**University of British Columbia, Vancouver**
Department of Computer Science

# CPSC 304 Project Cover Page

Milestone #: 3

Date: 2024/03/01

Group Number: 21

| Name | Student Number | CS Alias (Userid) | Preferred Email Address |
|---|---|---|---|
| Alex Lee | 4422902962290296 | al7031 | alexmy31@gmail.com |
| Erica Buchanan | 55077747 | erica4 | eeobuchanan@gmail.com |
| Brooklyn Cheng | 68614932 | bcheng7 | brooklyncheng2002@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 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

**University of British Columbia, Vancouver**
Department of Computer Science

# Project Description

**Brief Description**

The domain of the application is a need-based cookbook for university students. We will be creating a cooking database for university students with recipes that they can query for, given the ingredients they have. These recipes are also able to be reviewed and have their associated author so that the students can have all of the information they need before deciding on their recipe.

# Timeline

**UI Design (React, CSS, JS):**

| Task | Assignee | Deadline | Description |
|---|---|---|---|
| Basic UI | Brooklyn | March 25 | UI can communicate with Java backend and database values are visible to the user. |
| Data Manipulation | Brooklyn | March 30 | Values can be inserted, updated and deleted through the UI and updated values are shown to the user. |
| User Friendliness | Brooklyn | March 30 | UI is intuitive and does not require intricate query construction eg. would not need: attributeName <op> value to do a search. Instead will have dropdowns or other intuitive solutions to create queries. |
| Queries | Brooklyn | April 1 | Queries return the data that is needed and in the right order if required. No further filtering/sorting needed. Queries can be built using a dropdown |

| | | | |
|---|---|---|---|
| | | | menu or buttons, depending on the query. |
| Error Handling | Brooklyn | April 1 | Invalid operations like inserting a duplicate value, or invalid input return an error message visible to the user. |
| User Notification | Brooklyn | April 1 | Completion of queries and of operations will return a result or success message to the user so they can verify their result. |

**Queries (Java Backend):**

| Task | Assignee | Deadline | Description |
|---|---|---|---|
| INSERT operation | Alex | March 30 | Users should be able to insert any valid values into the RequestSaves, Ingredient, Allergy and Publishes tables. When the foreign key value does not exist the operation should be rejected and an appropriate error message returned to the user. |
| UPDATE operation | Alex | March 30 | The user should be able to edit non-primary key attributes in a relation, for relations with at least two non-primary key attribute (eg. User, ReviewMakesHas).<br><br>The relation used for the update operation must have at least two non-primary key attributes. At least one non-primary key attribute must have either a UNIQUE constraint or be |

| | | | |
|---|---|---|---|
| | | | a foreign key that references another table.<br><br>The GUI should present the tuples that are available so that the user can select which tuple they want to update. |
| DELETE operation | Alex | March 30 | Implement a cascade-on-delete situation. The user should be able to choose what values to delete (eg. deleting a Request). |
| SQL script | Erica | March 23 | Create all tables and data in database Should be able to Drop, Recreate, and Reload Tables. |
| Selection queries (WHERE clause) | Erica | March 23 | The user is able to specify the filtering conditions for a given table. That is, the user is able to determine what shows up in the WHERE clause (equalities only).<br><br>The user should be allowed to search for tuples using any number of AND/OR clauses using a dropdown. |
| Projection queries | Erica | March 30 | The user is able to choose any number of attributes to view from any relation in the database. Non-selected attributes must not appear in the result, and the database should be accessed for each query (needs to be fully up to date). All tables should appear in a dropdown for users. |

| | | | |
|---|---|---|---|
| | | | One or more tables must contain at least four attributes (eg. User). |
| Join queries | Erica | March 30 | Users can use a query which joins at least 2 tables and performs a meaningful query through the GUI. |
| Aggregation with group by | Erica | April 1 | Create one query with aggregation (min, max, average, or count are all fine), and a button/dropdown (preset options) to perform the aggregation through the GUI. |
| Aggregation w/ Having | Erica | April 1 | Create one query with the HAVING clause (preset) that the user can execute through the GUI (button). |
| Nested Aggregation w/ Group By | Erica | April 1 | Create one query that finds some aggregated value for each group (e.g., use a nested subquery, will be preset). This must be different from the aggregation query and the user should be able to make the query from the GUI. |
| Division | Alex | April 1 | Create one (preset) division query and provide an interface (button) for the user to execute this query. Query should execute against the most up to date version of the database. |
| Sanitization | Alex | April 1 | Basic Security Practices: Values from the user are not directly used in the database. Assertions are made beforehand to ensure sql statements are not malicious before they're performed. |

| | | | |
|---|---|---|---|
| README Extra Info or Formatting | Alex | April 1 | Update README based on milestone work as needed. |