Ground Fresh-Bridging the Gap between Farmers and Consumers with Mobile App

By

Antora Debnath
ID: CSE1903018097

Rajib Raihan
ID: CSE1903018113

Mst.Eti Akter
ID: CSE1903018078

Ommey Habiba Sweety
ID: CSE1903018061

Jannatul Ferdous
ID: CSE1903018079

Supervised by
Mohammad Naderuzzaman

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)

May 2023

APPROVAL

The Project titled **“Ground Fresh (Mobile Apps)**” submitted by Antora Debnath (CSE1903018097), Rajib Raihan (CSE1903018113), Mst.Eti Akter (CSE1903018078), Ommey Habiba Sweety (CSE1903018061), Jannatul Ferdous (CSE1903018079) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Mohammad Naderuzzaman
Assistant Professor,
Department of Computer Science and Engineering
Sonargaon University (SU)

Supervisor

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 1

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 2

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 3

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Mohammad Naderuzzaman,** Assistant Professor**,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned	Signature
------------------------------	-----------------------
Mohammad Naderuzzaman	**Antora Debnath**
Supervisor	ID: CSE1903018097

	Rajib Raihan
	ID: CSE1903018113

	Mst.Eti Akter
	ID: CSE1903018078

	Ommey Habiba Sweety
	ID: CSE1903018061

	Jannatul Ferdous
	ID: CSE1903018079

ABSTRACT

Ground Fresh is a mobile application that aims to bridge the gap between farmers and consumers by providing a platform for direct sales of agricultural products. The app allows farmers to create profiles, list their available products, and set prices. Consumers can browse the available products, place orders, and make payments directly to the farmers.

The app features a user-friendly interface that allows farmers to manage their profiles and products easily. Farmers can update their product listings, view orders, and track sales history. The app also features a messaging system that allows farmers and consumers to communicate directly with each other, enabling them to discuss orders, delivery options, and other details.

For consumers, Ground Fresh provides a convenient way to purchase fresh, locally sourced produce directly from farmers. Consumers can search for products based on location, type, and availability. They can view detailed information about the products, including photos, descriptions, and pricing. Once they have placed an order, they can track the delivery status and communicate with the farmer if necessary.

The main objective of the Ground Fresh app is to promote local agriculture and support small-scale farmers by providing them with a direct sales platform. By eliminating intermediaries, farmers can sell their products at better prices, while consumers can enjoy fresh produce at affordable prices. The app also helps to reduce food waste by enabling farmers to sell their products before they spoil.

ACKNOWLEDGMENT

At the very beginning, we would like to express my deepest gratitude to the Almighty Allah for giving us the ability and the strength to finish the task successfully within the schedule time.

We are auspicious that we had the kind association as well as supervision of **Mohammad Naderuzzaman,** Assistant Professor, Department of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out our project. His insights and feedback were invaluable in helping us to refine our ideas and turn them into a functional app.

We would like to convey our special gratitude to **Brig. Gen. (Retd) Prof. Habibur Rahman Kamal,** Dean, Faculty of Science & Engineering for kind concern, discretion, friendly behavior and precious suggestions.

We would also like to thank our team who worked tirelessly to bring the app to life. Their dedication and hard work were essential in ensuring that the app met the highest standards of quality and usability.

We are also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, our deepest gratitude and love to my parents for their support, encouragement, and endless love.

LIST OF ABBREVIATIONS

SDK	Software Development Kit
VS Code	Visual Studio Code
UI	User Interface
IDE	Integrated Development Environment
JIT	Just-in-Time
AOT	Ahead-of-Time
OOP	Object-Oriented Programming
DFD	Data Flow Diagram
ERD	Entity Relationship Diagram
GFA	Ground Fresh App

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGUREURES

CHAPTER 1

1. INTRODUCTION

1.1. Introduction

Ground Fresh is a mobile application that aims to revolutionize how consumers purchase fresh vegetables. With the rise of online shopping, there has been a growing trend towards buying fresh produce online. Ground Fresh provides a platform for buyers and sellers of fresh vegetables to connect seamlessly and efficiently.

Built using Flutter, a powerful and flexible open-source framework for developing high-performance, natively compiled mobile apps, Ground Fresh is designed to provide an intuitive and user-friendly interface for both buyers and sellers. The app is specifically tailored to the needs of the fresh produce market, with features that include a searchable database of fresh produce items, the ability to browse and compare prices from different sellers, and a seamless checkout process that makes purchasing fresh produce as easy as possible.

For buyers, Ground Fresh offers a one-stop shop for all their fresh produce needs. The app makes it easy to find the best deals on fresh vegetables, compare prices from different sellers, and place orders with just a few clicks. Buyers can also track their orders in real-time, so they always know when their fresh produce will arrive.

For sellers, Ground Fresh provides a powerful platform to connect with buyers and grow their businesses. The app makes it easy for sellers to list their products, manage their inventory, and track their sales. Sellers can also take advantage of Ground Fresh's marketing tools, such as targeted promotions and coupons, to attract new buyers and increase sales.

But Ground Fresh is more than just a platform for buying and selling fresh produce. It is also a platform for building relationships between buyers and sellers. By providing a transparent and trustworthy marketplace, Ground Fresh fosters trust and loyalty between buyers and sellers, helping to create a community of people who are passionate about fresh produce and sustainable farming practices.

In this book, we will explore the impact of Ground Fresh on the fresh produce market. We will examine how the app has transformed the way consumers buy fresh vegetables, and how it has helped to grow the businesses of small-scale farmers and local producers. We will also explore the challenges faced by Ground Fresh in its mission to create a transparent and trustworthy marketplace for fresh produce, and how the app has adapted to meet these challenges.

Through interviews with farmers, buyers, and sellers who have used Ground Fresh, we will gain insights into the benefits and drawbacks of using a mobile app to buy and sell fresh produce. We will also examine the role that technology can play in promoting sustainable farming practices and reducing food waste.

Ultimately, our goal is to provide a comprehensive analysis of Ground Fresh and its impact on the fresh produce market. We believe that this research paper will provide valuable insights for policymakers, farmers, and other stakeholders who are interested in promoting sustainable agriculture and creating more efficient and transparent markets for fresh produce.

1.2.Objectives

1.2.1. To analyze the factors that led to the development of Ground Fresh as a platform for buying and selling fresh vegetables.

1.2.2. To examine the features of the app that make it a unique and valuable platform for buyers and sellers of fresh produce.

1.2.3. To assess the impact of Ground Fresh on the fresh produce market, including its effects on prices, quality, and food waste.

1.2.4. To explore the challenges faced by Ground Fresh in its mission to create a transparent and trustworthy marketplace for fresh produce, and how the app has adapted to meet these challenges.

1.2.5. To analyze the potential of mobile apps like Ground Fresh to promote sustainable agriculture practices and reduce food waste.

1.3. Background & Significance

1.3.1. Traditional food supply chains face significant challenges and inefficiencies, including long transportation distances, excessive reliance on intermediaries, and a lack of transparency. These issues have far-reaching implications for the industry and highlight the need for improvements in the supply chain.

1.3.2. **Long transportation distances:** Traditional food supply chains often involve transporting food over long distances from farms or production centers to consumers. This leads to increased transportation costs, carbon emissions, and energy consumption. Moreover, extended travel times can result in delays, reduced freshness, and spoilage of perishable food items. The reliance on long-distance transportation also limits the availability of locally produced food, as consumers depend on goods shipped from distant regions.

1.3.3. Excessive reliance on intermediaries: Traditional food supply chains typically involve multiple intermediaries such as wholesalers, distributors, and retailers. Each intermediary adds their margin to the product cost, leading to higher prices for consumers. Additionally, the involvement of numerous intermediaries increases complexity, communication challenges, and potential disruptions in the supply chain. These inefficiencies can result in longer lead times, increased wastage, and difficulties in coordinating the demand and supply effectively.

1.3.4. Lack of transparency: The lack of transparency is a significant challenge in traditional food supply chains. Consumers often have limited visibility into the origin, production processes, and handling practices of the food they consume. This lack of information makes it difficult for consumers to make informed choices about the quality, safety, and sustainability of the products they purchase. Furthermore, it creates opportunities for fraud, mislabeling, and adulteration, which can compromise food safety and erode consumer trust.

To address these challenges, it is crucial to reimagine and transform traditional food supply chains. One possible solution is the adoption of shorter supply chains, which reduce transportation distances and support local economies. This can be achieved through the promotion of local and regional food systems, where farmers and producers are directly connected with consumers, reducing the need for extensive intermediaries

CHAPTER 2

2. FOOD SUPPLY CHAINS AND CHALLENGES

2.1.Overview of Traditional Food Supply Chains

Traditional food supply chains are the conventional systems that involve the production, processing, distribution, and retailing of food products. They typically consist of multiple stages and numerous intermediaries between producers and consumers. Here is an overview of the key components and processes involved in traditional food supply chains:

2.1.1. Production: Food production starts with farmers and agricultural producers who cultivate crops or raise livestock. They employ various methods such as planting, nurturing, harvesting, and breeding to produce raw agricultural commodities.

2.1.2. Processing: After the initial production stage, raw agricultural commodities undergo processing and transformation into intermediate or finished food products. This can involve activities such as cleaning, sorting, cutting, cooking, packaging, and preserving.

2.1.3. Distribution: Once the food products are processed and packaged, they are distributed to various locations. This step involves transporting the products from manufacturing facilities or processing centers to regional distribution centers, warehouses, or retailers. Transportation methods may include trucks, trains, ships, or airplanes, depending on the distance and urgency.

2.1.4. Intermediaries: Traditional food supply chains typically involve multiple intermediaries. These intermediaries include wholesalers, distributors, brokers, and retailers who facilitate the movement of products between different stages of the supply chain. They handle activities such as purchasing, storing, marketing, and selling the food products.

2.1.5. Retailing: The retail stage involves the sale of food products to consumers. This can occur through supermarkets, grocery stores, restaurants, foodservice establishments, or online platforms. Retailers may source products directly from wholesalers or distributors, and they often add their margin to the product's cost.

2.2.Challenges in Traditional Food Supply Chains

Challenges and inefficiencies in traditional food supply chains include:

2.2.1. Long Transportation Distances: Food often travels long distances between production and consumption points, leading to increased costs, carbon emissions, energy consumption, and potential issues with product freshness and quality.

2.2.2. Excessive Reliance on Intermediaries: The involvement of multiple intermediaries adds complexity, increases costs, and can lead to longer lead times, communication challenges, and potential disruptions in the supply chain.

2.2.3. Lack of Transparency: Consumers often have limited visibility into the origin, production processes, and handling practices of the food they consume. This lack of information hampers consumer choice, raises concerns about food safety, and undermines trust in the supply chain.

2.2.4. Limited support for small-scale farmers: Traditional food supply chains often favor larger-scale producers and suppliers, making it challenging for small-scale farmers to access markets and compete on an equal footing. The dominance of larger players in the supply chain can lead to unfair pricing, reduced profitability, and limited opportunities for smaller producers to thrive.

Addressing these challenges and inefficiencies in traditional food supply chains is crucial for promoting sustainability, reducing environmental impact, ensuring food safety, and supporting a more inclusive and resilient food system. Efforts are underway to adopt alternative models such as direct-to-consumer, and shorter supply chains, as well as leveraging technology and implementing transparency measures to enhance efficiency, reduce waste, and improve the overall performance of food supply chains.

2.3. Need for Innovation and Transformation

The need for innovation and transformation in food supply chains arises from various challenges faced by the traditional systems. These challenges include:

2.3.1. Food safety and traceability: Foodborne illnesses and contamination outbreaks continue to pose risks to public health. Traditional supply chains often lack comprehensive traceability systems, making it difficult to identify the source of contamination and prevent its spread. Ensuring food safety and traceability is crucial for consumer confidence and regulatory compliance.

2.3.2. Rising consumer expectations: Consumers are increasingly demanding transparency, ethical sourcing, sustainability, and social responsibility from the food industry. Traditional supply chains face challenges in meeting these expectations and providing accurate and reliable information to consumers about the origin, quality, and production practices of the food they consume.

2.3.3. Technological advancements: Rapid advancements in technology offer opportunities for transforming food supply chains. Technologies such as blockchain, IoT, data analytics, and automation can improve traceability, reduce waste, optimize inventory management, enhance supply chain visibility, and facilitate real-time monitoring of food quality and safety.

2.3.4. Market access for small-scale producers: Traditional supply chains often marginalize small-scale farmers and producers, limiting their access to markets and fair prices. Innovation is needed to create inclusive supply chain models that empower and support small-scale producers, fostering economic sustainability and reducing inequality in the food system. To address these challenges, there is a growing need for innovation and transformation in food supply chains.

2.3.5. Adopting sustainable and regenerative practices: Embracing environmentally-friendly and resource-efficient farming techniques, reducing waste, and implementing circular economy principles.

2.3.6. Enhancing supply chain visibility and traceability: Leveraging technologies like blockchain and IoT to enable end-to-end traceability, ensuring transparency, and facilitating quick identification and resolution of issues related to food safety and quality.

2.3.7. Developing direct-to-consumer models: Facilitating direct connections between producers and consumers through e-commerce platforms, farmers' markets, and farm-to-table initiatives, reducing dependence on intermediaries and promoting local food systems.

2.3.8. Embracing data-driven decision-making: Utilizing data analytics and predictive modeling to optimize supply chain operations, improve demand forecasting, and reduce inefficiencies in inventory management and transportation.

CHAPTER 3

3. GROUND FRESH CONCEPT

3.1. Definition and Principles of Ground Fresh

Ground Fresh, is an approach in the food industry that focuses on creating a direct and short supply chain from agricultural producers to consumers, minimizing the involvement of intermediaries. The goal is to provide consumers with fresh, high-quality, and locally sourced food while promoting transparency, sustainability, and supporting local economies. Here are the key principles of the Ground Fresh approach:

3.1.1. Local Sourcing: Ground Fresh emphasizes sourcing food from local farmers and producers within a defined geographic region. The focus is on reducing transportation distances and supporting local agricultural communities. By sourcing locally, the supply chain minimizes carbon emissions, energy consumption, and environmental impact associated with long-distance shipping.

3.1.2. Direct Producer-Consumer Connection: A fundamental principle of Ground Fresh is establishing a direct relationship between producers and consumers. Farmers and producers have the opportunity to engage directly with consumers, sharing information about their farming practices, product quality, and handling processes. This direct connection fosters trust, transparency, and accountability in the food system.

3.1.3. Seasonal and Fresh Products: Ground Fresh emphasizes the use of seasonal produce and fresh products. By reducing the distance between production and consumption, food reaches consumers in a shorter time, maintaining its freshness, flavor, and nutritional value. This principle promotes a closer connection to the natural growing cycles, encourages biodiversity, and reduces the need for excessive processing and preservation methods.

3.1.4. Sustainability and Responsible Practices: The Ground Fresh approach promotes sustainable and responsible agricultural practices. This includes organic farming, regenerative agriculture, and reducing the use of synthetic chemicals. By prioritizing sustainable practices, the approach aims to protect soil health, water resources, and biodiversity, ensuring long-term environmental sustainability.

3.1.5. Traceability and Transparency: Ground Fresh emphasizes traceability and transparency throughout the supply chain. It involves providing consumers with information about the origin of the food, farming practices employed, certifications, and quality control measures. This transparency enables consumers to make informed choices, builds trust, and ensures accountability for the entire supply chain.

3.1.6. Community Support and Economic Resilience: The Ground Fresh approach supports local economies and communities. By sourcing from local farmers and producers, it helps sustain their livelihoods, creates economic opportunities, and promotes a vibrant local food system. This principle fosters community engagement, strengthens relationships between producers and consumers, and contributes to the overall well-being of the community.

Overall, the Ground Fresh approach focuses on shortening the distance between producers and consumers, promoting local sourcing, transparency, sustainability, and community support.

3.2. Evolution and Growth of Ground Fresh Movement

The evolution and growth of the ground fresh movement have been driven by changing consumer preferences, increased awareness about food sourcing, and advancements in technology. Here is a detailed overview of the evolution and growth of the ground fresh concept:

3.2.1. Emergence of Local Food Movements: The ground fresh movement emerged as part of the broader local food movements that gained momentum in the late 20th century. Consumers started showing increased interest in knowing where their food comes from, supporting local farmers, and having a closer connection to the food they consume.

3.2.2. Farm-to-Table Movement: The farm-to-table movement played a significant role in popularizing the ground fresh concept. It emphasized sourcing ingredients directly from local farmers and serving them in restaurants or through direct-to-consumer channels. This movement focused on freshness, quality, and supporting local communities.

3.2.3. Increasing Consumer Demand: As consumers became more conscious of the environmental and health impacts of their food choices, the demand for ground fresh products grew. Consumers sought out products that were minimally processed, locally sourced, and had a smaller carbon footprint. This increased demand created opportunities for producers to offer ground fresh options.

3.2.4. Technological Advancements: The growth of the ground fresh movement has been facilitated by advancements in technology. Online platforms, e-commerce, and mobile apps have made it easier for consumers to access and purchase ground fresh products directly from local farmers and producers. These technological advancements have improved convenience and expanded market reach.

3.2.5. Collaboration between Producers and Retailers: Producers and retailers have recognized the potential of the ground fresh concept and have collaborated to meet consumer demand. Retailers have started partnering with local farmers and producers to offer ground fresh products in their stores. This collaboration has allowed producers to access a wider market and has provided retailers with unique and high-quality product offerings.

3.2.6. Focus on Food Safety and Traceability: The ground fresh movement has also emphasized food safety and traceability. Consumers want to know the origin and handling practices of their food. Producers and retailers have responded by implementing stringent quality control measures, ensuring safe handling practices, and providing transparent information about the sourcing and processing of ground fresh products.

3.2.7. Expansion of Product Offerings: Initially, the ground fresh concept was primarily associated with fruits, vegetables, and herbs. However, the movement has expanded to include other product categories such as ground meats, dairy products, bakery items, and even ready-to-eat meals. This expansion has diversified the options available to consumers and has increased the market potential for ground fresh products

The ground fresh movement has evolved from a niche concept to a mainstream trend driven by consumer demand, technological advancements, collaborations between producers and retailers, a focus on food safety and traceability, and supportive policies. As consumers continue to prioritize freshness, quality, and sustainability in their food choices, the ground fresh movement is expected to further grow and innovate, offering consumers a wide range of locally sourced, minimally processed, and high-quality food options.

3.3. Benefits of Ground Fresh Model

The ground fresh model, within the Ground Fresh approach, offers several benefits to consumers, producers, and the overall food system. Here are some of the key benefits of the ground fresh model:

3.3.1. Superior Quality and Freshness: Ground fresh products are known for their superior quality and freshness. By sourcing directly from local farmers and producers, the time between harvest and consumption is minimized, ensuring that the products retain their optimal flavor, texture, and nutritional value. This results in a more satisfying and enjoyable eating experience for consumers.

3.3.2. Local and Sustainable Sourcing: The ground fresh model promotes local sourcing, supporting local farmers and producers within the community. This reduces the carbon footprint associated with long-distance transportation and supports local economies. Additionally, ground fresh products often adhere to sustainable farming practices, such as organic or regenerative agriculture, minimizing environmental impacts and preserving natural resources.

3.3.3. Enhanced Food Safety: The ground fresh model prioritizes food safety through direct sourcing and shorter supply chains. By minimizing the number of intermediaries, there is better control and traceability throughout the production and distribution process. This improves the ability to identify and address any potential issues related to food safety, reducing the risk of contamination and ensuring a safer food supply for consumers.

3.3.4. Transparent and Trustworthy Supply Chain: Ground fresh products offer transparency and traceability, providing consumers with information about the origin, production methods, and handling practices. This transparency fosters trust and confidence in the food system, allowing consumers to make informed choices about the food they consume. Knowing where their food comes from and how it is produced builds a stronger connection between consumers and producers.

3.3.5. Support for Local Communities: The ground fresh model contributes to the vitality of local communities by supporting local farmers and producers. By purchasing directly from local sources, consumers help sustain local agricultural economies, create job opportunities, and foster a sense of community. This strengthens the local food system, enhances food security, and reduces dependence on distant sources.

3.3.6. **Promotion of Culinary Diversity:** Ground fresh products often reflect the regional diversity and culinary traditions of a specific area. Consumers have the opportunity to access unique and locally grown ingredients that may not be readily available through conventional supply chains. This promotes culinary diversity and encourages exploration and appreciation of local flavors and food culture.

3.3.7. **Increased Consumer Engagement and Education:** The ground fresh model encourages consumer engagement and education about food and agriculture. Consumers have the opportunity to connect directly with farmers and producers, learn about farming practices, and gain insights into the local food system. This promotes a deeper understanding of food production, fosters appreciation for the efforts of local farmers, and encourages a more conscious approach to food choices.

The ground fresh model offers numerous benefits, including superior quality and freshness, local and sustainable sourcing, enhanced food safety, transparency in the supply chain, and support for local communities, promotion of culinary diversity, and increased consumer engagement and education. These benefits contribute to a more sustainable, resilient, and consumer-centric food system that prioritizes freshness, quality, and community well-being.

CHAPTER 4

4. ROLE OF TECHNOLOGY IN FOOD SUPPLY CHAINS

4.1. Overview of Mobile Applications in Agriculture

4.1.1. Mobile applications have transformed the agricultural industry by providing farmers, agricultural workers, and agribusinesses with powerful tools and functionalities. These applications offer a range of features such as crop management and monitoring, weather forecasting, market information, farm management and record-keeping, precision agriculture, knowledge sharing and collaboration, direct farm-to-consumer connections, and pest and disease management. By leveraging mobile technology, these applications enable farmers to make data-driven decisions, optimize their farming practices, improve productivity, and enhance sustainability. They provide access to real-time data, market trends, and weather information, empowering farmers to manage their operations more efficiently. Mobile applications also promote transparency, traceability, and direct connections between farmers and consumers, fostering a closer relationship and fair trade practices. As technology advances, mobile applications are expected to continue playing a vital role in revolutionizing agriculture, driving innovation, and improving the overall efficiency and sustainability of the industry. Conditions. As technology continues to advance, mobile applications are expected to play an increasingly vital role in shaping the future of agriculture.

- Farm-to-Consumer Connections (Just Like Ground Fresh)
- Precision Agriculture and Farm Mapping
- Farm Management and Record Keeping
- Market Information and Pricing
- Pest and Disease Management
- Knowledge Sharing and Collaboration etc.

4.2. Ground Fresh Apps: Functionality

Ground fresh apps, within the Ground Fresh concept, offer specific functionalities to connect consumers with locally sourced, minimally processed, and high-quality food products. These apps focus on providing a seamless experience for users seeking ground fresh options. Here are some key functionalities of ground fresh apps:

4.2.1. **Product Listings:** Farm-to-consumer apps allow farmers to create listings of their products, including detailed descriptions, images, and pricing information. Consumers can browse through these listings to explore the available products.

4.2.2. **Ordering and Purchasing**: These apps enable consumers to place orders for farm-fresh products directly from farmers. Consumers can select the desired quantity, add items to their cart, and proceed to checkout for payment. Some apps may also offer options for subscription-based purchases.

4.2.3. **Customization and Personalization:** Farm-to-consumer apps often provide customization options to cater to individual preferences. Consumers can choose specific product attributes, such as organic, non-GMO, or locally sourced, based on their preferences and dietary needs.

4.2.4. **Delivery and Pickup Options:** To enhance convenience, farm-to-consumer apps typically offer different delivery and pickup options. Consumers can select their preferred method, including home delivery, local pickup locations, or farmers' markets, based on their location and preferences.

4.2.5. **Real-time Updates and Notifications:** These apps keep consumers informed about order statuses, delivery schedules, and any updates from the farmers. Real-time notifications ensure that consumers stay updated on the progress of their orders.

4.2.6. **Farmer Profiles and Stories:** Farm-to-consumer apps often feature profiles and stories of the participating farmers. This allows consumers to learn more about the farmers, their farming practices, sustainability initiatives, and the story behind their products. It helps establish a personal connection between farmers and consumers.

4.2.7. **Ratings and Reviews:** Consumers can provide ratings and reviews for the products and farmers on the app. These reviews serve as valuable feedback for farmers and help other consumers make informed decisions when selecting products.

4.2.8. **Community Engagement:** Farm-to-consumer apps often foster a sense of community by facilitating interaction among farmers and consumers. They may include features such as forums, discussion boards, or social media integration to encourage communication, sharing of experiences, and the exchange of ideas.

4.3.Tools & Technology

4.3.1. VS Code: Visual Studio is a powerful integrated development environment (IDE) created by Microsoft. It provides a comprehensive set of tools and features for software development across different platforms, including desktop, web, mobile, cloud, and gaming. With Visual Studio, developers can write, debug, and test their code efficiently. It offers a user-friendly code editor with advanced features like syntax highlighting and IntelliSense for faster coding. Visual Studio also includes robust debugging capabilities, allowing developers to diagnose and fix issues in their code easily. The IDE supports collaboration among team members through Git integration and code review features. Visual Studio offers project management tools, including templates, version control integration, and project conFigureuration settings. It supports various programming languages and frameworks, making it suitable for a wide range of development projects. With its extensibility, developers can enhance the IDE's functionality by adding extensions and customizing their development environment. Visual Studio integrates with Microsoft Azure, enabling seamless development and deployment of cloud-based applications. Whether you're developing applications for Windows, macOS, Linux, iOS, Android, or the web, Visual Studio provides a versatile and efficient development environment.

4.3.2. Flutter: Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. It allows developers to write code once and deploy it on multiple platforms. Flutter uses the Dart programming language, which is also developed by Google. With Flutter, developers can create high-performance, visually appealing, and fast applications with a reactive and flexible UI framework. It provides a rich set of customizable widgets and an extensive set of libraries and tools. Flutter's hot reload feature enables real-time code changes and instant UI updates, making the development process faster and more efficient. The framework also offers built-in testing and debugging tools, making it easier to identify and fix issues. Flutter has gained popularity for its cross-platform capabilities, its ability to deliver native-like performance, and its vibrant developer community. It is widely used by developers to create stunning and responsive user interfaces for mobile, web, and desktop applications.

4.3.3. Dart: Dart is a programming language developed by Google. It is primarily used for building cross-platform mobile, web, and desktop applications. Dart is known for its simplicity, scalability, and performance. It combines the best features of object-oriented programming (OOP) and functional programming (FP) paradigms. Dart is statically typed, which helps catch errors during compile-time and improves code quality. It also supports (JIT) and (AOT) compilation, allowing developers to optimize their code for different scenarios. Dart has a rich set of libraries and frameworks, such as Flutter, which is a popular UI toolkit for building beautiful and high-performance applications. Dart's focus on productivity and maintainability makes it a preferred choice for developers looking to create robust and efficient applications across different platforms.

4.3.4. Firebase: Firebase is a comprehensive mobile and web development platform provided by Google. It offers a set of backend services and tools that help developers build, deploy, and scale applications more efficiently. Firebase provides various features, including real-time database, cloud storage, authentication, hosting, cloud functions, and analytics. With Firebase, developers can quickly set up and manage backend services without the need for extensive server infrastructure or conFigureuration. The real-time database enables real-time synchronization of data across clients, while the cloud storage allows for secure and scalable file storage. Firebase's authentication service provides ready-to-use user authentication and identity management features, making it easy to implement secure user authentication in applications. The platform also offers hosting services to deploy web applications with ease. Cloud functions allow developers to run server-side code in a server less environment, while analytics provides insights into user behaviour and app performance. Firebase supports seamless integration with other Google Cloud services, enabling developers to leverage the full power of the Google Cloud ecosystem. With its ease of use, scalability, and extensive feature set, Firebase has become a popular choice for developers to build robust and feature-rich mobile and web applications.

4.3.5. Figma: Figma is a cloud-based design and prototyping tool used by designers and design teams to create user interfaces (UI) and collaborate on design projects. It provides a comprehensive platform for designing, prototyping, and sharing interactive and responsive designs. With Figma, designers can create and edit designs in real-time, making it easy for multiple team members to collaborate on the same project simultaneously. Figma offers a range of features such as vector editing tools, design components, and design version control. It allows designers to create interactive prototypes with animations and transitions, enabling stakeholders to experience the user flow and interaction of the design. Figma's cloud-based nature ensures that designs are accessible from anywhere, facilitating seamless collaboration and feedback gathering. It also offers features for design handoff, allowing developers to inspect and extract design assets for implementation. Figma's popularity is attributed to its intuitive interface, collaborative capabilities, and platform-agnostic nature, making it a preferred choice for designers and design teams to streamline their design workflow and create visually appealing and interactive designs.

4.4.Process Flow, DFD, ERD

4.4.1. Process Flow: A process flow, also known as a workflow or process diagram, is a visual representation of the steps or activities involved in completing a particular process or achieving a specific outcome. It helps to illustrate the sequence of actions, decisions, and interactions among various entities or components within the process.

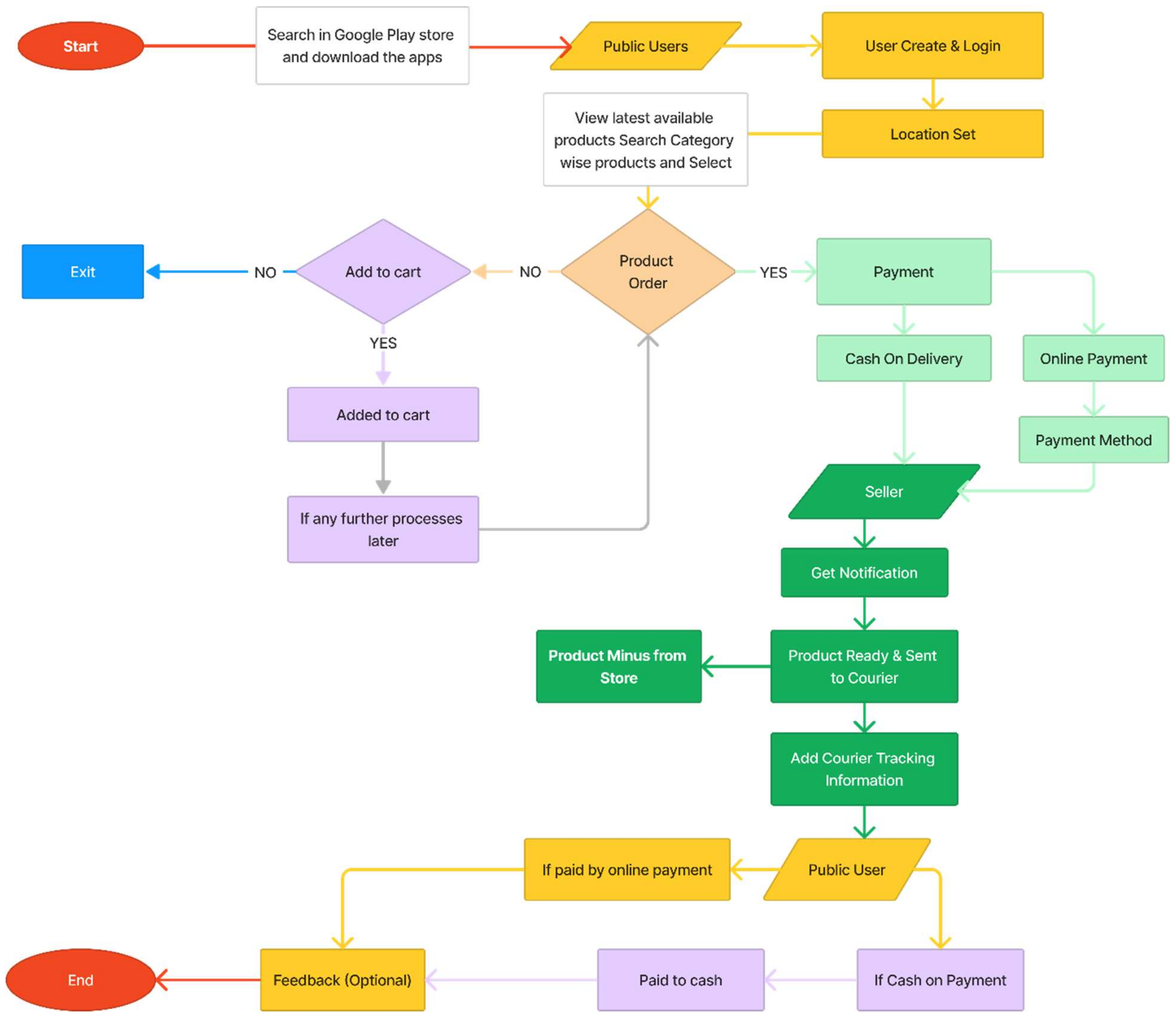


Fig 1: Process Flow of Ground Fresh

4.4.2. DFD: A Data Flow Diagram (DFD) is a visual representation that illustrates how data moves within a system or process. It uses symbols and arrows to show the flow of data between external entities, processes, and data stores. The external entities are sources or destinations of data, processes represent actions or transformations performed on the data, and data stores are where data is stored or retrieved. DFDs help in understanding the data flow, identifying interactions between different components, and providing a clear overview of the system's information exchange. They are commonly used in software engineering and systems analysis to analyze, design, and communicate system architectures and data dependencies.

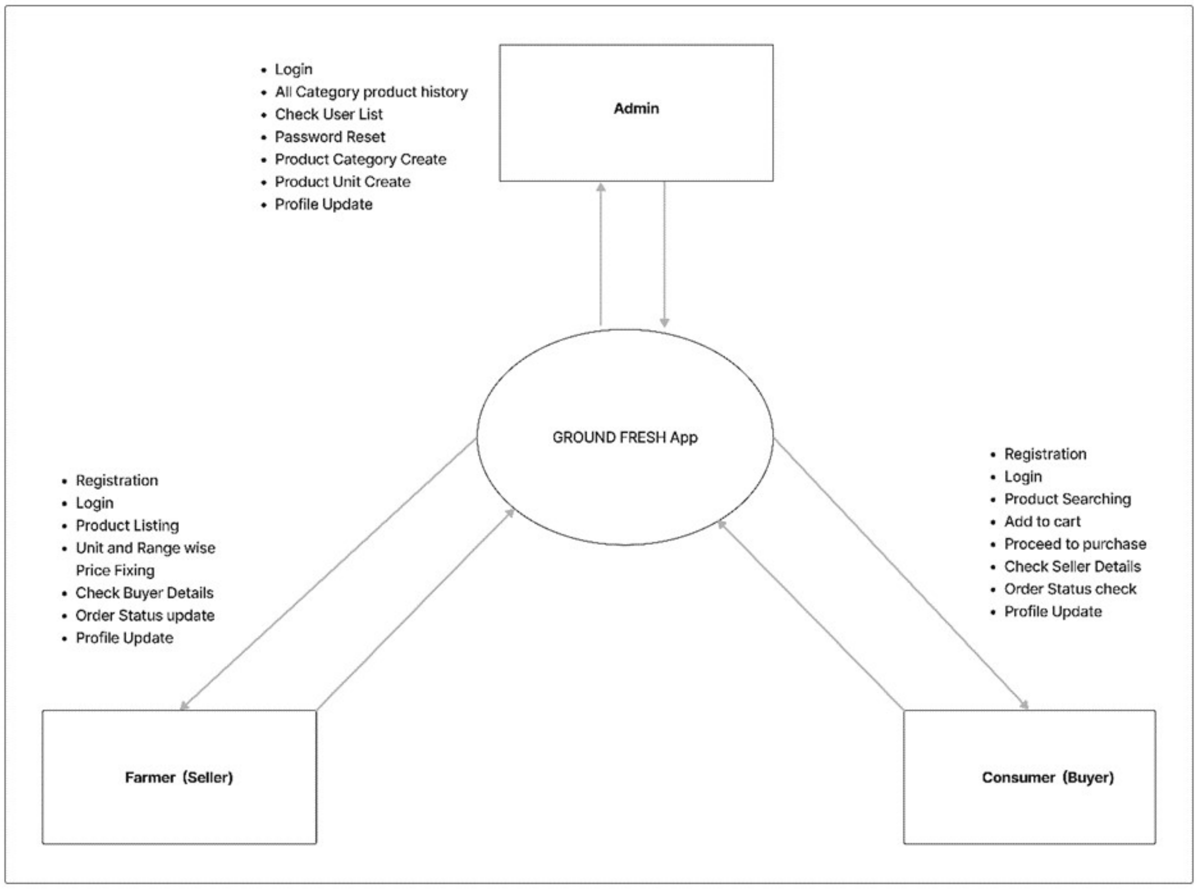


Fig 2: DFD of Ground Fresh

4.4.3. ERD: An Entity-Relationship Diagram (ERD) is a visual representation of the relationships between entities (objects or concepts) within a system or database. It depicts how different entities are connected through relationships and attributes. Entities are represented as rectangles, relationships are shown as lines connecting the entities, and attributes are listed within the entities. ERDs provide a clear and concise overview of the structure and organization of a system's data. They help in designing and analyzing databases, identifying key entities and their relationships, and ensuring data integrity and consistency. ERDs are widely used in database design and systems analysis to model complex systems and facilitate effective communication among stakeholders involved in system development.

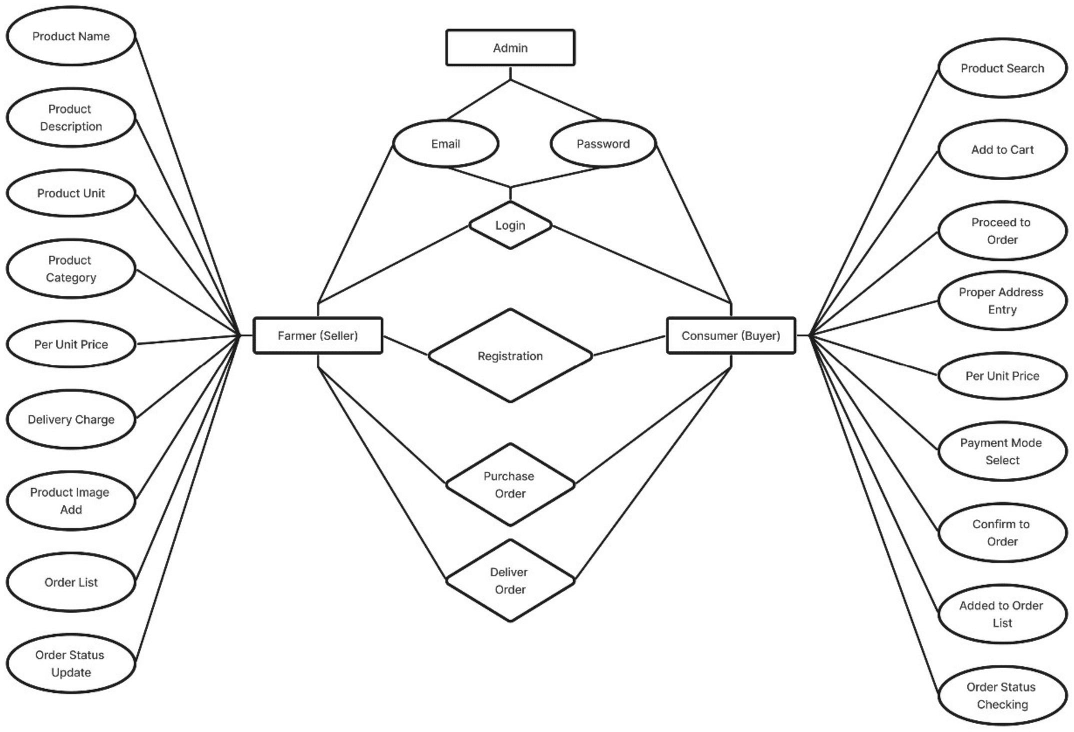


Fig 3: ERD of Ground Fresh

4.4.4. Use Case: A Use Case is a concise and descriptive representation of how a system or software application interacts with its users or external entities to achieve a specific goal or perform a particular function. It outlines the step-by-step interactions and behaviors between the user and the system, capturing the various scenarios and actions involved in a user's interaction with the system. Use cases typically consist of a title, a brief description, a list of steps or actions, and potential outcomes. They serve as a means to understand and define system requirements, validate system functionality, and guide the development and testing process. Use cases are an essential tool in software development and systems analysis, providing a clear and structured way to document and communicate user interactions and system behaviour.

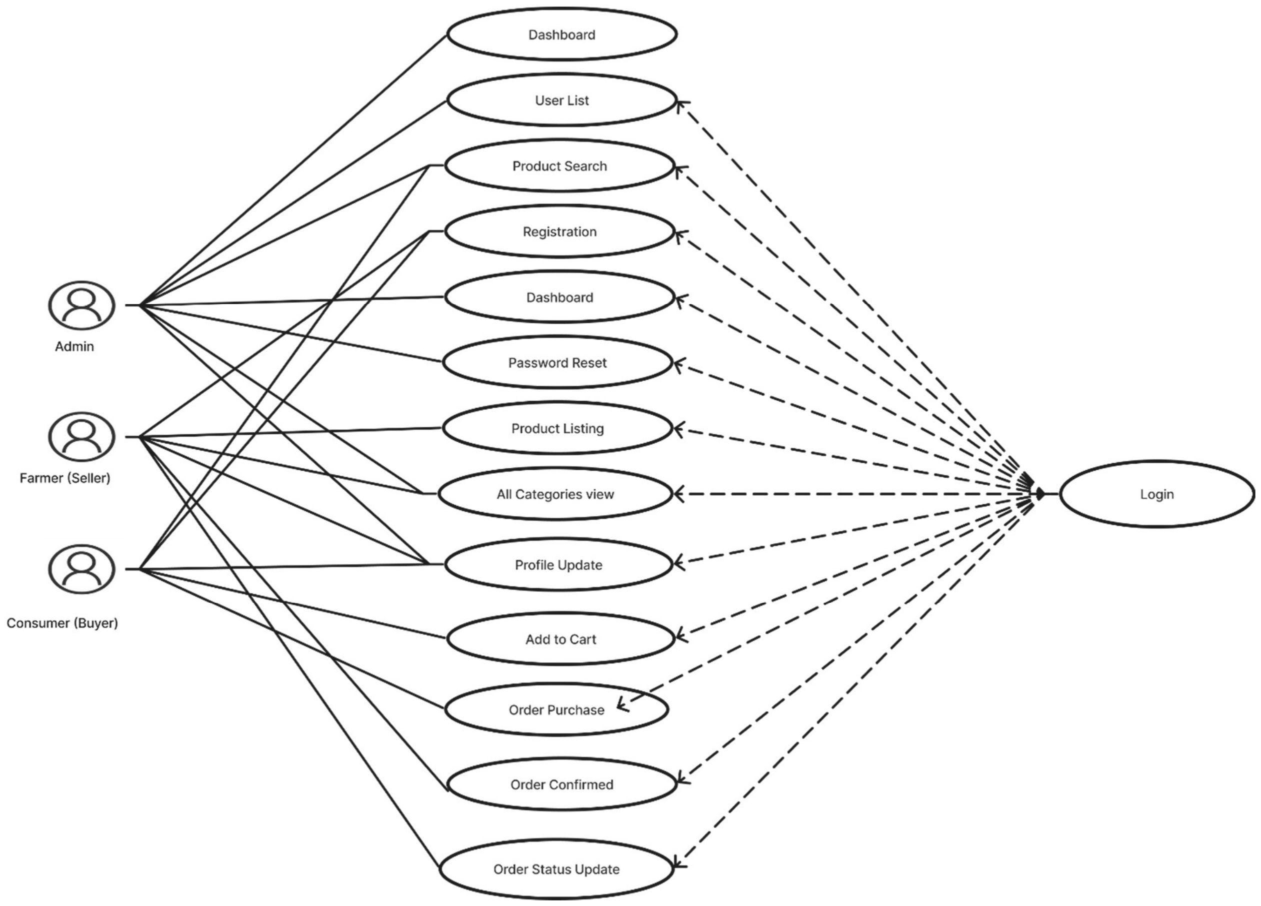
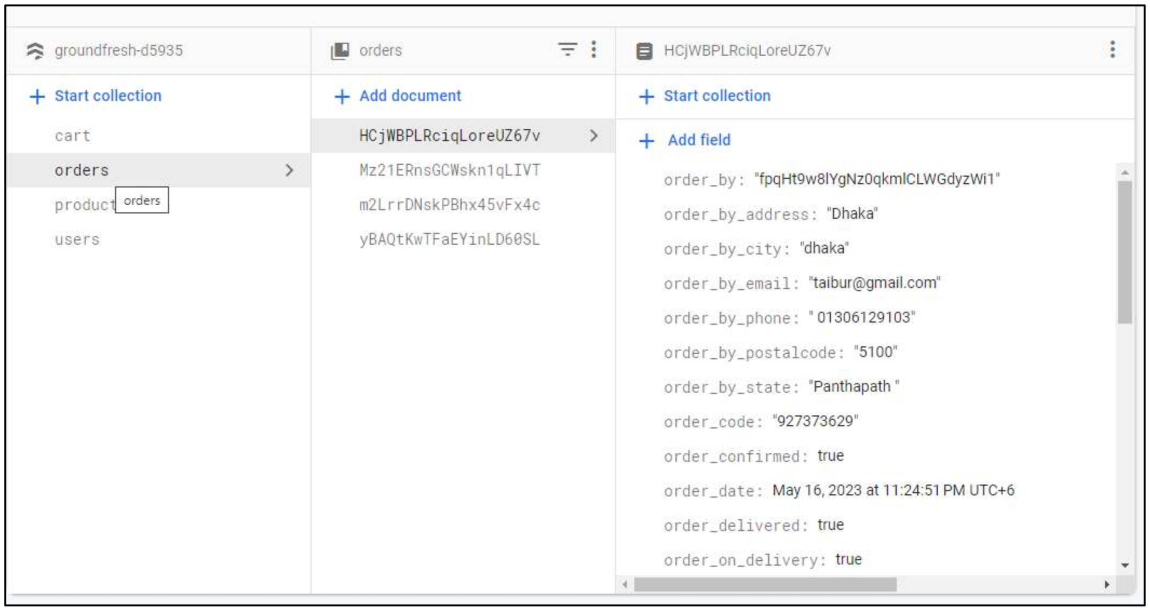


Fig 4: Use Case of Ground Fresh

4.5.Table on Firebase (Database)

4.5.1. Cart Details



groundfresh-d5935

orders

HCjWBPLRciqLoreUZ67v

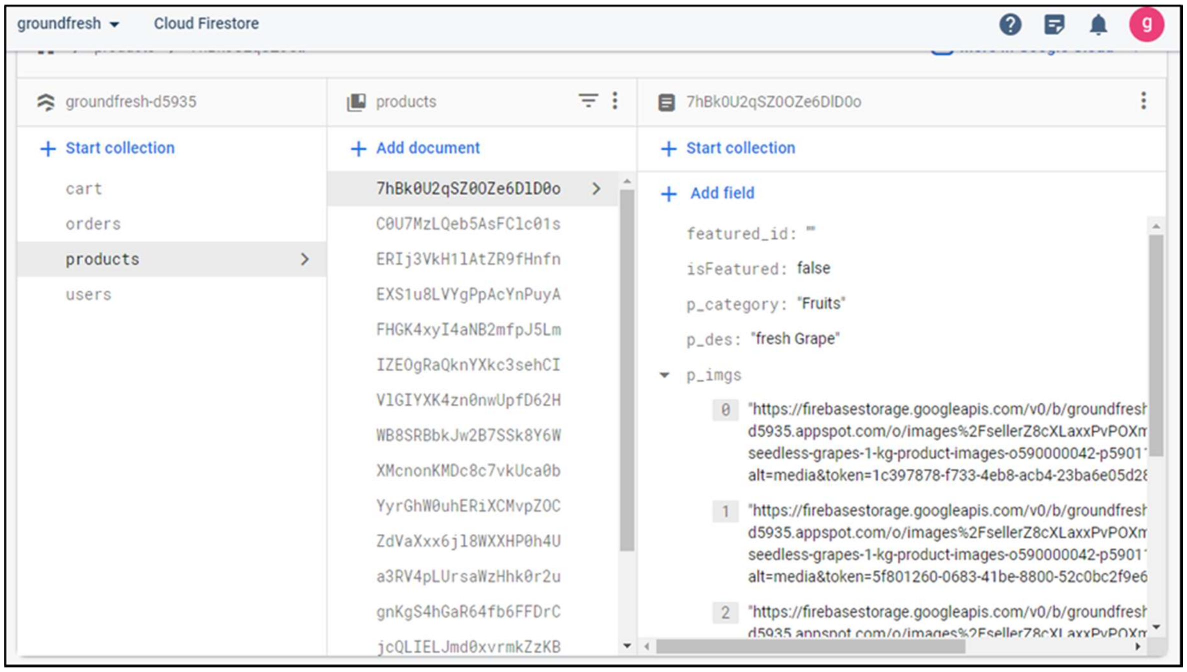
cart
orders
products
users

HCjWBPLRciqLoreUZ67v
Mz21ERnsGCWskn1qLIVT
m2LrrDNskPBhx45vFx4c
yBAQtKwTFaEYinLD60SL

```
order_by: "fpqHt9w8lYgNz0qkmlCLWGdyzWi1"
order_by_address: "Dhaka"
order_by_city: "dhaka"
order_by_email: "taibur@gmail.com"
order_by_phone: "01306129103"
order_by_postalcode: "5100"
order_by_state: "Panthapath "
order_code: "927373629"
order_confirmed: true
order_date: May 16, 2023 at 11:24:51 PM UTC+6
order_delivered: true
order_on_delivery: true
```

Fig 5: Cart Details

4.5.2. Product List



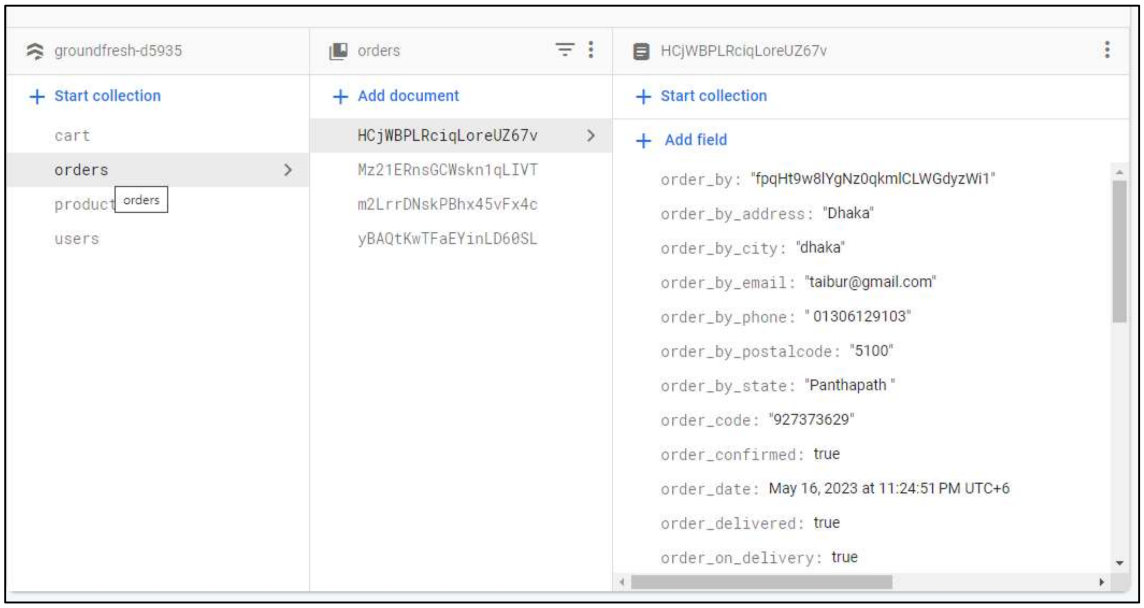
```
7hBk0U2qSZ0OZe6DlD0o

7hBk0U2qSZ0OZe6D1D0o
C0U7MzLQeb5AsFClc01s
ERIj3VkH1lAtZR9fHnfn
EXS1u8LVYgPpAcYnPuyA
FHGK4xyI4aNB2mfpJ5Lm
IZEOgRaQknYXkc3sehCI
V1GIYXK4zn0nwUpfD62H
WB8SRBbkJw2B7SSk8Y6W
XMcnonKMDc8c7vkUca0b
YyrGhW0uhERiXCMvpZOC
ZdVaXxx6jl8WXXHP0h4U
a3RV4pLUrsaWzHhk0r2u
gnKgS4hGaR64fb6FFDrC
jcQLIELJmd0xvrmkZzKB

featured_id: ""
isFeatured: false
p_category: "Fruits"
p_des: "fresh Grape"
p_imgs
  0 "https://firebasestorage.googleapis.com/v0/b/groundfresh
    d5935.appspot.com/o/images%2FsellerZ8cXLaxxPvPOXm
    seedless-grapes-1-kg-product-images-o590000042-p5901
    alt=media&token=1c397878-f733-4eb8-acb4-23ba6e05d2
  1 "https://firebasestorage.googleapis.com/v0/b/groundfresh
    d5935.appspot.com/o/images%2FsellerZ8cXLaxxPvPOXm
    seedless-grapes-1-kg-product-images-o590000042-p5901
    alt=media&token=5f801260-0683-41be-8800-52c0bc2f9e6
  2 "https://firebasestorage.googleapis.com/v0/b/groundfresh
```

Fig 6: Product List

4.5.3. Order Details:



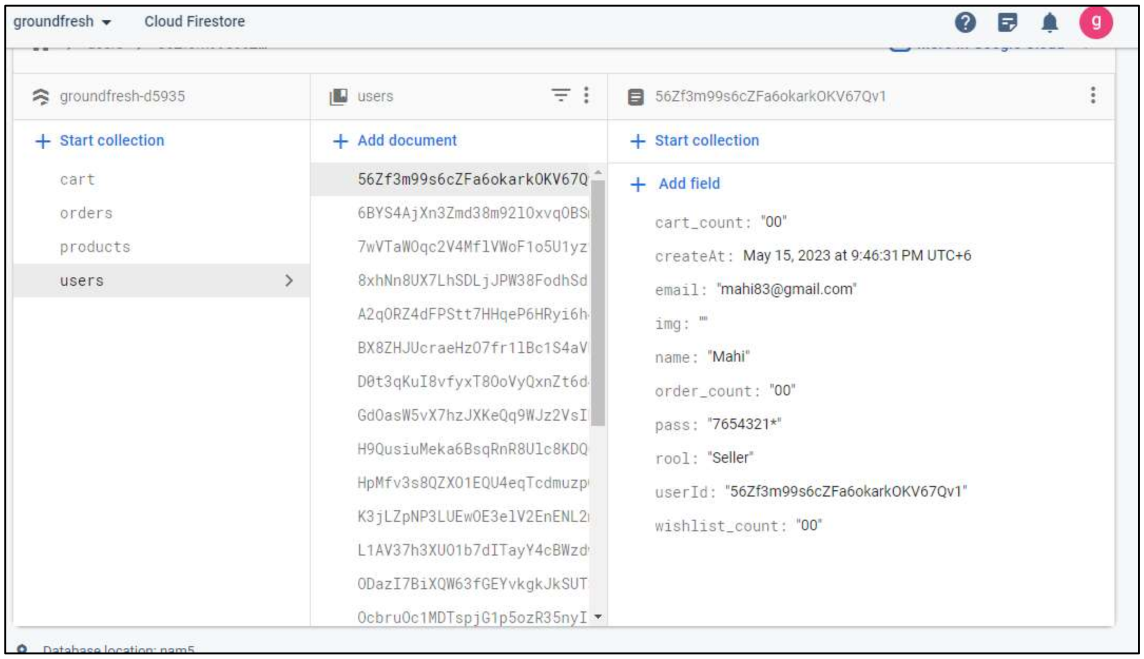
```
HCjWBPLRciqLoreUZ67v

HCjWBPLRciqLoreUZ67v
Mz21ERnsGCWskn1qLIVT
m2LrrDNskPBhx45vFx4c
yBAQtKwTFaEYinLD60SL

order_by: "fpqHt9w8lYgNz0qkmlCLWGdyzWi1"
order_by_address: "Dhaka"
order_by_city: "dhaka"
order_by_email: "taibur@gmail.com"
order_by_phone: " 01306129103"
order_by_postalcode: "5100"
order_by_state: "Panthapath "
order_code: "927373629"
order_confirmed: true
order_date: May 16, 2023 at 11:24:51 PM UTC+6
order_delivered: true
order_on_delivery: true
```

Fig 7: Order Details

4.5.4. User List:


```
groundfresh    Cloud Firestore

groundfresh-d5935          users                           56Zf3m99s6cZFa6okarkOKV67Qv1
+ Start collection         + Add document                  + Start collection
cart                       56Zf3m99s6cZFa6okarkOKV67Q      + Add field
orders                     6BYS4AjXn3Zmd38m921OxvqOBS      cart_count: "00"
products                   7wVTaWOqc2V4MflVWoF1o5U1yz      createAt: May 15, 2023 at 9:46:31 PM UTC+6
users                      8xhNn8UX7LhSDLjJPW38FodhSd      email: "mahi83@gmail.com"
                           A2qORZ4dFPStt7HHqeP6HRyi6h      img: ""
                           BX8ZHJUcraeHz07fr1lBc1S4aV      name: "Mahi"
                           D0t3qKuI8vfyxT8OoVyQxnZt6d      order_count: "00"
                           GdOasW5vX7hzJXKeQq9WJz2VsI      pass: "7654321*"
                           H9QusiuMeka6BsqRnR8Ulc8KDQ      rool: "Seller"
                           HpMfv3s8QZX01EQU4eqTcdmuzp      userId: "56Zf3m99s6cZFa6okarkOKV67Qv1"
                           K3jLZpNP3LUEwOE3elV2EnENL2      wishlist_count: "00"
                           L1AV37h3XU01b7dITayY4cBWzd
                           ODazI7BiXQW63fGEYvkgkJkSUT
                           Ocbru0c1MDTspjG1p5ozR35nyI
```

Fig 8: User List

4.6.Ground Fresh App Test Cases

Test Case No.	Test Case ID	Test Case Title	Test Steps	Expected Results	Status
1.	GFA-001	App Installation and Launch	Install the Ground Fresh app from the respective app store.	The app should be installed without any errors.	Passed
			Launch the Ground Fresh app.	The app should launch successfully.	Passed
2.	GFA-002	User Registration	Launch the Ground Fresh app.	The user should be successfully registered.	Passed
			Tap on the "Sign Up" or "Register" button.		
			Enter valid user details in the registration form.	The user should be redirected to the app's home screen or login page.	Passed
			Tap on the "Register" or "Create Account" button.		
3.	GFA-003		Launch the Ground Fresh app.	The user should be successfully logged in.	Passed
			Enter valid login credentials.		
			Tap on the "Login" or "Sign In" button.	The user should be directed to the app's home screen.	Passed

4.	GFA-004	Forgot Password	Launch the Ground Fresh app. Tap on the "Forgot Password" link on the login page. Enter the registered email address. Tap on the "Reset Password" button.	A password reset link should be sent to the user's registered email address.	Passed
5.	GFA-005	View Product Categories	Launch the Ground Fresh app. Login using valid credentials. Navigate to the product categories section.	The app should display a list of available product categories.	Passed
6.	GFA-006	View Products within a Category	Launch the Ground Fresh app. Login using valid credentials. Select a specific product category. View the list of products within the selected category.	The app should display a list of products within the selected category.	Failed
7.	GFA-007	Search for Products	Launch the Ground Fresh app. Login using valid credentials. Enter a specific product name or keyword in the search bar. Tap on the search button or press Enter.	The app should display relevant search results for the entered product name or keyword.	Passed
8.	GFA-008	Add Product to Cart	Launch the Ground Fresh app. Login using valid credentials. Search for a specific product. Tap on the desired product from the search results. On the product details page, tap on the "Add to Cart" button.	The selected product should be successfully added to the user's cart.	Passed
9.	GFA-009	Remove Product from Cart	Launch the Ground Fresh app. Login using valid credentials.	The selected product should be successfully removed from the user's cart.	Passed

			View the contents of the user's cart.		
			Select a product in the cart to remove.		
			Tap on the "Remove" or "Delete" button.		
10.	GFA-010	View Cart Details	Launch the Ground Fresh app.	The app should display the details of the user's cart, including the products, quantities, and total price.	Passed
			Login using valid credentials.		
			View the contents of the user's cart.		
11.	GFA-011	Place an Order	Launch the Ground Fresh app.	The order should be successfully placed, and the user should receive an order confirmation.	Passed
			Login using valid credentials.		
			Add products to the cart.		
			View the cart details.		
			Proceed to checkout.		
			Enter the delivery address and payment details.		
			Confirm the order.		
12.	GFA-012	View Order History	Launch the Ground Fresh app.	The app should display the user's order history, including previous orders and their details.	Passed
			Login using valid credentials.		
			Navigate to the order history section in profile.		
13.	GFA-013	View Store Locations	Launch the Ground Fresh app.	The app should display store Locations	Passed
			Login using valid credentials.		
			Show the store locations in product Details		
14.	GFA-014	View Deals and Offers	Launch the Ground Fresh app.	The app should display the available deals and offers.	
			Login using valid credentials.		
			Navigate to the deals and offers section.		
15.	GFA-015	Logout	Launch the Ground Fresh app.	The user should be successfully logged out of the app.	Passed
			Login using valid credentials.		

			Logout or sign out message shown when click Back Button.		
			If Yes then logged out		
			If No then stay in.	The user should be continue using the app.	Passed

Continue process

Table 4.1: Test Case of Ground Fresh Apps

CHAPTER 5

5. IMPLEMENTATION AND INTERFACE

5.1. Implementation

To implement a ground-up fresh app using Flutter, VS Code, and Firebase, we are follow these general steps:

- **Set up Flutter and VS Code:**
  - **Install Flutter SDK:** Visit the Flutter website (https://flutter.dev/) and follow the installation guide for your operating system.
  - **Install VS Code:** At First, Download and install Visual Studio Code (https://code.visualstudio.com/), which is a lightweight code editor.

  Install the Flutter extension: Launch VS Code, go to the Extensions tab (Ctrl+Shift+X), search for "Flutter," and install the official Flutter extension by Dart Code.

- **Create a new Flutter project:**

  Open VS Code and create a new Flutter project by running the command palette (Ctrl+Shift+P) and typing "Flutter: New Project." Provide a project name and location for your app. Wait for VS Code to create the project and set up the necessary files and folders.

- **ConFigureure Firebase:**

  Go to the Firebase website (https://firebase.google.com/) and create a new project.

  Set up Firebase Authentication: Enable Firebase Authentication in your project settings and conFigureure the desired authentication methods (e.g., email/password, Google Sign-In).

  Set up Firebase Firestore: Enable Firestore as your database, and define the necessary collections and fields for your app's data.

- **Connect Flutter app to Firebase:**

  Add the Firebase Flutter packages: Open the pubspec.yaml file in your Flutter project, and add the necessary Firebase packages like firebase_core, firebase_auth, and cloud_firestore. Save the file.

  Run flutter pub get in the terminal to fetch the packages and update your project.

  Initialize Firebase in your Flutter app: Open the main.dart file and import the necessary Firebase packages. Initialize Firebase using Firebase.initializeApp(). This step connects your app to the Firebase project.

- **Build app screens and functionality:**

  Design app's screens and UI using Flutter's widgets and layouts.

  Implement authentication features: Create login and registration screens, handle user authentication using Firebase Authentication, and manage user sessions.

  Integrate Firestore for data management: Implement CRUD (Create, Read, Update, and Delete) operations using Firestore to store and retrieve data.

  Implement additional features and functionality specific to your app's requirements.

- **Test and debug:**

  Run app in the emulator or on a physical device to test its functionality.

  Use debugging tools provided by VS Code and Flutter to identify and fix any issues or errors.

- **Deploy app:**

  ConFigureure app for deployment on different platforms (iOS, Android).

  Generate release builds for distribution or publishing to the respective app stores.

5.2. Code Annotations

5.2.1. main file:

In the context of Flutter, main.dart typically refers to the main entry point file of a Flutter application. It is a Dart programming language file that serves as the starting point of the application's execution.

When we create a new Flutter project, the main.dart file is automatically generated and placed in the lib directory of your project. This file contains the main() function, which is the first function called when the application is launched. The main() function typically calls the runApp() function, which initializes and runs the Flutter application.

main.dart file for Ground Fresh Apps:

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

import 'package:get/get.dart';
import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/views/splash_screen/splash_screen.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';

Future<void> main() async {
  SystemChrome.setSystemUIOverlayStyle(const SystemUiOverlayStyle(
    statusBarColor: scaffoldColor,
    statusBarIconBrightness: Brightness.dark,
    systemNavigationBarColor: scaffoldColor,
  ));
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );

  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Ground Fresh',
      theme: ThemeData(
          scaffoldBackgroundColor: scaffoldColor,
          appBarTheme: const AppBarTheme(
            iconTheme: IconThemeData(color: darkFontGrey),
            elevation: 0,
            backgroundColor: scaffoldColor,
          ),
          fontFamily: regular),
      //home: const LoginScreen(),
      home: const SplashScreen(),
    );
  }
}
```

5.2.2. auth_screen:

An authentication screen, also known as a login screen or sign-in screen, is a user interface that allows users to authenticate themselves before accessing an application or a specific set of features. It typically consists of input fields for username/email and password, along with buttons for signing in or creating a new account.

Auth Screen file of Ground Fresh Apps: There are login screen, Sign Up Screen & Terms & Conditions.

5.2.2.1. login_screen:

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

import 'package:firebase_auth/firebase_auth.dart';
import 'package:get/get.dart';
import 'package:groundfresh/adimn/admin_home.dart';
import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/seller/seller_home.dart';
import 'package:groundfresh/views/auth_screen/signup_screen.dart';
import 'package:groundfresh/views/controllers/auth_controller.dart';
import 'package:groundfresh/views/home_screen/dashboard_screen.dart';
import 'package:groundfresh/widget_common/appLogo_widget.dart';
import 'package:groundfresh/widget_common/customTextField.dart';
import 'package:groundfresh/widget_common/ourButton.dart';

class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  var emailcontroller = TextEditingController();
  var passwordcontroller = TextEditingController();

  @override
  Widget build(BuildContext context) {
    var controller = Get.put(AuthController());
    return Scaffold(
      body: Container(
        height: context.screenHeight,
        width: context.screenWidth,
        color: greencolor,
        child: SingleChildScrollView(
          physics: const BouncingScrollPhysics(),
          child: Column(
            children: [
              Container(
                  height: context.screenHeight / 3,
                  child: Column(
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                      appLogoWidget(),
                      10.heightBox,
                      loginto.text.fontFamily(semibold).white.make()
        ],
      )),
  Container(
    height: 700,
    padding:
        const EdgeInsets.symmetric(horizontal: 20, vertical: 10),
    decoration: const BoxDecoration(
      color: whiteColor,
      borderRadius: BorderRadius.only(
          topLeft: Radius.circular(15),
          topRight: Radius.circular(15)),
    ),
    child: Obx(
      () => Column(
        children: [
          customTextFeild(
              hint: emailhint,
              title: email,
              isPass: false,
              controller: emailcontroller),
          5.heightBox,
          customTextFeild(
              hint: passwordhint,
              title: password,
              isPass: true,
              controller: passwordcontroller),
          5.heightBox,
          Align(
            alignment: Alignment.centerRight,
            child: TextButton(
              onPressed: () {},
              child: forgetpass.text.color(greencolor).make(),
            ),
          ),
          5.heightBox,
          controller.isLoding.value
              ? const CircularProgressIndicator(
                  color: greencolor,
                )
              : ourButton(
                  color: greencolor,
                  textColor: whiteColor,
                  title: login,
                  onPress: () {
                    var email = emailcontroller.text.trim();
```

```
                              var pass = passwordcontroller.text.trim();
                              signIn(email, pass);
                             }).box.width(context.screenHeight - 50).make(),
                        5.heightBox,
                        createAnAccount.text.color(fontGrey).make(),
                        5.heightBox,
                        ourButton(
                          color: scaffoldColor,
                          textColor: greencolor,
                          title: signup,
                          onPress: () {
                            Get.to(() => const SignUpScreen());
                          }).box.width(context.screenHeight - 50).make(),
                      ],
                    ),
                  ),
                )
              ],
            ),
          ),
        ),
      );
    }

    void route() {
      User? user = FirebaseAuth.instance.currentUser;
      var kk = FirebaseFirestore.instance
          .collection('users')
          .doc(user!.uid)
          .get()
          .then((DocumentSnapshot documentSnapshot) {
        if (documentSnapshot.exists) {
          if (documentSnapshot.get('rool') == "Buyer") {
            Navigator.pushReplacement(
              context,
              MaterialPageRoute(
                builder: (context) => const DashboardScreen(),
              ),
            );
          } else if (documentSnapshot.get('rool') == "Seller") {
            Navigator.pushReplacement(
              context,
              MaterialPageRoute(
                builder: (context) => const SellerHome(),
              ),
            );
          } else {
            Navigator.pushReplacement(
              context,
              MaterialPageRoute(
                builder: (context) => const AdminHome(),
              ),
            );
          }
        } else {
          VxToast.show(context, msg: 'Document does not exist on the database');
        }
      });
    }

    void signIn(String email, String password) async {
      try {
        UserCredential userCredential =
            await FirebaseAuth.instance.signInWithEmailAndPassword(
          email: email,
          password: password,
        );
        route();
      } on FirebaseAuthException catch (e) {
        VxToast.show(context, msg: "wrong password");
      }
    }
  }
```

5.2.2.2. sign_up screen:

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

import 'package:get/get.dart';
import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/views/auth_screen/login_screen.dart';
import 'package:groundfresh/views/auth_screen/trems_and_condition.dart';
import 'package:groundfresh/views/controllers/auth_controller.dart';
import 'package:groundfresh/widget_common/appLogo_widget.dart';
import 'package:groundfresh/widget_common/customTextField.dart';
import 'package:groundfresh/widget_common/ourButton.dart';

class SignUpScreen extends StatefulWidget {
  const SignUpScreen({super.key});

  @override
  State<SignUpScreen> createState() => _SignUpScreenState();
```

```
}

class _SignUpScreenState extends State<SignUpScreen> {
 bool? isCheck = false;
 // var controller = Get.put(AuthController());
 var namecontroller = TextEditingController();
 var emailcontroller = TextEditingController();
 var passwordcontroller = TextEditingController();
 var retyppasswordcontroller = TextEditingController();

 var options = [
  'Buyer',
  'Seller',
 ];
 var _currentItemSelected = "Buyer";
 var rool = "Buyer";

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    body: Container(
     height: context.screenHeight,
     width: context.screenWidth,
     color: greencolor,
     child: SingleChildScrollView(
      child: Column(
       children: [
        Container(
          height: context.screenHeight / 3,
          child: Column(
           mainAxisAlignment: MainAxisAlignment.center,
           children: [
            appLogoWidget(),
            5.heightBox,
            "Sign Up to $appname"
              .text
              .fontFamily(semibold)
              .white
              .make()
           ],
          )),
        Container(
         height: 700,
         padding:
           const EdgeInsets.symmetric(horizontal: 20, vertical: 20),
decoration: const BoxDecoration(
  color: whiteColor,
  borderRadius: BorderRadius.only(
      topLeft: Radius.circular(15),
      topRight: Radius.circular(15)),
),
child: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    DropdownButton<String>(
      dropdownColor: whiteColor,
      isExpanded: true,
      iconEnabledColor: Colors.white,
      focusColor: greencolor,
      items: options.map((String dropDownStringItem) {
        return DropdownMenuItem<String>(
          value: dropDownStringItem,
          child: Text(
            dropDownStringItem,
            style: const TextStyle(
              color: greencolor,
              fontFamily: semibold,
              fontSize: 16,
            ),
          ),
        );
      }).toList(),
      onChanged: (newValueSelected) {
        setState(() {
          _currentItemSelected = newValueSelected!;
          rool = newValueSelected;
        });
      },
      value: _currentItemSelected,
    ).box.rounded.make(),
    10.heightBox,
    customTextFeild(
        hint: namehint,
        title: name,
        controller: namecontroller,
        isPass: false),
    5.heightBox,
    customTextFeild(
        keyboardType: TextInputType.emailAddress,
        hint: emailhint,
    title: email,
    controller: emailcontroller,
    isPass: false),
5.heightBox,
customTextFeild(
    hint: passwordhint,
    title: password,
    controller: passwordcontroller,
    isPass: true),
5.heightBox,
customTextFeild(
    validator: (value) {
      if (retyppasswordcontroller.text !=
          passwordcontroller.text) {
        return "Password did not match";
      } else {
        return null;
      }
    },
    hint: retypepass,
    title: retypepass,
    controller: retyppasswordcontroller,
    isPass: true),
10.heightBox,
Row(
  children: [
    Checkbox(
        activeColor: greencolor,
        value: isCheck,
        onChanged: (newValue) {
          setState(() {
            isCheck = newValue;
          });
        }),
    5.widthBox,
    Expanded(
      child: InkWell(
        onTap: () {
          Get.to(() => TermAndCondition());
        },
        child: RichText(
            text: const TextSpan(children: [
          TextSpan(
              text: "I agree to the ",
              style: TextStyle(
                fontFamily: regular, color: fontGrey)),
          TextSpan(
            text: termAndCondition,
            style: TextStyle(
                fontFamily: regular, color: greencolor)),
          TextSpan(
            text: "&",
            style: TextStyle(
                fontFamily: regular, color: fontGrey)),
          TextSpan(
            text: privacyPolicy,
            style: TextStyle(
                fontFamily: regular, color: greencolor))
        ])),
      ),
    )
  ],
),
10.heightBox,
ourButton(
    color: isCheck == true ? greencolor : scaffoldColor,
    textColor: whiteColor,
    title: signup,
    onPress: () async {
      var email = emailcontroller.text.trim();
      var name = namecontroller.text.trim();
      var pass = passwordcontroller.text.trim();

      if (isCheck != false) {
        FirebaseAuth.instance
            .createUserWithEmailAndPassword(
                email: email, password: pass)
            .then((value) => {
                  storageData(
                      name, pass, email, rool, context)
                })
            .then((value) {
          VxToast.show(context, msg: "Successfully signup");
          Get.to(() => const LoginScreen());
        });
      }
    }).box.width(context.screenHeight - 50).make(),
15.heightBox,
Row(
  children: [
                  alreadyhaveanAccount.text.color(fontGrey).make(),
                  TextButton(
                    onPressed: () {
                      Get.to(() => const LoginScreen());
                    },
                    child: login.text.color(greencolor).make())
                 ],
                )
               ],
              ),
             )
            ],
           ),
          ),
         ),
        );
      }
    }
```

5.2.3. Terms_and_conditions:

```
import 'package:groundfresh/consts/consts.dart';


class TermAndCondition extends StatelessWidget {
  const TermAndCondition({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: "Terms and Condition".text.color(darkFontGrey).make(),
      ),
      body: Container(
        padding: const EdgeInsets.all(15),
        child: Column(
          children: [
            "•  Users must agree to the terms and conditions to access and use the app."
                .text
                .color(darkFontGrey)
                .make(),
            5.heightBox,
            "•  Users are responsible for maintaining the confidentiality of their account
information."
                .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  Users must comply with all applicable laws and regulations."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  Farmers are responsible for the accuracy and legality of their listings."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  Users should review and verify information before making a purchase."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  The app facilitates transactions but is not responsible for disputes between
users and farmers."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  User data is collected and processed in accordance with the app's privacy
policy."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  The app is not liable for any damages or interruptions in app availability."
            .text
            .color(darkFontGrey)
            .make(),
          5.heightBox,
          "•  The app may modify the terms and conditions with notice to users."
            .text
            .color(darkFontGrey)
            .make()
        ],
      ),
    ),
  );
 }
}
```

5.2.4. Product_search Screen:

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

import 'package:get/get.dart';
import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/consts/loding_indicator.dart';
import 'package:groundfresh/views/categorise_screen/item_details.dart';
import 'package:groundfresh/views/controllers/firestore_services.dart';

class SearchScreen extends StatelessWidget {
  final String? title;
  const SearchScreen({super.key, this.title});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: title!.text.color(darkFontGrey).make(),
      ),
      body: FutureBuilder(
          future: FirestoreServices.searchproduct(title),
          builder:
              (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
            if (!snapshot.hasData) {
              return Center(
                child: lodingIndicator(),
              );
            } else if (snapshot.data!.docs.isEmpty) {
              return "No product found".text.makeCentered();
            } else {
              var data = snapshot.data!.docs;
              var filtereed = data
                  .where((element) => element["p_name"]
                      .toString()
                      .toLowerCase()
                      .contains(title!.toLowerCase()))
                  .toList();
              return Padding(
                padding: const EdgeInsets.all(8.0),
                child: GridView(
                  gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
                      crossAxisCount: 2,
                      mainAxisExtent: 260,
                mainAxisSpacing: 8,
                crossAxisSpacing: 8),
              children: filtereed
                .mapIndexed((currentValue, index) => Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                     Image.network(
                      filtereed[index]['p_imgs'][0],
                      height: 150,
                      width: 200,
                      fit: BoxFit.cover,
                     ),
                     const Spacer(),
                     5.heightBox,
                     "${filtereed[index]["p_name"]}"
                       .text
                       .fontFamily(semibold)
                       .color(darkFontGrey)
                       .overflow(TextOverflow.ellipsis)
                       .make(),
                     5.heightBox,
                     "${filtereed[index]["p_price"]}"
                       .numCurrency
                       .text
                       .color(greencolor)
                       .fontFamily(bold)
                       .size(16)
                       .make(),
                    ])
                    .box
                    .white
                    .margin(const EdgeInsets.only(right: 5))
                    .rounded
                    .padding(const EdgeInsets.all(8))
                    .make()
                    .onTap(() {
                   Get.to(() => ItemDetails(data: filtereed[index]));
                  }))
                .toList(),
             ),
            );
          }
        }),
      );
    }
```

```
}
```

5.2.5. Order Screen:

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:get/get.dart';
import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/consts/loding_indicator.dart';
import 'package:groundfresh/views/controllers/firestore_services.dart';
import 'package:groundfresh/views/order_screen/order_details.dart';

class OrderScreenB extends StatelessWidget {
  const OrderScreenB({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: "My order".text.fontFamily(semibold).color(darkFontGrey).make(),
      ),
      body: StreamBuilder(
          stream: FirestoreServices.getAllOrder(),
          builder:
              (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
            if (!snapshot.hasData) {
              return Center(
                child: lodingIndicator(),
              );
            } else if (snapshot.data!.docs.isEmpty) {
              return "No order yet!".text.color(darkFontGrey).makeCentered();
            } else {
              var data = snapshot.data!.docs;

              return ListView.builder(
                itemCount: data.length,
                itemBuilder: (BuildContext context, int index) {
                  return ListTile(
                    leading: "${index + 1}"
                        .text
                        .color(darkFontGrey)
                        .fontFamily(bold)
                        .make(),
                    title: data[index]["order_code"]
                        .toString()
                        .text
                        .color(darkFontGrey)
                        .make(),
                    subtitle: data[index]['total_amound']
                        .toString()
                        .numCurrency
                        .text
                        .make(),
                    trailing: IconButton(
                        onPressed: () {
                          Get.to(() => OrderDetails(
                                data: data[index],
                              ));
                        },
                        icon: const Icon(
                          Icons.arrow_forward,
                          color: darkFontGrey,
                        )),
                  );
                },
              );
            }
          }),
    );
  }
}
```

5.2.6. Logout Screen:

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


import 'package:groundfresh/consts/consts.dart';
import 'package:groundfresh/widget_common/ourButton.dart';

Widget exitDialog(context) {
  return Dialog(
    child: Column(
      mainAxisSize: MainAxisSize.min,
      children: [
        "Confrim".text.fontFamily(bold).size(18).color(darkFontGrey).make(),
        const Divider(
          height: 1,
          color: greencolor,
          endIndent: 10,
          indent: 10,
        ),
        10.heightBox,
        "Are you sure want to exit".text.size(16).color(darkFontGrey).make(),
        10.heightBox,
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            ourButton(
                onPress: () {
                  SystemNavigator.pop();
                },
                color: greencolor,
                textColor: whiteColor,
                title: "Yes"),
            ourButton(
                onPress: () {
                  Navigator.pop(context);
                },
                color: greencolor,
                textColor: whiteColor,
                title: "No"),
          ],
        )
      ],
    ).box.color(whiteColor).padding(const EdgeInsets.all(15)).make(),
  );
}
```

5.3. Interface

- **Home Page:**
  There is a Splash screen for 3 seconds. After 3 seconds we can view sign up page.
- **Sign up page:**
  There are two types of registration.
  - Buyer
  - Seller

  Buyer and Seller can complete registration here using the following information
  - Name
  - Email
  - Password
  - Confirm Password

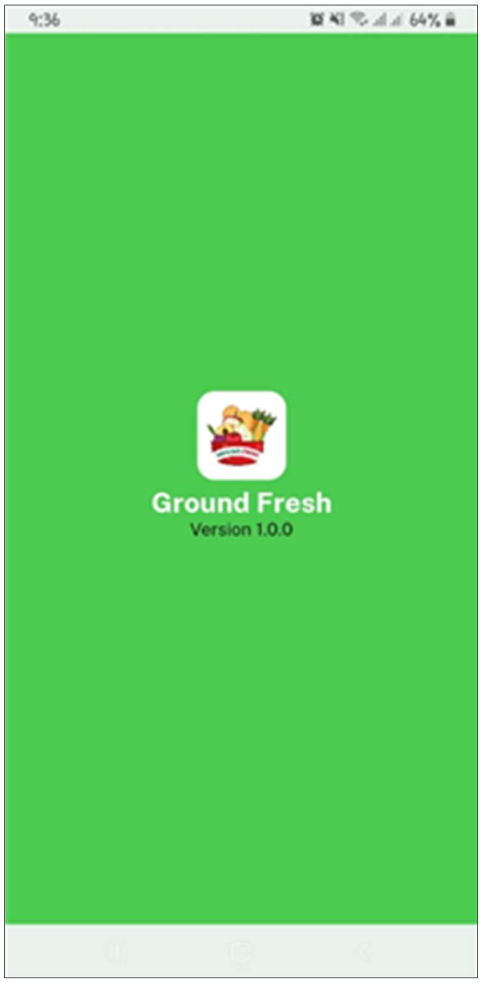


Fig 9: Home Page

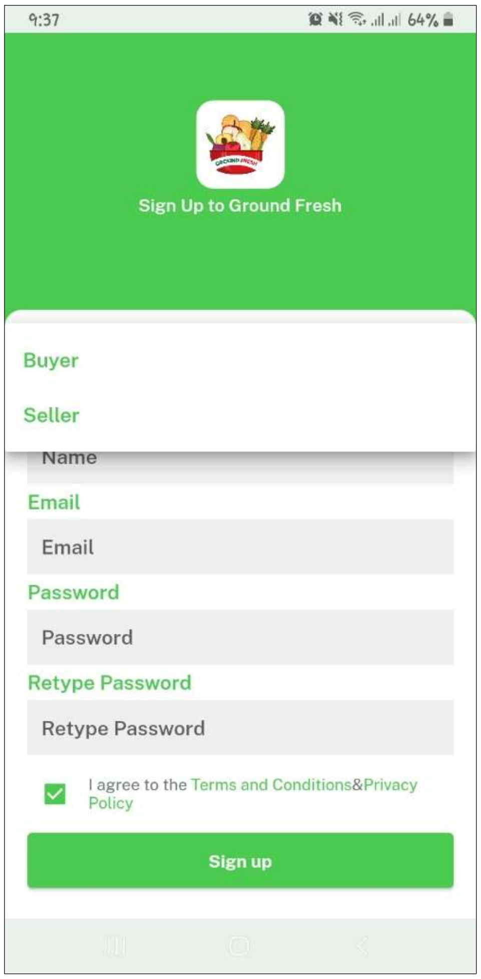


Fig 10 : Sign Up Page

- **Login Page**

  After registration all type users can login and system show role wise respective home page for the users

- **Consumer Users activity**
  - ➢ The buyer has a profile.
  - ➢ They can select the product, select the quantity of the product.
  - ➢ They show payment calculation according to product category and product quantity.
  - ➢ They can order products.
  - ➢ Can purchase products.
  - ➢ They can track order status.

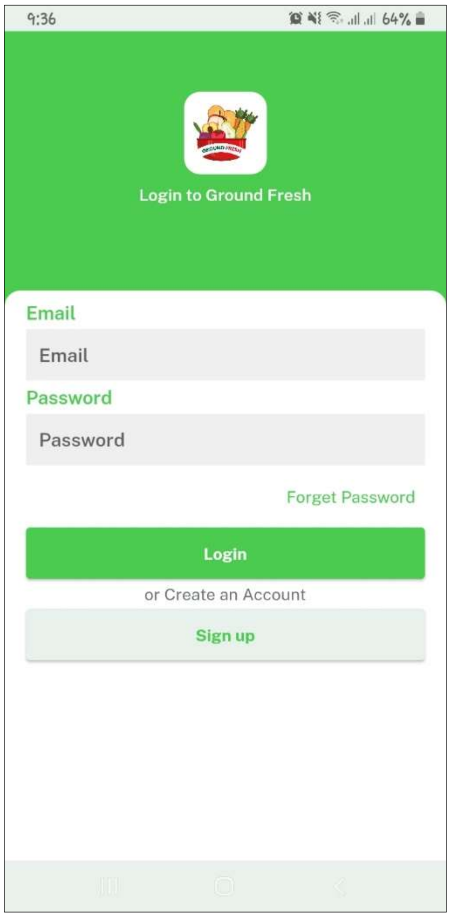


Fig 12 : Login Page

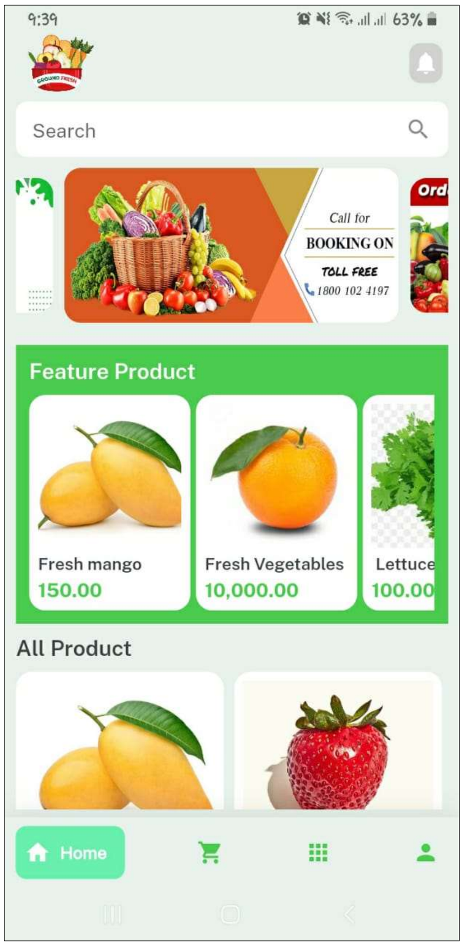
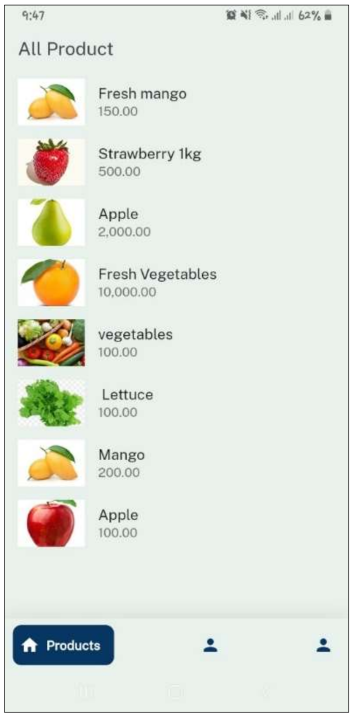


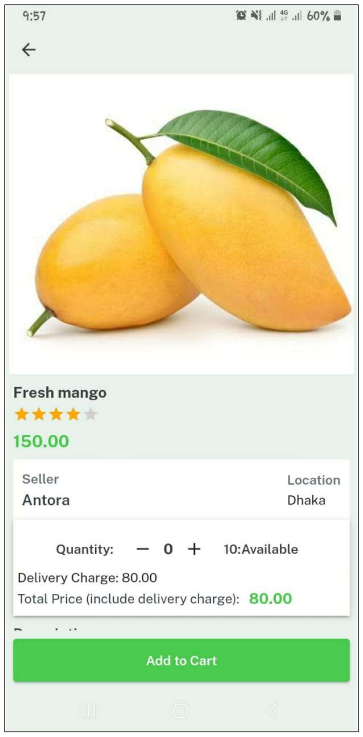
Fig 11: Consumer User Home


9:47 62%

All Product

Fresh mango	150.00
Strawberry 1kg	500.00
Apple	2,000.00
Fresh Vegetables	10,000.00
vegetables	100.00
Lettuce	100.00
Mango	200.00
Apple	100.00

Fig 13: Product Selection


9:57 60%

Fresh mango

150.00

Seller: Antora

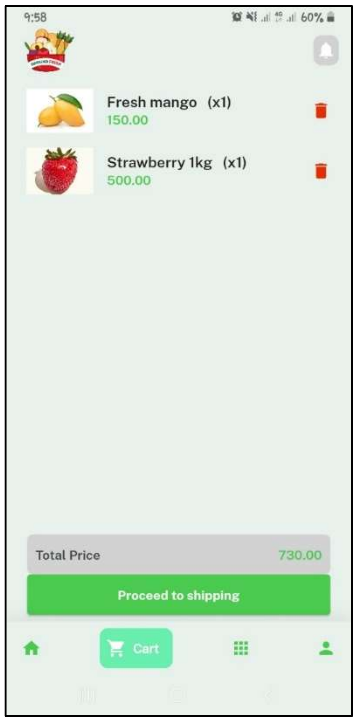
Location: Dhaka

Quantity: — 0 + 10:Available

Delivery Charge: 80.00

Total Price (include delivery charge): 80.00

Fig 14: Added to Cart

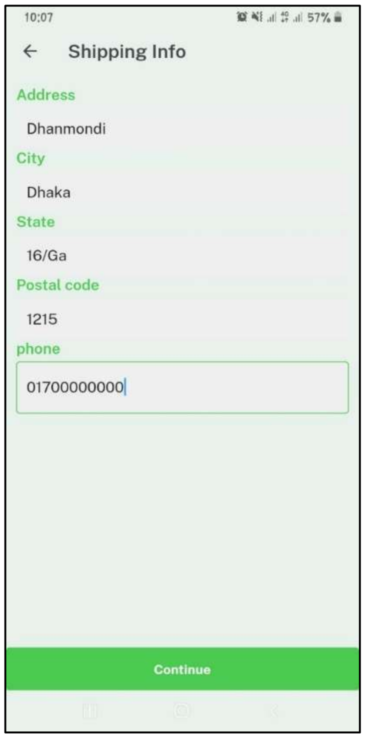

9:58 60%

Fresh mango (x1)
150.00

Strawberry 1kg (x1)
500.00

Total Price 730.00

Fig 16: Proceed to Shipping


10:07 57%

Shipping Info

Address: Dhanmondi

City: Dhaka

State: 16/Ga

Postal code: 1215

phone: 01700000000

Fig 15: Fill up Address and Information.

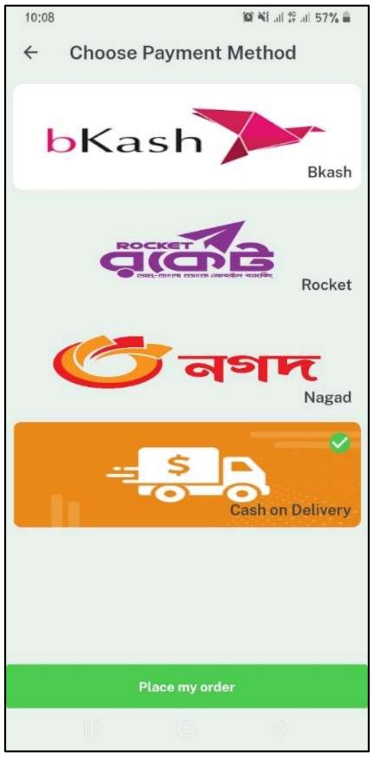


Fig 17: Order History

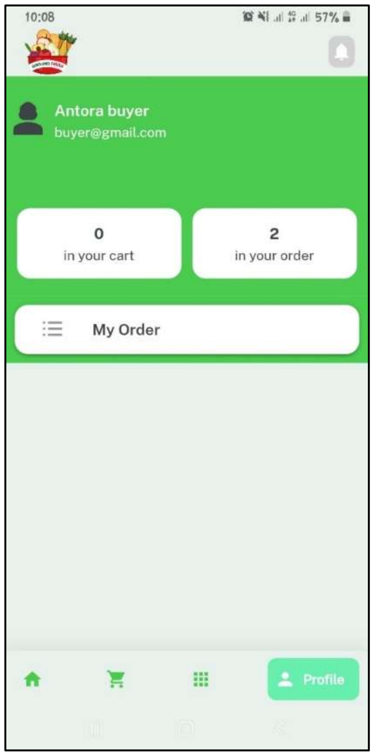
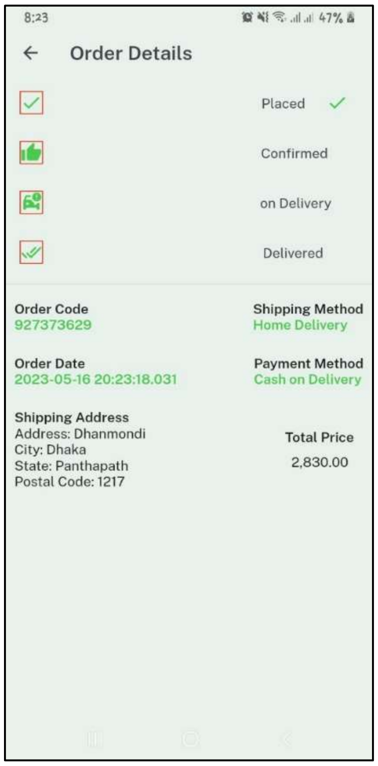


Fig 19: Added to Order list



Order Code
927373629

Shipping Method
Home Delivery

Order Date
2023-05-16 20:23:18.031

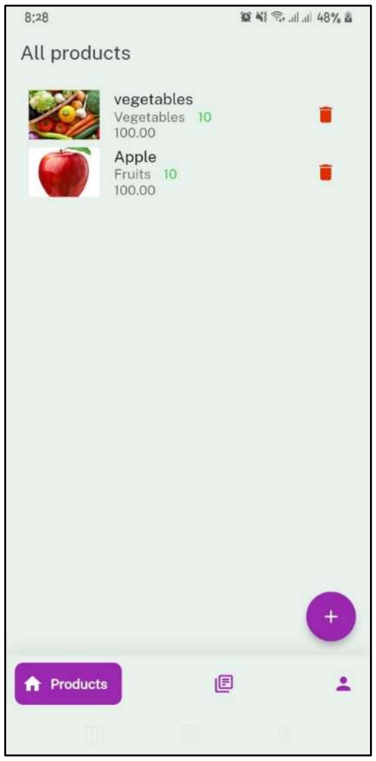
Payment Method
Cash on Delivery

Shipping Address
Address: Dhanmondi
City: Dhaka
State: Panthapath
Postal Code: 1217

Total Price
2,830.00

Fig 18: Proceed to Shipping

- **Seller Users activity**
  - ➢ Seller has a profile.
  - ➢ There is a facility to add location.
  - ➢ Ensure that products can be delivered to specific areas.
  - ➢ Products will be added according to product categories and units.
  - ➢ Price per unit can be fixed.
  - ➢ After adding the product, you can specify the quantity of the product. And it will be added to the store.
  - ➢ After the product is sold, the amount of that product will be subtracted from the total amount.
  - ➢ Product quantity being added.

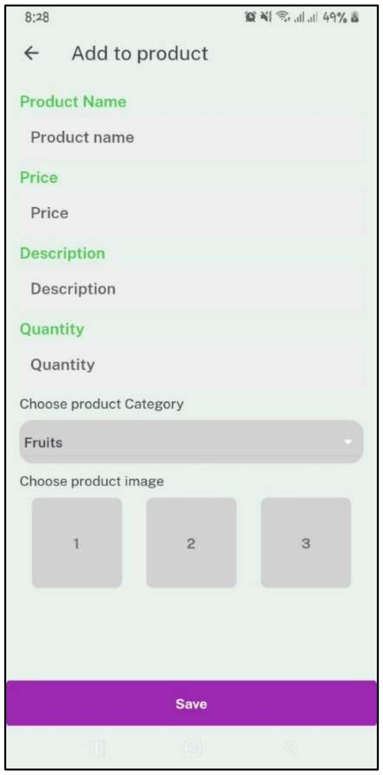

8:28

All products

vegetables
Vegetables 10
100.00

Apple
Fruits 10
100.00

Fig 21: Product List


8:28

Add to product

Product Name
Product name

Price
Price

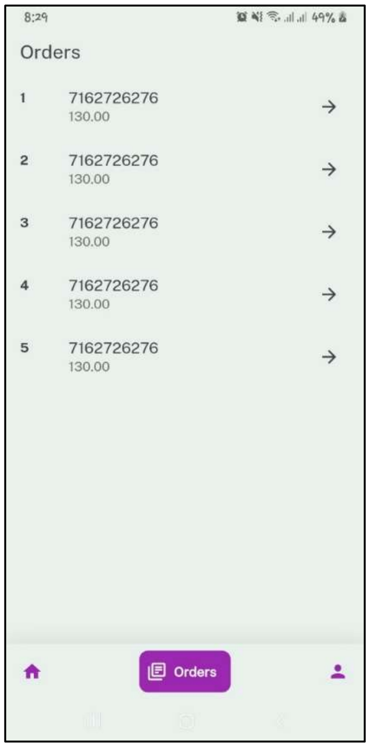
Description
Description

Quantity
Quantity

Choose product Category
Fruits

Choose product image

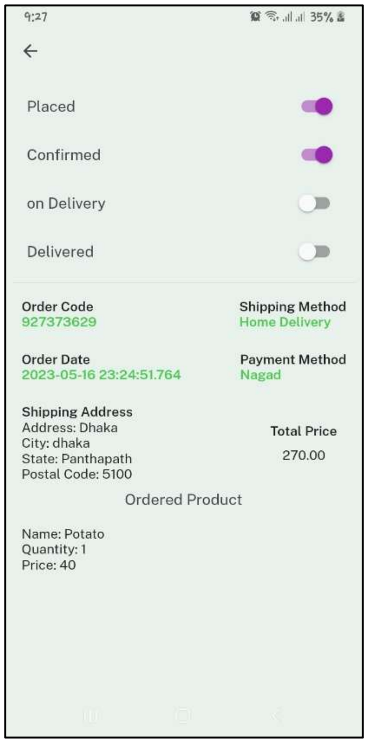
Fig 20: Add Product


8:29

Orders

1	7162726276 130.00	→
2	7162726276 130.00	→
3	7162726276 130.00	→
4	7162726276 130.00	→
5	7162726276 130.00	→

Fig 23: Order List


9:27

Placed ●
Confirmed ●
on Delivery ○
Delivered ○

Order Code
927373629

Shipping Method
Home Delivery

Order Date
2023-05-16 23:24:51.764

Payment Method
Nagad

Shipping Address
Address: Dhaka
City: dhaka
State: Panthapath
Postal Code: 5100

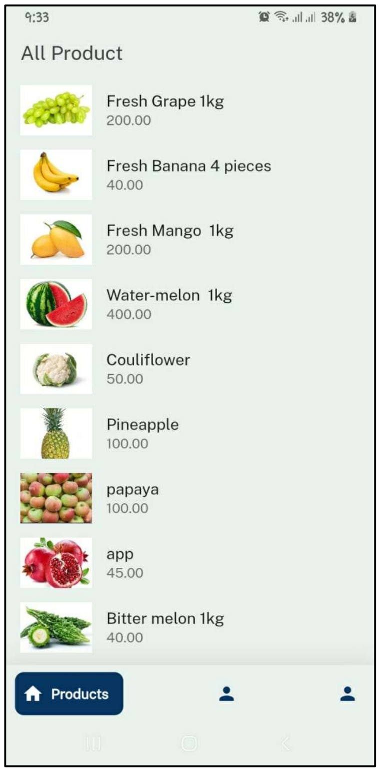
Total Price
270.00

Ordered Product

Name: Potato
Quantity: 1
Price: 40

Fig 22: Order Process

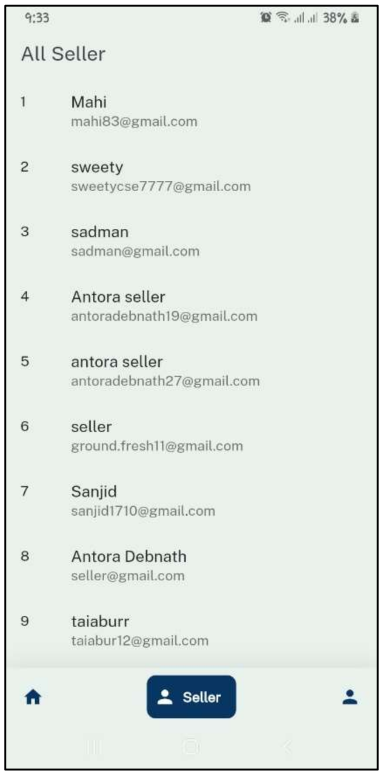
- **Admin Users activity**
  - All Category Product view
  - User List View
  - Password Reset
  - Apps activity Dashboard
  - Category Add


9:33 38%

All Product

Fresh Grape 1kg	200.00
Fresh Banana 4 pieces	40.00
Fresh Mango 1kg	200.00
Water-melon 1kg	400.00
Couliflower	50.00
Pineapple	100.00
papaya	100.00
app	45.00
Bitter melon 1kg	40.00

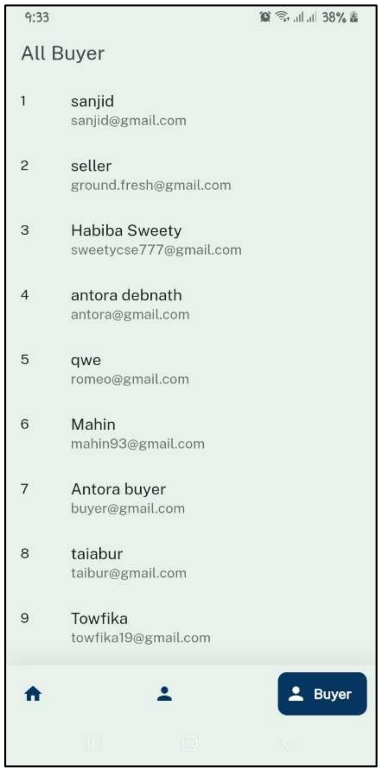
Fig 26: All Product


9:33 38%

All Seller

1	Mahi	mahi83@gmail.com
2	sweety	sweetycse7777@gmail.com
3	sadman	sadman@gmail.com
4	Antora seller	antoradebnath19@gmail.com
5	antora seller	antoradebnath27@gmail.com
6	seller	ground.fresh11@gmail.com
7	Sanjid	sanjid1710@gmail.com
8	Antora Debnath	seller@gmail.com
9	taiaburr	taiabur12@gmail.com

Fig 25: Seller List


9:33 38%

All Buyer

1	sanjid	sanjid@gmail.com
2	seller	ground.fresh@gmail.com
3	Habiba Sweety	sweetycse777@gmail.com
4	antora debnath	antora@gmail.com
5	qwe	romeo@gmail.com
6	Mahin	mahin93@gmail.com
7	Antora buyer	buyer@gmail.com
8	taiabur	taibur@gmail.com
9	Towfika	towfika19@gmail.com

Fig 24: Buyer List

5.4. Limitations of Ground Fresh App

While ground fresh apps offer numerous benefits, they also have certain limitations that should be considered. Here are some potential limitations of ground fresh apps:

5.4.1. **Limited Geographical Coverage:** Ground fresh apps may have limited coverage areas, particularly in rural or remote regions. This can restrict access to local produce for consumers living outside the app's delivery range.

5.4.2. **Dependence on Local Supply:** The availability and variety of products on ground fresh apps heavily rely on the participation of local farmers and producers. In areas with limited agricultural resources or low adoption of the app among farmers, the product selection may be limited.

5.4.3. **Seasonal Availability:** Depending on the region and climate, certain agricultural products may be available only during specific seasons. Ground fresh apps may face challenges in consistently providing a wide range of products throughout the year, leading to potential limitations in product availability during off-seasons.

5.4.4. **Delivery Logistics:** Maintaining efficient and cost-effective delivery operations can be a challenge, especially in areas with dispersed or low-density populations. Delivery costs, infrastructure limitations, and managing perishable products can impact the overall viability and convenience of the service.

5.4.5. **Consumer Adoption**: Ground fresh apps may face challenges in attracting a critical mass of users, especially in areas where traditional grocery shopping methods are deeply ingrained. Convincing consumers to shift from their established purchasing habits and adopt the app may require substantial marketing efforts and consumer education.

5.4.6. **Product Quality and Consistency:** Maintaining consistent product quality can be a challenge when working with multiple farmers and producers. Variations in farming practices, handling, and storage methods can affect the overall quality and freshness of the products received by consumers.

5.4.7. **Technological Barriers:** Some consumers, particularly those in older demographics or with limited access to smartphones or internet connectivity, may face challenges in using and navigating ground fresh apps. Ensuring a user-friendly interface and providing adequate support for users with varying levels of technological proficiency is essential.

5.4.8. Trust and Verification: Verifying the authenticity of claims made by farmers regarding their farming practices, certifications, or product attributes can be difficult. Establishing trust between farmers and consumers and implementing reliable verification mechanisms is crucial to ensure transparency and maintain consumer confidence.

5.4.9. Environmental Impact: While ground fresh apps aim to reduce the carbon footprint associated with long supply chains, there can still be environmental implications related to delivery logistics, packaging materials, and waste management. Ensuring sustainable practices throughout the entire process is important for minimizing environmental impact.

Understanding and addressing these limitations can help ground fresh apps overcome challenges and optimize their operations to better serve both farmers and consumers.

CONCLUSION

Ground Fresh is a mobile app designed to connect buyers and sellers of fresh vegetables. The app provides a platform for farmers and growers to showcase their produce and reach a wider audience, while also allowing buyers to purchase fresh vegetables conveniently and securely. With features such as search functionality, product listings, seller and buyer profiles, in-app purchases, real-time notifications, map integration, quality control, sustainability metrics, recipe suggestions, delivery tracking, rewards programs, and multilingual support, Ground Fresh offers a comprehensive solution for buying and selling fresh vegetables.

The development of Ground Fresh represents an exciting opportunity to address the growing demand for fresh and locally grown produce, while also supporting small-scale farmers and sustainable agriculture practices. By providing a platform for farmers to connect with buyers, the app promotes transparency and accountability in the food supply chain, while also fostering a sense of community and connection between producers and consumers.

Finally, there is a significant opportunity to expand the app beyond fresh vegetables and into other areas of the food supply chain. For example, the app could be used to connect buyers and sellers of locally produced meats, dairy products, and other specialty foods. By creating a comprehensive platform for buying and selling locally sourced foods, Ground Fresh could help to transform the food system and create a more sustainable and equitable food economy.

In conclusion, Ground Fresh is a mobile app with the potential to revolutionize the way we think about food production and consumption. By promoting transparency, sustainability, and community connection, the app represents a powerful tool for supporting local farmers and growers, while also providing consumers with convenient access to fresh and healthy foods. With further development and expansion, Ground Fresh could play a critical role in creating a more sustainable and equitable food system for generations to come.

REFERENCES

[1] H. Yoo, J. Kim, J. K. Lee, "Development of a Mobile App for Connecting Local Farmers and Consumers," IEEE Access, vol. 8, pp. 108046-108055, 2020.

[2] A. R. A. Rahman, A. M. A. Rahim, M. A. Shukor, "Design and Development of a Mobile Application for Online Vegetable Trading System," 2020 7th International Conference on Electrical and Electronics Engineering (ICEEE), Dhaka, Bangladesh, 2020, pp. 1-6.

[3] S. Kim, J. Kim, "Design and Development of a Mobile Application for Supporting Local Agricultural Products," 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, South Korea, 2020, pp. 1242-1245.

[4] H. Wu, J. Wu, "Design and Implementation of a Mobile E-commerce Platform for Fresh Agricultural Products," 2018 International Conference on Smart Agriculture and Environmental Monitoring (ICSAM), Shenzhen, China, 2018, pp. 131-136.

[5] A. Kumar, A. Kumar, A. Kumar, "Design and Development of Mobile App for Farmers for Selling Agricultural Products," 2019 International Conference on Automation, Computational and Technology Management (ICACTM), Ghaziabad, India, 2019, pp. 1-4.

[6] S. S. Kumar, S. S. Kumar, M. R. Hasan, "Design and Development of Mobile Application for Farmers," 2019 3rd International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2019, pp. 1322-1326.

[7] J. Kim, H. Kim, "Development of a Mobile Application for the Agricultural Market," 2018 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, South Korea, 2018, pp. 1102-1105.

[8] Y. Wang, X. Huang, "Design and Development of Mobile Commerce Platform for Agricultural Products," 2021 International Conference on Computer Network, Electronic and Automation (ICCNEA), Taiyuan, China, 2021, pp. 141-146.

[9] Y. Choi, S. Han, "Development of a Mobile Application for Farmers and Consumers," 2018 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), Jeju Island, South Korea, 2018, pp. 37-40.

[10] Y. Kim, H. Kim, "Design and Implementation of a Mobile Application for Agricultural Product Distribution," 2019 3rd International Conference on Information Management (ICIM), London, United Kingdom, 2019, pp. 7-11.

[11] H. Lu, J. Hu, "Design and Implementation of a Mobile Application for Online Shopping of Agricultural Products," 2018 14th International Conference on Computational Intelligence and Security (CIS), Hong Kong, China, 2018, pp. 291-294.