KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

KUMASI, GHANA

Developing a food donation and sharing mobile application to curb hunger crisis (SDG 2) and promote good health and well-being (SDG 3)

BY

Benjamin Ollennu (4628218)

Samuel Koduah Sarpong (4621118)

A thesis submitted to the Department of Computer Science,

in partial requirement for the award degree of

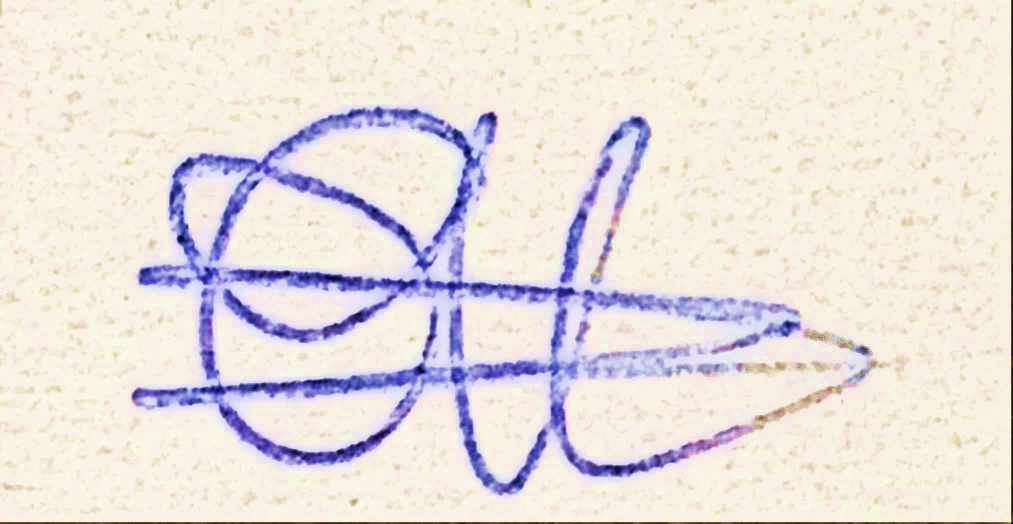
BACHELOR OF COMPUTER SCIENCE

October, 2022

# DECLARATION

We thus state that we prepared this project work in 2022 for the BSc in Award Of the degree at Kwame Nkrumah University of Science and Technology under the direction of Dr. Eric Osei Opoku, Department of Computer Science.

Furthermore, we certify that this project represents our own labor. and has not been submitted to another university for the purpose of conferring a degree or other credentials.

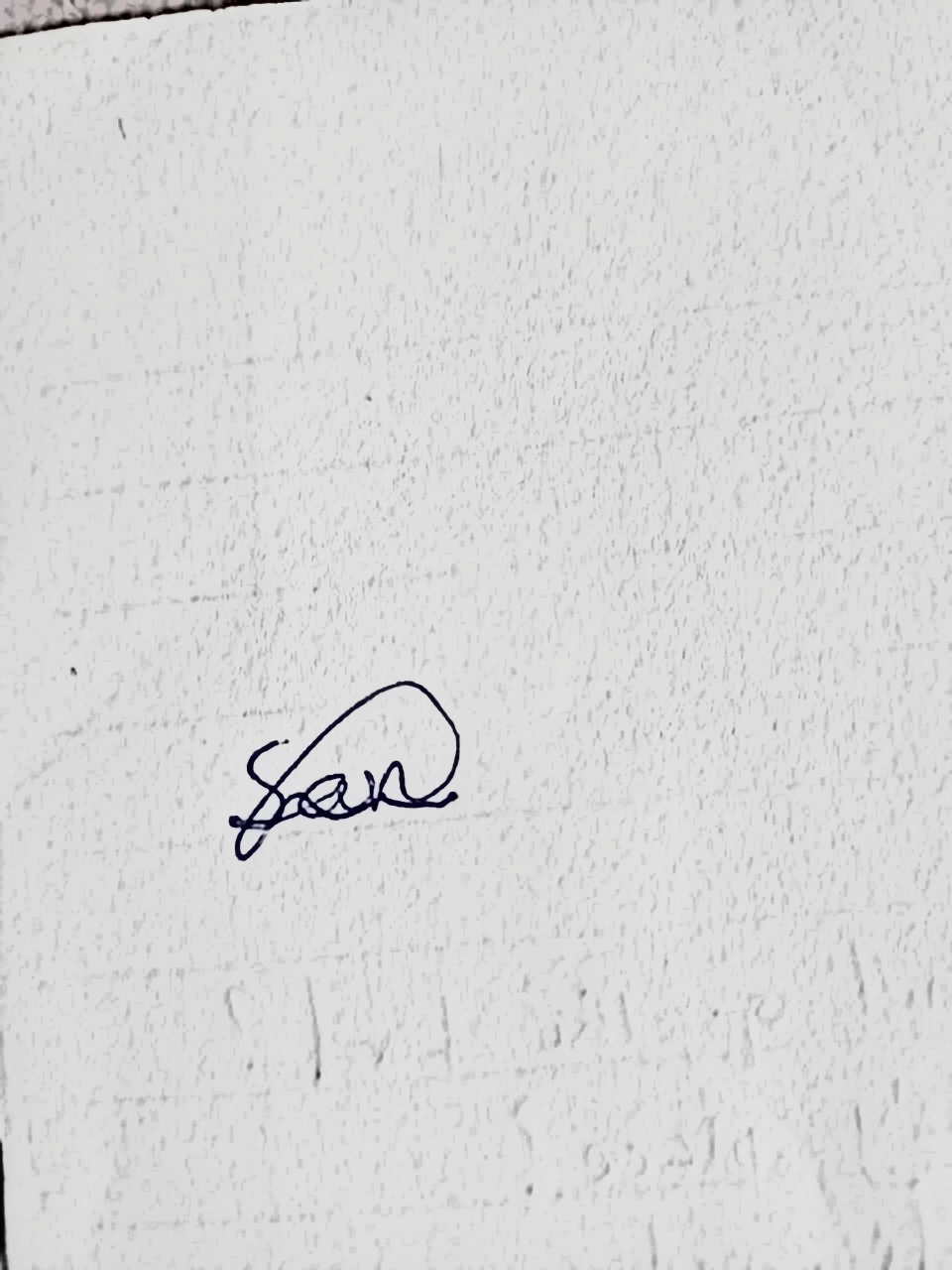


Ollennu Benjamin Larteh ……………

Signature

--------------------

(4628218) Date



--------------------

Samuel Koduah Sarpong ……………

Signature

(4621118) Date

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Certified by: |  |  |  |  |  |  |
| Dr. Eric Opoku Osei |  |  |  | …………………. |  | …………… |
| (Supervisor)      Certified by: |  |  |  | Signature |  | Date |
| Prof. Hayfron-Ben Acquah |  |  |  | …………………. |  | …………… |
| (Head of Department) |  |  |  | Signature |  | Date |

# DEDICATION

We dedicate this work to the All-Powerful God and to our project manager, Dr. Eric Opoku Osei, for his leadership and assistance during this research.

# ACKNOWLEDGEMENT

We give thanks to the All-Powerful God for His direction and favor throughout our effort. Additionally, we are very appreciative to Dr. Eric Opoku Osei, our supervisor, for his presence and the guidance he provided us with throughout the project. We are grateful.

# ABSTRACT

The world suffers from hunger, and it results in increased death counts. Many individuals are physically famished and begging for charity, despite the fact that many of them do not even receive two meals every day. In light of this, it is preferable to donate extra food to those who are in need rather than putting it to waste. Encouraging food donation and sharing in order to support SDG2 (end world hunger) and enhance wellbeing. The new system provides the feature to share leftover food with communities or individuals who are experiencing hunger, whether it is for free or at an affordable cost. The former systems offered fewer effective features. The new system includes key features such as indicating the number of nutrient-dense products and materials needed to produce meals and a chatroom to enhance user communication. Users and admins will both need to log in to customize their app experience.

# TABLE OF CONTENTS

**DECLARATION ............................................................................................................ii** **DEDICATION .............................................................................................................. iii** **ACKNOWLEDGEMENT ............................................................................................ iv** **ABSTRACT .................................................................................................................... v** **TABLE OF CONTENTS .............................................................................................. vi** **LIST OF TABLES ...................................................................................................... vii**

**LIST OF FIGURES ....................................................................................................... viii**

# CHAPTER ONE ............................................................................................................. 1 INTRODUCTION .......................................................................................................... 1

1.0 Introduction ................................................................................................................ 1

1.1 Background................................................................................................................. 1

1.2 Problem Statement…….............................................................................................. 1

1.3 Aim and Objectives.................................................................................................... 2

1.4 Scope of the Study ......................................................................................................2

1.5 Significance of the study ............................................................................................ 3

1.6 Limitations of the study .............................................................................................. 3

1.7 Organization of work ...................................................................................................3

# CHAPTER TWO ............................................................................................................ 4 LITERATURE REVIEW .............................................................................................. 4

2.1 Subject Matter Knowledge ......................................................................................... 4

2.1.1 Online Databse (Google Firebase) .......................................................................... 4

2.1.2 Android Operating System…… .............................................................................. 5

2.1.3 Key Takeaways…………………............................................................................. 5

2.1.4 Understanding the Android Operating System ........................................................ 5

2.1.5 Using Flutter for the Android Application ...............................................................6

2.2 Overview of Food Donation and Sharing Mobile Applications................................. 7

2.3 Review of Closely Related Papers …………............................................................ 9

2.4 Related Works Essay …………………………………………………................... 17

**CHAPTER THREE............................................................................................................. 34**

# METHODOLOGY AND SYSTEM DESIGN............................................................ 34

3.0 Analysis and Design ................................................................................................. 34

3.1 Requirements Gathering Methods ……................................................................... 34

3.2 Waterfall Methodology ……………........................................................................ 36

3.3 System Requirement…………….............................................................................. 38

3.4 Non-Functional Requirements .................................................................................. 39

# CHAPTER FOUR...............................................................................................................41 SYSTEM IMPLEMENTATION AND TESTING .................................................... 41

4.0 Introduction .............................................................................................................. 41

4.1 Testing of the New System ...................................................................................... 41

4.2 Functional Testing ………………………............................................................... 42

4.4 Implementation of the New System......................................................................... 44

4.5 System Documentation............................................................................................. 44

4.6 About the System…….............................................................................................. 44

4.6.1 User Access Level…….......................................................................................... 44

4.7 Conclusion................................................................................................................ 45

# CHAPTER FIVE ........................................................................................................ 46 CONCLUSION AND FUTURE WORK .................................................................. 46

5.0 Introduction ............................................................................................................ 46

5.1 Summary................................................................................................................. 46

5.2 Strength of the System............................................................................................ 46

5.3 Limitation (weakness) of the system ...................................................................... 46

5.4 Opportunities .......................................................................................................... 46

5.5 Threats .................................................................................................................... 47

5.6 Recommendations and Future Work....................................................................... 47

5.7 Conclusion............................................................................................................... 47

**REFERENCES ……............................................................................................................48** **APPENDIX ................................................................................................................. 50**

**LIST OF TABLES**

Table 1 Reviewing closely related papers ....................................................................... 9

**LIST OF FIGURES**

Fig. 1 Comparison between Firebase and Traditional Database........................................ 4

Fig. 2 Android Studio IDE................................................................................................. 7

Fig. 3 Food Related Application 1...................................................................................... 8

Fig. 4 Food Related Application 2 ..................................................................................... 9

Fig. 5 – Fig.15 Closely Related Works Images………………………………….….10 - 33

Fig. 16 – Waterfall methodology ..................................................................................... 38

Fig. 17 – Use Case Diagram ............................................................................................ 40

Fig. 18 –Startup Interface ................................................................................................ 42

Fig. 19 –Account Creation Interface................................................................................ 42

Fig. 20 –Sign in Page ....................................................................................................... 42

Fig. 21 –Create Account or Sign in Page ........................................................................ 42

Fig. 22 –User Side Bar ..................................................................................................... 43

Fig. 23 –Admin Page (Handle Reports) ........................................................................... 43

Fig. 24 –Admin Page (List of Users) ................................................................................ 43

Fig. 25 – Admin Page (Search Users) ............................................................................... 43

Fig. 26 – Offers Page ........................................................................................................ 43

Fig. 27 –Admin Page (Side Bar) ....................................................................................... 43

Fig. 28 – Admin Page (User Details) ................................................................................. 43

Fig. 29 – Records Page ..................................................................................................... 43

# CHAPTER ONE : INTRODUCTION

## 1.0 Introduction

This chapter largely focuses on the project's history, the various objectives the study aims to achieve, and the significance of the project. It provides an explanation of how the study is set up.

## 1.1 Background

The world faces a global hunger crisis of unprecedented proportions. Globally, 822 million people suffer undernourishment. Malnutrition and starvation claim the lives of 3.1 million children worldwide yearly. *(The World Counts, 2022)*. This is no small feat judging by the population*.* Every person must be able to know the components of what they ingest into their system. Online-ordering of food is the order of the day now because of the pandemic at hand.

Over half a million of people worldwide are living with extreme hunger. In countries like Nigeria, Syria, Yemen, India, Sudan etc. they have recorded the most cases of hunger crises. While some starving citizens get little to no amount of food, others sadly do not. It causes undernourishment and leads most of them to their deaths.

Since 2019, there have been an additional 135 million cases of acute food insecurity, bringing the total to 345 million. This means that up to 828 million individuals now experience nighttime hunger. 50 million people, spread across 45 different countries, are on the point of going hungry. *(World Food Programme, 2022).*

Over 99 million young children under the age of five are still underweight and undernourished globally, claims research, and hunger is to blame for about half of all child fatalities. Food is being squandered throughout.

The majority of people are unaware of the amount of food they waste every day, including uneaten leftovers, perishable foods, and fruit and vegetable portions that may be used in other ways.

Modern technology can no doubt take the fight against World hunger and food wastage to the next level.

## 1.2 Problem Statement

Every day, between 7,750 and 15,345 people pass away from hunger and malnutrition. This means that up to 11 people die from malnutrition every minute. According to the study, this is higher than the COVID death rate, which in July 2021 was around seven deaths per minute. *(Oxfam Media Briefing 2021).*

1.3 billion tonnes, or almost 1/3, of food produced for human use across the world is wasted every year. The majority of people are unaware of the amount of food they waste every day, including uneaten leftovers, perishable foods, and fruit and vegetable portions that may be used in other ways *(United States Environmental Protection Agency, 2022).*

By creating a food donation system that enables consumers to receive extra food from vendors, restaurants, residences, and volunteers, as well as supporting the battle against Zero Hunger (SDG 2).

## 1.3 Aims and Objectives

This project is targets easy and simple mode of connecting donors and receivers of shared meals or food products. with comfort and tracking of products being shared on through the app. The project’s objectives include:

1. To indicate the quality of the meal offered to customers.
2. To provide a platform for food vendors to share leftovers to the hungry.
3. To promote good health and well-being.
4. To improve existing systems and be ahead of competition.
5. To provide a system that will encourage restaurants and food vendors or individuals to help combat food wastage while contributing to greater goals (Sustainable Development Goals 2).
6. Indicating ingredients used in preparing meals.
7. Alert for the availability of leftovers.
8. Contributing percentage of payments for each meal.

## 1.4 Scope of the Study

The scope of this study includes:

1. This study focuses on developing a software mobile application for global usage.
2. Providing an acceptable means of donating extra quality food to people living in hunger-stricken communities or homes.

## 1.5 Significance of the study

With this study, we look to achieve SDG ‘Zero Hunger’ (SDG 2) by:

1. This study would, when adopted by the mass, massively reduce hunger crisis amongst certain communities and countries.
2. This study would reduce the rate of deaths caused by hunger and starvation globally mostly in children.
3. This study would, when adopted by restaurants, provide precise information about ingredients used in meal preparation to ensure good health provision to customers thus promoting good nourishments.

## 1.6 Limitations of the study

* An active internet connection is required. Only those with an internet connection can access the system. Therefore, you need cellular data, Wi-Fi, or an Ethernet connection to utilize the system.
* The project's inability to follow the package's whereabouts in real time is another drawback.

## 1.7 Organization of work

The first chapter highlights the introduction, background analysis, problem description, range of the investigation, and goals. The Second chapter consists of the literature review in which related works are compared to the recent study to help facilitate development for the new study. The Third chapter is the methodology which talks about the equipment and methods used to build the project. The Fourth chapter contains the results and analysis of the project. Chapter Five concludes the study with details about conclusions drawn from this project.

# CHAPTER TWO: LITERATURE REVIEW

This literature review explains the concept of a Courier Tracking System. It elaborates on what a Courier Tracking System is about and how it works. It also talks about the various advantages, disadvantages, and the development tools needed to develop a Courier tracking System. It also focuses on the subject knowledge, research environment, table of review papers, and an essay on the reviewed papers.

## 2.1 Subject Matter Knowledge

### 2.1.1 Online Database (Google Firebase)

A database that can be accessed online is one that is connected to the internet or a local network. Online databases are made available as a software service and are hosted on websites. A web browser can be used to access them.

A mobile development platform from Google called FIREBASE aids in the creation, enhancement, and expansion of apps. It offers a wide range of services, but its cloud storage and authentication service are crucial ones required for this project specifically.

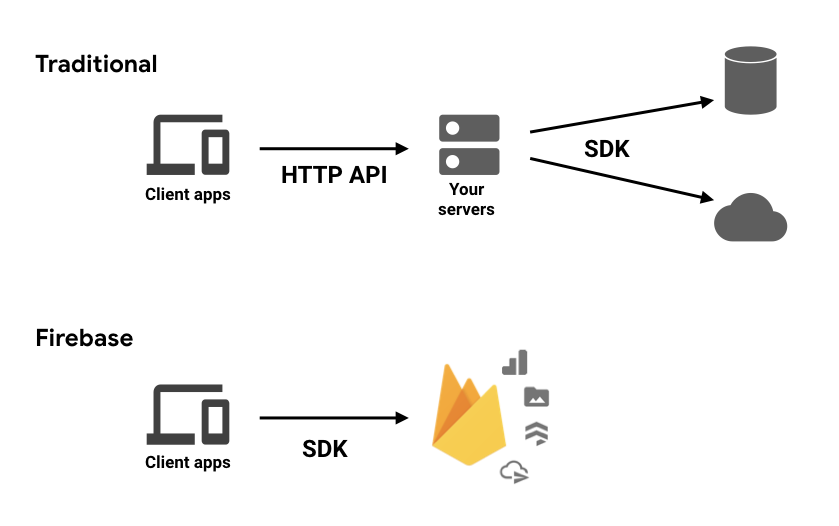


Fig. 1 Comparison between Firebase and Traditional database.

### 2.1.2 Android Operating System

Google (GOOGL) invented the Android OS is a mobile operating system with a Linux foundation that mostly powers smartphones and tablets. An operating system built on the Linux kernel, a GUI, a web browser, and end-user applications that can be downloaded are all included in the Android platform.

### 2.1.3 Key Takeaways

* Google (GOOGL) created the Android operating system to run on all of its touchscreen phones, tablets, and other mobile devices.
* Until purchased in 2005 by Google, this operating system was first created by Silicon Valley-based software startup Android, Inc.

Even while the Android source code is made available in an open-source format to assist create open standards for mobile devices, it is nevertheless offered on cell devices coupled with proprietary software.

### 2.1.4 Understanding the Android Operating System

Since the acquisition, the sincerity of Google's intentions to enter the mobile market has come under scrutiny from investors and analysts in the electronics industry. However, shortly after, Google announced that their first Android-powered smartphone would be available for purchase in 2007, even though it didn't launch until 2008. Since then, Android technology has made it possible for programmers to create mobile applications that are distributed through app stores like Google Play. Because Android was developed as a Google product, Android users also have the option to link their mobile devices to other Google products, such as cloud storage, email platforms, and video services.

The open-source Android source code is made accessible in order to create open standards for mobile devices. Despite being released as "open," Android still comes pre-installed on mobile devices with proprietary software.

### 2.1.5 Using flutter for the Android Application

**2.1.5.1 Flutter**

Google's UI software development kit, Flutter. Very common in the development of most mobile applications. Flutter Is classified as the top app development framework for Android developers in the twenty-first century. Flutter is an open-source software that provides both backend and front-end development. It facilitates a single language structure backend and its very powerful in app development. There are requirements for app creation when using flutter.

**2.1.5.2 Android Studio IDE**

Android Studio is the name of the authorized Android application development is done using an integrated development environment (IDE). It incorporates the developer and code editing tools from IntelliJ IDEA, a Java application integrated development environment. To aid with the creation of Android OS applications, emulators, code templates, GitHub connectivity, and a Gradle-based build mechanism are all utilized by Android Studio. Android Studio uses a feature called to upgrade active apps with fresh resources and code-named Instant Push. Applications made in Android Studio are then uploaded to the Google Play Store after an APK file has been developed. In May 2013 during Google I/O, the program was originally presented, and its first stable release was made available in December of that same year. Desktop PCs running Mac, Windows, and Linux can use Android Studio. It took on the function of the Eclipse Android Development Tools as the primary IDE for creating Android applications (ADT).

* Dart plugin
* Dart is a programming language created for client development, including the creation of PC, mobile, and online applications. A class-based, garbage-collected, object-oriented language with C-style syntax.
* Flutter plugin
* Java SDK
* Android SDK etc.

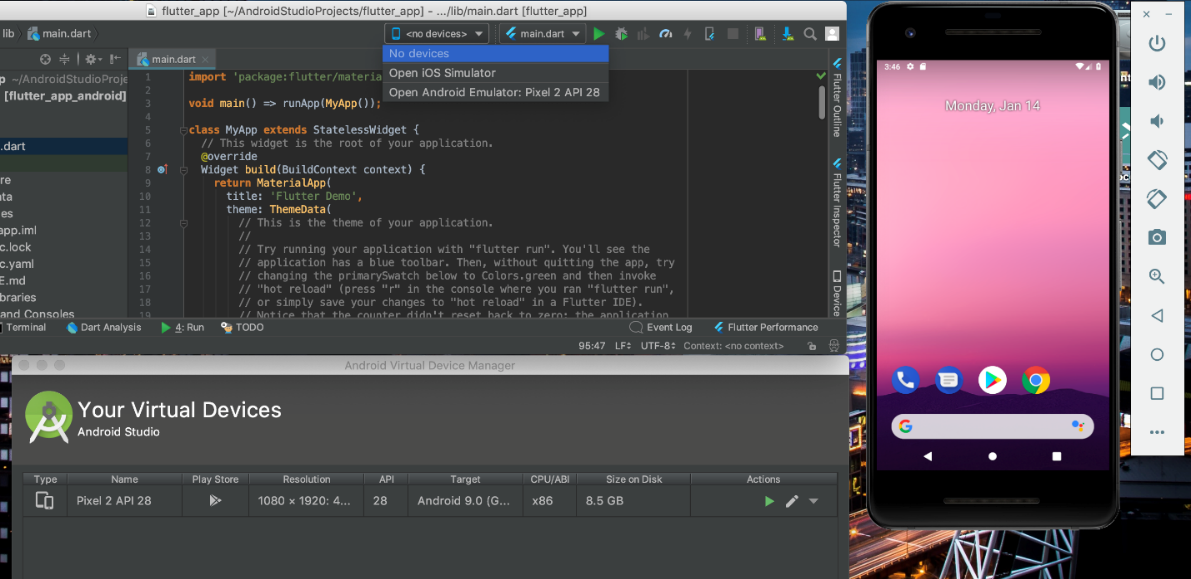


Fig 2. Android Studio IDE

## 2.2 Overview of Food Donation and Sharing mobile applications

The use of mobile software in food delivery is one of the leading applications in these pandemic times. It gained feat when Lockdown restrictions were placed on citizens for safety. It has since then improved in many ways. Not only can customers access the platform to donate from comfort of their homes, the system can also keep tab on the accounts of customers who access the services on the system. This is a reliable method that can be used to donate food.



Fig 3. Food related application image 1

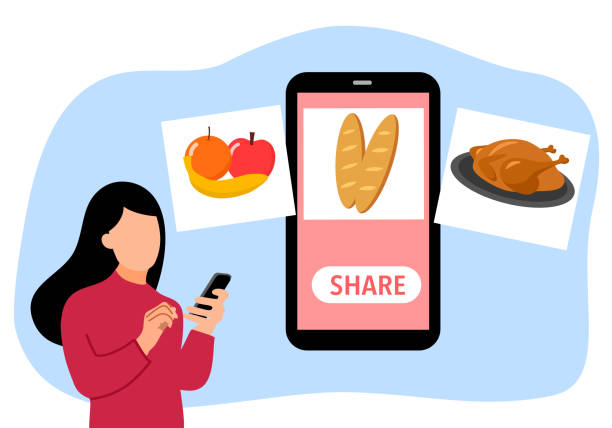


Fig 4. Food related application image 2

## 2.5 Review of closely related papers

Table 1 Review of closely related papers

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Author(s) and**  **Year** | **Country** | **Purpose of the Study /Research Objectives** | **Name /Label of Framework/Model designed by Author (Pictorial/drawing)** | **Technology Methods/Technique used by Tutor to solve the problem** | **Key Findings** | **Did you see Algorithm by Author?**  **Yes/No?** |
| Mrigank, M, Ishan, S, Vaishnavi R, S, Kalidass  2021 | India | Building an application to provide a forum to link charity donors to receivers or volunteers without the hinderance of data breaches or technicalities by intruders. | Aahar - Food Donation App | - Android-Studio IDE - Java programming language  - XML | - The use of technology to reduce food wastage. | Yes |
| Nívea, M, Davi, S, Breno, P, André, B, Beatriz, Trinchão, A, Sandra A  2018 | Brazil | Developing an application that connects food donors and receivers and makes charity organizations more visible | Combating Waste: A Mobile App for Food Donation | - Android studio components | - Using technology to redistribute food surpluses to help combat waste especially food wastage. | No |
| Zeel ,M, Darshan, G , Ketan, P, Ichhanshu, J  2022 | India | Building an online food donation service system to share unconsumed food with the needy | Food Waste Management App: Muskaan | - Android studio IDE - Cloud database | - Developing an app to reduce malnutrition in India and reduce food surplus wastage | No |
| Putri S, M, Sulistinayah S, P,  Rossy, A,  Lilit, R 2020 | Indonesia | Developing an android-based application to distribute leftover foods | Android Based Application Fawless (Food Assist Wasteless) As Innovative Solution On Reducing Food Waste | -Juggle Jack | - Encourage people to share their nearly-expired food and beverages to reduce the amount of food waste produced by society. | No |
| McDonald, D, Solovey, E 2017 | USA | Building a scheduling app  designed to facilitate the distribution of food that would  otherwise go to waste from grocery stores, to people  that need it. | AgileFood: Facilitating Adaptive Food Donation  to Address Hunger & Reduce Waste | - Android Studio 2.1  - Tested on 5554 Nexus 5 Emulator  - SQLite database | -Providing effective methods by building a mobile app for grocery stores to donate almost expired foods | No |
| George, Z, Marisabel, C, Yu, S, 2021 | USA | Building a data driven, easy to use, smart delivery app to assist users during corona pandemic , access food surpluses, groceries and shared meals. | A Mobile Platform For Food Donation  And Delivery System Using Ai And  Machine Learning | - Android studio  - Artificial Intelligence | - Using artificial intelligence to mediate between donors and targets to request and receive specific items.  -Also volunteers can see those requests and pick up the resources from food pantries, and deliver them directly to the homes of individuals | Yes |
| Hema S, M, Sai D, M, Bhanu S, K, Suvarna V, K,  2021 | India | Building a software application to connect people living in hunger and the donor and helpful to reduce the rate of deaths caused by starvation. | An Android Application on Food Salvage | - Android Studio IDE  - Google places API  - Google Firebase | - Using an android-based application to share surplus food and save lives.  -Food wastage can be avoided using food salvage app which helps the alms. | No |
| Carlos R, R, 2020 | Spain | Building a web application to streamline communication between food donors and food distribution entities | Development Of A Platform To Connect And Communicate Food Donors And Food Banks | - Microsoft Power Apps | - Developing a platform which allows donation management and provides fast and reliable communication between food-donating companies and the receiving and distributing entities. | Yes |
| Abhishek , D 2020 | India | Developing a mobile application to handle donations and connect the donors with the nearest needy person through a social mobile application. | Food Donation App | - Android studio IDE  - SQL server database  - Java programming language | - Using technology to serve as a medium to assist NGOs reach the needy and donate food to curb starvation and world hunger  -Waterfall model was used | Yes |
| Saveetha, G, Rajermani, T, Sathiapriya, R, 2019 | Malaysia | A study to investigate the understanding and daily practices of surplus food handling by the public as well as those in the food industry | Surplus Food Donation: An Analytical Review for The Lost Food Project (NGO) | Data driven Research | - The study uncovered the need for the project to take place  - The study also uncovers benchmarks required for the potential development of a surplus food donation system. | No |
| Al-Shammari A, F, N, Ahmed, F, 2020 | Iraq | Building a food waste map and collect data about generated waste and geolocate areas noted for food wastage | FoodWise: Geolocalised Food Wastes Tracking and  Management | - Mobile Software application  - Geolocation map  - Database access API. | - To identify the patterns of food wastes generated by using geolocalized data | Yes |
| Samantha, B 2016 | USA | Developing food recovery programs to help people with food insecurity | Cowboy Food Sharing: Supply the Soup Kitchen and Reduce  Waste | Data Driven research | - Taking high quality foods that would otherwise be wasted and donate to people who need them. | No |
| Isha, S 2015 | USA | creating a website application that gives people a sharing platform for home-cooked food | MEALS2SHARE  Neighborhood Home Cooked Food Sharing Web Application | - Web application  built in ASP.NET framework using MVC (Model View Controller) development model  - Requires SQL Server. | - An easy to use interface to make donors share the food they have prepared in their homes with pricing they desire and receiver user can search the food they would like to eat and locate the cook in geographical proximity  -This web application brings its own advantage to both donors and receivers by providing immense business opportunity to the service provider through this ecommerce web application.  - Based on the hybrid ecommerce – sharing economy model | Yes |
| Lucia, P,  Margherita P, 2017 | Italy | To highlight the need to solve food wastage globally | Responses To Food Waste In A Sharing Economy: We Have Fully Transitioned To A Participatory Culture, And Digital Technology Is Key Driver Of That Transition | Data driven research | -Proposing solutions to combat food wastage  -Identifying technology as the most effective method to help combat food wastage and ensure better food management | No |
| Morilla, JAR, Bagsic, FC, Cruz, MKD, et al ... 2021 | USA | - To develop a mobile and web application specifically for food donor and beneficiaries,  -A food donation application | Foodernity: A Mobile and Web Application for Food Sharing | A mobile and web- application | - Combating food waste and hunger by linking food producers with people in their area searching for quality food at fast food places, grocery shops, and other similar establishments  - Agile development method | Yes |
| Satish, R, S, Stella, R, Aishwarya, S | India | Developing an online blood, money, clothes and food donation | A Smart Platform for Donation Handling | - A mobile and web- application  - Client server computing technology  - MVC model  3 tiers architecture: Client tier, Server Tier, and Data Tier.  - Google Maps Web Services  - Mysql database  - JPA. | - State-of-the-art technologies to preserve privacy of big data at each stage of big data life cycle.  - Proposes a system that helps social organizations run projects transparently, effectively and reach more people in need.  - User decides what to donate; Blood, money, food, clothes, amenities etc.  Then the system shows list of NGOs.  -Donor can keep track of his donation to prevent scams. | Yes |

## 2.6 Related Works Essay

1. **Combating Waste: A Mobile App for Food Donation**

Nívea Melo, Davi Silva dos Santos, Breno Piva, André Britto, Beatriz Trinchão Andrade and Sandra Avila proposed a system for food donation to combat waste. This idea was suggested due to their discovery about the FAO. According to them, FAO, fish and seafood are excluded from the direct economic effects of food waste, which total around $680 billion for wealthy nations and $310 billion for developing nations. Although existing solutions seek to reduce the waste of food, they are more suitable to nations where residents feel safe to provide their addresses to a stranger or to go to an unknown person’s address to collect food. In Brazil, food donation can be done through direct contacting charitable organizations or through food banks. They then introduced the Combating Waste app, which is based on a donation network that searches for compatible donations and demands, generating rides that link them. When compared to other food donation apps, their approach is specially designed to Brazilian reality: it provides more safety for the users; it focuses on delivering food to people in need; anyone can contribute, even persons who have nothing to donate can help taking the donation rides.

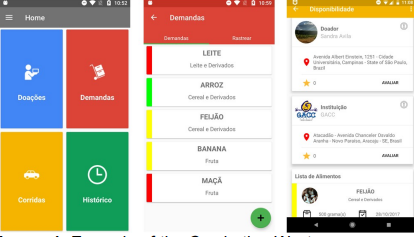


Fig. 5

The image above is an example of the combating waste app screen.

**How The App Works: Components of the App**

The Combating Waste app enables charitable organizations to register as food recipients and allows individuals, restaurants, merchants, companies, and manufacturers to register as food givers. Volunteer work does not simply rely on donations; several organizations can both donate food and assist in transporting it from a food donor location to a charitable organization. A volunteer on the one hand offers to contribute food.

The Combating Waste app enables charitable organizations to register as food recipients and allows individuals, restaurants, merchants, companies, and manufacturers to register as food givers. Volunteer work does not simply rely on donations; several organizations can both donate food and assist in transporting it from a food donor location to a charitable organization. A volunteer on the one hand offers to contribute food. They were able to learn about nearby activities and their intended audience as a result. The software keeps the essential data about its users and the food listed in the settings for both donation and demand. To help the transporters, it also generates rides using our ride composition technology. Examples of the Combating Waste app interfaces are shown in Fig. 1.0. Fig 1.0. An illustration of a Combating Waste app screen. Currently, the software only supports Portuguese. The home screen of the main app. (middle) Demands list (from a charitable institution point of view). Ride information screen (right) (from a transporter point of view).

1. **PantryGo: Mobile Platform for Food Donation and Delivery System Using AI and Machine Learning.**

The idea of creating a smartphone app for a food donation and delivery system using artificial intelligence and machine learning was put out by George Zhu, Marisabel Chang, and Yu Sun. The surge in unemployment rates during the COVID outbreak led to an economic disaster. About 35 million Americans struggled with hunger in their homes prior to the epidemic; this number has since escalated as a result of the pandemic's effects. With regards to this, George Zhu, Marisabel Chang et al decided to design a solution that can alleviate this hunger and food poverty using mobile platform.

**About The System**

A smart delivery platform called PantryGo was created in response to rising food insecurity as well as the medical and financial problems brought on by the new coronavirus. It is data-driven, user-friendly, and smart. Through interactions with "Services providers," such as local food pantries, supermarkets, and leading philanthropic organizations, it aims to assist "Clients," the most vulnerable and isolated residents in Orange County who are unable to access adequate food sources due to a number of barriers. To deliver groceries, meals, and other items to clients' doorsteps, the delivery company employs "Volunteers." PantryGo is not just helping those in need, but also reducing food waste at grocery stores and food banks by delivering predicted demand, food shelf life, food availability, resource allocation, and other artificially intelligent information via the website. Service Providers can trust PantryGo to know what kinds and how much of what is needed to meet those families' needs. The second is operational effectiveness, which implies that PantryGo provides the quickest service or delivery when compared to other food pantries. Technology is not typically employed in the world of food pantries or any group providing free resources to the most vulnerable. The distinctiveness of this product will allow food banks, grocery stores, and all service providers to better serve their customers by saving time, gas, and resources on a single platform.

PantryGo also has the advantage of being able to serve various customer types in any situation. The program makes it possible for everyone, regardless of their physical, mental, or geographical constraints, to purchase high-quality food and beverages.

The data from PantryGo also provides a picture of food insecurity and hunger in a particular town, revealing a wealth of information on the behavior of the clients. The most popular items are shown here, along with a map showing the locations of the clients and the typical volunteer delivery time. This information can help food banks and other organizations better understand the traits and behavior of these families so that they can modify their programs and services accordingly. Finally, by aiding our churches, governmental bodies, and decision-makers in allocating resources and tackling the root causes of food poverty, PantryGo builds a stronger community. Management groups can coordinate, execute, and resolve problems better than before.

**Challenges Faced in Building the Tracking System**

* One of the first and most difficult jobs we had to finish in order to deal with the problems that were developing at the food pantry was finding a solution. No one has come up with a means to expand the usage of food pantries or the food bank system since they were first proposed a few decades ago, or to upgrade them to a technology level. For PantryGo's early versions of the program and the way that food would be delivered, it was difficult to find the perfect structure and system because of this. Additionally, we did not want to alter the approach the food pantry now employs in search of a fix.
* Designing, developing, and engineering the PantryGo mobile application will probably be the key challenge for the experiment or project. Building a professional application from scratch as a coder with only a few weeks of experience was fairly challenging. Due to the implementation of code as well as continuing design and UI/UX improvement, numerous revisions and upgrades to the app's code were required. Furthermore, there were many instances where it was challenging to strike the right balance between a functional app with a basic UI/UX and a piece of professional software with a well-developed UI/UX.
* Promotion: Promoting PantryGo's mobile app and the company as a whole is a task that is regarded to be a never-ending issue. The creation of the promotion assets, such as the logo, website, social media, flyers, and graphic arts, necessitates a lengthy period of revision and refinement until satisfaction. After the theoretical invention and design of these assets are finished, there is a lengthy process in which the minute details and laborious development take place. Before being completely destroyed and recreated from scratch, many of the components also underwent multiple revisions and iterations. After completion, these assets had to be sent to the appropriate users.

**Solution**

Selecting one of three user types will allow PantryGo users to register. The first category is the Foodbank Profile. This profile will be used by Foodbank managers or anyone in charge of the specific food pantry's activities. Foodbank profile users will be able to publish details about their operating hours, inventory, location, and other data that other users can read in order to save clients and volunteers time, energy, and misunderstandings. The second category is the client or customer profile. People or families who want to purchase food from the food pantry will utilize this profile. Users can enter their location and other private information if they so choose. Additionally, customers can choose a shopping list with their top choices on it, choose a food pantry that will show the list's available items, and finally choose another shopping list. The third user is the volunteer profile. For liability and safety reasons, this person will be allowed to enter personal information so that other users may be aware of the specifics of their volunteer. The clients' requests for deliveries for the day will be available to the volunteers, who will also be able to check delivery details including the delivery location and pick-up times. Customers' requests will also be accessible through the food bank profiles, and as they select products, the food bank's inventory will change immediately.

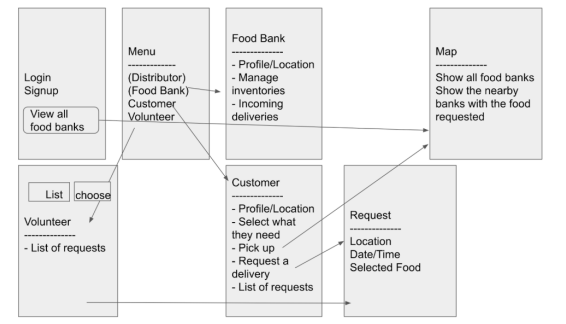


Fig 6 Components Of The App

The initial part of the system is the food bank user, and we wrote the code so that one of the user's main duties is being able to manage the stock of the food pantry. In other words, the administration of the food bank or pantry will have the ability to alter any food products or boxes that are regularly accessible for pickup or delivery. Users of the food bank will also be able to publish information and change their whereabouts for the viewing of other users, such as customers and volunteers.

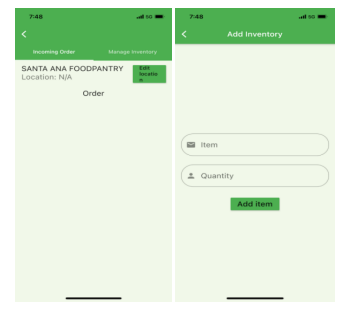


Fig 7: Customer User

The client user, which makes up the second element, performs a few key tasks. The consumer will be able to request a food list from a food bank in their neighborhood based on the goods that are available, and thanks to artificial intelligence, the food items they chose will immediately link with a food pantry that has those products. Customers will also be able to view all of their previous requests as well as the request's specifics; when a volunteer selects a customer's request, the request will automatically be marked as completed and moved into the request history.

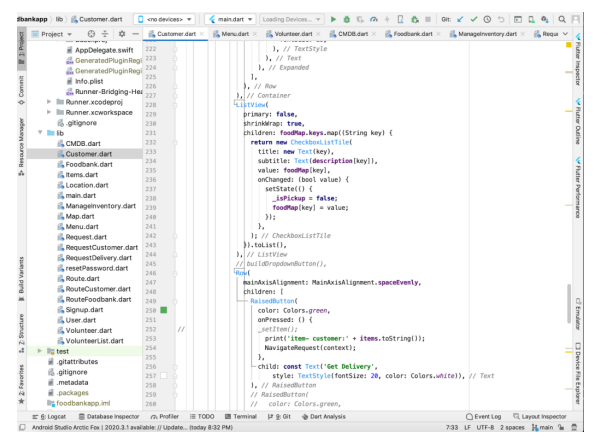


Fig 8 Code Snapshot

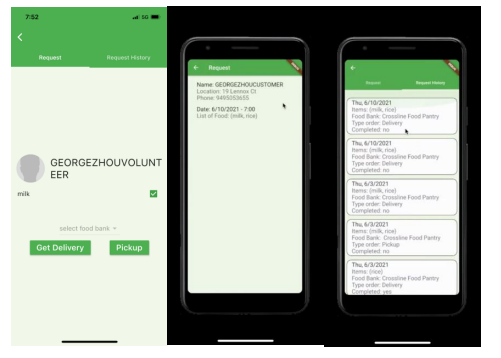


Fig 9: Volunteer User

The third element is the volunteer user, who is mostly in charge of choosing customer meal orders. Automatically the volunteer will receive each request for meals from a customer along with all pertinent details, such as the location, time, date, and food items. The request will move to the list of customers after the volunteer selects it, and after it is finished, the screens of the other two users will be updated. Figure 6 displays the code that enables volunteers to choose customer requests, and Figure 7 displays screenshots from the viewpoint of the volunteer user. Utilizing particular steps, the relationship between the three elements is formed. The request will be forwarded to the food pantry and the volunteer, who will then choose the required food, pick it up from the pantry, and deliver it to the person's or family's home.

**Experiment**

In order to evaluate the efficacy of our strategy, they collected data from the Crossline Church food pantry over a two-week period by tallying requests, weighing food waste, and surveying patrons and volunteers. Two experiments looked at the effectiveness of the treatment. The first experiment examines the potential impact of the application on food waste, and the second experiment gauges the potential impact of the application on volunteerism.

**Experiment 1: Pantry Go’s effect on Food Waste**

Data on the quantity of food waste that remains after each week are being gathered for this study. We observed the first week without using our solution, and the second week with its implementation. Food surpluses in the United States have had a negative carbon effect on the environment, making food waste a major global issue. The application expands the food pantry's audience and draws more visitors, which greatly reduces the amount of food that is wasted in facilities like food pantries. The fact that food waste was reduced by more than 50% after the application's implementation shows how effective it is at encouraging better use of the food pantry.

**Experiment 2: PantryGo’s effect on Volunteer Involvement**

Data and information will be gathered for this project from surveys given to different young volunteer groups asking them whether they would be interested in volunteering at the food pantry. The first survey was conducted before the control group was informed about the possibility of working as delivery drivers during the first week, and the second survey was conducted immediately after the group was informed about the app during the second week.

**3. Android Application On Food Salvage**

Hema Sowjanya Mamidi, Sai Divya Moguloju, and others came up with the concept for the Android app food salvage. With the help of this app, donors who are willing to provide food to feed hungry victims will be able to connect with hungry victims. We must stop wasting food and use it to feed the hungry. Every day, a large amount of food that is useless for anyone is thrown out. We are able to aid the hungry and those without access to food thanks to effective food management. Food is being wasted in significant amounts nowadays, yet there is no suitable program for helping the hungry. Numerous applications surfaced, and government organizations control food rescue numbers.

**Algorithm Used**

Google Places API was employed for locating the nearby orphanages. They are using Google Places API Google indexes its spatial and geographical data using S2 cell.

S2 cell hierarchy:

* Sufficient resolution to index geographical characteristics
* Querying using arbitrary locations quickly
* Each cell at a given level need to have a comparable area.

Google uses the A\* Algorithm to quickly determine the quickest path and alternate routes. An enhanced version of breadth-first search is the A\* algorithm. It selects the most promising option while avoiding the expensive path. A really clever algorithm, indeed. In practical applications, such as in maps, it is used to roughly represent the shortest path. In the case of the Google Map, the weight represents journey time. It is explained via weighted graphs. Starting with a single node (the source node) in a graph, it constructs a tree of pathways, developing each pathway one at a time until it reaches the predetermined destination node.

**Methodology**

In order to contribute or share food, a user must log into the app. After logging in, a site is shown so visitors can view messages about the excess food that is accessible. The donor must provide both his personal information and information pertaining to the food before clicking the contribute button. The post then displays on the home page, and if the recipient wants to order the food, he can type his location and see posts that are relevant to it. Therefore, he can accept the meals from a donor based on his needs. The information about the food will be displayed by clicking on the posts, and by clicking "Get food" button, a notification that someone is asking for his meal will be sent to the donor. The giver can then share his extra food with the recipient after getting in touch with them both. The donor can also look out nearby "Hunger Food Spots," such as orphanages and nursing homes, where they can visit and personally provide food. We are using the Google Places API to find the orphanages and nursing facilities.

**Implementation**

1. **Database**

There are two types of databases in Google Firebase. They are: Every user contains an unique id and then for that specified user we store the other details like name, email id, and default images.



Fig 11 Database structure

* Cloud Firestore
* Real-time Database

**Posts:**

The user must upload the food image and then provide the essential details in order to keep a post. The Google Firebase's storage component receives and stores the uploaded image.

**Storage:**

The Firebase SDKs for Cloud Storage are used by developers to upload and get files directly from users. The client can resume an operation where it left off if the network connection is weak, saving your users' time and bandwidth. Firebase and Google Cloud both provide access to files thanks to Cloud Storage, which stores them in a bucket on Google Cloud Storage. This gives you the freedom to use Google Cloud Platform's server-side processing tools, such as image filtering or video transcoding, to transfer files between mobile clients and a desktop computer utilizing the Firebase SDKs.

Cloud Storage scales automatically, meaning that there's no need to migrate to any other provider. Cloud Storage is used to upload files and also stores data on the Internet. It provides a feasibility to resume the operation where it is left. We can see how the images are stored in the storage and a link is generated the generated link is stored in the Posts reference of database as shown in Fig. 3. the details of the donor i.e., the name of the donor, their address, City of the donor, their email id, user\_id and the timestamp.

**Authentication:**

To contribute or claim food, a user must be logged in to the application. They must therefore create an account. To store user data, we employ Google Firebase. Identify users by using their email addresses and passwords. The Firebase Authentication SDK can be used to create and manage users who log in using their email addresses and passwords. Additionally, Firebase Authentication is used to send emails for password resets.

1. **The Android App "Fawless" (Food Assist Wasteless) as a Creative Way to Reduce Food Waste**

Pukri Sekar Pelati, Sulistinayah Suwaka Putri, et al. observed that humas produce trash known as clamp shells. At the moment, both locally and globally, there is a serious issue with food waste. In light of their findings, they proposed developing an Android app as a way to reduce instances of food waste, called "FAWless" (Food Assist Wasteless).

**Methodology:**

They used the Descriptive Analysis method of study for their findings. a technique that aids in providing an overview of the things investigated using data. In order to discover a solution for the issue of food waste, an android-based application is being developed.

Using the program "Juggle Jack," the application was made. Using the capabilities of the Juggle Jack app there is an opportunity to select for the building of the app. The app's features cover all other options that can be used during app creation, such as edit, add image, add icon, and many more.

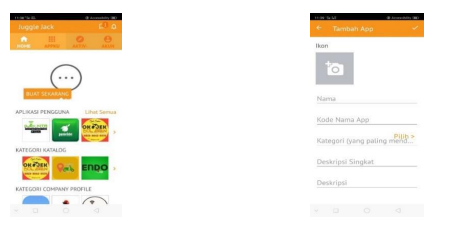


Fig 12 Application snapshots

**Experiment**

To download the app, there is a link that is provided by the “Juggle Jack” app which can be used to download the app. User’s email would be required after download after which the app is then ready for usage. The app is very user friendly.

1. **Food Management App: Muskaan**

Zeel Mehta, Darshan Gawade et al proposed that food can be managed to reduce the rate of wastage thus, some could be available for donation to feed the hungry victims. They decided to use this idea by developing a food management app for gathering excess food to make them available for donation.

**Features of The App:**

* User Account Management: Users are allowed to register and create profiles
* Management of profiles: The following elements make up the user profile: Name, Last Name, Email, Gender, Phone Number, Address, and Identity Proof
* Profile Improvement: The user will have the option to register as a representative of an NGO or restaurant, each with their own specialty. A user must provide the documentation proving membership in the organization.
* Process for receiving food: Users would get alerts for local locations.
* Search results shall enable filters and reviews/comments/ratings filters.
* Donor’s approval is waited for and received after applying for his or her approval
* Dine (Hoteliers, Restaurants):
* Only users whose accounts have been upgraded and validated as belonging to restaurant personnel are permitted to mention food that is available for ordering or dining. Users can look up nearby eateries and dine cheaply after business hours.
* Food Quality: While the application does not ensure the caliber of the food that will be delivered, recipients and NGOs can still be confident in the caliber by looking at the reviews of food provided at the time of applying. When a user takes a picture of food, the food's quality rating will be determined.
* KARMA Points: An interactive feature that allows users to earn rewards for food donations. KARMA points will be produced for each donation, which may then be redeemed for dining at restaurants, playing amusing games, and earning interesting coupons and prizes.

Muskaan Mingle: A social media component built specifically for food donation services can support and encourage user participation on the app. Additionally, it will serve as a bridge between those in need of assistance and those eager to provide it.

**6. Food donation app**

Datta proposed the development of an android application for food donation. The development was to be done using Android Studio and Java.

**Features:**

* Can be accessed by one entity that is, the user.
* Login credentials required for login at first for access.
* Admin page available for managing and updating list.

**Model:**

Abishek Datta sought to use the Water fall Model for development; a linear sequential flow. Progress is seen a steady flow in each phase of implementing the software.

**Project Design:**

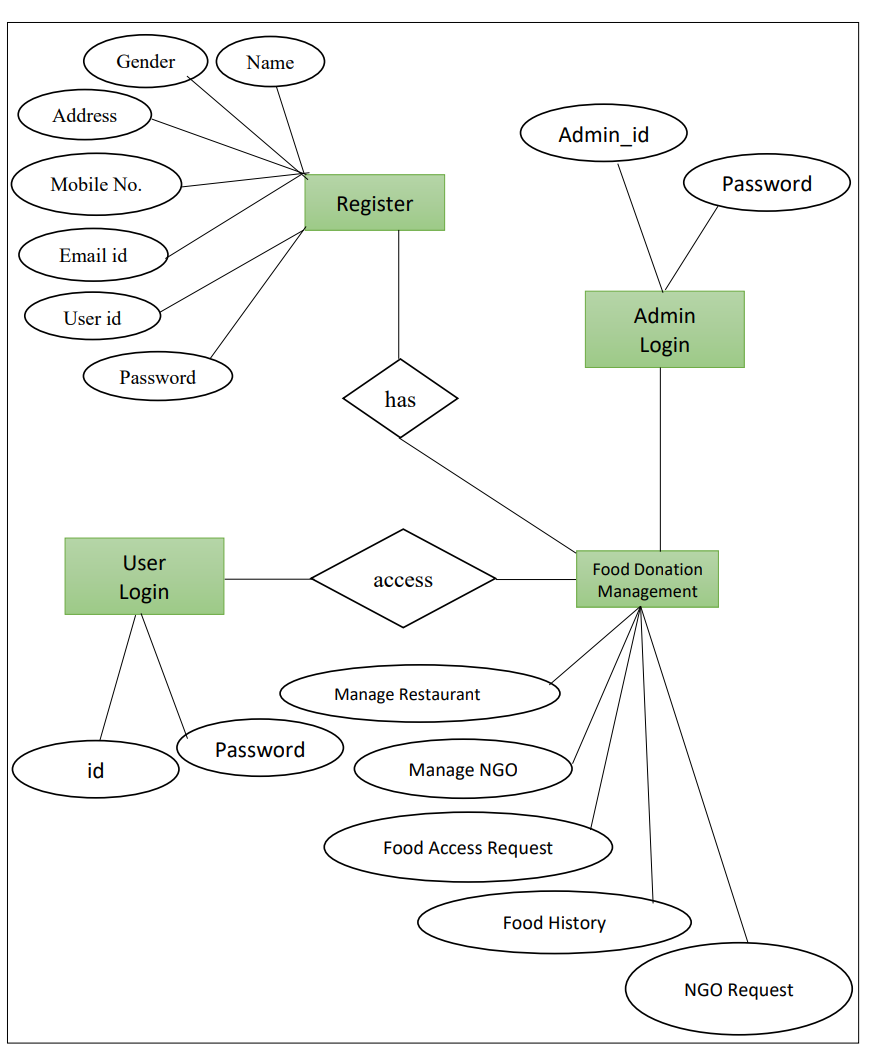


Fig 13. E-R Diagram

1. **A Smart Platform For Donation Handling**

Satish R. Shelar, Stella Raju Salve et al proposed the idea of developing a system for donating, blood, money, food and clothes online. With Distributed Client Server computing in mind, the entire project has been developed. A proper systematic management would be put in place to promote transparency amongst social organizations running this project.

**Features:**

* User Friendly
* System Architecture: The MVC architecture, which the system uses, allows for the separation of concerns notion.

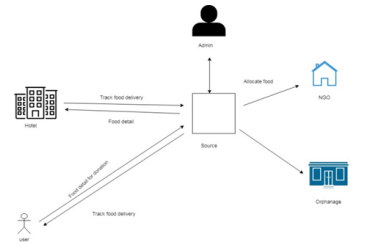


Fig 14. System Architecture

The diagram above is the System’s architecture, this follows the MVC model which has a common strategy for the Client, Server, and Data tiers of its three-tier architecture.

**Modules:**

* Android as well web application for registering or sign up with email, password or mobile phone number.
* User get welcoming mail from website after sign up.
* Users can watch current schemes of what is happening in the village.
* Admin can handle requests of all donors and receivers.

1. **Aahar Food Donation App**

To decrease food waste, Mrigank Mathur, Ishan Srivastava, and others suggested creating a food donation app. The framework needs to be straightforward but effective so that anyone may use it without worrying about technical details or data breaches. After successfully logging into the system, the "Aahar" app for Android smartphones offers donors and searchers forums where they may gather and donate food.

**Features**

* Android based, generated using Java and xml on Android Studio.
* Simple and easy to use framework’s UI

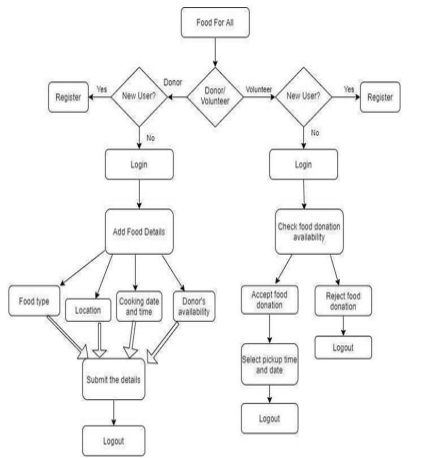


Fig 15 Activity flow diagram

* **Modules of the App**
* Login and Registration: Login and enlist is involved for both guests and agents.
* Administrator: Subtleties are kept up by the admin.
* Donator: Donor allocates excess food to the shelter

**CHAPTER THREE: METHODOLOGY AND SYSTEM DESIGN**

## 3.0 Analysis and design

The methodology of each research project is one of the most important chapters. The reader is provided with an overview of the specific research and design methodology used in this chapter as well as the type of analysis produced.. This chapter provides the data source and its collection methods while evaluating the existing system; its architecture, structure, and design. It introduces the selected Software Development Life Cycle method and how it was applied in the development or framework of the proposed system together with its functional and non-functional requirements for its implementation and justification thereof according to the laid-out objectives of the project.

This chapter is devoted to the creation of the proposed system, which is the Food Donation and Sharing System. It explains how the system's parts work together to carry out its functions.

**3.1 Requirements Gathering Methods**

There are several different sorts of methods available for obtaining requirements, including user observation, questionnaires, interviews, and document analysis, among others. The procedures used to collect requirements for the suggested system are described below.

**3.1.1 Interviews**

There are two main ways to conduct interviews: one-on-one and in groups. The most typical interviews are one-on-one approach for gathering needs and one of the primary sources of requirements. Stakeholder identification is necessary before conducting interviews with them. These can be users who interact with the old or new system, management, project financiers, or any other system players. The fact that more than one person is being interviewed is the only distinction between a group interview and an individual interview. Group interviews work best when the participants are in the same role or status. As a result of someone in the group stating or suggesting an idea that may have gone unnoticed by others, which in turn might lead to a conversation or provide more information on a specific subject, further ideas and discussion may be created. The interviewer can determine which topics are more widely agreed upon and which topics are more divisive. The interview's schedule may be a significant drawback. It could be challenging or take a lot of time when multiple people are involved. Regarding the suggested system A few restaurants conducted one-on-one interviews with wait staff and other personnel to learn more about meal service and how extras are stored, as well as to gather criteria regarding the suggested system. Group interviews with restaurant management staff were done to collect non-functional needs, management-level requirements, and details about the current manual process. A few NGO personnel were also interviewed to gather data for the suggested system.

**3.1.2 Analyzing Existing Documents**

Pre-existing articles can be effectively analyzed to gather needs, either independently or in conjunction with other techniques. By reviewing the current method and documentation, the analyst can have a deeper understanding of the business or system and its current status. The titles and names of all system stakeholders will be provided to the analyst from the paperwork that is currently in place. This will help the analyst develop questionnaires or interview questions to ask stakeholders in order to learn more. In order to gather data and needs for the proposed system, papers related to the current manual procedure, such as menu cards, payment instructions, request forms, documents connected to donation data, etc. were evaluated.

**3.1.3 User Observation**

The analyst may observe the user in person to gain a deeper understanding of them in their current work setting. By providing the analyst with a thorough understanding of the user's interactions with the system, user observation is useful. When stakeholders find it challenging to describe precisely what their tasks entail and what their requirements might be, watching the user might help offer the specifications. The validation of previously gathered data may also benefit from user observation. To better understand the current food management system and to obtain data for the suggested solution, user observation was conducted for the Daily Bite Foods restaurant personnel. This was quite helpful in determining what features and functions the beneficiaries of the proposed system should have access to.

**3.2 Chosen Methodology For This Project : Waterfall**

The Waterfall model, a sequential development process that moves through all project phases, including analysis, design, development, and testing, with each phase being finished in full before the beginning of the next.

The sequential method of the waterfall process is based on predetermined completion dates, conditions, and results. The various execution teams can work autonomously up until the point where specialist integrations are required thanks to this strategy, which also does away with the necessity for regular communication. The frequency of progress reports is lower than it would be under the Agile methodology because team members usually work alone. Normally, before starting a new phase, the prior one must be finished.

**Using a software development project as an example, the Waterfall process usually includes stages that look like this:**

1. **Requirements:**

The idea that all project specifications can be obtained and understood in advance is the basis of the Waterfall technique. The project manager goes above and above to fully comprehend the requirements of the project sponsor. Written requirements detail every aspect of the project, including the costs, assumptions, risks, dependencies, success indicators, and completion deadlines. Typically, a single document contains all of these requirements.

1. **Design**

Here, scenario, layout, and data models are developed by software designers to implement a technical fix for the problems mentioned in the product requirements. This system’s objectives, scope, general traffic patterns for each component, and connectivity points are initially laid out in a higher-level or logical design. Once complete, it is transformed using specific hardware and software technologies into a physical design.

1. **Implementation**

Once the design is complete, technical implementation starts. This stage of the Waterfall process might be the quickest because of the rigorous research and planning that has already been done. Based on project specifications, programmers build programs during this phase, with some testing and implementation also taking place. If significant changes are required during this phase, it could be essential to go back to the design phase.

1. **Verification or testing**

To ensure there are no bugs and that all requirements have been satisfied, testing is required before a product is made available to users. This will provide a great user experience with the application. The testing team's test cases will be built on the personas, use case scenarios, and design papers that the product manager has provided.

1. **Deployment and maintenance**

After the program has been made available to clients or introduced to the market, the maintenance phase begins. As defects are found and user modification requests are submitted, a team will be assigned to manage upgrades and the rollout of new software versions.

**3.2.1 Advantages of the Waterfall methodology**

The waterfall technique is a tried-and-true approach to project management that has a clear definition and is simple to apply.. Because the requirements are outlined in depth from the beginning, each contributor may properly schedule their time for the duration of the project.



Fig 16 Waterfall Methodology

**3.3 System Requirement**

**3.3.1 Hardware:**

The system development process would require a number of physical devices and components:

* Laptop with at least 8 GB of RAM and about 100 GB of free disc space
* Windows 10 or 11 operating system
* Flutter SDK installed

**3.3.1 Software:**

1. Visual Studio Code
2. Flutter SDK
3. Android Studio
4. Online Database (Firebase)

**3.4 Non-Functional Requirements**

**3.4.1 Usability**

The system should take into account the demands of managing the effectiveness of web-based systems, as well as managing customer orders and payments. It should generate pertinent data on each item of food and drink so that customers may base their orders on it. The system need to work better during busy hours.

**3.4.2 Availability**

System is an Android-based mobile app software that allows users to log in from anywhere in the world at any time (24 X 7 availability.) Additionally, the database backups should be kept at least weekly or monthly, and the system needs to secure access to private information (user details).

**3.4.3 Access Control**

Administrators and users should have proper access to and control over the system. Customer accounts are handled by administer, and super admin handles administrator accounts.

**3.4.4 Modifiable**

Only administration has the authority to alter menu items, pertinent photos, etc. Customers can examine certain details in read-only mode.

**3.4.5 Platform compatibility**

System is an android software. So, it is platform dependent.

**3.4.6 Flexibility**

The system should be adaptable enough to handle any workload. It should be able to handle customer orders and configure SMS notifications, which would be very desirable for future expansion.

**The technologies that were employed were:**

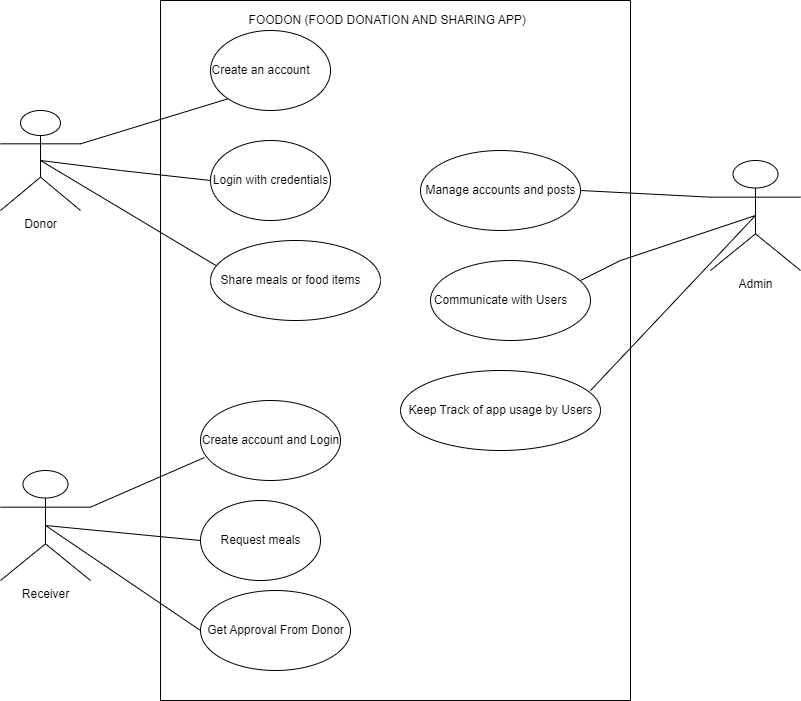
* Online Database (Firebase)
* Dart Plugin
* Visual Studio Code
* Android SDK
* Flutter plugin

Lack of knowledge about development tools was also a huge problem. Books, Video lessons and support from knowledgeable people were used to overcome this problem to some extent.

Future Enhancement:

* A feature for enabling a signal of the exact location at any time to display posts made by nearby donors.
* Including a feature for verification using IDs and Biometric.

Fig 17 Use Case Diagram



# CHAPTER FOUR: SYSTEM IMPLEMENTATION AND TESTING

## 4.0 Introduction

The system documentation is written information describing the operation, the structure and the use of the system. This section is also to explain the terms used in the work. It also includes most of the general flows of the program that clarify what goes on in the system. Moreover, the documentation is to guide users guide users navigate through the system once the user enters the system. In a nut shell, the documentation is to curb the tendency of forgetting the details of the system. For the system to be more efficient and flexible, user must be supplied with a proper documentation on loading procedure, input, control, halt points, dump medium, user messages, error messages, PF keys required, queries etc. during compilation and running time. System testing, implementation and documentation is all about testing the system to verify it. It meets the required specification, after which it is implemented in the proposed environment using the interface, device and power supply.

**4.2 Testing of the New System**

**4.2.1 Unit Testing**

The system should be adaptable enough to handle any workload. It should be able to handle customer orders and configure SMS notifications, which would be very desirable for future expansion.

**4.2.2 Login Page**

A text field for the username and password, as well as a login button, should be visible on the login page. The login button will transport you to the admin homepage if you enter the username and password for an admin. Additionally, the user is prompted when they enter the incorrect username or password. The administration can also employ the same strategy.

**4.2.3 Admin Home Menu**

When logging in using an admin username or password, the admin home page opens by default. Any admin page's home menu, when selected, directs you to the admin home page. Here, the admin users have the right to make changes to the platform, such as updating data on the foods that are currently available and newly introduced ingredients, assisting with in-app communication with users, and giving donors health advice in order to encourage them to donate only nutritious food items.

**4.2.4 Records Page**

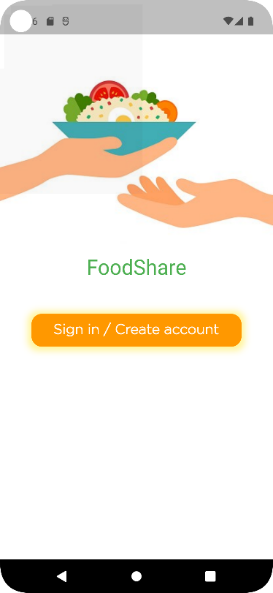
The records page is where the details of the food is recorded. The details include the ratings of food served, the number of foods shared, reviews by users on the platform. This page is only accessible by the admins.

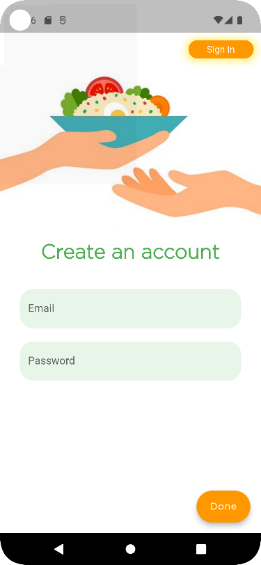
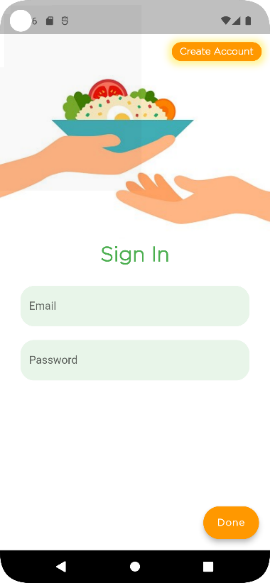
**4.2.5 Donation Page**

The donation page is where the donor users are allowed to input the food items, they are willing to add for donation. As per the Good Health and well-being part of the SDG’s certain criteria would be posted on the page so as to guide users on what food to go ahead and donate as this would protect the receivers from any future health complication as a result of poorly kept food items.

**4.2.6 Receivers Page**

The receiver’s page is where receiver users would be given the opportunity to access the available donated foods. Here, there is a portion for in app communication so as to communicate with users on the urgency with which they require these donated foods as this would help prevent unqualified users from being given these services.





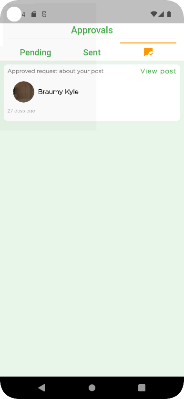
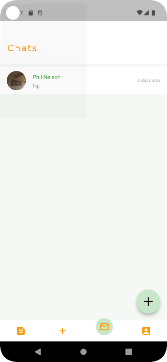


Fig 18

Fig 19

Fig 20

Fig 21

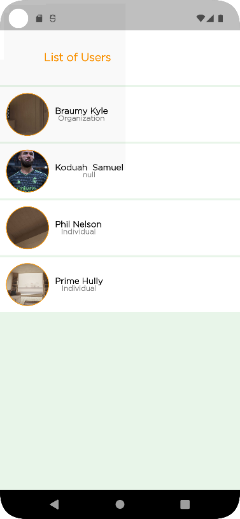
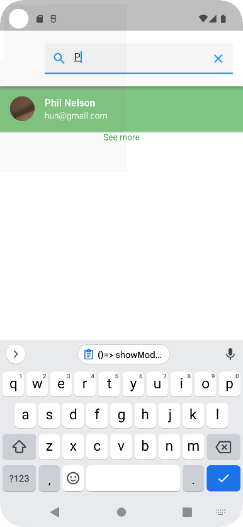
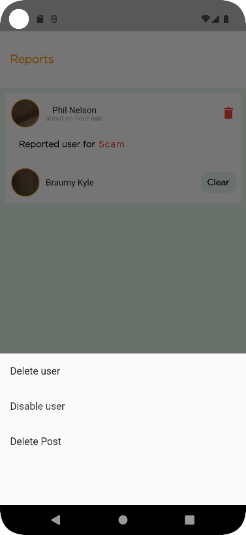
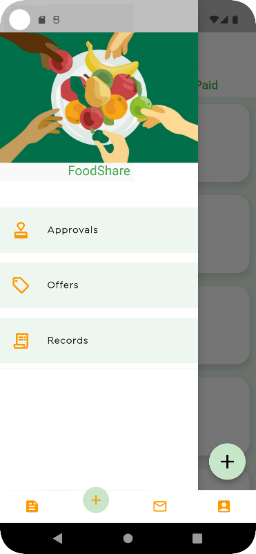
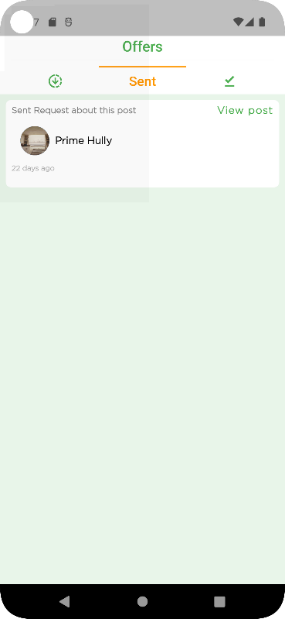


Fig 25

Fig 22

Fig 24

Fig 23



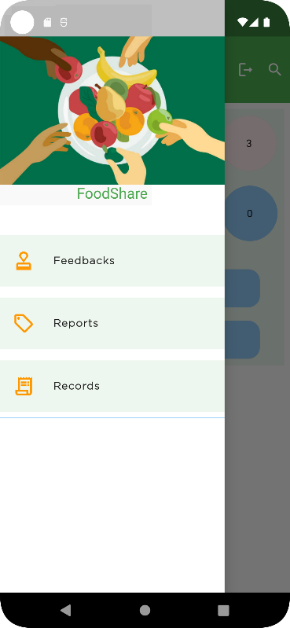
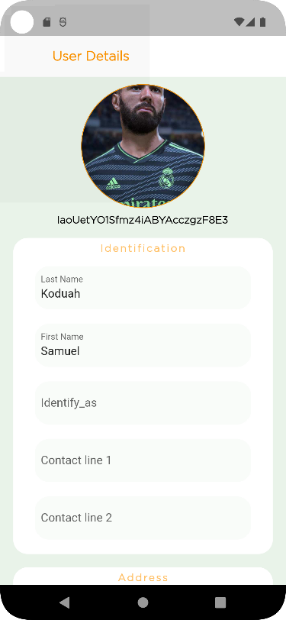
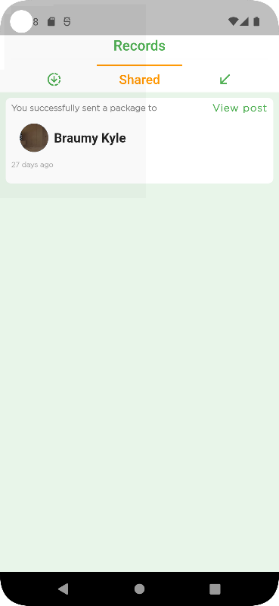


Fig 28

Fig 26

Fig 27

Fig 29

**4.3 Functional Testing**

Application testing against business requirements is known as functional testing. Functional testing is carried out utilizing the use cases provided by the design team or the client-provided functional specifications. The purpose of functional testing is to verify how the application behaves. This strategy aims to reduce hunger-related problems while also advancing human health and wellbeing. Unauthorized individuals would not be permitted because users must create personal accounts, and because of the in-app communication feature, users are screened based on how urgently they require this service. This system is to provide receivers with good and healthy foods and also make sure the issue of hunger is solved if not totally but brought to a minimum. With this system, admins can notice all unqualified users who would try to access the platform.

**4.4 Implementation of The New System**

The process of building a new system and putting it into production is known as systems implementation. The phased implementation technique will be followed in the implementation of this system.

**4.5 System Documentation**

The method, the structure, and the use of the system are mostly covered in the system documentation. It provides general program flows that describe what occurs in the system. Once a person has logged in, the documentation instructs them on how to move around the system.

**4.6 About the System**

About the system describes mainly what the system does. “Foodon”, the name of the proposed system is an android mobile application used for donating meals especially excess meals available from restaurants, vendors, homes and volunteers to needy victims. Volunteers or donors who decide to present their meals on the platform are entitled to posting only free or meals with subsidized prices that would be affordable to those in need.

**4.6.1 User Access Level**

User access level describes the access every user of this system has. In this system, the main users are; admins, donors and receivers. The donor is only allowed to tick the option to indicate that he or she has donated his or her food after being given access by the admin. The same applies with the receiver, where users are given the go ahead to come and receive their food after they have passed the criteria test, they would take on the platform.

* **Admin**

The admin is responsible for handling feedbacks, reports, blocking and deleting accounts that disturb users on the application and also checking the statistics of the application. The admin is the only one who has the authority to carry out the said operations. The admin can neither be a receiver nor a donor.

* **Donor**

Any authorized individual or organization who made a post willing to donate or share food or food items is referred to as the donor. A post is made by the donors to indicate and describe the item being donated. They make the post based on their terms. The donor can also be a receiver and vice versa.

* **Receiver**

Any authorized individual or organization which seeks to receive help from donors by sending a request to a donor's post is a receiver. Receivers can accept donations from donors by sending requests to a donor for a specific post they made. They can also get in touch with donor through in-app communication.

**4.7 Conclusion**

This system was able to meet requirement according to the objectives. It can be easily accessed and user friendly. Below are snap shots from the system.

# CHAPTER FIVE: CONCLUSION AND FUTURE WORK

## 5.0 Introduction

The "food donation/food sharing app" has been successfully designed and tested. It was created by integrating functionality from every piece of hardware used. During the course of this project, significant details that can be included were highlighted, including an in-app communication feature to assist users on how to conveniently benefit from our services. Another area noted and implemented is the logins feature which only gives access to only qualified users. The findings, recommendations, and conclusions from the study's analysis and research are condensed in this chapter.

## 5.1 Summary

The world is looking to reduce world hunger and combat food wastage and what better way to accomplish that than using modern technology. This system was introduced with certainty to provide an easy and effective method to locate starving individuals and repurpose food products to aid in the battle against World Hunger.

## 5.2 Strength of the System

1. The system has user authentication.
2. This feature prevents unauthorized access to the system.
3. It has an online database with powerful security to store user data and records.
4. Helps users (donor or receiver) connect through the apps chat feature.

## 5.3 Limitation (weakness) of the system

This system is a web application, and even though it can be accesses on a mobile device it would be good to have a mobile app version of it. Also, a client should be able to create an account in order to get benefits like discounts and others.

## 5.4 Opportunities

By turning leftover meals or food products into donations, this invention will reduce food waste. To use the app's features, users simply need to connect through it from anywhere at any time.

The concept offers a quick and straightforward approach to combat world hunger.

## 5.5 Threats

The number of competitors is ever growing as years go by; new applications are being developed every day, making it quite difficult for a particular system to survive in the industrial market. In view of this, developers are advised to indulge the various end-users in development to know their opinions concerning the development of these applications.

## 5.6 Recommendations and Future work

Users provided insightful comments and suggestions about potential ways to improve this system throughout system testing. Several of these suggestions include;

* Handling payment in the app.
* Making the app work offline.
* Real time tracking of package in app.
* Rewarding donors and special benefit schemes for users.
* Verification using IDs and biometrics
* Engaging government bodies to endorse users by some form of certification to prevent the use of the app for illegal activities and crimes such as mass poisoning.

## 5.7 Conclusion

After carefully reviewing other systems, we studied that these systems did have similarities with our system for certain features but as to the ingredient display as well as the in-app communication feature, they did not include them in their system. The inclusion of this ingredient feature in our system would encourage healthy nourishment since consumers would be aware of what they would be consuming and as a result this would reduce the health implications that might arise as a result of allergy related issues. Also, the communication feature will help users stay in touch. Users enjoyed the UI experience generally because of the user-friendly UI. We'd want to sum up by emphasizing that this system's development will promote food sharing and giving, hence reducing hunger. The required adjustments were performed, and the system was constructed to meet the requirements.

# REFERENCES

MATHUR, M, SRIVASTAVA, I, RAI, V, & KALIDASS, APS 2021, 'Aahar-Food Donation App'

MELO, NNB, AVILA, SEF DE, SANTOS, DS DOS, & ... 2018, 'Combating waste: a mobile app for food donation'

MEHTA, Z, GAWADE, D, PATIL, K, & JAISWAL, I, 'Food Waste Management App: Muskaan',

MELATI, P, PUTRI, S, PUTRI, R, & RUSYATI, L 2020, 'Android Based Application Fawless (Food Assist Wasteless) As Innovative Solution on Reducing Food Waste', *no. December*, researchgate.net

MCDONALD, DQ, & SOLOVEY, ET, 'AgileFood: Facilitating Adaptive Food Donation to Address Hunger & Reduce Waste', *users.wpi.edu*,

ZHOU, G, CHANG, M, & SUN, Y 2021, 'AMobile Platform For Food Donation And Delivery System Using Ai And Machine Learning', *CS & IT Conference Proceedings*, csitcp.com

MAMIDI, HS, MOGULOJU, SD, KOTHA, BS, & VANI, KS, 'An Android Application on Food Salvage', *ijfe.latticescipub.com*,

RAMZI, CR, 'Development of A Platform To Connect And Communicate Food Donors And Food Banks', *repositorio.comillas.edu*,

DATTA, A 2021, 'Food Donation App'

GANESAN, S, THINAKARAN, R, & RAMIAH, S, 'Surplus Food Donation: An Analytical Review for The Lost Food Project (NGO)', *researchgate.net*

AL-SHAMMARI, AFN, & FADHIL, A 2020, 'FoodWise: Geolocalised Food Wastes Tracking and Management', *Kerbala Journal for Engineering*

BRANT, SL 2016, 'Cowboy Food Sharing: Supply the Soup Kitchen and Reduce Waste'

SINGH, I 2015, 'MEALS2SHARE Neighborhood Home Cooked Food Sharing Web Application'

POTO, MP 2017, 'Responses to Food Waste in a Sharing Economy'

MORILLA, JAR, BAGSIC, FC, CRUZ, MKD, & ... 2021, 'Foodernity: A Mobile and Web Application for Food Sharing'

SHELAR, SR, SALVE, SR, & KEDARI, AS, 2020 'A Smart Platform for Donation Handling'.

# APPENDIX: SOURCE CODE

import 'package:flutter/material.dart';

import 'package:sates/authentication/auth.dart';

import 'package:firebase\_core/firebase\_core.dart';

import 'dart:async';

void main()async {

  WidgetsFlutterBinding.ensureInitialized();

  await Firebase.initializeApp();

  return runApp(Foodshare());

}

class Foodshare extends StatefulWidget {

  @override

  State<Foodshare> createState() => \_FoodshareState();

}

class \_FoodshareState extends State<Foodshare> {

  @override

  Widget build(BuildContext context) {

    return StreamProvider<USER?>.value(

      value: AuthService().user,

      initialData: null,

      child: MaterialApp(

        theme: ThemeData(

         primaryColor: Colors.green,

          accentColor:Colors.orange,

        ),

        debugShowCheckedModeBanner: false,

        initialRoute: '/start',

        routes: {

          '/start':(context) => Firstpage(),

        },

      ),

    );

  }

}

final chatsListRef = FirebaseFirestore.instance.collection('chats\_list');

final usersRef = FirebaseFirestore.instance.collection('users');

class Chats\_screen extends StatefulWidget {

  @override

  State<Chats\_screen> createState() => \_Chats\_screenState();

}

class \_Chats\_screenState extends State<Chats\_screen> {

  var myProfileUrl;

//  final currentUser= auth.currentUser;

  final Stream<QuerySnapshot?>\_chats\_withStream=FirebaseFirestore.instance

      .collection('chats\_list')

      .doc(w.auth.currentUser!.uid)

      .collection('chats\_with')

      .orderBy('LastMessageTime',descending: true)

      .snapshots();

  bool isLoading=false;

  @override

  void initState() {

    getMyURl();

    super.initState();

  }

  getMyURl()async{

    if (w.auth.currentUser!.uid==null)

      await usersRef

          .doc(w.auth.currentUser!.uid)

          .get()

          .then((ds){

        myProfileUrl=ds.data()!['ProfilePhotoUrl'];

       // print(currentUserName);

      }).catchError((e){

        print(e);

      });

  }

  @override

  Widget build(BuildContext context) {

    return StreamBuilder<QuerySnapshot?>(

      stream: \_chats\_withStream,

        builder: (BuildContext context, AsyncSnapshot<QuerySnapshot?> snapshot){

        if (snapshot.hasError){

          return Scaffold(

            appBar:AppBar(

              //centerTitle: true,

              iconTheme: IconThemeData(

                  color: Colors.orange

              ),

              automaticallyImplyLeading: false,

              elevation: 0,

              title:Column(

                // mainAxisAlignment: MainAxisAlignment.spaceBetween,

                children: [

                  SizedBox(

                    height: 30,

                  ),

                  Text('Chats',

                    style: TextStyle(

                        fontFamily: 'Gotham',

                        color: Colors.orange,

                        fontSize: 20,

                        fontWeight: FontWeight.bold,

                        letterSpacing: 3

                    ),

                    textAlign: TextAlign.start,

                  ),

                ],

              ),

              toolbarHeight:100,

              backgroundColor: Colors.white,

            ),

            body: Center(

                child:Column(

                  mainAxisAlignment: MainAxisAlignment.center,

                  children: [

                    CircularProgressIndicator(),

                    Text('Something went wrong'),

                  ],

                )

            ),

          );

        }

        if(snapshot.connectionState==ConnectionState.waiting){

          return Scaffold(

            appBar:AppBar(

              //centerTitle: true,

              iconTheme: IconThemeData(

                  color: Colors.orange

              ),

              automaticallyImplyLeading: false,

              elevation: 0,

              title:Column(

                // mainAxisAlignment: MainAxisAlignment.spaceBetween,

                children: [

                  SizedBox(

                    height: 30,

                  ),

                  Text('Chats',

                    style: TextStyle(

                        fontFamily: 'Gotham',

                        color: Colors.orange,

                        fontSize: 20,

                        fontWeight: FontWeight.bold,

                        letterSpacing: 3

                    ),

                    textAlign: TextAlign.start,

                  ),

                ],

              ),

              toolbarHeight:100,

              backgroundColor: Colors.white,

            ),

            body: Center(

              child:Column(

                mainAxisAlignment: MainAxisAlignment.center,

                children: [

                  CircularProgressIndicator(),

                  Text('Loading'),

                ],

              )

            ),

          );

        }

        if(snapshot.hasData &&  snapshot.data!.size==0){

          return Scaffold(

            appBar:AppBar(

              //centerTitle: true,

              iconTheme: IconThemeData(

                  color: Colors.orange

              ),

              automaticallyImplyLeading: false,

              elevation: 0,

              title:Column(

                // mainAxisAlignment: MainAxisAlignment.spaceBetween,

                children: [

                  SizedBox(

                    height: 30,

                  ),

                  Text('Chats',

                    style: TextStyle(

                        fontFamily: 'Gotham',

                        color: Colors.orange,

                        fontSize: 20,

                        fontWeight: FontWeight.bold,

                        letterSpacing: 3

                    ),

                    textAlign: TextAlign.start,

                  ),

                ],

              ),

              toolbarHeight:100,

              backgroundColor: Colors.white,

            ),

            body: Center(

                child:Column(

                  mainAxisAlignment: MainAxisAlignment.center,

                  children: [

                    Text('No chats yet...',

                    style: TextStyle(

                        color: Colors.black.withOpacity(0.5)

                    ),

                    ),

                  ],

                )

            ),

          );

        }

        return Scaffold(

          backgroundColor: Colors.green.shade100.withOpacity(0.1),

          appBar:AppBar(

            //centerTitle: true,

            iconTheme: IconThemeData(

                color: Colors.orange

            ),

            automaticallyImplyLeading: false,

            elevation: 0,

            title:Column(

             // mainAxisAlignment: MainAxisAlignment.spaceBetween,

              children: [

                SizedBox(

                  height: 30,

                ),

                Text('Chats',

                  style: TextStyle(

                    fontFamily: 'Gotham',

                      color: Colors.orange,

                      fontSize: 20,

                    fontWeight: FontWeight.bold,

                    letterSpacing: 3

                  ),

                  textAlign: TextAlign.start,

                ),

              ],

            ),

             toolbarHeight:100,

           backgroundColor: Colors.white,

          ),

          ///List of chats with other users

          body: ListView(

            children:snapshot.data!.docs

                .map((DocumentSnapshot document){

                  Map<String, dynamic> data=

                      document.data()! as Map<String,dynamic>;

                  var num=snapshot.data!.docs.length;

                    DateTime dt =(data['LastMessageTime']as Timestamp).toDate();

                  return Container(

                    color: Colors.white,

                    child: Column(

                      children: [

                        GestureDetector(

                          onTap: ()=>Navigator.push(context, MaterialPageRoute(

                              builder: (context)=> ChatDetail(

                                friendName:(data['Username']).toString(),

                                friendUid: (data['Id']).toString(),))),

                          child: ListTile(

                           dense: true,

                            title: Text((data['Username']).toString(),

                              style: TextStyle(

                                  color: Colors.green

                              ),

                            ),

                            subtitle: Text((data['LastMessage']).toString()),

                            trailing: Text(dt.toString(),

                              style: TextStyle(

                                fontSize: 11

                              ),

                            ),

                            leading: CircleAvatar(

                              radius: 25,

                              child: Center(

                                child: CachedNetworkImage(

                                  imageUrl:(data['ProfilePhotoUrl']).toString(),

                                  imageBuilder: (context, imageProvider) => Container(

                                    decoration: BoxDecoration(

                                      borderRadius: BorderRadius.circular(50),

                                      image: DecorationImage(

                                        image: imageProvider,

                                        fit: BoxFit.fill,

                                      ),

                                    ),

                                  ),

                                  placeholder: (context, url) => CircularProgressIndicator(),

                                  errorWidget: (context, url, error) => Icon(Icons.error),

                                ),

                              ),

                            ),

                          ),

                        ),

                        Divider(),

                      ],

                    ),

                  );

            })

                .toList()

                .cast(),

          ),

        );

        }

    );

  }

}

final usersRef = FirebaseFirestore.instance.collection('users');

final postsRef = FirebaseFirestore.instance.collection('posts');

class home extends StatefulWidget {

  @override

  State<home> createState() => \_homeState(

  );

}

class \_homeState extends State<home> {

  final currentUser= auth.currentUser!;

  String postId=Uuid().v4();

  final PageController \_pageController= PageController();

  var scaffoldKey = GlobalKey<ScaffoldState>();

  bool? NoUrl;

    int pageIndex=0;

    @override

    void initstate(){

      //checkUser();

     // getPhotoUrl();

     //getUsername();

   /\*  getProfilePosts();

     getUsername();\*/

     super.initState();

    }

  String? Url;

  String? Username;

  Future<String> url() async {

    DocumentSnapshot docS = await usersRef.doc(auth.currentUser!.uid).get();

    String urL = (docS.data() as Map)["ProfilePhotoUrl"];

    return urL;

  }

  Future<String> username() async {

    DocumentSnapshot docS = await usersRef.doc(auth.currentUser!.uid).get();

    String userName = (docS.data() as Map)["Username"];

    return userName;

  }

  @override

      void dispose(){

      \_pageController.dispose();

      super.dispose();

      }

    onPageChanged(int pageIndex){

      setState((){

        this.pageIndex = pageIndex;

      });

    }

    onTap(int pageIndex){

    \_pageController.animateToPage(

      pageIndex,

      duration: Duration(milliseconds: 250),

      curve: Curves.bounceInOut

    );

    }

  @override

  Widget build(BuildContext context) {

    final Size size =MediaQuery.of(context).size;

   return Scaffold(

      body: PageView(

        controller:  \_pageController,

        onPageChanged: onPageChanged,

        physics: NeverScrollableScrollPhysics(),

        children: [

          Timeline(),

          Requests\_screen(),

          Chats\_screen(),

          Profile(profileId:currentUser.uid),

        ],

      ),

      bottomNavigationBar: CurvedNavigationBar(

        height: 50,

        index: pageIndex,

        onTap: onTap,

        animationDuration: Duration(milliseconds: 100),

        buttonBackgroundColor: Colors.green.shade100,

        animationCurve: Curves.bounceInOut,

        backgroundColor: Colors.white,

        color: Colors.white,

        items: [

          Icon(Icons.feed,

            color: Colors.orange,

            semanticLabel: 'Home',

          ),

          Icon(Icons.add,

            color: Colors.orange,

            semanticLabel: 'Requests',

          ),

          Icon(Icons.mail\_outline,

            color: Colors.orange,

            semanticLabel: 'Messages',

          ),

          Icon(Icons.account\_box,

            color: Colors.orange,

            semanticLabel: 'Profile',

          ),

        ],

      ),

     /\* ConvexAppBar(

        height: 50,

        initialActiveIndex: pageIndex,

        onTap: onTap,

        backgroundColor: Colors.orange,

        color: Colors.green,

        activeColor: Colors.green,

        curveSize: 100,

        top: -20,

        items: [

          TabItem(icon:Icon(Icons.feed,

            color: Colors.white,

            semanticLabel: 'Home',

          ),

          ),

          TabItem(icon:Icon(Icons.mail\_outline,

            color: Colors.white,

            semanticLabel: 'Messages',

          ),

          ),

          TabItem(icon:Icon(Icons.account\_box,

            color: Colors.white,

            semanticLabel: 'Profile',

          ),

          ),

        ],

      ),\*/

    );

  }

}

class Profile extends StatefulWidget {

  final String? profileId;

  Profile({this.profileId});

  @override

  \_ProfileState createState() => \_ProfileState();

}

class \_ProfileState extends State<Profile> {

  final usersRef = FirebaseFirestore.instance.collection('users');

  final postsRef = FirebaseFirestore.instance.collection('posts');

  final recordsRef = FirebaseFirestore.instance.collection('records');

  final reviewsRef = FirebaseFirestore.instance.collection('reviews');

  final ratingsRef = FirebaseFirestore.instance.collection('ratings');

  var \_textController = new TextEditingController();

  int pageIndex=0;

  final PageController \_pageController= PageController();

  final AuthService \_auth= AuthService();

  final \_formkey= GlobalKey<FormState>();

  bool isLoading=false;

  int postCount=0;

  var reviewID= Uuid().v4();

  final currentUserId= FirebaseAuth.instance.currentUser!.uid;

  List<UserProfilePosts> pposts=[];