Technological Institute of the Philippines Quezon City

College of Computer Studies

SafePlate: A Mobile Application for Personalized Allergen-Free Recipe Recommendations of Philippine Food Utilizing Random Forest Algorithm

CS 304 - Final Project

In partial fulfillment for the course

CS 304 –Software Engineering 2

Submitted by:

Colesio, Christian Joseph A.
Gacer, Ronahld Redd B.
Madrigal, Danilo P. IV

Submitted to:

Date Submitted:

,

CHAPTER 1

Introduction

I. Background of the Study

Food allergy is a response by the immune system to certain foods that it identifies as harmful. When someone with a food allergy consumes a particular food, their immune system recognizes specific proteins in that food as dangerous substances and launches an attack against them. This immune response can lead to a range of symptoms, which can vary from mild to severe, and potentially life-threatening. Allergic reactions to food lead to over 30,000 emergency room visits and 2,000 hospitalizations per year (FDA 2018b; Radke et al. 2018). There are approximately 150 fatalities associated with food allergic reactions in the US annually (FDA 2018).

Each and every day, most of us rely on our phones to communicate, research, and entertain ourselves. We even need our phones to navigate in the car or on foot. With that in mind, it's unsurprising that cell phone usage has only grown over time (Jack Flynn, 2023). As of today, there are currently 6.8 billion smartphone users worldwide, and the number of mobile phone users in the Philippines in 2023 is 86.7 million.(Statista,2022). With that said, there is no doubt that we can create a mobile based application that utilizes random forest algorithms to recommend substitutions for ingredients in Philippine cuisines.

The growing knowledge of dietary restrictions and allergies has created a demand for individualized food suggestions that meet the needs of each individual. Several existing food recommendation systems frequently neglect such factors, making it difficult for people to identify

,

acceptable meals. Food recommendation systems should consider complex and diverse fine-grained user needs, such as allergies and lifestyle, in addition to the user's health needs. Because food restrictions and allergies are so diverse, it is critical to inquire about them when planning meals, designing menus, and coordinating functions. As a result, our study attempts to address this issue by developing a system that takes allergies and dietary limitations into consideration and provides individualized suggestions. Data gathering and allergy identification are all issues. Our goals include constructing a comprehensive food database, designing an algorithmic structure for allergy identification, providing an intuitive user interface, and assessing system performance. An improved user experience, higher dietary compliance, increased safety, and significant information into food allergies and dietary patterns are among the projected advantages. Finally, by giving precise and individualized meal suggestions, the researchers want to improve the eating experience for people with dietary restrictions.

II. Project Description

The project "SafePlate: Personalized Allergen-Free Recipe Recommendations" aims to develop a recommender system that is user-friendly and suggests dishes based on a user's dietary preferences, accessible ingredients, and nutritional needs. Users may also input their allergies in the system, which ensures that any recipes that include certain allergens are eliminated from the suggestions and that acceptable substitutes are offered. Other features such as filtering and sorting types of cuisine can be added to the system.

III. Project Objectives

,

The SafePlate's objective is to create and provide an application for individuals who do not have broad knowledge about different cuisines that can be done on limited ingredients or materials left on their groceries. By providing this application, it will also help those novice cooks by helping them get new ideas from the choices. Parents, for them to know what they can do with the remaining recipes. Lastly, nutritionists can share knowledge about the health benefits of such ingredients and the balancing while preparing food. The proposed object of our system is as follows:

- To help individuals spend less time on thinking what food to make in the remaining ingredients
- To identify allergies and provide them alternate ingredients that will avoid the given allergen while keeping the quality of the food
- To implement an application that is capable of getting a user input and providing an accurate output.
- To provide accurate dishes that can be made using the ingredients provided.

The ISO 9126 Model

The International Organization for Standardisation (ISO) was founded in 1946 in order to facilitate international trade, international coordination and unification of industrial standards by providing a single set of standards that would be recognised and respected (Praxiom Research Group). ISO 9126 was originally developed in 1991 to provide a framework for evaluating software quality and then refined over a further ten year period (Abran et al. 2003). Many studies criticize ISO 9126 for not prescribing specific quality requirements, but instead defining a general framework for the evaluation of software quality (Valenti 2002). We believe that this is in fact one of its strengths as it is more adaptable and can be used across many

,

systems, including e-learning systems. The original model defined six product characteristics (see Figure 1). These six characteristics are further subdivided into a number of sub characteristics (see Table 1).

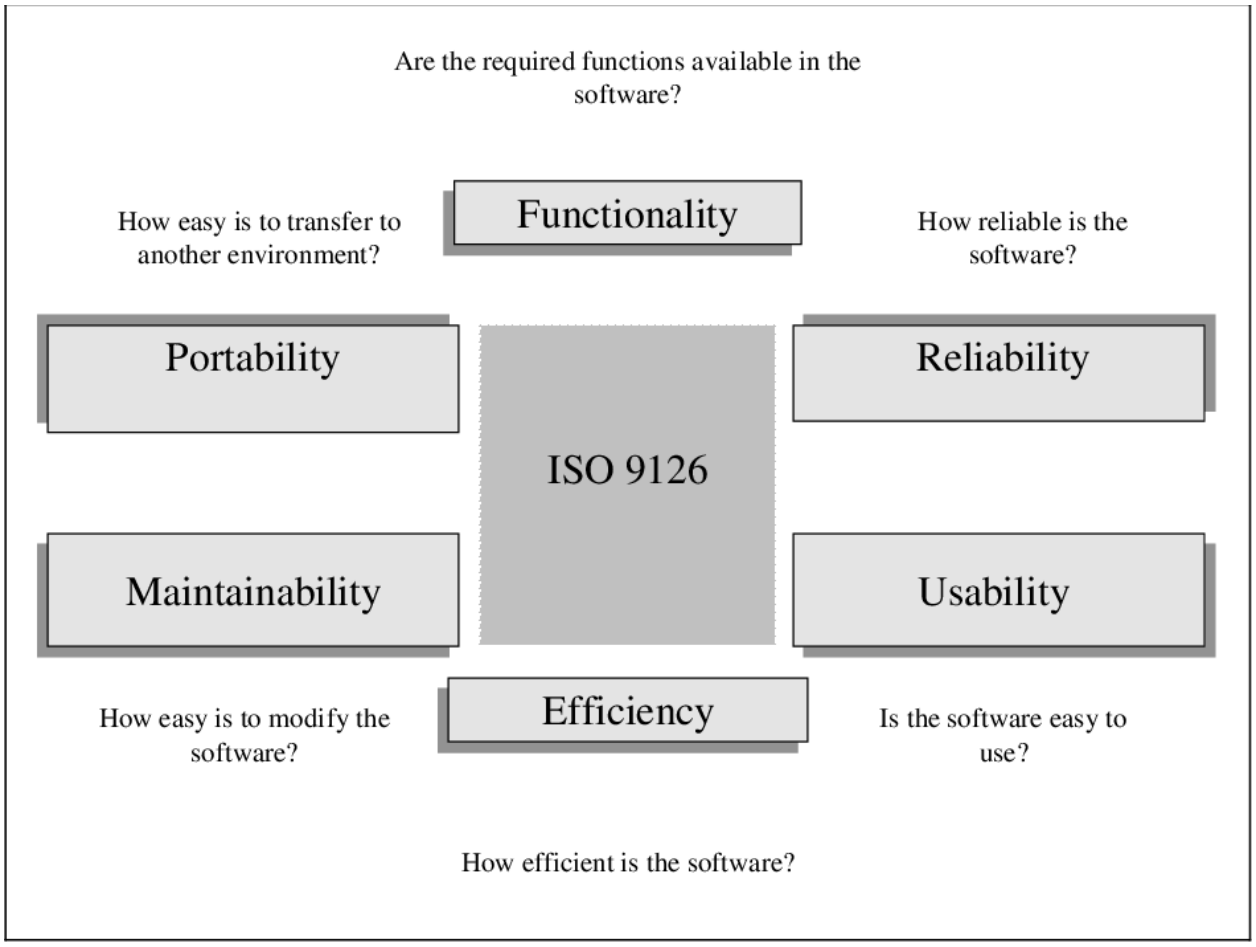


Figure 1: (Source: ISO 1991)

Table 1: ISO 9126 Characteristic and sub-characteristics (Source: ISO 1991; Abran 2003)

,

Characteristic	Sub-characteristic	Explanation
Functionality	Suitability	Can software perform the tasks required?
	Accurateness	Is the result as expected?
	Interoperability	Can the system interact with another system?
	Security	Does the software prevent unauthorised access?
Reliability	Maturity	Have most of the faults in the software been eliminated over time?
	Fault tolerance	Is the software capable of handling errors?
	Recoverability	Can the software resume working and restore lost data after failure?
Usability	Understandability	Does the user comprehend how to use the system easily?
	Learnability	Can the user learn to use the system easily?
	Operability	Can the user use the system without much effort?
	Attractiveness	Does the interface look good?
Efficiency	Time Behaviour	How quickly does the system respond?
	Resource Utilisation	Does the system utilise resources efficiently?
Maintainability	Analysability	Can faults be easily diagnosed?
	Changeability	Can the software be easily modified?
	Stability	Can the software continue functioning if changes are made?
	Testability	Can the software be tested easily?
Portability	Adaptability	Can the software be moved to other environments?
	Installability	Can the software be installed easily?
	Conformance	Does the software comply with portability standards?
	Replaceability	Can the software easily replace other software?
All characteristics	Compliance	Does the software comply with laws or regulations?

These characteristics and sub-characteristics represent a detailed model for evaluating any software system. Indeed, Abran, Khelifi, Suryn & Seffah (2003) claimed that, “Even though it is not exhaustive, this series constitutes the most extensive software quality model developed to date.” It is also an easy model for the non-specialist to employ, for example, simpler than the IEEE P1484.1 LTSA model, SCORM or IMS. Unlike these other frameworks, ISO 9126 covers a wide spectrum of system features, including both technical requirements and human interaction with the system. For example, ISO 9126 includes HCI features such as attractiveness of the interface, which is overlooked by the other standards.

One of the many objectives that the researchers want to achieve is to complete various characteristics that are present in ISO 9126. These characteristics aim to have the system assessed by model quality, data quality, and even to establish quality characteristics for target data utilized by humans

,

and the system. The researchers aim to establish four of its characteristics which are functionality, usability and efficiency.

IV. Significance of the Study

The project offers several significant benefits to various individuals, including:

- Individuals with Food Allergies - This group is the primary beneficiary of the project. The system will help them to discover new dishes that meet their dietary restrictions, expanding their menu options and helping to ensure they can enjoy a variety of foods safely.
- Novice Cooks and Food Explorers - This project can be a fantastic tool for those who are just starting their culinary journey or those who wish to broaden their culinary horizons. By providing a wide range of personalized, allergy-safe recipes, the system can help users discover new foods and cuisines.
- Parents or Caregivers - Those caring for individuals with food allergies, especially children, can have peace of mind. It can help them to come up with a diverse range of meals without risking allergen exposure.
- Dietitians and Nutritionists - These professionals can use this system as a tool to help clients with food allergies to build diverse and nutritionally balanced meal plans.
- Future Programmers and Data Scientists - This project can serve as a valuable case study for aspiring programmers, data scientists, and software engineers. It provides a real-world example of how artificial intelligence, machine learning, and data processing can be used to solve everyday problems and improve people's lives. Those interested in specializing in health tech,

,

personalization algorithms, or recommendation systems could particularly benefit from studying this project.

V. Scope and Delimitations

This study focuses on having meals or recipes that offer allergen-free foods to the users of the system. Awareness on what food to avoid and what food to eat will help people prevent such allergies to some people. This leads to the conclusion that it is important to guide people in their food decision process. Guidance in food decisions would be beneficial to people with allergies to the products presented in the system, and also to all people in general.

As an alternative guidance method, this study describes a method to promote an allergen-free eating by offering personalized recipe recommendations in a form of mobile application. For this purpose it is important to find out which characteristics of users within the target group of food allergic people and which characteristics of recipes are important for tailored recipe recommendations. It is necessary to be able to identify these characteristics automatically by the system. Knowledge about factors that influence food choice can provide further insight in how recipe recommendations can be optimized. These factors together can be used to find recipes that the user not only likes, but that he or she will actually be likely to prepare.

A limitation of this study is that we did not ask participants to actually prepare the dishes and what is the participant's available ingredients. It is possible that, although we have asked them specifically to report whether they want to prepare the dish or not for a specific day, they were still more focused on the liking of the dish, rather than preparation barriers. This may affect the frequency estimations. Additionally,

,

some recipes that the user wants may not be available to the system. Some people that has sickness or have a limited food choice may not be included in the system(e.g. Diabetes 2, Highblood). Lastly, some specific allergens may not be available to the system as it is limited to only nine major allergens for now.

,

CHAPTER 2

Theoretical Framework

Review of Related Literature

This chapter covers the local and foreign literature related to the study that supports its claims. It also covers the conceptual framework, system architecture, and definition of terms.

Food Allergies

According to 'Moen, Ø., Opheim, E., & Trollvik, A. (2019) Food allergies are one of the fastest growing public health concerns without a cure, affecting approximately 8% of the world's child population. Being the parent of a child with allergies may lead to concerns, and affect everyday life in the family. Food allergies can be a nuisance when you order or prepare some food without knowing that the food or ingredient you're allergic to is included in it. Allergic reactions to food lead to over 30,000 emergency room visits and 2,000 hospitalizations per year (FDA 2018b; Radke et al. 2018). There are approximately 150 fatalities associated with food allergic reactions in the US annually (FDA 2018). According to J. Andrew Bird, A. Wesley Burks (2023). Symptoms of a food allergy may range in severity from mild localized oral itching with ingestion of fresh fruits or vegetables in individuals with pollen-food allergy syndrome to life-threatening states in individuals with anaphylactic reactions to foods, such as peanut and etc.

The most common foods that trigger food allergies in the Philippines are wheat, eggs, milk, shellfish and peanuts. Wheat, which can be found in a variety of foods including breads, cereals and pizza can cause dermatitis to swollen airways and anaphylaxis. Eggs are the leading cause of eczema (a type of skin allergy) attacks in children. The primary trigger is the protein in egg white, although allergies to yolk are

,

uncommon. Crustaceans (crabs and shrimps) and mollusks (mussels, clams and similar seafood) can be just as harmful when consumed by an allergic individual. (Pelegrino,2019). Providing an application that will help individuals or groups lessen the time they make decisions on what to prepare or search for some substitution for their food while also preventing them from having an allergic reaction.

Recommendation Systems

Recommender Systems (RS) are the tools designed for interacting with voluminous and complex information spaces, and prioritizing those items for the users that may be of interest to them (Villegas, 2018) . When friends convene for dinner, they might share pictures on social media of the food they enjoy, and other people might turn to apps for recommendations when deciding what to eat. Today, diet preference is becoming increasingly important in meeting a variety of necessities, including basic nutrition, calories, taste, mental wellbeing, and sociocultural circumstances (J.-C. Kim and Chung, 2020, Premasundari and Yamini, 2019, Tran et al., 2021). Diet is a major contributor to the immense rise in the prevalence of obesity and diabetes (Bishop et al., 2021, Mario et al., 2022, Zhu et al., 2022). According to the Global Burden of Disease Study, dietetic patterns are a significant component to thresholds of malnourishment, fatness, and adiposity, and unhealthy diets cause 11 million preventable early deaths annually (James, et al., 2018). Food recommendation systems are emerging as a new branch of science to resolve these concerns (Ali et al., 2018, Chen and Toumazou, 2019, Vairale and Shukla, 2019). Food recommendation systems play a vital role across a wide range of online technology-based lifestyle applications, and they have become an essential component of many lifestyle services, which can, in turn, be used to influence people towards a healthy lifestyle. These systems seek appropriate food products that

,

accommodate users' daily diet preferences while enabling these individuals to satisfy their basic biological and physiological needs and adequately perform their daily activities (H. I. Lee et al., 2020, Norouzi et al., 2018, Subramaniyaswamy et al., 2019).

Food Recommendation System

Food recommender models can be viewed as intelligent algorithms that make personalized food recommendations from a wide range of choices. Using these models, researchers can efficiently address the well-known information overload problem. In recent years, food recommendation systems have gained increasing attention due to their applicability to healthy living (Mehrdad Rostami, et al., 2022).Many food recommendation systems have been proposed in recent years to predict people's preferences and/or to guide their choices based on predetermined criteria but, Despite the relative success of previous food recommender systems in learning individuals' preferences based on their historical interactions with food, these systems still suffer from several limitations (Mehrdad Rostami, 2023).

Most current studies in the food domain focus on providing users with recommendations for favorite food items based on their preferences and/or health problems (H. I. Lee, et al., 2020). But in this study, we aim to focus on a food recommendation system that substitutes allergens to non-allergen ingredients to prevent allergic reaction to the user.

Content-Based Algorithm

,

Content-based filtering focuses on recommending items to users based on the similarity of its features or attributes. In this project, contest-based filtering is used to recommend food to the users based on the activity they do such as visiting the food or adding the food to their favorites. Although there are limitations, such as its inability to get the actual user's preferences, it suffers from overspecialization where the system recommends only similar items to the user, leading to a lack of diversity in recommendations, and it requires a good understanding of each item's attributes and how they relate to the user's preferences. (Garipelly et al., 2021). Similarly, it may also lead to multidimensional analysis of recent data related with another user that may lead to biased recommendations (Zacarias et al., 2023).

Machine Learning

With the growing ubiquity of machine learning, everyone in business is likely to encounter it and will need some working knowledge about this field. A 2020 Deloitte survey found that 67% of companies are using machine learning, and 97% are using or planning to use it in the next year (Deloitte, 2020). Machine learning is a scientific study of algorithms which is used to learn from data without any human interaction.

Recommendation system or Recommender System comes under Machine learning method which has become one of the essential systems in e-commerce websites. Many techniques are proposed to efficiently capture the opinion of the users and to provide recommendations accurately. In our proposed project, we have decided to use Random Forest Algorithm to come up with our recommendation system that will focus solely on finding an alternative or recommend an alternative ingredient for the food that the user wants.

,

Random-Forest Algorithm

Random-Forest Algorithm is a machine learning algorithm that is extremely popular when it comes to classification and regression tasks. It is machine learning that utilizes decision trees to make a prediction, random forest works by creating a set of decision trees based on the subsets of the training data and features, and each decision tree is built independently by selecting a random subset to come up with the predicted value. In this project, a random forest algorithm is used to predict the food rating based on the ingredients of the dish and preparation time, with that said, the highest predicted rating will now be recommended to the user. According to (Abel Martinez-Gorospe, 2019) it is observed that the Random Forest Algorithm has a precision of 97.7381% when making the decision to recommend or not an ingredient according to the input data of each user. The advantages of random forest in the era of big data are reflected. Random forest has fast processing speed and high accuracy, which ensures the real-time and high performance of the recommendation algorithm(T. Liu, et al. 2022).

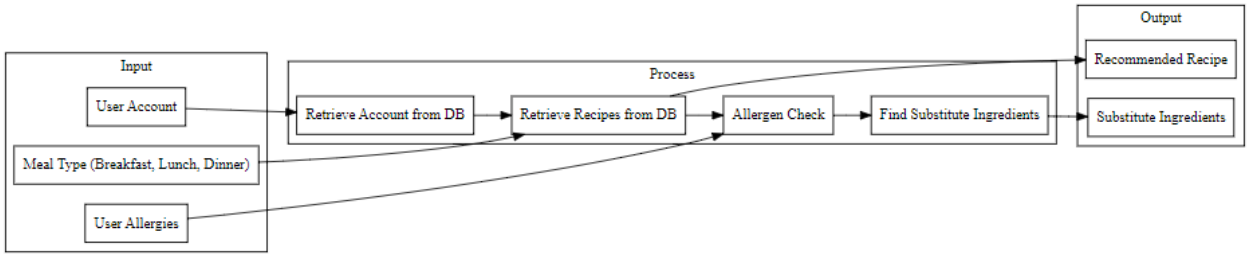
Food Substitution

According to 'Moen, Ø., Opheim, E., & Trollvik, A. (2019) Eating habits can play an important role in improving personal health. For example, dietitians recommend that patients diagnosed with diabetes monitor their intake of specific nutrients from foods (like carbohydrates and protein) to treat their condition (American Diabetes Association, 2020). If people need to adjust their intake of some nutrients, they may choose to modify their diet by substituting ingredients in their meals. Such substitutions can remove restricted types of ingredients (e.g., common allergens) or replace ingredients to adhere to some dietary constraint (e.g., replacing potatoes to reduce carbohydrate intake). Substituting individual ingredients rather

,

than strictly following a new meal plan allows people to eat familiar meals while maintaining their dietary goals. People who are looking to make substitutions have several common resources available. Popular recipe websites, such as AllRecipes and Food.com, have comment sections where community members discuss their opinions and modifications of different recipes. These websites also compile lists of common ingredient substitutions3. Another approach is to use popular search engines. Queries to search engines, using keywords such as “potato substitute,” return relevant results from a variety of websites, and queries targeting specific nutrients (e.g., “low-carb substitutes”) can sometimes find websites aggregating common substitution options for such diets. In our research, we will use food substitution to implement alternative options for users to select when they have a specific allergen.

IPO Architecture

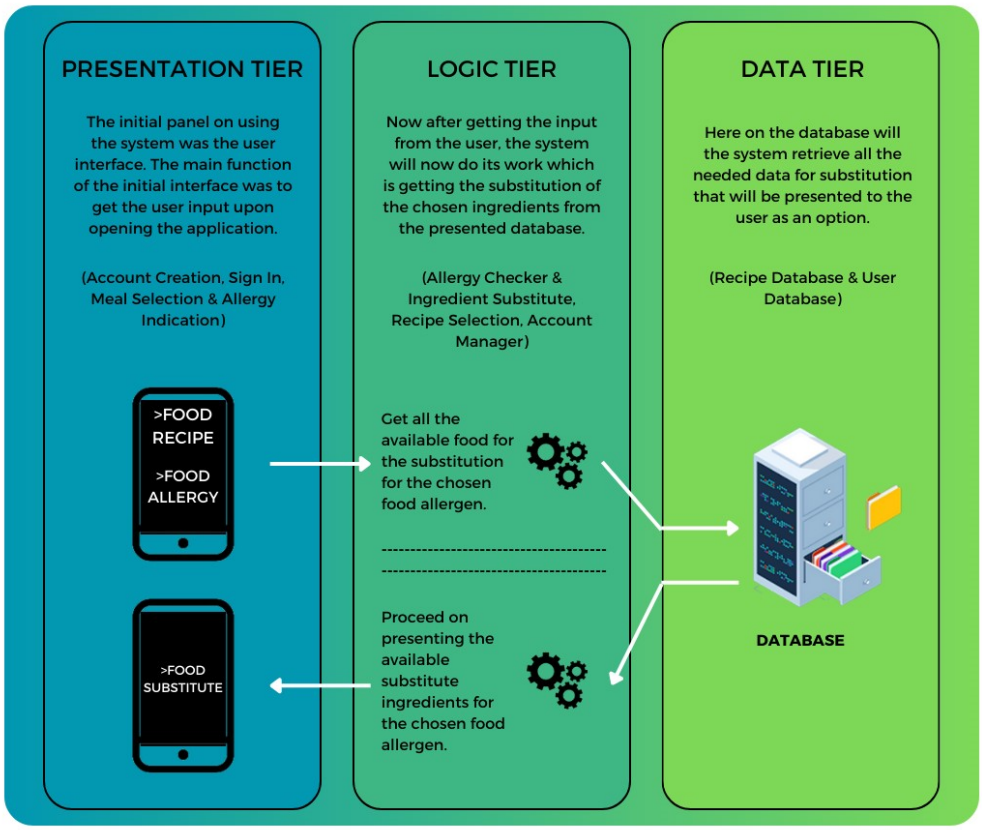


(Figure 1)

The researcher's IPO is divided into three parts: input, process, and output. The input portion needs the user to create a User account, which is used to propose recipes. The user may next choose their meal type, such as breakfast, lunch, or supper, and enter any allergies they may have. The system retrieves data from the database in the process portion, including the user's profile and meal substitutes. Finally, in the output portion, the system displays the recommended recipes and their replacement ingredients.

,

System Architecture



(Figure 2)

In this figure where there are 3 tiers, the first one was the presentation tier on which the user interacts with.

The user has an option to choose different recipes that he likes and change the ingredients that the user

,

has allergies with. The chosen ingredients will now act as an inputted variable and will now be directed to the database on where the substitute ingredients are stored. After that the system will now present it to the user as an option for the substitution of ingredients.

Definition of Terms

These are the study's operational terms that need to be defined.

Allergen-Free Recipe Recommendations - These are recipe suggestions provided by the SafePlate application that do not include ingredients that the user is allergic to. The application uses a food recommendation system and the Random Forest Algorithm to generate these recommendations.

Anaphylaxis - a common medical emergency and a life-threatening acute hypersensitivity reaction. It can be defined as a rapidly evolving, generalized, multi-system allergic reaction. Without treatment, anaphylaxis is often fatal due to its rapid progression to respiratory collapse.

Food Allergies - In the context of this study, food allergies refer to the immune system's response to certain foods that it identifies as harmful. This can lead to a range of symptoms from mild to severe, and potentially life-threatening. The study aims to address this issue by developing a system that takes allergies and dietary limitations into consideration and provides individualized suggestions.

Food Recommendation System - This is a specific type of recommendation system that makes personalized food recommendations from a wide range of choices. The system in this study uses a food

,

recommendation system to suggest dishes based on a user's dietary preferences, accessible ingredients, and nutritional needs, while also taking into account any allergies the user may have.

Food Substitution - This refers to the practice of replacing one ingredient in a meal with another in order to adhere to dietary constraints or to avoid allergens. In the context of this study, food substitution is used to provide alternative options for users when they have a specific allergen, thus preventing allergic reactions.

Random Forest Algorithm - This is a machine learning algorithm used in the study's recommendation system. It is observed to have a precision of 97.7381% when making the decision to recommend or not an ingredient according to the input data of each user.

Recommendation Systems - These are tools designed for interacting with voluminous and complex information spaces, and prioritizing those items for the users that may be of interest to them. In this study, the recommendation system is a mobile application that suggests personalized, allergen-free recipes based on a user's dietary preferences, accessible ingredients, and nutritional needs.

SafePlate - This is the name of the mobile application being developed in the study. It is designed to provide personalized, allergen-free recipe recommendations for users, taking into account their dietary preferences, accessible ingredients, and nutritional needs.

,

CHAPTER 3

Research Methodology

Project Design

In this section, diagrams show the evolution of the application's design process. The project design discussion includes the context diagram, use-case diagram, data flow diagram, and screen designs for the system.

,

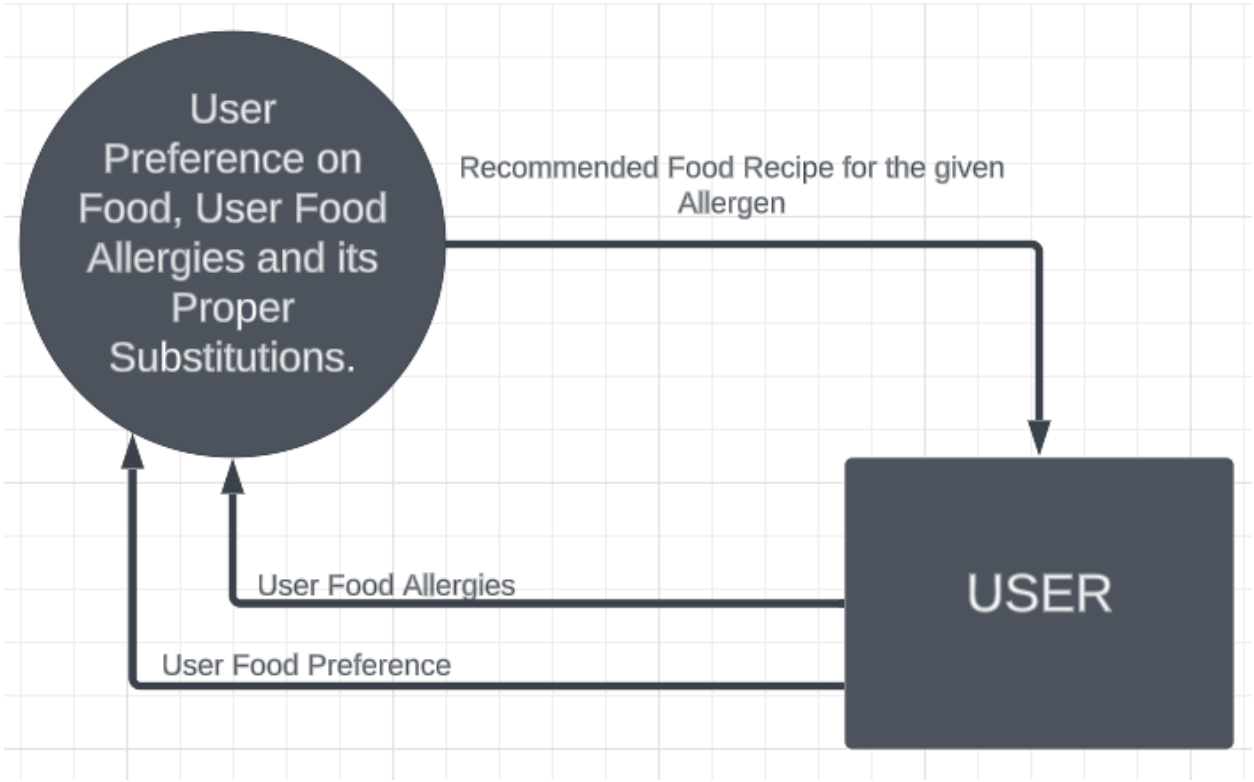


Figure 3 Context Diagram

Figure 3 shows the researcher's context diagram of their system. The food recommender system requires users to enter their favorite dish and recipe. Users must examine the items necessary to cook the dish after picking the recipe and select any substances to which they are allergic. The information is then processed by the system, which recommends acceptable substitutions for the specified food allergy, allowing the user to prepare the proposed dish with the food allergy exemption. To further enhance this process, the system might include nutritional information for the proposed dish as well as alternate recipes depending on dietary restrictions or preferences.

,

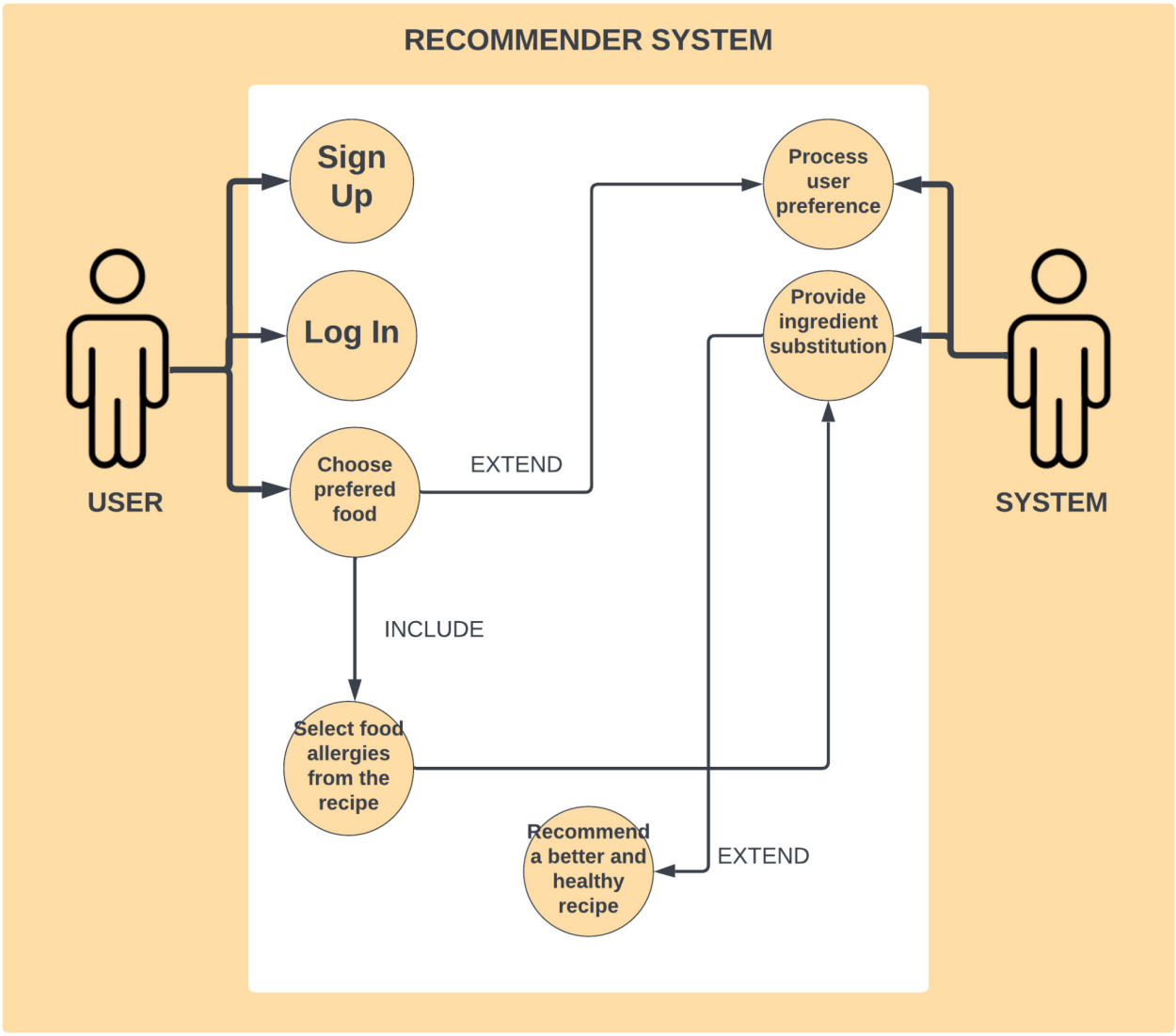


Figure 4 User-Case Diagram

The Use Case Diagram depicts all of the user's procedures for manipulating data within the program. The system will be activated by the users. It comprises Sign Up to establish a new user account, Log In to access the system by entering their credentials, and an error message if they are invalidated. Once authenticated, the system will direct the user to the Homepage; once inside the application, the system will provide an array of preferred food recipes ranging from lunch to dinner, followed by the activation of the

,

recommender system, which will provide various substitutions for the user's selected food allergies, resulting in a great and healthier me.

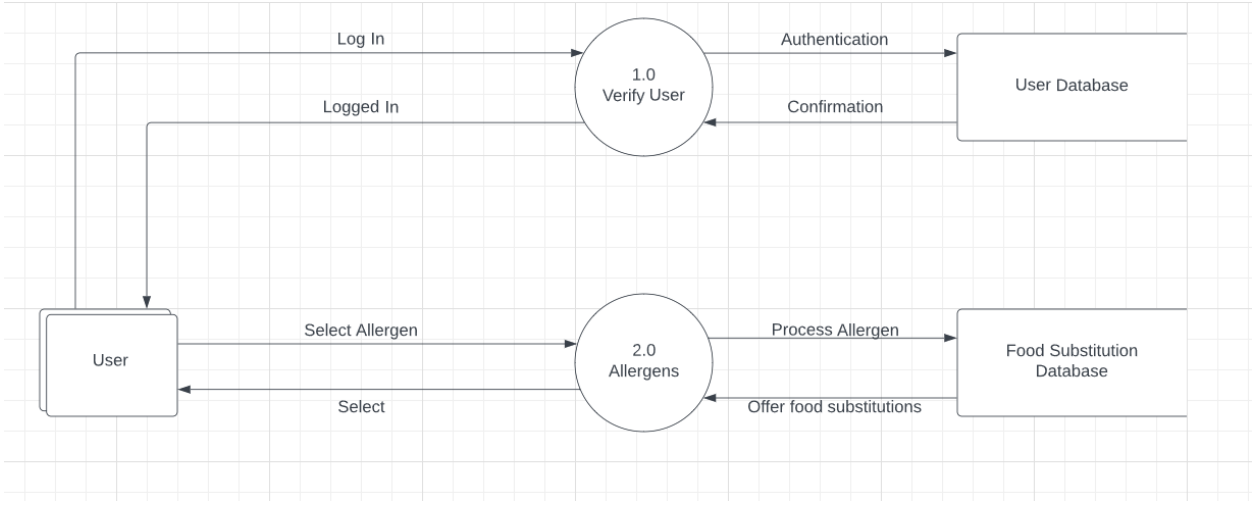


Figure 5 DFD-Level 0

In this Data Flow Diagram - Level 1 the first process is the user has to log in using his/her username and password. Once the user is verified from the user database he/she is logged in. Now the user can select the food allergen he/she is allergic to and the input is read by the allergen database. Now the system searches for the appropriate food substitute which is rated highly by other users and displays it to the user. Now the user can select recipes he wants to cook/prepare from the recommended list.

,

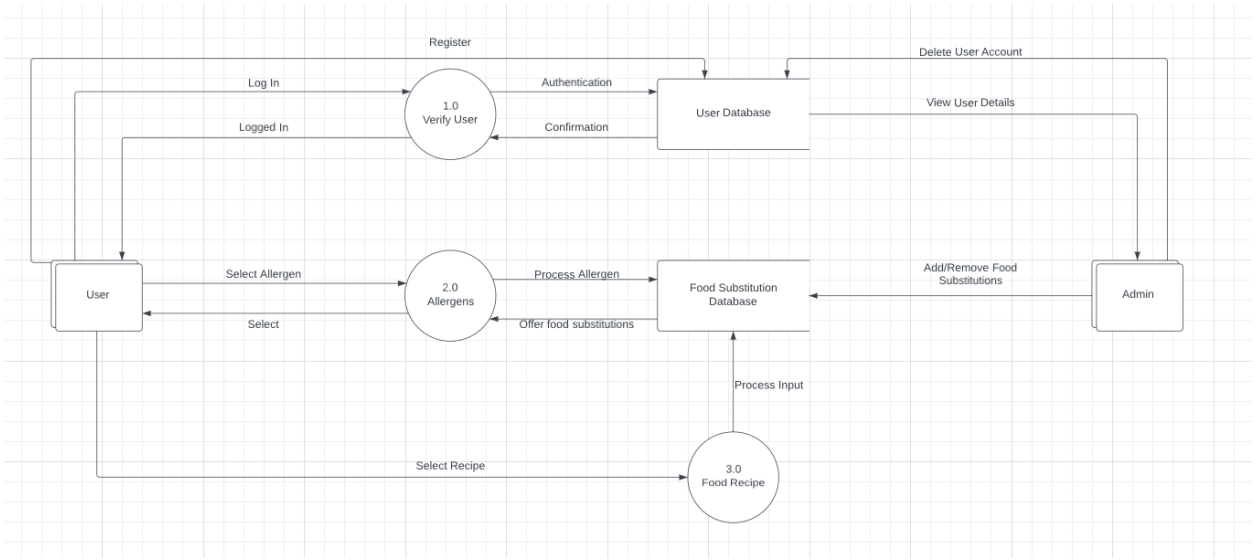


Figure 6 DFD Level 1

In this figure the researchers already expanded the needed connections for certain processes and added another entity which is admin on where it acts as an observer for the accounts being created and can do certain tasks such as deleting accounts. The researchers also added the register option for the user on where they have the ability to create an account for them to have a preference when it is inside the system, this will help to distinguish the different types of users and most likely to have a different perspective about food preferences.

,

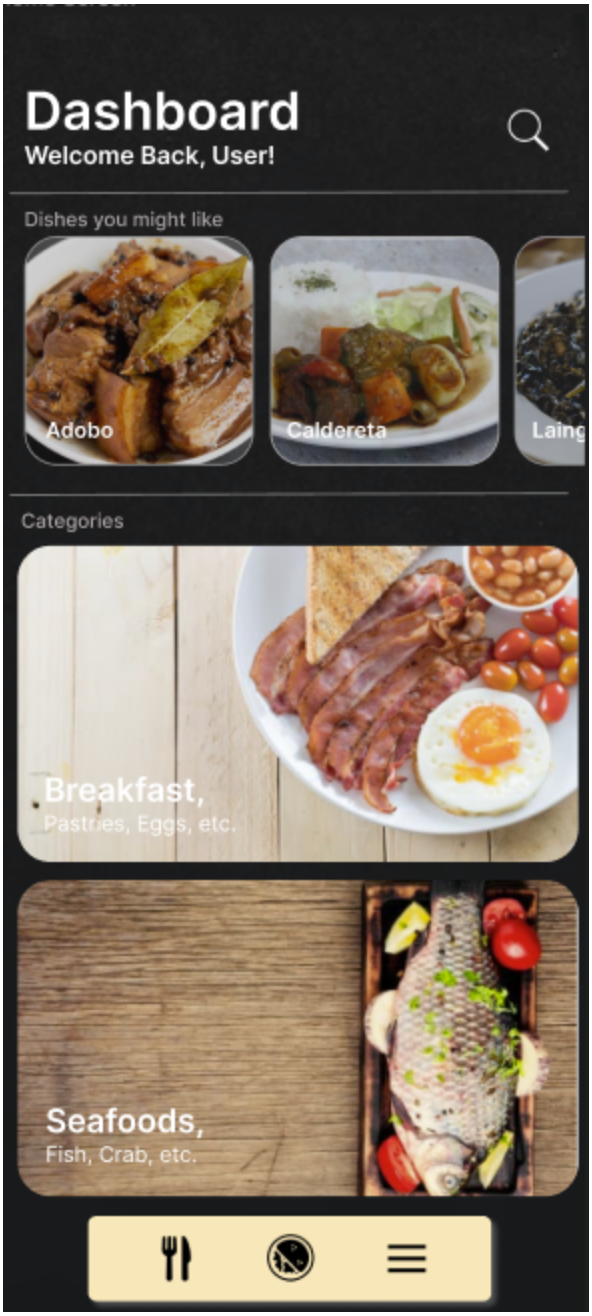


Figure 6. Sample Screen

Figure 6. shows the dashboard of the system on which the users can choose their preferred food for the day and the system will present a variant of cuisines that you can pick off. You can also see categories where it recommends things similar to your favorite food.

,

Project Development

The system's project development is discussed in this section. This contains the software development methodology used in this project with specific development phases.

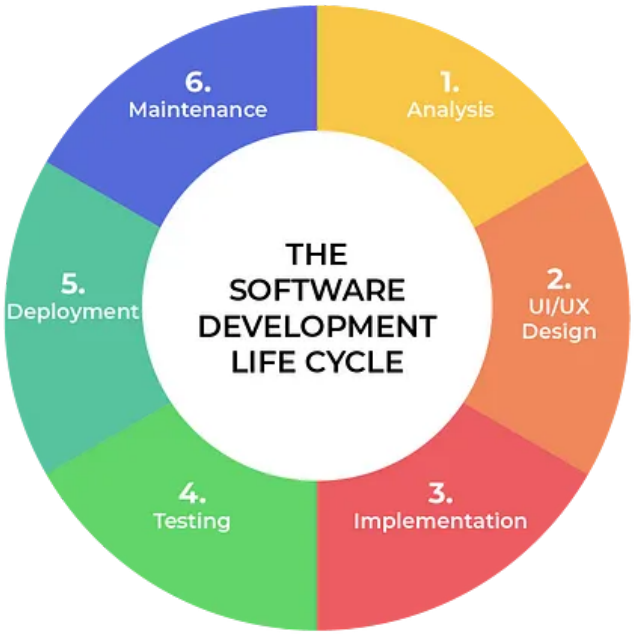


Figure 7. SDLC Model

In this figure it shows the model that was used for the whole development of the project. The SDLC model has six (6) phases. The Requirement Analysis, System Design, Implementation, Testing, Deployment and Maintenance. The SDLC model is suitable for developing the system since it works well in small projects such as this system. It is an upgraded model of waterfall model on which it is used for efficiency on creating projects with a predefined scope, deadline for completion and even a space for revisions. The requirements

,

are easily met and can be arranged in an efficient way for a better understanding of the program. By using this methodology we can easily finish a process and move onto another, while still having a possibility to come back for every detail that we have included on the procedure.

Requirement Analysis

In this part of the study, the researchers will gather the requirements needed to build up the project. The researchers will tackle the different perceptions of people on using different ingredients in a specific recipe to achieve the best outcome of their food. Also the researchers will determine the different top variants of allergens that most of the people affected with. They will conduct a survey of credible people about the different allergens that most of the population encounter. The researchers will evaluate gathered data to ensure that all requirements are met.

System Design

In this phase, the researchers created a context diagram, use case diagram, and screen design that will help to create this system. Also, they designed and created the system architecture of the study, the whole outline of the system, including the backend and the front-end of the system.

Implementation

The researchers developed the system to help people with different food allergies to make a healthy and allergen safe food recipe for them, and recommend to them a more beneficial and tasty food recipe. The project design was used in this specific way, and was made as a working application for mobiles and tablets. The researchers developed the front-end and back-end of the

,

system using Android Studio responsible for making its user interface, and Visual Studio Code that is responsible for making the back-end, also for connecting the UI to the main algorithm and Firebase for its database.

Testing

In this phase the researchers tested the system for any results and failures in multiple ways. The researchers used a certain type of testing and that is unit testing where it consists in testing individual methods and functions of classes, components or modules used by the application. The researchers used white-box testing to assess the code and black-box testing to evaluate the application internal structure.

Deployment

In this phase the researchers will conclude the testing of the application software on whether it passes the test or not. If the application passes the test, the researchers will now deploy the application for the available use of the product.

Maintenance and Support

In this phase the researchers will keep in touch about their product on which they will be updated on the issues of it for further development, add features, maintenance, and even make necessary updates.

,

Constraints and Trade-off Analysis

In this segment, the constraints and trade-off analysis of the program are used to determine what design and detail are more appropriate to use for the system. The researchers will compare three (3) different algorithms and compare them to each other for making the comparison analysis. By doing this it will justify why that specific algorithm was used in the system.

Constraints

This section includes the discussion of the criteria being used on the system's algorithm for the assessment of the overall capabilities of the system. Each criterion will come from calculations of existing related studies that also used the same algorithm. This will be the basis for which algorithms will be the best to be used in the system.

Accuracy. The accuracy of the system will be the one to reflect on how well the system can predict. This factor determines the accuracy of the algorithm that was being used currently onto the system to perform the recommender system.

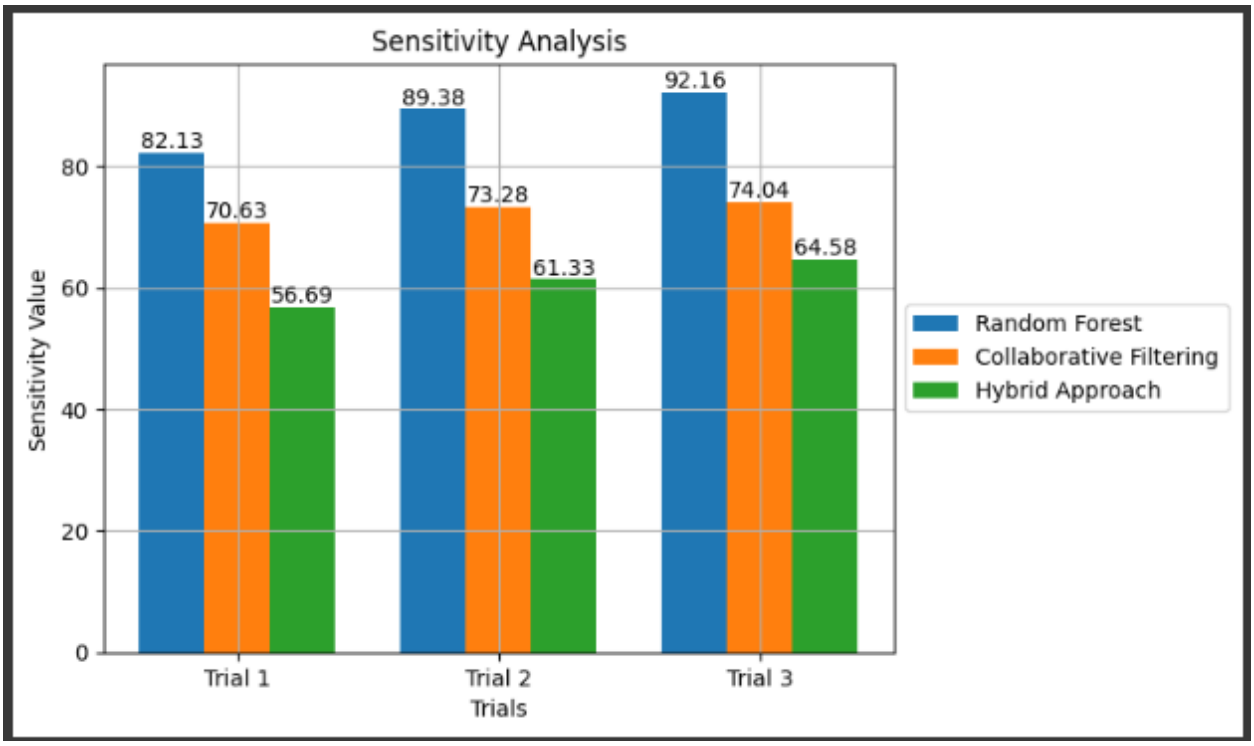
Memory Allocation. This criterion is important because it contributes to the system's efficiency. It will calculate how much an algorithm takes space or the memory needed to run a function.

Speed (Turnaround Time). This part refers to the time being taken to complete a single process. By this criterion it can now determine how quick the system will respond.

Sensitivity Analysis

,

Sensitivity Analysis Combination	Random Forest	Collaborative FIltering	Hybrid Approach
Trial 1	82.13	70.63	56.69
Trial 2	89.38	73.28	61.33
Trial 3	92.16	74.04	64.58



In this figure the researchers show the result of the sensitivity analysis. Coming from the three (3) algorithm, the Random Forest has the highest score compared to the two algorithms. This shows that the algorithm is the one that the researchers will use upon the creation of the system.

Algorithm Implementation

,

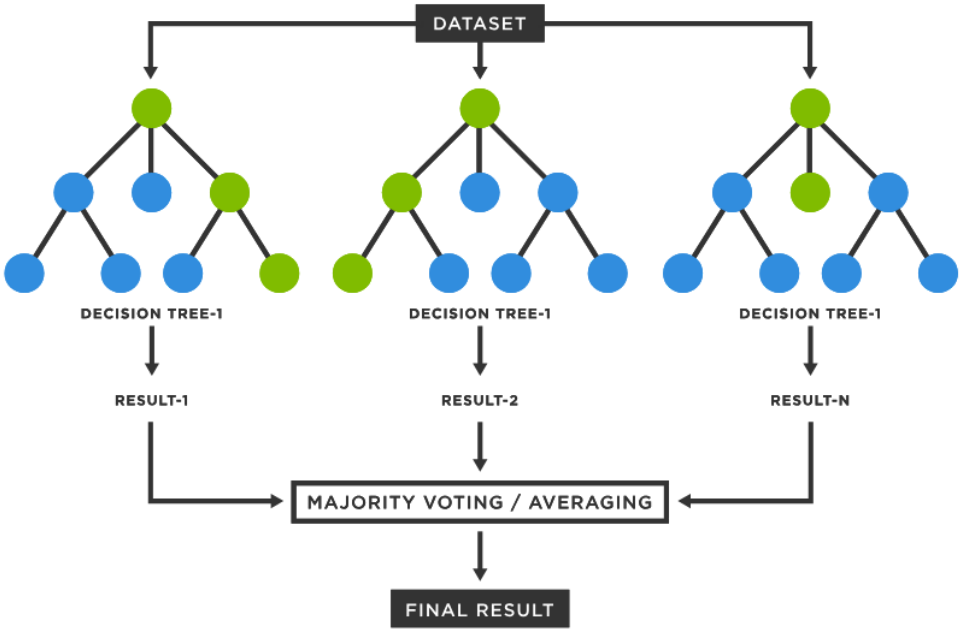
The Random Forest algorithm is proposed to be the focus of the study. The three (3) prediction models, Random Forest, Collaborative Filtering, and the Hybrid Approach were compared and conducted many trials for determining their constraints and differentiating them using the trade-off analysis. THe result showed that the Random Forest algorithm was more effective than the other two algorithms. The Random Forest algorithm has the highest accuracy compared to the other models which are the main focus for creating the recommender system.

Random Forest Algorithm

Random forest is a Supervised Machine Learning Algorithm commonly used in classification and regression problems. It constructs decision trees from various samples and uses their majority vote for classification and average for regression.

How does the algorithm work?

,



The Random Forests in this illustration are a mixture of decision trees. It generates many decision trees and merges them to get a more accurate and dependable forecast. Each tree's leaf node represents the decision tree's final result. A majority-voting mechanism or bagging is used to determine the final outcome. In this situation, the random forest's ultimate output is the output picked by the majority of decision trees. The amount of trees in the random forest influences the findings; the more trees in the forest, the more accurate and solid the prediction is.

Dataset - a collection of data.

Decision tree - When a decision tree receives a data set with features as input, it will generate a set

,

of prediction rules.

Prediction - a process that operates among multiple decision trees to achieve the best possible result by selecting most of them as the best value.

Non-Technical Evaluation

Evaluation tool for mobile application developers.

CRITERIA	1	2	3	4	5
FUNCTIONALITY					
The application give results based on the user preferences.					
The application successfully recommends food preference.					
USBAILITY					
The application user interface is easy to navigate					
The application is not complicated to use					
The application has a well designed user interface.					
EFFICIENCY					
The application has a quick load time					
The application responds quickly to user interaction					

Technical Evaluation

Evaluation tool for application development.

,

CRITERIA	1	2	3	4	5
FUNCTIONALITY					
The application give results based on the user preferences.					
The application gives clear instructions.					
The application successfully recommends food substitution based form the chosen food allergen.					
USBAILITY					
The application provides a clear function of buttons, fields and even color scheme assist.					
The application function as intended					
The application is simple to navigate and use					
EFFICIENCY					
The application can quickly responds based on the action of the user					
The application state is optimized					

Work Plan (Work Breakdown Structure)

Thesis 1	Lead	Week 1						Week 2							Week 3							Week 4						
		M	T	W	TH	F	S	S	M	T	W	TH	F	S	S	M	T	W	TH	F	S	S	M	T	W	TH	F	S
		12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6	7	8
Title Proposal	**Team**																											
Title Approval	**Team**																											
Chapter 1 Introdcution	**Team**																											
1.1 Background of the Study	**Team**																											
1.2 Objectives of the Study	**Team**																											
1.3 Significance of the Study	**Team**																											
1.4 Scope and Delimitation	**Team**																											
Revision of Chapter 1	**Team**																											
Chapter 2 Theoretical Frame	**Team**																											
2.1 Review of Related Literatur	**Team**																											
2.2 Design Concept	**Team**																											
2.3 System Architecture	**Team**																											
2.4 Definition of Terms	**Team**																											
Revisionof chapter 1 & 2	**Team**																											

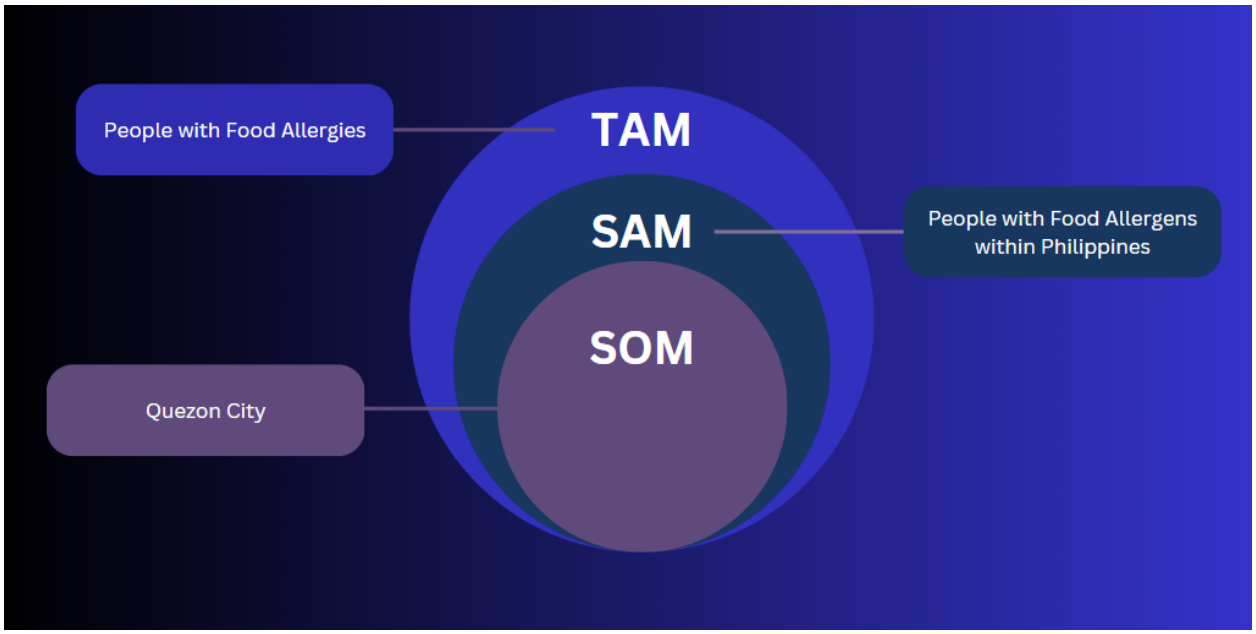
,

Chapter 3 Research Methodl	**Team**																											
3.1 Materials	**Team**																											
3.1.1 Software	**Team**																											
3.1.2 Hardware	**Team**																											
Algorithm Implementation	**Team**																											
Testing and Operating Procedu	**Team**																											
Project Evaluation	**Team**																											
Work Plan	**Team**																											
Potential for Commercializat	**Team**																											
Three Year Product Roadmap	**Team**																											
Go-To-Market	**Team**																											
Business Model	**Team**																											

Market Model

In this section the Total Available Market(TAM), Serviceable Available Market(SAM), and Share of Market(SOM) of this project is introduced. The researchers want to introduce the created system to the people who suffer from different allergens for them to have a healthy and trustworthy application that they can rely on creating their daily meal. The SAM focuses only on the small part of the population in the Philippines with specific food allergens where the system will introduce a variant of Filipino cuisine and a proper food substitutions for the chosen food allergy. While the SOM, will be introduced to people inside Quezon City.

,



Three years product roadmap

	2023	2024	2025
Performance	Accuracy, Speed & Efficiency	Accuracy, Speed & Efficiency	Accuracy, Speed & Efficiency
Price	Free	Free	Free
Features	•Allergen Exception •Food Substitution •Recipe Recommender	•Enhance GUI •Increase Recipe Option •Addidtional Allergens	•Food Recommenderwith illness exception

Go To Market Strategy

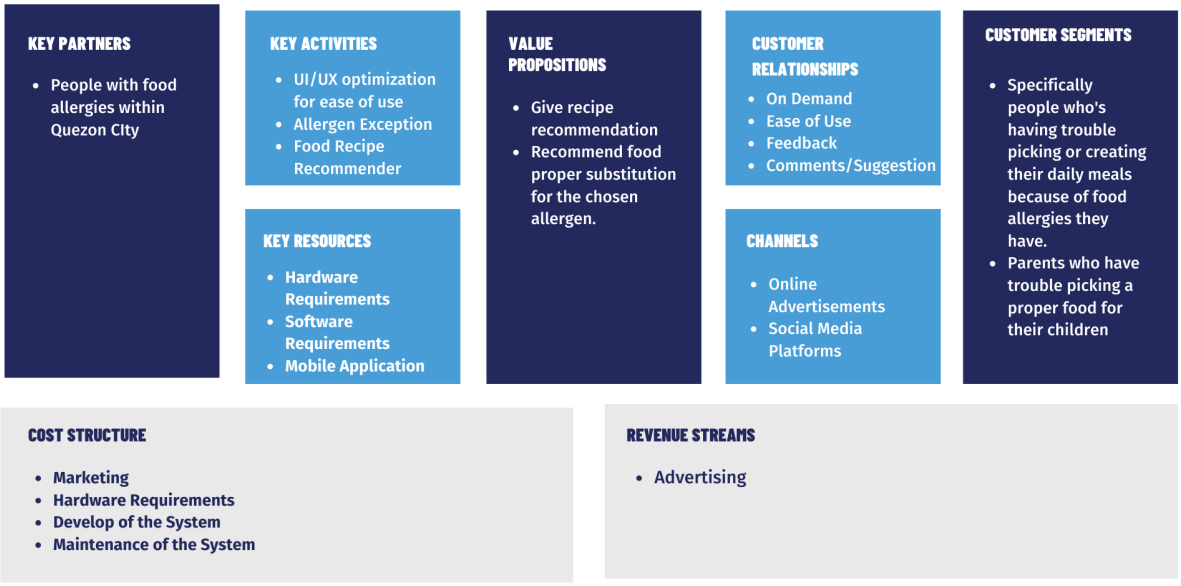
,

Go To Market Stratgey		
Offering	Customer	Channels
•Free to use	•People with allergens	•Advertisements
•User-Friendly	•Parents	•Social Media Page
•Efficient to use		

Strategy Business Model

SafePlate: A Mobile Application for Personalized Allergen-Free Recipe Recommendations of Philippine Food Utilizing Random Forest Algorithm

Business Model



,

In this figure the researchers wanted to show information about the application development, customer segmentation, cost structure, revenue sources, and other relevant business information.

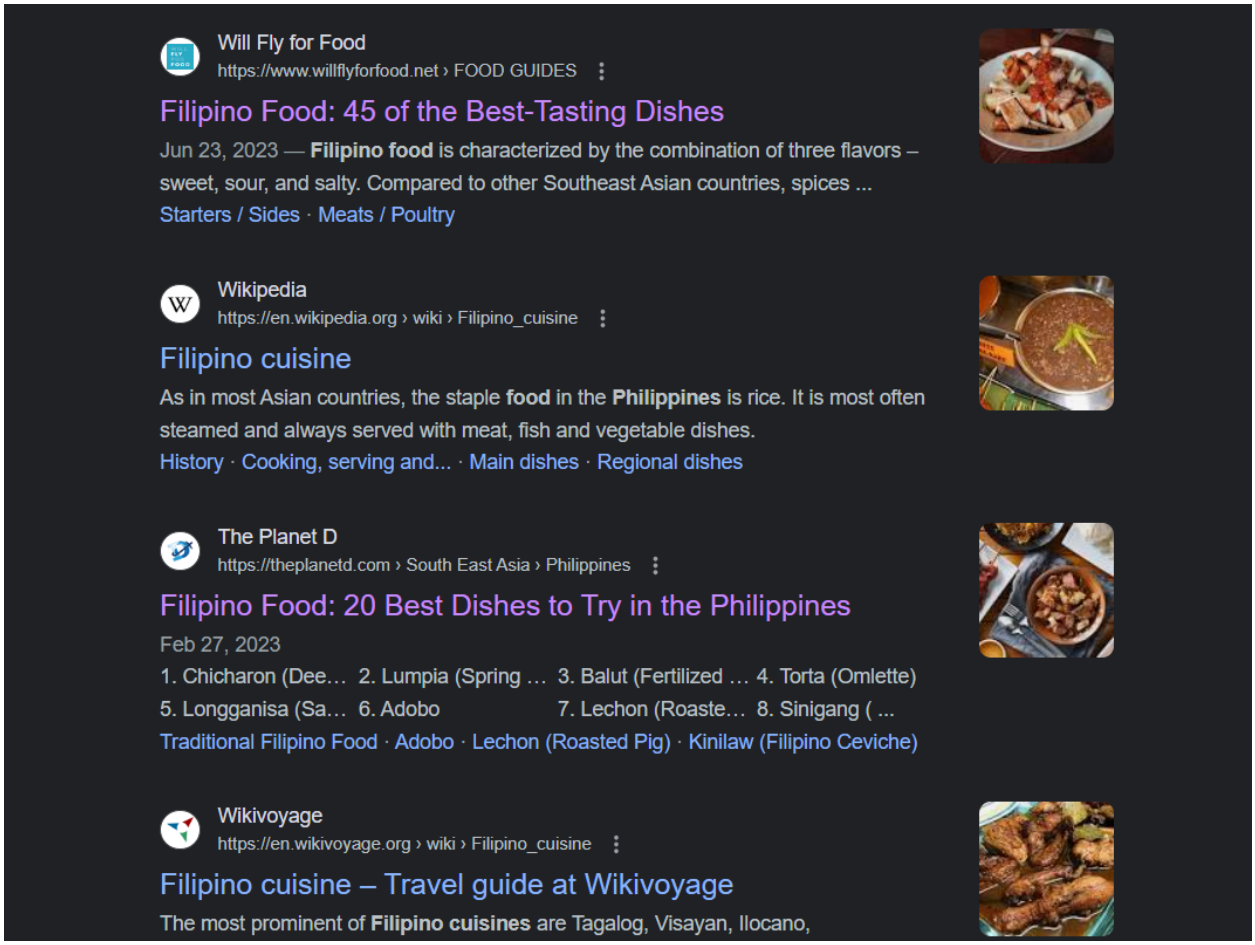
Chapter 4

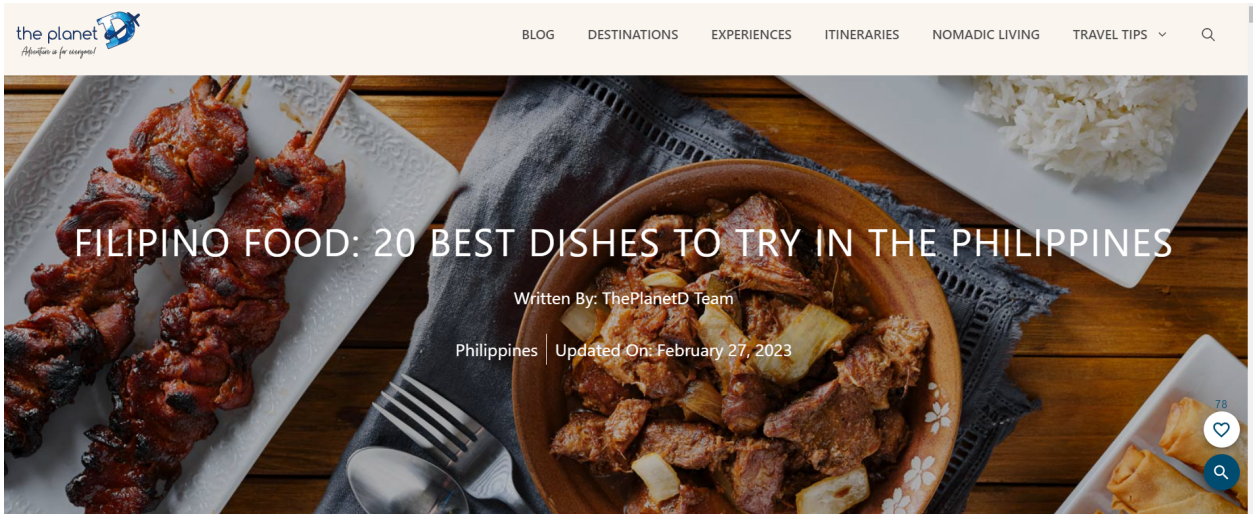
Result and Discussion

This chapter covers system development, feature prediction, data collecting, model training, data cleaning, database management, and system assessment. The proponents were able to fulfill their aims by employing the Iterative Model, which allowed them to assess each component and simplify the work that needed to be done at once in each of these stages.

Data Analysis

Dataset Gathering

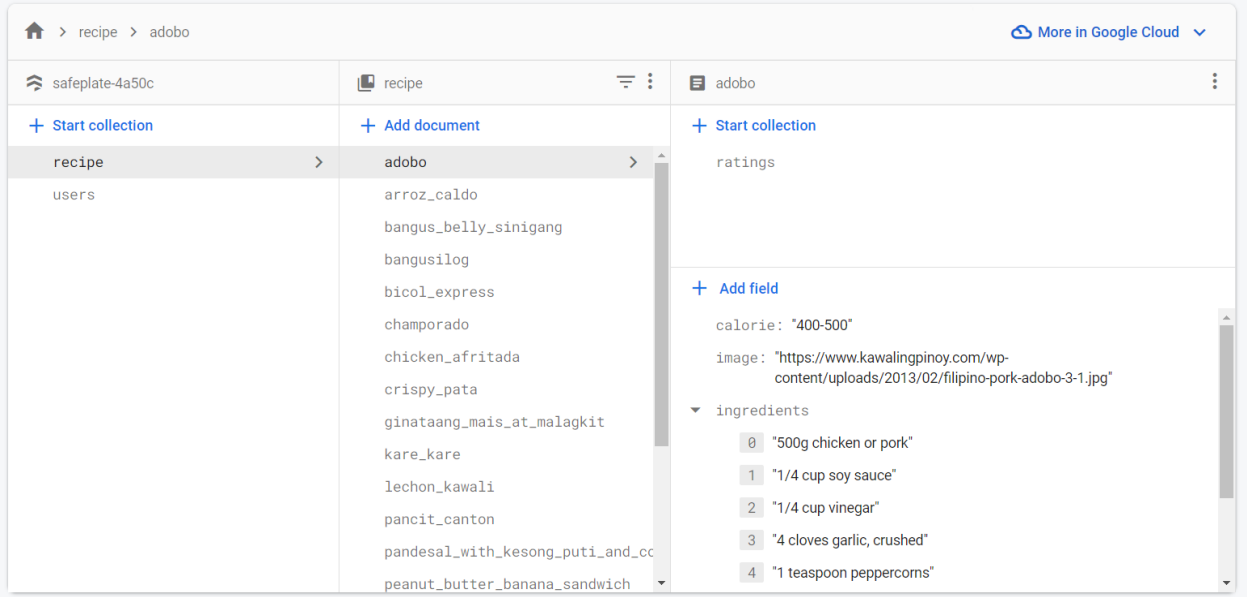

Will Fly for Food
https://www.willflyforfood.net › FOOD GUIDES
Filipino Food: 45 of the Best-Tasting Dishes
Jun 23, 2023 — Filipino food is characterized by the combination of three flavors – sweet, sour, and salty. Compared to other Southeast Asian countries, spices ...
Starters / Sides · Meats / Poultry
Wikipedia
https://en.wikipedia.org › wiki › Filipino_cuisine
Filipino cuisine
As in most Asian countries, the staple food in the Philippines is rice. It is most often steamed and always served with meat, fish and vegetable dishes.
History · Cooking, serving and... · Main dishes · Regional dishes
The Planet D
https://theplanetd.com › South East Asia › Philippines
Filipino Food: 20 Best Dishes to Try in the Philippines
Feb 27, 2023
1. Chicharon (Dee… 2. Lumpia (Spring … 3. Balut (Fertilized … 4. Torta (Omlette)
5. Longganisa (Sa… 6. Adobo 7. Lechon (Roaste… 8. Sinigang (...
Traditional Filipino Food · Adobo · Lechon (Roasted Pig) · Kinilaw (Filipino Ceviche)
Wikivoyage
https://en.wikivoyage.org › wiki › Filipino_cuisine
Filipino cuisine – Travel guide at Wikivoyage
The most prominent of Filipino cuisines are Tagalog, Visayan, Ilocano,


the planet
BLOG
DESTINATIONS
EXPERIENCES
ITINERARIES
NOMADIC LIVING
TRAVEL TIPS
FILIPINO FOOD: 20 BEST DISHES TO TRY IN THE PHILIPPINES
Written By: ThePlanetD Team
Philippines
Updated On: February 27, 2023

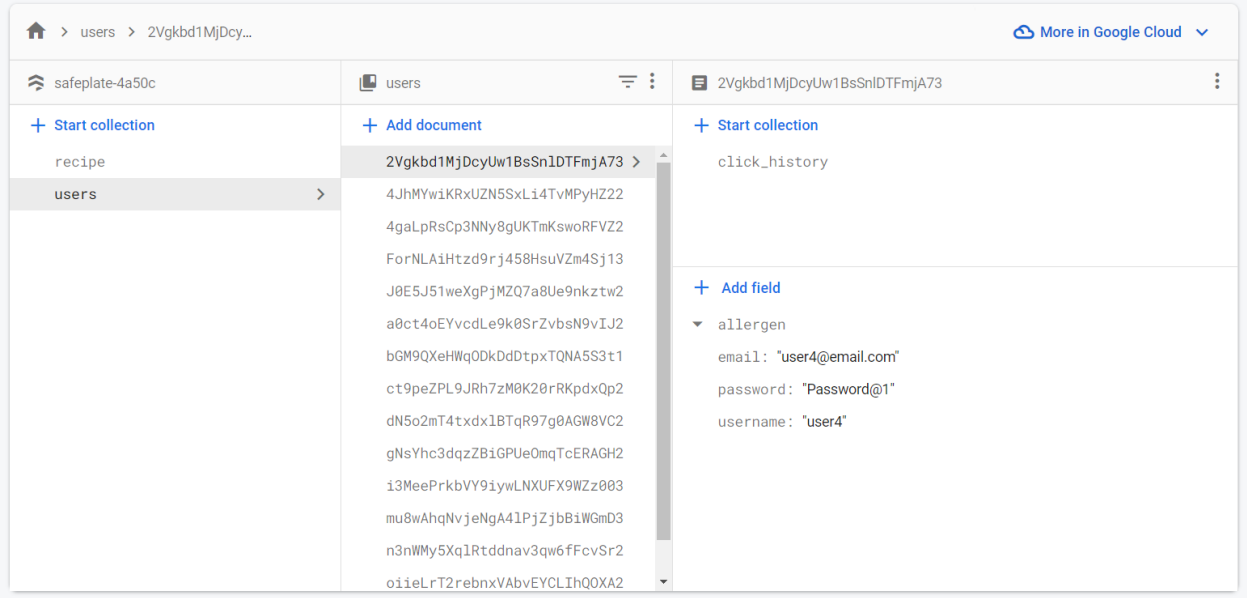
,

The figures above shows how the researchers gather their needed data for the system to work by browsing a lot of Filipino cuisines and different existing meals that are composed of different ingredients that might be a great example on having a food allergen and may provide an excellent output upon using.

Database



,



The figure above shows the use of Firestore, the researchers database needed for the system, consisting of two (2) collections which are the recipe and users. The recipe collection consists of different food recipes that will be used for the recommendation in the application, it also consists of different food allergens presented on the recipe and the rating of each user on that specific recipe, this will be needed for the functionality of the recommendation system. While the user database contents are the user's username, email, and password that will be needed for the authentication, it also consists of a document called user_history that tracks the recipes that the user used.

Database Connection and Model loading

Database connection to the system

,

Sign-Up Controller

```dart
class SignUpController extends GetxController {
  static SignUpController get instance => Get.find();

  final username = TextEditingController();
  final password = TextEditingController();
  final confirmPassword = TextEditingController();
  final email = TextEditingController();

  Future<void> registerUser(
      String email, String password, String username) async {
    await AuthenticationRepository.instance
        .createUserWithEmailAndPassword(email, password, username);
  }

  Future<bool> loginUser(String email, String password) async {
    try {
      await AuthenticationRepository.instance
          .signInWithEmailAndPassword(email, password);
      return true;
    } on SignInWithEmailAndPasswordFailure catch (e) {
      // Pass the error up to the UI
      throw e;
    } catch (e) {
      throw Exception("An unknown error occurred");
    }
  }
}
```

Authentication Repository

```dart
import 'package:firebase_auth/firebase_auth.dart';
import 'package:get/get.dart';
import 'package:splashscreen/LoginScreen.dart';
import 'package:splashscreen/SignupScreen.dart';
import 'package:splashscreen/SplashScreen.dart';
import 'package:splashscreen/allergenScreen.dart';
import 'package:splashscreen/signinWithEmailAndPasswordFailure.dart';
import 'package:splashscreen/signupWithEmailAndPasswordFailure.dart';
```

,

```
import 'package:cloud_firestore/cloud_firestore.dart';

class UsernameAlreadyExistsException implements Exception {
  final String message;
  UsernameAlreadyExistsException(this.message);
}

class AuthenticationRepository extends GetxController {
  static AuthenticationRepository get instance => Get.find();

  //Variables
  final _auth = FirebaseAuth.instance;
  late final Rx<User?> firebaseUser;

//to check if user is logged or not
  @override
  void onReady() {
    firebaseUser = Rx<User?>(_auth.currentUser);
    firebaseUser.bindStream(_auth.userChanges());
    ever(firebaseUser, _setInitialScreen);
  }

  _setInitialScreen(User? user) {
    user == null
        ? Get.offAll(() => const LoginScreen())
        : Get.offAll(() => const SplashScreen());
  }

  Future<void> createUserWithEmailAndPassword(
      String email, String password, String username) async {
    try {
      final usernameQuery = await FirebaseFirestore.instance
          .collection('users')
          .where('username', isEqualTo: username)
          .get();

      if (usernameQuery.docs.isNotEmpty) {
        throw UsernameAlreadyExistsException('Username already taken');
      }
      UserCredential userCredential = await _auth
          .createUserWithEmailAndPassword(email: email, password: password);
      firebaseUser.value = userCredential.user;
```

,

```
    // Save user data to Firestore
    await FirebaseFirestore.instance
        .collection('users')
        .doc(userCredential.user!.uid)
        .set({
      'email': email,
      'password': password,
      'username': username,
    });

    firebaseUser.value != null
        ? Get.offAll(() => const LoginScreen())
        : Get.to(() => const SignupScreen());
  } on FirebaseAuthException catch (e) {
    final ex = SignUpWithEmailAndPasswordFailure.code(e.code);
    print('FIREBASE AUTH EXCEPTION- ${ex.message}');
    throw ex;
  } catch (e) {
    print('EXCEPTION - ${e}');
    throw const SignUpWithEmailAndPasswordFailure();
  }
}

Future<void> signInWithEmailAndPassword(String email, String password) async {
  try {
    UserCredential userCredential = await _auth.signInWithEmailAndPassword(
        email: email, password: password);
    firebaseUser.value = userCredential.user;

    // Go to HomeScreen if login is successful
    Get.offAll(() => const allergenScreen());
  } on FirebaseAuthException catch (e) {
    final ex = SignInWithEmailAndPasswordFailure.code(e.code);
    print('FIREBASE AUTH EXCEPTION- ${ex.message}');
    throw ex;
  } catch (e) {
    print('EXCEPTION - ${e}');
    throw const SignInWithEmailAndPasswordFailure();
  }
 }
}
```

,

Splash Screen

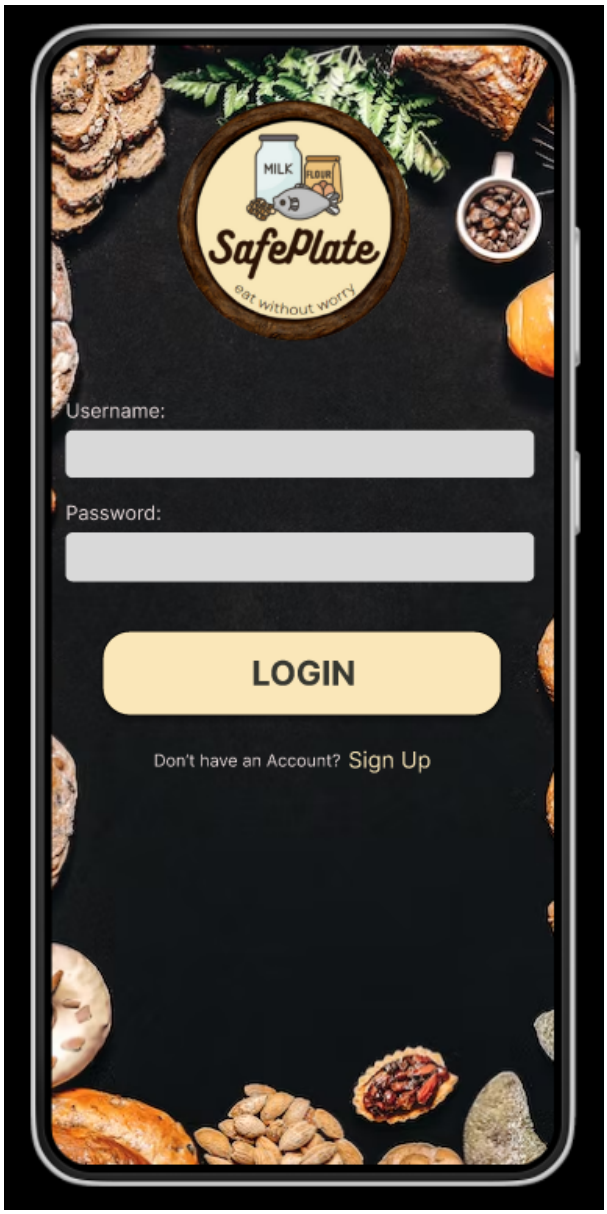

```
import 'package:flutter/material.dart';
import './LoginScreen.dart';

class SplashScreen extends StatelessWidget {
  const SplashScreen({Key? key}) : super(key: key);

  @override
```

,

```
Widget build(BuildContext context) {
  return Scaffold(
      body: Stack(
    children: [
      Image.asset(
        "assets/images/bgImgtest.png",
        width: 500,
        fit: BoxFit.fill,
      ),
      SafeArea(
        child:
            Column(crossAxisAlignment: CrossAxisAlignment.center, children: [
          const SizedBox(
            height: 42,
          ),
          Image.asset(
            "assets/images/logo.png",
            width: 500,
            height: 500,
          ),
          const SizedBox(
            height: 90,
          ),
          Text("Get Started ",
              style: TextStyle(
                  fontSize: 18,
                  fontWeight: FontWeight.w400,
                  color: Colors.white)),
          IconButton(
            iconSize: 40,
            color: Colors.white,
            icon: const Icon(Icons.keyboard_double_arrow_down),
            onPressed: () {
              Navigator.push(context,
                  MaterialPageRoute(builder: (context) => LoginScreen()));
            },
          ),
        ]),
      )
    ],
  ));
 }
}
```

,

Login Page




```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:fluttertoast/fluttertoast.dart';
```

,

```
import 'package:get/get.dart';
import 'package:splashscreen/controllers/signupController.dart';
import 'package:splashscreen/forgotPassScreen.dart';
import 'package:splashscreen/signinWithEmailAndPasswordFailure.dart';
import './SignupScreen.dart';

class LoginScreen extends StatefulWidget {
  const LoginScreen({Key? key}) : super(key: key);

  @override
  _LoginState createState() => _LoginState();
}

class _LoginState extends State<LoginScreen> {
  @override
  Widget build(BuildContext context) {
    final controller = Get.put(SignUpController());
    GlobalKey<FormState> _loginFormKey = GlobalKey<FormState>();

    return MaterialApp(
        home: Scaffold(
            resizeToAvoidBottomInset: false,
            body: Stack(children: <Widget>[
              Container(
                decoration: const BoxDecoration(
                    color: Colors.black,
                    image: DecorationImage(
                        image: AssetImage("assets/images/bgImgtest.png"),
                        fit: BoxFit.fill)),
                child: Scaffold(
                    backgroundColor: Colors.transparent,
                    body: Stack(
                      children: [
                        Container(
                          child: SingleChildScrollView(
                            child: SafeArea(
                              child: Column(
                                children: [
                                  Image.asset(
                                    "assets/images/logo.png",
                                    width: 390,
                                    height: 350,
                                  ),
```

,

```
Form(
  key: _loginFormKey,
  child: Column(
    children: <Widget>[
      Container(
          alignment: Alignment.centerLeft,
          decoration: kBoxDecorationStyle,
          width: 350,
          height: 60,
          child: TextFormField(
              controller: controller.email,
              keyboardType:
                  TextInputType.emailAddress,
              style: const TextStyle(
                  color: Colors.white),
              decoration: InputDecoration(
                  border: InputBorder.none,
                  contentPadding:
                      const EdgeInsets.only(
                          top: 14.0),
                  prefixIcon: const Icon(
                    Icons.people,
                    color: Colors.white,
                  ),
                  hintText:
                      'Enter your email',
                  hintStyle: kHintTextStyle),
              validator: (value) {
                if (value == null ||
                    value.isEmpty) {
                  return 'Please enter your email';
                }
                return null;
              })),
      const SizedBox(
        height: 10,
      ),
      Container(
          alignment: Alignment.centerLeft,
          decoration: kBoxDecorationStyle,
          width: 350,
          height: 60,
          child: TextFormField(
```

,

```
                controller: controller.password,
                obscureText: true,
                style: const TextStyle(
                    color: Colors.white),
                decoration: InputDecoration(
                    border: InputBorder.none,
                    contentPadding:
                        const EdgeInsets.only(
                            top: 14.0),
                    prefixIcon: const Icon(
                      Icons.lock,
                      color: Colors.white,
                    ),
                    hintText:
                        'Enter your password',
                    hintStyle: kHintTextStyle),
                validator: (value) {
                  if (value == null ||
                      value.isEmpty) {
                    return 'Please enter your password';
                  }
                  return null;
                })),
            const SizedBox(
              height: 5,
            ),
            Container(
              padding: const EdgeInsets.symmetric(
                  vertical: 25.0),
              width: 300,
              child: ElevatedButton(
                onPressed: () async {
                  if (_loginFormKey.currentState!
                      .validate()) {
                    try {
                      bool success =
                          await SignUpController
                              .instance
                              .loginUser(
                        controller.email.text
                            .trim(),
                        controller.password.text
                            .trim(),
```

,

```
                );

                if (success) {
                  Fluttertoast.showToast(
                    msg:
                        "Logged in successfully",
                    toastLength:
                        Toast.LENGTH_LONG,
                    gravity:
                        ToastGravity.BOTTOM,
                    backgroundColor:
                        Colors.green,
                    textColor: Colors.white,
                    fontSize: 16.0,
                  );
                }
              } on SignInWithEmailAndPasswordFailure catch (e) {
                Fluttertoast.showToast(
                  msg: e.message,
                  toastLength:
                      Toast.LENGTH_LONG,
                  gravity:
                      ToastGravity.BOTTOM,
                  backgroundColor: Colors.red,
                  textColor: Colors.white,
                  fontSize: 16.0,
                );
              } catch (e) {
                Fluttertoast.showToast(
                  msg:
                      "An unknown error occurred",
                  toastLength:
                      Toast.LENGTH_LONG,
                  gravity:
                      ToastGravity.BOTTOM,
                  backgroundColor: Colors.red,
                  textColor: Colors.white,
                  fontSize: 16.0,
                );
              }
            }
          },
          style: ButtonStyle(
```

,

```
                backgroundColor:
                    MaterialStateProperty.all(
                        const Color.fromARGB(
                            235, 250, 230, 185)),
                padding:
                    MaterialStateProperty.all(
                        const EdgeInsets.all(20)),
              ),
              child: const Text(
                'LOGIN',
                style: TextStyle(
                  color: Colors.black,
                  letterSpacing: 1.5,
                  fontSize: 18.0,
                  fontWeight: FontWeight.bold,
                  fontFamily: 'OpenSans',
                ),
              ),
            ),
          ),
          GestureDetector(
            onTap: () => {
              Navigator.pushReplacement(
                  context,
                  MaterialPageRoute(
                      builder: (context) =>
                          const SignupScreen()))
            },
            child: RichText(
              text: const TextSpan(
                children: [
                  TextSpan(
                    text:
                        'Don\'t have an Account? ',
                    style: TextStyle(
                      color: Colors.white,
                      fontSize: 12.0,
                      fontWeight: FontWeight.w400,
                    ),
                  ),
                  TextSpan(
                    text: 'Sign Up',
                    style: TextStyle(
```

,

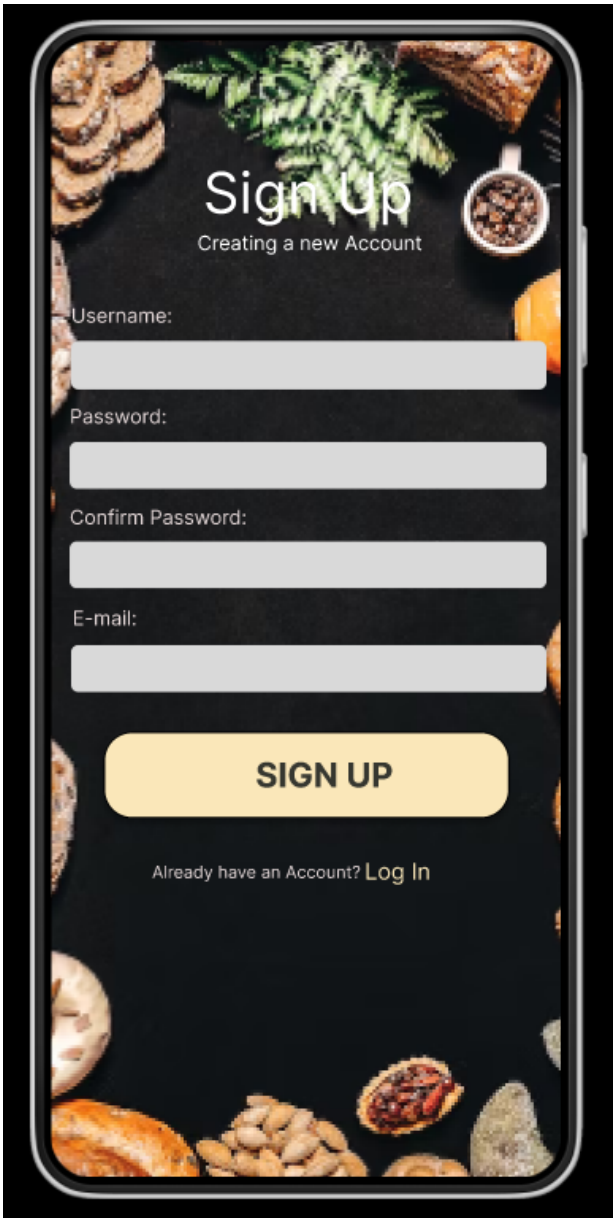
```
                                color: Colors.white,
                                fontSize: 12.0,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
                          ],
                        ),
                      ),
                    ),
                    const SizedBox(height: 50),
                    GestureDetector(
                      onTap: () => {
                        Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) =>
                                    const forgotPassScreen()))
                      },
                      child: RichText(
                        text: const TextSpan(
                          children: [
                            TextSpan(
                              text: 'Forgot Password',
                              style: TextStyle(
                                color: Colors.white,
                                fontSize: 12.0,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
                          ],
                        ),
                      ),
                    ),
                  ],
                ),
              ),
            ),
          ],
        )),
```

,

```
              )
            ])));
  }
}

final kBoxDecorationStyle = BoxDecoration(
  color: Colors.black12,
  borderRadius: BorderRadius.circular(15.0),
  boxShadow: [
    const BoxShadow(
      blurRadius: 2.0,
      offset: Offset(0, 2),
    ),
  ],
);

final kLabelStyle = const TextStyle(
  color: Colors.white,
  fontWeight: FontWeight.bold,
  fontFamily: 'OpenSans',
);

final kHintTextStyle = const TextStyle(
  color: Colors.white54,
  fontFamily: 'OpenSans',
);
```

Sign-Up Page

,




```
import 'package:flutter/material.dart';
import 'package:splashscreen/authenticationRepository.dart';
import 'package:splashscreen/controllers/signupController.dart';
import 'package:splashscreen/signupWithEmailAndPasswordFailure.dart';
import './LoginScreen.dart';
import 'package:get/get.dart';
```

,

```
import 'package:fluttertoast/fluttertoast.dart';

class SignupScreen extends StatefulWidget {
  const SignupScreen({Key? key}) : super(key: key);

  @override
  _SignupState createState() => _SignupState();
}

class _SignupState extends State<SignupScreen> {
  String? _password;
  String? _confirmPassword;
  @override
  Widget build(BuildContext context) {
    final controller = Get.put(SignUpController());
    final _signupFormKey = GlobalKey<FormState>();

    return MaterialApp(
        home: Scaffold(
            resizeToAvoidBottomInset: false,
            body: Stack(children: <Widget>[
              Container(
                decoration: BoxDecoration(
                    color: Colors.black,
                    image: DecorationImage(
                        image: AssetImage("assets/images/bgImgtest.png"),
                        fit: BoxFit.fill)),
                child: Scaffold(
                    backgroundColor: Colors.transparent,
                    body: Stack(
                      children: [
                        Container(
                          child: SingleChildScrollView(
                            child: SafeArea(
                              child: Column(
                                children: [
                                  Form(
                                    key: _signupFormKey,
                                    child: Column(
                                      children: [
                                        SizedBox(
                                          height: 100,
                                        ),
```

,

```
Container(
  alignment: Alignment.center,
  child: Text("Sign Up",
      style: TextStyle(
        color: Colors.white,
        fontSize: 40,
        fontWeight: FontWeight.w400,
      )),
),
SizedBox(
  height: 5,
),
Container(
    alignment: Alignment.center,
    child: Text(
        "Creating a new Account",
        style: TextStyle(
          color: Colors.white,
          fontSize: 14,
          fontWeight: FontWeight.w400,
        ))),
SizedBox(
  height: 50,
),
Container(
    alignment: Alignment.centerLeft,
    decoration: kBoxDecorationStyle,
    width: 350,
    height: 60,
    child: TextFormField(
        controller: controller.username,
        keyboardType:
            TextInputType.text,
        style: TextStyle(
            color: Colors.white),
        decoration: InputDecoration(
            border: InputBorder.none,
            contentPadding:
                EdgeInsets.only(
                    top: 14.0),
            prefixIcon: Icon(
              Icons.people,
              color: Colors.white,
```

,

```
            ),
            hintText:
                'Enter your username',
            hintStyle:
                kHintTextStyle))),
SizedBox(
  height: 7,
),
Container(
    alignment: Alignment.centerLeft,
    decoration: kBoxDecorationStyle,
    width: 350,
    height: 60,
    child: TextFormField(
      controller: controller.password,
      keyboardType: TextInputType.text,
      style: TextStyle(
          color: Colors.white),
      decoration: InputDecoration(
          border: InputBorder.none,
          contentPadding:
              EdgeInsets.only(
                  top: 14.0),
          prefixIcon: Icon(
            Icons.lock,
            color: Colors.white,
          ),
          hintText:
              'Enter your password',
          hintStyle: kHintTextStyle),
      validator: (val) {
        _password = val;
        if (val!.isEmpty) {
          return 'Please enter a password';
        } else if (val.length < 8) {
          return 'Password should be at least 8 characters';
        } else if (!val.contains(
            RegExp(r'[0-9]'))) {
          return 'Password should contain at least one number';
        } else if (!val.contains(
            RegExp(r'[A-Z]'))) {
          return 'Password should contain at least one uppercase letter';
        } else if (!val.contains(RegExp(
```

,

```
                r'[!@#$%^&*(),.?":{}|<>]'))) {
              return 'Password should contain at least one special character';
            }
            return null;
          },
        )),
    SizedBox(
      height: 7,
    ),
    Container(
        alignment: Alignment.centerLeft,
        decoration: kBoxDecorationStyle,
        width: 350,
        height: 60,
        child: TextFormField(
          controller:
              controller.confirmPassword,
          keyboardType:
              TextInputType.emailAddress,
          style: TextStyle(
              color: Colors.white),
          decoration: InputDecoration(
              border: InputBorder.none,
              contentPadding:
                  EdgeInsets.only(
                      top: 14.0),
              prefixIcon: Icon(
                Icons.lock,
                color: Colors.white,
              ),
              hintText:
                  'Confirm your password',
              hintStyle: kHintTextStyle),
          validator: (val) {
            _confirmPassword = val;
            if (val!.isEmpty) {
              return 'Please confirm your password';
            } else if (_password !=
                _confirmPassword) {
              return 'Passwords do not match';
            }
            return null;
          },
```

,

```
    )),
SizedBox(
  height: 7,
),
Container(
    alignment: Alignment.centerLeft,
    decoration: kBoxDecorationStyle,
    width: 350,
    height: 60,
    child: TextFormField(
        controller: controller.email,
        keyboardType:
            TextInputType.emailAddress,
        style: TextStyle(
            color: Colors.white),
        decoration: InputDecoration(
            border: InputBorder.none,
            contentPadding:
                EdgeInsets.only(
                    top: 14.0),
            prefixIcon: Icon(
              Icons.email,
              color: Colors.white,
            ),
            hintText:
                'Enter your email',
            hintStyle:
                kHintTextStyle))),
Container(
  padding: EdgeInsets.symmetric(
      vertical: 25.0),
  width: 300,
  child: ElevatedButton(
    onPressed: () async {
      if (_signupFormKey.currentState!
          .validate()) {
        // check if password and confirm password match
        if (controller.password.text !=
            controller
                .confirmPassword.text) {
          Fluttertoast.showToast(
            msg:
                "Password and Confirm Password doesn't match.",
```

,

```
          toastLength:
              Toast.LENGTH_LONG,
          gravity:
              ToastGravity.BOTTOM,
          backgroundColor: Colors.red,
          textColor: Colors.white,
          fontSize: 16.0,
        );
      } else {
        try {
          await SignUpController
              .instance
              .registerUser(
            controller.email.text
                .trim(),
            controller.password.text
                .trim(),
            controller.username.text
                .trim(),
          );

          Fluttertoast.showToast(
            msg:
                "Account created successfully",
            toastLength:
                Toast.LENGTH_LONG,
            gravity:
                ToastGravity.BOTTOM,
            backgroundColor:
                Colors.green,
            textColor: Colors.white,
            fontSize: 16.0,
          );
        } on SignUpWithEmailAndPasswordFailure catch (e) {
          Fluttertoast.showToast(
            msg:
                "Username already exists.",
            toastLength:
                Toast.LENGTH_LONG,
            gravity:
                ToastGravity.BOTTOM,
            backgroundColor:
                Colors.red,
```

,

```
                textColor: Colors.white,
                fontSize: 16.0,
              );
            } on UsernameAlreadyExistsException catch (e) {
              Fluttertoast.showToast(
                msg: e.message,
                toastLength:
                    Toast.LENGTH_LONG,
                gravity:
                    ToastGravity.BOTTOM,
                backgroundColor:
                    Colors.red,
                textColor: Colors.white,
                fontSize: 16.0,
              );
            } catch (e) {
              Fluttertoast.showToast(
                msg:
                    "An unknown error occurred",
                toastLength:
                    Toast.LENGTH_LONG,
                gravity:
                    ToastGravity.BOTTOM,
                backgroundColor:
                    Colors.red,
                textColor: Colors.white,
                fontSize: 16.0,
              );
            }
          }
        }
      },
      style: ButtonStyle(
        backgroundColor:
            MaterialStateProperty.all(
                Color.fromARGB(
                    235, 250, 230, 185)),
        padding:
            MaterialStateProperty.all(
                const EdgeInsets.all(20)),
      ),
      child: Text(
        'SIGN UP',
```

,

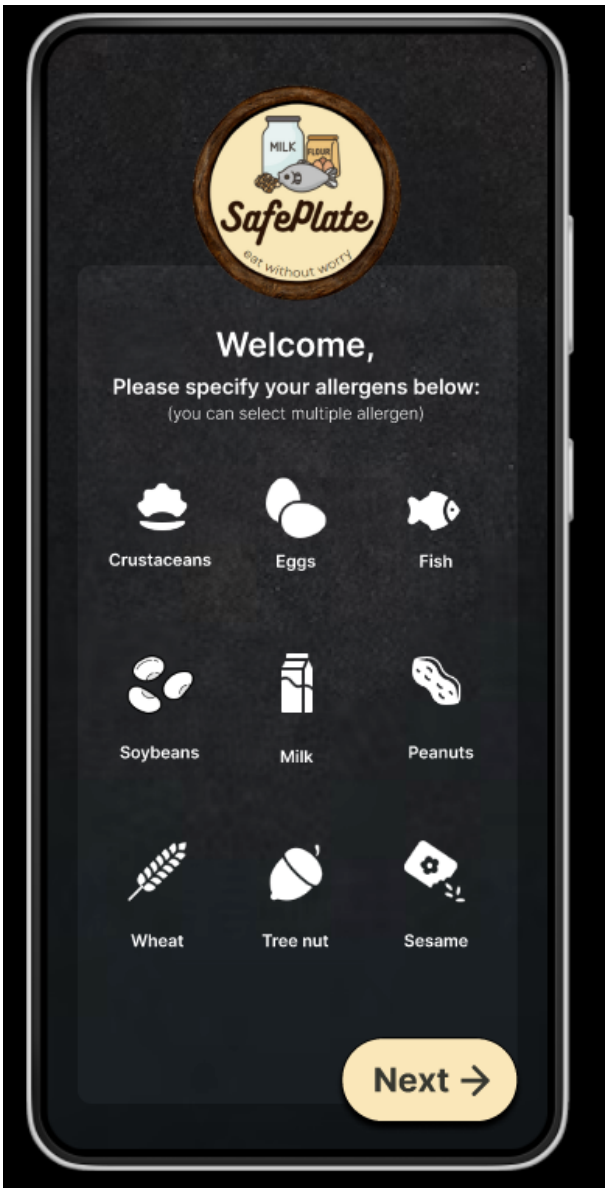
```
                    style: TextStyle(
                      color: Colors.black,
                      letterSpacing: 1.5,
                      fontSize: 18.0,
                      fontWeight: FontWeight.bold,
                      fontFamily: 'OpenSans',
                    ),
                  ),
                ),
              ),
              GestureDetector(
                onTap: () => {
                  Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) =>
                              LoginScreen()))
                },
                child: RichText(
                  text: TextSpan(
                    children: [
                      TextSpan(
                        text:
                            'Already Have an account? ',
                        style: TextStyle(
                          color: Colors.white,
                          fontSize: 12.0,
                          fontWeight: FontWeight.w400,
                        ),
                      ),
                      TextSpan(
                        text: 'Log in',
                        style: TextStyle(
                          color: Colors.white,
                          fontSize: 12.0,
                          fontWeight: FontWeight.bold,
                        ),
                      ),
                    ],
                  ),
                ),
              ),
            ],
```

,

```
                                ),
                              ),
                            ],
                          ),
                        ),
                      ),
                    ),
                  ],
                )),
            )
          ])));
  }
}
```

Allergen Option

,




```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:splashscreen/HomeScreen.dart';
import './LoginScreen.dart';
```

,

```
class allergenScreen extends StatefulWidget {
  const allergenScreen({Key? key}) : super(key: key);

  @override
  _allergenState createState() => _allergenState();
}

class _allergenState extends State<allergenScreen> {
  final double logoHeight = 144;
  late List<bool> selectedItems;
  late List<String> selectedAllergens;

  final images = [
    'assets/images/shell.png',
    'assets/images/eggs.png',
    'assets/images/fish.png',
    'assets/images/soybeans.png',
    'assets/images/milk.png',
    'assets/images/peanut.png',
    'assets/images/wheat.png',
    'assets/images/nut.png',
    'assets/images/sesame.png',
  ];

  final imageTexts = [
    'Shell',
    'Eggs',
    'Fish',
    'Soybeans',
    'Milk',
    'Peanut',
    'Wheat',
    'Nut',
    'Sesame',
  ];

  @override
  void initState() {
    super.initState();
    selectedItems = List.generate(images.length, (index) => false);
    selectedAllergens = [];
  }
```

,

```
@override
Widget build(BuildContext context) {
  return MaterialApp(
      home: Scaffold(
          resizeToAvoidBottomInset: false,
          body: Stack(
              clipBehavior: Clip.none,
              alignment: Alignment.center,
              children: <Widget>[
                Container(
                    decoration: const BoxDecoration(
                        color: Colors.black,
                        image: DecorationImage(
                            image: AssetImage("assets/images/bgImg2.png"),
                            fit: BoxFit.fill)),
                    child: Scaffold(
                        backgroundColor: Colors.transparent,
                        body: Stack(children: [
                          Container(
                              alignment: Alignment.center,
                              child: SingleChildScrollView(
                                  child: Stack(
                                      alignment: Alignment.topCenter,
                                      children: [
                                    Positioned(
                                      child: ingredientBg(),
                                    ),
                                    Positioned(
                                      top: 20,
                                      child: logoImage(),
                                    ),
                                    Positioned(
                                      top: 700,
                                      left: 212,
                                      child: addButton(
                                          'Next', selectedAllergens),
                                    ),
                                  ])))
                        ])))
              ]));
}

Widget logoImage() => Container(
```

,

```
      height: 180,
      width: 300,
      child: Image.asset(
        "assets/images/logofit.png",
      ),
    );

Widget ingredientBg() => Container(
    height: 850,
    width: 400,
    child: Container(
      decoration: const BoxDecoration(
          image: DecorationImage(
        image: AssetImage("assets/images/ingredientsBg.png"),
      )),
      child: Scaffold(
        backgroundColor: Colors.transparent,
        body: Stack(children: [
          Container(
              alignment: Alignment.center,
              child: SafeArea(
                child: Column(children: <Widget>[
              const SizedBox(
                height: 180,
              ),
              Container(
                  alignment: Alignment.center,
                  child: const Text('Welcome,',
                      style: TextStyle(
                          color: Colors.white,
                          fontSize: 24,
                          fontWeight: FontWeight.w600))),
              Container(
                alignment: Alignment.center,
                child: const Text("Please specify your allergens below: ",
                    style: TextStyle(
                      color: Colors.white,
                      fontSize: 15,
                      fontWeight: FontWeight.w600,
                    )),
              ),
              Container(
                alignment: Alignment.center,
```

,

```
  child: const Text("(you can select multiple allergen)",
      style: TextStyle(
        color: Colors.white,
        fontSize: 12,
        fontWeight: FontWeight.w400,
      )),
),
Padding(
  padding: const EdgeInsets.fromLTRB(5, 0, 5, 0),
  child: SizedBox(
    height: 400, // constrain height
    child: Padding(
      padding: const EdgeInsets.all(16.0),
      child: GridView.count(
        crossAxisSpacing: 1,
        crossAxisCount: 3,
        children: List.generate(images.length, (index) {
          return GestureDetector(
            onTap: () {
              setState(() {
                // Toggle selection status when item is tapped
                selectedItems[index] = !selectedItems[index];
                if (selectedItems[index]) {
                  selectedAllergens.add(imageTexts[index]);
                } else {
                  selectedAllergens.remove(imageTexts[index]);
                }
              });
            },
            child: Container(
              padding: const EdgeInsets.all(
                  17), // Adjust the padding here
              child: Container(
                decoration: BoxDecoration(
                  color: selectedItems[index]
                      ? Colors.grey[600]
                      : Colors.transparent,
                  // Add this for rounded corners
                  borderRadius: BorderRadius.circular(50),
                ),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
```

,

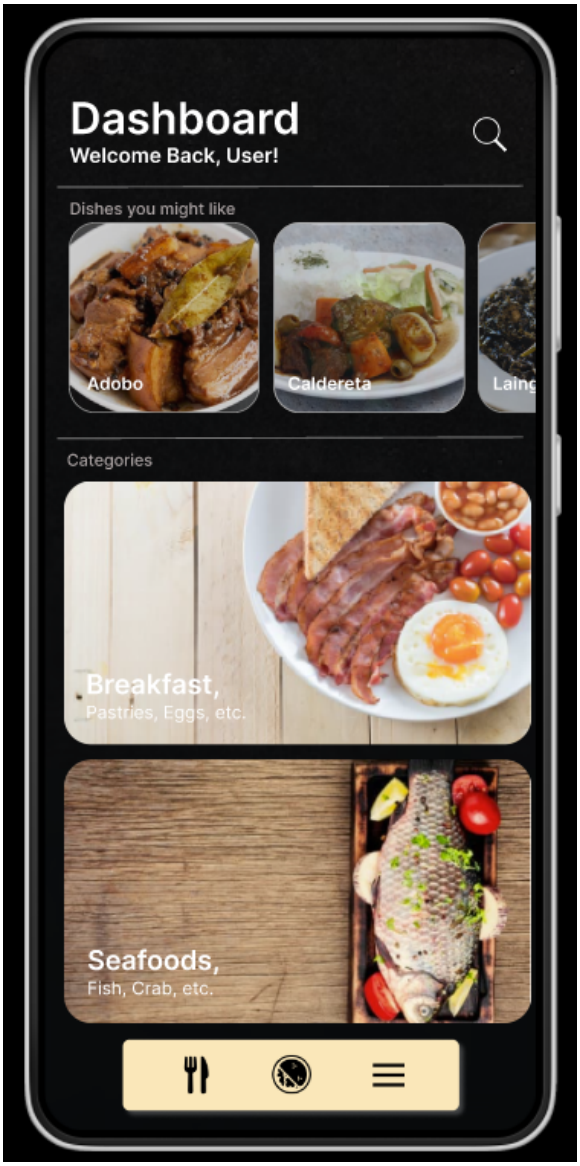
```
                                Image.asset(
                                  images[index],
                                  fit: BoxFit.cover,
                                ),
                                const SizedBox(height: 5),
                                Text(imageTexts[index],
                                    style: const TextStyle(
                                        color: Colors.white,
                                        fontSize: 14,
                                        fontWeight: FontWeight.bold)),
                              ],
                            ),
                          ),
                        ),
                      );
                    }),
                  ),
                ),
              ),
            )
          ])))
        ]),
      ),
    ));
}

Widget addButton(String btnText, List<String> selectedAllergens) {
  return Container(
    width: 140,
    height: 45,
    child: ElevatedButton(
      style: ElevatedButton.styleFrom(
        shape:
            RoundedRectangleBorder(borderRadius: BorderRadius.circular(20.0)),
        backgroundColor: const Color.fromARGB(
            235, 250, 230, 185), // background (button) color
      ),
      onPressed: () async {
        final String? currentUserId = FirebaseAuth.instance.currentUser?.uid;

        if (currentUserId == null) {
          print('Error: User not logged in');
          return;
```

,

```
      }

      await FirebaseFirestore.instance
          .collection('users')
          .doc(currentUserId)
          .set({'allergen': selectedAllergens}, SetOptions(merge: true));

      Get.offAll(() => const HomeScreen());
    },
    child: Row(
      children: [
        Text(btnText,
            style: const TextStyle(fontSize: 20, color: Colors.black)),
        const Spacer(),
        const Icon(
          // <-- Icon
          Icons.east,
          color: Colors.black,
          size: 24.0,
        ),
      ],
    ),
  ),
 );
}
```

Dashboard Screen / Home Screen

,




```
import 'package:flutter/material.dart';
import 'package:splashscreen/ProfileScreen.dart';
import 'package:splashscreen/dummy.dart';
import 'package:splashscreen/BreakfastScreen.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'dart:convert';
```

,

```
import 'package:http/http.dart' as http;

void main() {
  runApp(const HomeScreen());
}

class HomeScreen extends StatefulWidget {
  const HomeScreen({Key? key}) : super(key: key);

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class Category {
  final String imagePath;
  final String text1;
  final String text2;
  final Widget screen;

  Category(this.imagePath, this.text1, this.text2, this.screen);
}

final List<Category> categories = [
  Category('assets/images/breakfast.png', 'Breakfast,', 'Pastries, Eggs, etc.',
      const BreakfastScreen()),
  Category('assets/images/seafoods.png', 'Seafood,', 'Fish, Crab, etc.',
      const ProfileScreen()),
  Category('assets/images/meat.png', 'Meat,', 'Beef, Lamb, etc.',
      const YourNewScreen()),
  Category('assets/images/chicken.png', 'Chicken,', 'Poultry',
      const ProfileScreen()),
  Category('assets/images/pork.png', 'Pork,', 'Pig', const ProfileScreen()),
];

class _HomeScreenState extends State<HomeScreen> {
  void _onItemTapped(int index) {
    switch (index) {
      case 0:
        Navigator.push(
          context,
          MaterialPageRoute(builder: (context) => const ProfileScreen()),
        );
        break;
```

,

```
    case 1:
      Navigator.push(
        context,
        MaterialPageRoute(builder: (context) => const ProfileScreen()),
      );
      break;
    case 2:
      Navigator.push(
        context,
        MaterialPageRoute(builder: (context) => const ProfileScreen()),
      );
      break;
  }
}

void fetchData() async {
  List<Map<String, dynamic>> recipesData =
      await fetchRecipesWithDetailsAndRatings();

  // Print only the recipe name and the rating for each recipe
  for (Map<String, dynamic> recipeData in recipesData) {
    print('Recipe Name: ${recipeData['recipeName']}');
  }

  // List<Map<String, dynamic>> usersData = await fetchUsersWithClickHistory();
  // print('Users Data: $usersData');
}

@override
Widget build(BuildContext context) {
  String imageUrl;
  return MaterialApp(
    home: Scaffold(
      body: Container(
        decoration: const BoxDecoration(
          image: DecorationImage(
            image: AssetImage("assets/images/bgImg1.jpg"),
            fit: BoxFit.cover,
          ),
        ),
        child: Column(
          // use Column here
          children: [
```

,

```
            Expanded(
              flex: 2,
              child: Padding(
                padding:
                    const EdgeInsets.only(top: 70.0, left: 30.0, right: 10.0),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.start,
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    const Column(
                      crossAxisAlignment: CrossAxisAlignment
                          .start, // Align texts to the left side
                      children: [
                        Text(
                          "Dashboard",
                          style: TextStyle(
                            fontSize: 32,
                            fontWeight: FontWeight.w600,
                            color: Colors.white,
                          ),
                        ),
                        Text(
                          "Welcome Back, User!", // You can replace "User" with the actual user's name
from the db
                          style: TextStyle(
                            fontSize: 16,
                            fontWeight: FontWeight.w600,
                            color: Colors.white,
                          ),
                        ),
                      ],
                    ),
                    const Spacer(), // Takes up remaining space between Column and IconButton
                    IconButton(
                      icon: const Icon(Icons.search),
                      color: Colors.white,
                      iconSize: 50, // adjust the icon size
                      onPressed: () {
                        // Handle the search action here
                        //fetchData();
                        sendDatasetsToFlask();
                      },
                    ),
```

,

```
        ],
      ),
    ),
  ),
  Container(
    width: 350, // Adjust this to control the width of the divider
    child: const Divider(
      color: Colors.white,
      thickness: 0.5,
    ),
  ),
  Expanded(
    flex: 2,
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        const Padding(
          padding: EdgeInsets.only(left: 8.0),
          child: Text(
            'Dishes you might like',
            style: TextStyle(
              fontSize: 16,
              fontWeight: FontWeight.w700,
              color: Colors.white,
            ),
          ),
        ),
        Expanded(
          child: StreamBuilder<QuerySnapshot>(
            stream: FirebaseFirestore.instance
                .collection('recipe')
                .snapshots(),
            builder: (BuildContext context,
                AsyncSnapshot<QuerySnapshot> snapshot) {
              if (snapshot.hasError) {
                return Text('Something went wrong');
              }

              if (snapshot.connectionState ==
                  ConnectionState.waiting) {
                return Text("Loading");
              }
```

,

```
                return ListView.builder(
                  scrollDirection: Axis.horizontal,
                  itemCount: snapshot.data!.docs.length,
                  itemBuilder: (BuildContext context, int index) {
                    var document = snapshot.data!.docs[index];

                    try {
                      imageUrl = document.get('image');
                    } catch (e) {
                      print(
                          'The image field does not exist in the document');
                      imageUrl =
                          'default_image_url'; // Replace with your default image URL
                    }

                    return Container(
                      width: 160.0,
                      margin: const EdgeInsets.all(10.0),
                      child: Image.network(imageUrl),
                    );
                  },
                );
              },
            ),
          ),
        ],
      ),
    ),
    Container(
      width: 350, // Adjust this to control the width of the divider
      child: const Divider(
        color: Colors.white,
        thickness: 0.5,
      ),
    ),
    Expanded(
      flex: 4,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          const Padding(
            padding: EdgeInsets.only(left: 10.0),
            child: Text(
```

,

```
                    'Categories',
                    style: TextStyle(
                      fontSize: 16,
                      fontWeight: FontWeight.w700,
                      color: Colors.white,
                    ),
                  ),
                ),
                Expanded(
                  child: ListView.builder(
                    itemCount: categories.length,
                    itemBuilder: (BuildContext context, int index) {
                      return Padding(
                        padding: const EdgeInsets.fromLTRB(30, 0, 0, 10),
                        child: GestureDetector(
                          onTap: () {
                            Navigator.push(
                              context,
                              MaterialPageRoute(
                                builder: (context) =>
                                    categories[index].screen,
                              ),
                            );
                          },
                          child: Column(
                            children: [
                              AspectRatio(
                                aspectRatio: 2,
                                child: ClipRRect(
                                  borderRadius: BorderRadius.circular(10.0),
                                  child: Stack(
                                    children: <Widget>[
                                      Image.asset(
                                        categories[index].imagePath,
                                        fit: BoxFit.cover,
                                      ),
                                      Padding(
                                        padding: const EdgeInsets.all(10.0),
                                        child: Column(
                                          mainAxisAlignment:
                                              MainAxisAlignment.end,
                                          crossAxisAlignment:
                                              CrossAxisAlignment.start,
```

,

```
                                children: <Widget>[
                                  Text(
                                    categories[index].text1,
                                    style: const TextStyle(
                                      color: Colors.white,
                                      fontSize: 15,
                                      fontWeight: FontWeight.w700,
                                    ),
                                  ),
                                  Text(
                                    categories[index].text2,
                                    style: const TextStyle(
                                      color: Colors.white,
                                      fontSize: 15,
                                      fontWeight: FontWeight.w700,
                                    ),
                                  ),
                                ],
                              ),
                            ),
                          ],
                        ),
                      ),
                    ),
                  ],
                ),
              ),
            );
          },
        ),
      ),
    ],
  ),
),
Positioned(
  left: 20.0,
  right: 20.0,
  bottom: 10.0,
  child: ClipRRect(
    borderRadius: BorderRadius.circular(30.0),
    child: BottomNavigationBar(
      items: const <BottomNavigationBarItem>[
        BottomNavigationBarItem(
```

,

```
                  icon: Icon(Icons.restaurant),
                  label: 'Food',
                ),
                BottomNavigationBarItem(
                  icon: Icon(Icons.menu_book),
                  label: 'Plate',
                ),
                BottomNavigationBarItem(
                  icon: Icon(Icons.menu),
                  label: 'Menu',
                ),
              ],
              selectedItemColor: Colors.black,
              onTap: _onItemTapped,
              showSelectedLabels: false,
              showUnselectedLabels: false,
              backgroundColor: const Color.fromRGBO(250, 231, 185, 1),
            ),
          ),
        ),
      ],
    ),
  ),
 ),
 );
 }
}

Future<List<Map<String, dynamic>>> fetchRecipesWithDetailsAndRatings() async {
 // This list will hold the data we retrieve
 List<Map<String, dynamic>> data = [];

 // Fetch all recipes
 QuerySnapshot recipesSnapshot =
    await FirebaseFirestore.instance.collection('recipe').get();

 // Iterate over each recipe document
 for (QueryDocumentSnapshot recipe in recipesSnapshot.docs) {
  if (recipe.exists) {
    String recipeName = recipe.id;
    String calorie = recipe.get('calorie');
    String image = recipe.get('image');
    List<dynamic> ingredients = recipe.get('ingredients');
```

,

```
      String preparation_time = recipe.get('preparation_time');
      String type = recipe.get('type');

      QuerySnapshot ratingsSnapshot =
          await recipe.reference.collection('rating').get();

      for (QueryDocumentSnapshot rating in ratingsSnapshot.docs) {
        //Fetch the user ID and user rating
        String userID = rating.id;
        double userRating = rating.get('rating');

        // Add the data to the list
        data.add({
          'recipeName': recipeName,
          'calorie': calorie,
          'image': image,
          'ingredients': ingredients.join(', '),
          'preparation_time': preparation_time,
          'userID': userID,
          'rating': userRating,
        });
      }
    } else {
      print('Document does not exist on the database');
    }
  }

  return data;
}

// Future<List<Map<String, dynamic>>> fetchUsersWithClickHistory() async {
//   // This list will hold the data we retrieve
//   List<Map<String, dynamic>> data = [];

//   // Fetch all users
//   QuerySnapshot usersSnapshot =
//       await FirebaseFirestore.instance.collection('users').get();

//   // Iterate over each user document
//   for (QueryDocumentSnapshot user in usersSnapshot.docs) {
//     String userId = user.id;

//     // Fetch the click history documents for the current user
```

,

```
//     QuerySnapshot clickHistorySnapshot =
//         await user.reference.collection('click_history').get();

//     // Iterate over each click history document
//     for (QueryDocumentSnapshot clickHistory in clickHistorySnapshot.docs) {
//       // Fetch the click history details
//       String clickDetails = clickHistory.get('details');

//       // Add the data to the list
//       data.add({
//         'userId': userId,
//         'clickDetails': clickDetails,
//       });
//     }
//   }

//   return data;
// }

Future<void> sendDatasetsToFlask() async {
  final url = 'http://10.0.2.2:5000/recommend-recipes';
  final headers = {'Content-Type': 'application/json'};

  List<Map<String, dynamic>> recipesData =
      await fetchRecipesWithDetailsAndRatings();

  final body = jsonEncode({
    'recipesData': recipesData,
  });

  try {
    final response = await http
        .post(Uri.parse(url), headers: headers, body: body)
        .timeout(const Duration(seconds: 20));

    if (response.statusCode == 200) {
      final responseData = jsonDecode(response.body) as Map<String, dynamic>;
      final recipeRecommendations =
          responseData['recipeRecommendations'] as List<dynamic>;

      // Print the recipeRecommendations in the Flutter console
      print('Recipe Recommendations:');
      recipeRecommendations.forEach((recipeName) {
```

,

```
        print(recipeName);
      });
    } else {
      print('Failed to send datasets: ${response.statusCode}');
    }
  } catch (e) {
    print('Unknown Exception: $e');
  }
}
```

Respondents' Profile

In this project evaluation, we have identified two (2) categories of users as our respondents, which are the technical and non-technical users. The technical respondents were given a questionnaire that was validated by Computer Science students. The non-technical respondents were given a questionnaire answered by the black box testers who have a small amount of knowledge about the system and the project wholly.

,

Overall Rating of Conducted Survey

Technical Respondent

Sub Characteristics	Average	Descriptive Writing
Functionaility		
1. Suitability	4.9	The Technical respondesnts **Strongly Agreed** the the mobile application does perform well in terms of required tasks
2. Accurateness	4.9	The Technical respondents **Strongly Agreed** that the output were as expected
3. Interoperability	4.8	The Technical respondents **Strongly Agreed** that the mobile application can make or use of information effectively
Mean Score	**4.86**	

The table above shows the summary of the average rating that the technical respondents gave from the following criterion. The overall rating is 4.86 and it is interpreted as a **Strongly Agreed**, implying that the mobile application met the satisfactory performance.

,

Sub Characteristics	Average	Descriptive Writing
Usability		
1. Understandability	4.8	The Technical respondesnts **Strongly Agreed** the the mobile application can be easily to understand and can be fully used.
2. Learnability	4.6	The Technical respondents **Strongly Agreed** that the mobile application can be easily learned.
3. Operatability	4.9	The Technical respondents **Strongly Agreed** that the mobile application can be used without much effort.
Mean Score	**4.76**	

The table shows the summary of the average rating of technical respondents that responds to the Efficiency criterion. The overall average rating is 4.76 and implies as **Strongly Agreed**, making the system meet the expected performance.

,

Sub Characteristics	Average	Descriptive Writing
Efficiency		
1. Time Behavior	4.35	The Technical respondesnts **Strongly Agreed** the the mobile application can quickly respond.
2. Resource Utilization	4.44	The Technical respondents **Strongly Agreed** that the mobile application can use the information efficiently.
Mean Score	**4.39**	

The figure above shows that the average score the Technical respondent gave for the following criterion was 4.39, which implies that the mobile application met the intended efficient performance.

Technical Criteria Average

,

FACTORS	Average	Descriptive Writing
1. Functionality	4.86	The technical responders **Strongly Agreed** that the application function satisfies its targeted performance standards.
2. Usability	4.76	The technical responders **Strongly Agreed** that the application satisfies their expectations for ease of use and navigation.
3. Efficiency	4.39	The technical responders **Strongly Agreed** that the application satisfies their expectations in terms of efficiently utilizing its resources.
Total	4.67	Overall, Technical Respondents **Strongly Agreed** that mobile application meets the three (3) evaluation metrics

Non-Technical Respondent

Sub Characteristics	Average	Descriptive Writing
Functionaility		
1. Suitability	4.8	The Non-Technical respondesnts **Strongly Agreed** the the mobile application does perform well in terms of required tasks
2. Accurateness	4.9	The Non-Technical respondents **Strongly Agreed** that the output were as expected
3. Interoperability	4.7	The Non-Technical respondents **Strongly Agreed** that the mobile application can make or use of information effectively
Mean Score	**4.8**	

The table above shows the summary of the average rating that the Non-Technical respondents gave from the following criterion. The overall rating is 4.8 and it is interpreted as a **Strongly Agreed**, implying that the mobile application met the satisfactory performance.

,

Sub Characteristics	Average	Descriptive Writing
Usability		
1. Understandability	5	The Non-Technical respondesnts **Strongly Agreed** the the mobile application can be easily to understand and can be fully used.
2. Learnability	4.9	The Non-Technical respondents **Strongly Agreed** that the mobile application can be easily learned.
3. Operatability	4.9	The Non-Technical respondents **Strongly Agreed** that the mobile application can be used without much effort.
Mean Score	**4.93**	

The figure above shows that the average score the Non-Technical respondent gave for the following criterion was 4.93, which implies that the mobile application met the intended efficient performance.

Sub Characteristics	Average	Descriptive Writing
Efficiency		
1. Time Behavior	5	The Non-Technical respondesnts **Strongly Agreed** the the mobile application can quickly respond.
2. Resource Utilization	5	The Non-Technical respondents **Strongly Agreed** that the mobile application can use the information efficiently.
Mean Score	**5**	

The figure above shows that the average score the Non-Technical respondent gave for the following criterion was 5, which implies that the mobile application met the intended efficient performance.

,

Below are the detailed representation of the Technical and Non-Technical Respondents along with their corresponding questions shown in the table.

Non-Technical Criteria Average

FACTORS	Average	Descriptive Writing
1. Functionality	4.8	The non-technical responders **Strongly Agreed** that the application function satisfies its targeted performance standards.
2. Usability	4.93	The non-technical responders **Strongly Agreed** that the application satisfies their expectations for ease of use and navigation.
3. Efficiency	5	The non-technical responders **Strongly Agreed** that the application satisfies their expectations in terms of efficiently utilizing its resources.
Total	**4.91**	Overall, Non-Technical Respondents **Strongly Agreed** that mobile application meets the three (3) evaluation metrics

ISO Evaluation Table Data

Non-Technical Evaluation Table Data

,

CRITERIA	1	2	3	4	5
FUNCTIONALITY					
The application give results based on the user preferences.	-	-	-	-	-
The application successfully recommends food preference.	-	-	-	-	-
USBAILITY					
The application user interface is easy to navigate	-	-	-	-	-
The application is not complicated to use	-	-	-	-	-
The application has a well designed user interface.	-	-	-	-	-
EFFICIENCY					
The application has a quick load time	-	-	-	-	-
The application responds quickly to user interaction	-	-	-	-	-

Technical Evaluation Table Data

CRITERIA	1	2	3	4	5
FUNCTIONALITY					
The application give results based on the user preferences.	-	-	-	-	-
The application successfully recommends food substitution based form the chosen food allergen.	-	-	-	-	-
USBAILITY					
The application provides a clear function of buttons, fields and even color scheme assist.	-	-	-	-	-
The application function as intended	-	-	-	-	-
The application is simple to navigate and use	-	-	-	-	-
EFFICIENCY					
The application can quickly responds based on the action of the user	-	-	-	-	-
The application state is optimized	-	-	-	-	-

Summary of Evaluation for Technical and NOn-Technical Respondents

,

FACTORS	Average	Descriptive Writing
1. Functionality	4.83	The Technical and non-technical responders **Strongly Agreed** that the application function satisfies its targeted performance standards.
2. Usability	4.84	The Technical and non-technical responders **Strongly Agreed** that the application satisfies their expectations for ease of use and navigation.
3. Efficiency	4.69	The Technical and non-technical responders **Strongly Agreed** that the application satisfies their expectations in terms of efficiently utilizing its resources.
Total	**4.79**	Overall, the Technical and Non-Technical Respondents **Strongly Agreed** that mobile application meets the three (3) evaluation metrics

Model Development

Random Forest Algorithm

Chapter 5

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

This chapter focuses on the conclusion's summary findings and suggestions for correcting concerns uncovered during testing and evaluation.

,

Summary of the Study

A lot of people enjoy eating different meals a day since food is one of the most important things for us people to gain energy or even just to satisfy the cravings we have. Most of us have what we call food allergens, and food allergic reactions may be often mild to us people in the Philippines, sometimes they can be serious. What I mean 'serious' is to the point our allergy reaction progresses to a more severe life-threatening allergic reaction called anaphylaxis. Anaphylaxis and this causes constricted airways in the lungs and severe lowering of blood pressure and shock (anaphylactic shock). To help with this problem, the researchers conceptualize a way to help or assist people who suffer from food allergy on making or picking recipes for making their daily meals, by developing an application that make use of Random Forest Algorithm to train model and also to predict what food substitution is much more effective to use other than the food allergen presented on the recipe.

Conclusion

The application was capable of providing acceptable food dishes and alternatives for the selected food allergy. During the testing and assessment phase, the researchers concluded that the constructed system effectively met its objectives and scope.

1. A completely working system based on the Random Forest Algorithm, Collaborative Filtering and a Hybrid Approach has been created.
2. The trained model acquired an accuracy of 92% for Random Forest.
3. The application was created and built using the Random Forest Algorithm trained model in mind with the added two (2) more algorithms.

,

4. The application recommends the proper food substitution for the chosen allergen and also recommends a food recipe for its rating.
5. Based on the ISO-9125, the evaluation of the application has an average of ()() for the Technical Respondents and average of ()() for the Non-Technical Respondents, a total average of ()() which says that the respondents gain a satisfactory results and has shown that the application is functional, usable, and efficient.

In conclusion, a trained model based on the Random Forest Algorithm has created a completed evaluation and testing. The application allows the user to freely choose and determine if the food recipe is safe for them. A user evaluation based on the ISO-9126 regulated to test the whole system's performance and capabilities while also providing outstanding results. With all the objectives met, the researchers concluded that the primary goal of developing a personalized Allergen-Free Recipe Recommendations of Philippine Food have been met.

Recommendations

In future research, supervised learning methods will be used to train models for a recommender system. To reduce classification mistakes, it is advised that as many dataset samples as feasible be used. The study examined and applied the Random Forest Algorithm and the Collaborative Filtering Algorithm, as well as a hybrid of the two methods, yielding a result of 92% accuracy only for Random Forest algorithm, allowing the system to perform as predicted and perform efficiently as expected before implementing it as a usable application. In the systems participants or what we call users were not able to use the application in terms for people who have dietary restrictions or medical conditions such as having diabetes or high blood

,

pressure since the system does not include those, since only the nine (9)allergens are available to choose on the system and may not account for the other specific allergens.

References

Statista Research Department. (2021, July 15). *Philippines: Mobile phone users 2017-2025*. Statista. https://www.statista.com/forecasts/558756/number-of-mobile-internet-user-in-the-philippines

Taylor, P. (2023, March 30). *Mobile network subscriptions worldwide 2028*. Statista. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

'Moen, Ø., Opheim, E., & Trollvik, A. (2019). Parents Experiences Raising a Child with Food Allergy; A Qualitative Review.. Journal of pediatric nursing. https://doi.org/10.1016/j.pedn.2019.02.036'.

'Food recommender system based on weighted ingredients, body mass index and allergies; using the Random Forest algorithm' by Abel Martinez-Gorospe et. al.(2021)

Pelegrino, E. N. (n.d.). *Serious facts you need to know about food allergies*. National Nutrition Council.

,

https://nnc.gov.ph/regional-offices/mindanao/region-ix-zamboanga-peninsula/5597-serious-facts-you-need-to-know-about-food-allergies

‘Moen, Ø., Opheim, E., & Trollvik, A. (2019). Parents Experience Raising a Child with Food Allergy; A Qualitative Review.. Journal of pediatric nursing. https://doi.org/10.1016/j.pedn.2019.02.036’.

Pergamon. (2023, February 24). *A novel healthy and time-aware food recommender system using attributed community detection*. Expert Systems with Applications. https://www.sciencedirect.com/science/article/pii/S0957417423002208

Shirai, S. S., Seneviratne, O., Gordon, M. E., Chen, C.-H., & McGuinness, D. L. (2020, December 3). *Identifying ingredient substitutions using a knowledge graph of food*. Frontiers. https://www.frontiersin.org/articles/10.3389/frai.2020.621766/full

V. Garipelly, P. T. Adusumalli and P. Singh, "Travel Recommendation System Using Content and Collaborative Filtering - A Hybrid Approach," 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2021, pp. 1-4, doi: 10.1109/ICCCNT51525.2021.9579907.

H. Zacarias, G. Cangondo, L. Souza-Pereira, N. M. Garcia, B. Silva and N. Pombo, "Application of Content-Base Recommendation Algorithms on Mobile Travel Applications," 2023 1st International Conference on Advanced Innovations in Smart Cities (ICAISC), Jeddah, Saudi Arabia, 2023, pp. 1-5, doi: 10.1109/ICAISC56366.2023.10085680.

,

,