Faculty of Computers and Artificial Intelligence, Beni-Suef University, Egypt

Tryfood-ia

Food Classification And Macronutrients Detection Graduation Project

Presented by:

Ammar Yaser Mohammed

Fatma Ahmed Kamal

Rana Adel Abdelhameid

Hager Mohammed Ramadan

Sohaila Mahmoud Saied

Supervisor:

Prof. Mohammed Kayed

Dr. Ehab Ibrahim

Table Of Content

Acknowledgment.....4

Purpose of documentation.....4

Abstract.....5

Chapter 1 (INTRODUCTION)

1.1 Introduction.....6

1.2 Problem Statement.....7

1.3 Objectives.....8

1.4 Conclusion.....8

Chapter 2 (RELATED WORKS)

2.1 Introduction.....9

2.2 Related works.....9

2.3 TRYFOOD-ia Features.....10

2.5 Project Risk Management.....11

Chapter 3 (PROJECT METHODOLOGY)

3.1 Software Development Lifecycle.....15

3.2 Why agile is used 17

Chapter 4 (DATA & MACHINE LEARNING)

4.1 Dataset.....20

4.2 YOLO.....29

4.3 TRAINING MODEL.....42

Chapter 5 (MOBILE APPLICATION)

5.1 what is UI/UX ?.....48

5.2 Introduction 50

5.3 User Interface.....55

Chapter 6 (BACKEND)

6.1 what is DataBase?.....104

6.2 My SQL DataBase 106

6.3 Use Case Diagram.....111

6.4 class Diagram.....114

6.5 ERD.....117

6.6 Schema.....119

6.7 ASP.net.....121

6.8 FLASK Connect Model.....131

Chapter 7 (TOOLS & TECHNOLOGIES)

7.1 ML TOOLS 135

7.2 Mobile Application TOOLS.....136

Chapter 8 (FUTURE WORK)

FEATURES 138

REFERNCE.....139

Acknowledgment:

Firstly, the success of our project happens when we believe in ourselves, believe in doing the right things to help other people, and solve a community issue. Secondly, we would like to acknowledge and give our warmest thanks to our supervisor, **Dr. Mohammed Kayed**, who made this project possible and for his guidance and advice that carried us through all the stages of the project. Special thanks to **Dr. Ehab Ibrahim** for his support and interest in the project to bring it to this stage and for his efforts over 4 years with us.

Purpose of documentation

This document aims to describe the key elements of the project starting with the project idea, the software development life cycle of the project, discussing the design of the user interface, and outline the technical aspects of the project in detail what parts that will be implemented and which parts to be considered as future work.
With all of this information, this document becomes an absolute reference for the designers and developers during the implementation phase of this project.

ABSTRACT:

In 2019, the "100 Million Health" campaign conducted a survey revealing that 39.8% of Egyptians, approximately 49.7 million individuals, are classified as obese based on a Body Mass Index (BMI) exceeding 30 kg/m². The survey further highlighted a notable gender disparity, with obesity rates among women at 49.5% compared to 29.5% among men. This research underscored a prevalent lack of attention to healthy eating practices and nutritional balance in daily dietary choices across Egypt.

In response to these findings, our innovative application provides a comprehensive solution to address these nutritional challenges. Users can leverage the application to photograph their meals and receive detailed nutritional analyses, empowering them to better understand their dietary intake and identify areas for improvement. Moreover, the application offers personalized features such as regular consultations with nutritionists, precise tracking of nutritional intake, access to a diverse array of healthy recipes, and a supportive community platform. By fostering engagement and facilitating knowledge-sharing among users with shared health goals, the application aims to promote sustainable adherence to healthier eating habits and contribute to improved public health outcomes in Egypt.

Chapter 1 INTRODUCTION

1.1 Introduction

In the contemporary digital age, maintaining optimal health and nutrition has become increasingly important yet challenging. With the fast-paced lifestyle and the overwhelming amount of dietary information available, individuals often struggle to make informed nutritional choices.

Proper nutrition supports bodily functions, boosts the immune system, enhances energy levels, and reduces the risk of chronic diseases such as obesity, diabetes, and cardiovascular conditions. Additionally, a healthy lifestyle can improve mood, cognitive function, and longevity, making it essential for individuals to prioritize their health in daily life.

AI can enhance the accuracy and efficiency of nutrition tracking. Machine learning models can identify and quantify food items from photos, estimate portion sizes, and calculate nutritional content with high precision. This reduces the burden of manual tracking and ensures users receive accurate information about their dietary intake.

maintaining a healthy body is essential for overall well-being, and AI plays a pivotal role in facilitating this process. By providing personalized, accurate, and actionable insights, AI technology empowers individuals to make healthier choices and achieve their nutritional goals more effectively. As AI continues to evolve, its integration into health and nutrition management will undoubtedly become increasingly important, offering innovative solutions to support a healthier and more informed society.

1.2 Problem Statement

What is the importance of this problem?

In Egypt, a pervasive lack of awareness and interest in healthy eating and nutritional routines has led to widespread ignorance of the serious health risks associated with poor nutrition. This gap in knowledge contributes significantly to the prevalence of malnutrition-related diseases, which not only diminish individual health but also hinder societal progress by reducing productivity and economic potential. Moreover, the absence of accessible nutritionists for dietary guidance exacerbates the issue, as many individuals resort to unhealthy eating habits without understanding the long-term consequences. This cultural preference for fast food and restaurant meals further compounds the problem, perpetuating a cycle of nutritional deficiencies and health challenges among the population. Addressing these issues is crucial to improving public health outcomes, fostering economic growth, and ensuring a healthier future for generations to come in Egypt.

Current National Statistics:

Diabetes: Approximately 17% of Egyptian adults are diagnosed with diabetes, and an additional 10% exhibit pre-diabetic symptoms. Over 60% of diabetic patients do not receive adequate treatment due to awareness issues or lack of regular screenings

Chronic Kidney Disease: Egypt faces increasing numbers of chronic kidney disease cases compared to global averages. According to the Egyptian Society of Nephrology and Kidney Transplantation, the rate of individuals needing kidney dialysis reaches 650 cases per million, more than double the global rate.

Underweight: The prevalence of underweight individuals in Egypt is estimated at 12.3%

Obesity: Approximately 40% of adults in Egypt are classified as obese, highlighting pervasive issues with unhealthy weight management and dietary habits that contribute to long-term health complications.

These statistics underscore the critical importance of addressing nutritional education and healthcare accessibility to mitigate the growing burden of chronic diseases and promote healthier lifestyles among Egyptians.

What is the current solution?

Solution1 : increasing the number of nutritionists to suit the number of citizens and providing awareness programs

Solution2 : Increase investment in healthy eating restaurants to raise their quality and spread

How will our solution solve the problem?

Using the application, the user can know the nutritional value of his food in seconds without the need for a doctor. In the event of a need for a doctor, it provides quick access to doctors and continuous follow-up. It also provides widely available recipes. Finally, it provides an environment of people interested in nutrition and health, which encourages the user to continue. In a healthy routine.

1.3 Objectives

The goal of our project is to reach a stage where the healthy routine is the prevailing routine in our society, which preserves the general health of the community, because the nutritional status of individuals shows its direct effects on society.

1.4 Conclusion

At the end of this chapter, the reader will be able to form a basic idea about what is the problem that our project aims to solve,

- what is the importance of the idea
- The problem other existed solutions

• what we will add to our project to make it better than the traditional or existing methods to solve the same problem

Chapter 2 Related Works

2.1 Introduction

If we look at the entire world, we will find that the problem of malnutrition and related diseases is widespread, but in Egypt we do not have many applications that contribute to raising public awareness of nutrition compared to other countries and societies.

2.2 Related works

Nutritionix Track: is an app that helps users monitor their food intake, track macronutrients, and set personalized nutrition goals. It offers a large database of food items with detailed nutritional information

Fooducate: is a nutrition-focused app that helps users make healthier food choices. It offers features such as food tracking, barcodescanning, and-personalized-recommendations based on dietary preferences and health goals.

Calorie Counter - MyNetDiary: is a comprehensive calorie counter and diet tracker that helps users achieve their weight loss and health goals. It offers features such as food logging, exercise tracking, water intake monitoring, and personalized meal planning.

Lose It!: A weight loss app focusing on food tracking, exercise logging, and goal setting, with a supportive community aspect.

SparkPeople: Offers fitness tracking, meal planning, and community support, allowing users to log meals, track workouts, and access recipes.

Yazio: A nutrition app providing personalized meal plans, exercise tracking,and calorie counting, with a focus on managing macronutrients.

HealthifyMe: An all-in-one health and fitness app featuring calorie tracking, workout plans, personalized coaching, and connections to nutritionists and trainers.

2.3 TRYFOOD-ia Features

The application detects food and shows nutritional values and advice, making it easier for the user to maintain a healthy routine and diet.

1. AI-Based Food Recognition and Nutritional Information:

- Utilizes artificial intelligence to detect food through images.
- Provides nutritional value details like calories, protein, carbs, fats, etc.

2. Food & Plan Recording:

- Allows users to record their food intake via images or doctor-prescribed

plans.

3. Doctor Integration:

- Facilitates sharing the application data with specialist doctors to modify and monitor the user's nutritional plan.

4. Section for recipes:

- Offers meal recipes and advanced filtering options to help users find meals that meet their specific dietary requirements and preferences

5. Body Mass Index:

- where users can enter their height and weight to calculate their BMI. The app provides insights into what BMI means and provides recommendations for getting into a healthy BMI range. The results can also be shared with a nutritionist or any other platform

6. Health community:

- One of the key aspects of the app is its ability to create a vibrant community of health-conscious individuals. Users can join the app and share their progress

2.4 Risk management

First, we would like to explain the concept of risk; Risk is any unexpected event that can affect your project — for better or for worse. Risk can affect anything: people, processes, technology, and resources. An important distinction to remember is that risks are not the same as issues. So, risk management is known as the practice of identifying, evaluating, and preventing or mitigating risks to a project that have the potential to impact the desired outcomes.

To effectively manage risk, you must have a clear understanding of your objectives so you can identify any possible barriers that could impact the team's ability to produce results.

"Risk management is really about looking at your project objectives and figuring out what the threats to those objectives are, and what you can do to address them from the beginning," says Connie Emerson.[4]

The types of events or scenarios that fall under the category of risk can be broad and sometimes misinterpreted. While project managers or those tasked with overseeing a project may be inclined to view risks exclusively as threats, this is not always the case.

To clarify this common misconception, Emerson defines project risk as "…a future event that may or may not happen which, if it does happen, will have some impact on the objectives of the project. It could be positive—an opportunity, or negative—a threat."

Types of Project Risk

There are several types of risks that occur frequently, regardless of the specifics of the project. These common types of risks include:

• Cost: The risk of events that impact the budget, especially those that cause the project to be completed over budget.

• Schedule: The risk of unplanned scheduling conflicts, such as events that cause the project to be delayed.

• Performance: The risk of events that cause the project to produce results that are inconsistent with the project specifications.

• Security: The risk of events that cause loss of control over personal information.

So now let's look at some of our project risk:

Project_area	Project Scope Management
Project_risk	Lack of clarity in terms of expected deliverables
Risk_Type	Project Scope
sol.	We made a CR document to clarity expected Customer Requirements and SRS document to clarity expected System Requirement

project_area	Project Scheduling/ Project Timeline
Project_risk	Inaccurate project schedule estimation leading to disproportionate project work allocation
Risk_Type	Project Schedule
sol.	We have set a clear schedule that specifies the sufficient time to implement each part of the project with increased time to avoid any negligence in the time frame required to complete the project

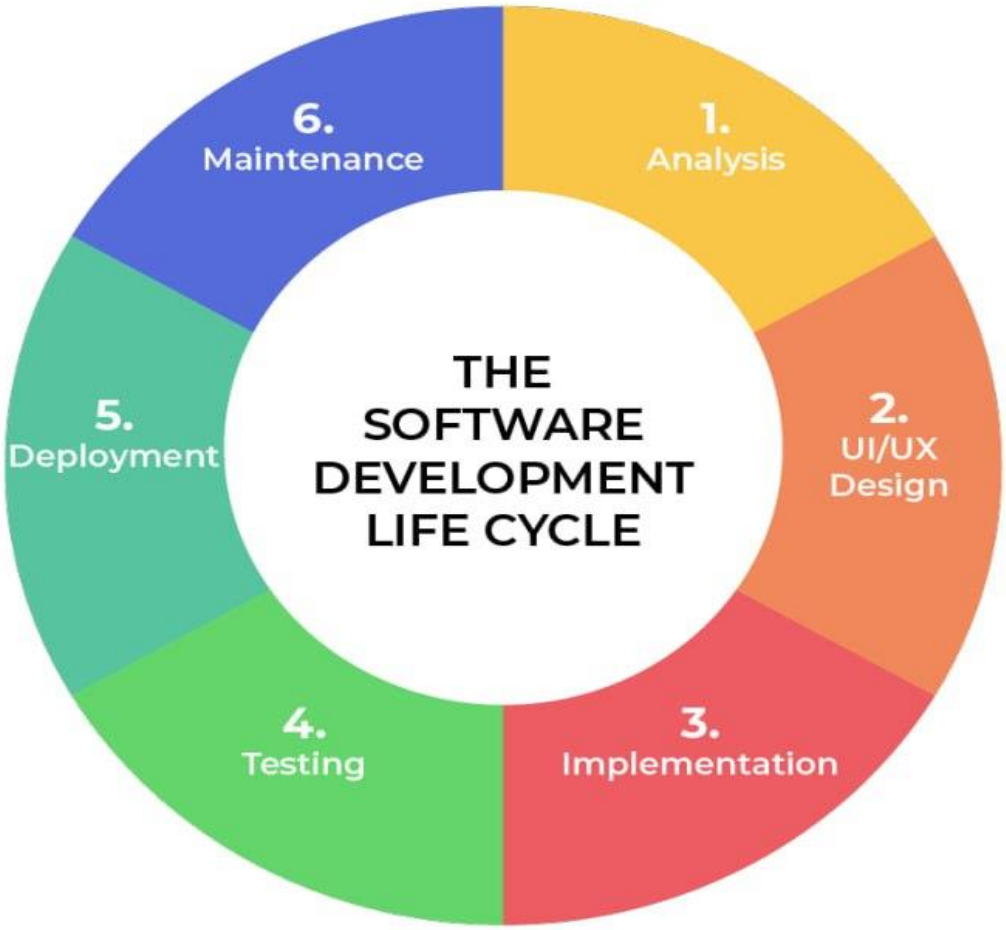
project_area	Resource Management
Project_risk	Lack of rightly skilled or experienced resources
Risk_Type	Skills
sol.	In the beginning, we had to choose the skills we needed to implement our project, the person who would be responsible for each part, and each person started to develop their skills so that they could implement what was required of them.

project_area	Project security
Project_risk	the potential loss of control over personal information
Risk_Type	security
sol.	Our project has gone through different stages of testing in order to be safe for users

Chapter 3 Project Methodolgy

3.1 Software Development Life Cycle

In this section, we are going to talk about the life cycle of our project. First of all, what is SDLC? The SDLC is a methodology for producing software projects, involving a detailed plan for development, maintenance, replacement, alteration, or enhancement. It helps improve software quality and overall development process. The typical SDLC consists of stages like:



Stage 1: Planning and Requirement Analysis Requirement analysis

is a crucial stage in SDLC, involving senior team members, customer input, sales department, market surveys, and industry experts. This information is used to plan project approaches, conduct product feasibility studies, and identify quality assurance requirements. The technical feasibility study outlines successful implementation methods with minimal risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved by the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle

Stage 3: Designing the Product Architecture

SRS is the reference for product masterminds to come out with the trim framework for the product to be developed. Predicated on the necessities specified in SRS, normally, farther than one design approach for the product framework is proposed and proved in a GDD- Design Document Specification. This GDD is reviewed by all the important stakeholders and predicated on polychrome parameters as peril assessment, product robustness, design modularity, budget, and time constraints, the trim design approach is named for the product. A design approach defines all the architectural modules of the product along with its communication and data exodus representation with the external and third- party modules (if any). The internal design of all the modules of the proposed structure should be definitely defined with the tiniest of the details in GDD.

Stage 4: Building or Developing the Product

In this stage of SDLC, the actual development starts, and the product is built. The programming code is generated as per GDD during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle. Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high-level programming languages such as C, C++, Pascal, Java, and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS

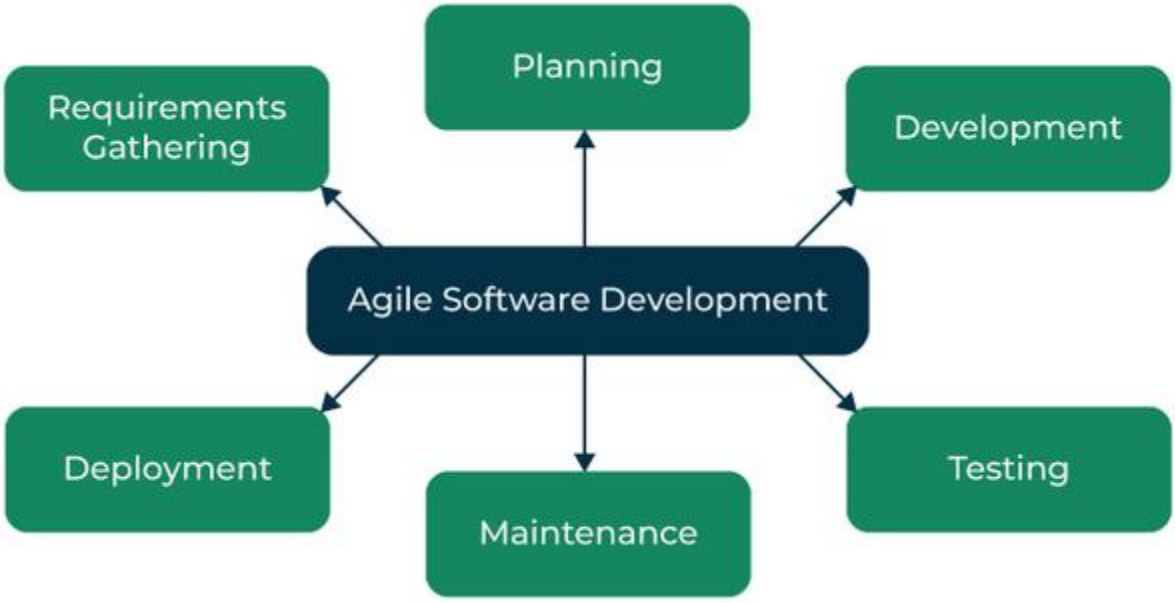
Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing). Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

3.2 WHY Agile is used?

1. **Creating Tangible Value:** Agile places a high priority on creating tangible value as soon as possible in a project. Customers can benefit from early delivery of promised advantages and opportunity for prompt feedback and modifications.
2. **Concentrate on Value-Added Work**: Agile methodology promotes teams to concentrate on producing functional and value-added product increments, hence reducing the amount of time and energy allocated to non-essential tasks.
3. **Agile as a Mindset:** Agile represents a shift in culture that values adaptability, collaboration, and client happiness. It gives team members more authority and promotes a cooperative and upbeat work atmosphere.
4. **Quick Response to Change:** Agile fosters a culture that allows teams to respond swiftly to constantly shifting priorities and requirements. This adaptability is particularly useful in sectors of the economy or technology that experience fast changes.
5. **Regular Demonstrations:** Agile techniques place a strong emphasis on regular demonstrations of project progress. Stakeholders may clearly see the project's status, upcoming problems, and upcoming new features due to this transparency.
6. **Cross-Functional Teams:** Agile fosters self-organizing, cross-functional teams that share information effectively, communicate more effectively and feel more like a unit.

The agile software development process



1. **Requirements gathering:** The customer's requirements for the software are gathered and prioritized.

2. **Planning:** The development team creates a plan for delivering the software, including the features that will be delivered in each iteration.

3. **Development:** The development team works to build the software, using frequent and rapid iterations.

4. **Testing:** The software is thoroughly tested to ensure that it meets the customer's requirements and is of high quality.

5. **Deployment:** The software is deployed and put into use.

6. **Maintenance:** The software is maintained to ensure that it continues to meet the customer's needs and expectations.

Agile Software Development is widely used by software development teams and is considered to be a flexible and adaptable approach to software development that is well-suited to changing requirements and the fast pace of software development. Agile is a time-bound, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver all at once.

Chapter 4 Data & Machine Learning

4.1 DATASET

This data set consists of different types of vegetables, fruits, meat, and dairy. It is divided into 7 classes based on the percentage of macronutrients in each type

1-Gathering Dataset

When we started searching for a data set that would meet our needs in terms of the number of images available for each type and the abundance of different types in order to cover most or a large part of our food resources in our Arab or Egyptian environment in particular, we were unable to find only a small group of types known to us, such as fruits and vegetables that Their number hardly exceeds 30 different types, with insufficient pictures of these types being available. As for the rest, most of them were Asian, European, or even African foods, which forced us to begin collecting our own data that is related to what we are familiar with and have available to us.

So, we first started by taking some species that already existed in another data set and which are considered species known to us in our environment.
Then we continued using web scraping to obtain the rest of the items until we had a plentiful set of items used and available in our environment.

Then we moved to the next step, which is collecting nutritional information or macronutrients (calories, protein, carbohydrates, fats), but unfortunately, we were not able to find accurate and clear information for most of our varieties, and even if we found it on some reliable sites, we had to communicate with specialists so that we could use it. This information is convenient, so we turned to a nutritionist, **Dr. Muhammad Hussein**, who helped us verify the accuracy of the information we have first, and then completed the missing part of it himself.

Here, when we looked at the information after it was complete and after analyzing and studying it, we found that there are some classes that are similar in the proportions of macronutrients. Therefore, we asked him to divide the 49 elements in our data set based on this similarity, which resulted in 7 classes that differ in their proportions.

After that, we added a column containing tips related to each individual item

2-Preparing

At this stage, we found that each type contains a different number of images, and it also contains many invalid images, whether of poor quality or not related to this type, because we brought them through web scraping, which does not guarantee that the images obtained are Only the ones you want, so we started scanning these images of all types and keeping only the good quality images that belong to this type.

Then we faced another problem, which is the difference in the number of images in each type, which will cause overfitting of the types in which the images are much larger than the other types. Therefore, we needed to standardize the number of images, based on the resources available to us, so we standardized them to 200 images for each type.

Finally, we named each image based on its type and ranked from 1 to 200

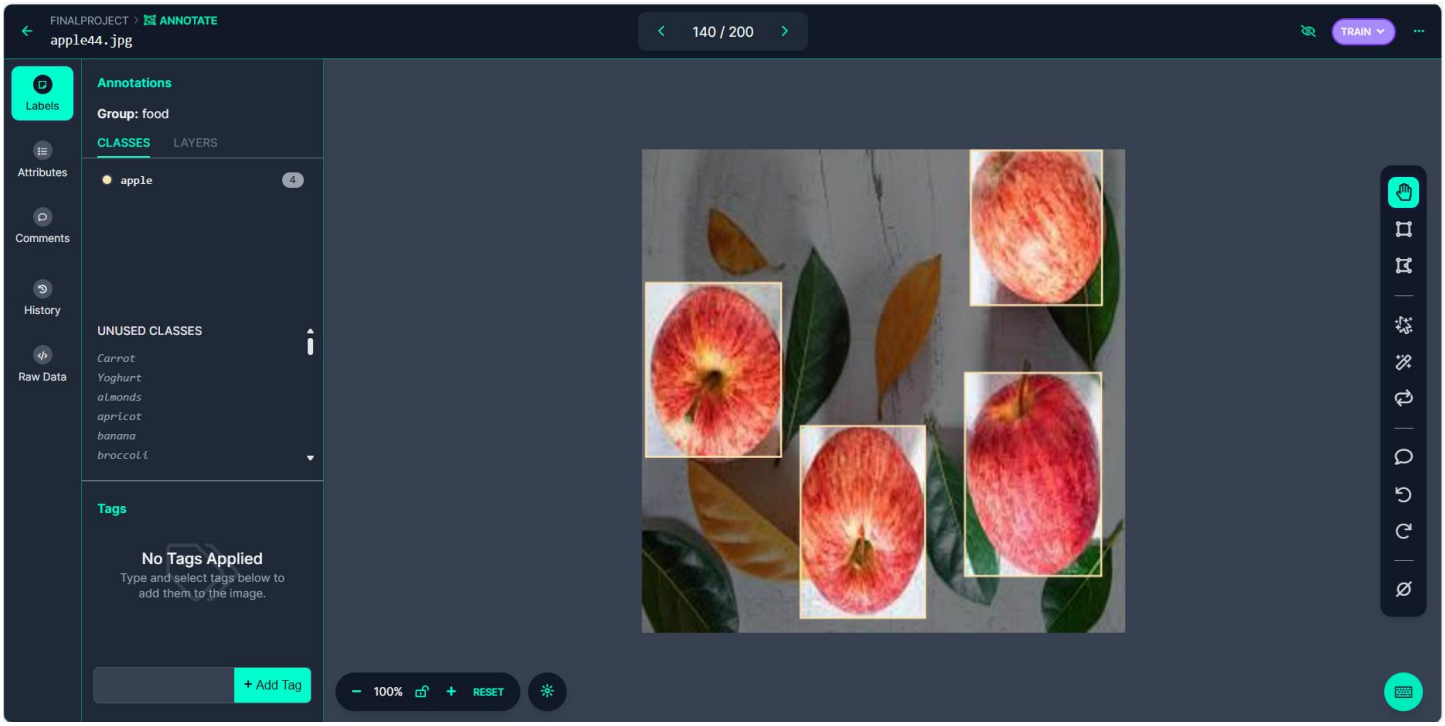
3-Assigning macronutrients classes

We consulted a nutritionist to know the proportion of macronutrients present in each type we have, and then distributed these types into categories describing the proportions of macronutrients, such as LHLM (from left to right: low calories, High Protein, Low Carb, Moderate Fat) and likewise for the rest: HLHL, HLLH, LLLL, MLML, MMLM, MMLH

4-Annotaion: Roboflow

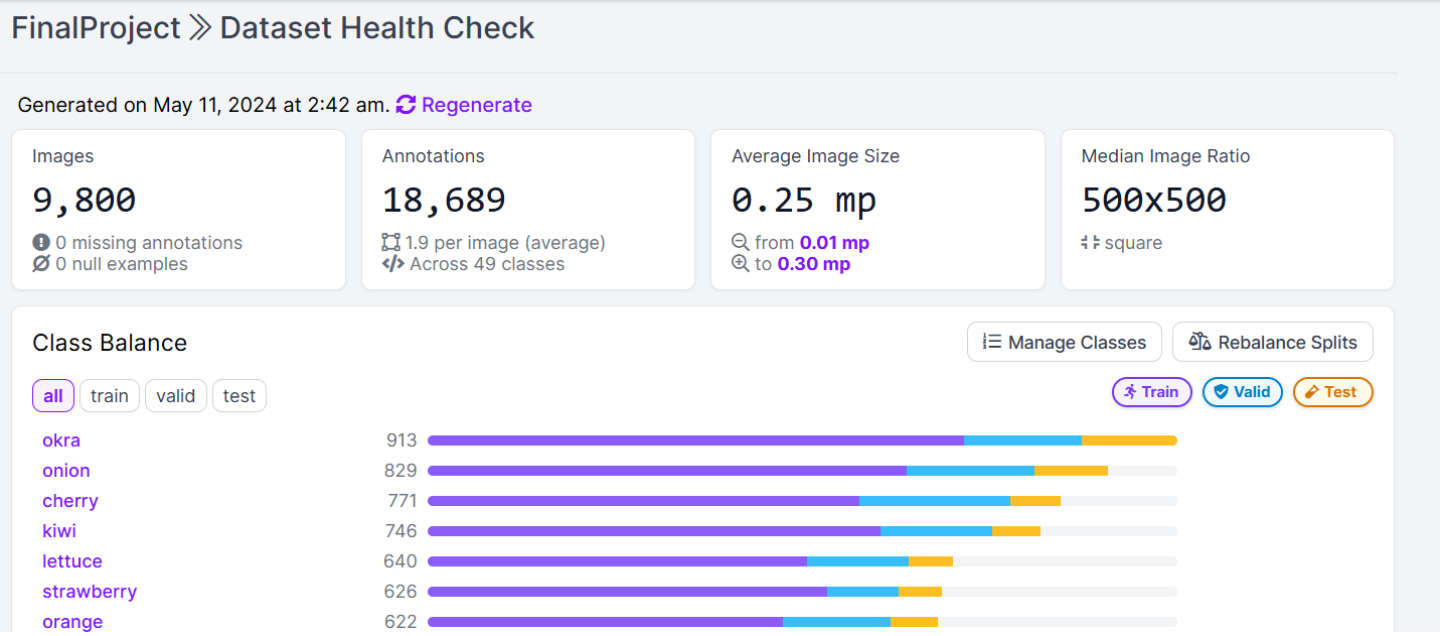
After searching for ways to annotate images, whether manually or automatically, we did not find anything better than the Roboflow website for manual annotation, which guarantees the highest quality.

Since the dataset was ready to start work, we uploaded each type separately and did a manual annotation for each image separately and determined the name of the class it belongs to. The datasheet became almost ready to start training our model on it.

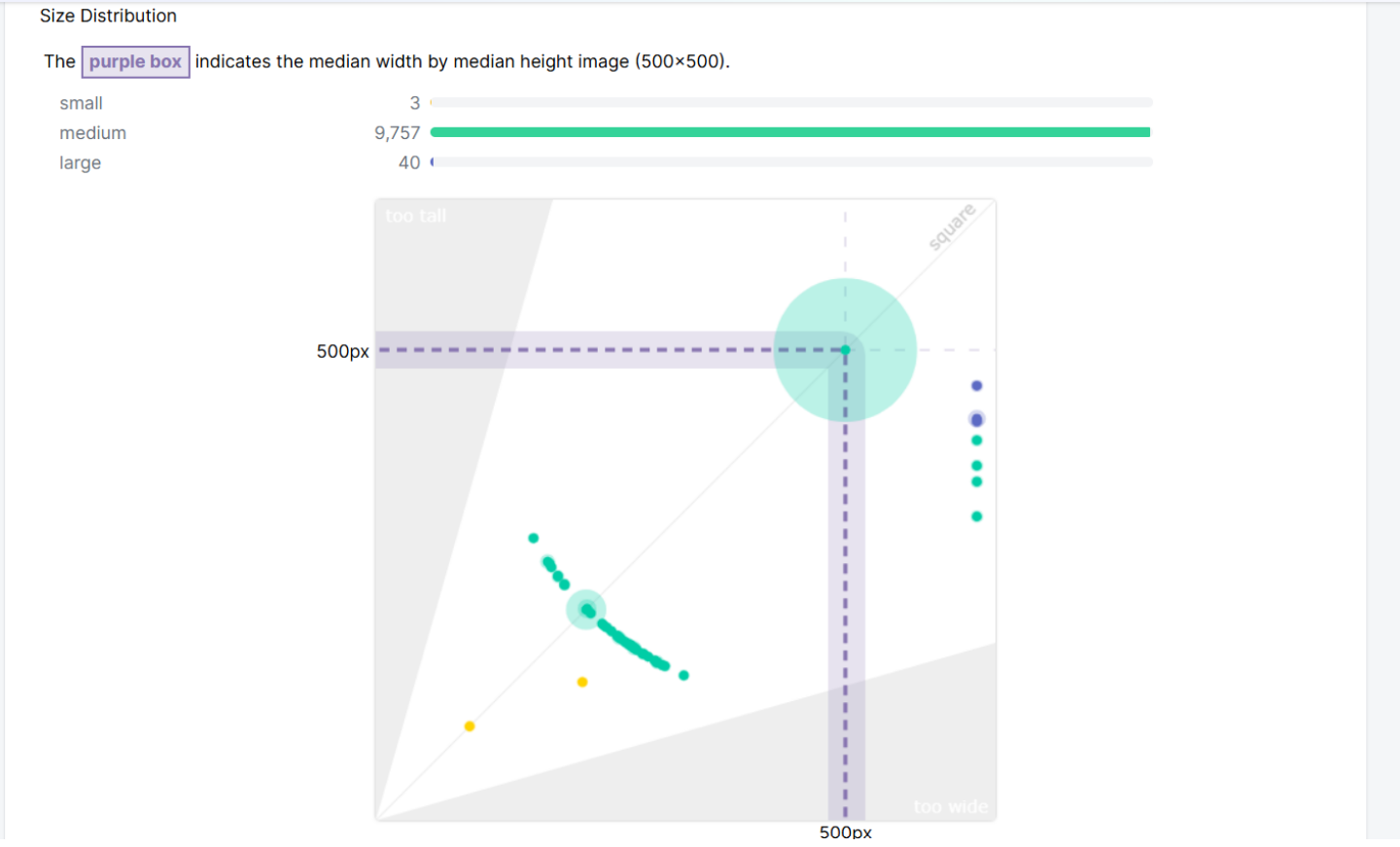


5- Dataset Health Check:

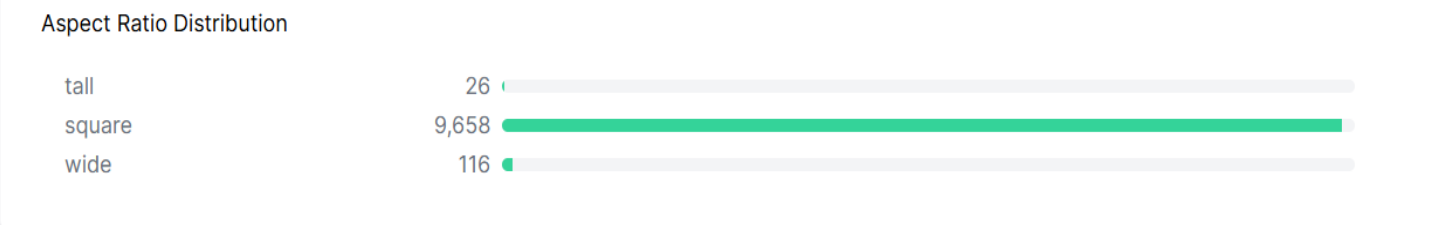
Class balance



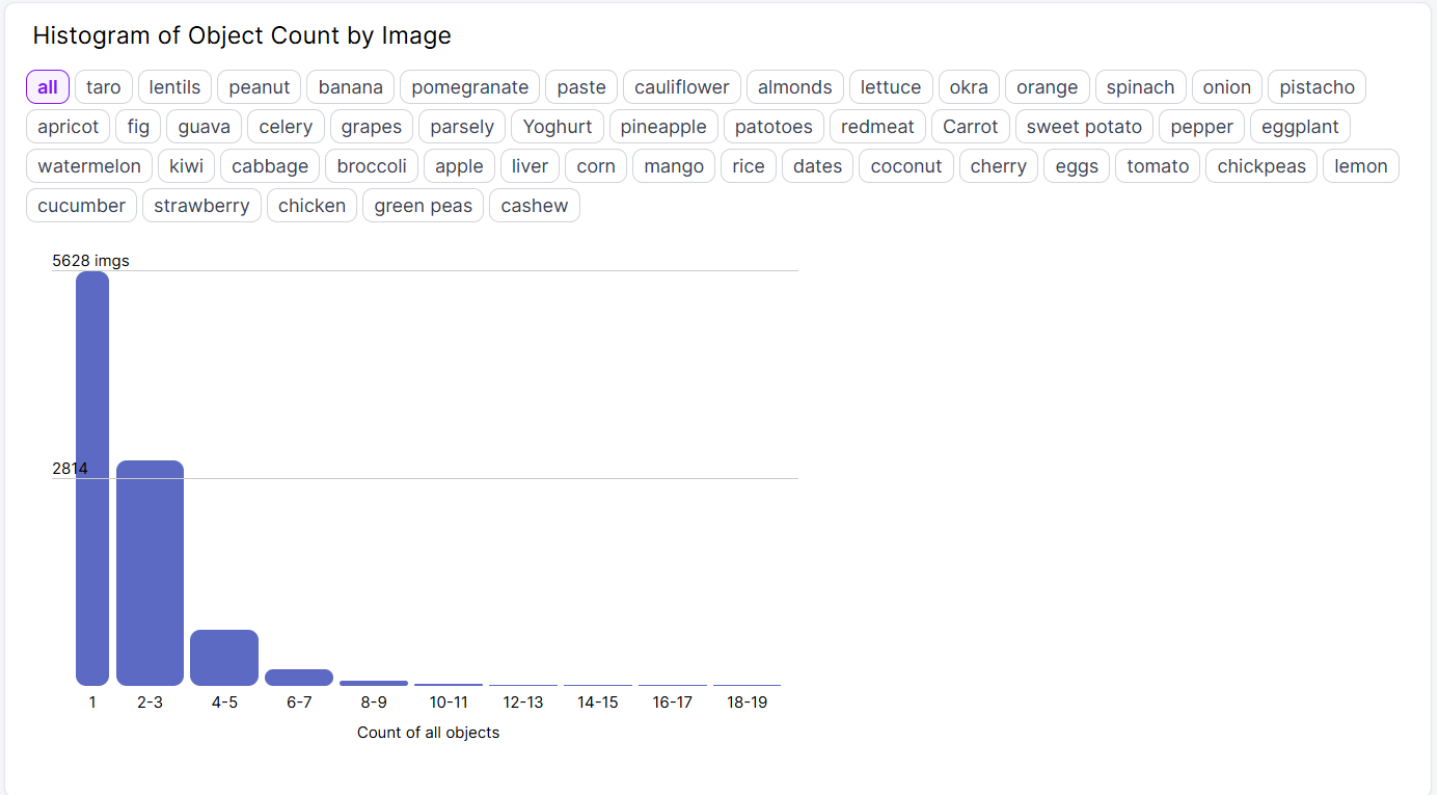
Dimension Insights



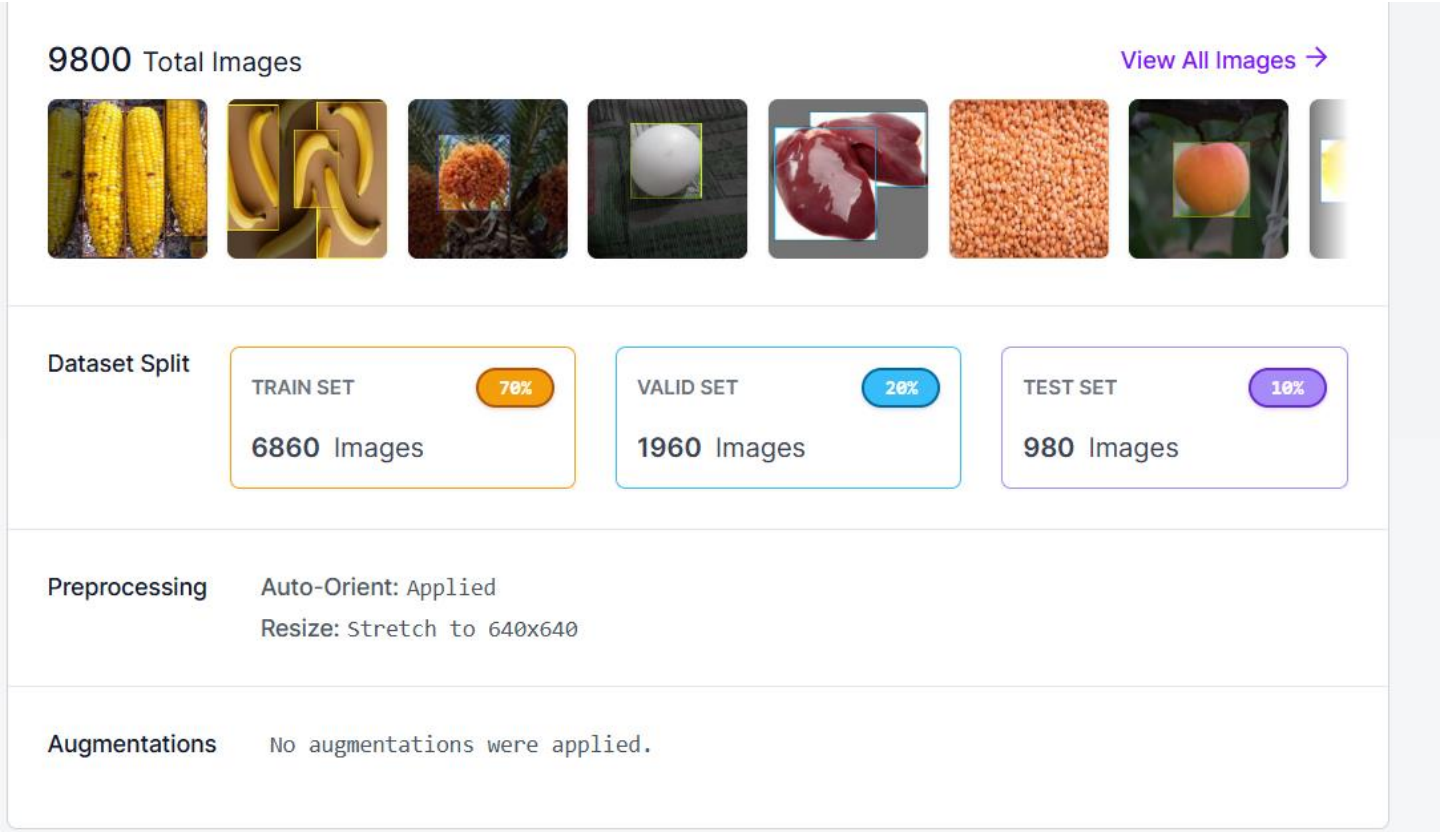
That represent the shape of box:



You can see that most image has only 1 object or from 1 – 3 objects

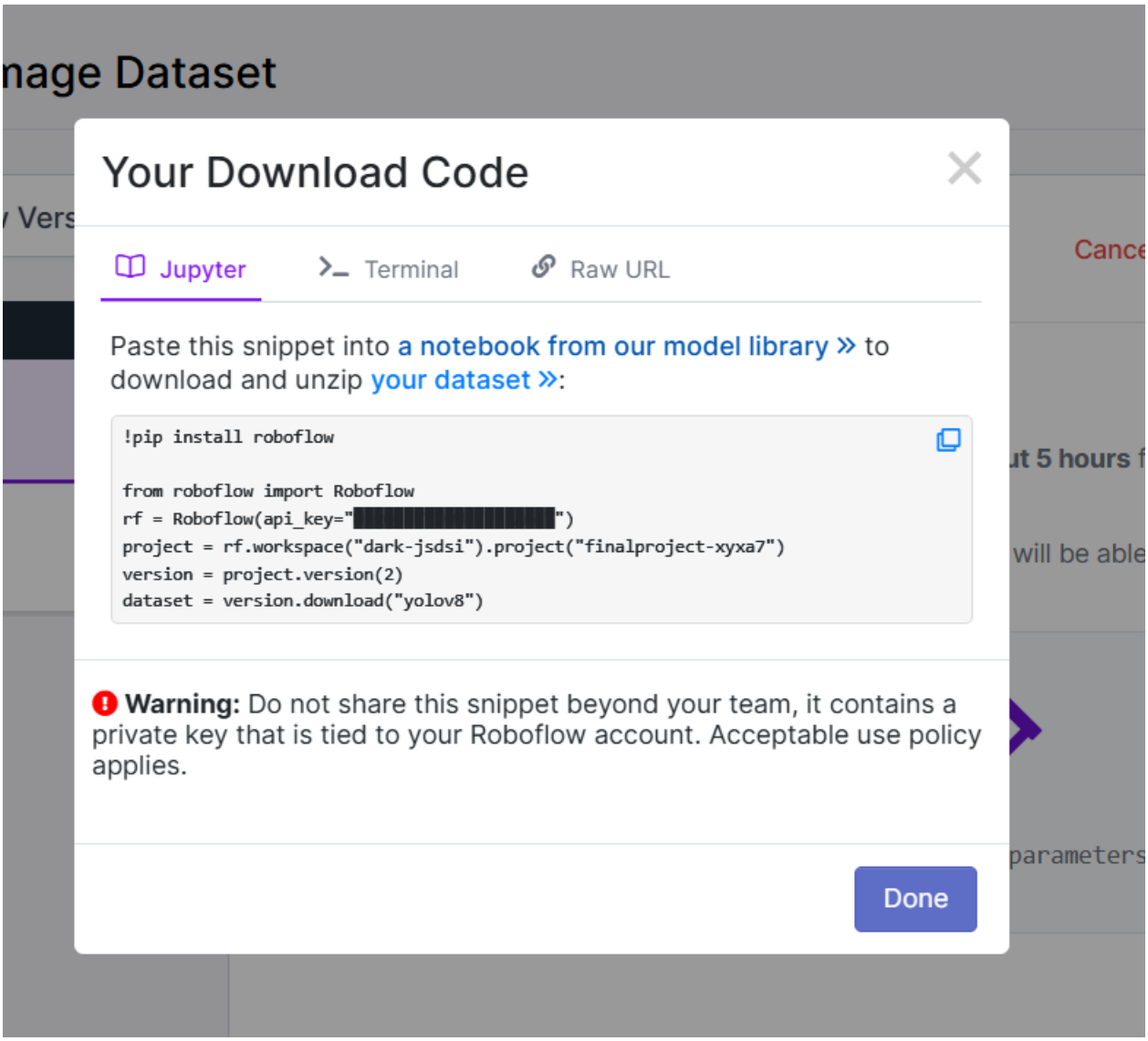


6- The final Dataset details



7-Export Dataset to train yoloV8 on Google Collab

Finally, after completing the work on the dataset, the Roboflow website allows you to obtain your dataset in different formats without downloading the dataset to the local machine. Here we chose this solution because we use Google Collab to train our model.



8-Dataset macronutrients details:

You can see that this file is linked to our data, as it provides detailed information about the types we have, which we will use to display to the user of our application. This file consists of 49 rows representing the different types we have, and 11 columns.

Sample:

	A	B	C	D	E	F	G	H	I	J	K
1	NAME	CLASS	D_Kcal	D_Ptr	D_Carb	D_Fat	Calories (KCal)	Protein (g)	Carbohydrates (g)	Fat (g)	NOTES:
2	coconut	HLHL	high	low	high	low	354	3.33	15.23	33.49	Coconut is high in healthy fats, providing a quick source of energy, an
3	corn	HLHL	high	low	high	low	86	3.27	19.02	1.35	Corn is a good source of carbohydrates for energy and fiber for digest
4	pasta	HLHL	high	low	high	low	131	5.47	25.77	1.02	Pasta is a carbohydrate-rich food, providing energy for the body, but
5	Almonds	HLLH	high	low	low	high	576	21.15	21.55	49.93	Almonds are high in healthy fats, protein, and fiber, supporting heart
6	Apple	LLLL	low	low	low	low	52	0.26	13.81	0.17	Apples are low in calories and high in fiber, supporting digestive heal
7	Apricot	MLML	moderate	low	moderate	low	48	1.4	11.12	0.39	Apricots are low in calories and high in fiber, vitamins A and C, and a
8	banana	HLHL	high	low	high	low	105	1.3	27	0.4	a modest amount of muscle-supporting protein. They're packed with
9	Broccoli	LLLL	low	low	low	low	34	2.82	6.64	0.37	Broccoli is low in calories and high in fiber, vitamins C and K, and anti
10	Cabbage	LLLL	low	low	low	low	25	1.28	5.8	0.1	Cabbage is low in calories and high in fiber and vitamin C, supporting
11	Carrot	LLLL	low	low	low	low	41	0.93	9.58	0.24	Carrots are rich in beta-carotene, supporting eye health, and are low
12	Cashew	HLLH	high	low	low	high	553	18.22	30.19	43.85	Cashews are a good source of healthy fats, protein, and minerals like
13	cauliflower	HLHL	high	low	high	low	25	1.9	4.97	0.28	Cauliflower is low in calories and carbohydrates but high in fiber, mal
14	Celery	LLLL	low	low	low	low	16	0.69	3	0.17	Celery is low in calories and high in water content, making it a hydrat
15	Cherry	LLLL	low	low	low	low	50	1.06	12.18	0.3	Cherries are rich in antioxidants and anti-inflammatory compounds, s
16	Chicken	LHLM	low	high	low	moderate	165	31	0	3.6	Chicken is a lean source of protein, supporting muscle growth and re
17	chickpeas	HLHL	high	low	high	low	164	8.86	27.42	2.59	Chickpeas are rich in protein, fiber, and carbohydrates, providing sust
18	Cucumber	LLLL	low	low	low	low	15	0.65	3.63	0.11	Cucumbers are low in calories and high in water content, providing h
19	dates	HLHL	high	low	high	low	282	2.45	75.03	0.39	Dates are high in natural sugars, providing quick energy, and are rich
20	Eggplant	LLLL	low	low	low	low	25	1	6	0.18	Eggplant is low in calories and carbohydrates but high in fiber, suppo

Name: represent the name of food or vegetables ...etc.

Class: Indicates the class to which it belongs based on the proportions of macronutrients

D_Kcal, D_Ptr, D_Carb, D_Fat: represent the percentage of calories, protein, carbohydrates, fat in this type

Calories, Protein, carbohydrates, Fat: represent the grams of calories, protein, carbohydrates, fat in this type

Notes: A brief paragraph explaining tips and warnings for each type

All of this information has been reviewed by a specialized nutritionist

4.2 YOU ONLY LOOK ONCE(YOLO)

You Only Look Once (YOLO) is a state-of-the-art, real-time object detection algorithm introduced in 2015 by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in their famous research paper “You Only Look Once: Unified, Real-Time Object Detection”.

The authors frame the object detection problem as a regression problem instead of a classification task by spatially separating bounding boxes and associating probabilities to each of the detected images using a single convolutional neural network (CNN).

4.2.1 WHAT MAKES YOLO POPULAR FOR OBJECT DETECTION?

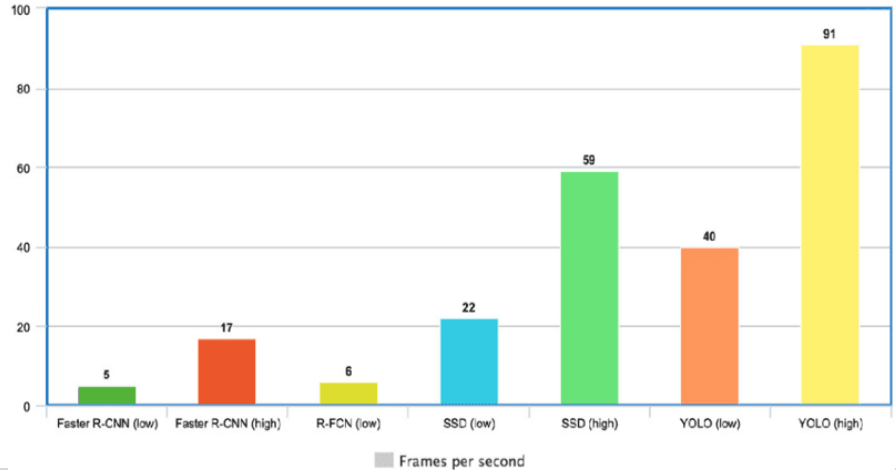
Some of the reasons why YOLO is leading the competition include its:

- Speed
- Detection accuracy
- Good generalization
- Open-source

1- Speed

YOLO is extremely fast because it does not deal with complex pipelines. It can process images at 45 Frames Per Second (FPS). In addition, YOLO reaches more than twice the mean Average Precision (mAP) compared to other real-time systems, which makes it a great candidate for real-time processing.

From the graphic below, we observe that YOLO is far beyond the other object detectors with 91 FPS.



2- High detection accuracy

YOLO is far beyond other state-of-the-art models in accuracy with very few background errors.

3- Better generalization

This is especially true for the new versions of YOLO, which will be discussed later in the article. With those advancements, YOLO pushed a little further by providing a better generalization for new domains, which makes it great for applications relying on fast and robust object detection.

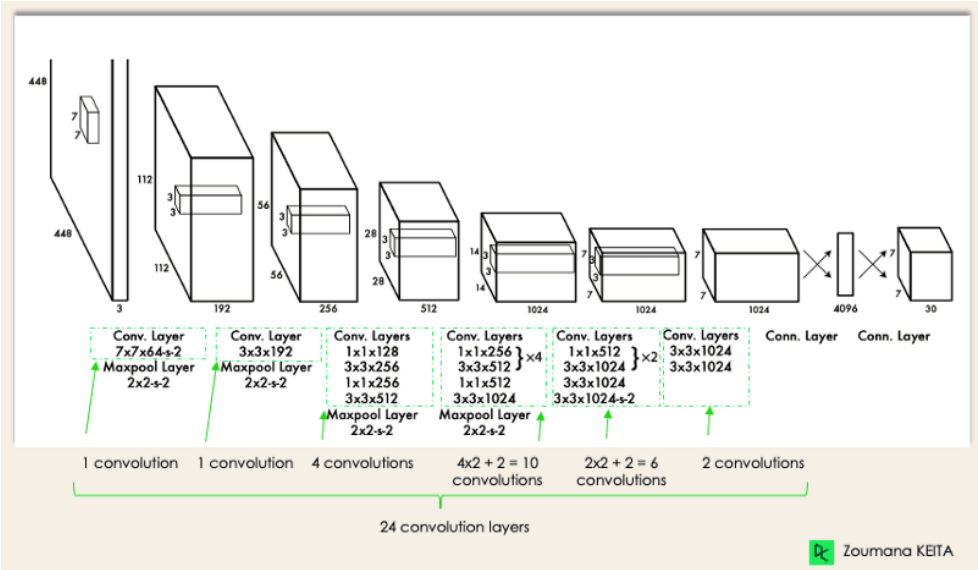
For instance the Automatic Detection of Melanoma with Yolo Deep Convolutional Neural Networks paper shows that the first version YOLOv1 has the lowest mean average precision for the automatic detection of melanoma disease, compared to YOLOv2 and YOLOv3.

4- Open source

Making YOLO open-source led the community to constantly improve the model. This is one of the reasons why YOLO has made so many improvements in such a limited time.

YOLO Architecture

YOLO architecture is similar to GoogleNet. As illustrated below, it has overall 24 convolutional layers, four max-pooling layers, and two fully connected layers.



The architecture works as follows:

- Resizes the input image into 448x448 before going through the convolutional network.
- A 1x1 convolution is first applied to reduce the number of channels, which is then followed by a 3x3 convolution to generate a cuboidal output.
- The activation function under the hood is ReLU, except for the final layer, which uses a linear activation function.
- Some additional techniques, such as batch normalization and dropout, respectively regularize the model and prevent it from overfitting.

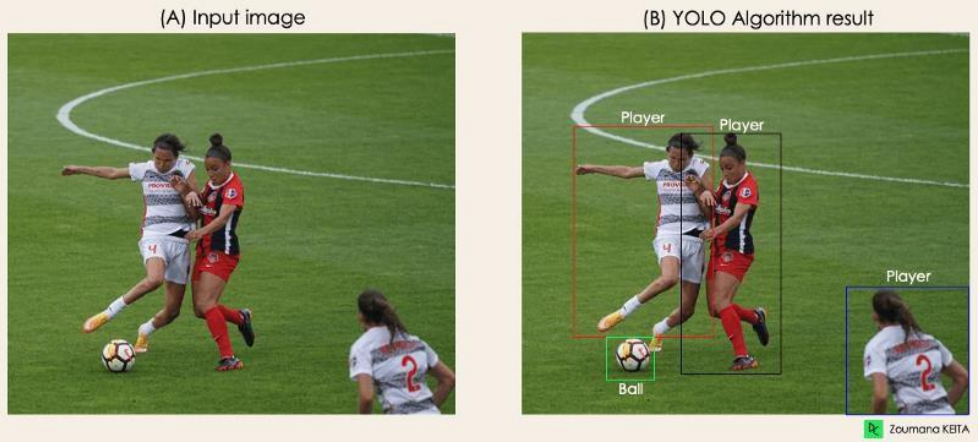
4.2.2 HOW DOES YOLO OBJECT DETECTION WORK?

Now that you understand the architecture, let's have a high-level overview of how the YOLO algorithm performs object detection using a simple use case.

"Imagine you built a YOLO application that detects players and soccer balls from a given image.

But how can you explain this process to someone, especially non-initiated people?

→ That is the whole point of this section. You will understand the whole process of how YOLO performs object detection; how to get image (B) from image (A)"



The algorithm works based on the following four approaches:

- Residual blocks
- Bounding box regression
- Intersection Over Unions or IOU for short
- Non-Maximum Suppression.

1- Residual blocks

This first step starts by dividing the original image (A) into NxN grid cells of equal shape, where N in our case is 4 shown on the image on the right. Each cell in the grid is responsible for localizing and predicting the class of the object that it covers, along with the probability/confidence value.

2- Bounding box regression

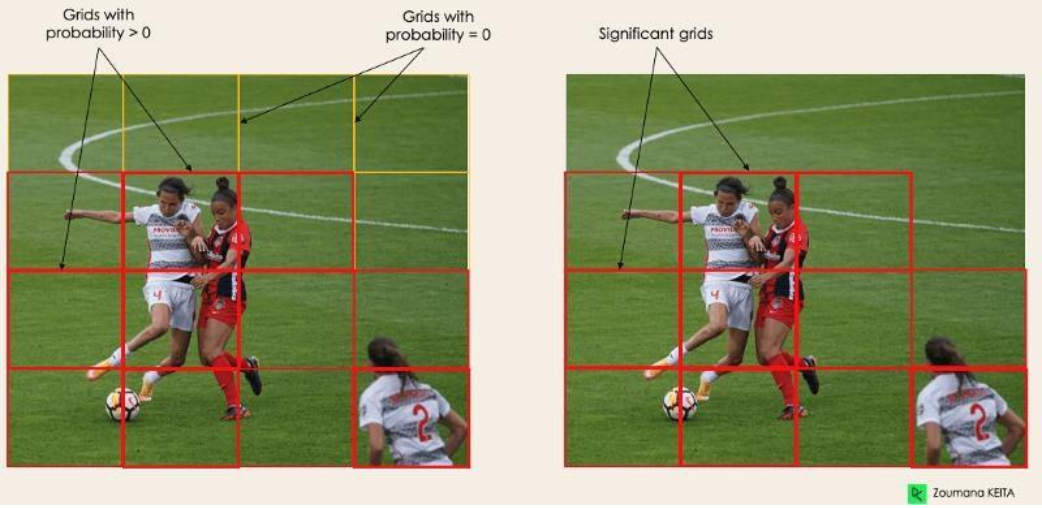
The next step is to determine the bounding boxes which correspond to rectangles highlighting all the objects in the image. We can have as many bounding boxes as there are objects within a given image.

YOLO determines the attributes of these bounding boxes using a single regression module in the following format, where Y is the final vector representation for each bounding box.

Y = [pc, bx, by, bh, bw, c1, c2]

This is especially important during the training phase of the model.

- pc corresponds to the probability score of the grid containing an object. For instance, all the grids in red will have a probability score higher than zero. The image on the right is the simplified version since the probability of each yellow cell is zero (insignificant).

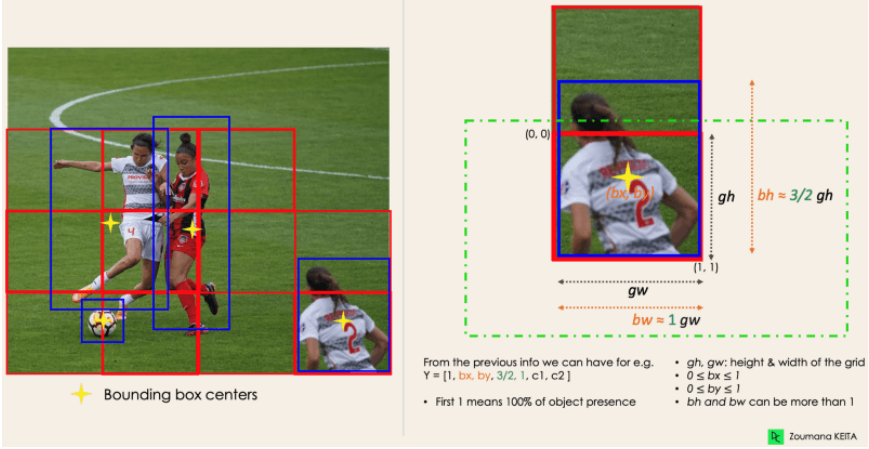


bx, by are the x and y coordinates of the center of the bounding box with respect to the enveloping grid cell.

bh, bw correspond to the height and the width of the bounding box with respect to the enveloping grid cell.

c1 and c2 correspond to the two classes Player and Ball. We can have as many classes as your use case requires.

To understand, let's pay closer attention to the player on the bottom right.

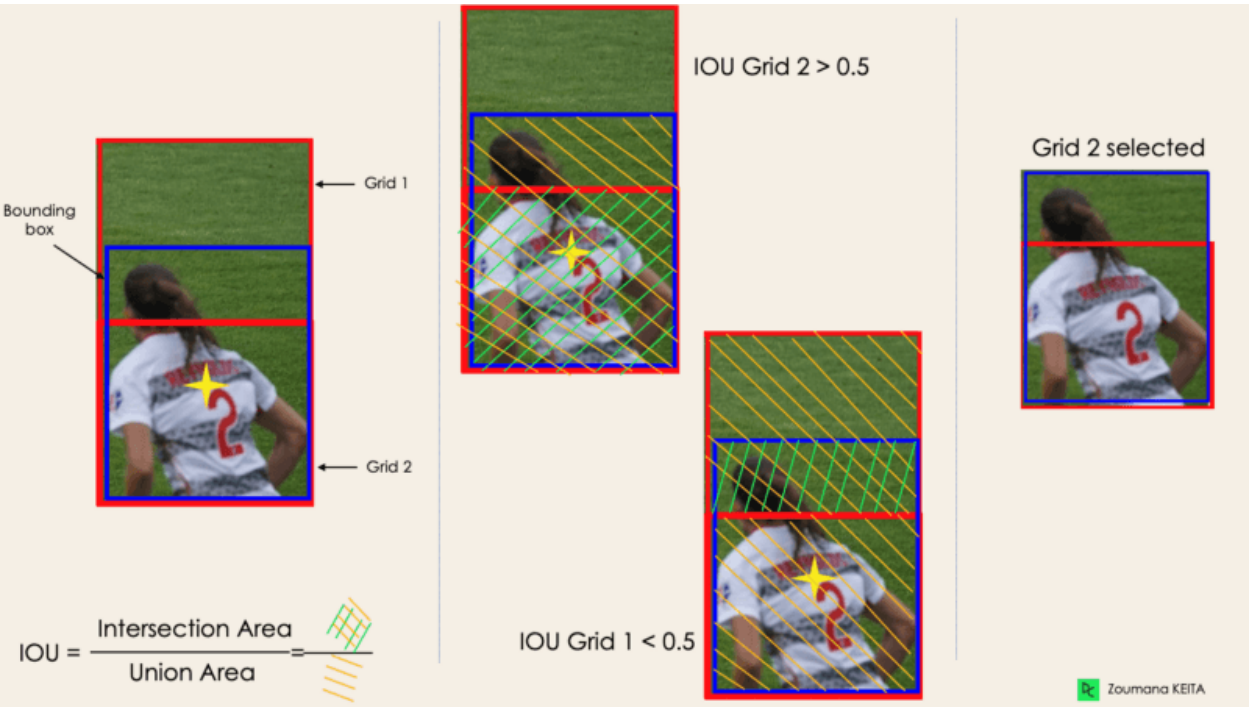


3- **Intersection Over Unions or IOU**

Most of the time, a single object in an image can have multiple grid box candidates for prediction, even though not all of them are relevant. The goal of the IOU (a value between 0 and 1) is to discard such grid boxes to only keep those that are relevant. Here is the logic behind it:

- The user defines its IOU selection threshold, which can be, for instance, 0.5.
- Then YOLO computes the IOU of each grid cell which is the Intersection area divided by the Union Area.
- Finally, it ignores the prediction of the grid cells having an IOU ≤ threshold and considers those with an IOU > threshold.

Below is an illustration of applying the grid selection process to the bottom left object. We can observe that the object originally had two grid candidates, then only "Grid 2" was selected at the end.



4- **Non-Max Suppression or NMS**

Setting a threshold for the IOU is not always enough because an object can have multiple boxes with IOU beyond the threshold, and leaving all those boxes might include noise. Here is where we can use NMS to keep only the boxes with the highest probability score of detection.

4.2.3 Compare between yoloV8 vs yoloV9:

Executive Summary:

The dynamic evolution of object detection algorithms reaches a pinnacle with the inception of YOLOv9, a groundbreaking iteration designed to surpass its predecessors in both performance and accuracy. This comprehensive analysis delves into the technical intricacies of YOLOv9, elucidating its architectural innovations and performance enhancements that distinguish it from YOLOv8. By unraveling the core advancements, this article aims to provide a holistic understanding of why YOLOv9 emerges as the undisputed champion in object detection. In this article we will discuss comparison between YOLOv9 vs YOLOv8.

Introduction:

YOLOv9 emerges as a cutting-edge model, boasting innovative features that will play an important role in the further development of object detection, image segmentation, and classification. The new top-tier features allow faster, sharper, and more versatile actions.

Architectural Evolution: YOLOv8 vs. YOLOv9:

The architectural disparity between YOLOv9 vs YOLOv8 underpins the substantial performance gains witnessed in the latter. YOLOv8, while revolutionary in its own right, lacked certain key components that impeded its detection prowess. YOLOv9 addresses these shortcomings through a series of strategic architectural enhancements.
In YOLOv8, the backbone architecture primarily relied on DarkNet-53, a deep neural network consisting of 53 convolutional layers. While effective, DarkNet-53 exhibited limitations in capturing fine-grained features across different scales, particularly for small and occluded objects. YOLOv9 addresses this limitation by integrating a feature pyramid network (FPN) directly into its architecture.

YOLOv9 vs YOLOv8 – What's Different?

The creators of YOLOv9 introduced a new idea called Programmable Gradient Information (PGI) to solve the problem of losing data during the process of passing information forward. This PGI concept helps generate reliable gradients through an extra reversible branch in the model. This branch works alongside the main task, ensuring that important features aren't lost. By applying PGI at different levels of meaning, they achieved the best training results. The reversible structure of PGI is included in the extra branch, so it doesn't add extra cost. PGI also allows for choosing suitable loss functions, solving issues faced in mask modeling. This PGI method can be used in various sizes of deep neural networks.

In their paper, they also developed a tool called Generalized ELAN (GELAN), which considers factors like parameter count, complexity, accuracy, and speed of inference. GELAN lets users pick different computational blocks for different devices for running the model efficiently.

Using PGI and GELAN, they created YOLOv9. They tested it on the MS COCO dataset and found it performed the best in all scenarios. This iteration seeks to outperform both convolution-based and transformer-based methods in object detection. YOLO v9 introduces four models, categorized by parameter count: v9-S, v9-M, v9-C, and v9-E, each targeting different use cases and computational resource requirements

Conclusion:

In conclusion, while YOLOv8 excels in accurately recognizing objects, its tendency to detect non-existent objects leads to a higher false positive rate. YOLOv9, on the other hand, takes a more conservative approach, resulting in fewer false positives but potentially missing some actual objects, leading to a higher false negative rate. Despite this trade-off, YOLOv9 represents a significant advancement in real-time object detection, thanks to its improved training methods and practices, as well as innovative solutions like PGI and GELAN. These enhancements not only boost efficiency, accuracy, and adaptability but also set a new standard for future research and applications in the field. As the AI community continues to progress, YOLOv9 serves as a testament to the collaborative spirit and innovative thinking driving technological advancements forward.

Difference between result in our dataset:

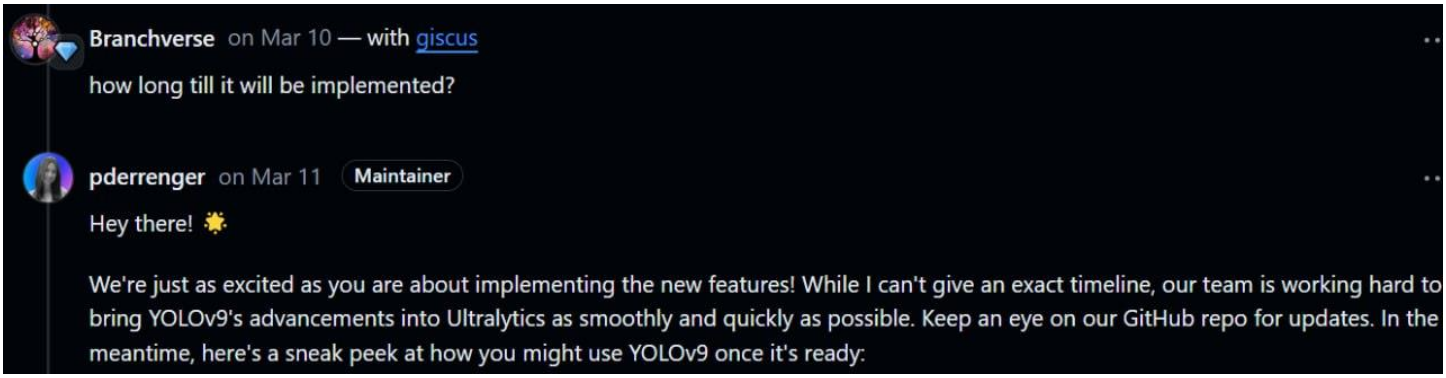
Result of YOLOV8:

```
/content
Ultralytics YOLOv8.0.196 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (NVIDIA L4, 22700MiB)
Model summary (fused): 168 layers, 3015203 parameters, 0 gradients, 8.1 GFLOPs
val: Scanning /content/datasets/FinalProject-2/valid/labels.cache... 1960 images, 0 backgrounds, 0 corrupt: 100% 1960/1960 [00:00<?, ?it
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95):   0% 0/123 [00:00<?, ?it/s]/usr/local/lib/pyth
  return F.conv2d(input, weight, bias, self.stride,
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 123/123 [00:14<00:00,  8.67it/s]
                   all       1960       3715      0.815      0.804      0.851       0.63
                Carrot       1960         37      0.794      0.784      0.849      0.655
               Yoghurt       1960         42      0.908      0.705      0.841      0.434
```

Result of YOLOV9:

```
gelan-c summary: 467 layers, 25448739 parameters, 0 gradients
val: Scanning /content/yolov9/FinalProject-2/valid/labels.cache... 1960 images, 0 backgrounds, 0 corrupt: 100% 1960/1960 [00:00<?, ?it/s
/usr/lib/python3.10/multiprocessing/popen_fork.py:66: RuntimeWarning: os.fork() was called. os.fork() is incompatible with multithreaded
  self.pid = os.fork()
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 62/62 [00:46<00:00,  1.34it/s]
                   all       1960       3715      0.858      0.842       0.89      0.685
                Carrot       1960         37       0.87      0.901      0.955      0.775
               Yoghurt       1960         42      0.847       0.69      0.883      0.518
```

From the results, it appears that yolov9 is superior to yolov8, but we had to use yolov8 because the use of yolov9 on the ultralytics library is not yet available.



4.2.4 Why should I use YOLOv8?

A few of the main reasons you should consider using YOLOv8 in your next computer vision project are:

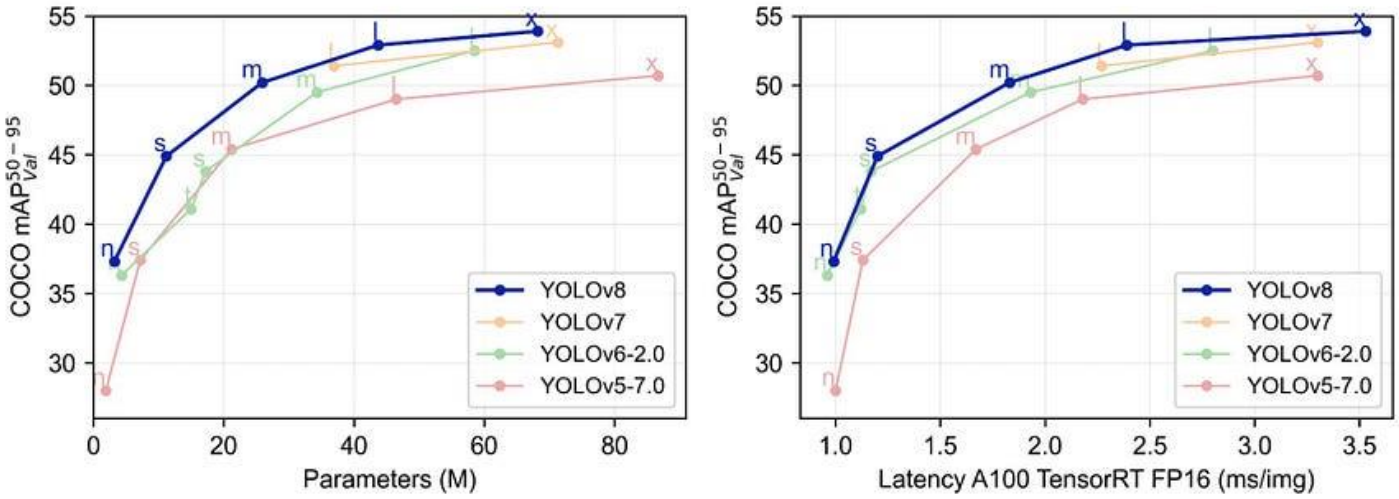
- YOLOv8 has better accuracy than previous YOLO models.
- The latest YOLOv8 implementation comes with a lot of new features, we especially like the user-friendly CLI and GitHub repo.
- It supports object detection, instance segmentation, and image classification.
- Training of YOLOv8 will be probably faster than the other two-stage object detection models.

One reason not to use YOLOv8:

- At the current time YOLOv8 does not support models trained in 1280 (in pixels) resolution, thus if you're looking to run inference at high resolution it is not recommended to use YOLOv8

4.2.5 How does YOLOv8 compare to previous models?

The Ultralytics team has once again benchmarked YOLOv8 against the COCO dataset and achieved impressive results compared to previous YOLO versions across all five model sizes.



When comparing the performance of the different YOLO lineages and model sizes on the COCO dataset we want to compare different metrics.

- **Performance:** Mean average precision (mAP)
- **Speed:** Speed of the inference (In fps)
- **Compute (cost):** The size of the model in FLOPs and params

For the object detection comparison of the 5 model sizes The YOLOv8m model achieved an **mAP of 50.2%** on the COCO dataset, whereas the largest model, YOLOv8x achieved **53.9%** with more than double the number of parameters.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

- mAP^{val} values are for single-model single-scale on COCO val2017 dataset. Reproduce by `yolo val detect data=coco.yaml device=0`
- **Speed** averaged over COCO val images using an Amazon EC2 P4d instance. Reproduce by `yolo val detect data=coco128.yaml batch=1 device=0/cpu`

4.3 Training YOLOV8 On our custom Dataset:

The code installs the "ultralytics" library for YOLO, clears the output screen for clarity, imports the library, and checks the system setup to ensure everything is ready for using the "ultralytics" features.

```
!pip install ultralytics==8.0.196

from IPython import display
display.clear_output()

import ultralytics
ultralytics.checks()
```

```
Ultralytics YOLOv8.0.196 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (NVIDIA L4, 22700MiB)
Setup complete ✅ (12 CPUs, 53.0 GB RAM, 30.2/201.2 GB disk)
```

Using YOLO Models: By importing the YOLO class, you can create, train, and use YOLO models for various computer vision tasks such as object detection.

Displaying Images: With the display and Image functions, you can visualize images directly within your Jupyter Notebook, which is useful for inspecting model predictions and results.

This code sets up a dataset for use with YOLOv8 by creating and navigating to a `datasets` directory, installing the `roboflow` library, and initializing a Roboflow object with an API key to access the Roboflow API. It retrieves a specific project and version within a Roboflow workspace and downloads the dataset in a format compatible with YOLOv8, making it ready for training or evaluation.

```
[ ] from ultralytics import YOLO

    from IPython.display import display, Image
```

```
!mkdir {HOME}/datasets
%cd {HOME}/datasets

!pip install roboflow --quiet

from roboflow import Roboflow
rf = Roboflow(api_key="WggQsHcV20WJfJ6uAGqT")
project = rf.workspace("dark-jsdsi").project("finalproject-xyxa7")
version = project.version(2)
dataset = version.download("yolov8")
```

```
/content/datasets
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 3.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 178.7/178.7 kB 7.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.5/54.5 kB 8.3 MB/s eta 0:00:00
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in FinalProject-2 to yolov8:: 100%|██████████| 385482/385482 [00:05<00:00, 65080.48it/s]

Extracting Dataset Version Zip to FinalProject-2 in yolov8:: 100%|██████████| 19612/19612 [00:02<00:00, 7743.60it/s]
```

This code initiates the training of a YOLOv8 model for object detection. It begins by navigating to the home directory, then uses a command to start training the model specified by yolov8n.pt using the dataset defined in data.yaml. The training is configured to run for 50 epochs with an image size of 640 pixels, and it enables plotting to visualize the training progress.

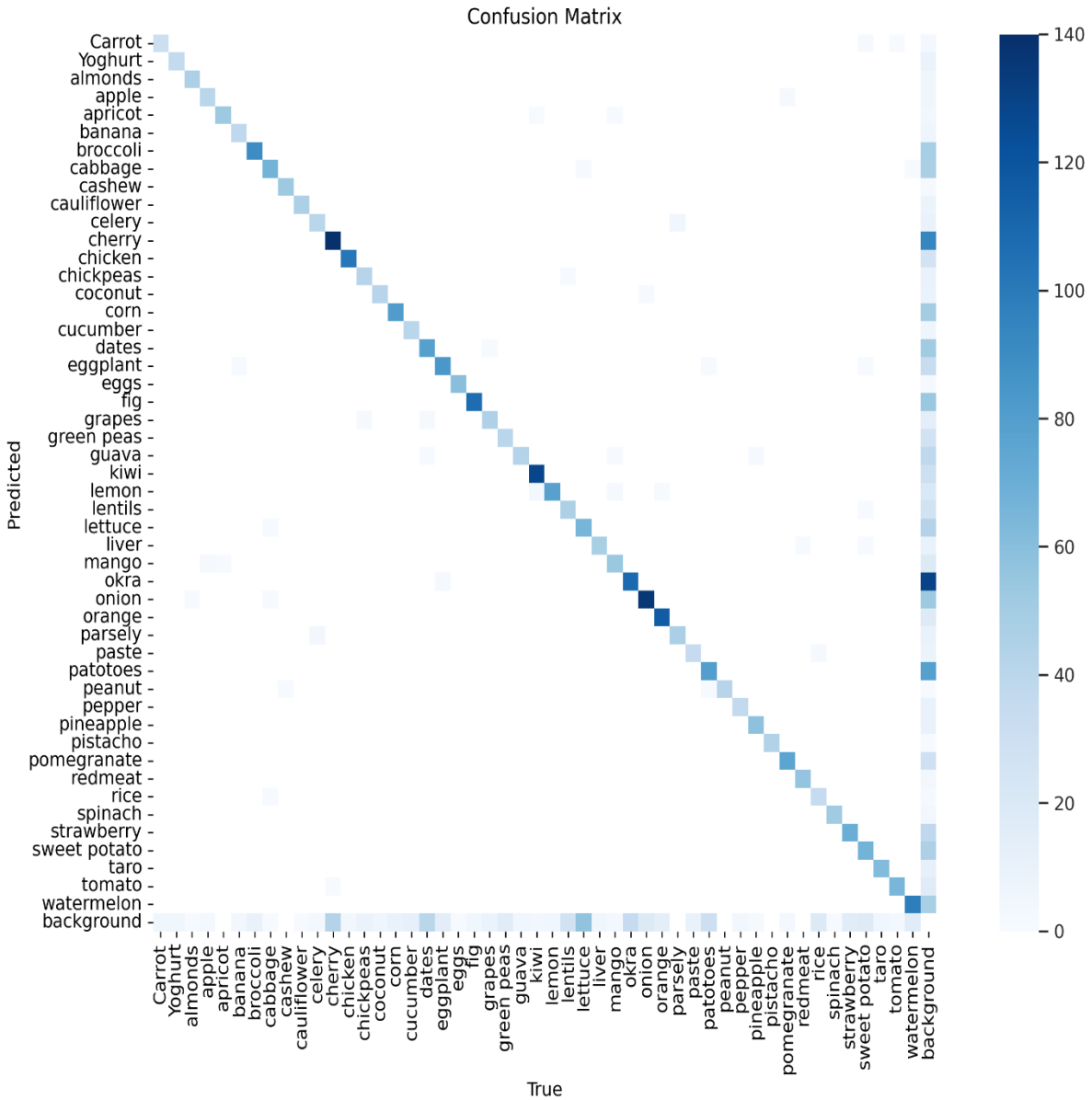
```
!yolo task=detect mode=train model= /content/yolov8n.pt data= /content/datasets/FinalProject-2/data.yaml epochs=50 imgsz=640 plots=True
```

```
/content
New https://pypi.org/project/ultralytics/8.2.36 available 😃 Update with 'pip install -U ultralytics'
Ultralytics YOLOv8.0.196 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (NVIDIA L4, 22700MiB)
engine/trainer: task=detect, mode=train, model=/content/yolov8n.pt, data=/content/datasets/FinalProject-2/data.yaml, epochs=50, patience
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultralytics/Arial.ttf'...
100% 755k/755k [00:00<00:00, 24.2MB/s]
2024-06-20 17:00:35.301704: E external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:9261] Unable to register cuDNN factory: Attempting
2024-06-20 17:00:35.301760: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:607] Unable to register cuFFT factory: Attempting
2024-06-20 17:00:35.303490: E external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1515] Unable to register cuBLAS factory: Attempti
Overriding model.yaml nc=80 with nc=49

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1     37248  ultralytics.nn.modules.block.C2f             [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    123648  ultralytics.nn.modules.block.C2f             [192, 128, 1]
 19                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 22        [15, 18, 21]  1    760867  ultralytics.nn.modules.head.Detect           [49, [64, 128, 256]]
Model summary: 225 layers, 3020403 parameters, 3020387 gradients, 8.2 GFLOPs
```

```
16m
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      45/50      2.21G      0.833     0.8476      1.308         21        640: 100% 429/429 [00:40<00:00, 10.53it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.68it/s]
                   all       1960       3715      0.802      0.802      0.845      0.624

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      46/50      2.21G     0.8255     0.8354      1.302         27        640: 100% 429/429 [00:40<00:00, 10.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.78it/s]
                   all       1960       3715      0.796      0.816      0.847      0.627

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      47/50      2.21G     0.8154     0.8213      1.295         19        640: 100% 429/429 [00:40<00:00, 10.66it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.76it/s]
                   all       1960       3715      0.815      0.801      0.844      0.625

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      48/50      2.21G     0.8192     0.8179      1.295         17        640: 100% 429/429 [00:40<00:00, 10.59it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.76it/s]
                   all       1960       3715      0.812      0.806      0.849      0.629

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      49/50      2.21G     0.8075     0.8003      1.289         15        640: 100% 429/429 [00:40<00:00, 10.60it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.72it/s]
                   all       1960       3715      0.816      0.802      0.851       0.63

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      50/50      2.21G     0.8053     0.7945      1.281         25        640: 100% 429/429 [00:40<00:00, 10.58it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 62/62 [00:10<00:00,  5.73it/s]
                   all       1960       3715      0.801      0.815      0.847      0.628

50 epochs completed in 0.733 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 6.3MB
Optimizer stripped from runs/detect/train/weights/best.pt, 6.3MB

Validating runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.0.196 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (NVIDIA L4, 22700MiB)
```

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/confusion_matrix.png', width=1200)
```

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. In the context of object detection, each cell in the matrix represents the number of predictions made for each class versus the actual instances of each class.

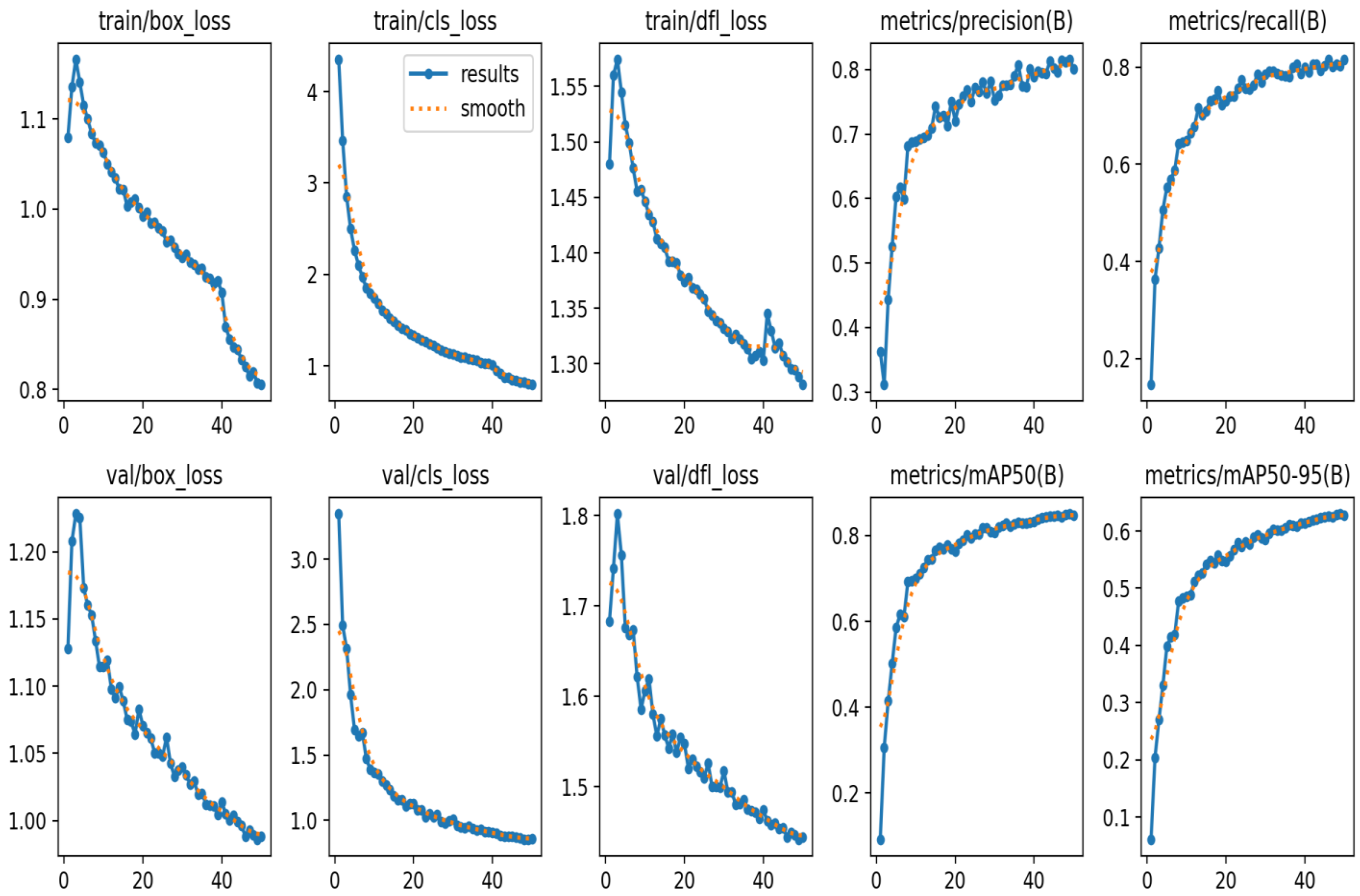


Importance of the Confusion Matrix

- **Accuracy Measurement**: It helps in calculating various performance metrics like precision, recall, and F1-score for each class.
- **Error Analysis**: By examining where the model makes mistakes (off-diagonal elements), you can identify specific classes that are frequently confused with each other, which can provide insights into model improvements.
- **Model Improvement**: Understanding the confusion matrix can guide you in augmenting your training data, adjusting model architecture, or applying different post-processing techniques to reduce specific types of errors.

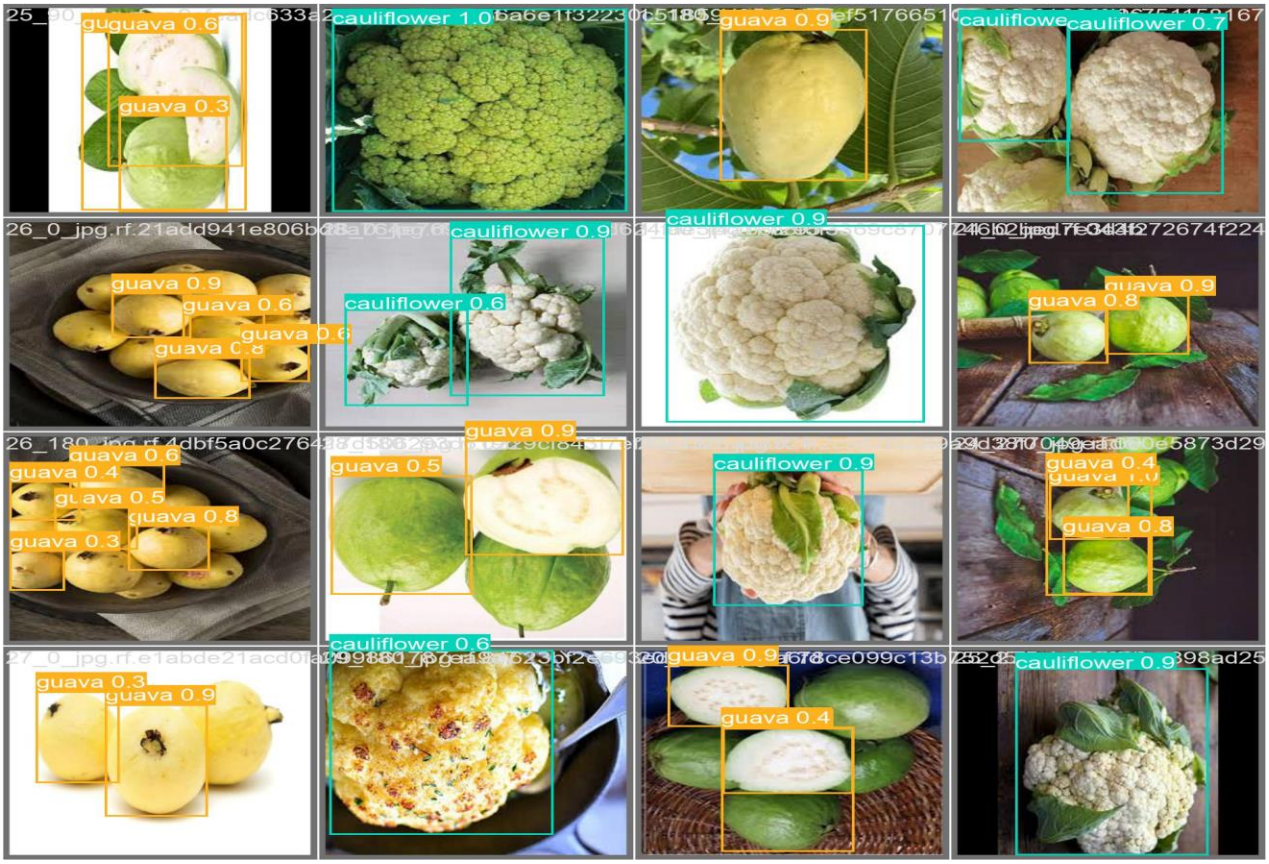
results of training model along 50 epochs.

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/results.png', width=1000)
```



Validation:

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/val_batch1_pred.jpg', width=600)
```



```
%cd {HOME}

!yolo task=detect mode=val model={HOME}/runs/detect/train/weights/best.pt data={dataset.location}/data.yaml
```

```
/content
Ultralytics YOLOv8.0.196 🚀 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (NVIDIA L4, 22700MiB)
Model summary (fused): 168 layers, 3015203 parameters, 0 gradients, 8.1 GFLOPs
val: Scanning /content/datasets/FinalProject-2/valid/labels.cache... 1960 images, 0 backgrounds, 0 corrupt: 100% 1960/1960 [00:00<?, ?it
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95):   0% 0/123 [00:00<?, ?it/s]/usr/local/lib/pyth
  return F.conv2d(input, weight, bias, self.stride,
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 123/123 [00:14<00:00,  8.67it/s]
                   all       1960       3715      0.815      0.804      0.851       0.63
                Carrot       1960         37      0.794      0.784      0.849      0.655
               Yoghurt       1960         42      0.908      0.705      0.841      0.434
```

Chapter 5 MOBILE APPLLICATION

5.1 what is UI/UX ?

5.1.1 What is user experience (UX)?

User experience, or UX, is a term used to describe the overall experience a user has when interacting with a product or service in a given context. Depending on how the product or service is designed, the experience can range from delightful to downright frustrating!

You'll often hear about UX in relation to digital products, such as websites and apps—but UX isn't limited to the digital space. Anything that can be experienced can be designed, from the packaging of a toothbrush to the wheels of an orthopedic chair.

The impact of good (and bad!) UX is everywhere. That's one of the reasons it's such an exciting field, and also explains why you already know a lot more about UX than you realize. Every time you curse a push door that has a pull bar, or close a confusing website in frustration, you're making a judgment on the quality of its UX design.

So: UX is all about the user's interaction or experience with a product or service. With that in mind, let's move on to part two…

5.1.2 What is user interface (UI)?

UI (User Interface), on the other hand, focuses on the visual and interactive elements of a digital product. It deals with the design of the user interface components, such as buttons, icons, menus, forms, and layouts.

UI design aims to create an aesthetically pleasing and visually appealing interface that enhances the user experience. It involves selecting colors, typography, graphics, and other visual elements to create a cohesive and attractive design.

UI designers work closely with UX designers to ensure that the interface aligns with the overall user experience goals and provides a seamless and visually engaging interaction.

In summary, UX design is concerned with the overall user experience, while UI design is focused on the visual and interactive aspects of the user interface. Both UX and UI design are crucial in creating successful and user-friendly digital products, and they often work in tandem to deliver a cohesive and enjoyable user experience.

5.2 INTRODUCTION

In the contemporary digital age, maintaining optimal health and nutrition has become increasingly important yet challenging. With the fast-paced lifestyle and the overwhelming amount of dietary information available, individuals often struggle to make informed nutritional choices.

Proper nutrition supports bodily functions, boosts the immune system, enhances energy levels, and reduces the risk of chronic diseases such as obesity, diabetes, and cardiovascular conditions. Additionally, a healthy lifestyle can improve mood, cognitive function, and longevity, making it essential for individuals to prioritize their health in daily life.

AI can enhance the accuracy and efficiency of nutrition tracking. Machine learning models can identify and quantify food items from photos, estimate portion sizes, and calculate nutritional content with high precision. This reduces the burden of manual tracking and ensures users receive accurate information about their dietary intake.

In conclusion, maintaining a healthy body is essential for overall well-being, and AI plays a pivotal role in facilitating this process. By providing personalized, accurate, and actionable insights, AI technology empowers individuals to make healthier choices and achieve their nutritional goals more effectively. As AI continues to evolve, its integration into health and nutrition management will undoubtedly become increasingly important, offering innovative solutions to support a healthier and more informed society.

APP ROLE

Our mobile application is designed to serve as a comprehensive platform for individuals who are committed to maintaining and improving their health. By fostering a community-centric environment, the app not only provides personalized nutrition and health management tools but also encourages users to connect, share experiences, and support each other in their wellness journeys. Key features of the app include access to online doctors for professional guidance, and a machine learning component that analyzes food images to calculate nutritional content and record meals throughout the day.

Our APP stands out as a multifaceted tool designed to enhance the health and well-being of its users. By fostering a community of health enthusiasts, providing access to online doctors, and utilizing cutting-edge machine learning for nutritional analysis, the app offers a comprehensive and user-friendly solution for health management. This innovative approach empowers individuals to take control of their health, make informed decisions, and ultimately lead healthier, more fulfilling lives.

5.2.1 Features of the Application

Our mobile application is designed to be an all-encompassing tool for individuals focused on improving and maintaining their health. It boasts a range of features that cater to various aspects of health management, from community engagement to professional medical advice and advanced nutritional tracking. Here are the key features of the application:

- User Authentication
- Health-Focused Community
- Professional Medical Support
- Advanced Nutritional Analysis
- Health Metrics Tracking
- Meal Planning and Recipes

1- User Authentication

Login and Registration:

- The application offers secure login and registration pages to ensure that user data is protected. New users can create an account by providing basic information and setting up their profile, while returning users can log in with their credentials.
- Options for social media login or single sign-on (SSO) with platforms like Google to streamline the authentication process.

2- Health-Focused Community

Community Section:

One of the key aspects of the app is its ability to create a vibrant community of health-conscious individuals. Users can join the app, participate in discussions, share their progress and exchange tips and recipes. This community approach fosters a sense of belonging and motivation, as users can draw inspiration and support from others who share similar health goals. This promotes friendly competition and group accountability, which are proven ways to enhance adherence to health goals.

- A dedicated section for community interaction where users can publish posts, share their health journeys, success stories, and wellness tips.
- Users can engage with each other by liking and commenting on posts, fostering a supportive and interactive environment.

3- Professional Medical Support

Access to Professional Medical Guidance:

The inclusion of online doctors within the app significantly enhances its utility and credibility. Users can schedule virtual consultations, seek advice on dietary and health-related concerns, and receive personalized medical guidance. This feature ensures that users have access to professional support, helping them make informed decisions about their health. The availability of expert advice also bridges the gap between general health information and personalized medical care, providing users with a holistic approach to health management.

- Access a network of doctors online for consultations and follow-ups. Users can seek advice about health issues and receive personalized medical recommendations.
- Secure messaging with doctors for quick questions and follow-up discussions.

4- Advanced Nutritional Analysis through Machine Learning

Machine Learning for Food Identification:

- A standout feature of the app is its machine learning capability, which allows for the precise calculation of nutritional components in food through image analysis. By simply taking a photo of their meal, users can receive detailed information about its nutritional content, including calories, macronutrients, and micronutrients. This innovative technology not only simplifies the process of meal tracking but also enhances its accuracy, helping users maintain a comprehensive and accurate food diary.
- The machine learning algorithm continuously learns and improves from user data, ensuring that its nutritional assessments become more accurate over time. This feature is particularly useful for users with specific dietary requirements or those who are managing chronic conditions such as diabetes or hypertension. By providing real-time feedback and insights, the app empowers users to make healthier food choices and adjust their diets to meet their nutritional needs.

5- Health Metrics Tracking

BMI Calculator:

Built-in Body Mass Index (BMI) calculator where users can enter their height and weight to calculate their BMI. The app provides insights into what BMI means and provides recommendations for getting into a healthy BMI range.

The results can also be shared with a nutritionist or any other platform.

6- Meal Planning and Recipes

Meal Section:

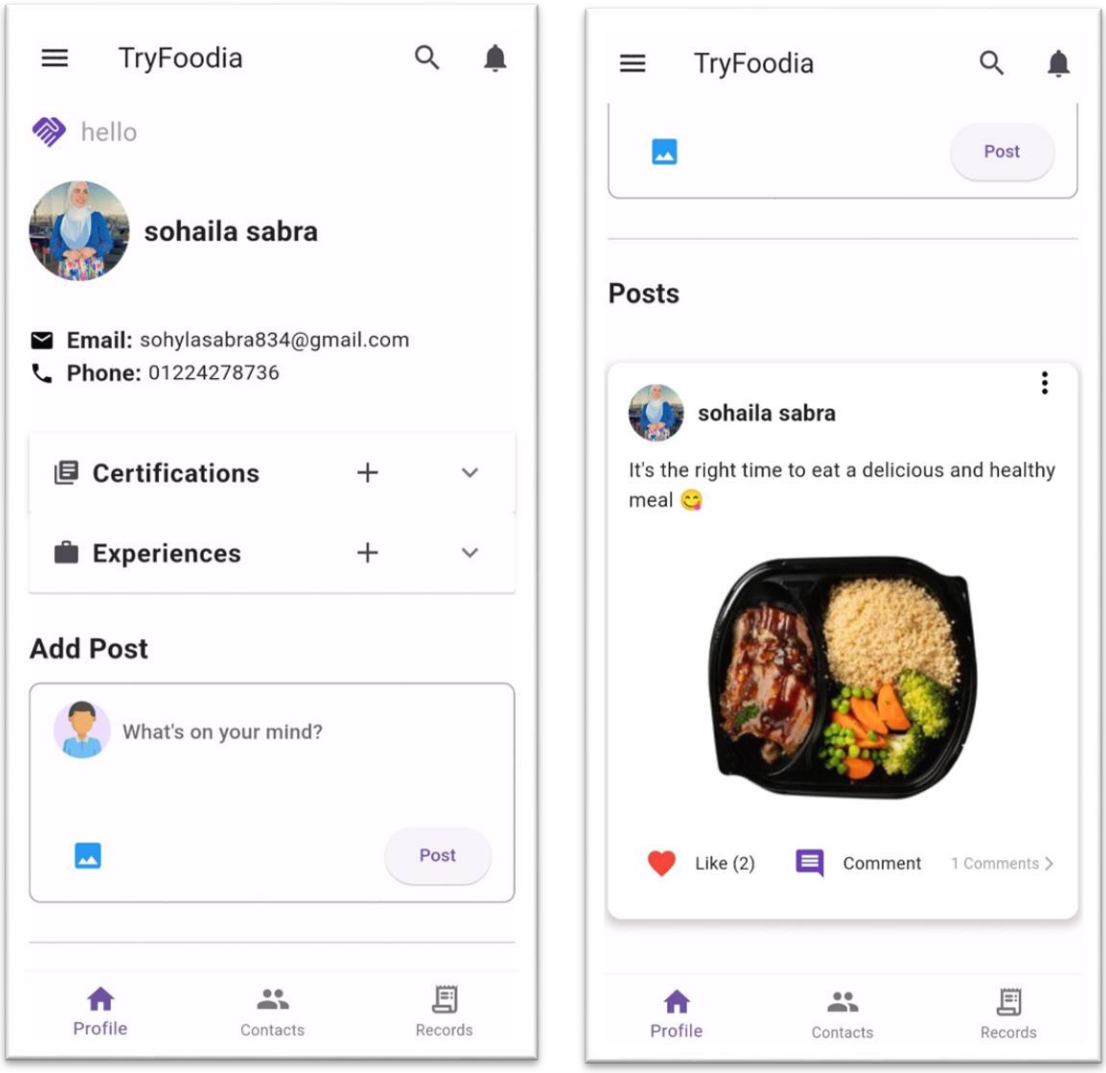
- A comprehensive meal section featuring a wide variety of recipes and meal ideas categorized by type (breakfast, lunch, dinner, snacks) and dietary needs (vegetarian, vegan, lactose, gluten).
- Advanced filtering options to help users find meals that meet their specific dietary requirements and preferences.
- Detailed nutritional information for each recipe, making it easy for users to plan balanced and healthy meals.
- Option to save favorite recipes.

5.3 THE USER INTERFACE

5.3.1 Profile Page

Overview

The Profile Page class is a pivotal component of a mobile application, designed to manage user profiles. It provides essential functionalities for displaying user information, managing posts, interacting with contacts, and searching for other users. This page enhances user engagement by enabling seamless navigation between different sections of the profile and facilitating interactions such as creating, editing, and liking posts.



Key Features

1. **User Profile Display**:
    - Displays user's profile information including their username, email, and phone number.
    - Allows users to update their profile picture with options to view, delete, or change the image.
2. **Posts Management**:
    - Users can create new posts with text and images.
    - Posts are displayed in a list format where users can interact with each post by liking, commenting, editing, or deleting them.
    - Each post displays the username, avatar, text content, and image (if any).
3. **Search Functionality**:
    - A search bar in the app bar allows users to search for other users by their username.
    - The search results are displayed dynamically as the user types.
4. **Navigation and Layout**:
    - The page includes a bottom navigation bar with three main sections: Profile, Contacts, and Records.
    - Uses a PageView to manage the different sections of the profile, allowing smooth transitions between pages.

5. **Loader and Shimmer Effect**:
    - Implements a shimmer effect to indicate loading state while user data is being fetched, enhancing user experience by providing visual feedback during data loading.

Detailed Breakdown

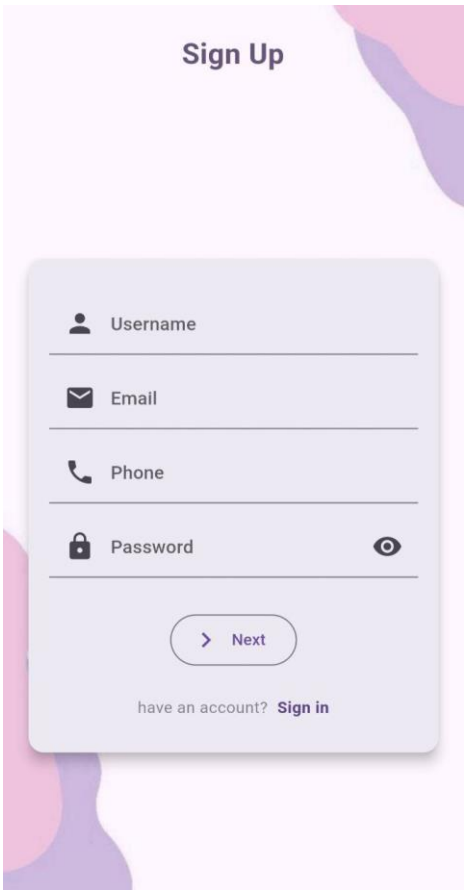
User Profile Section

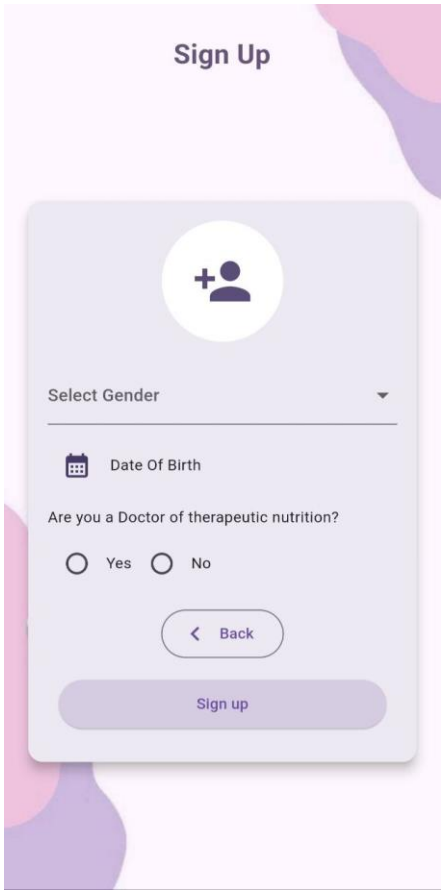
- **Profile Picture**: Users can tap on their profile picture to open a modal with options to delete, view, or change the image. The profile picture is displayed using a CircleAvatar widget.
- **Profile Information**: Basic user information such as email and phone number is displayed with corresponding icons for easy identification.

5.3.2 Sign Up screen

Overview

The Signup Page is a Flutter page that allows users to sign up for an account. The sign-up process includes filling out basic information, selecting a profile image, specifying gender, date of birth, and whether the user is a doctor of therapeutic nutrition. The form is split across two pages with the help of a Page Controller.





Key Components

1. **Stateful Widget**:
    - **SignupAPIPage**: The main widget.
    - **_SignupAPIPageState**: The state class that manages the UI and logic for user interaction.

2. **Controllers**:
    - _usernameController: Captures the username input.
    - _emailController: Captures the email input.
    - _passwordController: Captures the password input.
    - _phoneController: Captures the phone number input.

3. **State Variables**:
    - _selectedImage: Stores the profile image file selected by the user.
    - passwordVisible: Toggles the visibility of the password field.
    - _gender: Stores the selected gender.
    - selectDate: Stores the selected date of birth.
    - _selectedIfDoctor: Stores the selection indicating if the user is a doctor.

4. **Methods**:
    - _pickImage: Allows the user to pick an image from their gallery.
    - _goToNextPage and _goToPreviousPage: Navigate between the two pages of the form.
    - _handleSignUp: Handles the sign-up process, sending the collected data to the server.

Key Methods Explained

1. **_pickImage**:
    - Uses the ImagePicker package to select an image from the gallery.
    - Updates the state with the selected image file.
2. **_goToNextPage** and **_goToPreviousPage**:
    - Use the PageController to navigate between the pages smoothly.
3. **_handleSignUp**:
    - Collects user inputs and constructs a multipart HTTP request to the server.
    - Handles both image upload and form data submission.
    - Provides user feedback based on the response from the server.

UI Components

1. **First Page**:
    - Contains text fields for username, email, phone, and password.
    - "Next" button navigates to the second page.
    - Link to sign in if the user already has an account.
2. **Second Page**:
    - Allows the user to select a profile image.
    - Contains dropdown for gender selection.
    - Date picker for date of birth.
    - Radio buttons to specify if the user is a doctor.
    - "Back" button to return to the first page.
    - "Sign Up" button to submit the form.

Post Management

- **Add Post**:
    - Users can add new posts using a text field for content and an option to add an image from the gallery.
    - Posts are displayed in a card-like layout with the user's avatar, name, post content, and image.
    - Interaction options include liking a post, adding comments, and a menu for editing or deleting the post.

- **Display Posts**:
    - Posts are rendered in a list with each post showing the number of likes and comments.
    - Users can click on the like button to toggle likes and view or add comments to posts.

Search Functionality

- **Search Bar**: A search bar in the app bar toggles between a search field and the title "TryFoodia".
    - When active, users can input a query to search for other users by their username.
    - The search results update in real-time based on the input.

Navigation

- **Bottom Navigation Bar**:
    - Includes three items: Profile, Contacts, and Records, allowing users to navigate between different sections of the app.
    - Each navigation item corresponds to a page in the PageView, ensuring a consistent and smooth navigation experience.

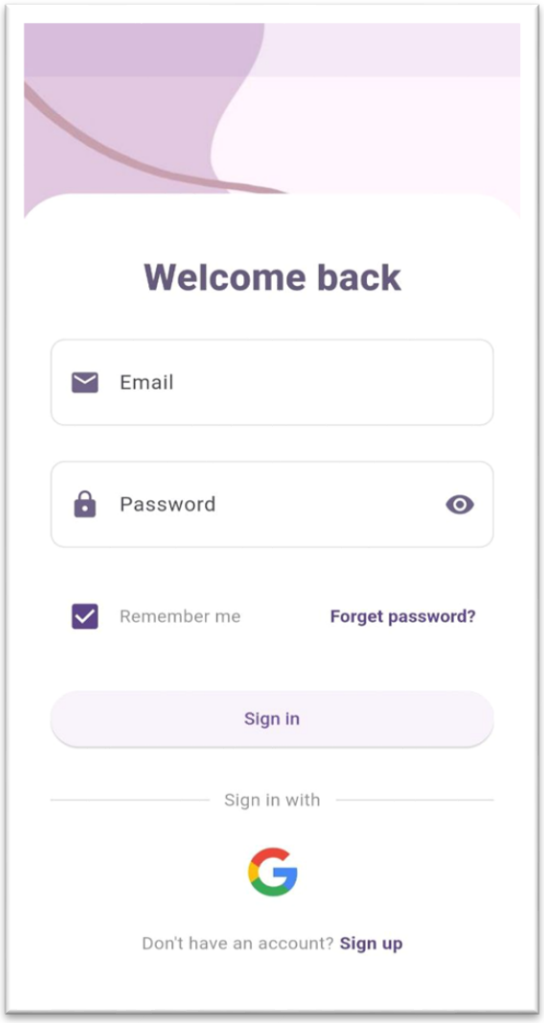
Shimmer Effect

- **Loading Indicator**:
    - A shimmer effect is used to indicate loading states, providing a placeholder while the actual content is being fetched.
    - This is particularly useful when loading user profile data, enhancing the overall user experience by giving a visual indication that data is being loaded.

5.3.3 Sign In Page

Overview

The Sign In Screen is a Flutter page that provides a user interface for signing in with email/password or through Google Sign-In. This page also includes options for remembering the user, password visibility toggle, and a link to reset the password. Additionally, it offers navigation to a signup page for users who do not have an account.



Components and Functionality

1. **Stateful Widget**
    - **SignInScreen**: A stateful widget that maintains the state of the sign-in form.
    - **_SignInScreenState**: The state class that manages the UI and logic for user interaction.

2. **Form Key and Controllers**
    - _formSignInKey: A GlobalKey for validating the form.
    - _emailController: A TextEditingController for capturing the user's email input.
    - _passwordController: A TextEditingController for capturing the user's password input.

3. **State Variables**
    - rememberPassword: A boolean to track whether the user wants to remember their password.
    - passwordVisible: A boolean to toggle password visibility.
    - _googleSignIn: An instance of GoogleSignIn for Google authentication.

4. **Methods**
    - **_handleLogin**: Handles the login process using basic authentication with the provided email and password.
    - **_handleGoogleSignIn**: Handles Google Sign-In and updates the email input field with the Google account email if successful.
    - **_handleForgotPassword**: Navigates to the forgot password screen.

5. **UI Structure**
    - **CustomScaffold**: A custom scaffold that includes a background image and safe area for child widgets.
    - **Form**: A form widget containing text fields and buttons for user input and actions.
    - **TextFormField**: Widgets for email and password input with validation.
    - **Buttons**: Includes a sign-in button, Google Sign-In button, and navigation buttons for forgot password and signup.

6. **Design Elements**
    - **AppBar**: Transparent app bar to match the background image.
    - **Background Image**: Set as the background of the scaffold to enhance visual appeal.
    - **Padding and Spacing**: Used to create a clean and organized layout.
    - **Input Decoration**: Customized input fields with icons and rounded borders.
    - **Visibility Toggle**: Allows users to toggle password visibility.

Custom Methods

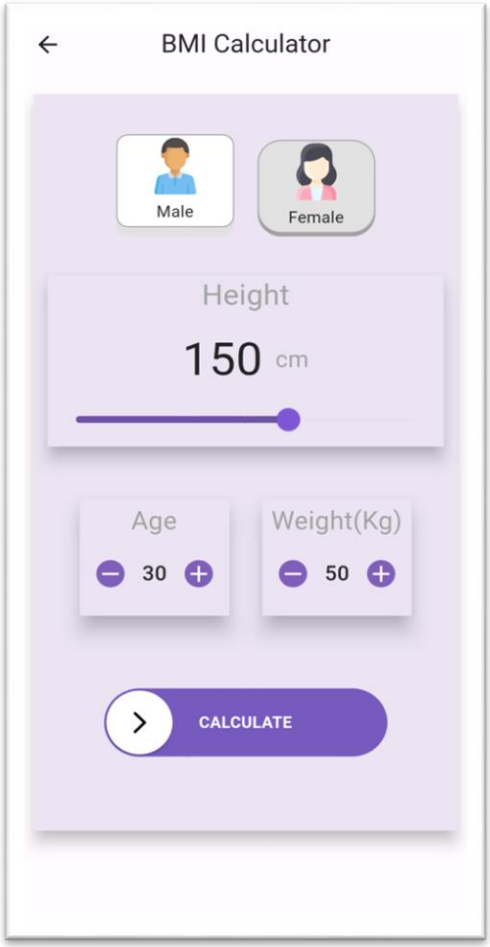
1. **_handleLogin**
    - **Parameters**:
        - context: The build context to navigate or show snackbar messages.
    - **Functionality**:
        - Retrieves email and password from the text controllers.
        - Performs a POST request to the login API endpoint with basic authentication.
        - On successful response, navigates to the DoctorPage.
        - Displays a snackbar message if login fails.

2. **_handleGoogleSignIn**
    - **Parameters**: None.
    - **Functionality**:
        - Initiates Google Sign-In.
        - On success, updates the email input field with the Google account email.
        - Logs an error message if Google Sign-In fails.
3. **_handleForgotPassword**
    - **Parameters**: None.
    - **Functionality**:
        - Navigates to the ForgotPasswordScreen.

5.3.4 BMI Home Screen

1-Overview

The Body Mass Index is a Flutter widget designed to provide users with a tool to calculate their Body Mass Index (BMI). The screen collects user input for gender, height, age, and weight, and upon calculation, navigates to a results screen displaying the BMI score. This page utilizes a combination of custom widgets and Flutter packages to create an interactive and visually appealing user experience.



2- Components

StatefulWidget:

- The BMIHomeScreen extends StatefulWidget, allowing it to maintain and update the state based on user interactions.

State Variables:

- _gender: Stores the selected gender (1 for male, 2 for female).
- _height: Stores the height input by the user.
- _age: Stores the age input by the user.
- _weight: Stores the weight input by the user.
- _isFinished: Indicates if the BMI calculation process is finished.
- _bmiScore: Stores the calculated BMI score.

Widgets Used:

- GenderWidget: Allows users to select their gender.
- HeightWidget: Allows users to input their height using a slider.
- AgeWeightWidget: Allows users to input their age and weight using increment/decrement buttons.

App Bar:

- Provides a title for the screen, centered at the top.

Card:

- Used to create a visually distinct container for the main content of the screen.

SwipeableButtonView:

- Provides a swipe button that, when swiped, calculates the BMI and navigates to the results screen.
-

3- Key Features

Gender Selection:

- The GenderWidget allows users to select their gender using 3D choice chips, which provide a visually appealing way to make a selection.

Height Input:

- The HeightWidget lets users input their height using a slider, with real-time display of the current value.

Age and Weight Input:

- The AgeWeightWidget provides increment and decrement buttons for users to adjust their age and weight values.

BMI Calculation:

- The BMI is calculated using the formula:

$$\text{BMI} = \frac{\text{weight}}{(\text{height}/100)^2}$$

- The calculation is triggered by swiping the button in the SwipeableButtonView.

Navigation to Results Screen:

- Upon finishing the BMI calculation, the app navigates to the ScoreScreen to display the BMI score.

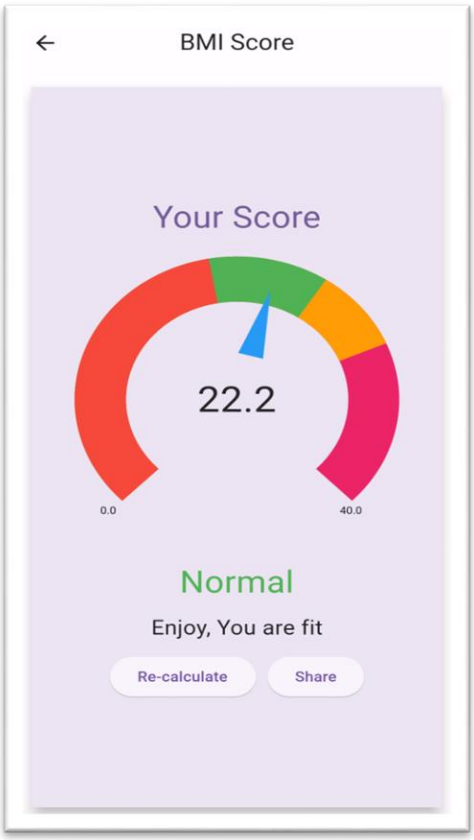
4 - User Interaction

- **Selecting Gender**:
  - Users can select their gender by tapping on the male or female 3D choice chips. The selected chip is highlighted to indicate the current selection.
- **Adjusting Height**:
  - Users can adjust their height by dragging the slider, with the current height value displayed in real-time.
- **Adjusting Age and Weight**:
  - Users can adjust their age and weight using the plus and minus buttons provided in the AgeWeightWidget.
- **Calculating BMI**:
  - Users swipe the "CALCULATE" button to trigger the BMI calculation. The app calculates the BMI and navigates to the results screen to display the score.

5.3.5 BMI Score Screen

1 - Overview

The Score Screen is a StatefulWidget in Flutter designed to display the user's BMI score, that displays the calculated Body Mass Index (BMI) score along with an interpretation of the result based on the user's age. It uses a gauge to visually represent the BMI score and provides options to recalculate the BMI or share the results. This screen is navigated to from the BMIHomeScreen after the BMI calculation is complete.



2- Components

1. **StatefulWidget**:
    - The ScoreScreen extends StatefulWidget, which allows the widget to maintain and update the state based on the BMI score.

2. **State Variables**:
    - bmiStatus: Stores the status of the BMI (e.g., Underweight, Normal, Overweight, Obese).
    - bmiInterpretation: Provides a textual interpretation of the BMI status.
    - bmiStatusColor: Stores the color associated with the BMI status.

3. **Constructor**:
    - The Score Screen constructor takes in two required parameters: BMI Score and age.

3- Widgets and Layout

1. **AppBar**:
    - The AppBar provides a title ("BMI Score") and is centered at the top of the screen.
2. **Container and Card**:
    - A Container with padding holds a Card that provides elevation and shape to the main content of the screen.
3. **Text Widgets**:
    - Displays the "Your Score" label, the BMI score, and the textual interpretation of the BMI status.
4. **Pretty Gauge**:
    - A custom gauge widget that visually represents the BMI score with different segments for various BMI ranges (Underweight, Normal, Overweight, and Obese).
5. **Buttons**:
    - Two ElevatedButton widgets allow the user to either recalculate their BMI or share their BMI score.

4-Key Features

BMI Gauge:

- The PrettyGauge widget visually represents the BMI score with colored segments indicating different BMI categories (Underweight, Normal, Overweight, and Obese). The needle points to the current BMI score.

BMI Interpretation:

- The setBmiInterpretation method sets the bmiStatus, bmiInterpretation, and bmiStatusColor based on the BMI score. This information is displayed to the user in a readable and visually appealing manner.

Recalculate and Share Buttons:

- The "Re-calculate" button allows users to go back to the previous screen to recalculate their BMI.
- The "Share" button uses the Share package to share the BMI score and age with others through available sharing options on the device.

5 - User Interaction

Viewing BMI Score:

- Users can see their BMI score on the gauge and receive a textual interpretation of their BMI status.

Recalculating BMI:

- Users can click the "Re-calculate" button to return to the previous screen and input new data to recalculate their BMI.

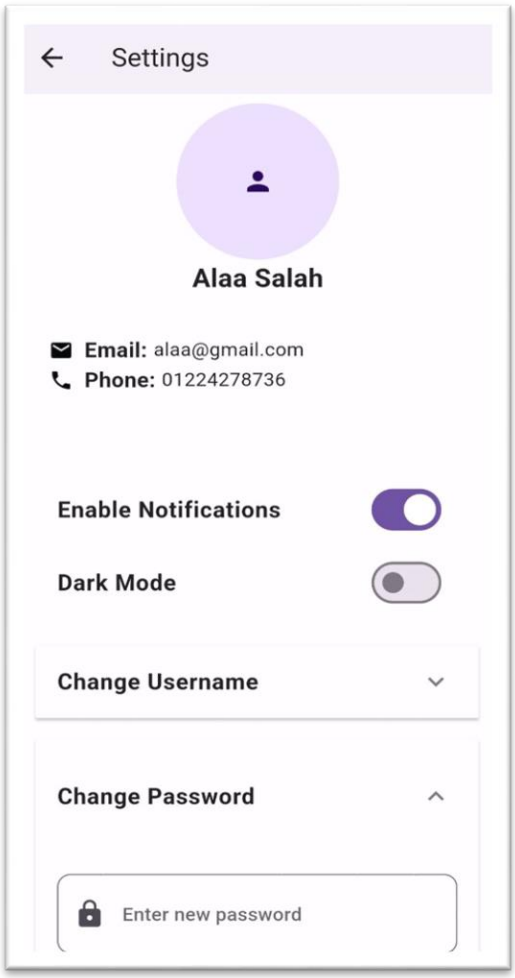
Sharing BMI Score:

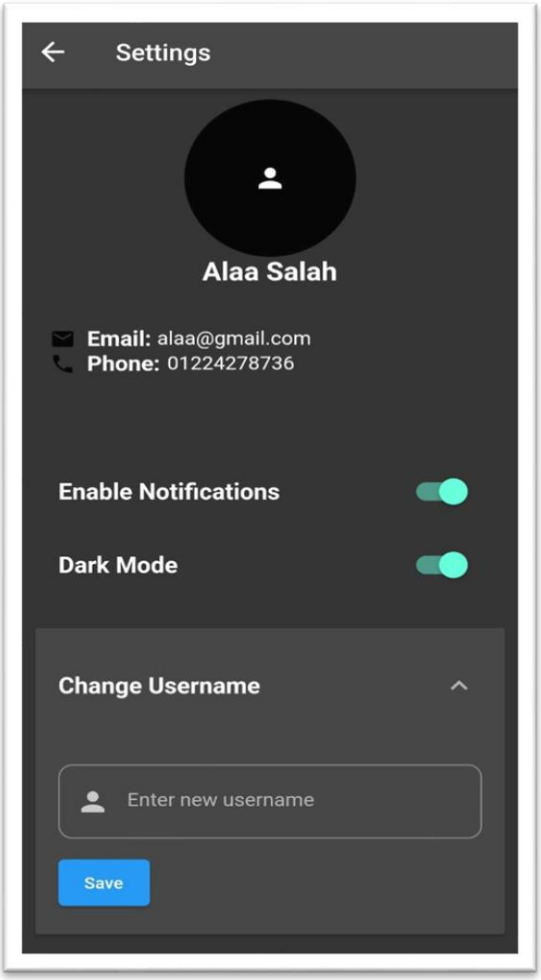
- Users can click the "Share" button to share their BMI score with others, using the device's sharing options.

5.3.6 Settings Screen

1-Overview

The Settings Page in our Flutter application is a comprehensive interface that allows users to manage their preferences and personal information. This page includes features such as enabling or disabling notifications, toggling dark mode, changing the username, and updating the password. The page utilizes the Provider package for state management, ensuring that user preferences are updated in real-time throughout the application.





2-Widgets and State Management

SettingsPage Widget

- **Stateful Widget**: Manages state using _SettingsPageState.
- **Text Editing Controllers**: Manages input fields for new password, confirm password, and new username using TextEditingController.
- **User Data**: Retrieves and displays user data fetched from the API.
- **Loading Indicator**: Displays a shimmer effect while user data is being fetched (isLoading).

initState and _fetchUserData

- **Initialization**: Initializes state and fetches user data using _fetchUserData during widget initialization (initState).
- **HTTP GET Request**: Fetches user data from the API endpoint using http.get with basic authentication (Authorization header).

buildProfilePage Widget

- **Profile Display**: Displays user profile information including username, email, and phone number.
- **Conditional Rendering**: Shows a shimmer effect while data is loading, or displays user information once loaded.

Consumer<SettingsModel> Widget

- **State Management**: Utilizes Provider for managing state (SettingsModel).
- **SwitchListTile Widgets**: Toggle switches for enabling notifications and dark mode, with corresponding onChanged handlers that update state in SettingsModel.

ExpansionPanelList.radio

- **Username and Password Expansion Panels**:
  - Allows users to expand/collapse panels to change username or password.
  - Text fields for entering new username and password.
  - Save button triggers changeUsername or changePassword methods in SettingsModel.

SnackBar Feedback

- **User Feedback**: Displays SnackBar messages to inform users of successful or failed operations (e.g., username or password update).

3-Models and Business Logic

SettingsModel (ChangeNotifier)

- **State Management**: Manages state for notifications, dark mode, username, and password using ChangeNotifier.
- **Methods**: Includes methods to toggle notifications and dark mode (toggleNotifications, toggleDarkMode), set unit (setUnit), and update username/password (changeUsername, changePassword).
- **HTTP PUT Requests**: Updates username and password on the server using http.put, with error handling and state updates upon success.

Networking and Authentication

- **HTTP Requests**: Communicates with the remote API (http package) to fetch user data and update settings.
- **Basic Authentication**: Uses Basic Authentication with username and password encoded using base64Encode.

UI Components and Styling

- **Widgets**: Utilizes TextField, ElevatedButton, SwitchListTile, ExpansionPanelList, and SnackBar for user interaction and feedback.
- **Shimmer Effect**: Displays a loading shimmer effect (Shimmer.fromColors) while user data is being fetched.

Error Handling

- **Try-Catch Blocks**: Implements error handling for network requests and updates, displaying relevant error messages via SnackBar.

Dependencies

- **flutter**, **flutter/material.dart**: Core Flutter framework and material design widgets.
- **provider**: For state management and dependency injection.
- **http**: For making HTTP requests.
- **shimmer**: For creating shimmer effects while loading data.

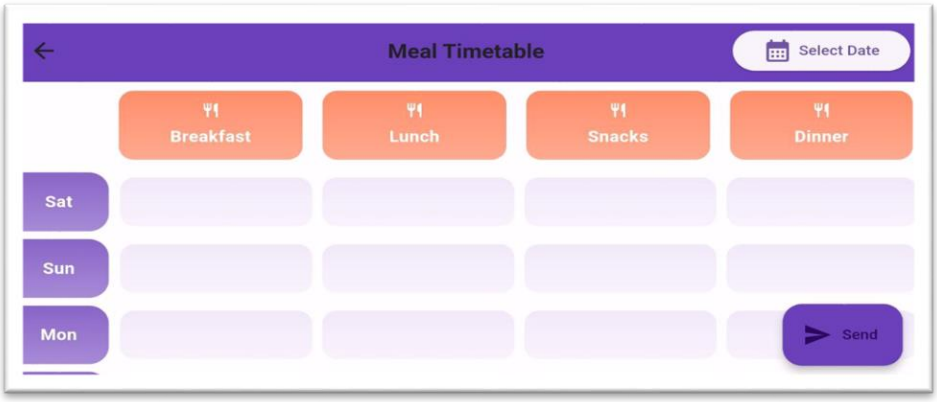
4- User Interaction

- **Notifications and Dark Mode**:
  - Users can toggle the switches for notifications and dark mode.
  - Changes are instantly reflected in the UI through the Provider.
- **Changing Username**:
  - Users input a new username and press the save icon.
  - The app sends a request to update the username on the server.
  - If successful, the username is updated locally and a success message is displayed. If not, an error message is shown.
- **Changing Password**:
  - Users input a new password and confirm it.
  - The app sends a request to update the password on the server.
  - If successful, the password is updated locally and a success message is displayed. If not, an error message is shown.

5.3.7 Timetable Screen

1- Overview

The Timetable Page is designed to manage and display a weekly meal timetable. Doctors can input meal data for each day of the week for patient and send this data to a remote server using HTTP POST requests.



2- Widgets and State Management

TimetablePage Widget

AppBar:

- Displays the title "Meal Timetable" and includes an action button to select a date using showDatePicker.

Body:

- **Meal Categories Row**: Displays categories such as Breakfast, Lunch, Snacks, and Dinner horizontally using an Expanded widget within a Row.
- **Days of the Week and Meal Data**:
  - For each day of the week (daysOfWeek), it displays a row where each cell represents a meal category (meals).
  - mealData is a 2D list (List<List<String>>) that holds the meal data input by the user. Each cell can be tapped to open a dialog (_showMealInputDialog) where users can input or modify meal information.

FloatingActionButton:

- Enables users to submit the timetable data (_send function), triggering an HTTP POST request to a specified endpoint.

showMealInputDialog Function

- **Dialog**: Allows users to input or modify meal information for a specific day and meal category. It displays an AlertDialog with a TextField for user input and "Save" and "Cancel" buttons to confirm or cancel the operation.

send Function

- **HTTP POST Request**: Constructs a list of Meal objects using the entered meal data (mealData) for each day of the week and sends it to a remote server.
- **Authentication**: Uses Basic Authentication (basicAuth) with a username and password encoded using base64Encode.

3- Models

Meal Class

- Represents a single meal entry with properties for date, day index, and meal categories (breakfast, lunch, snacks, dinner).
- **Serialization (toJson)**: Converts Meal objects to JSON format suitable for sending data over HTTP.
- **Deserialization (fromJson)**: Parses JSON data received from the server into Meal objects.

Networking

- **HTTP POST Request**: Sends the serialized timetable data (timetable) to the server endpoint (http://rana3del-001-site1.gtempurl.com/api/Meal/submitTimetable).
- **Error Handling**: Includes basic error handling for network errors and responses with status codes other than 200.

Orientation Handling

- **initState and dispose**: Ensures that the page is locked in landscape orientation (landscapeLeft and landscapeRight) during its lifetime and resets it to portrait orientation (portraitUp and portraitDown) upon disposal.

UI Styling

- **Gradient Decoration**: Utilizes linear gradients to style the background of meal category and day headers, enhancing visual appeal and readability.

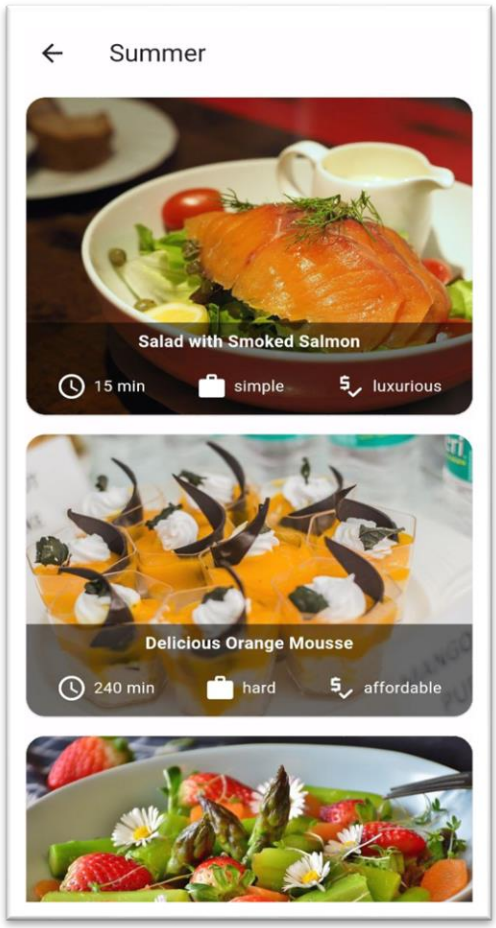
Dependencies

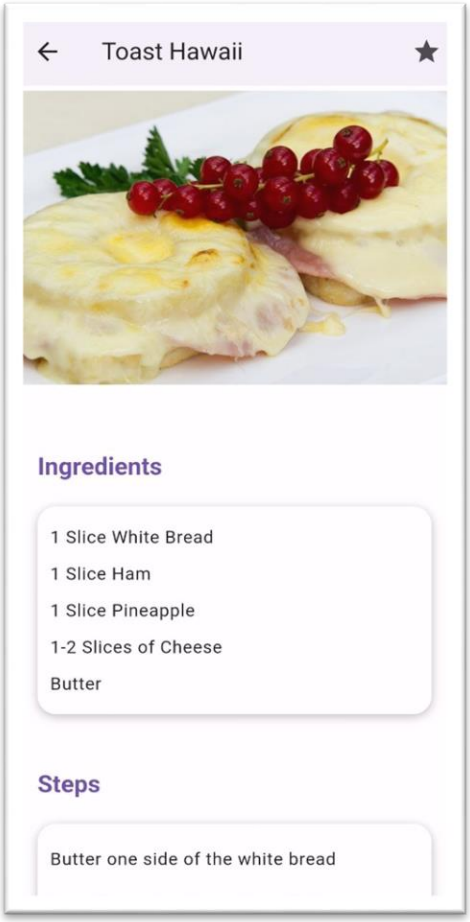
- **flutter**, **flutter/material.dart**: Core Flutter framework and material design widgets.
- **flutter/services.dart**: For managing device orientation.
- **http**: For making HTTP requests to the server.

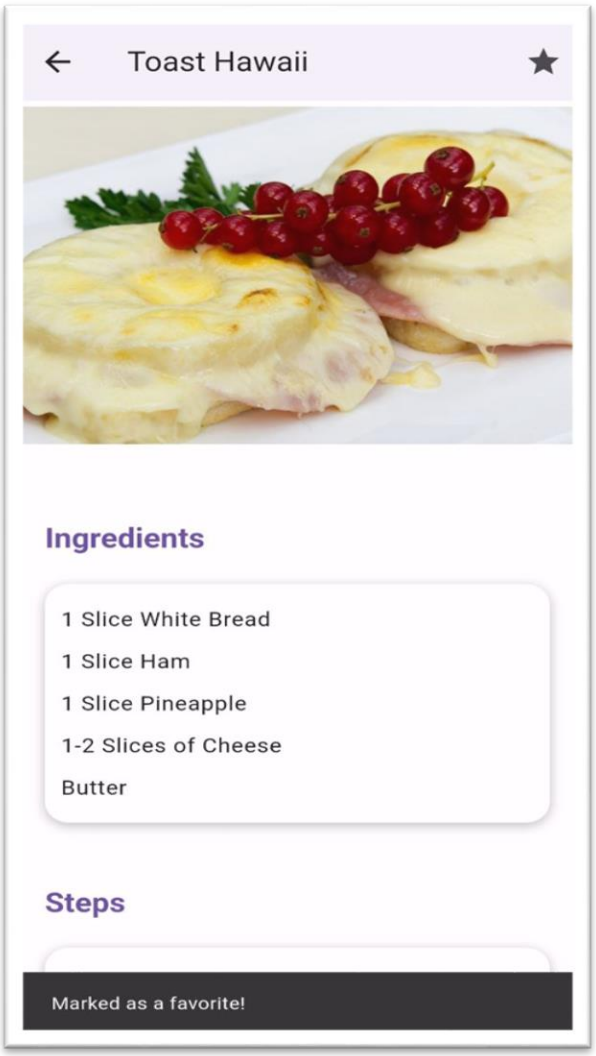
5.3.8 Meals screen

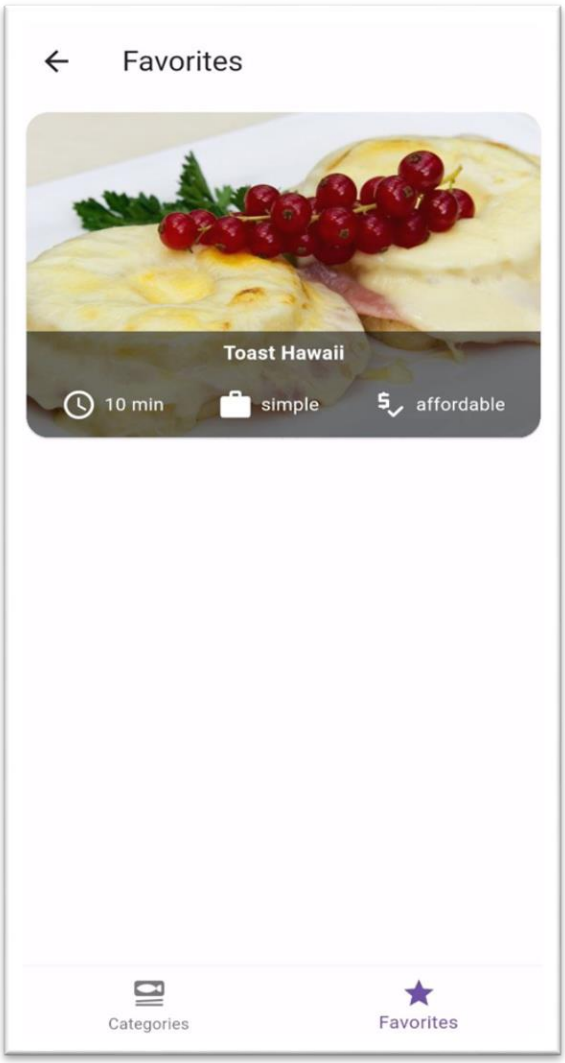
1- Overview

This is a meal planner that helps users discover and categorize meals based on different dietary filters. The app consists of several screens including categories, meal details, and filter settings. It supports marking meals as favorites and filtering meals based on specific dietary restrictions like gluten-free, lactose-free, vegan, and vegetarian.









2-Key Features

1. **Categories Screen**: Displays a grid of meal categories.
2. **Meals Screen**: Shows a list of meals within a selected category.
3. **Meal Detail Screen**: Provides detailed information about a specific meal.
4. **Filters Screen**: Allows users to set dietary filters such as gluten-free, lactose-free, vegan, and vegetarian.
5. **Favorites**: Users can mark meals as favorites and view them in a dedicated tab.

3- Screens and Navigation

Categories Screen

The Categories Screen is the primary interface where users can view a grid of meal categories. Each category is represented visually and can be selected to view meals belonging to that category. The screen uses a grid layout to organize categories, ensuring a user-friendly and visually appealing display.

Tabs Screen

The Tabs Screen manages the main navigation of the app, featuring two primary tabs: Categories and Favorites. A BottomNavigationBar at the bottom allows users to switch between these tabs. The screen is also responsible for handling the functionality to toggle favorite status and apply filters. When the Filters option is selected from the drawer, the Filters Screen is displayed, allowing users to customize their dietary preferences.

Meals Screen

The Meals Screen displays a list of meals for a selected category. Each meal is shown with basic details such as the title, image, and duration. When a meal is selected, the app navigates to the Meal Detail Screen, providing a deeper look into the meal's ingredients and preparation steps.

Meal Detail Screen

The Meal Detail Screen provides comprehensive information about a meal, including its ingredients and preparation steps. It also allows users to mark the meal as a favorite. This screen enhances user engagement by providing detailed content and interactive functionality.

Filter Screen

The Filter Screen enables users to set dietary filters to tailor the meal recommendations to their needs. The screen includes switches for gluten-free, lactose-free, vegan, and vegetarian options. The selected filters are applied to filter the available meals, ensuring that users only see meals that meet their dietary restrictions.

4- Data Models

The app uses data models to represent meals and categories. Each meal includes properties such as title, image URL, ingredients, steps, duration, complexity, affordability, and dietary suitability. Categories are represented by a title and color, helping to visually distinguish different meal types.

5- Initial Data

The app includes initial hardcoded data for meals and categories for demonstration purposes. This data helps in providing a working prototype of the app, showcasing its functionality and user interface.

6- Widgets and State Management

In the Meal Planner, a variety of widgets are used to create a dynamic and responsive user interface. State management is employed to handle user interactions and updates to the UI. This section outlines the key widgets and state management techniques used in the app.

7- Key Widgets

Scaffold

- **Purpose**: Provides a structure for the basic material design visual layout of the app.
- **Usage**: Utilized in all primary screens, such as the CategoriesScreen, MealsScreen, MealDetailScreen, and TabsScreen.
- **Components**: Includes an AppBar, Drawer, BottomNavigationBar, and FloatingActionButton where necessary.

GridView

- **Purpose**: Displays a grid of items.
- **Usage**: Used in the CategoriesScreen to display meal categories.
- **Components**: Each item in the grid is a custom widget representing a category.

ListView

- **Purpose**: Displays a scrollable list of items.
- **Usage**: Used in the MealsScreen to list meals and in the MealDetailScreen to list ingredients and preparation steps.
- **Components**: Each item in the list is a custom widget representing a meal or a detail element.

Drawer

- **Purpose**: Provides a hidden menu that slides in from the side.
- **Usage**: Used in the TabsScreen to provide access to the FiltersScreen.
- **Components**: Contains navigation links to different sections of the app.

SwitchListTile

- **Purpose**: Displays a switch with a label.
- **Usage**: Used in the FiltersScreen for toggling dietary preferences like gluten-free, lactose-free, vegan, and vegetarian.
- **Components**: Each switch is associated with a specific filter option.

BottomNavigationBar

- **Purpose**: Provides navigation between major sections of the app.
- **Usage**: Used in the TabsScreen to switch between the categories and favorites sections.
- **Components**: Contains icons and labels for each tab.
-

Card

- **Purpose**: Provides a rectangular container with rounded corners and a shadow.
- **Usage**: Used in both the CategoriesScreen and MealsScreen to present categories and meals, respectively.
- **Components**: Each card contains an image, title, and other relevant information.

8 - State Management

StatefulWidget

- **Purpose**: Manages state that can change over time or in response to user actions.
- **Usage**: Utilized in screens and widgets where dynamic updates are necessary, such as MealDetailScreen for toggling favorites and FiltersScreen for applying filters.
- **Components**: Contains a State object to hold the current state and setState() method to update the UI.

Provider

- **Purpose**: Manages and provides state to different parts of the widget tree efficiently.
- **Usage**: Used to manage global state such as the list of favorite meals and applied filters.
- **Components**: Includes ChangeNotifier for managing state and Consumer for accessing state within widgets.

ChangeNotifier

- **Purpose**: Notifies listeners of changes to the state.
- **Usage**: Used in conjunction with Provider to manage and update the state related to filters and favorites.
- **Components**: Contains methods to update state and notify listeners.

Example Flow

Categories Screen:

- Users select a category.
- Navigation to MealsScreen with the selected category data.

Meals Screen:

- Users view a list of meals.
- Select a meal to navigate to MealDetailScreen.

Meal Detail Screen:

- Users view meal details.
- Toggle favorite status.
- setState() updates the UI to reflect changes.

Tabs Screen:

- Users switch between categories and favorites using BottomNavigationBar.
- Access FiltersScreen from the drawer.

Filters Screen:

- Users set dietary preferences.
- ChangeNotifier updates the filter state.
- Consumer rebuilds widgets based on new state.

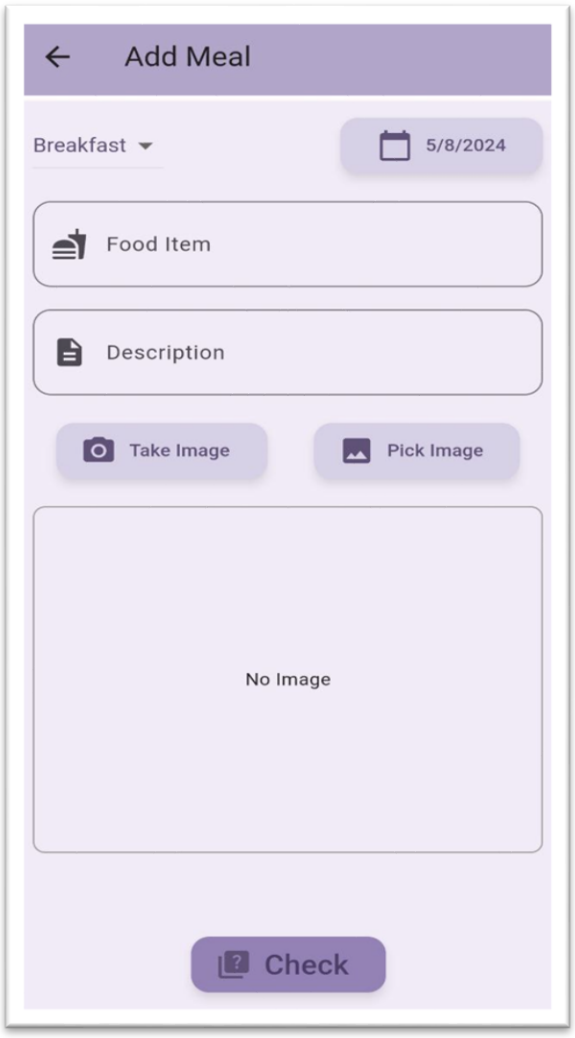
9- User Interaction

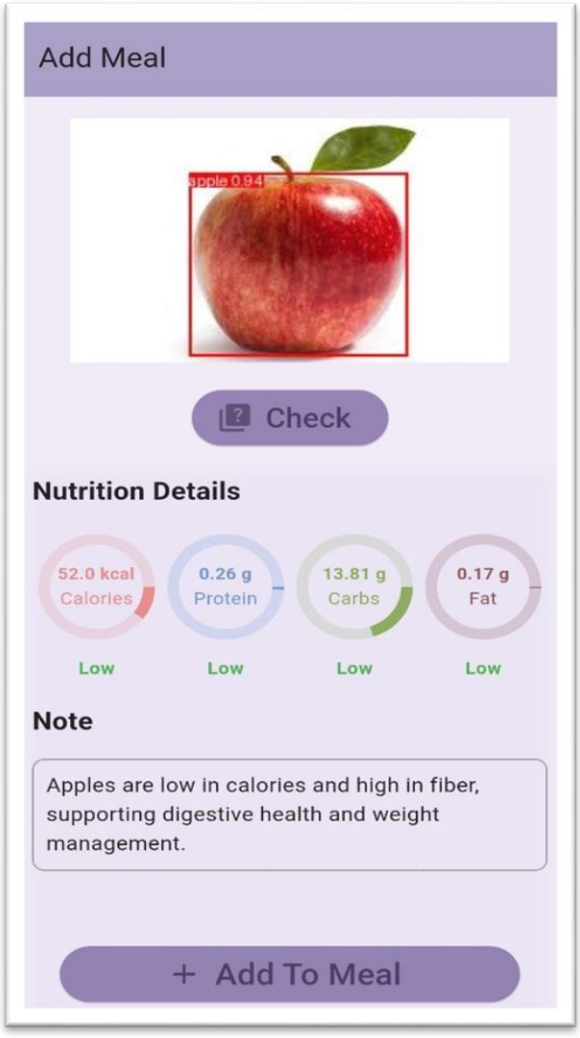
- **Category Selection**: Users can select a category from the Categories Screen to view the meals associated with it.
- **Meal Selection**: From the Meals Screen, users can select a specific meal to view detailed information on the Meal Detail Screen.
- **Favorite Meals**: Users can mark meals as favorites from the Meal Detail Screen, and view their favorite meals in the Favorites tab.
- **Setting Filters**: Users can set dietary filters on the Filter Screen to customize the meal recommendations according to their dietary preferences.

5.3.9 Scan Meals

Overview

The "Add Meal" page in the Flutter application serves the purpose of allowing users to input details about a meal they have consumed, including selecting the meal type, adding a food item description, capturing or selecting an image of the meal, and viewing nutritional details if an image of the meal is uploaded.





Components and Functionality

1. **AppBar**
    - Positioned at the top of the screen, providing a title "Add Meal" for easy navigation and context.

2. **Meal Type Drop1down**
    - Allows users to select the type of meal from options such as Breakfast, Lunch, Dinner, or Snacks using a DropdownButton.
3. **Date Selection**
    - Utilizes a DateTimePicker to let users choose the date when the meal was consumed, enhancing accuracy and record-keeping.
4. **Text Fields**
    - **Food Item**: Allows users to input the name of the food item they consumed using a TextField.
    - **Description**: Enables users to add additional notes or descriptions related to the meal using another TextField.
5. **Image Selection**
    - Provides functionality for users to either take a photo using the device's camera (_pickImage) or select an image from the gallery (_loadImage). Selected images are displayed within a container on the screen.
6. **Image Analysis**
    - Utilizes an API (_uploadImage) to analyze the uploaded image for detected classes of food items. If successful, displays the detected classes and an image preview of the detected item.
7. **Nutritional Details**
    - If an image is successfully analyzed, displays nutritional details (_user object) related to the detected food item using a set of gauges (_buildNutritionGauge) and labels (_buildNutritionLabel). Nutritional values include calories, protein, carbohydrates, and fat.
8. **Save and Add to Meal Button**
    - Allows users to save the entered meal details by uploading the data, including optional image attachment, to the backend server (saveMeal). Upon successful save, navigates users to the MealRecord page where they can view their recorded meals.

9. **Loading Indicator**
    - Displays a loading indicator (CircularProgressIndicator) while waiting for the image analysis or fetching nutritional details, ensuring user awareness of ongoing operations.

State Management

- **StatefulWidget and State**
    - The AddMeal class extends StatefulWidget, managing mutable state such as selected meal type, food item, description, selected image, selected date, detected classes, nutritional details, and loading state.
- **setState() Method**
    - Used to update the UI based on user interactions or asynchronous operations, ensuring a responsive and dynamic user experience.
- **Http and API Integration**
    - Utilizes http package for making HTTP requests to backend APIs (ApiService) to fetch nutritional details (getMealData) and save meal records (saveMeal).

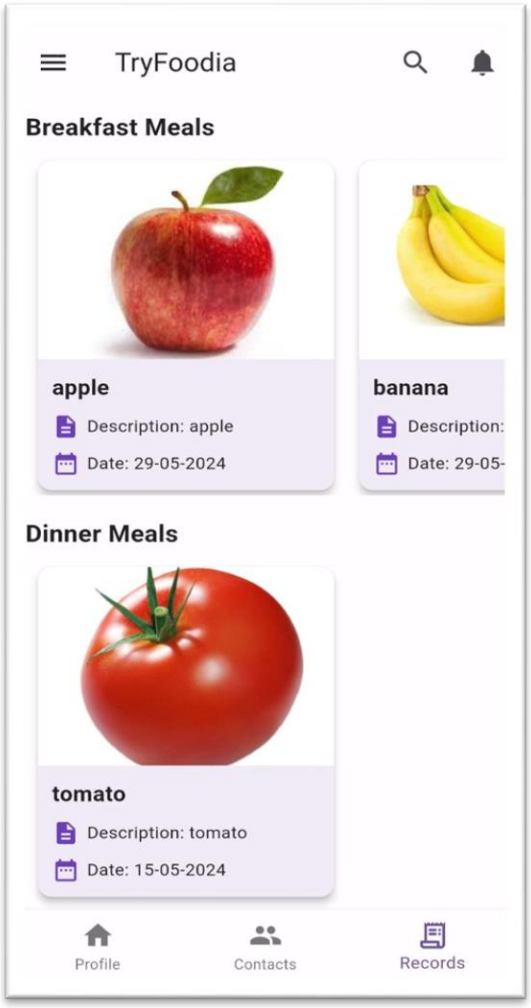
Design Considerations

- **User Interface (UI)**
    - Designed using Material Design principles, ensuring consistency and familiarity for users of Android and iOS platforms.
    - Utilizes appropriate widgets (TextField, DropdownButton, ElevatedButton, Image, Scaffold, etc.) to enhance user interaction and readability.
- **Error Handling**
    - Implements error handling for HTTP requests (try-catch blocks), displaying appropriate error messages (print statements) and ensuring graceful degradation in case of network failures or API errors.

5.3.10 Meal Record Page

Overview

The MealRecord page in the Flutter application displays a list of meal records for a given user. It organizes the meals by type (e.g., Breakfast, Lunch, Dinner, Snacks) and allows users to view details about each meal, including an image, description, and date. Users can tap on a meal to navigate to a detailed view of the meal.



Components and Functionality

1. **Constructor**
    - **Parameters**:
        - Email: The user's email address.
        - mealsList: A list of meal objects.
    - **Purpose**: Initializes the page with the user's email and the list of meals.

2. **State Management**
    - **StatefulWidget**: MealRecord extends StatefulWidget to manage the state of the meal records.
    - **State**: _MealRecordState maintains the state and handles the logic for fetching and displaying meal records.

3. **Future Initialization**
    - **Future<List<RecordWithImage>> futureRecords**: Initialized in initState to fetch records when the page is first built.
    - **fetchRecords**: Function that fetches the meal records from the backend using the user's email.

4. **Grouping Records**
    - **groupRecordsByType**: A method that groups the fetched records by meal type. It returns a map where the key is the meal type and the value is a list of RecordWithImage objects.

5. **Scaffold and AppBar**
    - **Scaffold**: Provides the basic structure for the page.
    - **AppBar**: (Commented out) Intended to display the title "Meals List."

6. **FutureBuilder**
    - **Purpose**: Manages the asynchronous fetching of meal records.
    - **States**:
        - **waiting**: Displays a loading indicator.
        - **hasError**: Displays an error message.
        - **no data**: Indicates no records were found.
        - **hasData**: Displays the records grouped by meal type in a ListView.

7. **ListView and Meal Display**
    - **ListView**: Displays grouped records as a list of columns, each representing a meal type.
    - **Padding and Column**: Used to format the display of each meal type section.
    - **SingleChildScrollView**: Allows horizontal scrolling of meal records within each type section.
    - **GestureDetector**: Wraps each meal record to enable navigation to the MealDetailPage on tap.

8. **Meal Record Card**
    - **Container and Card**: Each meal is displayed in a card with a fixed width for consistency.
    - **Image**: If an image is available, it is displayed at the top of the card.
    - **Details**: The meal item name, description, and date are displayed below the image.

Error Handling

- **fetchRecords Function**
    - **HTTP Status 200**: Parses the response body into a list of RecordWithImage objects.
    - **HTTP Status 404**: Throws an exception indicating no records were found.
    - **Other Errors**: Throws a general exception for failed requests.

API Integration

- **HTTP Request**: Uses the http package to send a GET request to the backend API.
- **Authorization**: Basic Auth is used for authorization, encoded in the request headers.

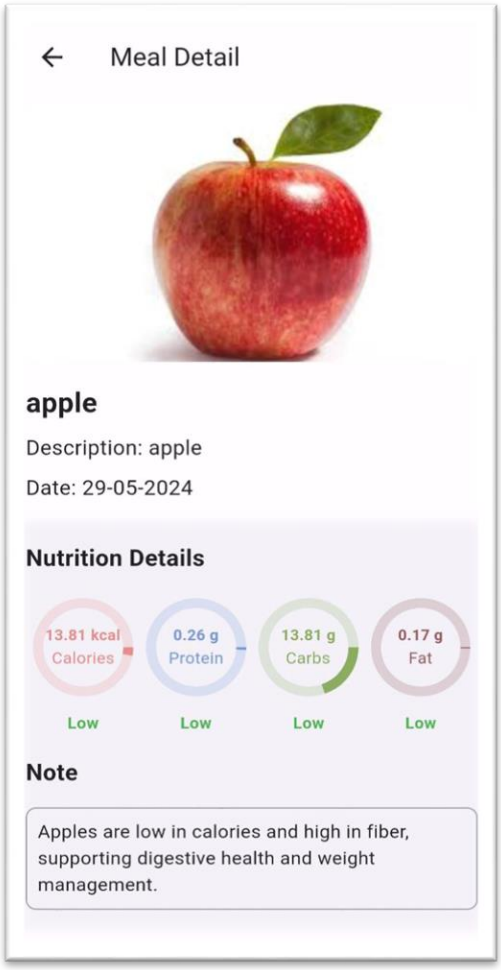
Design Considerations

- **UI Consistency**: Maintains a consistent look and feel with the rest of the application.
- **User Interaction**: Provides intuitive navigation and interaction by grouping meals and allowing detailed views.
- **Performance**: Efficiently manages asynchronous data fetching and rendering using FutureBuilder.

5.3.11 Meal Detail Page

Overview

The MealDetailPage is a Flutter page designed to display detailed information about a specific meal record. This page includes various attributes of the meal such as an image, item name, description, date, and detailed nutritional information. Additionally, it provides a note section for any extra information related to the meal.



Components and Functionality

1. **Constructor**
    - **Parameters**:
        - record: An instance of RecordWithImage containing the meal details to be displayed.
    - **Purpose**: Initializes the page with the provided meal record data.

2. **UI Structure**
    - **Scaffold**: Provides the basic structure of the page, including an AppBar and a Body that contains the meal details.
    - **AppBar**: Displays the title 'Meal Detail'.

3. **Meal Information Display**
    - **Padding**: Adds padding around the content for a clean layout.
    - **SingleChildScrollView**: Allows vertical scrolling of the content to ensure all information is accessible on smaller screens.
    - **Column**: Organizes the content vertically.
    - **Conditional Image Display**: If the meal record contains an image, it is displayed at the top. Otherwise, an empty container of fixed height is shown.

4. **Text Information**
    - **Item Name**: Displayed prominently with a larger font size and bold styling.
    - **Description**: Provides additional details about the meal.
    - **Date**: Shows the date when the meal was recorded, formatted as 'dd-MM-yyyy'.

5. **Nutrition Details**
    - **Container**: Encapsulates the nutrition details with a decorative shadow for visual separation.
    - **Column**: Organizes nutrition details and note section vertically.
    - **Row**: Displays four key nutritional values (Calories, Protein, Carbs, Fat) side by side using radial gauges.
    - **Radial Gauges**: Visual representation of the nutritional values using SfRadialGauge.
        - **_buildNutritionGauge Method**: Constructs each radial gauge with label, value, maximum value, and color.
        - **Gauge Annotations**: Annotates each gauge with the value and label.
    - **Nutrition Level Labels**: Displays the level (High, Moderate, Low) for each nutritional attribute with appropriate colors.

6. **Note Section**
    - **Text**: Displays any additional notes related to the meal.
    - **Container with Border**: Encapsulates the note text, adding a visual border for emphasis.

Custom Methods

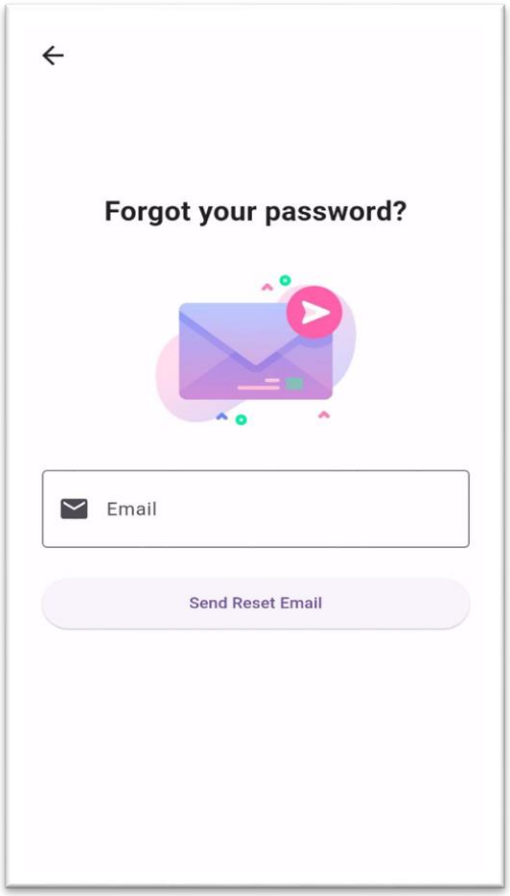
1. **_buildNutritionGauge**
    - **Parameters**:
        - label: The label for the nutritional attribute.
        - value: The actual value of the nutritional attribute.
        - max: The maximum possible value for the attribute.
        - color: The color to be used in the gauge.
    - **Returns**: A widget containing a radial gauge with the specified parameters.
    - **Purpose**: Creates and styles radial gauges to display nutritional values.

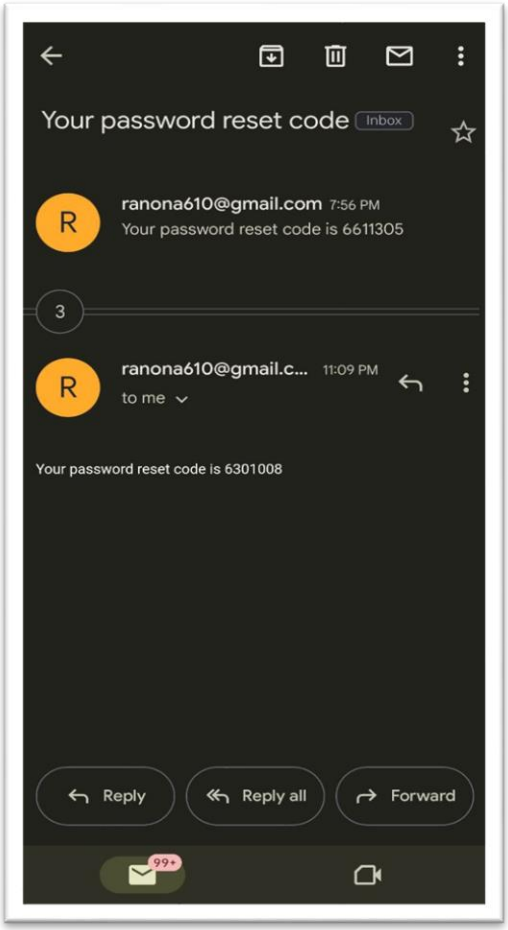
2. **_buildNutritionLabel**
    - **Parameters**:
        - Level: The level of the nutritional attribute ('high', 'low', 'moderate').
    - **Returns**: A text widget with appropriate styling based on the level.
    - **Purpose**: Provides a text label indicating the nutritional level with color coding.

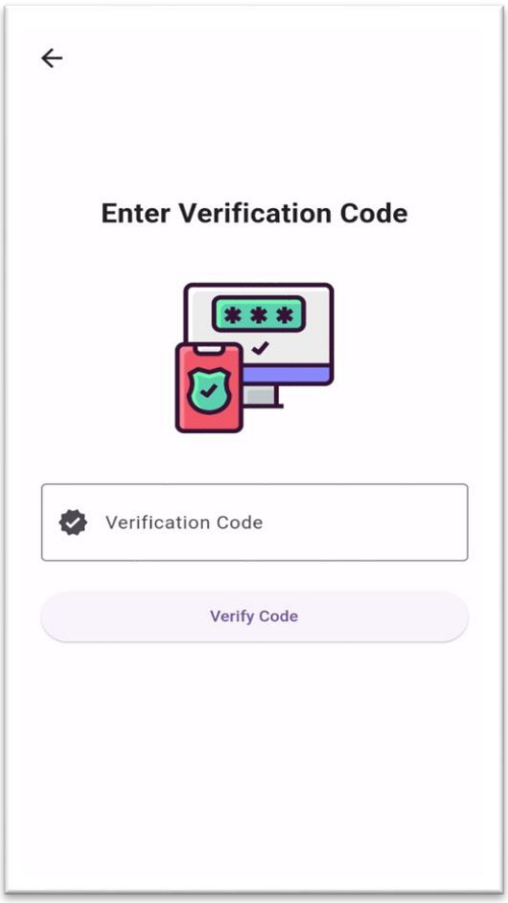
5.3.12 Forgot Password Screen

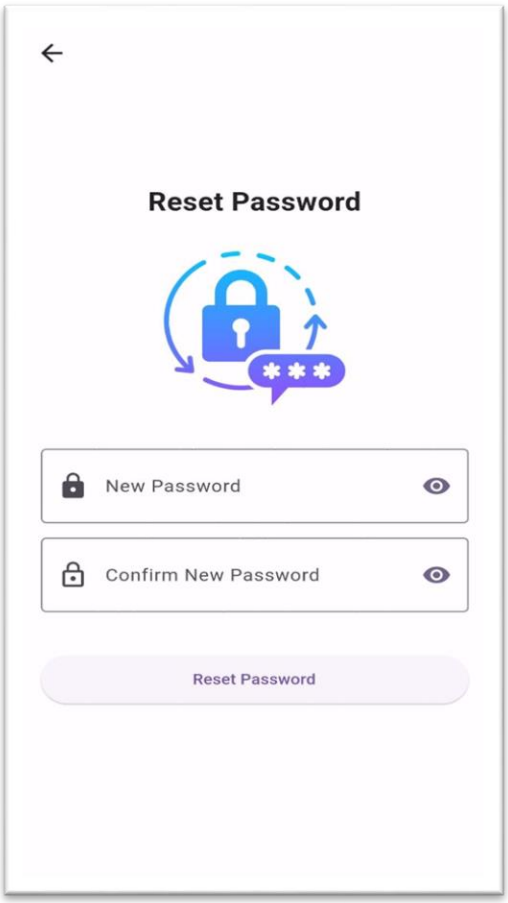
Overview

The Forgot Password Screen enables users to reset their passwords by following a series of steps: entering their email, verifying a code sent to their email, and setting a new password. This page handles the entire process of password recovery through API calls to the backend.









Components and Functionality

1. **Stateful Widget**
    - **ForgotPasswordScreen**: A stateful widget that represents the password recovery process.
    - **_ForgotPasswordScreenState**: The state class that manages the UI and logic for user interaction.

2. **Controllers**
    - _emailController: A TextEditingController for capturing the user's email input.
    - _codeController: A TextEditingController for capturing the verification code sent to the user's email.
    - _newPasswordController: A TextEditingController for capturing the new password input.
    - _confirmNewPasswordController: A TextEditingController for capturing the confirmation of the new password input.

3. **State Variables**
    - _errorMessage: A string to display error messages to the user.
    - _showCodeInput: A boolean to toggle the visibility of the code input field.
    - _showPasswordInput: A boolean to toggle the visibility of the new password input fields.

4. **Methods**
    - **_handleResetPassword**: Initiates the password reset process by sending a reset email to the provided email address.
    - **_handleVerifyCode**: Verifies the code sent to the user's email to proceed with resetting the password.
    - **_handleResetPasswordWithNewPassword**: Sets a new password for the user after code verification.
    - **_showSuccessMessage**: Displays a success message upon successful password reset.

5. **UI Structure**
    - **AppBar**: Displays the title "Forgot Password".
    - **Body**: Contains a form that dynamically changes based on the current step (email input, code input, or new password input).
    - **TextFields**: Input fields for email, verification code, new password, and confirmation of the new password.
    - **ElevatedButton**: A button that triggers the corresponding action based on the current step.
    - **Error Message**: Displays an error message if any step fails.

6. **Design Elements**
    - **Padding and Spacing**: Used to create a clean and organized layout.
    - **Input Decoration**: Customized input fields with labels and borders.
    - **Conditional Rendering**: UI elements are shown or hidden based on the state variables.

Custom Methods

1. **_handleResetPassword**
    - **Parameters**: None.
    - **Functionality**:
        - Retrieves the email from _emailController.
        - Sends a POST request to the backend to initiate the password reset process.
        - Shows the code input field if the request is successful.
        - Displays an error message if the request fails.

2. **_handleVerifyCode**
    - **Parameters**: None.
    - **Functionality**:
        - Retrieves the email and code from _emailController and _codeController.
        - Sends a POST request to the backend to verify the code.
        - Shows the new password input fields if the verification is successful.
        - Displays an error message if the verification fails.

3. **_handleResetPasswordWithNewPassword**
    - **Parameters**: None.
    - **Functionality**:
        - Retrieves the email, new password, and confirm password from the respective controllers.
        - Sends a PUT request to the backend to update the password.
        - Shows a success message if the password reset is successful.
        - Displays an error message if the password reset fails.

4. **_showSuccessMessage**
    - **Parameters**:
        - message: The success message to be displayed.
    - **Functionality**:
        - Displays an alert dialog with the success message and navigates back upon confirmation.

Chapter 6: Backend

6.1 What is a Database?

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.

Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

A Relational Database Management System (RDBMS) is a software that :

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

RDBMS Terminology

Before we proceed to explain the MySQL database system, let us revise a few definitions related to the database.

- **Database:** A database is a collection of tables, with related data.

• **Table**: A table is a matrix with data. A table in a database looks like a simple spreadsheet.

• **Column:** One column (data element) contains data of one and the same kind, for example the column postcode.

• **Row:** A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.

• **Redundancy:** Storing data twice, redundantly to make the system faster.

• **Primary Key:** A primary key is unique. A key value cannot occur twice in one table. With a key, you can only find one row.

• **Foreign Key:** A foreign key is the linking pin between two tables.

• **Compound Key:** A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.

• **Index:** An index in a database resembles an index at the back of a book.

• **Referential Integrity:** Referential Integrity makes sure that a foreign key value always points to an existing row.

6.2 MySQL Database

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.
Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.
Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

A Relational Database Management System (RDBMS) is a software that:

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses.
MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons.
MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
MySQL uses a standard form of the well-known SQL data language.
MySQL works on many operating systems and with many languages including .NET , Python, PHP, PERL, C, C++, JAVA, etc.
MySQL works very quickly and works well even with large data sets.
MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
MySQL is customization. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

Open source

Open source means that you're free to use and modify it. Anybody can install the software. You can also learn and customize the source code to better accommodate your needs. However, The GPL (GNU Public License) determines what you can do depending on conditions. The commercially licensed version is available if you need more flexible ownership and advanced support.

Client-server

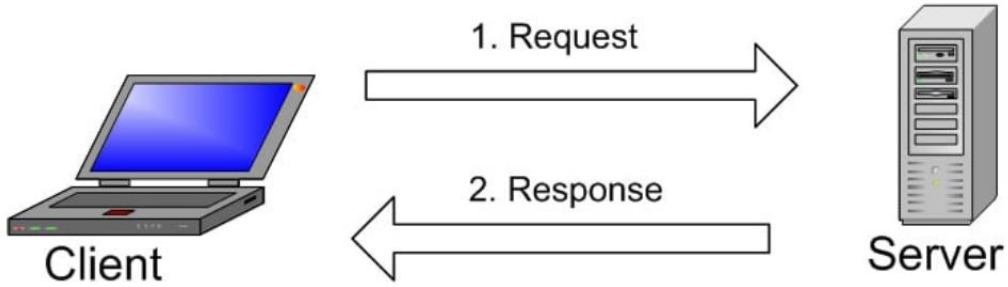
model Computers that install and run RDBMS software are called clients. Whenever they need to access data, they connect to the RDBMS server. That's the "client-server" part. MySQL is one of many RDBMS software options. RDBMS and MySQL are often thought to be the same because of MySQL's popularity. A few big web applications like Facebook, Twitter, YouTube, Google, and Yahoo! all use MySQL for data storage purposes. Even though it was initially created for limited usage, it is now compatible with many important computing platforms like Linux, macOS, Microsoft Windows, and Ubuntu.

SQL

MySQL and SQL are not the same. Be aware that MySQL is one of the most popular RDBMS software's brand names, which implements a client-server model. So, how do the client and server communicate in an RDBMS environment? They use a domain-specific language – Structured Query Language (SQL). If you ever encounter other names that have SQL in them, like PostgreSQL and Microsoft SQL server, they are most likely brands which also use Structured Query Language syntax. RDBMS software is often written in other programming languages, but always use SQL as their primary language to interact with the database. MySQL itself is written in C and C++.
Computer scientist Ted Codd developed SQL in the early 1970s with an IBM based relational model. It became more widely used in 1974 and quickly replaced similar, then outdated languages, ISAM and VISAM. History aside, SQL tells the server what to do with the data. It is similar to your WordPress password or code. You input it into the system to gain access to the dashboard area. In this case, SQL statements can instruct the server to perform certain operations:

- **Data query:** requesting specific information from the existing database.
- **Data manipulation**: adding, deleting, changing, sorting, and other operations to modify the data, the values or the visuals.
- **Data identity:** defining data types, e.g., changing numerical data to integers. This also includes defining a schema or the relationship of each table in the database.
- **Data access control:** providing security techniques to protect data, this includes deciding who can view or use any information stored in the database

How Does MySQL Work?



The image explains the basic structure of the client-server structure. One or more devices (clients) connect to a server through a specific network. Every client can make a request from the graphical user interface (GUI) on their screens, and the server will produce the desired output, as long as both ends understand the instruction. Without getting too technical, the main processes taking place in a MySQL environment are the same, which are:

- MySQL creates a database for storing and manipulating data, defining the relationship of each table.
- Clients can make requests by typing specific SQL statements on MySQL.
- The server application will respond with the requested information and it will appear on the clients' side.

Why is MySQL so Popular?

MySQL is indeed not the only (R)DBMS on the market, but it is one of the most popular ones based on the reviews of the developers on the Stack Overflow platform when scored using critical parameters like the number of mentions in search results, professional profiles on LinkedIn, and frequency of technical discussions on internet

forums. The fact that many major tech giants rely on it further solidifies the well deserved position. Why so? Here are the reasons:

1- Flexible and easy to use: You can modify the source code to meet your own expectations, and don't need to pay anything for this level of freedom, including the options for upgrading to the advanced commercial version. The installation process is relatively simple, and shouldn't take longer than 30 minutes.

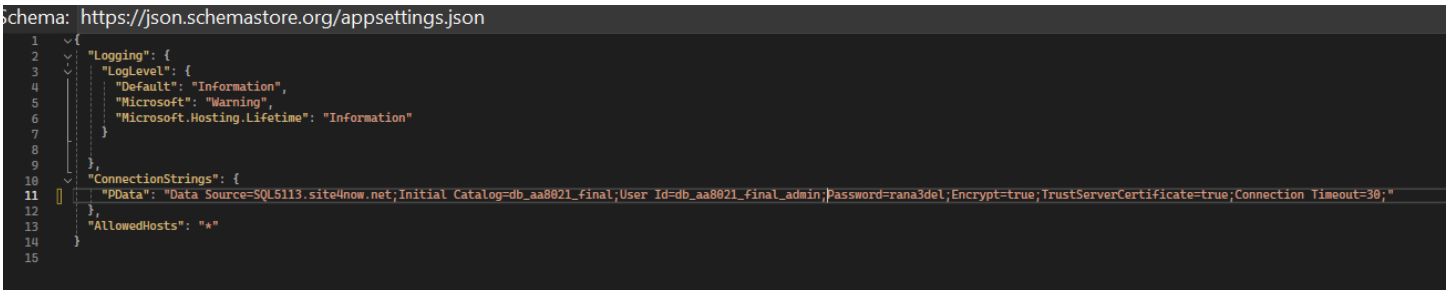
2- High performance: A wide array of cluster servers backs MySQL. Whether you are storing massive amounts of big e-Commerce data or doing heavy business intelligence activities, MySQL can assist you smoothly with optimum speed.

3- An industry standard: Industries have been using MySQL for years, which means that there are abundant resources for skilled developers. MySQL users can expect rapid development of the software and freelance experts willing to work for a smaller wage if they ever need them.

4- Secure: Your data should be your primary concern when choosing the right RDBMS software. With its Access Privilege System and User Account Management, MySQL sets the security bar high. Host-based verification and password encryption are both available.

Setup MySQL database in .NET

Before we start and showing you our schema that contain all tables, we used in project first we need to make a database configuration in .NET, in appsettings.json, and add this:



6.3 What is a Use case Diagram?

A use case diagram is a visual representation of the functional requirements of a system. It illustrates the interactions between the users (actors) and the system itself (use cases) to achieve a particular goal. Use case diagrams are part of Unified Modeling Language (UML) and are used to describe the functionality of a system from a user's perspective.

Key Components of a Use Case Diagram:

1. **Actors:**
    - Represent the roles that users or other systems play in interacting with the system.
    - Actors can be human users or other systems/software.

2. **Use Cases:**

- Represent the actions or services that the system performs in response to an actor's request.
- Use cases are typically named with verb-noun phrases like "Create Account," "Login," or "Generate Report."

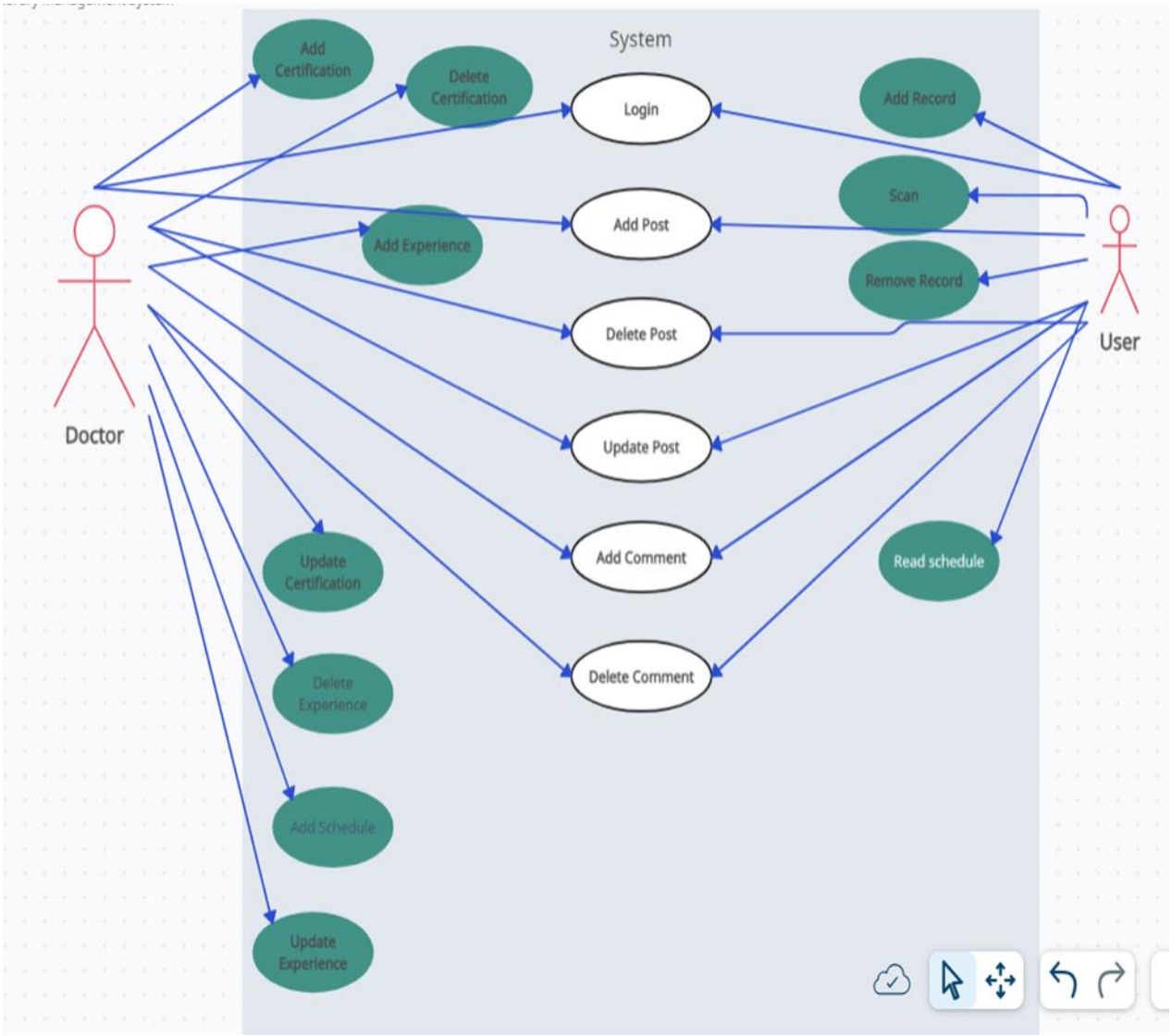
3. **System Boundary:**

- A rectangle that defines the scope of the system. Use cases are placed inside the system boundary, while actors are placed outside.

4. **Relationships:**

    - Association: A line connecting an actor to a use case, indicating that the actor participates in the use case.
    - Include: A dotted line with an arrow pointing to the included use case, indicating that the base use case always includes the behavior of the included use case.
    - Extend: A dotted line with an arrow pointing to the extended use case, indicating that the behavior of the use case can be extended by another use case under certain conditions.
    - Generalization: A line with a hollow triangle pointing to the parent actor or use case, indicating inheritance.

Use Case Diagram:



6.4 What is a Class Diagram?

A class diagram is a type of static structure diagram in Unified Modeling Language (UML) that describes the structure of a system by showing its classes, attributes, operations (methods), and the relationships among the objects. It is one of the most common types of UML diagrams and is used to illustrate the logical view of a system.

Key Components of a Class Diagram:

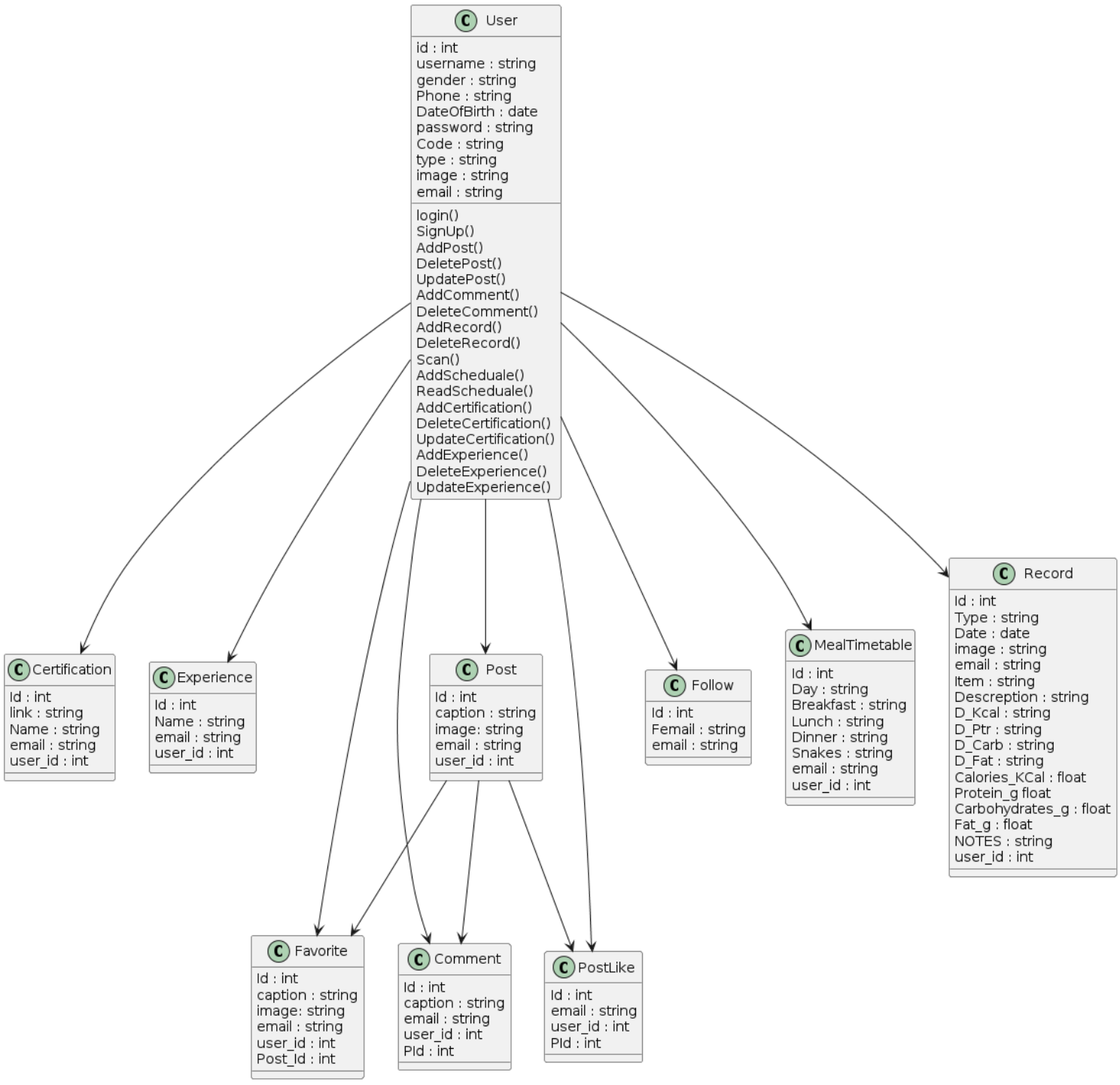
1. **Classes:**
   - Represent the main building blocks of the system.
   - Each class is depicted as a rectangle with three compartments:
   - The top compartment contains the class name.
   - The middle compartment lists the attributes.
   - The bottom compartment lists the operations (methods).

2. **Attributes:**
   - Characteristics or properties of the class.
   - Typically, an attribute is depicted with its visibility, name, and type

3. **Operations (Methods):**
   - Functions or procedures that the class can perform.
   - Typically, an operation is depicted with its visibility, name, and parameters

4. **Relationships:**
    - Associations: Represent the relationships between classes. Can include multiplicity
    - Inheritance (Generalization): A line with a hollow triangle pointing to the parent class, indicating a subclass inherits from the parent class.
    - Aggregation: A line with a diamond at the aggregate end, representing a whole-part relationship where the part can exist independently of the whole.
    - Composition: A line with a filled diamond at the composite end, representing a whole-part relationship where the part cannot exist independently of the whole.
    - Dependency: A dashed line with an arrow, indicating that a class depends on another class.

Class Diagram:



6.5 What is an ERD?

An Entity-Relationship Diagram (ERD) is a visual representation of the entities within a system and the relationships between those entities. It is commonly used in database design to illustrate the logical structure of databases. ERDs help in modeling data and are fundamental in the process of database design.

Key Components of an ERD:

1. **Entities:**
    - Represent objects or concepts that can have data stored about them.
    - Depicted as rectangles.

2. **Attributes:**
    - Describe properties or details of an entity.
    - Depicted as ovals connected to their entity.

3. **Relationships:**
    - Represent associations between entities.
    - Depicted as diamonds (in Chen notation) or just lines (in Crow's Foot notation).

4. **Primary Key (PK):**
    - An attribute that uniquely identifies each instance of an entity.
    - Underlined in the diagram.

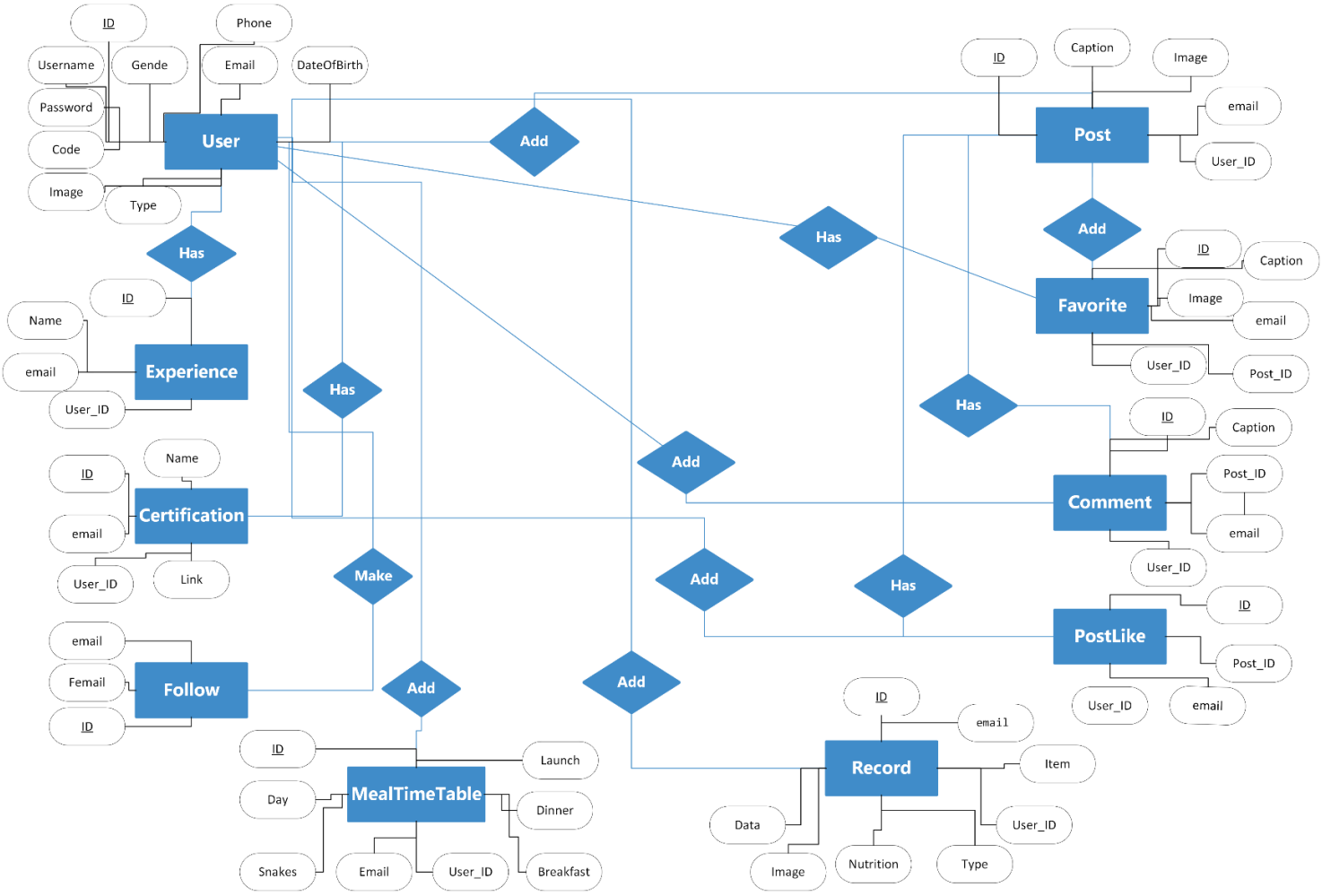
5. **Foreign Key (FK):**

- An attribute that creates a link between two entities.

6. **Cardinality:**

- Indicates the number of instances of one entity that can or must be associated with each instance of another entity.
- Commonly represented as one-to-one (1:1), one-to-many or many-to-many

Entity-Relationship Diagram (ERD):



6.6 What is a Schema?

A database schema is a blueprint or architecture of how our data will look. It doesn't hold data itself, but instead describes the shape of the data and how it might relate to other tables or models. An entry in our database will be an instance of the database schema.

It will contain all of the properties described in the schema. Think of a database schema as a type of data structure. It represents the framework and arrangement of the contents of an organization's data.

A database schema will include:

- All-important or relevant data
- Consistent formatting for all data entries
- Unique keys for all entries and database objects
- Each column in a table has a name and data typeThe size and complexity of your database schema depends on the size of your project. The visual style of a database schema allows programmers to structure the database and its relationships properly before jumping into the code. The process of planning a database design is called data modeling.

Schema Diagram:

6.7 What is ASP.NET?

ASP.NET is a high-level web development framework designed by Microsoft. Like many modern frameworks, ASP.NET is free and open source. It allows developers to build robust and scalable web applications, services, and APIs efficiently.

History and Evolution

Initially released in 2002, ASP.NET has evolved significantly over the years. The framework was developed to provide developers with a powerful and efficient way to build dynamic web applications. With the introduction of ASP.NET Core in 2016, the framework became cross-platform, allowing applications to run on Windows, macOS, and Linux.

Architecture

ASP.NET relies on the traditional model-view-controller (MVC) architecture, which promotes a clear separation of concerns:

- **Model:** Represents the data and the business logic of the application.
- **View:** Represents the UI, displaying data to the user.
- **Controller:** Handles user input and interactions, updating the model and view accordingly.

Key Features

1. **High Performance:**

ASP.NET Core, in particular, is known for its high performance and ability to handle a large number of requests efficiently. It leverages asynchronous programming and optimized runtime to achieve fast response times.

2. **Cross-Platform:**

   ASP.NET Core applications can run on Windows, macOS, and Linux, providing flexibility in deployment and development environments.

3. **Security:**

   Built-in security features, such as authentication and authorization, help protect web applications from common threats. ASP.NET also includes protections against CSRF, XSS, and SQL injection.

4. **Rich Ecosystem:**

   The .NET ecosystem includes a wide range of libraries, tools, and frameworks that integrate seamlessly with ASP.NET. This ecosystem supports various application types, including web, desktop, mobile, and cloud

5. **Development Tools:**

   Visual Studio, Microsoft's integrated development environment (IDE), provides comprehensive tools for developing, debugging, and deploying ASP.NET applications. Visual Studio Code, a lightweight and cross-platform editor, is also popular among ASP.NET developers.

Getting to Know the ASP.NET Framework

With a myriad of web frameworks to choose from, why pick the ASP.NET framework? It might not be the simplest to use and it's certainly not the newest. Nevertheless, the ASP.NET framework might be the right fit when you're building a web app that involves complex requirements and is expected to handle a large number of users or a complex set of features, such as API connectivity or user authentication. Based on the number of projects on GitHub and enterprise adoption, it's also very popular.

Anyone proficient in the C# programming language and its syntax should be able to start a project using the ASP.NET framework to build a web app. The framework supports development in multiple languages, including C#, F#, and VB.NET. Intermediate to advanced ASP.NET developers can better capitalize on ASP.NET's powerful features, which enable the building of sophisticated and high-performing applications.

Packaging Essential Features

ASP.NET has been in use for more than two decades and has been thoroughly tested and enhanced by a very active community and robust support from Microsoft. ASP.NET's greatest strength is its comprehensive feature set—with extensive libraries and tools, the framework covers virtually anything you'll need a web application to do. Features include:

API Development: ASP.NET makes it easy to build RESTful services that can be consumed by various clients, including browsers and mobile devices.
Content Management Systems: There are numerous CMS solutions built on ASP.NET, providing flexibility and customization for content-heavy applications.
User Authentication: Built-in authentication and authorization mechanisms, including support for various authentication protocols (OAuth, OpenID Connect).
Form Validation and Security: Comprehensive tools for form validation, data protection, and security features to guard against common threats like CSRF, XSS, and SQL injection.

.Net Library

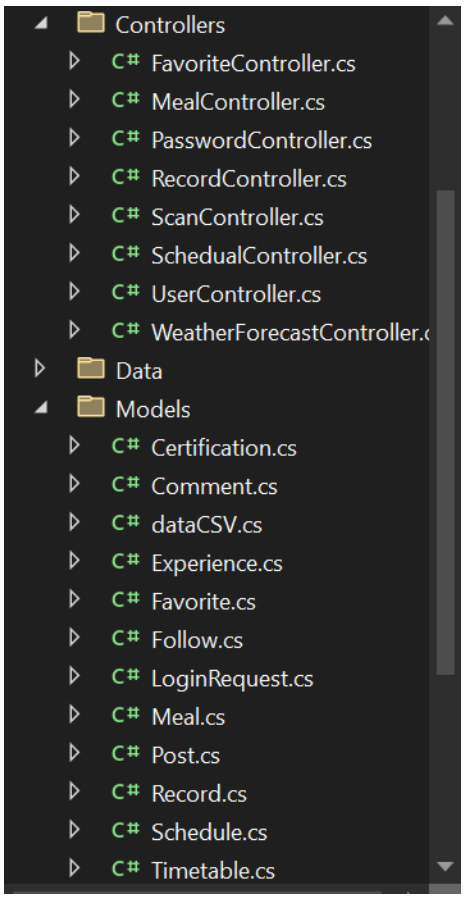
Extensive library simplifies complex programming tasks, ensuring developers can focus on writing high-quality code rather than reinventing the wheel.

```
using Final5.Models;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Configuration;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Threading.Tasks;
```

.NET Controller and Models

The controller in the .NET framework handles incoming HTTP requests, processes user input, and determines the appropriate response, often by interacting with the model.

The model represents the application's data and business logic, encapsulating the core functionalities and ensuring data integrity, which the controller uses to render views or return data to the user.



.Net APIs Examples:

1-Login

Input: Email, Password

Output: Login successful or Invalid email or password

```
[HttpPost]
[Route("login")]
0 references
public async Task<IActionResult> Login(LoginRequest loginRequest)
{
    try
    {
        string connectionString = _configuration.GetConnectionString("PData");

        using (SqlConnection con = new SqlConnection(connectionString))
        {
            string query = "SELECT COUNT(*) FROM [Usser] WHERE email = @Email AND password = @Password";
            using (SqlCommand cmd = new SqlCommand(query, con))
            {
                cmd.Parameters.AddWithValue("@Email", loginRequest.email);
                // Hash the password before comparing it to the stored hash in the database
                string hashedPassword = HashPassword(loginRequest.password);
                cmd.Parameters.AddWithValue("@Password", hashedPassword);

                await con.OpenAsync();
                int userCount = (int)await cmd.ExecuteScalarAsync();

                if (userCount > 0)
                {
                    return Ok(new
                    {
                        message = "Login successful",
                    });
                }
                return BadRequest("Invalid email or password");
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

2- **Sign Up**

Input: User Data

Output: Return Data or BadRequest or Failed to insert data

```
[HttpPost("usser")]
0 references
public IActionResult Usser([FromForm] Usser user)
{
    try
    {
        using (SqlConnection con = new SqlConnection(_configuration.GetConnectionString("PData")))
        {
            string query = "SELECT COUNT(*) FROM Usser WHERE email = @Email";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", user.email);
            con.Open();
            int existingUserCount = (int)cmd.ExecuteScalar();

            if (user.phone.Length != 11)
            {
                return BadRequest("Invalid phone number");
            }
            else if (existingUserCount > 0)
            {
                return BadRequest("Email already exists.");
            }
            else if (!user.email.EndsWith("@gmail.com"))
            {
                return BadRequest("Invalid email format.");
            }
            string fileName = null;
            string imagePath = null;
            if (user.image != null && user.image.Length > 0)
            {
                fileName = Guid.NewGuid().ToString() + Path.GetExtension(user.image.FileName);

                // Save image to server
                var directory = Path.Combine(Directory.GetCurrentDirectory(), "wwwroot", "images");
                if (!Directory.Exists(directory))
                {
                    Directory.CreateDirectory(directory);
                }
                imagePath = Path.Combine(directory, fileName);
```

```
                imagePath = Path.Combine(directory, fileName);

                using (var stream = new FileStream(imagePath, FileMode.Create))
                {
                    user.image.CopyTo(stream);
                }
            }

            query = "INSERT INTO Usser ( image,username, gender, dateOfBirth, phone, email, password,type) " +
                    "VALUES(@Image, @Username, @Gender, @DateOfBirth, @Phone, @Email, @Password,@Type)";
            cmd.CommandText = query;
            cmd.Parameters.Clear();
            cmd.Parameters.AddWithValue("@Image", fileName ?? (object)DBNull.Value);
            cmd.Parameters.AddWithValue("@Username", user.username);
            cmd.Parameters.AddWithValue("@Gender", user.gender);
            cmd.Parameters.AddWithValue("@DateOfBirth", user.dateOfBirth);
            cmd.Parameters.AddWithValue("@Phone", user.phone);
            cmd.Parameters.AddWithValue("@Email", user.email);
            cmd.Parameters.AddWithValue("@Password", user.password);
            cmd.Parameters.AddWithValue("@Type", user.type);

            int rowsAffected = cmd.ExecuteNonQuery();

            if (rowsAffected > 0)...
            else
            {
                if (fileName != null)
                {
                    System.IO.File.Delete(imagePath);
                }
                return StatusCode(StatusCodes.Status500InternalServerError, "Failed to insert data");
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

3-Forget Password

First -> Enter Email:

Input: Email
Output: Send Verification Code

```
[HttpPost("ForgetPass/{email}")]
0 references
public IActionResult ForgetPass(string email)
{
    try
    {
        using (SqlConnection con = new SqlConnection(_configuration.GetConnectionString("PData")))
        {
            string query = "UPDATE Usser SET Code = @Code WHERE email = @Email";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", email);
            con.Open();

            string newCode = CreateRandomToken();
            cmd.Parameters.AddWithValue("@Code", newCode);

            int rowsAffected = cmd.ExecuteNonQuery();

            if (rowsAffected > 0)
            {
                // Send email
                bool emailSent = SendEmail(email, newCode);
                if (emailSent)
                {
                    return Ok("User code updated and email sent successfully");
                }
                else
                {
                    return StatusCode(StatusCodes.Status500InternalServerError, "User code updated but failed to send email");
                }
            }
            else
            {
                return NotFound("Username not found for the given email");
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

```
private string CreateRandomToken()
{
    return RandomNumberGenerator.GetInt32(1000000, 9999999).ToString();
}

1 reference
private bool SendEmail(string toEmail, string code)
{
    try
    {
        var smtpClient = new SmtpClient("smtp.gmail.com")
        {
            Port = 587,
            Credentials = new NetworkCredential("ranona610@gmail.com", "jpud owiu ifwy kzyq"), // Replace with your Gmail email and App Password
            EnableSsl = true,
        };

        var mailMessage = new MailMessage
        {
            From = new MailAddress("ranona610@gmail.com"),
            Subject = "Your password reset code",
            Body = $"Your password reset code is {code}",
            IsBodyHtml = false,
        };
        mailMessage.To.Add(toEmail);

        smtpClient.Send(mailMessage);
        return true;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Failed to send email: {ex.Message}");
        return false;
    }
}
```

Second -> Enter Verification Code:

Input: Email, Code

Output: Successful, Code Error, Email Error

```
[HttpPost("verifyCode/{email}/{code}")]
0 references
public IActionResult VerifyCode(string email, string code)
{
    try
    {
        using (SqlConnection con = new SqlConnection(_configuration.GetConnectionString("PData")))
        {
            con.Open();

            string query = "SELECT Code FROM Usser WHERE Email = @Email";
            using (SqlCommand cmd = new SqlCommand(query, con))
            {
                cmd.Parameters.AddWithValue("@Email", email);
                object result = cmd.ExecuteScalar();

                if (result != null && result != DBNull.Value)
                {
                    string storedCode = result.ToString();

                    if (storedCode == code)
                    {
                        return Ok("Verification successful");
                    }
                    else
                    {
                        return BadRequest("Verification code does not match");
                    }
                }
                else
                {
                    return BadRequest("Email not found");
                }
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

Third -> Change Password:

Input: Email, Password, Confirm Password
Output: Successful, Password Error, Email Error

```
public IActionResult ResetPassword(string email, Usser user)
{
    try
    {
        if (user.NewPassword != user.ConfirmPassword)
        {
            return BadRequest("Password and confirmation password do not match");
        }

        using (SqlConnection con = new SqlConnection(_configuration.GetConnectionString("PData")))
        {
            con.Open();

            string updatePasswordQuery = "UPDATE Usser SET password = @NewPassword WHERE email = @Email";
            using (SqlCommand updatePasswordCmd = new SqlCommand(updatePasswordQuery, con))
            {
                updatePasswordCmd.Parameters.AddWithValue("@NewPassword", user.NewPassword);
                updatePasswordCmd.Parameters.AddWithValue("@Email", email);

                int rowsAffected = updatePasswordCmd.ExecuteNonQuery();

                if (rowsAffected > 0)
                {
                    return Ok("Password updated successfully");
                }
                else
                {
                    return NotFound("User not found for the given email");
                }
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

4- Get User Data

- Input: Email
- Output: User Data or Invalid email

```
[HttpGet("getUsers/{email}")]
0 references
public IActionResult GetUsers(string email)
{
    try
    {
        using (SqlConnection con = new SqlConnection(_configuration.GetConnectionString("PData")))
        {
            string query = "SELECT * FROM Usser WHERE email = @Email";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", email);
            con.Open();

            using (SqlDataReader reader = cmd.ExecuteReader())
            {
                if (reader.Read())
                {
                    Usser user = new Usser
                    {
                        username = reader.GetString(reader.GetOrdinal("username")),
                        gender = reader.GetString(reader.GetOrdinal("gender")),
                        dateOfBirth = reader.GetString(reader.GetOrdinal("dateOfBirth")),
                        phone = reader.GetString(reader.GetOrdinal("phone")),
                        email = reader.GetString(reader.GetOrdinal("email")),
                        type = reader.GetString(reader.GetOrdinal("type")),
                        imagePath = reader["image"] != DBNull.Value ? ConvertImageToBase64(reader.GetString(reader.GetOrdinal("image"))) : null
                    };

                    return Ok(user);
                }
                else
                {
                    return NotFound("User not found");
                }
            }
        }
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}
```

6.8 Using Flask to Connect to a YOLO Model

Flask is an ideal framework for creating web applications and APIs that can serve machine learning models like YOLO (You Only Look Once), a popular real-time object detection system. Here's an overview of how Flask can be used to connect to and serve a YOLO model.

-Key Features of Flask for Serving a YOLO Model:

1. **Lightweight and Minimalistic:**

- Flask's simplicity and minimalism make it easy to set up and use, allowing developers to quickly create endpoints to handle requests and serve the YOLO model.

2. **Extensible:**

- Flask can be extended with various libraries and tools necessary for handling machine learning models, such as image processing libraries (e.g., OpenCV, PIL) and deep learning frameworks (e.g., TensorFlow, PyTorch).

3. **Flexibility:**

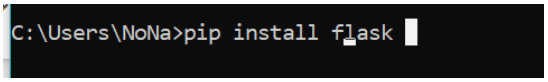
- Flask offers great flexibility in defining routes and handling requests, making it easy to build a RESTful API that accepts images, processes them through the YOLO model, and returns the detection results.

4. **Development Convenience:**

- Flask's built-in development server and debugging tools help streamline the development process, making it easier to test and refine the model-serving logic.

Steps to Connect Flask to a YOLO Model:

a. **Set Up Flask:**



b. **Load the YOLO Model:**

- Load your pre-trained YOLO model

```python
from flask import Flask, request, jsonify
from PIL import Image, ImageDraw, ImageFont
import io
import base64
from ultralytics import YOLO

app = Flask(__name__)

# Load your YOLOv8 model directly
model = YOLO('bestv8.pt')  # Adjust the path as necessary
```

c. **Create Flask API:**

- Create API to handle image uploads and return detection results
-

```python
@app.route( rule: '/detect', methods=['POST'])
def detect():
    if 'image' not in request.files:
        return jsonify({'error': 'No image file provided'}), 400

    file = request.files['image']
    img_bytes = file.read()
    img = Image.open(io.BytesIO(img_bytes))

    # Run the YOLO model
    results = model(img)

    # List to store detected class names
    detected_classes = []

    # Extract the detected class names and draw bounding boxes
    draw = ImageDraw.Draw(img)
    font = ImageFont.load_default()

    for box in results[0].boxes:
        cls = int(box.cls)
        label = model.names[cls]
        probability = float(box.conf)  # Convert tensor to float
```

```
            detected_classes.append(label)

            # Get bounding box coordinates
            x1, y1, x2, y2 = box.xyxy[0].tolist()
            x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)

            # Draw the bounding box
            draw.rectangle( xy: [x1, y1, x2, y2], outline="red", width=2)

            # Draw the label and probability
            text = f'{label} {probability:.2f}'
            text_size = draw.textbbox( xy: (x1, y1), text, font=font)  # Calculate text size
            text_width = text_size[2] - text_size[0]
            text_height = text_size[3] - text_size[1]
            draw.rectangle( xy: [x1, y1, x1 + text_width, y1 + text_height], fill="red")
            draw.text( xy: (x1, y1), text, fill="white", font=font)

    # Save the result image to a bytes buffer
    img_bytes = io.BytesIO()
    img.save(img_bytes, format='JPEG')
    img_bytes.seek(0)

    # Encode the image to base64
    encoded_img = base64.b64encode(img_bytes.read()).decode('utf-8')
```

```
    response = {
        'detected_classes': detected_classes,
        'image': encoded_img
    }

    return jsonify(response)

if __name__ == '__main__':
    app.run(port=5000)
```

Chapter 7 PROJECT TOOLS & TECHNOLOHIES

7.1 Machine Learning tools:

7.1.1 python

We use Python because it enhances productivity, is a Portable programming language, and is widely used for AI, ML, and DL with a huge community and library.

7.1.2 web scrapping

technique used to extract information from websites. It involves fetching the web page and extracting the relevant data like image from it.

7.1.3 ROBOFLOW

We used ROBOFLOW to perform Manual Annotation for the images in our data set, then ensure the quality and efficiency of the data set and extract it in a way that we can use to train our model.

7.1.4 Ultralytics

We use it to introduce the YoloV8 model to be faster, simpler and more accurate by accessing YOLO () Function.

7.1.5 Google Collab

Google Collab provides a variety of pre-installed libraries, making it easy to get started with machine learning projects. One of the key features of Google Collab is the ability to collaborate on notebooks with other users. Google Collab provides access to powerful computing resources, including GPUs. These can be used to speed up machine learning projects and reduce training times

7.2 Mobile Application tools

7.2.1 Flutter SDK

Flutter SDK is the cornerstone of our mobile application development. Flutter, an open-source UI software development kit created by Google, allows us to build natively compiled applications for mobile (iOS and Android), web, and desktop from a single codebase.

7.2.2 Dart Programming Language

Dart is the programming language used to write Flutter applications. It is optimized for building mobile, desktop, server, and web applications.

7.2.3 Integrated Development Environments (IDEs)

Two primary IDEs support Flutter and Dart development:

- **Android Studio**
- **Visual Studio Code (VS Code**

7.2.4 Networking

- For making network requests to the backend services, we use the **http** package.

7.2.5 Image Handling

To manage images within the app, we use:

- **Image Picker**
- **Base64 Encoding/Decoding**

7.2.6 UI and Animation Packages

- Enhancing the visual appeal and interactivity of the app is achieved through various UI and animation packages:

7.2.7 NET is the programming language Using Visual Studio Code

7.2.8 Flask: to connect with model Using Pycharm

7.2.9 SQL Server management: to connect with Database and Diagrams

.

Chapter 8 future work

Features:

-Mahine Learning team:

- Adding a new model that detects harmful food for certain diseases, such as chronic diseases, and also determines the appropriate food for those suffering from these diseases.
- Increasing the number of items that the model can detect, such as full meals from more than one type.
- Adding more information to the data set about different types and meals from around the world.

-Mobile Development team:

- Providing a section to help users reach places with healthy eating and nearby gym places via Google Map
- Adding a section specifically for exercise, whether thin or obese
- Providing a chat feature between users or the user and his doctor

REFERNCE :

https://www.geeksforgeeks.org/software-engineering-agile-software-development/

https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/

https://www.datacamp.com/blog/yolo-object-detection-explained

https://www.bing.com/ck/a?!&&p=0f772ff15bf249eeJmltdHM9MTcxOTAxNDQwMCZpZ3VpZD0yNWNmNGJjNC01NDFjLTY0M2QtMThjOS00NGE5NTU1YjY1MjEmaW5zaWQ9NTIwNA&ptn=3&ver=2&hsh=3&fclid=25cf4bc4-541c-643d-18c9-44a9555b6521&psq=Obesity%3A+Approximately+40%25+of+adults+in+Egypt+are+classified+as+obese%2C+highlighting+pervasive+issues+with+unhealthy+weight+management+and+dietary+habits+that+contribute+to+long-term+health+complications&u=a1aHR0cHM6Ly93d3cubmNiaS5ubG0ubmloLmdvdi9wbWMvYXJ0aWNsZXMvUE1DODQyOTkyOS8&ntb=1

https://www.bing.com/ck/a?!&&p=ab63da2bd37fce8eJmltdHM9MTcxOTAxNDQwMCZpZ3VpZD0yNWNmNGJjNC01NDFjLTY0M2QtMThjOS00NGE5NTU1YjY1MjEmaW5zaWQ9NTIxMA&ptn=3&ver=2&hsh=3&fclid=25cf4bc4-541c-643d-18c9-44a9555b6521&psq=Underweight%3A+The+prevalence+of+underweight+individuals+in+Egypt+is+estimated+at+12.3%25&u=a1aHR0cHM6Ly93d3cubmNiaS5ubG0ubmloLmdvdi9wbWMvYXJ0aWNsZXMvUE1DNjk4MTkyMC8&ntb=1

https://journals.lww.com/esnt/fulltext/2022/22010/egyptian_renal_data_system__erds__2020__an_annual.1.aspx

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9073222/