CPSC 304 Project Cover Page

Milestone #: 3

Date: Oct 29th 2023

Group Number: 28

Name	Student Number	CS Alias (Userid)	Preferred Email Address
Ram Jayakumar	15967981	ramj21	ramjayakumar21@gmail.com
Amin Fahiminia	13006549	afahimi	afahimi@student.ubc.ca
Mercury Mcindoe	85594505	merc0606	mercurymcindoe@gmail.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.

1. **A brief (~2-3 sentences) summary of your project. Many of your TAs are managing multiple projects so this will help them remember details about your project. You can reuse the summary from milestone 2.**

The application is set in the realm of fantasy role-playing games (RPGs), where both players and non-playable characters (NPCs) coexist and interact in a dynamically evolving game universe. Within this realm, characters traverse through diverse landscapes, engage in social exchanges, align themselves with factions, embark on quests, and oversee inventories filled with a wide array of items.

2. Timeline and task breakdown/assignment: The breakdown should be at a level of detail that demonstrates that the group has spent time meaningfully considering what there is left to do.

<u>Week 1: Setup, Planning, and Database Connection [Oct 30 - Nov 5]</u>

Frontend:

- Research and decide on a front-end framework or library
    - *(Ram, Amin, Mercury)*
- Design the initial HTML and CSS structure with Figma.
    - *(Ram)*
- Verify SQL DDL statements for generating every table in the database.
    - *(Amin)*
- Implement a basic frontend for verifying/testing our database output
    - *(Amin, Ram, Mercury)*

Backend:

- Research and decide on a backend framework compatible with Oracle.
    - *(Ram, Amin, Mercury)*
- Initial setup for connecting to the Oracle database.
    - *(Mercury)*
- Implement and test the Oracle database connection to the backend.
    - *(Ram)*
- Add basic routing to serve backend data to the frontend application, presenting a single chosen table from the database.
    - *(Amin)*

Week 2: Basic SQL Operations and Views [Nov 6 - Nov 12]

Frontend:

- Implement views to change the displayed table to a different one within the database
  - *(Amin)*
- Implement a button and input box for inserting new tuples
  - *(Ram)*
- Implement a checkbox option button for deleting specific tuples from the database
  - (*Mercury)*
- Implement an update option for editing specific tuples from the database, saving the change once the user clicks a button
  - (*Ram)*
- Implement a dropdown element to select specific columns from the database
  - *(Amin)*

Backend:

- Implement API endpoints to fetch all tables from the database.
  - *(Amin)*
- Implement API endpoint to insert new data to a specific table and connect to frontend
  - (*Ram)*
- Implement API endpoint to delete multiple tuples from a specific table and connect to frontend
  - (*Mercury)*
- Implement API endpoint to update multiple tuples from a specific table and connect to frontend
  - (*Ram)*
- Implement API endpoint to update specific columns from the database given a condition
  - *(Amin)*

Week 3: Advanced SQL Operations and Views [Nov 13 - Nov 19]

Frontend:

- Implement a projection option for querying specific tuples from the database
  - *(Amin)*
- Implement all join operations to show two tables combined into one (left join, middle join, right join)
  - (*Mercury, Amin*)

- Implement an aggregation with group-by to collect tuples with identical values
  - (*Ram)*
- Implement an aggregation by having to collect tuples with identical values filtered based on some property
  - (*Ram)*

Backend:

- Implement API endpoint to support querying specific tuples from the database
  - (*Amin*)
- Implement API endpoint to allow all join operations on the database.
  - (*Mercury, Amin*)
- Implement API endpoint to support aggregation with “Group By” and “Having”
  - (*Ram)*

Week 4: Final Touches, Documentation, and Presentation [Nov 20 - Nov 26]

Frontend:

- Implement nested aggregation with UI on the frontend of the application
  - *(Ram, Amin, Mercury)*
- Implement division with UI on the frontend of the application
  - *(Mercury)*
- Finalize the styling and user interface after tests.
  - *(Ram, Amin, Mercury)*

Backend:

- Implement API endpoint to allow nested aggregation operations on the database.
  - *(Ram, Amin, Mercury)*
- Implement API endpoint to allow all division operations on the database.
  - *(Mercury)*
- Complete any remaining backend optimizations after testing.
  - *(Ram, Amin, Mercury)*

Both:

- Complete Milestone 4 requirements (pdf document)
- Present demo application for Milestone 5

3. **The deliverables from milestones 1 and 2 have been added to the repository.**

4. **Each group member has made a commit to the repository. Use your UBC-provided account to make the commits. The commits do not have to be**

code-related. For example, one group member can commit the milestone 1 deliverables, and another the milestone 2 deliverables, and the third member the milestone 3 timeline.

5. **In the milestone 3 assignment on Canvas, submit the URL to your group's repository**