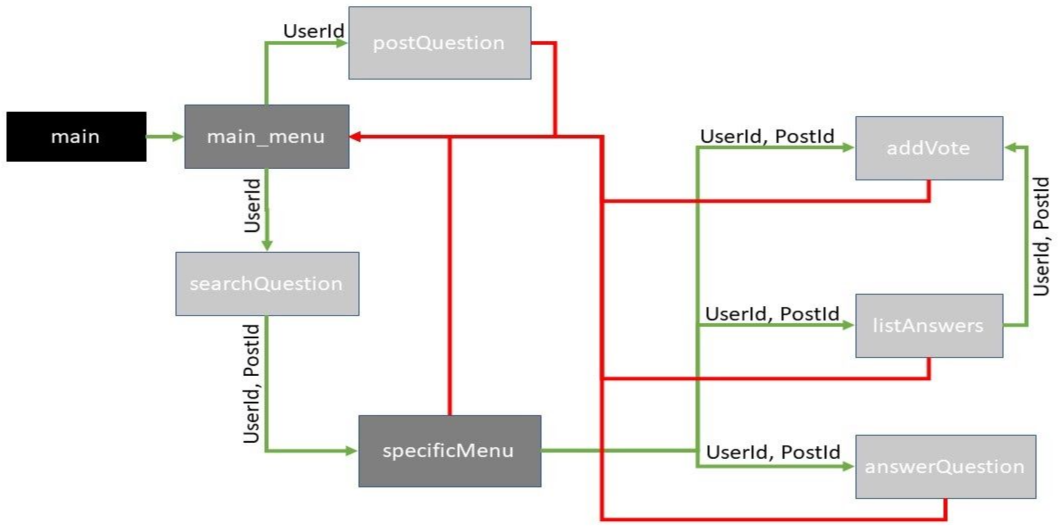
Design Document

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 enter the program by giving a user id or continuing anonymously. Then the user will be able to post a question or search for questions by providing key-words. If the user selects a question from the search results, they are able to add an answer for that question or list all the answers that question has. If the user chooses an answer from the list they can vote on the answer. The user can return to the main menu or exit the program at any time throughout the program.



Control Flow Diagram

Detailed Design

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

mainMenu*:* The main menu function gives the user the option to provide a user id. If they do this, the user report is shown then they are provided with the options ‘post a question’, ‘search for posts’, or ‘exit program’. If the user chooses ‘post a question’, the post question function will be called. If the user chooses ‘search for question’, the search question function will be called. If the user chooses ‘exit program’ the program will terminate.

displayReport*:* The display report function takes the userID as an input and returns various statistics about the user’s activity within the database, results include: the number of questions owned and the average score for those questions, the number of answers owned and the average score for those answers, and the number of votes registered for the user. This function is only applicable for users that provide a userid.

postQuestion: The post 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.

searchQuestion*:* The search posts function allows the user to search for posts by providing one or more keywords and is given a list of all posts that contain at least one key word either in title, body or tag fields. 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.

specificMenu*:* The specific menu function gives the user a list of possible menu options, including ‘post an answer’, ‘list answers’, ‘return to main menu’ or ‘exit program’. Once the user has chosen an option from one of these two lists, it calls the appropriate function, or terminates the program

answerQuestion: The answer question function is provided with a user id and a post id when called. The user can answer the question by providing text for the body. The answer is added to the database and the user is returned to the main menu.

***listAnswers**:* The list answers function is provided with a user id and a question id and lists the post id, text, creation date, and score of all the answers for that question, with the accepted answer displayed first and marked with a star. A user can select an answer by typing its id and is shown all information for that answer. Then they can choose to vote on the answer or return to the main menu.
***addVote**:* The add vote function is provided with a user id and a post id. The function is called when the user chooses a post to vote on. The vote is added if the user has not already voted on this post and the score is incremented on the specified post, then they are returned to the main menu.

Testing Strategy

Our main testing strategy follows closely to the testing strategy in Mini Project 1, where each team member thoroughly tested their own queries 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. Our main goal for this project was to stress test it so it doesn't crash under any circumstances.

Group Work Breakdown

Task	Partner(s)	Time Spent	Progress Made
Phase 1	Jesse	~25 minutes	-check if collections exist in db and drop if they do, create new collections, string operations to extract terms from each post
	Muhammad	~4 hours	-created the collections and the indexing, optimized code so that phase 1 runs faster
	Ahmad	~45 minutes	Reduced the runtime using the bulk insert commands the library ijson with the help of Muhammad
Connect to Database	Ahmad	~10 minutes	-added pymongo connectivity functionality.
Main Menu	Jesse	~10 minutes	-added menu options and function calls
Specific Menu	Jesse	~10 minutes	-added menu options and function calls
	Ahmad	~15 minutes	-edited certain menu options and ensured no loops lead to crashes and infinite loops.
User Report	Jesse	~1 hour	-added queries to find data and print report
Query 1 - Post a Question	Ahmmad	~2 hours	-added post a question query, sanitizing the tags input and inserting as shown in the example database.
Query 2 - Search for Questions	Muhammad	~3 hours	-implemented the search functionality so users can input keywords and get search results. Also implemented post selection and actions
Query 3 - Question action-Answer	Jesse	~20 minutes	-ask user for input, insert a new question to the Posts collection

Query 4 - Question action-List answers	Jesse	~1.5 hours	-list all answers with the question id provided then allow the user to select one and vote or return to main menu
Query 5 - Question/Answer action-Vote	Ahmad	~2.5 hours	-implemented the add vote functionality with all the required constraints.
Testing	Ahmad	~2 hours	-thoroughly tested all the functionalities to match them with what is specified in the project spec.
	Muhammad	~ 2 hours	-thoroughly tested all the functionalities to match them with what is specified in the project spec.
	Jesse	~1 hour	-tested list answers, display report, post answer functions
Design Document	Jesse	~45 minutes	-overview and user guide, control flow diagram, detailed design, group work breakdown
	Muhammad	~ 30 minutes	-worked on group work breakdown, worked on assumptions, and testing strategy
	Ahmad	~45 minutes	-method of coordination, dependencies, testing strategy

Method of Coordination:

The main tool for communication between team members was Discord, we carried out a total of 3 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:

pymongo : the official mongodb library for python, can be installed using the command:

```
pip install pymongo
```

ijson: a json loading library capable of unpacking large json files, can be installed using the command:

```
pip install ijson
```

Prettytable: a table output library intended to make search results more readable, can be installed using the command:

```
pip install prettytable
```

Assumptions:

When entering any keywords and tags, users must input them separated by a comma (,).

Limitations:

Program does not account for invalid user IDs. Users must input valid user ID when running the program.