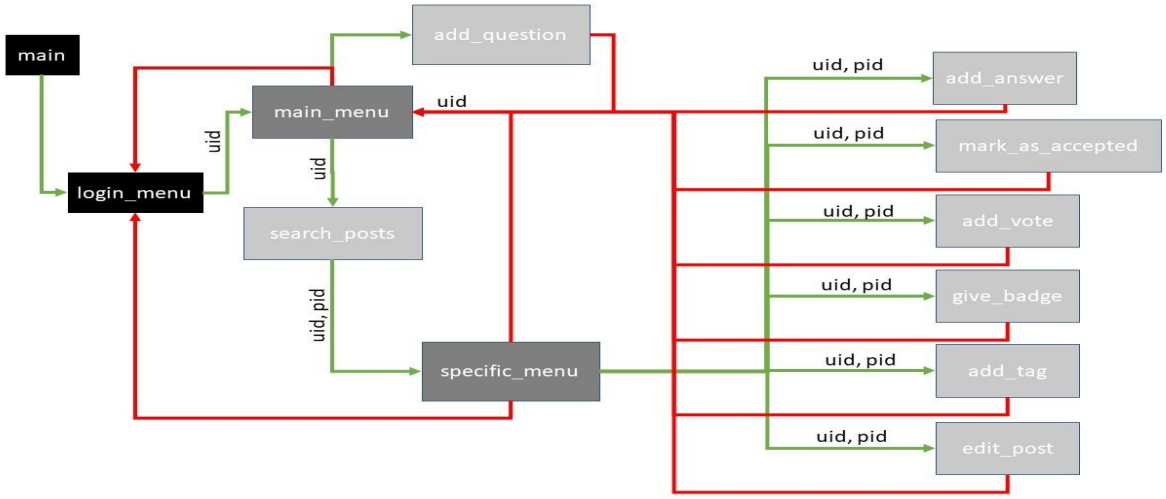
Design Document

By Ahmad Amali, Muhammad Mazhar Hussain, and Jesse Grywacheski

CCIDS: amali, hussain2, jgrywach

Overview & User Guide

This program stores and retrieves data from an inputted database. The user can either register to use the program, or log in with their existing user id and password. The password will be kept hidden for the user's safety. Once the user has logged in they are able to search through the posts stored in the database, vote on posts, or make their own post. A post can either be a question or an answer. If the user is a privileged user they also have the ability to mark answers as the accepted answer, give out badges to other users, add tags to posts, and edit posts. The user can return to the main menu, log out, or exit the program at different times throughout the program.



Control Flow Diagram

Detailed Design

***main**:* The main function welcomes the user and connects to their inputted database, then calls the login menu function.

***login_menu**:* The login menu function allows the user to either login, by typing '1', or register, by typing '2', or exit the program by typing '0'. One the user has registered by providing their user id, name, city, and password, or logged in by providing their user id and password successfully, the user is brought to the main menu.

***main-menu**:* The main menu function gives the user the option to 'post a question', 'search for posts', or 'log out'. If the user chooses 'post a question', the add question function will be called. If the user chooses 'search for posts', the search posts function will be called. If the user chooses 'log out' the user will be returned to the login menu.

add_question: The add question function allows the user to post a question by providing a title and body for the post. Once the question has been added to the database, the user is returned to the main menu.

***search_posts**:* The search posts function allows the user to search for posts by providing one or more keywords and is given a list of up to 5 posts at a time. The user can choose a post to interact with by typing a post id shown in the list. This will bring them to the specific menu. The user can also press enter to show the next page of up to 5 posts or enter 0 to return to the main menu.

***specific_menu**:* The specific menu function gives the user a list of possible menu options, including 'answer a question', 'vote on a post', 'mark accepted answer', 'give a badge to a user', 'add a tag to a post', and 'edit the title or body of a post'. Once the user has chosen an option from one of these two lists, it calls the appropriate function.

add_answer: The add answer function is provided with a user id and a post id when called. The user can answer the question by providing an answer title and body. Once the answer is added to the database, the user is returned to the main menu.

add _vote*:* The add vote function is provided with a user id and a post id when called. The function adds one vote to the post, then returns the user to the main menu.
mark_as_accepted*:* The mark as accepted function is provided with a user id and a post id when called. If the user is not a privileged user and is not allowed to use this function, the user will be returned to the specific menu. If the user is a privileged user, then the function will mark that post as the accepted answer, if it is an answer not a question, and return the user to the main menu.
give_badge*:* The give badge function is provided with a user id and a post id when called. If the user is not a privileged user and is not allowed to use this function, the user will be returned to the specific menu. If the user is a privileged user, then the user can give a badge to the poster of a post by providing a valid badge name. When the badge is given and added to the database, the function returns the user to the main menu.
add_tag: The add tag function is provided with a user id and a post id when called. If the user is not a privileged user and is not allowed to use this function, the user will be returned to the specific menu. If the user is a privileged user, then the user can add a tag to the post by providing a unique tag. When the tag is added to the post and the database, the function returns the user to the main menu.
edit_post: The edit post function is provided with a user id and a post id when called. If the user is not a privileged user and is not allowed to use this function, the user will be returned to the specific menu. If the user is a privileged user, then the user can edit the title and/or body of the post by providing a new title and/or new body When the updated post is added to the database, the function returns the user to the main menu.

Testing Strategy

Our testing strategy included each team member thoroughly testing each of the queries they mainly wrote, this is detailed in the group work breakdown, the tests were then detailed as such: no input, wrong input (uppercase), wrong input (lower case), correct input (upper case), correct input (lowercase) and to see if any reproducible bugs were present, bugs were then fixed periodically upon discovery and the tests were re-ran to ensure that the fix was effective and did not affect any other scenarios that previous passed.

Group Work Breakdown

Task	Partner(s)	Time Spent	Progress Made
Connect to Database	Ahmad	~10 minutes	- skeleton code to connect to a given database name provided as a command line argument
Login/Register Menu	Muhammad	~5 minutes	-skeleton code for login
	Jesse	~10 minutes	-changed login to accept and use uid in program rather than name
	Ahmad	~1.5 hours	-wrote main logic for both functions, periodically fixed and changed by other group members.
Main Menu	Muhammad	~5 minutes	-added return to main menu after each post action
	Jesse	~5 minutes	-added user menu options and user input intake
Specific Menu (Post Actions)	Ahmad	~20 minutes	-skeleton code for login and registration menu

	Jesse	~15 minutes	-added user menu options and user input intake
SQL Injection Prevention	Ahmad	~10 minutes	-added password hiding during login using the getpass module
	Jesse	~10 minutes	-ensured all queries used safe sql injection techniques with variables
Query 1 - Post a Question	Ahmad	~1 hour	-implemented posting a question based on inputs from the user pertaining to question title and body
Query 2 - Search for Posts	Muhammad	~6 hours	-implemented post searching based on user inputted keywords and printing results
	Ahmad	~1 hour	-helped fix bugs pertaining to the printing of the results 5 at a time
Query 3 -Post action-Answer	Muhammad	~1.5 hours	-implemented posting answer to a question functionality
Query 4 - Post action-Vote	Muhammad	~2 hours	-implemented adding a vote to the selected post
Query 5 - Post action-Mark as accepted	Ahmad	~3 hours	-implemented marking answer as accepted and main functionality of the query
	Jesse	~1 hour	-changed function to use the inputted selected pid as aid rather than having user input aid -added a check if user is privileged function
Query 6 - Post action-Give a badge	Ahmad	~1.5 hours	-implemented giving a badge to the user and checking the badge exists.
Query 7 - Post action-Add a tag	Jesse	~1 hour	-implemented function to add the users tag to a post -later implemented the unique tag per post condition -added a check if user is privileged function
Query 8 - Post action-Edit a Post	Jesse	~40 minutes	-implemented function to update the post with the user's new title -added a check if user is privileged function
Testing	Ahmad	~2 hours	-fixed numerous bugs including case-insensitivity and certain sql injection prevention code among general testing on most of the queries
	Muhammad	~2 hours	-search functionality -login/registering functionality -add post and add answer functionality
	Jesse	~2 hours	-log in and registration -add tag and edit post functions -mark as accepted function
Design Document	Jesse	~2 hours	-detailed design of each primary program -general overview and user guide

			-group work breakdown
	Muhammad	~ 30 minutes	-dependencies -group work breakdown
	Ahmad	~ 45 minutes	-testing strategy -group work breakdown -Method of coordination

Method of Coordination:

The main tool for communication between team members was Discord, we carried out a total of 5 voice chat meetings to discuss the project at various stages from start to finish, although the text chat was used all throughout the length of the project for simple communication that required no need for a set time period, as for the development of the mini project, we used github as our version control system in order to collaborate on the project as efficiently as possible, all bugs that were found were mentioned in the chat or during meetings and fixed periodically using tests.

Dependencies

To make our code work we utilized a python module named PrettyTable that allowed us to print our results from the search functionality in a presentable way. PrettyTable may need to be installed prior to running our code and can be done so by running the following command in terminal:

```
pip install -U prettytable
```

Our code can then be run normally using the following command:

```
python3 mainp1.py {database file name}
```