



**Ain Shams University**  
**Faculty of Computer & Information Sciences**  
**Computer Science Department**



**June 2023**



**Ain Shams University**  
**Faculty of Computer & Information Sciences**  
**Computer Science Department**

# **CalorieMe: Image-based Calorie Estimator System**

**By:**

Mohamed Ibrahim Mohamed  
Waleed Mohamed Mohamed  
Bavly Magid Wadea  
Mazen Mohamed Bakr  
Mazen Khaled Ibrahim  
Mohamed Sabry Youssef

**Under Supervision of:**

Dr. Hanan Hindy [BSc, MSc, PhD],  
Computer Science Department,  
Faculty of Computer and Information Sciences,  
Ain Shams University.

TA. Yomna Ahmed [BSc],  
Computer Science Department,  
Faculty of Computer and Information Sciences,  
Ain Shams University.

## Acknowledgement

All praise and thanks to ALLAH, who provided us with the ability to complete this work. I hope to accept this work from us.

We are grateful of *our parents* and *our families* who are always providing help and support throughout the whole years of study. we hope we can give that back to them.

We also offer our sincerest gratitude to our supervisors, *Dr. Hanan Hindy. and T.A. Yomna Ahmed.* who have supported us throughout our thesis with their patience, knowledge and experience.

Finally, we would like to thank our friends and all people who gave us support and encouragement.

# Abstract

In recent years, owing to the rise in healthy eating, various food photo recognition applications for recording meals have been released. However, some of these applications require human assistance for calorie estimation; such as manual input or the help of a nutrition expert. Additionally, even with automated ones, food categories are often limited, or images from multiple viewpoints are required. Meanwhile, methodologies on image recognition have advanced greatly because of the advent of Convolutional Neural Network (CNN). CNN has improved accuracies of various image recognition tasks such as classification and object detection.

This project proposes a CNN-based food calorie estimation for multiple ingredient food photos. The proposed method estimates food ingredient locations and food calories simultaneously by using two CNN models; one for recognizing the food and one to segmenting the recognized ingredient. Currently, there is no dataset of multiple-ingredient food photos annotated with bounding boxes, segmentation masks and food calories. In this work, two types of datasets are used alternately for training the two CNN models. The two datasets are: multiple-dish/ingredient food photos annotated with bounding boxes [UECFood-100] and multiple-ingredient food photos with food masks [UECFoodPixComplete]. Finally, the proposed model compares the food type and portion size against a dataset of food types and their respective calorific value per fixed serving to estimate the total number of calories. The presented results confirm the practicality of the approach and showed that our proposed method fulfilled a promising potential in the food image-based calorie estimation field compared to the state-of-the-art results.

## ملخص

في السنوات الأخيرة، نظرًا لارتفاع الوعي العام بأساليب الأكل الصحي، تم إطلاق تطبيقات متعددة للتعرف على صور الطعام وتسجيل القيمة الغذائية لها ومتابعة الوجبات على مدار اليوم. ومع ذلك، يتطلب استخدام هذه التطبيقات المساعدة بشرية لتقدير السرعات الحرارية مثل الإدخال اليدوي لكميات الطعام أو مساعدة خبير تغذية. بالإضافة إلى ذلك، من الملاحظ أنه في التطبيقات التي تعتمد على الاستخدام التلقائي (Automated applications) تكون فئات الطعام غالبًا محدودة، أو تتطلب من المستخدم تصوير الطعام بأكثر من زاوية. في خلال الأعوام السابقة، تطورت منهجيات التعرف على الصور بشكل كبير بسبب ظهور الشبكات العصبية الالتفافية (CNNs). أدى استخدام الـ CNNs إلى تحسن واضح في دقة مهام التعرف على الصور المختلفة مثل التصنيف وتمييز الأشياء. استنادًا إلى ذلك، نقترح في هذا المشروع طرح تطبيق مبني على استخدام الـ CNN قادر على استخراج وتقدير القيمة الغذائية للوجبات المتعددة المكونات مستندًا إلى صور الطعام فقط. من خلال هذا المشروع تتمكن الآلة من التعرف على موقع المكونات الغذائية و تحديد السرعات الحرارية في نفس الوقت عن طريق استخدام نموذجين من نماذج الـ CNN يعملان في آن واحد، أحدهما مخصص للتعرف على أنواع الطعام والآخر لتجزئة أصناف الطعام التي تم التعرف عليها. حاليًا، لا يوجد قاعدة بيانات من صور الطعام المتعددة المكونات مشروحة بمربعات التحديد وأقنعة التجزئة والسرعات الحرارية للطعام. في هذا العمل نستخدم نوعين من قواعد البيانات بالتوازي لتدريب نموذجي الـ Deep Learning أولهما هي UECFOOD-100 وهي قاعدة بيانات مكونة من العديد من صور الطعام متعدد المكونات مزودة بمربعات تحديد المخصصة لتدريب الشبكة المسؤولة عن التعرف على نوع الطعام. أما الأخرى فهي UECFoodPixComplete وهي قاعدة بيانات مكونة من العديد من صور الطعام المتعدد المكونات مرفقة بأقنعة Image masks لهذه الصور وهي مخصصة لتدريب الشبكة المسؤولة عن تجزئة مكونات الطعام. وأخيرًا نقوم بمقارنة نوع الطعام وحجم حصة الطعام مقابل قاعدة بيانات مكونة من أنواع الطعام و القيمة الغذائية الموجودة داخل حصة محددة لكل صنف من أصناف الطعام الموجودة. وقد أظهرت نتائج هذا المشروع أن الطريقة المقترحة قد حققت إمكانات واعدة في مجال تقدير السرعات الحرارية القائم على صور الطعام.

# Table of Contents

Acknowledgement.....	i
Abstract.....	ii
List of Figures.....	vi
List of Tables.....	viii
List of Abbreviations .....	ix
1 Introduction.....	1
1.1 Motivation.....	1
1.2 Problem Definition .....	2
1.3 Objective .....	2
1.4 Time Plan.....	2
1.5 Document Organization.....	3
2 Literature Review .....	4
2.1 Project Overview.....	4
2.2 Theoretical Background.....	5
2.3 Related Work.....	7
2.3.1 Food Image Recognition and Segmentation Tasks.....	7
2.3.2 Food Portion Estimation Task.....	15
3 System Architecture and Methods.....	20
3.1 System Architecture .....	20
3.2 Description of Methods and Procedures.....	21
3.3 System Users.....	29
4 System Implementation and Results .....	30
4.1 Datasets.....	30
4.2 Software Tools Used .....	33
4.3 Setup Configuration (Software).....	34
4.4 Experiments and Results.....	36

4.4.1 Dataset Selection .....	36
4.4.2 Segmentation & Recognition Experiments .....	37
4.4.3 Reference Object Detection Experiments.....	43
4.4.4 Real-Life Scenario and Results .....	47
5 User Manual .....	48
6 Conclusion and Future Work.....	60
6.1 Conclusion .....	60
6.2 Future Work.....	61
References .....	62

# List of Figures

Figure 1.1: Time Plan.....	2
Figure 2.1: SSD Architecture [23] .....	12
Figure 2.2: Image Recognition Flow [14].....	13
Figure 2.3: Candidate Region Detection Methods [14].....	14
Figure 2.4: Coin as a Reference Object [25] .....	18
Figure 2.5: Proposed Image Taking Strategy [26].....	19
Figure 2.6: International Food Unit Experiment [26] .....	19
Figure 3.1: Three Tier System Architecture.....	20
Figure 3.2: DeepLab V3 Plus Model Architecture [27] .....	22
Figure 3.3: YOLO V5 Model Architecture [28].....	25
Figure 4.1: UECFOOD-100 Dataset Samples [14].....	31
Figure 4.2: UECFOODPIXCOMPLETE Dataset Samples [24].....	32
Figure 4.3: Seefood Model Sample Output.....	37
Figure 4.4: DeepLabV3+ Sample Output Mask.....	39
Figure 4.5: YOLOV5 & DeepLabV3+ Sample Output Recognition and Mask Generation.....	41
Figure 4.6: SIFT Results .....	44
Figure 4.7: ID-Card Segmentation Model Results.....	44
Figure 4.8: Hough Lines Transform Results .....	45
Figure 4.9: Virtual Card Boundaries Method.....	46
Figure 4.10: Contour Analysis Results .....	46
Figure 5.1: Landing Page .....	48
Figure 5.2: Registration with Gmail.....	48
Figure 5.3: Registration with Gmail Process.....	49



Figure 5.4: Registration via Email Process .....	50
Figure 5.5: Login via Email Process .....	50
Figure 5.6: Login via Gmail Process .....	51
Figure 5.7: Adding A Meal.....	52
Figure 5.8: Adding A Meal Options Menu .....	52
Figure 5.9: Capture Photo via Camera Option .....	53
Figure 5.10: Upload Image from Gallery Option .....	54
Figure 5.11: Meal Nutrition Value.....	55
Figure 5.12: Add to Meals Option.....	55
Figure 5.13: User Dashboard Update .....	56
Figure 5.14: Deleting a Meal Process.....	57
Figure 5.15: User Profile Management Process .....	58
Figure 5.16: Save Changes Done to Profile .....	58
Figure 5.17: Settings Option.....	59
Figure 5.18: Application's Dark Theme.....	59

## List of Tables

Table 2.1: Traditional Approaches Results for Food Image Classification [4] .....	8
Table 2.2: Deep learning Approaches Results for Food Image Classification [4].	10
Table 2.3: VGG-16 Results [23] .....	12
Table 2.4: UEC-FoodPix Complete Dataset Results [24] .....	15
Table 2.5: Volume Estimation Approaches [4] .....	17
Table 2.6: Volume Estimation by Reference Object Results [25] .....	18
Table 4.1: DeepLabV3+ Evaluation [32].....	37
Table 4.2: DeepLabV3+ Model Results (Ours vs. [24]) .....	38
Table 4.3: YOLOv8 vs YOLOv5 Results .....	42
Table 4.4: GourmetNet vs. DeepLabv3+ only vs. DeepLabv3+ & YOLOV5 .....	42
Table 4.5: Whole System Results.....	47

## List of Abbreviations

Abbreviation	Description
API	Application Programming Interface.
CNN	Convolution Neural Networks.
CPU	Center Processing Unit
DPM	Deformable parts model (DPM) are a collection of graphical models used for object detection.
HOG	Histogram of Oriented Gradient
JSEG	A segmentation method that was introduced to segment the images with a mixture of both texture and color features.
mIoU	mean Intersection over Union; a model evaluation criteria.
NN	Neural Network
OpenCV	Open-Source Computer Vision
SaaS	Software as a Service.
SIFT	Scale-Invariant Feature Transform
SSD	Single Shot Detector
SSL	Secure Socket Layer
SVM	Support Vector Machine
TOF	The Time-Of-Flight (TOF) principle is a method for measuring the distance between a sensor and an object.
VGG-16	VGG16 refers to the VGG model, also called VGGNet; a convolution neural network (CNN) model supporting 16 layers.
YOLO	You Only Look Once; an end-to-end Neural Network that makes predictions of bounding boxes and class probabilities all at once.

# 1 Introduction

## 1.1 Motivation

In recent years, communities became more aware of health and fitness domain, which reflects on monitoring eating habits and dietary plans. The diets and eating habits can affect health of human beings. In particular, diabetics, allergic people, and so on, should strictly monitor and control their dietary behavior.

Food recognition and classification is an important task that could help human beings record the daily diets. Images of food are one of the most important information to reflect the characteristics of food. Moreover, image sensing is a relatively easy and low-cost information acquisition tool for food appearance analysis. For natural products, like food and processed food, the large variations in food shape, volume, texture, color, and compositions make food type and portion recognition challenging compared to other recognition tasks. Various background and layout of food also introduce variations for food recognition and classification.

At present, due to the common use of Convolutional Neural Networks (CNN), image analysis has been the most commonly used pattern in food recognition and classification which helps in replacing the manual methods in tracking the nutrition plan with an AI model which estimates the volume and calories of each ingredient just by capturing a photo of the dish only using one tool which saves time, money and effort. CalorieMe, the proposed end-to-end solution, is developed to ease the process of monitoring calorie intake and encourage people to keep track of what they eat in an easy and usable way.

## 1.2 Problem Definition

Accurately estimating the total amount of calories and portion sizes of food from phone captured image.

## 1.3 Objective

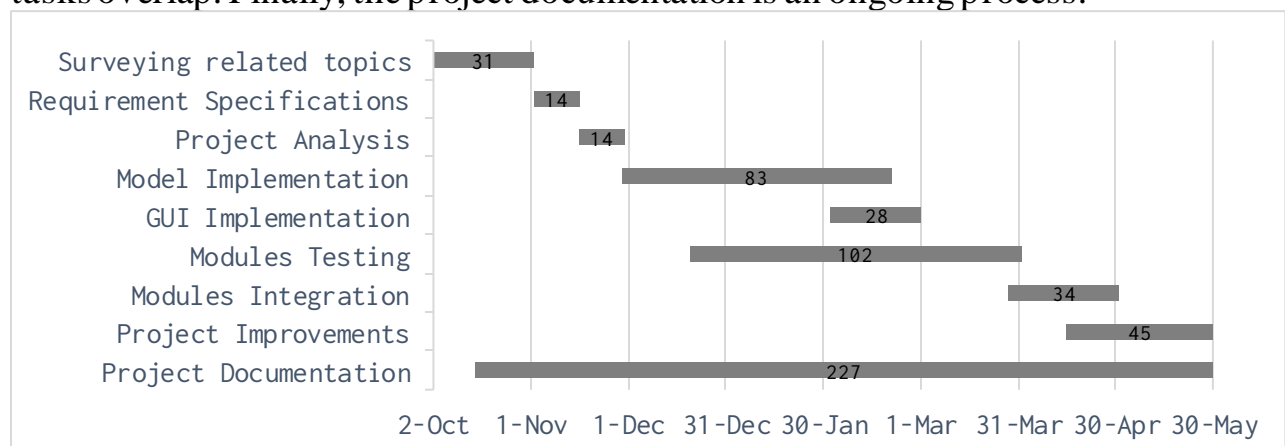
The main objective of the project is to estimate meal calories, this can be achieved by:

- Identifying a given meal ingredients.
- Estimating the quantity of each ingredient.
- Calculating the total meal calories based on each ingredient quantity.

Building an interactive user-friendly platform that encourages users to manage their nutrition plan.

## 1.4 Time Plan

**Figure 1.1** shows the overall project time plan. The time plan manifests the main stages of the project and their corresponding time span. The project process starts with the literature review through until building the final product. As seen, some tasks overlap. Finally, the project documentation is an ongoing process.



**Figure 1.1: Time Plan**

## 1.5 Document Organization

- **Chapter 2:** This chapter discusses the background of the project with respect to its scientific basis and its intended usage field, as well as a brief description of other similar projects.
- **Chapter 3:** This chapter discusses the three-tier system architecture with description to each of its components and the architecture of used deep learning models. The chapter also includes the system's intended users with their characteristics (basic knowledge required for the user to be able to benefit from the project).
- **Chapter 4:** This chapter explains in detail each function, technique and algorithm implemented in the project, as well as testing procedures done.
- **Chapter 5:** This chapter outlines the detailed user's manual on how to use the system with a step by step screenshot guide.
- **Chapter 6:** This chapter concludes this documentation and suggests intended future improvements to be done in order to maximize the project's potential.

## **2 Literature Review**

### **2.1 Project Overview**

With the widespread use of smart phones, many mobile health applications have been launched to help people keep track of what they eat throughout the day and encourage them to follow a healthy lifestyle by facilitating the process of tracking the nutritional value and the quality of food they consume. Existing applications include: MyDietCoach, Yazio, MyFitnessPal, Foodnotes, MyFoodDiary and FatSecret. Such mobile applications, however, require users to manually enter the food types and consumed weight which are tedious and burdensome. Recently, various automated food calorie estimation techniques employing image recognition have been proposed. Mainly to achieve the goal of estimating nutrition value via image recognition, there is a need for a proper dataset to train a proposed model. Methodologies on image recognition have advanced greatly because of the advent of Convolutional Neural Network (CNN). CNN has improved accuracies of various kinds of image recognition tasks such as classification and object detection. This chapter discusses some of the approaches employed to solve the tasks of food image recognition. Also, in order to determine the food portion size without the help of manual input, the approaches used to estimate volume through the image are explained.

## 2.2 Theoretical Background

Computer vision is a field of Artificial Intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs and take actions or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand. Computer vision works much the same as human vision, except humans have a head start. Human sight has the advantage of lifetimes of context to train how to tell objects apart, how far away they are, whether they are moving and whether there is something wrong in an image. Computer vision, on the other hand, needs lots of data. It runs analyses of data over and over until it discerns distinctions and ultimately recognize images. For example, to train a computer to recognize automobile tires, it needs to be fed vast quantities of tire images and tire-related items to learn the differences and recognize a tire, especially one with no defects. Mainly such tasks are carried out via the usage of a CNN which is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data [1].

Additionally, object detection is a computer vision technique responsible for locating instances of objects in images or videos. In object detection, a bounding box is usually used to describe the spatial location of an object. The bounding box is rectangular and is determined by the x and y coordinates of the upper-left corner of the rectangle and the such coordinates of the lower-right corner. The box contains information about the object and coordinates carries information of where it is located in the image. Object detection algorithms typically leverage machine learning or deep learning techniques to produce meaningful results. When humans look at images or video, one can recognize and locate objects of interest within a



matter of moments. The goal of object detection is to replicate this intelligence using a computer.

Object detection is a critical task for the process of tracking the nutritional value through food images. A variety of techniques can be used to perform object detection. Popular deep learning-based approaches using CNNs, such as R-CNN and YOLO v2, automatically learn to detect objects within images. Image recognition and object detection are similar techniques and are often used together. Image recognition identifies which object or scene is in an image, object detection finds instances and locations of those objects in images. A deep learning approach to image recognition can involve the use of a CNN to automatically learn relevant features from sample images and automatically identify those features in new images [2].

Lastly, image segmentation is a commonly used technique in digital image processing and analysis that aims to partition an image into multiple parts or regions, often based on the characteristics of the pixels in the image. Image segmentation could involve separating foreground from background, or clustering regions of pixels based on similarities in color or shape. For example, a common application of image segmentation in medical imaging is to detect and label pixels in an image or voxels of a 3D volume that represent a tumor in a patient's brain or other organs. Using CNNs, a deep learning technique called semantic segmentation allows the association of every pixel of an image with a class label. Applications for semantic segmentation include autonomous driving, industrial inspection, medical imaging, and satellite image analysis and in our case food image segmentation. Popular deep learning-based approaches using CNNs are FCN, U-Net and Mask RCNN [3].

## 2.3 Related Work

This section discusses the commonly used approaches in the food image recognition, segmentation and food portion size estimation fields with their methodologies and results. The following subsections discuss them in detail as subsection 2.3.1 outlines papers related to the food image recognition and segmentation tasks while subsection 2.3.2 discusses papers related to food portion estimation task.

### 2.3.1 Food Image Recognition and Segmentation Tasks

Wen Lo et al. [4], published in their paper a discussion about common approaches utilized in image recognition task as follows.

**1- Conventional Image Recognition Approach with Manually Designed Features** describes the conventional approach to food classification using manually designed features. This approach involves two main tasks: feature extraction and classification. Several commonly used feature extraction techniques are mentioned, including SIFT, HOG, Gabor filter, MR8 filter, and LBP. These techniques can be combined to improve the recognition rate of food classification. The paper also discusses the use of visual words, which are representative descriptors over a set of descriptors in the same cluster, to improve the efficiency of feature extraction. Different linear and non-linear techniques for classification were evaluated, with linear SVM marginally outperforming the others as shown in **Table 2.1**.

*Table 2.1: Traditional Approaches Results for Food Image Classification [4]*

<b>Authors</b>	<b>Method(s)</b>	<b>Dataset</b>	<b>Top 1 Accuracy</b>
<b>Kong et al. [5] 2011</b>	Nearest-neighbor classifier: DoG and SIFT	61-food classes	84.0%
<b>Kong et al. [6] 2011</b>	Nearest-neighbor classifier: SIFT and visual words	5-food classes	92.0%
<b>Kawano et al. [7], 2014</b>	One-vs-rest classifier: Fisher vector with RootHoG	256-food classes	50.1%
<b>He et al. [8] 2014</b>	KNN classifier: DCD, MD-SIFT, SCD and SIFT	42-food classes	64.5%
<b>Tamma et al. [9] 2014</b>	SVM classifier: BoF, SFTA and color histogram	5-food classes	70.0%
<b>Anthim et al. [10] 2014</b>	SVM classifier: BoF, hsvSIFT and color moment invariant	11-food classes	78.0%
<b>Pouladz et al. [11] 2015</b>	SVM classifier: color texture, size, shape features	30-food classes	94.5%
<b>Beijbom et al. [12] 2015</b>	SVM classifier: HoG, SIFT, LBP, MR8 Filter and LLC	50-food classes	77.4%

**2- End-to-end Image Recognition with Deep Learning Approach** describes the use of deep learning for food recognition. Deep learning has shown outstanding performance in various artificial intelligence applications, but has only been considered in a few works for food recognition. The paper discusses the use of convolutional neural networks for dietary monitoring and compares their performance to traditional SVM-based techniques using handcrafted features. The CNNs outperformed the traditional methods by 10%. Google has also proposed deep learning methods for dietary assessment, using a pre-trained GoogLeNet fine-tuned on the Food101 dataset [13]. The results outperformed traditional methods based on handcrafted features and SVM classifiers by 28%. The paper also mentions other research works based on deep learning and concludes that deep learning methods outperform traditional ones in food recognition.

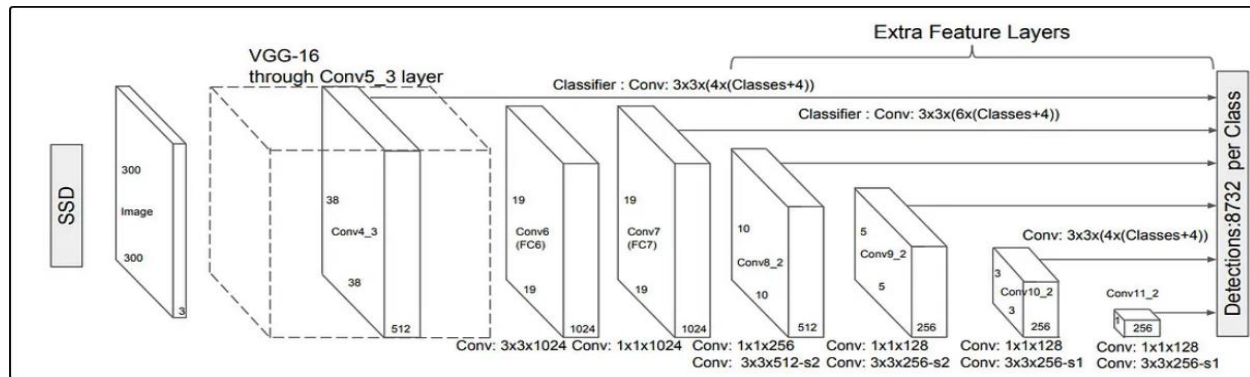
**Table 2.2** shows comprehensive studies that have been carried out on the practicality of deep learning technique and examined different types of networks on several large publicly available datasets including UEC-FOOD100 [14], UEC-FOOD256 [15] and Food101 [13].

*Table 2.2: Deep learning Approaches Results for Food Image Classification [4]*

<b>Authors</b>	<b>Method(s)</b>	<b>Dataset</b>	<b>Top 1 Accuracy</b>
<b>Kagaya et al. [16] 2014</b>	CNN	Own dataset	73.7%
<b>Christ et al. [17] 2015</b>	Patch-wise model + CNN	Own dataset	84.9%
<b>Qui et al. [18] 2019</b>	PAR-NET	Food-101	90.4%
<b>Min et al. [19] 2019</b>	IG-CMAN	Food-101	90.4%
<b>Tan et al. [20] 2019</b>	EffecientNet	Food-101	93.0%
<b>Hassane et al. [21] 2016</b>	Inception V3	UEC-FOOD100	81.5%
<b>Foresti et al. [22] 2018</b>	WISeR	UEC-FOOD100	89.6%
<b>Hassane et al. [21] 2016</b>	Inception V3	UEC-FOOD256	76.2%
<b>Foresti et al. [22] 2018</b>	WISeR	UEC-FOOD256	83.2%

Takumi Ege and Keiji Yanai [23] proposed an approach in their paper under the name of “Simultaneous Estimation of Dish Locations and Calories with Multi-Task Learning” which imposed a multi-task learning technique in recognition of food dish location and food categories using the dataset. Their results showed that their multi-task method achieved higher accuracy, higher speed and smaller network size than a sequential model of food detection and food calorie estimation. They proposed a network for multi-task learning of dish detection and food calorie estimation. They modified the network of SSD based on VGG16 as shown in **Figure 2.1**, so that it can output a food calorie value on each food bounding box. To modify the SSD, the authors carried out multi-task learning of food calorie estimation as well as dish detection. In their proposed method, they added the output layer of estimated food calories as well as bounding boxes and categories so that their network estimates food calories in addition to bounding boxes and categories. Hence, the total number of channels of our output feature map is defined as  $B \times (4 + C + 1)$ . To estimate food calories for each of the estimated bounding boxes, a food image dataset annotated with both bounding boxes and the calorie values of the foods in each of the bounding boxes is required. However, such datasets do not exist at present. Therefore, in this work, they created calorie-annotated multiple-dish food photos with pseudo bounding boxes using the following procedure. Firstly, a food image is prepared as a background image. Secondly, some random size calorie-annotated food images are embedded on the random positions. For smoothing of the boundary, an alpha blending process is used to embed calorie-annotated food images into a background. Then the embedded image region is set as a ground-truth bounding box.

The problem with this paper is that it used an additional manually collected dataset pre-annotated with calories to estimate calories as per serving no matter the amount of food present in the image but their model performed well in the food recognition task.



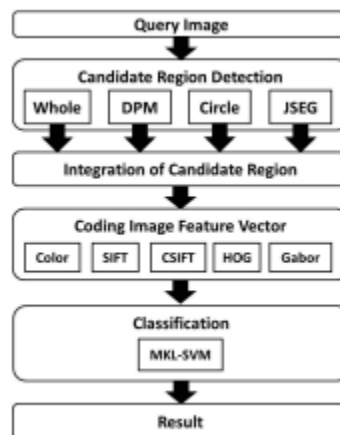
*Figure 2.1: SSD Architecture [23]*

**Table 2.3** shows the results of food calorie estimation from single-dish food photos against multiple-dish food photos. The multiple-dish results see a significant drop in precision against single-dish which highlights the challenge of recognizing food in multiple-dish food scenario.

*Table 2.3: VGG-16 Results [23]*

	<i>Rel.err.(%)</i>	<i>Abs.err.(Kcal)</i>	<i>≤ 20% err.(%)</i>	<i>≤ 40% err.(%)</i>	<i>Precision(%)</i>	<i>Recall(%)</i>
<i>Single-dish (top1)</i>	28.2	106.0	46.7	76.7	100.0	100.0
<i>Single-dish (threshold)</i>	32.7	110.4	43.3	73.3	93.8	100.0
<i>Multiple-dish (threshold)</i>	29.7	292.3	40.0	70.0	85.7	86.4





Another paper published by Matsuda et al. [14], under the title “Recognition of Multiple-Food Images by Detecting Candidate Regions” proposed a two-step method to recognize multiple-food images by detecting candidate regions shown in **Figure 2.2** with several methods and classifying them with various kinds of features. In the first step, they detect several candidate regions by fusing outputs of several region detectors including Felzenszwalb’s Deformable Part Model (DPM), a circle detector and the JSEG region segmentation. In the second step, they apply a feature-fusion-based food recognition method for bounding boxes of the candidate regions with various kinds of visual features including bag-of-features of SIFT and CSIFT with spatial pyramid (SP-BoF), Histogram of oriented Gradient (HoG), and Gabor texture features. In the experiments, they estimated ten food candidates for multiple-food images in the descending order of the confidence scores. As results, they have achieved the 55.8% classification rate, which improved the baseline result in case of using only DPM by 14.3 points, for a multiple-food image data set. This demonstrates that the proposed two-step method is effective for recognition of multiple-food images.



*Figure 2.2: Image Recognition Flow [14]*



**Figure 2.3** compares candidate region detection methods with respect to the number of candidate regions, advantages and disadvantages.

Method				
	<b>Whole</b>	<b>DPM</b>	<b>Circle</b>	<b>JSEG</b>
# of candidates	1	100 (1 for each item)	4 (avg.)	14 (avg.)
Advantage	is suitable for larger food items	detect regions by HoG-based part model	detect dishes by contours of dishes.	detect dishes by segmentation.
Disadvantage	is unsuitable for small food items	is based on only gradient-based features.	dishes are not always circular.	segmentation sometimes fails.

*Figure 2.3: Candidate Region Detection Methods [14]*

Kaimu Okamoto and Keiji Yanai [24] developed and published a paper under the title “UEC-FoodPix Complete: A Large-scale Food Image Segmentation Dataset” where they introduced their own version of image segmentation dataset. It was published as a large-scale food image dataset with a segmentation mask and a modification to the UEC-FoodPix by Ege et al, as Ege’s UEC-FoodPix dataset contained some incomplete segmentation masks on the boundaries of the food regions, since they were generated automatically from the bounding box annotations. Therefore, in this paper, they created “UECFoodPix Complete” as a higher quality dietary image segmentation dataset by updating UECFoodPix manually. They used the Web-based pixel-wise annotation tool implemented by Pongsate et al. This tool allows easy synthesis and separation of food regions with super-pixels. To make the annotation higher quality and more reliable. This dataset showed promising results in food segmentation related tasks.

**Table 2.4** compares Pixel wise accuracy and mean intersection over union carried over the older version of UEC-FoodPix dataset against the complete version with

the complete version outperforming older version by achieving a 66.8% pixel accuracy and a mean intersection over union of 0.555.

*Table 2.4: UEC-FoodPix Complete Dataset Results [24]*

Training Dataset	Acc	mIoU
UEC-FoodPix(all automatic)	0.560	0.416
Partial UEC-FoodPix Complete(2000 hand annotation)	0.597	0.436
UEC-FoodPix Complete(all 9000 hand annotation)	0.668	0.555

### 2.3.2 Food Portion Estimation Task

There were multiple approaches used to estimate the portion size ranging from approaches using depth cameras to others utilizing the usage of multiple captures of different viewpoints of the food to 3D-Model reconstruction. Wen Lo et al. [4], discussed in their paper these approaches and classified them into 5 categories which are:

**1-Stereo-based approach:** Stereo-based approach refers to using multiple frames to reconstruct the 3D structure of food objects by finding pixel correspondences between image frames and using the extrinsic parameters to reproject the pixels from image coordinate to world coordinate.

**2-Model-based approach:** Model-based approach refers to pre-building shape templates (mathematical models) so that the volume of objects can be determined by model selection followed with model scaling and rotation, which is also known as image registration.

**3-Depth camera-based approach:** Apart from monocular cameras, other visual sensors are involved in dietary assessment. The most commonly used sensor is the

depth camera such as Time Of Flight (TOF) camera. In using depth camera-based approach, the actual scale of object items can be obtained without any reference object such as a fiducial marker.

**4-Perspective transformation approach:** This refers to the method of estimating object volume based on a single image. By using perspective transformation, a bird's eye view image can be obtained and a rough estimate on the size of the object can be derived. This method does not rely on pre-built shape templates, thus it is typically used to estimate objects with irregular shapes.

**5-Deep learning approach:** Deep neural networks have been extensively used in volume estimation. Several research works proposed using a single RGB image to infer the depth map. Voxel representation has been used to present the depth map and the volume can be estimated by counting the number of voxels occupied. Recently, research explored the use of point cloud completion to achieve volume estimation. Furthermore, the advantages and challenges for the different approaches are also highlighted in the paper. To evaluate the performance of dietary assessment, various food datasets have been constructed by different research groups, however, most of the publicly known datasets are constructed to examine the performance of food classification only, instead of food volume estimation. To examine the accuracy of volume estimation, authors tend to construct their own databases, which is relatively small in scale compared to the publicly known one. Thus, it seems important to keep in mind which database is used in each case when comparing results. **Table 2.5** summarizes the five volume estimation approaches by stating required preparation for each approach and the outline of the carried out procedure.

*Table 2.5: Volume Estimation Approaches [4]*

Volume Estimation					
Method	Stereo-based	Model-based	Perspective transformation	Depth camera based	Deep learning
<b>Preparation</b>	Camera calibration (Intrinsic matrix)	Model library construction (Pre-build 3D shape models)	Strong constraints required (Camera viewing angle)	Depth camera calibration (Intrinsic matrix)	Model training
<b>Procedure</b>	1. Camera calibration (Extrinsic matrix) 2. 3D model reconstruction 3. 3D meshing and scale determination 4. Volume estimation	1. Food recognition 2. Model selection from the library 3. Model registration by rotating and scaling 4. Volume determination by pre-build models	1. Surface plane equation fitting 2. Perspective transformation 3. Scale determination 4. Volume calculation by geometric relationships	1. Depth map construction 2. 3D model reconstruction based on point cloud or voxel representation 3. Volume estimation	1. Single RGB image captured 2. Depth map estimation/ 3D Shape completion 3. 3D model reconstruction 4. Volume estimation

Kadam et al. [25] published a paper under the title of “FVEstimator: A novel food volume estimator Wellness model for calorie measurement and healthy living”. They utilized the usage of fiducial marker (Reference object) which is an object of known and standard dimensions put in frame with the food while image is being captured (e.g. a checkerboard card/a coin/a credit card) as shown in **Figure 2.4**, the paper proposed a coin as reference object in order to estimate food portion size. **Table 2.6** Shows volume estimation results against actual volume across different food shape types with a drop in accuracy in scenario 1 and 2 as amorphous and convex food shapes volumes are harder to estimate as they take the shape of their container and that could be much trickier to estimate.

*Table 2.6: Volume Estimation by Reference Object Results [25]*

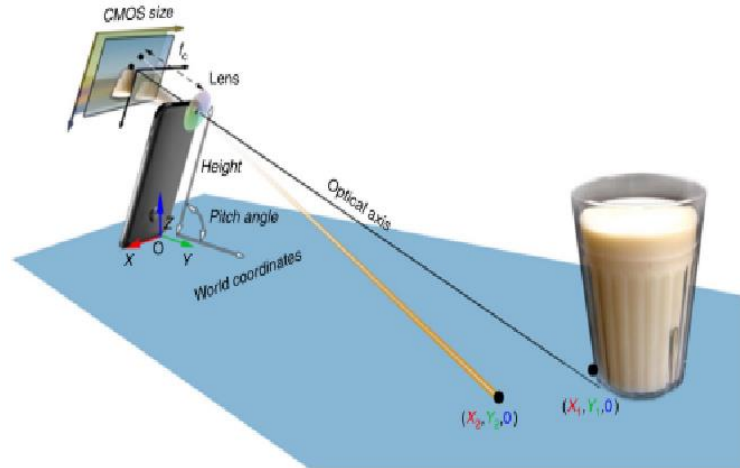
Volume estimation				
Scenario	Shape of food item	Actual volume [ $Vol_A$ ]	Estimated volume [ $Vol_g$ ]	Accuracy
1	Amorphous	270.615	299.154	90.46%
2	Convex	441.036	485.14	90.9%
3	Regular(square)	130	132.1	98.5%
4	Regular(circle)	79.0321	78.125	98.9%



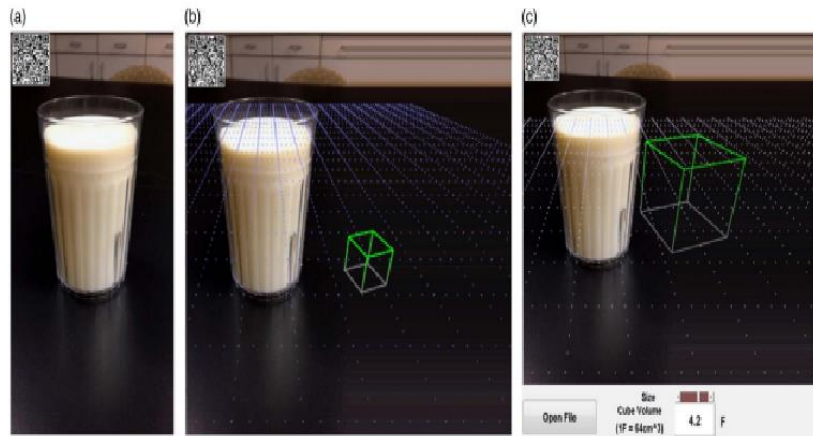
*Figure 2.4: Coin as a Reference Object [25]*

“Image-based food portion size estimation using a smartphone without a fiducial marker” paper published by Yang et al. [26], proposed a different approach to replace the usage of fiducial marker, The authors presented a new method for food volume estimation without a fiducial marker. Their mathematical model indicates that, using a special picture-taking strategy shown in **Figure 2.5**, the smartphone-based imaging system can be calibrated adequately if the physical length of the

smartphone and the output of the motion sensor within the device are known. They also presented and tested a new virtual reality method for food volume estimation using the International Food Unit™ shown in **Figure 2.6** and a training process for error control.



*Figure 2.5: Proposed Image Taking Strategy [26]*

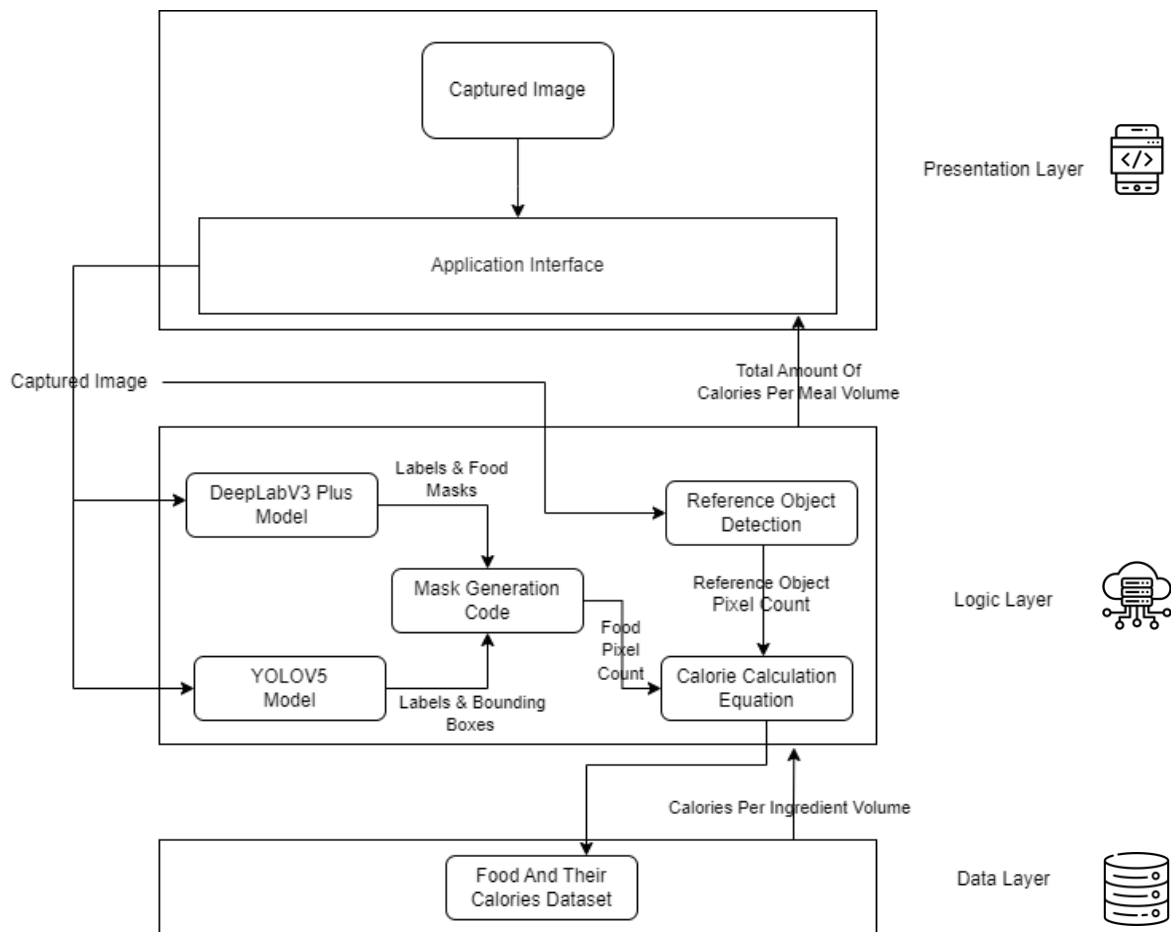


*Figure 2.6: International Food Unit Experiment [26]*

## 3 System Architecture and Methods

### 3.1 System Architecture

**Figure 3.1** visualizes the system architecture of the project. The system architecture is composed of three layers. The three-tier architecture is divided into Presentation, Logic and Data layers. Section 3.2 discusses each layer in detail.



*Figure 3.1: Three Tier System Architecture*

## 3.2 Description of Methods and Procedures

### 1. Presentation Layer

User Interface and communication layer of the application where the end-user interacts with the application.

#### 1.1 Application Interface

It works on collecting the image captured by the user and sending it to logic layer (server) to process the image. It's also works on retrieving and displaying the data after being processed. OpenSSL is used to generate self-signed SSL certificate to encrypt the connection between the application and the server to assure security of user's data.

### 2. Logic Layer (Application Layer)

Information collected from presentation layer is processed by the included models to apply the segmentation, recognition, and calorie estimation process and prepare it for comparison against other information in the data layer.

It takes the image as an input and produces a food estimate volume and food label to compare against the data layer.

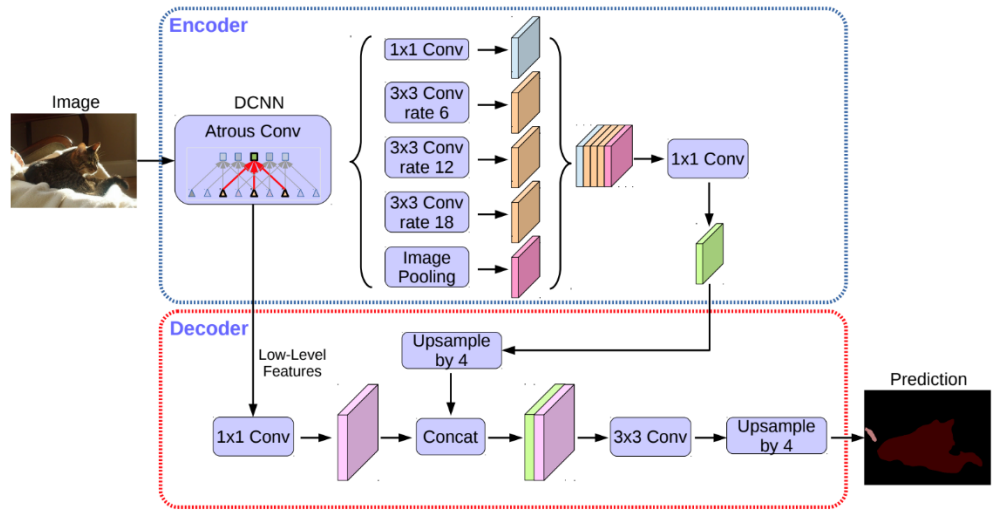
#### 2.1 DeepLabV3Plus

**Figure 3.2** shows DeepLab V3 Plus model architecture. DeepLab V3 Plus is a state-of-the-art architecture for multi-class semantic segmentation. It extends DeepLab V3 by adding an encoder-decoder structure. The encoder module processes multiscale contextual



information by applying dilated convolution at multiple scales, while the decoder module refines the segmentation results along object boundaries. DeepLab V3 Plus is an improved version of the DeepLab V3 architecture. It uses an encoder-decoder structure to improve the segmentation results. The encoder module processes multiscale contextual information by applying dilated convolution at multiple scales. The decoder module refines the segmentation results along object boundaries.

The architecture is designed to perform well on semantic segmentation benchmarks.



*Figure 3.2: DeepLab V3 Plus Model Architecture [27]*

## 2.2 YOLOv5

Object detection is a computer vision task that involves identifying and locating objects in images or videos. It is an important part of many applications, such as surveillance, self-driving cars, or robotics. Object detection algorithms can be divided into two main categories: single-shot detectors and two-stage detectors. The two categories are based on how many times the same input image is passed through a network. Single-shot object detection uses a single pass of the input image to make predictions about the presence and location of objects in the image. It processes an entire image in a single pass, making them computationally efficient.

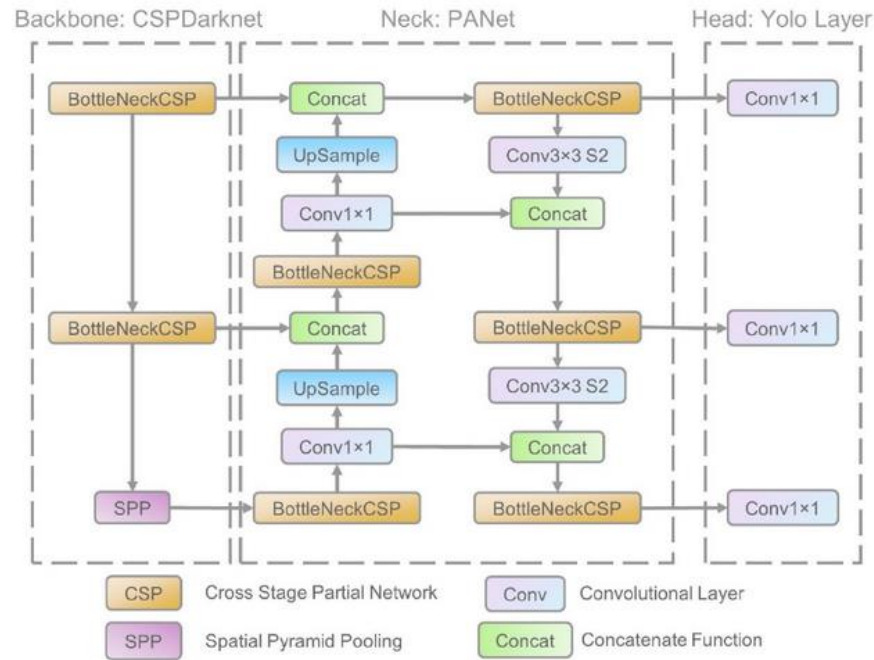
You Only Look Once (YOLO) proposes using an end-to-end neural network that makes predictions of bounding boxes and class probabilities all at once. It differs from the approach taken by previous object detection algorithms, which repurposed classifiers to perform detection. Following a fundamentally different approach to object detection, YOLO achieved state-of-the-art results, beating other real-time object detection algorithms by a large margin. While algorithms like Faster RCNN work by detecting possible regions of interest using the Region Proposal Network and then performing recognition on those regions separately, YOLO performs all of its predictions with the help of a single fully connected layer.

YOLO is a single-shot detector that uses a fully (CNN) to process an image. Methods that use Region Proposal Networks perform multiple iterations for the same image, while YOLO gets away with a single iteration. The YOLO algorithm takes an image as input and then uses a

simple deep convolutional neural network to detect objects in the image.

**Figure 3.3** shows the YOLOv5 model architecture, YOLOv5 is a model in the YOLO family of computer vision models. YOLOv5 is commonly used for detecting objects. YOLOv5 comes in four main versions: small (s), medium (m), large (l), and extra-large (x), each offering progressively higher accuracy rates. Each variant also takes a different amount of time to train. The YOLOv5 repository is a natural extension of the YOLOv3 PyTorch repository by Glenn Jocher. The YOLOv3 PyTorch repository was a popular destination for developers to port YOLOv3 Darknet weights to PyTorch and then move forward to production. After fully replicating the model architecture and training procedure of YOLOv3, Ultralytics began to make research improvements alongside repository design changes with the goal of empowering thousands of developers to train and deploy their own custom object detectors to detect any object in the world.

In conclusion, Ultralytics YOLOv5 is a cutting-edge, State-Of-The-Art (SOTA) model that builds upon the success of previous YOLO versions and introduces new features and improvements to further boost performance and flexibility. YOLOv5 is designed to be fast, accurate, and easy to use, making it an excellent choice for a wide range of object detection, instance segmentation and image classification tasks.



*Figure 3.3: YOLO V5 Model Architecture [28]*

## 2.3 Mask Generation

The two models were integrated: YOLOv5 for object recognition and DeepLabV3+ for precise segmentation mask generation. This collaborative integration allowed us to leverage the unique strengths of each model. The YOLOv5 model demonstrated proficiency in identifying food classes, while the DeepLabV3+ model excelled in generating accurate segmentation masks.

The output generated by YOLOv5m was combined with the image produced by DeepLabV3+ and generate the mask.

## 2.4 Reference Object Detection

To extract the ID card, Contour analysis is applied, which is a fundamental technique in image processing and computer vision used to extract and analyse the boundaries of objects or regions within an

image. A contour is a curve that represents the outline of an object or the boundaries of a region with similar pixel intensity or colour. Contour analysis involves preparing the image by applying various techniques like thresholding, edge detection, morphological operations to enhance the regions of interest and suppress unwanted details or noise.

- Edge Detection

1. Canny

2. Sobel

- Morphological Operations (Dilation)

This process expands the boundaries of objects in an image by adding pixels to the object regions. It involves a sliding window called a structuring element that examines each pixel in the image and replaces it with the maximum value within the corresponding neighbourhood defined by the structuring element. Dilation is useful for tasks like filling gaps, joining disconnected components, and enlarging objects.

- Gaussian Blur which is used to reduce image noise and smooth out details. It is named after the Gaussian function, also known as the bell curve, which is used as a weighting function in the blurring process.

The steps of extracting the reference objects goes as following:

- 1- Convert the image to HSV colour form which is a representation based on three components: Hue (H), Saturation (S), and Value (V).

These components are combined to describe a particular colour in the HSV colour space.

- Hue (H): represents the dominant wavelength of the colour.
- Saturation (S): Saturation represents the intensity or purity of the colour.
- Value (V): Value represents the perceived brightness or intensity of the colour.

2- Apply Gaussian Blur on the S channel of image.

3- Apply Sobel edge detection on the output from Gaussian Blur.

4- Apply Dilation.

5- Apply series of threshold values to the grayscale channel of the row image and processed one we apply the Contour analysis and find the contours which are close the area and angles of the reference object (ID Card).

6- Get the mask of the largest Contours, which is the ID card.

## 2.5 Volume Calculation Equation

Assuming that the volume is directly correlated to the weight and the weight is directly correlated to the size of surface of the ingredient [29], The real size of food regions is estimated based on the size of the reference object. Since the real size of reference object is known, the real size of foods,  $F_r$ , can be obtained by the following equation:

$$F_r = S_r \times \frac{F_p}{S_p}$$

where  $F_p$  represents the number of pixels of the region of the target food item,  $S_p$  represents the number of pixels of the region of the

reference object, and  $S_r$  represents the real size of the top view of the reference object which is expected to be registered in advance [29].

### **3. Data Layer**

Data access layer or back-end, is where the information processed by the application is stored and managed. The users' data (username, password, history, etc, are stored using Firebase. The main data in the data layer, is the one related to food data as explained in Section 3.1.

It takes the food estimate volume and food label as an input and returns the nutritional value per estimated portion size.

#### **3.1 Food And Their Calories [30]**

The dataset contains a csv file with more than 300 foods each with the amount of Calories, Fats, Proteins, Saturated Fats, Carbohydrates, Fibers labelled for each food. The ingredient label & volume are the field of interest; where they are compared against the dataset and return the calorie and macros count per ingredient volume.

### 3.3 System Users

#### *A. Intended Users:*

The traditional way to calculate the calories is tedious way and it takes a lot of time and calculations. CalorieMe is made to make this process easier for the intended users who are:

1. **Fitness enthusiasts:** as fitness enthusiasts' main goal is to stay in shape and keep track of what they eat during the day. CalorieMe will speed up this process.
2. **People who suffer from chronic diseases:** People who suffer from chronic diseases such as heart diseases, diabetes, .. etc must keep track of their diet and their macro nutrient portions (fats/carbohydrates) to ensure their safety. CalorieMe should help them to monitor their health.
3. **Casual users:** are people who aim to keep track of their calories intake throughout the day to keep a healthy lifestyle. Calorie me will encourage them to do so.

#### *B. User Characteristics*

1. Basic knowledge of what is meant by calorie intake and macro nutrients (proteins/fats/carbohydrates).
2. Basic knowledge of smart phone use.



## 4 System Implementation and Results

### 4.1 Datasets

Three datasets are used to train and evaluate the different models implemented in this project. Each of the datasets is explained below, outlining the data format, number of entries, etc. The datasets are: UECFOOD-100 [14], UECFOODPIXCOMPLETE [24], and Food and their calories [30].

**UECFOOD-100** [14]: 100-kind food dataset

The dataset "UECFOOD-100" contains 100-kind food photos including 11561 images for training and 1417 images for testing. Each food photo has a bounding box indicating the location of the food item in the photo. Most of the food categories in this dataset are popular foods in Japan. Therefore, some categories might not be familiar with other nationalities than Japanese. This dataset was built to implement a practical food recognition system which is intended to be used in Japan. [Figure 4.1](#) shows sample images from the UECFOOD-100 dataset.



*Figure 4.1: UECFOOD-100 Dataset Samples [14]*

This project's recognition task involves identifying and locating the food items in the images. To achieve this goal, we trained a YOLO v5 model on this dataset, which contains various food categories and annotations.

#### **UECFOODPIXCOMPLETE [24]: 100-kind of food dataset**

UECFOODPIXCOMPLETE is a food images dataset with segmentation masks including 9,000 images for training and 1,000 image for testing. The segmentation masks are augmented by food category. In UECFOODPIXCOMPLETE there is provided manually. The mask images have pixel-wise food 103 class labels, and only R (red) channel have these labels. Some sample images from the dataset are shown in **Figure 4.2**.



*Figure 4.2: UECFOODPIXCOMPLETE Dataset Samples [24]*

The segmentation task, performed in this project, requires extracting and labeling the pixels of the food items in the images. To achieve this goal, a DeepLabv3+ model is trained on this dataset, which provides pixel-level annotations for various food categories.

**Food and their calories (Extended Version):** this dataset contains a CSV file with 300+ labeled foods and their Calories values.

We manually extended this dataset to match the categories found in UEC-FOOD 100 and UECFOODPIXCOMPLETE which we used in our project.

## 4.2 Software Tools Used

**Flutter:** Flutter is an open-source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase which used in building the user interface.

**Python:** Python is a programming language that lets you work quickly and integrate systems more effectively. In the project, we leverage certain frameworks and applications of Python, such as:

- TensorFlow: Used in building segmentation model (DeepLapV3+).
- PyTorch: Used in building recognition model (YOLOv5).
- Flask: Used in building REST API for the application.

**Azure Virtual Machine:** Azure is a cloud computing platform run by Microsoft, which offers access, management, and development of applications and services through global data centres. It provides a range of capabilities, including Software as a Service (SaaS) which is used as a host for the API.

**Kaggle:** Kaggle is a machine learning and data science community that provides code, data, notebooks, courses, and competitions for data enthusiasts which used in training the models. The main reason why Kaggle was used is that the model development environment is that it provided the resources needed to process a huge amount of data especially the GPU resources it offered as our project's data consisted of images which was difficult to do on standard PC GPUs.

### 4.3 Setup Configuration (Software)

**Git:** Git is a distributed version control system used for tracking changes in source code during software development. It allows multiple developers to collaborate on a project, manage different versions of the code, and facilitate code merging.

**Flask:** Flask is a popular web framework in Python used for building web applications. It provides a simple and flexible way to handle HTTP requests, define routes, and render templates.

**Pandas:** Pandas is a powerful library for data manipulation and analysis in Python. It provides data structures and functions for efficiently handling and analyzing structured data, particularly tabular data.

**Fuzzywuzzy:** Fuzzywuzzy is a Python library for fuzzy string matching and similarity calculations. It offers various algorithms for comparing and matching strings based on their similarity, even when dealing with partial or incorrect matches.

**Numpy:** Numpy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a wide range of mathematical functions to operate on these arrays efficiently.

**Python-Levenshtein:** Python-Levenshtein is a library that implements the Levenshtein distance algorithm. It is used for calculating the minimum number of single-character edits (insertions, deletions, and substitutions) required to transform one string into another.

**Tensorflow-cpu:** Tensorflow-cpu is a version of the TensorFlow library optimized for running on Central Processing Units (CPUs). TensorFlow is an open-source machine learning framework that enables the creation and training of various types of neural networks.

**Opencv-python-headless:** Open Source Computer Vision (OpenCV) is a library used for computer vision and image processing tasks. The opencv-python-headless package provides a headless version of OpenCV that does not require a Graphical User Interface (GUI) and is typically used in server environments.

**Keras:** Keras is a high-level neural networks API that runs on top of TensorFlow (or other backend engines). It provides an easy-to-use interface for building and training neural networks.

**OpenSSL:** OpenSSL is an open-source library that provides cryptographic functions and tools. It includes implementations of various encryption, hashing, and secure communication protocols.

**Systemctl:** Systemctl is a command-line tool used to manage and control system services in Linux-based operating systems. It allows you to start, stop, restart, enable, disable, and check the status of services.

**Daemon:** A daemon is a background process that runs independently of the user and does not require direct interaction. In software development, a daemon often refers to a program that runs as a service, providing certain functionalities or performing tasks in the background.

## 4.4 Experiments and Results

### 4.4.1 Dataset Selection

**FoodSeg 103:** FoodSeg103 [31] is a new food image dataset containing 7,118 images. Images are annotated with 104 ingredient classes and each image has an average of 6 ingredient labels and pixel-wise masks. It's provided as a large-scale benchmark for food image segmentation.

This dataset was rejected because the classification label was not provided.

#### **UECFood-100:** 100-kind food dataset

The dataset "UECFood-100" contains 100-kind food photos including 11561 images for training and 1417 images for testing. Each food photo has a bounding box indicating the location of the food item in the photo. It is used to develop the YOLO v5 recognition model mentioned in detail in **section 4.4.2**.

#### **UECFoodPIXCOMPLETE:** 100-kind of food dataset

UECFoodPIXCOMPLETE is a food images dataset with segmentation masks including 9,000 images for training and 1,000 image for testing. It was used to develop DeepLab v3+ semantic segmentation model mentioned in detail in **section 4.4.2**.

**Food and their calories (Extended Version):** this dataset contains a CSV file with 300+ labeled foods and their Calories values. It is used in the volume estimation part mentioned in **section 3.1**.



#### 4.4.2 Segmentation & Recognition Experiments

**Firstly**, the pretrained model called "Seefood" was tested. The model was pretrained and evaluated on a Google-internal food ingredient parsing dataset.

The model was rejected because it has very small variety of food categories (26 category), and the categories were vague (i.e., too general). For example, the model classifies both apple and orange as one category (Fruits).



*Figure 4.3: Seefood Model Sample Output*

**Secondly**, a DeepLabV3+ model was constructed for the purpose of food identification and segmentation, inspired by the approach outlined in the research paper [32]. Our model exhibited promising results for this particular task, as demonstrated in **Table 4.1**:

*Table 4.1: DeepLabV3+ Evaluation [32]*

<i>Method</i>	<i>mIoU (%)</i>
<i>UEC Foodpix</i>	55.55%
<i>DeepLabV3+</i>	61.54%



The reported results in the paper [32] are based on evaluations conducted using the UECFoodPixComplete dataset.

Also referring to Kaimu Okamoto and Keiji Yanai's [24] paper where they released the UEC-Foodpix Complete dataset, they evaluated it using the state-of-the-art semantic segmentation method, DeepLab V3+ as proposed in this work. As shown in **Table 4.2**, the implemented model is able to achieve a higher pixel accuracy over that of Okamoto's and Yanai's which was crucial in the portion size estimation stage. However, the proposed model came behind in the mean intersection over union criteria which affected the model's ability to correctly classify the food categories. Another approach stood out that mended this flaw, which is what we finally decided to use. This is discussed later in this section.

*Table 4.2: DeepLabV3+ Model Results (Ours vs. [24])*

<i>Model</i>	<i>Acc (%)</i>	<i>mIOU (%)</i>
<i>DeepLabV3+ ([24])</i>	66.8%	<b>55.55%</b>
<i>DeepLabV3+ (Ours)</i>	<b>74%</b>	45%

The DeepLabV3+ model was trained on UECFoodPixComplete dataset using the parameters:

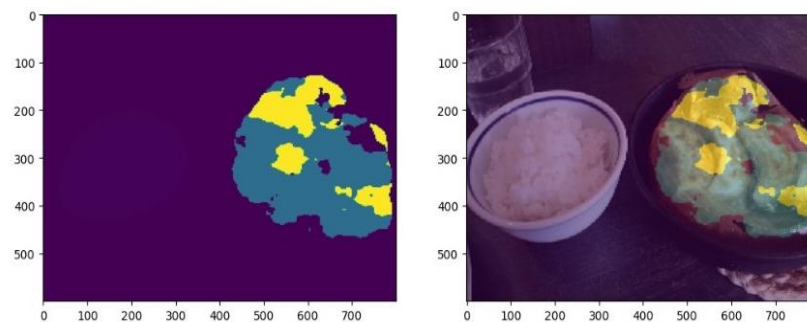
- Image size set to 256
- Batch size of 16
- SGD optimizer
- Cross-entropy loss function
- Learning rate of 0.001

- Momentum of 0.9
- Weight decay of 0.0005
- ResNet101 as a backbone for 100 epochs

the model yielded a pixel accuracy 74% and mIOU result of only 45% on the test data from UECFoodPixComplete.

This performance fell short of the desired result due to the challenges posed by the dataset's heterogeneity and diverse nature, underscoring the difficulties faced in accurately segmenting and identifying the food items.

The model exhibited accurate performance in drawing the segmentation mask; however, it encountered challenges in accurately classifying the food category as shown in **Figure 4.4** where the mask gives a respectable coverage to the food present in the image but the variance in colors indicates different classes which are not present.



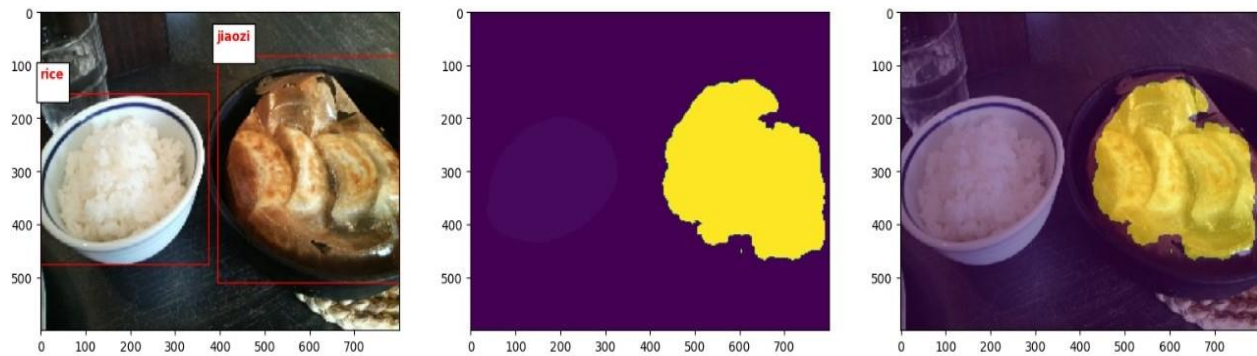
*Figure 4.4: DeepLabV3+ Sample Output Mask*

**Thirdly**, a collaborative approach was adopted to address the challenges faced by the DeepLabV3+ model in accurately determining the class of food images. Inspired by the methodology presented in the research paper [29], two models were integrated: YOLOv5 for object recognition and DeepLabV3+ for precise segmentation mask generation. This collaborative integration allowed us to leverage the unique strengths of each model. The YOLOv5 model demonstrated proficiency in identifying food classes, while the DeepLabV3+ model excelled in generating accurate segmentation masks. The combination of these models resulted in significant improvements to the system's overall performance, effectively overcoming the limitations encountered previously.

The YOLOv5m model was trained on the UEC-Food100 dataset, which is the recognition version of the UECFoodPixComplete dataset. The training process utilized the following parameters:

- 130 epochs
- Batch size of 64
- Image size of 413
- SGD optimizer

After the training, the integration of the two models was performed. The output generated by YOLOv5m was combined with the image produced by DeepLabV3+. After integrating the outputs of YOLOv5m and DeepLabV3+ models, the combined model achieved a high Mean IoU (mIOU) reaching 79.25%. This significant improvement in performance signifies the effectiveness of leveraging the strengths of both models to enhance the accuracy and quality of the results as shown in **Figure 4.5**.



*Figure 4.5: YOLOV5 & DeepLabV3+ Sample Output Recognition and Mask Generation*

Despite the satisfying results achieved by the combined YOLOv5m and DeepLabV3+ model, attempts to upgrade the food recognition component to the newer YOLOv8 version did not yield favorable outcomes as it resulted in a lower classification precision. The trial with YOLOv8 was rejected as it produced inferior results compared to YOLOv5. The decision was made to retain the successful YOLOv5 model for food recognition, considering its superior performance over the newer version.

**Table 4.3** Shows the results of both YOLOv8 and YOLOv5 models which shows that the later outperformed the v8 scoring a higher precision score of 82% vs 69.7%.

**Table 4.3: YOLOv8 vs YOLOv5 Results**

	<b>MAP50 (%)</b>	<b>MAP50-95 (%)</b>	<b>Precision (%)</b>	<b>Recall (%)</b>
<b>YOLOv5</b>	<b>86.9%</b>	<b>68.6%</b>	<b>82%</b>	<b>79.3%</b>
<b>YOLOv8</b>	72.5%	56.365%	69.7%	69.449%

**Table 4.4** shows a comparison between the related work models and our work, both before and after the improvement. Based on the evaluation conducted using the UECFoodPixComplete dataset, a significant improvement in mean intersection over union criteria for the two models approach can be observed, thus the improved ability of the proposed model to correctly classify food categories.

**Table 4.4: GourmetNet vs. DeepLabv3+ only vs. DeepLabv3+ & YOLOV5**

<b>Method</b>	<b>mIoU</b>
<b>Gourmetnet [32]</b>	65.13%
<b>DeepLabV3+ [ours]</b>	45%
<b>DeepLabV3+ &amp; YOLOV5 [ours]</b>	<b>79.25%</b>

#### 4.4.3 Reference Object Detection Experiments

Following the detection and segmentation of a given food image, comes the volume estimation. To accurately achieve this, a reference object detection is needed as it is a main component in the proposed volume estimation method and to yield acceptable results there must exist an efficient yet accurate technique to achieve the reference object detection and segmentation, in the following section discusses some of the proposed reference object detection and segmentation techniques.

**SIFT:** SIFT stands for Scale-Invariant Feature Transform. It is a computer vision algorithm that can detect and describe local features or key points in images. These features are invariant to changes in scale, rotation, and affine transformations. SIFT can be used for object detection by matching the key points of an object in a reference image to the key points of the same object in a test image. SIFT uses a four-stage process to detect and describe key points: scale-space extrema detection, key point localization, orientation assignment, and key point descriptor.

SIFT failed to detect the reference object as shown **Figure 4.6** because there are not enough key points detected in the ID card segmentation. Therefore, this algorithm could not be able to find a sufficient number of matches to establish a reliable correspondence.



*Figure 4.6: SIFT Results*

**ID-Card-Segmentation Model:** Pretrained segmentation of ID Cards model using U-Net. However, this pre-trained model was sensitive to the surrounding objects and could not detect the right id card and the mask was not satisfying for the desired result as shown in [Figure 4.7](#).



*Figure 4.7: ID-Card Segmentation Model Results*

**Hough Lines Transform:** The Hough Lines Transform is a method to detect straight lines in an image. It works by transforming each point in the image to a sinusoid in a parameter space of line equations. The intersections of these sinusoids correspond to potential lines in the image. The more points that lie on a line, the

more sinusoids intersect at that point. The Hough Lines Transform can be used for object detection by finding the lines that form the boundaries or contours of an object. There are two types of Hough Lines Transform: the standard and the probabilistic. The standard one returns the parameters of the lines, while the probabilistic one returns the endpoints of the lines [33].

This method could not get the right lines. As shown in **Figure 4.8**, it can get the id card trying different combinations of thresholding and Noise reduction and enhancement methods including Gaussian Blur, Median Blur and Morphological Operations.



*Figure 4.8: Hough Lines Transform Results*

**Virtual Card Boundaries:** The main idea is the presence of virtual box drawn in the bottom third of the user's view as shown in **Figure 4.9** where the user is requested to manually adjust the placement of the reference object within the boundaries of that box. It was a successful trial but prone to human error, as the user had to manually adjust reference object placement with respect to the viewfinder boundaries.





*Figure 4.9: Virtual Card Boundaries Method*

**Contour Analysis:** In this method, the aim is to smooth the image with a Gaussian blur, detect edges with a Canny algorithm, find contours with four sides and a certain area and convexity, check the angle between the sides to identify rectangles, and return a list of detected rectangles. This experiment was successful and promising results were achieved as shown in **Figure 4.10**.



*Figure 4.10: Contour Analysis Results*

#### 4.4.4 Real-Life Scenario and Results

After the segmentation and recognition processes, a calorie estimation technique was employed to estimate the calorie content of the identified food (as mentioned in Chapter 3).

To assess the system's performance in real-life scenarios, a test was conducted using various dishes. The results revealed an average error of 36% in terms of calorie estimation. Although some deviation was observed, it is important to note that accurately estimating the calorie content of food can be challenging due to variations in portion sizes, cooking methods, and ingredient composition. Despite the margin of error, the system demonstrates promising potential in providing useful insights into the approximate calorie content of different dishes.

**Table 4.5** Shows the results of the system performance and results on real life scenarios.

*Table 4.5: Whole System Results*

<i><b>Food Name</b></i>	<i><b>Real calorific value (Kcal)</b></i>	<i><b>Estimated calories (Kcal)</b></i>	<i><b>Error ( Predicted-Actual )</b></i>	<i><b>Error(%)</b></i>
<i>French fries</i>	277	316	39	14%
<i>Rice</i>	198	200	2	1%
<i>Egg sunny side up</i>	90	158	68	75%

To sum-up, these are the techniques used for each of the stages:

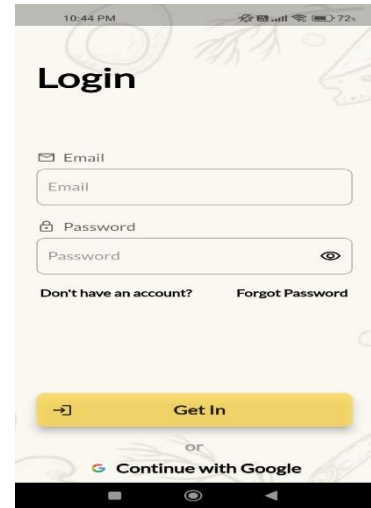
- 1. YOLO v5(m) & DeepLab v3+** for the recognition and segmentation stages.
- 2. Contour Analysis** for the reference object detection and segmentation stage.

# 5 User Manual

This chapter is a walkthrough of the whole application.

- **Landing Page**

- This is the first page to be shown after launching the mobile application as shown in **Figure 5.1**.
- It contains fields for user to enter his credentials to log in to the application.
- User can also log in via Gmail directly.
- It also contains an option for Registering a new user either via Gmail or entering his/her email.

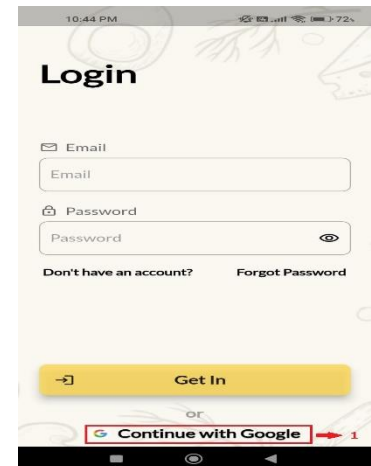


*Figure 5.1: Landing Page*

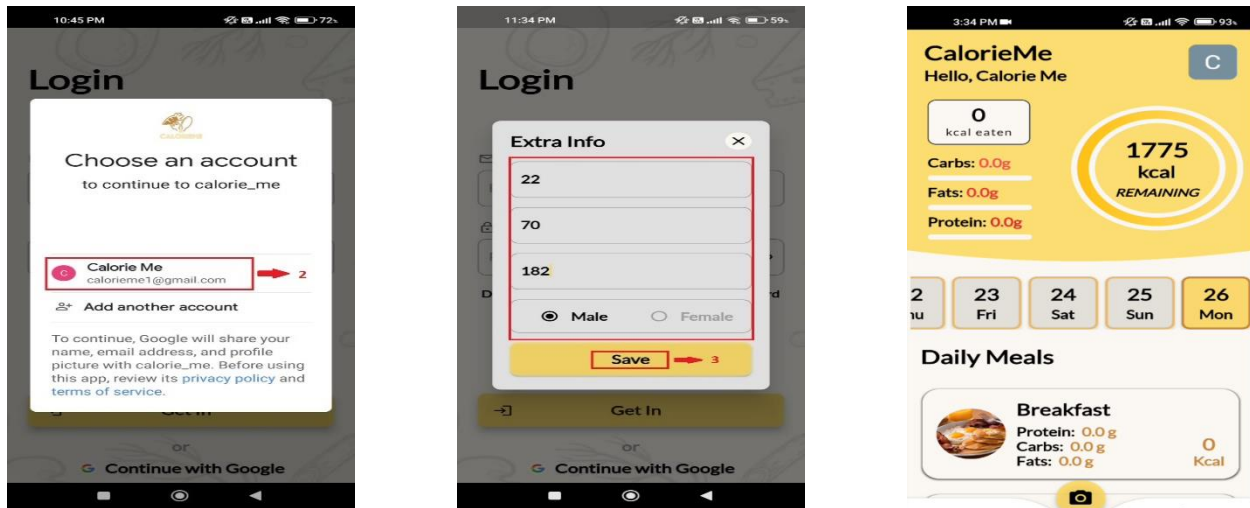
- **User Registration**

1. **Registration via Gmail**

- 1.1 User taps on the option highlighted in red shown as **no.1** in **Figure 5.2**.
- 1.2 A window will pop up asking to continue via Google account registered on smart phone as shown as **no.2** in **Figure 5.3**.
- 1.3 Another window will show up requiring additional information critical for the app like age, height, weight and gender as shown as **no.3** in **Figure 5.3**.
- 1.4 Finally, the user will be directed to the dashboard/homepage as shown on the last image in **Figure 5.3**.



*Figure 5.2: Registration with Gmail*



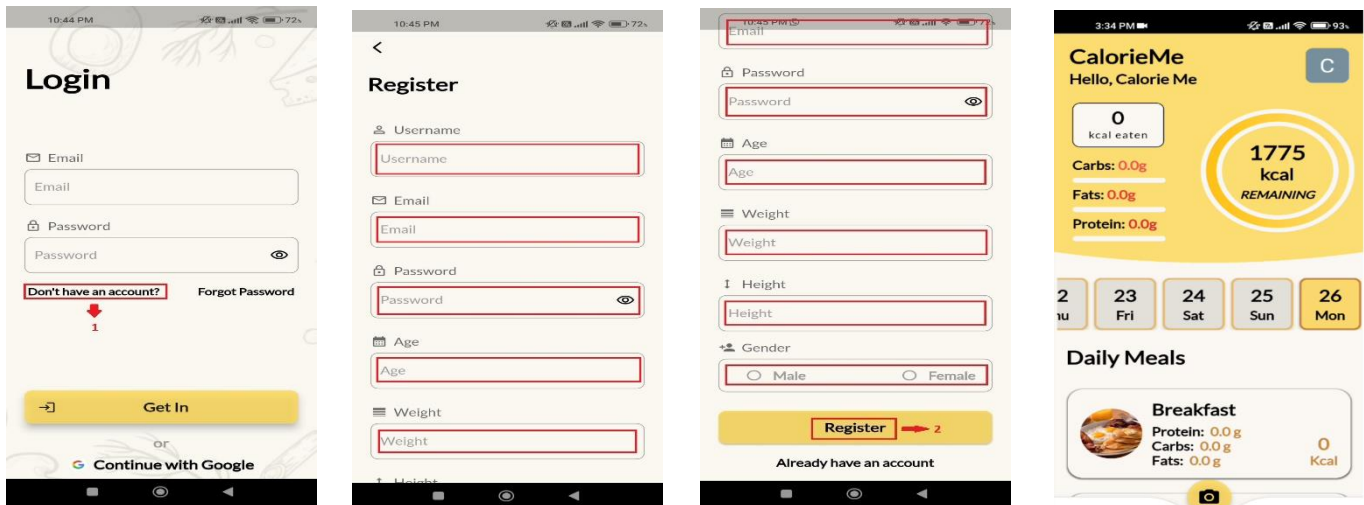
*Figure 5.3: Registration with Gmail Process*

## 2. Registration via Email

2.1 The user taps on (Don't have an account) so he/she can register via email, shown as **no.1** in **Figure 5.4**.

2.2 A registration page will appear requiring the user to fill needed data i.e., username, email, password, age, height, weight, and gender. The user then proceeds by tapping Register button, shown as **no.2** in **Figure 5.4**.

2.3 The user will be redirected to the dashboard/homepage as shown in last image of **Figure 5.4**.

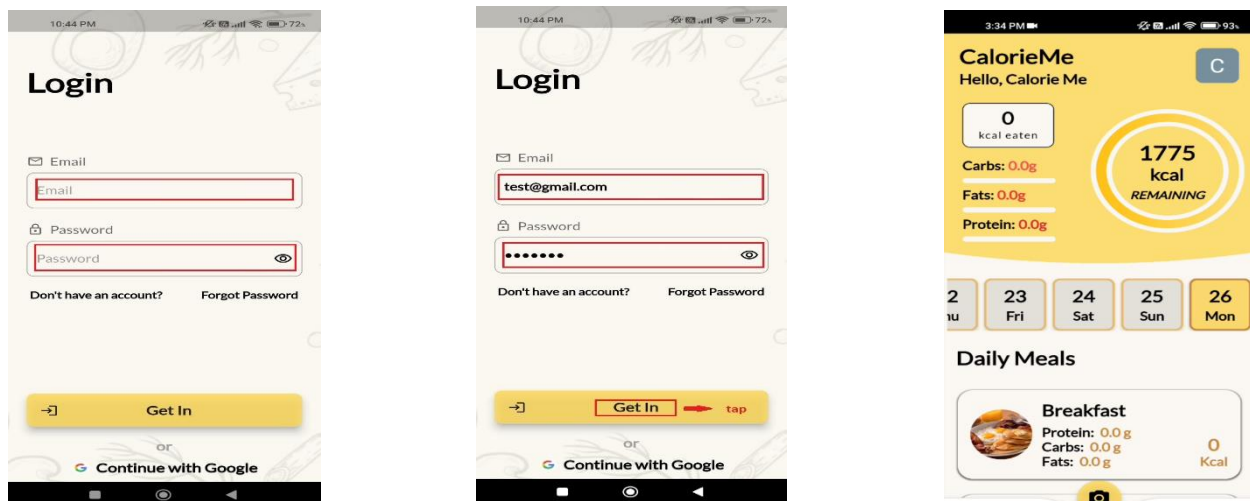


*Figure 5.4: Registration via Email Process*

- **User Login**

1. **Login via Email**

- 1.1 User enters email and password in their respective fields and taps “Get In” button as shown in the first two images of **Figure 5.5**.
- 1.2 If the email and password are written correctly user will be redirected to dashboard/homepage as shown in the last image of **Figure 5.5**.



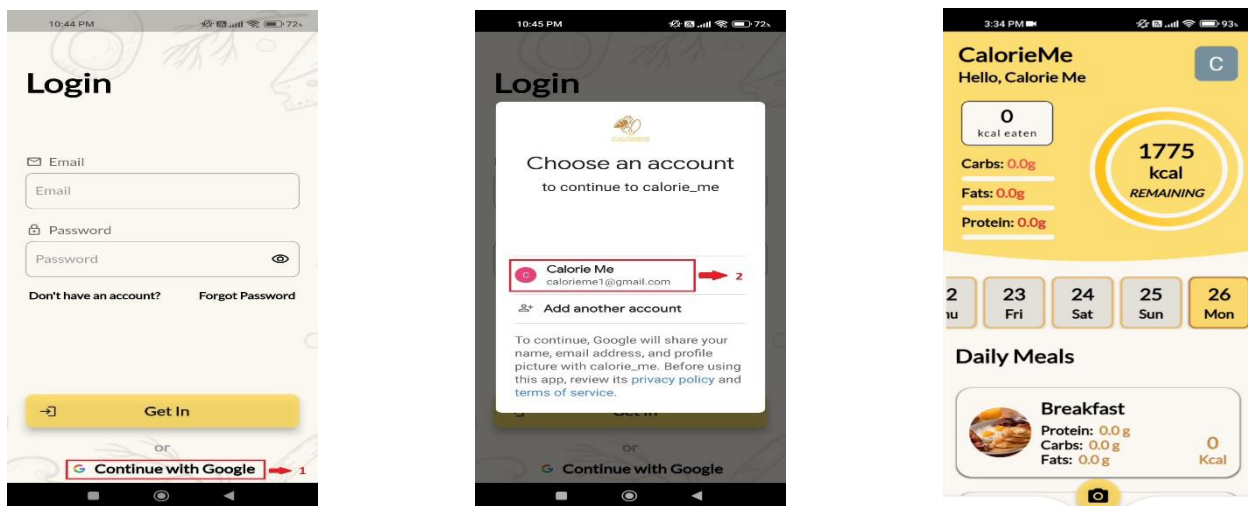
*Figure 5.5: Login via Email Process*

## 2. Login via Gmail

2.1 The user taps on the option highlighted in red shown as **no.1** in **Figure 5.6**.

2.2 A window will pop up asking to continue via Google account registered on smart phone, shown as **no.2** in **Figure 5.6**.

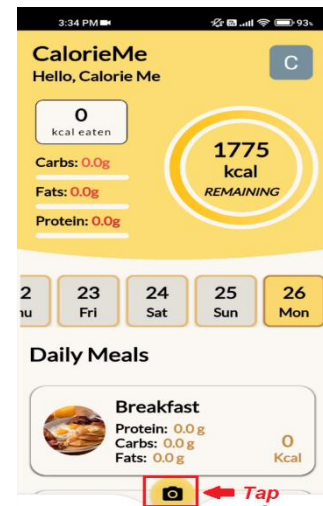
2.3 When the user chooses the Gmail, he/she will be redirected to dashboard/homepage as shown in the last image of **Figure 5.6**.



*Figure 5.6: Login via Gmail Process*

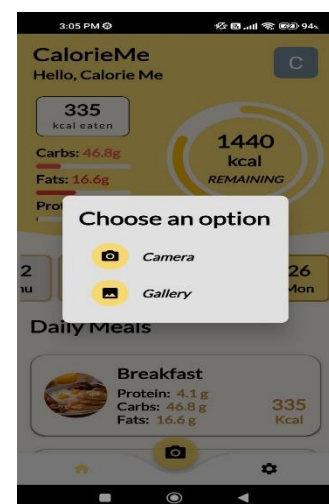
- **Adding Meal**

- After logging into the application, the user will be able to see the homepage containing daily maintenance calories on the top right and calories consumed during the day on the top left as shown in **Figure 5.7**.
- When the user wants to add a meal he/she has to either upload a photo of that meal or capture the meal's photo via the application's viewfinder.
- To either upload the photo or capture it, the user must first tap the camera icon at the bottom center of the page as shown in **Figure 5.7**.



*Figure 5.7: Adding A Meal*

- An option pop-up will then appear to show either choice that a photo might be uploaded as shown in **Figure 5.8**.



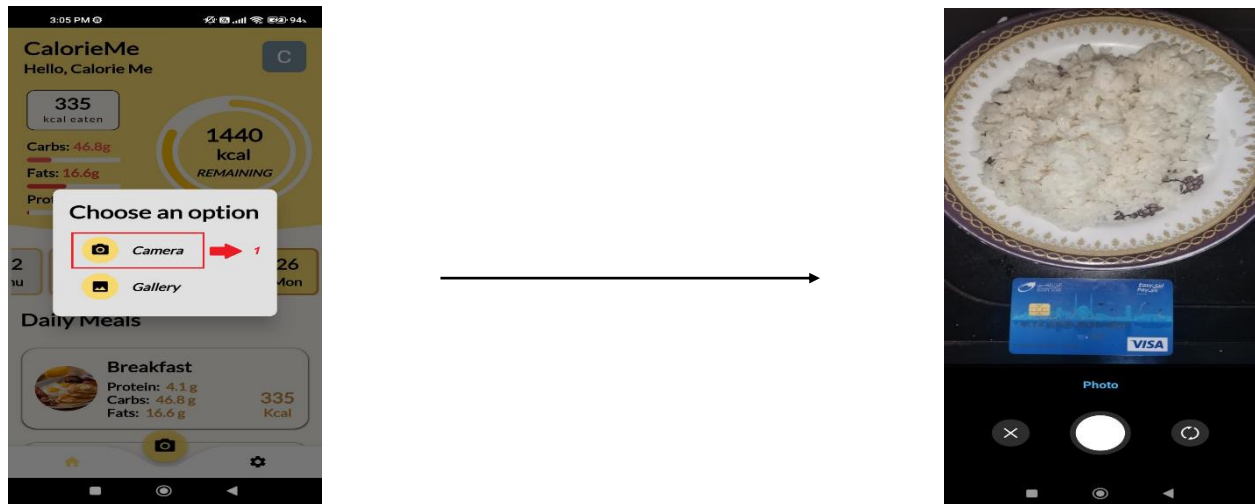
*Figure 5.8: Adding A Meal Options Menu*

## 1. Capture photo via camera

1.1 If the user wants to capture the meal's image then he/she will tap on camera option, shown as **no.1** in **Figure 5.9**.

1.2 The user will then be directed to a viewfinder where he/she will be able to capture a photo of the meal with the reference object as shown in the second image of **Figure 5.9**.

**Note:** the captured image must contain the reference object in frame with the meal as it is critical in estimating the portion size. Reference object could be a credit card or an ID card or any card with dimensions of 85.6mm x 53.98mm.



*Figure 5.9: Capture Photo via Camera Option*

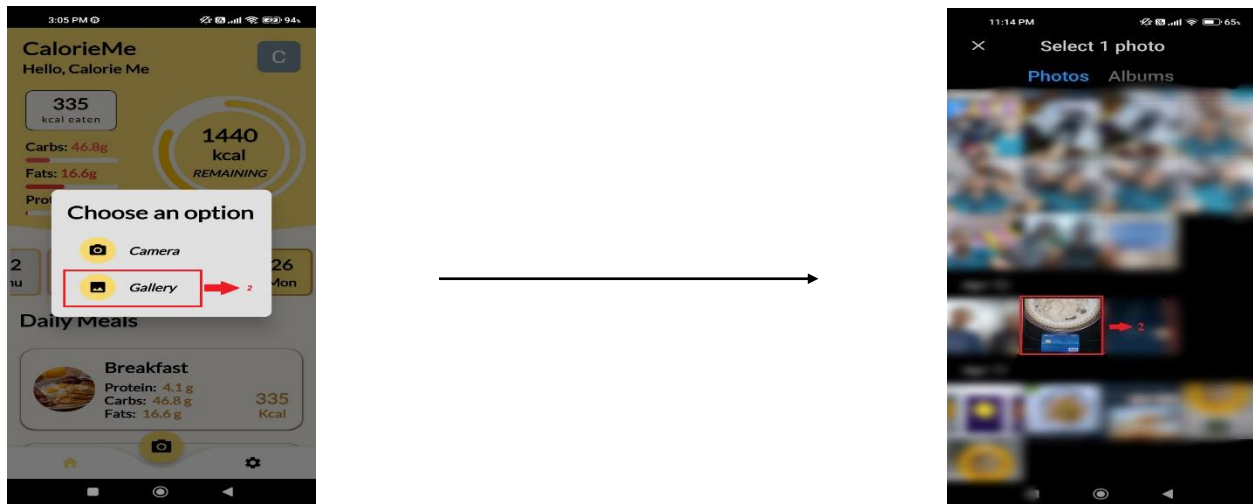


## 2. Upload image from gallery

2.1 If the user wants to upload the meal's image then he/she will tap on gallery option, shown as **no.2** in **Figure 5.10**.

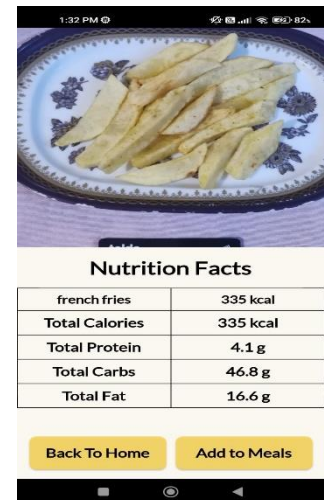
2.2 The user will then be directed to his/her gallery where he/she will be able to choose a photo of the meal with the reference object.

**Note: the uploaded image must contain the reference object in frame with the meal as it is critical in estimating the portion size. Reference object could be a credit card or an id card or any card with dimensions of 85.6mm x 53.98mm.**



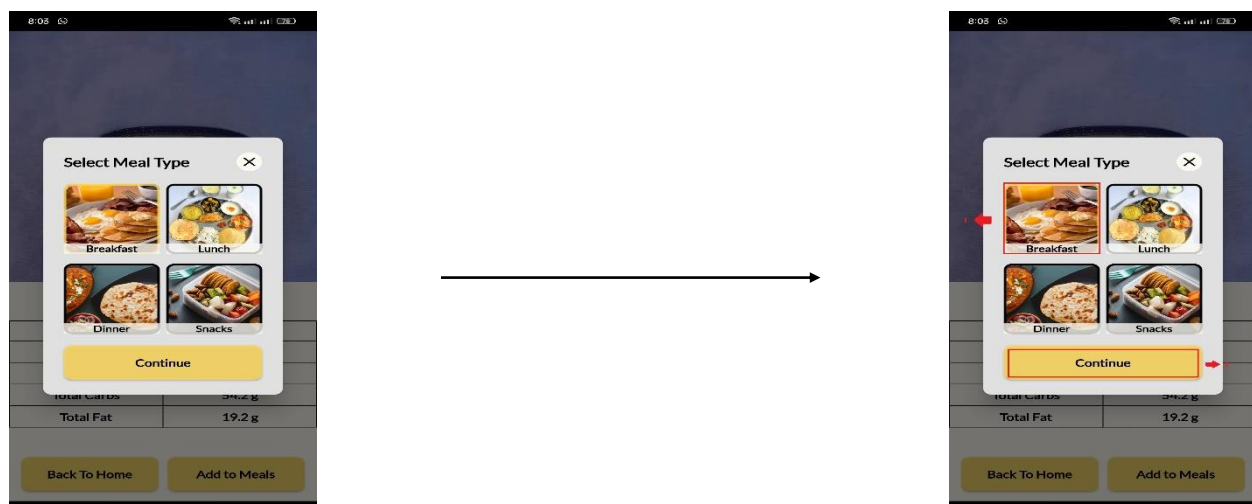
*Figure 5.10: Upload Image from Gallery Option*

- After uploading/capturing the image, it will be uploaded to the server, processed, then the meal's details will be returned to the user (meal components name/s with total number of calories and macros analysis of the meal's components) as shown in **Figure 5.11**.
- The user has the option to add the meal to his/her daily calorie consumption or to return to homepage without adding the meal.



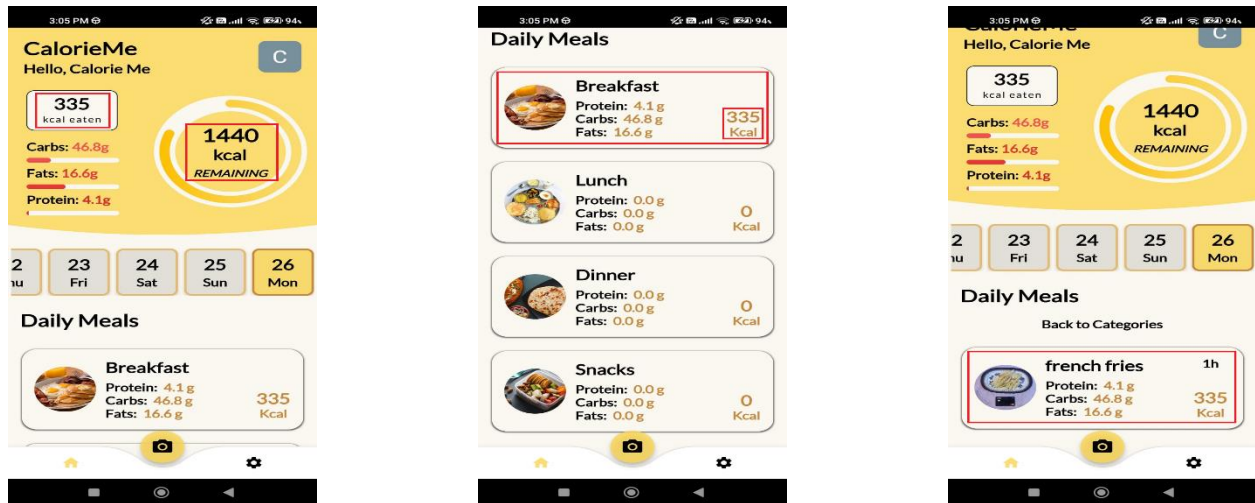
**Figure 5.11: Meal Nutrition Value**

- If the user chose "Add to Meals" option then a pop-up will show to ask where this meal fits within the four meals of the day (breakfast/lunch/dinner/snack).
- The user then chooses the meal-type then taps continue button as shown in **Figure 5.12**.



**Figure 5.12: Add to Meals Option**

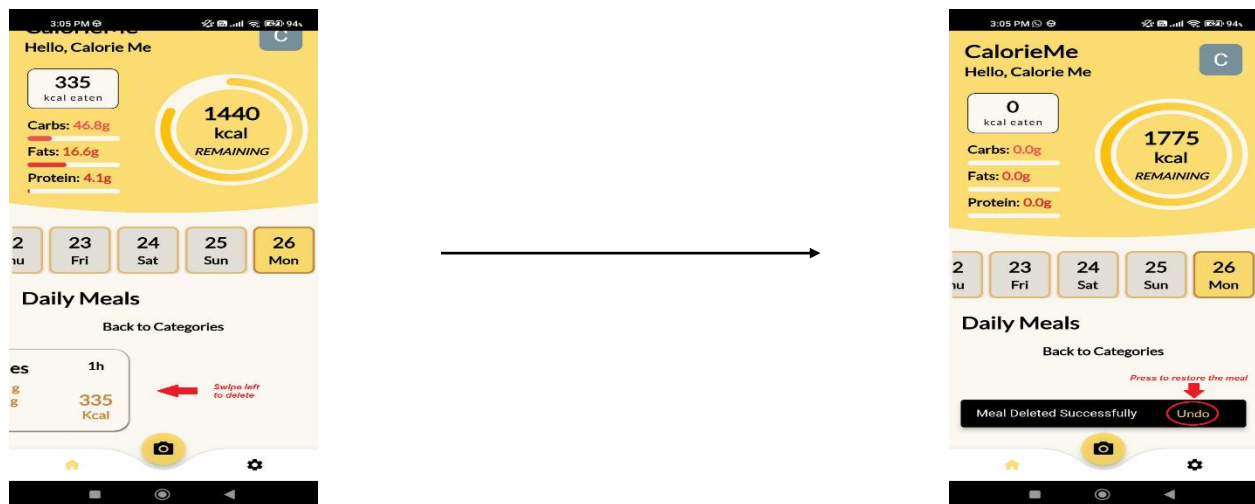
- After adding the meal, the dashboard will be updated by subtracting the meals total calories from the users daily maintenance calories and users meals will be updated by adding the meals name and total calories inside its meal-type also the users total macros intake will be updated by the returned meal's macros as shown in **Figure 5.13**.



*Figure 5.13: User Dashboard Update*

- **Delete Meal**

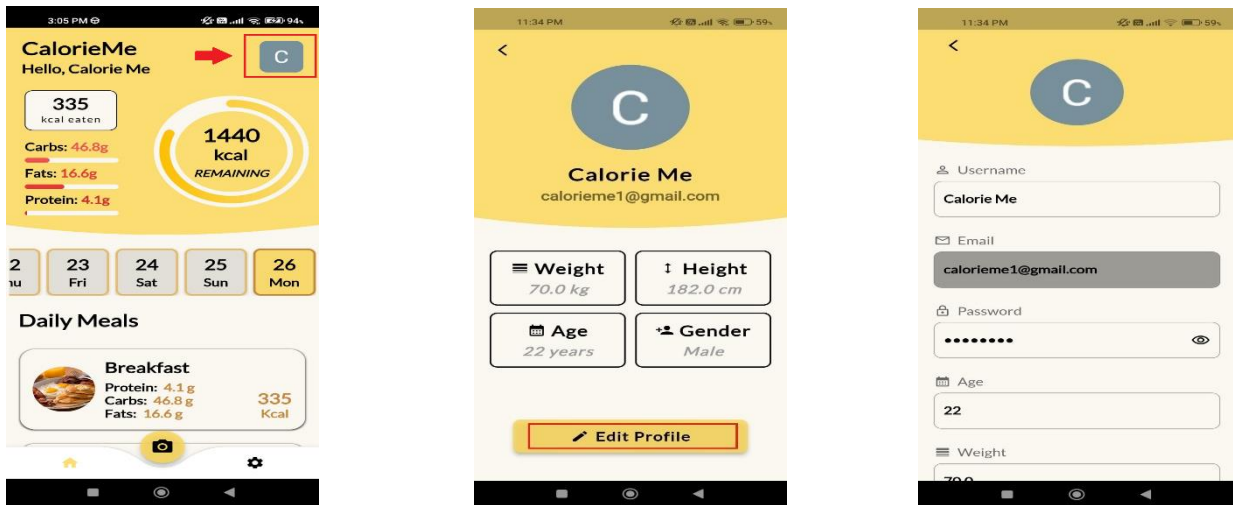
- Users can delete a meal by simply swiping left on it as shown in the first image of **Figure 5.14**.
- If deleted by accident an undo pop-up will show so you can restore it as shown in the second image in **Figure 5.14**.



*Figure 5.14: Deleting a Meal Process*

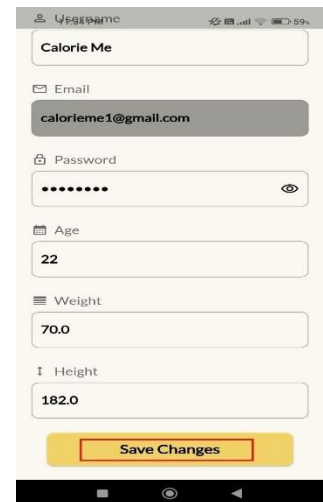
- **User profile management**

- Users can view their personal data and edit their profile any time by simply pressing on their image at the top right corner it will open their profile page as shown in the first two images of **Figure 5.15**.
- A user can edit any field of his profile by pressing the edit profile button shown in the middle image of **Figure 5.15**.



*Figure 5.15: User Profile Management Process*

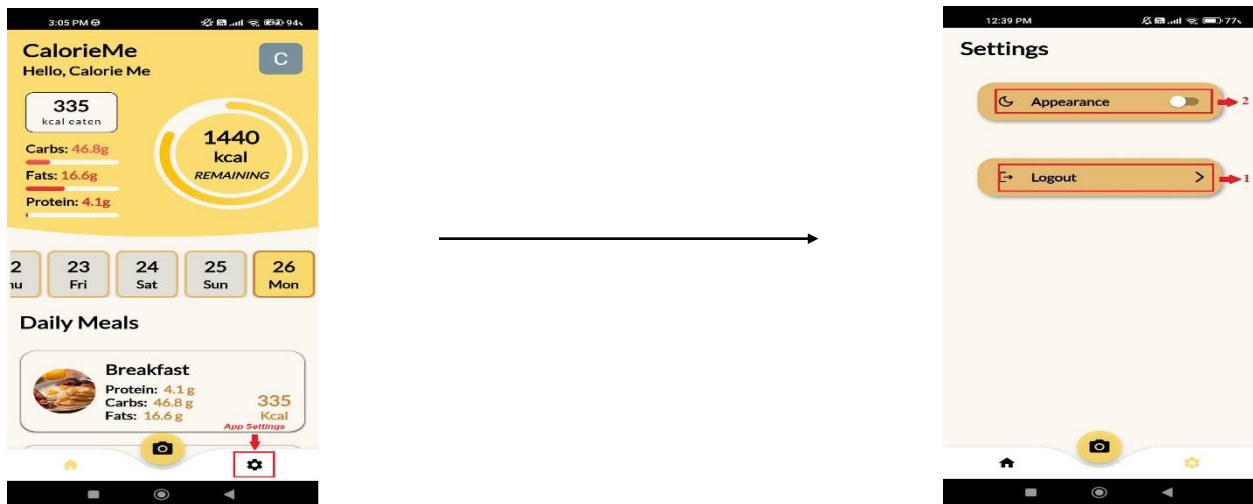
- After editing his information a user can confirm these modifications by pressing “Save Changes” button as shown in **Figure 5.16**.



*Figure 5.16: Save Changes Done to Profile*

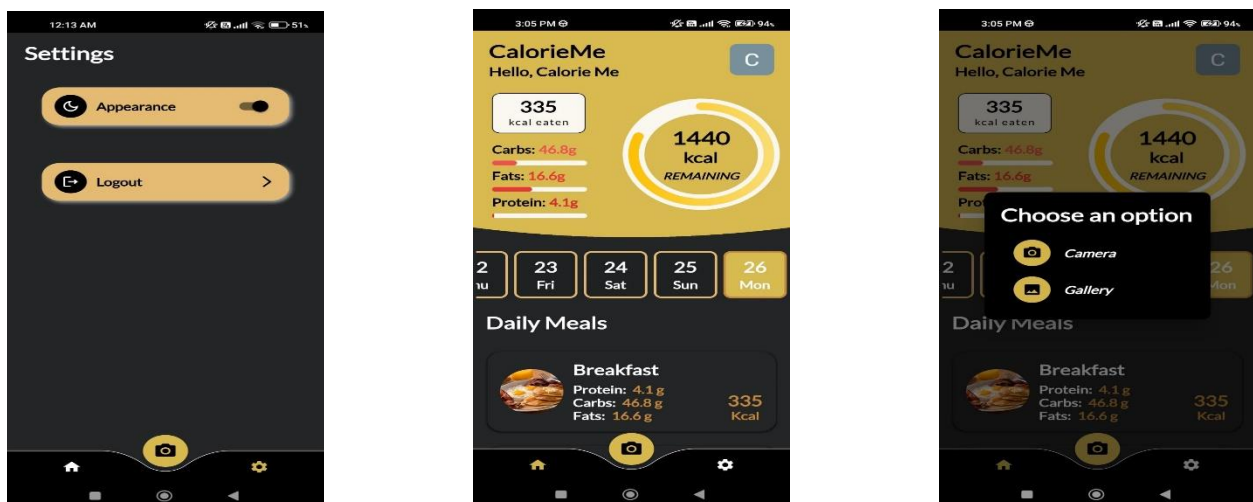
- **Settings**

- A gear icon in the homepage indicates settings page which contains a log out button (1) which will sign the user out of the application, and an appearance dial (2) shown in **Figure 5.17** which switches back and forth between light and dark as shown in **Figure 5.18**.



*Figure 5.17: Settings Option*

- **Dark theme**



*Figure 5.18: Application's Dark Theme*

## 6 Conclusion and Future Work

### 6.1 Conclusion

In this work, a food calorie estimation method is proposed. This is achieved through image via the usage of food image recognition and segmentation techniques. The proposed approach utilizes the presence of **fiducial marker** (reference object) to approximately estimate food portion size and return nutritional value based on estimated portion size and food type. **UECFood100** and **UECFoodPixComplete** datasets were used to train two models to work simultaneously, **YOLOv5** for food image recognition trained on the first mentioned dataset and **DeepLab v3 Plus** for food image segmentation trained on the later dataset. Additionally, **food and their calories** dataset was used which is a CSV file containing food categories and their nutrition values.

This work opens the research opportunities in images-based automated food calorie and food volume estimation using deep learning methods. This will help enhance automatic image-based dietary monitoring systems, which will help the patients, athletes, weight-watchers, etc., to record their diet and maintain a healthy lifestyle by tracking the calorie count by correctly identifying and estimating the portion size of the food they are eating.

## 6.2 Future Work

The future directions of this project include; model accuracy improvements by fine-tuning the different models and expanding the training to include larger datasets to include more food categories from different cultures. Furthermore, the project can be extended by building a complete immersive system with weekly reports and goals to help the user enjoy

the process through a gamified environment. Finally, the application could make use of the depth sensors to eliminate the need for a reference object, however, this would limit the application user base on the sensor's availability in smart phones.



## References

- [1] "What is Computer Vision?," IBM, [Online]. Available: <https://www.ibm.com/topics/computer-vision>. [Accessed 25 6 2023].
- [2] "What Is Image Recognition," Mathworks, [Online]. Available: <https://www.mathworks.com/discovery/image-recognition-matlab.html>. [Accessed 25 6 2023].
- [3] "What Is Image Segmentation?," Mathworks, [Online]. Available: <https://www.mathworks.com/discovery/image-segmentation.html>. [Accessed 25 6 2023].
- [4] F. Lo, Y. Sun, J. Qiu and B. Lo, "Image-Based Food Classification and Volume Estimation for Dietary Assessment: A Review," *IEEE journal of biomedical and health informatics*, vol. 24, no. 7, pp. 1926-1939, 2020.
- [5] F. Kong and J. Tan, "Dietcam: Regular shape food recognition with a camera phone," *Proceedings of the International Conference on Body Sensor Networks*, pp. 127-132, 2011.
- [6] F. Kong and J. Tan, "Dietcam: Automatic dietary assessment with mobile camera phones," *Pervasive and Mobile Computing*, vol. 8, no. 1, pp. 147-163, 2012.
- [7] Y. Kawano and K. Yanai, "Food image recognition with deep convolutional features," *Proceedings of ACM Int. Conf. Multimedia*, pp. 761-762, 2014.
- [8] N. Tammachat and N. Pantuwong, "Calories analysis of food intake using image recognition," *Proceedings of 2014 6th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 1-4, 2014.
- [9] Y. He, C. Xu, N. Khanna, C. J. Boushey and E. J. Delp, "Analysis of food images: Features and classification," *Proceedings of 2014 IEEE international conference on image processing (ICIP)*, p. 2744-2748, 2014.
- [10] M. M. Anthimopoulos, L. Gianola, L. Scarnato, P. Diem and S. G. Mougiakakou, "A food recognition system for diabetic patients based on an optimized bag-of-features mode," *IEEE journal of biomedical and health informatics*, vol. 18, no. 4, pp. 1261-1271, 2014.
- [11] P. Pouladzadeh, S. Shirmohammadi, A. Bakirov, A. Bulut and A. Yassine, "Cloud-based SVM for food categorization,," *Multimedia Tools and Applications*, vol. 74, no. 14, pp. 5243-5260, 2015.
- [12] O. Beijbom, N. Joshi, D. Morris, S. Saponas and S. Khullar, "Menu-match: Restaurant-specific food logging from images," *Proceedings of 2015 IEEE*

- Winter Conference on Applications of Computer Vision*, pp. 844-851, 2015.
- [13] L. Bossard, M. Guillaumin and L. Van Gool, "Food-101 – Mining Discriminative Components with Random Forests," *ECCV2014.*, vol. 8694, 2014.
  - [14] Y. Matsuda, H. Hoashi and K. Yanai, "Recognition of multiple-food images by detecting candidate regions," *2012 IEEE International Conference on Multimedia and Expo*, pp. 25-30, 2012.
  - [15] Y. Kawano and K. Yanai, "Automatic Expansion of a Food Image Dataset Leveraging Existing Categories with Domain Adaptation," *ECCV Workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV)*, 2014.
  - [16] H. Kagaya, K. Aizawa and M. Ogawa, "Food detection and recognition using convolutional neural network," *Proceedings of the 22nd ACM international conference on Multimedia*, p. 1085–1088, 2014.
  - [17] S. Christodoulidis, M. Anthimopoulos and S. Mougiakakou, "Food recognition for dietary assessment using deep convolutional neural networks," *Proceedings of New Trends in Image Analysis and Processing --ICIAP 2015 Workshops*, p. 458–465, 2015.
  - [18] J. Qiu, F. P.-W. Lo, Y. Sun, S. Wang and B. Lo, "Mining discriminative food regions for accurate food recognition," *Proceeding of Brit. Mach. Vision conference*, 2019.
  - [19] W. Min, L. Liu, Z. Luo and S. Jiang, "Ingredient-guided cascaded multiattention network for food recognition," *Proceedings of the 27th ACM International Conference on Multimedia*, p. 1331–1339, 2019.
  - [20] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *Proceedings of International conference on machine learning*, pp. 11905-11946, 2019.
  - [21] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Munari, M. Mordonini and S. Cagnoni, "Food image recognition using very deep convolutional networks," *Proceedings of the 2nd international workshop on multimedia assisted dietary management*, pp. 41-49, 2016.
  - [22] N. Martinel, G. L. Foresti and C. Micheloni, "Wide-slice residual networks for food recognition," *Proceedings of 2018 IEEE Winter Conference on applications of computer vision (WACV)*, p. 567–576, 2018.
  - [23] T. Ege and K. Yanai, "Simultaneous Estimation of Dish Locations and Calories with Multi-Task Learning," *IEICE TRANSACTIONS on Information and Systems*, vol. 102, no. 7, 2019.

- [24] K. Okamoto and K. Yanai, "UEC-FoodPix Complete: A Large-scale Food Image Segmentation Dataset," *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10--15, 2021, Proceedings, Part V*, pp. 647-659, 2021.
- [25] P. Kadam, S. Pandya, S. Phansalkar, M. Sarangdhar, N. Petkar, K. Kotecha and D. Garg, "FVEstimator: A novel food volume estimator Wellness model for calorie measurement and healthy living," *Elsevier Measurement*, vol. 198, 2022.
- [26] Y. Yang, W. Jia, T. Bucher, H. Zhang and M. Sun, "Image-based food portion size estimation using a smartphone without a fiducial marker," *Public health nutrition*, vol. 22, no. 7, pp. 1180-1192, 2021.
- [27] "Review: DeepLabv3+- Atrous Separable Convolution (Semantic Segmentation)," Medium, 29 9 2019. [Online]. Available: <https://sh-tsang.medium.com/review-deeplabv3-atrous-separable-convolution-semantic-segmentation-a625f6e83b90>. [Accessed 5 3 2023].
- [28] "YOLO v5 model architecture [Explained]," OpenGenus, [Online]. Available: <https://iq.opengenus.org/yolov5/>. [Accessed 5 3 2023].
- [29] K. Okamoto and K. Yanai, "An Automatic Calorie Estimation System of Food Images," *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*, pp. 63-70, 2016.
- [30] N. Pandit, "Nutritional Facts for most common foods," Kaggle, [Online]. Available: <https://www.kaggle.com/datasets/niharika41298/nutrition-details-for-most-common-foods>. [Accessed 5 3 2023].
- [31] X. Wu, X. Fu, Y. Liu, E.-P. Lim, S. C.H. Hoi and Q. Sun, "A Large-Scale Benchmark for Food Image Segmentation," 2021.
- [32] U. Sharma, "GourmetNet: Food Segmentation Using Multi-Scale Waterfall," Rochester Institute of Technology, 2021.
- [33] "Image Transforms-Hough Transform," ed.ac.uk, [Online]. Available: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm>. [Accessed 5 3 2023].