CPSC 304 Project Cover Page

Milestone #: 3

Date: 07/28/2023

Group Number: 22

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Selin Uz	55505440	z6s4s	eceselinuz2@gmail.com
Guzide Irmak Bayir	11179330	w1q0i	irmakbayir@gmail.com
Huseyin Kutluay Cakadur	47261367	p6n2b	hk.cakadur@hotmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia.

Completed so far:

1. Hold a meeting to discuss the topic and what the project aims to do.
2. Define the database schema and relationships between different entities (tables).
3. Identify user roles and their permissions within the application.
4. Identify which platforms to use.
5. Identify Functional Dependencies and normalize database relationships accordingly to avoid redundancy.
6. Determine the domains of attributes and write CREATE TABLE statements.
7. Creation of database tables

Features Required on the GUI and backend for Milestone 4

- Query: **INSERT**
  - The user creates a Post and when the Create button is clicked, the Post tuple is inserted.
  - Adding new needed roles by inserting to PostNeedsRole
- Query: **DELETE**
  - The user deletes a Post and when the Delete button is clicked, the Post tuple is deleted
- Query: **UPDATE**
  - Updating the description, title of a **Post**
- Query: **Selection**
  - Filtering functionality for the Feed
    - Selecting **PostNeedsRole**.RoleName = "<selected_roles>"
    - Selecting **Project.Status =** "<selected_statuses>"
- Query: **Projection**
  - Having a "view" selection in the feed where the users can select how much detailed information they want to see about a project. Users can choose to remove project descriptions, needed roles etc. to simplify their view.
- Query: **Join** (need to confirm)
  - Users can input their userID to see the tasks assigned to them. This query would join User and Task.
- Query: **Aggregation with GROUP BY**
  - For displaying the progress of the project, count all tasks of it using the COUNT aggregation_function (then we can count all tasks that have been completed by doing "selection(status="completed") + aggregation with group by (for each type of task)", and convert this into a user-friendly value –like a percentage– that communicates the progress)
- Query: **Aggregation with HAVING**
  - We can have another filter on the feed that allows the user to select a minimum number of tasks left in a project
    - SELECT ProjectID FROM Bug (and also for Testing, Feature, Security)
    - GROUP BY ProjectID HAVING COUNT(ProjectID) >= user_input ;

- Query: **Nested Aggregation with GROUP BY**
    - Maximum Number of Tasks Completed per User in Each Team
- Query: **Division** (need to confirm)
    - Doing **Post** / **User** on UserID to see all posts that a user has posted

User Stories: *(only highlighted in yellow are needed for Milestone 4)*

- **User Registration**
    - As a new user, I want to create an account with my username, password, and skills, so I can access the collaborative platform.
- **User Profile Management**
    - As a user, I want to edit my profile information, including my skills and bio, to showcase my expertise to others.
- **Creating Project Ideas (Post)**
    - As a user, I want to create a project idea post, specifying the project's title, description, and required skills, to attract potential team members.
- **Viewing Project Ideas (Posts on Feed)**
    - As a user, I want to view a list of project idea posts created by other users, so I can find interesting projects to join.
- **Requesting to Join a Project**
    - As a user, I want to send a collaboration request to join a project, so the project owner can review my skills and approve my participation.
- **Approval/Rejection of Collaboration Requests**
    - As a project owner, I want to receive collaboration requests from interested users and have the authority to approve or reject them.
- **Team Formation**
    - As a project owner, I want to accept approved users into my project, forming a team with assigned roles and tasks based on their skills and preferences.
- **Task Management**
    - As a team member, I want to view and manage my assigned tasks, including bug fixes, features, tests, and security tasks, with their respective statuses.
- **Notification System**
    - As a user, I want to receive notifications when someone sends me a collaboration request, and when my request to join a project is approved or rejected.
- **Project Progress Tracking**
    - As a team member, I want to track the progress of the project, including adding new tasks, viewing completed tasks and pending tasks, to stay updated on the project's status.
- **Security and Privacy**
    - As a user, I want assurance that my password and personal information are stored securely and not accessible to unauthorized users.

Project Timeline: *(greens are already done)*

1. **Project/Framework Setup:** (EVERYONE deadline: 07/28)
    a. Setting up the database and connecting to MySQL
    b. Making sure frameworks necessary for connecting to MySQL are set
    c. Configuring Apache server settings
    d. Configuring database settings if necessary

2. **Creation of database tables:** (EVERYONE deadline: 07/29)
    a. Connecting to MySQL and executing CREATE TABLE statements.
    b. Populating tables that map values (that are decomposed after normalization and are between two or less attributes). The tables referred to this by are:
        i. TestingCoverageStatusMap
        ii. SecurityCvssSeverityMap
        iii. NotificationTypeDescMap
        iv. Role
    c. Making sure data validation constraints (unique, not null etc.) are implemented

3. **Creation of classes that will communicate to the database:** (deadline: 08/02)
    a. Create classes that send the following requests to the database: GET, POST, PUT, DELETE --Kutluay
    b. Create classes for entities that require to do so (eg: Post, Team, User etc.). This will be based on which http requests can be performed with which input according to the Entity's schema --Selin
    c. Creation of classes for further backend operations (adding the user to a team in the database –as the project owner– when a tuple in the post is created, modifying the UserHasRole table depending on which RequestToJoin's status changes to "approved" etc.) –classes that communicate the changes in the backend with other classes in the backend– --Irmak

4. **Creation of endpoints** (deadline: 08/03) --Kutluay --Selin--Irmak

5. **Creation of classes that will communicate with the GUI:** (deadline: 08/05)
    a. Creation of classes that will listen to the actions of the user on the GUI (creating a new user, providing text input etc.), and call functions from classes mentioned in step 3 –classes that communicate the changes in the frontend with the backend– --Selin
    b. Creation of classes that will listen to the actions of the user on the GUI (an error message that pops up after an action, clicking a button and a tab opening etc.) and make changes in the GUI accordingly –classes that communicate the changes in the frontend with the frontend– --Kutluay

6. **Implementing GUI features:** (deadline: 08/08) *(only highlighted in yellow are needed for Milestone 4)*
    a. Setting up the homepage (feed where posts will be seen) --Irmak
    b. My projects tab
    c. My request tab
    d. My notifications tab
    e. My tasks tab
        i. Bug task view
        ii. Feature task view
        iii. Security task view
        iv. Testing task view
    f. My teams tab
    g. My user profile tab
    h. Creation of new task --Selin
    i. Creation of new user
    j. Creation of Post --Kutluay
    k. Authentication tab

7. **Checking implementation work against the rubric** (EVERYONE deadline: 08/09)
    a. Making sure the infrastructure has been created in a way that the following operations from the rubric of Milestone 4 can be performed:
        i. INSERT
        ii. DELETE
        iii. UPDATE
        iv. Selection
        v. Projection
        vi. Join
        vii. Aggregation with GROUP BY
        viii. Aggregation with HAVING
        ix. Nested Aggregation with GROUP BY
        x. Division

Description of challenges:

Some things that might go wrong during the steps described above are:

- Unexpected bugs during the testing process taking longer than expected to debug
- The GUI having different views on different platforms (the buttons, text boxes and other UI features not being aligned on different devices)
- The process of figuring out how the backend and the frontend should communicate
- Learning php in a very quick span of time