**CONTENTS**

**Topic Page no.**

1. Title page i
2. Certificate ii - iii
3. Declaration iv
4. Acknowledgment v
5. Abstraction vi
6. INTRODUCTION 2-4
   1. Object
   2. Problem Definition
   3. Project Overview/Specification
   4. Introduction
7. LITERATURE SURVEY 5-6
   1. Existing System
   2. Proposed System
8. SYSTEM ANALYSIS AND DESIGN 7-20
   1. Requirement Specification
   2. Flowchart
   3. Design and Test Step
   4. Algorithms and Pseudo Code
   5. Hardware Specification
   6. Software Specification
9. IMPLEMENTATION 21-29
10. TESTING PROCESS 30-32

1. RESULTS/OUTPUT 33-37

1. CONCLUSIONS/FUTURE WORK 38-40

**CHAPTER 1**

**INTRODUCTION**

* 1. Object
  2. Problem Definition
  3. Project Overview/Specification
  4. Introduction

1. **INTRODUCTION**
   1. **Object**

* By this project, our purpose to make farmer more independent and also give a choice to compare the fertilizer & pesticide of different companies, and alsocompare the cost of it.
  1. **Problem Definition**
* Dependent on shopkeeper
* Not have a choice to compare the pesticides and fertilizer.
* Waste of time to go the shop.
  1. **Problem Solution**
* They easily search the fertilizer & pesticide which the require .
* They have more companies to compare qualities , price and quantity of fertilizer & pesticide.
* They save their time by order the fertilizer & pesticide by there phone only.
  1. **Introduction**
* KISHAN MITRA Application “ is an application for Smart phones use in Android Operating system”.
* This application allow you to purchases the fertilizer & pesticide .
* This application is design specially for farmers .
* This application gives you complete knowledge of crops and fertilizer & pesticide .
* This application make the direct connection between the farmers and companies.

**CAHPTER 2**

**LITERATURE SURVEY**

* 1. Existing System
  2. Proposed System

1. **LITERATURE SURVEY**

**2.1.** **Existing System**

In the existing system, the manual process, receiving information of pesticide and fertiliser from shopkeeper, other farmer . These record share entered in manual process. In this process will take long time, and difficult to get proper quality of fertiliser and pesticide. Many farmer have to suffer lot in to get proper quality of crop because they trust on shopkeeper about the quality of pesticide and fertilise. Farmer have to pay more cost to perches the fertiliser and pesticide because they have more shop to compare to pesticide.

**2.2.** **Proposed System**

The main objective of the existing system is to provide a user-friendly interface the system, which is processed, now computerizes all the details that are maintain. Once the details fed into the computer is no need for various persons to deal with separate sections Only a single person is enough to maintain all there reports. The security can also be given as per the requirement of the user.

In order to make the software dynamic and more interactive we have tried to include a database link tour to our software. Owner can register their selves easily and access there

An academic details. This the software becomes more interactive and accessible to user.

* large volume of data can be stored with case.
* Maintenance of file is flexible.
* Records stored are update now and then.
* Stored data and procedures can be easily edited.
* Reports can be generated with case.
* Accurate calculations are made.

**CHAPTER 3**

**SYSTEM ANALYSIS AND DESIGN**

**3.1. Required Specification**

**3.2. Flowchart**

**3.3. Designing and Test Steps**

**3.4. Hardware Specification**

**3.5. Software Specification**

**System Analysis:**

System analysis is the process of gathering and interpreting facts, diagnosing problem and using the information to recommended improvements on the system. System analysis is problem solving activity that requires intensive communication between the system user and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analysed the system analyst plays the role of an interrogator and dwells deep into the working of the present system.

**Elements of the analysis model**

1. **Scenario based element**

* This type of element represents user point of view.
* Scenario based elements are use case diagram, user stories.

1. **Class based elements**

* The object of this type of element manipulated by the system.
* It defines the object, attributes and relationship. •The collaboration is occurring between the classes.
* Class based elements are the class diagram, collaboration diagram.

1. **Behavioural elements**

* Behavioural elements represent state of the system and how it is changed by the external events.
* The behavioural elements are sequenced diagram, state diagram

1. **Flow oriented elements**

* An information flows through a computer-based system it gets transformed. It shows how the data objects are transformed while they flow between the various system
* functions. The flow elements are data flow diagram, control flow diagram.

**Analysis Rules of Thumbs**

The Rules of thumb that must be followed while creating the analysis model.

**The rules are as follows:**

* The model focuses on the requirements in the business domain. The level of abstraction must be high Le. there is no need to give details
* Every element in the model helps in understanding the software requirement and focus on the information, function and behaviour of the system.
* The consideration of infrastructure and non-functional model delayed in the design. For example: The database is required for a system, but the classes, functions and behaviour of the database are not initially required. If these are initially considered then there is a delay in the designing.
* Throughout the system minimum coupling is required. The interconnections between the modules is known as "coupling
* The analysis model gives value to all the people related to model.
* The model should be simple as possible. Because simple model always helps in easy understanding of the requirement.

**Software analytics** is the measurement, collection, analysis and reporting of software data for purpose of understanding and optimizing software usage. However, software analytics is not just a process for measuring software traffic but can be used as a tool for business and market research and to assess and improve the effectiveness of a software, Software analytics applications can also help companies measure the results of traditional print or broadcast advertising companies. It helps one to estimate how traffic to a software changes after the launch of a new advertising campaign. Software analytics provides information about the number of visitors to a software and the number of page views. It helps traffic popularity trends which is useful for market research.

**Requirement Specification :**

**Role of a System analyst:**

The analyst starts requirements gathering and analysis by collecting all information from the customer which could be used to develop the requirements of the system. He then anal the collected information to obtain a clear and through understanding of the product to be developed, with a view to removing all ambiguities and inconsistencies from the initial customer perception of the problem. The following basic questions, pertaining to the project should be clearly understood by the analyst in order to obtain a good grasp of the problem.

After the analyst has understood the exact customer requirements, he proceeds to identify and resolve the various requirements problem. The most important requirements problems that the analyst has to identify and eliminate are the problems of anomalies, inconsistencies, and incompleteness. When the analyst detects any inconsistencies, anomalies or incompleteness in the gathered requirements, he resolves them by carrying our further discussion with the end-users and the customers.

**Parts of a SRS document:**

**The important parts of SRS document are:**

* Functional requirements of the system.
* Non-functional requirements of the system.
* Goals of implementation.

**Factional requirements:**

The functional requirements part discusses the functionalities required from the system. The system is considered to perform a set of high-level functions.

**Non- functional requirements:**

Non-functional requirements deal with the characteristics of the system which cannot be expressed as function such as the maintainability of the system, portability of the system, usability of the system, etc. Non- functional requirements may include: # reliability issues, # accuracy of results, # constraints on the system implementation, etc.

Identifying functional requirements from a problem description.

The high-level functional requirements often need to be identified either from an informal problem description document or from a conceptual understanding of the problem. Each high-level requirement ) characterizes a ways of system usage by some user to perform some meaningful piece of work.

**Documenting functional requirements**

For documenting the functional requirements, we need to specify the set of functionalities supported by the system. A function can be specified by identifying the state at which the data is to be input data domain, the output data domain, and the type of processing to be carried on the input data to obtain the output data. Let us first try to document the withdraw-cash function of an ATM (Automated Teller Machine) system. The withdraw-cash is a high-level requirement. It has several sub-requirements corresponding to the different user interaction. These different interaction sequences capture the different scenarios.

**3.2 Flow Chart**

**3.2.1 Data Flow Diagram**

Data Flow Diagram (DFD)is a graphical representation of the flow of the data through an information system. Modelling is its process aspects. DFD can also be used for the visualization data processing A DFD shows that what kind of data will be store. As its name includes its focus is on the flow of information flow and functional detail. Four simple notations are used to complete a DFD.

**3.2.2. Data Flow Diagram Notations**

There are essentially two different types of notations for data flow diagram defining different visual representation for process., data stores, data flow and external entities Your don and Code type.

**3.2.3. Process -**

A process transforms incoming data flow into outgoing data flow

Fig: 3.1

**3.2.4 Store Data -**

Store are repositories of data in the system. They are some time also referred to as files.

DATA FLOW

Fig: 3.2

**3.2.5 Data Flow -**

Data flows are pipelines through packets of information flow. Level the arrows with the name of the data that moves through it.

DATA FLOW

Fig: 3.3

**3.2.6 External Entity -**

External entities are objects outside the system, with which the system communication.

'External entities are sources and destination of the system inputs and outputs.

Fig: 3.4

**USE CASE DIAGRAM FOR ADMIN**

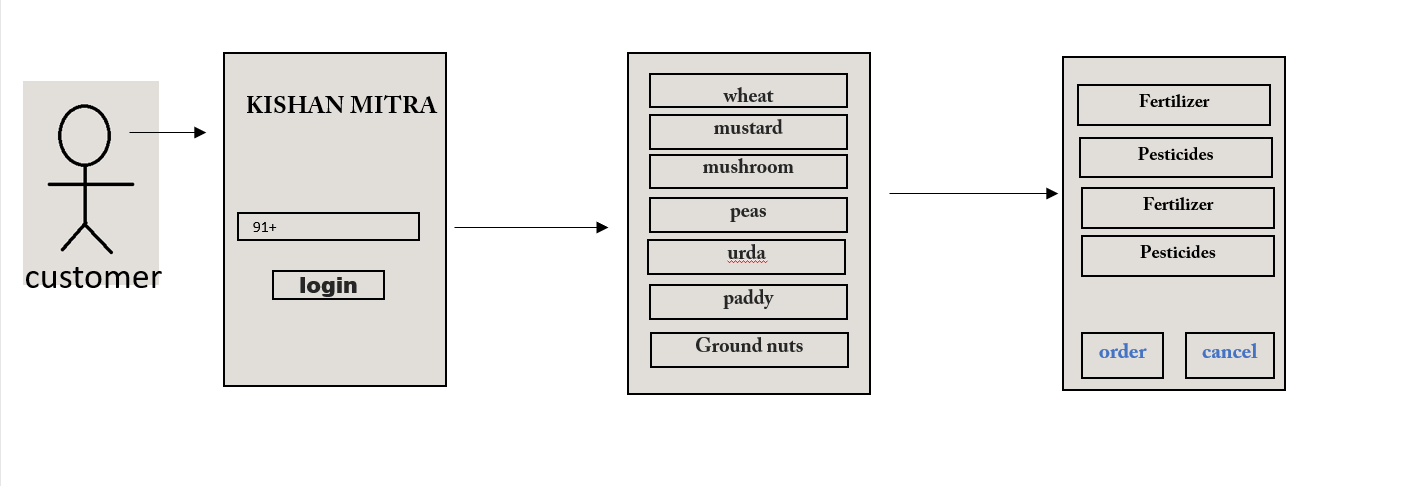
****

Fig: 3.5

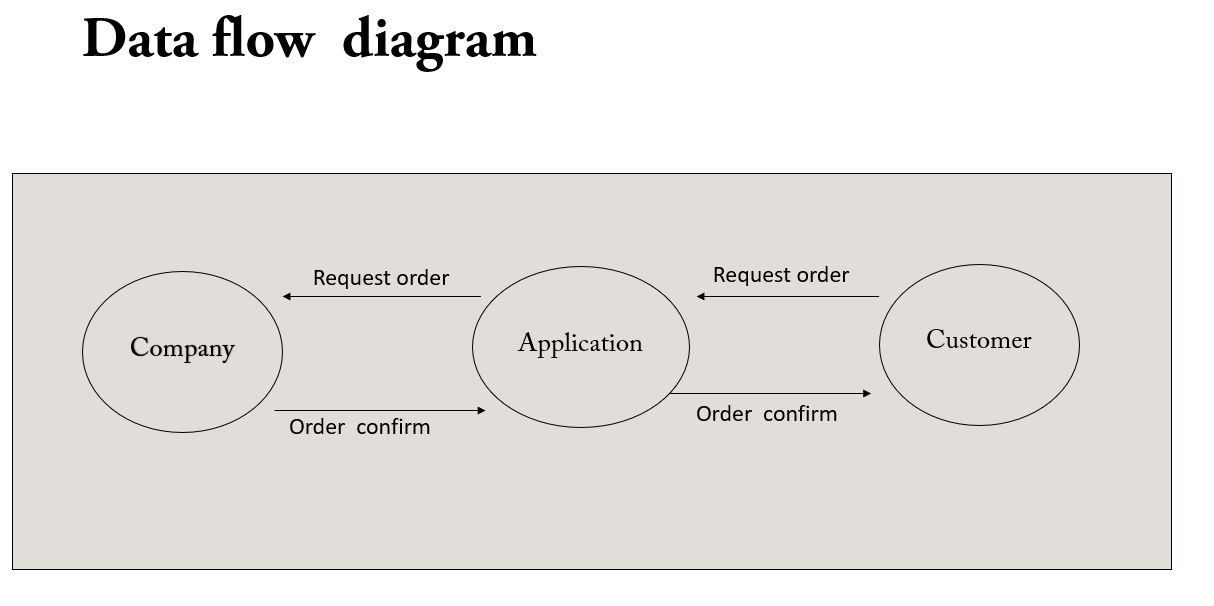


Fig:3.6

**DESIGN**

SUVIDHA

Ravi crop

Kharib Crop

Jayad Crop

Fig 3.7

Kishan Mitra ra

Ravi Crop

Fertilizer1

Pesticide 1

Fertilizer 2

Pesticide 2

Pesticide 3

Fertilizer 3

Fertilizer4

Fig: 3.8

**Hardware Specification**

* Ram : 512 MB or higher
* memory : 200MB
* Processor : 1 GHz or higher
* other : GPS enable

**Software Specification**

* Android Studio 2021.0.1
* Java 15
* XML
* Firebase 2020.0.1

**FRONT END :**

**Extensible Markup Language** (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. ... The design goals of XML emphasize simplicity, generality, and usability across the Internet. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML

XML is a software and hardware-independent tool for storing and transporting. XML is a markup much like HTML. XML was designed to store and transport data. XML was designed to be self-descriptive. XML is a Recommendation.

XML is usually used for descriptive purposes, but there are ways to display XML data. If you don't define a specific way for the XML to be rendered, the raw XML is displayed in the browser.

One way to style XML output is to specify [CSS](https://developer.mozilla.org/en-US/docs/Web/CSS) to apply to the document using the xml-stylesheet processing instruction.

[**Example**](https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction#example)

<?xml version="1.0" encoding="UTF-8"?>

<message>

<warning>

Hello World

<!--missing </warning> -->

</message>

Fig: 3.9

**XML Does Not Use Predefined Tags**

* The XML language has no predefined tags
* The tags in the example above (like to> and <form>) are not defined in any XM standard.
* HTML works with predefined tags like phl table, etc.
* With XML, the author must define both the tags and the document structure

**XML ATTRIBUTES**

* XML elements can have attributes, just like HTML
* Attributes are designed to contain data related to specific element.

**ADVANTAGES OF XML**

* XML uses human, not computer, language, XML is readable and understandable, even by novices, and no more difficult to code than HTML
* XML is completely compatible with Java and 100% portable. Any application that can process XML can use your information, regardless of platform XML is extendable.

**Java**

One of the most widely used programming languages, Java is used **as the server-side language for most back-end development Projects**, including those involving big data and Android development. Java is also commonly used for desktop computing, other mobile computing, games, and numerical computing.

Java is used to desktop and mobile applications, big data processing, embedded systems, and According to Oracle, the company that owns Java, Java rums on 3 billion devices worldwide, which m Java one of the most popular programming languages,

**Why Use Java?**

* Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
* It is one of the most popular programming language in the world.
* It is open-source and free. It is secure, fast and free.
* It has a huge community support (tens of millions of developers)
* Java is an object oriented language which gives a clear to be reused, lowering , development cont. • As Java is close to C++ and C# it makes it easy for programming to Java or vice versa.

Java Syntax

In the previous chapter, we created a Java called Main java, and we used the following print "Hello World" to the screen. code to

My Class java

Public class Main {

Public static void main(String[] args)

{ system.out.println(“Hello World”);

}

}

**Advantage of Java:**

1. **Simple -** Java is a simple programming language since it is easy to learn and easy to understand.
2. **Object-Oriented-** Java uses an object-oriented paradigm, which makes it more practical. Everything in Java is an object which takes care of both data and behaviour.
3. **Secured-** Java is a secured programming language because it doesn't use Explicit pointers Also, Java programs sun inside the virtual machine sandbox.
4. **Robust -**Java is a robust programming language since it uses strong memory Management we can also handle exceptions through the Java code.
5. **Platform Independent-** Java code can run multiple platforms directly, we need not Compile it every time.
6. **Multi-Threaded -** Java uses a multi-threaded environment in which a bigger task can be converted into various threads and run separately.

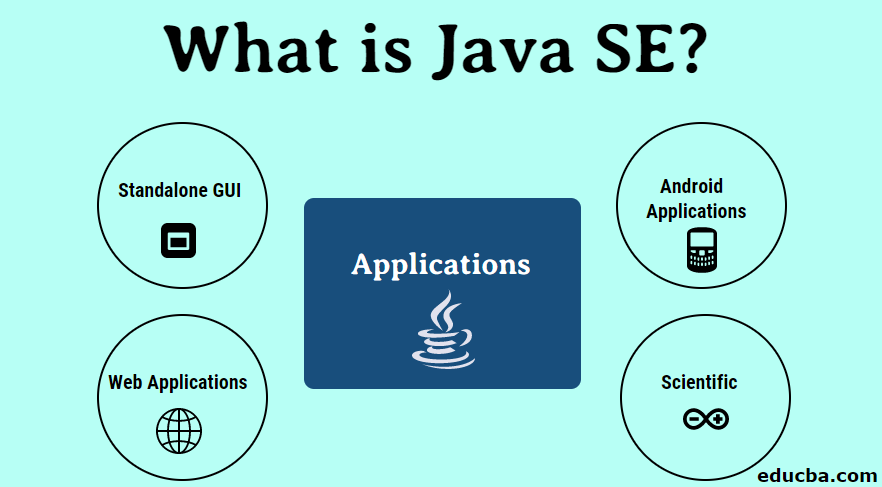


Fig: 3.10

**Firebase**

Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development.

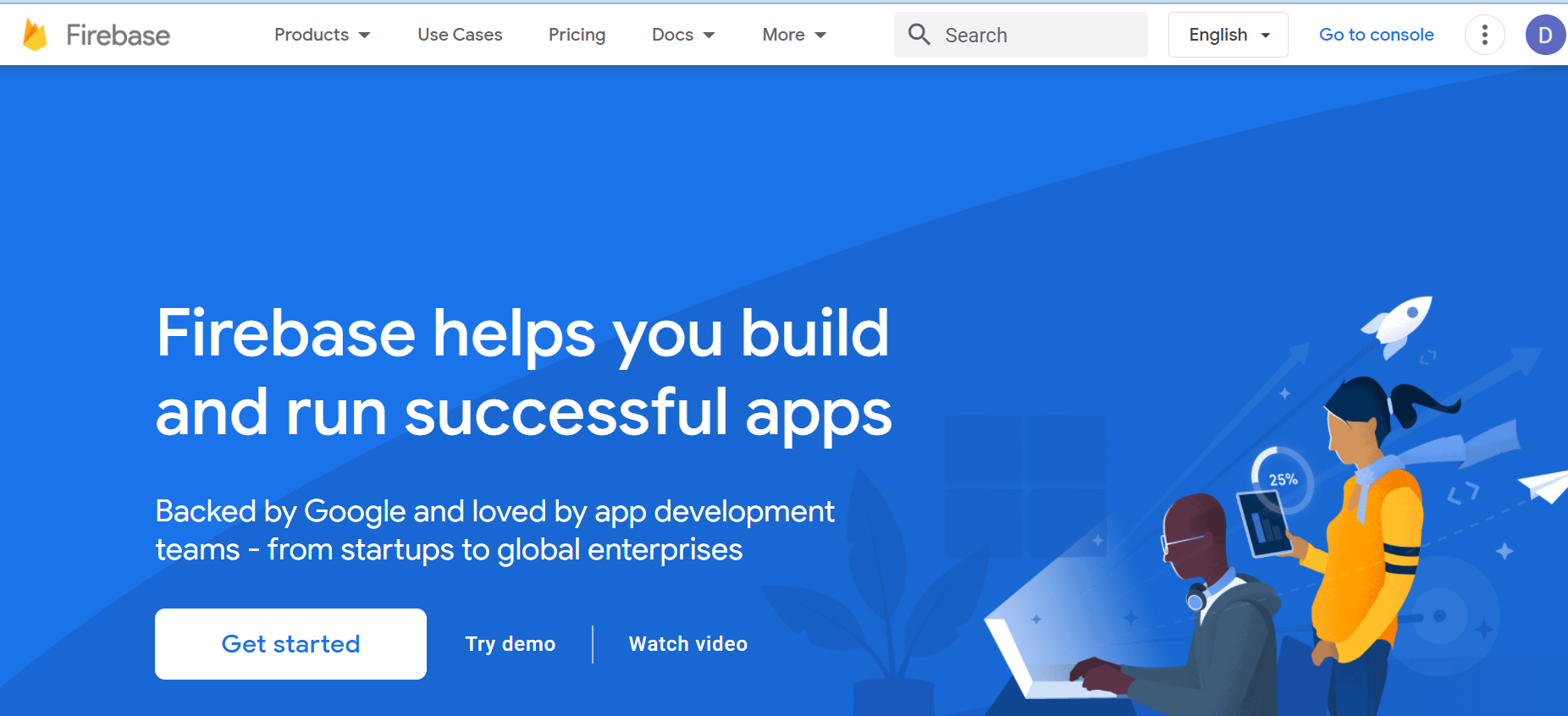


Fig: 3.11

**CHAPTER 4**

**IMPLEMENTATION**

**IMPLEMENTATION**

* implementation uses the contact number user get confirmation to produce code
* Demonstration that the program satisfies its specification validates the code.
* Typically sample runs of the program demonstrating the behaviour for expected data values and boundary values are required.
* Implementation begins with the user-invoked user and other user toward the modules that do not call any other user .The implementation may user depth-first or breath-first.

**Main\_Activity.java**

*package* com.example.b01;  
  
*import* androidx.annotation.NonNull;  
*import* androidx.appcompat.app.AppCompatActivity;  
  
*import* android.animation.Animator;  
*import* android.animation.AnimatorListenerAdapter;  
*import* android.content.Intent;  
*import* android.os.Bundle;  
*import* android.text.TextUtils;  
*import* android.view.View;  
*import* android.view.*WindowManager*;  
*import* android.view.animation.Animation;  
*import* android.widget.Button;  
*import* android.widget.EditText;  
*import* android.widget.ImageView;  
*import* android.widget.TextView;  
*import* android.widget.Toast;  
*import* com.google.android.gms.tasks.*OnCompleteListener*;  
*import* com.google.android.gms.tasks.Task;  
*import* com.google.firebase.FirebaseException;  
*import* com.google.firebase.auth.*AuthResult*;  
*import* com.google.firebase.auth.FirebaseAuth;  
*import* com.google.firebase.auth.PhoneAuthCredential;  
*import* com.google.firebase.auth.PhoneAuthOptions;  
*import* com.google.firebase.auth.PhoneAuthProvider;  
*import* com.google.firebase.messaging.FirebaseMessaging;  
*import* java.util.Timer;  
*import* java.util.TimerTask;  
*import* java.util.concurrent.TimeUnit;  
  
*public class* MainActivity *extends* AppCompatActivity {  
  
  
 ImageView imageView;  
 TextView textView;  
 Timer timer;  
 Animation topanim;  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 getWindow().setFlags(*WindowManager*.LayoutParams.***FLAG\_FULLSCREEN***,  
 *WindowManager*.LayoutParams.***FLAG\_FULLSCREEN***);  
 setContentView(R.layout.***activity\_main***);  
  
 FirebaseMessaging.*getInstance*().subscribeToTopic("Notification");  
  
  
 imageView = (ImageView) findViewById(R.id.***image\_view***);  
 textView = (TextView) findViewById(R.id.***textView7***);  
  
 imageView.animate().alpha(0f).setDuration(0);  
 textView.animate().alpha(0f).setDuration(0);  
  
 imageView.animate().alpha(1f).setDuration(1000).setListener(*new* AnimatorListenerAdapter() {  
 @Override  
 *public void* onAnimationEnd(Animator animation) {  
 textView.animate().alpha(1f).setDuration(800);  
 }  
 });  
 timer = *new* Timer();  
 timer.schedule(*new* TimerTask() {  
 @Override  
 *public void* run() {  
 Intent intent = *new* Intent(MainActivity.*this*, HomeActivity.*class*);  
 startActivity(intent);  
 finish();  
 }  
 }, 4000);}}

**Main\_Activity.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"  
 *android:background*="@color/white"  
 *tools:context*=".MainActivity">  
  
 <TextView  
 *android:id*="@+id/textView7"  
 *android:layout\_width*="412dp"  
 *android:layout\_height*="33dp"  
 *android:layout\_marginBottom*="50dp"  
 *android:fontFamily*="serif"  
 *android:includeFontPadding*="false"  
 *android:text*="Presented by Tiger"  
 *android:textAlignment*="center"  
 *android:textAllCaps*="true"  
 *android:textColor*="@color/browon"  
 *android:textSize*="20sp"  
 *app:layout\_constraintBottom\_toBottomOf*="parent"  
 *app:layout\_constraintEnd\_toEndOf*="parent"  
 *app:layout\_constraintStart\_toStartOf*="parent"  
 *app:layout\_constraintTop\_toBottomOf*="@+id/image\_view"  
 *app:layout\_constraintVertical\_bias*="0.747" />  
  
 <androidx.constraintlayout.utils.widget.ImageFilterView  
 *android:id*="@+id/image\_view"  
 *android:layout\_width*="300dp"  
 *android:layout\_height*="300dp"  
 *android:layout\_marginLeft*="100dp"  
 *android:layout\_marginRight*="100dp"  
 *android:adjustViewBounds*="false"  
 *android:src*="@drawable/logo"  
 *app:layout\_constraintBottom\_toBottomOf*="parent"  
 *app:layout\_constraintEnd\_toEndOf*="parent"  
 *app:layout\_constraintHorizontal\_bias*="0.477"  
 *app:layout\_constraintStart\_toStartOf*="parent"  
 *app:layout\_constraintTop\_toTopOf*="parent" />  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**Login\_page.java**

*package* com.example.b01;  
  
*import* androidx.annotation.NonNull;  
*import* androidx.appcompat.app.AppCompatActivity;  
  
*import* android.content.Intent;  
*import* android.os.Bundle;  
*import* android.text.TextUtils;  
*import* android.view.View;  
*import* android.view.*WindowManager*;  
*import* android.widget.Button;  
*import* android.widget.EditText;  
*import* android.widget.Toast;  
  
*import* com.google.android.gms.tasks.*OnCompleteListener*;  
*import* com.google.android.gms.tasks.Task;  
*import* com.google.firebase.FirebaseException;  
*import* com.google.firebase.auth.*AuthResult*;  
*import* com.google.firebase.auth.FirebaseAuth;  
*import* com.google.firebase.auth.PhoneAuthCredential;  
*import* com.google.firebase.auth.PhoneAuthOptions;  
*import* com.google.firebase.auth.PhoneAuthProvider;  
  
*import* java.util.concurrent.TimeUnit;  
  
*public class* Register *extends* AppCompatActivity {  
  
 EditText phone,otp;  
 Button btngenOTP,btnverify;  
 String mVerificationId;  
 FirebaseAuth mAuth;  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 getWindow().setFlags(*WindowManager*.LayoutParams.***FLAG\_FULLSCREEN***,  
 *WindowManager*.LayoutParams.***FLAG\_FULLSCREEN***);  
 setContentView(R.layout.***activity\_register***);  
  
  
 phone =findViewById(R.id.***phone***);  
 otp = findViewById(R.id.***otp***);  
 btngenOTP = findViewById(R.id.***btmGenerateOtp***);  
 btnverify = findViewById(R.id.***btnVerifyOtp***);  
 mAuth = FirebaseAuth.*getInstance*();  
  
 *if* (mAuth.getCurrentUser() != *null*)  
 {  
 startActivity(*new* Intent(getApplicationContext(),MainActivity.*class*));  
 finish();  
 }  
 btngenOTP.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
  
 otp.setVisibility(view.***VISIBLE***);  
 btnverify.setVisibility(view.***VISIBLE***);  
 *if*(TextUtils.*isEmpty*(phone.getText().toString()))  
 {  
 Toast.*makeText*(Register.*this*,"Enter Valid Phone no",Toast.***LENGTH\_SHORT***).show();  
 }  
 *else* {  
 String number = phone.getText().toString();  
 sendverificationcode(number); }} });  
  
 btnverify.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
  
 *if*(TextUtils.*isEmpty*(otp.getText().toString()))  
 {  
 Toast.*makeText*(Register.*this*,"Wrong OTP Enterd",Toast.***LENGTH\_SHORT***).show();  
  
 }  
 *else* verifycode(otp.getText().toString());  
  
 }  
 });  
 }  
  
 *private void* sendverificationcode(String phoneNumber) {  
  
 PhoneAuthOptions options =  
 PhoneAuthOptions.*newBuilder*(mAuth)  
 .setPhoneNumber("+91"+phoneNumber) .setTimeout(60L, TimeUnit.***SECONDS***) *// Timeout and unit* .setActivity(*this*) *// Activity (for* .setCallbacks(mCallbacks) .build();  
 PhoneAuthProvider.*verifyPhoneNumber*(options);  
  
 }  
 *private* PhoneAuthProvider.OnVerificationStateChangedCallbacks  
  
 mCallbacks = *new* PhoneAuthProvider.OnVerificationStateChangedCallbacks() {  
  
 @Override  
 *public void* onVerificationCompleted(PhoneAuthCredential credential) {  
  
  
 *final* String code = credential.getSmsCode();  
 *if*(code!=*null*)  
 {verifycode(code); }}  
  
 @Override  
 *public void* onVerificationFailed(FirebaseException e) {  
  
 Toast.*makeText*(Register.*this*,"Verification failed",Toast.***LENGTH\_SHORT***).show();  
 }  
  
 @Override  
 *public void* onCodeSent(@NonNull String verificationId,  
 @NonNull PhoneAuthProvider.ForceResendingToken token)

{*super*.onCodeSent(verificationId,token);  
 mVerificationId = verificationId; }};  
  
 *private void* verifycode(String Code) {  
 PhoneAuthCredential credential=PhoneAuthProvider.*getCredential*(mVerificationId,Code);  
 signinCredentials(credential);  
  
 }  
  
 *private void* signinCredentials(PhoneAuthCredential credential) {  
 FirebaseAuth firebaseAuth = FirebaseAuth.*getInstance*();  
 firebaseAuth.signInWithCredential(credential)  
 .addOnCompleteListener(*new* OnCompleteListener<*AuthResult*>() {  
 @Override  
 *public void* onComplete(@NonNull Task<*AuthResult*> task) {  
 *if*(task.isSuccessful())  
 {  
 Toast.*makeText*(Register.*this*,"Login Successfully",Toast.***LENGTH\_SHORT***).show();  
 startActivity(*new* Intent(Register.*this*, MainActivity.*class*)); }  
 }  
 });}}

**Login\_page.xml**

<?*xml version*="1.0" *encoding*="utf-8"?>  
<RelativeLayout *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*=".Register">  
  
 <View  
 *android:id*="@+id/view"  
 *android:layout\_width*="wrap\_content"  
 *android:layout\_height*="120dp"  
 *android:background*="@color/design\_blue" />  
  
  
 <View  
 *android:id*="@+id/view1"  
 *android:layout\_width*="match\_parent"  
 *android:layout\_below*="@+id/view"  
 *android:layout\_height*="100dp"  
 *android:background*="@drawable/ic\_wave\_\_2"/>  
  
 <ImageView  
 *android:id*="@+id/imageView"  
 *android:layout\_width*="75dp"  
 *android:layout\_height*="58dp"  
 *android:layout\_alignParentRight*="true"  
 *android:layout\_marginTop*="30dp"  
 *android:layout\_marginRight*="20dp"  
 *app:srcCompat*="@drawable/ic\_baseline\_perm\_identity\_24" />  
  
  
 <Button  
 *android:id*="@+id/btmGenerateOtp"  
 *android:layout\_width*="244dp"  
 *android:layout\_height*="48dp"  
 *android:layout\_below*="@id/phone"  
 *android:layout\_centerInParent*="true"  
 *android:layout\_marginTop*="20dp"  
 *android:background*="@drawable/round\_border"  
 *android:backgroundTint*="@color/design\_blue"  
 *android:text*="Generate Otp"  
 *android:textColor*="#FAF4F4" />  
  
 <EditText  
 *android:id*="@+id/otp"  
 *android:layout\_width*="252dp"  
 *android:layout\_height*="59dp"  
 *android:layout\_below*="@id/btmGenerateOtp"  
 *android:background*="@drawable/round\_border"  
 *android:drawableStart*="@drawable/ic\_baseline\_smartphone\_24"  
 *android:drawablePadding*="10dp"  
 *android:layout\_marginTop*="20dp"  
 *android:layout\_centerInParent*="true"  
 *android:ems*="10"  
 *android:hint*="OTP"  
 *android:visibility*="invisible"  
 *android:inputType*="number"  
 *android:padding*="16dp" />  
 <Button  
 *android:id*="@+id/btnVerifyOtp"  
 *android:layout\_width*="244dp"  
 *android:layout\_marginTop*="20dp"  
 *android:layout\_height*="48dp"  
 *android:layout\_below*="@id/otp"  
 *android:layout\_centerInParent*="true"  
 *android:background*="@drawable/round\_border"  
 *android:backgroundTint*="@color/design\_blue"  
 *android:text*="Verify Otp"  
 *android:visibility*="invisible"  
 *android:textColor*="#FAF4F4" />  
  
</RelativeLayout>

**Home\_page.java**

*package* com.example.b01;  
  
*import ….*   
  
*import* com.google.android.material.bottomnavigation.BottomNavigationView;  
  
*public class* HomeFragment *extends* Fragment {  
  
 *public* ImageView ravi, kharif, jayat;  
 @Override  
  
 *public* View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  
 View view = inflater.inflate(R.layout.***fragment\_home***, container, *false*);  
  
 ravi=view.findViewById(R.id.***ravi***);  
 ravi.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Intent intent = *new* Intent(getActivity(), ravi.*class*);  
 startActivity(intent);  
 Toast.*makeText*(getActivity(), "loading", Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 kharif=view.findViewById(R.id.***kharif***);  
 kharif.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Intent intent = *new* Intent(getActivity(), kharib.*class*);  
 startActivity(intent);  
 Toast.*makeText*(getContext(), "loading", Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 jayat=view.findViewById(R.id.***jayat***);  
 jayat.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Intent intent = *new* Intent(getActivity(), Jayad.*class*);  
 startActivity(intent);  
 Toast.*makeText*(getContext(), "loading", Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
  
  
 *return* view;  
 }  
}

**Home\_page.xml**

<?*xml version*="1.0" *encoding*="utf-8"?>  
<RelativeLayout *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"  
 *tools:context*=".HomeFragment">  
  
  
 <HorizontalScrollView  
 *android:id*="@+id/horizontalScrollView"  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="190dp"  
 *android:fillViewport*="true"  
 *android:layout\_alignParentTop*="true"  
 *tools:ignore*="SpeakableTextPresentCheck">  
  
  
 <LinearLayout  
 *android:layout\_width*="wrap\_content"  
 *android:layout\_height*="match\_parent"  
 *android:gravity*="bottom"  
 *android:orientation*="horizontal">  
  
  
  
 <ImageView  
 *android:id*="@+id/imageView2"  
 *android:layout\_width*="400dp"  
 *android:layout\_height*="wrap\_content"  
 *app:srcCompat*="@drawable/img" />  
  
 <ImageView  
 *android:id*="@+id/imageView4"  
 *android:layout\_width*="400dp"  
 *android:layout\_height*="wrap\_content"  
 *app:srcCompat*="@drawable/image2"/>  
 </LinearLayout>  
  
 </HorizontalScrollView>  
  
 <ImageView  
 *android:id*="@+id/ravi"  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="170dp"  
 *android:layout\_below*="@+id/horizontalScrollView"  
 *android:src*="@drawable/ravi"/>  
 <ImageView  
 *android:id*="@+id/kharif"  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="170dp"  
 *android:layout\_below*="@+id/ravi"  
 *android:src*="@drawable/kharif"  
 />  
  
</RelativeLayout>

**CHAPTER 6**

**TESTING PROCESS**

**TESTING**

**Test Information flow:-**

Test configuration

Reliability

predicted

Expected result

Testing result

error

Evolution configuration

error

Correction

**Testing**

**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. Test techniques include, but not necessarily limited to:

analysing the product requirements for completeness and correctness in various contexts like industry perspective, business perspective, feasibility and viability of implementation, usability, performance, security, infrastructure considerations, etc. reviewing the product architecture and the overall design .

CHAPTER 6

RESULT/ OUTPUT

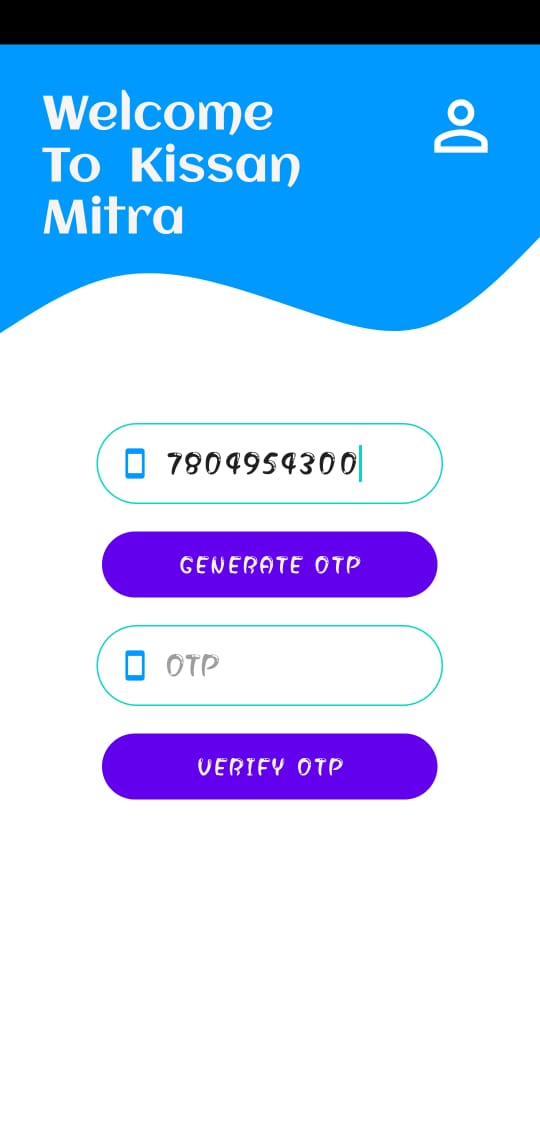


fig: 6.1



Fig: 6.2



Fig 6.3



Fig 6.4

**CHAPTER 7**

**CONCLUTION / FETURE WORK**

**Future scope**

* This project will be put up on the cloud platform, so that it will be accessible by every Android user.
* The application will prove beneficial for every farmers , or even the common people as well as shopkeeper .
* In future we can also add the agriculture tool so that farmers can easily purchase the tool in good cost.

**CONCLUTION**

**“KISHAN MITRA ”** is an application for Smart phones that works on Android Operating system. This application makes direct connection between farmers and company’s. This application allow you to compare the quality, cost and quantities of different company’s Fertilizer and pesticides.