MANIPAL
ACADEMY of HIGHER EDUCATION
(Institution of Eminence Deemed to be University)

FARMERA

SUBMITTED BY:	REGISTRATION NO:	ROLL NO:
Devika Vinayrajan	180911102	20
Praatibh Surana	180911148	22
Abhishek Amonkar	180911186	27

ABSTRACT

The agricultural industry is an essential part of survival and farmers are the drivers of this trade. With its allied sectors, it is the largest source of livelihoods in India. Seventy percent of our rural households still depend primarily on agriculture for their livelihood, with eighty two percent of farmers being small and marginal.

In 2017-2018 alone, total food grain production was estimated at **275 MT [1]**. However, they are not always provided with the best of facilities [2]. There is a severe shortage of resources owing to the large income inequality in India. A wide gap between the availability and need for information restricts farmers from achieving the most optimal production capability. Several factors such as weather conditions, natural disasters and terrain affect the quality of crops as well.

We propose an application, **Farmera**, that will act as a bridge between farmers and a centralized repository/organization that will help provide some respite and easier solutions.

ABBREVIATIONS

1. MT: Mega Tonne
2. UI: User Interface
3. UPI: Unified Payments Interface
4. JVM: Java Virtual Machine
5. API: Application Programming Interface

INTRODUCTION

Farmera is an app built on Android Studio using Java, and designed using Figma. The app was made with the objective of helping farmers by bridging the gap between authorities and farmers.

The app features an easy-to-use UI that will assist farmers in making better decisions with respect to crop selection based on past data that they will have access to via simple searches. This ensures that crops they select will be best-suited to the area they choose and will therefore flourish.

It also provides a list of pesticides, the category they belong to and the amount of consumption in MT during recent years.
The weather option is a dynamic feature, dedicated to displaying the current weather at the location chosen.
An additional payment interface has been added to assist the user in making monetary transactions through any UPI-based apps on their phone.
The security features implemented involve using biometric/pin validation in order to access the app.

LANGUAGE DESCRIPTION

The app was written in Java 8 and built on the Android Studio platform as instructed. Android Studio as a platform is extremely robust through its flexible Gradle-based build system, feature rich emulator, Git integration and extensive testing tools. [3]

Features of Java [4]

1. Object-Oriented
2. Portable
3. Platform independent
4. Secured
5. Robust
6. Interpreted
7. High Performance
8. Multithreaded
9. Dynamic

Advantages of Java [5]

1. Being object-oriented allows programmers to create modular reusable code.
2. Through its JVM, it compiles code into a platform independent byte code, which can then be interpreted on different kinds of machines, making it architecture neutral.
3. Java has fewer identified vulnerabilities due to its popularity leading to more bug hunters. [6]
4. Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU.

DATABASE DESCRIPTION

We have attempted to use data that is relevant while still keeping it concise. The most recent data available on crops [7] and pesticides [8] was used.

The pesticides database highlights the usage of certain pesticides over the years 2017, 2018 and 2019, and their chemical group.

The crops database is a little more descriptive and helps understand the area to produce comparison for crops based on season depending on the entered location. The data for crops is displayed based on the location entered which can be either the user's state or district.

Both databases contain information in MT units.

PROJECT DESCRIPTION

Problem Statement

1. While factors such as soil terrain and natural disasters cannot be predicted, how can we collate data collected over the years into a single platform?
2. How can we feature instantaneous and dynamic weather data?
3. How can we add security to the app so that the payment feature is not compromised?
4. How can we implement partial functionality in the absence of a stable internet connection, in order to make it more accessible?
5. What other information can assist in the process of growing and maintaining crops?
6. How can we make the UI understandable and easy-to-use?
7. What updates can be rolled out to widen the scope of the app in the future?

Objective

Implement an application that assists farmers in making various agricultural decisions based on factors relevant to their circumstances, such as location. This will help boost yield rates and help the farmers be best prepared for the future. The users of the app can access data such as crops suited to their area, pesticides, current weather conditions, make payments and change their location, if needed.

PROJECT DESCRIPTION

Proposed Methodology

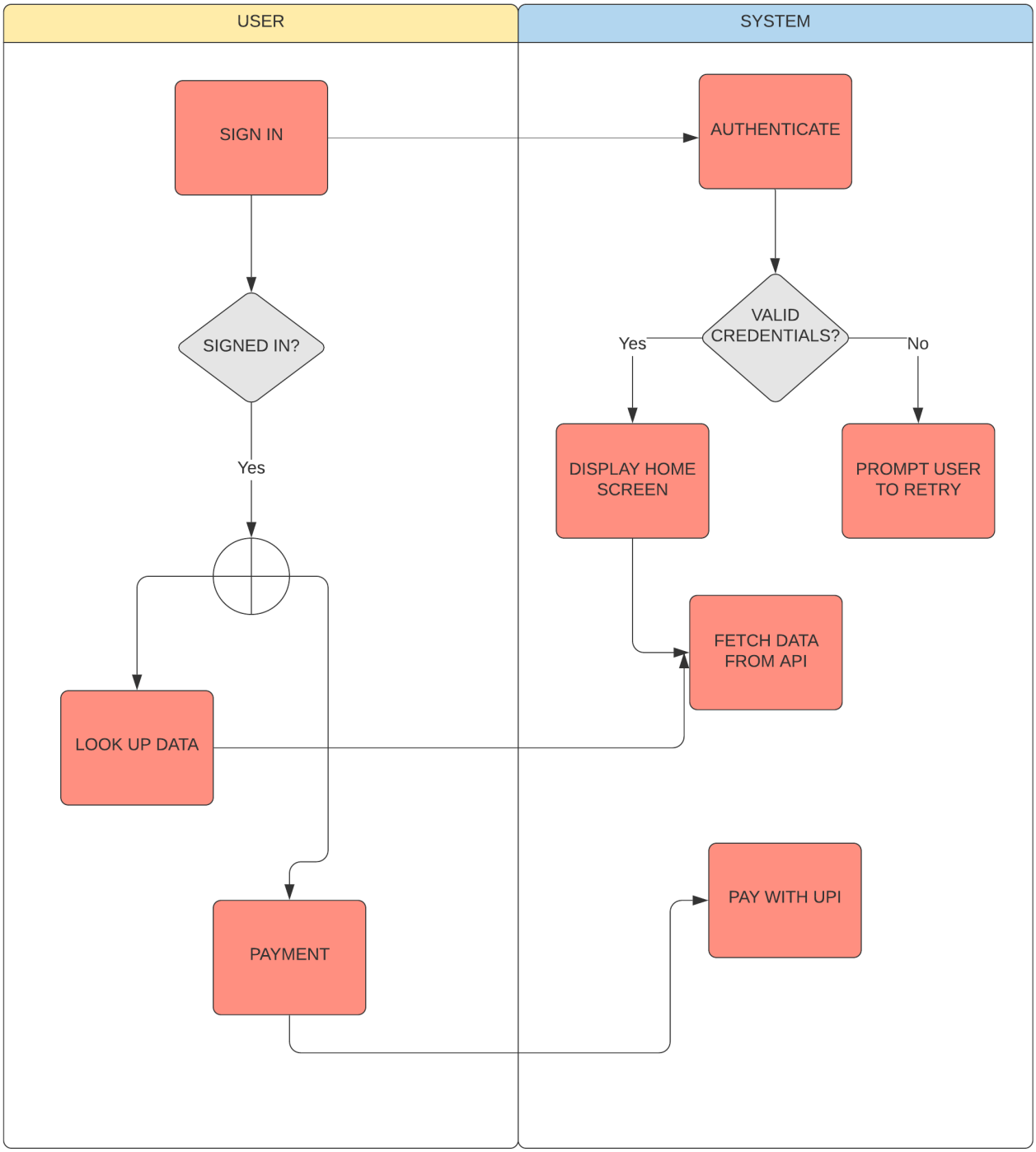
For authentication, the app makes use of the fingerprint sensor to validate the user's biometric. If a device has no fingerprint sensor, the app makes use of the device PIN and the user is prompted to enter it after which they can access the app.
Once the authentication is complete, the user is taken to the home page where they are required to provide their location, preferably city/district.
After this, as seen from the UI screenshots, the user can press the “Find Data” button which takes them to the menu page offering various functionalities as given below.

- **Weather:** On selecting this option, the weather for the entered location is displayed. This weather is real-time and is fetched from an API. It displays temperature, humidity, wind speed, etc.
- **Crops:** On selecting this option, the app pulls data from a repository and then filters out tuples depending on the entered location. It displays crops, the state, district and produce in MT.
- **Pesticides:** On selecting this option, data is pulled from a repository containing pesticide data. The units of usage of these pesticides is given in MT.
- **Help:** On selecting this option, the user is provided with a brief description on the usage of the app as well as various contact information in case of any discrepancies.
- **Make Payment:** This option enables the user to make UPI payments from any application on the device that allows it. As of now, the payment API does not have actual transactional functionality as this app is not in production.
- **Reset Location:** Lastly, the user can change their location by selecting this option.

These features are further discussed in detail in the ‘Implementation’ section of this report. We have also taken into account a situation wherein the user may not have access to the internet. This is handled via accessing the repository loaded on the app itself. This too, will be discussed in the implementation section.

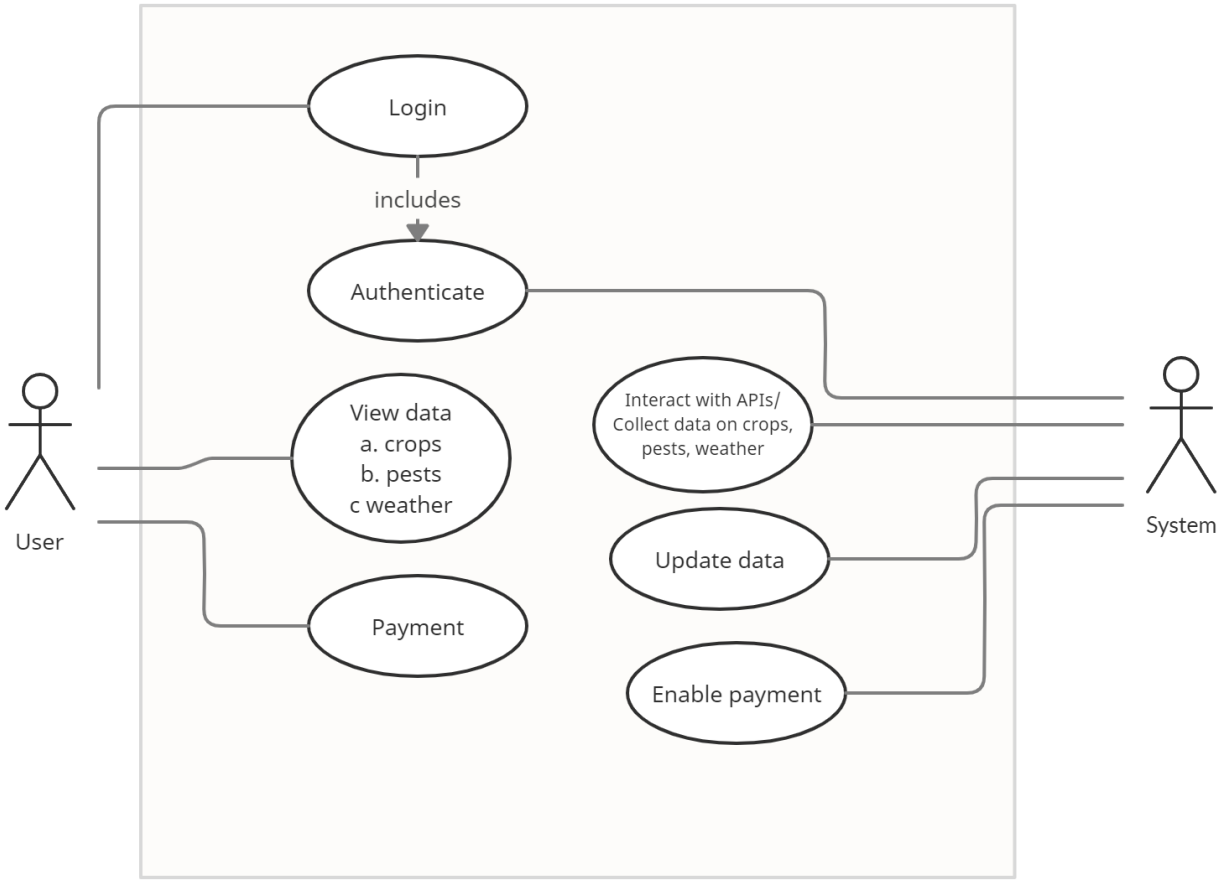
DESIGN

Activity Diagram



DESIGN

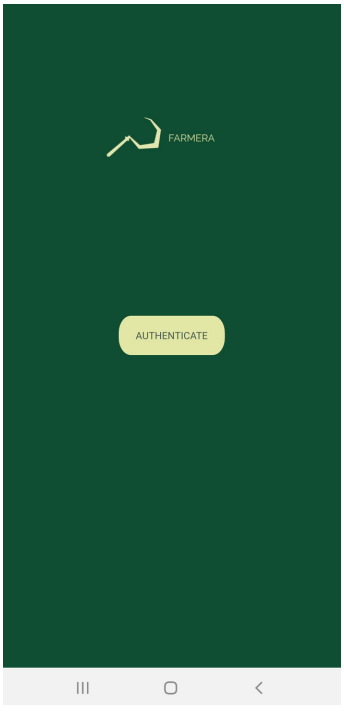
Use Case Diagram



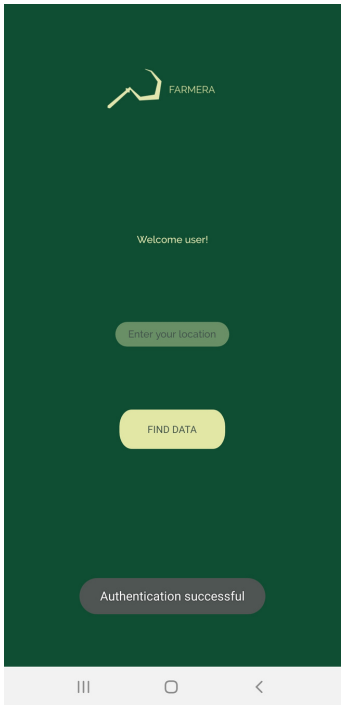
IMPLEMENTATION

The application was programmed in Java in Android studio as discussed earlier. The implementation followed a set template wherein we worked on individual functionalities and then pieced the application together. The general function triggering was caused when the corresponding buttons were pressed. Each function had different functionality which was visible usually on a new screen as is visible in the UI screenshots given.

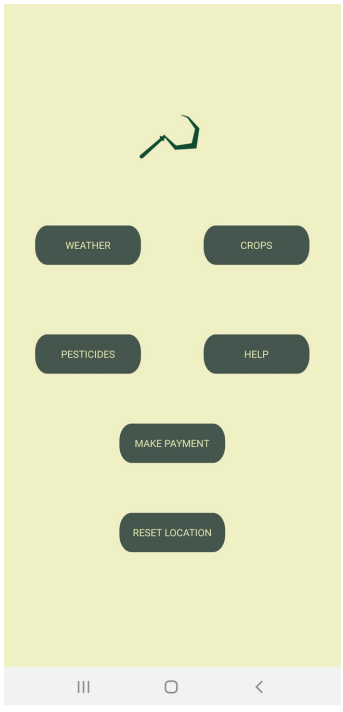
Authentication: We made use of the device's fingerprint sensor for authentication. In cases where a fingerprint sensor is not present on the device, the app makes use of the device lock pattern/PIN. The 'biometric' module for android was used for the same. To get this module to work, changes need to be made to AndroidManifest.xml to get permission to access the device's sensor. In addition, the biometric module must also be added as a dependency to the build.gradle file.



Authentication



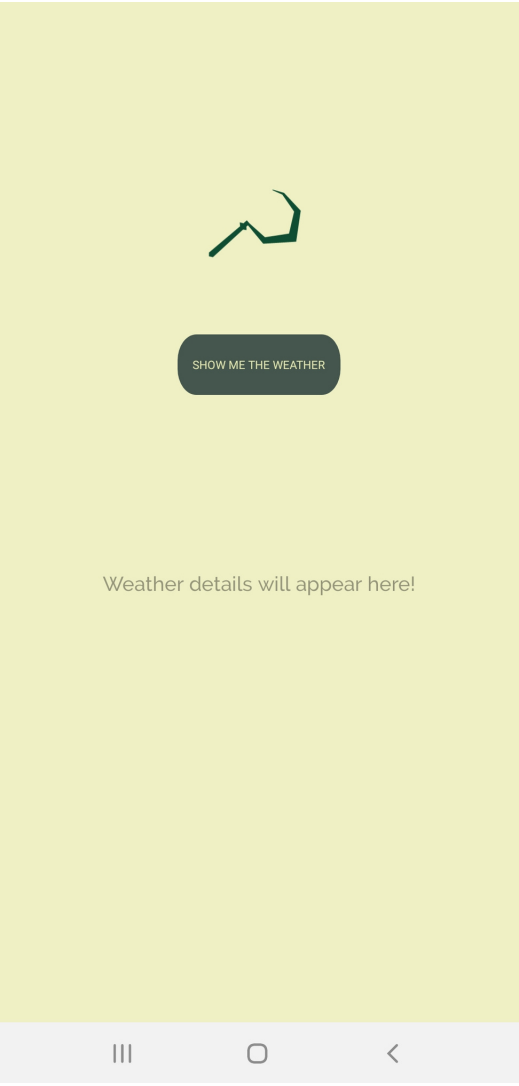
Information Entry



Home

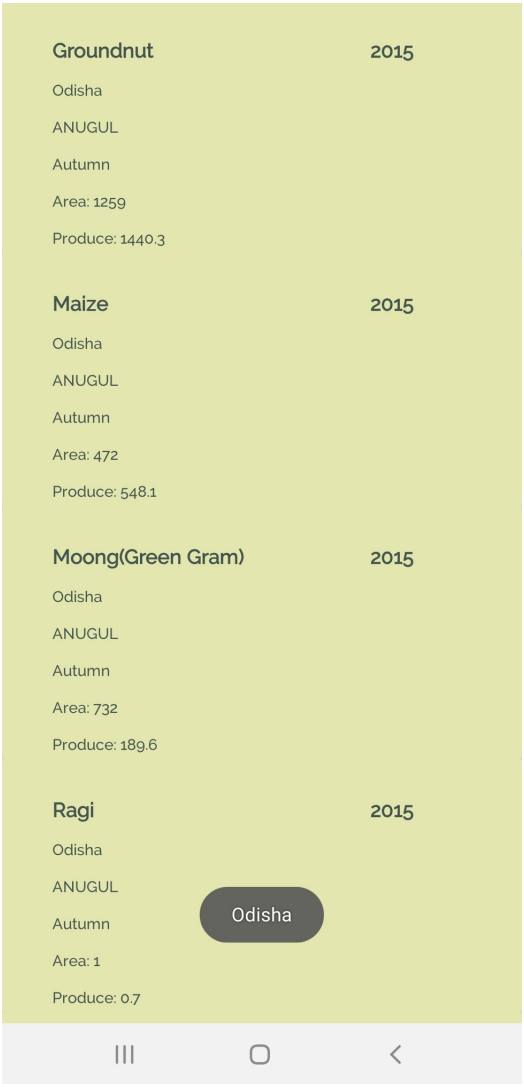
IMPLEMENTATION

Weather API: We accessed openweathermap.org's real time weather API to display the weather. Android's 'volley' module was used to get access to the internet and ping the API. The location entered initially is sent as a 'POST' request to the API which then returns a JSON file consisting of the real-time weather information. Again, to ensure smooth operation, changes need to be made to AndroidManifest.xml to get internet access permissions and build.gradle to import the volley module.

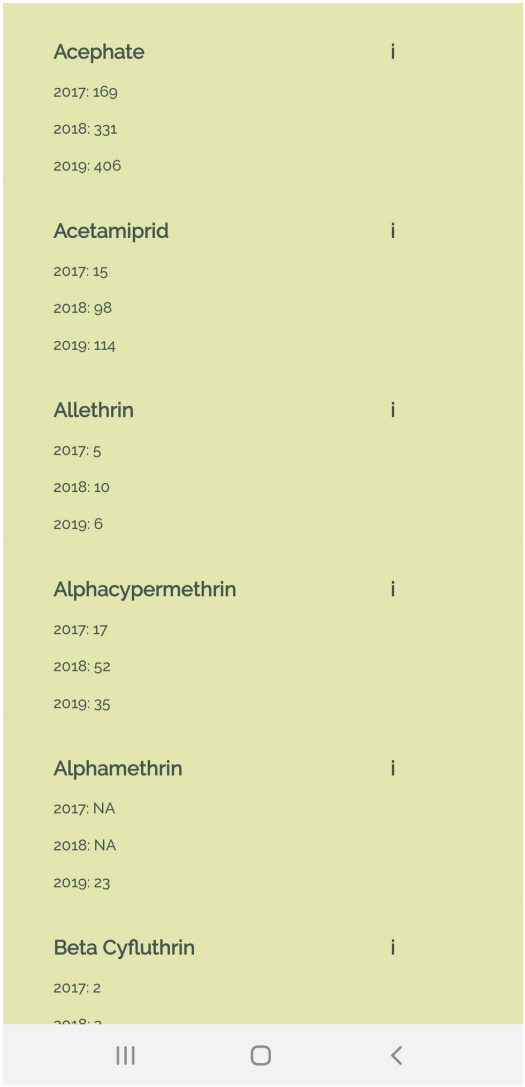


IMPLEMENTATION

Getting and displaying crop and pesticide data: We used the combination of the Android AsyncHTTP API(AHC) and JExcelApi(JXL) to handle the excel database. The Async API allows us to pull the excel file from the internet, using git for our app, and stores it in a workbook. This allows us to update the data in the excel as required as long as the URL remains the same. The JXL API then enables us to store and display the data. The JXL API lets us parse the sheet by storing it in a workbook. The data is then passed as ArrayLists to the Adapter's constructor. The Adapter class lets us relay the custom xml layout to the RecyclerView which we used to display the multiple entries for crops and pesticides.



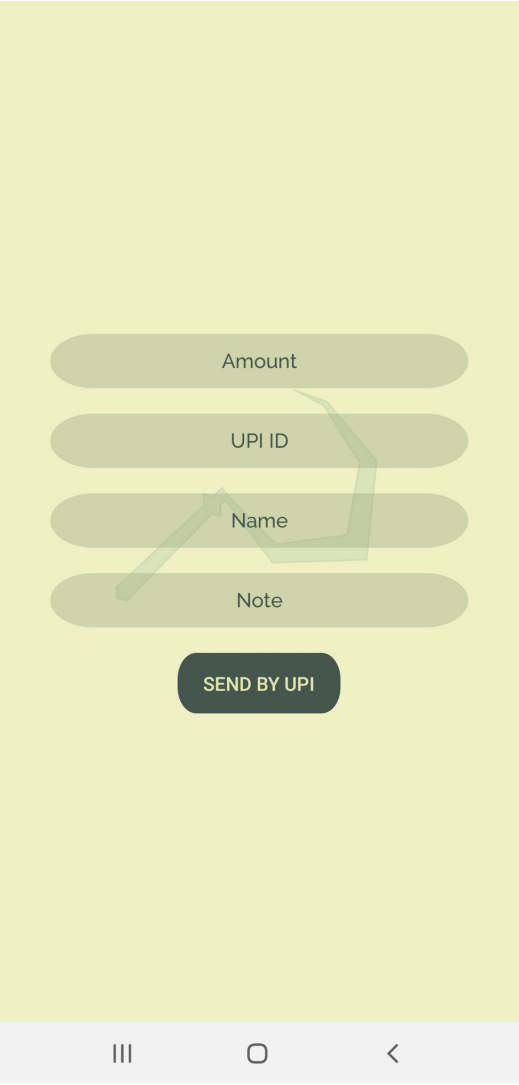
Crops



Pesticides

IMPLEMENTATION

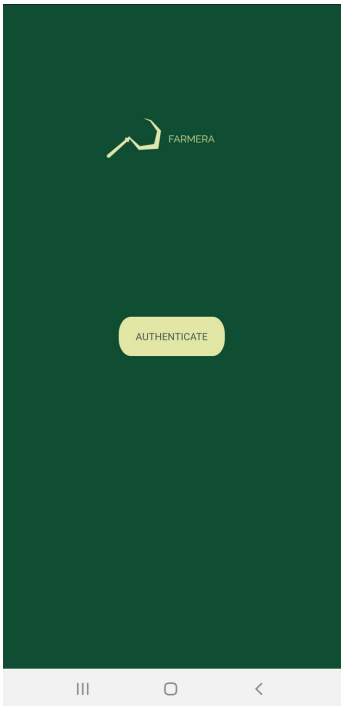
UPI Payments: We made use of google play services wallet API to allow UPI transactions from any application. Again, changes need to be made to AndroidManifest.xml to ensure permissions. The android ‘gms wallet’ dependency must be added to build.gradle to make use of the UPI service. As of now, transactions don’t actually go through as the app is still in developer mode.



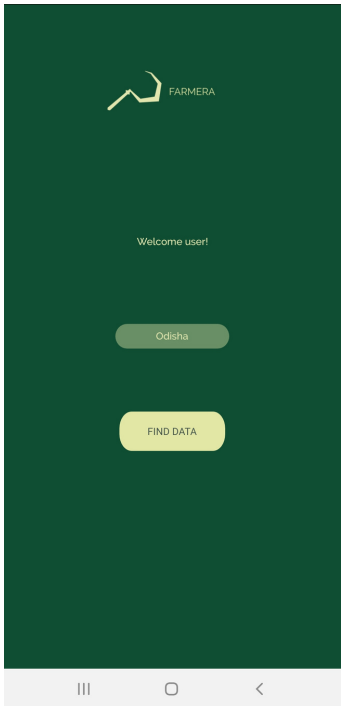
Payments

EXPERIMENTAL RESULT

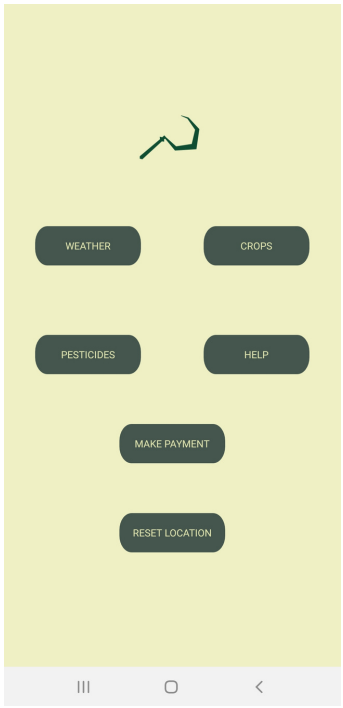
The results are shown in the screenshots below. We were able to implement all functionalities smoothly. Error handling was also taken care of and appropriate toast messages were also displayed.



Authentication

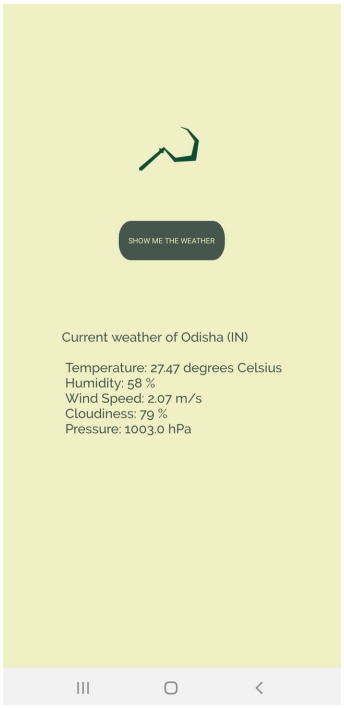


Information Entry



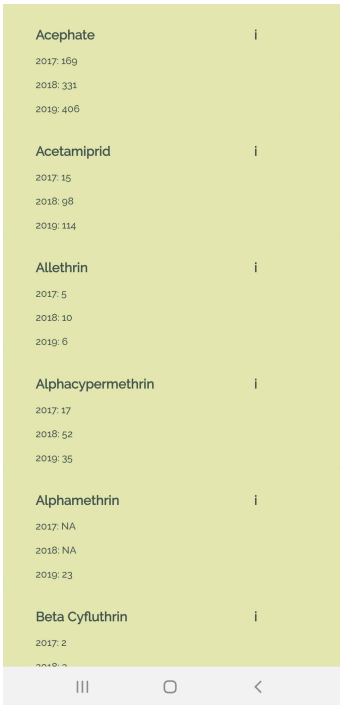
Home

EXPERIMENTAL RESULT

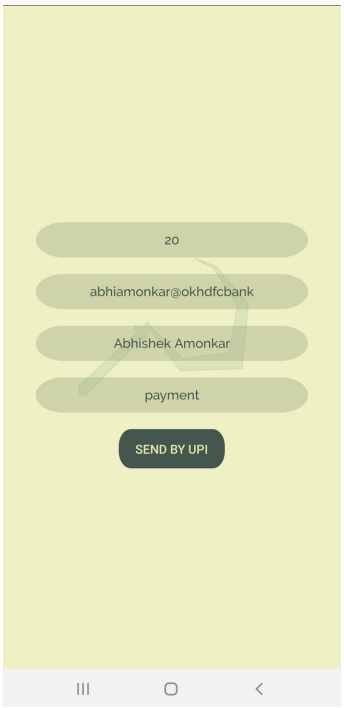


Weather

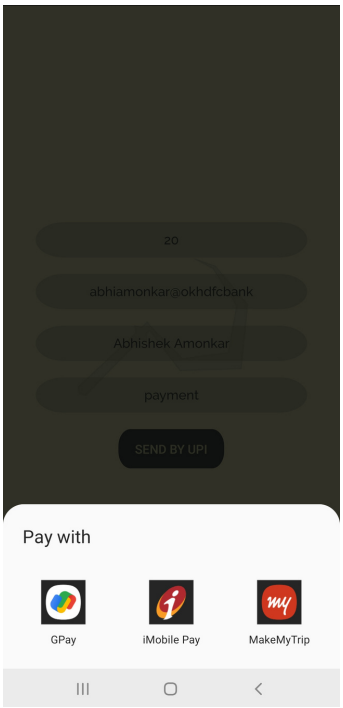
Crops



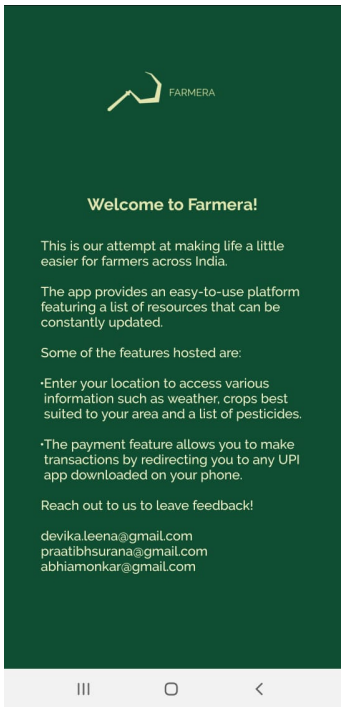
Pesticides



Payments



Payment redirection to UPI



Help

APP DEMO LINK

https://drive.google.com/file/d/1Qvt8OSmT0ziypo414WWVz-65Ou1PwnEs/view?usp=sharing

CONCLUSION

The goal of this application is to act as a bridge of information between the central repositories of data and farmers. The app provides them with information in a concise and direct manner while still being user-friendly. Even someone without a lot of experience with technology and access to resources should be able to successfully navigate the application and make full use of it.

FUTURE WORK

Future work can be aimed at making user experience a lot smoother. Currently, the application is a little slow due to the large volume of data that it is handling. We plan on optimizing our code to improve the overall speed of our application.
We also plan on rolling out an update to implement an expense manager for field related costs. Apart from this, efforts will be made to add more features in case of a lack of internet access.

REFERENCES

1. http://www.fao.org/india/fao-in-india/india-at-a-glance/en/
2. https://brainly.in/question/4427855
3. https://developer.android.com/studio/intro
4. https://www.javatpoint.com/features-of-java
5. https://www.ibm.com/docs/en/ssw_aix_72/performance/advantages_java.html
6. https://www.infoworld.com/article/3537561/how-secure-is-java-compared-to-other-languages.html
7. https://data.world/thatzprem/agriculture-india
8. http://ppqs.gov.in/statistical-database