# A SUPPLY CHAIN MANAGEMENT SOFTWARE FOR POULTRY FARMING

By

SULE EMMANUEL NGBEDE

17010301033

A PROJECT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, COLLEGE OF BASIC AND APPLIED SCIENCES,

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE

AUGUST, 2021

## DECLARATION

I hereby declare that this project has been written by me and is a record of my own research work. It has not been presented in any previous application for a higher degree of this or any other University. All citations and sources of information are clearly acknowledged by means of reference.

…….

SULE, EMMANUEL NGBEDE

……

Date

## CERTIFICATION

This is to certify that this project titled '**A Supply Chain Management Software for Poultry Farming'** was prepared and submitted by SULE EMMANUEL NGBEDE in partial fulfillment of the requirements for the degree of BACHELOR OF SCIENCE IN COMPUTER SCIENCE.

The original research work was carried out by him under my supervision and is hereby accepted.

_____ (Signature and Date)

Matthew O. Adewole, PhD

Supervisor

_____ (Signature and Date)

Matthew O. Adewole, PhD

Coordinator, Department of Computer Science and Mathematics

## DEDICATION

I dedicate this project work to God Almighty, the giver of life.

**ACKNOWLEDGEMENT**

I first of all want to thank almighty God for his grace upon my life. From start to completion of this project, God has giving me wisdom, strength and the courage to pull through.

My profound gratitude goes to Dr. B.O. Okunoye for approving the project topic and providing the initial guidance I needed to start the project. I specially appreciate my project supervisor Dr. M.O. Adewole who saw me through the completion of the project. He was there for me when I needed guidance and always sought to help me out despite his busy schedule as the Head of the Department. I also appreciate the efforts of the Dean of Post graduate studies Prof. I.O. Akinyemi for his continued efforts in ensuring continued excellence, discipline and high standards within the department. Special thanks also go out to the other lecturers in the department in persons of Dr. F.A. Kasali, Dr. Igiri, Mr. J.A. Balogun, Miss. Anabelle, Mr. I.O. Ebo, Mr. Michael and other members of staff not mentioned. Your positive impact upon my life will not be forgotten.

I am forever grateful to my parents Mr. & Mrs. Sule for giving me this opportunity to further my education. I am grateful for their continued love and support both financially and morally upon my life. I also want to thank my siblings and other members of my family for their continued prayers and constant words of encouragement.

Finally, I want to say thank you to the all the management and staff of the University for their commitment to hard work and excellence. And to all my friends who made my stay in the university worthwhile I say thank you.

## TABLE OF CONTENTS

## LIST OF FIGURES

## ABSTRACT

Supply chains have become a key player to any business activity that revolves around serving of goods to customers. The poultry sector is no stranger to the applications of supply chains and its impact in delivering value when done right. Technology has been a major driving force in the advancement of supply chains in relation to the building of supply chain management software for private companies looking to enhance their supply chains. The objective of this project is to develop a mobile application that can be used by poultry farmers to manage their supply chain activities

This project focuses on building a forward supply chain management software using agile techniques to software development. The key users that the software is intended for are the farmers and distributors. The farmer to distributor relationship is modelled in the application with the aim of helping farmers better manage the supply of their goods and services to distributors.

Ordering modules were implemented in the software to help automate the process of booking farm orders from distributors to farmers. Stock inventory modules are also implemented in the system for better collection and storage of farm data on the cloud.

In conclusion, the implemented software is capable of automating key farm activities that farmers rely on so as to better enhance their supply chain. For future work it is recommended that other researchers build upon the results of this project to develop software that can account for reverse chain activities that go on in the poultry supply chain such as return of farm goods.



Keywords: Supply chain, poultry, management software, stock inventory and orders.

# CHAPTER ONE
# INTRODUCTION

## 1.1 Background of Study

Consumption of poultry products in Nigeria is rapidly increasing due to urbanization and the continued growth of the poultry sector. The Nigerian poultry sector accounts for 25% of the nation's agricultural GDP, which amounts to 6% of the nation's total GDP. With an estimated net worth of over 1.6 trillion naira, it is regarded as the most commercialized of all Nigeria's agricultural sub-sectors (Awojulugbe, 2019). Indirectly and directly, the poultry industry employs about 14 million Nigerians. Nigeria currently has the second largest chicken population in Africa with a stock value of approximately 180 million birds. Every year, about 454,000 metric tonnes of chicken meat and over 14 billion eggs are produced locally (Netherlands enterprise agency, 2020).

Chickens account for the most types of birds that are reared for their meat and eggs in poultry farms. Chickens that are raised for their meat are called broilers, while those raised for eggs are called layers. Typically farmers buy these chickens from hatcheries as chicks and groom the chicks to maturity. Raw materials such as chicken feeds, saw dust, drinking bowls, cages, food trays etc., are then used to groom the chickens for their meat and eggs. These raw materials are typically sourced from local vendors and markets by the poultry farmers. Once the chickens are old enough they start laying eggs in the case of layers. The eggs are collected and then packaged to be sold off to distributors who in turn sell these eggs to customers such as hotels, eateries, homes etc. After a period of time, the chickens are eventually sold off for their meat. This above scenario of a poultry farms activities is called a supply chain.

A supply chain is a network used to deliver goods and services from raw materials to end customers. It involves the series of steps that are involved in getting a product to the final consumer. Elements of a supply chain begin with the receiving of orders from customers for certain products and meeting up with these requests (Kenton, 2020). A key aspect to the stability of any supply chain lay in the coordination of activities that go on along the chain. The coordination of these activities is the collective responsibility of stakeholders involved with the chain. For a poultry supply chain, farmers are the main stakeholders as they play an important role in keeping the chain stable (Fajemisin & Ogunribido, 2018).The primary goal of supply chains is to bring sustainability or stability to the network so as to gain the benefits of meeting up with the demands of the final consumer (Shamsuddoha, Quaddus, & Klass, 2013).The aim

of achieving sustainability in the network has led to various studies in supply chain management as a means to design supply chains that are effective in their own usage scenarios.

### 1.2 Statement of the Problem

Many inadequacies exist in the Nigerian poultry supply chain. Amongst these issues the most pressing issue is the lack of automated software tools to manage the poultry farm business activities. This is caused due to process inefficiencies that exist within the supply chain networks used by poultry farmers to serve their goods to customers. This lack of exposure to easily accessible distribution networks for poultry farmers to sell their goods tend to lead to damage of the product due to overstocking. Adequate software tools are also not readily made available for poultry farmers to keep count of stock inventory at hand.

### 1.3 Aim and Objectives

The aim of this project is to develop a software that can be used by poultry farmers to automate farm activities with the goal of enhancing their supply chain. The specified objectives are as follows:

i. identify the key users involved in the poultry supply chain.
ii. identify the requirements of an adequate supply chain management software.
iii. specify the design of the proposed system.
iv. implement the specified system design for the supply chain management software.
v. test the implemented poultry supply chain management software.

### 1.4 Proposed Methodology

In order fulfill the stated objectives stated above, the following steps will be undertaken.

**a.** An informal survey will be done to identify the various key stakeholders involved in the poultry supply chain.
**b.** An analysis will be done to elicit the functional and non-functional requirements of the system.
**c.** A system design will be illustrated using UML (Unified modelling language) diagrams such as use case diagrams, sequence diagrams, activity diagrams etc.
**d.** Firebase will be used to model and design the backend architecture of the system.
**e.** A cross platform framework called Flutter will be used to develop the software.
**f.** A version control software called git will be used in the development phase to manage various versions of the software.

**g.** Unit and Acceptance testing will be done to ensure that the system meets the specified user need.

### 1.5 Scope and Limitations

The software to be implemented in this project will be restricted only to farmers of the Nigerian poultry sector. The implemented software also has the capability to be used by researchers in other countries looking to design management software for farmers in their countries. Some of the possible limitations expected to be faced while working on the project are as follows:

- Lack of access to physical devices to test a working version of the software on.
- Slow internet connection present in the university.

### 1.6 Significance of Study

This project work is set to benefit Nigerian poultry farmers by providing a software solution capable of enhancing key business activities via the use of automation. The proposed software will help improve the way farmers serve their goods to customers by connecting the farmer to more distributors willing to buy their farm goods. This in turn will reduce overstocking of farm goods in warehouses and lead to increased revenue. Data collection features will also be made available in the software so as to properly store generated farm data on the cloud. This collected data can prove useful to both the farmers in relation to their business activities and even researchers who need access to such data for future research purposes.

### 1.7 Definition of Terms

- Supply chain: a network used to deliver products and services from raw materials to end consumers.
- Supply chain management software: are software tools used in the execution of transactions, supplier relations and control of business processes in a supply chain.
- Poultry: animal husbandry involved in raising of chickens for their eggs and meat.
- Stock: is a term used to describe a store of products that are ready to be sold.
- Inventory: the stock count of an item on hand at a particular business location.
- Flutter: cross platform UI framework for building mobile, desktop and web apps.
- Firebase: a mobile application development platform, designed for building applications quickly.
- Git: a version control software for managing various versions of software.

# CHAPTER TWO
# LITERATURE REVIEW

## 2.0 Introduction

This chapter gives a conceptual review on literature that abounds in the field of supply chain management. The first few sections gives detailed descriptions the topic of supply chains and its types. After which the term supply chain management is introduced with a section detailing its origins and its relation to the design of supply chain management software. Later sections give an insight into the chosen software methodology for the design of the system and the rationale between picking a methodology over the other based on the given usage scenario. This chapter concludes with the reviews of other works in relation to supply chain management.

## 2.1 Supply Chain

A supply chain is a network used to deliver goods and services from raw materials to the final consumer. It consist of the network of individuals, information, organizations, resources, activities and technology involved in the manufacturing and distribution of a product. Multiple activities take place within the chain to provide a certain niche of consumers their desired products. These activities typically involve the packaging, storage, transportation and distribution of goods and services to the end customer (Kenton, 2020).

The segment of the supply chain that is involved with getting the final goods from the point of manufacture to the consumer is called a distribution channel (Lutkevich, 2020). The transportation of the final goods to the customer in a timely fashion along these distribution routes involves the use of logistics by a host of transportation companies responsible for making sure the goods sent are of the right quality (Fajemisin & Ogunribido, 2018).

As highlighted by Lutkevich (2020), there are six fundamental steps in any supply chain. The time taken for any one of this processes to complete is known as the lead time. These steps are as follows:

i. Sourcing of raw materials: this involves the gathering of resources necessary to produce a product.
ii. Conversion of the raw materials into basic parts: once the raw materials have been gathered, they ought to be refined via manufacturing processes into parts that can be combined to form the final product.
iii. Combination of the basic parts to form a product: parts necessary to build a product are combined at this stage to form the final product.

iv. Order fulfillment/Sales: this involves the fulfillment of order requests from customers by the sales team.

v. Deliveries: after the product for a given customer has been procured, it is packed and shipped to the address of the customer. This stage involves the use of logistics in order to get the product out to the consumer in a timely fashion.

vi. Customer support and returns: in situations where the delivered product doesn't meet up to the consumer expectations or the wrong product was shipped, provisions ought to be made for the return of such items back to the manufacturer. The customer support personnel is responsible for overseeing this stage.


Figure 2.1: A generic supply chain (corporatefinanceinstitute.com)

### 2.1.1 Types of Supply Chains

There are 3 types of supply chains namely, forward supply chains, reverse supply chains and combined forward and reverse supply chains.

a. Forward supply chain (FSC): this is a step by step process that starts from the raw material collection phase up until the consumption of the finished product by the final consumer. It links all the internal and external partners of the suppliers, investors, carriers, policymakers, intermediary companies and systems information providers as one system (Shamsuddoha, Quaddus, & Klass, 2013).

Following this definition, the poultry forward supply chain starts with the collection of suitable parent breed, their subsequent rearing for collection of hatch able

eggs, hatching of the eggs in a hatchery, the distribution of the hatched chicks via the use of middlemen, rearing of the chicks for a certain period of time by the farmer and then selling of the meat and/or eggs to the final consumer. Value of products increase in value as it passes through each stage in the supply chain on route to the final consumer. Hence, the smoother the supply flow, the more benefits and rewards lye in store for the stakeholders involved in the supply chain as sustainability within the chain is achieved.

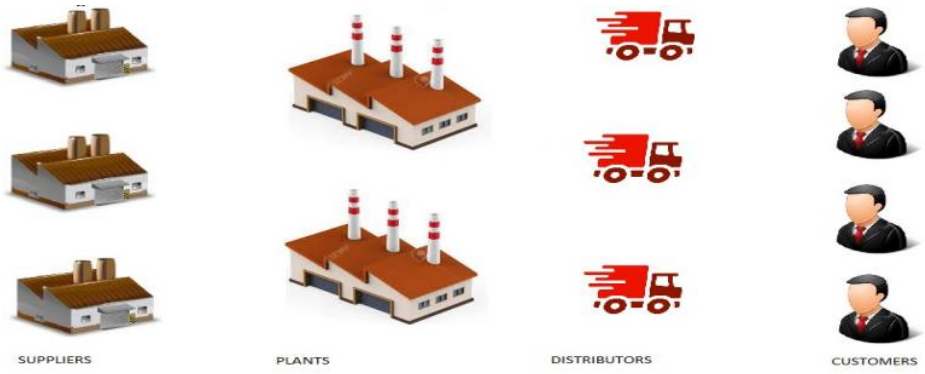


Figure 2.2: Forward supply chain design (Arumugham, C, & Parameswaran, 2017)

b. Reverse supply chain (RSC): was introduced to address the issues of disposal, recycling and remanufacturing of rejected wastage or products. These are all by-products of the forward supply chain. Steps such as product re-design, manufacturing of by-products, extension of product life, end life of a product and possible recovery processes at product end of life (Shamsuddoha, Quaddus, & Klass, 2013). It focuses on the movement of goods back to its point of origin. The reverse supply chain starts at the final consumers end, and then to the wholesaler or manufacturer. The returned goods are meant to either be properly disposed, remanufactured, refurbished in the case of defective products or even recycle.

   The more goods that are returned can lead to the supply chain being busier, quicker and economically efficient. It is also important to note that the value of products tend to decline as it passes through more stages in a reverse supply chain, due to the extra time and resources added during the return process.


Figure 2.3: Reverse supply chain design (Arumugham, C, & Parameswaran, 2017)

c. Combined forward and reverse supply chains: this refers to supply chains that integrate both forward and reverse chain methods into their business workflow. According to Shamsuddoha, Quaddus, & Klass (2013), poultry industries have no chance of product retrievals, returns or reconditioning due to the perishable nature of poultry goods i.e. eggs and meat. They do however agree that opportunities lye in the recycling of poultry wastes for the production of valuable products such as fertilizers, pillows, biogas, bakery items and charcoal.

   Examples of poultry farm wastes include feathers, manure, cracked eggs and egg shells. The feathers can be used in the manufacture of pillows and beds; the chicken manure can be processed to create fertilizers to be used in growing of crops; cracked eggs can be sold to bakeries and eateries at a cheaper price for their own usage; and egg shells which contain high levels of calcium can be used as ingredients for beauty products.

## 2.2 Supply Chain Management

Supply chain management refers to the set of approaches undertaken to efficiently integrate suppliers, manufacturers, warehouses and stores so that products are produced and distributed at the right quantity, location and time, so as to minimize service cost along the chain while also satisfying service level requirements (Simchi-Levi & Kaminski, 2008). The term supply chain management refers to the management of the flow of goods and services within a supply chain. The individual responsible for overseeing all the activities in a supply chain is called a supply chain manager. This flow of goods and services could be upstream and downstream value-added

flow of materials, goods and information amongst the suppliers, organization and end consumer (Lutkevich, 2020).

The goal of supply chain management is to create and implement supply chains that are economically cost efficient. Supply chain management aims at centrally controlling the production, shipment and distribution of a product.

### 2.2.1 History of Supply Chain Management

The term supply chain management was first coined by a consultant at Booz Allen Hamilton named Keith Oliver. He introduced the term to the public in an interview with the Financial Times in the year 1982. Nevertheless, research work related to supply chain management trace back to as early as the 1950s where multiple researchers were interested in finding optimal policies relating to inventory management. An example of such work was by Clark & Scarf (1958), who developed models for managing inventory at multiple echelons (Lauren & Swaminathan, 2015).

Regardless of the term first appearing in the 1980s, significant research work on supply chain management did not begin until the mid-1990s. Since then the number of publications on supply chain management have increased greatly. The advent of the 21$^{st}$ century saw what could otherwise be called a supply chain revolution. Multiple organizations began to establish high level position with the supply chain title (Rafaela & Carmen, 2009).

### 2.2.2 Supply Chain Management Software

Supply chain management software refers to a set of software modules or tools designed to help or assist supply chain managers in managing the flow of information, execution of supply chain transactions, control of supplier relationships and the overall controlling of business processes within a supply chain.

Globally, companies keep striving to integrate all their business processes from their suppliers to their customers so as to reduce their time to market, reduce the gap between their supply and demand and raise the level of their customer service. By doing so they have improved their business performance and saved cost. Four key factors are crucial to any supply chain project regardless of the technology behind it, they are as follows right leadership, right business focus, right result and the right approach (Jose & Andrew, 2005).

Below are some of the functionalities or features which supply chain management software commonly include

i. Customer requirement processing

ii. Purchase order processing
iii. Sales and distribution
iv. Inventory management
v. Goods receipt and warehouse management
vi. Supplier management or sourcing

### 2.3 System Development Life Cycle

A Software development life cycle (SDLC) describes all the activities that need to be carried out for a given software to evolve all through in its life cycle. It is represented graphically, showing each stage of the life cycle and the transitions amongst these phases. Textual descriptions of activities to be done are also included in the graphical depiction (Rajib, 2014). It is a systematic process of building software that ensures the quality and correctness of the implemented software. The goal of the SDLC is to produce software that meets up to the user's expectation.

One of the primary reasons for using software development cycle techniques when building software, is that it encourages the development of software in a systematic and thorough manner. Different kinds of systems require different approaches to its software development processes. For example a critical control systems such as an airplane system will require more careful and thought out planning in its development cycle than a gaming software designed for kids (Sommerville, 2011).

#### 2.3.1 Phases of the System Development Life Cycle

The software development life cycle is divided up into six phases. These six phases feed off one another i.e. the result of a completed phase is passed unto the next phase. The SDLC works by lowering the cost of development while also maintaining the expected quality of a software with reduced production time (Altvater, 2020). The six phases are described below.

i. Requirement analysis: the goal of this phase is to specify in detail the user and system requirements of the system. It is at this phase that the functional and non-functional requirements of the system are derived. The end product of this phase is the requirement specification document which is used throughout the system development life cycle.
ii. Planning: this phase is concerned with the mitigating risk in the project. The goal here is to identify the risks associated with the project and plan out the technical approaches that can be applied to solving these risks. The analysis done at this stage is what is known as feasibility study (Tutorialspoint, 2020).

iii. Architectural design: this is the design of the systems modules and their interactions with one another. The flow of data between each module and their functionality is shown at this phase. There are two kinds of design documents developed at this phase which are the High level design and the low level design.

iv. Software development: this is the phase where actual coding out of the system begins. All the identified components of the system are developed at this phase. The entire project is broken up into phases or components and coded out by different developers depending on the size of the team.

v. Testing: this is the analysis of the implemented software. The goal here is to find bugs and defects in the software and report it to the developers for fixing. The quality assurance team is typically the group responsible for handling testing.

**vi.** Deployment: at this phase the software is deployed into a production environment where users of the software can interact with it and make use of its functionalities.

### 2.3.2 Software Development Life Cycle Models

SDLC models can be seen as simplified representation of a software process. Each model represents a given process from certain perspectives. They are abstractions of the process that can be used to explain different approaches to software development (Sommerville, 2011).

Due to the scope, comprehensiveness, unpredictability, complexity and unstable requirements of supply chain management software, agile methodologies are more suited for the development of supply chain management software (Deepti & Alok, 2007). Their argument was that it was not possible to develop such software with predictable development process models like waterfall. Their solution to this was through the application an agile method known as extreme programming on a supply chain project so as to mitigate risks and handicap at each phase of the project.

Discussed below are 2 software process models, namely the waterfall model, and the agile development model.

1. Waterfall model: the waterfall model is a plan driven process model that came into prominence in the early 1970s. It is classified as a plan driven process model due to the planning and scheduling of process activities before any work is done (Sommerville, 2011). Each phase in the waterfall model is a process activity cascaded on top one another with each phase feeding off the other and passing information unto the next phase. At the end of each phase, the result is typically one or more approved documents.

The next phase in the development cycle cannot proceed until the former has been approved.

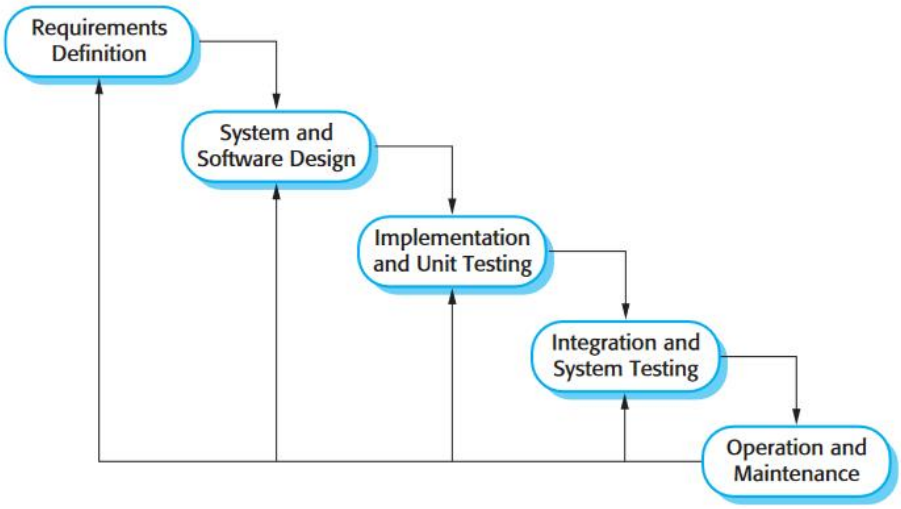


Figure 2.4: The waterfall model (Sommerville, 2011)

2. Agile development model: agile methods are incremental approaches used in building software. Agile methods focus on rapid development, frequent release of software updates, reduction of process overheads, and production of high quality code. One selling point of agile methods is its direct involvement of customers in the development process. The agile model was designed to help projects adapt to frequent changes in the system and user requirements. Its primary aim is to facilitate quick completion of projects (Rajib, 2014).
   The agile model aids the quick completion of projects by scaling down all process bureaucracy associated with software releases in an organization. Rather than relying overtly on the requirement specification documents to guide the software development phase, focus is put more into breaking up the project into smaller components which can then be iteratively built as software increments. Each successive increments builds upon the other hence, adding more functionality with each version.

   It is advisable to implement agile techniques for projects where the requirements can't be fully fledged out at the beginning. Businesses and companies also involved in software development are also shifting to agile methods of development so as to keep up with the users demands.

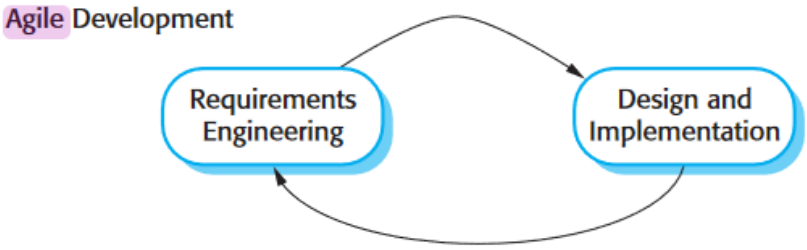


Figure 2.5: Agile model specification (Sommerville, 2011)

### 2.3.3 Extreme Programming

Extreme programming is an agile methodology that focuses on iterative development and good practices to software development. Requirement are expressed as user stories in extreme programming. Deepti & Alok, (2007) noted in their paper that the best way to develop supply chain management software was through the use of agile methods. Their reasoning behind this was that the business nature of supply chains posed a difficult challenge in planning out from top to bottom the development process without having to change some significant portion of the project to changing user requirements and the business landscape. Hence, the use of plan driven approaches wasn't encouraged as a development for supply chain projects as such models aren't prone to change tolerance.

Extreme programming being an agile model allows for the quick incremental delivery of software and embraces change in system requirements. This is why extreme programming methods will be used for the software development cycle of this project.

## 2.4 UML (Unified Modelling Language)

UML is a language used for documenting models. It is a set of 13 different diagram types that can be used to model software systems (Sommerville, 2011). It has its own syntax and semantics. Contained in the UML are a set of graphical notations such as triangles, squares, lines, arrows etc., which can be combined in certain was following the language syntax so as to document the design and analyze results (Rajib, 2014). One of the major advantages of using the UML is that it can be used to document Object Oriented style programs and design results obtained from any methodology used.

The UML emerged from studies in the 1990s on OOP (Object Oriented Programming) modelling where similar OOP methodologies were combined to create the UML. It is currently recognized as a standard tool used for the development of models for software systems.

UML diagrams are divided into 3 categories namely

i. Structural diagrams: illustrate a static view of the system. It shows various objects making up the system. E.g. class diagrams, object diagrams etc.
ii. Behavioral diagrams: this class of diagrams represent the functioning of the system. They illustrate the dynamic parts of the system. E.g. activity diagrams, use case diagrams etc.
iii. Interaction diagrams: show the interaction between two or more entities in a system. E.g. sequence diagrams, timing diagrams etc.

### 2.5 System Development Tools

This section gives an overview of the software tools that will be used to develop the proposed system. For the user interface, the framework of choice chosen for the project is flutter. The primary language of flutter is a programming language called dart. Firebase will be used to model the backend infrastructure of the system.

#### 2.5.1 Flutter Framework

Flutter is a cross-platform UI development kit created by Google and published as an open source project to the public in May 2017. Using a single codebase, it is possible to create software for Android, iOS, the web, Linux, Mac, Windows, and more platforms. C, C++, and the Dart programming language are actively used to develop the framework.

Flutter applications run on the Dart virtual machine, which enables hot reloading of changes made to the source file without having to recompile the entire application. Flutter apps are compiled ahead-of-time (AOT) in order to achieve faster performance when released, i.e. directly to machine code regardless of the CPU instruction set or to JavaScript if targeting the web. (flutter.dev, 2021).

Flutter is a layered system that may be expanded. Each layer is made up of libraries that are dependent on the layer behind them. Each aspect of the framework is designed to be optional and changeable, and no layer has privileged access to the layer below it. Flutter applications are bundled in the same way that any other native app (applications built using platform specific languages such as java for android and swift for iOS) is. This is accomplished through the use of a platform-specific embedder that communicates with the operating system to gain access to services like render surface configuration. (flutter.dev, 2021).

#### 2.5.2 The Dart Programming Language

Dart is a client optimized programming language that can be used to create quick applications on any platform. Google created the language, and it was originally launched on October 10th,

2011. Dart is a garbage-collected, object-oriented, class-based language with C-style syntax. When used on mobile devices, it compiles to native code, and when used on the web, it compiles to JavaScript.

Flutter uses the Dart programming language as its language of choice. Both dynamic and static types are supported. Dart's dynamic nature is enabled via the var keyword, as well as language data types like integers, strings, double, Boolean and others data types that allow for static style typing in the language. Dart enables concurrency through the usage of isolates, despite being a single threaded programming language.

### 2.5.3 Firebase Mobile Development Platform

Google's Firebase is a mobile application development platform. It includes a set of handy tools for developers to swiftly design and scale apps based on user demand. It's called a mobile backend as a service since it automates backend development, allowing the developer to concentrate on the frontend or client side. Firebase comes with a plethora of services that are useful for building applications, they are as follows.

- Firebase authentication: provides easy to use SDKs and readymade UI toolkits for securing your applications users. It accepts a variety of authentication methods, including email, phone number, Google, Facebook, Twitter, and GitHub, among others.
- Firebase web hosting: provides an enterprise-grade web content hosting platform. It allows both static and dynamic web applications to be deployed unto the internet.
- Cloud firestore: a scalable NoSQL document database for mobile and web applications that enables for data storage, synchronization, and querying of data.
- Cloud functions: enable the automatic execution of backend/server side code in response to particular event triggers from firebase features and https requests.
- Cloud storage: SDKs allow for file uploads and downloads within Firebase applications, regardless of network quality. Cloud storage can be used to store images, movies, audio, and other user-generated content.

## 2.6 Review of Related Works

Haikun, Tiemin, Cheng, Jiayuan, & Yang, (2021) developed a poultry farm information management system. They incorporated the use of cloud databases to store and synchronize all farm generated data. The system was broken up into 3 layers i.e. the management system application layer, data servicing layer and an information sesning layer. Each layer functioned independently of the other while also cooperating with one another to form the information

management system. They were able to integrate wireless sensor networks into the aplication to collect environmental data from farms which was in turn analyzed to monitor the health of the farm.

Esnola-Gonzalez, et al., (2020) worked on a PCM (poultry chain management) platform whose aim was to collect data across the different phases in a poultry supply chain. The collected data was in trun used to analyze the quality of each phase and further used to identify critical issues which caused ineffciencies within the chain. Data collected was also used to support decision making of stakeholders involved in the chain. However, not all the data collected across certain phases in the supply chain could be automated. Use of file storage system was employed in such phases which caused integration to the centralized PCM platform to be rather not as seamless as it could be.

In a research work done by Balasaheb et al., (2020) a technological based solution was highlighted to make use of intelligent systems which utilized an embedded framework and a smart phone for monitoring farm environmental parameters. The goal was to design a fully decentralized automatic control system which made use of IOT (internet of things) based technology to store all the sensor data gotten from the farm in a cloud database.

Fajemisin & Ogunribido, (2018) worked on a research paper which focused on identifying the opportunities which lay along the Nigerian poultry supply chain. They were able to highlight in their paper that poultry farmers are the major stakeholders in a poultry supply chain. In their paper also it was highlighted that poultry waste materials also tend to go to waste due to lack of materials needed to recycle such by-products.

Goud & Sudharson, (2015) developed an internet based smart poultry farm mobile system which integrated wireless sensors to alert persons in charge of farms who had registerd mobile numbers about various environmental parameters such as temperature, humidity and water levels. One of the major draw backs of their system was the use of GSM to send alerts to persons in charge of the farms. Such a system that rely on GSM for its information transfer flow will fail in areas of no service.

# CHAPTER THREE
# METHODOLOGY

## 3.0 Introduction

The adopted methodology used in designing the software are discussed in detail in this chapter. The functional and non-functional requirements of the system are also identified with their method of identification. The overall system architecture is introduced in subsequent sections. With the aid of UML (Unified modelling language) diagrams, illustrations are given on the various system components so as to provide better understanding on how these system components interact. The subsequent sections provide a better understanding on how the Frontend and Backend of the system was designed

## 3.1 Method of Identification of User and System Requirements

This section gives an overview on how the system and user requirements for the application were identified. The identified system and user requirements were sourced through the use of informal surveys, online articles, books, journals etc. Observations were carried out on existing systems being used in real world scenarios. From the result of these observations, features that needed to be implemented in the system were identified.

For the system requirements, the first point of concern was identifying what platform the system should run on i.e. web, mobile or desktop. In order to make an informed decision, an analysis was carried out on the age demographic of poultry farmers in Nigeria. It is estimated that 20% of farmers are young adults within the ages of 24–35 (Liverpool-Tasie, et al., 2017). Informally, most poultry farmers make use of their mobile devices to carry out their business activities as it makes keeping up with customers much easier. Hence, the decision to build the software as an android application.

An informal survey was carried out on the business practices of small and medium scale poultry farmers. Tools used for the survey include information from articles on the current landscape of poultry businesses in Nigeria, videos explaining the current trends and monitoring of farm activities from actual poultry farms. From this informal analysis, it was determined that the two essential users which made up a typical poultry supply chain were the farmers themselves and the distributors responsible for buying the farm products.

### 3.1.1 System Requirements

a) Functional requirements: below is a list of all the identified functional requirement of the system.

1. New users should be able to register unto the platform with their email and password.

2. Farmers should be able to perform stock count of items (eggs and chickens) on their farm.
3. Distributors should be able to book orders from farmers.
4. Farmers should be able to confirm or decline a distributor's order request.
5. Farmers should be able to create new orders for distributors who are not using the platform.
6. Farmers should be able to adjust prices of their products.

b) Non-functional requirements: below is a list of the identified non-functional requirements of the system.
    1. All users shall be authenticated via their email and password before gaining access to the platform.
    2. The user interface of the mobile application ought to be intuitive and responsive at all times.
    3. The designed system shall be portable on all android devices, regardless of the phones screen size.

c) Hardware requirements: In order for the specified software to work, the below hardware specifications must be met.
    1. RAM (500MB or greater)
    2. CPU (single core processor or higher)
    3. Storage space (1GB or greater)
    4. Internet connectivity (3G or higher)
    5. An android smartphone

d) Software requirements: In order for the specified system to work, the below software specifications must be met.
    1. Android OS (5.0 or greater)
    2. Android SDK version (16 or greater)

### 3.1.2 User Requirements

From the above section 3.1.1, six functional requirements were identified. In the below paragraph, a list of the corresponding user requirements to each functional requirement is listed out.

a) All users shall be authenticated via their email and password before gaining access to the platform.
    i. Users should input both their email and password in their specified input fields, the fields shall be validated to meet correct standard i.e. email must follow email format and the password must be 6 digits long.
    ii. Once the fields have been validated for correctness, a requests is sent to the Firebase authentication backend to check if the email has been registered. If the email is not registered, feedback is sent to the user via the UI that the email is not registered. If the email is registered but the password is incorrect, user receives feedback also. If both email and password are validated by the backend as valid, user gets approved and gains access to the platform.
b) Farmers should be able to perform stock count of items (eggs and chickens) on their farm.
    i. Farmers have the option of filling stock count for both eggs and chickens independently.
    ii. On selecting egg stock count, system navigates to the egg stock screen where a form is made visible. They are required to fill two fields, the total eggs collected and the number of bad eggs for the day.
    iii. Once done with filling the fields, the confirm button is tapped on and the stock count for the day gets stored in the database.
    iv. On selecting chicken stock count, system navigates to the chicken report screen where farmers perform stock count in batches i.e. chickens are grouped into batches. They have the option of creating a new batch of chickens or doing stock count on an already existing batch.
    v. On tapping on a batch of chickens, they fill a form which contains two fields, the number of dead chickens and the comment field where reasons are given. Once done, they tap the confirm button and the data gets stored in a database.
c) Distributors should be able to book orders from farmers.
    i. Distributors have access to the order menu where they can select from a list of farms they want to book their order from. On picking a farm, their details get displayed.
    ii. The distributor is then required to pick the items they want to order from the farm with their quantity. A price summary is then displayed at the bottom of the screen. Once done they tap on the confirmation button and the order details get sent to the farm.
d) Farmers should be able to confirm or decline a distributor’s order request.

i. All orders show up as either open or closed within the application. Open orders refer to orders that have not been attended to by the farm and vice versa.
ii. On tapping an open order, the farmer has an option of either fulfilling or declining the order request. Once either option is chosen, the order status changes to that of closed.

e) Farmers should be able to create new orders for distributors who are not using the platform.
   i. On the order section of the app farmers have access to the create order button.
   ii. On tapping, they get navigated to the order form screen, where they can input details of the order they want to create for a customer. Once done, they tap on the confirm button and the order details get saved on the database.

f) Farmers should be able to adjust market prices of their products.
   i. On the account section, an option to adjust product prices is made available.
   ii. On tapping, a form field gets shown with two options one for changing the unit price of a crate of egg and the other for changing the unit price of a chicken.
   iii. Once done, tapping the confirmation button shows an alert dialog with details of changes made. The data gets stored in both the system cache and on the database.

### 3.2 System Design Methods

The system was built using extreme programming. The method of determining the functional requirements was through an iterative approach. First an initial version of the software was designed with select features. After which the version was exposed to proposed users of the software for constructive criticism so as to check if the version meet its specified functional requirements. Through this process, omissions in the software were identified and the next iteration/release cycle focused on implementing the new refined functional requirements.

#### 3.2.1 Architectural Diagram

The system architecture illustrated below was modelled using the layered architectural design. Each layer represents the different system components with the arrows providing description on the interaction between each layer. The firebase authentication and cloud firestore SDKs provide abstractions for communicating with the database of the software.

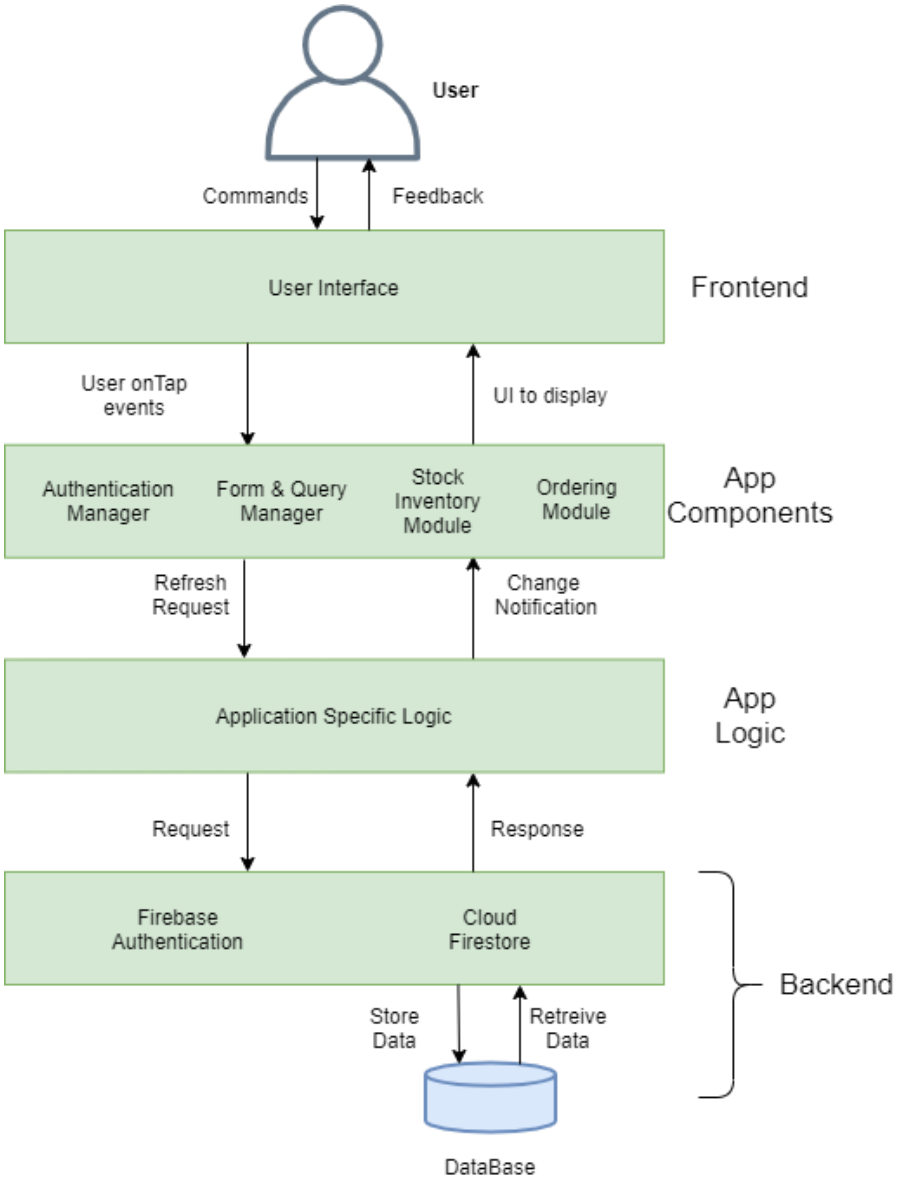


Figure 3.1: System architectural diagram

### 3.2.2 Use Case Diagrams

This section gives an illustration of the various actors/system users of the designed software and their interactions with the system using use case diagrams.

a. Admin use case diagram: the role of the admin is to manage the interactions between the two other system users i.e. the farmers and distributors. They are responsible for performing CRUD (create, read, update, delete) operations on user generated content in the database. Figure 3.2 illustrates the administrator use case scenarios.
b. Farmer use case diagram: figure 3.3 illustrates the farmer use case diagrams. They are responsible for either confirming or cancelling received orders from distributors and inputting farm stock inventory data for both eggs and chickens.

c. Distributor use case diagram: the distributor's primary role is to provide farmers with alternatives for selling their farm goods, by making order requests to farmers with their contact details. Once a requests has been sent, the farmer receives the request and can decide to perform on of two actions on the order, either cancel of confirm. The action taking by the farmer reflects on the distributors UI after which contact is made between both users in the real world to complete the transaction.

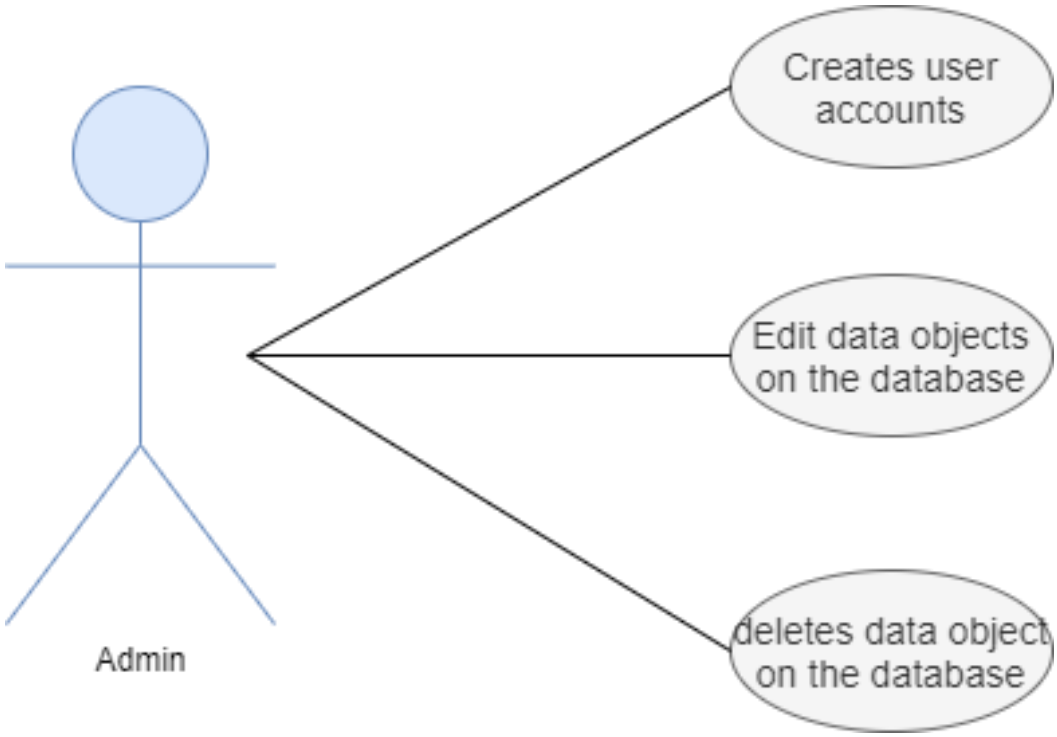


Figure 3.2: Admin use case diagram

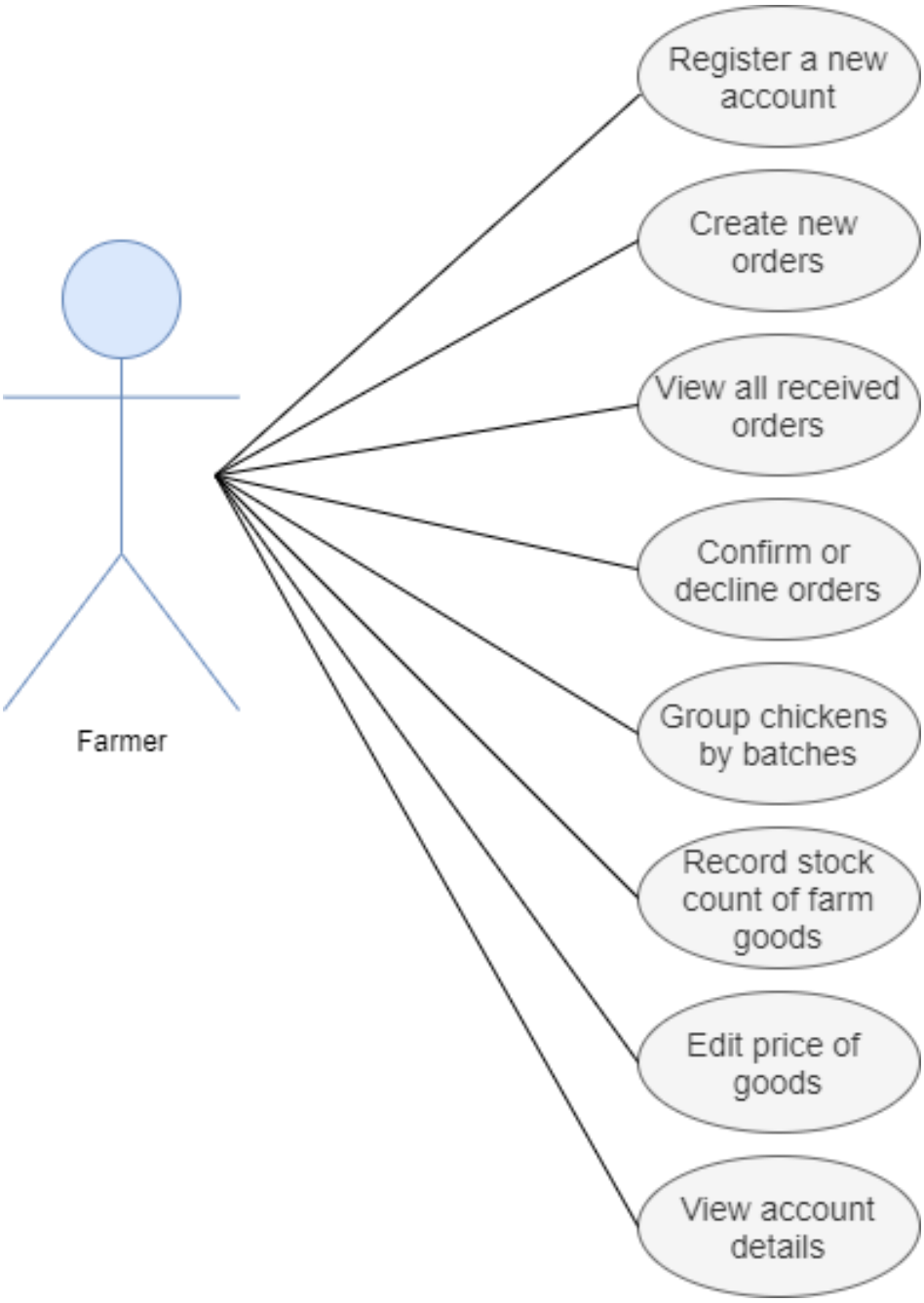


Figure 3.3: Farmer use case diagram

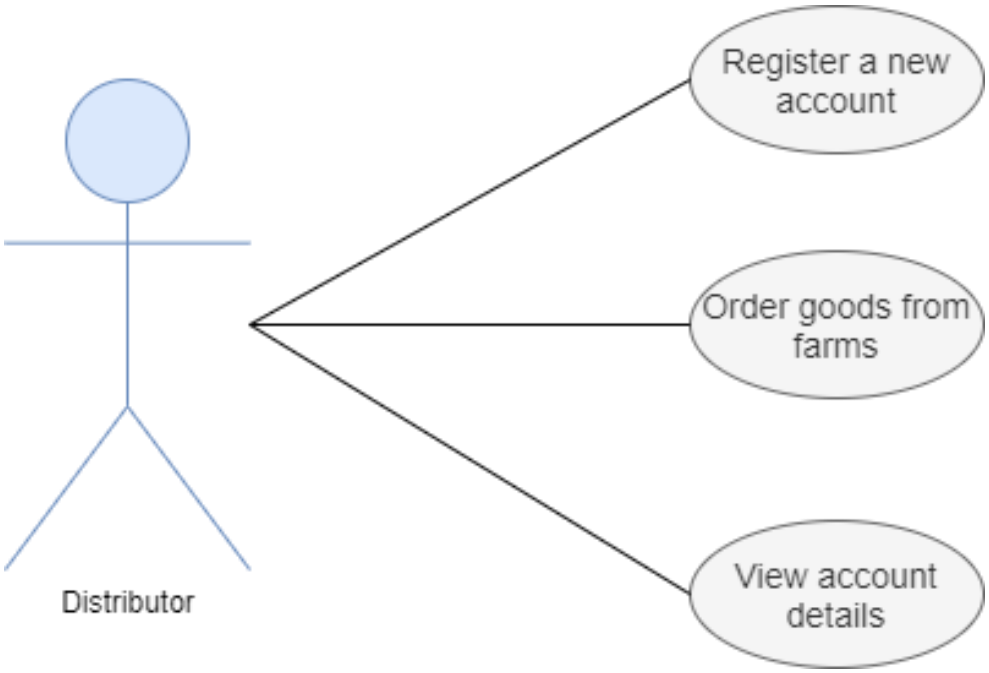


Figure 3.4: Distributor use case diagram

### 3.2.3 Sequence Diagram

The below sequence diagram illustrate the sequence of activities that take place within the ordering module of the system for the distributor user.

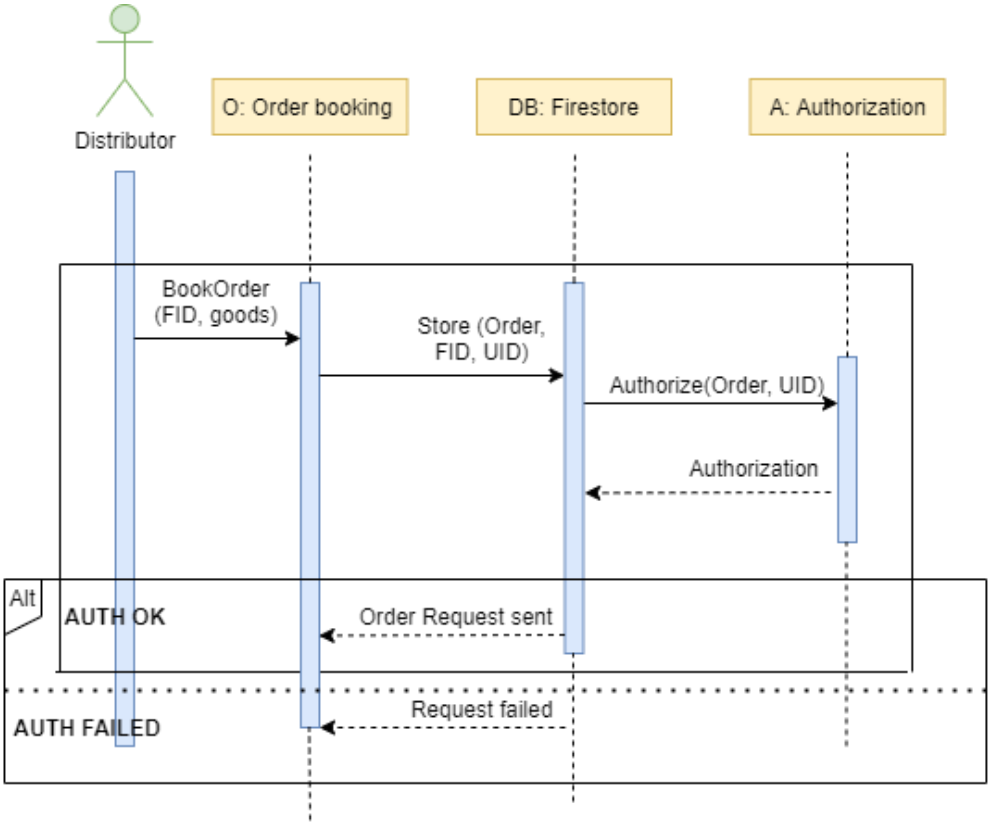


Figure 3.5: Ordering sequence diagram

### 3.2.4 Activity Diagram

Figure 3.6 illustrates the activity diagram of the system showing the typical workflow involved with using the system. Each rectangle represents a state in the application, with the arrows and text providing details on certain actions that drives the movement of one state to the other.

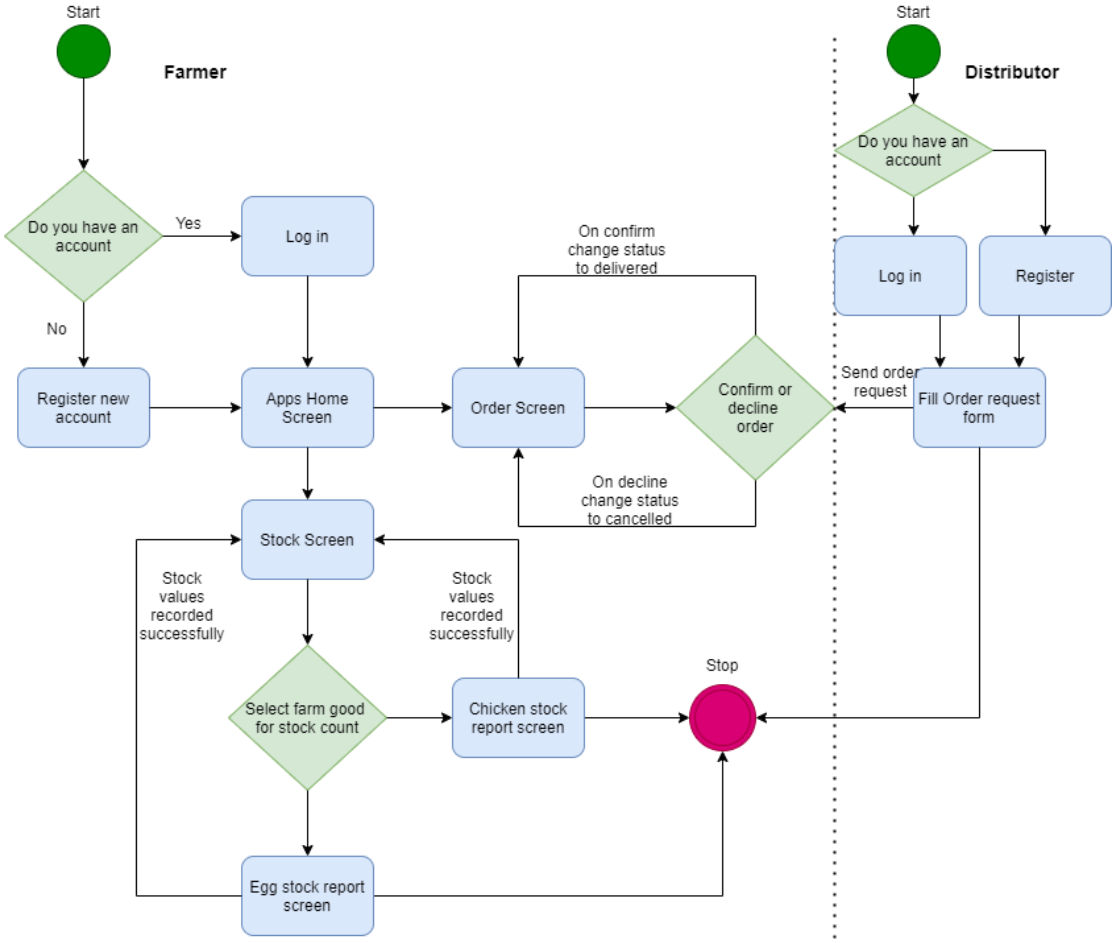


Figure 3.6: Activity diagram illustrating systems workflow

### 3.2.5 Class Diagrams

The specified system was built using the Object Oriented Programming paradigm. As such classes were used to model the objects of the system. Below are UML class diagrams illustrating the various classes of the proposed system.

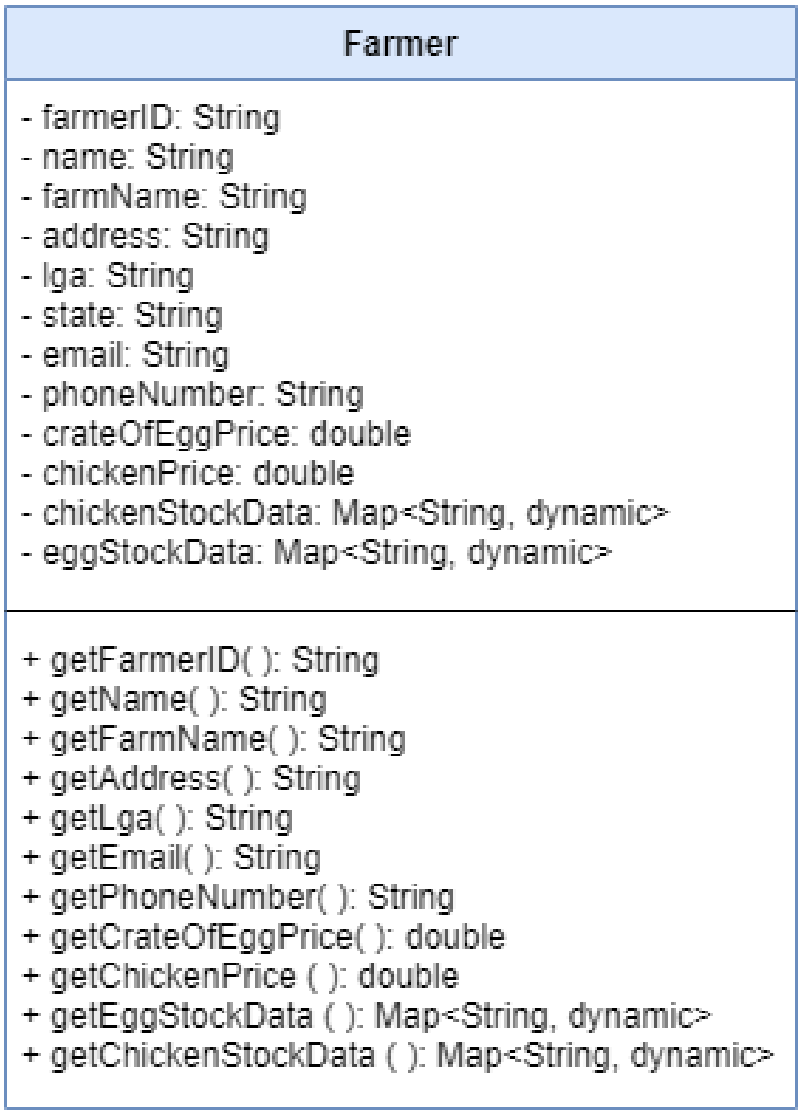


Figure 3.7: Farmer class diagram

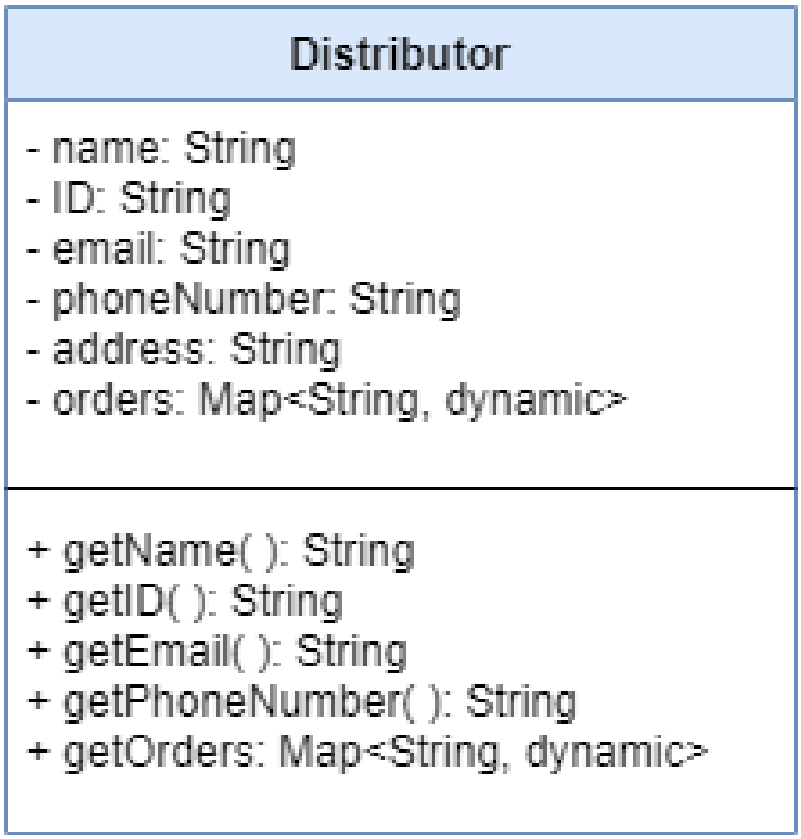


Figure 3.8: Distributor class diagram

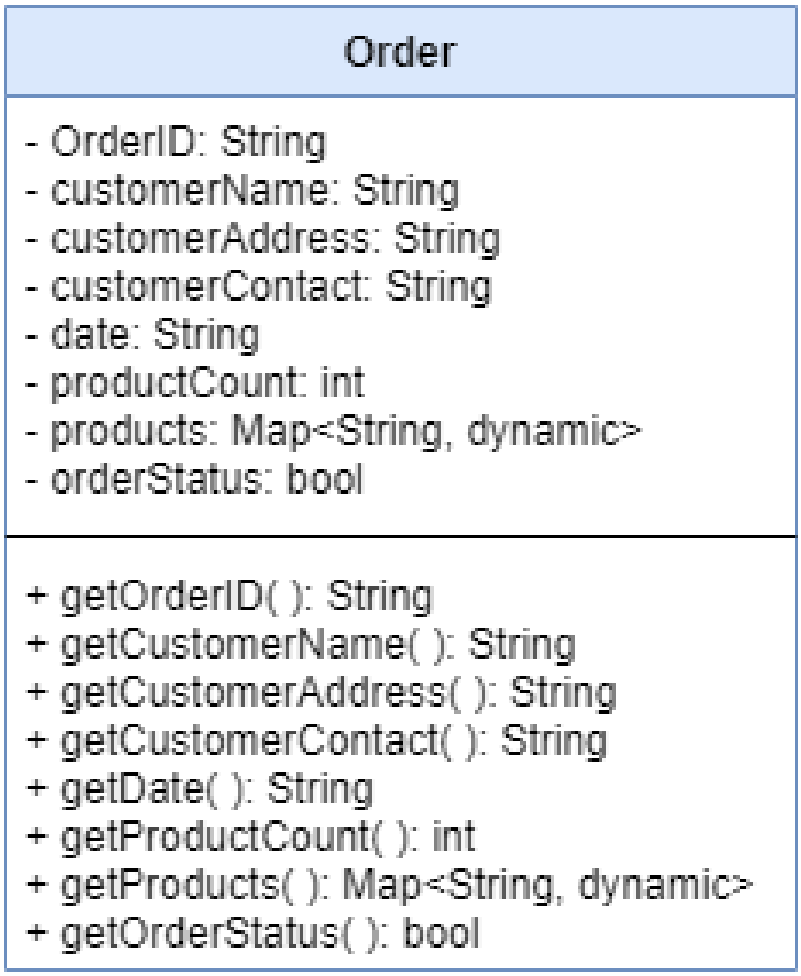


Figure 3.9: Order class diagram

### 3.3 System Testing Approach

The designed system will be tested for correctness using two methods of testing namely

i. Unit testing: unit testing was done using the flutter test package. It provides functionality out of the box to test the classes and logic of the code written for the system. The goal of having these unit tests set up is to ensure that whatever new code or new functionality that gets added to the code base, don't end up breaking the system. Unit tests were written for all the major classes of the system and tested before every build of the application was run.

ii. Acceptance testing: this will be done on completion of the project to ensure that the designed system meets the specified needs of the identified users. Whatever omissions to the software or bugs that are discovered at this stage of testing will be duly noted and fixed.

# CHAPTER FOUR
# IMPLEMENTATION AND TESTING

## 4.0 Introduction

This chapter shows the implementation of the system. For the purpose of implementation, the application was run on an android simulator. The source code for the application can be found on GitHub. The backend was simulated on the Firebase emulator suite which is an offline tool for testing out applications. Screenshot of the application shown in this section are actual snapshot of the systems user interface. The system was developed using the dart programming language.

## 4.1 Backend Implementation

The backend of the system was implemented using googles firebase SDK (software development kit). The SDK provides multiple firebase services, but for the purpose of building the system, only two were needed which are Firebase authentication for validating users with their registered email and password, and Cloud firestore as the database for storing all user generated content.

For testing purposes, the firebase authentication and cloud firestore were run in a test environment using the emulator suite. To begin running an instance of the cloud firestore and firebase authentication emulator on your laptop the following steps need to be followed.

i. Clone the source code from the GitHub link https://github.com/ngbede/poultry.
ii. Have flutter installed and working on your laptop.
iii. Have the firebase emulator suite installed on your laptop.
iv. Navigate to the project root directory on the terminal.
v. Install all the package dependencies from the pubspec.yaml file.
vi. Run the below command in the terminal.
    firebase emulators:start --import=./db --export-on-exit
vii. The firebase authentication and cloud firestore instance will run now run on port 9099 and 8080 respectively.

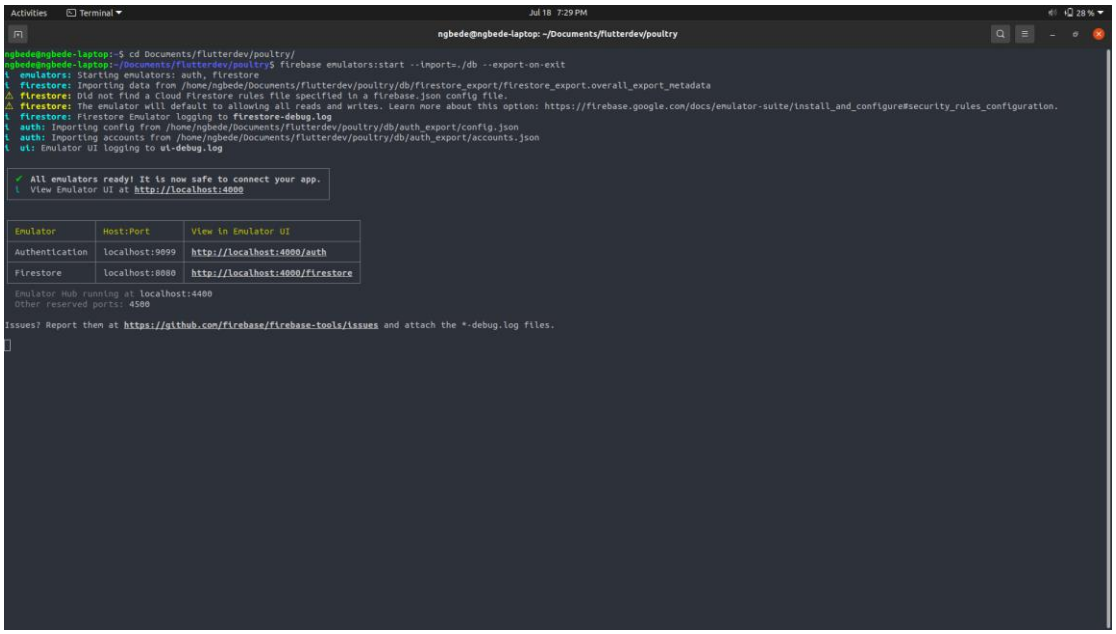


Figure 4.1: firebase emulator suite launched via terminal

### 4.1.1 Database Implementation

The database of the system was implemented using firebase's cloud firestore. Cloud firestore is a No-SQL database, which stores data objects in the form of JSON (JavaScript object notation). The database tables are segregated in the form of collections with each document in the collection having its own attributes.

Figure 4.2 shows the collections present in the applications instance of cloud firestore. There are five collections which are farmers, orders, stock_chickens, stock_eggs and the users collections. They all store different kinds of information using JSON style syntax in respect to their collection names.

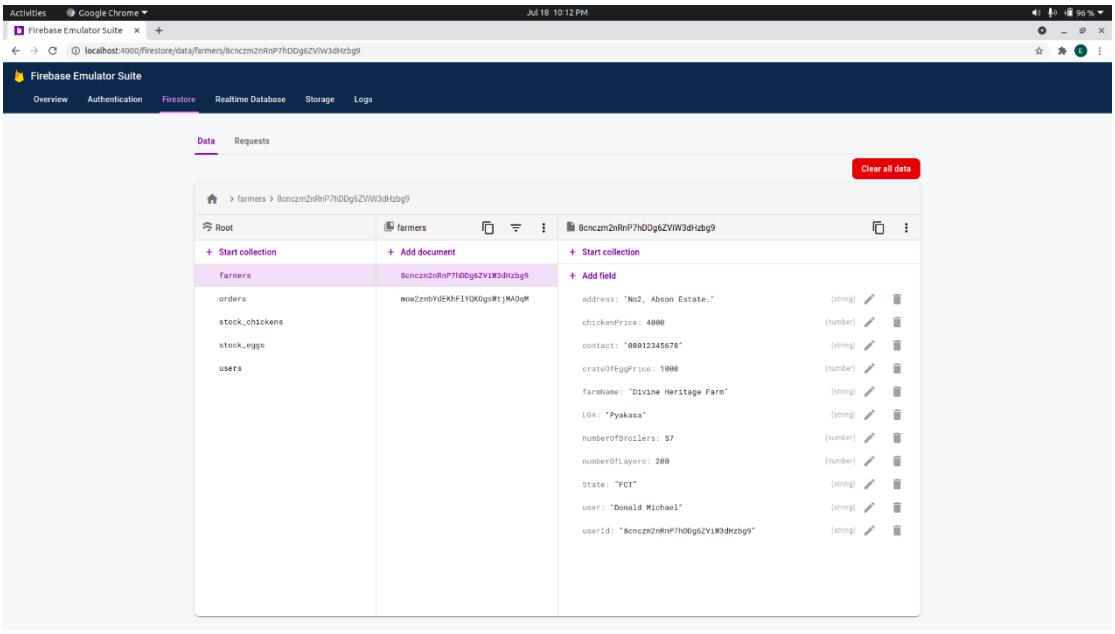


Figure 4.2: Cloud firestore collections

### 4.1.2 Authentication Implementation

Firebase authentication was used to implement the user validation via the use of email and password. New users were on boarded unto the platform using the register with email and password method of the authentication SDK. Once an email was taken by a user, it can no longer be used to register unto the platform. The constraints set for the authentication input are listed below.

i. The email must be a valid email.
ii. The email must not be in use by any other registered user i.e. it must be unique.
iii. The password must be at least six digits long.

Once an email is registered, it can then be used to log unto the platform with the valid credentials.

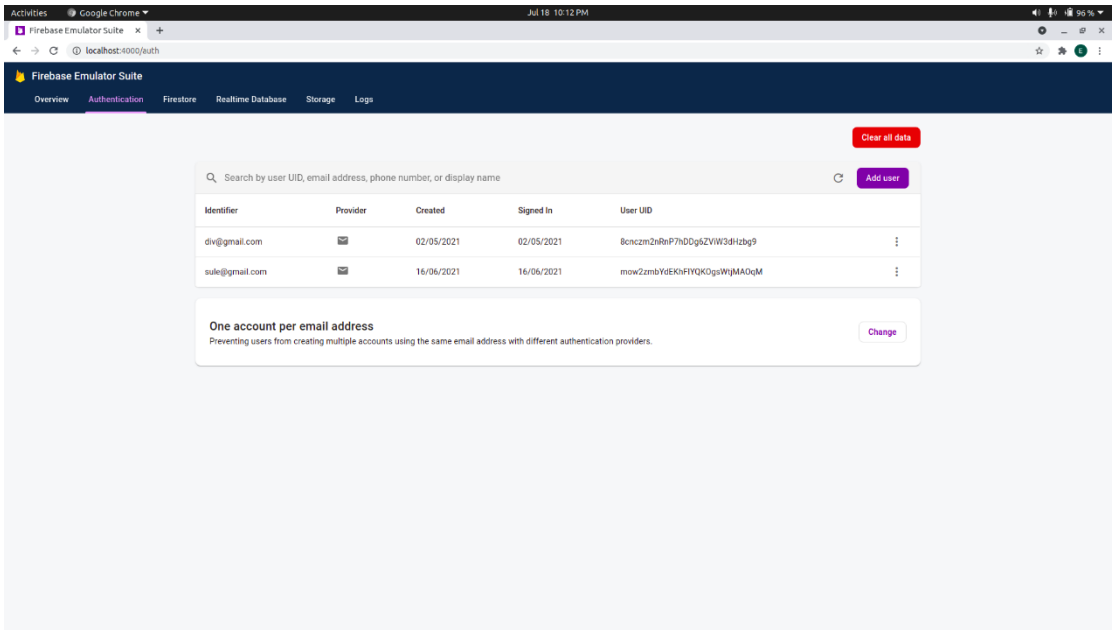


Figure 4.3: Firebase authentication panel

## 4.2 Frontend Implementation

The frontend of the application was implemented using flutter. Adobe XD was used to prototype and test out some of the UI concepts that were required to run the application. With the use of the flutter widget library, the user interface was built to be intuitive and responsive. Subsequent sections show the UI implementation of the application.

### 4.2.1 Registration and Login

New users are on boarded unto the platform via the registration screen. They are required to fill in their details in the form provided. A new user is categorized into one of two user types which is either a farmer or a distributor. For users who select the farmer class of users, they are redirected to fill a form on details about their farm such as the name, address, price of chicken crates etc. Distributors on the other hand get redirected to the apps main section on registering.

After a user has been on boarded unto the app, they can now make use of the login screen where they are required to only input their email and password for authentication. All user errors when filling the login and registration form are handled using form validators which check to see if field inputs are valid.

Below are screenshots of the applications onboarding screens.

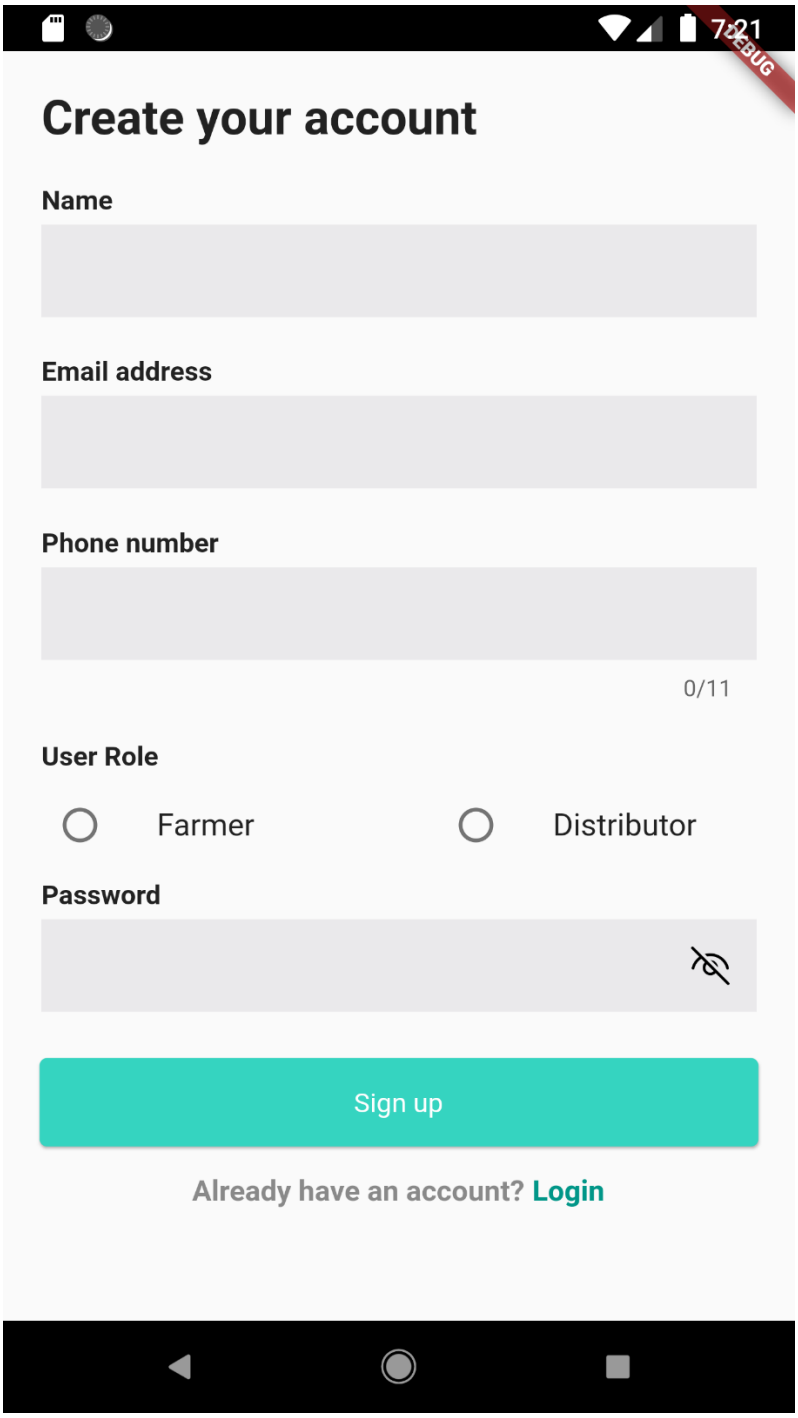


Figure 4.4: Registration screen for new users

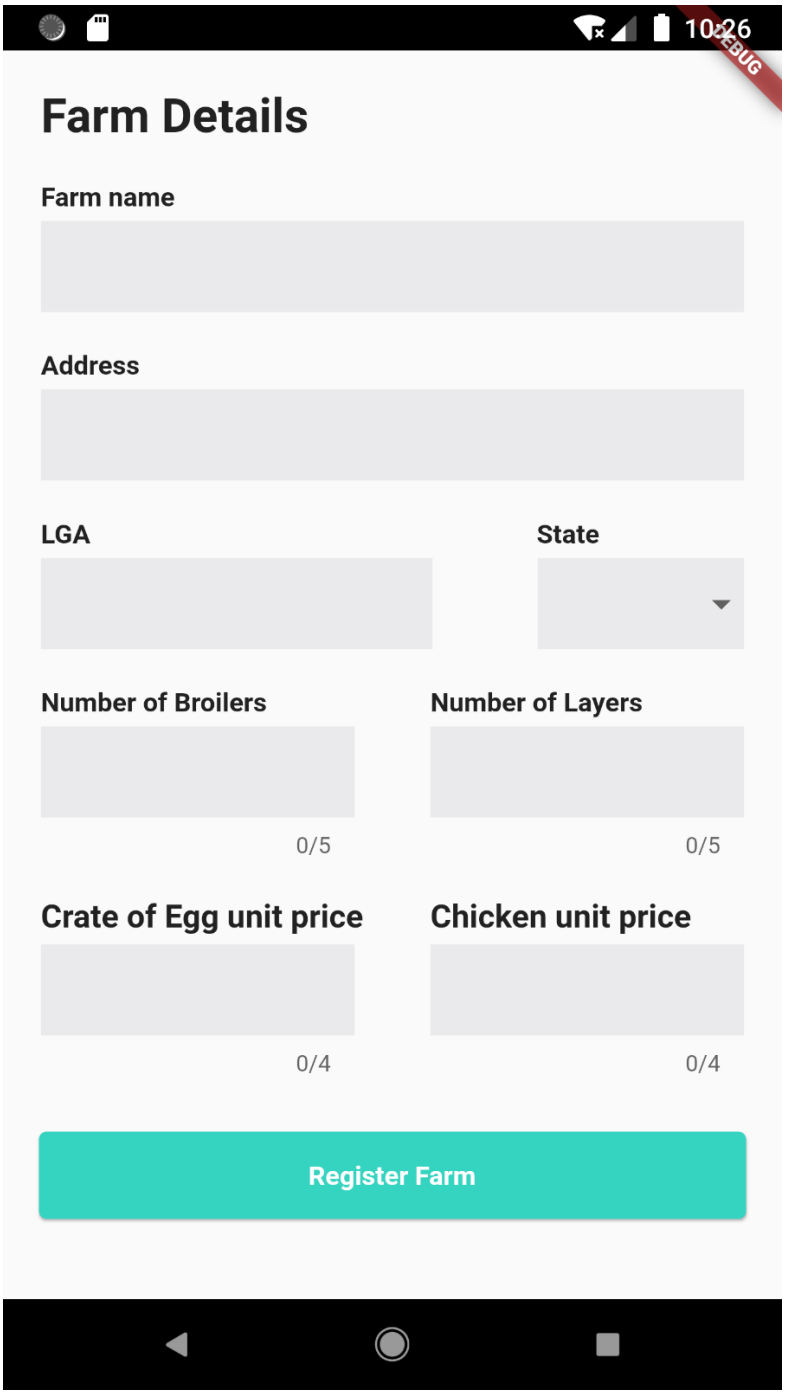


Figure 4.5: Farm details form

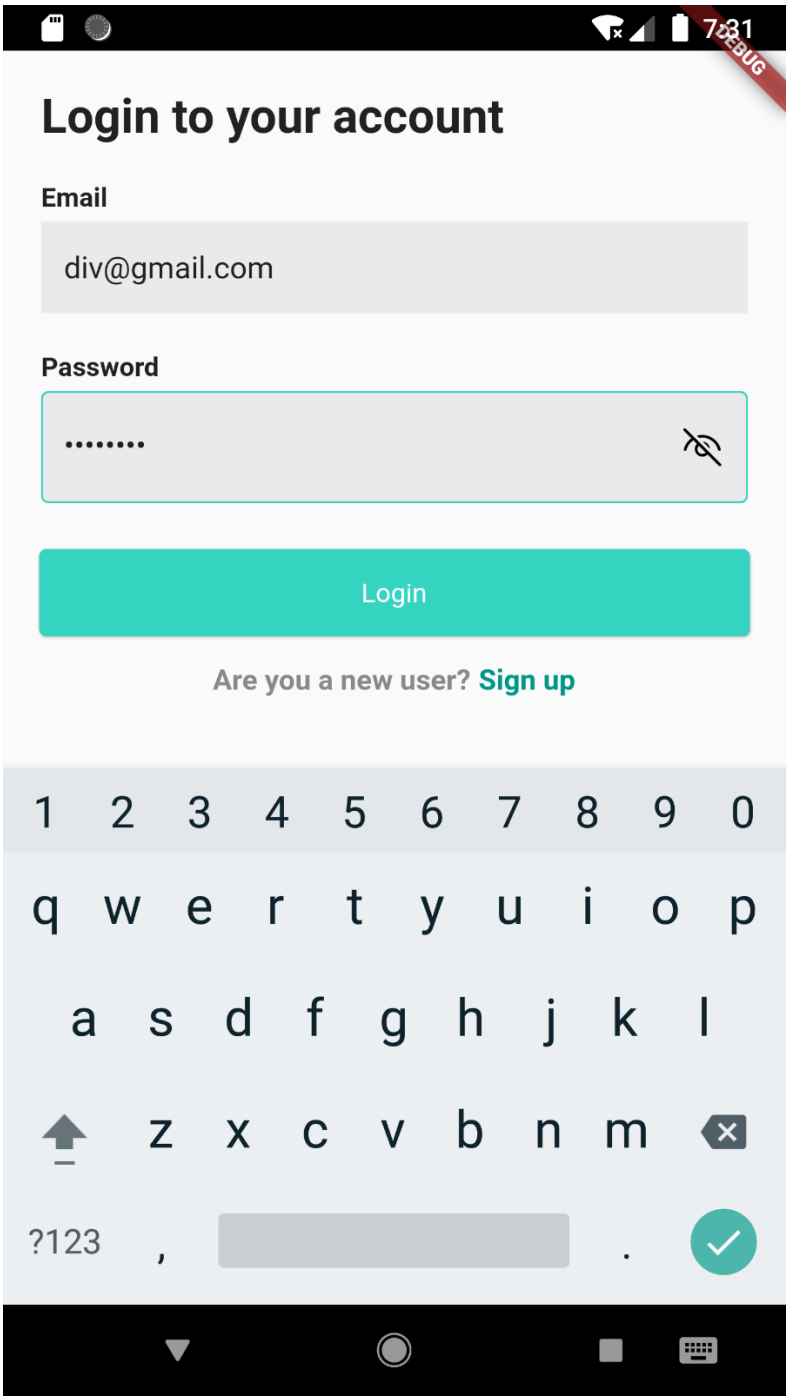


Figure 4.6: Login screen

### 4.2.2 Home Screen

The home screen was designed to show quick information on a farms current state. It also provided quick navigation to the key functionalities in the app such as orders and stock count.

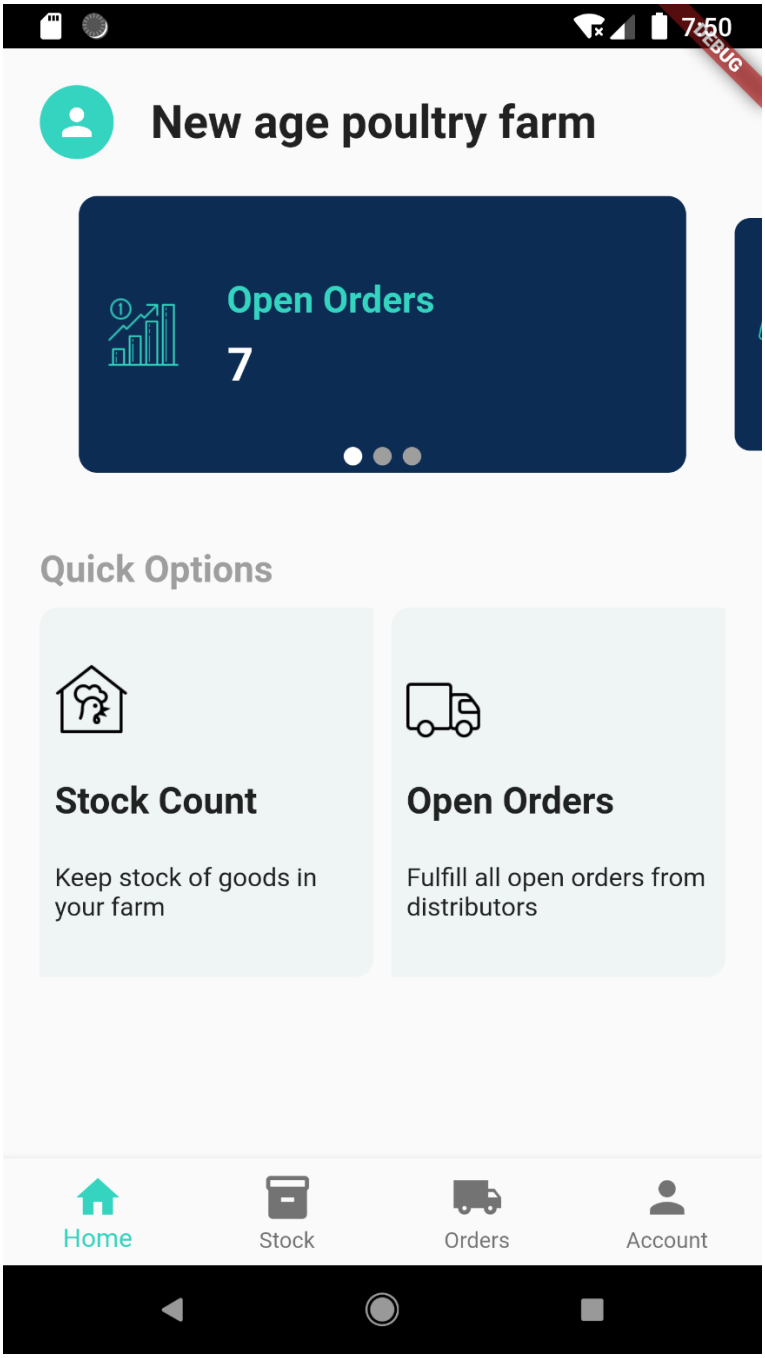


Figure 4.7: Home screen UI showing farm details

### 4.2.3 Stock Counting Module

The stock inventory module allows farmers to record their daily stock of their farm goods, i.e. chickens and eggs. The stock reporting screen provides a form to the user to input their records

for the day. A comment field is also provided so as to leave reasons why the values were recorded. All chicken stock data were stored in the stock_chickens collection on cloud firestore, while all egg stock data were stored in the stock_eggs collection.

The chicken stock count was done in batches, i.e. for every new batch of chickens brought into a farm, the farmer could segregate the chickens into their individual batch and do stock count on a batch independent of the other batches. This was done via the add batch button.

Once stock values were recorded for both eggs and chickens, a circular progress bar would refresh to indicate the progress thus far for the stock count done for the day.

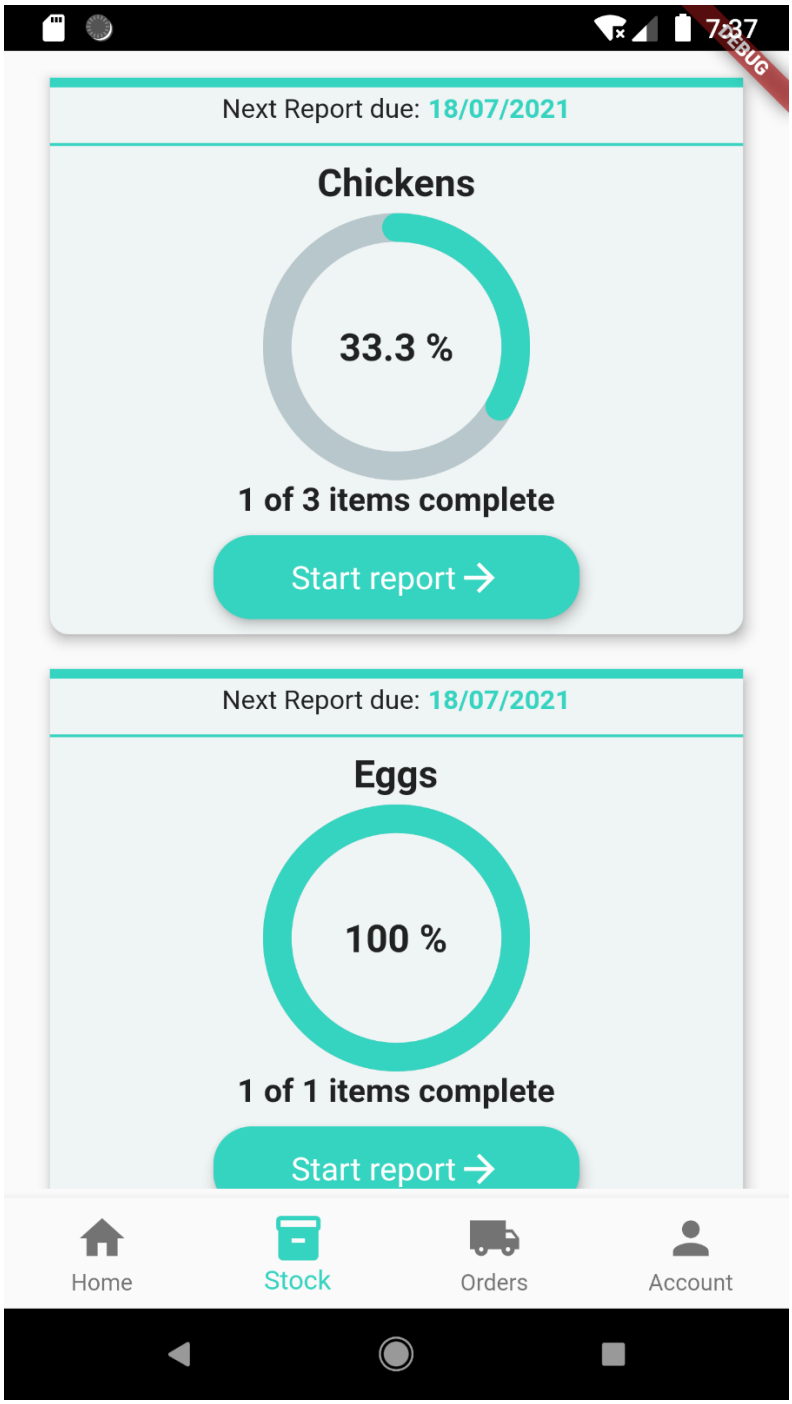


Figure 4.8: Circular progress bar indicating stock count progress

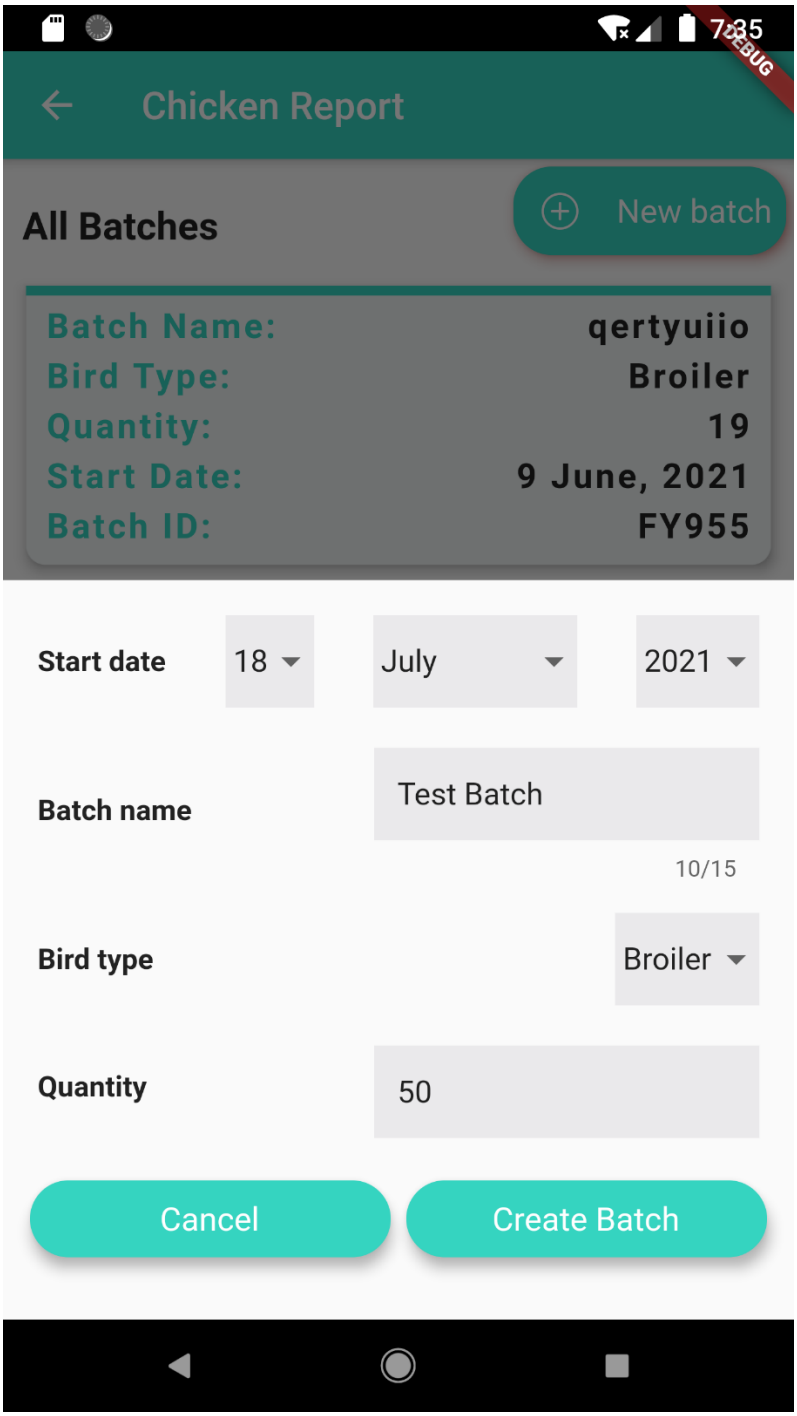


Figure 4.9: Batch addition modal form for chicken stock

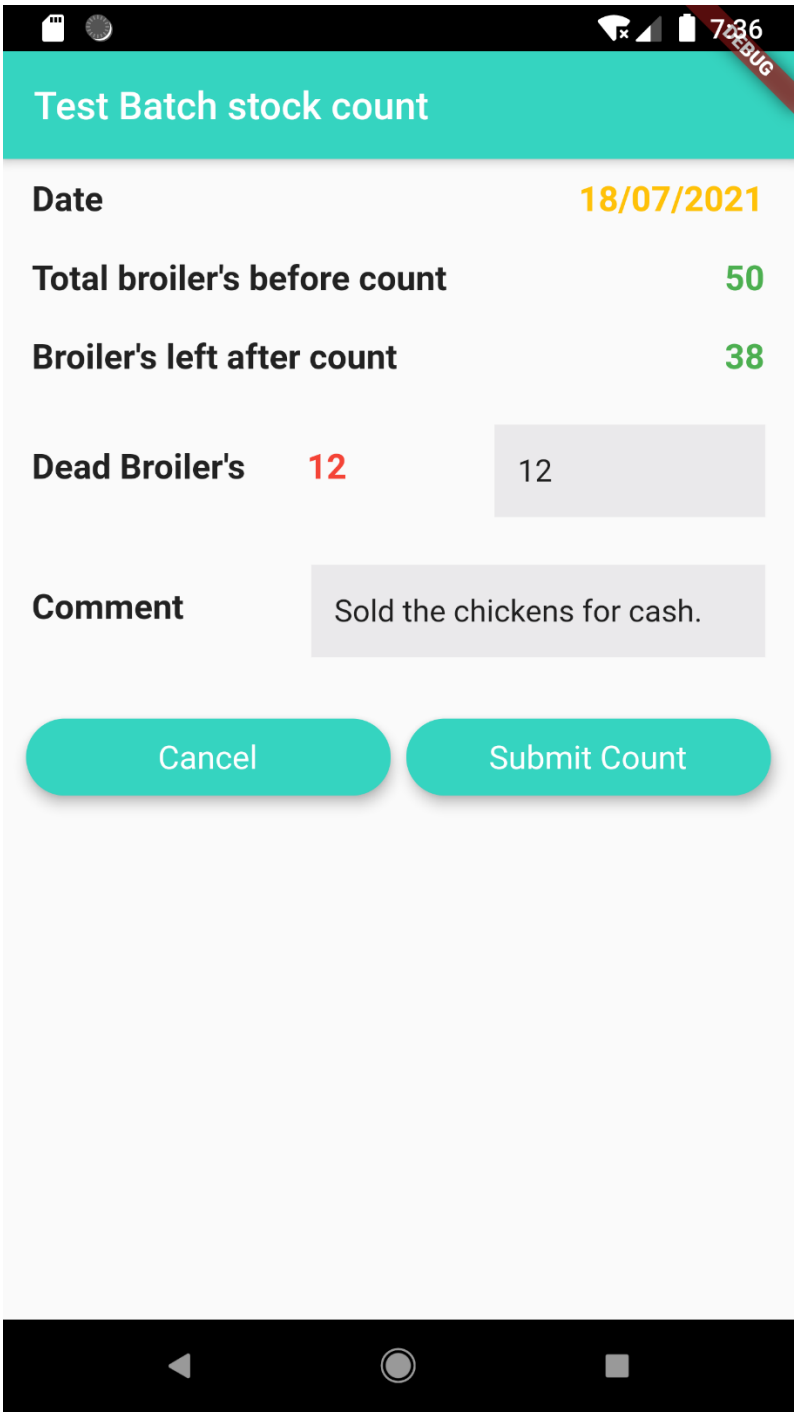


Figure 4.10: Stock count reporting form

### 4.2.4 Ordering Module

The ordering functionality begins from the distributors end. A distributor logs unto the platform using their credentials, after which they are given access to the ordering menu and their account details. On the ordering section, they have access to the book order button where they can begin the process of booking farm goods from poultry farms present on the platform. The farms current farm good price are displayed with a price summary showing the total value of all the goods they have picked. Once done with picking the items they want to order, they tap on the book order button to send an order request to the selected farm.

Once an order has been sent from a distributor to a farm, the order shows up in the farms open orders where the status of the order is set to pending. The status can only change based on the farmer's choice of action which is either to deliver the order to the specified address or cancel the order. When an order is fulfilled its status changes to delivered and it moves from the open tab on the orders screen to the closed tab. In the case where an order is cancelled, its status changes from pending to cancelled and gets moved to the closed tab.

Farmers have the ability to adjust their product prices. This way, they are able to control the current prices that show up on a distributors ordering page.

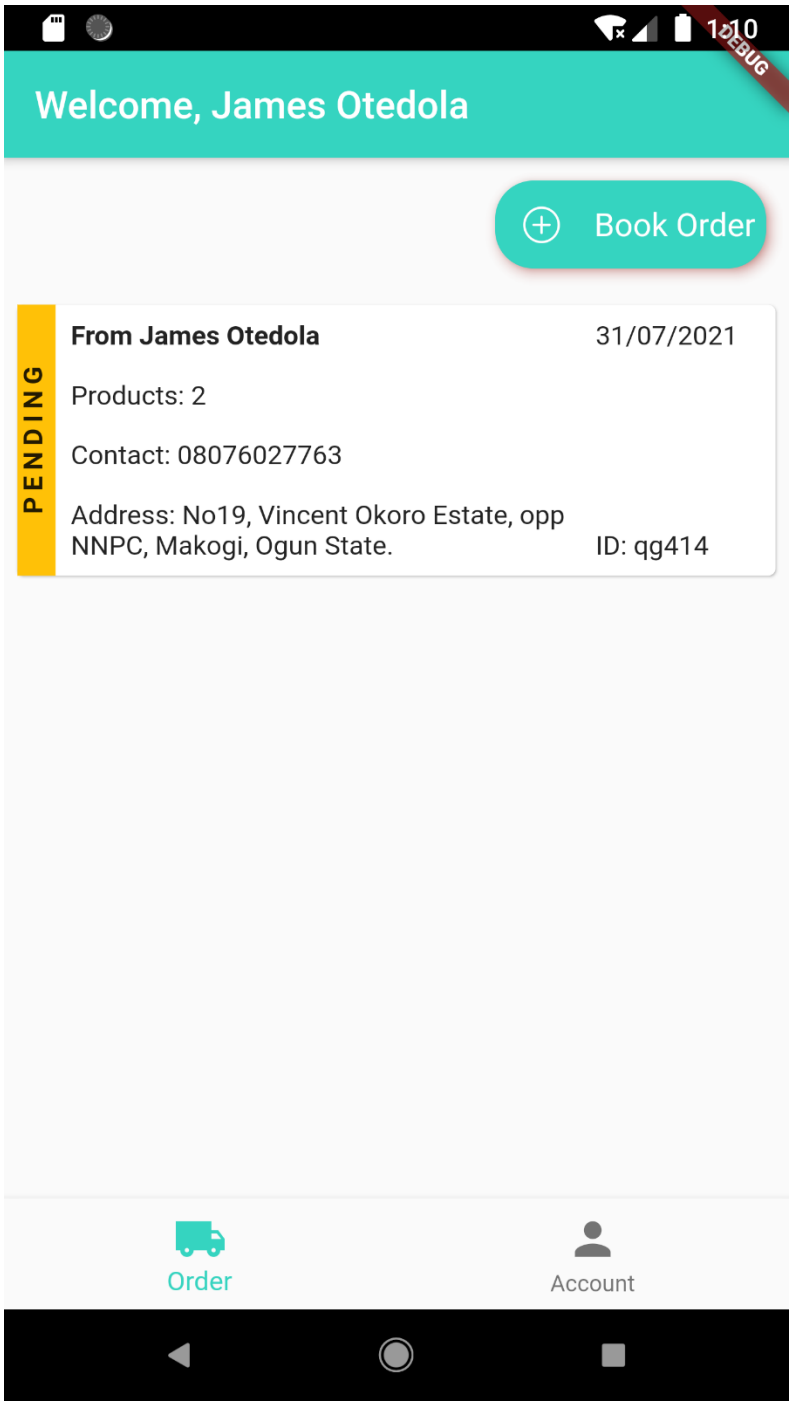


Figure 4.11: Distributors ordering menu

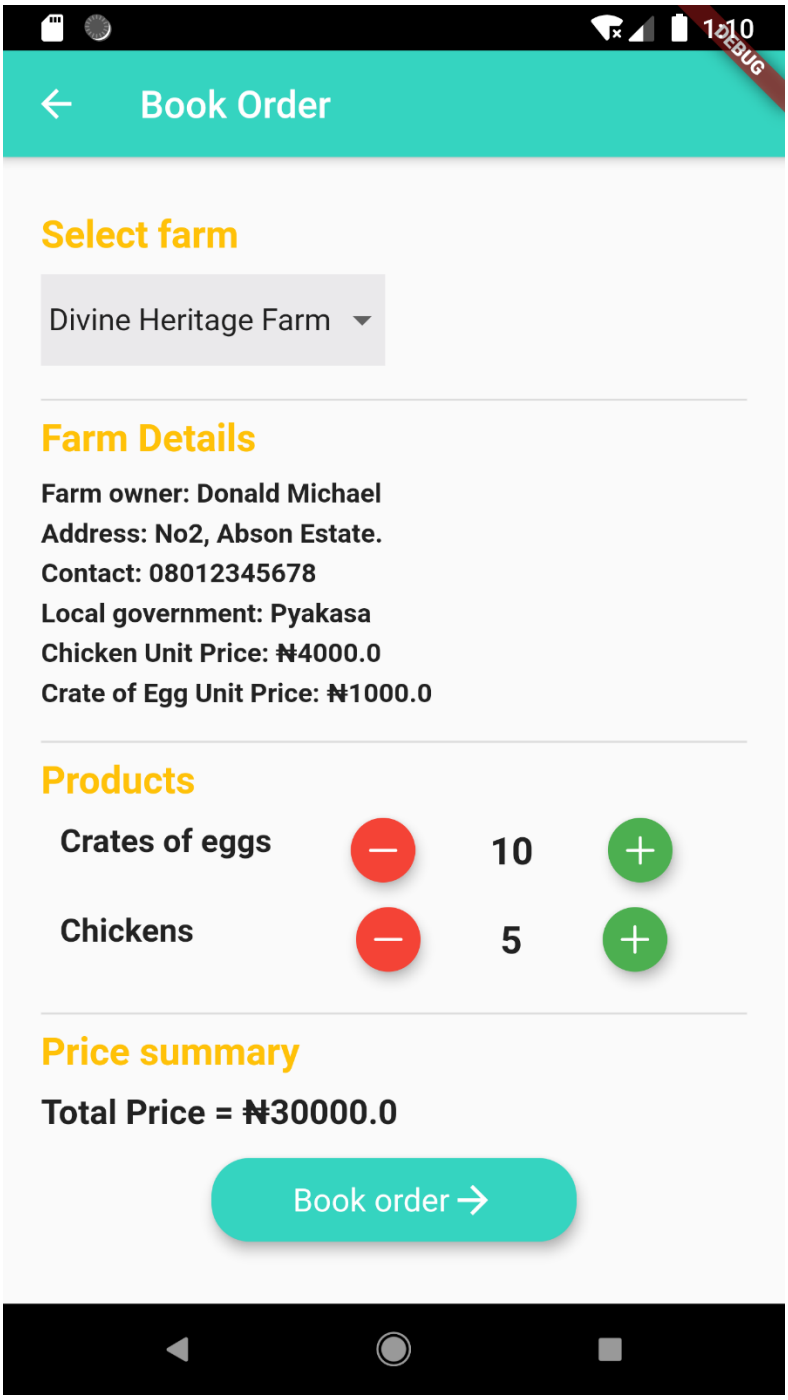


Figure 4.12: Distributors ordering form field

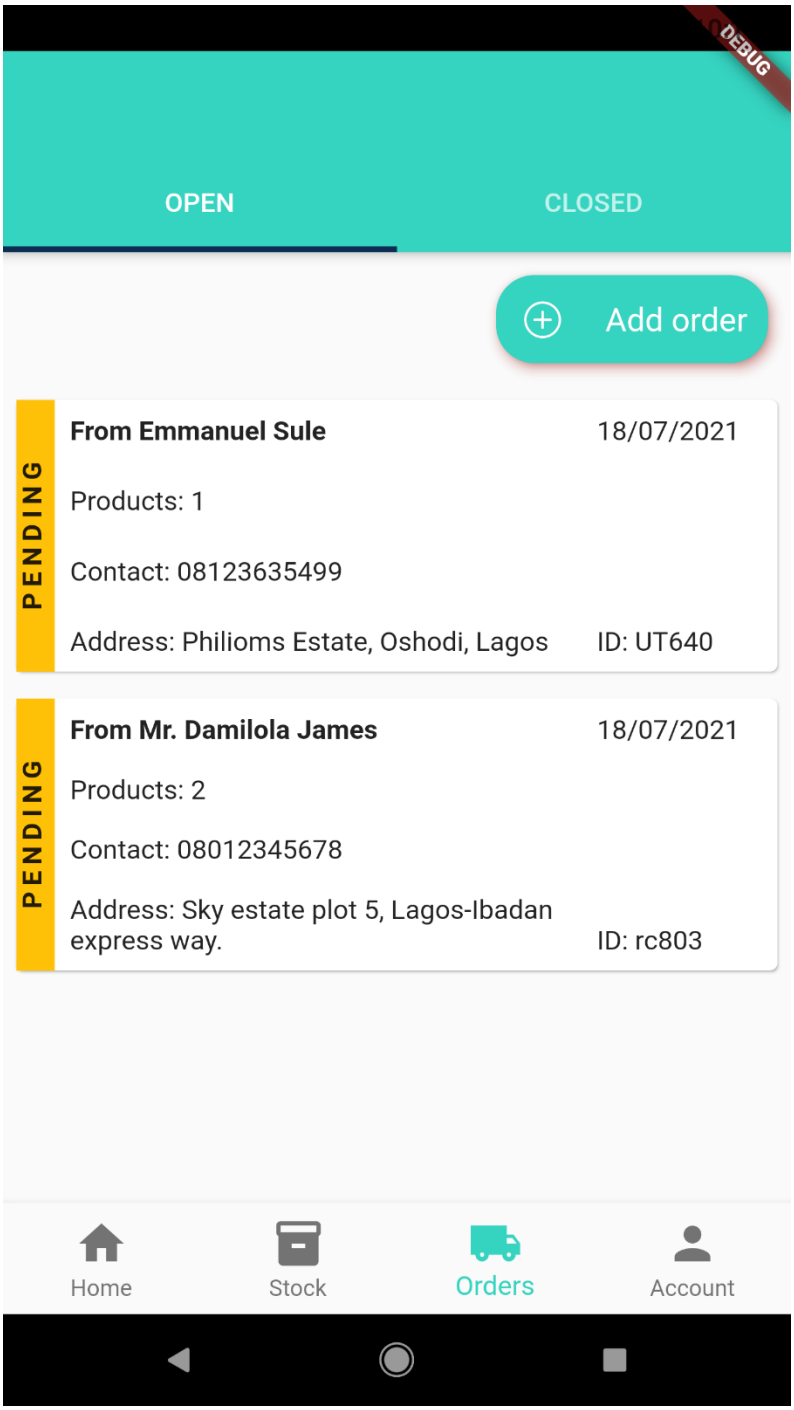


Figure 4.13: Farmers ordering menu showing all received orders with their status

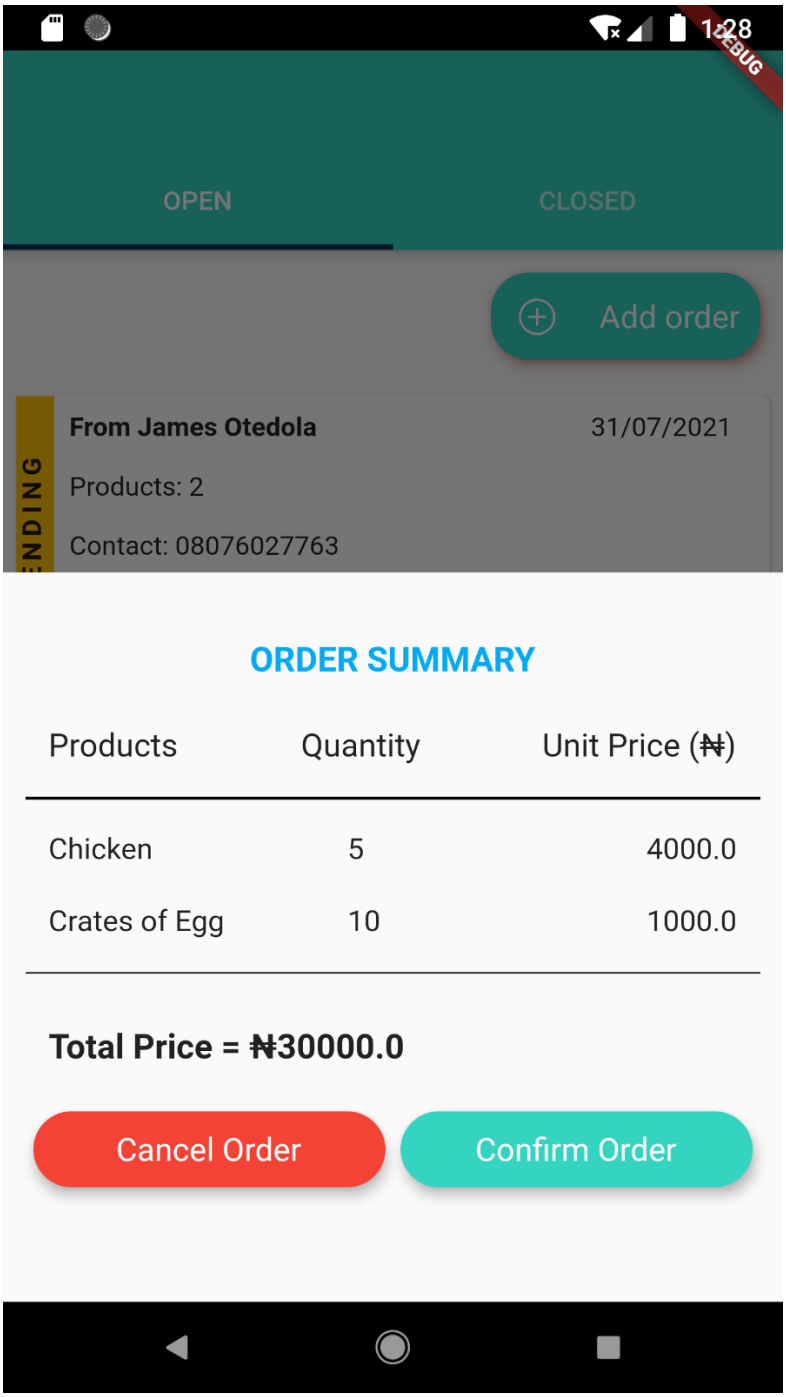


Figure 4.14: Order summary form for cancellation and confirmation of orders

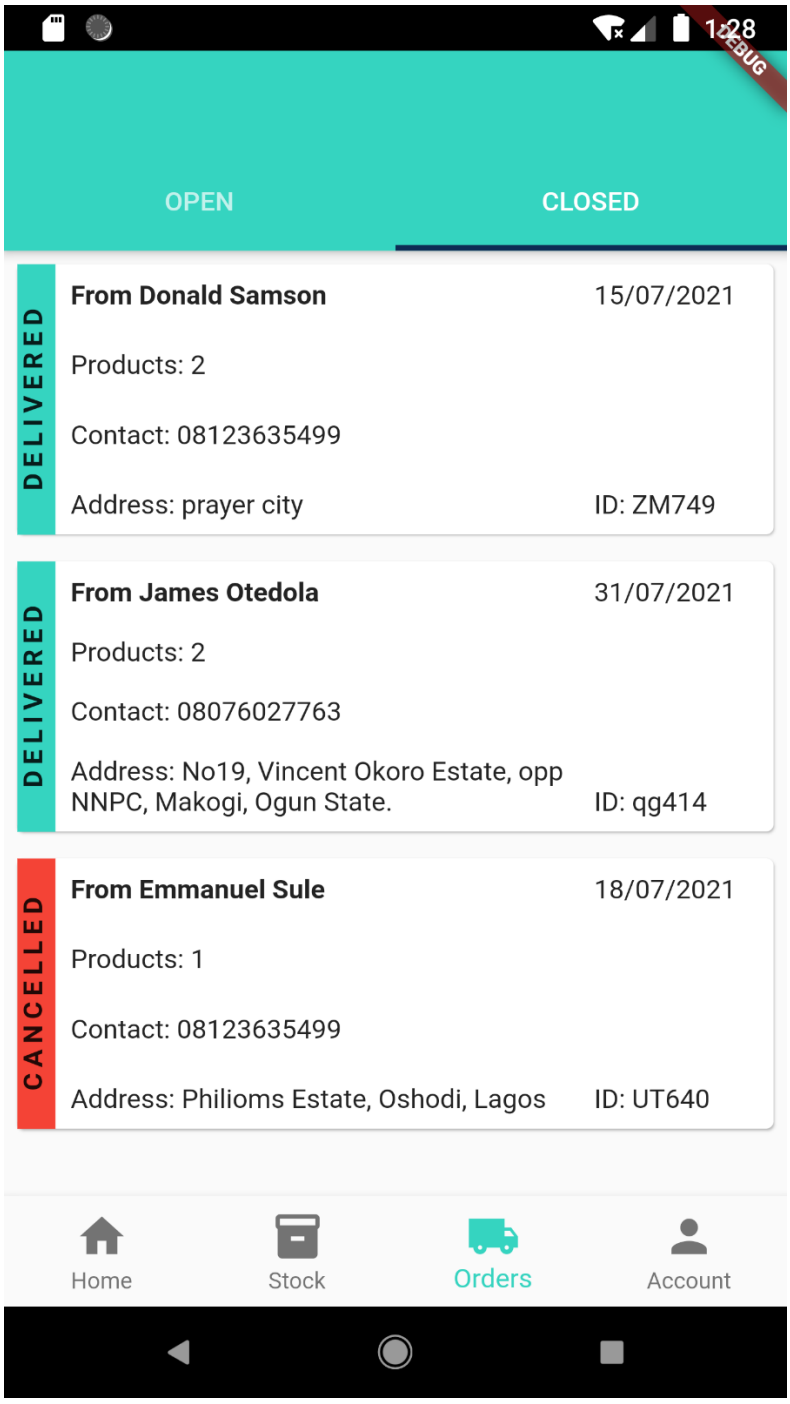


Figure 4.15: Confirmed and cancelled deliveries showing up in closed tab

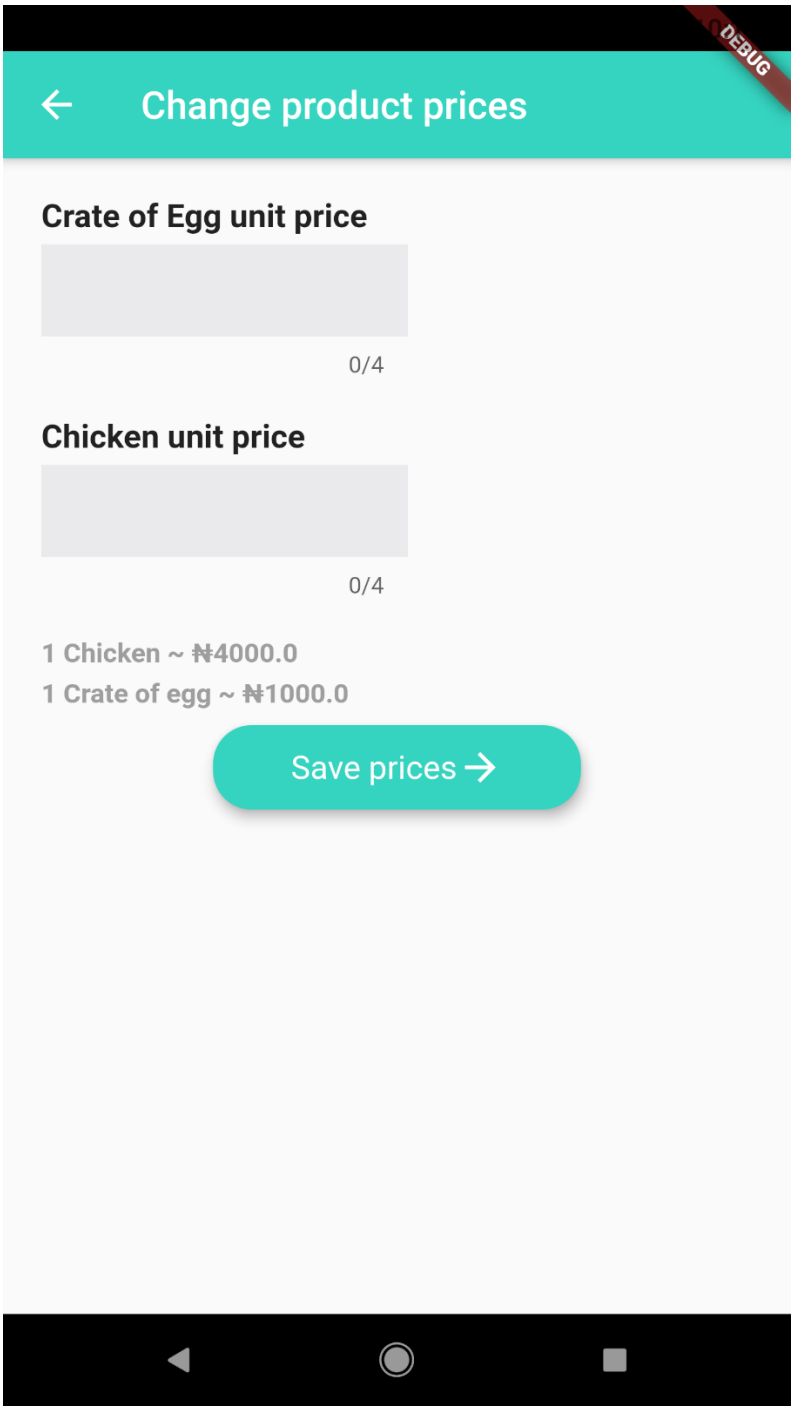


Figure 4.16: Price adjustment screen for farm goods

### 4.3 System Testing

As highlighted in chapter 3, two approaches will be used to test the implemented software i.e. unit testing and acceptance testing. Unit tests were written for each of the classes in the software. The below figure 4.17 shows the results of the unit tests that were run for a particular build of the application. All tests cases passed proving that the specific build had no bugs.

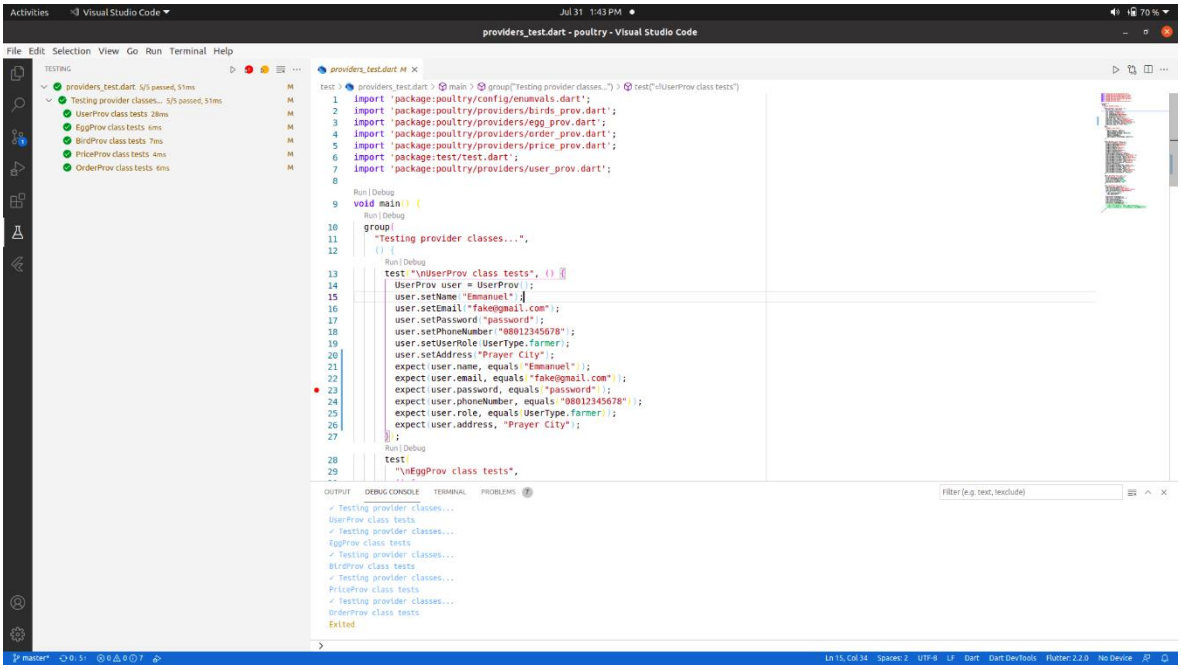


Figure 4.17: Results of unit tests ran for classes in the software

# CHAPTER FIVE
# SUMMARY AND CONCLUSION

## 5.0 Summary and Conclusion

The global effort to improve the efficiency of supply chains show the value that effective supply chain management have on the global economy. Multiple companies today are reaping the rewards of having effective supply chains such example is Amazon. The integration of technological based solutions with supply chain management will only further the cause of having sustainable supply chains that are capable of meeting up with user demands.

This project work presents a mobile first solution which can be applied to poultry supply chains to help automate key business activities which in turn help to enhance poultry supply chains. The implemented forward supply chain model adopted by the software focused on the farmer to distributor relationship. The software modules developed show that farm generated stock data is best stored on the cloud rather than relying on manual methods of storage. The ordering module also highlight the value that lye in having a centralized system that exposes farmers to a wide variety distributors willing to buy their goods.

From the above results, it is evident that technological solutions have a very important role to play in the goal of sustainability in supply chains. Efforts should always be made to integrate all the data generated in supply chains as they can prove useful in supporting decision making at crucial stages.

## 5.1 Limitations

Some of the limitations experienced while working on this project are as follows.

i. Lack of access to physical devices to test the implemented software on.
ii. Time constraints

## 5.2 Recommendations

For future work on the project, I intend to focus my efforts on studying possible methods on how both forward and reverse chain techniques can be best integrated into a poultry supply chain management software for automation. Despite poultry goods being perishable in nature, there is still tremendous value to be harnessed from the study of its reverse chain and better management of poultry by-products.

## REFERENCES


Altvater, A. (2020, April 8). *What is SDLC*. Retrieved from Stackify: https://stackify.com/what-is-sdlc/

Arumugham, A. J., C, K., & Parameswaran, R. (2017). Review of mathematical models for supply chain network designs. *International journal of innovative research in advanced engineering*, 12-21.

Awojulugbe, O. (2019, July 8). *CBN: Nigeria's poultry industry now worth N1.6trn*. Retrieved from The Cable.ng: www.thecable.ng/cbn-nigeria-poultry-industry-worth-n1-6trn

Balasaheb, P. B., Londhe, V., & Arudkar, A. (July 2020). A poultry farm management system. *International journal of research in engineering science and management*, 41-43.

Deepti, M., & Alok, M. (2007). Achieving Success in Supply Chain Management Software by Agility. (pp. 237-246). Berlin: Springer-Verlag Berlin Heidelberg.

Emmanuel, O., Hiver, B., Orsolya, M., & Ugo, P.-C. (2018). *Livestock and livelihood spotlight in Nigeria, cattle and poultry sectors.* Abuja.

Esnola-Gonzalez, I., Gómez-Omella, M., Ferreiro, S., Fernandez, I., I. L., & García, E. (2020, March 11). An IoT platform towards the enhancement of poultry production chains.

Fajemisin, A., & Ogunribido, T. (2018). Harnessing the values along the food supply chain of poultry production in Nigeria. *Nigerian journal of animal science*, 162-172.

flutter.dev. (2021). *Flutter architectural overview*. Retrieved from Flutter: https://flutter.dev/docs/resources/architectural-overview

Goud, K. S., & Sudharson, A. (2015). Internet based smart poultry farm. *Indian journal of science and technology*.

Haikun, Z., Tiemin, Z., Cheng, F., Jiayuan, Z., & Yang, X. Y. (2021). Design and Implementation of Poultry Farming Information Management System Based on Cloud Database. *Animals Precisions Poultry Farming*, 900-915.

Jose, F., & Andrew, F. (2005). Supply chain software implementations: getting it right. *Supply Chain Management: An International Journal* , 241-243.

Kenton, W. (2020, July 7). *Supply chain*. Retrieved from Investopedia: https://www.investopedia.com/terms/s/supplychain.asp

Lauren, X. L., & Swaminathan, J. M. (2015). Supply chain management. *International Encyclopedia of Social and Behavioral Sciences, 2nd edition, Vol 23*, 709-713.

Liverpool-Tasie, L. S., Omonona, B., Sanou, A., Ogunleye, W., Padilla, S., & Reardon, T. (2017). Growth and transformation of food systems in Africa: evidence from the poultry value chain in Nigeria. *Nigerian Journal of Agricultural Economis*, 1-15.

Lutkevich, B. (2020). *Supply chain explained*. Retrieved from Techtarget whatIs: https://whatis.techtarget.com/definition/supply-chain

Netherlands enterprise agency. (2020). *Poultry sector study Nigeria.* Prinses Beatrixlaan 2: Netherlands enterprise agency.

Rafaela, A.-L., & Carmen, M.-L. (2009). Supply Chain Management: Unheard of in the 1970s, core to today's company. *Business History*, 202-221.

Rajib, M. (2014). *Fundamentals of software engineering.* Delhi: Asoke K.Gosh, PHI learning private limited.

Shamsuddoha, M., Quaddus, M., & Klass, D. (2013). Poultry supply chain: a system approach.

Simchi-Levi, D., & Kaminski, P. (2008). *Designing and managing the supply chain: Concepts strategies and case studies.* Boston: McGraw-Hill.

Sommerville, I. (2011). *Software engineering.* Boston: Addison-Wesley.

Tutorialspoint. (2020). *SDLC-Overview*. Retrieved from Tutorialspoint web site: https://www.tutorialspoint.com/sdlc/sdlc_overview.htm