B.Sc. (Hons) in Software Development

Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Food Logix

By
Emma Garvey

April 22, 2024

Minor Dissertation

Department of Computer Science & Applied Physics,
School of Science & Computing,
Atlantic Technological University (ATU), Galway.

Contents

1 Appendix **2**
- 1.0.1 Frontend Repository . . . 2
- 1.0.2 Backend Repository . . . 2

2 Introduction **3**
- 2.1 Project Context . . . 3
  - 2.1.1 Project Scope . . . 3
- 2.2 Project Objectives . . . 5
  - 2.2.1 Identify Food using Machine Learning . . . 5
  - 2.2.2 Calculate Calories . . . 5
  - 2.2.3 Real Time Meal Tracking . . . 5
  - 2.2.4 Display Nutritional Information . . . 6
- 2.3 Explanation of Dissertation Sections . . . 6
- 2.4 Github links . . . 7
  - 2.4.1 Frontend Repository . . . 7
  - 2.4.2 Python Repository . . . 7

3 Methodology **8**
- 3.1 Agile Approach to development . . . 8
  - 3.1.1 Planning . . . 8
  - 3.1.2 Analysis . . . 8
  - 3.1.3 Design . . . 8
  - 3.1.4 Development . . . 10
  - 3.1.5 Testing . . . 11
  - 3.1.6 Deployment . . . 11
- 3.2 Validation and testing . . . 11
- 3.3 Research Methodology . . . 12
  - 3.3.1 Research Process . . . 12
  - 3.3.2 Finding . . . 12
- 3.4 GitHub . . . 12
- 3.5 Technology Selection criteria . . . 13
  - 3.5.1 Language selection . . . 13
  - 3.5.2 platform selection . . . 13
  - 3.5.3 Model Selection . . . 13
  - 3.5.4 Hosting platform selection . . . 14

4 Technology Review **15**
4.1 Frontend . . . 15
4.1.1 Vue . . . 15
4.1.2 Angular . . . 16
4.1.3 React . . . 16
4.2 Data visualization . . . 18
4.3 Database . . . 18
4.3.1 MongoDB . . . 18
4.3.2 SQL . . . 18
4.4 Backend . . . 19
4.4.1 Backend Framework . . . 19
4.4.2 Language for machine learning . . . 19
4.4.3 Machine learning models . . . 20
4.4.4 Training a model . . . 21
4.5 Model Training . . . 22
4.5.1 Hardware requirements for training a model . . . 22
4.5.2 Training process . . . 22
4.5.3 Problems . . . 23
4.6 Hosting . . . 24
4.6.1 Database . . . 24
4.6.2 Frontend . . . 24
4.6.3 Backend . . . 26
4.6.4 Problems Encountered . . . 27

5 System Design **28**
5.1 Components . . . 28
5.1.1 Client Interface . . . 28
5.1.2 Authentication . . . 28
5.1.3 Registration . . . 28
5.1.4 Login . . . 30
5.1.5 Web Server . . . 31
5.1.6 Backend Service . . . 32
5.1.7 Database Layer . . . 32
5.2 Sequence Diagram . . . 33
5.3 Coupling . . . 33

6 System Evaluation **36**
6.1 Robustness . . . 36
6.1.1 Manual Testing . . . 36
6.1.2 Automated Testing . . . 37
6.2 Performance . . . 37

6.2.1 Response Time 37
6.2.2 Throughput 37
6.3 Achievement of Objectives 37
6.4 Project Milestones 38
6.5 Future Work 39
6.6 Limitations 40

7 Conclusion **43**
7.1 Context 43
7.2 Objectives 43
7.3 Findings from System evaluation 44
7.4 Opportunities for Future Investigation 44
7.5 Impact on Health and Well-being 45
7.6 Data Privacy and Security 45
7.7 Conclusion and Future Outlook 45

List of Figures

3.1 Kanban board 9
3.2 Kanban board Done Issues 9
3.3 Project Roadmap 10

4.1 Model training in Github Repository 23
4.2 Database hosted on AWS 25
4.3 Railway 26

5.1 Architecture Diagram 29
5.2 Sign Up page 29
5.3 Login Page 30
5.4 Invalid password validation in login 31
5.5 User must enter both a login and email validation 31
5.6 Email Validation Login 31
5.7 Login Sequence Diagram 34
5.8 Sign Up Sequence Diagram 34
5.9 Capture Picture Sequence Diagram 35

List of Tables

4.1 Frontend framework comparison 17

Chapter 1

Appendix

1.0.1 Frontend Repository

`https://github.com/E-E-Inc/FYP_Frontend`

1.0.2 Backend Repository

`https://github.com/E-E-Inc/FYP_Python`

Chapter 2

Introduction

2.1 Project Context

For the final year project, an app oriented around food tracking was developed. It addressed the problem of finding the calories in food by just capturing a picture of it. In terms of rationale, there were many interesting reasons behind the choice of this project idea. In terms of user experience, it supplied a simple and unique approach to food tracking while utilising machine learning. Existing food-tracking apps rely on manual data entry or require the user to scan the food's bar code. The Fitbit app is an example of a food app. However, an apple or a carrot may not have a bar code or a label. This could often be tedious and inaccurate, as users had to choose from a list that may not necessarily contain their specific item. This project aimed to fix that. The aim of the project was to address this problem while implementing machine learning and image recognition. The idea was for the user to be able to simply take a picture of the food they had and be able to get the calorie content of the food and store the calories. The motivation behind developing this project was to gain knowledge and expertise in the field of machine learning. Throughout the project, valuable experience was gained in the area. The overall goal was to provide a user-friendly solution to calorie tracking. Machine learning is such an interesting area as it is constantly growing, and new things are being discovered every day. It was exciting that this project used and experimented with things such as machine learning models and datasets.

2.1.1 Project Scope

The project's scope defined the boundaries of what was intended to be achieved. The goal was to create a practical and user-friendly calorie-tracking solution that utilized machine learning. In the market, there wasn't such an app like the one being proposed; therefore, there was a gap to be filled. In the app, the user is asked

to log in or sign up. Once passed this step, A user takes a picture of the food item they have. For example, a banana or a packet of crisps. One of the key features of the app is to use machine learning libraries to identify food in a picture captured by the app. OpenCV and TensorFlow are machine-learning libraries that were used for image recognition. After the user uploads the picture, the application analyzes it. It then calculates calories with possible user input for portion size after the picture is taken. Users are able to see their calorie intake in the form of a calorie dashboard. The users could possibly choose to view daily or weekly calorie intake. The app is user-friendly, which encourages user engagement. The development of this project included learning new things, such as using Python to detect food in images taken in the app.

Project Timeline

The project timeline was carefully planned to ensure that all tasks were completed on time. This ensured that the project was delivered on time and with excellent quality. The project began with sprints 1 and 2 in October. During these sprints, the project idea was finalized and created a comprehensive project proposal. The primary goal of these sprints was to establish end-to-end communication in the app, which is an essential feature to ensure a seamless user experience. Moving on to November, sprints 3 and 4 were worked on. During these sprints, Identifying food items in an image was worked on. Additionally, work was done on implementing a database, which was an integral part of the app. The team also focused on improving the front-end design of the app, which enhanced the overall look and feel of the app. In December, Sprint 5 was worked on, which involved calculating the number of calories found in the food that was identified in the image. This feature is essential for users who are concerned about their calorie intake. Work was also done on the calorie dashboard, which provided users with a comprehensive overview of their calorie consumption. Moving on to January, work was done on sprint 6, which involved testing and improving the app's functionality and usability. This sprint was crucial as it helped highlight any issues within the app and allowed for necessary improvements to ensure that the app was user-friendly and working correctly. Finally, in February, work was done on Sprint 7 and Sprint 8. These sprints involved working on the dissertation, which was an essential part of the project. The focus was on creating a detailed and comprehensive dissertation that was to showcase the hard work and dedication throughout the project. Overall, the project timeline had been carefully planned to make sure that all tasks were completed on time and efficiently. The development of the project was highly motivated to deliver a high-quality app that was user-friendly and provided users with a seamless experience.

2.2 Project Objectives

The success of this project was determined by its ability to achieve its objectives. It was extremely important to set clear and realistic project objectives that could be measured and evaluated. To ensure its success, the aim was to establish and execute a plan that effectively meets the project objectives.

2.2.1 Identify Food using Machine Learning

The plan was to utilize machine learning technology to identify the food item captured in an image. Libraries such as Open CV and TensorFlow could be used to process images and analyze their features, such as objects or faces. Open CV is one of the biggest computer vision libraries. This library contains functions used for real-time computer vision tasks. It is open source, containing over 2500 algorithms. TensorFlow is an open-source platform used for machine learning that focuses on developing and training machine learning models. Machine learning algorithms and datasets were used to identify food in images. Datasets such as Food-101 and Food-11 were used in the identification of food.

2.2.2 Calculate Calories

Research was done into estimating portion size using machine learning. It was discovered that it is quite unreliable and inaccurate. This was because the machine learning model had difficulty determining the size and density of the object. In order to address this, the plan was to get user input for the portion size of the food identified. Possible user input is a portion is equal to 100g and the user must enter how many portions in relation to that. This is done after the image is taken, and the name of the food identified is used to query the Calorie database through an API. Calorie information for a specific food is obtained by querying a food database such as the USDA National Nutrient Database or food-data Central. There are other food databases available online, but initial research began with the USDA National Nutrient Database.

2.2.3 Real Time Meal Tracking

To ensure that users' information about food and calories is correct and reliable, it is essential to update the user's calories in the database regularly. This means keeping the calorie dashboard up to date with the latest information to help users make informed decisions about their diets. Utilizing pictures to locate food was a beneficial capability. Nonetheless, it was essential to guarantee that the fundamental information was precise and current.

2.2.4 Display Nutritional Information

The plan was to display the nutritional information to the user in a calorie dashboard. The user can select whether they want to see their calories on a weekly or monthly basis. The calorie information is fetched from the database and displayed to the user. The main function of the dashboard is to provide the user with information about their daily or weekly calories all in one place. This allows them to make informed decisions about their diet.

2.3 Explanation of Dissertation Sections

This dissertation contained several sections that provided a project overview, its scope, objectives, and development process. The introduction section outlined the project and provided a brief overview of its scope and objectives. It described the problem the project was trying to solve and why it is important. This section also included links to related GitHub repositories that contained documentation, code, and other resources related to the project. The methodology section described the approach taken by the team to develop and test the project. It provided details on how the team worked together, the tools and technologies used, and the testing and validation methods applied. This section also explained how GitHub was used for collaboration and version control, which allowed the team to work efficiently and effectively. The technology review section provided an overview of the technology used in the project. It presented the technology in a similar structure to a literature review, focused on its conceptual aspects and its relevance to the project. The system design section showed the system architecture and how the project worked. It detailed the components of the system and how they interacted. It also explained how the project was designed to meet the objectives and requirements. This section provided a comprehensive overview of how the project was built and how it functioned. The system evaluation section evaluated the project's objectives and highlighted any limitations encountered during its development. It provided an assessment of how the project met its objectives and requirements and discussed any issues or challenges encountered. This section helped the reader understand the strengths and weaknesses of the project and its potential for future development. Finally, the conclusion section summarised the project's context, objectives, and results of the system evaluation. It provided a concise overview of the project's achievements, the lessons learned, and the potential for future work. Overall, the report provided a comprehensive overview of the project, its development process, and its potential impact.

2.4 Github links

The final year project utilized GitHub as the primary tool for keeping track of project versions.

2.4.1 Frontend Repository

This repository contained a React application. It contained each page of the application along with routing. This was the part of the application that the user interacted with.

2.4.2 Python Repository

The repository had a Flask service that receives requests from the React front-end. It contained Python code for image recognition using different machine-learning models, such as InceptionV2 and YOLOv8. Additionally, a model was trained on a custom dataset. The objective of this model was to detect more food types in YOLOv8. This repository contained a requirements text file, which was used when creating virtual environments in Python.

Virtual Environment

A virtual environment is a private space on a computer where Python code and packages for a specific project are kept. The project greatly benefited from virtual environments, which enabled easy installation of all necessary packages through a few simple commands. There are many great reasons to use a virtual environment in Python. If a machine has many different Python versions, switching between them can get complicated.

Chapter 3

Methodology

3.1 Agile Approach to development

The preferred development methodology for the project was agile. This approach was considered the best because it broke down the project into manageable chunks that were distributed over sprints.

3.1.1 Planning

In this stage, the foundations of the project were set. The project scope, requirements and important features were decided. To determine project requirements, user stories were developed to capture the requirements. These user stories were then broken down into tasks, which were added to Jira. Jira is a project management tool that allows for issue tracking. Tasks that need to be completed are stored in the backlog. When a task was ready to be completed, it was moved to ready. When it was being worked on, it was in progress and then finally moved to done. The Kanban board can be seen in Figure 3.1, which was used for project management. In figure 3.2, a list of done issues can be seen.

3.1.2 Analysis

In this phase, the user stories and requirements identified were analyzed. This phase allowed for the identification of potential blocks and challenges that could occur.

3.1.3 Design

In this phase, the following elements of the project were mapped out:

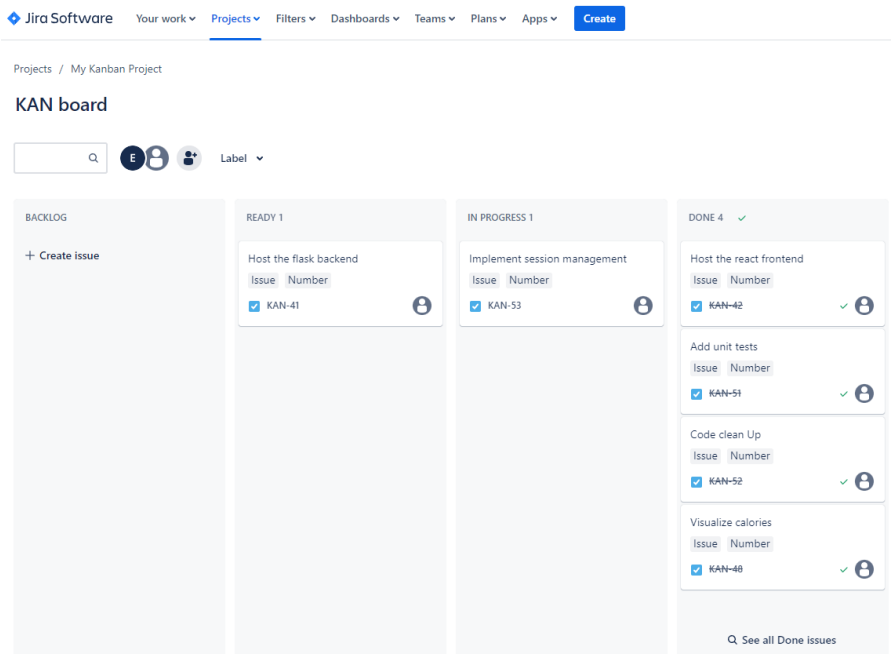

Figure 3.1: Kanban board

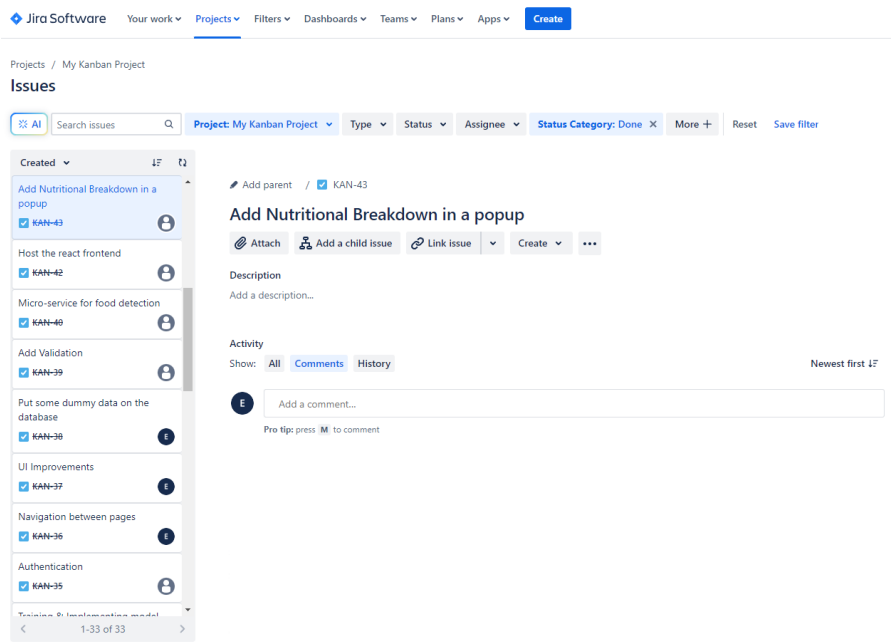

Figure 3.2: Kanban board Done Issues

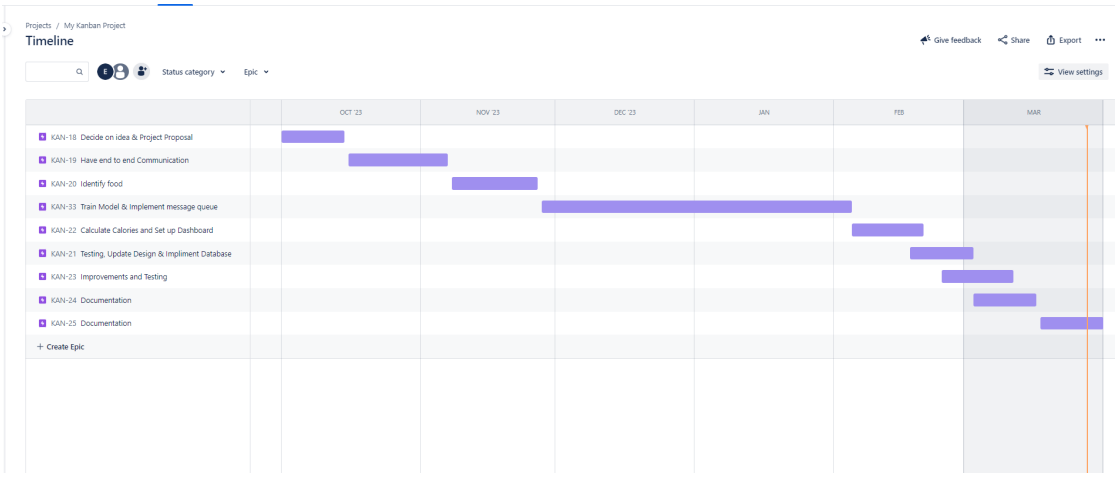

Figure 3.3: Project Roadmap

- Map out the system architecture in an architecture diagram. This showed the relationship between each part of the project.
- The initial design of the user interface was user-friendly and intuitive.
- Database design defines the structure and relationships between data elements in a database system.

3.1.4 Development

In software development, sprints were a common approach to managing a project's progress. Each sprint lasts about two weeks on average. A set of tasks was worked on, and a piece of code was added to the project's functionality at the end of each sprint. This approach helped the project stay focused and motivated. During each sprint, a set of tasks from the product backlog is developed. To give a clear picture of the overall project plan, a roadmap was often used. Figure 3.3 shows a project roadmap that outlines the major milestones and deliverables for the project. This helped the project to stay focused and on track.

Meetings

Throughout the project, weekly meetings were held with the project supervisor. These meetings were essential in keeping the project on track and ensuring that progress was being made in the right direction. The meetings provided a clear objective for the week ahead, which helped to keep the project focused and moving

forward. During these meetings, the progress of the project was presented to the supervisor. The supervisor was then given the opportunity to provide feedback and suggestions on how to improve the project. This feedback was crucial in ensuring that any issues or potential roadblocks were identified and addressed promptly. The project supervisor's feedback also allowed the project to make any necessary adjustments and improvements as it progressed. This helped to ensure that the final product met the desired specifications and requirements.

Milestones

A milestone is a crucial point of progress in a project that serves as an essential checkpoint. It defined a clear target and indicated progress toward the overall project objective. Meeting milestones was crucial because they were critical points that had to be achieved to avoid delays in the project schedule.

3.1.5 Testing

During the development of a project, various manual tests were carried out to ensure the functionality of the system. However, during the testing phase, the project was extensively tested to check whether it met the acceptance criteria defined at the beginning of the project. This phase was critical as it ensured that the code was bug-free and all edge cases were resolved, thereby ensuring that the application or system worked as intended. Comprehensive testing was essential to identifying any issues or bugs and fixing them before the final product was delivered to the customer. Proper testing can save time and money and ensure that the end product meets the expectations of the users. Therefore, it was important to give proper attention to testing during the development cycle to deliver high-quality applications that meet the requirements of the customers.

3.1.6 Deployment

This phase involved deploying the software to the end user, hosting the app, and ensuring that it can be accessed by users.

3.2 Validation and testing

Testing at the beginning of development was primarily done during ticket development. This testing process involved ensuring that the code was functioning efficiently and did not break any previously working functionalities. Both unit tests and manual testing were done through the development of the project. Unit

tests were added to some aspects of the project, such as login and sign-up. Jest is a testing framework used for testing the Frontend code. It was developed by Facebook and is known for its simplicity and ease of use. Although unit tests were only developed for some aspects of the project, given more time, adding more unit tests would be a priority.

3.3 Research Methodology

To conduct the project research, an empirical approach was used.

3.3.1 Research Process

First, research was conducted to identify available food-tracking apps and analyze potential gaps in their functionality. This was done by examining what apps were available and how they worked.

3.3.2 Finding

The research observed that many apps were available to track food. These apps rely on the user to search for the food and enter the portion size. Food-tracking apps that utilize barcode scanning are also available. However, there was a lack of apps that detect food by taking a picture of it. For example, the Fitbit app does not currently have a feature for identifying food from images. [1]

3.4 GitHub

Github was used throughout the whole development of the project. An organization was created as it allowed for multiple repositories to be contained in one place. An organization also allows for multiple users to work together on a repository within the organization. This could be beneficial for teamwork. The use of a virtual environment could also be very beneficial for teamwork. It allowed for multiple users to get the correct packages needed for the project. This was a great tool that helped save time and allowed for more concentration on development work. The Python repository has a guide on how to set up the virtual environment.

3.5 Technology Selection criteria

3.5.1 Language selection

- The selection of a programming language for the project was based on the objective of using something new.
- The language needs to be used in developing projects in machine learning and image recognition.
- It should also have machine learning libraries for machine learning and AI.

3.5.2 platform selection

- When it comes to selecting a platform, there were a few key requirements that needed to be met in order to ensure a smooth and successful experience.
- First and foremost, it was essential that the platform was scalable, meaning that it could grow and adapt as needs change over time.
- A platform that is not scalable can quickly become outdated and inefficient, leading to frustration and decreased productivity.

3.5.3 Model Selection

- The machine learning model chosen must be suitable for object detection.
- When selecting a machine learning model for food recognition, it was important to choose one large enough to identify a wide variety of food items.
- The model should be able to accurately identify a large amount of food, including different types of food, different colours, and different shapes.
- This would help ensure that the system can recognize a wide variety of food items, even those that are less commonly encountered.
- The model should not be under-fitted or over-fitted.

Under-fitting

A model is seen to be under-fitted when it cannot capture the relationship between the input data and the target value, leading to poor performance on training data.

Over-fitting

A model is seen to be over-fitted when it performs well on the training data but not on the evaluation data. The model memorizes the data it has seen and can't generalize unseen examples.

3.5.4 Hosting platform selection

- The platform had to be relatively easy to use and deploy the application
- The price of the hosting platform had to be cost-effective and evaluated for potential hidden cost
- The platform should have the ability to expand and accommodate the growing needs of the application.
- It should have been capable of handling an increasing amount of traffic.
- It should have had comprehensive documentation and an active community.

Chapter 4

Technology Review

4.1 Frontend

When it comes to choosing the framework to develop the Frontend in, extensive research was conducted to determine what would be the best fit for the particular project at hand. Factors such as ease of use, scalability, maintenance, and community support were all taken into account during the selection process. Ultimately, a thorough evaluation led to the identification of a framework that would not only meet the project's requirements but also provide a seamless and intuitive user experience. The chosen framework enabled a highly functional and visually appealing interface that met the needs and expectations of the end-users.

4.1.1 Vue

Vue is a JavaScript framework that is commonly used to create user interfaces. It is lighter and more adaptable than other available frameworks. Vue can be used in the development of web and mobile apps. Vue also contains a model view controller which is used for displaying web pages. [2]

MVC

It consists of a model, a view, and a controller. It provides a way of organizing code so each section has a purpose. The model defines the components of the app. For example, in relation to the project being developed, it would define what a food and user are. A view consists of the project's functionality. The controller acts as a middleman between the model and the view, receiving user input and directing it where it's supposed to go.

Advantages and Disadvantages

- VUE is a small and fast framework that is suitable for beginners.
- It is also a component-based architecture like many other JavaScript frameworks.
- While it is a good framework for beginners, it is not suitable for the development of large projects. [3]

Vue was not chosen due to its limited scalability, fewer libraries and tools compared to React, and a smaller community.

4.1.2 Angular

Angular is a typescript-based framework that could be used in the development of single-page web applications. Angular is a dynamic web application as the data is updated and displayed without the need for updating by the developer. The project architecture is similar to the view. Its based on components where each component is responsible for part of the user interface and the behaviour relating to it. Components consist of a view, a modal and the actual component class. The view is the HTML of what the end user sees. The model is the data the component is dealing with such as retrieving data from an API. [4]

Advantages and Disadvantages

- Angular is also a component-based architecture like Vue.
- It has a large ecosystem and a strong community
- It can be quite complex for beginners and suffers from version control and mitigation challenges

4.1.3 React

React is a front-end JavaScript library that is used for building user interfaces. Like Angular and Vue, React is component-based. It is used to develop and build web, mobile and desktop applications. Reacts main concern is the user interface and rendering components to the dom. This fact makes applications rely on libraries for things such as routing. [5]

Framwork	Advantages	Disadvantages
Vue	• Small and fast • Component-based architecture	• Not suitable for large projects • Small community
Angular	• Component-based architecture • Big ecosystem • Large community	• Complex for beginners • Version migration challenges
React	• Easy to learn and use • Component-based architecture	• Boilerplate code required • Lack of routing

Table 4.1: Frontend framework comparison

Advantages and Disadvantages

- React is a component-based architecture.
- It is easy to learn for beginners and for experienced developers alike to use
- However, it relies on boilerplate code for things like state management.
- As mentioned above, It requires libraries for routing as it does not contain routing internally.

For a more condensed comparison of the frameworks, please see Table 4.1. In the end, after research, React was chosen due to its flexibility, scalability, and widespread use in the industry, making it an ideal choice for the project.

4.2 Data visualization

It was decided to display both recommended and consumed calories on the dashboard. Chart.js is a JavaScript library used for creating charts on web pages. It is a simple way to visualize data for the user.

4.3 Database

In the beginning stages of the project, research was done to determine which database to use based on the structure and size of the data.

4.3.1 MongoDB

MongoDB is a cross-platform document-oriented database. Each record in this database is stored in a document that is described in a BSON. A BSON is a binary representation of the data. When the application wants to retrieve information, it is in a JSON format. MongoDB can handle a large amount of data and is scalable. In regards to this project, where it may be necessary for each food item to have different attributes, MongoDB allows the storage of data with different structures. MongoDB allows for the storage of user information, such as login, through references and relationships between documents. However, developers are required to manage the integrity of the relationships in regard to the application logic. [6]

4.3.2 SQL

MySQL is an open-source relational database management system. Data is stored in tables, rows, and columns. In terms of the project, tables could be users, which store user information, and foods, which store the user's foods. These tables could be connected via a one-to-many relationship. For example, this could mean one user has many foods. [7] In the end, it was decided to use SQL as the database to store the information from the project, such as users and food. Although SQL can't be as flexible as MongoDB, It was the preferred option as the project had more structured data, for example, storing user's login Information and user calories.

4.4 Backend

4.4.1 Backend Framework

When developing an application, a backend framework such as Express or Flask must be chosen to connect the front end and the database. This framework is required to save and load information from a database.

Express

Express is a popular framework used in React apps. It is built on top of Node.js. Node.js is a JavaScript runtime environment that allows JavaScript to be run outside of the web browser. It was created in 2009 and became popular in the development community. It allows developers to build scalable and high-performance network applications. Express is widely used to develop both small-scale and large-scale applications. Although Express is a user-friendly platform, it may require more effort to learn than other similar tools. Both Flask and Express have a very active community of developers as well as substantial documentation. When it comes to performance, both Flask and Express can handle high loads while being lightweight.

Flask

Flask is a framework used in Python. It follows the WSGI framework, which is a calling convention used to forward requests to web applications. [8] Flask is popular in the Python community and has a large ecosystem of libraries and tools. Flask is known to be simple and more minimalist, making it more beginner-friendly. It is flexible but requires a lot more configuration for larger projects. Flask was chosen as the preferred framework for the project, given that the individuals involved had previous experience with Express. The decision to use a new framework was motivated by the desire to expand knowledge and skills. Additionally, Python was being utilized for the machine learning component of the project, making Flask a suitable choice. [9]

4.4.2 Language for machine learning

When the project idea was decided, research was completed to determine which language had the best support and libraries for object detection. Many great options for developing machine learning code exist, including C++ and Python.

C++

The first choice evaluated was C++. C++ is a compiled language, which means it is directly executed on the computer's processor. Python is an interpreted language, which means it must be translated into machine code at runtime. In this sense, C++ is faster than Python. In C++, developers are able to control memory, which makes it more memory efficient, as a developer can allocate and deallocate memory. However, it requires responsible memory management, irresponsible management can result in memory leaks and other problems which can decrease the performance. [10] However, it does have some drawbacks. Using C++ requires much more programming knowledge and experience than Python. C++ also has fewer machine-learning libraries than Python, which can be seen as a disadvantage. The community that uses C++ for machine learning is also quite smaller than Python, which can be a disadvantage when it comes to solving issues and community support.

Python

As mentioned above, Python is an interpreted language that can be slower than C++. Python is known for being easy to learn and use, which is a great advantage over C++ for developers learning machine learning. The community surrounding Python is vibrant and active, which contributes to its development. The community provide useful resources and tutorials. However, Python is widely used for machine learning and has many machine learning libraries, such as YOLOv8. YOLOV8 was developed by Ultralytics, a company that contributes to computer vision and deep learning projects.

4.4.3 Machine learning models

Selecting a suitable model was crucial for the project's success. In the initial stages of development, Experimentation was done with the Inception-v3 model.

Image classification models

Inception-v3 was the first model discovered during the project's research. It is an image classification model with an accuracy rate of 78 per cent. The model is trained on the ImageNet dataset.[11] Research and experimentation were also done on MobileNetV2. This model is also a classification. The suitability for real-time classification was considered to be beneficial for the project. Initially, it was thought that these models worked well, but with more discovery, the realization occurred that instead of an image classification model, an object detection model should be used. An image classification model classifies the image as a whole. For

example, if a picture was taken of a banana and an apple. It would either identify one of the food items or identify them as fruit. Unfortunately, this is not the best solution for the project.

Object Detection models

While researching, It was discovered that the most popular model for object detection was YOLOv8. YOLOv8 is a pre-trained model trained on the COCO dataset. The coco dataset is a large dataset used in training object detection models. It contains over 330 thousand images, with over 200k being labelled. It is categorized into 80 object categories. [12]. During the process of using the model, it was observed that it performed impressively well for the given project, accurately recognizing food items such as apples, bananas, and pizza. However, it was noticed that the model's training was limited to only a few food categories. Since YOLOv8 only identified a restricted number of food items, the app would be limited to identifying a narrow range of food items. The next suitable approach was to train YOLOv8 on a food dataset. [13]

4.4.4 Training a model

Dataset Requirements

The dataset must meet a few requirements to be suitable for training a model. The first one is more general: The dataset must contain the correct classes, such as a training set, validation set, and Evaluation set. This is essential to train a working model. The dataset must also contain labelled images to identify food. Lastly, the images must contain bounding boxes. This is necessary for object detection models to get the object location in the image.

Food-101

In 2014, a research team created the Food 101 dataset to solve the problem of identifying food. This dataset consists of 101 food classes, with 101k images for each class. Although the training set contained a considerable amount of noise, it was still useful. However, it is important to note that this dataset is an image classification dataset and not an object detection dataset. Therefore, it does not contain labels and bounding boxes. To efficiently train the model, an object detection dataset with labels and bounding boxes was needed. More research was conducted to find such a dataset that met the criteria. [14]

Food-11

While researching, the food-11 dataset was discovered. The Food 11 dataset was created in 2020. This dataset answered the key problem of having a food dataset with labels and bounding boxes so it can be used to train object detection models. It contains 11 general categories of food, such as bread, dairy and meat. [15]

4.5 Model Training

The next step in development was to train a model capable of detecting a wide range of foods. Unfortunately, the development machine used for testing did not meet these requirements, so cloud resources were used.

4.5.1 Hardware requirements for training a model

Research then had to be done into what hardware the virtual machine would need to train the model in a good amount of time. During the online research, a GitHub issue on ultralytics was discovered, providing valuable insights on the hardware requirements for training a custom model on yoloV8. The requirements are as follows:

- NVIDIA GPU with 24GB of memory
- Requires an Intel i7 and an AMD Ryzen 9
- 32GB of RAM
- 1 TB SSD

[16]

4.5.2 Training process

GitHub, which can transfer code via git clones, was utilized to facilitate the transfer of code between a local machine and a virtual one used for training. As mentioned above, a dataset with bounding boxes and labels is required for training YOLOv8 on a custom model. The command below was run on the virtual machine to train the new model.

```
yolo task=detect
mode=train
model=yolov8n.pt
```

```
imgsz=640
data=/home/comp08011/Documents/FYP_Python/src/Dataset/data.yaml
epochs=10
batch=8
name=Custom_dataset
```

The above command intends to create a model for object detection, and a training model is specified. The model is chosen as YOLOv8, and it will be trained on a custom dataset. The data.yaml file was also specified. This file contained information about class names, image paths, and labels. The number of epochs was set to 10, so it passed through the training dataset 10 times. The training process typically took 20 minutes, which was a lot better than previous training times on the development machine.

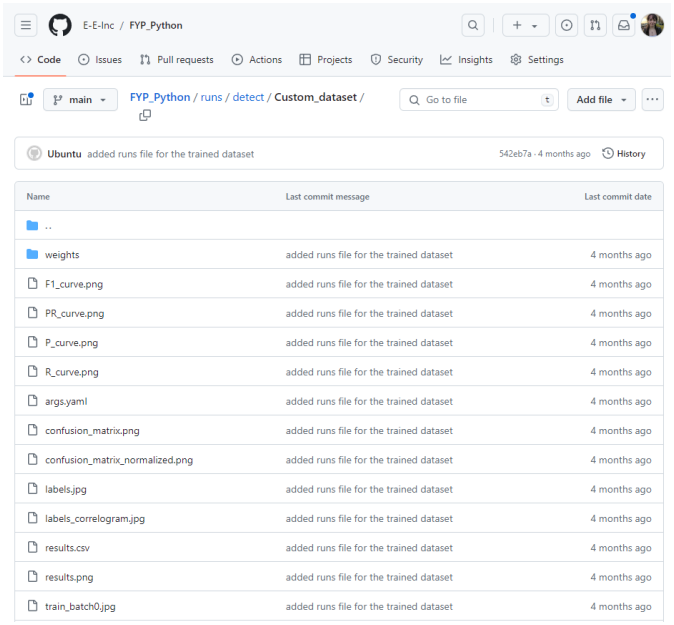

Figure 4.1: Model training in Github Repository

4.5.3 Problems

In order to use and validate the model, the new model can be used in the Python code, as seen above. The first port of call was to see if it recognized what it was trained on, which was successful. The next thing was to test if the model could recognize what it previously knew. Unfortunately, it didn't retain that information. More research was done, and it was discovered that a thing called

catastrophic forgetting occurred. This can happen when the dataset the model is being trained on is greatly different from the classes the data is already trained on. It was concluded that the project should use the original yoloV8 model for development.

4.6 Hosting

During the final stages of development, work was done on hosting the project. The following sections outline the research and final decision.

4.6.1 Database

Azure

Azure is also great for hosting databases such as SQL. It's easy to use, as it has an Azure portal that allows users to manage their databases. When it comes to security, Azure provides built-in security features such as access management and threat detection. However, if the database that needs to be hosted is large, it can often carry a substantial cost.

Google cloud platforms

The Google Cloud Platform offers Cloud SQL to host MySQL databases. Many businesses around the world use Cloud SQL. It allows for the easy creation of databases. A great feature is the automation of backups, replication, encryption, and storage. It also offers reliability, scalability, and security. It has the great feature of giving 300 dollars in credits to new users. [17]

AWS

AWS offers a service called Amazon RDS. It allows for the creation, operation, and scalability of a database. AWS is a valuable skill for future graduates. It was utilized during an internship and proved to be a beneficial experience. A database can be set up within 30 minutes. Teams benefit from local database access so multiple users can work together simultaneously. In the end, we decided to choose AWS as it is widely used in the industry and easy to use.

4.6.2 Frontend

Research was also completed into what services are available to host the Frontend of the project.

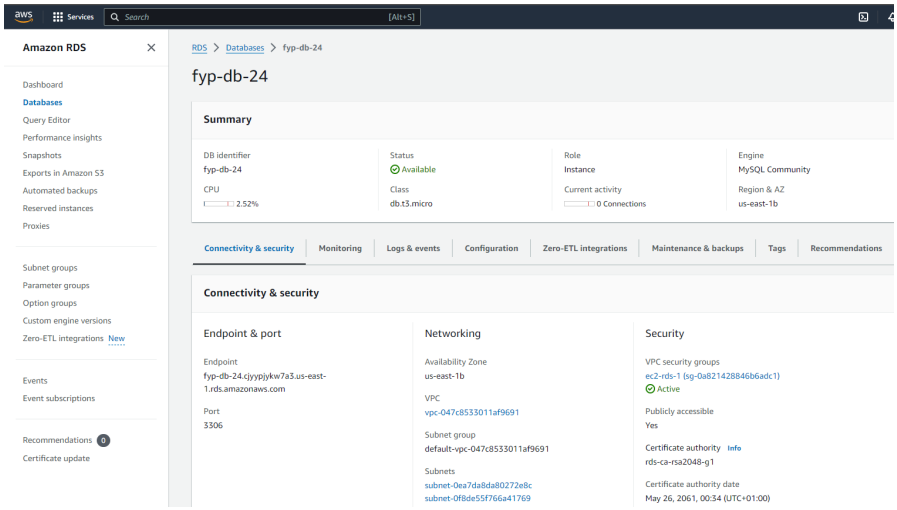

Figure 4.2: Database hosted on AWS

Github Pages

Firstly, experimentation was done on hosting the react on Github pages. Github pages are easy and free to use. However, they do have limitations. Github pages are very good at hosting static web apps but not so good at hosting dynamic web apps. This wouldn't be a good choice for hosting the app.

AWS

AWS, however, is a good option for hosting the front end. It also requires the developer to be familiar with AWS as it is more complex to host on AWS than Railway or GitHub pages. AWS is known to be scalable but can be expensive.

Railway

Railway is a hosting platform that enables users to host web applications, services, and databases. As a developer, it offers an ideal balance of usability and scalability. Setting up the hosting is easy - all that is needed is to log in to Github and select a repository. Additionally, a docker file must be set up to ensure that the app is rendered correctly. The scalability of Railway depends on the infrastructure of the app being hosted. While Railway does offer a free tier, there are limitations. During the development phase of the project, the hobby tier was utilized for hosting purposes, which cost 5 euros per month. In the end, it was decided that the railway was the best option to host the front end. It is the perfect middleman between scalability, cost and ease of use.

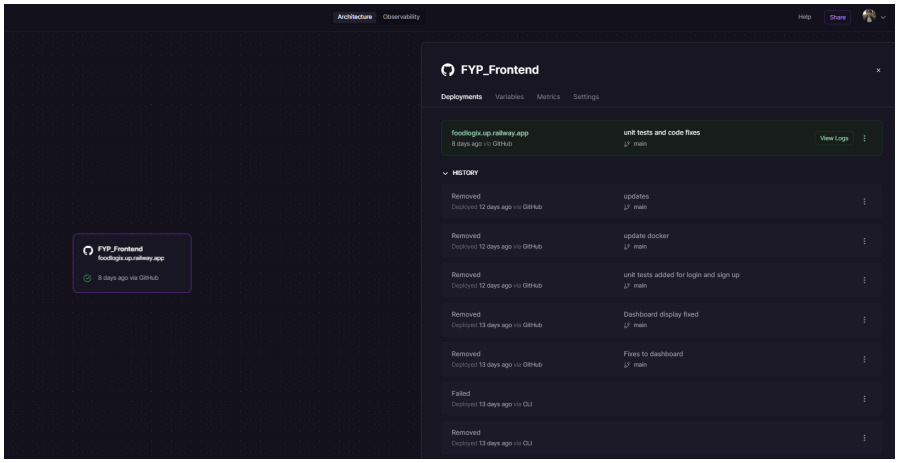

Figure 4.3: Railway

4.6.3 Backend

While developing this project, research was completed into the best service that can be used to host the flask service and food recognition microservice.

Heroku

Heroku is a cloud platform for hosting applications. Its beauty is that it abstracts away infrastructure management to allow developers to focus on building and deploying applications. Given this fact, Heroku streamlines the development process. Heroku is also integrated with GIT. Heroku also offers scalability features which allow for automatic scaling by increasing or decreasing the number of containers based on the application load. However, the scalability factor is more limited than AWS. [18]

Railway

As mentioned previously, Railway is a platform that allows users to host web applications, services, and databases. It strikes an excellent balance between usability and scalability. The hosting process is straightforward: log in to GitHub, select a repository, and set up a Docker file to ensure proper application rendering.

AWS

AWS offers a wide range of services for hosting backend services, including Elastic Compute Cloud (EC2) for virtual servers and Elastic Beanstalk for easy deployment and management of applications. AWS also offers full control of infrastruc-

ture and services. AWS allows developers to scale both horizontally and vertically. AWS uses auto-scaling and elastic loading for scalability. It offers auto-scaling based on demand by adjusting the number of EC2 instances in an auto-scaling group. Elastic load balancing, however, offers the distribution of incoming traffic across multiple targets, for example, containers or IP addresses.

It was decided to use Railway for hosting the backend.

4.6.4 Problems Encountered

Setting up the backend was straightforward initially. It involved creating a Dockerfile that exposed the ports for both the service and microservice. Some CORS issues were encountered during testing. Fixing these issues involved setting up the service to take requests from other sites. The outlook seemed promising, but unfortunately, the microservice still did not connect. Both services were hosted on a railway, but they were unable to properly use the food identification microservice.

Chapter 5

System Design

Food Logix is a food-tracking app that utilises machine learning to identify food captured in an image in the app. The user can track their food, see their calories for the day, and see how they compare to the recommended calories. The system was designed to be user-friendly, resilient and secure. The components are discussed below.

5.1 Components

5.1.1 Client Interface

The user interacts with the system through a user interface designed using React. The user interface communicates with the backend through restful API requests, such as HTTP get and HTTP post requests. It was designed to be user-friendly and meet the project requirements.

5.1.2 Authentication

When a user first loads the app, they are met with a Login and Sign Up page where authentication is completed. On the user side, it is a simple login and sign-up page, but on the backend, manual authentication is completed.

5.1.3 Registration

When the user clicks register, the password is hashed, and an HTTP post request is sent to the flask server. Passwords are encrypted with B-Crypt, a hashing algorithm that secures sensitive information. The algorithm is secure but limits passwords to 72 bytes. Next, the information is inserted into the SQL database. During the initial registration, the user enters their email and password. If the

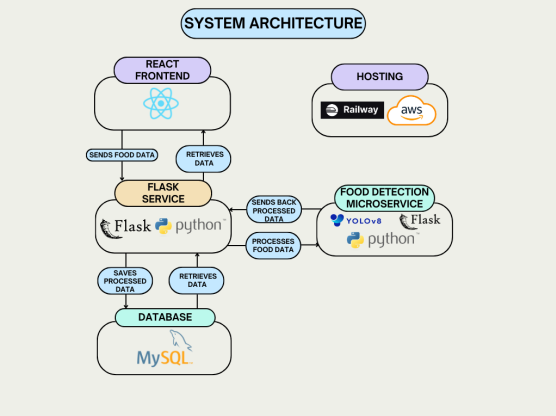

Figure 5.1: Architecture Diagram

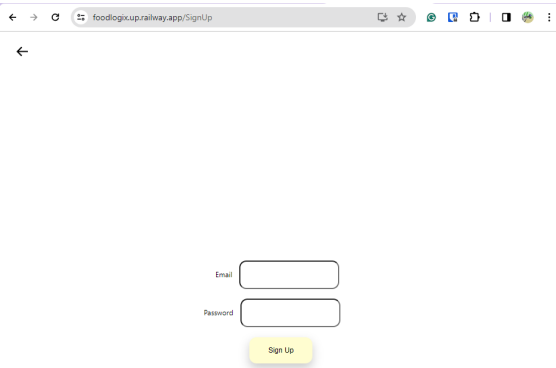

Figure 5.2: Sign Up page

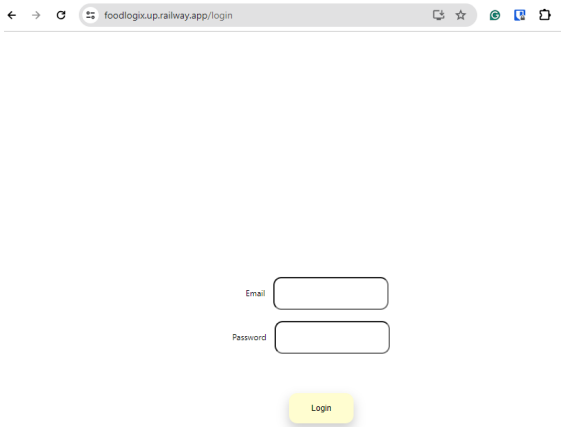

Figure 5.3: Login Page

initial registration is successful, a modal pops up for the user to enter additional information such as Gender, Age, Weight, and Height. This information is needed in order to calculate the recommended calories needed for the day.

5.1.4 Login

When the user clicks login, an HTTP post request is sent to the Flask server. The SQL database is queried with the email and password to see if the information exists. If it does, the user is logged in and redirected to the home page.

Validation

Validation is also done during authentication, where the user's entered email and password values are stored.

- If there is no information entered, an error message appears to ask the user to enter an email and password.
- In order to prevent duplicate entries, a get request is sent to the backend to check if the email has already been registered. If it has, an error message appears asking the user to enter an unregistered email.
- In order to prevent invalid email entries, the user must enter a valid email to register or log in. If an invalid email is entered, an error message appears, asking a user to enter a valid email.

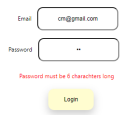

Figure 5.4: Invalid password validation in login

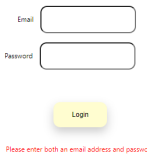

Figure 5.5: User must enter both a login and email validation

- Validation is set up so that the user must enter at least 6 characters in order to sign up and log in.

5.1.5 Web Server

The web server is built using the Python programming language and the server is implemented using the Flask framework. Flask was very easy to use and implement. Flask provides a consistent and reliable way to handle HTTP requests and responses. It allows developers to define routes for functionality such as Login and sign up. Flask supports HTTP methods such as get, post, put and delete. The web server is run locally.

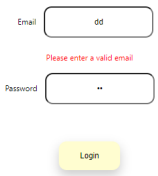

Figure 5.6: Email Validation Login

5.1.6 Backend Service

The backend service is composed of two flask services. One is the main service and one is a microservice. The main service handles the following:

- Connection to the SQL database
- Sign up and update information endpoint to handle user registration
- Login endpoint to handle user login
- Information endpoint to return the user calories for a specific day
- The needed calories endpoint returns to the recommended for a user
- Get Nutrition endpoint returns the calorie breakdown for the food specified

The microservice is responsible for using machine learning to identify different food items. The microservice and service communicate via HTTP requests such as HTTP get and HTTP post.

5.1.7 Database Layer

The SQL database is hosted on AWS. The developer can access the database via the SQL console on a local machine. This is beneficial for teams as multiple users can access the database at the same time. The user must enter the following command to interact with the database locally. Please note that the following command is a general command. After this command is entered, the user must enter the password. The password is specified during the creation of the database on AWS RDS.

```
mysql -h {AWS database url} -P 3306 -u { AWS database Username} -p
```

Security Breach

During the development process, a security breach was identified. During database creation, the database username was set to root and the password was set to root. It was also set to be accessible by all IP addresses. Unfortunately, the database was hacked due to the username and password being so generic. It is suspected that bots scanning the internet for open databases with common usernames and passwords found the database and were able to access it. All of the data stored on the database was removed and replaced with a ReadMe table. Inside was a

message stating, "Please send 0.1 bitcoin to this address to get your data back". Thankfully, this is a college project, so no important user data was stored. If this happened in industry, it would be a huge problem. In order to recover from this, the database instance on AWS RDS had to be deleted and another database instance created. A more secure username and password were chosen to ensure that the database was secure. Thankfully, the database script was saved, so the script had to be put into the SQL console while logged into the database instance.

Database Schema Creation

MySQL Workbench was utilized during the project's development. It delivers visual tools for creating, developing, and managing an SQL database. The database was designed using MySQL Workbench as a visualization method that helps to illustrate the relationships between each table. After that, forward engineering was finished. Forward engineering is the automated process of converting a class diagram into an SQL script. This script was then used to make a database through the MySQL console.

5.2 Sequence Diagram

A sequence diagram is a type of UML diagram. It focuses on the process and objects that live simultaneously and on the messages and data exchanged between them to perform a function. The sequence diagram in Figure 5.7 depicts how data is exchanged between components of the project during login. The sequence diagram in Figure 5.8 depicts how data is exchanged between components of the project during registration. Figure 5.9 shows a sequence diagram of how the image captured by the user is processed and returned.

5.3 Coupling

One of the best practices for designing software systems is to promote flexibility and maintainability through the use of loosely coupled components. This means that different parts of the system are designed to work independently of each other, making it easier to modify and update individual components without affecting the entire system. To achieve this, components in modern software systems often communicate using the HTTP protocol and REST architectural style. RESTful APIs allow components to exchange data and functionality in a standardized way, making it easier to integrate different components and services into a cohesive system. By using HTTP and REST, software developers can create scalable, modular, and robust systems that can be easily maintained and extended over time. This

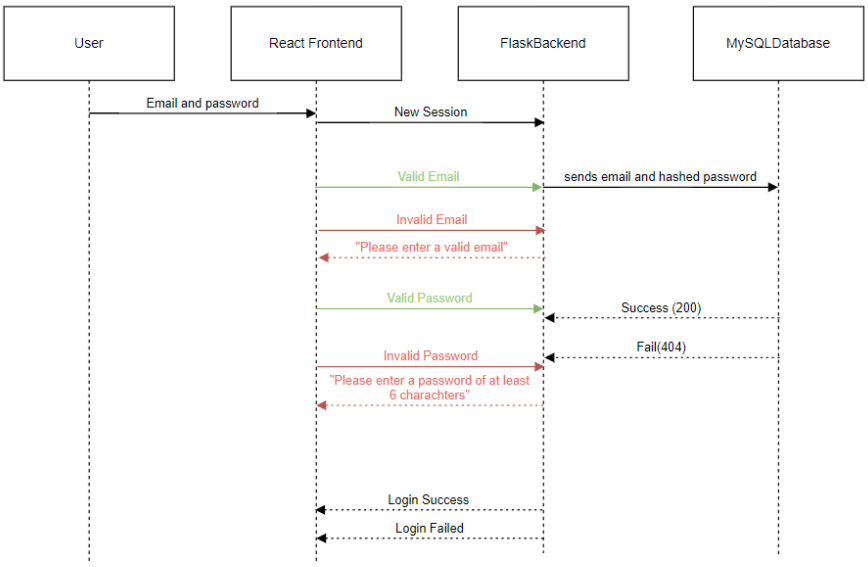

Figure 5.7: Login Sequence Diagram

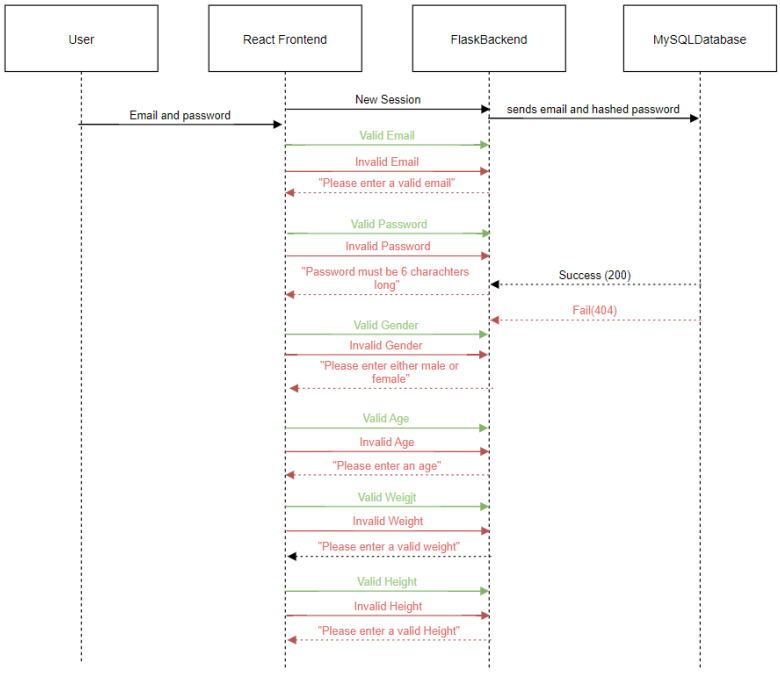

Figure 5.8: Sign Up Sequence Diagram

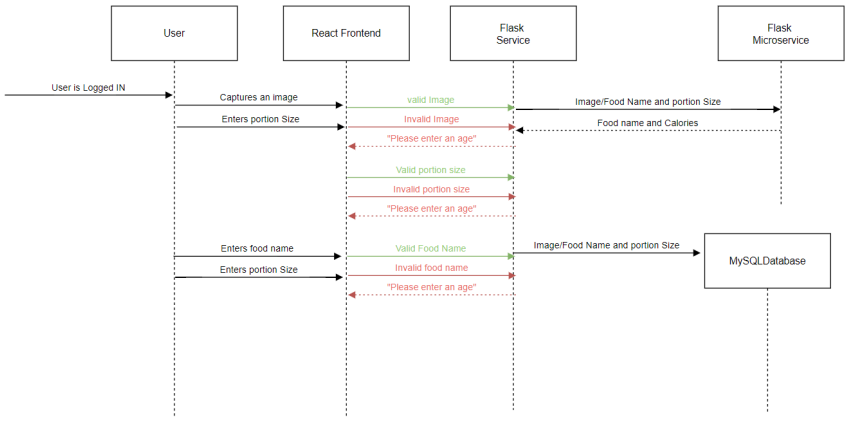

Figure 5.9: Capture Picture Sequence Diagram

approach promotes the separation of concerns, reduces dependencies, and ensures that each component can be tested and optimized independently.

Chapter 6

System Evaluation

The strengths and weaknesses of the project were thoroughly examined, providing insight into what went well and what could have been improved during development. In addition, the limitations of the project were explored to understand their impact on the overall performance of the system. Finally, the project's performance was analyzed in its entirety, assessing its effectiveness in achieving its intended goal, as well as the quality of its outputs. The aim of this evaluation was to provide valuable insights to inform the development of future projects.

6.1 Robustness

Ensuring a software project's robustness is a crucial step in its development cycle. Several techniques are employed to achieve this, including unit tests and manual testing. In the case of this project, both of these techniques were used to ensure its robustness.

6.1.1 Manual Testing

To ensure the system was robust, manual testing was carried out to ensure it worked efficiently for the user. During manual testing, potential edge cases were explored to ensure that all possible scenarios were accounted for. During development, it was discovered that any user can access every page without having an account. In order to solve this, these routes are protected so that only logged-in users can access them, such as the home, dashboard and login page. Another edge case that was evaluated was to ensure that only valid inputs could be submitted. This ensured that no invalid data could enter the database and be used. This was accomplished through error-prone methods on input fields such as gender and age in the additional user information model that appears when the user signs up. The

purpose of manual testing is to replicate real-life scenarios and prevent unexpected errors.

6.1.2 Automated Testing

Unit tests were implemented for the sign-up and login pages to evaluate their performance and ensure their correct functionality. The unit tests for login and sign-up accomplished a 90% coverage rate, which is excellent.

Overall, the combination of unit tests and manual testing helped to ensure that the project was as robust as possible. By employing these techniques, issues are identified and resolved early on in the development process, which helps to ensure the project's success.

6.2 Performance

6.2.1 Response Time

The system's response time was evaluated, and it was found to be within acceptable limits as specified in the project objectives. On average, it took around 6 seconds to process the picture and produce output while running the backend locally. However, if the application was hosted, the response time varied. The app is highly responsive, providing a smooth user experience.

6.2.2 Throughput

Throughput refers to a system's efficiency in processing incoming requests. This involves evaluating the system's ability to handle a reasonable volume of requests within a reasonable timeframe. Typically, this assessment is carried out towards the end of the development process to ensure that the system can handle the expected load without any issues.

6.3 Achievement of Objectives

- **Identifying Food using Machine Learning**
  - Utilized YoloV8 machine learning model for food identification in images.
  - YoloV8 exhibited high accuracy in identifying food items.
  - Limited set of food items for identification.

- **Calorie Calculation based on Portion Size**
  - To determine the total number of calories per portion size, calculations must be done.
  - It is often complex and inaccurate to identify the portion size in an image
  - Implemented manual input for portion size due to image limitations.
  - Employed a reliable API for accurate calorie and nutrition information.
  - The API offered a comprehensive database of food items for precise calculations.
- **Real-time Meal Tracking**
  - Implemented a feature for real-time meal tracking.
  - Users input their food, and the system displays total calories on the dashboard.
  - During sign-up, users provide additional information (gender, age, weight, height) used to calculate Basal Metabolic Rate (BMR).
  - BMR serves as a benchmark for recommended daily calorie intake.
  - Calories consumed are compared to recommended calories, visually represented via a bar chart.
- **Display nutritional information**
  - Beside each food item, there is an information button to allow the user to view the nutritional information of the food item.
  - The project utilized an API to fetch nutritional information in order to display accurate nutritional information for each food item based on portion size
  - The API used was reliable and contained an accurate breakdown of food items.
  - The API contained a wide range of food items and their nutritional information.

6.4 Project Milestones

Throughout the development of the project, certain milestones and goals were set. When milestones were set, demonstrations were held during weekly supervisor meetings to show the work completed. Some of the main milestones are as follows:

1. Establishing a frontend connection to the backend system.
2. Demonstration of food identification using machine learning
3. Demonstration of database connection by implementing a login and sign up feature
4. Demonstration of database connection by saving food information for users
5. Demonstration of the dashboard for displaying the user's food intake.
6. Recommended calorie intake that is used for comparing with calorie consumption

Towards the end of development, demonstrations became a weekly occurrence as the project was becoming more advanced.

6.5 Future Work

- **Enhanced Food Recognition**
  - In the future, investigations will be conducted on machine learning models with more food items or more time spent on exploring and training the custom model.
  - Although identifying portion size automatically in images would be complex, an exploration into potential methods would be conducted.
  - Given this research and exploration, this would expand the system's ability to identify a wider range of food items.
- **Improved user Experience**
  - The addition of meal planning and recipe suggestions based on dietary requirements would be a valuable feature to implement.
  - Additionally, the user interface continues to be developed with a continued focus on being user-friendly and intuitive.
- **Integration with Wearable Devices or IoT**
  - A cool future addition would be to integrate the app with wearable devices such as a fitness tracker or a smartwatch
  - Wearable devices such as fitness trackers and smart scales can be used to improve meal tracking and estimate calorie expenditure.

- **Implement Social features**
  - The addition of social features such as user profiles, sharing, and community forums could enhance user engagement and support.
  - The use of gamification can motivate and encourage individuals to maintain healthy eating habits and reach fitness goals.
  - Encourage users to create a sense of community and accountability by sharing their experiences, recipes, and fitness tips with each other
- **Privacy and Security Enhancements**
  - It is important to continue to comply with GDPR regulations and implement strong security measures to safeguard the personal and sensitive health data of users.
  - It is critical to continue to implement strong authentication, authorization, and encryption mechanisms to protect user accounts and data integrity.
- **Real-Time Feedback and Coaching**
  - A future incorporation could be to introduce real-time feedback for the user on their dietary choices.
  - One potential implementation could be meal scoring based on the nutritional value of the food. There could also be alerts for accessive intake of certain nutrients.
  - Exploration could also be done into the implementation of an AI coach to guide users of healthier eating habits to achieve fitness goals.
- **Multimodal Input Support**
  - Exploration could also be done into the integration of inputting food data through various means such as voice commands and barcode scanning.
  - This would enhance the usability and accessibility of the app.

6.6 Limitations

- **Manual Input for Portion Sizes**
  - The system depended on the user to provide information regarding the portion size.

  - The accuracy of the data could be compromised if the user enters an inaccurate number of portions.
  - Also, users may find it difficult to estimate portion size, which may lead to data inaccuracies in calorie calculations.
  - Manual input for portion size introduces a source of errors and could be an inconvenience for users

- **Dependency on External APIs**
  - The system relies on external APIs to fetch nutritional information for food.
  - The dependency on third-party APIs could introduce a risk of disruptions to services if there are changes in the API endpoint, which could affect the performance and reliability of the system

- **Limited number of food items**
  - As previously mentioned, although yoloV8 has a list of food items, it is very limited.
  - During the development process, an attempt was made to train YOLOv8 on a custom model. Regrettably, the task gradually became more difficult. If additional time had been allocated, further research and training of the model would have been undertaken.

- **Performance and Scalability Challenges**
  - As the user base and data size increase, there could be issues with performance and scalability.
  - Due to this, it may be necessary to expand the system architecture and database scalability in order to maintain good performance.

- **Resource Constraints**
  - As with any software system, there may be challenges that arise during the development and implementation process.
  - Due to limited development resources, such as time, budget, and expertise, our system may face difficulties in implementing certain features, addressing technical issues, and adapting to users' needs.
  - The project team maintains a continuous dedication to improving the system to meet the ever-changing needs of users.

- **Recognition Limits**
  - When using machine learning models like YoloV8 for food recognition, there may be limitations in accurately identifying certain foods, such as those with a complex or ambiguous appearance.
  - Accuracy can also depend on conditions such as lighting, image quality and the diversity of the food in the dataset.
  - Continued development hopes to improve the accuracy of the food in certain conditions
- **Cultural and Regional Variation**
  - As yoloV8 is trained on a very general and limited range of food such as fruit, vegetables and pizza
  - Therefore, the application may struggle to identify a more diverse range of cuisines, which could lead to decreased user engagement and inaccuracies.
  - This could result in inaccuracies in tracking and recommending diets for users with non-Western eating habits.
  - The user of a more diverse dataset would be a huge benefit in the expansion of the project.
- **User Engagement and Adherence**
  - Over time, there could be challenges in maintaining user engagement.
  - Factors such as user motivation. For example, the user may be highly motivated at the beginning of signing up for the app but decrease over time.
  - Work on the Development of long-term user engagement strategies in order to sustain user interest, motivation, and participation over time.
  - Introduce gamification, rewards, challenges, and incentives to encourage continued user engagement and adherence to healthy lifestyle behaviours.

Chapter 7

Conclusion

The Food Logix app is an innovative application that is designed to help users keep track of their daily food intake and monitor their calorie consumption. The app uses advanced technology to recognize food items present in the images captured by the user. With the help of machine learning, the Food Logix app identifies the food items in the images with high accuracy and provides the user with the necessary information, such as calorie count, nutritional value, and more. The overall goal of this project was to encourage users to monitor their food intake and make informed decisions about their nutrition. Through this conclusion, The context and objectives were revisited. The key findings of the system evaluation were highlighted. Future investigations were also explored.

7.1 Context

As mentioned throughout the project, Food Logix was a food-tracking app that utilized machine learning. This app makes tracking food intake easier, making manual data entry and scanning barcodes a thing of the past. Tracking foods such as a carrot or banana may not have a bar code, which can make tracking tedious and inaccurate. The aim of this project was to fix that. The aim was to provide a hassle-free way for users to obtain the calorie count of their food by simply capturing a picture. This enabled them to store the calorie information for future reference.

7.2 Objectives

The accomplishment of this project depended on its capacity to attain these objectives.

- Identify food using machine learning
- Calculate calories for food items identified
- Real-Time meal tracking
- Display Nutritional Information to users

The Overall objective was to create a system capable of accurately identifying food items, calculating their calories, and delivering real-time feedback to users.

7.3 Findings from System evaluation

The chapter on System Evaluation provides a comprehensive analysis of the project's performance and effectiveness in achieving its objectives. Findings include the following:

- Provided a robust application through the development of manual and automated tests. The system demonstrated a strong resilience against potential errors.
- The evaluations conducted indicated that the response times and throughput were satisfactory, meeting the user expectations for efficiency and responsiveness.
- The project successfully implemented the objectives, including food identification, calorie calculation, real-time meal tracking, and display of nutritional information.
- As with all projects, there are limitations. Some limitations of this project included the limited list of food items in the YOLOv8 machine-learning model and the need for user input for portion size.

7.4 Opportunities for Future Investigation

- As mentioned earlier, In the future, enhanced food recognition could be developed. This could be done by working on training a modal that recognises more food items.
- In order to continue to increase the usability of the app, there could be continued improvements made to the app. Improvements could be made in areas such as meal planning and coaching.

- Potential future work could be the integration of wearable devices, which would greatly increase the functionality and accuracy of the app.
- The implementation of social features such as sharing fitness goals and collecting points could also introduce a gamification aspect of the app

7.5 Impact on Health and Well-being

- Throughout this document, the main goal of the project was to encourage users to adopt healthier eating habits.
- By providing real-time feedback on calorie consumption to users, the app was valuable too for users who aim to eat well and track their food
- The app's ability to accurately recognize a variety of food items, including those without barcodes, removes obstacles to precise tracking.

7.6 Data Privacy and Security

- Security and data privacy are always a top priority for any application.
- Ensuring the security of sensitive user information was of utmost importance for the app. Continual efforts had to be made to maintain its security.
- One way to ensure that sensitive data was secure was by using encryption methods. This involves converting passwords and secret keys into code that is difficult to decipher, thereby enhancing their protection during storage and sessions.

7.7 Conclusion and Future Outlook

To sum up, the project for the food-tracking app had accomplished considerable progress towards achieving its goals, establishing a strong base for future development and innovation. Despite some challenges and limitations, the project was well-positioned for expansion and advancement. By embracing opportunities for further investigation, the system could further enhance its effectiveness, user-friendliness and influence. With a hopeful outlook and a dedication to continuous improvement, the project concludes on a positive note, prepared to embark on the next phase of its journey to encourage healthier dietary habits and enable individuals to make informed decisions about their nutrition.

Bibliography

[1] Fitbit. Say hi to the all-new fitbit app. https://www.fitbit.com/global/us/technology/fitbit-app, 2023.

[2] Vue. The progressive javascript framework. https://vuejs.org/, 2024.

[3] Code Academy. Mvc: Model, view, controller. https://www.codecademy.com/article/mvc, 2024.

[4] Angular. Introduction to the angular docs. https://angular.io/docs#introduction-to-the-angular-docs, 2024.

[5] React. React. https://react.dev/, 2024.

[6] MongoDB. Why use mongodb and when to use it? https://www.mongodb.com/why-use-mongodb, 2024.

[7] AWS. What is sql? https://aws.amazon.com/what-is/sql/#:~:text=You%20can%20use%20SQL%20statements,maintain%20and%20optimize%20database%20performance., 2024.

[8] Wikipedia. Toggle the table of contents web server gateway interface. https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface#:~:text=The%20Web%20Server%20Gateway%20Interface,in%20the%20Python%20programming%20language., 2023.

[9] Flask. Flask. https://flask.palletsprojects.com/en/3.0.x/, 2024.

[10] ML Concepts. C++ vs. python for machine learning: Which language should you choose? https://www.linkedin.com/pulse/c-vs-python-machine-learning-which-language-should-you, 2023.

[11] Google. Advanced guide to inception v3. https://cloud.google.com/tpu/docs/inception-v3-advanced#:~:text=Inception%20v3%20is%20an%20image,multiple%20researchers%20over%20the%20years., 2024.

[12] COCO. What is coco? https://cocodataset.org/#home, 2024.

[13] ultralytics. Object detection. https://docs.ultralytics.com/tasks/detect/, 2024.

[14] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[15] Kaggle. Food-11 image dataset. https://www.kaggle.com/datasets/trolukovich/food11-image-dataset/data, 2020.

[16] Ultralytics. Hardware requirements. https://github.com/ultralytics/ultralytics/issues/4106, 2023.

[17] Google Cloud. Cloud sql. https://cloud.google.com/sql?hl=en, 2024.

[18] Heroku Dev Centre. Preparing a codebase for heroku deployment. https://devcenter.heroku.com/articles/preparing-a-codebase-for-heroku-deployment, 2023.