BENGALURU NORTH UNIVERSITY

Tamaka, Kolar – 560 103

VI Sem BCA Project Report

MY FARM GUIDE

By

TEJAS R

(R1911939)

Under the Guidance of

Nithya Kalyani S

(Department of Computer Applications)

Smt. Danamma Channabasavaiah College of Arts, Commerce, Science And Management Studies

Kodiramasandra, NH – 75, Bypass, Kolar – 563103

(NAAC Accredited with 'B' Grade and affiliated to Bengaluru North University)

Academic Year 2019 - 22

Smt. Danamma Channabasavaiah College of Arts, Commerce, Science And Management Studies

Kodiramasandra, NH – 75, Bypass, Kolar – 563103

(NAAC Accredited with 'B' Grade and affiliated to Bengaluru North University)

DISCIPLINE & QUALITY

S.D.C. GROUP OF INSTITUTIONS

Certificate

Certified that the seminar entitled "**MY FARM GUIDE**" carried out by **TEJAS R** bonafide student of **Smt. Danamma Channabasavaiah College of Arts, Commerce, Science and Management Studies** in partial fulfilment for the award of the degree of **Bachelor of Computer Applications** of **Bengaluru North University** during the year 2019-2022. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed by the said degree.

Nithya Kalyani S	**Mr. Satyajit Ray**
Project Guide	HOD of Computer Science

External

Name of the Examiner	**Signature with Date**

Smt. Danamma Channabasavaiah College of Arts, Commerce, Science And Management Studies

Kodiramasandra, NH – 75, Bypass, Kolar – 563103

(NAAC Accredited with 'B' Grade and affiliated to Bengaluru North University)

I student of Sixth semester B.C.A, at the department of Computer Applications, Smt. Danamma Channabasavaiah College of Arts, Commerce, Science and Management Studies, declare that the Project entitled **"MY FARM GUIDE"** has been presented by me and submitted in partial fulfillment of course requirements for the award of degree in Bachelor of Computer Applications of Bengaluru North University, Tamaka, Kolar during the academic year 2019-2022.

Place: Kolar

Date:

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have a great pleasure in expressing my deep sense of gratitude to my respected founder chairperson **Smt. Usha Gangadhar** for having provided me with great infrastructure and well-furnished labs.

I take this opportunity to express my profound gratitude to our respected Principal **Prof. Pushpalatha K** for her constant support and encouragement.

I am grateful to the Vice Principal and Head of the Department Mr. Satyajit Ray, Department of Computer Science and Applications for his unfailing encouragement and suggestion given to me in the course of Project work.

Guidance and deadlines play a very important role in successful completion of the project report on time. I also convey my gratitude to **Nithya Kalyani S** , Assistant Professor, Department of Computer Science and Applications for having constantly monitored the development of the project report and setting up precise deadlines.

Finally, a note thanks to the Department of Science and Applications, both teaching and non-teaching staff for their co-operation extend to me.

ABSTRACT

Agriculture in India depends heavily on weather and climate conditions. Weather forecasts are useful for decisions regarding crop choice, crop variety, planting/harvesting dates, and investments in farm inputs such as irrigation, fertilizer, pesticide, herbicide etc. Hence, improved weather forecast based agromet advisory service greatly helps farmers to take advantage of benevolent weather and mitigate the impacts of malevolent weather situation. Medium range weather forecast based agro-meteorological advisory service of NCMRWF strives to improve and protect agricultural production, which is crucial for food security of the country. The weather forecast and advisories have been helping the farming community to take advantage of prognosticated weather conditions and form the response strategy. On many occasions Agro-Meteorological Field Units have reportedly saved the crop from unfavourable weather condition. Also the service, on many instances, helped farmers over different regions to minimize crop losses as a result of extreme weather conditions. Such reports were included in the Annual Progress Reports submitted by the Agro-Advisory Service (AAS) units as well as discussed during different review meetings of the project. But these were sporadic cases and could not be inter-compared mainly due to non uniform use of the methodology. Hence, a project entitled "“Economic Impact of AAS of NCMRWF” was formulated and launched in November 2003 to assess the use and value of the service, with a view not only to assess the economic impact of the service but also to assess its usage pattern and identify strengths and weaknesses to further improve it. The case studies include estimates for both perfect and imperfect forecasts. From a practical perspective, perfect forecasts are an unrealistic expectation, and on the other hand the less accurate forecasts also help farmers to determine farm management action and add information for decision making. Also, reporting a range of advisories which are based on weather forecasts with lower skill levels is also helpful in determining the degree of accuracy that is needed to further improve the service. Most of the economic evaluations of weather forecasts based advisories presented in the report are based on comparison of a set of information obtained from users against non-users and recorded at the individual farm level, on a per hectare basis. Majority of these studies, base the value of weather forecasts on precipitation and temperature forecasts which can aid in numerous farm level decision making strategies. Assessing impacts of weather forecast application in farm management sector is a stupendous task. The task becomes even more challenging if one is attempting to quantify the value of weather forecast based agro-advisories. It was difficult to consider all crop and all agro-climatic situations, hence a conscious decision was taken to undertake the study at 15 representative sites covering principal crops. The project was implemented at 15 AAS units. The study period was spread over three years comprising of 3 Kharif and 3 Rabi seasons. National Centre for Agriculture Economics and Policy Research (NCAP), who was engaged as consultant for the project, helped to formulate the study plan, including devising sampling method, preparation of questionnaire, monitoring its implementation and data analysis.

TABLE OF CONTENTS

1	Synopsis	1-2
2	Software Development Life Cycle	3-7
3	Software Requirement Specification	8-10
4	Software and Connectivity	11-18
5	Feasibility Study	19-20
6	System Analysis	21-22
7	ER Diagram	23-24
8	Data Flow Diagram	25-26
9	Snapshots	27-32
10	Coding	33-64
11	Data Base	65-68
12	Testing	69-72
13	Implementation	73-74
14	Future Enhancements	75-76
15	Conclusion	77-78
16	Bibliography	79-80

SYNOPSIS

Project Title: My Farm Guide

Brief Introduction: My Farm is one of the android application for the farmers which provide information to grow the Organic products in a proper manner. Organic food is grown without the use of synthetic chemicals, such as human-made pesticides and fertilizers, and does not contain genetically modified organisms and our application consists of the tasks to grow the plant and earn money and also the climate alerts for farmers to safeguard their crops from huge rain and protect the crops from infections. So, the farmer can grow good organic product which is healthy to life and also profitable. Now a days more chemical fertilizers are damaging the plants and products that are not healthy to life so grow an organic product for a healthy life is a main motive of our application.

A farmer is a person engaged in agriculture, raising living organisms for food or raw materials. The term usually applies to people who do some combination of raising field crops, orchards, vineyards, poultry, or other livestock. A farmer might own the farm land or might work as a labourer on land owned by others. In most developed economies, a "farmer" is usually a farm owner (landowner), while employees of the farm are known as farm workers (or farmhands). However, in other older definitions a farmer was a person who promotes or improves the growth of plants, land or crops or raises animals (as livestock or fish) by labour and attention.

Over half a billion farmers are smallholders, most of whom are in developing countries, and who economically support almost two billion people. Globally, women constitute more than 40% of agricultural employees.

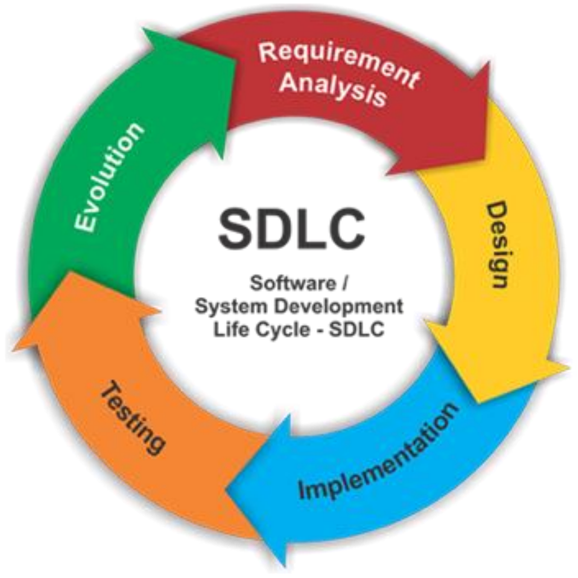
Agriculture or farming is the practice of cultivating plants and livestock. Agriculture was the key development in the rise of sedentary human civilization, whereby farming of domesticated species created food surpluses that enabled people to live in cities. The history of agriculture began thousands of years ago. After gathering wild grains beginning at least 105,000 years ago, nascent farmers began to plant them around 11,500 years ago. Pigs, sheep, and cattle were domesticated over 10,000 years ago. Plants were independently cultivated in at least 11 regions of the world. Industrial agriculture based on large-scale monoculture in the twentieth century came to dominate agricultural output, though about 2 billion people still depended on subsistence agriculture.

The major agricultural products can be broadly grouped into foods, fibers, fuels, and raw materials (such as rubber). Food classes include cereals (grains), vegetables, fruits, oils, meat, milk, eggs, and fungi. Over one-third of the world's workers are employed in agriculture, second only to the service sector, although in recent decades, the global trend of a decreasing number of agricultural workers continues, especially in developing countries, where smallholding is being overtaken by industrial agriculture and mechanization that brings an enormous crop yield increase.

Modern agronomy, plant breeding, agrochemicals such as pesticides and fertilizers, and technological developments have sharply increased crop yields, but cause ecological and environmental damage. Selective breeding and modern practices in animal husbandry have similarly increased the output of meat but have raised concerns about animal welfare and environmental damage. Environmental issues include contributions to global warming, depletion of aquifers, deforestation, antibiotic resistance, and other agricultural pollution. Agriculture is both a cause of and sensitive to environmental degradation, such as biodiversity loss, desertification, soil degradation, and global warming, all of which can cause decreases in crop yield. Genetically modified organisms are widely used, although some are banned in certain countries.

SOFTWARE DEVELOPMENT LIFE CYCLE

Software Development Life Cycle (SDLC):



SDLC is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customer expectations. The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life Cycle has its own process and deliverables that feed into the next phase. SDLC stands for **Software Development Life Cycle** and is also referred to as the Application Development life-cycle.

Phases of SDLC are:

- Requirement Analysis
- Design
- Implementation
- Testing
- Evolution

1. Requirement Analysis:

The requirement is the first stage in the SDLC process. It is conducted by the senior team members with inputs from all the stakeholders and domain experts in the industry. Planning for the quality assurance requirements and reorganization of the risks involved is also done at this stage.

2. Design:

In this phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture.

High-Level Design (HLD)

- Brief description and name of each module
- An outline about the functionality of every module
- Interface relationship and dependencies between modules

Low-Level Design (LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface

3. Implementation:

Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process.

4. Testing:

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.

During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system.

5. Evolution:

Once the system is deployed, and customers start using the developed system, following 3 activities occur

- Bug fixing – bugs are reported because of some scenarios which are not tested at all
- Upgrade – Upgrading the application to the newer versions of the Software
- Enhancement – Adding some new features into the existing software

The main focus of this SDLC phase is to ensure that needs continue to be met and that the system continues to perform as per the specification mentioned in the first phase.

Software process models:

Software process model is a simple description of software process. Some popular types of software process models are:

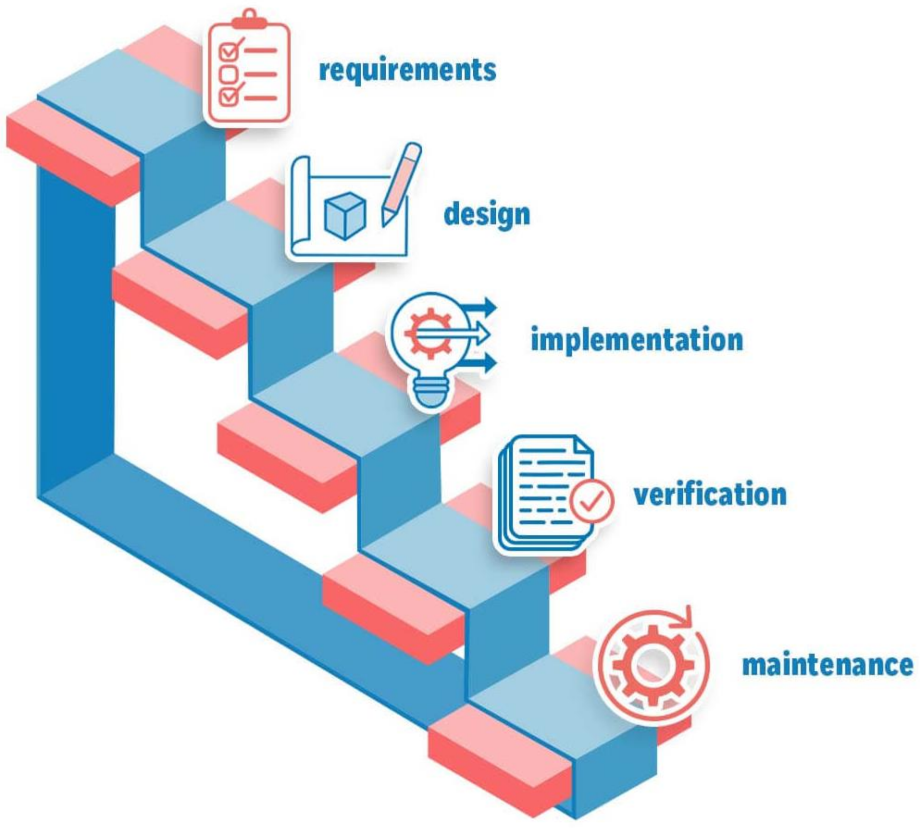
- Waterfall Model
- Spiral Model
- Iterative Enhancement Model

For the development of our project, we used **Waterfall Model.**

Waterfall Model (Linear Sequential Model):

The Waterfall model—is a sequential development process that flows like a waterfall through all phases of a project (analysis, design, development, and testing, for example), with each phase completely wrapping up before the next phase begins.

It is said that the Waterfall methodology follows the adage to "measure twice, cut once." The success of the Waterfall method depends on the amount and quality of the work done on the front end, documenting everything in advance, including the user interface, user stories, and all the features' variations and outcomes. With the majority of the research done upfront, estimates of the time needed for each requirement are more accurate, and this can provide a more predictable release date. With a Waterfall project, if parameters change along the way, it's harder to change course than it is with Agile methodology.



The Waterfall methodology follows a chronological process and works based on fixed dates, requirements, and outcomes. With this method, the individual execution teams aren't required to be in constant communication and, unless specific integrations are required, are usually self-contained. Team members also tend to work independently and aren't expected to provide status reports as often as with the Agile approach. Usually, one phase doesn't begin until the previous one is finished.

1. **Requirements:**
   The Waterfall methodology depends on the belief that all project requirements can be gathered and understood upfront. The project manager does their best to get a detailed understanding of the project sponsor's requirements. Written requirements, usually contained in a single document, are used to describe each stage of the project.

2. **Design:**
   Here, software developers design a technical solution to the problems set out by the product requirements, including scenarios, layouts, and data models. First, a higher-level or logical design is created that describes the purpose and scope of the project, the general traffic flow of each component, and the integration points. Once this is complete, it is transformed into a physical design using specific hardware and software technologies.

3. **Implementation:**
   Once the design is complete, technical implementation starts. This might be the shortest phase of the Waterfall process, because painstaking research and design have already been done. In this phase, programmers code applications based on project requirements and specifications, with some testing and implementation taking place as well. If significant changes are required during this stage, this may mean going back to the design phase.

4. **Verification:**
   Before a product can be released to customers, testing needs to be done to ensure the product has no errors and all of the requirements have been completed, ensuring a good user experience with the software. The testing team will turn to the design documents, personas, and user case scenarios supplied by the product manager to create their test cases.

5. **Maintenance:**
   Once the software has been deployed in the market or released to customers, the maintenance phase begins. As defects are found and change requests come in from users, a team will be assigned to take care of updates and release new versions of the software.

Advantages of Waterfall Model:

- Developers can catch design errors during the analysis and design stages, helping them to avoid writing faulty code during the implementation phase.
- The total cost of the project can be accurately estimated, as can the timeline, after the requirements have been defined.
- With the structured approach, it is easier to measure progress according to clearly defined milestones.

Disadvantages of Waterfall Model:

- Projects can take longer to deliver with this chronological approach than with an iterative one, such as the Agile method.
- Clients often don’t fully know what they want at the front end, opening the door to requests for changes and new features later in the process when they’re harder to accommodate.
- Clients are not involved in the design and implementation stages.

SOFTWARE REQUIREMENT SPECIFICATION

Software Requirement Specification:

The production of the requirements stage of the software development process is **Software Requirements Specifications (SRS)** (also called a **requirements document**). This report lays a foundation for software engineering activities and is constructing when entire requirements are elicited and analysed. **SRS** is a formal report, which acts as a representation of software that enables the customers to review whether it (SRS) is according to their requirements. Also, it comprises user requirements for a system as well as detailed specifications of the system requirements.

The SRS is a specification for a specific software product, program, or set of applications that perform particular functions in a specific environment. It serves several goals depending on who is writing it. First, the SRS could be written by the client of a system. Second, the SRS could be written by a developer of the system. The two methods create entirely various situations and establish different purposes for the document altogether. The first case, SRS, is used to define the needs and expectation of the users. The second case, SRS, is written for various purposes and serves as a contract document between customer and developer.

Following are the features of a good SRS document:

1. Correctness: User review is used to provide the accuracy of requirements stated in the SRS. SRS is said to be perfect if it covers all the needs that are truly expected from the system.

2. Completeness: All essential requirements, whether relating to functionality, performance, design, constraints, attributes, or external interfaces.

3. Consistency: The SRS is consistent if, and only if, no subset of individual requirements described in its conflict. There are three types of possible conflict in the SRS:

4. Unambiguousness: SRS is unambiguous when every fixed requirement has only one interpretation. This suggests that each element is uniquely interpreted. In case there is a method used with multiple definitions, the requirements report should determine the implications in the SRS so that it is clear and simple to understand.

5. Ranking for importance and stability: The SRS is ranked for importance and stability if each requirement in it has an identifier to indicate either the significance or stability of that particular requirement.

6. Modifiability: SRS should be made as modifiable as likely and should be capable of quickly obtain changes to the system to some extent. Modifications should be perfectly indexed and cross-referenced.

7. Verifiability: SRS is correct when the specified requirements can be verified with a cost-effective system to check whether the final software meets those requirements. The requirements are verified with the help of reviews.

8. Traceability: The SRS is traceable if the origin of each of the requirements is clear and if it facilitates the referencing of each condition in future development or enhancement documentation.

Functional Requirements

General description of inputs and outputs. The system has basically a menu driven input format. The officer has to choose from the menu, the appropriate options.

External Interface Requirements

User interface is of the most important parts of the effective software. A menu driven system is to be developed using which the user does it. The interface should reinforce.

Hardware and Software Specifications

- **Hardware Specification:**

Processor	: Clock Speed 1.80 GHz or Above
RAM	: 2.00 GB or Above
Hard Disk	: 80GB or Above
System Type	: 64 – bit Operating System or Above
Monitor	: Any Display Unit
Input Type	: Any Input Devices

- **Software Specification:**

Operating System	: Windows 7 or Above
Frontend	: Android Studio (XML , Java)
Backend	: Firebase (Java)
Database	: Firebase Database

- **Mobile Phone Specification:**

Processor	: Clock Speed 1.55 GHz or Above
RAM	: 1.00 GB or Above
ROM	: 50 MB or Above
Operating System	: Android 5.0 and Above (With active Internet Connection)

SYSTEM TOOLS AND CONNECTIVITY

Database

In computing, a **database** is an organized collection of data stored and accessed electronically. Small databases can be stored on a file system, while large databases are hosted on computer clusters or cloud storage. The design of databases spans formal techniques and practical considerations, including data modelling, efficient data representation and storage, query languages, security and privacy of sensitive data, and distributed computing issues, including supporting concurrent access and fault tolerance.

A **database management system** (**DBMS**) is the software that interacts with end users, applications, and the database itself to capture and analyse the data. The DBMS software additionally encompasses the core facilities provided to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a **database system.** Often the term "database" is also used loosely to refer to any of the DBMS, the database system or an application associated with the database.

Computer scientists may classify database management systems according to the database models that they support. Relational databases became dominant in the 1980s. These model data as rows and columns in a series of tables, and the vast majority use SQL for writing and querying data. In the 2000s, non-relational databases became popular, collectively referred to as NoSQL, because they use different query languages.

Existing DBMSs provide various functions that allow management of a database and its data which can be classified into four main functional groups:

- **Data definition** – Creation, modification and removal of definitions that define the organization of the data.
- **Update** – Insertion, modification, and deletion of the actual data.
- **Retrieval** – Providing information in a form directly usable or for further processing by other applications. The retrieved data may be made available in a form basically the same as it is stored in the database or in a new form obtained by altering or combining existing data from the database.
- **Administration** – Registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information that has been corrupted by some event such as an unexpected system failure.

FIREBASE

Firebase evolved from Envolve, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that were not chat messages. Developers were using Envolve to sync application data such as game state in real time across their users. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firebase as a separate company in 2011 and it launched to the public in April 2012.

Firebase's first product was the Firebase Realtime Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firebase's cloud. The product assists software developers in building real-time, collaborative applications.

In May 2012, a month after the beta launch, Firebase raised $1.1 million in seed funding from venture capitalists Flybridge Capital Partners, Greylock Partners, Founder Collective, and New Enterprise Associates. In June 2013, the company further raised $5.6 million in Series A funding from Union Square Ventures and Flybridge Capital Partners.

In 2014, Firebase launched two products: Firebase Hosting and Firebase Authentication. This positioned the company as a mobile backend as a service.

In October 2014, Firebase was acquired by Google. A year later, in October 2015, Google acquired Divshot, an HTML5 web-hosting platform, to merge it with the Firebase team.

In May 2016, at Google I/O, the company's annual developer conference, Firebase introduced Firebase Analytics and announced that it was expanding its services to become a unified backend-as-a-service (BaaS) platform for mobile developers. Firebase now integrates with various other Google services, including Google Cloud Platform, AdMob, and Google Ads to offer broader products and scale for developers. Google Cloud Messaging, the Google service to send push notifications to Android devices, was superseded by a Firebase product, Firebase Cloud Messaging, which added the functionality to deliver push notifications to both iOS and web devices.

In July 2016, Google announced that it was acquiring the mobile developer platform LaunchKit, which specialized in app developer marketing, and would be folding it into the Firebase Growth Tools team. In January 2017, Google acquired Fabric and Crashlytics from Twitter to add those services to Firebase.

In October 2017, Firebase launched Cloud Firestore, a real-time document database as the successor product to the original Firebase Realtime Database.

ANDROID STUDIO

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' 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 (E-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.

On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development. Java is still supported, as is C++.

The following features are provided in the current stable version:

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems
- Pro Guard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Android Studio supports all the same programming languages of IntelliJ (and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

Once an app has been compiled with Android Studio, it can be published on the Google Play Store. The application has to be in line with the Google Play Store.

XML

Extensible Markup Language (**XML**) is a markup language and file format for storing, transmitting, and reconstructing arbitrary data. It defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications—all of them free open standards—define XML.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.

The main purpose of XML is serialization, i.e. storing, transmitting, and reconstructing arbitrary data. For two disparate systems to exchange information, they need to agree upon a file format. XML standardizes this process. XML is analogous to a lingua franca for representing information.

As a markup language, XML labels, categorizes, and structurally organizes information. XML tags represent the data structure and contain metadata. What's within the tags is data, encoded in the way the XML standard specifies. An additional XML schema (XSD) defines the necessary metadata for interpreting and validating XML. (This is also referred to as the canonical schema.) An XML document that adheres to basic XML rules is "well-formed"; one that adheres to its schema is "valid."

IETF RFC 7303 (which supersedes the older RFC 3023), provides rules for the construction of media types for use in XML message. They are used for transmitting raw XML files without exposing their internal semantics. RFC 7303 further recommends that XML-based languages.

Further guidelines for the use of XML in a networked context appear in RFC 3470, also known as IETF BCP 70, a document covering many aspects of designing and deploying an XML-based language.

Some Commonly used Tags

Tags	Use
(<XML>. . . </XML>)*	The entire XML document
(<RELATIVE LAYOUT>	The Relative layout for an XML android page
(<IMAGEVIEW>	Inserting Image in the XML Page
<TEXTVIEW>	Inserting Text in the XML Page
<BUTTON >	Inserting Button in the XML Page
<SHAPE>	Shape to the android page will be Designed in the Drawable by the shape tag.
<BACKGROUND>	Background to the shape or a layout
<SEARCHVIEW>	**Search Bar** in the android page
<!- . . . ->	**Comment** The comments you write in the middle will not show up on the page when viewed.
<FONT-SIZE >	**Font size** to the text or string
<STYLES>	This xml is used to define different styles and looks for the UI(User Interface) of application. We define our custom themes and styles in this file.
<COLOR>	This file is used to define the color codes that we used in our app. We simply define the color's in this file and used them in our app from this file.
< DIMENSION >	This xml file is used to define the dimensions of the View's. Suppose we need a Button with 50dp(density pixel) height then we define the value 50dp in dimens.xml file and then use it in our app from this file.
< STRINGS >	This xml file is used to replace the Hard-coded strings with a single string. We define all the strings in this xml file and then access them in our app(Activity or in Layout XML files) from this file. This file enhance the reusability of the code.
<TABLE>	"Table"=Starts a table. "TR" (Table Row) = Starts a row. "TD" (Table Data) = Starts a cell to enter data.

JAVA

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client–server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems. It was released in May 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own Hot Spot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open-source software and used by most developers and is the default JVM for almost all Linux distributions.

As of March 2022, Java 18 is the latest version, while Java 17, 11 and 8 are the current long-term support (LTS) versions. Oracle released the last zero-cost public update for the legacy version Java 8 LTS in January 2019 for commercial use, although it will otherwise still support Java 8 with public updates for personal use indefinitely. Other vendors have begun to offer zero-cost builds of OpenJDK 18 and 8, 11 and 17 that are still receiving security and other upgrades.

Oracle (and others) highly recommend uninstalling outdated and unsupported versions of Java, due to unresolved security issues in older versions. Oracle advises its users to immediately transition to a supported version, such as one of the LTS versions (8, 11, 17).

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

Sun Microsystems released the first public implementation as Java 1.0 in 1996. It promised write once, run anywhere (WORA) functionality, providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run Java applets within web pages, and Java quickly became popular. The Java 1.0 compiler was re-written in Java by Arthur van Hoff to comply strictly with the Java 1.0 language specification. With the advent of Java 2 (released initially as J2SE 1.2 in December 1998 – 1999), new versions had multiple configurations built for different types of platforms. J2EE included technologies and APIs for enterprise applications typically run in server environments, while J2ME featured APIs optimized for mobile applications. The desktop version was renamed J2SE. In 2006, for marketing purposes, Sun renamed new J2 versions as Java EE, Java ME, and Java SE, respectively.

In 1997, Sun Microsystems approached the ISO/IEC JTC 1 standards body and later the Ecma International to formalize Java, but it soon withdrew from the process. Java remains a de facto standard, controlled through the Java Community Process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine (JVM) as free and open-source software (FOSS), under the terms of the GPL-2.0-only license. On May 8, 2007, Sun finished the process, making all of its JVM's core code available under free software/open-source distribution terms, aside from a small portion of code to which Sun did not hold the copyright.

Sun's vice-president Rich Green said that Sun's ideal role with regard to Java was as an evangelist. Following Oracle Corporation's acquisition of Sun Microsystems in 2009–10, Oracle has described itself as the steward of Java technology with a relentless commitment to fostering a community of participation and transparency.[35] This did not prevent Oracle from filing a lawsuit against Google shortly after that for using Java inside the Android SDK (see the Android section).

On April 2, 2010, James Gosling resigned from Oracle.

In January 2016, Oracle announced that Java run-time environments based on JDK 9 will discontinue the browser plugin.

FEASIBILITY STUDY

Feasibility study:

A feasibility study is a detailed analysis that considers all of the critical aspects of a proposed project in order to determine the likelihood of it succeeding. Success in business may be defined primarily by return on investment, meaning that the project will generate enough profit to justify the investment. However, many other important factors may be identified on the plus or minus side, such as community reaction and environmental impact. Although feasibility studies can help project managers determine the risk and return of pursuing a plan of action, several steps should be considered before moving forward.

1. Technical Feasibility

The technological resources that are used by the company should undergo an examination. It assists businesses in determining whether technical resources are sufficient for the job and whether the technical team is capable of executing planned concepts. To assess the technical viability factors like the system's hardware, software, and other possible technical needs are taken into account.

2. Economic Feasibility

A cost-benefit analysis of the project is frequently included in this review, which helps firms determine the project's viability, cost, and benefits before investing financial resources. It also serves as an unbiased project evaluation, enhancing project credibility by supporting decision-makers in identifying the proposed project's favorable economic benefits to the company.

3. Legal feasibility

This assessment looks into if any component of the proposed project violates any regulations, such as zoning laws, data protection legislation, or social media laws. Let us take an example that a company that desires to build a new office building at a particular location. A feasibility study may discover that the desired location for the company is not designated for that sort of business.

4. Operational Feasibility

This assessment entails researching to evaluate whether — and to what extent — the project can meet the organization's needs. Operational feasibility studies also look at how a project plan meets the requirements specified during the requirement analysis phase.

5. Scheduling Feasibility

Scheduling feasibility evaluation is crucial for project success; after all, the project will fail if it is not completed on time. A company forecasts the amount of time it will take for project execution when scheduling feasibility.

SYSTEM ANALYSIS

System Analysis

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

Analysis specifies **what the system should do**.

Existing System

There are some applications for the farmers to buy and sell their product directly to the customers with any middlemen so the farmer can make good profit. And other information for the farmers but the farmers will not getting the benefits of the technologies.

Proposed System

In our application we provide the information for the farmers how to grow the plants in a farm and how to protect the plant with the diseases and Grow a plant step by step. We have provided the information weekly basis. This will help the farmer to do task on weekly basis and protect the farm and crop.

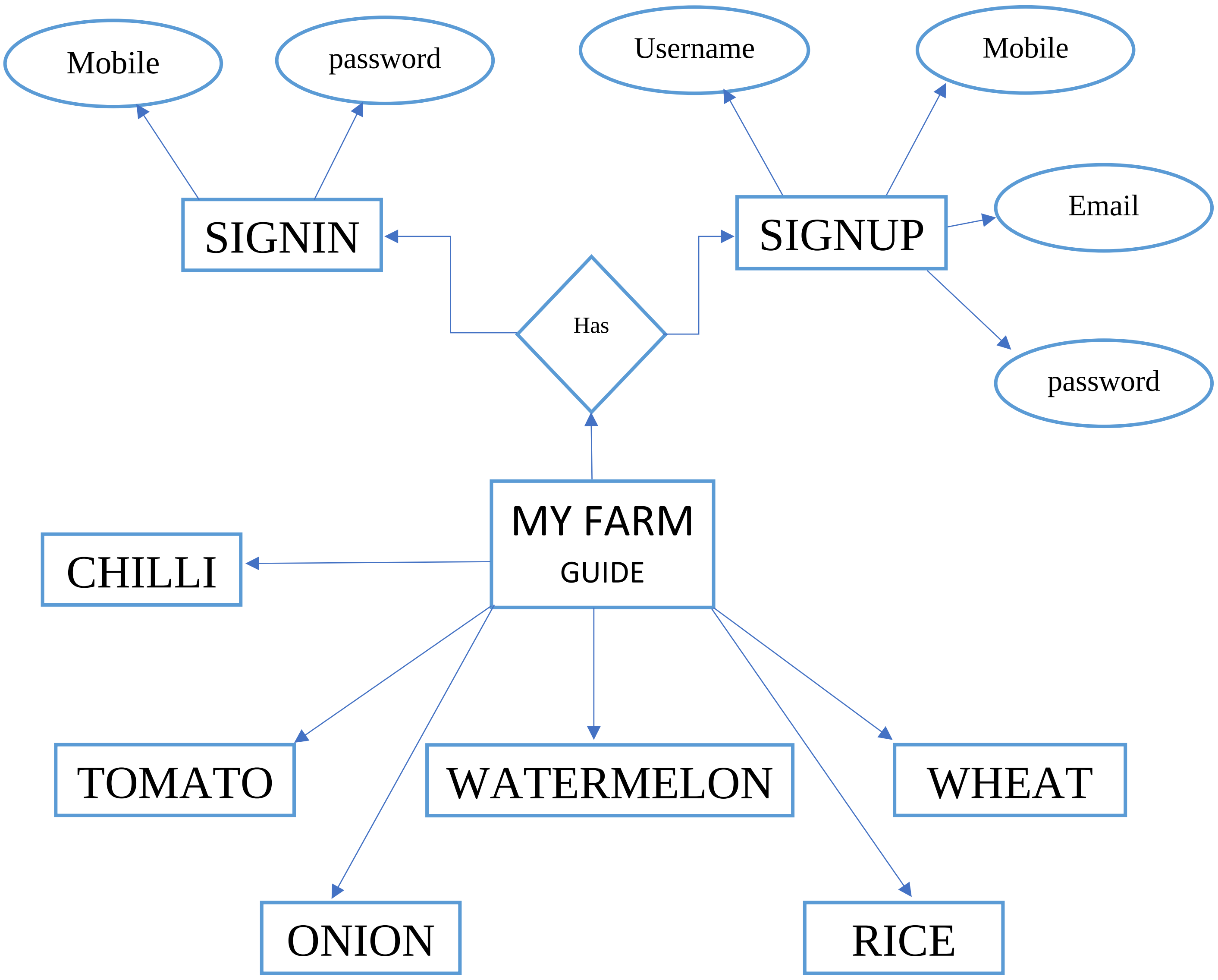
ER DIAGRAM
(Entity – Relationship)

Entity Relationship Diagram

An **entity–relationship model** (or **ER model**) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between entities (instances of those entity types).

In software engineering, an ER model is commonly formed to represent things a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model, that defines a data or information structure which can be implemented in a database, typically a relational database.

Entity–relationship modelling was developed for database and design by Peter Chen and published in a 1976 paper, with variants of the idea existing previously. Some ER models show super and subtype entities connected by generalization-specialization relationships, and an ER model can be used also in the specification of domain-specific ontologies.



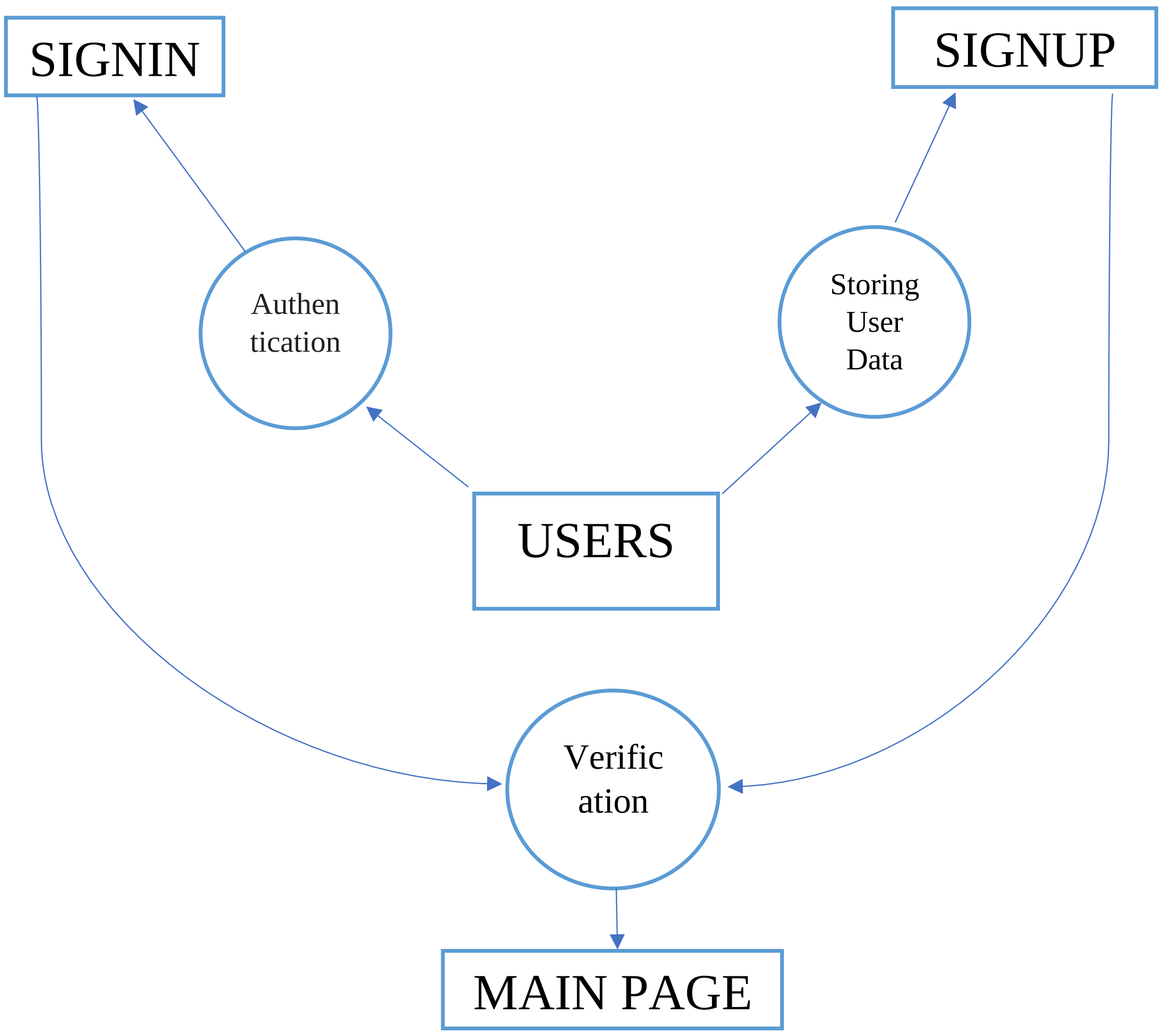
DFD

Data Flow Diagram

Data Flow Diagram

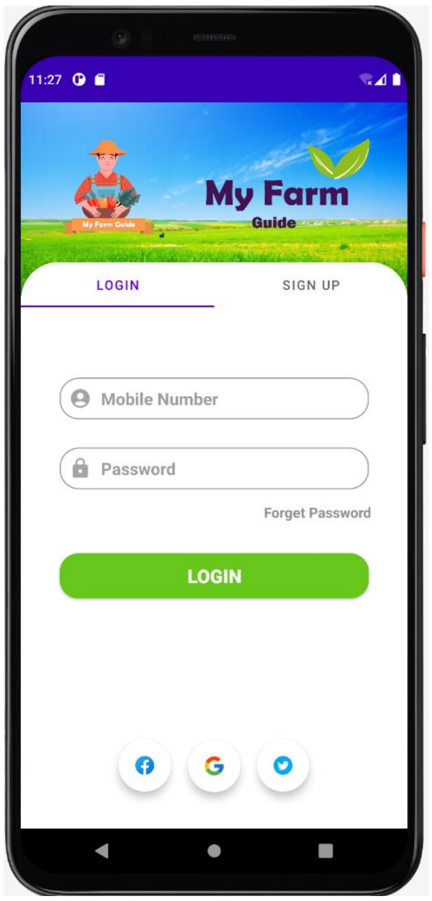
A **data-flow diagram** is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart. There are several notations for displaying data-flow diagrams. The notation presented above was described in 1979 by Tom DeMarco as part of structured analysis.

For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes. The data-flow diagram is a tool that is part of structured analysis and data modelling. When using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan. Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.

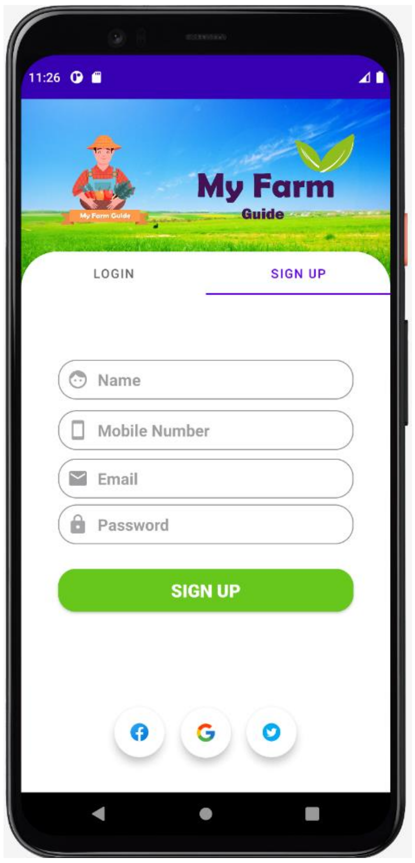


SNAPSHOTS

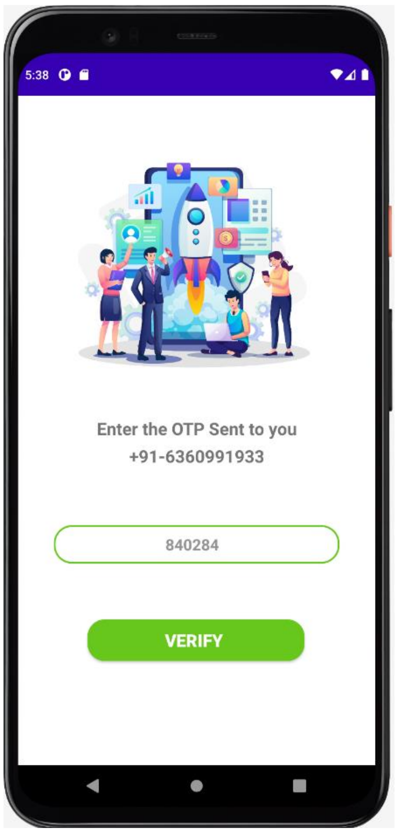
1) Login Page



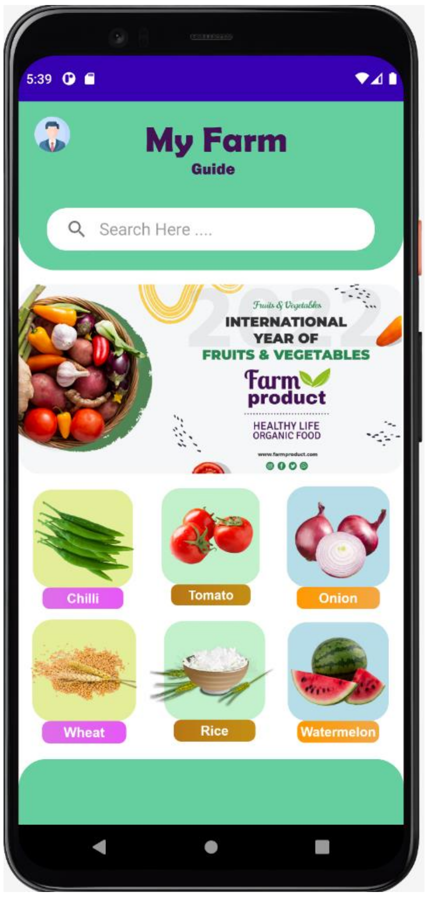
2) Signup Page



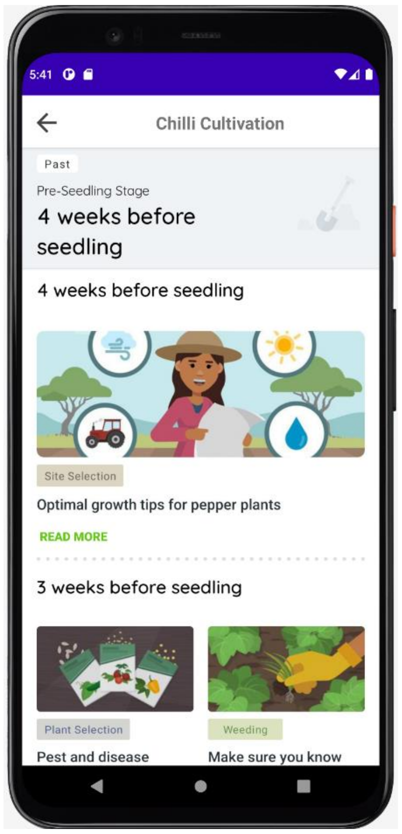
3) OTP Verification



4) Main Page



5) Chilli Cultivation


5:41

Chilli Cultivation

Past

Pre-Seedling Stage

4 weeks before seedling

4 weeks before seedling

Optimal growth tips for pepper plants

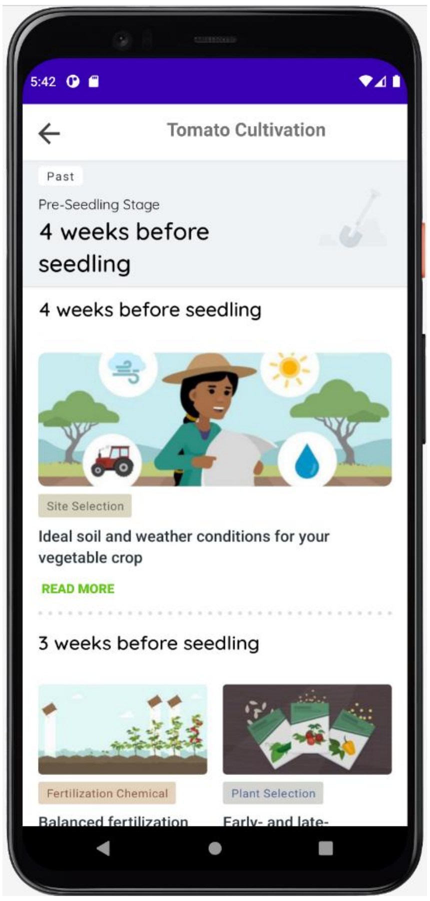
READ MORE

3 weeks before seedling

Pest and disease

Make sure you know

6) Tomato Cultivation


5:42

Tomato Cultivation

Past

Pre-Seedling Stage

4 weeks before seedling

4 weeks before seedling

Ideal soil and weather conditions for your vegetable crop

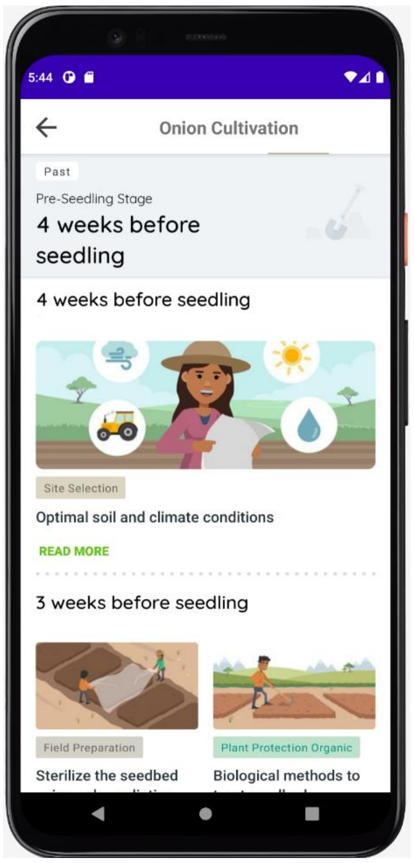
READ MORE

3 weeks before seedling

Balanced fertilization

Early- and late-

7) Onion Cultivation


5:44

Onion Cultivation

Past

Pre-Seedling Stage

4 weeks before seedling

4 weeks before seedling

Site Selection

Optimal soil and climate conditions

READ MORE

3 weeks before seedling

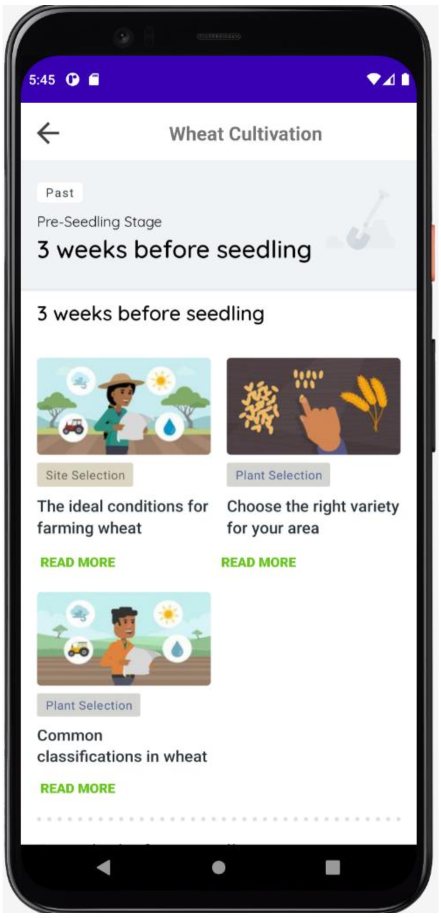
Field Preparation

Sterilize the seedbed

Plant Protection Organic

Biological methods to

8) Wheat Cultivation


5:45

Wheat Cultivation

Past

Pre-Seedling Stage

3 weeks before seedling

3 weeks before seedling

Site Selection

The ideal conditions for farming wheat

READ MORE

Plant Selection

Choose the right variety for your area

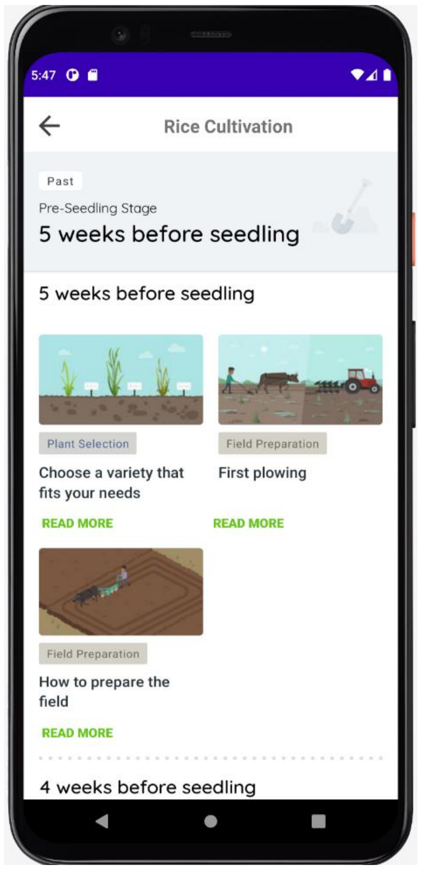
READ MORE

Plant Selection

Common classifications in wheat

READ MORE

9) Rice Cultivation



5:47

Rice Cultivation

Past

Pre-Seedling Stage

5 weeks before seedling

5 weeks before seedling

Plant Selection

Choose a variety that fits your needs

READ MORE

Field Preparation

First plowing

READ MORE

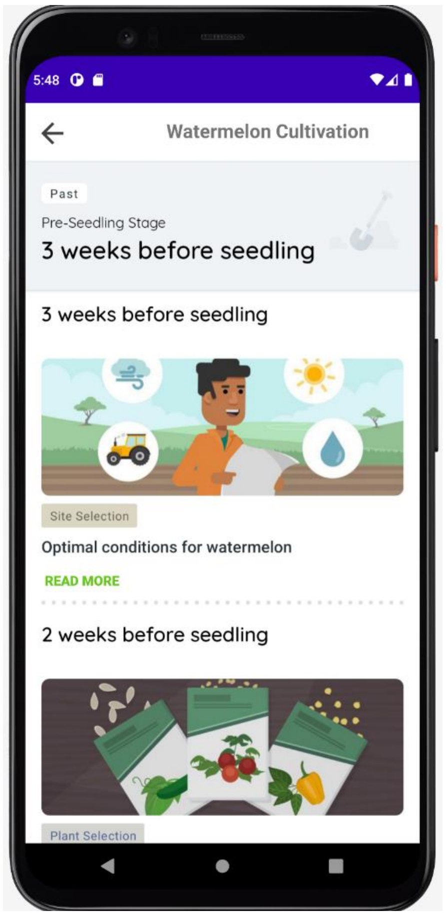
Field Preparation

How to prepare the field

READ MORE

4 weeks before seedling

10) Watermelon Cultivation



5:48

Watermelon Cultivation

Past

Pre-Seedling Stage

3 weeks before seedling

3 weeks before seedling

Site Selection

Optimal conditions for watermelon

READ MORE

2 weeks before seedling

Plant Selection

CODING

1) Mainpage.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".LoginActivity">

   <ImageView
      android:id="@+id/imageView"
      android:layout_width="wrap_content"
      android:layout_height="350dp"
      android:scaleType="centerCrop"
      android:src="@drawable/green"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHeight_percent=".27"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      app:layout_constraintVertical_bias="0"
      tools:ignore="MissingConstraints" />

   <androidx.constraintlayout.widget.ConstraintLayout
      android:id="@+id/constraintLayout"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:background="@drawable/rect"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHeight_percent=".78"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      app:layout_constraintVertical_bias="1"
      tools:ignore="MissingConstraints">

      <androidx.viewpager2.widget.ViewPager2
         android:id="@+id/view_pager"
         android:layout_width="match_parent"
         android:layout_height="0dp"
         app:layout_constraintBottom_toBottomOf="parent"
         app:layout_constraintEnd_toEndOf="parent"
         app:layout_constraintHeight_percent=".7"
         app:layout_constraintStart_toStartOf="parent"
         app:layout_constraintTop_toBottomOf="@+id/tab_layout"
         app:layout_constraintVertical_bias="0" />

      <com.google.android.material.tabs.TabLayout
         android:id="@+id/tab_layout"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0"
        android:background="@drawable/rect"/>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/google"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:elevation="35dp"
        android:src="@drawable/google"
        app:backgroundTint="@color/white"
        app:fabSize="normal"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/view_pager"
        app:tint="@null" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/facebook"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:elevation="35dp"
        android:src="@drawable/facebook"
        app:backgroundTint="@color/white"
        app:fabSize="normal"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/google"
        app:layout_constraintHorizontal_bias="0.852"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/view_pager"
        app:layout_constraintVertical_bias="0.493"
        app:tint="@null" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/twitter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:elevation="35dp"
        android:src="@drawable/twitter"
        app:backgroundTint="@color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
```

```
            app:layout_constraintHorizontal_bias="0.14"
            app:layout_constraintStart_toEndOf="@+id/google"
            app:layout_constraintTop_toBottomOf="@id/view_pager"
            app:layout_constraintVertical_bias="0.493"
            app:tint="@null" />

    </androidx.constraintlayout.widget.ConstraintLayout>

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:src="@drawable/logo_login"
        app:layout_constraintBottom_toTopOf="@+id/constraintLayout"
        app:layout_constraintEnd_toEndOf="@+id/imageView"
        app:layout_constraintHorizontal_bias="0.138"
        app:layout_constraintStart_toStartOf="@+id/imageView"
        app:layout_constraintTop_toTopOf="@+id/imageView"
        app:layout_constraintVertical_bias="0.573" />

    <ImageView
        android:layout_width="200dp"
        android:layout_height="150dp"
        android:src="@drawable/text"
        app:layout_constraintBottom_toTopOf="@+id/constraintLayout"
        app:layout_constraintEnd_toEndOf="@+id/imageView"
        app:layout_constraintHorizontal_bias="0.924"
        app:layout_constraintStart_toStartOf="@+id/imageView"
        app:layout_constraintTop_toTopOf="@+id/imageView"
        app:layout_constraintVertical_bias="1.0" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

2) Signin.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <EditText
        android:id="@+id/username"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/user_icon"
        android:drawablePadding="10dp"
        android:hint="Mobile Number"
        android:textColor="@color/gray"
        android:maxLength="10"
        android:inputType="number"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.199"
        app:layout_constraintWidth_percent=".8" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/login_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/button_bg"
        android:text="Login"
        android:textColor="@color/white"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/password"
        app:layout_constraintVertical_bias="0.378"
        app:layout_constraintWidth_percent=".8" />

    <TextView
        android:id="@+id/forgetpass"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Forget Password"
        android:textSize="15dp"
        android:textColor="@color/gray"
        android:textStyle="bold"
        app:layout_constraintBottom_toTopOf="@+id/login_button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.871"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/password"
        app:layout_constraintVertical_bias="0.299" />

    <EditText
        android:id="@+id/password"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/lock_icon"
        android:drawablePadding="10dp"
        android:hint="Password"
        android:inputType="textPassword"
        android:textColor="@color/gray"
        android:paddingRight="10dp"
        android:paddingLeft="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/username"
        app:layout_constraintVertical_bias="0.042"
        app:layout_constraintWidth_percent=".8" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

3) Signup.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <EditText
        android:id="@+id/name"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/name_icon"
        android:drawablePadding="10dp"
        android:hint="Name"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        android:textColor="@color/gray"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.192"
        app:layout_constraintWidth_percent=".8" />

    <EditText
        android:id="@+id/password_signup"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/ic_baseline_email_24"
        android:drawablePadding="10dp"
        android:hint="Email"
        android:textColor="@color/gray"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/mobile_num"
        app:layout_constraintVertical_bias="0.046"
        app:layout_constraintWidth_percent=".8" />

    <EditText
        android:id="@+id/cfp_signup"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/lock_icon"
        android:drawablePadding="10dp"
        android:hint="Password"
        android:textColor="@color/gray"
        android:inputType="textPassword"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/password_signup"
        app:layout_constraintVertical_bias="0.046"
        app:layout_constraintWidth_percent=".8" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/signup_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/button_bg"
        android:text="Sign Up"
        android:textColor="@color/white"
        android:textSize="20sp"
        android:textStyle="bold"
```

```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/cfp_signup"
        app:layout_constraintVertical_bias="0.284"
        app:layout_constraintWidth_percent=".8" />

    <EditText
        android:id="@+id/mobile_num"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawableStart="@drawable/ic_baseline_smartphone_24"
        android:drawablePadding="10dp"
        android:hint="Mobile Number"
        android:maxLength="10"
        android:inputType="number"
        android:textColor="@color/gray"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/name"
        app:layout_constraintVertical_bias="0.046"
        app:layout_constraintWidth_percent=".8" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

4) OTPVerification.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".otpforgetpas">


    <ImageView
        android:id="@+id/imageView3"
        android:layout_width="match_parent"
        android:layout_height="350dp"
        android:layout_marginTop="20dp"
        android:src="@drawable/otp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0"
        tools:ignore="MissingConstraints" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="120px"
        android:text="Enter the OTP Sent to you"
        android:textAlignment="center"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView3"
        app:layout_constraintVertical_bias="0.0"
        tools:ignore="MissingConstraints" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="100px"
        android:text=""
        android:textAlignment="center"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView3"
        app:layout_constraintVertical_bias="0.085"
        tools:ignore="MissingConstraints" />


    <ProgressBar
        android:id="@+id/progressbar"
        android:layout_width="50dp"
        android:layout_height="50dp"
        app:layout_constraintBottom_toTopOf="@+id/verify_button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/otp"
        app:layout_constraintVertical_bias="0.793"
        tools:ignore="MissingConstraints" />

    <EditText
        android:id="@+id/otp"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="@drawable/custom_bg"
        android:drawablePadding="10dp"
        android:hint="OTP"
        android:paddingLeft="140dp"
        android:paddingTop="10dp"
        android:paddingRight="10dp"
        android:paddingBottom="10dp"
        android:textColor="@color/gray"
        android:textStyle="bold"
        app:layout_constraintBottom_toTopOf="@+id/progressbar"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView1"
        app:layout_constraintWidth_percent=".8"
        tools:ignore="MissingConstraints" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/verify_button"
        android:layout_width="250dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="120dp"
        android:background="@drawable/button_bg"
        android:text="Verify "
        android:textColor="@color/white"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.496"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView1"
        app:layout_constraintVertical_bias="1.0"
        app:layout_constraintWidth_percent=".8" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

5) Dashboard.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
tools:context=".MainActivity">

<ImageView
    android:id="@+id/imageView6"
    android:layout_width="match_parent"
    android:layout_height="180dp"
    android:src="@drawable/dashboard"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0"
    tools:ignore="MissingConstraints" />



<ImageView
    android:id="@+id/imageView9"
    android:layout_width="match_parent"
    android:layout_height="210dp"
    android:src="@drawable/page1"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView6"
    app:layout_constraintVertical_bias="0.029"
    tools:ignore="MissingConstraints" />

<ImageView
    android:id="@+id/imageView8"
    android:layout_width="match_parent"
    android:layout_height="70dp"
    android:src="@drawable/dashboard_bottom"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView6"
    app:layout_constraintVertical_bias="1.0"
    tools:ignore="MissingConstraints" />

<ImageView
    android:id="@+id/imageView7"
    android:layout_width="160dp"
    android:layout_height="80dp"
    android:src="@drawable/logotext2"
    app:layout_constraintBottom_toBottomOf="parent"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.537"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/imageView6"
        app:layout_constraintVertical_bias="0.022"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:layout_width="40dp"
        android:layout_height="38dp"
        android:src="@drawable/profile"
        app:layout_constraintBottom_toBottomOf="@+id/imageView6"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.043"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/imageView6"
        app:layout_constraintVertical_bias="0.112" />

    <androidx.appcompat.widget.SearchView
        android:id="@+id/searchView"
        android:layout_width="350dp"
        android:layout_height="45dp"
        android:background="@drawable/search"
        app:iconifiedByDefault="false"
        app:layout_constraintBottom_toBottomOf="@+id/imageView6"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.491"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.844"
        app:queryHint="Search Here ...."
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/tomato"
        android:layout_width="130dp"
        android:layout_height="125dp"
        android:src="@drawable/tomato"
        app:layout_constraintBottom_toTopOf="@+id/rice"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.647"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/chilli"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:src="@drawable/chilli"
        app:layout_constraintBottom_toTopOf="@+id/wheat"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.092"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/onion"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:src="@drawable/onion"
        app:layout_constraintBottom_toTopOf="@+id/watermelon"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.086"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/wheat"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:src="@drawable/wheat"
        app:layout_constraintBottom_toTopOf="@+id/imageView8"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.896"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/rice"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:src="@drawable/rice"
        app:layout_constraintBottom_toTopOf="@+id/imageView8"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.516"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.891"
        tools:ignore="MissingConstraints" />

    <ImageView
        android:id="@+id/watermelon"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:src="@drawable/watermelon"
        app:layout_constraintBottom_toTopOf="@+id/imageView8"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView9"
        app:layout_constraintVertical_bias="0.891"
        tools:ignore="MissingConstraints" />


    </androidx.constraintlayout.widget.ConstraintLayout>
```

6) Mainactivity.java

```
package com.example.myfarm;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.viewpager2.widget.ViewPager2;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;

import com.google.android.material.tabs.TabLayout;
import com.google.android.material.tabs.TabLayoutMediator;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class LoginActivity extends AppCompatActivity {


  ImageView goolge;
  ImageView fb;
  ImageView twitter;
  Fragmentadapter loginadapter;
  TabLayout tabLayout;
  ViewPager2 viewPager2;
  float v=0;

  private Fragmentadapter adapter;
  private  String[] titles = new String[] {"Login","Sign Up"};
```

```
   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_login);

   tabLayout=findViewById(R.id.tab_layout);
   viewPager2=findViewById(R.id.view_pager);

   loginadapter = new Fragmentadapter(this);

   viewPager2.setAdapter(loginadapter);
   new TabLayoutMediator(tabLayout,viewPager2,((tab, position) ->
tab.setText(titles[position]))).attach();

       goolge = findViewById(R.id.google);
       fb = findViewById(R.id.facebook);
       twitter = findViewById(R.id.twitter);

       goolge.setTranslationY(300);
       fb.setTranslationY(300);
       twitter.setTranslationY(300);

       goolge.setAlpha(v);
       fb.setAlpha(v);
       twitter.setAlpha(v);

       goolge.animate().translationY(0).alpha(1).setDuration(1000).setStartDelay(400).start();
       fb.animate().translationY(0).alpha(1).setDuration(1000).setStartDelay(600).start();
       twitter.animate().translationY(0).alpha(1).setDuration(1000).setStartDelay(800).start();
   }
}
```

7) Loginactivity.java

```
package com.example.myfarm;

import static android.widget.Toast.makeText;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.activity.result.contract.ActivityResultContracts;
import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class login_tab extends Fragment{
    EditText username;
    EditText password;
    TextView forgetpass;
    Button login_button;
    float v=0;
    private FirebaseAuth mAuth;
    DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReferenceFromUrl("https://myfarm-cc36d-default-
rtdb.firebaseio.com/");


@Override
    public View onCreateView(LayoutInflater inflater,ViewGroup container, Bundle savedInstanceState) {

        ViewGroup root = (ViewGroup) inflater.inflate(R.layout.login_tab, container,false);

    username = root.findViewById(R.id.username);
    password = root.findViewById(R.id.password);
    forgetpass = root.findViewById(R.id.forgetpass);
    login_button = root.findViewById(R.id.login_button);

    username.setTranslationX(300);
    password.setTranslationX(300);
    forgetpass.setTranslationX(300);
    login_button.setTranslationX(300);

    username.setAlpha(v);
    password.setAlpha(v);
    forgetpass.setAlpha(v);
    login_button.setAlpha(v);

    username.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(400).start();
    password.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(600).start();
    forgetpass.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(800).start();
    login_button.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(1000).start();

    login_button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String mobile = username.getText().toString();
            String pass = password.getText().toString();

            String noWhiteSpace = "\\A\\w{4,20}\\z";
            if (mobile.isEmpty()) {
                username.setError("Mobile Number is required!");
                username.requestFocus();
                return;
            } else if (mobile.length() > 10) {
                username.setError("Mobile Number should be 10 digits");
                username.requestFocus();
                return;
            } else if (mobile.length() < 10) {
                username.setError("Mobile Number should be 10 digits");
                username.requestFocus();
                return;
            } else if (!mobile.matches(noWhiteSpace)) {
                username.setError("White Spaces are not allowed");
                return;
            }
            if (pass.isEmpty()) {
                password.setError("Field cannot be empty");
                return;
            }else
            {
              databaseReference.child("users").addListenerForSingleValueEvent(new ValueEventListener() {
                  @Override
                  public void onDataChange(@NonNull DataSnapshot snapshot) {
                      if(snapshot.hasChild(mobile)){
                           String getpass = snapshot.child(mobile).child("Password").getValue(String.class);
                          if(getpass.equals(pass)){
                      Toast.makeText(getActivity(), "Successfully Logged in", Toast.LENGTH_SHORT).show();
                              startActivity(new Intent(getActivity(),MainActivity.class));
                          }else
                          {
                      Toast.makeText(getActivity(), "Wrong Mobile Number or Password",
Toast.LENGTH_SHORT).show();
                          }
                      }
                  }
                  @Override
                  public void onCancelled(@NonNull DatabaseError error) {

                  }
              });
            }
        }
    });
    forgetpass.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent(getActivity(), Forgetpassword.class);
            startActivity(intent);
        }
    });
    return root;
    }
}
```

8) Signupactivity.java

```
package com.example.myfarm;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.google.firebase.FirebaseException;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.PhoneAuthCredential;
import com.google.firebase.auth.PhoneAuthProvider;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.lang.ref.Reference;
import java.util.concurrent.TimeUnit;
import java.util.prefs.PreferenceChangeEvent;

public class Signup_tab extends Fragment {

    DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReferenceFromUrl("https://myfarm-cc36d-default-
rtdb.firebaseio.com/");
    FirebaseAuth mAuth;
    EditText name;
    EditText mobile_numb;
    EditText password_signup;
    EditText cfp_signup;
    Button signup_button;
    float v=0;



    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
{
        ViewGroup root = (ViewGroup) inflater.inflate(R.layout.signup_tab, container, false);

        name = root.findViewById(R.id.name);
        mobile_numb = root.findViewById(R.id.mobile_num);
        password_signup = root.findViewById(R.id.password_signup);
        cfp_signup = root.findViewById(R.id.cfp_signup);
        signup_button = root.findViewById(R.id.signup_button);

        name.setTranslationX(300);
        mobile_numb.setTranslationX(300);
        password_signup.setTranslationX(300);
        cfp_signup.setTranslationX(300);
        signup_button.setTranslationX(300);

        name.setAlpha(v);
        mobile_numb.setAlpha(v);
        password_signup.setAlpha(v);
        cfp_signup.setAlpha(v);
        signup_button.setAlpha(v);

        name.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(200).start();
        mobile_numb.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(400).start();
        password_signup.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(600).start();
        cfp_signup.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(800).start();
        signup_button.animate().translationX(0).alpha(1).setDuration(1000).setStartDelay(1000).start();


        signup_button.setOnClickListener(new View.OnClickListener(){


            @Override
            public void onClick(View view) {
                String nm = name.getText().toString();
                String mobile = mobile_numb.getText().toString();
                String pas = password_signup.getText().toString();
                String cnfpas = cfp_signup.getText().toString();

                if (nm.isEmpty()) {
```

```
            name.setError("Name is required!");
            name.requestFocus();
            return;
        }
        String noWhiteSpace = "\\A\\w{4,20}\\z";
        if (mobile.isEmpty()) {
            mobile_numb.setError("Mobile Number is required!");
            mobile_numb.requestFocus();
            return;
        } else if (mobile.length() > 10) {
            mobile_numb.setError("Mobile Number should be 10 digits");
            mobile_numb.requestFocus();
            return;
        } else if (mobile.length() < 10) {
            mobile_numb.setError("Mobile Number should be 10 digits");
            mobile_numb.requestFocus();
            return;
        } else if (!mobile.matches(noWhiteSpace)) {
            mobile_numb.setError("White Spaces are not allowed");
            return;
        }
        String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";

        if (pas.isEmpty()) {
            password_signup.setError("Field cannot be empty");
            return;
        } else if (!pas.matches(emailPattern)) {
            password_signup.setError("Invalid email address");
            return;
        }
        String passwordVal = "^" +
                "(?=.*[0-9])" +         //at least 1 digit
                "(?=.*[a-z])" +         //at least 1 lower case letter
                "(?=.*[A-Z])" +         //at least 1 upper case letter
                "(?=.*[a-zA-Z])" +      //any letter
                "(?=.*[@#$%^&+=])" +    //at least 1 special character
                "(?=\\S+$)" +           //no white spaces
                ".{4,}" +               //at least 4 characters
                "$";

        if (cnfpas.isEmpty()) {
            cfp_signup.setError("Field cannot be empty");
            return;
        } else if (!cnfpas.matches(passwordVal)) {
            cfp_signup.setError("Password should consist of atleast 1 uppercase, 1 lower case and 1 
special character");
            return;
        }
```

```
                databaseReference.child("users").addListenerForSingleValueEvent(new ValueEventListener()
{
                    @Override
                    public void onDataChange(@NonNull DataSnapshot snapshot) {
                        if (snapshot.hasChild(mobile)) {
                            Toast.makeText(getActivity(), "Mobile Number already exist",
Toast.LENGTH_SHORT).show();
                        } else {
                            Intent intent = new Intent(getActivity().getApplicationContext(), OTPverification.class);
                            intent.putExtra("mobile", mobile);
                            intent.putExtra("name", nm);
                            intent.putExtra("email", pas);
                            intent.putExtra("password", cnfpas);
                            startActivity(intent);
                        }
                    }
                    @Override
                    public void onCancelled(@NonNull DatabaseError error) {

                    }
                });
            }
        });
        return root;
    }
}
```

9) Fragment Adapter.java

```
package com.example.myfarm;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentActivity;
import androidx.viewpager2.adapter.FragmentStateAdapter;

public class Fragmentadapter extends FragmentStateAdapter {

    private  String[] titles = new String[] {"Login","Sign Up"};

    public Fragmentadapter(@NonNull FragmentActivity fragmentActivity) {
        super(fragmentActivity);
    }

    @NonNull
    @Override
    public Fragment createFragment(int position) {
        switch (position)
        {
            case 0:
                return new login_tab();
            case 1:
                return new Signup_tab();
        }
        return new login_tab();
    }

    @Override
    public int getItemCount() {

        return 2;
    }
}
```

10) Dashboard.java

```
package com.example.myfarm;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    ImageView chilli,onion,tomato,wheat,rice,watermelon;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        chilli=findViewById(R.id.chilli);
        onion=findViewById(R.id.onion);
        tomato=findViewById(R.id.tomato);
        wheat=findViewById(R.id.wheat);
        rice=findViewById(R.id.rice);
        watermelon=findViewById(R.id.watermelon);
        chilli.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, chillipage.class));
                finish();
            }
        });

        onion.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, onion.class));
                finish();
            }
        });

        tomato.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, tomato.class));
                finish();
            }
        });
        wheat.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, wheat.class));
                finish();
            }
        });
        rice.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, rice.class));
                finish();
            }
        });
        watermelon.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, watermelon.class));
                finish();
            }
        });

    }
}
```

11) Chilli page.java

```
package com.example.myfarm;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
```

```
public class chillipage extends AppCompatActivity {

    ImageView arrow;
    Button
read1,read2,read3,read4,read5,read6,read7,read8,read9,read10,read11,read12,read13,read14,read15,read1
6,read17,read18,read19,read20,
            read21,read22,read23,read24,read25,read26,read27,read28,read29,read30,read31,read32,read33;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_chillipage);

        arrow=findViewById(R.id.arrow);

        arrow.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, MainActivity.class));
                finish();
            }
        });

        read1=findViewById(R.id.read1);

        read1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli1.class));
                finish();
            }
        });
        read2=findViewById(R.id.read2);

        read2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli2.class));
                finish();
            }
        });read3=findViewById(R.id.read3);

        read3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli3.class));
                finish();
            }
        });read4=findViewById(R.id.read4);

        read4.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli4.class));
            finish();
        }
    });read5=findViewById(R.id.read5);

    read5.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli5.class));
            finish();
        }
    });read6=findViewById(R.id.read6);

    read6.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli7.class));
            finish();
        }
    });read7=findViewById(R.id.read7);

    read7.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli8.class));
            finish();
        }
    });read8=findViewById(R.id.read8);

    read8.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli9.class));
            finish();
        }
    });read9=findViewById(R.id.read9);

    read9.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli10.class));
            finish();
        }
    });read10=findViewById(R.id.read10);

    read10.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli11.class));
            finish();
        }
    });
    read11=findViewById(R.id.read11);

    read11.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli12.class));
            finish();
        }
    });
    read12=findViewById(R.id.read12);

    read12.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli13.class));
            finish();
        }
    });read13=findViewById(R.id.read13);

    read13.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli14.class));
            finish();
        }
    });read14=findViewById(R.id.read14);

    read14.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli15.class));
            finish();
        }
    });read15=findViewById(R.id.read15);

    read15.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli16.class));
            finish();
        }
    });read16=findViewById(R.id.read16);

    read16.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli17.class));
            finish();
        }
    });read17=findViewById(R.id.read17);

    read17.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli18.class));
            finish();
        }
    });read18=findViewById(R.id.read18);

    read18.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli19.class));
            finish();
        }
    });read19=findViewById(R.id.read19);

    read19.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli20.class));
            finish();
        }
    });read20=findViewById(R.id.read20);

    read20.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli21.class));
            finish();
        }
    });
    read21=findViewById(R.id.read21);

    read21.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli22.class));
            finish();
        }
    });
    read22=findViewById(R.id.read22);

    read22.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(chillipage.this, readchilli23.class));
            finish();
        }
```

```
});read23=findViewById(R.id.read23);

read23.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli24.class));
        finish();
    }
});read24=findViewById(R.id.read24);

read24.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli25.class));
        finish();
    }
});read25=findViewById(R.id.read25);

read25.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli26.class));
        finish();
    }
});read26=findViewById(R.id.read26);

read26.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli27.class));
        finish();
    }
});read27=findViewById(R.id.read27);

read27.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli28.class));
        finish();
    }
});read28=findViewById(R.id.read28);

read28.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(chillipage.this, readchilli29.class));
        finish();
    }
});read29=findViewById(R.id.read29);

read29.setOnClickListener(new View.OnClickListener() {
```

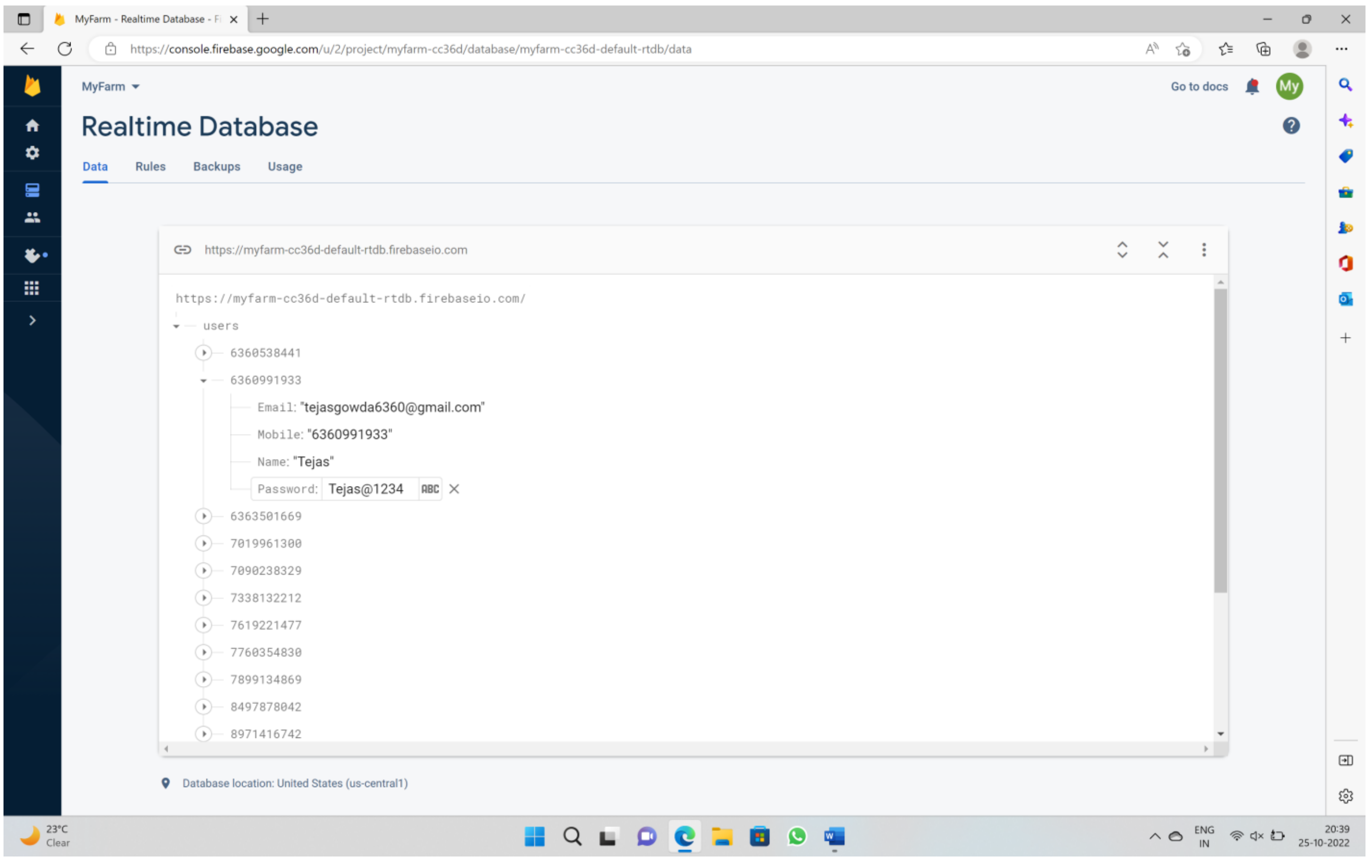
```
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli30.class));
                finish();
            }
        });read30=findViewById(R.id.read30);

        read30.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli31.class));
                finish();
            }
        });
        read31=findViewById(R.id.read31);

        read31.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli32.class));
                finish();
            }
        });
        read32=findViewById(R.id.read32);

        read32.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli33.class));
                finish();
            }
        });
        read33=findViewById(R.id.read33);

        read33.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(chillipage.this, readchilli34.class));
                finish();
            }
        });
    }
}
   }
}
```

12) Tomato page.java

```
package com.example.myfarm;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

public class tomato extends AppCompatActivity {

    ImageView arrow;
    Button
tomatoread1,tomatoread2,tomatoread3,tomatoread4,tomatoread5,tomatoread6,tomatoread7,tomatoread8,t
omatoread9,tomatoread10,tomatoread11,

tomatoread12,tomatoread13,tomatoread14,tomatoread15,tomatoread16,tomatoread17,tomatoread18,tomat
oread19,tomatoread20,tomatoread21,tomatoread22,

tomatoread23,tomatoread24,tomatoread25,tomatoread26,tomatoread27,tomatoread28,tomatoread29,tomat
oread30,tomatoread31,tomatoread32,
    tomatoread33,tomatoread34,tomatoread35,tomatoread36;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_tomato);

        arrow=findViewById(R.id.arrow);

        arrow.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, MainActivity.class));
                finish();
            }
        });
        tomatoread1=findViewById(R.id.tomatoread1);

        tomatoread1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato1.class));
                finish();
            }
        });
        tomatoread2=findViewById(R.id.tomatoread2);

        tomatoread2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato2.class));
                finish();
            }
        });
        tomatoread3=findViewById(R.id.tomatoread3);

        tomatoread3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato3.class));
                finish();
            }
        });
        tomatoread4=findViewById(R.id.tomatoread4);

        tomatoread4.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato4.class));
                finish();
            }
        });
        tomatoread5=findViewById(R.id.tomatoread5);

        tomatoread5.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato5.class));
                finish();
            }
        });
        tomatoread6=findViewById(R.id.tomatoread6);

        tomatoread6.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato6.class));
                finish();
            }
        });
        tomatoread7=findViewById(R.id.tomatoread7);

        tomatoread7.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato7.class));
                finish();
            }
        });
        tomatoread8=findViewById(R.id.tomatoread8);

        tomatoread8.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato8.class));
                finish();
            }
        });
        tomatoread9=findViewById(R.id.tomatoread9);

        tomatoread9.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato9.class));
                finish();
            }
        });
        tomatoread10=findViewById(R.id.tomatoread10);

        tomatoread10.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato10.class));
                finish();
            }
        });
        tomatoread11=findViewById(R.id.tomatoread11);

        tomatoread11.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato11.class));
                finish();
            }
        });
        tomatoread12=findViewById(R.id.tomatoread12);

        tomatoread12.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato12.class));
                finish();
            }
        });
        tomatoread13=findViewById(R.id.tomatoread13);

        tomatoread12.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(tomato.this, readtomato13.class));
                finish();
            }
        })
```

DATABASE

Signup Database



```
https://myfarm-cc36d-default-rtdb.firebaseio.com/
users
    6360538441
    6360991933
        Email: "tejasgowda6360@gmail.com"
        Mobile: "6360991933"
        Name: "Tejas"
        Password: Tejas@1234
    6363501669
    7019961300
    7090238329
    7338132212
    7619221477
    7760354830
    7899134869
    8497878042
    8971416742
```

Database location: United States (us-central1)

Coding

```
package com.example.myfarm;
import org.w3c.dom.Text;

import java.io.CharArrayWriter;
import java.util.concurrent.TimeUnit;

public class OTPverification extends AppCompatActivity {

    DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReferenceFromUrl("https://myfarm-cc36d-default-
rtdb.firebaseio.com/");
    String verificationCodeBySystem;
    ProgressBar progressBar;
    EditText otp;
    FirebaseAuth mAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_otpverification);
```

```
         progressBar = findViewById(R.id.progressbar);
        otp = findViewById(R.id.otp);
        Button verifybutton = findViewById(R.id.verify_button);

        progressBar.setVisibility(View.GONE);

        TextView textView = findViewById(R.id.textView1);
        textView.setText(String.format(
                "+91-%s", getIntent().getStringExtra("mobile")
        ));

        String mobile = getIntent().getStringExtra("mobile");
        String nm = getIntent().getStringExtra("name");
        String pas = getIntent().getStringExtra("email");
        String cnfpas = getIntent().getStringExtra("password");

        sendverificationcodetouser(mobile);

        verifybutton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String code = otp.getText().toString();
                if (code.isEmpty() || code.length()<6){
                 otp.setError("Wrong OTP....");
                 otp.requestFocus();
                 return;
                }
                progressBar.setVisibility(View.VISIBLE);
                verifiCode(code);
            }
        });

    }

    private void sendverificationcodetouser(String mobile) {
        PhoneAuthOptions options = PhoneAuthOptions.newBuilder()
                        .setPhoneNumber("+91" +mobile)       // Phone number to verify
                        .setTimeout(60L, TimeUnit.SECONDS) // Timeout and unit
                        .setActivity(OTPverification.this)                 // Activity (for callback binding)
                        .setCallbacks(mCallbacks)          // OnVerificationStateChangedCallbacks
                        .build();
        PhoneAuthProvider.verifyPhoneNumber(options);
        // [END start_phone_auth]
    }

    private PhoneAuthProvider.OnVerificationStateChangedCallbacks mCallbacks = new
PhoneAuthProvider.OnVerificationStateChangedCallbacks() {
        @Override
        public void onCodeSent(@NonNull String s, @NonNull PhoneAuthProvider.ForceResendingToken
forceResendingToken) {
            super.onCodeSent(s, forceResendingToken);
            verificationCodeBySystem=s;
        }

        @Override
        public void onVerificationCompleted(@NonNull PhoneAuthCredential phoneAuthCredential) {
            String code = phoneAuthCredential.getSmsCode();
           if(code!=null){
               progressBar.setVisibility(View.VISIBLE);
               verifiCode(code);
           }
        }

        @Override
        public void onVerificationFailed(@NonNull FirebaseException e) {
Toast.makeText(OTPverification.this,e.getMessage(),Toast.LENGTH_SHORT).show();
        }
    };

    private void verifiCode(String codebyuser){
    PhoneAuthCredential credential =
PhoneAuthProvider.getCredential(verificationCodeBySystem,codebyuser);
    signInTheUserByCredentials(credential);
}
    private void signInTheUserByCredentials(PhoneAuthCredential credential){
    FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
    firebaseAuth.signInWithCredential(credential)
            .addOnCompleteListener(OTPverification.this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
               if (task.isSuccessful()){
                   Intent intent = new Intent(getApplicationContext(),MainActivity.class);
                   intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
                   startActivity(intent);
                   databaseReference.child("users").addListenerForSingleValueEvent(new
ValueEventListener() {
                       @Override
                       public void onDataChange(@NonNull DataSnapshot snapshot) {
                           String mobile = getIntent().getStringExtra("mobile");
                               String nm = getIntent().getStringExtra("name");
                               String pas = getIntent().getStringExtra("email");
                               String cnfpas = getIntent().getStringExtra("password");

                               databaseReference.child("users").child(mobile).child("Name").setValue(nm);
                               databaseReference.child("users").child(mobile).child("Mobile").setValue(mobile);
                               databaseReference.child("users").child(mobile).child("Email").setValue(pas);
                               databaseReference.child("users").child(mobile).child("Password").setValue(cnfpas);

                               Toast.makeText(OTPverification.this, "User Register Successfully",
Toast.LENGTH_SHORT).show();       }
} }
```

TESTING

Introduction

Software testing is the act of examining the artifacts and the behaviour of the software under test by validation and verification. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.

Types of Testing

Unit tests

Unit tests are very low level and close to the source of an application. They consist in testing individual methods and functions of the classes, components, or modules used by your software. Unit tests are generally quite cheap to automate and can run very quickly by a continuous integration server.

Integration tests

Integration tests verify that different modules or services used by your application work well together. For example, it can be testing the interaction with the database or making sure that microservices work together as expected. These types of tests are more expensive to run as they require multiple parts of the application to be up and running.

Functional tests

Functional tests focus on the business requirements of an application. They only verify the output of an action and do not check the intermediate states of the system when performing that action.

There is sometimes a confusion between integration tests and functional tests as they both require multiple components to interact with each other. The difference is that an integration test may simply verify that you can query the database while a functional test would expect to get a specific value from the database as defined by the product requirements.

End-to-end tests

End-to-end testing replicates a user behavior with the software in a complete application environment. It verifies that various user flows work as expected and can be as simple as loading a web page or logging in or much more complex scenarios verifying email notifications, online payments, etc...

Acceptance testing

Acceptance tests are formal tests that verify if a system satisfies business requirements. They require the entire application to be running while testing and focus on replicating user behaviors. But they can also go further and measure the performance of the system and reject changes if certain goals are not met.

Performance testing

Performance tests evaluate how a system performs under a particular workload. These tests help to measure the reliability, speed, scalability, and responsiveness of an application. For instance, a performance test can observe response times when executing a high number of requests, or determine how a system behaves with a significant amount of data. It can determine if an application meets performance requirements, locate bottlenecks, measure stability during peak traffic, and more.

Smoke Testing

Smoke tests are basic tests that check the basic functionality of an application. They are meant to be quick to execute, and their goal is to give you the assurance that the major features of your system are working as expected.

Types of Manual Testing

1. White Box Testing

White box testing involves testing the product's underlying structure, architecture, and code to validate input-output flow and enhance design, usability, and security.

2. Black Box Testing

Black box testing involves testing against a system where the code and paths are invisible.

Test Report

Test Report is a document which contains a summary of all test activities and final test results of a testing project. Test report is an assessment of how well the Testing is performed. Based on the test report, stakeholders can evaluate the quality of the tested product and make a decision on the software release.

Test Cycle System Test

EXECUTED	Passed			130	
	Failed			0	
	(Total) TESTS EXECUTED (PASSED + FAILED)				130
PENDING					0
IN PROGRESS					0
BLOCKED					0
(Sub – Total) TEST PLANNED					130
(PENDING+IN PROGRESS+BLOCKED +TEST EXECUTED)					

Functions	Description	%TCs Executed	%TCs Passed	TCs Pending	Priority	Remarks
User Registration	Check new user added	100%	100%	0	High	
Login Registered user	Check user exists or not	100%	100%	0	High	
Chilli Page	How to grow the chilli	100%	100%	0	High	
Tomato Page	How to grow The tomato	100%	100%	0	High	
Onion Page	How to grow The Onion	100%	100%	0	High	
Rice Page	How to grow The rice	100%	100%	0	High	
Wheat Page	How to grow The wheat	100%	100%	0	High	
Watermelon Page	How to grow The watermelon	100%	100%	0	High	

IMPLEMENTATION

Implementation

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigating of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Technologies used

Android Studio

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems
- ProGuard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Firebase

- Realtime Database: The Firebase Realtime Database is a cloud-based NoSQL database that manages your data at the blazing speed of milliseconds. In simplest term, it can be considered as a big JSON file.

- Cloud Firestore: The cloud Firestore is a NoSQL document database that provides services like store, sync, and query through the application on a global scale. It stores data in the form of objects also known as Documents.

- Authentication: Firebase Authentication service provides easy to use UI libraries and SDKs to authenticate users to your app. It reduces the manpower and effort required to develop and maintain the user authentication service.

- Remote Config: The remote configuration service helps in publishing updates to the user immediately. The changes can range from changing components of the UI to changing the behavior of the applications. These are often used while publishing seasonal offers and contents to the application that has a limited life.

- Hosting: Firebase provides hosting of applications with speed and security. It can be used to host Stati or Dynamic websites and microservices. It has the capability of hosting an application with a single command.

FUTURE ENHANCEMENT

Future Enhancement

- Buying and selling of seeds and fertilizers in our app to the farmers for there doorstep
- Provides a chat bot support in our app which helps the farmer to clarify there doubt's
- Plant expert to visit the farmers land for there help the farmer can book an appointment in our app in future.
- Climate alert to the farmer for there safety to protect there crops from heavy rain
- All the local Languages it will helps the farmer to understand
- Audiobook

CONCLUSION

Conclusion

The project entitled “ **MY FARM GUIDE** ” was completed on time with total satisfaction after testing with possible sample data. The performance was found to be efficient and error free. This is a user-friendly packaged application which is very easy to access and understand. Anyone with Knowledge of Computers will find it very easy to use this software and perform various operations on it. In this project, first an attempt has been made to find the need of the system. To fulfil the needs, a detailed study had been conducted to find the various requirements of the system. This particular system has been designed in an attractive manner, so that even a user with minimum knowledge can be able to operate the system easily.

This software combines the best of both the world i.e.; programming language (XML), and (JAVA) with Database (FIREBASE) providing easy accessibility and security. It was developed to benefit the organizations and the customers. Finally, the system was tested with real data and everything worked successfully. Thus, the system has fulfilled all the objectives identified and is able to replace the existing system.

Goals Achieved

- Protect the Crops from Diseases
- Tells about the trending technologies to Farmers
- User Friendly screens to help non-technical users
- Provide Guidelines
- Provides Information to Farmer

BIBLIOGRAPHY

References

Books

1. Android Crash Course: A Hands-On, Project-Based Introduction to Programming (2nd Edition) By Eric Matthes
2. Learn Android the Hard Way: 3rd Edition by *Zed A. Shaw*
3. Android for APIs: Build web APIs by William S Vincent

Web Forums

1. https://www.geeksforgeeks.org/
2. https://www.stackoverflow.com/