CMPUT 291 Mini Project 2 Report

a) General overview of your system with a small user guide:

This program is based on a question-answer system. Users are allowed to either login with an ID or be anonymous. After logging in, users are able to search for questions and post a question. When searching a question, they could find all the answers for that question or answer that question. While looking for all the answers that are listed, they could choose to vote for the question or the answer.

b) Detailed design of your software with a focus on the components required to deliver the major functions of your application:

Phase 1:

The json files are opened and their data is stored in a dictionary. For posts.json, the task was longer so we first extracted the data from the file and then processed the data before inserting it into the database for efficiency . To do so, we extracted the title and body from every post individually if available as some posts did not have a title. After that the title and body were converted to a string both combined or individually depending on whether the title was available or not. After that each letter in the whole text was individually checked for punctuation using string.punctuation. Where punctuation wasn't found the letter was added to the 'result' and if it was found it was replaced with a " " in the result. After that we use .split to make a list of all the terms and we individually check each term if its length is >=3 . If it's not then we do not append it to the 'terms' list in lower case which we renew every time the loop runs. We remove the duplicates and finally add the terms to the intended document. After that we insert all the documents inside the database and create an index on the terms. The data in posts.json and votes.json is just extracted and sent to the database.

Phase 2:

Report Function: If the user enters an User ID, it will report 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. Otherwise, the function will not appear.

Post a Question: The users are able to post a question by giving a title, body, and tags. After these are typed, the system will automatically assign a Post ID and creation date. Other fields will also be added by the system. When the program shows: Successfully posted, that means the user has posted the question successfully.

Search for Question: Users are able to search for questions here. Note that the post answers will not appear in the result. Once the user typed the keyword (users could type as many keywords as they want), the program will print all the results that are in Title, Body, and Tags. Users are able to select a question they want to view.

Question Action - Answer: Selecting the question from the function b-2-3), users are able to answer the question. An answer post requires a body text, after the user inputs the body text, the system will automatically insert the body text in the database. The program will inform the user the Post ID.

Question Action - List Answer: After a user passes in the pid to this function, it will first check if the pid is that of a question and if it isn't, will return the user to the main menu once again. Then, it will get a list of all the answer pids which are answers to the question pid passed into the function. It will also limit the body text to a maximum of 80 characters via slicing (if there isn't 80 characters it will print out the entire body). Once the list is printed, the user will be prompted to select a pid from those shown and then the user will be given the option to vote on the question posts(passed in pid) , the selected answer pid or return to the main menu Parameters for Question/Answer Action - Vote (more below) are then adjusted.

Question/Answer Action - Vote: After passing in a post from the earlier function (the user decides whether it is the selected answer from the List Answer function or it is the question post itself from search for a question). After the correct parameters are passed in, it will check if the passed in user has already voted on the selected post. If the person has voted on it, the system will tell the user they already have voted on the post and will redirect the user to the main menu. The above assumes they have already voted or the user is not anonymous. If the user is anonymous, UserID is left blank, otherwise it is filled in with the passed in uid. The function will then call a separate function to randomly generate a voteID and check if that ID is not in use. It will constantly generate a new vote id until the vote ID has not been used and return said ID. Once the VoteID is returned, the variables are added to Votes collection as a new entry and the score field in the Posts collection will be incremented.

c) Testing strategy: (The program is coded by Python3, and application of MongoDB)

Tested both inside and outside lab machines. Initially we tested all the functions using the standard, hardcoded port number. Once we finished testing the individual functions and system functionality, we then changed the standard port number and switched it to taking in an input from the user. We tested all functions twice, once under the anonymous ID and another under a random ID chosen by one of the members. For both anonymous and user IDs, we tested many cases such as but not limited to:

a) False, erroneous data and see if the system handles the errors correctly (inputting a for user ID)
b) Correct Data: Data that is correct and see if the system handles the system as intended

After we used correct data, we then opened another terminal which we used to check if the data is uploaded correctly into the json files.

d) Group work break-down strategy:

Phase 1): Worked on by All Members. Group of 3 portion worked on by Mutaal

Phase-2):

- *Report for User*: Worked on by Leo Chang (1.3 hr)
- *Post a Question*: Worked on by Leo Chang (1.5 hr)
- *Search for questions*: Worked on by Leo Chang (62 hr) and Sanad Masannat
- *Question action-Answer*: Worked on by Leo Chang (2.1 hr)
- *Question action-List answers*: Worked on Sanad Masannat and Leo Chang (5.2 hr)
- *Question/Answer action-Vote*: Worked on by Sanad Masannat and Leo Chang (4.2 hr)

Time Spent by Each Members:

Leo: 76.3 hours for coding and editing. Approximately 45 hours to debug. 121.3 hours in total

Sanad: 27 Hours of work spread over 1.5 weeks.

Mutaal: 18-20 hours