CST 438: Software Engineering, Spring 2024

Assignment 2: Code Review Version: 2024-04-02

Objective.

As software engineers, conducting code reviews is a crucial aspect of our workflow. In this assignment, you will have the opportunity to implement a solution for a problem relevant to our course project and engage in the essential practice of code review with your teammates. Through this hands-on experience, you will refine your coding skills, enhance collaboration within your team, and gain valuable insights into the importance of thorough code evaluation.

Tasks.

The assignment consists of two main tasks:

- Task 1: Creating a new git repository and developing a program to solve a problem.
- Task 2: Creating a pull request to merge your program into a teammate's repository, followed by conducting a code review of another teammate's pull request.

Further details on these tasks will be provided in the following sections.

Task 1.

In this task, you will implement a program to solve a problem inspired by our course project.

Problem definition.

Given the following:

1. User Information: Set of users with their information, including:
   - **username**: the unique id of the user.
     Example values: `goldenlover1, petpal4ever, whiskerwatcher`
   - **display name**: the display name of the user.
     Example values: `John Doe`, `Jane Doe`.
   - **state**: the location of the user by state.
     Example values: `CA, NY, WV.`
   - **friends**: list of usernames whose are friends with the user.
     Example values: `[petpal4ever, goldenlover1]`

2. Post Information: Set of posted created by the users. Each post has the following info:
   - **post ID**: the unique id of the post.
     Example values: `p1924, p9367.`
   - **user ID**: the username of the user who created the post.
     Example values: `petpal4ever, goldenlover1, whiskerwatcher`
   - **visibility**: the visibility of the post, with only two possible values:
     `public`: the post is visible to everyone on the platform.
     `friend`: the post is exclusive viewable by the user's friends.

Note that for simplicity, this assignment does not take into account the contents of the posts.

Write a program with the following functionalities:

- **Check visibility**: Determine if a user can view a post by providing the post ID and username.
- **Retrieve posts**: Fetch all posts accessible to a user, excluding their own posts, by specifying their username.
- **Search users by location**: Obtain all users whose location matches the specified state.

Provided Input Files.

- `user-info.txt`: this file contains user information as explained above. Each line has the form: `<username>;<display_name>;<state>;<friends_list>`
  , where:
  - `<username>`: the user's username.
  - `<display_name>`: the user's display name.
  - `<state>`: the user's location by state.
  - `<friends_list>`: The user's list of friends enclosed in brackets ("`[]`") and separated by commas ("`,`").

Example:

```
goldenlover1;Jane Doe;CA;[petpal4ever,whiskerwatcher]
whiskerwatcher;John Doe;NY;[goldenlover1]
petpal4ever;Great Name;WV;[goldenlover1]
```

- `post-info.txt`: this file contains post information as explained above. Each line has the form: `<post_id>;<user_id>;<visibility>`
  , where:
  - `<post_id>`: the post's ID.
  - `<user_id>`: the post author's username.
  - `<visibility>`: the post's visibility.

Example:

```
post1112;goldenlover1;friend
post2123;whiskerwatcher;friend
post3298;petpal4ever;public
```

Requirements.

- Programming language: You are free to choose any programming language, provided that the code reviewer is familiar with the chosen language. Further guidance on code reviewer assignments will be provided later.
- The program does not require integration of front-end and back-end components. The program should run with standard input and standard output.
- The program should display the menu with the following options:
  1. `Load input data.`
     - The program asks for the file paths of the user and post information files, then loads the data from these files.
  2. `Check visibility.`
     - The program asks for a post ID and a username, then checks if the user can view the post. It outputs "`Access Permitted`" if access is granted and "`Access Denied`" otherwise.
  3. `Retrieve posts.`
     - The program asks for a username and retrieves all posts that the user can view, outputting the post IDs.
  4. `Search users by location:`
     - The program asks for a state location, retrieves users matching the state, and outputs their **display names**.
  5. `Exit:`
     - Select this option to exit the program.

Sample Run. The following is a sample run of the program.
The program displays 5 options.

- Choosing option 1 to load data:
  - Assuming that we use the same files with content as specified in the examples of the "Provided Input Files" section.
- Choosing option 2 to check visibility.
  - Input post ID: `post2123`, input username: `petpal4ever`
  - Output: `Access Denied`
- Choosing option 3 to retrieve posts.
  - Input: `whiskerwatcher`
  - Output: `post1112, post2123, post3298`
- Choosing option 4 to search for users by location.
  - Input: `NY`
  - Output: `John Doe`
- Choosing option 5 to terminate the program.

Task 2.

In this task, you will work with your teammates to perform code review and grading processes.

Coder Review Assignment.

Within your team, designate one code review for each member's program, ensuring that every team member reviews one program. Please make sure that the assigned code reviewer is familiar with the programming language used for the program.

Team Repository.

Create a new team GitHub repository for this assignment. Your team should design several test cases. A test case consists of a user information file, a post information, and an expected output file when testing with the input information files. Include these test cases in the GitHub repo.

Every team member should create a new branch to implement their program for Task 1. Once the code is completed and tested, push your branch to the team GitHub repository. Please ensure that branch names and program file names are unique to avoid duplication.

Next, open a pull request to merge your branch into the main branch. The assigned code reviewer will then conduct a review of the corresponding pull request. Additionally, the code reviewer needs to download the program from the pull request and test it with the provided test cases.

Tasks.

Code reviewer:

- Thoroughly review the assigned pull request. Provide constructive comments, feedback, and suggestions on the code.
- Download the program from the pull request (GitHub provides this option) and run it with the predetermined test cases.

Code author:

- Provide clear instructions to the code reviewer on how to run your program.
- After receiving the review, carefully address any feedback or suggestions provided by the code reviewer.
- Make necessary changes to the code based on the reviewer's comments and evaluate their feedback.

Grading.
In this assignment, team members will also participate in the grading process. Each member will receive two grades, and the final grade is the sum of the two:

- Implementation grade: This grade reflects your implementation of Task 1 and is assigned by the code reviewer, following to the grading instructions provided below.
- Review grade: This grade reflects your review of a pull request and is assigned by the code author of the pull request that you review, following the grading instructions provided below.

Implementation Grading Rubric (Code Reviewer)	
Grading criteria	**Points**
Open a pull request properly. Program compiles and runs successfully.	10
Well-commented code.	10
Program passes all test cases.	10
Well-structured code.	5
Consistent coding style (including naming coventions)	5
Total	**40**

Review Grading Rubric (Code Author)	
Grading criteria	**Points**
The reviewer identify relevant issues and provide constructive feedback.	10
The reviewer's comments and explanations are clear.	10
The reviewer communicate their feedback in a respectful manner.	20
The feedback provided by the reviewer lead to meaningful enhancements or insights for the code author.	20
Total	**60**

Grading Spreadsheet.
Each team need to complete a team grading spreadsheet, which can be downloaded from the assignment G-drive folder as an .xlsx file.

Each team member has a different grading sheet. Please complete the grading following the instructions above to the corresponding sheet.

After all grades in the individual sheets are assigned, complete the main "Team Grades" sheet for the final grades.

Please save the completed document and ensure the file is named in the following format:

`<TEAM_NAME>-as2-team-grading.xlsx`, where `<TEAM_NAME>` is your team's name (*the team's name should have the format* `Team<x>`*, where* `<x>` *is your team number*).

Possible Problems.

- To avoid any potential delays with pull requests, each team should establish a specific deadline for completing Task 1 and submitting pull requests to the main branch.
- The team leader should thoroughly review all grading sheets submitted by team members before final submission, confirming their accuracy and addressing any instances of academic dishonesty if present.

Submission.

Important notes:

- *The policy of dropping the lowest assignment grade does not apply to this assignment.*
- *You may utilize the grace period for this assignment.*

Please submit the completed grading spreadsheet file as described above on Canvas.