**Andy Gaspar, Kyler Veenstra**

**Application Proposal**

**CSC 3220 Applications Programming**

**Professor Vickers**

**June 5, 2024**

Table of Contents

[Product Description 3](#_Toc168521705)

[Features 4](#_Toc168521706)

[Data management 4](#_Toc168521707)

[Planned UX/UI 6](#_Toc168521708)

[Target Platforms, Development language, and Framework(s). 7](#_Toc168521709)

[Project Level Effort in Hours 7](#_Toc168521710)

# **Product Description**

A common way to adopt healthy habits is to keep and maintain a calorie count. The ordinary solution to this problem is a pen-and-paper approach; However, this approach often results in repetitively recording common meals. This repetitive action of keeping an overall calories count can discourage people new to meal tracking. Our application’s aim is to log common meals based on mealtimes, track entries, set common meals to be entered, and receive useful data regarding their overall calorie intake.

The proposed application aims to assist with tracking calories by making calorie logging quick, efficient, and requiring a minimum of effort. The application aims to allow the user to catalog their food intake from a stock list of common meals, select either default meals, or user-defined meals. Also, the application should allow users to categorize meals based on mealtimes. This lets the user catalog various individual statistics based on their meal choices. The user should also be able to correct meals they’ve consumed on previous days, as they may have entered incorrect data.

In return for recording this information, the user should receive a valuable archive of usable data from their overall calorie intake. For example, this information could calculate overall calorie counts, monthly calorie counts, daily averages, favorite foods per meal, which mealtime has the highest intake on average, and so on. This information alone should be enough for the user to continue cataloging their calories.

The initial design for the back-end server is required to accommodate several needs. A selection of meals needs to be stored, whether entered by the user, or included as a default. The database also needs to classify, group, and search the list of meals extremely efficiently. The database also is required to classify and store the meals that the user has consumed or recorded consuming.

A few complementary features would expand the possibilities of this application. On a basic note, integrating calorie goals with the application would allow for a user to keep their overall consumption at a minimum for a day, week, or month. Another feature that could be useful is setting “off days,” days that aren’t recorded in the official data set, for whatever reason. Another feature could be to track when the user doesn’t enter a meal, so the application could automatically remove that date from the statistics. Or, if the user wishes, the application could remind them which recent days are missing from the statistical tracker and inform them of the missing data.

# **Features**

* Logging
* CRUD

Create, Read, Update, Delete.

* Hamburger menu
* Quick add
* Line chart showing calorie intake
* Dark mode
* Export data
* Military time or standard time

# **Data management**

This application is intended to be a single user application without the need to connect to an outside database. Therefore, SQLite will be the best option for the database for this application. Below is Our ERD which laid the foundation of the relationships for the SQL Queries.



# **Planned UX/UI**



Above is the home screen of our final wireframe. This wireframe is meant to be high-fidelity as its purpose it to show functionality. Through this wireframe we also created a navigation diagram that is meant to guide stakeholders though the application. One interesting aspect of balsamic (the software we used to create the wireframe) is that It allows for links. Thus, by clicking icons such as the hamburger menu and widgets you are guided to another page.

# **Target Platforms, Development language, and Framework(s).**

Our target platform is android as we believe it would be easier for testing and deployment. We plan to use qt to begin with but then we will move onto react native to see which is better for future projects. Since we are starting with QT the languages that we will use are C++, some JavaScript, and QML.

# **Project Level Effort in Hours**

Since both my teammate and I have little experience with QT we will have to learn some fundamentals about the framework. This will take some time. Therefore, along with implementation and testing, we assume it will take approximately 80 hours for a fully working and deployable MVP.