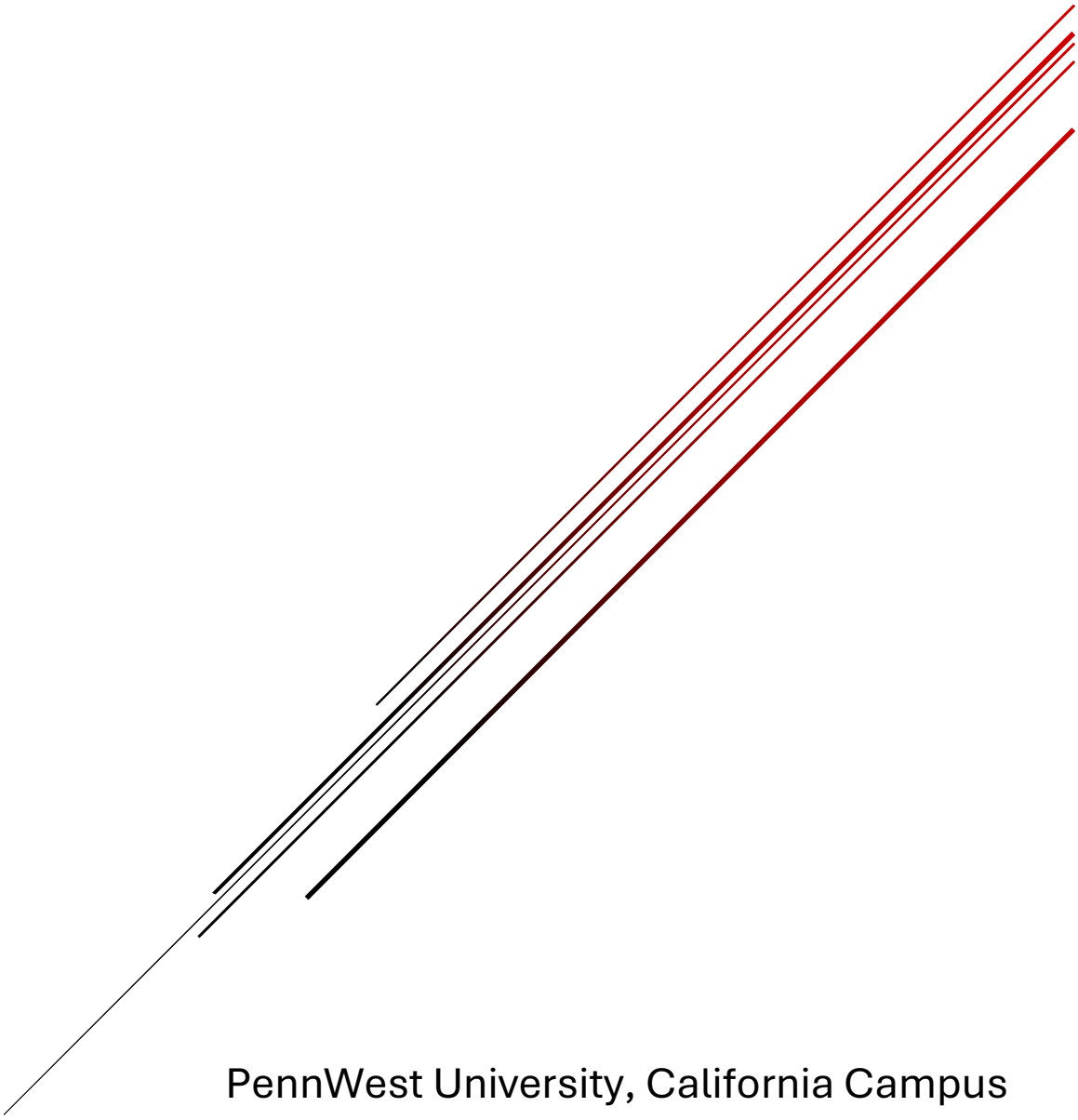
# INQUISITIVE STUDY SITE

## USER MANUAL



PennWest University, California Campus
Senior Project II
Caleb Ruby, Caleb Rachocki, Caleb Massey, Ibrahim Alani

# INSTRUCTOR COMMENTS / EVALUATION

# TABLE OF CONTENTS

# PROJECT OVERVIEW

## What is the Inquisitive Study Site?

The Inquisitive Study Site (ISS) is a web-based learning platform designed for learners of all levels. It promotes good study habits and self-education by offering a community-driven knowledge base that users can both contribute to and learn from using built-in tools. This project was inspired by our experiences with other academic study platforms, many of which have become increasingly restrictive—placing essential features behind paywalls or inundating users with ads. In contrast, ISS is entirely free to use, unmonetized, and offers full functionality with no limitations. As a browser-based application, ISS requires no downloads and is accessible from any device with an internet connection. Users can register for an account to create and store "study sets," which are then added to the site's growing public knowledge base. All study sets are searchable and accessible to both registered and unregistered users, making the platform a truly open educational resource.

## Intended Audience

Accessibility is a core goal of this project, and we believe it can benefit learners of all types. While we've drawn heavily on our own academic experiences to shape the project's direction—making it particularly useful for students—the platform is designed with a gentle learning curve that makes it approachable for users of all ages. At the same time, it incorporates proven educational tools that remain valuable even at the highest levels of education. Whether someone is in K–12, college, or pursuing a graduate degree, they can find value in the platform. This is made possible by its user-driven, freeform design, which allow users to personalize both the content and how they engage with it. With that in mind, we've carefully designed each element of the ISS to be intuitive and user-friendly, while maintaining a clean, professional look.

## Social Implications of the Project

The ISS allows students to store and use their own study material or explore content from our continually growing knowledge base. A key feature of the ISS is open access, meaning users can view and utilize any study set available on the platform. Ultimately, it is the users who will populate the database with their own knowledge, making the Inquisitive Study Site a community driven product.

The ISS doesn't support communication between users, this is not stopping the ISS from being used in a social setting; teachers and professors may also use the ISS for an easy access digital-based review of content, while students can use it to form study groups and work through certain sets of problems. All it takes is one person to create a given study set, and then provide a link to that specific set for anybody to access it. From there, users can generate quizzes with multiple types of questions, explore different types of questions available, and even utilize flashcards for a more simplified form of content review.

# SYSTEM BLOCK DIAGRAM

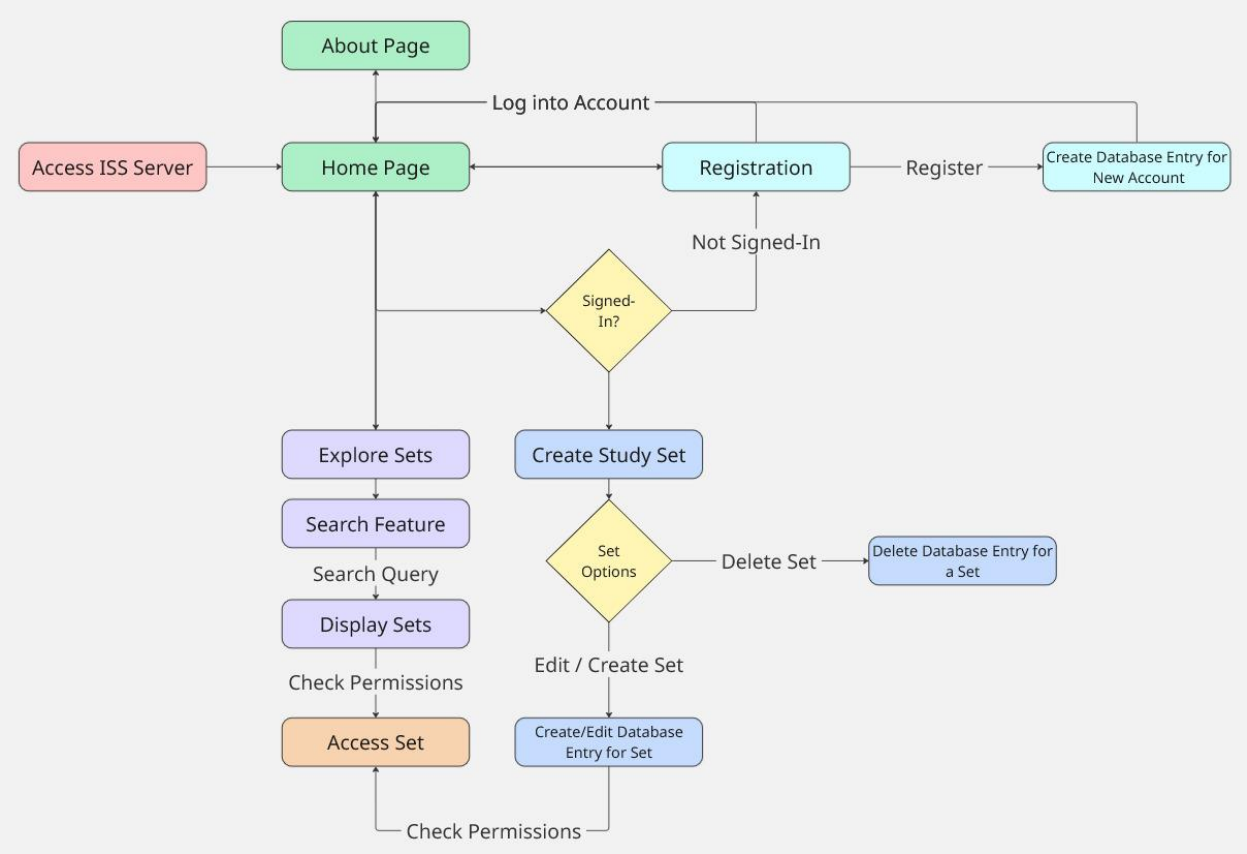



Figure 1. System Block Diagram

# Project Implementation Details

## General Implementation

The ISS has a frontend written in HTML, CSS, and JavaScript. These three working in unison provide the user with their experience on the website. The UI is primarily CSS and HTML, while JavaScript handles more dynamic features, such as tables for study set usage, as well as search features, and various other aspects of the project.

The backend communications of the ISS are handled through JavaScript, MySQL, and Node.JS, with JavaScript acting as the go-between for the website and the database. These three working together provide the frontend with its functionality, thus giving users the full experience of the ISS. Without the SQL database connection, users would not be able to access accounts, the search feature, or generally anything that isn't a static webpage.

With this in mind, users should know that the ISS requires both the frontend and the backend for the project to function properly. Without one, the other will fail to do as instructed. If the backend SQL database was down for some unspecified reason, the frontend would have incredibly limited capabilities. If the frontend was down for some unspecified reason, the backend wouldn't even be accessible to the user. Fortunately, the likelihood of one being down and not the other is slim to none, since both the database and the website are hosted from the same server through AWS, unless we were to specifically shut down one and not the other. However, it's entirely possible for both the frontend and backend to be down simultaneously.

## Differences from Previous Documentation

In making the ISS, we were required to make 3 separate 30-to-60-page documents, all detailing what we would and would not do when working on the project. While we have implemented most, if not all features we had previously specified, we did not do it in the way we initially intended to. Almost all of the classes, functions, and even variables went unused throughout the development of the ISS, rendering that section of the specification document meaningless to the reader. This is a result of our inexperience with the languages we have used in making the ISS; we were not expecting it to be as simple as it was to work with JavaScript.

We also feel that some, though not all of our diagrams have been inaccurate in the long run. Again, this is a result of inexperience and a lack of knowledge on building a website when writing the previous documentation. As of now, we feel we understand what it takes to build a website, and could therefore handle newer diagrams better. This is evident in our system block diagram, present in this document.

## Challenges during Implementation

As students, we of course faced life's challenges through the fall and spring semesters, both expected and unexpected; time conflicts, other projects, various exams, differences in ideas, general illnesses, family emergencies, and even flat tires. Ultimately, these were challenges that we had no way to solve, though they did not pose a threat to the progress of the ISS in any way. These were simply some unfortunate setbacks that ultimately slowed us down, but not enough to disrupt our efforts in implementing the project in a timely manner.

We've all agreed that the biggest challenge overall was our inexperience. None of us fully understood SQL going into the project, though one of us was aware of how to utilize it. Three of us also had minimal experience with HTML, CSS, and JavaScript. Because of our inexperience, it took us longer than anticipated to initially get the project moving, leading to us being behind on key features for at least two weeks. While this setback was a critical issue that was preventing us from working on the rest of the ISS, it did not stop us from completing it.

We also faced another problem, completely unrelated to languages; a server. For the ISS to work properly, we needed access to a webserver that would allow us to do what we needed to in the backend, and allow it to communicate to the frontend of the project. We initially used a school-provided server to host a static version of the website, but it ultimately did not suit our needs. After hours of research, we agreed on using *Amazon Web Services* (AWS), an Amazon based cloud computing solution. We chose AWS for a variety of reasons; Amazon being a trusted and reliable name, full access to set up what we needed, full support for any problems we may have faced, and a free three-month trial that covered the majority of the semester's costs. Even when the trial runs out, AWS still keeps the cost of hosting the ISS at a minimum.

In writing the code for the backend, we faced various difficulties in creating many features of the website. For example, when designing the flashcards portion of the study set template page, we were forced to deal with the same issue on three different occasions, all due to an indexing error. While it remains unclear what caused this issue, it was quickly resolved every time it was encountered by simply changing the initial value of the index variable. We believe this issue stemmed from the table that stored the study set information from our database, but we are unsure of why the values needed changed depending on the work done with the table.

In making the ISS, we had to design a user-friendly UI/UX to go along with the backend code in order for clients to properly use the ISS to its fullest potential. However, we ultimately made the decision to make the backend functionality a top priority first, thus meaning the ISS was considerably less user-friendly in the beginning of the project's implementation. While this was greatly beneficial in freeing up time and resources to work on the backend, it came at the expense of extra time and resources required to make the frontend in the latter half of the implementation timeline. Redesigning the entire website initially proved challenging, but eventually became easier as we put more effort into working on both the frontend and backend simultaneously, allotting a similar amount of time and resources into both.

There were also some difficulties in making the ISS compatible with mobile devices as well, i.e., size constraints. While the mobile version of the ISS is basic in its appearance, all features remain functional. We unfortunately were faced with the same problem as before though; a lack of experience. None of us have every designed a mobile interface for a project before now, which explains the simplicity of the ISS on mobile devices.

# SOFTWARE ENGINEERING PRINCIPLES

Modularity – The project is extremely modular, requiring many parts to work properly. We have approached this by separating each file type into different folders in the project, thus keeping everything organized and simple to navigate. Throughout the duration of the project's development, we continued to add more files into these folders, basing them off of their specific functionality. For example, because we have responsive designs implemented, we chose to separate the CSS files based on whether they implemented responsive design or not and aptly named the responsive files "responsiveFileExample.css" as opposed to the standard "fileExample.css." This practice applied to every aspect of the project and helped greatly in navigating and maintaining it.

Abstraction – The ISS is a complex project, requiring many moving parts working in unison to provide users with the best experience we could possibly provide. With this in mind, we tried to simplify working on the project as much as possible. We initially planned on using a PHP header for our navbar, but ultimately decided it would be easier to apply the navbar at the top of the screen to each webpage individually; this would also grant us the ability to alter it, should we have felt the need to do so. This also made the code for the navbar generally easier to work with. One of our biggest goals for the ISS was simplicity, both in the programming, and in understanding the project itself. We also made the index.CSS file our "master editor" by having it called in every HTML file, thus applying the same general styling to every webpage in the project. When working with a study set, users are able to either take a quiz based on the study set, or practice using individual elements of the quiz; this is where we reused some of our code, since the code for one piece already existed. Using this code, we simply applied it to the other forms, thereby eliminating redundancy in the code.

Encapsulation – Much of the backend data in the ISS is visible, but has private modification applied to almost every user aside from that which created it. We have also applied a level encryption to the database, hiding passwords to accounts to ensure user accounts remain safe. Many features of the ISS work through classes, such as the forms that provide our questions, as well as the study sets themselves. The database itself is also divided into various tables, each table storing various pieces of user data.

Separation of Concerns – As previously described in the *Abstraction* section, we have reused certain elements of our code; particularly in the case of styling the website, as every HTML page relies on the index.CSS in some capacity. This is also the case for our JavaScript. Despite this, many other features of the ISS are functionally isolated in their own files, such as registration.

# THE USER'S MANUAL AND INSTRUCTION

## 1.1 Accessing the ISS

To access the ISS, simply navigate to the webpage https://inquisitive-ss.com/index.html through the provided URL. User's may access the ISS from any computer or mobile device that has access to the internet. These devices include, but are not limited to personal computers, laptops, tablets, and smartphones. The ISS is meant to be accessible from any modern device that can easily access the internet. The ISS is accessible and fully functional in a mobile phone environment as well. While the mobile version has a different appearance than the desktop version, all features remain fully functional.

## 1.2 Navigating the Webpages

To properly utilize the ISS, users must be aware of all of the webpages they can access, with or without an account. In the following subsections, we will discuss which pages are which, and how users may access them. Webpages that do not require an account, dubbed *Limited-Access Webpages*, will not be re-discussed in the *Account Webpages* subsection, as all webpages of ISS are available to account users at all times; however, we will briefly discuss the access an account would give to a user in the *Limited-Access Webpages* subsection. At any point, a client may access the explore page using the navbar, or search for a specific set using the navbar's search feature.

### 1.2.1 Limited-Access Webpages

Limited-Access Webpages are as follows: *Home, Explore, About, Sign-Up,* and *Login*. These pages are navigable by all clients who visit the ISS, regardless of whether or not they have an account, or are logged into an account. These pages will be accessible to all users at all times, barring maintenance or server-downtime.

#### *1.2.1.1 HOME PAGE*

The Home page is the webpage users will be directly sent to as soon as they access the ISS from the internet. Users may also be brought to this webpage by clicking the *Home* button located in the navbar at any time on any other webpage. This webpage serves as the landing for a user who wishes to utilize this project; from here, they will have a plethora of options and directions, allowing them to access any other webpage from here.

The three most prevalent options are account registration, study set creation, and study set exploration. Should a user sign-in to their unique account, their study set history will appear on the home page as well, allowing for quick access to their most recently viewed study sets. Should a logged-out user view this page, the history area will appear empty to them until they sign-in. Should a user who is not signed-in attempt to create a new study set, they will be redirected to the *Registration* webpages. If the user is already signed in and attempts to create a new study set, they will be redirected to their personal *Library* webpage, which will be further discussed in a later part of this document. If a user attempts to access the *Explore* webpage, regardless of their registration status, they will be redirected to that webpage. The same applies to the *Registration*, *About*, and *Home* webpages, accessible through the upper navbar available on every webpage.

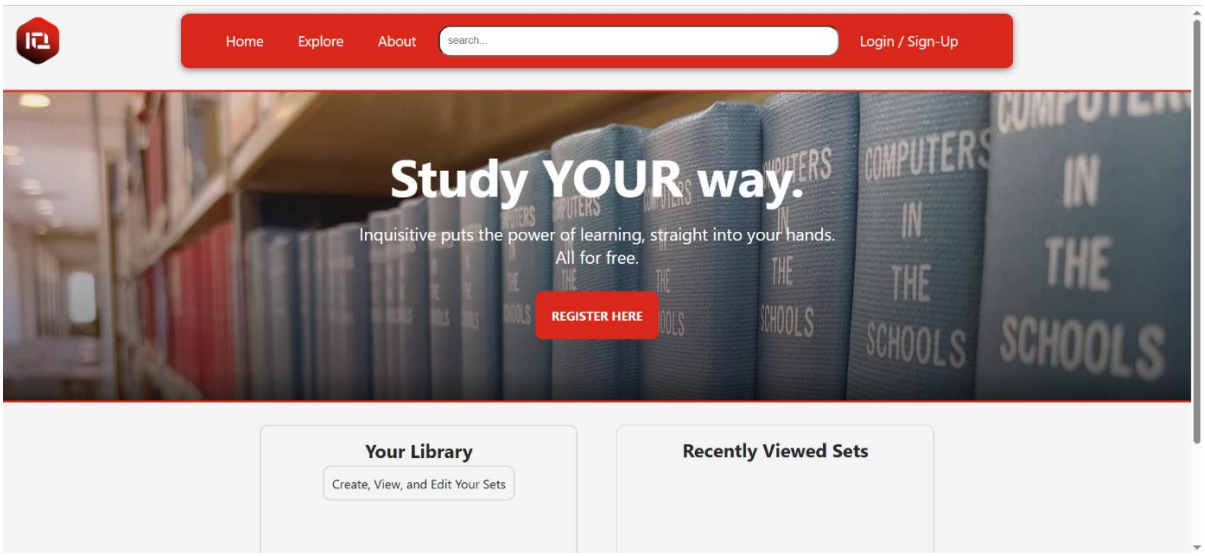



Figure 2: *Home Page*

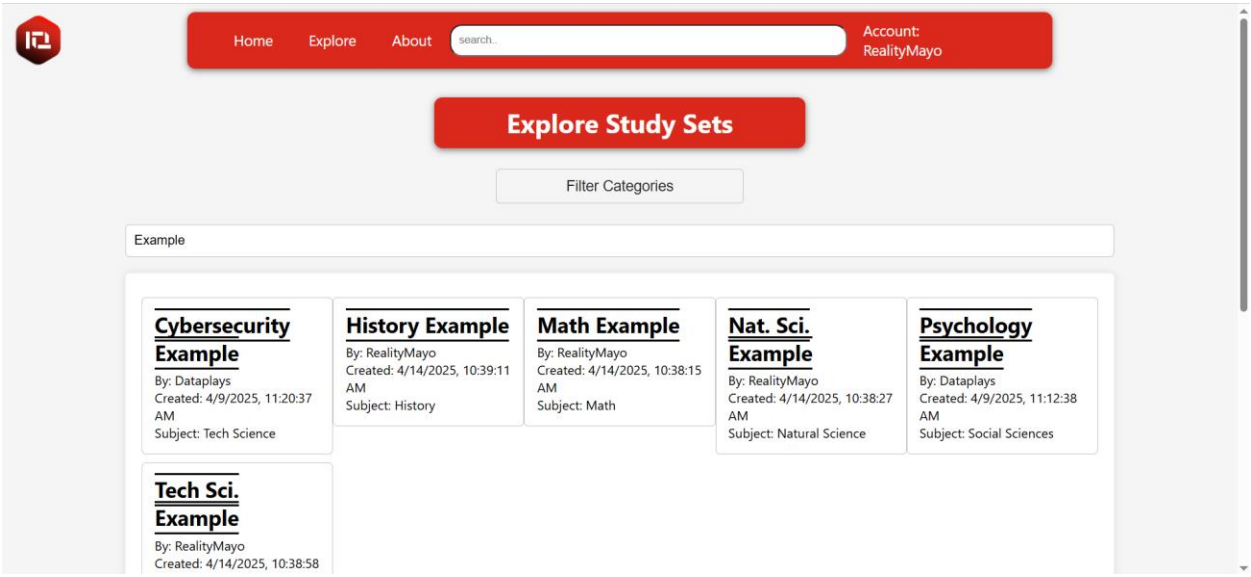



Figure 3: *Populated Explore Page*

*(Development Image; contents are likely to change.)*

#### *1.2.1.2 EXPLORE PAGE*

The *Explore* page is the heart of the project. This is where all clients will be directed if they're intending to use to the ISS for their educational purposes. The explore page, available to all clients connected to the ISS, is used to find any study set. When a client initially connects to this webpage, they will be presented with a blank page that has no search results. If a client would like to find a study set they'd like to use, they are able to do so by various means.

Should the user wish to find a specific study set, they must navigate to either search bar available on their screen and type their desired study set by name. That search query will be sent to the database, which will provide a response depending on what it finds. Through the use of identical searching, the database will provide study sets that only identical matches to the search query, as well as sets that have the searched word/title in their own title; this was seen as an opportunity to provide more sets to users. Users must press the enter key to perform their search.

Users will also be able to use the filter feature in their searches, which will narrow down further the study sets that the database can respond with. For example, if a set has a category of "Math," the search query will only ask for study sets of a given name that are tagged with the "Math" category, producing results that are only in that category. Multiple categories may be selected at a time during a search, as to prevent a type of bottleneck in the user's experience. Filters can be applied before or after a search, but they themselves do not perform a search.

Once a user has found their desired study set, they simple navigate their mouse to that set. Hovering over that set, user's will immediately notice the visual change, indicating they're about to select this study set. Once selected, the user will be redirected to the study set template webpage, which will be discussed further in section 1.2.1.6.

#### 1.2.1.3 ABOUT PAGE

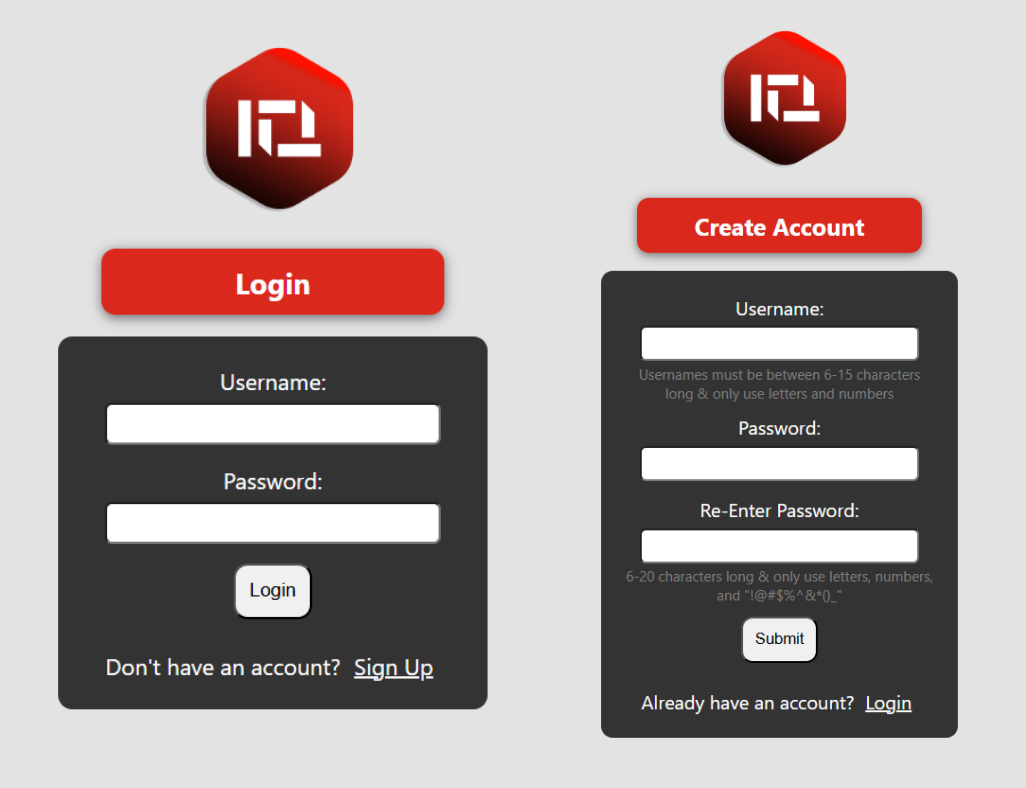
The *About* page serves a simple purpose; provide clients with an overview of the project, and provide a link to weekly documentation for the project. This page also provides a brief description of the team members, as well as why we decided to create the ISS. From this page, clients may navigate to the *Weekly Updates* website. This website has every piece of documentation created for the ISS project available at any time. It's worth noting that this website will likely be shut down and moved to the main ISS website when the semester ends.

#### 1.2.1.4 SIGN-UP PAGE

The *Sign-Up Page* is the counterpart of the *Login Page*. This page is targeted to clients who do not yet have an account to fully utilize the ISS. To register for an account, a user must navigate to this webpage via any means we have provided. Once on this webpage, a client will input their desired username, following the requirements provided. Usernames must be six (6) to fifteen (15) characters in length, and can only be alphanumeric; no symbols are permitted, otherwise, an error will return to the user. Usernames must also be unique, and cannot match a pre-existing account's username. Clients will then create a password on the same page, and must verify that they know it by entering it twice. Passwords also have strict criteria; they must be six (6) to twenty (20) characters in length, and alphanumeric, however they're able to include the symbols "!@#$%^&*()_" (quotes not included). Once account information is entered, the user is able to submit their information, which will be encrypted, sent to, and finally stored in the database. Users are not required to provide any private information aside from a unique username and password. From here, they'll be sent back to the *Home Page*, now logged in to their account. Users are solely responsible for their own account information; if it is lost, they cannot retrieve it unless they contact the developers.

### *1.2.1.5 LOGIN PAGE*

The *Login Page* is the counterpart to the *Signup Page*. This page is meant for returning users who already have an account associated with the ISS. This page functions similarly to the *Signup Page*. Users will need to provide their unique username and password in order to login. The user will identify their account by placing their username and password in the appropriate places and clicking the submit button. Because of the simple requirements and hands-off nature of the ISS, we have no way to recover accounts, so users are solely responsible for their account; if they lose their information, they cannot retrieve it unless they contact the developers directly. After logging into their personal account, users will be redirected back to the *Home Page*.




Figures 4 & 5:
*Login and Sign-Up Forms*

#### *1.2.1.6 STUDY SET PAGE – LIMITED FUNCTIONALITY*

This page is where users will be directed to once they've selected a given study set from the *Explore Page*. This page appears identical across all study sets, since it's a dynamic template page that simply fills in variable data into certain fields across the webpage. This data is pulled from the database we have set up that stores all information regarding a given study set, where each set is individually stored and maintained. Once on the webpage, users will be able to view the set in its entirety by scrolling through the webpage. Users will be able to interact with all features available in the form of buttons at the top of the webpage, located beneath the title of the selected study set.

SELECTING TERMS; In order to access any of the study resources available, users must first select which terms-definition combos they would like to use. This ensures that the user may freely customize their experience further, allowing them to skip certain term-definition combos they may already know. To select a term-definition combo, the user must navigate their mouse to which term-definition combo they would like, then navigate to the left-most side of that row. They will encounter a checkbox, located on the left side of the term; users will simply click this checkbox to select the term-definition combo. This term-definition combo will be used in any study resource on that page. Should the user wish to un-select, follow the same procedure just described, except ensure that the term-definition combo does not have a checkmark in its associated checkbox. Should the user wish to select or unselect all terms at once, they simply need to navigate to the "Controls" label, located under the study set title. Under this label, there are two buttons; one is to check all terms in the set (Select All), and the other is to uncheck all terms in a set (Select None). Clicking either of these will perform their associated action, enabling all terms or disabling all terms respectively.

FLASH CARDS; This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the flashcards. Once the flashcards have opened, the user will see a box with a single term appear on their screen. Once this box has been clicked, this term will turn into its associated definition, with the box changing its appearance as well to provide a visual indication that card has been “flipped.” Users will be able to navigate through any term-definition combo by utilizing the *Next* and *Back* buttons at the top of the flashcards. Once done with the flashcards, the user simply navigates to the *Close Flashcards* button in the upper right corner of the flashcards, which will return them to their study set.

MULTIPLE-CHOICE; This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the multiple-choice questions. Once the form for the questions has appeared on the user’s screen, they’ll be presented with a single term, which can be cycled through similarly to the flashcards. Each term will have four possible answers, where only one will be correct. Users must read and locate the correct answer for the associated term that’s been listed above. Once they find the answer they think is correct, the click the radio button to the left of it, and must click the “Check” button at the bottom of the box. This will verify the user’s answer, telling them it’s either correct, or telling them it’s incorrect while providing the correct answer. To move on to the next term, user’s simply click the “Next” button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the “Exit” button. Once they’re done reviewing this information and close the prompt, they will be returned to the study set page.

FILL-IN-THE-BLANKS; This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the fill-in-the-blank questions. Upon opening this resource, users will be presented with a term, and must fill in the definition of the given term in the associated text-field using their keyboard. Once finished with that term, users can use the “Check” button to verify their answer. The website will respond with either the correct answer, or will notify the user that they have entered the correct answer. To move on to the next term, user’s simply click the “Next” button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the “Exit” button. Once they’re done reviewing this information and close the prompt, they will be returned to the study set page.

TRUE / FALSE; This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the true and false questions. Upon opening this resource, users will be presented with a term, a random definition from the study set, and two radio buttons labeled true / false. Each question will only present one term and definition at a time. Users will read the term, then determine if the definition provided by the form is correct; if it is, then the user will select true, otherwise they will select false. Once answered, users will use the “Check” button to verify their answer. The website will respond with either the correct answer, or will notify the user that they have entered the correct answer. To move on to the next term, user’s simply click the “Next” button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the “Exit” button. Once they’re done reviewing this information and close the prompt, they will be returned to the study set page.

QUIZ; This button is located under the "Test Yourself" label, under the title of the study set. Clicking this button will apply the selected term-definition combos to a quiz. A quiz is compiled of all possible learning resources we have available on the website, excluding the flashcards. To be clear, a quiz includes multiple-choice questions, fill-in-the-blank questions, and true or false questions. Should a user want to learn more about these specific types of questions, refer to the previous underlined sections; they describe what to do with each type of question, as well as how they operate. Otherwise, once a user is done taking their quiz, they click the submit button located at the bottom of the quiz form, which will report correct and incorrect answers, as well as an overall grade. Any questions left blank will be counted towards their score. Once the user is done reviewing this information, they can close the prompt, and will be returned to the study set page.

### 1.2.2 Account Webpages

#### *1.2.2.1 LIBRARY PAGE*

The *Library Page* is where the user is directed when they want to view all of their previously created sets, or wish to create a new set. There is no other information on this page. From here, users will be able to directly access their created study sets. Users will also be able to delete their study sets from here, which will remove the set from the website's database. Doing this will make it unavailable to anyone who attempts to find and/or use it, as it will no longer exist. To delete a set, a user must navigate to the desired set they would like to delete, then navigate to the delete button in the associated row. They will be prompted to confirm their selection beforehand; after confirming, the set will be removed from the ISS database, and will no longer be available in any capacity. Once a set sis deleted, it's gone forever; there is no way to recover an accidentally deleted set.

### *1.2.2.2 STUDY SET PAGE – ACCOUNT FUNCTIONALITY*

If the reader is looking to learn more about the functionality of the overall *Study Set Page*, please refer this section's counterpart, *1.2.1.6 STUDY SET PAGE – LIMITED FUNCTIONALITY.* Users will learn more about how to use this webpage there. Should the user have an account and wish to learn more about what this means and how they can alter their personal study sets, please continue reading this section.

Should the user be the creator of the study set, they will be able to add, alter, and remove terms and definitions freely. These changes will immediately take effect once a user has confirmed their action. To delete a term and definition combo, the set creator must navigate to the desired term/definition combo, then navigate to the end of that specific row; from there, press the red "Delete" button; the term/definition combo will be removed from the database entry for that study set, and thus from the study set itself. To edit a term/definition combo, the set creator must navigate to the desired term/definition combo, then navigate to the end of that specific row; from there, simply press the "Edit" button, which will enable the set creator to edit either text field of that term/definition combo. Both the term and the definition are able to be altered, but it is not required to edit both. Once the desired changes are made, the set creator must apply changes by navigating to where the "Edit" button previously was, and must click "Save." Once saved, the changes will immediately be applied. Edits may be made anytime.

Adding a term to a study set is just as simple as editing. Should he user feel the need to do so, they simply navigate to the text-fields labeled "term" and "definition," filling in both with their appropriate information. Once the information is filled in, the user only has to press the button labeled "add term," which will immediately add that term/definition combo to the study set. The database will be updated immediately as well.

Should an account user view a study set that is not theirs, they will mostly have limited functionality applied to that set. To learn more about what an account user can do with a study set that is not theirs, please refer to *1.2.1.6 STUDY SET PAGE – LIMITED FUNCTIONALITY*. Just below, we have listed the actions a user with an account can take with a set that is not theirs.

KNOWN / UNKNOWN; This is an extension to our term selection. For the sake of the user, we have implemented a system that allows the user to skip their "known" terms. To mark a term as known or unknown, a user simply navigates to the known/unknown button, which has a colored block to the left of it. Green indicates that the term is known, red indicates the term is unknown. When a term is unknown, it will be used by the learning tools; when a term is known, it's no longer included in the learning tools for that user specifically. Terms may be selected as known or unknown at any time. The status of terms is unique to each user and each set, so only the user will be able to see which terms they've selected as known/unknown. Known/unknown terms do not carry over to other sets.

COPY SET; This feature is simple; simply click the "Copy Set" button located at the top of the study set webpage. This creates a unique copy of the study set, given the original name of the set as well as an appended ([USERS] Copy). Both sets will exist separately, and edits made to one will not affect the other. Users are able to do anything they wish with their copied set, as they are now the owner of it, while the original will remain unchanged.

### *1.2.2.3 PROFILE PAGE*

The *Profile Page* is where a user will find basic account information, as well as a few basic account options. The information displayed is associated to a user's specific account, i.e., their name, date of account creation, etc. The other options that are available on this page are as follows: account deletion, password change, and account logout. After any of the actions described below are taken, users will be returned to the *Home Page.*

To delete an account, a user should simply navigate to the "Delete Account" button, which will require a confirmation from the user. After confirming, the account will be removed from the ISS database. Due to the nature of the ISS, there's nothing we can do should there be an accidental deletion. The same applied to passwords, as we have no way to automatically recover them for users. Accounts are ultimately the responsibility of the user; we simply provide the means to have an account and use the ISS with one. However, should the user wish to change their password to their account, they are able to do so on this page, following the same directions as described in *1.2.1.4 SIGN-UP PAGE*. The user is also able to logout of their account from this webpage, should they feel the need to. A user will simply navigate to the logout button, and provide confirmation. Users will be able to login and out of their account as often as they wish.

### 1.2.3 Conclusion of Webpage Navigation

To reiterate, every webpage is available to a user who is signed in to their personal account through the ISS. While users who do not wish to use an account will still have access to the project and its contents, they will not be able to fully utilize the features of the ISS, that being making their own personal study sets. We did this to provide a free, safe, and educational space for students who are willing to take the extra steps in their studying habits, even if a user does not need to make their own set. This also creates opportunity for teachers to use the project to create review material that's accessible from anywhere. Users have nothing to fear either, as there is absolutely no personal data involved in using the ISS. Ultimately, the goal is to create a space to learn, and nothing more.

## 1.3 Disclaimers and EULA

By using this product, which is freely available online to anybody, the user or any other form of clientele agrees to the following terms and conditions;

I. Accounts are the sole responsibility of the client; the ISS, nor its development team members, or admins are liable to solve any problem caused by a bug or unintended feature discovered by a user that may permanently alter their account, its information, or any associated data tied to that account i.e., the study sets associated with that account.

II. Account information, i.e., usernames and passwords, are also the sole responsibility of users of the ISS. If this information is lost for any reason, regardless of if it's at the fault of the user or not, the ISS, its development team, or admins are not responsible for retrieving account information for a specific user.

III. By using this product, you, the client, agrees that we have reserved the right to manipulate, alter, or terminate your account(s) and/or their associated data at any time. Accounts and/or their associated data, including their personal study sets, may be terminated without warning or notification for any reason at any time. There are no appeals or disputes to make when an account is terminated, as there is no way to recover account information or its associated data in any ways once it is removed from the ISS database.

IV. The ISS, the development team, or its admins are not responsible for any infection or damage done to your personal device while utilizing or accessing any webpage/feature of the ISS.

V. The ISS, the development team, or its admins cannot be held responsible for content you may find distressing in any capacity. While we try our best to moderate content available to everyone on the ISS, we unfortunately cannot always remove all inappropriate content.

VI. The end-user will not intentionally upload malicious, inappropriate, or incorrect content to the ISS. The ISS is meant to be a safe space available to students of all ages of all grades. Should the user upload any content of this variety, their account is liable to be deleted entirely. Malicious and inappropriate includes, but is not limited to, suggestive content, threats of any capacity, foul language, harassment of any capacity, heavily opinionated information or topics, any form of links to other webpages, programming scripts, or direct communications between study sets.

VII. The end-user will not utilize an ISS account for any form of cheating. Cheating includes, but is not limited to, directly uploading homework, quiz, test, or exam questions and answers to the ISS. Doing so may result in the culprit user's account and study set data being terminated upon discovery by administration.

VIII. By creating an account and uploading any amount of your personal study set information, you agree that we have reserved the right to use your uploaded study set data in any way we desire. No data provided will be sold, simply stored in the database, and altered at any given time, as you have given us the right to do so.

## 1.4 Appearance of the Project

The ISS is adorned in colors that should be familiar to PennWest University students. We have chosen to use a red that was specific to the California campus, known as *California Red* with an RGB value of (218, 41, 28). This red was chosen as the primary color of the project. We have done this show an appreciation to the campus we have spent the last four years on, and to signify an association of the project to our campus specifically. However, because California University merged with Clarion and Edinboro to become PennWest University, we have also decided to incorporate the *PennWest Blue* that's associated with the university. All other colors were chosen purely for aesthetic purposes.


Generally speaking, we have attempted to craft the website in such a way that everything is self-explained. Buttons being labeled appropriately, easy access to study sets, search from anywhere on the website, and various other quality of life features were discussed and implemented. Ultimately, we decided to keep the project simple in appearance, but professional enough to present.

# SOURCES

Amazon Web Services (AWS) (n.d.). *What is AWS?* AWS Amazon. Retrieved March 31, 2025, from https://aws.amazon.com/what-is-aws/?nc1=f_cc

HTML. (2025, April 1). In *Wikipedia*. https://en.wikipedia.org/wiki/HTML

CSS. (2025, April 2). In *Wikipedia*. https://en.wikipedia.org/wiki/CSS

JavaScript. (2025, March 29). In *Wikipedia*. https://en.wikipedia.org/wiki/JavaScript

Node.JS. (2025, March 26). In *Wikipedia*. https://en.wikipedia.org/wiki/Node.js

MySQL. (2025, April 6). In *Wikipedia*. https://en.wikipedia.org/wiki/MySQL

# GLOSSARY

***Study Set*** – A user-defined collection of data stored in the ISS' offsite database, containing any number of term/definition combos.

***Term/Definition Combos*** – A user-defined set of data encapsulated within a given study set. Consists of a Term and its associated Definition.

***Term*** – A user-defined piece of identifier data, tied to a specific definition. Ideally a single word, or short series of words.

***Definition*** – A user-defined piece of semantic data, tied to a specific term. Ideally a longer series of words that apply meaning to the associated term.

***ISS*** – Abbreviated form of the project's name, "Inquisitive Study Site." This abbreviation is used only in written documentation, and occasionally in conversation. Verbally, the project is simply known as "Inquisitive."

***AWS*** – An abbreviation for "Amazon Web Services." Defined as a cloud-computing service provider, capable of many technological feats such as web-communications, hosting, computations, storage, database managements, machine learning, A.I., analytics, and more.

***HTML*** – (Hypertext Markup Language); Used as the standard markup language for webpages and documents that are specifically for web browsers, defining both structure and content.

***CSS*** – (Cascading Style Sheets); Style sheet language used to make HTML and various others have a better appearance. Dictates the content and presentation of the content available on an HTML based webpage.

***JavaScript*** – Frontend programming language that grants a majority of the world's websites their functionality. Allow for communication between frontend features and backend protocols, databases, servers, and more. Commonly used in conjunction with Node.JS.

***MySQL*** – Open-source relational database management system that allows for data tables and organized, structured content.

***Node.JS*** – JavaScript runtime environment for server-side scripting, allowing for dynamic webpage content.

***System Block Diagram*** – General structure of a piece of software and how it interacts with hardware; a simplified view for readers and consumers to understand how a piece of software works

***EULA*** – End User License Agreement; defined as a contract between software developers, and the client-end user, in which the client agrees to the terms and conditions set by the developers.

***Fuzzy Searching*** – A means by which the data being searched for is permitted to have typos and errors; accounts for human error in searching the database by allowing the query "xample" to use the result "Example."

# APPENDIX: TEAM DETAILS

The work done on the ISS project was handled by all members of the team accordingly, each putting as much time as possible into their best-suited areas. Ideally, we all would've worked on the same features and implementations simultaneously, however this was impossible due to our ambitious project. Key features of the ISS were divided among group members, which were further divided still. While some group members primarily worked on the backend, others worked primarily in the frontend; despite this, we managed to come together to work on paperwork and discuss our weekly meetings and presentations.

While this project was the largest one we've worked on during our time at PennWest, we managed to stay on top of it throughout the semester, despite difficulties in the beginning. Because of Caleb Ruby's persistence and dedication to the entire backend of the ISS, the other group members were able to pick up and assist in making the website more functional overall. Furthermore, because of Caleb Ruby, the database was nearly fully operational by week 3. Granted, we still needed to add in many tables, queries, etc., but it was a great benefit to have the database running as quickly as we did. Once this was achieved, the rest of the project could continue to be worked on despite a roadblock before doing this. Caleb Ruby worked on the JavaScript for the search and filter features of the ISS, key parts in letting users find a study set. Caleb Ruby also focused heavily on data manipulation and storage in the SQL database, encrypting information as needed and providing security where needed. If not for his efforts, the project would be in a less-than-ideal state.

Caleb Rachocki has also made great efforts as a frontend/backend intermediary, effectively and efficiently implementing JavaScript that communicated with Caleb Ruby's code in the backend of the ISS. Both Rachocki and Ruby implemented a majority of the JavaScript of the project, with some organization of the code coming later from Caleb Massey. Caleb Rachocki also initially set up our HTML pages over winter break, providing a phenomenal groundwork to begin building the ISS on, ultimately providing a head start on the project. Caleb Rachocki also provided much of the JavaScript used in the study set template page, the a key part of the ISS.

Caleb Massey handled a majority of the styling of the ISS, taking notes from his peers, other team members, and visiting a variety of other websites to take inspiration from them. He also handled a majority of the responsive design that allowed the project to work on mobile devices. Massey worked on small amount of JavaScript, but not anything note-worthy; most his JavaScript work was to retrieve and display some extra information from the database about a particular entry. Caleb Ruby worked on frontend design and styling as well, cleaning up and providing consistency across each webpage. Caleb Massey also handled the weekly paperwork, weekly PowerPoints, the User Manual documentation, a majority of the general design documentation and the overall appearance of the final presentation.

Ibrahim Alani put his efforts in general assistance. Ibrahim, known as Abe to the team, worked on various aspects of the project, never really staying in one place; his work came in the form of providing quick solutions to various problems through his research. Abe also provided help in designing the ISS, as well as separating and organizing some of the files of the project. Caleb Ruby ultimately restructured the entire project, but it was Abe who initially started to separate files based on their functionality. Abe also worked with JavaScript when needed, working on various study set features with Caleb Rachocki.

# APPENDIX: WRITING CENTER REPORT

The following notes related to your 4/14/2025 11:00 AM EDT appointment with Tatiana Kostovny have been shared with you:

**Grammar**
- Commas: When using commas before a conjunction, be sure that it's separating two independent clauses. I noticed a few sentences where the comma is not needed.
- Hyphens: There are a few hyphen errors. Look over the comments to see if they occur anywhere else that I didn't highlight.
- Verb Tense: Make sure that your verb tense is consistent throughout your writing.

**Simple Errors:**
- There are a couple of spelling errors and words that are not needed. I would suggest proofreading your document thoroughly to find any that I did not.

**Overall**:
No major errors. Very well written with good vocabulary.

** Addendum by ISS Team

We have reviewed the document and resolved these issues to the best of our abilities.

# APPENDIX: CODE LISTING

** We have not included the code used in the Node.JS packages we used since we did not write it ourselves.

## BACKEND JAVASCRIPT

SEARCHROUTE.JS ----------

```
module.exports = function(app, db) {
  // Basic title search route using POST method
  app.post('/study-sets/search', (req, res) => {
    let { set_name, categories } = req.body;

    // Ensure `categories` is an array or fallback to a default array
    if (!Array.isArray(categories)) {
      categories = []; // Default to empty array
    }

    const wildcardQuery = `%${set_name}%`;

    let sql = `
      SELECT *
      FROM StudySets
      WHERE set_name LIKE ?
    `;

    const params = [wildcardQuery];

    // Add the category filter only if categories are selected
    if (categories.length > 0) {
      sql += ` AND category IN (${categories.map(() => '?').join(', ')})`;
```

```
    params.push(...categories); // Bind category values dynamically
  }

  sql += ' ORDER BY set_name';

  // Execute the query
  db.query(sql, params, (err, results) => {
    if (err) {
      console.error('Database query error:', err.message);
      return res.status(500).json({ error: err.message });
    }
    res.status(200).json(results.length ? results : []);
  });
});

// New route to search for public study sets from other users
app.post('/study-sets/explore', (req, res) => {
  const { set_name } = req.body;

  const wildcardQuery = `%${set_name}%`;

  let sql = `
    SELECT StudySets.id, StudySets.set_name, StudySets.description, Users.username
    FROM StudySets
    JOIN Users ON StudySets.user_id = Users.id
    WHERE StudySets.set_name LIKE ? AND StudySets.is_public = 1
  `;

  const params = [wildcardQuery];
```

```
  // Execute the query
  db.query(sql, params, (err, results) => {
    if (err) {
      console.error('Database query error:', err.message);
      return res.status(500).json({ error: err.message });
    }
    res.status(200).json(results.length ? results : []);
  });
});

// New route to copy a study set to the user's study sets
app.post('/study-sets/add', (req, res) => {
  // Retrieve user_id from session (or token if using JWT)
  const user_id = req.session.user_id; // Ensure session middleware is configured
  const { study_set_id } = req.body;

  if (!user_id) {
    return res.status(401).json({ error: 'User not authenticated' });
  }

  // Query to copy the study set
  const sql = `
    INSERT INTO StudySets (user_id, set_name, description, category, is_public)
    SELECT ?, set_name, description, category, 0
    FROM StudySets
    WHERE id = ?
  `;
```

```
    const params = [user_id, study_set_id];

    // Execute the query
    db.query(sql, params, (err, result) => {
      if (err) {
        console.error('Database query error:', err.message);
        return res.status(500).json({ error: 'Failed to add study set' });
      }
      res.status(200).json({ message: 'Study set added successfully' });
    });
  });
};
```

---

SERVER.JS ----------

```
/*
~~~~~~~~~~Initialization~~~~~~~~~~
*/
// Change the working directory to the root of the repository
const path = require('path');
process.chdir(path.resolve(__dirname, '../../'));

// Load environment variables from .env file
require('dotenv').config();

//Requirements
const session = require('express-session'); //allows for 'sessions'
```

```
const bcrypt = require('bcrypt'); //for password hashing
const express = require('express');  //middle-ware: serves front, handles communication to back
const mysql = require('mysql2');  //database features
const levenshtein = require('fast-levenshtein');



//create instances and constants
const bodyParser = require('body-parser');
const app = express();
const port = process.env.PORT || 3001;

// create MySQL Connection from environment variables
const db = mysql.createPool({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASS,
  database: process.env.DB_NAME,
  waitForConnections: true,
  connectionLimit: 10, // Adjust based on your app's needs
  queueLimit: 0
});

/*

//will ping server and keep it alive if needed: 'heartbeat'
setInterval(() => {
  db.query('SELECT 1', (err) => {
    if (err) console.error('Database heartbeat failed:', err);
  });
```

```
}, 60000); // Ping every 60 seconds

*/



// Use sessions to track user login status
app.use(session({
  secret: process.env.SESSION_SECRET,
  /*
   this will be kept in the environment variables for security pracices, even if we don't really expect any attacks

    can be thought of as a random seed and a password:
     ensures constistency by creating a digital signature for the session ID cookie
  */
  resave: false,
  saveUninitialized: true,
  cookie: { secure: false }  // Set to 'true' when we use HTTPS
}));

//middleware to parse JSON communications from front-end
app.use(bodyParser.json());

// Import and use the routes
require('./studySetRoutes')(app, db);
require('./userRoutes')(app, db);
require('./searchRoutes')(app, db);
```

```
/*
Starting the server, database connection, and serving static files
*/

// Serve static files from the "frontend" directory
app.use(express.static(path.join(__dirname, '../frontend')));
app.use(express.static(path.join(__dirname, '../frontend/html')));
app.use(express.static(path.join(__dirname, '../frontend/css')));
app.use(express.static(path.join(__dirname, '../frontend/js')));
/*
 path.join() creates the file path using following parameters
  __dirname gets current file directory
  '..' navigates back a directory
  'frontend' moves into frontend directory

 result: express serves static files (frontend) from the frontend directory
*/

// Connect to MySQL
db.getConnection((err, connection) => {
 if (err) {
  console.error('Error connecting to the database:', err);
 } else {
  console.log('MySQL Connected...');
```

```
    connection.release(); // Release the connection back to the pool
  }
});

db.on('error', (err) => {
  console.error('Database error:', err);
});




// Start the server
app.listen(port, () => {
  console.log(` Server started successfully on port: ${port}` );
});

-----------------------------------------



STUDYSETROUTES.JS ------------------------
module.exports = function(app, db)
{
//transaction helper function
const executeTransaction = (db, callback) => {
    db.getConnection((err, connection) => {
        if (err) {
            console.error('Error getting connection from pool:', err);
            return callback(err);
```

```
    }

    connection.beginTransaction((err) => {
      if (err) {
        connection.release();
        console.error('Error starting transaction:', err);
        return callback(err);
      }

      callback(null, connection, (commitErr) => {
        if (commitErr) {
          return connection.rollback(() => {
            connection.release();
            console.error('Transaction rolled back:', commitErr);
          });
        }

        connection.commit((err) => {
          connection.release();
          if (err) {
            console.error('Error committing transaction:', err);
            return callback(err);
          }
          console.log('Transaction committed successfully');
        });
      });
    });
  });
};
```

```
//-----check answer route-----
const levenshtein = require('fast-levenshtein');


  app.post('/check-answer', (req, res) => {
    const { userInput, correctAnswer } = req.body; // Expect user input and correct answer in the request body


    // Compute Levenshtein distance
    const distance = levenshtein.get(userInput, correctAnswer);
    const threshold = Math.floor(correctAnswer.length/100 * 80); // tolerance set at ~80% character length


    if (distance <= threshold) {
      res.json({ success: true, message: "true" });
    } else {
      res.json({ success: false, message: "false" });
    }
  });



  // Route to get study sets for the current user
  app.get('/study-sets', (req, res) => {
   const username = req.session.user.username;
   if (!username) {
```

```
      return res.status(401).json({ error: 'Unauthorized' });
    }

    const query = 'SELECT * FROM StudySets WHERE username = ?';
    db.query(query, [username], (err, results) => {
      if (err) {
        return res.status(500).json({ error: err.message });
      }
      res.status(200).json(results);
    });
  });

  // Route to add a study set
app.post('/study-sets', (req, res) => {
  const username = req.session.user.username;
  if (!username) {
    return res.status(401).json({ error: 'Unauthorized' });
  }

  const { set_name, category } = req.body;

  // Validate the category input (optional, to ensure valid categories)
  const validCategories = [
    "Math", "Natural Science", "Tech Science", "History",
    "Social Sciences", "Language", "Test Prep", "Other"
  ];
  if (!validCategories.includes(category)) {
    return res.status(400).json({ error: 'Invalid category' });
  }
```

```
  // Insert set_name and category into the database
  const query = 'INSERT INTO StudySets (username, set_name, category) VALUES (?, ?, ?)';
  db.query(query, [username, set_name, category], (err, result) => {
    if (err) {
      return res.status(500).json({ error: err.message });
    }
    res.status(201).json({ id: result.insertId, set_name, category });
  });
});

  //route to modify set name
  app.put('/study-sets/:setId', (req, res) => {
    const username = req.session?.user?.username; // Avoid undefined errors
    if (!username) {
      console.error("Unauthorized request: No username found.");
      return res.status(401).json({ error: 'Unauthorized' });
    }

    const { set_name } = req.body;
    const { setId } = req.params;

    if (!set_name?.trim()) {
      console.error("Invalid input: Empty study set name.");
      return res.status(400).json({ error: 'Set name cannot be empty' });
    }

    const query = 'UPDATE StudySets SET set_name = ? WHERE set_id = ? AND username = ?';
    db.query(query, [set_name, setId, username], (err, result) => {
```

```
    if (err) {
      console.error("Database error:", err);
      return res.status(500).json({ error: err.message });
    }

    if (result.affectedRows === 0) {
      console.error(`Update failed: Study set ${setId} not found or unauthorized.`);
      return res.status(404).json({ error: 'Study set not found or unauthorized' });
    }

    res.json({ success: true, set_name });
  });
});



// Route to add terms to a set
app.post('/study-sets/:set_id/terms', (req, res) => {
 const { set_id } = req.params;
 const { term, definition } = req.body;
 const query = 'INSERT INTO Terms (set_id, term, definition) VALUES (?, ?, ?)';
 db.query(query, [set_id, term, definition], (err, result) => {
   if (err) {
     return res.status(500).json({ error: err.message });
   }
   res.status(201).json({ id: result.insertId, term, definition });
 });
});
```

```
// Route to get terms from a set
app.get('/study-sets/:set_id/terms', (req, res) => {
  const { set_id } = req.params;
  const query = 'SELECT * FROM Terms WHERE set_id = ?';
  db.query(query, [set_id], (err, results) => {
    if (err) {
      return res.status(500).json({ error: err.message });
    }
    res.status(200).json(results);
  });
});

//get study set infor from set ID
app.get('/study-set-info/:studySetId', (req, res) => {
  const { studySetId } = req.params;
  const query = 'SELECT * FROM StudySets WHERE set_id = ?';
  db.query(query, [studySetId], (err, result) => {
    if (err) {
      console.error('Database query error:', err);
      return res.status(500).json({ error: err.message });
    }
    console.log(result[0]);
    console.log('Query:', query);
    console.log('Study Set ID:', studySetId);
    if (result.length === 0) {
      return res.status(404).json({ message: 'Study set not found' });
    }
    res.status(200).json(result[0]);
  });
```

```
  });

// Route to get study set name by ID
app.get('/study-sets/:studySetId', (req, res) => {
  const { studySetId } = req.params;
  const query = 'SELECT set_name FROM StudySets WHERE set_id = ?';
  db.query(query, [studySetId], (err, result) => {
    if (err) {
      console.error('Database query error:', err);
      return res.status(500).json({ error: err.message });
    }

    console.log(result[0]);
    if (result.length === 0) {
      return res.status(404).json({ message: 'Study set not found' });
    }
    res.status(200).json(result[0]);
  });
});

// Route to delete a term
app.delete('/terms/:term_id', (req, res) => {
  const { term_id } = req.params;
  const query = 'DELETE FROM Terms WHERE term_id = ?';
  db.query(query, [term_id], (err, result) => {
    if (err) {
      return res.status(500).json({ error: err.message });
    }
    res.status(200).json({ message: 'Term deleted successfully' });
```

```
  });
});

// Route to update a term
app.put('/terms/:term_id', (req, res) => {
  const { term_id } = req.params;
  const { term, definition } = req.body;
  const query = 'UPDATE Terms SET term = ?, definition = ? WHERE term_id = ?';
  db.query(query, [term, definition, term_id], (err, result) => {
    if (err) {
      return res.status(500).json({ error: err.message });
    }
    res.status(200).json({ message: 'Term updated successfully' });
  });
});

// Route to delete a study set and associated terms
app.delete('/study-sets/:set_id', (req, res) => {
  const { set_id } = req.params;

  executeTransaction(db, (err, connection, finalizeTransaction) => {
    if (err) {
      return res.status(500).json({ error: 'Database connection error' });
    }

    const deleteTermsQuery = 'DELETE FROM Terms WHERE set_id = ?';
    connection.query(deleteTermsQuery, [set_id], (err, result) => {
      if (err) {
        return finalizeTransaction(err);
```

```
        }

        const deleteStudySetQuery = 'DELETE FROM StudySets WHERE set_id = ?';
        connection.query(deleteStudySetQuery, [set_id], (err, result) => {
          if (err) {
            return finalizeTransaction(err);
          }

          // If all queries succeed, finalize the transaction
          finalizeTransaction(null);
          res.status(200).json({ message: 'Study set and associated terms deleted successfully' });
        });
      });
    });
});


// Route to get recent study sets for the current user
app.get('/recent-study-sets', (req, res) => {
  const username = req.session.user.username;
  if (!username) {
    return res.status(401).json({ error: 'Unauthorized' });
  }

  const query = `
    SELECT s.set_id, s.set_name, MAX(v.visit_timestamp) AS latest_visit
    FROM VisitHistory v
    JOIN StudySets s ON v.set_id = s.set_id
    WHERE v.username = ?
```

```
  GROUP BY s.set_id, s.set_name
  ORDER BY latest_visit DESC
  LIMIT 50
 `;


 db.query(query, [username], (err, results) => {
    if (err) {
       console.error('Error fetching recent study sets:', err);
       return res.status(500).json({ error: err.message });
    }
    res.status(200).json(results);
 });
});

// Route to add a visit to the visit history
app.post('/update-visit-history', (req, res) => {
 const username = req.session.user.username;
 const { set_id } = req.body;

 if (!username || !set_id) {
    return res.status(400).json({ error: 'Bad Request' });
 }

 const query = `
  INSERT INTO VisitHistory (username, set_id, visit_timestamp)
  VALUES (?, ?, NOW())
 `;
```

```
    db.query(query, [username, set_id], (err, results) => {
      if (err) {
        return res.status(500).json({ error: err.message });
      }
      res.status(200).send('Visit history updated');
    });
  });



//-----learning status routes---------
  //get status for specific term and user
  app.get('/get-term-status', (req, res) => {
    const { username, termId } = req.query;
    const query = `
      SELECT status
      FROM TermStatus
      WHERE username = ? AND term_id = ?;
    `;
    db.query(query, [username, termId], (err, results) => {
      if (err) {
        console.error(err);
        res.status(500).send({ message: 'Error retrieving term status' });
      } else if (results.length > 0) {
        res.send({ status: results[0].status });
      } else {
        res.send({ status: 0 }); // Default to 'unknown'
      }
    });
```

```
  });

  app.post('/update-term-status', (req, res) => {
    const { termId, username, status } = req.body;

    const query = `
      INSERT INTO TermStatus (username, term_id, status)
      VALUES (?, ?, ?)
      ON DUPLICATE KEY UPDATE status = VALUES(status);
    `;

    db.query(query, [username, termId, status], (err) => {
      if (err) {
        console.error('Error updating term status:', err);
        res.status(500).send({ message: 'Failed to update term status' });
      } else {
        res.send({ message: 'Term status updated successfully' });
      }
    });
  });



//----copy study set------
app.post('/copy-study-set', (req, res) => {
  const username = req.session.user?.username;
  const { set_id } = req.body;

  if (!username || !set_id) {
    return res.status(400).json({ error: 'Bad Request' });
```

```
  }

  // Fetch the original study set details
  const getOriginalSetQuery = 'SELECT set_name, category FROM StudySets WHERE set_id = ?';
  db.query(getOriginalSetQuery, [set_id], (err, results) => {
    if (err) {
      return res.status(500).json({ error: err.message });
    }

    if (results.length === 0) {
      return res.status(404).json({ error: 'Study set not found' });
    }

    const originalSet = results[0];
    const newSetName = `${originalSet.set_name} (${username}'s copy)`;

    // Insert a new study set for the current user
    const createNewSetQuery = 'INSERT INTO StudySets (username, set_name, category) VALUES
(?, ?, ?)';
    db.query(createNewSetQuery, [username, newSetName, originalSet.category], (err, result) => {
      if (err) {
        return res.status(500).json({ error: err.message });
      }

      const newSetId = result.insertId;

      // Fetch terms from the original set
      const getTermsQuery = 'SELECT term, definition FROM Terms WHERE set_id = ?';
      db.query(getTermsQuery, [set_id], (err, terms) => {
```

```
            if (err) {
                return res.status(500).json({ error: err.message });
            }

            if (terms.length === 0) {
                return res.status(200).json({
                    message: 'Study set copied successfully!',
                    newSetId
                });
            }

            // Insert terms into the new study set
            const insertTermsQuery = 'INSERT INTO Terms (set_id, term, definition) VALUES ?';
            const termsValues = terms.map(term => [newSetId, term.term, term.definition]);

            db.query(insertTermsQuery, [termsValues], (err) => {
                if (err) {
                    return res.status(500).json({ error: err.message });
                }

                res.status(200).json({
                    message: 'Study set copied successfully!',
                    newSetId
                });
            });
        });
    });
});
```

```
};

----------------------------------------



USERROUTES.JS ----------------------------
const bcrypt = require('bcrypt');

module.exports = function(app, db)
{
 // Route to handle adding a new user account
 app.post('/add-user', async (req, res) => {
  const { username, password } = req.body;

  try {
   const hashedPassword = await bcrypt.hash(password, 10);
   const query = 'INSERT INTO user_credentials (username, password) VALUES (?, ?)';
   db.query(query, [username, hashedPassword], (err, results) => {
    if (err) {
     console.error('Error inserting user credentials:', err);
     res.status(500).json({ error: 'Database error' });
    } else {
     res.status(200).json({ message: 'User credentials added successfully', userId: results.insertId
});
    }
   });
```

```
  } catch (err) {
    res.status(500).send('Server error');
  }
 });

  // Route to check if user is authenticated
 app.get('/check-auth', (req, res) => {
  if (req.session.user) {
    res.json({ loggedIn: true, username: req.session.user.username });
  } else {
    res.json({ loggedIn: false });
  }
 });

 //just get current username
 app.get('/current-user', (req, res) => {
  const currentUser = req.session?.user || null; // Retrieve user or return null
  res.json({ user: currentUser });
});



//get user info
 app.get('/api/user/info', (req, res) => {
    if (!req.session.user) {
       return res.status(401).json({ error: 'Unauthorized' });
    }

    const username = req.session.user.username;
    const query = 'SELECT username, created_at FROM user_credentials WHERE username = ?';
```

```
  db.query(query, [username], (err, results) => {
    if (err) {
      console.error('Error fetching user info:', err);
      return res.status(500).json({ error: 'Database error' });
    }

    if (results.length === 0) {
      return res.status(404).json({ error: 'User not found' });
    }

    res.json(results[0]);
  });
});



// Route to check for the presence of a user account in the database
app.post('/check-username', async (req, res) => {
 const { username } = req.body;
 const query = 'SELECT * FROM user_credentials WHERE username = ?';
 db.query(query, [username], async (err, results) => {
   if (err) {
     return res.status(500).send('Server error');
   }
   if (results.length != 0) {
     return res.status(401).send('This username exists');
   }
   return res.status(200).send('Username does not exist');
```

```
  });
});


  // Route to handle user login
  app.post('/login', async (req, res) =>
  {
  const { username, password } = req.body;
  const query = 'SELECT * FROM user_credentials WHERE username = ?'; //first queries to match entry with that username
  db.query(query, [username], async (err, results) =>
  {
    if (err)
    {
      return res.status(500).send('Server error');  //exits on error
    }
    if (results.length === 0)
    {
      return res.status(401).send('Invalid username or password');  //exits if no entry found
    }
    const user = results[0];  //sets user to value of found username
    const isPasswordValid = await bcrypt.compare(password, user.password);  //using the hashing library, compares input password to stored password
    if (!isPasswordValid) {


      return res.status(401).send('Invalid username or password');  //if not valid, exits
    }


    //after passing all those checks, a session is created, and the user is logged in, and success sent back to scripts in frontend
    req.session.user = user;
```

```
      res.send('Login successful');
    });
    });

    // User Logout Route
   app.post('/logout', (req, res) => {
    req.session.destroy(err => {
      if (err) {
        return res.status(500).send('Server error');
      }
      res.clearCookie('connect.sid', { path: '/' }); // Clear the session cookie
      res.send('Logout successful');
    });
   });

// Route to handle changing password
app.post('/change-password', async (req, res) => {
  const username = req.session.user.username;
  if (!username) {
      return res.status(401).json({ error: 'Unauthorized' });
  }
  const { password } = req.body;


  try {
      const hashedPassword = await bcrypt.hash(password, 10);
      const query = 'UPDATE user_credentials SET password = ? WHERE username = ?';
      db.query(query, [hashedPassword, username], (err, results) => {
          if (err) {
```

```
            console.error('Error updating user password:', err);
            res.status(500).json({ error: 'Database error' });
        } else {
            res.status(200).json({ message: 'Password changed successfully' });
        }
    });
  } catch (err) {
    res.status(500).send('Server error');
  }
});


  // Route to handle deleting account
  app.post('/delete-account', (req, res) => {
   if (!req.session.user.username) {
    return res.status(401).json({ error: 'Unauthorized' });
   }

   const username = req.session.user.username;

   console.log(` Attempting to delete account for user: ${username}` );

   const query = 'DELETE FROM user_credentials WHERE username = ?';
   db.query(query, [username], (err, results) => {
    if (err) {
      console.error('Error deleting user account:', err);
      res.status(500).json({ error: 'Database error' });
    }
    else if (results.affectedRows === 0) {
```

```
      console.warn(` No account found for user: ${username}` );
      res.status(404).json({ message: 'Account not found' });
    }
    else {
      console.log(` Account for user ${username} deleted successfully` );
      req.session.destroy((err) => {
        if (err) {
          console.error('Error destroying session:', err);
        }
        res.status(200).json({ message: 'Account deleted successfully' });
      });
    }
  });
 });
}
```

---

## FRONTEND JAVASCRIPT

CHANGEUSERNAME.JS ----------

```
const express = require('express');
const router = express.Router();
const sql = require('mssql');
const { connectToDatabase } = require('../dbConfig');

router.post('/change-username', async (req, res) => {
  const { username } = req.body;
   if (!username) {
     return res.status(400).json({ message: 'Please enter a username' });
   }
```

```
    try {
      await connectToDatabase();
      const request = new sql.Request();
      const query = ` UPDATE Users SET username = '${username}' WHERE id = ${req.user.id}` ;
      request.input('username', sql.VarChar, username);
      request.input('userId', sql.Int, req.session.userId);
      await request.query(query);

      res.json({ success: true});
    }
    catch (err) {
      console.error(err);
      res.status(500).json({ success: false, message: 'An error occurred while updating your
username. Please try again.' });
    }
   });
   module.exports = router;
```

---

EXPLORE.JS ----------

```
// Function to get query parameter value by name
function getQueryParameter(name) {
   const urlParams = new URLSearchParams(window.location.search);
   return urlParams.get(name);
}

// Fetch and display search results
```

```
function fetchSearchResults(query, categories = []) {
  console.log('Fetching search results for query:', query, 'with categories:', categories); // Debugging log


  // Ensure that categories are sent as an array
  fetch('/study-sets/search', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({ set_name: query, categories }) // Send both query and categories
  })
  .then(response => {
    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }
    return response.json();
  })
  .then(data => {
    console.log('Received search results:', data); // Debugging log
    if (Array.isArray(data)) {
      displaySearchResults(data); // Use your existing function to display the results
    } else {
      console.error('Unexpected response format:', data);
    }
  })
  .catch(error => {
    console.error('Error fetching search results:', error); // Handle network or backend errors
  });
```

```
    console.log('Sending data to backend:', {
        set_name: query,
        categories: categories,
    });

}


// Display search results
function displaySearchResults(results) {
    const searchResultDiv = document.getElementById('searchResult');
    searchResultDiv.innerHTML = ''; // Clear previous results

    if (results.length === 0) {
        searchResultDiv.textContent = 'No results found.';
        return;
    }

    results.forEach(result => {
        const resultItem = document.createElement('div');
        resultItem.classList.add('result-item');
        resultItem.innerHTML = `
            <h3>${result.set_name}</h3>
            <p>By: ${result.username}</p>
            <p>Created: ${new Date(result.created_at).toLocaleString()}</p>
            <p>Subject: ${result.category}</p>
        `;
        // Add onclick event to update visit history and redirect
        resultItem.style.cursor = 'pointer';
```

```
        resultItem.onclick = () => {
            // Send request to update visit history
            fetch('/update-visit-history', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({ set_id: result.set_id })
            }).then(() => {
                window.location.href = `studySetTemplate.html?id=${result.set_id}`;
            }).catch(error => {
                console.error('Error updating visit history:', error);
            });
        };

        searchResultDiv.appendChild(resultItem);
    });
}

// Use the function to retrieve the 'search' query parameter and fetch search results
document.addEventListener("DOMContentLoaded", function() {
    const searchbar = document.getElementById("exploreSearchbar"); // The search bar on the explore page
    const searchQuery = getQueryParameter('search'); // Extract 'search' query parameter

    if (searchQuery) {
        searchbar.value = decodeURIComponent(searchQuery); // Prefill the search bar with the query
        fetchSearchResults(searchQuery, getAllCategories()); // Fetch results with the query
    }
```

```
  // Add an event listener for the Enter key
  searchbar.addEventListener("keypress", function(event) {
    if (event.keyCode === 13) {
      const query = searchbar.value.trim();
      if (query) {
        fetchSearchResults(query, getSelectedCategories());
      }
    }
  });
});

// Helper function to get query parameters
function getQueryParameter(name) {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get(name);
}

// Helper function to get all categories (if all are selected by default)
function getAllCategories() {
  return Array.from(document.querySelectorAll('.category-checkbox')).map(checkbox =>
checkbox.value);
}

// Helper function to get selected categories
function getSelectedCategories() {
  return Array.from(document.querySelectorAll('.category-checkbox:checked')).map(checkbox =>
checkbox.value);
}
```

```
//handle the category selector
document.addEventListener('DOMContentLoaded', () => {
  const popupOverlay = document.getElementById('popupOverlay');
  const openPopupBtn = document.getElementById('openPopupBtn');
  const closePopupBtn = document.getElementById('closePopupBtn');
  const selectAllBtn = document.getElementById('selectAllBtn');
  const deselectAllBtn = document.getElementById('deselectAllBtn');
  const applyFiltersBtn = document.getElementById('applyFiltersBtn');
  const categoryCheckboxes = document.querySelectorAll('.category-checkbox');

  // Open the popup
  openPopupBtn.addEventListener('click', () => {
    popupOverlay.style.display = 'flex';
  });

  // Close the popup
  closePopupBtn.addEventListener('click', () => {
    popupOverlay.style.display = 'none';
  });

  // Select All Categories
  selectAllBtn.addEventListener('click', () => {
    categoryCheckboxes.forEach(checkbox => {
      checkbox.checked = true;
    });
  });
```

```
  // Deselect All Categories
  deselectAllBtn.addEventListener('click', () => {
    categoryCheckboxes.forEach(checkbox => {
      checkbox.checked = false;
    });
  });

  // Apply Filters (optional: log selected categories)
  applyFiltersBtn.addEventListener('click', () => {
    const selectedCategories = Array.from(categoryCheckboxes)
      .filter(checkbox => checkbox.checked)
      .map(checkbox => checkbox.value);
    console.log('Selected Categories:', selectedCategories);

    // Optionally pass these categories to your search function
    // fetchSearchResultsWithCategories(selectedCategories);

    popupOverlay.style.display = 'none'; // Close popup after applying
  });
});

async function fetchExploreStudySets(searchTerm) {
  try {
    const response = await fetch('/study-sets/explore', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ set_name: searchTerm }),
```

```
      });

      if (!response.ok) {
        throw new Error('Failed to fetch study sets');
      }

      const studySets = await response.json();
      displayExploreStudySets(studySets);
    } catch (error) {
      console.error('Error fetching study sets:', error);
    }
}

function displayExploreStudySets(studySets) {
    const exploreContainer = document.getElementById('explore-container');
    exploreContainer.innerHTML = ''; // Clear previous results

    if (studySets.length === 0) {
      exploreContainer.innerHTML = '<p>No study sets found.</p>';
      return;
    }

    studySets.forEach((set) => {
      const setElement = document.createElement('div');
      setElement.className = 'study-set';

      setElement.innerHTML = `
        <h3>${set.set_name}</h3>
        <p>${set.description}</p>
```

```
        <p><strong>Created by:</strong> ${set.username}</p>
      `;

      exploreContainer.appendChild(setElement);
    });
}
```

---

INDEX.JS ----------

```
// Function to get query parameter value by name
function getQueryParameter(name) {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get(name);
}

// Fetch and display search results
function fetchSearchResults(query, categories = []) {
  console.log('Fetching search results for query:', query, 'with categories:', categories); // Debugging log

  // Ensure that categories are sent as an array
  fetch('/study-sets/search', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
```

```
    body: JSON.stringify({ set_name: query, categories }) // Send both query and categories
  })
  .then(response => {
    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }
    return response.json();
  })
  .then(data => {
    console.log('Received search results:', data); // Debugging log
    if (Array.isArray(data)) {
      displaySearchResults(data); // Use your existing function to display the results
    } else {
      console.error('Unexpected response format:', data);
    }
  })
  .catch(error => {
    console.error('Error fetching search results:', error); // Handle network or backend errors
  });
  console.log('Sending data to backend:', {
    set_name: query,
    categories: categories,
  });

}


// Display search results
function displaySearchResults(results) {
```

```
const searchResultDiv = document.getElementById('searchResult');
searchResultDiv.innerHTML = ''; // Clear previous results

if (results.length === 0) {
  searchResultDiv.textContent = 'No results found.';
  return;
}

results.forEach(result => {
  const resultItem = document.createElement('div');
  resultItem.classList.add('result-item');
  resultItem.innerHTML = `
    <h3>${result.set_name}</h3>
    <p>By: ${result.username}</p>
    <p>Created: ${new Date(result.created_at).toLocaleString()}</p>
    <p>Subject: ${result.category}</p>
  `;
  // Add onclick event to update visit history and redirect
  resultItem.style.cursor = 'pointer';
  resultItem.onclick = () => {
    // Send request to update visit history
    fetch('/update-visit-history', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ set_id: result.set_id })
    }).then(() => {
      window.location.href = `studySetTemplate.html?id=${result.set_id}`;
```

```
        }).catch(error => {
            console.error('Error updating visit history:', error);
        });
    };

    searchResultDiv.appendChild(resultItem);
  });
}

// Use the function to retrieve the 'search' query parameter and fetch search results
document.addEventListener("DOMContentLoaded", function() {
  const searchbar = document.getElementById("exploreSearchbar"); // The search bar on the explore page
  const searchQuery = getQueryParameter('search'); // Extract 'search' query parameter

  if (searchQuery) {
    searchbar.value = decodeURIComponent(searchQuery); // Prefill the search bar with the query
    fetchSearchResults(searchQuery, getAllCategories()); // Fetch results with the query
  }

  // Add an event listener for the Enter key
  searchbar.addEventListener("keypress", function(event) {
    if (event.keyCode === 13) {
      const query = searchbar.value.trim();
      if (query) {
        fetchSearchResults(query, getSelectedCategories());
      }
    }
  });
```

```
});

// Helper function to get query parameters
function getQueryParameter(name) {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get(name);
}

// Helper function to get all categories (if all are selected by default)
function getAllCategories() {
  return Array.from(document.querySelectorAll('.category-checkbox')).map(checkbox =>
checkbox.value);
}

// Helper function to get selected categories
function getSelectedCategories() {
  return Array.from(document.querySelectorAll('.category-checkbox:checked')).map(checkbox =>
checkbox.value);
}



//handle the category selector
document.addEventListener('DOMContentLoaded', () => {
  const popupOverlay = document.getElementById('popupOverlay');
  const openPopupBtn = document.getElementById('openPopupBtn');
  const closePopupBtn = document.getElementById('closePopupBtn');
  const selectAllBtn = document.getElementById('selectAllBtn');
  const deselectAllBtn = document.getElementById('deselectAllBtn');
  const applyFiltersBtn = document.getElementById('applyFiltersBtn');
```

```
const categoryCheckboxes = document.querySelectorAll('.category-checkbox');

// Open the popup
openPopupBtn.addEventListener('click', () => {
  popupOverlay.style.display = 'flex';
});

// Close the popup
closePopupBtn.addEventListener('click', () => {
  popupOverlay.style.display = 'none';
});

// Select All Categories
selectAllBtn.addEventListener('click', () => {
  categoryCheckboxes.forEach(checkbox => {
    checkbox.checked = true;
  });
});

// Deselect All Categories
deselectAllBtn.addEventListener('click', () => {
  categoryCheckboxes.forEach(checkbox => {
    checkbox.checked = false;
  });
});

// Apply Filters (optional: log selected categories)
applyFiltersBtn.addEventListener('click', () => {
  const selectedCategories = Array.from(categoryCheckboxes)
```

```
            .filter(checkbox => checkbox.checked)
            .map(checkbox => checkbox.value);
        console.log('Selected Categories:', selectedCategories);

        // Optionally pass these categories to your search function
        // fetchSearchResultsWithCategories(selectedCategories);

        popupOverlay.style.display = 'none'; // Close popup after applying
    });
});

async function fetchExploreStudySets(searchTerm) {
    try {
        const response = await fetch('/study-sets/explore', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ set_name: searchTerm }),
        });

        if (!response.ok) {
            throw new Error('Failed to fetch study sets');
        }

        const studySets = await response.json();
        displayExploreStudySets(studySets);
    } catch (error) {
        console.error('Error fetching study sets:', error);
```

```
    }
}

function displayExploreStudySets(studySets) {
    const exploreContainer = document.getElementById('explore-container');
    exploreContainer.innerHTML = ''; // Clear previous results

    if (studySets.length === 0) {
        exploreContainer.innerHTML = '<p>No study sets found.</p>';
        return;
    }

    studySets.forEach((set) => {
        const setElement = document.createElement('div');
        setElement.className = 'study-set';

        setElement.innerHTML = `
            <h3>${set.set_name}</h3>
            <p>${set.description}</p>
            <p><strong>Created by:</strong> ${set.username}</p>
        `;

        exploreContainer.appendChild(setElement);
    });
}

----------------------------------------
```

LOGIN.JS ---------------------------------

```
// Function to get query parameter value by name
function getQueryParameter(name) {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get(name);
}

// Fetch and display search results
function fetchSearchResults(query, categories = []) {
  console.log('Fetching search results for query:', query, 'with categories:', categories); // Debugging log

  // Ensure that categories are sent as an array
  fetch('/study-sets/search', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({ set_name: query, categories }) // Send both query and categories
  })
  .then(response => {
    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }
    return response.json();
  })
  .then(data => {
    console.log('Received search results:', data); // Debugging log
```

```
        if (Array.isArray(data)) {
            displaySearchResults(data); // Use your existing function to display the results
        } else {
            console.error('Unexpected response format:', data);
        }
    })
    .catch(error => {
        console.error('Error fetching search results:', error); // Handle network or backend errors
    });
    console.log('Sending data to backend:', {
        set_name: query,
        categories: categories,
    });

}


// Display search results
function displaySearchResults(results) {
    const searchResultDiv = document.getElementById('searchResult');
    searchResultDiv.innerHTML = ''; // Clear previous results

    if (results.length === 0) {
        searchResultDiv.textContent = 'No results found.';
        return;
    }

    results.forEach(result => {
        const resultItem = document.createElement('div');
```

```
        resultItem.classList.add('result-item');
        resultItem.innerHTML = `
            <h3>${result.set_name}</h3>
            <p>By: ${result.username}</p>
            <p>Created: ${new Date(result.created_at).toLocaleString()}</p>
            <p>Subject: ${result.category}</p>
        `;
        // Add onclick event to update visit history and redirect
        resultItem.style.cursor = 'pointer';
        resultItem.onclick = () => {
            // Send request to update visit history
            fetch('/update-visit-history', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({ set_id: result.set_id })
            }).then(() => {
                window.location.href = `studySetTemplate.html?id=${result.set_id}`;
            }).catch(error => {
                console.error('Error updating visit history:', error);
            });
        };

        searchResultDiv.appendChild(resultItem);
    });
}

// Use the function to retrieve the 'search' query parameter and fetch search results
```

```
document.addEventListener("DOMContentLoaded", function() {
  const searchbar = document.getElementById("exploreSearchbar"); // The search bar on the explore page
  const searchQuery = getQueryParameter('search'); // Extract 'search' query parameter

  if (searchQuery) {
    searchbar.value = decodeURIComponent(searchQuery); // Prefill the search bar with the query
    fetchSearchResults(searchQuery, getAllCategories()); // Fetch results with the query
  }

  // Add an event listener for the Enter key
  searchbar.addEventListener("keypress", function(event) {
    if (event.keyCode === 13) {
      const query = searchbar.value.trim();
      if (query) {
        fetchSearchResults(query, getSelectedCategories());
      }
    }
  });
});

// Helper function to get query parameters
function getQueryParameter(name) {
  const urlParams = new URLSearchParams(window.location.search);
  return urlParams.get(name);
}

// Helper function to get all categories (if all are selected by default)
function getAllCategories() {
```

```
  return Array.from(document.querySelectorAll('.category-checkbox')).map(checkbox =>
checkbox.value);
}

// Helper function to get selected categories
function getSelectedCategories() {
  return Array.from(document.querySelectorAll('.category-checkbox:checked')).map(checkbox =>
checkbox.value);
}



//handle the category selector
document.addEventListener('DOMContentLoaded', () => {
  const popupOverlay = document.getElementById('popupOverlay');
  const openPopupBtn = document.getElementById('openPopupBtn');
  const closePopupBtn = document.getElementById('closePopupBtn');
  const selectAllBtn = document.getElementById('selectAllBtn');
  const deselectAllBtn = document.getElementById('deselectAllBtn');
  const applyFiltersBtn = document.getElementById('applyFiltersBtn');
  const categoryCheckboxes = document.querySelectorAll('.category-checkbox');

  // Open the popup
  openPopupBtn.addEventListener('click', () => {
    popupOverlay.style.display = 'flex';
  });

  // Close the popup
  closePopupBtn.addEventListener('click', () => {
    popupOverlay.style.display = 'none';
```

```
  });

  // Select All Categories
  selectAllBtn.addEventListener('click', () => {
    categoryCheckboxes.forEach(checkbox => {
      checkbox.checked = true;
    });
  });

  // Deselect All Categories
  deselectAllBtn.addEventListener('click', () => {
    categoryCheckboxes.forEach(checkbox => {
      checkbox.checked = false;
    });
  });

  // Apply Filters (optional: log selected categories)
  applyFiltersBtn.addEventListener('click', () => {
    const selectedCategories = Array.from(categoryCheckboxes)
      .filter(checkbox => checkbox.checked)
      .map(checkbox => checkbox.value);
    console.log('Selected Categories:', selectedCategories);

    // Optionally pass these categories to your search function
    // fetchSearchResultsWithCategories(selectedCategories);

    popupOverlay.style.display = 'none'; // Close popup after applying
  });
});
```

```
async function fetchExploreStudySets(searchTerm) {
  try {
    const response = await fetch('/study-sets/explore', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ set_name: searchTerm }),
    });

    if (!response.ok) {
      throw new Error('Failed to fetch study sets');
    }

    const studySets = await response.json();
    displayExploreStudySets(studySets);
  } catch (error) {
    console.error('Error fetching study sets:', error);
  }
}

function displayExploreStudySets(studySets) {
  const exploreContainer = document.getElementById('explore-container');
  exploreContainer.innerHTML = ''; // Clear previous results

  if (studySets.length === 0) {
    exploreContainer.innerHTML = '<p>No study sets found.</p>';
    return;
```

```
  }

  studySets.forEach((set) => {
    const setElement = document.createElement('div');
    setElement.className = 'study-set';

    setElement.innerHTML = `
      <h3>${set.set_name}</h3>
      <p>${set.description}</p>
      <p><strong>Created by:</strong> ${set.username}</p>
    `;

    exploreContainer.appendChild(setElement);
  });
}
```

---

MYSCRIPT.JS ----------

```
//debug statement
console.log('myscript.js is loaded');     //debug, remove later


/*
Checking if the user is logged in and displaying correct nav button
*/
document.addEventListener("DOMContentLoaded", () => {
```

```
  fetch('/check-auth')
    .then(response => response.json())
    .then(data => {
      const authLink = document.getElementById('auth-link');
      const altRegisterLink = document.getElementById('altRegisterLink');

      if (data.loggedIn) {
        authLink.innerHTML = `<a href="profile.html">Account: ${data.username}</a>`;
        altRegisterLink.style.visibility = "hidden"; // Hides element without affecting layout
      } else {
        altRegisterLink.style.visibility = "visible"; // Shows element when logged out
      }
    })
    .catch(error => console.error('Error:', error));
});



/*
'Protected' pages
*/
//function to check if user logged in
function checkAuth() {
  fetch('/check-auth')
  .then(response => response.json())
  .then(data => {
    if (!data.loggedIn) {
      // Redirect to login page if not authenticated
      window.location.href = '../html/login.html';
```

```
      }
    })
    .catch(error => {
      console.error('Error checking authentication:', error);
    });
}
//on page load...
document.addEventListener('DOMContentLoaded', () => {
    //set of 'protected' pages
    const protectedPages = [
      'profile.html',
      //'studySetTemplate.html',
      'userLibrary.html'
    ];

    //get current page
    const currentPage = window.location.pathname.split('/').pop();
    console.log(currentPage);

    //if current page is protected
    if (protectedPages.includes(currentPage)) {
      checkAuth();
    }
});
```

---

PROFILE.JS ---------------------------------

```
//script to handle logging out
document.addEventListener('DOMContentLoaded', () => {
  const logoutBtn = document.getElementById('logoutBtn');



  logoutBtn?.removeEventListener('click', handleLogout);
  logoutBtn?.addEventListener('click', handleLogout, { once: true });
});

function handleLogout() {
  const userConfirmed = confirm('Are you sure you want to log out?');

  if (userConfirmed) {
    console.log("Logging out...");

    fetch('/logout', { method: 'POST' })
      .then(response => response.text())
      .then(message => {
        alert(message);
        window.location.href = '../html/login.html';
      })
      .catch(error => {
        console.error('Error during logout:', error);
        alert('An error occurred during logout. Please try again.');
      });
  }
}
```

```
document.addEventListener('DOMContentLoaded', function() {
  const passwordChangeContainer = document.getElementById('passwordChangeContainer');
  const changePasswordBtn = document.getElementById('changePasswordBtn');

  // Function to create the password change form
  function createPasswordChangeForm() {
    const formHTML = `
      <form id="changePasswordForm">
        <label for="new_password">New Password:</label>
        <input type="password" id="new_password" required>
        <br>
        <label for="confirm_password">Confirm Password:</label>
        <input type="password" id="confirm_password" required>
        <br>
        <button type="submit">Change Password</button>
      </form>
    `;
    passwordChangeContainer.innerHTML = formHTML;

    // Add event listener to the form
    const changePasswordForm = document.getElementById('changePasswordForm');
    changePasswordForm.addEventListener('submit', function(event) {
      event.preventDefault();

      const newPassword = document.getElementById('new_password').value;
      const confirmPassword = document.getElementById('confirm_password').value;
```

```
        // Password validation regex
        const passwordRegex = /^[a-zA-Z0-9!@#$%^&*()_+]{6,20}$/;

        // Check if passwords match
        if (newPassword !== confirmPassword) {
            alert('Passwords do not match.');
            return;
        }

        // Check if passwords meet the criteria
        if (!passwordRegex.test(newPassword)) {
            alert('Password must be 6-20 characters long and contain only letters, numbers, and "!@#$%^&*()_"');
            return;
        }

        // Fetch request to change password
        fetch('/change-password', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ password: newPassword }),
        })
        .then(response => response.json())
        .then(data => {
            alert(data.message);
            passwordChangeContainer.innerHTML = ''; // Clear the form on success
        })
```

```
        .catch(error => {
          console.error('Error:', error);
        });
      });
  }

  // Add event listener to the button
  changePasswordBtn.addEventListener('click', createPasswordChangeForm);
});
```

```
document.addEventListener('DOMContentLoaded', function() {
  const deleteAccountBtn = document.getElementById('deleteAccountBtn');
  if (deleteAccountBtn) {
    deleteAccountBtn.addEventListener('click', function() {
      const userConfirmed = confirm('Are you sure you want to delete your account?');

      if (userConfirmed) {
        fetch('/delete-account', {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json',
          }
        })
        .then(response => response.json())
        .then(data => {
```

```
            alert(data.message);
            if (data.message === 'Account deleted successfully') {
              fetch('/logout', {
                method: 'POST',
                headers: {
                  'Content-Type': 'application/json',
                }
              })
              .then(response => response.text())
              .then(data => {
                console.log(data);
                if (data === 'Logout successful') {
                  window.location.href = '../html/login.html'; // Redirect to signup page after logout
                } else {
                  alert(data); // Display error message
                }
              })
              .catch(error => {
                console.error('Error during logout:', error);
              });
            }
          })
          .catch(error => {
            console.error('Error:', error);
          });
        }
      });
    }
});
```

```
//section which fetches and updates user info
var userData = null; //variable to store the fetched user info

document.addEventListener('DOMContentLoaded', async function () {
  await fetchUserInfo();
  updateUserInfo();
});

async function fetchUserInfo() {
  try {
    const response = await fetch('/api/user/info', {
      method: 'GET',
      credentials: 'include'
    });

    if (!response.ok) {
      throw new Error('Failed to fetch user info');
    }

    userData = await response.json();
    console.log('Fetched Data:', userData);
  } catch (error) {
    console.error('Error fetching user info:', error);
    userData = { name: 'Guest', joinDate: 'N/A' }; // Fallback values
  }
}
function updateUserInfo() {
```

```
  const nameEl = document.getElementById('userName');
  const dateEl = document.getElementById('userJoinDate');

  if (nameEl && dateEl && userData) {
    nameEl.textContent = userData.username;
    dateEl.textContent = new Date(userData.created_at).toLocaleDateString();
  }
}
```

---

SEARCHBAR.JS ----------

```
//script to handle logging out
document.addEventListener('DOMContentLoaded', () => {
  const logoutBtn = document.getElementById('logoutBtn');


  logoutBtn?.removeEventListener('click', handleLogout);
  logoutBtn?.addEventListener('click', handleLogout, { once: true });
});

function handleLogout() {
  const userConfirmed = confirm('Are you sure you want to log out?');

  if (userConfirmed) {
    console.log("Logging out...");
```

```
    fetch('/logout', { method: 'POST' })
      .then(response => response.text())
      .then(message => {
        alert(message);
        window.location.href = '../html/login.html';
      })
      .catch(error => {
        console.error('Error during logout:', error);
        alert('An error occurred during logout. Please try again.');
      });
  }
}



document.addEventListener('DOMContentLoaded', function() {
  const passwordChangeContainer = document.getElementById('passwordChangeContainer');
  const changePasswordBtn = document.getElementById('changePasswordBtn');

  // Function to create the password change form
  function createPasswordChangeForm() {
    const formHTML = `
      <form id="changePasswordForm">
        <label for="new_password">New Password:</label>
        <input type="password" id="new_password" required>
        <br>
        <label for="confirm_password">Confirm Password:</label>
        <input type="password" id="confirm_password" required>
        <br>
```

```
        <button type="submit">Change Password</button>
      </form>
    `;
    passwordChangeContainer.innerHTML = formHTML;

    // Add event listener to the form
    const changePasswordForm = document.getElementById('changePasswordForm');
    changePasswordForm.addEventListener('submit', function(event) {
      event.preventDefault();

      const newPassword = document.getElementById('new_password').value;
      const confirmPassword = document.getElementById('confirm_password').value;

      // Password validation regex
      const passwordRegex = /^[a-zA-Z0-9!@#$%^&*()_+]{6,20}$/;

      // Check if passwords match
      if (newPassword !== confirmPassword) {
        alert('Passwords do not match.');
        return;
      }

      // Check if passwords meet the criteria
      if (!passwordRegex.test(newPassword)) {
        alert('Password must be 6-20 characters long and contain only letters, numbers, and "!@#$%^&*()_".');
        return;
      }
```

```
        // Fetch request to change password
        fetch('/change-password', {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json',
          },
          body: JSON.stringify({ password: newPassword }),
        })
        .then(response => response.json())
        .then(data => {
          alert(data.message);
          passwordChangeContainer.innerHTML = ''; // Clear the form on success
        })
        .catch(error => {
          console.error('Error:', error);
        });
      });
    }

    // Add event listener to the button
    changePasswordBtn.addEventListener('click', createPasswordChangeForm);
});
```

```
document.addEventListener('DOMContentLoaded', function() {
    const deleteAccountBtn = document.getElementById('deleteAccountBtn');
```

```
if (deleteAccountBtn) {
  deleteAccountBtn.addEventListener('click', function() {
    const userConfirmed = confirm('Are you sure you want to delete your account?');

    if (userConfirmed) {
      fetch('/delete-account', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
        }
      })
      .then(response => response.json())
      .then(data => {
        alert(data.message);
        if (data.message === 'Account deleted successfully') {
          fetch('/logout', {
            method: 'POST',
            headers: {
              'Content-Type': 'application/json',
            }
          })
          .then(response => response.text())
          .then(data => {
            console.log(data);
            if (data === 'Logout successful') {
              window.location.href = '../html/login.html'; // Redirect to signup page after logout
            } else {
              alert(data); // Display error message
            }
```

```
                })
                .catch(error => {
                    console.error('Error during logout:', error);
                });
            }
        })
        .catch(error => {
            console.error('Error:', error);
        });
    }
  });
}
});


//section which fetches and updates user info
var userData = null; //variable to store the fetched user info

document.addEventListener('DOMContentLoaded', async function () {
  await fetchUserInfo();
  updateUserInfo();
});

async function fetchUserInfo() {
  try {
    const response = await fetch('/api/user/info', {
      method: 'GET',
      credentials: 'include'
    });
```

```
      if (!response.ok) {
        throw new Error('Failed to fetch user info');
      }

      userData = await response.json();
      console.log('Fetched Data:', userData);
    } catch (error) {
      console.error('Error fetching user info:', error);
      userData = { name: 'Guest', joinDate: 'N/A' }; // Fallback values
    }
}
function updateUserInfo() {
    const nameEl = document.getElementById('userName');
    const dateEl = document.getElementById('userJoinDate');

    if (nameEl && dateEl && userData) {
      nameEl.textContent = userData.username;
      dateEl.textContent = new Date(userData.created_at).toLocaleDateString();
    }
}
```

---

SIGNUP.JS ----------

```
//script to handle logging out
document.addEventListener('DOMContentLoaded', () => {
```

```
  const logoutBtn = document.getElementById('logoutBtn');



  logoutBtn?.removeEventListener('click', handleLogout);
  logoutBtn?.addEventListener('click', handleLogout, { once: true });
});

function handleLogout() {
  const userConfirmed = confirm('Are you sure you want to log out?');

  if (userConfirmed) {
    console.log("Logging out...");

    fetch('/logout', { method: 'POST' })
      .then(response => response.text())
      .then(message => {
        alert(message);
        window.location.href = '../html/login.html';
      })
      .catch(error => {
        console.error('Error during logout:', error);
        alert('An error occurred during logout. Please try again.');
      });
  }
}



document.addEventListener('DOMContentLoaded', function() {
```

```
const passwordChangeContainer = document.getElementById('passwordChangeContainer');
const changePasswordBtn = document.getElementById('changePasswordBtn');

// Function to create the password change form
function createPasswordChangeForm() {
  const formHTML = `
    <form id="changePasswordForm">
      <label for="new_password">New Password:</label>
      <input type="password" id="new_password" required>
      <br>
      <label for="confirm_password">Confirm Password:</label>
      <input type="password" id="confirm_password" required>
      <br>
      <button type="submit">Change Password</button>
    </form>
  `;
  passwordChangeContainer.innerHTML = formHTML;

  // Add event listener to the form
  const changePasswordForm = document.getElementById('changePasswordForm');
  changePasswordForm.addEventListener('submit', function(event) {
    event.preventDefault();

    const newPassword = document.getElementById('new_password').value;
    const confirmPassword = document.getElementById('confirm_password').value;

    // Password validation regex
    const passwordRegex = /^[a-zA-Z0-9!@#$%^&*()_+]{6,20}$/;
```

```
        // Check if passwords match
        if (newPassword !== confirmPassword) {
            alert('Passwords do not match.');
            return;
        }

        // Check if passwords meet the criteria
        if (!passwordRegex.test(newPassword)) {
            alert('Password must be 6-20 characters long and contain only letters, numbers, and "!@#$%^&*()_"');
            return;
        }

        // Fetch request to change password
        fetch('/change-password', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ password: newPassword }),
        })
        .then(response => response.json())
        .then(data => {
            alert(data.message);
            passwordChangeContainer.innerHTML = ''; // Clear the form on success
        })
        .catch(error => {
            console.error('Error:', error);
        });
```

```
    });
  }

  // Add event listener to the button
  changePasswordBtn.addEventListener('click', createPasswordChangeForm);
});
```

```
document.addEventListener('DOMContentLoaded', function() {
  const deleteAccountBtn = document.getElementById('deleteAccountBtn');
  if (deleteAccountBtn) {
    deleteAccountBtn.addEventListener('click', function() {
      const userConfirmed = confirm('Are you sure you want to delete your account?');

      if (userConfirmed) {
        fetch('/delete-account', {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json',
          }
        })
        .then(response => response.json())
        .then(data => {
          alert(data.message);
          if (data.message === 'Account deleted successfully') {
            fetch('/logout', {
```

```
            method: 'POST',
            headers: {
              'Content-Type': 'application/json',
            }
          })
          .then(response => response.text())
          .then(data => {
            console.log(data);
            if (data === 'Logout successful') {
              window.location.href = '../html/login.html'; // Redirect to signup page after logout
            } else {
              alert(data); // Display error message
            }
          })
          .catch(error => {
            console.error('Error during logout:', error);
          });
        }
      })
      .catch(error => {
        console.error('Error:', error);
      });
    }
  });
}
});


//section which fetches and updates user info
```

```
var userData = null; //variable to store the fetched user info

document.addEventListener('DOMContentLoaded', async function () {
  await fetchUserInfo();
  updateUserInfo();
});

async function fetchUserInfo() {
  try {
    const response = await fetch('/api/user/info', {
      method: 'GET',
      credentials: 'include'
    });

    if (!response.ok) {
      throw new Error('Failed to fetch user info');
    }

    userData = await response.json();
    console.log('Fetched Data:', userData);
  } catch (error) {
    console.error('Error fetching user info:', error);
    userData = { name: 'Guest', joinDate: 'N/A' }; // Fallback values
  }
}
function updateUserInfo() {
  const nameEl = document.getElementById('userName');
  const dateEl = document.getElementById('userJoinDate');
```

```
  if (nameEl && dateEl && userData) {
    nameEl.textContent = userData.username;
    dateEl.textContent = new Date(userData.created_at).toLocaleDateString();
  }
}
----------------------------------------
```

STUDYSETTEMPLATE.JS ----------

```
//script to handle logging out
document.addEventListener('DOMContentLoaded', () => {
  const logoutBtn = document.getElementById('logoutBtn');


  logoutBtn?.removeEventListener('click', handleLogout);
  logoutBtn?.addEventListener('click', handleLogout, { once: true });
});

function handleLogout() {
  const userConfirmed = confirm('Are you sure you want to log out?');

  if (userConfirmed) {
    console.log("Logging out...");

    fetch('/logout', { method: 'POST' })
      .then(response => response.text())
      .then(message => {
        alert(message);
```

```
            window.location.href = '../html/login.html';
        })
        .catch(error => {
            console.error('Error during logout:', error);
            alert('An error occurred during logout. Please try again.');
        });
    }
}



document.addEventListener('DOMContentLoaded', function() {
    const passwordChangeContainer = document.getElementById('passwordChangeContainer');
    const changePasswordBtn = document.getElementById('changePasswordBtn');

    // Function to create the password change form
    function createPasswordChangeForm() {
        const formHTML = `
            <form id="changePasswordForm">
                <label for="new_password">New Password:</label>
                <input type="password" id="new_password" required>
                <br>
                <label for="confirm_password">Confirm Password:</label>
                <input type="password" id="confirm_password" required>
                <br>
                <button type="submit">Change Password</button>
            </form>
        `;
        passwordChangeContainer.innerHTML = formHTML;
```

```
    // Add event listener to the form
    const changePasswordForm = document.getElementById('changePasswordForm');
    changePasswordForm.addEventListener('submit', function(event) {
      event.preventDefault();

      const newPassword = document.getElementById('new_password').value;
      const confirmPassword = document.getElementById('confirm_password').value;

      // Password validation regex
      const passwordRegex = /^[a-zA-Z0-9!@#$%^&*()_+]{6,20}$/;

      // Check if passwords match
      if (newPassword !== confirmPassword) {
        alert('Passwords do not match.');
        return;
      }

      // Check if passwords meet the criteria
      if (!passwordRegex.test(newPassword)) {
        alert('Password must be 6-20 characters long and contain only letters, numbers, and "!@#$%^&*()_"');
        return;
      }

      // Fetch request to change password
      fetch('/change-password', {
        method: 'POST',
        headers: {
```

```
            'Content-Type': 'application/json',
          },
          body: JSON.stringify({ password: newPassword }),
        })
        .then(response => response.json())
        .then(data => {
          alert(data.message);
          passwordChangeContainer.innerHTML = ''; // Clear the form on success
        })
        .catch(error => {
          console.error('Error:', error);
        });
      });
    }

    // Add event listener to the button
    changePasswordBtn.addEventListener('click', createPasswordChangeForm);
});



document.addEventListener('DOMContentLoaded', function() {
    const deleteAccountBtn = document.getElementById('deleteAccountBtn');
    if (deleteAccountBtn) {
        deleteAccountBtn.addEventListener('click', function() {
            const userConfirmed = confirm('Are you sure you want to delete your account?');
```

```
if (userConfirmed) {
  fetch('/delete-account', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    }
  })
  .then(response => response.json())
  .then(data => {
    alert(data.message);
    if (data.message === 'Account deleted successfully') {
      fetch('/logout', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
        }
      })
      .then(response => response.text())
      .then(data => {
        console.log(data);
        if (data === 'Logout successful') {
          window.location.href = '../html/login.html'; // Redirect to signup page after logout
        } else {
          alert(data); // Display error message
        }
      })
      .catch(error => {
        console.error('Error during logout:', error);
      });
```

```
            }
          })
          .catch(error => {
            console.error('Error:', error);
          });
        }
      });
  }
});


//section which fetches and updates user info
var userData = null; //variable to store the fetched user info

document.addEventListener('DOMContentLoaded', async function () {
  await fetchUserInfo();
  updateUserInfo();
});

async function fetchUserInfo() {
  try {
    const response = await fetch('/api/user/info', {
      method: 'GET',
      credentials: 'include'
    });

    if (!response.ok) {
      throw new Error('Failed to fetch user info');
    }
```

```
      userData = await response.json();
      console.log('Fetched Data:', userData);
    } catch (error) {
      console.error('Error fetching user info:', error);
      userData = { name: 'Guest', joinDate: 'N/A' }; // Fallback values
    }
}
function updateUserInfo() {
    const nameEl = document.getElementById('userName');
    const dateEl = document.getElementById('userJoinDate');

    if (nameEl && dateEl && userData) {
      nameEl.textContent = userData.username;
      dateEl.textContent = new Date(userData.created_at).toLocaleDateString();
    }
}
```

---

USERLIBRARY.JS ----------

```
//script to handle logging out
document.addEventListener('DOMContentLoaded', () => {
    const logoutBtn = document.getElementById('logoutBtn');

    logoutBtn?.removeEventListener('click', handleLogout);
```

```
  logoutBtn?.addEventListener('click', handleLogout, { once: true });
});

function handleLogout() {
  const userConfirmed = confirm('Are you sure you want to log out?');

  if (userConfirmed) {
    console.log("Logging out...");

    fetch('/logout', { method: 'POST' })
      .then(response => response.text())
      .then(message => {
        alert(message);
        window.location.href = '../html/login.html';
      })
      .catch(error => {
        console.error('Error during logout:', error);
        alert('An error occurred during logout. Please try again.');
      });
  }
}



document.addEventListener('DOMContentLoaded', function() {
  const passwordChangeContainer = document.getElementById('passwordChangeContainer');
  const changePasswordBtn = document.getElementById('changePasswordBtn');

  // Function to create the password change form
```

```
function createPasswordChangeForm() {
  const formHTML = `
    <form id="changePasswordForm">
      <label for="new_password">New Password:</label>
      <input type="password" id="new_password" required>
      <br>
      <label for="confirm_password">Confirm Password:</label>
      <input type="password" id="confirm_password" required>
      <br>
      <button type="submit">Change Password</button>
    </form>
  `;
  passwordChangeContainer.innerHTML = formHTML;

  // Add event listener to the form
  const changePasswordForm = document.getElementById('changePasswordForm');
  changePasswordForm.addEventListener('submit', function(event) {
    event.preventDefault();

    const newPassword = document.getElementById('new_password').value;
    const confirmPassword = document.getElementById('confirm_password').value;

    // Password validation regex
    const passwordRegex = /^[a-zA-Z0-9!@#$%^&*()_+]{6,20}$/;

    // Check if passwords match
    if (newPassword !== confirmPassword) {
      alert('Passwords do not match.');
      return;
```

```
        }

        // Check if passwords meet the criteria
        if (!passwordRegex.test(newPassword)) {
            alert('Password must be 6-20 characters long and contain only letters, numbers, and "!@#$%^&*()_"');
            return;
        }

        // Fetch request to change password
        fetch('/change-password', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ password: newPassword }),
        })
        .then(response => response.json())
        .then(data => {
            alert(data.message);
            passwordChangeContainer.innerHTML = ''; // Clear the form on success
        })
        .catch(error => {
            console.error('Error:', error);
        });
    });
}

// Add event listener to the button
```

```
    changePasswordBtn.addEventListener('click', createPasswordChangeForm);
});



document.addEventListener('DOMContentLoaded', function() {
    const deleteAccountBtn = document.getElementById('deleteAccountBtn');
    if (deleteAccountBtn) {
        deleteAccountBtn.addEventListener('click', function() {
            const userConfirmed = confirm('Are you sure you want to delete your account?');

            if (userConfirmed) {
                fetch('/delete-account', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json',
                    }
                })
                .then(response => response.json())
                .then(data => {
                    alert(data.message);
                    if (data.message === 'Account deleted successfully') {
                        fetch('/logout', {
                            method: 'POST',
                            headers: {
                                'Content-Type': 'application/json',
                            }
```

```
            })
            .then(response => response.text())
            .then(data => {
              console.log(data);
              if (data === 'Logout successful') {
                window.location.href = '../html/login.html'; // Redirect to signup page after logout
              } else {
                alert(data); // Display error message
              }
            })
            .catch(error => {
              console.error('Error during logout:', error);
            });
        }
      })
      .catch(error => {
        console.error('Error:', error);
      });
    }
  });
});


//section which fetches and updates user info
var userData = null; //variable to store the fetched user info

document.addEventListener('DOMContentLoaded', async function () {
  await fetchUserInfo();
```

```
  updateUserInfo();
});

async function fetchUserInfo() {
  try {
    const response = await fetch('/api/user/info', {
      method: 'GET',
      credentials: 'include'
    });

    if (!response.ok) {
      throw new Error('Failed to fetch user info');
    }

    userData = await response.json();
    console.log('Fetched Data:', userData);
  } catch (error) {
    console.error('Error fetching user info:', error);
    userData = { name: 'Guest', joinDate: 'N/A' }; // Fallback values
  }
}
function updateUserInfo() {
  const nameEl = document.getElementById('userName');
  const dateEl = document.getElementById('userJoinDate');

  if (nameEl && dateEl && userData) {
    nameEl.textContent = userData.username;
    dateEl.textContent = new Date(userData.created_at).toLocaleDateString();
  }
```

```
}
```

---

## HTML

ABOUT.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Inquisitive | About</title>
        <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/about.css">
  <link rel="stylesheet" href="../css//responsive/responsiveIndex.css">
        <link rel="stylesheet" href="../css/responsive/responsiveAbout.css">
  <link rel="icon" type="image/png" href="../images/fav.png">
</head>
<body>
  <nav>
                <ul>
                        <li><a href="index.html">Home</a></li>
                        <li><a href="explore.html">Explore</a></li>
                        <li><a href="about.html">About</a></li>

                </ul>
                <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">
                <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
        </nav>
```

```
<div class="logoContainer">
  <img src="../images/logo.png">
</div>

    <div class="content-container">
        <div class="subheader">
            <h1>About Inquisitive</h1>
        </div>

        <div class="content">
            <h2>About the Project</h2>
            <p>
                  The Inquisitive Study Site (ISS) exists to provide a safe, free, and accessible learning and study environment to students who are looking to further educate themselves
                on a given topic. The project will be continually worked on for the next 14 to 15 weeks, and wll receive many updates, following a rough gantt chart timeline.
                This project is meant to combat the "paywall products" of the modern age, providing a similar but distinct enough service to satisfy a given user's needs, all within the capabilities of the website of course.
            </p>

            <h2>About the Team</h2>
            <p>
                  Consisting of 4 members, Caleb Ruby, Caleb Rachocki, Caleb Massey, and Ibrahim Alani, the team working on the ISS is dedicated to the performance, appearance, functionality, and friendliness of the project.
                We are continuing to work on the project, and are open to feedback and user criticisms, both requested and unrequested.
```

All 4 members of the team are experienced in different aspects of computing, whether it be hardware or software.

This project serves as a "Magnum Opus" for our college careers, applying as much as we can from our experiences over the last 4 years.

Our desire overall is to put forth our best effort in making the ISS everything we would like it to be.

We have experienced poor resources and services in our academic years; we ultimately desire to make a difference.

<h3>

Weekly Updates Website

</h3>

<li><a href="https://students.pennwest.edu/mas9309/weeklyUpdates-SP/frontend/HTML/index.HTML">Weekly Reports Available Here!</a></li>

<h2>Tutorials</h2>

<h3>Finding and Using a Set</h3>

When a client initially connects to this webpage, they will be presented with a blank page that has no search results. If a client would like to find a study set they'd like to use, they are able to do so by various means. Should the user wish to find a specific study set, they must navigate to either search bar available on their screen and type their desired study set by name. That search query will be sent to the database, which will provide a response depending on what it finds. Through the use of identical searching, the database will provide study sets that only identical matches to the search query, as well as sets that have the searched word/title in their own title; this was seen as an opportunity to provide more sets to users. Users must press the enter key to perform their search. Users will also be able to use the filter feature in their searches, which will narrow down further the study sets that the database can respond with. For example, if a set has a category of "Math," the search query will only ask for study sets of a given

name that are tagged with the “Math” category, producing results that are only in that category. Multiple categories may be selected at a time during a search, as to prevent a type of bottleneck in the user’s experience. Filters can be applied before or after a search, but they themselves do not perform a search.In order to access any of the study resources available, users must first select which terms-definition combos they would like to use. This ensures that the user may freely customize their experience further, allowing them to skip certain term-definition combos they may already know. To select a term-definition combo, the user must navigate their mouse to which term-definition combo they would like, then navigate to the left-most side of that row. They will encounter a checkbox, located on the left side of the term; users will simply click this checkbox to select the term-definition combo. This term-definition combo will be used in any study resource on that page. Should the user wish to un-select, follow the same procedure just described, except ensure that the term-definition combo does not have a checkmark in its associated checkbox. Should the user wish to select or unselect all terms at once, they simply need to navigate to the “Controls” label, located under the study set title. Under this label, there are two buttons; one is to check all terms in the set (Select All), and the other is to uncheck all terms in a set (Select None). Clicking either of these will perform their associated action, enabling all terms or disabling all terms respectively.

```
            </p>
        </div>
        <div class="flashcardTutorial">
            <h3>Using Flashcards</h3>
            <p>
```

This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the flashcards. Once the flashcards have opened, the user will see a box with a single term appear on their screen. Once this box has been clicked, this term will turn into its associated definition, with the box changing its appearance as well to provide a visual indication that card has been “flipped.” Users will be able to navigate through any term-definition combo by utilizing the Next and Back buttons at the top of the flashcards. Once done with the flashcards, the user simply navigates to the Close Flashcards button in the upper right corner of the flashcards, which will return them to their study set.

```
            </p>
        </div>
        <div class="multipleChoiceTutorial">
            <h3>Using Multiple Choice</h3>
            <p>
```

This button is located under the “Learn!” label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the multiple-

choice questions. Once the form for the questions has appeared on the user's screen, they'll be presented with a single term, which can be cycled through similarly to the flashcards. Each term will have four possible answers, where only one will be correct. Users must read and locate the correct answer for the associated term that's been listed above. Once they find the answer they think is correct, the click the radio button to the left of it, and must click the "Check" button at the bottom of the box. This will verify the user's answer, telling them it's either correct, or telling them it's incorrect while providing the correct answer. To move on to the next term, user's simply click the "Next" button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the "Exit" button. Once they're done reviewing this information and close the prompt, they will be returned to the study set page.

```
            </p>
        </div>
        <div class="FITBTutorial">
            <h3>Using Fill-in-the-Blanks</h3>
            <p>
```

This button is located under the "Learn!" label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the fill-in-the-blank questions. Upon opening this resource, users will be presented with a term, and must fill in the definition of the given term in the associated text-field using their keyboard. Once finished with that term, users can use the "Check" button to verify their answer. The website will respond with either the correct answer, or will notify the user that they have entered the correct answer. To move on to the next term, user's simply click the "Next" button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the "Exit" button. Once they're done reviewing this information and close the prompt, they will be returned to the study set page.

```
            </p>
        </div>
        <div class="booleanTutorial">
            <h3>Using True or False</h3>
            <p>
```

This button is located under the "Learn!" label, under the title of the study set. Clicking this button will apply the selected term-definition combos to the true and false questions. Upon opening this resource, users will be presented with a term, a random definition from the study set, and two radio buttons labeled true / false. Each question will only present one term and definition at a time. Users will read the term, then determine if the definition provided by the form is correct; if it is, then the user will select true, otherwise they will select false. Once answered, users will use the "Check" button to verify their answer. The website will respond with either the correct answer, or will notify the user that they have entered the correct answer. To move

on to the next term, user's simply click the "Next" button. This process is repeated until there are no more questions left to answer, at which point users can continue to cycle the same questions, or exit using the "Exit" button. Once they're done reviewing this information and close the prompt, they will be returned to the study set page.

```
                </p>
            </div>
            <div class="quizTutorial">
                <h3>Using a Quiz</h3>
                <p>
                    This button is located under the "Test Yourself" label, under the title
of the study set. Clicking this button will apply the selected term-definition combos to a quiz. A quiz
is compiled of all possible learning resources we have available on the website, excluding the
flashcards. To be clear, a quiz includes multiple-choice questions, fill-in-the-blank questions, and
true or false questions. Should a user want to learn more about these specific types of questions,
refer to the previous underlined sections; they describe what to do with each type of question, as
well as how they operate. Otherwise, once a user is done taking their quiz, they click the submit
button located at the bottom of the quiz form, which will report correct and incorrect answers, as
well as an overall grade. Any questions left blank will be counted towards their score. Once the user
is done reviewing this information, they can close the prompt, and will be returned to the study set
page.
                </p>
            </div>
        </div>

        <footer></footer>

        <script src="../js/myscript.js"></script>
        <script src="../js/searchBar.js"></script>
</body>
</html>
```

---

EXPLORE.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Inquisitive | Explore</title>
  <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/explore.css">
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="stylesheet" href="../css/responsive/responsiveExplore.css">
  <link rel="icon" type="image/png" href="../images/fav.png">
</head>
<body>
  <nav>
                <ul>
                        <li><a href="index.html">Home</a></li>
                        <li><a href="explore.html">Explore</a></li>
                        <li><a href="about.html">About</a></li>

                </ul>
                <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">
                <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
        </nav>

  <div class="logoContainer">
```

```
    <img src="../images/logo.png">
  </div>


        <header>
            <div class="subheader">
                <h1>Explore Study Sets</h1>
            </div>
        </header>



  <!--  Search Filtering and options   -->
    <!-- Category Filters -->
    <div id="popupOverlay" style="display: none;">
      <div id="popupMenu">
        <h3>Select Categories</h3>
        <div id="categories">
          <label><input type="checkbox" class="category-checkbox" value="Math" checked>
Math</label>
          <label><input type="checkbox" class="category-checkbox" value="Natural Science"
checked> Natural Science</label>
          <label><input type="checkbox" class="category-checkbox" value="Tech Science"
checked> Tech Science</label>
          <label><input type="checkbox" class="category-checkbox" value="History" checked>
History</label>
          <label><input type="checkbox" class="category-checkbox" value="Social Sciences"
checked> Social Sciences</label>
          <label><input type="checkbox" class="category-checkbox" value="Language" checked>
Language</label>
          <label><input type="checkbox" class="category-checkbox" value="Test Prep" checked>
Test Prep</label>
          <label><input type="checkbox" class="category-checkbox" value="Other" checked>
Other</label>
```

```
        </div>

        <button id="selectAllBtn">Select All</button>
        <button id="deselectAllBtn">Deselect All</button>
        <button id="applyFiltersBtn">Apply</button>
        <button id="closePopupBtn">Close</button>
      </div>
    </div>

    <div class="search-container">
      <div class="filterContainer">
        <button id="openPopupBtn">Filter Categories</button>
        <p>Search for a set, then press enter! Use the filters to narrow your search!</p>
      </div>
      <input type="text" id="exploreSearchbar" placeholder="Search...">
    </div>



    <div id="searchResult">
    </div>




          <script src="../js/myscript.js"></script>
    <script src="../js/searchBar.js"></script>
    <script src="../js/explore.js"></script>
</body>
```

```
<footer></footer>
</html>
```

---

INDEX.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Inquisitive</title>
  <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="icon" type="image/png" href="../images/fav.png">
</head>
<body>
  <nav>
            <ul>
                  <li><a href="index.html">Home</a></li>
                  <li><a href="explore.html">Explore</a></li>
                  <li><a href="about.html">About</a></li>

            </ul>
            <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">
            <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
```

```
        </nav>

<div class="logoContainer">
  <img src="../images/logo.png">
</div>

<div class="homeBanner">

  <div class="welcomeContainer">
    <h1>Study YOUR way.</h1>
    <p1>Inquisitive puts the power of learning, straight into your hands.<br>All for free.</p1>
    <li><a id="altRegisterLink" href="signup.html">REGISTER HERE</a></li>
  </div>

  <img src="../images/banner3.jpg">
</div>

<div class="homeSetContainer">
  <div id="userSets">
    <h2>Your Library</h2>
    <a class="userLibraryButton" href="userLibrary.html"> Create, View, and Edit Your Sets</a>
  </div>
  <div id="viewedSets">
    <h2>Recently Viewed Sets</h2>
    <ul id="visitHistory">
      <!-- List items will be dynamically added here -->
    </ul>
  </div>
</div>
```

```
    <footer></footer>

    <script src="../js/myscript.js"></script>
    <script src="../js/searchBar.js"></script>
    <script src="../js/index.js"></script>
</body>
</html>
```

---

LOGIN.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, height=device-height initial-scale=1.0">
    <title>Inquisitive | Login</title>
    <link rel="stylesheet" href="../css/index.css">
    <link rel="stylesheet" href="../css/login.css">
    <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
    <link rel="stylesheet" href="../css/responsive/responsiveLogin.css">
    <link rel="icon" type="image/png" href="../images/fav.png">

</head>
<body>
```

```
<div class="outer-container">
  <nav>
    <ul>
      <li><a href="index.html">Home</a></li>
      <li><a href="explore.html">Explore</a></li>
      <li><a href="about.html">About</a></li>

    </ul>
    <input type="text" placeholder="Search..." id="searchbar" onkeypress="clickPress(event)">
    <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
  </nav>
</div>

<div class="content-container">
  <div class="logoContainer">
    <img src="../images/logo.png">
  </div>
  <div class="subheader">
    <h1>Login</h1>
  </div>

  <div id="login">
    <form id="loginForm">
      <label>Username:</label>

      <input type="text" id="username" name="username" required>

      <label>Password:</label>
```

```
        <input type="password" id="password" name="password" required>

        <button type="submit" id="submit">Login</button>
      </form>
      <script src="../js/login.js"></script>
      <br>
      <p>
        Don't have an account? <a href="signup.html">Sign Up</a>
      </p>
    </div>
  </div>

  <footer></footer>

  <script src="../js/myscript.js"></script>
  <script src="../js/searchBar.js"></script>
</body>
</html>
```

---

PROFILE.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  <title>Inquisitive | Profile</title>
  <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="stylesheet" href="../css/profile.css">
  <link rel="stylesheet" href="../css/responsive/responsiveProfile.css">
  <link rel="icon" type="image/png" href="../images/fav.png">

</head>
<body>
  <nav>
            <ul>
                  <li><a href="index.html">Home</a></li>
                  <li><a href="explore.html">Explore</a></li>
                  <li><a href="about.html">About</a></li>

            </ul>
            <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">

      </nav>

  <div class="logoContainer">
    <img src="../images/logo.png">
  </div>

  <header>
    <div class="subheader">
      <h1>Account Details</h1>
    </div>
```

```
    <div class="content">
      <!-- User Info Section -->
      <div id="userInfo">
        <h2>Welcome, <span id="userName">[User Name]</span></h2>
        <p>Member since: <span id="userJoinDate">[Join Date]</span></p>
      </div>

      <!-- Actions Section -->
      <div id="Actions">
        <button id="deleteAccountBtn">Delete Account</button>
        <button id="changePasswordBtn">Change Password</button>
        <button id="logoutBtn">Log Out</button>
      </div>


      <div id="passwordChangeContainer"></div>
      <script src="../js/profile.js"></script>
    </div>
  </header>
  <footer></footer>
  <script src="../js/profile.js"></script>
  <script src="../js/myscript.js"></script>
  <script src="../js/searchBar.js"></script>
</body>
</html>
```

---

SIGNUP.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, height=device-height initial-scale=1.0">
  <title>Inquisitive | Sign-Up</title>
  <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/signup.css">
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="stylesheet" href="../css/responsive/responsiveSignup.css">
  <link rel="icon" type="image/png" href="../images/fav.png">

</head>
<body>

  <div class="outer-container">
    <nav>
      <ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="explore.html">Explore</a></li>
        <li><a href="about.html">About</a></li>

      </ul>
      <input type="text" placeholder="Search..." id="searchbar" onkeypress="clickPress(event)">
      <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
    </nav>
  </div>
```

```
  <div class="content-container">
    <div class="logoContainer">
      <img src="../images/logo.png">
    </div>
    <div class="subheader">
      <h1>Create Account</h1>
    </div>

    <div id="signup">
      <form id="signupForm" method="POST" action="/add-item">
        <label for="signup_username">Username:</label>

        <input type="text" id="signup_username" name="username" required>
        <p id="userReqs">Usernames must be between 6-15 characters long & only use letters and
numbers<p>

        <label for="signup_password">Password:</label>

        <input type="password" id="signup_password" name="password" required>

        <label for="password_verification"><br>Re-Enter Password:</label>

        <input type="password" id="password_verification" name="password" required>
        <p id="passReqs">6-20 characters long & only use letters, numbers, and
"!@#$%^&*()_"</p>

        <button type="submit" id="createAccount" name="submit">Submit</button>
      </form>
```

```
        <br>

        <p>
          Already have an account? <a href="login.html">Login</a>
        </p>
      </div>
    </div>

    <footer></footer>

    <script src="../js/myscript.js"></script>
    <script src="../js/signup.js"></script>
    <script src="../js/searchBar.js"></script>
</body>
</html>
```

---

SIGNSETTEMPLATE.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inquisitive | Study</title>
    <link rel="stylesheet" href="../css/index.css">
        <link rel="stylesheet" href="../css/studySetTemplate.css">
```

```
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="stylesheet" href="../css/responsive/responsiveStudySetTemplate.css">
  <link rel="icon" type="image/png" href="../images/fav.png">


</head>
<body id="body">
  <nav>
                <ul>
                        <li><a href="index.html">Home</a></li>
                        <li><a href="explore.html">Explore</a></li>
                        <li><a href="about.html">About</a></li>


                </ul>
                <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">
                <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
        </nav>


  <div class="logoContainer">
    <img src="../images/logo.png">
  </div>


  <header id="SSheader">
    <div class="subheader" id="studySetName">
      <h1>Study Set Title</h1>
      <div class="details-section">
        <p><strong>Date Created:</strong> March 25, 2025</p>
        <p><strong>Creator:</strong> Pappy</p>
        <p><strong>Category:</strong> Educational</p>
```

```
      </div>
    </div>
    <div class="button-container hidden">
      <button id="copy-Btn" class="header-btn">Copy Set</button>
    </div>


  </header>
  <div class="horizontal-bar"></div>

  <div class="content" id="SScontent">
    <ul class="StudySetOptions">
      <div class="controls">
        <h3>Controls</h3>
        <div class="controlButtons">
          <button id="selectAllButton">Select All</button>
          <button id="selectNoneButton">Select None</button>
          <button id="selectKnownButton">Select Known</button>
          <button id="selectUnknownButton">Select Unknown</button>

        </div>
      </div>
      <div class="learnOptions">
        <h3>Learn!</h3>
        <div class="learnOptionsButtons">
          <li><button onclick="flashCards()" id="flashcardbtn" class="studysetbutton" >Flash
Cards</button></li>
          <li><button onclick="mconly()"  id="mconlybtn" class="studysetbutton">Multiple
Choice</button></li>
```

```
            <li><button onclick="oeonly()" id="oeonlybtn" class="studysetbutton">Fill in the blank</button></li>

            <li><button onclick="tfonly()" id="tfonlybtn" class="studysetbutton">True or False</button></li>

          </div>

        </div>

        <div class="testOptions">

          <h3>Test Yourself!</h3>

          <div class="testOptionsButtons">

            <li><button onclick="generate_quiz()" id="quizbtn" class="studysetbutton"> Quiz </button></li>

          </div>


        </div>


      </ul>

      <div class="StudySet-Container">

        <form id="termForm" class = "hidden">

          <label for="term">Term:</label>

          <input type="text" id="term" name="term" required><br><br>

          <label for="definition">Definition:</label>

          <input type="text" id="definition" name="definition" required><br><br>

          <button type="submit">Add term</button>

        </form>


        <div class="horizontal-bar"></div>


        <table id="myTable">

          <thead>

            <tr>
```

```
                <th>Select</th>
                <th>Term(s)</th>
                <th>Definition(s)</th>
                <th>Actions</th>
                <th>Learned Status</th>
              </tr>
            </thead>
            <tbody>
              <!-- Rows will be dynamically added here -->
            </tbody>
          </table>


      </div>
    </div>
    <footer id="SSfooter"></footer>

    <script src="../js/myscript.js"></script>
    <script src="../js/studySetTemplate.js"></script>
    <script src="../js/searchBar.js"></script>


</body>

</html>
```

---

USERLIBRARY.HTML ----------

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>User Study Sets | Library</title>
  <link rel="stylesheet" href="../css/index.css">
  <link rel="stylesheet" href="../css/userLibrary.css">
  <link rel="stylesheet" href="../css/responsive/responsiveIndex.css">
  <link rel="stylesheet" href="../css/responsive/responsiveUserLibrary.css">
  <link rel="icon" type="image/png" href="../images/fav.png">
</head>
<body>
  <nav>
                <ul>
                        <li><a href="index.html">Home</a></li>
                        <li><a href="explore.html">Explore</a></li>
                        <li><a href="about.html">About</a></li>

                </ul>
                <input type="text" placeholder="Search..." id="searchbar"
onkeypress="clickPress(event)">
                <li id="auth-link"><a href="login.html">Login / Sign-Up</a></li>
        </nav>
  <div class="logoContainer">
    <img src="../images/logo.png">
  </div>
```

```
<div class="wrapper">
  <div class="content-container">
    <!-- Study Sets List -->
    <section id="studySetsList">
      <h2>Your Study Sets</h2>
      <ul id="studySets">
        <!-- List items will be dynamically added here -->
      </ul>
    </section>

    <!-- Create New Study Set -->
    <section id="createStudySet">
      <h2>Create New Study Set</h2>
      <form id="createStudySetForm">
        <label for="setTitle">Set Title:</label>
        <input type="text" id="setTitle" name="setTitle" required><br><br>

        <label for="category">Category:</label>
        <select id="category" name="category" required>
          <option value="">--Select a Category--</option>
          <option value="Math">Math</option>
          <option value="Natural Science">Natural Science</option>
          <option value="Tech Science">Tech Science</option>
          <option value="History">History</option>
          <option value="Social Sciences">Social Sciences</option>
          <option value="Language">Language</option>
          <option value="Test Prep">Test Prep</option>
          <option value="Other">Other</option>
```

```
            </select><br><br>

            <button type="submit">Create Set</button>
          </form>

        </section>
      </div>

      <footer>
        <!-- Footer content -->
      </footer>
    </div>

    <script src="../js/myscript.js"></script>
    <script src="../js/searchBar.js"></script>
    <script src="../js/userLibrary.js"></script>

</html>
```

---

## CSS – STANDARD

ABOUT.CSS ----------

```
@media only screen and (min-width:600px){

        /*************************/
        /* ABOUT WEBPAGE STYLING */
        /*************************/

        .outer-container {
```

```
        display: contents;
}



.subheader{
        margin-top: 2vh;
        margin-bottom: -1vh;
        display: flex;
        width: 50%;
        background: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
        border-style: none;
        font-size: 10px;
}

.subheader h1{
        color: white;
        background: rgb(218, 41, 28);
}

.logoContainer {
        margin-top: 0;
}

.logoContainer img {
        width: 80px;
        height: 80px;
```

```
        margin: 5px;
}

/* ---------- PROJECT DETAILS ---------- */
.content-container {
        margin: auto;
        height: 100%;
        width: 70%;
        align-items: center;
        justify-content: center;
        background-color: rgb(245, 245, 245);
        display: flex;
        flex-direction: column;
        overflow: visible;
}
.content {
        width: 100%;
        align-items: center;
        justify-content: center;
        align-content: center;
        display:flex;
        flex-direction: column;
}
.content p{
        width: 70%;
        text-align: left;
}
.content h2 {
        text-decoration: underline;
```

```
        text-align: center;
        color:rgb(40, 40, 40);
}
.content h3 {
        margin-top: 25px;
        margin-bottom: 5px;
        text-align: center;
        text-decoration: underline;
        color:rgb(40, 40, 40);
        display: flex;
}
.content ul {
        list-style-type: none;
        margin-top: 1vh;
        margin-bottom: 5vh;
}
.content li {
        list-style: none;
        background-color: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
}
.content li a {
        color: white;
        text-decoration: none;
        font-size: 18px;
        padding: 20px;
        display: block;
        transition: 0.3s;
```

```
            border-radius: 10px;
    }
    .content a:hover {
            background-color: rgb(173, 36, 36);
            border-radius: 10px;
    }
    /* ---------- PROJECT DETAILS ---------- */



    /* ---------- PROJECT TUTORIALS ---------- */
    .tutorialContainer {
            display: flex;
flex-direction: column;
            margin: auto;
            margin-top: 10px;
width: 70%;
padding: 30px;
background-color: #fff;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
border-radius: 5px;
            min-height: 700px;
    }
    .tutorialContainer h3{
            text-decoration: underline;
    }
    .tutorialContainer h2{
            text-decoration: underline;
    }
```

```
.generalSetTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
.flashcardTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
.multipleChoiceTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
.FITBTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
.booleanTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
.quizTutorial {
        margin-bottom: 10px;
        padding: 20px;
}
/* ---------- PROJECT TUTORIALS ---------- */


/*edits the footer div*/
footer {
```

```
                margin-top: 5vh;
                background-color: #c2c2c2;
                width: 100%;
                position: relative;
                min-height: 10vh;
                bottom: 0px;
        }

}
----------------------------------------
```

COMPONENT.CSS ----------

```
/* to reuse components like buttons, input, and modals*/
button {
  padding: 10px;
  border-radius: 15px;
  border: none;
  background: #333;
  color: white;
}

button:hover {
  background-color: grey;
}

input {
  padding: 10px;
```

```
  border-radius: 15px;
  border: 1px solid #ccc;
}

----------------------------------------



EXPLORE.CSS ----------------------------
@media only screen and (min-width: 600px) {

  /* ---------------------------------- SUBHEADER STYLING ---------------------------------- */
  .subheader {
    margin: auto;
    margin-top: 50px;
    background-color: rgb(218, 41, 28);
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
    border: none;
    text-align: center; /* Centering text within subheader */
  }

  .subheader h1 {
    color: white;
    padding: 10px;
    font-size: 24px; /* Specified font size for consistency */
    margin: 0; /* Remove default margin to prevent overflow */
  }
  /* ---------------------------------- HEADER STYLING ---------------------------------- */
```

```
/* ---------------------------------- FOOTER STYLING ---------------------------------- */
footer {
   background-color: #c2c2c2;
   text-align: center;
   position:fixed;
   margin: 5% 0 0;
   min-height: 5rem;
   width: 100%;
   bottom: 0;
}

/* ---------------------------------- SEARCH TOOLS STYLING ---------------------------------- */
#exploreSearchbar {
   display: block;
   width: 100%;
   margin: 20px auto;
   margin-top: 3px;
   padding: 10px;
   font-size: 16px;
   border: 1px solid #c2c2c2;
   border-radius: 5px;
   outline: none; /* Remove default outline */
   transition: border-color 0.3s; /* Smooth border transition */
}

#exploreSearchbar:focus {
   border-color: rgb(218, 41, 28); /* Highlight border on focus */
}
```

```
/* ---------------------------------- SEARCH TOOLS STYLING ---------------------------------- */


/* ---------------------------------- FILTER STYLING ---------------------------------- */
#popupOverlay {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.5);
    display: flex;
    justify-content: center;
    align-items: center;
    z-index: 1000;
}

#popupMenu {
    background: white;
    padding: 20px; /* Padding inside the filter menu */
    border-radius: 10px; /* Rounded corners */
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2); /* Soft shadow for depth */
    text-align: left; /* Align text to the left */
}

.filter-header {
    font-size: 24px; /* Larger font for the header */
    font-weight: bold; /* Bold text for emphasis */
```

```
    margin-bottom: 15px; /* Spacing below the header */
    color: rgb(40, 40, 40); /* Dark gray color */
}

#categories {
    display: flex;
    flex-direction: column; /* Organizing categories vertically */
    gap: 10px; /* Spacing between filters */
    margin-bottom: 20px; /* Space below the categories */
}

.filter-item {
    display: flex; /* Flex layout for items */
    align-items: center; /* Center vertically */
    justify-content: space-between; /* Space out label and checkbox */
    padding: 10px; /* Padding for each filter item */
    border-radius: 8px; /* Rounded corners */
    transition: background-color 0.3s; /* Smooth background transition */
}

.filter-item:hover {
    background-color: rgba(218, 41, 28, 0.1); /* Subtle hover effect */
}

.filter-label {
    font-size: 18px; /* Font size for filter labels */
    color: #333; /* Dark grey for readability */
}
```

```
.filter-checkbox {
    width: 20px; /* Checkbox size */
    height: 20px; /* Checkbox size */
    cursor: pointer; /* Pointer on hover */
    margin-left: 10px; /* Spacing between label and checkbox */
}
```

```
.search-container {
    display: flex;
    align-self: center;
    flex-direction: column; /* Stacks elements vertically */
    align-items: center; /* Centers them horizontally */
    width: 80%;
}
```

```
.filterContainer {
    width: 100%;
    display: flex;
}
.filterContainer p {
    align-self: center;
    text-align: center;
    margin-left: 20px;
}
```

```
#openPopupBtn {
  border-color: #c2c2c2;
  color: rgb(40, 40, 40);
  padding: 10px;
  padding-top: 5px;
  padding-bottom: 5px;
  cursor: pointer;
  border-radius: 5px;
  transition: 0.3s;
  text-align: center;
  align-self: flex-start;
  font-size: 16px;
  width: fit-content;
  transition: 0.3s;
}
#openPopupBtn:hover {
  background-color: #003d77;
  border-color: #003d77;
  color: white;
}
/* ---------------------------------- FILTER STYLING ---------------------------------- */


/* ---------------------------------- RESULT STYLING ---------------------------------- */
/* Edits the box that holds the search results */
#searchResult {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
```

```
    grid-auto-flow: row;
    width: 80%;
    margin: 0 auto; /* Center the search results */
    padding: 20px;
    background-color: #fff;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    border-radius: 5px;
}

/* Edits each individual search result item */
.result-item {
    border: 1px solid #e0e0e0;
    border-radius: 10px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
    margin-right: 5px;
    margin-bottom: 15px;
    padding: 15px;
    transition: transform 0.3s ease, box-shadow 0.3s ease, border-inline 0.3s ease;
    text-align: left;

}
.result-item:hover {
    transform: translateY(-5px); /* Lift effect on hover */
    box-shadow: 0 6px 20px rgba(0, 0, 0, 0.2); /* Deeper shadow when hovered */
    border-inline: solid 10px #003d77;
    cursor: pointer;
}

/* Edits header in search results */
```

```
.result-item h3 {
    margin: 0;
    font-size: 20px;
    color: #333;
    text-decoration: underline 2px;
    text-underline-offset: 5px;
    cursor: pointer;
}
.result-item h3:hover {
    cursor: pointer;
}

/* Edits mundane data for each study set result */
.result-item p {
    margin: 10px 0;
    font-size: 16px;
    color: #555;
    cursor: pointer;
}
.result-item p:hover {
    cursor: pointer;
}

/* Edits the subject tag */
.result-item .tag {
    display: inline-block;
    background: rgb(218, 41, 28);
    color: white;
    border-radius: 12px;
```

```
    padding: 5px 10px;
    font-size: 14px;
    margin-right: 5px;
    margin-top: 10px;
  }

  /* ---------------------------------- RESULT STYLING ---------------------------------- */
}
-----------------------------------------
```

INDEX.CSS ----------

```
@media only screen and (min-width:600px){

        /* Adds these attributes to the entire webpage*/
        * {
                margin: 0;
                padding: 0;
                box-sizing: border-box;
        }
        html, body {
                height: 100%;
                width: 100%;
                margin: 0;
        }
```

```
/* Body attributes */
body {
	font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serifd;
	background-color: rgb(245, 245, 245);
	display: flex;
	flex-direction: column;
	height: auto;
}
body {
	cursor: default; /* Standard cursor for the page */
}

/* Apply only to truly empty/non-editable areas */
*:not(input):not(textarea):not([contenteditable="true"]):not(button):not(a) {
	user-select: none; /* Prevent accidental selection */
	cursor: default; /* Stops text cursor from appearing */
}




/* Header attributes */
/* header includes subheader */
header{
	width: 100%;
	margin-top: -3vh;
	margin-bottom: 3vh;
	justify-items: center;
}
```

```
/* Edits subheader */
.subheader{
        display: flex;
        grid-column: auto;
        margin-top: 30px;
        width: 30%;
        align-self: center;
        justify-content: center;
        background: rgb(245, 245, 245);
        border-radius: 10px;
        border-color:#c2c2c2;
        border-style: solid;
        border-width: 2px;
}

/*this is to edit the attributes of the header inside the subheader, this is the text that currently just displays the name of the page*/
.subheader h1{
        color:rgb(40, 40, 40);
        padding: 10px;
}

/* ---------------------------------- NAVBAR STYLING ---------------------------------- */
/* Navbar attributes */
nav {
        background-color: rgb(218, 41, 28);
```

```
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        width: 70%;
        padding: 0px 20px;
        margin-top: 10px;
        position: absolute;
        height: 70px;
        border-radius: 10px;
        display: inline-flex;
        align-items: center;
        align-self: center;
        justify-content: space-between;
}

/* Edits unordered list that stores the links in the navbar*/
nav ul {

        margin-left: 15px;
        list-style-type: none;
        display:flex;
        align-items: center;
}

/* Edits unordered list items*/
nav ul li {
        margin: 0 2px;                  /* Creates spaces on the side of each link*/
}

/* Edits text of the link buttons*/
nav ul li a {
```

```
            color: white;
            text-decoration: none;
            font-size: 18px;
            padding: 20px;
            display: block;
            transition: 0.3s;
        }

        /* Edits navbar links with mouse hovering*/
        nav a:hover {
            background-color: rgb(173, 36, 36);
            height: auto;
            transform: scale(1.1);
            display: block;
        }

        /*This is for the login/signup button*/
        nav #auth-link{
            margin-left: 15px;
            list-style-type: none;
            display:flex;
            align-items: left;
        }

nav #auth-link a {
            color: white;
            text-decoration: none;
            font-size: 18px;
            padding: 20px;
```

```
        display: block;
        margin-right: 40px;
        transition: 0.3s;
    }
    nav #auth-link a:hover {
        background-color: rgb(173, 36, 36);
        height: auto;
        transform: scale(1.1);
        display: block;
    }

    /* Edits the part of the searchbar you type into */
    nav #searchbar{
        background-color: white;
        width: 50%;
        padding: 10px;
        justify-self: left;
        justify-content: left;
        justify-items: left;
        align-self: left;
        text-align: left; /* Centers the text inside */
        border-radius: 15px;
        outline: none;
    }

/* ---------------------------------- NAVBAR STYLING ---------------------------------- */
```

```
/* ---------------------------------- IMAGE STYLING ---------------------------------- */

        .homeBanner img {
                border: 3px solid rgb(218, 41, 28);
                border-left: none;
                border-right: none;
                margin-top: 2vh;
                width: 100%;
                height: 400px;
                object-fit:         cover;
        }

        .logoContainer img {
                width: 80px;
                height: 80px;
                margin: 5px;
        }

/* ---------------------------------- IMAGE STYLING ---------------------------------- */




/* ---------------------------------- CONTENT STYLING ---------------------------------- */

        /*This edits all paragraph attributes if they are in the content class*/
        .content {
                width: 100%;
```

```
        align-items: center;
        justify-content: center;
        align-content: center;
        display:flex;
        flex-direction: column;
}
.content p{
        width: 50%;
        text-align: center;
}
.content a{
        margin:auto;
}
.content h2 {
        margin-top: 2vh;
        width: 50%;
        text-align: center;
        color:rgb(40, 40, 40);
}



/* Edits container for welcomeContainer and banner img */
.homeBanner {
        position: relative;
        text-align: center;
        color: white;
}
```

```
/* Edits text header on the home index page, under the navbar, above the banner */
.welcomeContainer {
	width: 100%;
	height: 100%;
	position:absolute;
	align-content: center;
}
.welcomeContainer h1 {
	text-decoration: none;
	margin-bottom: 15px;
	font-size: 66px;
}
.welcomeContainer p1 {
	font-size: 22px;
}
.welcomeContainer li {
	list-style-type: none;
	margin-top: 50px;
	margin-bottom: 25px;
}
.welcomeContainer li a {
	text-decoration: none;
	font-weight: bold;
	padding: 20px;
	background-color: rgb(218, 41, 28);
	color: white;
	border-radius: 10px;
```

```
        transition: .3s;
}
.welcomeContainer li a:hover {
        background-color: rgb(173, 36, 36);
}



/* this edits the attributes of the content div*/
.homeSetContainer {
        /*margin-top: 10px;*/
        text-align: center;
        display: flex;
        width: 80%;
        align-self: center;
        min-height: 40vh;
        flex: 1;
        justify-content: center; /* Centers items */
overflow: hidden; /* Prevents clipping */
}

/* Edits the User Library Buttons */
.homeSetContainer a {
        display: inline-block;
        border-radius: 10px;
        border-color: #d8d7d7;
        border-width: 2px;
        border-style: solid;
        padding-top: 1rem;
```

```
        padding-bottom: 1rem;
        color: rgb(40, 40, 40);
        text-decoration: none;
        background-color: rgb(245, 245, 245);
        transition: 0.3s;
        margin: 10px;
        text-align: center;
        white-space: nowrap; /* Prevents unwanted text wrapping */
        max-width: 100%; /* Ensures it doesn't exceed the parent */
        overflow: hidden; /* Prevents any text or border from spilling out */
        align-self: center;
}


.homeSetContainer a:hover {
        color: white;
        background-color: rgb(218, 41, 28);
        border-color: rgb(218, 41, 28);
        transition: 0.3s;
}



/*This edits the two divs of userSets and viewedSets inside of the content div*/
#userSets, #viewedSets{
        width: 50%;
        padding: 10px;
        margin: 25px;
        background: rgb(245, 245, 245);
```

```
        color:rgb(40, 40, 40);
        grid-column: auto;
        border-radius: 10px;
        border-color:#d8d7d7;
        border-style: solid;
        border-width: 2px;
        max-height: 300px;
}

/* Edit the headers in the "Your Library" and "Recently Viewed Sets" */
#userSets h2, #viewedSets h2{
        width: 70%;
        padding: 5px;
        margin: auto;
        margin-bottom: 10px;
        color:rgb(40, 40, 40);
        grid-column: auto;
}

/* Edit the links in the "Recently Viewed Sets" */
#viewedSets ul #visitHistory {
        text-decoration: none;
        padding: 10px;
        border-radius: 10px;
        color: white;
        text-decoration: none;
        background-color: rgb(218, 41, 28);
}
```

```
/*edits the visit history list*/
.styled-list-item {
    list-style-type: none;                  /* Removes bullets */
    border-bottom: 1px solid #ccc;          /* Adds a thin line between items */
    padding: 10px 0;                        /* Adds spacing for better readability */
}

/*make visit history a scrollable field */
#viewedSets {
    max-height: 300px;                      /* Define the maximum height for the container */
    overflow-y: auto;                       /* Enable vertical scrolling when content exceeds the height */
    border: 1px solid #ccc;                 /* Optional: Add a border around the container */
    padding: 10px;                          /* Optional: Add padding inside the container */
    background-color: rgb(245, 245, 245);   /* Optional: Background for better visuals */
}

#visitHistory {
    list-style-type: none;          /* Remove bullets */
    margin: 0;                      /* Reset default margins */
    padding: 0;                     /* Reset default padding */
}

#visitHistory li {
    border-bottom: 1px solid #ddd;          /* Add a thin line between list items */
```

```
        padding: 10px 0;                                    /* Add spacing for
better readability */
        transition: 0.3s;
    }

    #visitHistory li:hover {
        border-color: #003d77;                  /* Change border Color for
consistency */
        background-color: #003d77;          /* Change BG color to indicate what's selcted
*/
        color: white;                               /* Change text color for readability */
    }

/* ---------------------------------- CONTENT STYLING ---------------------------------- */


    /* Edits the footer div*/
    footer {
        background-color: #c2c2c2;
        width: 100%;
        position: relative;
        min-height: 10vh;
        bottom: 0px;
    }
```

```
        /*-------class----------*/
        .disabled-link {
                pointer-events: none; /* Prevents clicking */
                user-select: none; /* Prevents text selection */
                text-decoration: none !important; /* Removes hover effects */
                color: inherit !important; /* Keeps default text color */
        }

        .editing-mode {
                cursor: text !important; /* Force I-beam cursor */
                pointer-events: auto; /* Allow interaction */
                user-select: text; /* Enable text selection */
        }



}

----------------------------------------



LOGIN.CSS -------------------------------
@media only screen and (min-width:600px){

        /************************/
        /* LOGIN WEBPAGE STYLING */
```

```
/************************/

.outer-container {
        display: flex;
        flex-direction: column;
}

.content-container {
        margin: auto;
        height: 100%;
        width: 40%;
        align-items: center;
        justify-content: center;
        background-color: #e3e3e3;
        border-left: solid #c2c2c2;
        border-right: solid #c2c2c2;
        display: flex;
        flex-direction: column;
        overflow: auto;
}

.subheader{
        margin-top: auto;
        margin-bottom: 15px;
        display: flex;
        width: 40%;
        background: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
```

```
        border-style: none;
        font-size: 10px;
}

.subheader h1{
        color: white;
        background: rgb(218, 41, 28);
        text-align: center;
}

.logoContainer {
        height: 25%;
        width: 25%;
        margin-top: 120px;
}

.logoContainer img {
        height: 100%;
        width: 100%;
}

/*this edits the login div*/
#login{
        color: white;
        margin-bottom: 15vh;
        width: 50%;
        background-color: #333;
        padding: 20px;
        border-radius: 10px;
```

```
        display: flex;
        flex-direction: column;
        justify-content: center;
        text-align: center;
}

/* Edits the inputs inside of the login div */
#login input{
        padding:2%;
        margin-bottom: 5%;
        margin-top: 2%;
        border-radius: 5px;
}

/*edits the username and password inputs specifically*/
#login #username, #password{
        width: 90%;
}

/*edits links inside of the login div*/
#login a{
        color: white;
        margin: 5px;
}

/*edits the submit button on the login/signup page*/
#login button{
        padding: 10px;
        border-radius: 10px;
```

```
        }

        footer {
                background-color: #e3e3e3;
                border-left: solid #c2c2c2;
                border-right: solid #c2c2c2;
                margin: auto;
                width: 40%;
                align-items: center;
                justify-content: center;
                display: flex;
                flex-direction: column;
                height: 150px;
        }

}
```

---

PROFILE.CSS ----------

```
@media only screen and (min-width:600px) {

   .subheader {
      margin:auto;
      margin-top: 6vh;
      background-color: rgb(218, 41, 28);
      box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
      border: none;
```

```
}
.subheader h1{
   color:white;
   padding: 10px;
}

.content {
   flex: 1;
   display: flex;
   flex-direction: column;
   align-items: center;
   padding: 30px;
}

#Actions {
   display: flex;
   flex-direction: column;
   align-items: center;
   gap: 12px;
   margin-top: 20px;
}

#Actions button {
   padding: 12px 24px;
   border-radius: 8px;
   border: none;
   background: #d32f2f;
   color: white;
   cursor: pointer;
```

```
    font-size: 18px;
    transition: all 0.3s ease-in-out;
}
#Actions button:hover {
    background: #b71c1c;
    transform: scale(1.1);
}

#passwordChangeContainer {
    background: #333;
    color: white;
    padding: 25px;
    border-radius: 12px;
    width: 50%;
    text-align: center;
    margin-top: 25px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);
}

#passwordChangeContainer form {
   display: flex;
    flex-direction: column;
    gap: 12px;
}

#passwordChangeContainer input {
    padding: 12px;
    border-radius: 8px;
    border: 1px solid #bbb;
```

```
    font-size: 16px;
  }

  #passwordChangeContainer input:focus {
    outline: none;
    border-color: #d32f2f;
  }

/* User Info Section */
#userInfo {
  margin-bottom: 20px;
  padding: 20px;
  background-color: #f9f9f9;
  border: 1px solid #ddd;
  border-radius: 8px;
  width: 80%; /* Align width with other boxes */
  box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
  text-align: center; /* Center-align content */
}

#userInfo h2 {
  margin: 0;
  font-size: 1.8em;
  color: #333;
}

#userInfo p {
  margin: 10px 0 0;
  font-size: 1.2em;
```

```
    color: #555;
}

/* Recent Activity Section */
#recentActivity {
    margin-top: 30px;
    padding: 20px;
    background-color: #f9f9f9;
    border: 1px solid #ddd;
    border-radius: 8px;
    width: 80%; /* Align width with other boxes */
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
    text-align: left; /* Left-align content */
}

#recentActivity h2 {
    font-size: 1.8em;
    margin-bottom: 15px;
    color: #333;
}

#activityList {
    list-style-type: none;
    padding: 0;
}

#activityList li {
    padding: 10px 0;
    border-bottom: 1px solid #ddd;
```

```
  font-size: 1.1em;
  color: #555;
}

#activityList li:last-child {
  border-bottom: none;
}

/* General Box Alignment */
.content > div {
  margin-bottom: 20px;
  display: flex;
  flex-direction: column;
  align-items: center;
}
}
```

---

SIGNUP.CSS ----------

```
@media only screen and (min-width:600px){

        /*************************/
        /* SIGNUP WEBPAGE STYLING */
        /*************************/

        .outer-container {
                display: flex;
```

```
        flex-direction: column;
}

.content-container {
        margin: auto;
        height: 100%;
        width: 40%;
        align-items: center;
        justify-content: center;
        background-color: #e3e3e3;
        border-left: solid #c2c2c2;
        border-right: solid #c2c2c2;
        display: flex;
        flex-direction: column;
        overflow: auto;
}

.subheader{
        margin-top: auto;
        margin-bottom: 15px;
        display: flex;
        width: 40%;
        background: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
        border-style: none;
        font-size: 10px;
}
```

```
.subheader h1{
        color: white;
        background: rgb(218, 41, 28);
        text-align: center;
}

.logoContainer {
        height: 25%;
        width: 25%;
        margin-top: 120px;
}

.logoContainer img {
        height: 100%;
        width: 100%;
}

/*this edits the login div*/
#signup{
        color: white;
        margin-bottom: 15vh;
        width: 50%;
        background-color: #333;
        padding: 20px;
        border-radius: 10px;
        display: flex;
        flex-direction: column;
        justify-content: center;
        text-align: center;
```

```
}

/* Edits the inputs inside of the login div */
#signup input{
        padding:2%;
        margin-bottom: 5%;
        margin-top: 2%;
        border-radius: 5px;
}

/* Edits the username and password inputs specifically */
#signup #signup_username, #signup_password, #password_verification{
        width: 90%;
}

/*edits links inside of the login div*/
#signup a{
        color: white;                                   /*sets their text to
white*/
        margin: 5px;
}

/*edits the submit button on the login/signup page*/
#signup button{
        padding: 10px;                                  /*gives the button 10
pixels of padding*/
        border-radius: 10px;                            /*rounds the corners of the
button*/
}
```

```
        #userReqs, #passReqs{
                margin-top: -4%;
                margin-bottom: 10px;
                font-size: 12px;
                color: #808080;
        }

        footer {
                background-color: #e3e3e3;
                border-left: solid #c2c2c2;
                border-right: solid #c2c2c2;
                margin: auto;
                width: 40%;
                align-items: center;
                justify-content: center;
                display: flex;
                flex-direction: column;
                height: 150px;
        }

}
```

---

STUDYSETTEMPLATE.CSS ----------

```
@media only screen and (min-width:600px) {

  /*Start of Template Page CSS*/
```

```
  .hidden {
    display: none;
  }

/* ------------------------ HEADER ------------------------ */

  #SSheader, #SScontent, #SSfooter{
    opacity: 1;
    transition: opacity 2s;
  }
  #SSheader {
    display: flex;
    align-items: stretch; /* Align items to the top */
    justify-content: center; /* Center the header and its contents */
    margin: auto;
    padding: 10px;
    background-color: #f4f4f4;
    gap: 1px;
  }

  .horizontal-bar {
    width: 100%; /* Makes the bar span the width of the header */
    height: 2px; /* Adjust thickness */
    background-color: #33333367; /* Bar color */
    margin-top: 10px; /* Optional spacing */
  }


  .button-container {
```

```
    display: flex;
    flex-direction: column; /* Arrange buttons vertically */
    gap: 1px; /* Space between buttons */
    margin-top: 0; /* Remove any unwanted spacing */
    align-self: flex-start; /* Align the buttons to the top of the header */
    margin-top: 40px;
    height: 100%;
    justify-content: center; /* Align buttons vertically within the container */
    height: auto; /* Automatically stretch to match the height of the subheader */
    align-self: stretch; /
}
.button-container.hidden {
    display: none !important;
}


.header-btn{
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    padding: 15px;
    margin: 1px;
    transition: 0.3s;
    cursor: pointer;
    font-size: 14px;
    flex-grow: 1; /* Make each button grow to fill available space */
    display: flex;
    align-items: center; /* Center the button content vertically */
```

```
    justify-content: center; /* Center the button content horizontally */
}
.header-btn:hover{
    background-color: #003e77;
    border-color: #003d77;
    color: white;
}

/* DO NOT DELETE -- We have a weird bug... */
nav #user-info a {
                color: white;
                text-decoration: none;
                font-size: 18px;
                padding: 20px;
                display: block;
                transition: 0.3s;
}

/* Covers both #studySetName and subheader */
.subheader {
    display: flex;
    flex-direction: column;    /* Ensures elements stack vertically */
    align-items: start;       /* Aligns items to the left */
    width: 40%;

    margin-top: 40px;
}

/* Header details, i.e., date, creator, etc. */
```

```
  .details-section {
    font-size: 14px;
    color: #555;
    margin-top: 5px;
    padding-top: 10px;
    border-top: 1px solid #ddd;
    padding: 10px;
  }

  .footer {
    position: fixed;
    bottom: 0;
    left: 0;
    width: 100%;
    padding: 20px;
    margin: 0;
    text-align: center;
    background-color: #f4f4f4;
    border-top: 1px solid #ddd;
    z-index: 100;
  }

/* ------------------------ HEADER ------------------------ */




/* ------------------------ RESOURCES ------------------------ */
```

```
/* Div containing buttons and headers for study resources */
.StudySetOptions {
   display: flex;
   list-style-type: none;
   border-radius: 10px;
   align-self: center;
   justify-content: center;
   text-align: center;
   width: 70%;
   background: rgb(245, 245, 245);
   color: rgb(40, 40, 40);
   padding: 10px;
   height: 10%;
   gap: 10px;
}

.StudySetOptions button {
   border-color: #c2c2c2;
   border-style: solid;
   border-width: 2px;
   border-radius: 10px;
   padding: 15px;
   margin: 5px;
   transition: all 0.3s ease;
   background: rgb(245, 245, 245); /* Matches the container background */
   box-shadow: 5px 5px 10px rgba(0, 0, 0, 0.1), -5px -5px 10px rgba(255, 255, 255, 0.7);
   color: rgb(40, 40, 40);
   cursor: pointer;
}
```

```
.StudySetOptions button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
    box-shadow: 6px 6px 12px rgba(0, 0, 0, 0.2), -6px -6px 12px rgba(255, 255, 255, 0.6);
    transform: translateY(-2px); /* Adds depth on hover */
}

.StudySetOptions button:active {
    background-color: #002a52;
    border-color: #002a52;
    color: white;
    box-shadow: inset 4px 4px 8px rgba(0, 0, 0, 0.2), inset -4px -4px 8px rgba(255, 255, 255, 0.6);
    transform: translateY(2px); /* Simulates button press */
}
/* Learning Options Button/Header Div */
.learnOptions {
    margin: auto;
}
.learnOptionsButtons {
    display: flex;
}



/* Test Options Button/Header Div */
.testOptions {
    margin: auto;
}
```

```
.testOptionsButtons {
    display: flex;
}


/* Control Options Button/Header Div */
.controls {
    margin: auto;
}
.controlButtons {
    display: flex;
}
/* Edits control buttons specifically */
.controls #selectAllButton, #selectNoneButton {
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    padding: 15px;
    margin: 5px;
    transition: 0.3s;
}
.controls #selectAllButton:hover, #selectNoneButton:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}

/* ------------------------ RESOURCES ------------------------ */
```

```
/* ------------------------ STUDY SET TABLE / FORM ------------------------ */

  .StudySet-Container {
    display: flex;
    flex-direction: column;
    width: 70%;
    gap: 20px;
  }
  #termForm, #myTable {
    text-align: left;
    max-width: 100%;
    min-width: 100%;
    box-sizing: border-box;
    color: rgb(40, 40, 40);
    background-color: rgb(245, 245, 245);
    margin: auto;
    padding: 10px;
  }
  #termForm {
    position: relative;
    margin-top: 20px;
  }
  /* Edits individual entries in a table */
  td {
    padding: 5px;
```

```
    border-top: 1px solid #ddd;
  }

  /* Edits the input bars for set creator/editor */
  #termForm input {
    flex: 1;              /* Input will take up the available space */
    padding: 5px;
    border-radius: 5px;
    width: 100%;
    align-items: center;
    box-sizing: border-box;    /* Include padding and border in the element's total width and height
*/
  }

  /* Edits the "Add Term" button */
  #termForm button {
    border-color: #c2c2c2;
    border-style: solid;
    margin-top: -15px;
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
  }
  #termForm button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
  }
```

```
/* Edits each "Delete" button for a given term/def combo */
.delete-button {
   border-color: rgb(218, 41, 28);
   background-color: rgb(218, 41, 28);
   color: white;
   border-style: solid;
   border-width: 2px;
   border-radius: 10px;
   padding: 10px;
   margin-right: 10px;
   transition: 0.3s;
}
.delete-button:hover {
   border-color: rgb(173, 36, 36);
   background-color: rgb(173, 36, 36);
}

/* Edits each "Edit" button for a given term/def combo */
.edit-button {
   border-color: #c2c2c2;
   border-style: solid;
   border-width: 2px;
   border-radius: 10px;
   padding: 10px;
   transition: 0.3s;
}
.edit-button:hover {
   background-color: #003d77;
```

```
    border-color: #003d77;
    color: white;
}

.save-button {
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
}
.save-button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}

/* Edits the Checkboxes */
.row-checkbox {
    width: 20px;
    height: 20px;
    background: #FFF;
}

/* Learning Status Column Styles */
.learning-status {
    display: inline-block;
    padding: 5px 10px;
```

```
    margin: 5px;            /* Adds a small margin around the box */
    border-radius: 10px;    /* Rounds the corners */
    color: white;
    font-size: 12px;
    text-align: center;
}
.known {
    background-color: green;
    padding: 5px 10px;      /* Adds spacing inside the bubble */
    margin: 5px;            /* Adds a small margin around the text */
    border-radius: 10px;    /* Rounds the corners around the text */
    color: white;           /* Text color */
    display: inline-block;  /* Keeps the styling contained to the text */
    font-size: 12px;
    text-align: center;
}
.unknown {
    background-color: red;
    padding: 5px 10px;      /* Adds spacing inside the bubble */
    margin: 5px;            /* Adds a small margin around the text */
    border-radius: 10px;    /* Rounds the corners around the text */
    color: white;           /* Text color */
    display: inline-block;  /* Keeps the styling contained to the text */
    font-size: 12px;
    text-align: center;
}
.toggle-button {
    border-color: #c2c2c2;
    border-style: solid;
```

```
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
  }



/* ------------------------ STUDY SET TABLE / FORM ------------------------ */



/* ------------------------ QUIZZES ------------------------ */

  #Quiz, #settings {
    width: 60%;
    max-width: 800px;
    min-height: 400px;
    position: absolute;
    margin-top: 90px;
    gap: 15%;
    align-self: center;
    text-align: center;
    background: #ffffff;
    color: #333;
    transition: all 0.3s ease;
    border-radius: 15px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
```

```
    padding: 20px;
    opacity: 1;
}
#Quiz .details-section {
    color: #333;
    font-weight: 500;
    margin-bottom: 20px;
}
#quiz-QA-selector {
    padding: 15px;
    text-align: left;
    background: #f5f5f5;
    border-radius: 10px;
    margin: 10px 0;
}
.quiz-option {
    padding: 12px 15px;
    margin: 8px 0;
    background: #fff;
    border: 1px solid #ddd;
    border-radius: 8px;
    cursor: pointer;
    transition: all 0.2s ease;
}
.quiz-option:hover {
    background: #f0f0f0;
    transform: translateY(-2px);
}
```

```
/* ------------------------ QUIZZES ------------------------ */




/* ------------------------ FLASHCARDS ------------------------ */

  #Flashcards {
    width: 70%;
    max-width: 900px;
    height: 60%;
    background: #ffffff;
    position: absolute;
    left: 0;
    right: 0;
    top: 20%;
    margin-inline: auto;
    color: #333;
    transition: all 0.3s ease;
    text-align: center;
    border-radius: 20px;
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
    perspective: 1000px;
  }
  #flashCardTextHolder {
    width: 100%;
    height: 90%;
    transition: transform 0.3s;
  }
```

```
#flashcardText {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100%;
    font-size: 1.5em;
    padding: 20px;
    color: #333;
}
.clicked {
    background-color: #c2c2c2;
    border-bottom-left-radius: 15px;
    border-bottom-right-radius: 15px;
}
#exitButton {
    position: absolute;
    top: 15px;
    right: 15px;
    padding: 10px;
    border-radius: 15px;
    color: #333;
    background: #f5f5f5;
    border: none;
    width: 150px;
    height: 40px;
    cursor: pointer;
    transition: all 0.2s ease;
}
#exitButton:hover {
```

```
    background: #e0e0e0;
    transform: scale(1.1);
  }
  #nextButton, #backButton {
    padding: 12px 24px;
    margin: 10px;
    background: #003e77;
    color: white;
    border: none;
    border-radius: 25px;
    cursor: pointer;
    transition: all 0.2s ease;
  }
  #nextButton:hover, #backButton:hover {
    background: #002a52;
    transform: translateY(-2px);
  }

/* ------------------------ FLASHCARDS ------------------------ */




/* ------------------------ OTHER BUTTONS ------------------------ */

  #submitbtn{
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
```

```
    border-radius: 10px;
    margin: auto;
    margin-bottom: 20px;
    padding: 10px;
    transition: 0.3s;
  }

/* ------------------------ OTHER BUTTONS ------------------------ */



/* ------------------------ LEARNING TOOLS ------------------------ */

#learningTool {
  width: 70%;
  max-width: 900px;
  height: 60%;
  background: #ffffff;
  position: absolute;
  left: 0;
  right: 0;
  top: 20%;
  margin-inline: auto;
  color: #333;
  transition: all 0.3s ease;
  text-align: center;
  border-radius: 20px;
  box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
  padding: 30px;
```

```
}
#question-container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 70%;
  font-size: 1.3em;
  padding: 20px;
  background: #f8f9fa;
  border-radius: 15px;
  margin-bottom: 20px;
}
#learning-tool-options {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 20%;
  gap: 15px;
}
#submitbutton, #nextbutton, #exitbutton {
  background: #003e77;
  color: white;
  border: none;
  border-radius: 25px;
  margin: 10px;
  padding: 12px 24px;
  cursor: pointer;
  transition: all 0.2s ease;
  font-weight: 500;
```

```
}
#submitbutton:hover, #nextbutton:hover, #exitbutton:hover {
   background: #002a52;
   transform: translateY(-2px);
   box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}
/* ----------------------- LEARNING TOOLS ----------------------- */




}
-----------------------------------------



USERLIBRARY.CSS --------------------------
@media only screen and (min-width:600px){

/* ------------------------- WRAPPER / CONTENT CONTAINER ------------------------- */
   /* Flex container to hold header, content, and footer */
   .wrapper {
      display: flex;
      flex-direction: column;
      min-height: 100vh;
   }

   /* Make the content take available space */
   .content-container {
      flex: 1;
```

```
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 2rem; /* Add some space between sections */
    padding: 2rem; /* Add padding around the entire container */
  }

  /* Flexbox container for the sections */
  .content-container section {
    width: 80%;
    min-height:fit-content; /* Ensures the section has a minimum height */
    border: 1px solid #ccc;
    padding: 1rem;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    display: flex;
    flex-direction: column;
    align-items: flex-start; /* Align items at the top (start) */
    justify-content: flex-start; /* Align items at the top (start) */
  }

/* ------------------------- WRAPPER / CONTENT CONTAINER ------------------------- */




/* ------------------------- CONTROL FEATURES ------------------------- */

  #studySetsList h2 {
```

```
    font-size: 36px;
    text-decoration: underline;
}
/* Style for the study sets list */
#studySetsList ul {
    list-style-type: none;
    padding: 0;
    width: 100%;
}
/* Style for individual study set items */
#studySetsList ul li {
    padding: 0.5rem;
    border-bottom: 3px solid #ccc;
    transition: 0.3s;
    width: 100%;
}
#studySetsList ul li:hover {
    border-bottom: 3px solid rgb(218, 41, 28);
}



/* Style for form elements */
#createStudySetForm label {
    display: block;
    margin-bottom: 0.5rem;
    text-decoration: underline;
}
#createStudySetForm input {
    width:auto;
```

```
    padding: 0.5rem;
    margin-bottom: 1rem;
    border: 1px solid #ccc;
    border-radius: 5px;
}
#createStudySetForm button {
    padding: 0.5rem 1rem;
    background-color: rgb(218, 41, 28);
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: 0.3s;
}
#createStudySetForm button:hover {
    background-color: rgb(173, 36, 36);
}



/* CSS to style the delete buttons */
.deleteBtn {
    align-self: right;
    justify-self: right;
    background-color: rgb(218, 41, 28);
    color: white;
    border: none;
    padding: 10px;
```

```
    cursor: pointer;
    border-radius: 5px;
    transition: 0.3s;
    align-self: right;
}
.deleteBtn:hover {
    background-color: rgb(173, 36, 36);
}

/* CSS to style the delete buttons */
.edit-button {
    align-self: right;
    justify-self: right;
    border-color: #c2c2c2;
    color: rgb(40, 40, 40);

    padding: 10px;
    cursor: pointer;
    border-radius: 5px;
    transition: 0.3s;
    align-self: right;
}
.edit-button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}

/* ------------------------- CONTROL FEATURES ------------------------- */
```

}

---

## CSS – RESPONSIVE

RESPSONSIVEABOUT.CSS ----------

```
/* THIS IS FOR SMALLER VIEWPORTS */

@media only screen and (max-width:600px){

        /***********************************/
        /* RESPONSIVE ABOUT WEBPAGE STYLING */
        /***********************************/

        .subheader{
                margin-top: -80px;
                margin-bottom: -1vh;
                display: flex;
                width: 70%;
                background: rgb(218, 41, 28);
                box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
                border-radius: 5px;
                border-style: none;
                font-size: 10px;
        }

        .content-container {
    margin-top: 10vh;
                height: 100%;
                width: 90%;
                border-radius: 10px;
```

```
                display: flex;
                align-items: center;
                align-self: center;
                flex-direction: column;
                overflow: visible;
        }

        .content {
                width: 90%;
   text-align: left;
                display:flex;
                flex-direction: column;
                align-items: center;
        }

.content p {
   text-align: left;
}

.content h2 {
                text-decoration: underline;
   margin-top: 30px;
                margin-bottom: 10px;
                text-align: center;
                color:rgb(40, 40, 40);
        }

        .content h3{
                text-decoration: underline;
```

```
            margin-top: 30px;
            margin-bottom: 25px;
            text-align: center;
            color:rgb(40, 40, 40);
        }
        .content li {
            list-style-type: none;
        }

.content li a {
            background-color: rgb(218, 41, 28);
            border-radius: 10px;
            padding: 20px;
            color: white;
            text-decoration: none;
            margin-top: 35px;
            transition: 0.3s;
        }
        .content li a:hover {
            background-color: rgb(173, 36, 36);
        }


        /* ---------- PROJECT TUTORIALS ---------- */
        .tutorialContainer {
            display: flex;
   flex-direction: column;
            margin: auto;
            margin-top: 10px;
```

```
width: 90%;
padding: 30px;
background-color: #fff;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
border-radius: 5px;
            min-height: 700px;
    }
    .tutorialContainer h3{
            text-decoration: underline;
    }
    .tutorialContainer h2{
            text-decoration: underline;
    }
    .generalSetTutorial {
            margin-bottom: 10px;
            padding: 10px;
    }
    .flashcardTutorial {
            margin-bottom: 10px;
            padding: 10px;
    }
    .multipleChoiceTutorial {
            margin-bottom: 10px;
            padding: 10px;
    }
    .FITBTutorial {
            margin-bottom: 10px;
            padding: 10px;
    }
```

```
        .booleanTutorial {
                margin-bottom: 10px;
                padding: 10px;
        }
        .quizTutorial {
                margin-bottom: 10px;
                padding: 10px;
        }
        /* ---------- PROJECT TUTORIALS ---------- */

}
-----------------------------------------
```

RESPSONSIVEEXPLORE.CSS ----------

```
/* THIS IS FOR SMALLER VIEWPORTS */
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE EXPLORE WEBPAGE STYLING */
        /*************************************/

  .logoContainer {
    display: none;
  }

  .subheader {
    margin:auto;
```

```
    margin-top: 180px;
                display: flex;
                width: 70%;
                background: rgb(218, 41, 28);
                box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
                border-radius: 5px;
                border-style: none;
                font-size: 10px;
}

#exploreSearchbar {
    display: block;
    width: 80%;
    margin: 20px auto;
    padding: 10px;
    font-size: 16px;
    border: 1px solid #c2c2c2;
    border-radius: 5px;
}

#searchResult {
    width: 80%;
    margin: 20px auto;
    background-color: #fff;
    padding: 20px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    border-radius: 5px;
}
```

```
  footer {
    background-color: #c2c2c2;
    text-align: center;
    padding: 10px 0;
    position: relative;
    width: 100%;
    bottom: -15vh;
  }
/* ---------------------------------- FILTER STYLING ---------------------------------- */

  /* Example css for the category search options */
  #popupOverlay {
    position: fixed;
    top: 0;
    left: 0;
    width: 100vw;
    height: 100vh;
    background-color: rgba(0, 0, 0, 0.5);
    display: flex;
    justify-content: center;
    align-items: center;
    z-index: 1000;
  }

  #popupMenu {
    background: white;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
```

```
    width: 300px;
    text-align: left;
}

#categories {
    display: flex;
    flex-direction: column;
    gap: 10px;
    margin-bottom: 15px;
}



.filterContainer {
    width: 100%;
    display: flex;
    flex-direction: column;
}
.filterContainer p {
    text-align: center;
    margin-top: 15px;
}

#openPopupBtn {
    border-color: #c2c2c2;
    color: rgb(40, 40, 40);
    padding: 10px;
    padding-top: 5px;
    padding-bottom: 5px;
    cursor: pointer;
```

```
    border-radius: 5px;
    transition: 0.3s;
    text-align: center;
    align-self: center;
    font-size: 16px;
    width: fit-content;
    transition: 0.3s;
}
#openPopupBtn:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}



/* Edits each individual search result item */
.result-item {
    border: 1px solid #e0e0e0;
    border-radius: 10px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
    margin-right: 5px;
    margin-bottom: 15px;
    padding: 15px;
    transition: transform 0.3s ease, box-shadow 0.3s ease, border-inline 0.3s ease;
    text-align: left;
}
.result-item:hover {
    transform: translateY(-5px); /* Lift effect on hover */
    box-shadow: 0 6px 20px rgba(0, 0, 0, 0.2); /* Deeper shadow when hovered */
```

```
    border-inline: solid 5px #003d77;
    cursor: pointer;
  }

/* ---------------------------------- FILTER STYLING ---------------------------------- */



}
----------------------------------------



RESPONSIVEINDEX.CSS ----------------------
/* THIS IS FOR SMALLER VIEWPORTS */
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE INDEX WEBPAGE STYLING */
        /*************************************/

        * {
                margin: 0;
                padding: 0;
                box-sizing: border-box;
        }

        /* Body attributes */
```

```
body {
        font-family: Arial, sans-serif;
        background-color: rgb(245, 245, 245);
        display: flex;
        flex-direction: column;
        height: auto;
}

/* Header attributes */
/* header includes subheader */
header{
        width: 100%;
        margin-bottom: 1rem;
        margin-top: -80px;
        justify-items: center;
        justify-self: center;
}

/* Navbar attributes */
nav {
        background-color: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        width: 100%;
        padding: 0px 16px;
        position: relative;
        height: 125px;
        display: inline-grid;
        justify-content: center;
}
```

```
nav #searchbar {
        margin-top: 5px;
        width: 80%;
        justify-self: center;
}

nav #login_signup {
        background-color: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        width: 180px;
        border-radius: 10px;
        justify-self: center;
        position: relative;
        height: 60px;
        margin-top: 180px;
        text-align: center;
        align-content: center;
        display: block;
        text-decoration: none;
        color: white;
        font-size: 18px;
}

nav #user-info {
        background-color: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        width: 180px;
        border-radius: 10px;
```

```
        justify-self: center;
        position: relative;
        height: 60px;
        margin-top: 180px;
        text-align: center;
        align-content: center;
        display: block;
        text-decoration: none;
        color: white;
        font-size: 18px;
}

/* Edits unordered list that stores the links in the navbar*/
nav ul {
        list-style-type: none;
        display:flex;
}

/* Edits unordered list items*/
nav ul li {
        margin: 0 20px;
}

/* Edits text of the link buttons*/
nav ul li a {
        color: white;
        text-decoration: none;
        font-size: 18px;
        padding: 20px;
```

```
        display: block;
        transition: 0.3s;
}

/* Edits navbar links with mouse hovering*/
nav a:hover {
        background-color: rgb(173, 36, 36);
        height: auto;
        transition: 0.3s;
}

/*this is for the div that contains the searchbar*/
#searcharea {
        background-color: white;
        width: 40%;
        padding: 5px;
        border-radius: 10px;
        display: flex;
}

/* Edits the part of the searchbar you type into */
#searchbar{
        background-color: white;
        width: 50%;
        padding: 10px;
        border-radius: 15px;
        outline: none;
}
```

```
/* Edits subheader */
.subheader{
        display:flex;
        margin-top: 80px;
        width: 30%;
        justify-content: center;
        align-content: center;
        background: #333;
}

/*this is to edit the attributes of the header inside the subheader, this is the text that currently just displays the name of the page*/
.subheader h1{
        color: white;
        padding: 20px;
        text-align: center;
}

/*this edits the attributes of the content div*/
.homeSetContainer {
        margin-top: 90px;
        text-align: center;
        display: flex;
        flex-direction: column;
        width: 90%;
        align-items: center;
        align-self: center;
        min-height: 50vh;
        flex: 1;
```

```
}
.homeSetContainer a {
        border-radius: 10px;
        border-color: #d8d7d7;
        border-width: 2px;
        border-style: solid;
        padding: 10px;
        color: rgb(40, 40, 40);
        text-decoration: none;
        background-color: rgb(245, 245, 245);
        transition: 0.3s;
}
.homeSetContainer a:hover {
        color: white;
        background-color: rgb(218, 41, 28);
        border-color: rgb(218, 41, 28);
        transition: 0.3s;
}


/*This edits all paragraph attributes if they are in the content class*/
.content p{
        width: 100%;                                    /*This will make the
paragraphs 100% of the content width*/
        text-align: center;                             /*This will align the
text inside the paragraph to center*/
}

.content a{
```

```
        margin:auto;
}

/*This edits the two divs of userSets and viewedSets inside of the content div*/
#userSets, #viewedSets{
        padding: 10px;
        margin: 25px;
        background: rgb(245, 245, 245);
        color:rgb(40, 40, 40);
        grid-column: auto;
        border-radius: 10px;
        border-color:#d8d7d7;
        border-style: solid;
        border-width: 2px;
        max-height: 300px;
        min-height: 250px;
        min-width: 350px;
}

/* Edit the headers in the "Your Library" and "Recently Viewed Sets" */
#userSets h2, #viewedSets h2{
        width: 70%;
        padding: 5px;
        margin: auto;
        margin-bottom: 10px;
        color:rgb(40, 40, 40);
        grid-column: auto;
}
/* Edit the links in the "Recently Viewed Sets" */
```

```
#viewedSets ul #visitHistory {
        text-decoration: none;
        padding: 10px;
        border-radius: 10px;
        color: white;
        text-decoration: none;
        background-color: rgb(218, 41, 28);
}

/*edits the visit history list*/
.styled-list-item {
        list-style-type: none;                  /* Removes bullets */
        border-bottom: 1px solid #ccc;          /* Adds a thin line between items */
        padding: 10px 0;                        /* Adds spacing for better readability */
}

/*make visit history a scrollable field */
#viewedSets {
        max-height: 300px;                      /* Define the maximum height for the container */
        overflow-y: auto;                       /* Enable vertical scrolling when content exceeds the height */
        border: 1px solid #ccc;                 /* Optional: Add a border around the container */
        padding: 10px;                          /* Optional: Add padding inside the container */
        background-color: rgb(245, 245, 245);   /* Optional: Background for better visuals */
}
```

```
#visitHistory {
    list-style-type: none;          /* Remove bullets */
    margin: 0;                      /* Reset default margins */
    padding: 0;                     /* Reset default padding */
}

#visitHistory li {
    border-bottom: 1px solid #ddd;  /* Add a thin line between list items */
    padding: 10px 0;                /* Add spacing for better readability */
    transition: 0.3s;
}

#visitHistory li:hover {
    border-color: #003d77;          /* Change border Color for consistency */
    background-color: #003d77;      /* Change BG color to indicate what's selcted */
    color: white;                   /* Change text color for readability */
}

.homeBanner {
    display: none;
}

.logoContainer {
    margin-top: 0;
    justify-self: center;
    align-self: center;
}
```

```
.logoContainer img {
	width: 150px;
	height: 150px;
}

#studySetCreation{
	margin: 25px;
	padding: 15px;
	border-radius: 15px;
	color:white;
	background:grey;
}

h4 {
	text-align: center;
	margin-top: 20px;
}

h5 {
	text-align: center;
}

/*edits the footer div*/
footer {
	text-align: center;
	color: white;
	background-color: #c2c2c2;
	width: 100%;
```

```
                position: relative;
                height: 25vh;
                margin-top: 10vh;
        }
}
----------------------------------------
```

RESPONSIVELOGIN.CSS ----------

```
/* THIS IS FOR SMALLER VIEWPORTS */
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE LOGIN WEBPAGE STYLING */
        /*************************************/

  #login_signup, #user-info {
    display: none;
  }

  nav #searchbar {
    margin-bottom: 5vh;
                width: 90%;
                justify-self: center;
        }

        .outer-container {
                display: flex;
```

```
        flex-direction: column;
}

.content-container {
        margin: auto;
        height: 100%;
        width: 90%;
        align-items: center;
        justify-content: center;
        background-color: #e3e3e3;
        border-left: solid #c2c2c2;
        border-right: solid #c2c2c2;
        display: flex;
        flex-direction: column;
        overflow: auto;
}

.subheader{
        margin-top: auto;
        margin-bottom: 15px;
        display: flex;
        width: 70%;
        background: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
        border-style: none;
        font-size: 10px;
}
```

```
.subheader h1{
        color: white;
        text-align: center;
}

/*this edits the login div*/
#login{
        color: white;
        margin-bottom: 15vh;
        width: 80%;
        background-color: #333;
        padding: 20px;
        border-radius: 10px;
        display: flex;
        flex-direction: column;
        justify-content: center;
        text-align: center;
}

/* Edits the inputs inside of the login div */
#login input{
        padding:2%;
        margin-bottom: 5%;
        margin-top: 2%;
        border-radius: 5px;
}

/*edits the username and password inputs specifically*/
#login #username, #password{
```

```
        width: 90%;
    }

    /*edits links inside of the login div*/
    #login a{
        color: white;
        margin: 5px;
    }

    /*edits the submit button on the login/signup page*/
    #login button{
        padding: 10px;
        border-radius: 10px;
    }

    footer {
        background-color: #e3e3e3;
        border-left: solid #c2c2c2;
        border-right: solid #c2c2c2;
        margin: auto;
        width: 90%;
        align-items: center;
        justify-content: center;
        display: flex;
        flex-direction: column;
        height: 150px;
    }

}
```

---

RESPONSIVEPROFILE.CSS ----------

```
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE PROFILE WEBPAGE STYLING */
        /*************************************/

  .subheader {
    margin:auto;
    margin-top: 80px;
    background-color: rgb(218, 41, 28);
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
    border: none;
    padding: 20px;
    width: 80%;
    border-radius: 15px;
  }
  .subheader h1{
    color:white;
    padding: 10px;
  }

  .content {
    flex: 1;
    display: flex;
```

```
    flex-direction: column;
    align-items: center;
    padding: 30px;
}

#Actions {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 12px;
    margin-top: 20px;
}

#Actions button {
    padding: 12px 24px;
    border-radius: 8px;
    border: none;
    background: #d32f2f;
    color: white;
    cursor: pointer;
    font-size: 18px;
    transition: all 0.3s ease-in-out;
}
#Actions button:hover {
    background: #b71c1c;
    transform: scale(1.1);
}

#passwordChangeContainer {
```

```
    background: #333;
    color: white;
    padding: 25px;
    border-radius: 12px;
    width: 90%;
    text-align: center;
    margin-top: 25px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);
}

#passwordChangeContainer form {
  display: flex;
    flex-direction: column;
    gap: 12px;
}

#passwordChangeContainer input {
    padding: 12px;
    border-radius: 8px;
    border: 1px solid #bbb;
    font-size: 16px;
}

#passwordChangeContainer input:focus {
    outline: none;
    border-color: #d32f2f;
}

/* User Info Section */
```

```
#userInfo {
  margin-bottom: 20px;
  padding: 20px;
  background-color: #f9f9f9;
  border: 1px solid #ddd;
  border-radius: 8px;
  width: 80%; /* Align width with other boxes */
  box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
  text-align: center; /* Center-align content */
}

#userInfo h2 {
  margin: 0;
  font-size: 1.8em;
  color: #333;
}

#userInfo p {
  margin: 10px 0 0;
  font-size: 1.2em;
  color: #555;
}

/* Recent Activity Section */
#recentActivity {
  margin-top: 30px;
  padding: 20px;
  background-color: #f9f9f9;
  border: 1px solid #ddd;
```

```
    border-radius: 8px;
    width: 80%; /* Align width with other boxes */
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
    text-align: left; /* Left-align content */
}

#recentActivity h2 {
    font-size: 1.8em;
    margin-bottom: 15px;
    color: #333;
}

#activityList {
    list-style-type: none;
    padding: 0;
}

#activityList li {
    padding: 10px 0;
    border-bottom: 1px solid #ddd;
    font-size: 1.1em;
    color: #555;
}

#activityList li:last-child {
    border-bottom: none;
}

/* General Box Alignment */
```

```
.content > div {
   margin-bottom: 20px;
   display: flex;
   flex-direction: column;
   align-items: center;
}

}
----------------------------------------
```

RESPONSIVESIGNUP.CSS ----------

```
/* THIS IS FOR SMALLER VIEWPORTS */
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE SIGNUP WEBPAGE STYLING */
        /*************************************/

   nav #login_signup, #user-info {
      display: none;
   }

   nav #searchbar {
      margin-bottom: 5vh;
                width: 90%;
                justify-self: center;
        }
```

```
.outer-container {
        display: flex;
        flex-direction: column;
}

.content-container {
        margin: auto;
        height: 100%;
        width: 90%;
        align-items: center;
        justify-content: center;
        background-color: #e3e3e3;
        border-left: solid #c2c2c2;
        border-right: solid #c2c2c2;
        display: flex;
        flex-direction: column;
        overflow: auto;
}

.subheader{
margin-top: auto;
        margin-bottom: 15px;
        display: flex;
        width: 70%;
        background: rgb(218, 41, 28);
        box-shadow: 0 2px 8px rgba(0, 0, 0, 0.5);
        border-radius: 10px;
        border-style: none;
```

```
        font-size: 10px;
}

.subheader h1{
        color: white;
        text-align: center;
}

/*this edits the login div*/
#signup{
        color: white;
        margin-bottom: 15vh;
        width: 80%;
        background-color: #333;
        padding: 20px;
        border-radius: 10px;
        display: flex;
        flex-direction: column;
        justify-content: center;
        text-align: center;
}

/* Edits the inputs inside of the login div */
#signup input{
        padding:2%;
        margin-bottom: 5%;
        margin-top: 2%;
        border-radius: 5px;
}
```

```
	/* Edits the username and password inputs specifically */
	#signup #signup_username, #signup_password, #password_verification{
		width: 90%;
	}

	/*edits links inside of the login div*/
	#signup a{
		color: white;                                   /*sets their text to
white*/
		margin: 5px;
	}

	/*edits the submit button on the login/signup page*/
	#signup button{
		padding: 10px;                                  /*gives the button 10
pixels of padding*/
		border-radius: 10px;                            /*rounds the corners of the
button*/
	}

	#userReqs, #passReqs{
		margin-top: -4%;
		margin-bottom: 10px;
		font-size: 12px;
		color: #808080;
	}

	footer {
		background-color: #e3e3e3;
```

```
                border-left: solid #c2c2c2;
                border-right: solid #c2c2c2;
                margin: auto;
                width: 90%;
                align-items: center;
                justify-content: center;
                display: flex;
                flex-direction: column;
                height: 150px;
        }


}
----------------------------------------


RESPSONSIVESTUDYSETTEMPLATE.CSS ----------
@media only screen and (max-width:600px){

        /*************************************/
        /* RESPONSIVE STUDYSET WEBPAGE STYLING */
        /*************************************/

  /*Start of Template Page CSS*/
  .hidden {
    display: none;
  }
```

```
  .logoContainer {
    display: none;
  }

/* ------------------------ HEADER ------------------------ */

  #SSheader, #SScontent, #SSfooter{
    opacity: 1;
    transition: opacity 2s;
  }
  #SSheader {
    display: flex;
    flex-direction: column;
    align-items: center;
    margin: auto;
    padding: 20px 15px;
    background-color: #f4f4f4;
    width: 80%;
    box-sizing: border-box;
    gap: 15px;
  }
  #SSheader h1 {
    font-size: 24px;
    color: rgb(40, 40, 40);
    border: 1px solid #ddd;
    border-radius: 8px;
    padding: 12px 16px;
    margin: 0;
    text-align: center;
```

```
    width: 100%;
    box-sizing: border-box;
}

.button-container {
    display: flex;
    flex-direction: column;
    gap: 12px;
    width: 100%;
    margin: 10px 0;
    padding: 0;
    box-sizing: border-box;
}
.button-container.hidden {
    display: none !important;
}



.header-btn{
    border-color: #ddd;
    border-style: solid;
    border-width: 1px;
    border-radius: 10px;
    margin: auto;
    margin-top: -20px;
    padding: 15px;
    cursor: pointer;
    font-size: 16px;
    width: 50%;
```

```
    display: flex;
    align-self: center;
    justify-content: center;
    background-color: #f4f4f4;
    transition: 0.3s;
}
.header-btn:hover{
    background-color: #003e77;
    border-color: #003d77;
    color: white;
}

/* DO NOT DELETE -- We have a weird bug... */
nav #user-info a {
                color: white;
                text-decoration: none;
                font-size: 18px;
                padding: 20px;
                display: block;
                transition: 0.3s;
}

/* Covers both #studySetName and subheader */
.subheader {
    align-items: center;
    border: none;
    background-color: rgba(255, 0, 0, 0);
    display: flex;
    flex-direction: column;
```

```
    width: 100%;
    margin-top: 10px;
  }

  /* Header details, i.e., date, creator, etc. */
  .details-section {
    font-size: 14px;
    background-color: #f4f4f4;
    color: #555;
    margin-top: 5px;
    border: 1px solid #ddd;
    border-radius: 5px;
    padding: 15px;
  }

  .footer {
    position: absolute;
    margin-bottom: 50px;
  }

/* ------------------------ HEADER ------------------------ */



/* ------------------------ RESOURCES ------------------------ */

  /* Div containing buttons and headers for study resources */
  .StudySetOptions {
    align-self: center;
```

```
    justify-self: center;
    justify-content: center;
    text-align: center;
    width: 100%;
    background: rgb(245, 245, 245);
    color: rgb(40, 40, 40);
    padding: 10px;
    height: 10%;
    display: grid;
    list-style-type: none;
    border-radius: 10px;
}
.StudySetOptions button {
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 12px;
    padding: 16px;
    margin: 8px;
    transition: 0.3s;
    min-height: 48px;
    font-size: 16px;
    width: calc(100% - 16px);
    background-color: white;
    box-shadow: 0 1px 3px rgba(0,0,0,0.1);
    cursor: pointer;
}
.StudySetOptions button:hover {
    background-color: #003d77;
```

```
    border-color: #003d77;
    color: white;
}

/* Learning Options Button/Header Div */
.learnOptions {
    margin: auto;
}
.learnOptionsButtons {
    display: flex;
}
.learnOptionsButtons button {
    min-height: 100px;
}

/* Learning Options Button/Header Div */
.learnOptions {
    margin: auto;
}
.learnOptionsButtons {
    display: flex;
}


/* Test Options Button/Header Div */
.testOptions {
    margin: auto;
    margin-top: 15px;
}
```

```
.testOptionsButtons {
    display: flex;
    justify-content: center;
}
.testOptionsButtons button {
    text-align: center;
}



/* Control Options Button/Header Div */
.controls {
    margin: auto;
}
.controlButtons {
    display: flex;
}
/* Edits control buttons specifically */
.controls #selectAllButton, #selectNoneButton {
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 12px;
    padding: 16px;
    margin: 8px;
    transition: 0.3s;
    min-height: 48px;
    font-size: 16px;
    width: calc(100% - 16px);
    background-color: white;
```

```
    box-shadow: 0 1px 3px rgba(0,0,0,0.1);
  }
  .controls #selectAllButton:hover, #selectNoneButton:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
  }

/* ------------------------ RESOURCES ------------------------ */




/* ------------------------ STUDY SET TABLE / FORM ------------------------ */

  .StudySet-Container {
    display: flex;
    flex-direction: column;
    width: 100%;
    gap: 25px;
    padding: 0 12px;
    box-sizing: border-box;
    margin-bottom: 35px;
  }
  #termForm, #myTable {
    text-align: center;
    width: 100%;
    box-sizing: border-box;
    color: rgb(40, 40, 40);
```

```
    background-color: rgb(245, 245, 245);
    margin: 0;
    padding: 20px 15px;
    border-radius: 16px;
    box-shadow: 0 2px 8px rgba(0,0,0,0.1);
}
#myTable {
    width: 100%;
    border-collapse: separate;
    border-spacing: 0 15px;
}
tr {
    display: grid;
    grid-template-columns: repeat(1, 1fr);
    padding: 20px;
    border: 1px solid #e0e0e0;
    font-size: 16px;
    line-height: 1.5;
    vertical-align: middle;
    background-color: white;
    border-radius: 8px;
    margin-bottom: 20px;
}
td {
    display: flex;
    flex-direction: row-reverse;
    justify-content: center;
}
.row-checkbox {
```

```
    width: 28px;
    height: 28px;
    margin: 8px;
    cursor: pointer;
}
#termForm {
    margin-top: 20px;
}
#termForm input {
    flex: 1;
    padding: 18px;
    border-radius: 12px;
    width: 100%;
    align-items: center;
    box-sizing: border-box;
    border: 2px solid #ddd;
    font-size: 16px;
    margin-bottom: 15px;
    background-color: white;
}
#termForm button {
    border-color: #c2c2c2;
    border-style: solid;
    margin-top: -15px;
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
}
```

```
#termForm button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}

/* Edits each "Delete" button for a given term/def combo */
.delete-button {
    border-color: rgb(218, 41, 28);
    background-color: rgb(218, 41, 28);
    color: white;
    border-style: solid;
    border-width: 2px;
    border-radius: 12px;
    padding: 12px 16px;
    margin: 8px;
    transition: 0.3s;
    min-height: 44px;
    font-size: 16px;
    width: calc(100% - 16px);
}
.delete-button:hover {
    border-color: rgb(173, 36, 36);
    background-color: rgb(173, 36, 36);
}

/* Edits each "Edit" button for a given term/def combo */
.edit-button {
    border-color: #c2c2c2;
```

```
    border-style: solid;
    border-width: 2px;
    border-radius: 12px;
    padding: 12px 16px;
    transition: 0.3s;
    min-height: 44px;
    font-size: 16px;
    width: calc(100% - 16px);
    margin: 8px;
    background-color: white;
}
.edit-button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
}

.save-button {
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
}
.save-button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
```

```
}

/* Edits the Checkboxes */
.row-checkbox {
  width: 20px;
  height: 20px;
  background: #FFF;
}

/* Learning Status Column Styles */
.learning-status {
  display: inline-block;
  padding: 5px 10px;
  margin: 5px;            /* Adds a small margin around the box */
  border-radius: 10px;    /* Rounds the corners */
  color: white;
  font-size: 12px;
  text-align: center;
}
.known {
  background-color: green;
  min-width: 30%;
  color: white;
  border-radius: 10px;
  padding: 10px;
  transition: 0.3s;
}
.unknown {
  background-color: red;
```

```
    min-width: 30%;
    color: white;
    border-radius: 10px;
    padding: 10px;
    transition: 0.3s;
  }
  .toggle-button {
    border-color: #c2c2c2;
    min-width: 30%;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    padding: 10px;
    margin-right: 15px;
    transition: 0.3s;
  }

/* ------------------------ STUDY SET TABLE / FORM ------------------------ */



/* ------------------------ QUIZZES ------------------------ */

  #Quiz, #settings {
    width: 90%;
    max-width: 800px;
    min-height: 400px;
    position: absolute;
    margin-top: 150px;
```

```
    gap: 15%;
    align-self: center;
    text-align: center;
    background: #ffffff;
    color: #333;
    transition: all 0.3s ease;
    border-radius: 15px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
    padding: 20px;
    opacity: 5;
}
#Quiz .details-section {
    color: #333;
    font-weight: 500;
    margin-bottom: 20px;
}
#quiz-QA-selector {
    padding: 15px;
    text-align: left;
    background: #f5f5f5;
    border-radius: 10px;
    margin: 10px 0;
}
.quiz-option {
    padding: 12px 15px;
    margin: 8px 0;
    background: #fff;
    border: 1px solid #ddd;
    border-radius: 8px;
```

```
    cursor: pointer;
    transition: all 0.2s ease;
  }
  .quiz-option:hover {
    background: #f0f0f0;
    transform: translateY(-2px);
  }

/* ---------- QUIZZES ---------- */

  #Flashcards {
    width: 90%;
    max-width: 900px;
    height: 60%;
    background: #ffffff;
    position: fixed;
    left: 0;
    right: 0;
    top: 20%;
    margin-inline: auto;
    color: #333;
    transition: all 0.3s ease;
    text-align: center;
    border-radius: 20px;
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
    perspective: 1000px;
  }
  #flashCardTextHolder {
    width: 100%;
```

```
    height: 90%;
    transition: transform 0.3s;
}
#flashcardText {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100%;
    font-size: 1.5em;
    padding: 20px;
    color: #333;
}
.clicked {
    background-color: #c2c2c2;
    border-bottom-left-radius: 15px;
    border-bottom-right-radius: 15px;
}
#exitButton {
    position: absolute;
    top: 60px;
    right: 25%;
    padding: 10px;
    border-radius: 15px;
    color: #333;
    background: #f5f5f5;
    border: none;
    width: 50%;
    height: 40px;
    cursor: pointer;
```

```
    transition: all 0.2s ease;
  }
  #exitButton:hover {
    background: #e0e0e0;
    transform: scale(1.1);
  }
  #nextButton, #backButton {
    padding: 12px 24px;
    margin: 10px;
    background: #003e77;
    color: white;
    border: none;
    border-radius: 25px;
    cursor: pointer;
    transition: all 0.2s ease;
  }
  #nextButton:hover, #backButton:hover {
    background: #002a52;
    transform: translateY(-2px);
  }

/* ------------------------ FLASHCARDS ------------------------ */



/* ------------------------ OTHER BUTTONS ------------------------ */

  #submitbtn{
```

```
    border-color: #c2c2c2;
    border-style: solid;
    border-width: 2px;
    border-radius: 10px;
    margin: auto;
    margin-bottom: 20px;
    padding: 10px;
    transition: 0.3s;
  }

/* ------------------------ OTHER BUTTONS ------------------------ */



/* ------------------------ LEARNING TOOLS ------------------------ */

  #learningTool {
    width: 90%;
    max-width: 900px;
    height: 60%;
    background: #ffffff;
    position: fixed;
    left: 0;
    right: 0;
    top: 20%;
    margin-inline: auto;
    color: #333;
    transition: all 0.3s ease;
    text-align: center;
```

```
    border-radius: 20px;
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
    padding: 30px;
}
#question-container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 70%;
    font-size: 1.3em;
    padding: 20px;
    background: #f8f9fa;
    border-radius: 15px;
    margin-bottom: 20px;
}
#learning-tool-options {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 20%;
    gap: 15px;
}
#submitbutton, #nextbutton, #exitbutton {
    background: #003e77;
    color: white;
    border: none;
    border-radius: 25px;
    margin: 10px;
    padding: 12px 24px;
```

```
    cursor: pointer;
    transition: all 0.2s ease;
    font-weight: 500;
  }
  #submitbutton:hover, #nextbutton:hover, #exitbutton:hover {
    background: #002a52;
    transform: translateY(-2px);
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  }

/* ------------------------ LEARNING TOOLS ----------------------- */
}
```

---

RESPSONSIVEUSERLIBRARY.CSS ----------

```
/* THIS IS FOR SMALLER VIEWPORTS */
@media only screen and (max-width:600px){

/* ------------------------- WRAPPER / CONTENT CONTAINER ------------------------- */
  /* Flex container to hold header, content, and footer */
  .wrapper {
    display: flex;
    flex-direction: column;
    min-height: 100vh;
  }
```

```
  /* Make the content take available space */
  .content-container {
    flex: 1;
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: 2rem; /* Add some space between sections */
    padding: 2rem; /* Add padding around the entire container */
  }

  /* Flexbox container for the sections */
  .content-container section {
    width:max-content;
    max-width: 600px;
    min-height:fit-content; /* Ensures the section has a minimum height */
    border: 1px solid #ccc;
    padding: 1rem;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    display: flex;
    flex-direction: column;
    align-items: flex-start; /* Align items at the top (start) */
    justify-content: flex-start; /* Align items at the top (start) */
  }

/* ------------------------- WRAPPER / CONTENT CONTAINER ------------------------- */
```

```
/* ------------------------- CONTROL FEATURES ------------------------- */

  #studySetsList {
    width: 90%;
    align-items: center;
  }
  #studySetsList h2 {
    font-size: 26px;
    text-decoration: underline;
  }
  /* Style for the study sets list */
  #studySetsList ul {
    list-style-type: none;
    padding: 0;
    width: 100%;
  }
  /* Style for individual study set items */
  #studySetsList ul li {
    padding: 0.5rem;
    border-bottom: 3px solid #ccc;
    transition: 0.3s;
  }
  #studySetsList ul li:hover {
    border-bottom: 3px solid rgb(218, 41, 28);
  }
```

```
/* Style for form elements */
#createStudySetForm label {
  display: block;
  margin-bottom: 0.5rem;
}
#createStudySetForm input {
  width:auto;
  padding: 0.5rem;
  margin-bottom: 1rem;
  border: 1px solid #ccc;
  border-radius: 5px;
}
#createStudySetForm button {
  padding: 0.5rem 1rem;
  background-color: rgb(218, 41, 28);
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: 0.3s;
}
#createStudySetForm button:hover {
  background-color: rgb(173, 36, 36);
}
```

```
/* CSS to style the delete buttons */
.deleteBtn {
    align-self: right;
    justify-self: right;
    background-color: rgb(218, 41, 28);
    color: white;
    border: none;
    padding: 10px;
    cursor: pointer;
    border-radius: 5px;
    transition: 0.3s;
}
.deleteBtn:hover {
    background-color: rgb(173, 36, 36);
}



/* CSS to style the delete buttons */
.edit-button {
    align-self: right;
    justify-self: right;
    border-color: #c2c2c2;
    color: rgb(40, 40, 40);

    padding: 10px;
    cursor: pointer;
    border-radius: 5px;
    transition: 0.3s;
    align-self: right;
```

```
  }
  .edit-button:hover {
    background-color: #003d77;
    border-color: #003d77;
    color: white;
  }

/* ------------------------- CONTROL FEATURES ------------------------- */

}
```

---

# APPENDIX: WORKFLOW AUTHENTICATION

I, Caleb Massey, hereby testify that any and all work on all documentation and the ISS project was done so by adhering to the requirements placed throughout the previous semesters by ourselves as well as the professor, Dr. Chen.

**SIGNATURE:** 

**Date:** 4/19/25

I, Caleb Rachocki, hereby testify that any and all work on all documentation and the ISS project was done so by adhering to the requirements placed throughout the previous semesters by ourselves as well as the professor, Dr. Chen.

**SIGNATURE:** 

**Date:** 4/19/25

I, Caleb Ruby, hereby testify that any and all work on all documentation and the ISS project was done so by adhering to the requirements placed throughout the previous semesters by ourselves as well as the professor, Dr. Chen.

**SIGNATURE:** 

**Date:** 4/19/25

I, Ibrahim K. Alani, hereby testify that any and all work on all documentation and the ISS project was done so by adhering to the requirements placed throughout the previous semesters by ourselves as well as the professor, Dr. Chen.

**SIGNATURE:** 

**Date:** 4/19/25