AGRO BUDDY

A Mini Project Report Submitted by

Prathvik Shervegar (4NM18CS119)
Moghe Sudheesh Rao (4NM17CS087)
Rohan Kamat (4NM18CS137)

UNDER THE GUIDANCE OF

Mr. Ramesha Shettigar
Assistant Professor
Department of Computer Science and Engineering

in partial fulfilment of the requirements for the award of the Degree of

Bachelor of Engineering in
Computer Science & Engineering

from

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum) (AICTE approved, NBA Accredited, ISO 9001:2008 Certified) NITTE -574 110, Udupi District, KARNATAKA.

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering
B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

"AGRO BUDDY"

is a bonafide work carried out by

Prathvik Shervegar(4NM18CS119)
Moghe Sudheesh Rao(4NM18CS094)
Rohan Kamat(4NM18CS137)

in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering prescribed by Visvesvaraya Technological University, Belagavi during the year 2020-2021.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report.

The Mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide　　　　　　　　　　Signature of HOD

ACKNOWLEDGEMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, we express our deep sense of gratitude and indebtedness to our guide **Mr. Ramesha Shettigar**, Assistant Professor, Department of Computer Science and Engineering, for his inspiring guidance, constant encouragement, support and suggestions for improvement during the course of our project.

We sincerely thank **Dr. Jyothi Shetty**, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

Our sincere thanks to our beloved principal, **Dr. Niranjan N. Chiplunkar** for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.
We also thank all those who have supported us throughout the entire duration of our project.

Finally, we thank the staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

Prathvik Shervegar(4nm18cs119)

Moghe Sudheesh Rao(4nm18cs094)

Rohan Kamat(4nm18cs137)

ABSTRACT

This application provides a new technique for both farmers and Buyers to communicate over a single social medium.

A Farmer can list and price his/her produce on our app to begin a new venture .The app provides very basic user interface for the farmers to work with. It also has many features with which a farmer can keep track of his daily affairs.

The buyer or the consumer on the other hand will get a dedicated user interface for his/her affairs. A commercial consumer can list the crops which he/she wants to buy and start a dialogue with the farmers. Two types of consumers can register themselves, commercial and non-commercial.

We at AGRO BUDDY aim at providing Freedom for farmers to sell their produce at the price they want to and also have a direct interaction with the consumers

Table of Contents:

CHAPTER 1

INTRODUCTION

1.1 Scope

Agro Buddy APP is a flexible, easy to use and securely designed to benefit the farmers and consumers. It is used to digitize the market of retail selling and wholesale all in one single android mobile phone and manage them efficiently.

1.2 Importance

Because of third party vendors, farmers never get to price their produce or get any profits out of it. Consumers on the other hand have to pay more price to the third party vendors. With Agro Buddy the whole ecosystem of third party vendors will be restricted from entering one way business between consumers and farmers.

1.3 Objective:

The objective of this project is a mobile application developed for managing the list of food grains or agricultural produces that a farmer wants to sell and also for the buyers to see what is available in the market and the price which the would have to pay.

CHAPTER 2

LITERATURE SURVEY

2.1 Technical Background:

In this app the information of Farmers and consumers are stored in the backend i.e., in Google Firebase Database. GUI of the application enables to access and modify data efficiently.

2.2 Existing system:

The existing system has UI for Farmers to communicate with consumers and vice versa. With functions like trade history and Income generated by a particular entity.

2.3 Proposed System:

The proposed system will try to make the app more efficient while also bringing new features with which the ease of doing business over the app will increase significantly , and the world of business will become a smaller place

CHAPTER 3

SYSTEM REQUIREMENT AND SPECIFICATION

3.1 Introduction

Requirements are during early stages of a system development as a specification of what should be implemented or as a constraint of some kind of on the system. They may be a user level facility description, a detailed specification of expected system behaviour, a general system property, a specific constraint on the system, and information on how to carry out some computation or a constraint on the development of the system. The end product of the requirement analysis phase is a requirement specification. The requirement specification is a reconstruction of the result of this analysis phase. Its purpose is to communicate this result to others. System requirements are more detailed descriptions of the user requirements. They may serve as the basis for a contract to the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point of system design. In principle, the system requirements should state what the system should do and not how it should be implemented. However, at the level of detail required to specify the system completely, it is virtually impossible to exclude all design information.

Natural language is often used to write system requirements specifications. Further problems with natural language can arise when it is used for more detailed specification:

1 Natural language understanding relies on the specification of the readers and writers using the same words for the same concept. This leads to misunderstandings because of the ambiguity of the natural language.

2 A natural language requirements specification is over-flexible. You can say the same thing in completely different ways. It is up to the reader to find out when requirements are same and when they are distinct.

3.2 Functional Requirements

The functional requirements are the statement of services the system should provide, how system reacts to particular inputs and how system should behave in particular situation. It describes the functionality that the system provides.

Our app requires:

I Active internet connection.

II A firebase console to store the data

3.3 User Requirements

All the Entities using this app require Active Internet

3.4 Software Requirements

1 Operating System: Windows 7/8/10 (32-bit or 64-bit)
2. Android SDK
3. Android Studio
4. Firebase

3.4.1 Android SDK

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android. The ADT bundle includes the essential Android SDK components and a version of the Eclipse IDE with built-in Android Developer Tools to streamline the Android app development. ADT bundle consists of following components for developing the application II. Eclipse ADT plugin.

- Android SDK Tools
- Android Platform-tools
- The latest Android platform
- The latest Android system image for the emulator

3.4.2 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on Jet Brains IntelliJ IDEA software and designed
Specifically, for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as the primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. The current stable version is 3.3, which was released in January 2019.

3.4.3 Firebase

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014.As of October 2018, the Firebase platform has 18 products, which are used by 1.5 million apps. Firebase provides a real-time database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. Firebase Storage provides secure file uploads and downloads for Firebase apps, regardless of network quality. The developer can use it to store images, audio, video, or other user-generated content. Firebase Storage is backed by Google Cloud Storage.

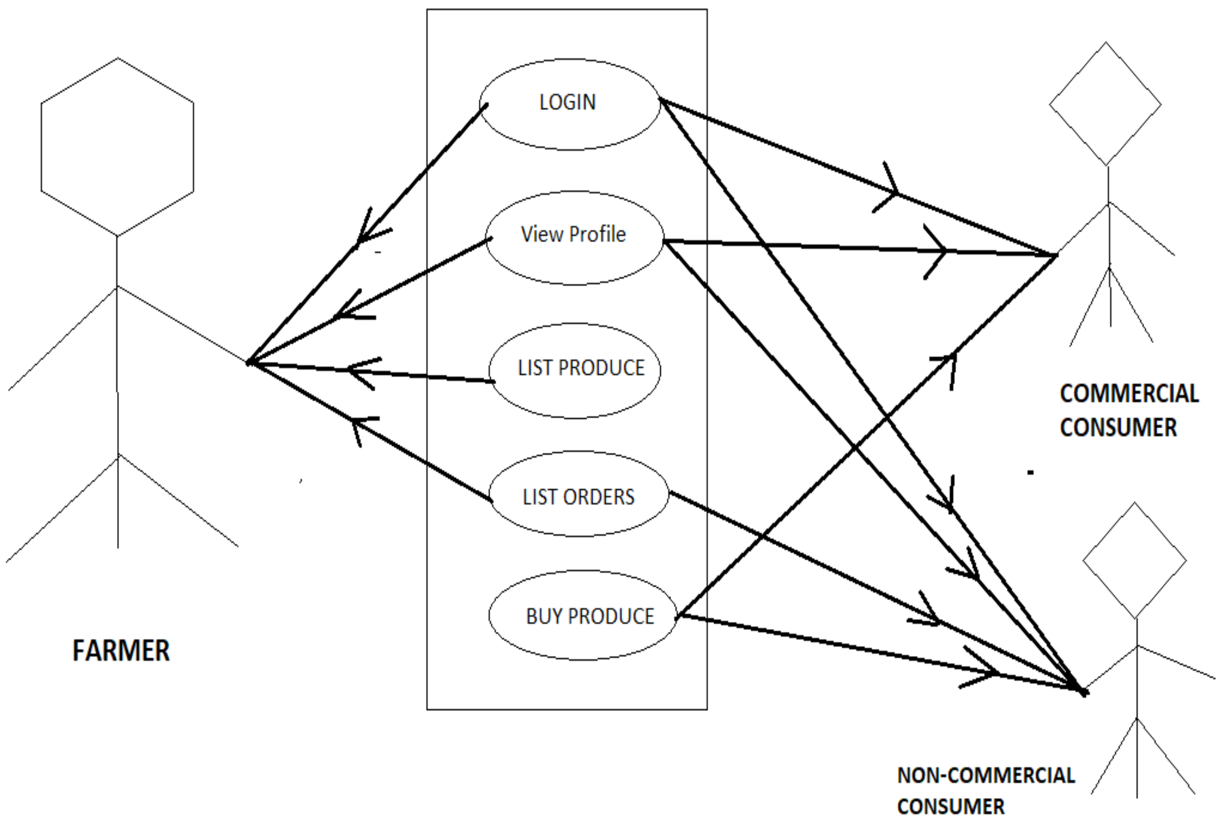
3.5 HARDWARE REQUIREMENTS

1. Minimum 4 GB RAM (8GB recommended).
2. 5GB free disk space
3. USB 2.0 or higher
4. Android Device

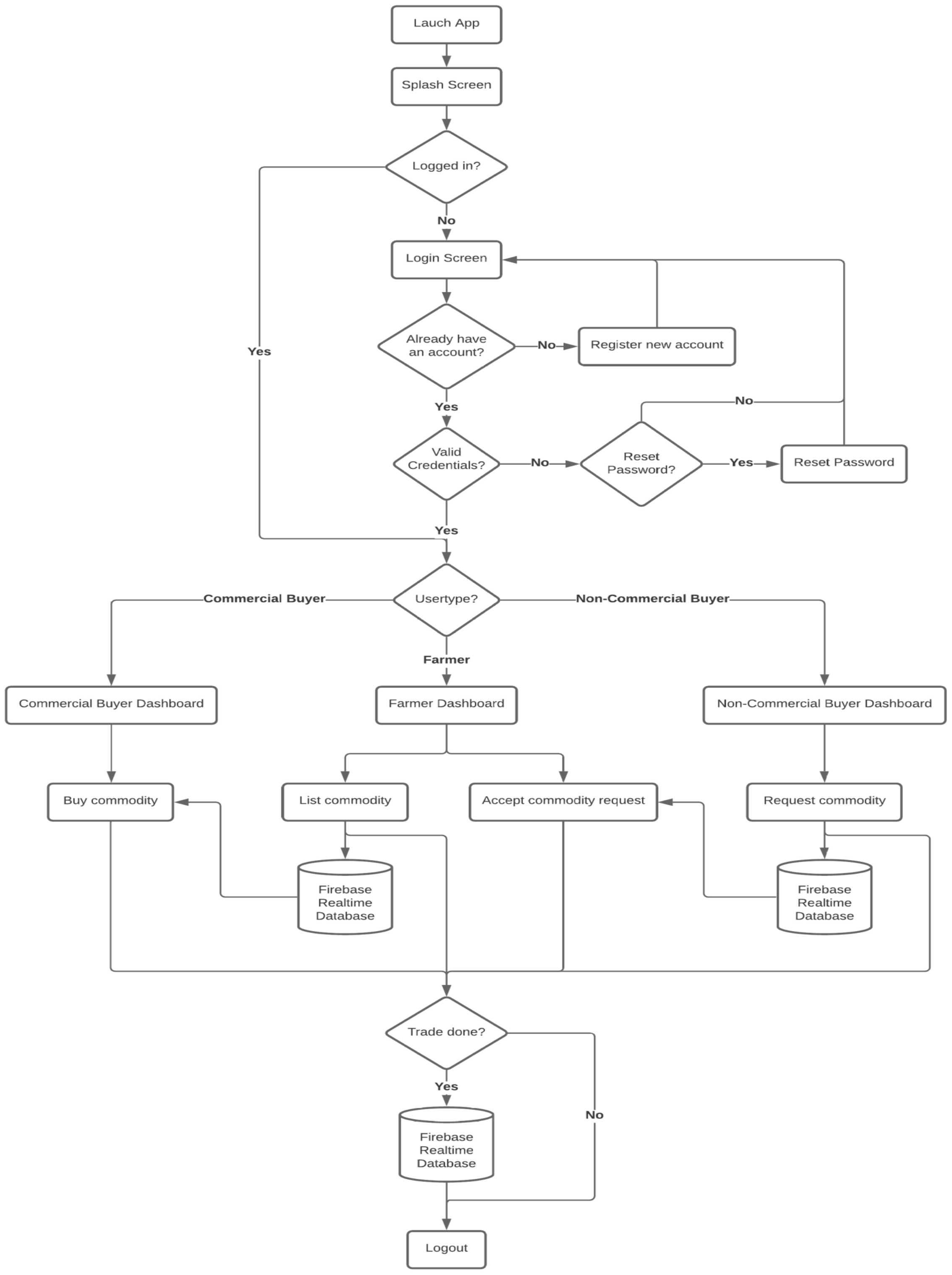
CHAPTER 4

SYSTEM DESIGN

4.1 USE CASE MODEL



4.2 Dataflow Diagram



FARMER:

- Adds the products to be sold and prices them.
- Can access the list of requests given by non-commercial consumers

COMMERCIAL CONSUMER:

- Can read the list of produce given by farmers
- Buy the products

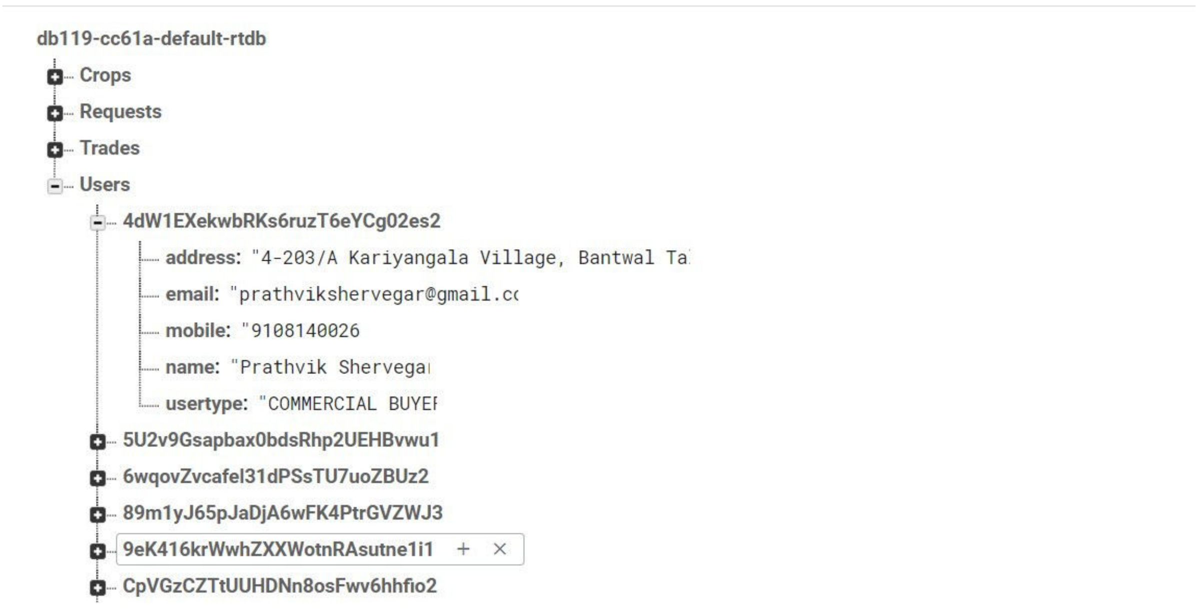
NON COMMERCIAL CONSUMER:

- Can request the products farmers usually produce as they buy in small-scale
- Requests need to be accepted by farmer to complete trade

4.3 Database Design

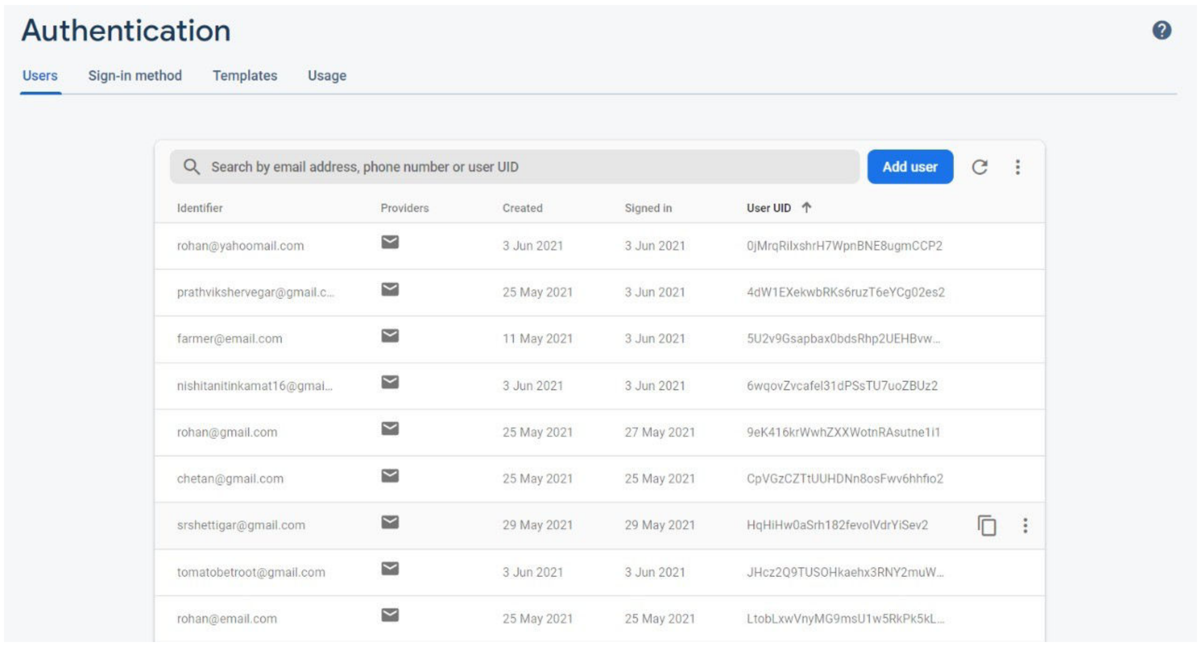
The database is designed using Google Firebase Console in which data is stored in popular data structure known as JSON tree (JavaScript Object Notation). Every time when the data transfer happens from client end, the information given to the UI is converted into JSON tree structure which is efficient and faster way to retrieve and store data.

A snapshot of the data in the firebase database



```
db119-cc61a-default-rtdb
 + Crops
 + Requests
 + Trades
 - Users
    - 4dW1EXekwbRKs6ruzT6eYCg02es2
         address: "4-203/A Kariyangala Village, Bantwal Ta
         email: "prathvikshervegar@gmail.cc
         mobile: "9108140026
         name: "Prathvik Shervegai
         usertype: "COMMERCIAL BUYEF
    + 5U2v9Gsapbax0bdsRhp2UEHBvwu1
    + 6wqovZvcafel31dPSsTU7uoZBUz2
    + 89m1yJ65pJaDjA6wFK4PtrGVZWJ3
    + 9eK416krWwhZXXWotnRAsutne1i1   +  ×
    + CpVGzCZTtUUHDNn8osFwv6hhfio2
```

A snapshot of the data authenticated



Authentication

Users Sign-in method Templates Usage

Search by email address, phone number or user UID

Add user

Identifier	Providers	Created	Signed in	User UID ↑
rohan@yahoomail.com	✉	3 Jun 2021	3 Jun 2021	0jMrqRilxshrH7WpnBNE8ugmCCP2
prathvikshervegar@gmail.c...	✉	25 May 2021	3 Jun 2021	4dW1EXekwbRKs6ruzT6eYCg02es2
farmer@email.com	✉	11 May 2021	3 Jun 2021	5U2v9Gsapbax0bdsRhp2UEHBvw...
nishitanitinkamat16@gmai...	✉	3 Jun 2021	3 Jun 2021	6wqovZvcafel31dPSsTU7uoZBUz2
rohan@gmail.com	✉	25 May 2021	27 May 2021	9eK416krWwhZXXWotnRAsutne1i1
chetan@gmail.com	✉	25 May 2021	25 May 2021	CpVGzCZTtUUHDNn8osFwv6hhfio2
srshettigar@gmail.com	✉	29 May 2021	29 May 2021	HqHiHw0aSrh182fevolVdrYiSev2
tomatobetroot@gmail.com	✉	3 Jun 2021	3 Jun 2021	JHcz2Q9TUSOHkaehx3RNY2muW...
rohan@email.com	✉	25 May 2021	25 May 2021	LtobLxwVnyMG9msU1w5RkPk5kL...

CHAPTER 5

IMPLEMENTATION

5.1 SOLUTION APPROACH/METHODOLOGY

We are here using xml and java for the front end and firebase for the backend as a server.

5.1.1 FIREBASE

Firebase is considered as web application platform. It helps query for inserting, updating, deleting or adding data to it. It is the backend of a system that is used as a database for storing data

Firebase real-time database feature is very easy to use. Once the Firebase and database dependency is added to the app, unstructured data can be added to database.

5.1.2 STORAGE

Image files can be stored in the app. The data stored is highly secured and is robust in nature, means it resumes from the last point if any network error occurs. The steps below are to be followed to use storage feature in Android application: added to the application, create instance of

5.1.3 FIREBASE AND ANDROID APP

An Android application has been developed for the demonstration of Firebase. In this app images along with strings are loaded to Firebase and retrieved from Firebase similar to Instagram. For the development of an Android app to demonstrate the use of Firebase, prototyping model has been followed.
Steps for connecting App to Firebase:

Step1: An account in the Firebase Login has to be created at https://www.firebase.com/login/ using the Google account.

Step2: Creating a new application on Firebase. Firebase creates a new application when one logs in for the first time. Also, at the bottom left corner, one can find an option to create a new application on the Firebase server. The app url has to be unique among all applications deployed on Firebase.

Step3: Next step is to add Firebase as a project dependency. Make changes to the following lines to the build.gradle file, which is located in the app's project folder, and not the root folder. After adding any dependency, one has to make sure to sync the application. If there is any build error complaint about duplicate files then one can choose to exclude those files by adding the packaging Options directive to the build.gradle file: android

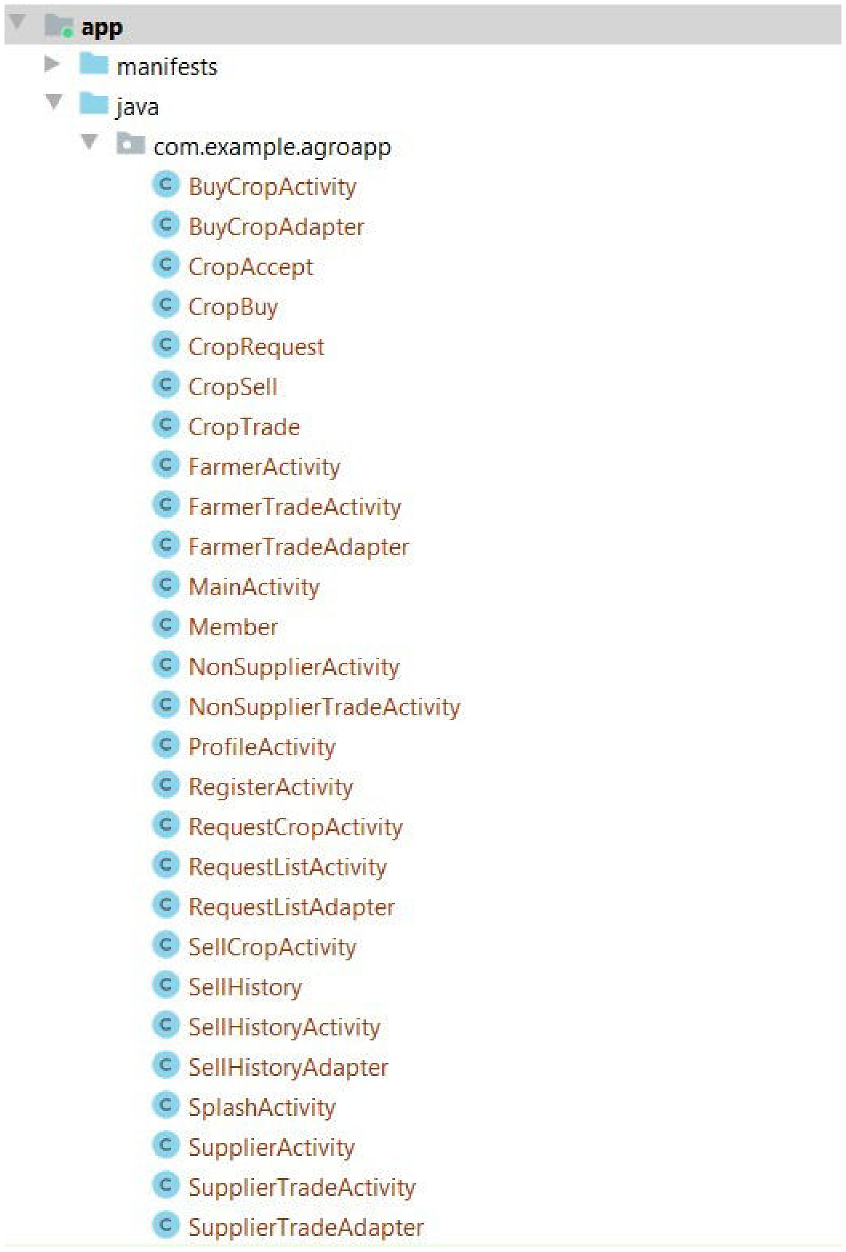
Step4: Next, add permissions to Android application, add network permission to the app, the same way it has been done for parse earlier. Now add the following line to the AndroidManifest.xml file:
<uses-permission android:name="android.permission.INTERNET" /
Firebase is a Backend-as-a-Service—BaaS—that started as an YC11 start up and grew up into a next-generation app-development platform on Google Cloud Platform.

5.1.4 Java

There are several ways to create apps for Android devices, but the recommended method for most developers is to write native apps using Java and the Android SDK. Java for Android apps is both similar and quite different from other types of Java applications.
If you have experience with Java (or a similar language) then you'll probably feel comfortable diving right into the code and learning how to use the Android SDK to make your app run. But if you're new to programming or object-oriented languages then you'll probably want to get familiar with the syntax of the Java language and how to accomplish basic programming tasks before learning how to use the Android SDK.

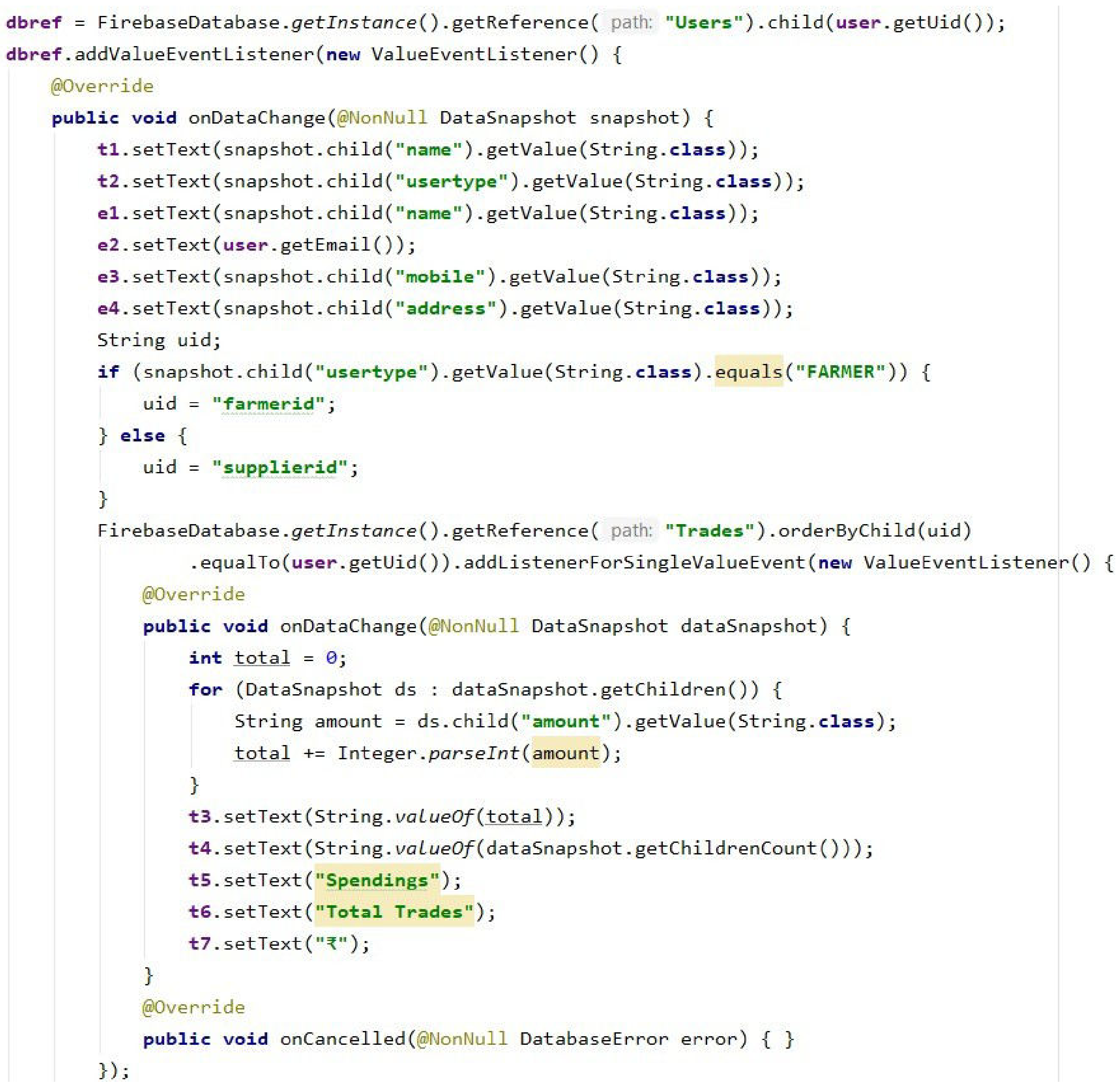
5.2 IMPLEMENTATION CODE

Overview of all the project files:



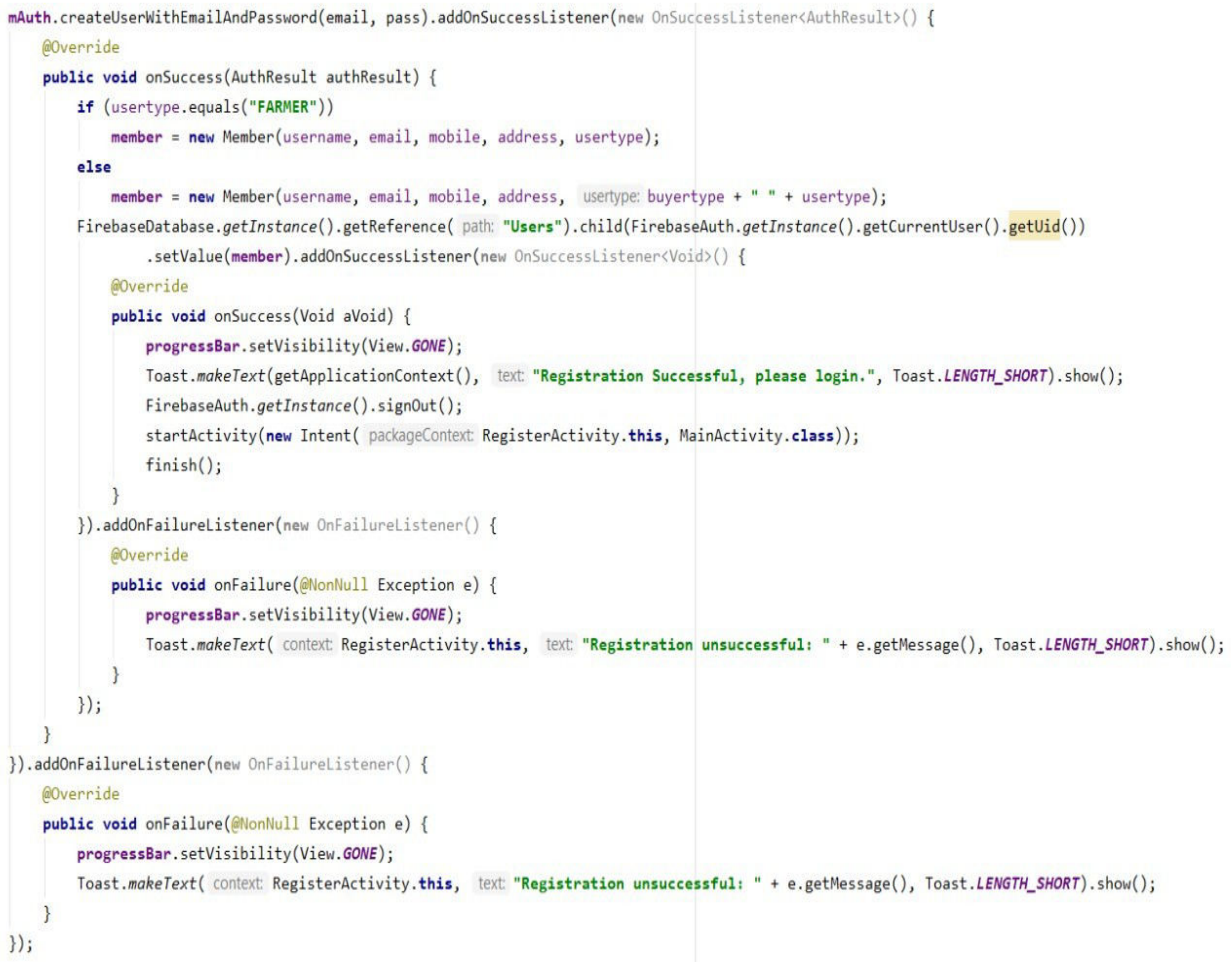
```
app
  manifests
  java
    com.example.agroapp
      BuyCropActivity
      BuyCropAdapter
      CropAccept
      CropBuy
      CropRequest
      CropSell
      CropTrade
      FarmerActivity
      FarmerTradeActivity
      FarmerTradeAdapter
      MainActivity
      Member
      NonSupplierActivity
      NonSupplierTradeActivity
      ProfileActivity
      RegisterActivity
      RequestCropActivity
      RequestListActivity
      RequestListAdapter
      SellCropActivity
      SellHistory
      SellHistoryActivity
      SellHistoryAdapter
      SplashActivity
      SupplierActivity
      SupplierTradeActivity
      SupplierTradeAdapter
```

Profile Activity:



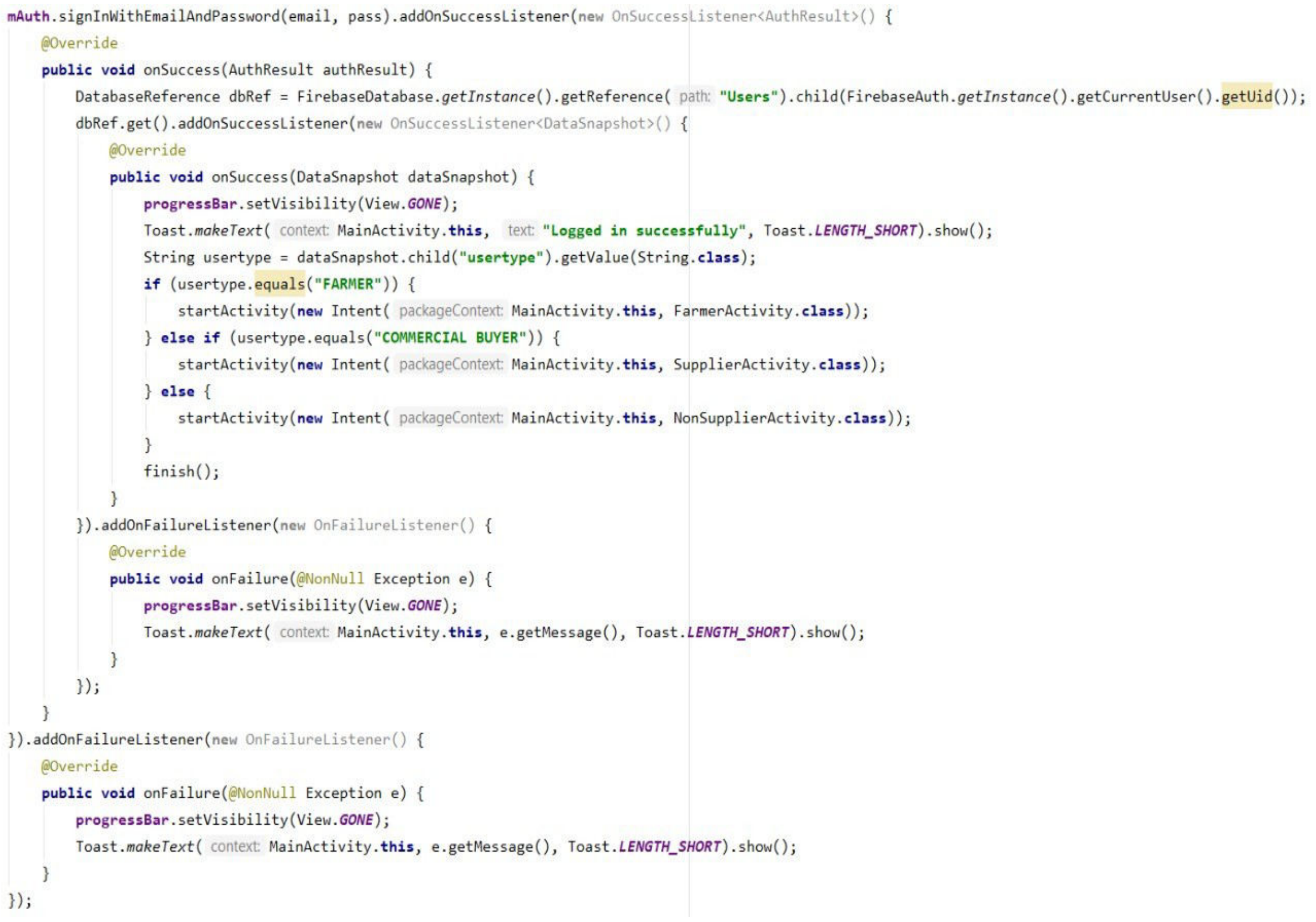
```
dbref = FirebaseDatabase.getInstance().getReference( path: "Users").child(user.getUid());
dbref.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {
        t1.setText(snapshot.child("name").getValue(String.class));
        t2.setText(snapshot.child("usertype").getValue(String.class));
        e1.setText(snapshot.child("name").getValue(String.class));
        e2.setText(user.getEmail());
        e3.setText(snapshot.child("mobile").getValue(String.class));
        e4.setText(snapshot.child("address").getValue(String.class));
        String uid;
        if (snapshot.child("usertype").getValue(String.class).equals("FARMER")) {
            uid = "farmerid";
        } else {
            uid = "supplierid";
        }
        FirebaseDatabase.getInstance().getReference( path: "Trades").orderByChild(uid)
                .equalTo(user.getUid()).addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                int total = 0;
                for (DataSnapshot ds : dataSnapshot.getChildren()) {
                    String amount = ds.child("amount").getValue(String.class);
                    total += Integer.parseInt(amount);
                }
                t3.setText(String.valueOf(total));
                t4.setText(String.valueOf(dataSnapshot.getChildrenCount()));
                t5.setText("Spendings");
                t6.setText("Total Trades");
                t7.setText("₹");
            }
            @Override
            public void onCancelled(@NonNull DatabaseError error) { }
        });
```

Registration Page:



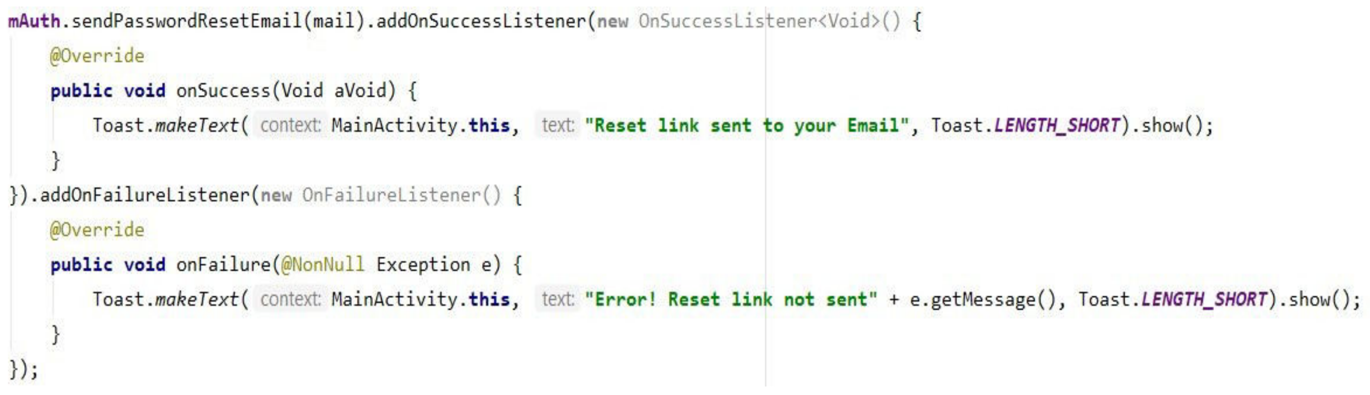
```
mAuth.createUserWithEmailAndPassword(email, pass).addOnSuccessListener(new OnSuccessListener<AuthResult>() {
    @Override
    public void onSuccess(AuthResult authResult) {
        if (usertype.equals("FARMER"))
            member = new Member(username, email, mobile, address, usertype);
        else
            member = new Member(username, email, mobile, address,  usertype: buyertype + " " + usertype);
        FirebaseDatabase.getInstance().getReference( path: "Users").child(FirebaseAuth.getInstance().getCurrentUser().getUid())
                .setValue(member).addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {
                progressBar.setVisibility(View.GONE);
                Toast.makeText(getApplicationContext(),  text: "Registration Successful, please login.", Toast.LENGTH_SHORT).show();
                FirebaseAuth.getInstance().signOut();
                startActivity(new Intent( packageContext: RegisterActivity.this, MainActivity.class));
                finish();
            }
        }).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                progressBar.setVisibility(View.GONE);
                Toast.makeText( context: RegisterActivity.this,  text: "Registration unsuccessful: " + e.getMessage(), Toast.LENGTH_SHORT).show();
            }
        });
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        progressBar.setVisibility(View.GONE);
        Toast.makeText( context: RegisterActivity.this,  text: "Registration unsuccessful: " + e.getMessage(), Toast.LENGTH_SHORT).show();
    }
});
```

Login Page:



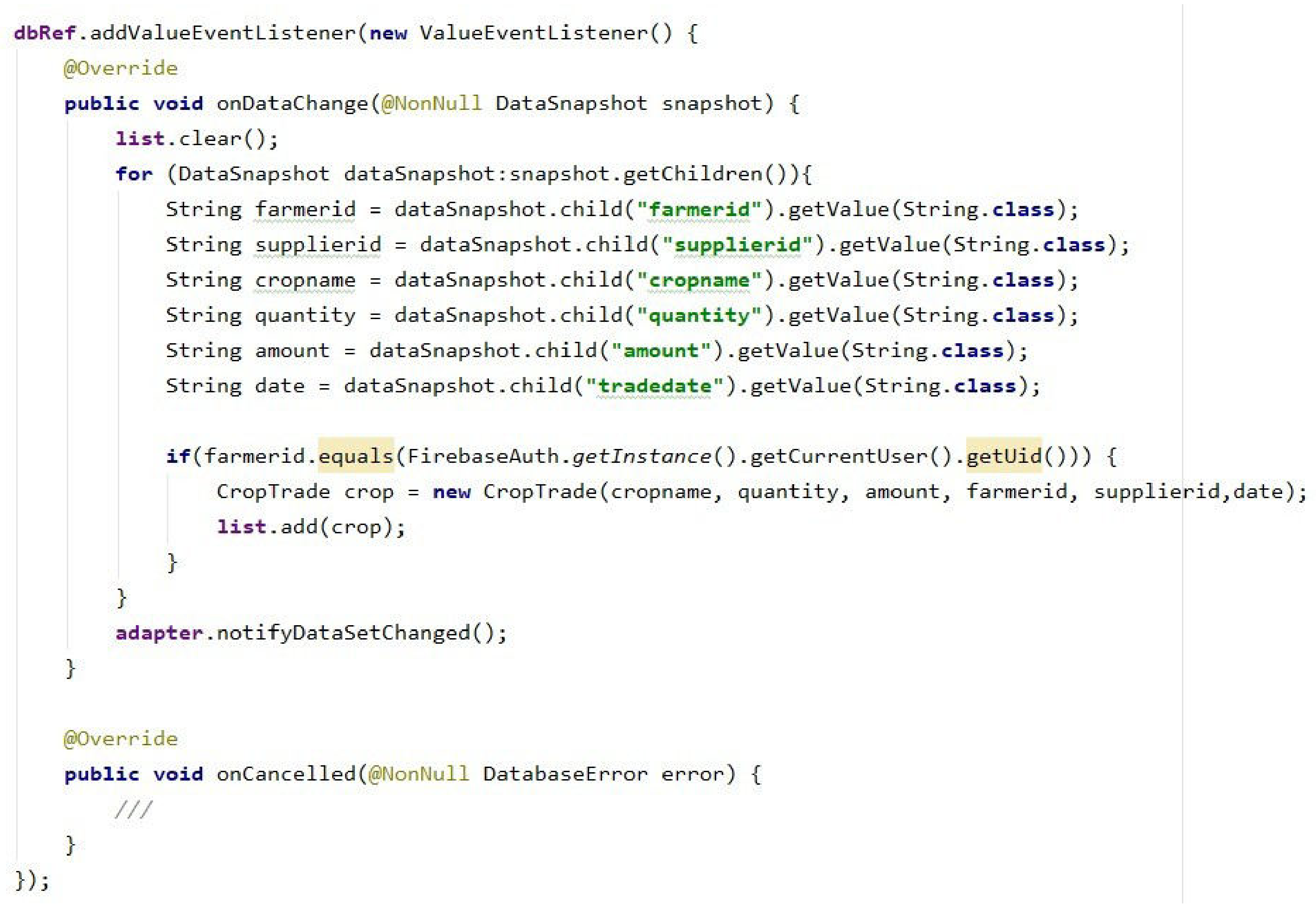
```
mAuth.signInWithEmailAndPassword(email, pass).addOnSuccessListener(new OnSuccessListener<AuthResult>() {
    @Override
    public void onSuccess(AuthResult authResult) {
        DatabaseReference dbRef = FirebaseDatabase.getInstance().getReference( path: "Users").child(FirebaseAuth.getInstance().getCurrentUser().getUid());
        dbRef.get().addOnSuccessListener(new OnSuccessListener<DataSnapshot>() {
            @Override
            public void onSuccess(DataSnapshot dataSnapshot) {
                progressBar.setVisibility(View.GONE);
                Toast.makeText( context: MainActivity.this,  text: "Logged in successfully", Toast.LENGTH_SHORT).show();
                String usertype = dataSnapshot.child("usertype").getValue(String.class);
                if (usertype.equals("FARMER")) {
                    startActivity(new Intent( packageContext: MainActivity.this, FarmerActivity.class));
                } else if (usertype.equals("COMMERCIAL BUYER")) {
                    startActivity(new Intent( packageContext: MainActivity.this, SupplierActivity.class));
                } else {
                    startActivity(new Intent( packageContext: MainActivity.this, NonSupplierActivity.class));
                }
                finish();
            }
        }).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                progressBar.setVisibility(View.GONE);
                Toast.makeText( context: MainActivity.this, e.getMessage(), Toast.LENGTH_SHORT).show();
            }
        });
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        progressBar.setVisibility(View.GONE);
        Toast.makeText( context: MainActivity.this, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
});
```

Password Forgot:



```
mAuth.sendPasswordResetEmail(mail).addOnSuccessListener(new OnSuccessListener<Void>() {
    @Override
    public void onSuccess(Void aVoid) {
        Toast.makeText( context: MainActivity.this,  text: "Reset link sent to your Email", Toast.LENGTH_SHORT).show();
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        Toast.makeText( context: MainActivity.this,  text: "Error! Reset link not sent" + e.getMessage(), Toast.LENGTH_SHORT).show();
    }
});
```

Trade History Page:



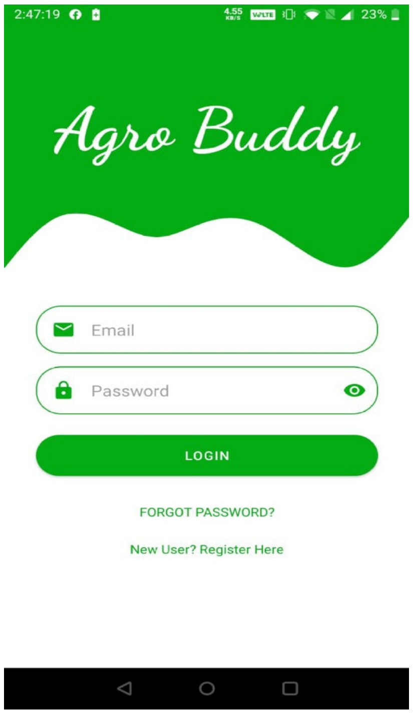
```java
dbRef.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {
        list.clear();
        for (DataSnapshot dataSnapshot:snapshot.getChildren()){
            String farmerid = dataSnapshot.child("farmerid").getValue(String.class);
            String supplierid = dataSnapshot.child("supplierid").getValue(String.class);
            String cropname = dataSnapshot.child("cropname").getValue(String.class);
            String quantity = dataSnapshot.child("quantity").getValue(String.class);
            String amount = dataSnapshot.child("amount").getValue(String.class);
            String date = dataSnapshot.child("tradedate").getValue(String.class);

            if(farmerid.equals(FirebaseAuth.getInstance().getCurrentUser().getUid())) {
                CropTrade crop = new CropTrade(cropname, quantity, amount, farmerid, supplierid,date);
                list.add(crop);
            }
        }
        adapter.notifyDataSetChanged();
    }

    @Override
    public void onCancelled(@NonNull DatabaseError error) {
        ///
    }
});
```

CHAPTER 6

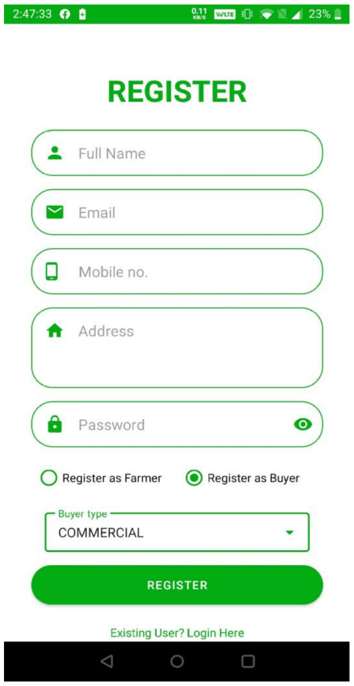
SCREENSHOTS

Introduction:

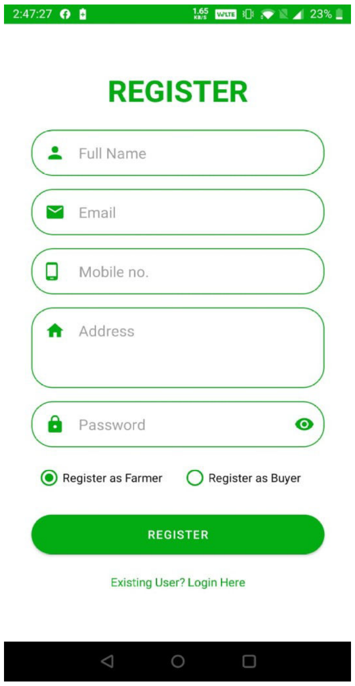
Login Page:



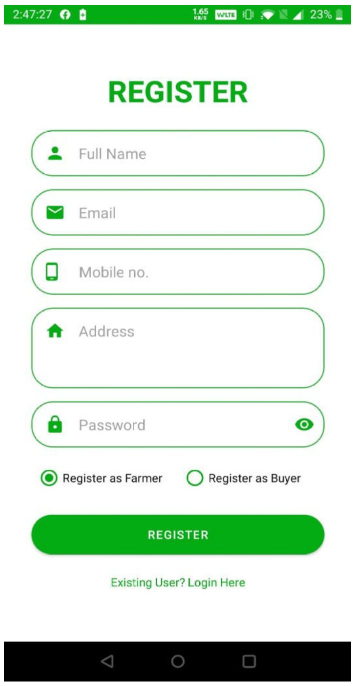
Sign up Activity:



Non-Commercial consumer

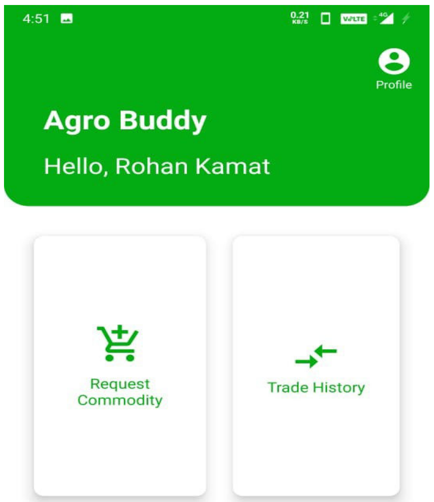


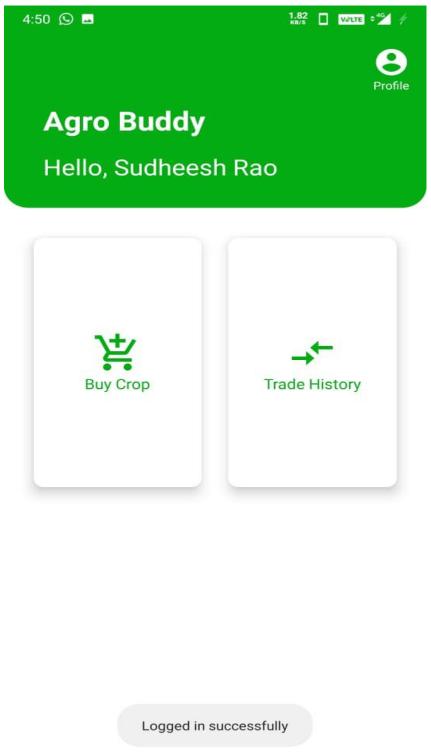
Commercial Consumer

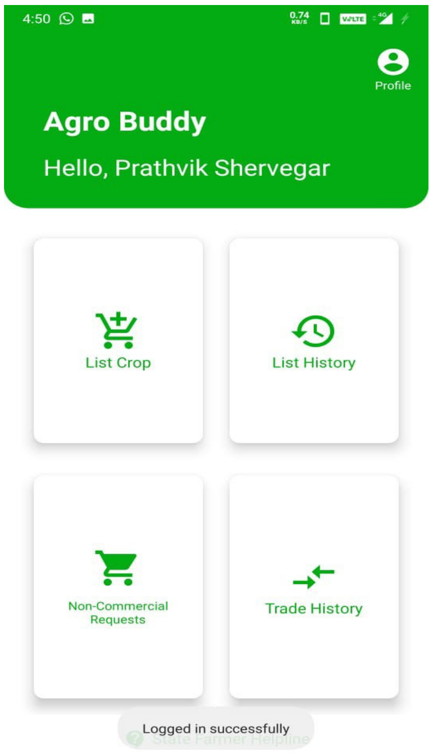


Farmer

Main Page:

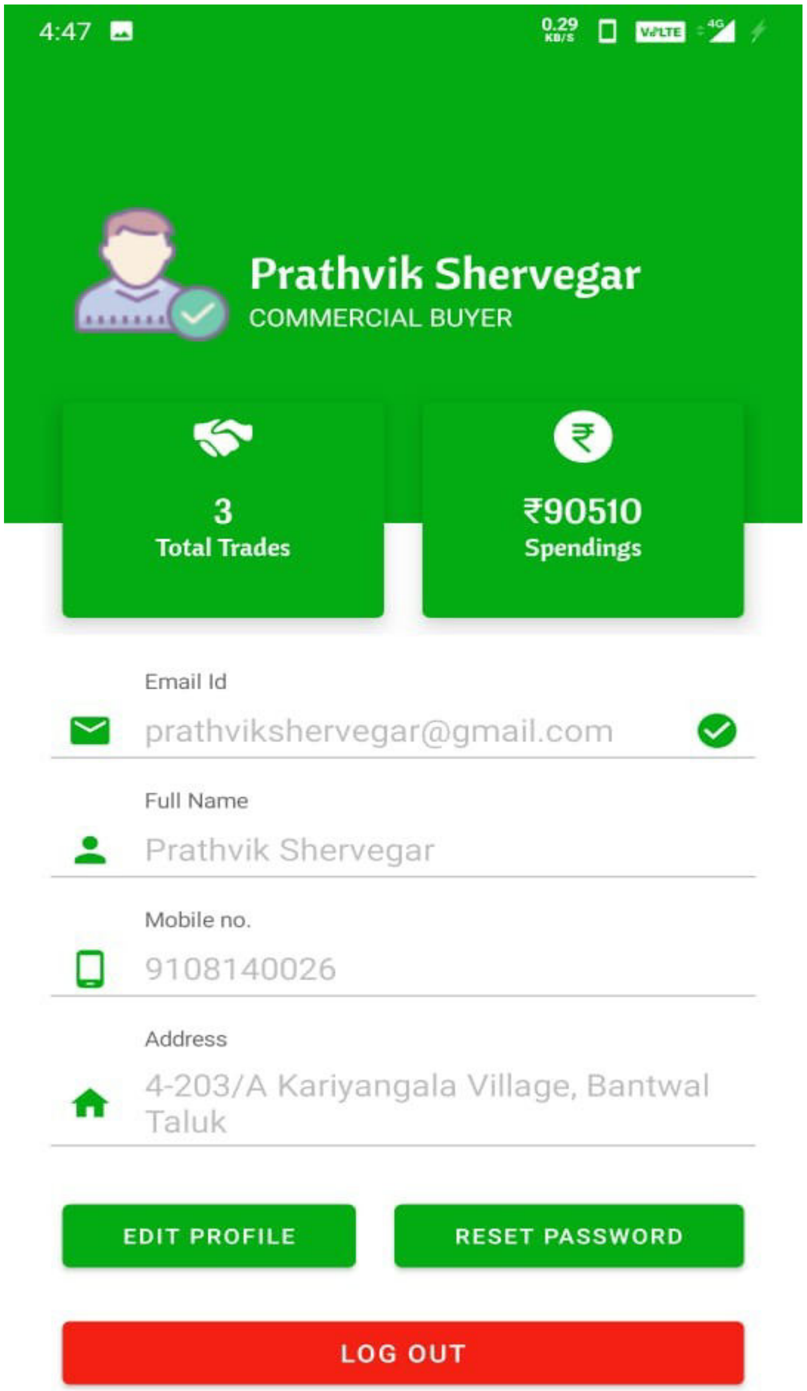






Non-Commercial Commercial Farmer

Profile Activity:


4:47

Prathvik Shervegar
COMMERCIAL BUYER

3
Total Trades

₹90510
Spendings

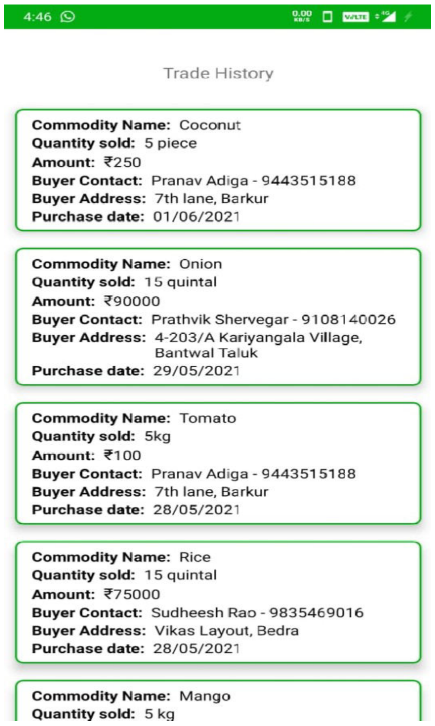
Email Id
prathvikshervegar@gmail.com

Full Name
Prathvik Shervegar

Mobile no.
9108140026

Address
4-203/A Kariyangala Village, Bantwal Taluk

Trade History :



4:46

Trade History

Commodity Name: Coconut
Quantity sold: 5 piece
Amount: ₹250
Buyer Contact: Pranav Adiga - 9443515188
Buyer Address: 7th lane, Barkur
Purchase date: 01/06/2021

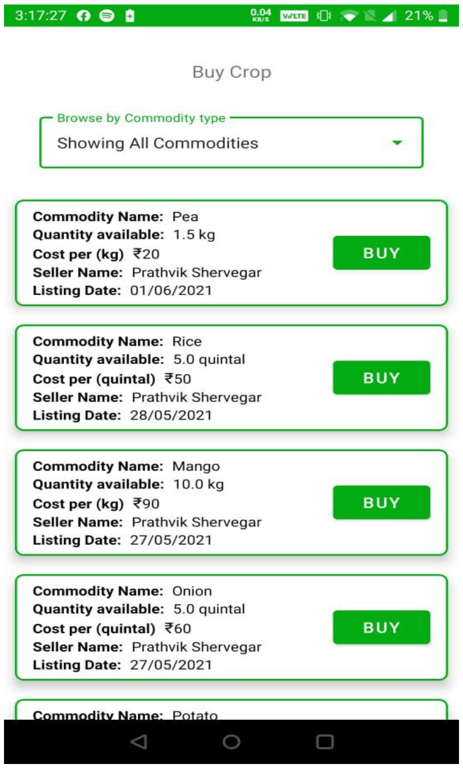
Commodity Name: Onion
Quantity sold: 15 quintal
Amount: ₹90000
Buyer Contact: Prathvik Shervegar - 9108140026
Buyer Address: 4-203/A Kariyangala Village, Bantwal Taluk
Purchase date: 29/05/2021

Commodity Name: Tomato
Quantity sold: 5kg
Amount: ₹100
Buyer Contact: Pranav Adiga - 9443515188
Buyer Address: 7th lane, Barkur
Purchase date: 28/05/2021

Commodity Name: Rice
Quantity sold: 15 quintal
Amount: ₹75000
Buyer Contact: Sudheesh Rao - 9835469016
Buyer Address: Vikas Layout, Bedra
Purchase date: 28/05/2021

Commodity Name: Mango
Quantity sold: 5 kg

Items Available:



3:17:27 21%

Buy Crop

Browse by Commodity type
Showing All Commodities

Commodity Name: Pea
Quantity available: 1.5 kg
Cost per (kg) ₹20
Seller Name: Prathvik Shervegar
Listing Date: 01/06/2021
BUY

Commodity Name: Rice
Quantity available: 5.0 quintal
Cost per (quintal) ₹50
Seller Name: Prathvik Shervegar
Listing Date: 28/05/2021
BUY

Commodity Name: Mango
Quantity available: 10.0 kg
Cost per (kg) ₹90
Seller Name: Prathvik Shervegar
Listing Date: 27/05/2021
BUY

Commodity Name: Onion
Quantity available: 5.0 quintal
Cost per (quintal) ₹60
Seller Name: Prathvik Shervegar
Listing Date: 27/05/2021
BUY

Commodity Name: Potato

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Results/Conclusion:

- Agro Buddy app is a Commercial app used for Buying and Selling of goods
- Farmers can list their produce. Commercial buyers can buy in bulk and normal consumers can place their daily orders
- As it uses android platform it supports all Android devices.

7.2 Future Works

In the future, we may extend this project by Introducing it in IOS devices

- Providing this service for multiple Operating System users
- More options for authentication.

8 References

- "Overview Guides Reference Samples Libraries Support Go to console" Documentation Firebase, https://firebase.google.com/docs/
- Stack Overflow, https://stackoverflow.com
- YouTube, https://www.youtube.com