RecogFood – An Image Processing & Computer Vision Application

Patrick Bondaruk

Student Number - N00201161

Report submitted in partial fulfilment of the requirements for the BSc (Hons) in Creative Computing at the Institute of Art, Design and Technology (IADT).

Declaration of Authorship

The incorporation of material without formal and proper acknowledgement (even with no deliberate intent to cheat) can constitute plagiarism.

If you have received significant help with a solution from one or more colleagues, you should document this in your submitted work and if you have any doubt as to what level of discussion/collaboration is acceptable, you should consult your lecturer or the Programme Chair.

WARNING: Take care when discarding program listings lest they be copied by some-one else, which may well bring you under suspicion. Do not to leave copies of your own files on a hard disk where they can be accessed by others. Be aware that removable media, used to transfer work, may also be removed and/or copied by others if left unattended.

Plagiarism is considered to be an act of fraudulence and an offence against Institute discipline.

Alleged plagiarism will be investigated and dealt with appropriately by the Institute. Please refer to the Institute Handbook for further details of penalties.

The following is an extract from the B.Sc. in Computing (Hons) course handbook. Please read carefully and sign the declaration below

Collusion may be defined as more than one person working on an individual assessment. This would include jointly developed solutions as well as one individual giving a solution to another who then makes some changes and hands it up as their own work.

Declaration

I am aware of the Institute's policy on plagiarism and certify that this thesis is my own work.

Signed: Patrick

Date: 03/05/2024

Failure to complete and submit this form may lead to an investigation into your work.

Abstract

Image recognition is a computer vision related task that involves identifying and categorizing patterns and objects within the digital images and videos. It is also known as object detection or image classification. The purpose of researching how different image recognition models can detect and categorize certain objects and patterns in different images and videos, a decision came to develop a mobile application based on object detection within the food items. The main types of machine learning models that will be researched is the convolutional neural networks (CNN) and the deep learning. The object detection model that will researched is YOLO which is the primary known model that is used for object detection. The purpose of the application is to see different results and accuracies on certain food items that the model detects with the use of an application.

Acknowledgements

I would like to thank and give an appreciation to my supervisor Catherine Noonan for the supervision and feedback of the project, and Cyril Connolly for conducting the initial meetings. I acknowledge the rest of the tutors along with friends and family for offering support and assistance.

Contents

1. Introduction

The technology in this current time of the world has been progressing and advancing at a noticeable increase in speed, especially within the area of Artificial Intelligence. Even around a decade ago, the concept of Artificial Intelligence in certain areas of application was still new and in the process of advancing the AI technology. Nowadays AI is more known and blown in popularity as of recent developments within the use of AI that people hear about. People are more curious about the advancement of AI and leads to think about the further development of AI in the future. Many AI tools are available out there to people like Google Maps which is an AI as a service, different search engines like Google, Firefox, and Internet Explorer, and shopping applications that have a system of recommendations.

This project will specifically go through the AI on machine learning that is based on object detection. Each of the chapters will go through the different stages of the process on developing the project. This first part of the chapter will go through the research section, which will look at the general information about image processing and how it works. Then will look at the certain applications on image processing, research on convolutional neural networks (CNN) and deep learning, and then the research on YOLO object detection model. The next chapter will cover on the requirements analysis of the application, which will research on existing similar applications that are out there and make comparisons. Then will go through the certain technologies and tools to develop the application. Surveys and interviews are conducted to develop personas to target the user demographic.

The feasibility study is carried out to examine specific risks and make a project plan for the development of the application. The next chapter will go through the design of the application with the chosen tools and technologies. The wireframe of the application with the user flow diagram, and style guide with the font and colour palette that was chosen. The next chapter is the implementation or construction which goes through each of the technology component that was implemented and the code structure that was used. Then the next chapter is the testing and analysis, which goes through the actual testing of using the application and trying to use the camera functionality to detect primarily the chosen food items and potentially other objects. Then seeing what the result is and the output.

The discussion chapter will go through the outcome of the application, potential changes and additions on the applications. Then the conclusion chapter will go through the outcome of the overall project and the information learned from it.

Research

The research document is written up to examine the resources, technologies and information that are available to guide in developing a mobile application with a machine learning model. For the research document, the current research on machine learning based on image processing and object detection will be looked at and then conducting it.

1.1. Image Processing

Image processing is the process of taking an image and converting it into a digital formatted image and then applying specific operations to retrieve certain information that would be useful, according to André (2013).

1.1.1. Image processing & How it works

The system that performs image processing, treats the images as two-dimensional signals. The signals being the functions of two variables of x and y coordinates. There are multiple adapted filtering technologies such as object detection, pattern recognition, classification. According to André (2013), when using filtering techniques it uses optical channels that are being applied to pattern recognition. Optical processing has the attractions the moderate cost, the simplicity, and the speed of processing large amounts of information.

According to Jähne (2005), features of the images, there needs to be understanding on how the digital signal is related to it. The sensor for image processing uses an electric signal that is converted from the incident irradiance. The electric signal is then converted into digital numbers that is processed by a computer to retrieve the data.

1.2. Image processing applications in food items

Image processing system for the industry related in food has been recognised for a while (Tillet, 1990). One of the top ten ranked industries that use the techniques on image processing is in the food industry. Gunasekaran (1996).

The common tools in image processing that are used for quality assessment of food products include variety of pre-processing techniques such as contrast enhancement, segmentation and noise removal. These tools help in improving the quality of images, extracting relevant information, and analysing the structure of the food items (Timmermans 1998; Sun, 2004). Additionally, the technology of computer vision utilizes image processing techniques.

Computer vision has been used for evaluating internal and external quality attributes of food items such as sweetness, acidity, size scaling, colour intensity, surface texture and bruises. Furthermore, various studies have implemented image processing and computer vision for tasks such as fruit grading, inspecting the colour of apples and also potatoes, and estimating the weight of fruits. These techniques and tools play a crucial role in providing objective, consistent and quantitative evaluation of food product quality.

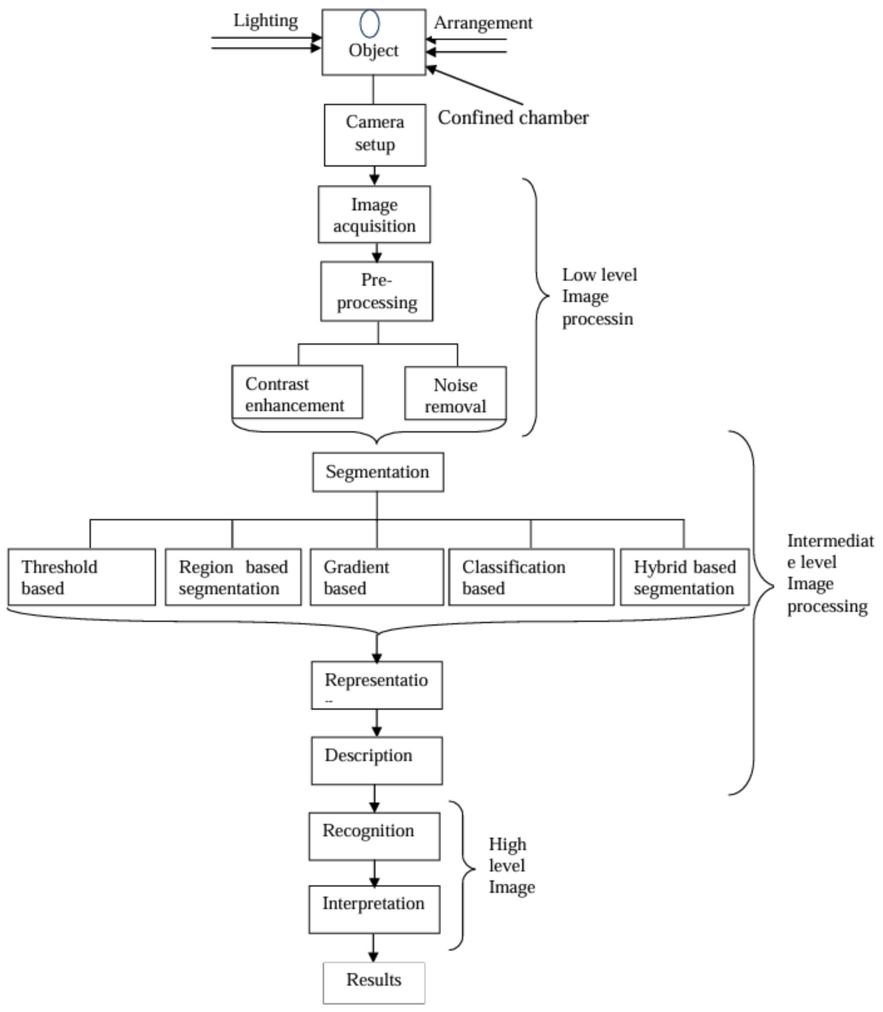


Figure 1.1 - Various techniques of image processing ranging from low, mid and high level (2015 Jakraya Publications Ltd)

1.3. Machine Learning in Image processing

1.3.1. Convolutional Neural Networks (CNN)

Convolutional Neural Network (CNN) is an algorithm of deep learning, and it is commonly put to use for image recognition, object detection, image classification and segmentation tasks. Convolutional Neural Networks have a design to independently learn and to extract different features from the data like images, making them very effective for computer vision tasks. Girish D. et al (n.d.). It starts with the simple layers which is detecting the edges from the image, and then it moves up to more complex layers like detecting the shape from the image.

1.3.1.1. How CNN works

The CNN architecture typically involves different layers like convolutional, pooling, and fully connected. Those layers form the network architecture when they are stacked. The model of a CNN is designed for using tools like MatConvNet, which allows for quick creation of prototype of new CNN architectures and compute efficiently on CPU and GPU to use for training complicated models on large sets of data.

CNNs also include down sampling layers, and that can be used for a variety of tasks such as object detection, scene labelling, and image segmentation.

The CNN has a core set of a convolutional layer. The layer consists of a set of filters (or kernels) that can learn, and with a small receptive field that extend through the full depth of the input volume. During the forward pass, each filter or kernel is convolved across the width and height of the input volume, computing the dot product and producing a 2-dimensional activation map of that filter. This allows the network to learn about the filters and activate when they detect specific features at certain positions in the input. Hossain, Md. A. et al (2019).

1.3.1.2. Learning applied to Image processing

The primary approach for learning that's applied to image processing is deep learning. It is a subset of machine learning that involves training the artificial neural networks to take large amounts of data to learn from.

Sets of algorithms has been applied to diverse application areas in imaging, computer vision and recognition, and speech detection. Few examples of applications include:

Computational Imaging: Develops efficient and interpretable network architectures.

Medical Imaging: Potential on offering to enhance the efficiency in this domain.

Vision and Recognition: Demonstrates its potentiality in developing an efficient and high-performing network architectures for these applications.

These examples of practical applications highlight the potential and versatility of an algorithm to unroll in a variety of domains on image processing, showcasing its influence on real world problems and research areas. Monga, V. et al (2021).

1.3.2. Deep Learning in Image processing

Deep learning in image processing involves the use of deep neural networks to analyse and manipulate images. Deep learning models such as previously mentioned, convolutional neural networks (CNNs), have demonstrated astonishing performance in various tasks for image processing. These tasks include image classification, object detection, image generation, and image segmentation.

Image Classification: The models for deep learning can be trained to classify images into different categories or classes. For example, a deep neural network can be trained to recognize and differentiate between different objects, animals, or scenes within images.

Object Detection: Deep learning models can be used to detect and localize objects within images. This involves identifying the presence of specific objects and drawing bounding boxes around them.

Image Generation: Deep learning models, such as generative adversarial networks (GANs), can generate realistic images based on learned patterns and features. This has applications in creating art, generating synthetic data, and enhancing image quality.

Image Segmentation: Deep learning techniques can segment images into different regions or objects, assigning each pixel to a specific category. This is particularly useful in medical imaging, where it can aid in identifying and analysing specific structures within the human body.

1.4. Object Detection

Object detection is a computer vision task that identifies and locates objects within an image or a video. Object detection goes beyond image classification by not only recognizing the existence of objects but also providing information about the object's spatial locations. Object detection is used in applications in a wide variety such as autonomous vehicles, surveillance cameras, factories, and image retrieval.

1.4.1. Object Detection & How it works

Object detection involves certain amount of steps:

The process starts with an input of an image or a frame from a video.

Then an image is processed to extract relevant features that can help to identify objects. In deep learning-based approaches, this often involves using convolutional neural networks (CNNs) that extracts hierarchical features containing in the image.

Then it aims to identify the presence of objects and locate them within the image.

In addition to locating the objects, the algorithm also classifies the objects into predefined categories or classes. This step involves assigning a label to each object that is detected, indicating what type of object it is.

After the initial detection and classification, post-processing steps may be applied to refine the results, such as non-maximum suppression to merge overlapping bounding boxes.

Object detection can be approached using a variety of techniques, this includes traditional methods of computer vision and approaches of deep learning-based.. Models such as Region-based CNNs (R-CNN), Fast R-CNN, and You Only Look Once (YOLO) has popular architectures for deep learning that are used for object detection.

These models leverage the power of deep learning to simultaneously perform object localization and classification, making them well-suited for real-time applications and scenarios where accurate and efficient object detection is crucial.

1.5. YOLO object detection

You Only Look Once (YOLO) is the algorithm for object detection that reframes the regression problem from the object detection problem, rather than a classification problem. This means that YOLO looks at the entire image only once to detect objects and their positions, making it a single-stage object detector. YOLO divides the input image into a grid and each grid cell is responsible for predicting bounding boxes and class probabilities for the objects present in that cell. Diwan, T. et al (2022).

1.5.1. How YOLO works

The way YOLO (You Only Look Once) works is that it divides the image that is inputted into a grid and then in a single pass it processes the entire image to detect objects and the object's positions. Those grids are the cells, and those cells has the responsibility to detect the objects that fall within it. Then for each cell the YOLO model predicts the bounding boxes that could enclose objects. These bounding boxes have the coordinates of the box's centre, height, width, and the confidence score for the location of an object within the box.

After that it predicts different classes for each bounding box, and the classes would have different probabilities. YOLO estimates the likelihood of each bounding box to contain a variety object classes, for example like bicycle, person, cat, chair.

When the predictions of classes are made, the model uses a non-max suppression technique, that filters low-confidence or duplicate predictions. That way it will retain the detections that is most accurate and confident with. YOLO's overall output is a set of bounding boxes and each of them have a class label and a confidence score that's associated with, which indicates the presence of the objects in the image.

YOLO uses its base network called Darknet53 and it adds 53 additional layers to make it suitable for object detection. It includes heuristics like residual blocks, skip connections and up sampling. YOLO has three different scales that generates feature maps by down-sampling the input at factors of 32, 16, and 8. It then uses (1×1) convolution on these feature maps to detect objects.

One of the essential advantages of YOLO is its potentiality to work on highly probable regions only for object detection, making it efficient and fast. Additionally YOLO's simple architectural design, low complexity, and easy implementation have made it a common choice in production.

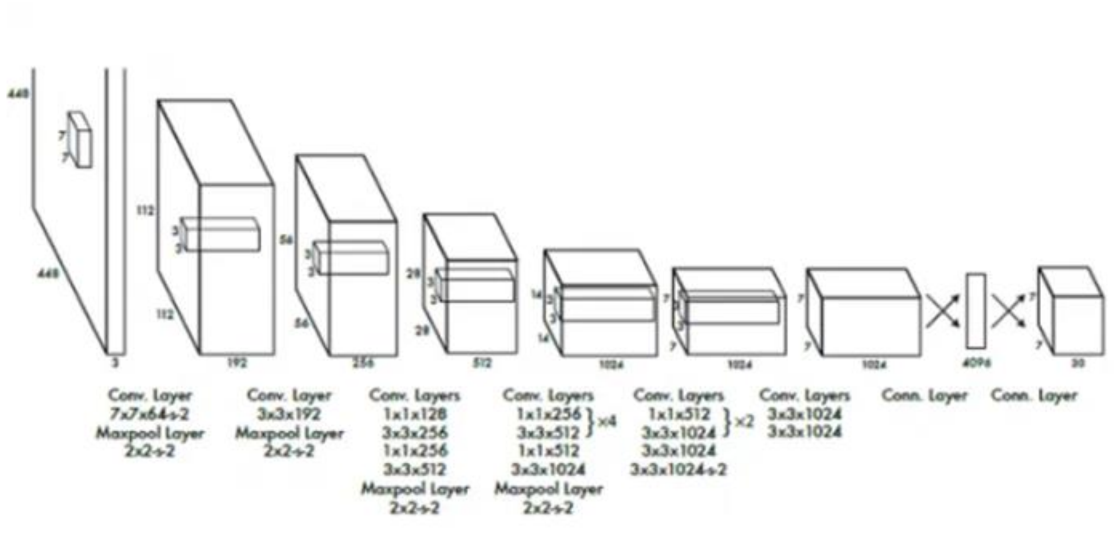

448
448
3
112
112
192
56
56
256
28
28
512
14
14
1024
7
7
1024
1024
4096
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 1.3 – YOLO object detection algorithm by using convolutional layers

1.5.2. Advancements in YOLO

The YOLO algorithm have improved its performance and accuracy at a significant level in detecting certain objects. Here are some of the key developments in the advancements.

YOLO V3 Multi-Scale Detection: A multi-scale detection that was introduced is a YOLO V3, that addresses the issue of detecting targets that were small. It was a limitation in other previous versions like YOLO and YOLO V2. This enhancement empowered YOLO V3 to effectively detect objects with a variety of different sizes from three different scales. Peiyuan Jiang et al. (2022).

Optimizations in YOLO V4: YOLO V4 implemented and sorted various optimizations throughout the entire procedure, and resulted in improved performance. It was able to achieve an increase of 10% in Average Precision (AP) and an increase of 12% in FPS (Frames Per Second) compared to the previous YOLO V3. Additionally, YOLOv4 runs at a twice speed as EfficientDet while maintaining equivalent performance levels. Peiyuan Jiang et al. (2022).

YOLO V5 Flexibility and Performance: YOLO V5 introduced a flexible model architecture, and it can be scaled from between 10+ million parameters to 200+ million parameters. Regardless of this flexibility, even the smaller models of YOLO V5 have demonstrated an impressive performance. The network diagrams as a whole of YOLO V3 to YOLO V5 have remained similar. But the aim on detecting objects of a variety of different sizes from multiple scales, continued to be a priority.

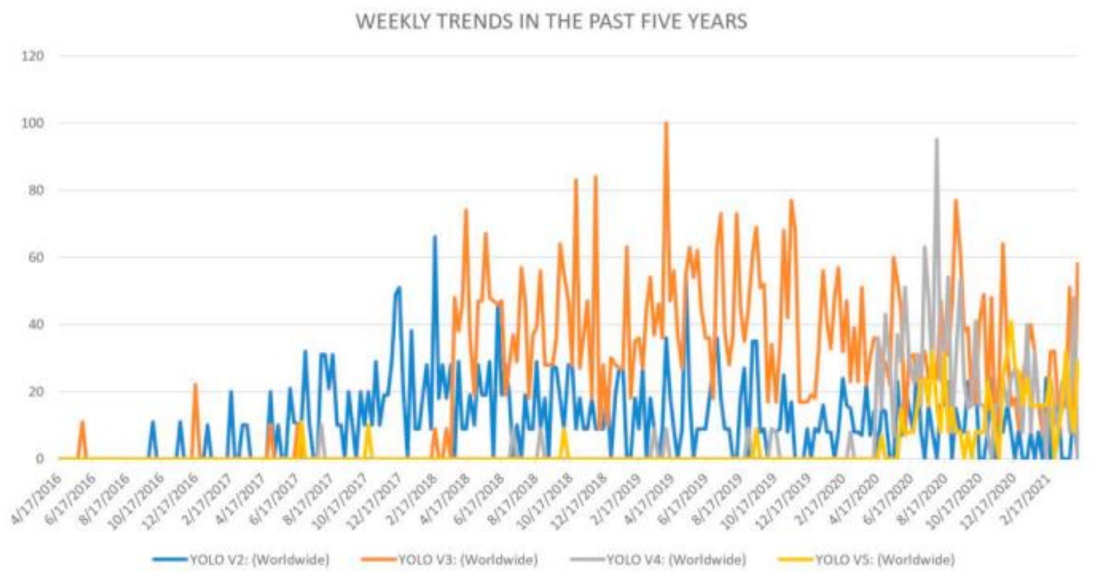


Figure 1.2 – Trends for YOLO versions in the past five years

By incorporating detection of multiple scales, optimizations, flexibility in model architecture, and a commitment towards continuous improvement. The advancements in the YOLO algorithm have led to enhancements at a significant level in performance and accuracy in detecting different objects across various scales and scenarios.

1.5.3. Applications in utilizing the YOLO algorithm

Utilizing the YOLO algorithm in various fields and industries offers a wide range of applications and other implications due to its capabilities in object detection. Here are some of the applications and other implications that are utilized in:

Surveillance & Security: YOLO has a capability of real-time object detection that makes it ideal for surveillance systems. It enables quick and accurate identification of certain objects and individuals in a security footage. It can be applied in public areas, banks, airports, and other areas that are security-sensitive to enhance monitoring and other threat detection. Peiyuan Jiang et al. (2022).

Autonomous Vehicles: YOLO has an ability to detect objects with high speed and accuracy, and it's crucial for autonomous vehicles to navigate safely around environments that can be complex. By utilizing YOLO for object detection, autonomous vehicles can identify and detect pedestrians, road signs, other vehicles and obstacles in real-time. That contributes to improved road safety and efficiency.

Healthcare: YOLO has object detection capabilities that can be applied in healthcare scenarios for multiple purposes. It can be applied to such as medical image analysis, detecting equipment, and patient monitoring. By accurately identifying, detecting and tracking medical tools, anomalies in medical images, or movement from the patients, professionals in healthcare can improve diagnosis accuracy, treatment efficiency, and overall care for patients. Peiyuan Jiang et al. (2022).

Industrial Automation: In the industrial settings, YOLO can be utilized for quality control, detecting defections, and optimization. When accurately detecting and classifying objects on production lines, monitoring conditions of the equipment, and identifying anomalies, manufacturers can streamline operations, reduce any errors, and increase productivity.

The applications of the YOLO algorithm within various industries and fields, have the potentiality to revolutionize processes, efficiency enhancement, and improvement in decision-making by providing accurate and real-time capabilities of object detection. Its implications span across sectors, and offer innovative solutions to challenges that are complex and progress the way for advancements in artificial intelligence and computer vision technologies.

1.6. Summary

Image processing takes an image and converts it into a digital formatted image and applies specific operations to create suitable model, to then retrieve certain information that would be useful. There can be multiple methods used such as Object detection and Image classification which are commonly utilized.

CNNs work by using convolutional layers to learn and detect features in input data like images, down sampling layers to reduce dimensionality, and fully connected layers for classification, making them highly effective for the tasks related to images.

Deep learning in image processing leverages the power of neural networks to make it automatic and improve various aspects of image analysis, manipulation and understanding. The capability of deep learning models to learn and comprehend complex hierarchical representations from raw data, has led to significant advancements in the field of image processing.

YOLO object detection as a regression problem, its efficient use of convolutional neural networks, and its single-stage detection process makes it a powerful and widely used algorithm in the area of computer vision and object detection. Diwan, T. et al (2022).

2. Requirements

2.1. Introduction

The development of the application is based on image processing on objection detection. The use of the application is to select and input images or videos of the food items, it can be as basic as an apple, orange, bread, bottle of soda, potato and onion. Once the photo/video is processed on the application, it detects the contents in the photo/video and outputs a list of food items that it recognizes. With the list of food items that it outputs, it gives nutritional information for each of the food items and gives a list of possible recipes to make with the selected food items.

The purpose of the application is to give the users to take the food items that they don't know the name of or the nutritional value in them, and/or to find the possible recipes that they can make with those food items. This is mainly for the foreign-language people that find the food items in the store that they don't recognise and want to find the necessary information by using this application.

2.2. Requirements Analysis

This section will research the application's requirements analysis. This is the necessity to cover the functionality of the application and the contents within it. The surveys/interviews are carried out to get data on what the users want or desire to have in the application. User profiles/personas are created to get an understanding of the demographic and what type of users would be using the application. Functional and non-functional requirements will be then determined for the application.

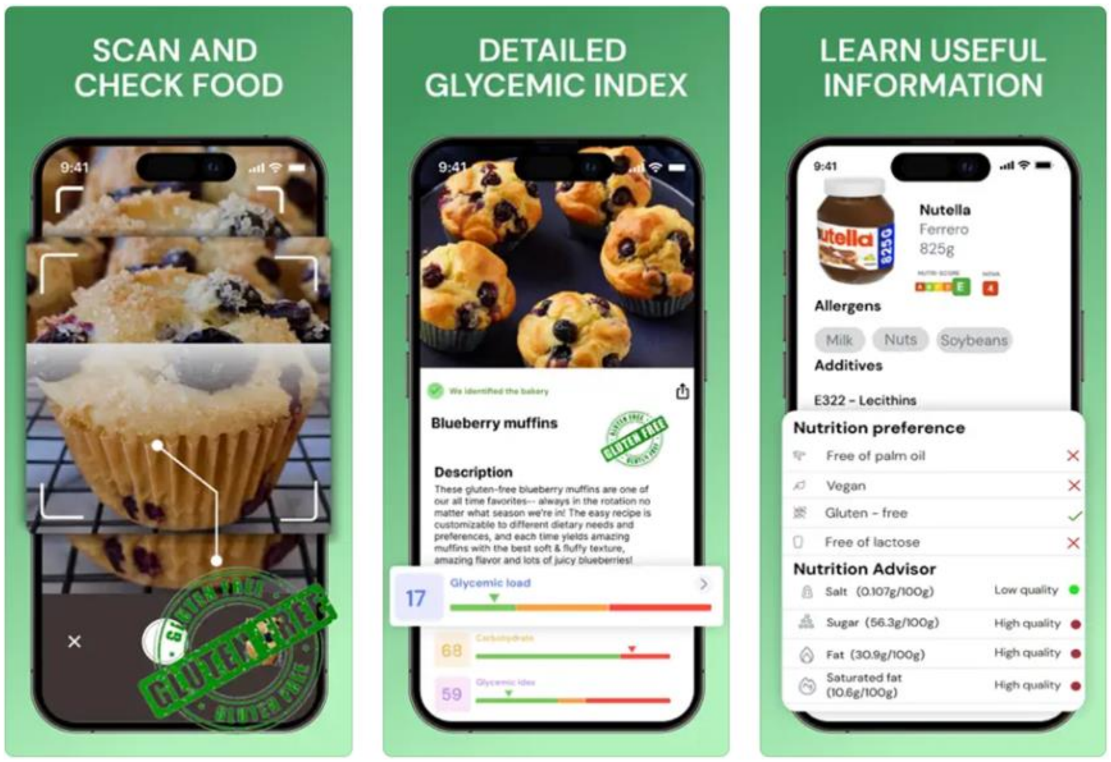
2.2.1. Existing Applications

The research on the existing or similar applications that relate to food products was carried out to get an insight on the current available features on the applications.

A list of mobile apps on food products:

Identify Food - Meal Scanner

This app identifies food products such as vegetables and bread, and it indicates their glycemic index (GI). By scanning the food products, it can tell if fruits and vegetables are low in carbs.



These gluten-free blueberry muffins are one of our all time favorites-- always in the rotation no matter what season we're in! The easy recipe is customizable to different dietary needs and preferences, and each time yields amazing muffins with the best soft & fluffy texture, amazing flavor and lots of juicy blueberries!

Figure 3.1 - Identify Food - Meal Scanner pages 1

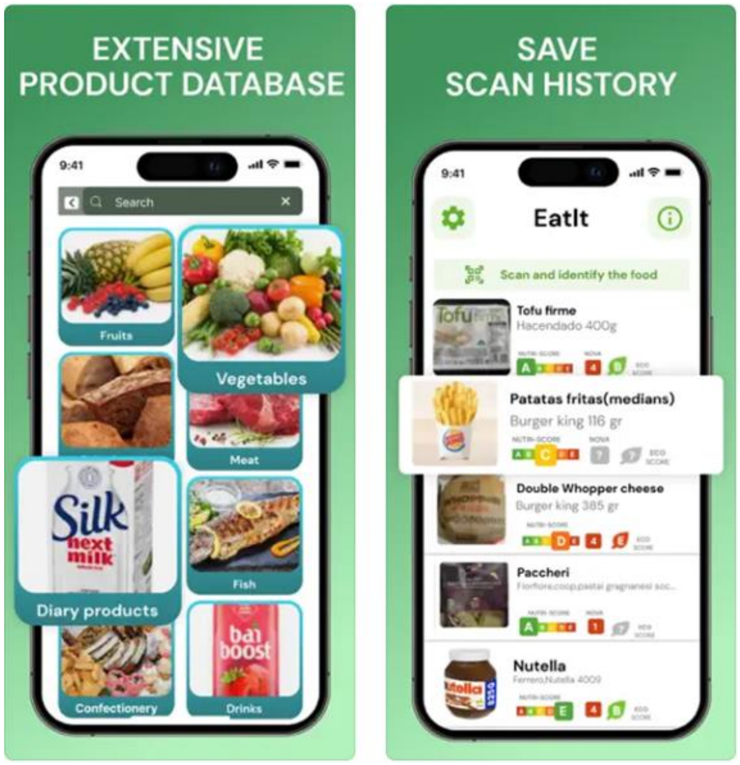
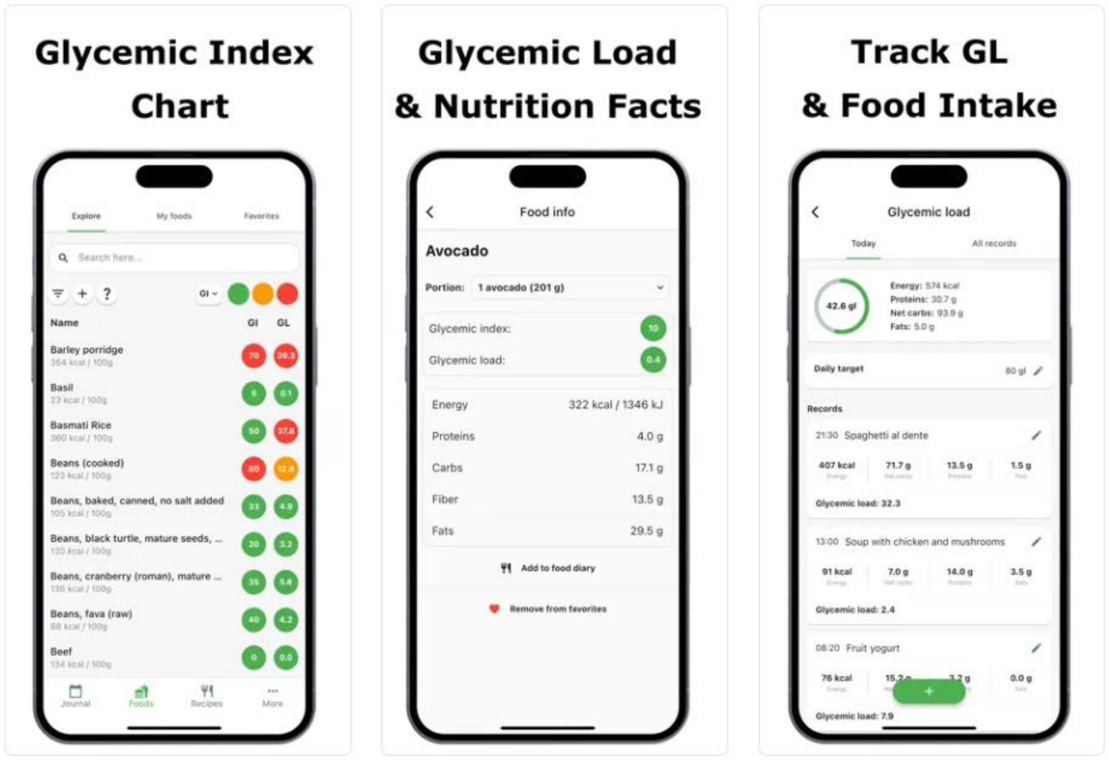


Figure 2.2 - Identify Food - Meal Scanner pages 2

Glycemic Index & Load Tracker

This app features a comprehensive glycemic index & glycemic load chart. It provides info to see which foods are low-GI and high-GI, allowing the user to make informed choices.



Name	GI	GL
Barley porridge 354 kcal / 100g	70	39.3
Basil 23 kcal / 100g	5	0.1
Basmati Rice 360 kcal / 100g	50	37.8
Beans (cooked) 123 kcal / 100g	80	12.8
Beans, baked, canned, no salt added 105 kcal / 100g	33	4.9
Beans, black turtle, mature seeds, ... 130 kcal / 100g	20	3.2
Beans, cranberry (roman), mature ... 136 kcal / 100g	35	5.6
Beans, fava (raw) 88 kcal / 100g	40	4.2
Beef 134 kcal / 100g	0	0.0

Avocado

Portion: 1 avocado (201 g)

Glycemic index: 10

Glycemic load: 0.4

Energy	322 kcal / 1346 kJ
Proteins	4.0 g
Carbs	17.1 g
Fiber	13.5 g
Fats	29.5 g

42.6 gl

Energy: 574 kcal
Proteins: 30.7 g
Net carbs: 93.9 g
Fats: 5.0 g

Daily target 80 gl

Records

21:30 Spaghetti al dente
407 kcal 71.7 g 13.5 g 1.5 g
Glycemic load: 32.3

13:00 Soup with chicken and mushrooms
91 kcal 7.0 g 14.0 g 3.5 g
Glycemic load: 2.4

08:20 Fruit yogurt
76 kcal 15.2 g 3.2 g 0.0 g
Glycemic load: 7.9

Figure 3.3 - Glycemic Index & Load Tracker pages 1

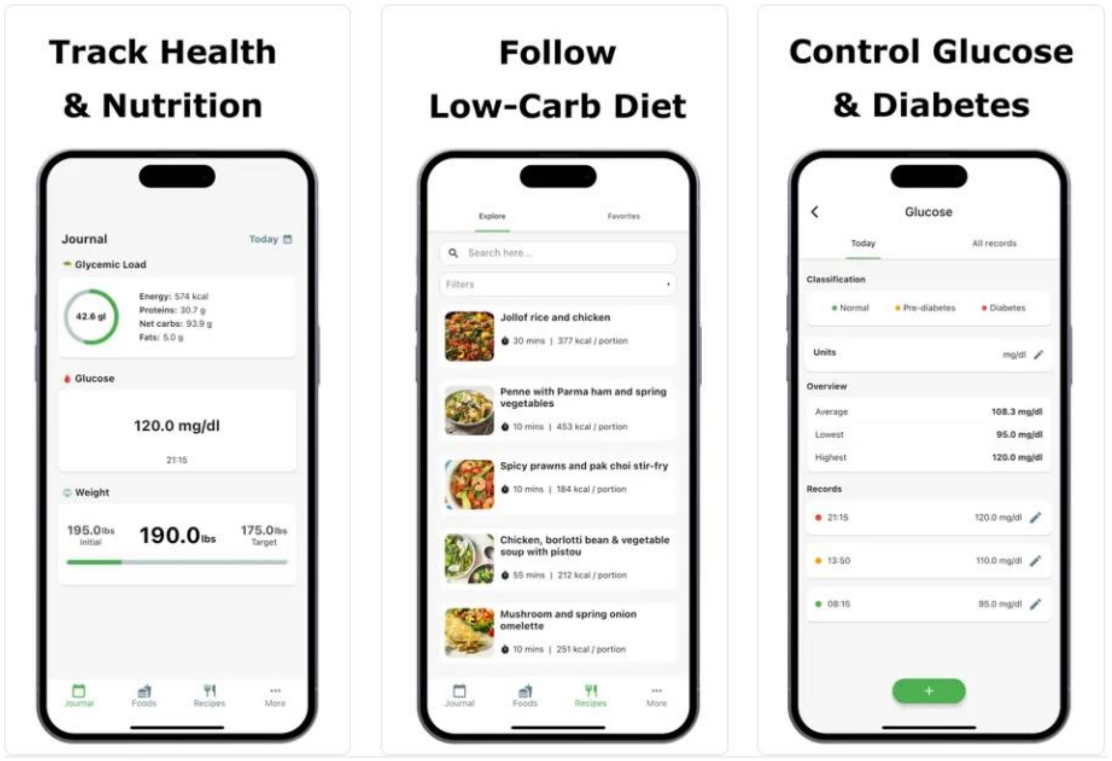
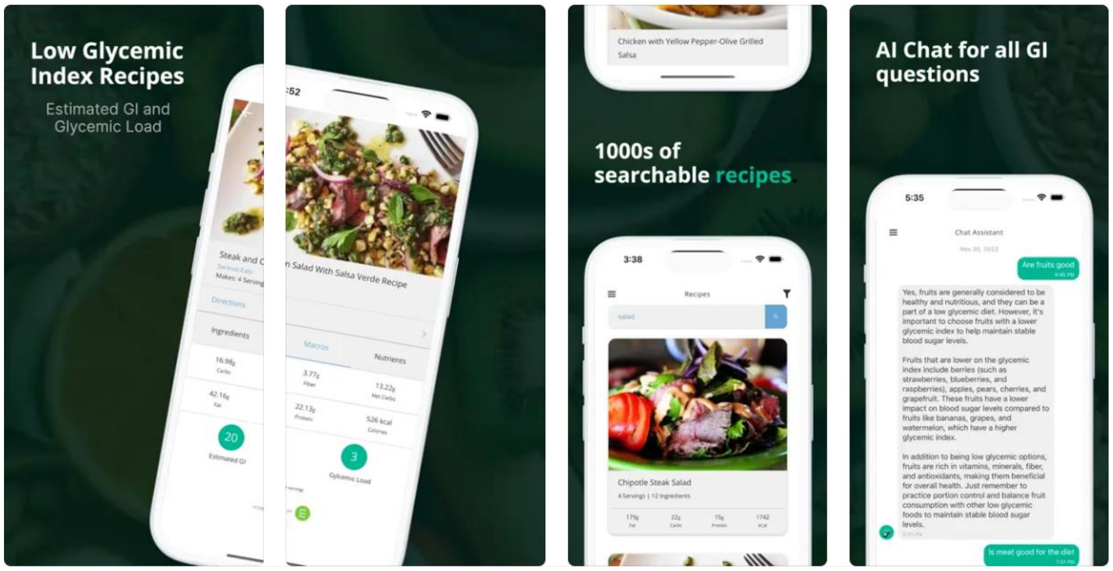


Figure 3.4 - Glycemic Index & Load Tracker pages 2

Glycemic Index & Load Recipes

An AI chat tool that provides food recipes the nutritional value of the foods.


Low Glycemic Index Recipes
Estimated GI and Glycemic Load
Steak and
Directions
Ingredients
20
Estimated GI
n Salad With Salsa Verde Recipe
Macros
Nutrients
3
Glycemic Load
Chicken with Yellow Pepper-Olive Grilled Salsa
1000s of searchable recipes.
3:38
Recipes
salad
Chipotle Steak Salad
AI Chat for all GI questions
5:35
Chat Assistant

Are fruits good

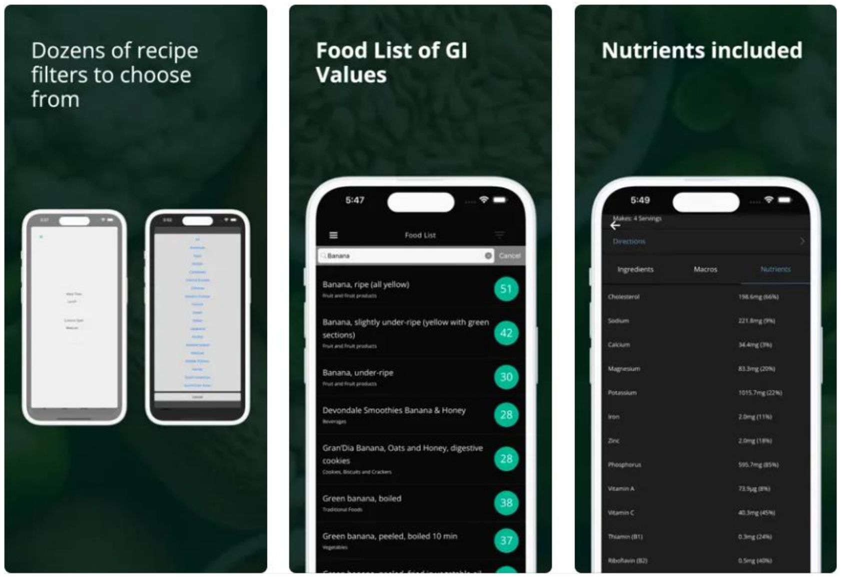
Yes, fruits are generally considered to be healthy and nutritious, and they can be a part of a low glycemic diet. However, it's important to choose fruits with a lower glycemic index to help maintain stable blood sugar levels.

Fruits that are lower on the glycemic index include berries (such as strawberries, blueberries, and raspberries), apples, pears, cherries, and grapefruit. These fruits have a lower impact on blood sugar levels compared to fruits like bananas, grapes, and watermelon, which have a higher glycemic index.

In addition to being low glycemic options, fruits are rich in vitamins, minerals, fiber, and antioxidants, making them beneficial for overall health. Just remember to practice portion control and balance fruit consumption with other low glycemic foods to maintain stable blood sugar levels.

Is meat good for the diet

Figure 3.5 - Glycemic Index & Load Recipes pages 1



Banana, ripe (all yellow) 51
Fruit and Fruit products
Banana, slightly under-ripe (yellow with green sections) 42
Fruit and Fruit products
Banana, under-ripe 30
Fruit and Fruit products
Devondale Smoothies Banana & Honey 28
Beverages
Gran'Dia Banana, Oats and Honey, digestive cookies 28
Cookies, Biscuits and Crackers
Green banana, boiled 38
Traditional Foods
Green banana, peeled, boiled 10 min 37
Vegetables

Cholesterol	198.6mg (66%)
Sodium	221.8mg (9%)
Calcium	34.4mg (3%)
Magnesium	83.3mg (20%)
Potassium	1015.7mg (22%)
Iron	2.0mg (11%)
Zinc	2.0mg (18%)
Phosphorus	595.7mg (85%)
Vitamin A	73.9µg (8%)
Vitamin C	40.3mg (45%)
Thiamin (B1)	0.3mg (24%)
Riboflavin (B2)	0.5mg (40%)

Figure 3.6 - Glycemic Index & Load Recipes pages 2

Yuka - Food & Cosmetic scanner

An app that allows the user to scan the barcodes of food products and personal care products to show the impact on the person's health. A rating and detailed information is provided to the user to understand the analysis of each product.

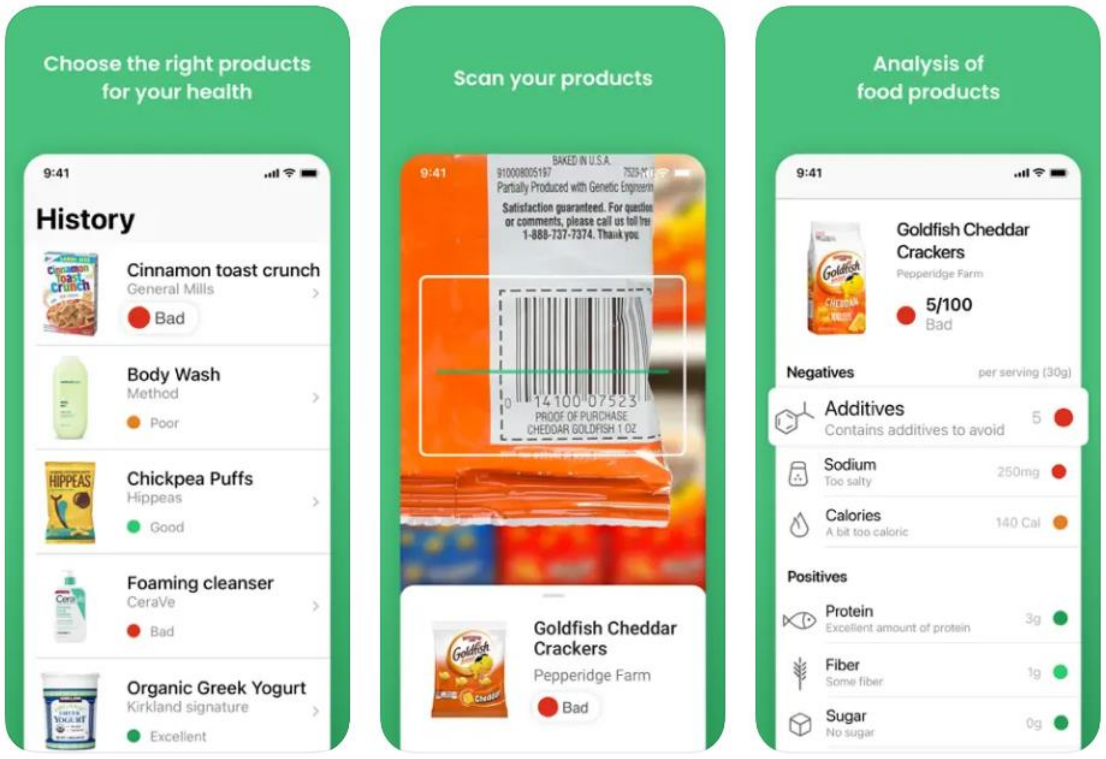


Figure 3.7 - Yuka - Food & Cosmetic scanner pages 1

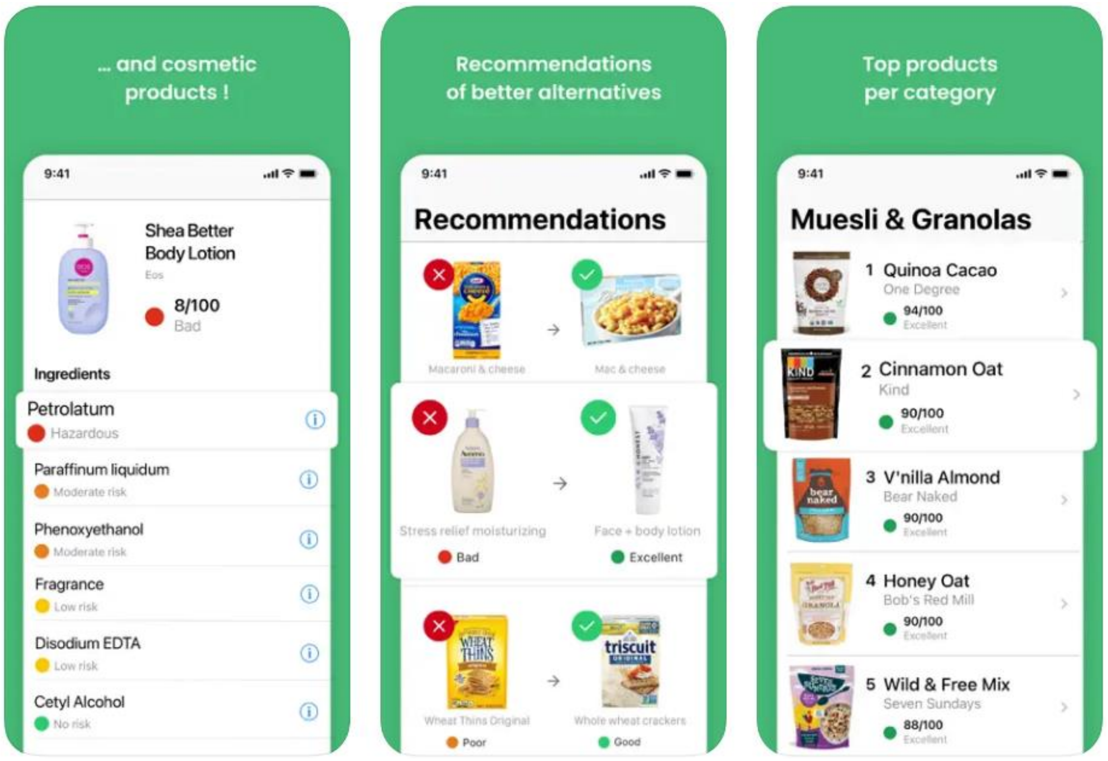


Figure 3.8 - Yuka - Food & Cosmetic scanner pages 2

Trash Panda Food Scanner

An app that scans barcode that provides info to see if the food products have potentially harmful ingredients. It can display if the product is gluten-free, dairy-free, low sugar, organic, plant-based, vegan or vegetarian.

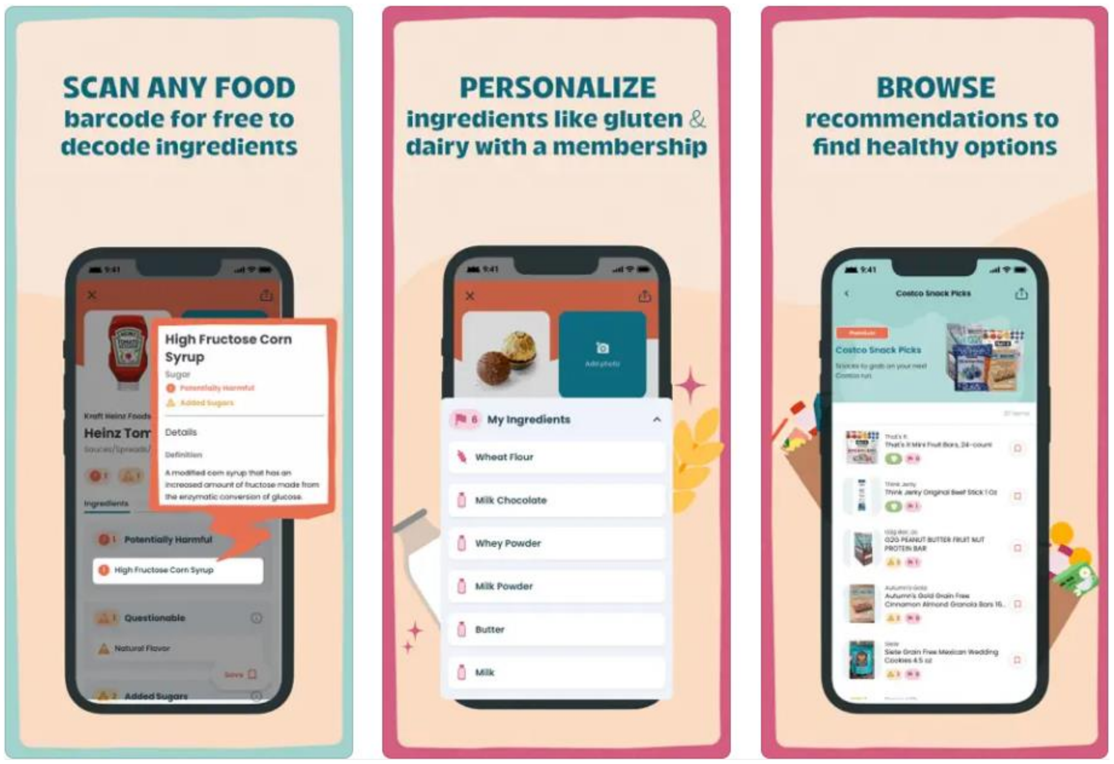


Figure 3.9 - Trash Panda Food Scanner pages 1

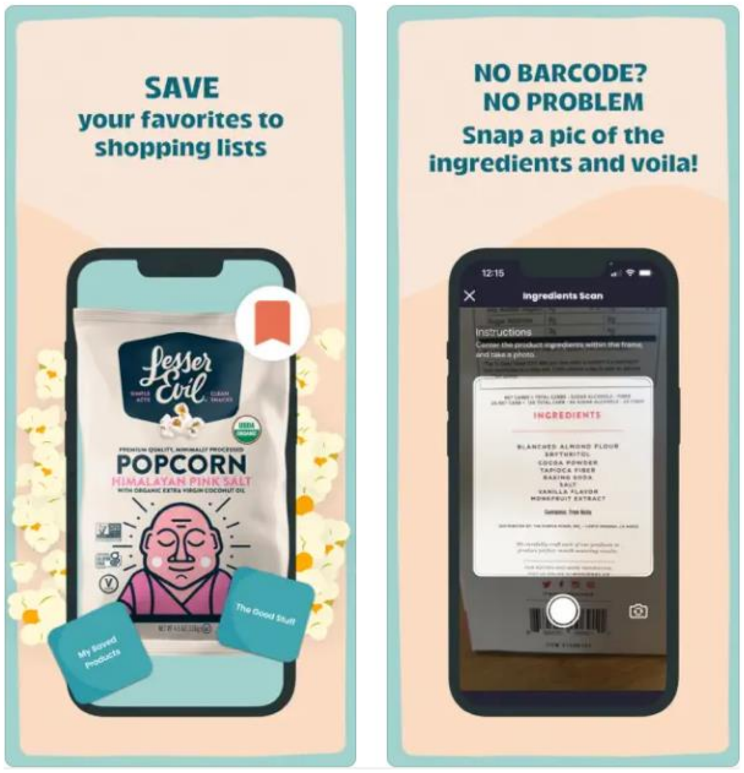


Figure 3.10 - Trash Panda Food Scanner pages 2

A comparison was made between the two apps by their relativity and the number of users. Taking the existing applications and making a comparison study helps to determine the functionality that the apps provide and the features that may possibly be included in the development of the application.

Application	**Identify Food - Meal Scanner**	**Yuka - Food & Cosmetic scanner**
Platform	• IOS	• IOS • Android
Cost	Installation and need to be a subscriber to use the features, $4.99 for a week	Install and sign-up for free, premium subscription from $9.99 - $19.99 per year

Description	This app identifies food products such as vegetables and bread, and it indicates their glycemic index (GI). By scanning the food products, it can tell if fruits and vegetables are low in carbs. The app counts the carbs to keep track of calorie intake.	An app that allows the user to scan the barcodes of food products and personal care products to show the impact on the person’s health. A rating and detailed information is provided to the user to understand the analysis of each product. The app can provide similar products for healthier alternatives.
Technology	• Camera scanning food products • Image recognition algorithms	• Camera scanning barcodes • Search bar
Advantages	• Available in many languages. • Intuitive interface • Keeps track of carbs • Indicates the GI-index of the products.	• Free to register. • Shows different impacts on the products.
Disadvantages	• A subscription is paid to use the features. • Not available on Android devices. • Fairly expensive	• A subscription is paid to use the search bar, offline mode and unlimited history.

3.3. User Profile

3.3.1. Interviews

By conducting the interviews from 3-4 users that tested the application, feedbacks were compiled. The first question was ‘from 1 to 10 how easy was it to use the application’. By compiling the answers, the average score was 9.5 and the users said that it was very easy to navigate, really intuitive and no the application functioned as intended. The other question was if ‘there’s anything that they would like to change’, and the majority said not particularly but the additions like more food coverage and refreshing would be a bonus. The other question was if ‘there’s any features they would like in a food related app’, and the compiled answers were to have a camera focus control and a grocery list would be handy, and to have a breakdown of nutrients from the selected food items if applicable.

3.3.2. Survey

A survey is carried out to obtain an understanding of a demographic of potential users. By taking and combining the results from the survey, personas can be created to support the project’s progression.

All the participants of the survey are aged between 22-25 years old, along with all of them being male. Majority of the participants were students with a good portion of them employed. 80% of the participants have an android operating system on their smartphone device, with the other 20% of the participants having a redmi or other alternative operating system. By compiling the results, up to 60% of participants haven’t used or heard of food related apps like calorie counter or searching recipes. While other 40% of the participants have either used or heard of familiar food related apps.

The selected food related apps that were researched are listed in the survey for the participants to select the applications that they’ve used or heard of before. Upon the inspection of the results, 40% of the participants have heard of the listed applications which were ‘Yuka – Food & Cosmetic Scanner’ and ‘Glycemic Index & Load Recipes’. While 60% of the participants haven’t used or heard of the listed applications.

60% of the participants answered that they would give try using those or any other familiar apps, while the other 40% answered that they wouldn’t use those apps. For those that have tried using those or other familiar apps, 75% of them wouldn’t use those apps again, and the other 25% of them would use them again. Majority of the participants answered between not likely and unlikely that they would use a smartphone application to get food recipes or track calorie intake. The other portion of participants answered between likely and more likely that they would use a such smartphone application.

All the visual graphs of the results are listed below.

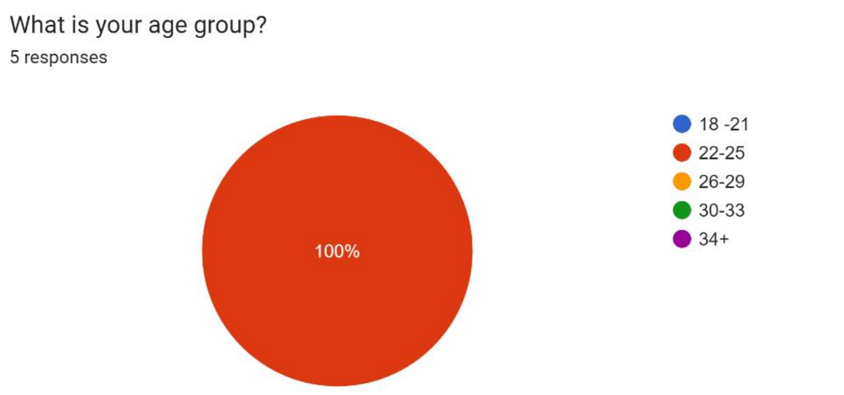


Figure 3.11 - Survey result 1

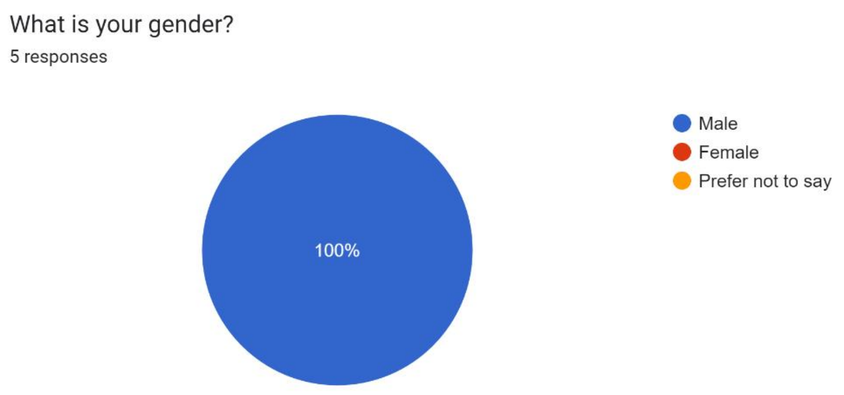


Figure 3.12 - Survey result 2

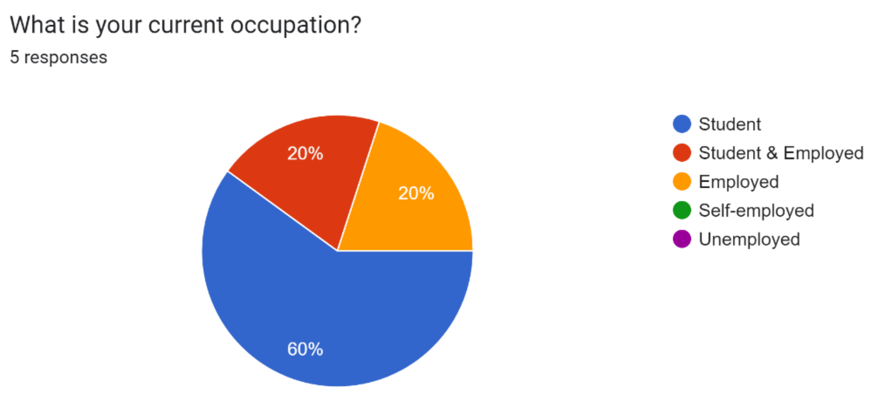


Figure 3.13 - Survey result 3

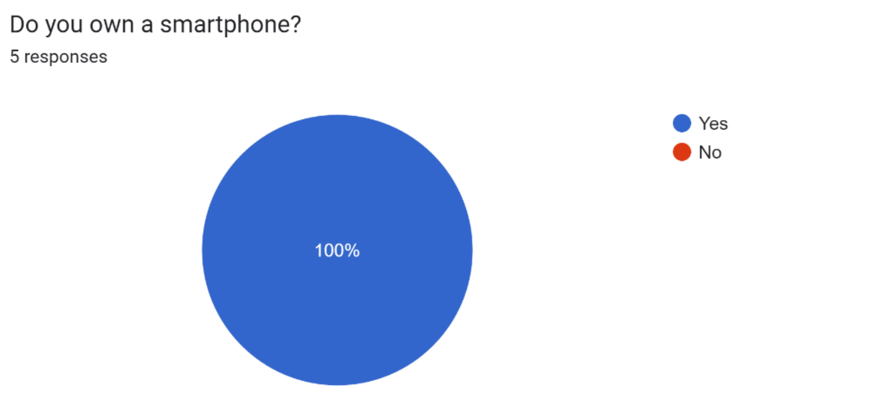
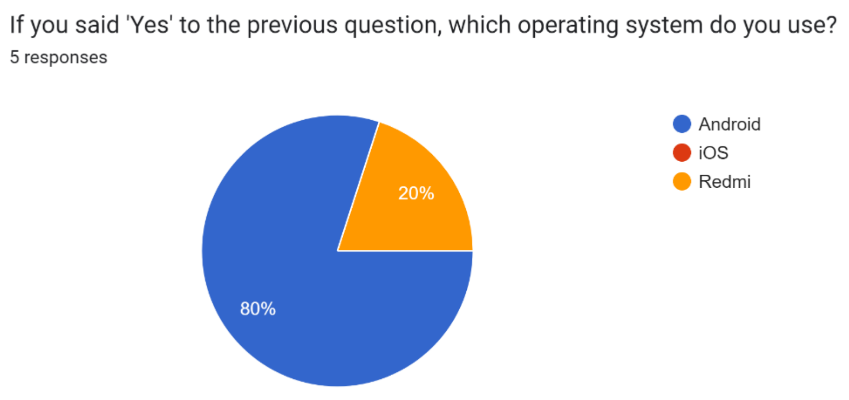
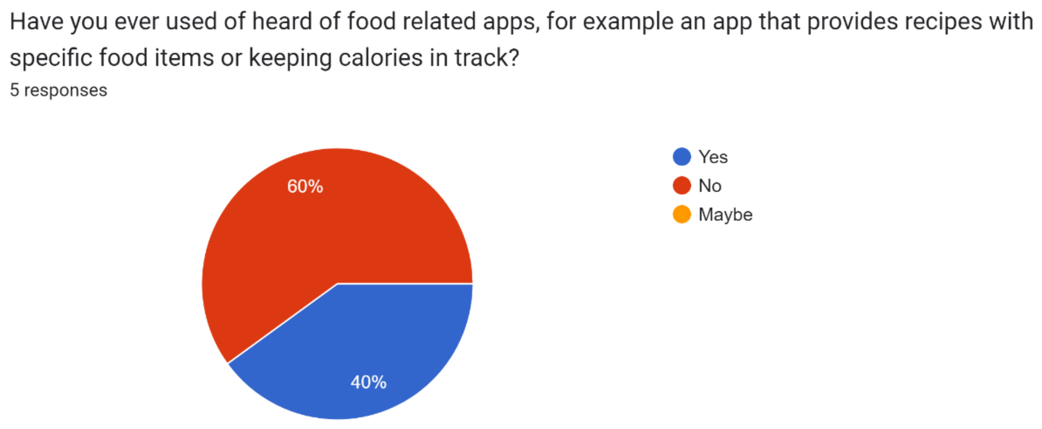


Figure 3.14 - Survey result 4


If you said 'Yes' to the previous question, which operating system do you use?

5 responses

Figure 3.15 - Survey result 5


Have you ever used of heard of food related apps, for example an app that provides recipes with specific food items or keeping calories in track?

5 responses

Figure 3.16 - Survey result 6

Here are a list of food related applications, select the ones that you've used or heard of before.
5 responses

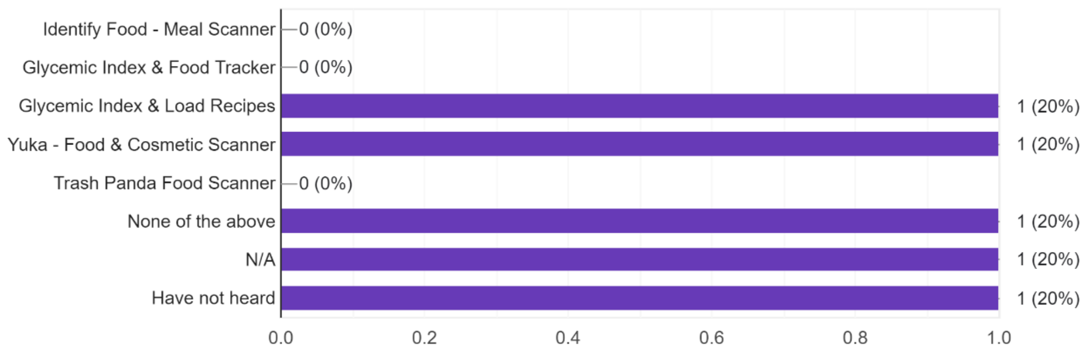


Figure 3.17 - Survey result 7

If you haven't used those or any familiar apps before, would you ever try using them?
5 responses

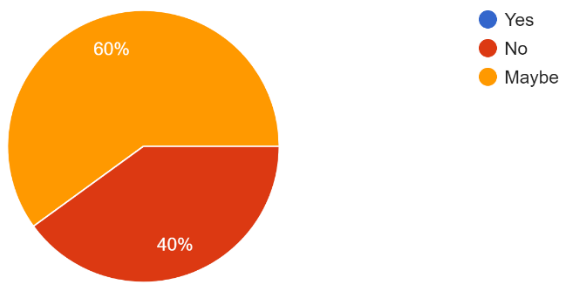


Figure 3.18 - Survey result 8

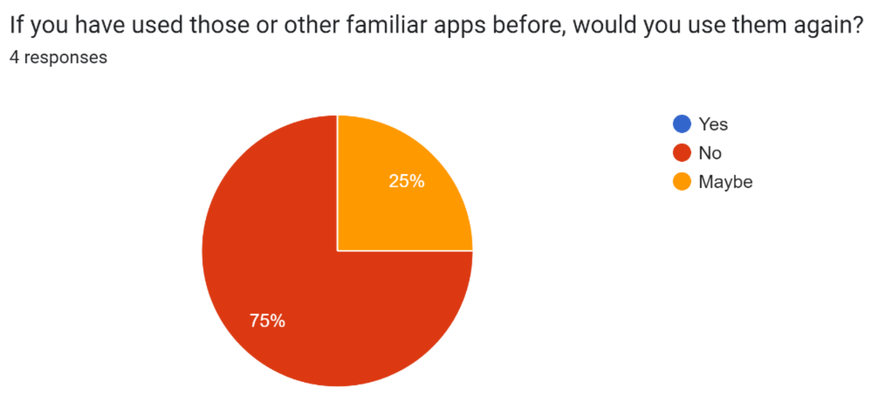


Figure 3.19 - Survey result 9

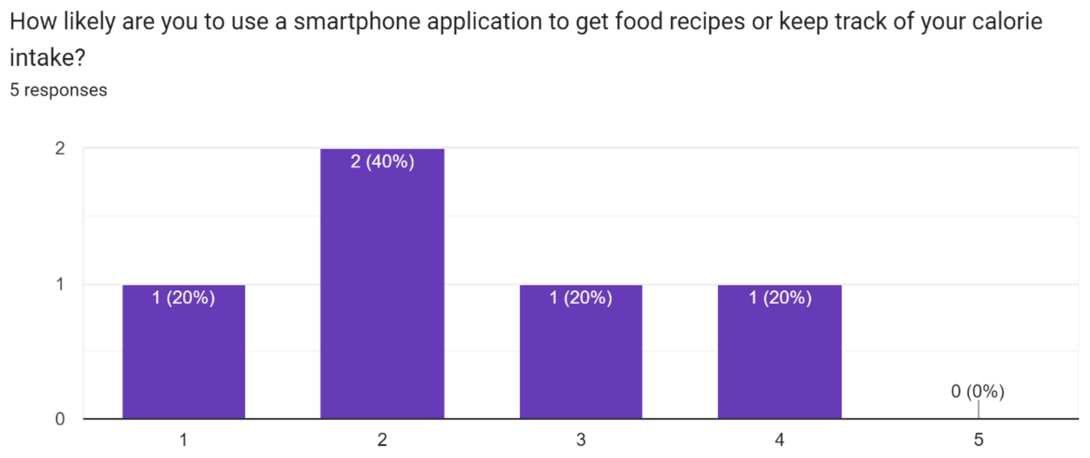
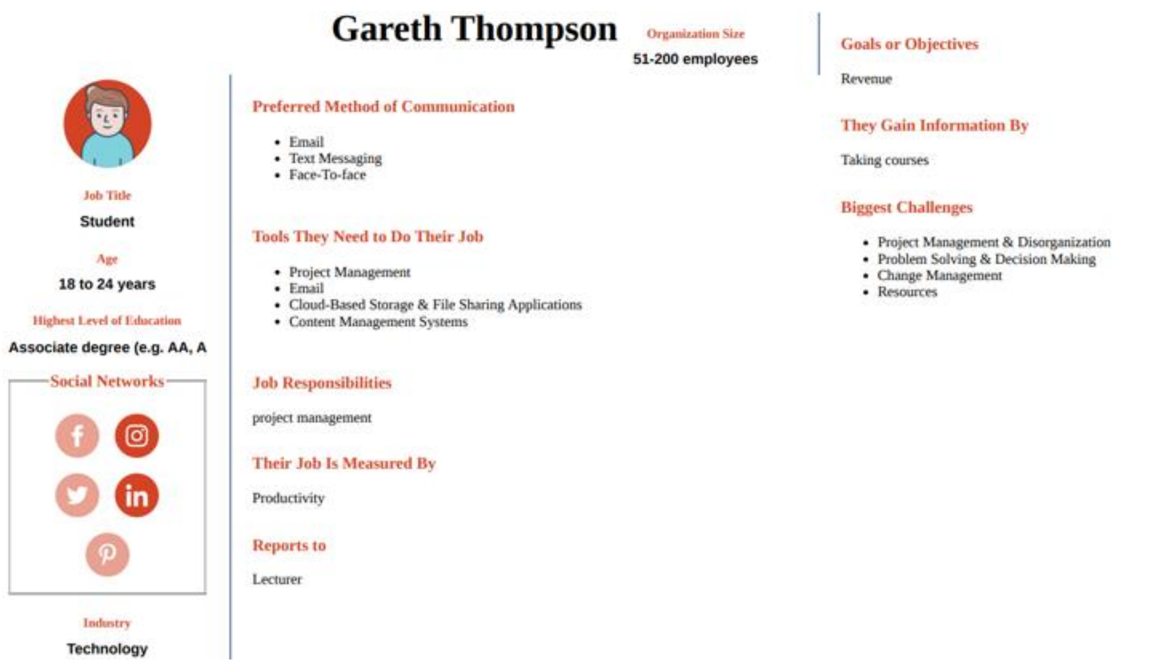


Figure 3.20 - Survey result 10

3.3.3. Personas

Taking the results from the survey that was carried out, personas are created. Personas are fictional characters that are developed from the user research. It can visualise different types of users and give a demographic for the application. By creating the personas it can give a better understanding of the users’ needs and helps the project move in the right direction.



Gareth Thompson

Organization Size

51-200 employees

Job Title

Student

Age

18 to 24 years

Highest Level of Education

Associate degree (e.g. AA, A

Social Networks

Industry

Technology

Preferred Method of Communication

- Email
- Text Messaging
- Face-To-face

Tools They Need to Do Their Job

- Project Management
- Email
- Cloud-Based Storage & File Sharing Applications
- Content Management Systems

Job Responsibilities

project management

Their Job Is Measured By

Productivity

Reports to

Lecturer

Goals or Objectives

Revenue

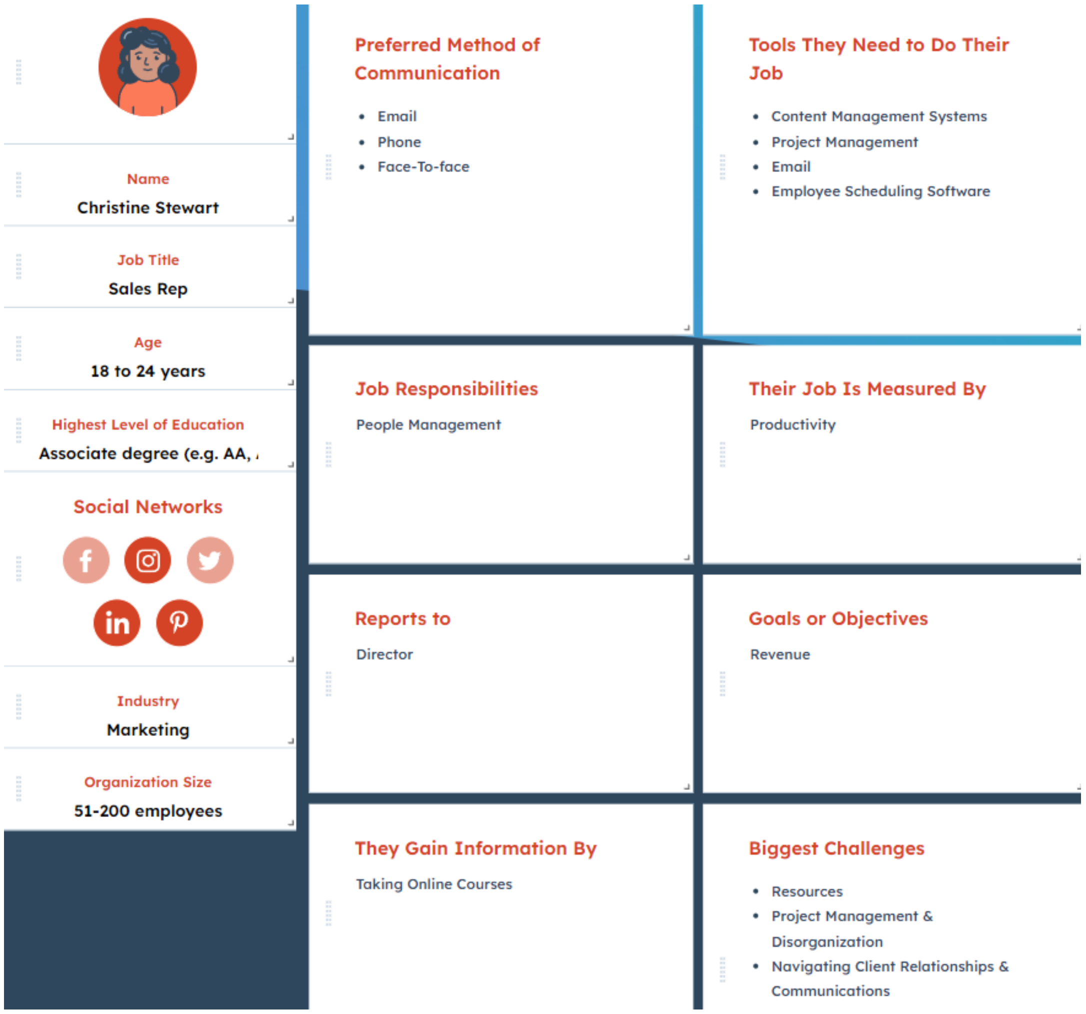
They Gain Information By

Taking courses

Biggest Challenges

- Project Management & Disorganization
- Problem Solving & Decision Making
- Change Management
- Resources

Figure 3.21 - Persona 1



Name
Christine Stewart

Job Title
Sales Rep

Age
18 to 24 years

Highest Level of Education
Associate degree (e.g. AA,

Social Networks

Industry
Marketing

Organization Size
51-200 employees

Preferred Method of Communication
- Email
- Phone
- Face-To-face

Tools They Need to Do Their Job
- Content Management Systems
- Project Management
- Email
- Employee Scheduling Software

Job Responsibilities
People Management

Their Job Is Measured By
Productivity

Reports to
Director

Goals or Objectives
Revenue

They Gain Information By
Taking Online Courses

Biggest Challenges
- Resources
- Project Management & Disorganization
- Navigating Client Relationships & Communications

Figure 3.22 - Persona 2

The personas that were developed, Gareth and Christine want to make a routine of preparing their own meals and track their calorie intake. They want to use an app that would help them to know the calorie and nutritional value for each food item that they intake, and potentially know the meals that they can prepare with specific food items. That way it would make their time efficient and help them to keep track of the food eaten.

3.4. Requirement Modelling

3.4.1. Functional Requirements

Outputting names of the food products	The application should then output the names of the products that it recognises.

Displaying nutritional value and recipes	When the names of the food products is listed, it should also display the nutritional value and possible recipes from those products.
Login Authentication	The user should be able to create an account, and authenticate whenever the user logs in.
Using the camera on the application	The user should have an option to open and use the camera within the application to take images/videos of the current food products in front of them.
Searching recipes with a search bar	A user can search general recipes within the search bar.
Storing Images & Videos	An application can save and store images/videos that were used previously.

3.4.2. Non-functional Requirements

Security	An application can keep the data secure to the user.
Storing history	The application can save and store previous results when using the app.

3.5. System Model and System Requirements

The application will have a set of components that will be used to develop an application.

These are the list of components:

Android Studio	This is the standard IDE for coding Java and Kotlin to develop mobile applications.
TFLite	This will be a TensorFlow machine learning model in a mobile format.

AWS	A back-end system to store the data of images/videos.

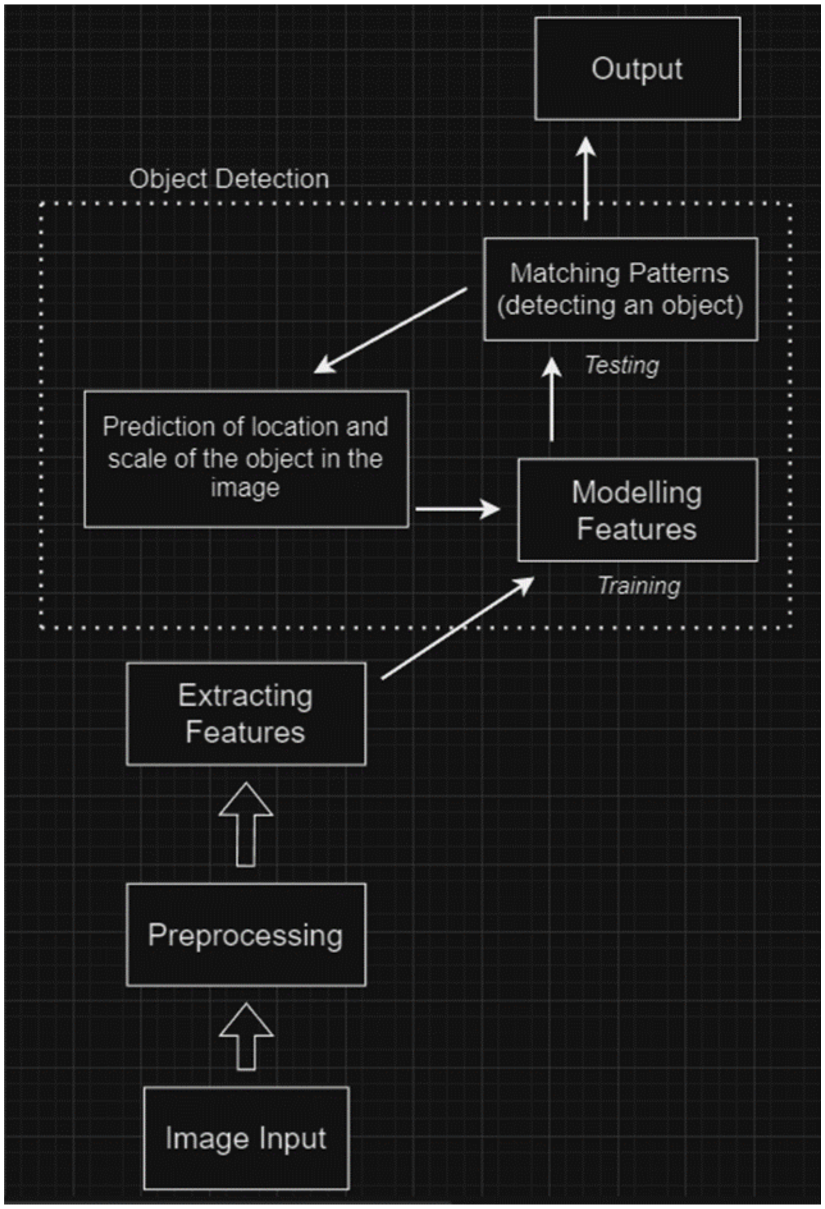

Output
Object Detection
Matching Patterns (detecting an object)
Testing
Prediction of location and scale of the object in the image
Modelling Features
Training
Extracting Features
Preprocessing
Image Input

Figure 3.23 - System Model

3.6. Feasibility Study

During the development of this project, the feasibility study will examine the risks that will possibly be encountered. This project is an Android Studio Java application built on a mobile device. The primary risk is that Android Studio is a technology that wasn't taught in any of the modules, so it would have to be self-taught in the process. However the fundamentals of Java programming were taught in one of the modules so the self-teaching process would be slightly easier. There are different tutorials available to guide the learning process.

The machine learning model will be coded with Python, which is a language that was taught in two modules of 'Data Analytics' and 'Artificial Intelligence', which gives an advantage. Additionally the IDE Jupyter Notebook and Google Colab Notebook were taught within those modules, and Google Colab Notebook is used to build the machine learning model. This gives a low risk since the student had experience with those tools and technologies. The only risk is using different resources and tools in Python that were not taught, so the student will have to self-learn those resources and tools with available tutorials to guide the learning.

The student will take time to self-teach those technologies and tools, and it comes to time management which is the main risk. If the time is well managed, the application with minimum functional requirements can be developed, which is the primary objective of this project.

3.7. Project Plan

The project plan will use an agile methodology, and iterations will be made with a set of goals. That way the project can be planned and time managed, when progressing with each step. Trello board will be used to make an iterative process for the project by managing tasks and setting goals. A set of tables will be created when completing certain tasks, and the primary tables that would be used are 'To Do', 'Doing' and 'Done', along with any additional tables like 'Questions' and 'Resources'. The tasks for each of the tables will be colour coded to effectively visualise it, so the 'To Do' tasks would be in orange, the 'Doing' tasks would be in yellow, and the 'Done' tasks would be in green. There can be due dates set for certain tasks to be completed, and if there are any tasks that are overdue they can be colour coded in red. That way it can be managed, and maintain the project's workflow.

3.8. Test Plan

A test plan will be created to the application's quality. This test plan will help to identify and highlight any gaps, flaws or functionalities that the application may need.

3.8.1. Unit Testing

Unit testing will be used to test separate pieces of code to ensure these codes perform as intended. The code will be separated into smallest piece that can be individually tested. Any changes that will be made to the code, the unit testing will be carried out.

3.8.2. Integration Testing

This is a software testing technique to test a group of components or individual software modules together. That way it will ensure that this group testing will work together as expected.

3.8.3. System Testing

When the application is developed, system testing is then carried out. This test determines whether the system meets what is required from it. By meeting the requirements is to have the system to succeed this test, and if the system fails this test it must be changed and adjusted.

3.8.4. User Testing

The testers will be given a task to complete on the application, and during the test the testers will be monitored. This ensures the applications' ease of use for the users, and the user feedback from these tests will be taken note of that will help the application to improve in user experience.

4. Design

4.3. Introduction

This chapter will describe the design, architecture and user interface of an image recognition application on food items that is developed. This system will be developed into a mobile application along with the mobile emulator to demonstrate the functionality of the system.

The application for this project is an image recognition model that processes and recognizes certain food items from the images that are inputted.

4.4. Program Design

It will be broken down into the image recognition model which is the fundamental part of the application. The architecture system of the application, along with the permissions of the use of camera on the device and an AI system.

4.4.1. Technologies

The technologies being used to create this application are:

o Java (Android Studio)

o Python

o TensorFlow

These technologies were chosen with the following reasons.

Java:

- **Platform Independence:** It means that any device that has a JVM (Java Virtual Machine), it can run a Java written code. Meaning that it allows the developers to deploy the code on multiple platforms when they have written the code once.

- **Android Studio IDE:** The official IDE (Integrated Development Environment) for Android development is Android Studio, which uses Java and Kotlin language. This IDE offers powerful tools and features created specifically for Android app development. It provides an intuitive interface, along with debugging tools, and seamless integration with the Android SDK.

- **Performance:** Java has good performance when it's used with the Android SDK. So the Java code is being optimised in Android Studio, which ensures that applications run smooth and efficient on Android devices.
- **Security:** Developers can write secure code since Java provides built-in security features. Android Studio provides tools for implementing security such as network communication, encrypting data, and user authentication.
- **Compatibility:** Java is natively supported by Android devices, making it a preferred language for developing Android apps. It insures that the applications that are developed are compatible with a wide range of Android devices, like smartphones, tablets, and wearables like smartwatches.

Python:

- **Ease of Use and Learning:** Python is known as one of the beginner-friendly languages for programming. It has a readable syntax that makes it easy to understand and learning it.
- **Libraries:** Python provides a rich ecosystem of libraries and frameworks that are specifically made and designed for data science and machine learning. The popular libraries include Keras, PyTorch, scikit-learn, and TensorFlow which was the chosen library for the machine learning model. The libraries provide pre-built algorithms and functions for tasks such as training models, evaluating, and preprocessing data. They significantly speed up the time for development.
- **Versatility:** Python is a versatile programming language that can be utilised not only for machine learning. It can be applied also for other wide range of tasks such as web development, data analysis, automation and more.
- **Interoperability:** Python integrates well with other programming languages and tools that are commonly used in the ecosystem of machine learning.
- **Support for Rapid Prototyping:** Python's interactive development environment and expressive syntax makes it well suited rapid prototyping. Researchers can test on different algorithms and visualize results in real-time with tools like Google Colab notebook and Jupyter notebook.

TensorFlow:

- **Scalability:** TensorFlow is scalable in training and deploying machine learning models efficiently across numerous configurations, which include GPUs, CPUs, and TPUs (Tensor Processing Units). It makes it applicable for experimentation on small-scales and up to production deployments on large-scales.

- **Flexibility:** TensorFlow has flexible and modular architecture, which gives developers to train, customize, and build different range of machine learning models. The models can be customized for deep neural networks, CNN (convolutional neural networks), and recurrent neural networks. TensorFlow gives low-level APIs for control over model design, and high-level APIs for prototyping and experimentation.

- **Ecosystem:** TensorFlow provides a rich ecosystem of tools and resources. Those tools provide support on various stages of the machine learning workflow, which include data preprocessing, training, deployment, and evaluation. This ecosystem includes tools like TensorFlow Extended (TFX) that's used for end-to-end ML pipelines, TensorFlow Hub that's used for reusable model components, and TensorFlow Lite which is used for deploying machine learning models on mobile devices.

- **Integration with Python:** TensorFlow integrates with Python and leverages it's simplicity. It provides developers with leveraging existing tools and workflows from Python while training and building machine learning models with TensorFlow.

- **Ease of Use:** TensorFlow provides high-level APIs that offer intuitive interfaces for specifying training parameters, defining model architectures, and monitoring the performance of the model. It makes it easier to get started with deep learning for developers.

Other possible technologies which could have been used were React Native JavaScript and C++. Although React Native JavaScript provides many advantages on mobile app development, it wasn't chosen with reasons for limitations on performance, platform-specific issues, debugging, and third-party libraries. To achieve optimal performance developers may need to write code that is platform-specific. With platform-specific issues there can be inconsistencies with differences in platform behaviour and components for UI, which may require tweaks and optimizations.

Debugging for React Native apps can be challenging compared to native development environments. There are debugging tools out there like React Native Debugger, but they may not offer the same level of functionality and integration as native debugging tools.

React Native relies on third-party libraries for accessing native functionalities, components for UI, and integrations. While it can provide a variety of features and accelerate development, it can introduce features that may be incompatible or outdated with updated versions of React Native.

Although C++ has advantages on performance and provides low-level control that may be beneficial for certain machine learning applications, the main drawbacks would be its complexity, limited ecosystem, and steep learning curve. Comparing to Python, C++ is a complex language that requires more effort and expertise to write code along with maintaining it and debugging it. It gives a disadvantage for those who have limited experience with C++ or other low-level programming languages, and the development can make it slower more error-prone.

While C++ also has a variety of libraries and frameworks like Python, it lags behind when it comes to libraries and tools that are specifically created for machine learning and data science. That makes it more challenging on finding relevant resources and leveraging existing codebases. C++ can be challenging to learn particularly for those who’ve done higher-level languages, since it requires an understanding of memory management, data structures, and low-level concepts that may not be as intuitive. So the language makes it less suitable for data science and machine learning projects compared to higher-level languages like Python.

4.4.2. Structure of Android Studio

The technology stack of Android Studio is the official IDE for developing Android apps. The IDE consists several layers and key components. The code editor in Android Studio include Java and Kotlin, which are the two primary languages used for developing Android apps. It provides different features such as syntax highlighting, code navigation, code completion, and integrated documentation. The folder structure in Android Studio when working on a project, it typically follows a standard layout that’s defined by the Android project structure and the Gradle build system. This is an overview of the main folders in Android Studio and their purposes.

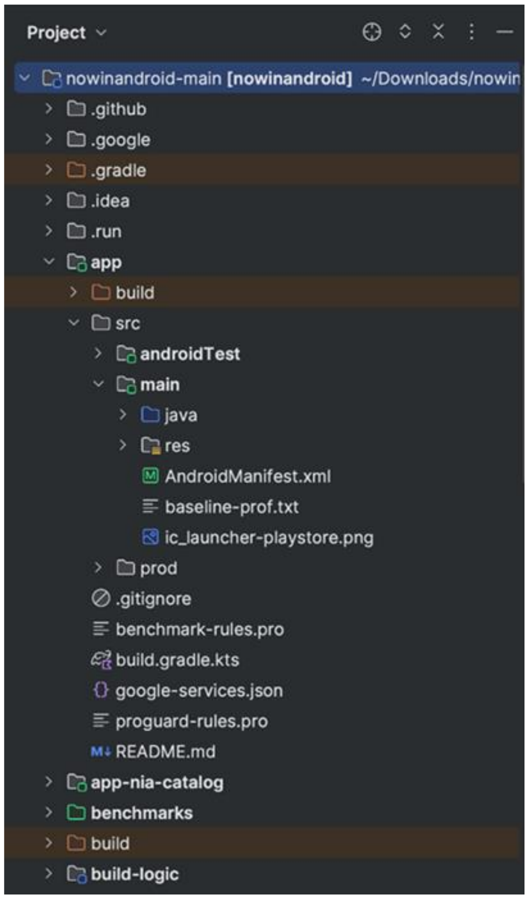
‘app’ Folder:

The main module of the Android project that contains the source code, specific resources, and configuration files that’s specific to the app. In the ‘app’ folder, contain subdirectories such as ‘src’ that contains the Java and resource files for different build variants. ‘res’ contains resources like layout XML files, string resources, drawable assets, and other resources used by the app. ‘manifests’ contains the AndroidManifest.xml file, which provides necessary information about the app, such as its package name, permissions, services, and activities. The raw asset files contain in the app that may need a runtime such as fonts, HTML or custom data files.

‘gradle’ Folder:

This folder includes files like, ‘wrapper’ which contains Gradle wrapper files that allow the app project to specify a specific Gradle version. That ensures consistent builds across different development environments. ‘buildSrc’ can contain custom plugins build scripts written in Java. ‘build’ is an auto-generated folder by Gradle, containing intermediate build files, code that is compiled, and other artifacts that are generated on the build process. The folder can be usually deleted safely without affecting the project since it’s usually ignored in version control systems. ‘.gradle’ folder contain configuration files that improve performance and maintain state.

Those list of folders are project-level build configuration files that are written in Groovy or Kotlin DSL. They define project-wide settings, build configurations, and dependencies that are shared across all modules in the project. This folder structure provides an organization that’s logical for Android projects by separating the source code, resources, build configurations, and other project-related files. The structure efficiently navigates and manages their projects.



```
Project
nowinandroid-main [nowinandroid] ~/Downloads/nowin
  .github
  .google
  .gradle
  .idea
  .run
  app
    build
    src
      androidTest
      main
        java
        res
        AndroidManifest.xml
        baseline-prof.txt
        ic_launcher-playstore.png
      prod
    .gitignore
    benchmark-rules.pro
    build.gradle.kts
    google-services.json
    proguard-rules.pro
    README.md
  app-nia-catalog
  benchmarks
  build
  build-logic
```

Figure 4.1 - Android Studio folder structure

4.4.3. Design Patterns

The development in Android Studio, the Model-View-Controller (MVC) architecture is commonly adapted into a variant architecture known as Model-View-ViewModel (MVVM) or Model-View-Presenter (MVP). This is the list of components that map to the Android Studio structure.

Model:

The Model is the data and business logic of the application. In Android, this includes data structures, database operations, network requests, and other operations that are data-related.

In Android Studio, the Model components are often represented by:

Data classes: Java classes that encapsulate data and define data structures.

Database-related classes that are responsible for database operations using SQLite, or other persistence libraries.

Network-related classes that are responsible for making network requests using libraries like Retrofit, OkHttp, or Volley.

View:

The View are the user interface (UI) components of the application. This includes UI widgets, UI-related logic, and layout XML files in Android.

In Android Studio, the View components are commonly represented by:

Activity classes in Java or Kotlin that serve as the entry point for UI interactions and the lifecycle of UI screens that is managed.

Fragment classes in Java or Kotlin that are reusable portions of UI screens and their own lifecycle that is managed.

Layout of XML files that define the appearance and structure of UI screens using views, view groups, and other UI components.

Controller / ViewModel / Presenter:

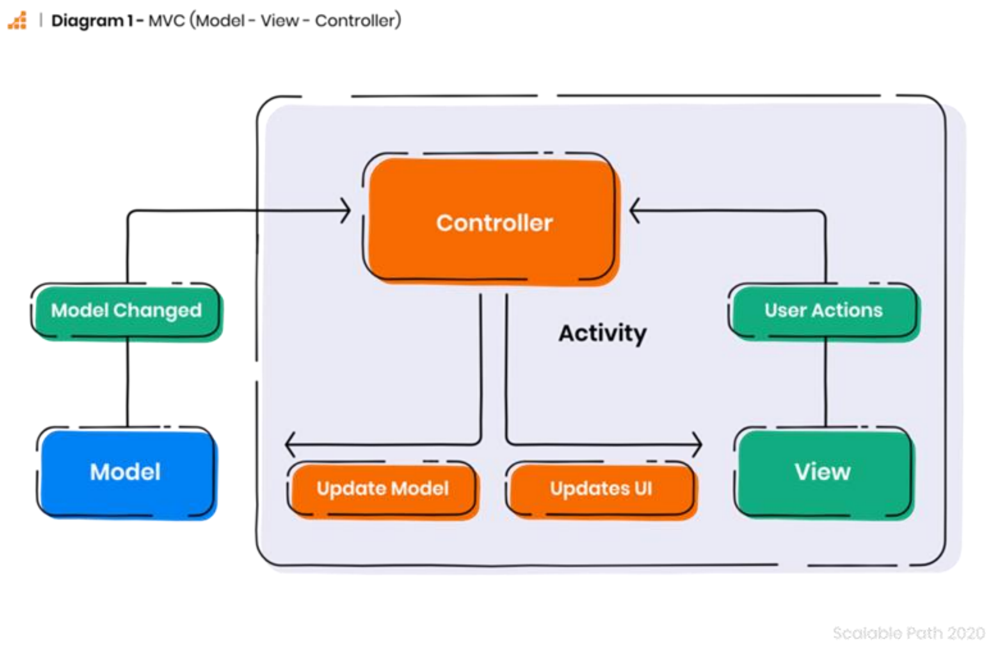
The Controller (in MVC), Presenter (in MVP), or ViewModel (in MVVM) acts as an intermediary between the Model and the View. In the Model, data is retrieved, processes it, and updates the View accordingly. It handles and input events and user interactions.

In Android Studio, the Controller/ViewModel/Presenter components are commonly represented by:

ViewModel classes, which are part of the Android Architecture Components that store and manage UI-related data.

Controller classes, which are less common in Android development, but those classes can still be implemented using custom controller classes that manage interactions between the Model and the View.

The architecture of Android Studio doesn't apply a strict adherence to MVC, MVVM, or MVP patterns, and Android projects are commonly structured by using these architectural patterns to improve code organization, and facilitate testing and maintenance. These patterns are supported by tools and features that is provided by Android Studio, such as code generators for creating ViewModel classes, layout editors for designing UI screens, and debugging tools for inspecting data and UI interactions.



4.2.4. Database Design

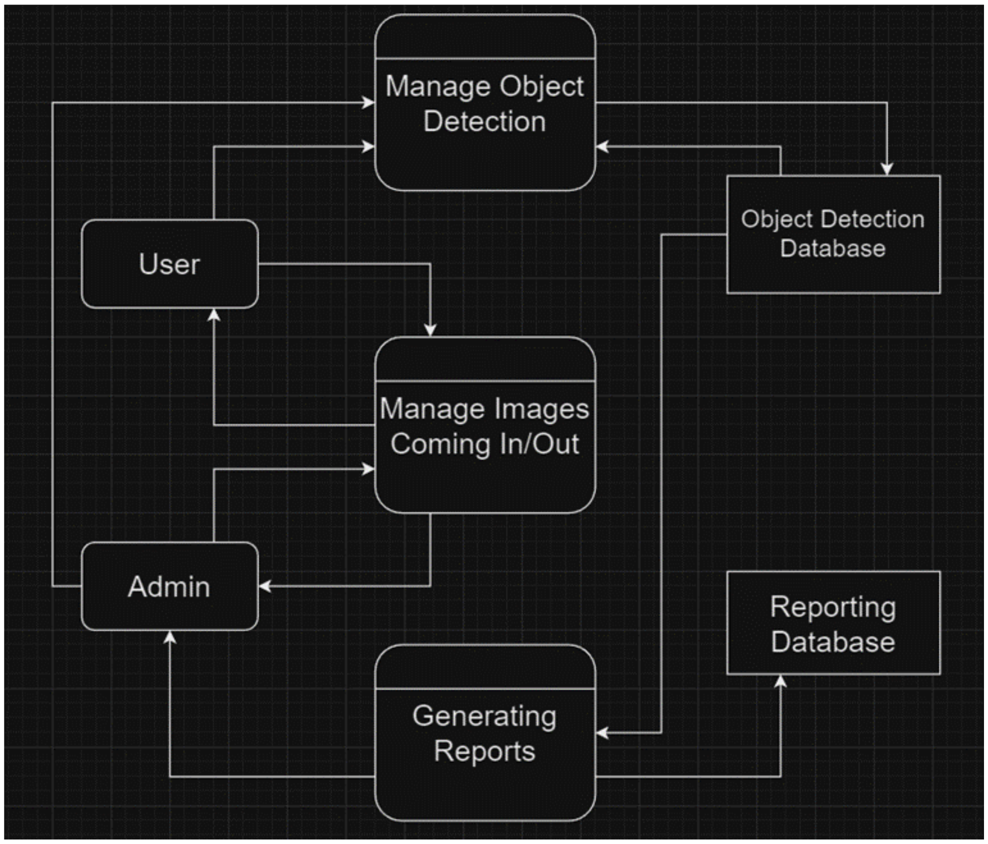


Figure 4.2 – Database design

4.3. User Interface Design

The interface of the application has a simple design with basic features of a homepage, a profile page, and a search page. The homepage provides a button below of the main page that opens a camera that is used to process selected food items and then outputs the result.

4.3.4. Wireframe

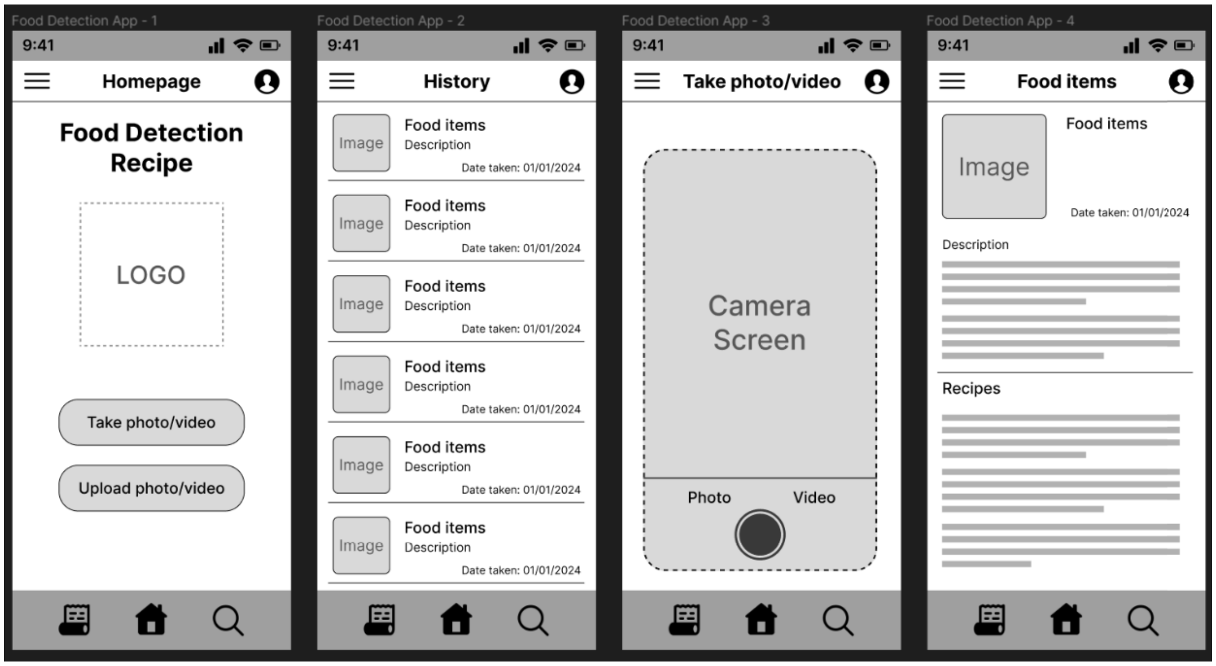


Figure 4.3 - Wireframe application

4.3.5. User Flow Diagram

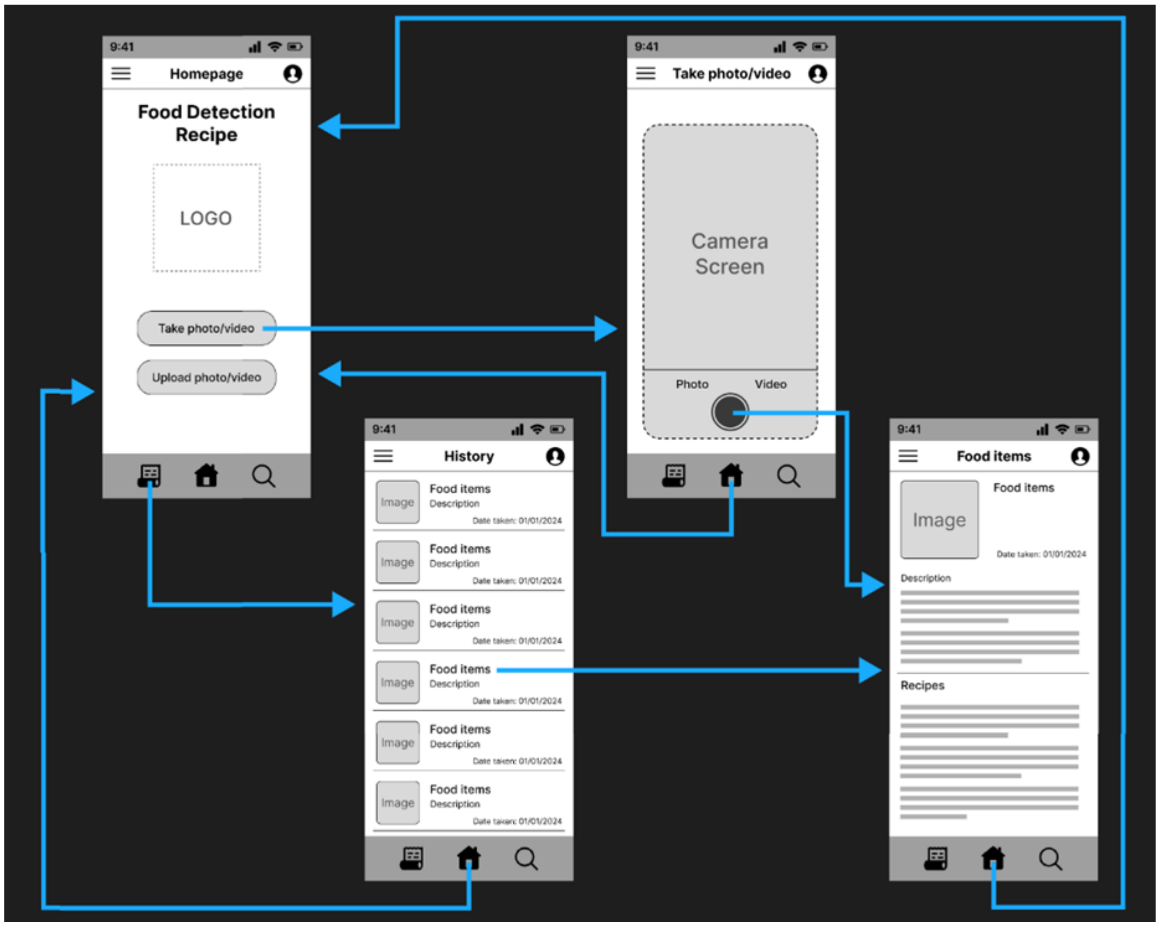


Figure 4.4 - User flow diagram

4.3.6. Style Guide

The colour palette and font were chosen for the application which are shown in Figure 5 below. The colour palette consists of primary colours and secondary colours for the application, and the font style that was chosen was ‘gotham’ style. The choice and mixture of colours that are chosen were subtle and warm, but also had good colour-blocking for colour contrast for the user interface. The choice of font style ‘gotham’, is a geometric sans-serif typeface that gives an appealing readable text.

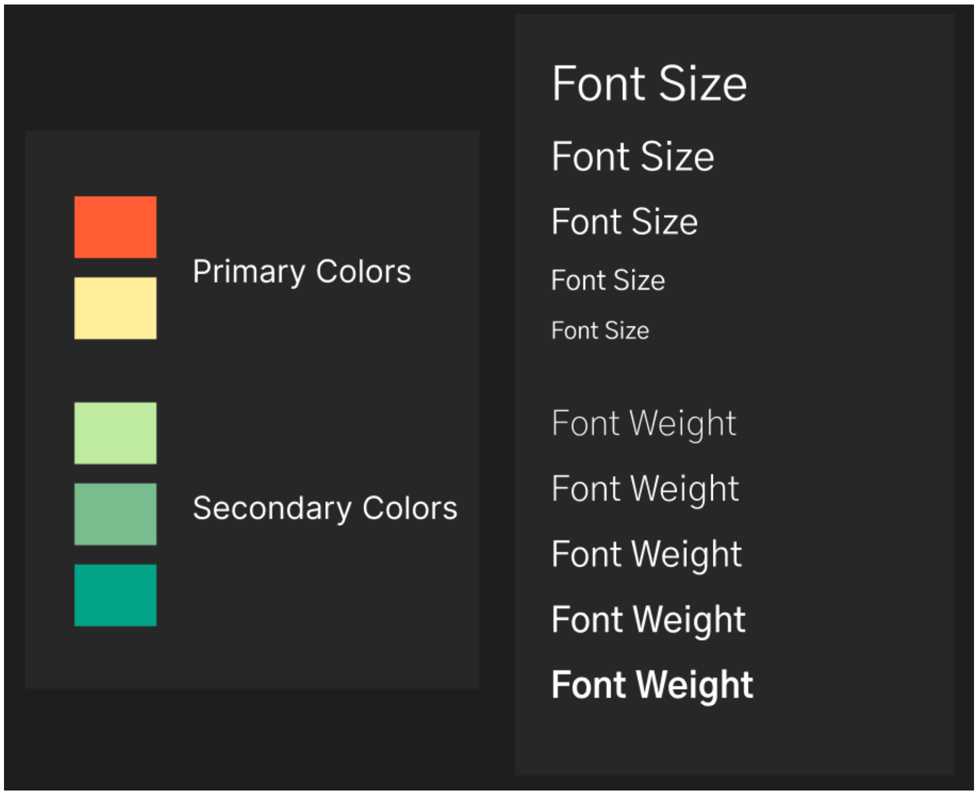


Figure 4.5 - Style guide

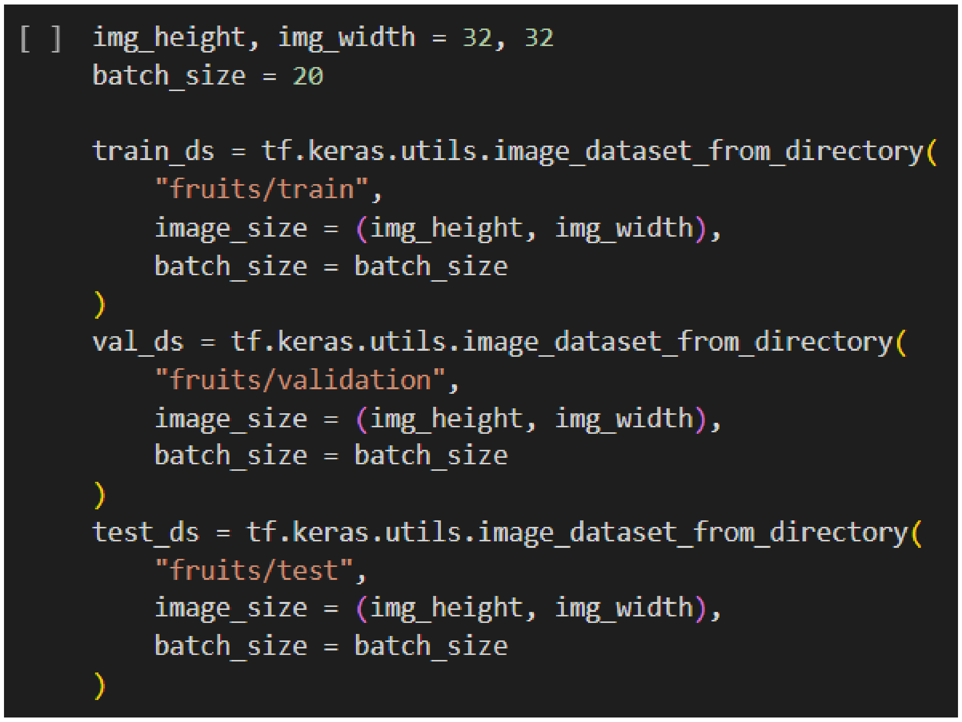
5. Implementation (or Construction)

The implementation of each component will be elaborated in terms of the code structure and how it was implemented. One of the components is the object detection model that is built by using Python programming language. The other component is using a TensorFlow library which is an open-source framework for machine learning, and it is utilised for various tasks. Various types of machine learning models can be built with TensorFlow, and it includes convolutional neural networks (CNN) which is for image recognition, along with other models like natural language processing (NLP) and recurrent neural networks (RNN). The other component is the main front-end application that handles the user interactivity to use a camera for taking images of the food items, and it’s developed by using Java programming language.

5.1. Python model

Initially the sample model that was trained and tested was deployed in an ONNX (Open Neural Network Exchange) file format. Since the optimized model format for a mobile device was a TensorFlow Lite format, the process in converting the ONNX file to a TensorFlow Lite format went with trial and error. Different methods were attempted to convert the file format, but it kept being stubborn and couldn’t convert the file format. So a different model was then trained and tested that was able to be extracted and deployed in a TensorFlow Lite format.

The code below loads the image datasets for training, validation and testing from directories using the function ‘image_dataset_from_directory’ from TensorFlow. In this dataset is a set of fruits that is being trained and validated. The first line of the code sets the height and width of the image to 32 by 32 pixels. The batch size is set to 20 images, it determines the number of samples that will be propagated at a time during the training process. The template of the code that is split into train, validation, and test, creates TensorFlow datasets train_ds, val_ds, and test_ds from the images that is stored in the specified directories, along with the image size and the batch size.

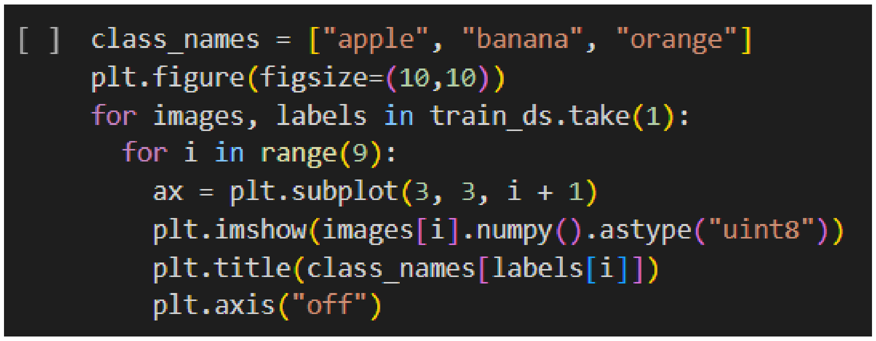


```
[ ]  img_height, img_width = 32, 32
     batch_size = 20

     train_ds = tf.keras.utils.image_dataset_from_directory(
         "fruits/train",
         image_size = (img_height, img_width),
         batch_size = batch_size
     )
     val_ds = tf.keras.utils.image_dataset_from_directory(
         "fruits/validation",
         image_size = (img_height, img_width),
         batch_size = batch_size
     )
     test_ds = tf.keras.utils.image_dataset_from_directory(
         "fruits/test",
         image_size = (img_height, img_width),
         batch_size = batch_size
     )
```

Figure 5.1.1 – Loading the image datasets

This code visualises a subset of images in the training dataset, and it defines a list of class names that correspond to the labels in the dataset. It creates a new figure with a size of 10 by 10 inches by using plt from Matplotlib. The code 'for i in range(9)' is the loop that iterates over the first 9 images in the batch, and then the code 'ax = plt.subplot(3, 3, i + 1)' creates a subplot within the figure grid of 3 by 3 and the i + 1 determines the current subplot position within the grid. Then it displays the image located at 'i' within the batch, then the .numpy() converts it to a NumPy array an.astype("uint8") converts the pixel values to unsigned integers to display the image correctly.

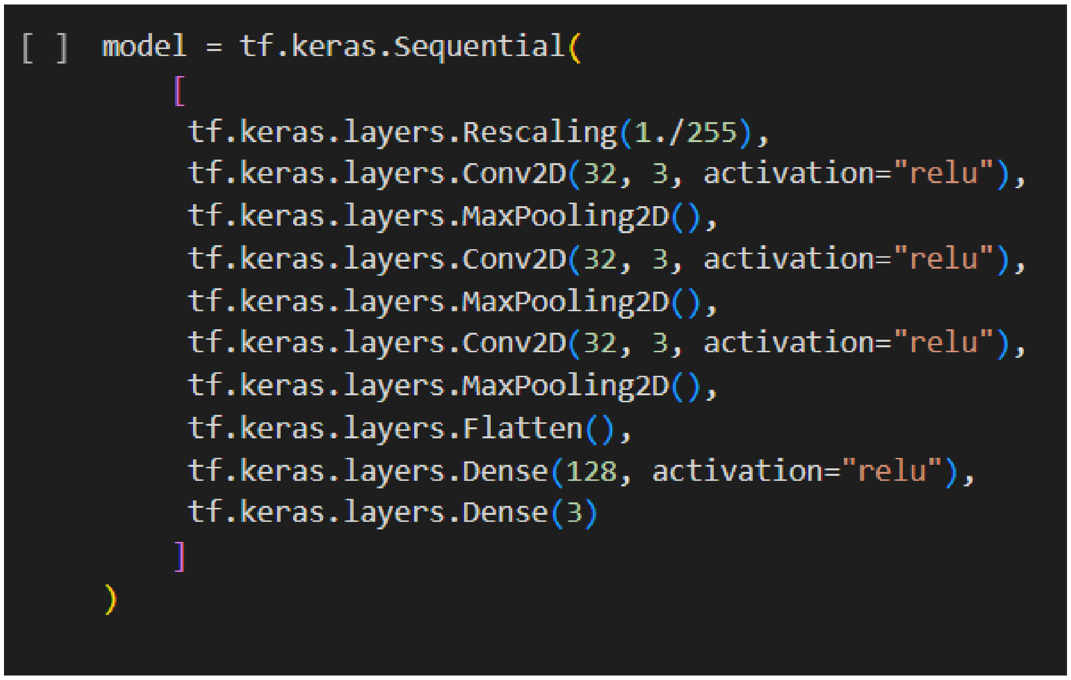


```
[ ]  class_names = ["apple", "banana", "orange"]
     plt.figure(figsize=(10,10))
     for images, labels in train_ds.take(1):
       for i in range(9):
         ax = plt.subplot(3, 3, i + 1)
         plt.imshow(images[i].numpy().astype("uint8"))
         plt.title(class_names[labels[i]])
         plt.axis("off")
```

Figure 5.1.2 – Defining a list of class names

This defines the model into a convolutional neural network (CNN) by using the keras API tool from TensorFlow. It creates a model with a linear stack of layers called Sequential. The first layer rescales the input pixel values from 0,255 to 0,1 and it's a common image processing task called normalization. Then it adds a 2D convolutional layer, in this example it's with 32 filters, each with a 3 by 3 size. The 'relu' function is used which is Rectified Linear Unit.

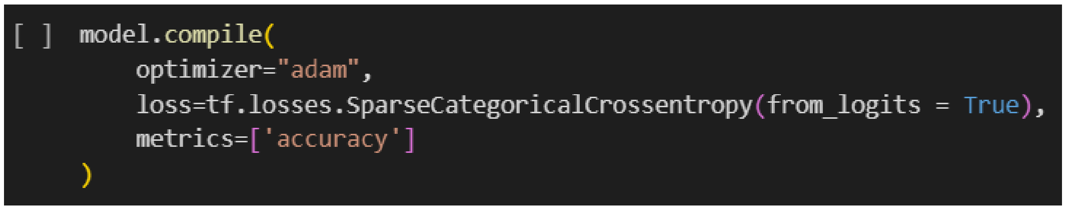
Then the next layer performs a max pooling operation that reduces the spatial dimensions of the feature maps that were obtained from the previous convolutional layers. In this code it's repeated twice, then it flattens the input feature maps into a one dimensional array. Then it's the dense layer and it has a number unit that corresponds to the number of classes in the classification task, in this case it's 3 classes. Then a suitable activation function can be applied later during the training or evaluation.



```
[ ]  model = tf.keras.Sequential(
         [
          tf.keras.layers.Rescaling(1./255),
          tf.keras.layers.Conv2D(32, 3, activation="relu"),
          tf.keras.layers.MaxPooling2D(),
          tf.keras.layers.Conv2D(32, 3, activation="relu"),
          tf.keras.layers.MaxPooling2D(),
          tf.keras.layers.Conv2D(32, 3, activation="relu"),
          tf.keras.layers.MaxPooling2D(),
          tf.keras.layers.Flatten(),
          tf.keras.layers.Dense(128, activation="relu"),
          tf.keras.layers.Dense(3)
         ]
     )
```

Figure 5.1.3 – Defining the model into a convolutional neural network

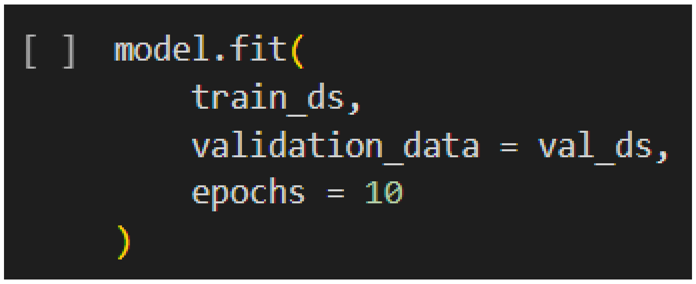
The training process is then configured for the model, and then compiles by specifying the optimizer to be used during the training which is ‘adam’. It’s a popular optimizer choice for gradient-based optimization algorithms. The next code defines the loss function to be optimised during the training, and ‘SparseCategoricalCrossentropy’ is suitable for multi-class classification problems with integer labels. The last line of code specifies the metrics to be evaluated during the training and testing, in this code it uses ‘accuracy’ that calculates the accuracy of the model’s predictions.



```
[ ]  model.compile(
         optimizer="adam",
         loss=tf.losses.SparseCategoricalCrossentropy(from_logits = True),
         metrics=['accuracy']
     )
```

Figure 5.1.4 – Configuring the training process for the model

Then it trains the model by using the ‘train_ds’ dataset and validates by using the ‘val_ds’ dataset. Then the number of epochs is specified for which the model will be trained on and in this case it’s 10 epochs. Each epoch is one complete pass through the entire training dataset, and during the training the model will update its weights and biases. That will use the ‘adam’ optimizer to minimize the loss function which is ‘SparseCategoricalCrossentropy’, and after each epoch the accuracy of the training and validation datasets will be calculated.

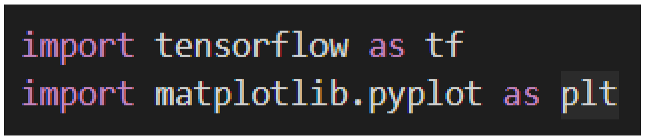


```
[ ]  model.fit(
         train_ds,
         validation_data = val_ds,
         epochs = 10
     )
```

Figure 5.1.5 – Training the model

5.2. TensorFlow library

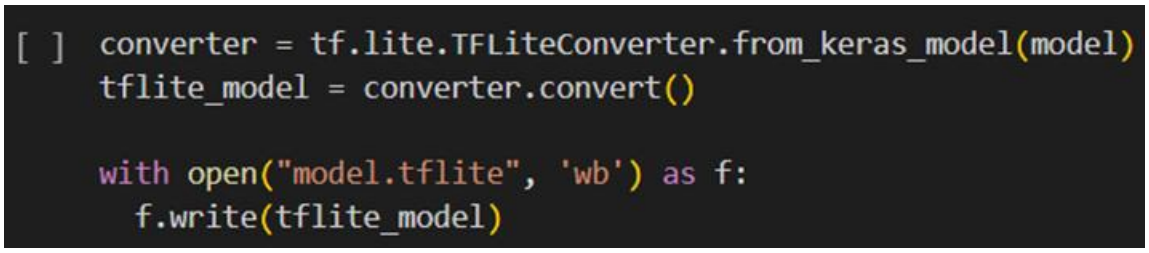
The TensorFlow library is imported in python with a simple code.



```
import tensorflow as tf
import matplotlib.pyplot as plt
```

Figure 5.2.1 – Importing the TensorFlow library

The trained keras model is then converted into a TensorFlow Lite format, which is optimized for mobile devices. The model converts into a model.tflite format when deployed. The second line of the code invokes the convert() method of the converter object, and it converts the keras format to a TensorFlow Lite format. In the third line of code f.write(tflite_model) writes the contents of the model to a ‘model.tflite’ file.



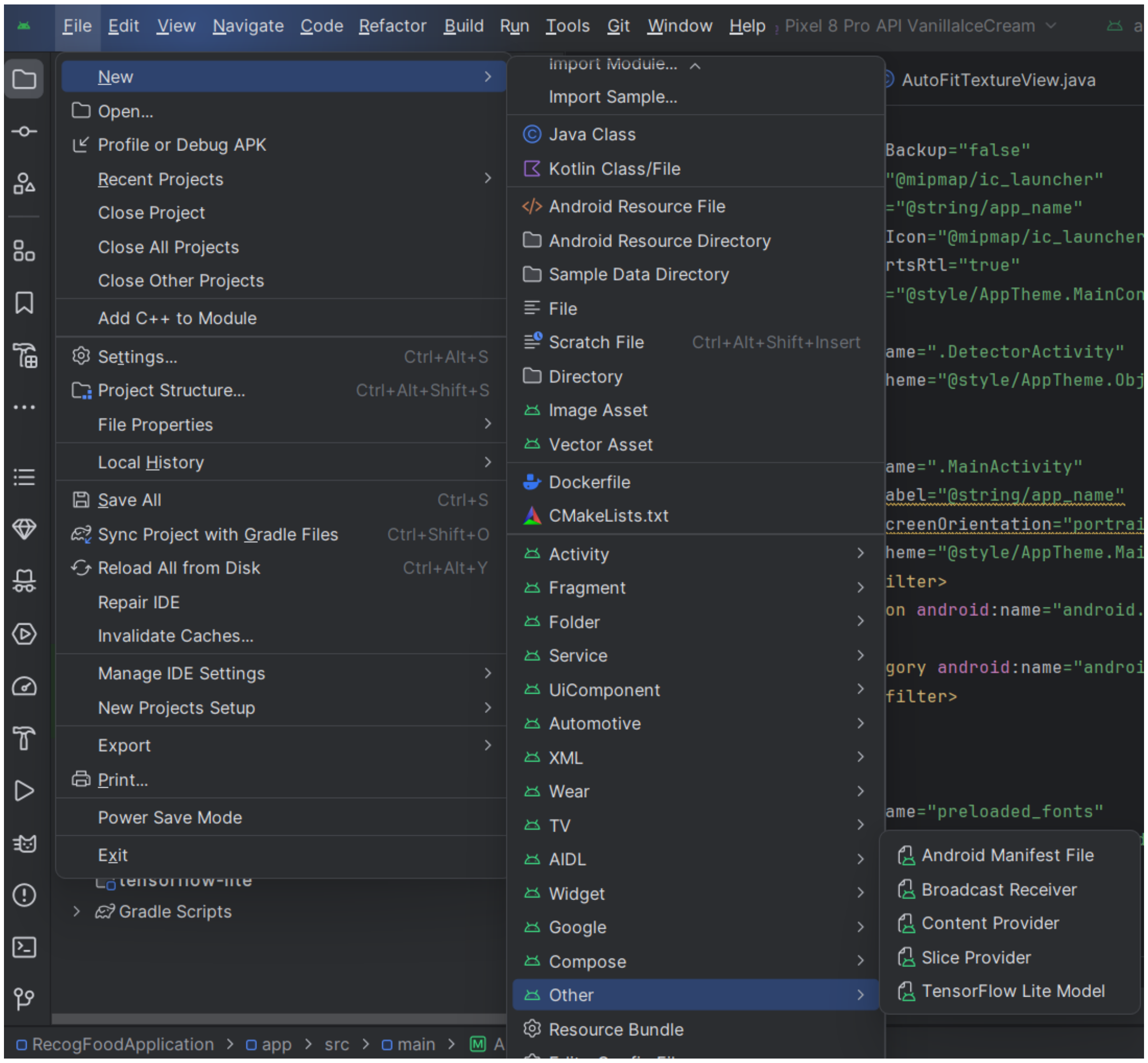
```
[ ]  converter = tf.lite.TFLiteConverter.from_keras_model(model)
     tflite_model = converter.convert()

     with open("model.tflite", 'wb') as f:
       f.write(tflite_model)
```

Figure 5.2.2 – Converting the model into a TensorFlow Lite format

5.3. Java application

The model was then implemented in a Java application through the Android Studio IDE. In the file system there’s an option to import a TensorFlow Lite model at the bottom right of the figure below, which is then added into the assets folder.



```
Backup="false"
"@mipmap/ic_launcher"
="@string/app_name"
Icon="@mipmap/ic_launcher
rtsRtl="true"
="@style/AppTheme.MainCon

ame=".DetectorActivity"
heme="@style/AppTheme.Obj

ame=".MainActivity"
abel="@string/app_name"
creenOrientation="portrai
heme="@style/AppTheme.Mai
ilter>
on android:name="android.

gory android:name="androi
filter>

ame="preloaded_fonts"
```

Figure 5.3.1 – Android Studio file system

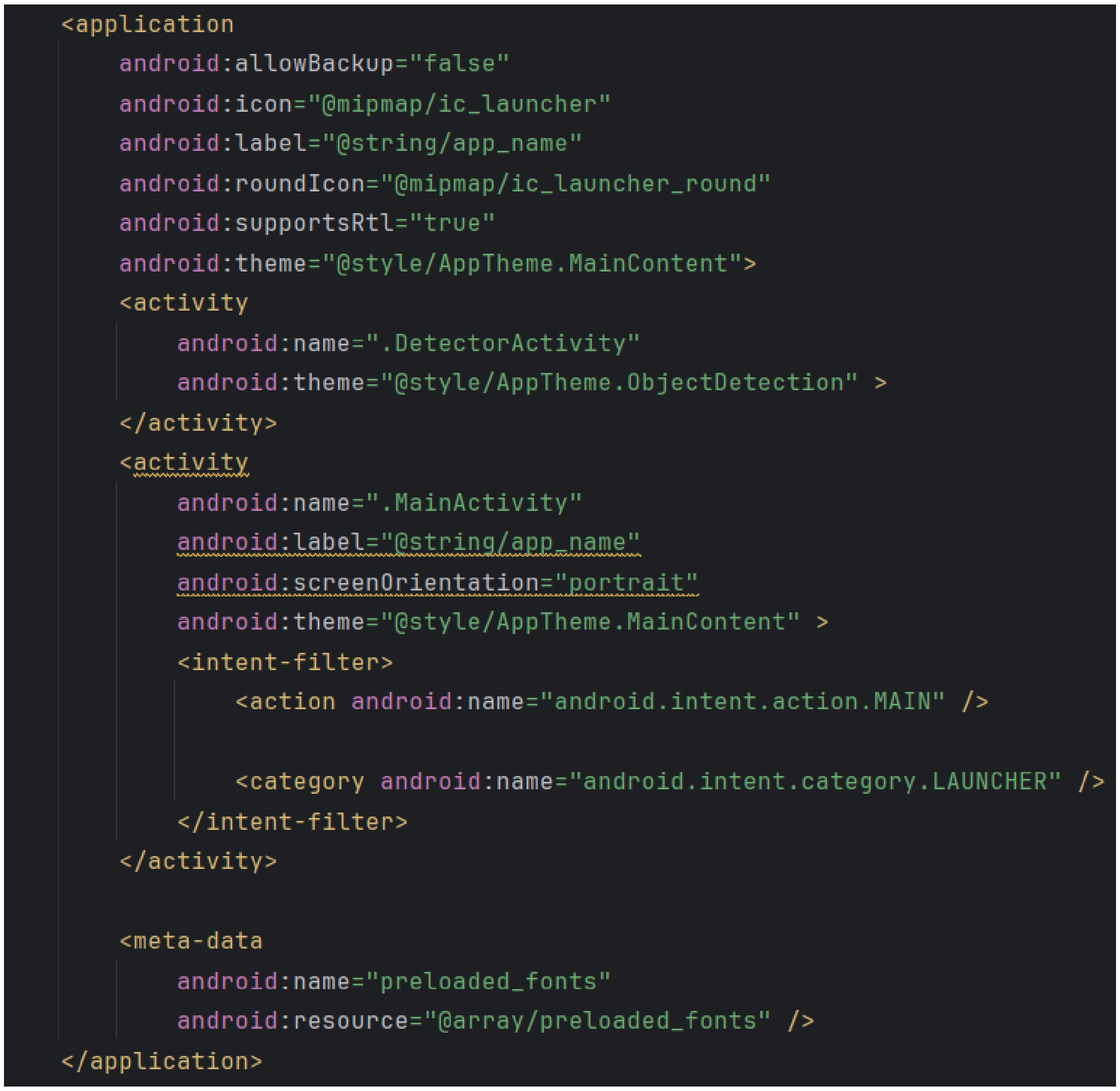
The code below specifies the permission declaration to access the camera on the device. It also declares the presence of a camera and the camera to support autofocus on the device.

```
    <uses-permission android:name="android.permission.CAMERA" />

    <uses-feature android:name="android.hardware.camera" />
    <uses-feature android:name="android.hardware.camera.autofocus" />
```

Figure 5.3.2 – Camera function permission declaration

This is the application element and its child elements in an xml file. It defines various attributes, activities, and metadata for an Android application, including its main and secondary activities, icons, labels, and themes.



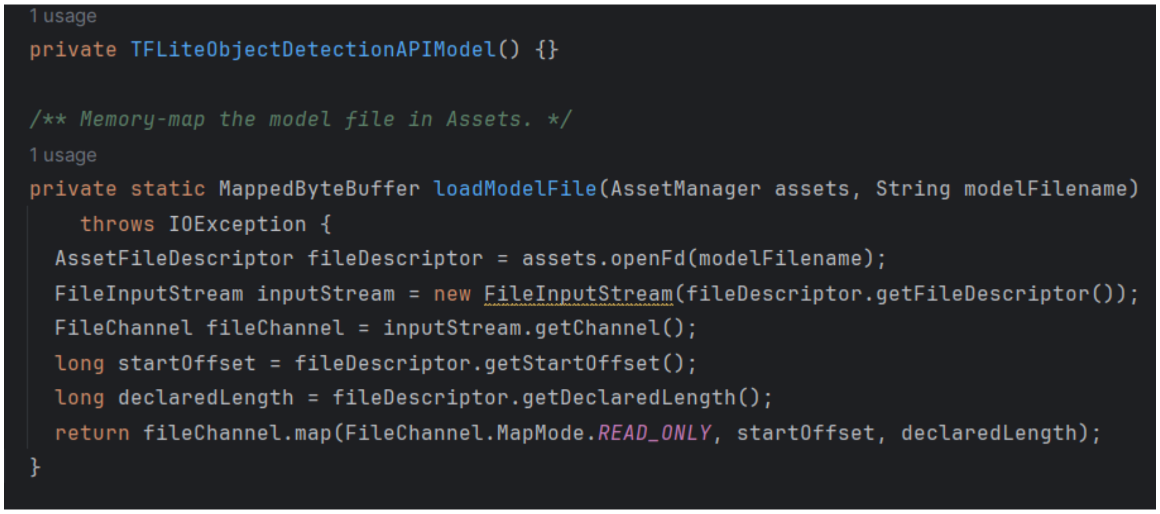
```
<application
    android:allowBackup="false"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme.MainContent">
    <activity
        android:name=".DetectorActivity"
        android:theme="@style/AppTheme.ObjectDetection" >
    </activity>
    <activity
        android:name=".MainActivity"
        android:label="@string/app_name"
        android:screenOrientation="portrait"
        android:theme="@style/AppTheme.MainContent" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <meta-data
        android:name="preloaded_fonts"
        android:resource="@array/preloaded_fonts" />
</application>
```

Figure 5.3.3 – Application element and its child elements

This loads a TensorFlow Lite model file from the assets folder directory.

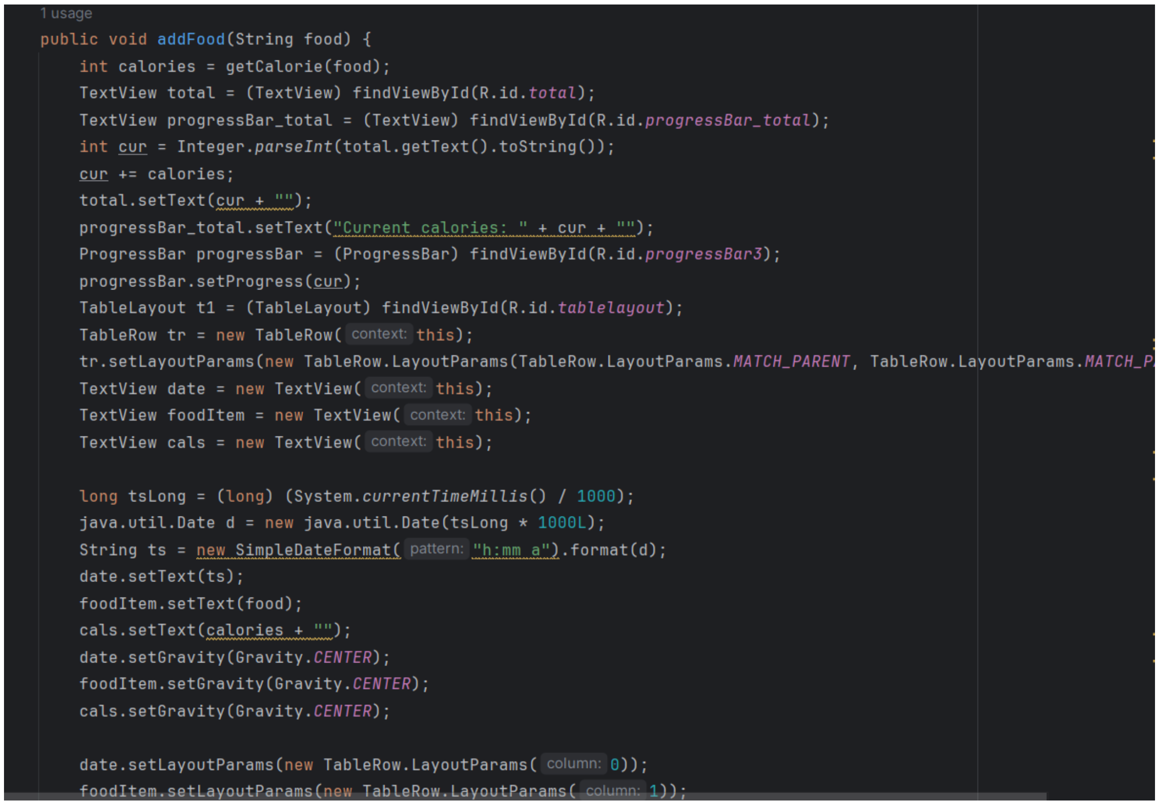


```
1 usage
private TFLiteObjectDetectionAPIModel() {}

/** Memory-map the model file in Assets. */
1 usage
private static MappedByteBuffer loadModelFile(AssetManager assets, String modelFilename)
    throws IOException {
  AssetFileDescriptor fileDescriptor = assets.openFd(modelFilename);
  FileInputStream inputStream = new FileInputStream(fileDescriptor.getFileDescriptor());
  FileChannel fileChannel = inputStream.getChannel();
  long startOffset = fileDescriptor.getStartOffset();
  long declaredLength = fileDescriptor.getDeclaredLength();
  return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength);
}
```

Figure 5.3.4 – Loading a TensorFlow Lite model

This is a partial snippet of the main activity on the application where the users track the calorie intake when certain food items are added and display the calorie count.



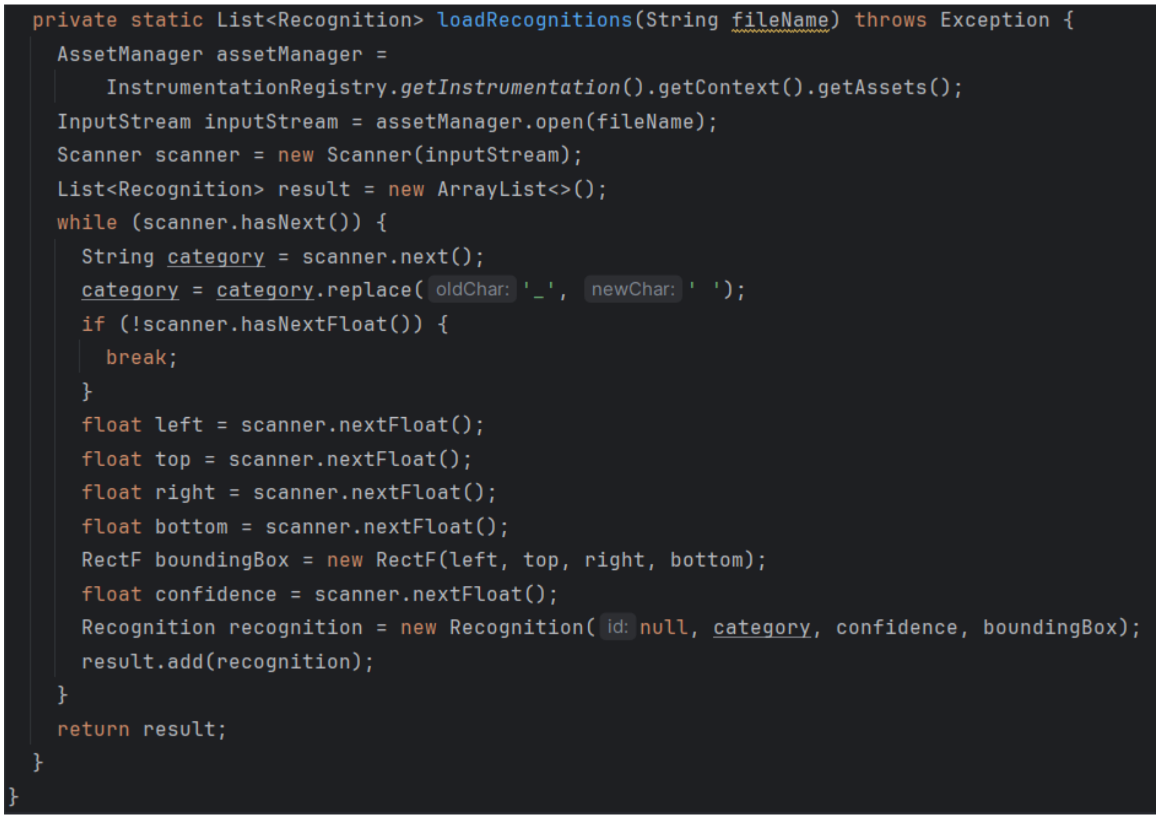
```
1 usage
public void addFood(String food) {
    int calories = getCalorie(food);
    TextView total = (TextView) findViewById(R.id.total);
    TextView progressBar_total = (TextView) findViewById(R.id.progressBar_total);
    int cur = Integer.parseInt(total.getText().toString());
    cur += calories;
    total.setText(cur + "");
    progressBar_total.setText("Current calories: " + cur + "");
    ProgressBar progressBar = (ProgressBar) findViewById(R.id.progressBar3);
    progressBar.setProgress(cur);
    TableLayout t1 = (TableLayout) findViewById(R.id.tablelayout);
    TableRow tr = new TableRow( context: this);
    tr.setLayoutParams(new TableRow.LayoutParams(TableRow.LayoutParams.MATCH_PARENT, TableRow.LayoutParams.MATCH_P
    TextView date = new TextView( context: this);
    TextView foodItem = new TextView( context: this);
    TextView cals = new TextView( context: this);

    long tsLong = (long) (System.currentTimeMillis() / 1000);
    java.util.Date d = new java.util.Date(tsLong * 1000L);
    String ts = new SimpleDateFormat( pattern: "h:mm a").format(d);
    date.setText(ts);
    foodItem.setText(food);
    cals.setText(calories + "");
    date.setGravity(Gravity.CENTER);
    foodItem.setGravity(Gravity.CENTER);
    cals.setGravity(Gravity.CENTER);

    date.setLayoutParams(new TableRow.LayoutParams( column: 0));
    foodItem.setLayoutParams(new TableRow.LayoutParams( column: 1));
```

Figure 5.3.5 – Calorie count activity

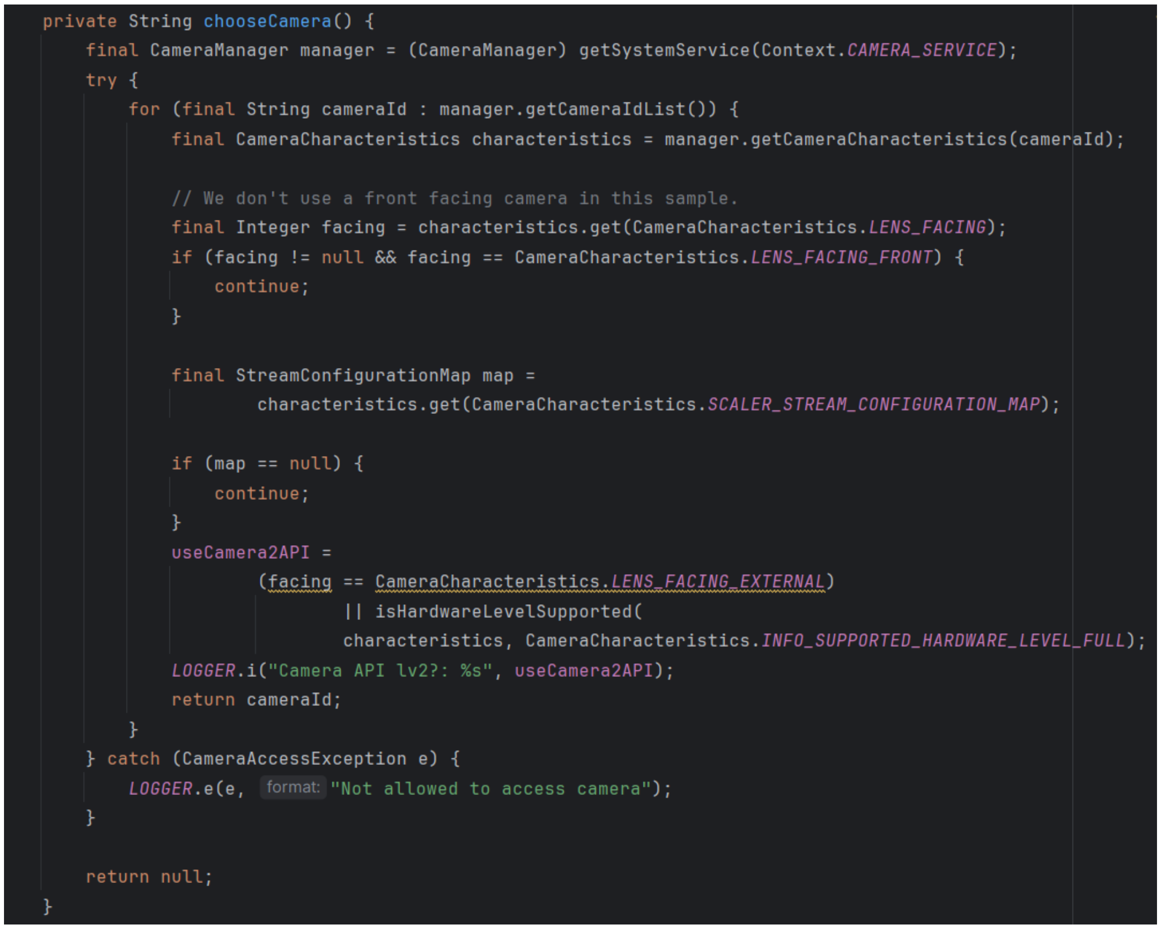
This partial snippet of the code is the method loads expected recognition output results from a text file. The full code tests and ensures the object detection algorithm detects intended results when applied to a specific image.



```java
  private static List<Recognition> loadRecognitions(String fileName) throws Exception {
    AssetManager assetManager =
        InstrumentationRegistry.getInstrumentation().getContext().getAssets();
    InputStream inputStream = assetManager.open(fileName);
    Scanner scanner = new Scanner(inputStream);
    List<Recognition> result = new ArrayList<>();
    while (scanner.hasNext()) {
      String category = scanner.next();
      category = category.replace( oldChar: '_', newChar: ' ');
      if (!scanner.hasNextFloat()) {
        break;
      }
      float left = scanner.nextFloat();
      float top = scanner.nextFloat();
      float right = scanner.nextFloat();
      float bottom = scanner.nextFloat();
      RectF boundingBox = new RectF(left, top, right, bottom);
      float confidence = scanner.nextFloat();
      Recognition recognition = new Recognition( id: null, category, confidence, boundingBox);
      result.add(recognition);
    }
    return result;
  }
}
```

Figure 5.3.6 – Loading recognition results

This a partial snippet from the code that provides a framework for implementing a camera functionality for image processing. This determines capability and the hardware support of the chosen camera.



```
private String chooseCamera() {
    final CameraManager manager = (CameraManager) getSystemService(Context.CAMERA_SERVICE);
    try {
        for (final String cameraId : manager.getCameraIdList()) {
            final CameraCharacteristics characteristics = manager.getCameraCharacteristics(cameraId);

            // We don't use a front facing camera in this sample.
            final Integer facing = characteristics.get(CameraCharacteristics.LENS_FACING);
            if (facing != null && facing == CameraCharacteristics.LENS_FACING_FRONT) {
                continue;
            }

            final StreamConfigurationMap map =
                    characteristics.get(CameraCharacteristics.SCALER_STREAM_CONFIGURATION_MAP);

            if (map == null) {
                continue;
            }
            useCamera2API =
                    (facing == CameraCharacteristics.LENS_FACING_EXTERNAL)
                            || isHardwareLevelSupported(
                            characteristics, CameraCharacteristics.INFO_SUPPORTED_HARDWARE_LEVEL_FULL);
            LOGGER.i("Camera API lv2?: %s", useCamera2API);
            return cameraId;
        }
    } catch (CameraAccessException e) {
        LOGGER.e(e, format: "Not allowed to access camera");
    }

    return null;
}
```

Figure 5.3.7 – Camera activity

6. Testing and Analysis

6.1. Testing with food items

The testing and analysis will go through the use test of the application, which will go through using the camera functionality and detecting selected food items and potentially other objects. The testing went by first trying to detect a simple food item like an orange or an apple, and the result was that it outputted the intended result of the food item which was an orange or an apple that was detected. Then other food items were tested like bread and pizza, which most of the time detected and outputted the correct name of the food item. A few tests gave incorrect predictions to certain food items like a muffin instead of bread or pancakes instead of pizza. Those tests showed that the model predicts certain food items that look similar to other foods, and recognizes the same patterns.

6.2. Testing with non-food items

Certain non-food items were tested like a mug or a mobile phone, and most of the cases it did not detect and predict any names of food. Except for at certain angles and glimpses of the image, the model detected the mug as a muffin and a fist as an apple. That again goes to the model’s accuracy on detecting similar patterns of the object within the image. This use test showed that the model can have different accuracies depending on certain data it was trained with and the quantity of data it trained with.

7. Discussion

The application turned out with the bare minimum requirements of detecting the food with names that are outputted. The feature of having certain amount of calories for each food item was added that adds up on the counter for every food item it detects, and keeps track of the calorie intake. If there is anything that would be changed in this project is to recreate the application in React Native JavaScript. The reason is for a more universal deployment of a mobile application in terms of an operating system. The other addition that would be adjusted is the User Interface a bit with some separate pages added. If there was better time management in the process of development, additional features would be implemented of having the API connected to generate the recipes of certain food items that would detect and output. Additionally, with the features of a search bar system and a log history storage.

8. Conclusion

The development of this project was informative in the process of using a host of technologies which were, Java in Android Studio, Python, and TensorFlow library. There were concerns of deploying the machine learning in the correct format and then implementing it in a mobile application. Since Android Studio had a feature tool to implement certain applicable machine learning models, it eased the processed slightly and it came together stable. Users were able to use the camera functionality to detect the food items that were chosen, and output the name of the food item and the calorie quantity that they intake.

The aim was to understand more about machine learning that is used in image recognition applications. That aim was achieved with the research in different types of image recognition applications, methods and techniques that are used. The application gave a hands-on experience to develop and implement, while researching different tools and documents. That gave a valuable experience on working with the development process. It would be interesting for personal development and to test different optimal techniques and methods to create machine learning models that can be utilized.

References

Du, C.-J., & Sun, D.-W. (2004). Recent developments in the applications of image processing techniques for food quality evaluation. Trends in Food Science & Technology, 15(5), 230–249. https://doi.org/10.1016/j.tifs.2003.10.006

Jähne, B. (2005). Digital Image Processing. In Google Books. Springer Science & Business Media. https://books.google.ie/books?hl=en&lr=&id=GVWYBrDXMNkC&oi=fnd&pg=PA3&dq=image+processing+and+how+it+works&ots=0LOGMb_Ry4&sig=d7e_vE9iEGiyPWwdTrGnSM-sJhM&redir_esc=y#v=onepage&q=image%20processing%20and%20how%20it%20works&f=false

Kk, P., Ma, K., & Dk, S. (2015). Image Processing Tools and Techniques Used in Computer Vision for Quality Assessment of Food Products: A Review Image Processing Tools and Techniques Used in Computer Vision for Quality Assessment of Food Products: A Review. International Journal of Food Quality and Safety | Year-2015 |, 1, 1–16.

WANG, P. S. P. (2000). 3D ARTICULATED OBJECT UNDERSTANDING, LEARNING, AND RECOGNITION FROM 2D IMAGES. International Journal of Pattern Recognition and Artificial Intelligence, 14(07), 863–873. https://doi.org/10.1142/s0218001400000544

Girish, D., Singh, V., & Ralescu, A. (n.d.). Unsupervised clustering based understanding of CNN.

Hossain, Md. A., & Alam Sajib, Md. S. (2019). Classification of Image using Convolutional Neural Network (CNN). Global Journal of Computer Science and Technology, 13–18. https://doi.org/10.34257/gjcstdvol19is2pg13

Monga, V., Li, Y., & Eldar, Y. C. (2021). Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing. IEEE Signal Processing Magazine, 38(2), 18–44. https://doi.org/10.1109/msp.2020.3016905

Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82(6), 9243–9275. https://doi.org/10.1007/s11042-022-13644-y

Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82(6), 9243–9275. https://doi.org/10.1007/s11042-022-13644-y