

# **Mini Project -1**

**(2020-2021)**

**“TRADE-APP”**

**FINAL PROJECT REPORT**



**SUBMITTED BY:-**

***YASH KR. SINGH***

***(University Roll No-181500458)***

***BALARAM SAMANTA***

***(University Roll No-181500188)***

**Under the Guidance of:-**

**Mr. VAIBHAV DIWAN SIR**

**Technical Trainer**

**(Department of Computer Science Engineering & Applications)**

## **TEAM DETAILS**

**TEAM MEMBER 1 NAME: YASH KUMAR SINGH ([yashsinghcs](#))**

**TEAM MEMBER 2 NAME: BALARAM SAMANTA ([balaram714](#))**

**PROJECT IDEA: “TRADE FABRICS”**

## **DECLARATION**

**I hereby declare that I have completed my three months at\_(GLA UNIVERSITY) from\_(15 AUGUST) to\_(25 NOVEMBER) under the guidance of Mr. Vaibhav Diwan sir. I declare that I have worked with full dedication , my learning outcomes fulfill the requirements of GLA UNIVERSITY,MATHURA**

**Yash Kr. Singh**

**Balaram Samanta**

**Date:**

## **ACKNOWLEDGEMENT**

**I wish to express my gratitude to,my trainer at GLA UNIVERSITY for providing me an opportunity to do my project work in “Android App Development”.Under his guidance I have completed my project and tried my best to implement what I had learnt till now.**

**I sincerely thank sir, our coordinator for their guidance and encouragement to do my MINI PROJECT.He also help me by updating us about the information of what to do and not to do during our Project and help us with all.I also thanks my friend for helping me with my problem that I face in my project.**

## **TABLE OF CONTENT**

1. Introduction about android
2. Knowing about android
3. Overview and starting with android
4. Overview And Starting With Android
5. Index
6. Problem Statement
7. Literature Review
8. UML Diagrams
9. WireFrames
10. Project Code

# **Introduction**

## **1.1 What is an app?**

App is an abbreviated form of the word "application". An application is a software program that is designed to a specific function directly for the user which can be accessed easily.

## **1.2 What is an Android?**

Android is an open source operating System for mobile devices such as smart-phones and tablets,computers. Android offers a unified approach to application development for mobile devices which means developers need to develop only for Android, and their applications should be able to run on different devices powered by android.

Android was developed by the Open Handset Alliance(OHA) , led by Google, and other companies.Android is mainly based on direct manipulation ,using touch gestures that loses correspond to real world actions. It provide us with the manipulate on-screen object,along with a virtual keyboard for text input.

### **1.3 History of Android :**

Android was initially developed by Android Inc., the code names of android ranges from A to N.

Founded by Andy Rubin in Palo Alto, California, United States in Oct 2003.

Android word is actually referred to as robot

Android was named after the nickname of Andy Rubin by their co-workers for his love for robots.

Google acquired android Incorporation on 17th Aug, 2005.

Android was developed by the Open Handset Alliance(OHA) for camera phones, led by Google, and other companies.

But shifted to smart-phones due to the low market of cameras at that time.

HTC launches the first android mobile.

In 2010, Google launched it's Nexus series of devices in the smart-phone world with android OS.

### **1.4 Android Version:**

1. Android 1.0, 1.1(Base, Base\_1\_1)
2. Android 1.5(Cupcake)
3. Android 1.6(Donut)
4. Android 2.0(Eclair)
5. Android 2.0.1(Eclair\_0\_1)
6. Android 2.1.x(Eclair\_MR1)
7. Android 2.2.x(Froyo)

8. Android 2.3 - 2.3.2(Gingerbread)
9. Android 2.3.3 - 2.3.4(Gingerbread\_MR1)
10. Android 3.0.x, 3.1.x, 3.2(HoneyComb, HC \_MR1, HC\_Mr2)
11. Android 4.0 - 4.0.2, 4.0.3 - 4.0.4((Ice\_Cream\_Sandwich,ICS\_MR1) 12.  
Android 4.1 - 4.1.1, 4.2 - 4.2.2,4.3(Jelly\_Bean, JB\_MR1, JB\_MR2)
13. Android 4.4, 4.4W (Kitkat, K\_Watch)
14. Android 5.0, 5.1(Lollipop, L\_MR1)
15. Android 6.0 (Marshmallow)
16. Android 7.0 (Nougat)

## **KNOWING ANDROID**

We will start our Android application development on any of the following operating systems:

Microsoft Window XP or later version.



Mac OS X 10.5.8 or later version with Intel chip Linux including GNU Library with Intel chip.

## **2.1 Android App Development.**

Android app is a combination of different source code in a single place whose action can be performed just by a single touch.

Example:- Suppose if we want to add two no's then we just have to click on the calculator app and enter two no's and the operand that we have to perform. It makes our work much easier and this are much user friendly. So what happened ? How it calculated the answer.

Android programming is based on java programming language so if we have basic understanding on Java programming then it will be a fun to study Android app development.

## **2.2 Java in Android App Development.**

Java is a programming language that doesn't compile to native processor code but rather it refers to a virtual machine which understands an intermediate format i.e; java byte-code. Each platform that uses java to run needs a virtual device.

An android app uses an android application that runs on android platform. It build on custom virtual machine that gives its user the addition usage and application power and a user friendly environment. Android's actual virtual machine is called Dalvik.

### **2.3 Android Software Development Kit(Android SDK) :**

Apps that extend the functionality of a device are written using Android SDK and often using java programming language. The SDK includes a set of development tools, including a debugger, software libraries, a handset emulator, sample code, etc.

Initially Google supported Integrated Development Environment(IDE) i.e; Eclipse using the Android Development Tools(ADT) plugins. Other development tools are also available such as Native Development Kit(NDK). Android Studio is based on IntelliJ IDEA developed by Google as its primary IDE for android app development. Android is a selection of third-party applications which can be acquired by users by downloading and installing the Android Application Package(APK) file.

### **2.4 What is API Level ?**

API Level is an integer value that uniquely identifies the framework API revision offered by a version of the Android platform.

### **2.5 Features of Android :**

It is an open source user friendly software It has beautiful user interface

It reduced cost of development

It has rich development environment Inter application Integration

It support single and bi-directional text

It uses Dalvik virtual machine- optimized version for mobile

### **2.6 Android Application :**

They are generally developed in the java language using the Android SDK  
Android applications can be packaged easily and sold out either through a store such as Google play,etc.

There are many android applications that we already know and uses them few are music,news,weather,etc

Many android applications are also available for free, most of them are already available in play stores.

These apps are compatible with almost every platform but few are also available for the specific platform..

## **OVERVIEW AND STARTING WITH ANDROID**

### **3.1 Overview**

Android applications are freely available and can be downloaded from the Web. Following is the list of software's you will need before you start your Android application programming.

This all software required in the installation of Android application.This setup is required for the configuration with RAM less or more than 4gb:

Java JDK5 or JDK6

Download the latest version of Java JDK and install the JDK and set the environmental path for it.

## Android SDK

Download Android SDK from Android's official website :

<http://developer.android.com/sdk/index.html>

If you install SDK either on Mac OS or Linux, follow the instructions and set up the environment path.

Launch Android SDK Manager using option All Program>Android SDK Tools>SDK Manager

## Eclipse IDE for Java Developers

Check for the version that is compatible with your device and install it.

Or, Android Development Tools (ADT) Eclipse Plugin

This step will help you in setting Android Development Tool plugin for Eclipse.

Let's start with launching Eclipse and then, choose Help > Software Updates > Install New Software

### 3.2 Android Virtual Device

To test our android application we will need a virtual Android device. Before start writing our code we create an Android Virtual Device Android operating system is a stack of software components which is roughly divided into five sections and four main layers :

#### Application

You will find all the Android applications at the top layer. You will write your application to be installed on this layer only

#### Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes.

## Libraries

There is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database, etc.

And , Android Runtime :

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android

## Linux Kernel

At the bottom of the layers is Linux, This provides basic system functionality like process management, memory management, device management like camera, keypad, display etc.

### **3.3 Application Component**

Application components are the essential building blocks of an Android application. These component are loosely coupled by the application manifest file, i.e;

Android-

-Manifest.xml

Following four main component that can be used within an Android application :

1) Activities :- They dictate the UI and handle the user interaction to the smartphone screen public class MainActivity extends Activity

```
{ }
```

2) Services :- They handle background processing associated with an application  
public class MyService extends Service

```
{ }
```

3) Broadcast Receivers :- They handle communication between Android OS and applications  
public class MyReceiver extends BroadcastReceiver

```
{ }
```

4) Content Providers :- They handle data and database management issues.  
public class MyContentProvider extends ContentProvider

```
{ }
```

### 3.4 Additional Component :

There are additional components which will be used in the construction of above mentioned entities, their logic, and wiring between them. They are :

1) Fragments :- Represent a behavior or a portion of user interface in an Activity.

2) Views :- UI elements that are drawn onscreen including buttons, lists forms etc

3) Layouts :- View hierarchies that control screen format and appearance of the views

4) Intents :- Messages wiring components together.

5) Resources :- External elements, such as strings, constants and drawable pictures

6) Manifest :- Configuration file for the application

## **Problem Statement**

### **1. What do we want to create?**

- 1.1. A smartphone application, Which on further development may add web application

### **2. What is our idea about?**

- 2.1. This project aims to develop online shopping for customers with the goal that it is very easy to shop your loved things from an extensive number of merchants available on the web .On further development may convert into a app plus and a website Where people who are manufacturers of different clothes on different scale can come together and sell their products on the single platform which allows them to have a hassle-free

payment as well as good selling of the product. With the help of this buyers can carry out an online shopping from home. Here is no compelling reason to go to the crowded stores or shopping centers during and after lockdown . You simply require a mobile or a laptop and one important payment sending option to shop online or to sell . To get to this online shopping system all the users will need to have an email and password to login and proceed your shopping . The login credentials for an online shopping system are under high security and nobody will have the capacity to crack it easily. Upon successful login the customers can purchase a wide range of fabrics such as Arganj Shalabh fabric, Butter Crepe Nylon, Chanderi Buti Fabric, Handloom Tussar Pure Silk Fabric etc. can be dispatched using an online shopping system. And of course you will get your requested ordered items at your door step. It is simple. You will pick your favourite items from a variety of online merchants looking at cost and quality. No need to go physical shops with this you will have more time to spend with your family. It Just need a mobile and a payment making options like net banking, credit card, debit card or online wallets. Almost a wide range of things can be brought through online shopping system. You can purchase goods from foreign places from your bedroom and you will get your goods at your home. It is extremely secure. Customer service is accessible.



- 2.2. Many online shopping which allows to sell different types of things at a single place but this app is mainly focused on a single product fabrics because it has been seen that many merchants are facing problem while selling these online as people who are willing to buy direct fabrics for readymade clothes but they fail as there is no online platform for particularly this work. There are a vast number of merchants other merchants who are in the work of converting fabrics to clothes needed this app or a website so that they could also buy fabrics online and could expand the business a bit faster rather than to buy fabrics physically going to manufacturers of fabrics.
- 2.3. Something that makes the project different is that we would be not selling readymade clothes other than that we would be selling fabrics which would be made directly from machines then after that people would buy them and use them to make Sherwani and clothes like kurta etc.
- 2.4. We would have 2 options for the user to login either as merchant or as customer/buyer where merchants would be able to list the item to sell and customers would buy them.
- 2.5. every merchant would be allowed to request for verification by the admin where the people who are DBA are allowed to assign them either verified or non verified merchants which would help the buyer to know and to be sure where to buy products from.
- 2.6. In addition to this we would charge a very small amount of commission from both the clients.

2.7. And the main highlight would be that the amount we would earn no matter what 50% of it would be given to charity every year and the same will be displayed to all the users for their contribution made.

**3. Has the idea/project been implemented by someone else in the past?**

3.1. Yes, we have seen many online websites and apps selling things online and similar to them we also aim to sell our goods online as well as of other merchants that are willing to sell their products online. For example: amazon, flipkart, indiamart etc. most important feature of our app is **“its only for Fabrics ,not for any other items.As there is no applications till now which is selling fabrics. All of them are selling readymade items.But there are no facilities for the people who want to buy fabrics. Releasing this application we can make an end to this problem.”**

## **Literature Review**

- ❑ Our application is basically based on online shopping where people can order from their home their desired things. But these kinds of applications have already been given by established companies like AMAZON, FLIPKART etc . There are also various other well known companies who have implemented the same.
- ❑ In all the shopping apps , there are various users all over the world. Each of them will have an ID and password for login. After successful login the user can freely find his desired item. After selecting the item the person has to go for the payment process via any means. After successful payment the user will be given a specific date within which the item will be delivered. This is the complete concept of an online shopping application.
- ❑ Our app will be similar to that. But there are some changes which will make it unique. The first and the most

important feature of our app is **“its only for Fabrics ,not for any other items.As there is no applications till now which is selling fabrics. All of them are selling readymade items.But there are no facilities for the people who want to buy fabrics. Releasing this application we can make an end to this problem.”**

- ❑ Second important uniqueness is **“this application is both user and seller friendly.Seller can very easily put their Fabrics with all the required data with it.According to the users requirement the data will be shown to the screen.”**



The image shows a digital signpost for Bhadohi, India. The signpost has a yellow rectangular display showing "BHADOHI" in English and "भदोही" in Hindi. Below the signpost is a login form titled "BUSSINESS OF TEXTILES". The form includes fields for "USER ID:" (containing "yash") and "PASSWORD:" (containing "...."). Below these fields are two buttons: "FORGETPASSWORD" and "LOGIN". The "LOGIN" button has a small "LOGIN" label next to it.

this would be the login page of our project and there would be a signup button too .where different users can sign up according to the choice that weather they want to be merchant or buyer.But login page will be same for all and once logged in users will have a choice selection for which role they wanna log in but in buyers case he can only log in as buyer on the other hand seller can act as both.

Register

Username

Email

Password

Confirm Password

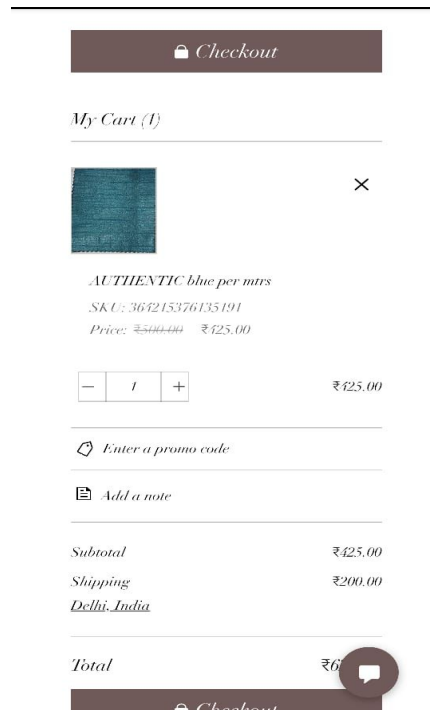
Sign Up

Already Have An Account? [Login](#)

this is how the registration page would look like but with more fields and before registration a choice selection will take place asking for which role user wants to sign up/register for normal users this much info is ok but for merchants there would be more columns for bank details.



After login the particular user this would be a role selection page after the login in which only merchants can select both options after successful registration/authentication but buyers/normal users can only select buyers image i.e. to buy only not to sell.(*the top img is for buyers and bottom one is for merchants*).



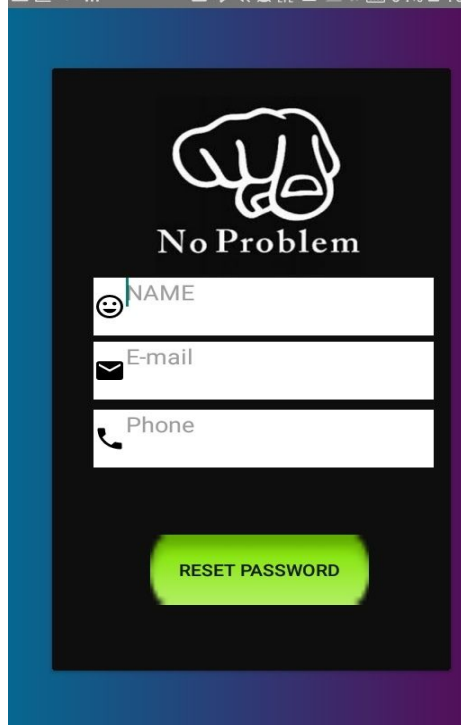
this would be the check out page where invoice generation will take place for buyer and he would get the required amount to be paid



this would be the catalog viewing for the buyers where they can select the fabric according to their choice i.e which color or type of fabric etc they want.

- There will also be a page for forget password by which a user can change his or her password. By requesting a reset password link on their registered mail id.





No Problem

NAME

E-mail

Phone

RESET PASSWORD

- So after all these, all the items which are put up by all the sellers will be visible to the users in the proper list. He or she just has to search for the item and proceed for the payment.

## **SPECIFIC REQUIREMENTS:**

### **★Hardware Requirements(for developers):**

- Processor: i5 8 gen and above
- Minimum of 8 Gb ram and 256 Gb rom.

### **★Hardware Requirements(for clients):**

- Internet enabled mobile phone
- Development must support at least version 7.0 of Android.
- Minimum of 512 MB ram and 8 GB of rom

### ★Software Requirements(for developers):

- Android Studio version 3.6 and up.
- All the data is Stored on Google Firebase.

### ★Functional Requirements:

- Login:there must be a login for clients
- Client Details:separate details after registration of clients
- Merchants Details:separate details after registration of merchants
- Inventory: the stocks left are stored in here and displayed.
- Sales: track of amount sold and left with money transfer done
- Login Detail: for every login attempt made there must be a login detail of clients.

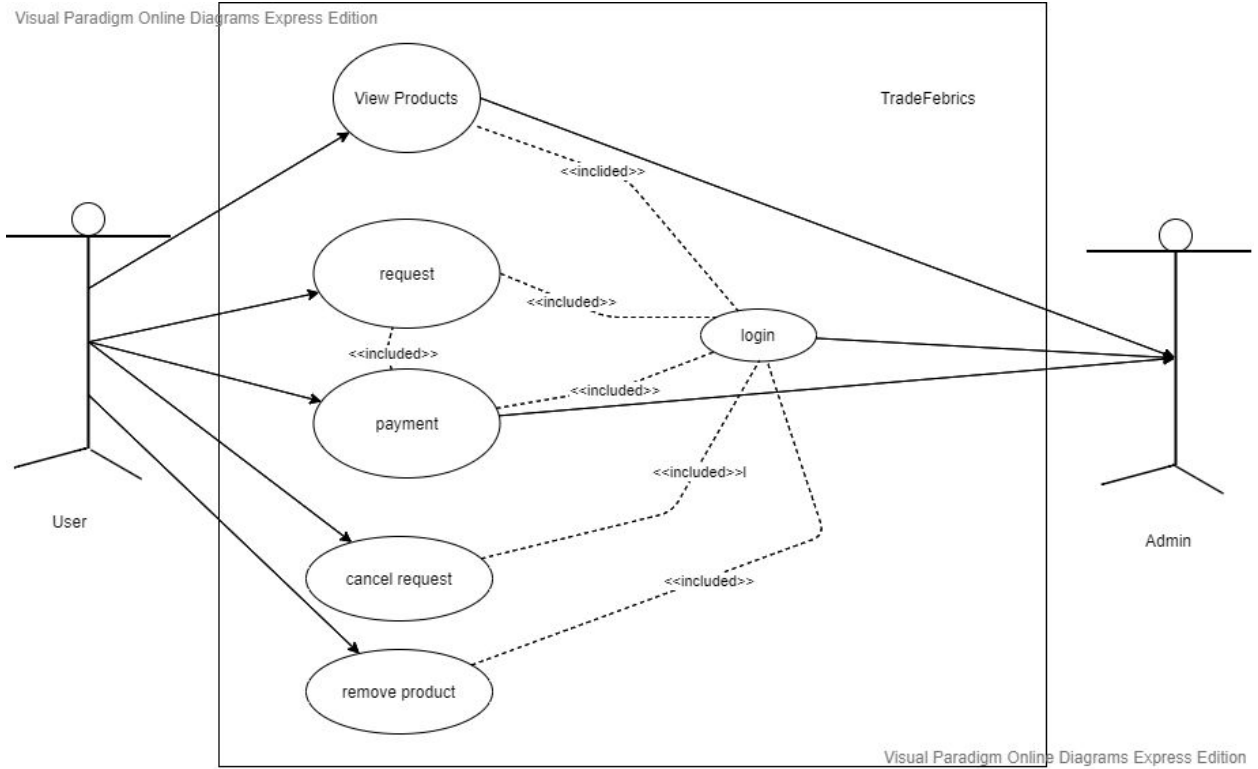
### ★Non-Functional Requirements:

- System should be able to handle multiple users.

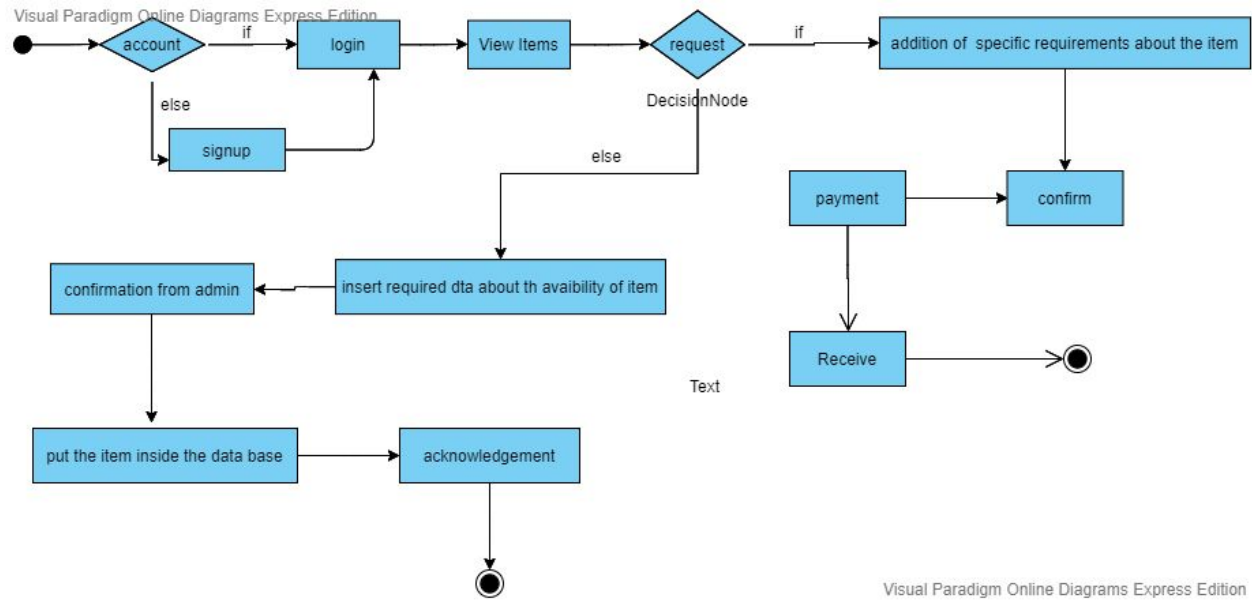
- Login by username, password should be incorporated wherever necessary
- Should be user friendly and display easy to understand error messages

## **UML Diagrams**

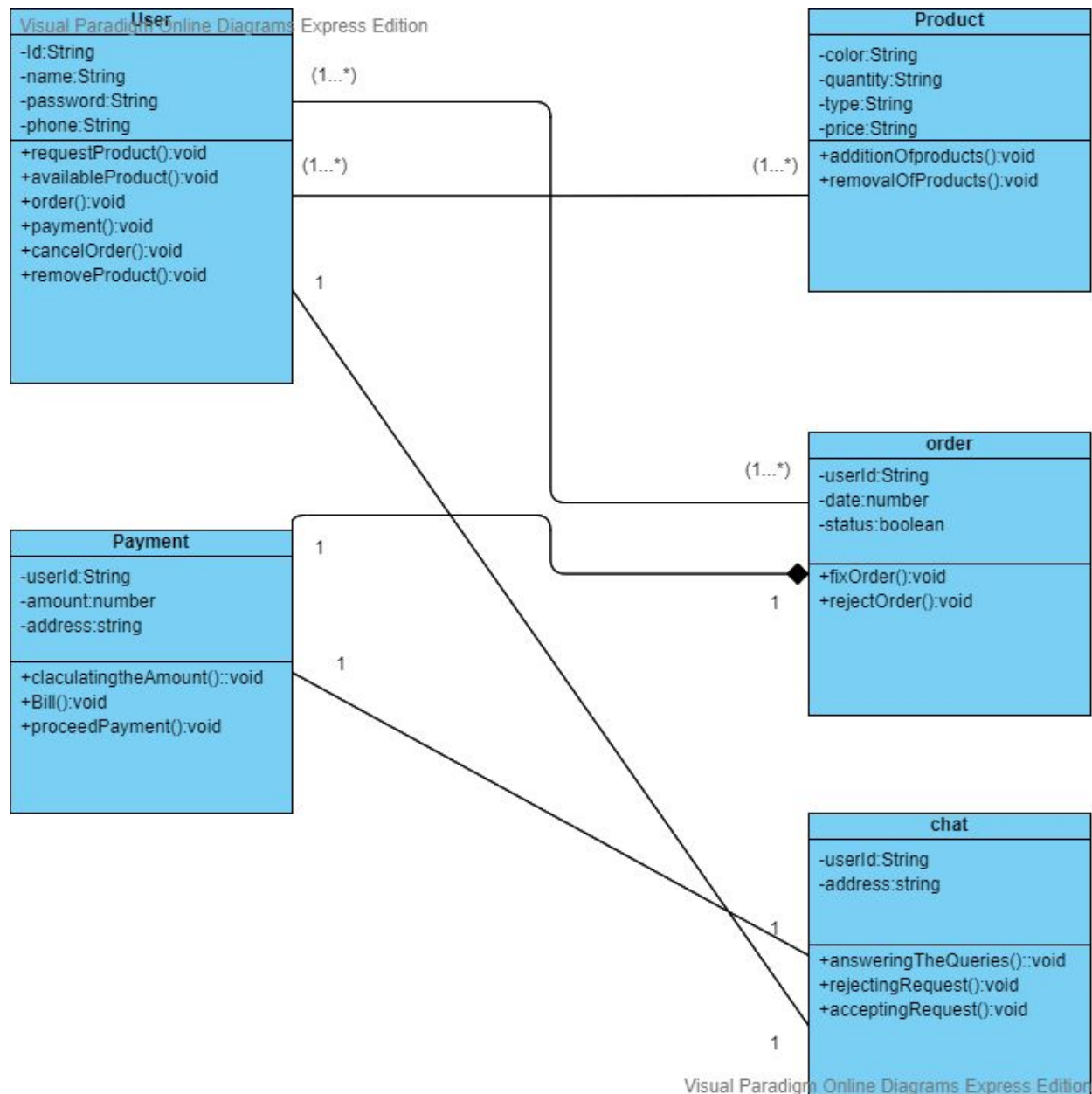
### **USE CASE DIAGRAM:**



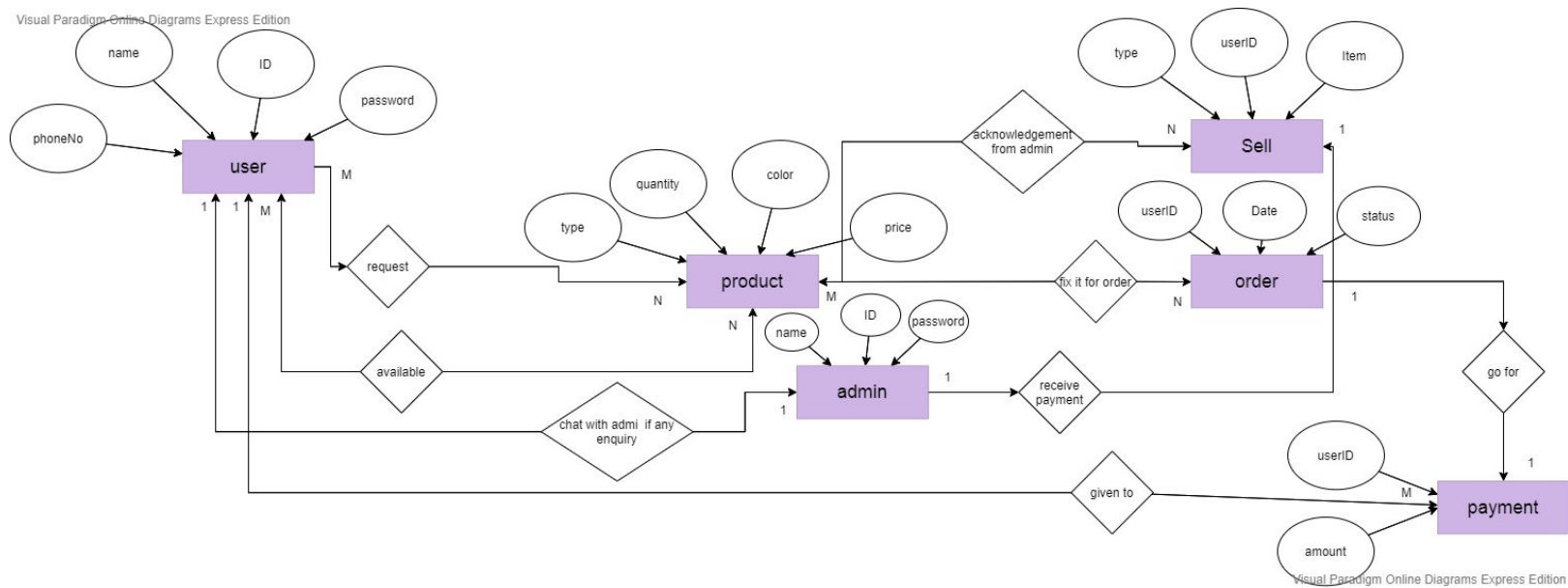
## Activity Diagram



## Class Diagram



## E-R Diagram



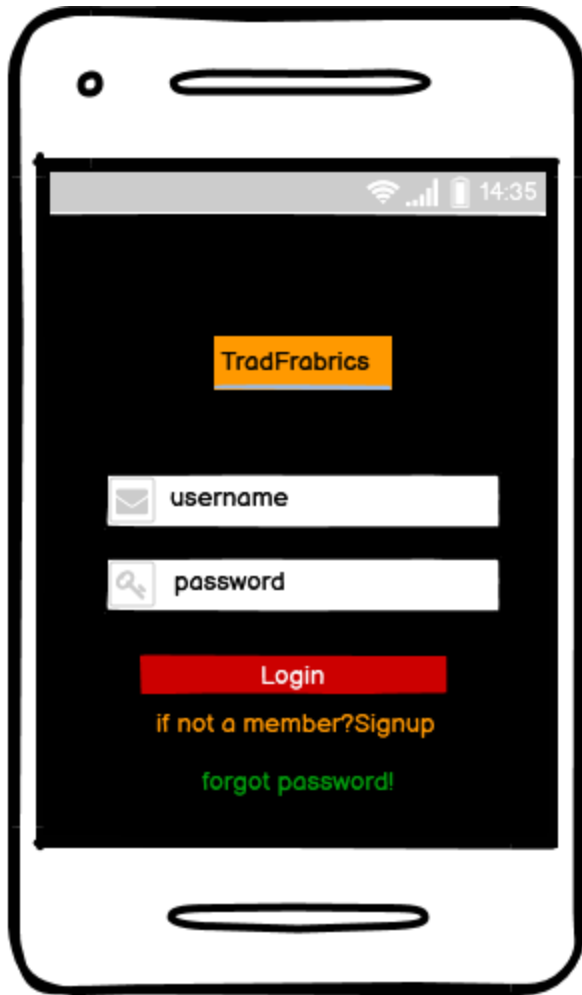
## WireFrames:

### 1.first page

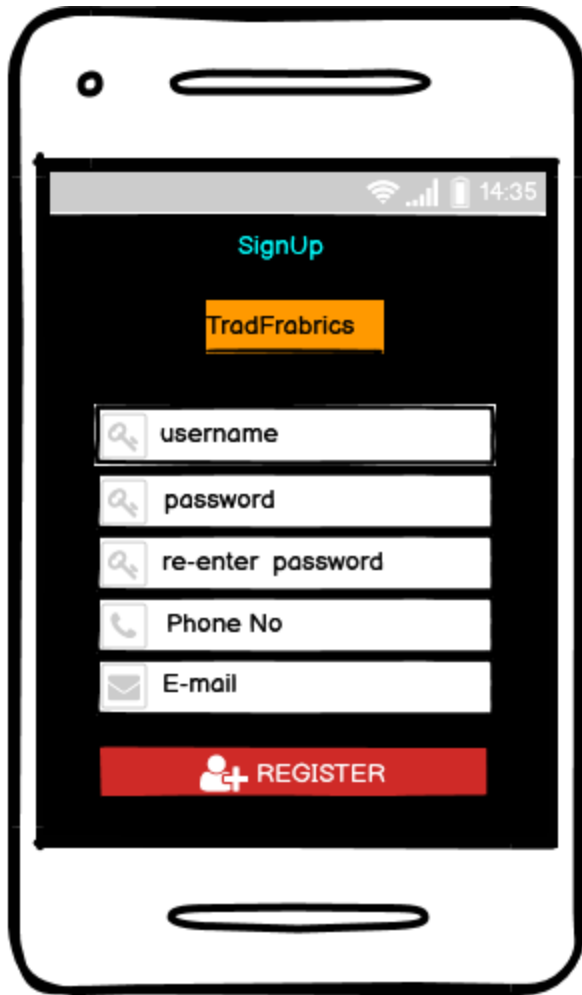


## 2.Login Page





### 3.SignUp Page



#### 4.Forgot Password page



5.after login page



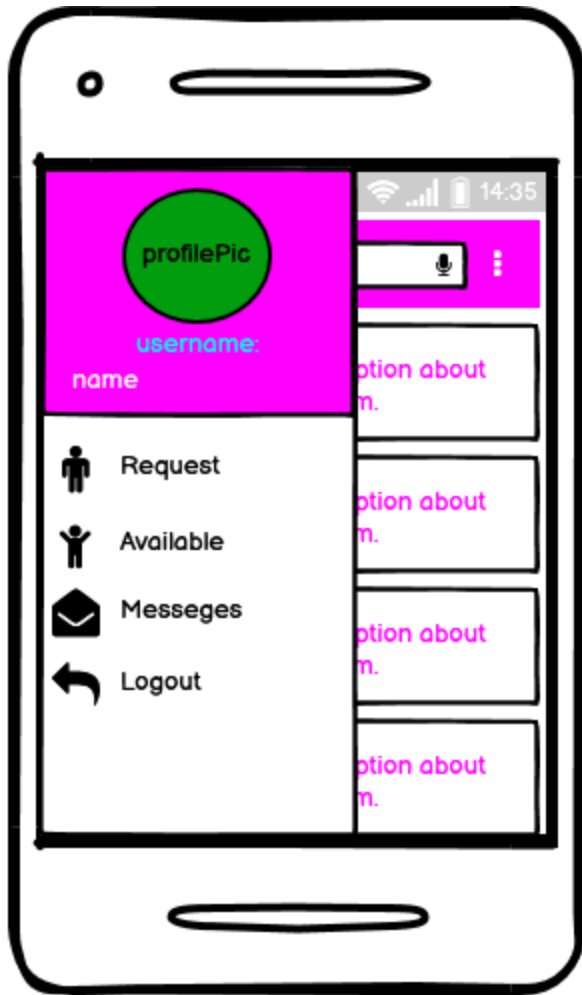
## 6.Available products enter page



## 7.Requeste a products page



## 8. user navigation page



## **PROJECT CODES:**

### **KOTLIN FILES:**

#### **1. MostFront page:**

```
package munik.android.projects.tradeapp
```

```
import android.content.Intent
```

```

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import java.util.*

class MostFrontPage : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.frontpage)

        //added timer so that ,after the 5 seconds of the apps opening it will go
        to the login page
        val timer:Timer= Timer()
        timer.schedule(object : TimerTask() {
            override fun run() {
                //this method will take us to the login Activity
                goToLoginPage()
            }
        }, 3500)
    }
    private fun goToLoginPage(){

        val goToLoginPage = Intent(this, LoginPage::class.java)
        startActivity(goToLoginPage)
        finish()
    }
}

```

## **2.Login Page:**

```

package munik.androidprojects.tradeapp

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity

```



```
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.TextView
import android.widget.Toast
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.auth.FirebaseUser
import java.nio.charset.MalformedInputException
```

```
class LoginPage : AppCompatActivity() {
    private lateinit var signup_button_LoginPage:TextView
    private lateinit var forgetpassword:TextView
    private lateinit var loginbutton : Button
    private lateinit var Auth: FirebaseAuth
    private lateinit var username: EditText
    private lateinit var password: EditText
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login_page)

        //initialising the View objects.
        signup_button_LoginPage = findViewById(R.id.signup)
        forgetpassword = findViewById(R.id.forgetpassword_textview)
        loginbutton = findViewById(R.id.login)
        username=findViewById(R.id.username)
        password=findViewById(R.id.password)
        Auth=FirebaseAuth.getInstance()
        //FirebaseAuth.getInstance().signOut();
        /* if (Auth!!.currentUser != null) {
            startActivity(Intent(applicationContext, LoginPage::class.java))
            finish()
        }*/
        signup_button_LoginPage.setOnClickListener {
```

```

        startActivity(Intent(this,SignupPage::class.java))
    }

    forgetpassword.setOnClickListener {
        startActivity( Intent(this,ForgetPassword::class.java))
    }

    loginbutton.setOnClickListener {
        doLogin()
        //startActivity( Intent(this,ItemLoginPageAfterLoginUser::class.java))

    }
}

private fun doLogin(){
    fun updateUI(currentuser:FirebaseUser?){
        if(currentuser!=null){
            Toast.makeText(baseContext,"Login sucseessful",
Toast.LENGTH_SHORT).show()
            startActivity(Intent(this,ItemLoginPageAfterLoginUser::class.java))
            finish()
        }
        else{
            Toast.makeText(baseContext,"Login Failed",
Toast.LENGTH_SHORT).show()
        }

    }

    if (username.text.toString().isEmpty()) {
        username.error = "please enter username"
        username.requestFocus()
        return
    }
}

```

```

    if (password.text.toString().isEmpty()) {
        password.error = "please enter password"
        password.requestFocus()
        return
    }

```

```

Auth.signInWithEmailAndPassword(username.text.toString(),password.text
.toString()).addOnCompleteListener{
    task ->
    if(task.isSuccessful){
        val user: FirebaseUser?=Auth.currentUser
        updateUI(user)
    }

    else{

        updateUI(null)
    }
}
}
public override fun onStart() {
    super.onStart()
    val user:FirebaseUser?=Auth.currentUser
    if(user!=null){
        Toast.makeText(baseContext,"Login successful",
Toast.LENGTH_SHORT).show()
        startActivity(Intent(this,ItemLoginPageAfterLoginUser::class.java))
        finish()
    }

}
}
}

```

### **3.SignUp page:**

```
package munik.androidprojects.tradeapp

import android.content.Intent
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.firestore.DocumentReference
import com.google.firebase.firestore.FirebaseFirestore
import java.nio.charset.MalformedInputException

class SignupPage : AppCompatActivity() {

    private lateinit var loginButton_SignupPage: Button
    private lateinit var auth: FirebaseAuth
    private lateinit var email : EditText
    private lateinit var password : EditText
    private lateinit var re_enter_password : EditText
    private lateinit var phoneNo : EditText
    private lateinit var e_mail : EditText
    var db = FirebaseFirestore.getInstance()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_signup_page)

        //initializing the View class object
        loginButton_SignupPage =
            findViewById(R.id.sighupverify_button_OTP)
```

```

email = findViewById(R.id.username_signUp)
password = findViewById(R.id.password_signUp)
re_enter_password = findViewById(R.id.reEnterPassword_signUp)
phoneNo = findViewById(R.id.Phoneno_signUp)
e_mail = findViewById(R.id.Email_signUp)

auth = FirebaseAuth.getInstance()

if (auth!!.currentUser != null) {
    startActivity(Intent(applicationContext,
MalformedURLException::class.java))
    finish()
}
loginButton_SignupPage.setOnClickListener {
    signUpUser()
}
}

private fun signUpUser(){
    if (email.text.toString().isEmpty()) {
        email.error = "please enter username"
        email.requestFocus()
        return
    }
    if (password.text.toString().isEmpty()) {
        password.error = "please enter password"
        password.requestFocus()
        return
    }

    auth.createUserWithEmailAndPassword(email.text.toString(),password.text
.toString()).addOnCompleteListener(this){task ->
        if(task.isSuccessful){

```

```

        doAddition()
        Toast.makeText(baseContext,"account created
succsesfully",Toast.LENGTH_SHORT).show();
        startActivity(Intent(this,LoginPage::class.java))
        finish()
    }
    else{
        Toast.makeText(baseContext,"Signup Failed.Try again after
some time",Toast.LENGTH_SHORT).show();
    }
}
}

```

```

private fun doAddition() {
    if (re_enter_password.text.toString().isEmpty()) {
        re_enter_password.error = "please enter enterbookname"
        re_enter_password.requestFocus()
        return
    }
    if (phoneNo.text.toString().isEmpty()) {
        phoneNo.error = "please enter phoneNo"
        phoneNo.requestFocus()
        return
    }
    if (e_mail.text.toString().isEmpty()) {
        e_mail.error = "please enter email"
        e_mail.requestFocus()
        return
    }
    val userdetails: MutableMap<String, Any> = HashMap()
    userdetails["phoneNo"] = phoneNo.text.toString()
    userdetails["email"] = e_mail.text.toString()
    userdetails["username"] = email.text.toString()
}

```

```
//Toast.makeText(Signup_student.this, "user created",  
Toast.LENGTH_SHORT).show();  
    //putting other data like name ,email etc into the fire base collection  
name users
```

```
//Toast.makeText(Signup_student.this, "user created",  
Toast.LENGTH_SHORT).show();  
    //putting other data like name ,email etc into the fire base collection  
name users  
    val userId_techer = auth.getCurrentUser()?.getUid()  
    val documentReference: DocumentReference =  
        db.collection("STUDENT").document(userId_techer as String)  
    documentReference.set(userdetails)  
        .addOnSuccessListener { // Log.i("info", "on success:user  profile is  
created" + userId_techer);  
            // Log.i("info","on success:user  profile is created"+userId);  
            Toast.makeText(baseContext,"added succsesfully",  
Toast.LENGTH_SHORT).show()  
        }  
        .addOnFailureListener {  
            Toast.makeText(baseContext,"not added due to some reason  
please try again", Toast.LENGTH_SHORT).show()  
        }  
    /*db.collection("userdetails")  
        .add(userdetails)  
        .addOnSuccessListener { documentReference ->  
            Toast.makeText(baseContext,"added succsesfully",  
Toast.LENGTH_SHORT).show()  
        }  
        .addOnFailureListener { e ->  
            Toast.makeText(baseContext,"not added due to some reason  
please try again", Toast.LENGTH_SHORT).show()
```

```
    }*/  
}  
}
```

#### **4.Item list after login :**

```
package munik.androidprojects.tradeapp  
  
import android.content.Intent  
import android.os.Bundle  
import android.util.Log  
import android.view.MenuItem  
import android.view.View  
import android.widget.SearchView  
import android.widget.TextView  
import androidx.appcompat.app.ActionBarDrawerToggle  
import androidx.appcompat.app.AppCompatActivity  
import androidx.drawerlayout.widget.DrawerLayout  
import androidx.fragment.app.FragmentTransaction  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.google.android.material.navigation.NavigationView  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.FirebaseUser  
  
class ItemListPageAfterLoginUser : AppCompatActivity() {  
  
    private lateinit var recyclerView : RecyclerView  
    private lateinit var searchView: SearchView  
    lateinit var toggle : ActionBarDrawerToggle  
    lateinit var drawerlayout : DrawerLayout  
    lateinit var navView : NavigationView  
    lateinit var available: Available
```



```

lateinit var request: Request
lateinit var messages: Messages
private lateinit var adapter :CustomAdapter
private lateinit var headText : TextView
private lateinit var user : FirebaseUser
private lateinit var FirebaseAuth : FirebaseAuth
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_item_list_page_after_login_user)

    //initializing the objects
    recyclerView = findViewById(R.id.recycler_view_item) as
RecyclerView
    searchView = findViewById(R.id.searchView) as SearchView
    drawerlayout =
findViewById(R.id.drawerlayout_ItemListPageAfterLoginUser)
    navView = findViewById(R.id.navView)

    FirebaseAuth.getInstance()
    user = FirebaseAuth.getCurrentUser()!!;

    var headerView : View = navView.getHeaderView(0)
    headText = headerView.findViewById(R.id.EMAILid)
    toggle =
ActionBarDrawerToggle(this,drawerlayout,R.string.open,R.string.close)

    recyclerView.layoutManager = LinearLayoutManager(this)
    drawerlayout.addDrawerListener(toggle)
    toggle.syncState()
    supportActionBar?.setDisplayHomeAsUpEnabled(true)

    val docRef =

```

```

        db.collection("STUDENT").document(user.getId())
docRef.get().addOnCompleteListener { task ->
    if (task.isSuccessful) {
        val document = task.result
        if (document!!.exists()) {
            headText.text = document.data?.get("username").toString()
        }
    } else {
        Log.d("info", "get failed with ", task.exception)
    }
}
val user = ArrayList<dataModel>()
/* user . add(dataModel("balaram"))
user . add(dataModel("ishita"))
user . add(dataModel("goggo"))
user . add(dataModel("billi"))
user . add(dataModel("ballu"))
user . add(dataModel("akshat"))
user . add(dataModel("dubeyji"))
user . add(dataModel("himanshu"))
user . add(dataModel("willy"))
user . add(dataModel("ballu"))*/

// recyclerView.adapter = adapter
db.collection("items").get().addOnSuccessListener { documents ->
    for (document in documents) {
        val a = document.data.get("name_of_item") as String
        val b = document.data.get("price") as String + "/"
        val c = document.data.get("quantity") as String + "m"

        //val k = a.toString().substring(10,a.toString().length-1)
        Log.d("info", "get failed with =" + document.data.toString())
        user.add(dataModel(a,b ,c ))
    }
}

```

```

    }
    adapter = customeAdapter(user)
    recyclerView.adapter = adapter
}.addOnFailureListener { exception ->
    Log.d("info", "get failed with ", exception)
}

```

```

searchView.setOnQueryTextListener( object :
SearchView.OnQueryTextListener{
    override fun onQueryTextSubmit(query: String?): Boolean {
        return false
    }

    override fun onQueryTextChange(newText: String?): Boolean {
        adapter.filter.filter(newText)
        return false
    }

})

```

```

navView.setNavigationItemSelectedListener {
    when(it.itemId){
        R.id.available -> {
            available= Available()
            supportFragmentManager.beginTransaction()
                .replace(R.id.container, available)

.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_OPEN).comm
it()

            searchView.setVisibility(View.GONE)
            recyclerView.setVisibility(View.GONE)

```

```

    }
    R.id.request ->{
        request = Request()
        supportFragmentManager.beginTransaction()
            .replace(R.id.container, request)

        .setTransition(FragmentTransaction.TRANSIT_FRAGMENT_OPEN).commit()

        searchView.setVisibility(View.GONE)
        recyclerView.setVisibility(View.GONE)
    }
    R.id.messeges ->{
        messeges = Messeges()
        supportFragmentManager.beginTransaction()
            .replace(R.id.container, messeges)

        .setTransition(FragmentTransaction.TRANSIT_FRAGMENT_OPEN).commit()

        searchView.setVisibility(View.GONE)
        recyclerView.setVisibility(View.GONE)
    }
    R.id.logout ->{
        FirebaseAuth.getInstance().signOut();
        startActivity(Intent(this, LoginPage::class.java))
        finish()
    }
}
}
true
}
}

```

```

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    if(toggle.onOptionsItemSelected(item)){

```

```

        return true
    }
    return super.onOptionsItemSelected(item)
}

override fun onBackPressed() {
    super.onBackPressed()
    startActivity(Intent(this,ItemListPageAfterLoginUser::class.java))
}
}

```

## **5.Available item add :**

```
package munik.androidprojects.tradeapp
```

```

import android.content.Intent
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.firestore.FirebaseFirestore

```

```

// TODO: Rename parameter arguments, choose names that match
// the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
private const val ARG_PARAM1 = "param1"

```

```
private const val ARG_PARAM2 = "param2"
```

```
/**
```

```
 * A simple [Fragment] subclass.
```

```
 * Use the [Available.newInstance] factory method to
```

```
 * create an instance of this fragment.
```

```
 */
```

```
private lateinit var name_of_item : EditText
```

```
private lateinit var quantity :EditText
```

```
private lateinit var price : EditText
```

```
private lateinit var auth: FirebaseAuth
```

```
private lateinit var proceed_button : Button
```

```
var db = FirebaseFirestore.getInstance()
```

```
class Available : Fragment() {
```

```
    // TODO: Rename and change types of parameters
```

```
    private var param1: String? = null
```

```
    private var param2: String? = null
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
```

```
        arguments?.let {
```

```
            param1 = it.getString(ARG_PARAM1)
```

```
            param2 = it.getString(ARG_PARAM2)
```

```
        }
```

```
    }
```

```
    override fun onCreateView(
```

```
        inflater: LayoutInflater, container: ViewGroup?,
```

```
        savedInstanceState: Bundle?
```

```
    ): View? {
```

```
        // Inflate the layout for this fragment
```

```
        val view = inflater.inflate(R.layout.fragment_available, container, false)
```

```

name_of_item = view.findViewById(R.id.name_of_text)
quantity = view.findViewById(R.id.sizeInMeters_availsble)
price = view.findViewById(R.id.price)
prooced_button = view.findViewById(R.id.proocceed)
auth = FirebaseAuth.getInstance()

prooced_button.setOnClickListener {
    val userdetails: MutableMap<String, Any> = HashMap()
    userdetails["name_of_item"] = name_of_item.text.toString()
    userdetails["quantity"] = quantity.text.toString()
    userdetails["price"] = price.text.toString()

    db.collection("items")
        .add(userdetails)
        .addOnSuccessListener { documentReference ->
            Toast.makeText(activity,"added succsesfully",
Toast.LENGTH_SHORT).show()
        }
        .addOnFailureListener { e ->
            Toast.makeText(activity,"not added due to some reason
please try again", Toast.LENGTH_SHORT).show()
        }
        startActivity(Intent(context,ItemListPageAfterLoginUser::class.java))
    }
    return view
}

```

```

companion object {

```

```

    /**

```

```

        * Use this factory method to create a new instance of

```

```

        * this fragment using the provided parameters.

```

```

        *

```

```

        * @param param1 Parameter 1.

```

```

* @param param2 Parameter 2.
* @return A new instance of fragment Available.
*/
// TODO: Rename and change types and number of parameters
@JvmStatic
fun newInstance(param1: String, param2: String) =
    Available().apply {
        arguments = Bundle().apply {
            putString(ARG_PARAM1, param1)
            putString(ARG_PARAM2, param2)
        }
    }
}
}
}

```

## **6.Request for required item:**

```
package munik.androidprojects.tradeapp
```

```

import android.content.Intent
import android.os.Bundle
import android.util.Log
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.firestore.FirebaseFirestore

```

```
// TODO: Rename parameter arguments, choose names that match
```



```
// the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
private const val ARG_PARAM1 = "param1"
private const val ARG_PARAM2 = "param2"
```

```
/**
```

```
 * A simple [Fragment] subclass.
 * Use the [Request.newInstance] factory method to
 * create an instance of this fragment.
 */
```

```
private lateinit var name_of_item : EditText
```

```
private lateinit var quantity : EditText
```

```
private lateinit var price : EditText
```

```
private lateinit var auth: FirebaseAuth
```

```
private lateinit var proceed_button : Button
```

```
private var cunter : Int = 0;
```

```
class Request : Fragment() {
```

```
    // TODO: Rename and change types of parameters
```

```
    private var param1: String? = null
```

```
    private var param2: String? = null
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
```

```
        arguments?.let {
```

```
            param1 = it.getString(ARG_PARAM1)
```

```
            param2 = it.getString(ARG_PARAM2)
```

```
        }
```

```
    }
```

```
    override fun onCreateView(
```

```
        inflater: LayoutInflater, container: ViewGroup?,
```

```
        savedInstanceState: Bundle?
```

```
    ): View? {
```

```
        // Inflate the layout for this fragment
```

```

val view = inflater.inflate(R.layout.fragment_request, container, false)
val db = FirebaseFirestore.getInstance()
name_of_item = view.findViewById(R.id.name_of_request)
quantity = view.findViewById(R.id.sizeInMeters_request)
price = view.findViewById(R.id.price_request)
prooced_button = view.findViewById(R.id.proocceed_request)
auth = FirebaseAuth.getInstance()

prooced_button.setOnClickListener {
    db.collection("items").get().addOnSuccessListener { documents ->
        for (document in documents) {
            val a = document.data.get("name_of_item") as String
            val b = document.data.get("price") as String
            val c = document.data.get("quantity") as String
            if((a.equals(name_of_item.text.toString())) &&
(b.equals(price.text.toString())) && (c.equals(
                quantity.text.toString())) {
                cunter = 1;
                Toast.makeText(context, "the item is available",
Toast.LENGTH_SHORT).show()
                Toast.makeText(context, "proocceed for payment",
Toast.LENGTH_SHORT).show()

startActivity(Intent(context,ItemListPageAfterLoginUser::class.java))
                break
            }

            //val k = a.toString().substring(10,a.toString().length-1)
            // Log.d("info", "get failed with =" + document.data.toString())

        }
        if( cunter == 0){

```

```

        Toast.makeText(context, "out of stok",
Toast.LENGTH_SHORT).show()

startActivity(Intent(context,ItemListPageAfterLoginUser::class.java))
    }

    }.addOnFailureListener { exception ->
        Log.d("info", "get failed with ", exception)

    }
}
return view
}

```

```

companion object {
    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment Request.
     */
    // TODO: Rename and change types and number of parameters
    @JvmStatic
    fun newInstance(param1: String, param2: String) =
        Request().apply {
            arguments = Bundle().apply {
                putString(ARG_PARAM1, param1)
                putString(ARG_PARAM2, param2)
            }
        }
}
}

```

```
}
```

## **7.Custom adapter fpr recycler view in item list:**

```
package munik.android.projects.tradeapp
```

```
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Filter
import android.widget.Filterable
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView
import java.util.*
import kotlin.collections.ArrayList
```

```
class customeAdapter(var arraylist : ArrayList<dataModel>) :
    RecyclerView.Adapter<customeAdapter.viewHolder>() ,Filterable {
    var arrayListFilter = ArrayList<dataModel>()
```

```
    init {
        arrayListFilter = arraylist
    }
```

```
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
viewHolder {
        val v =
LayoutInflater.from(parent?.context).inflate(R.layout.item_holder,parent,fals
e)
        return viewHolder(v)
    }
```

```

override fun getItemCount(): Int {
    return arrayListFilter.size
}

override fun onBindViewHolder(holder: viewHolder, position: Int) {
    val user : dataModel = arrayListFilter[position]
    holder.textViewName?.text = user.header
    holder.textViewPrice?.text = user.header1
    holder.textViewLength?.text = user.header2
    holder.image.setImageResource(R.drawable.ic_baseline_available)
}

class viewHolder(itemView : View) :
RecyclerView.ViewHolder(itemView){
    val textViewName = itemView.findViewById(R.id.description)as
TextView
    val textViewPrice = itemView.findViewById(R.id.description1)as
TextView
    val textViewLength = itemView.findViewById(R.id.description2)as
TextView
    val image = itemView.findViewById(R.id.image) as ImageView
}

override fun getFilter(): Filter {
    return object : Filter() {
        override fun performFiltering(constraint: CharSequence?):
FilterResults {
            val charSearch = constraint.toString()
            if(charSearch.isEmpty()) {
                arrayListFilter = arraylist
            } else{
                val resultList =ArrayList<dataModel>()

```

```

        for(row in arraylist) {

if(row.header.toLowerCase(Locale.ROOT).contains(charSearch.toLowerCase(Locale.ROOT))){
            resultList.add(row)
        }
    }
    arrayListFilter = resultList
}
val filterResult = FilterResults()
filterResult.values = arrayListFilter
return filterResult
}

    override fun publishResults(constraint: CharSequence?, results:
FilterResults?) {
        arrayListFilter = results?.values as ArrayList<dataModel>
        notifyDataSetChanged()
    }

}
}
}

```

### **Data model:**

```

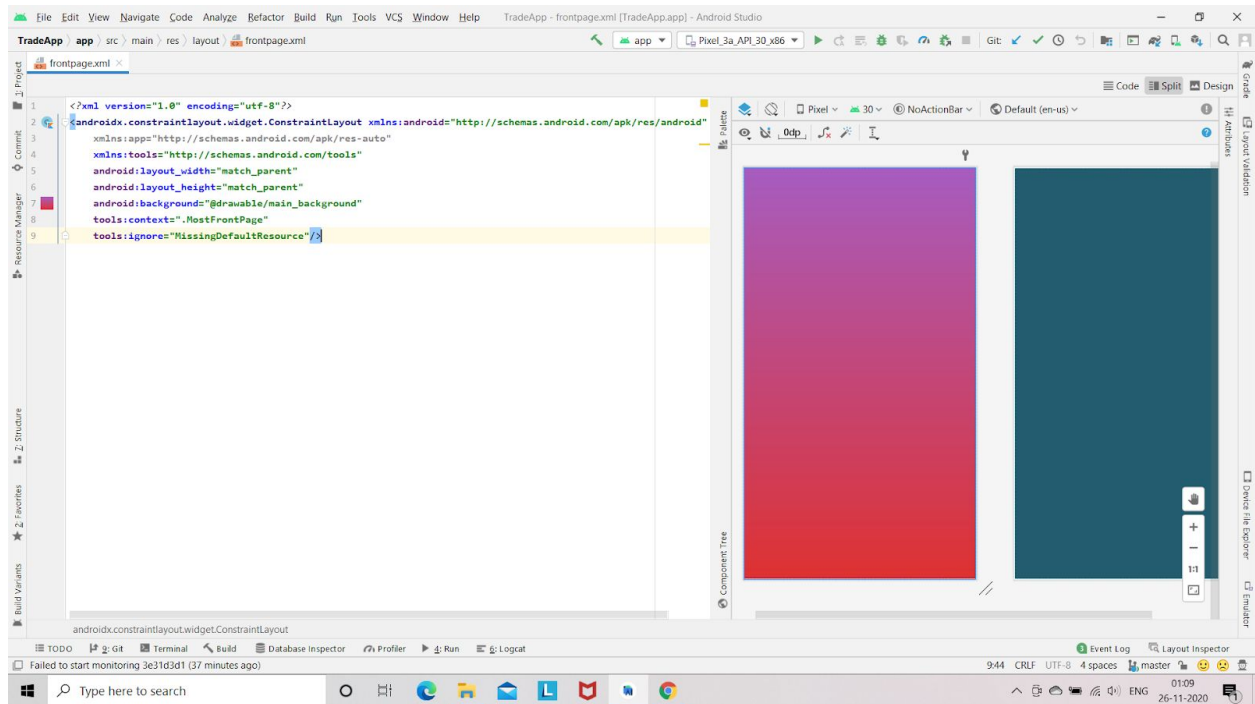
package munik.androidprojects.tradeapp

class dataModel (var header :String,var header1 : String,var header2 :
String) {
}

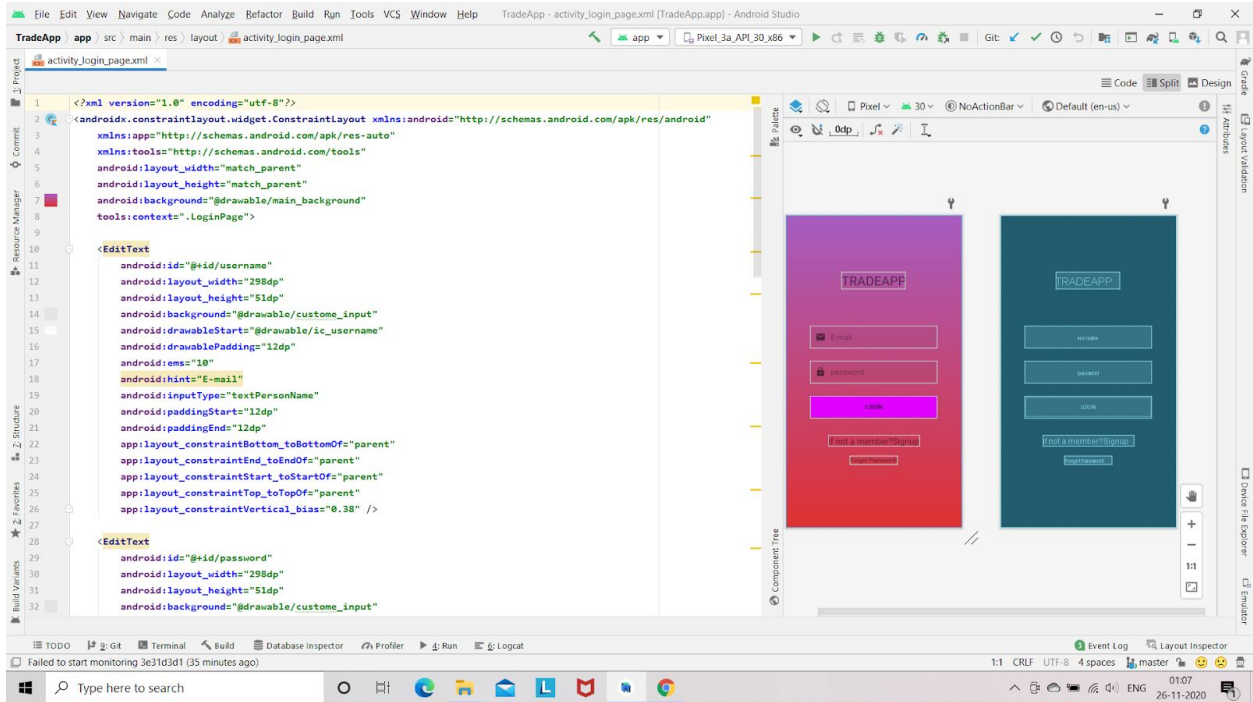
```

# Layouts:

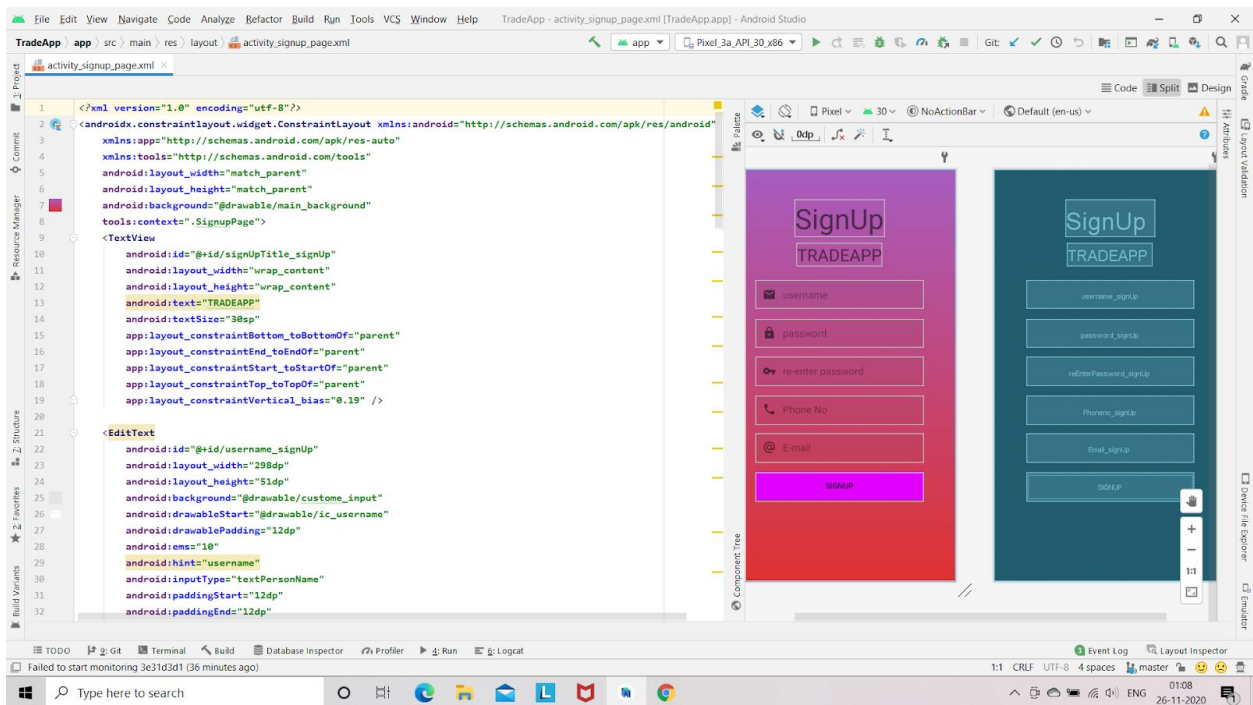
## 1. Most Front Page



## 2.Login Page

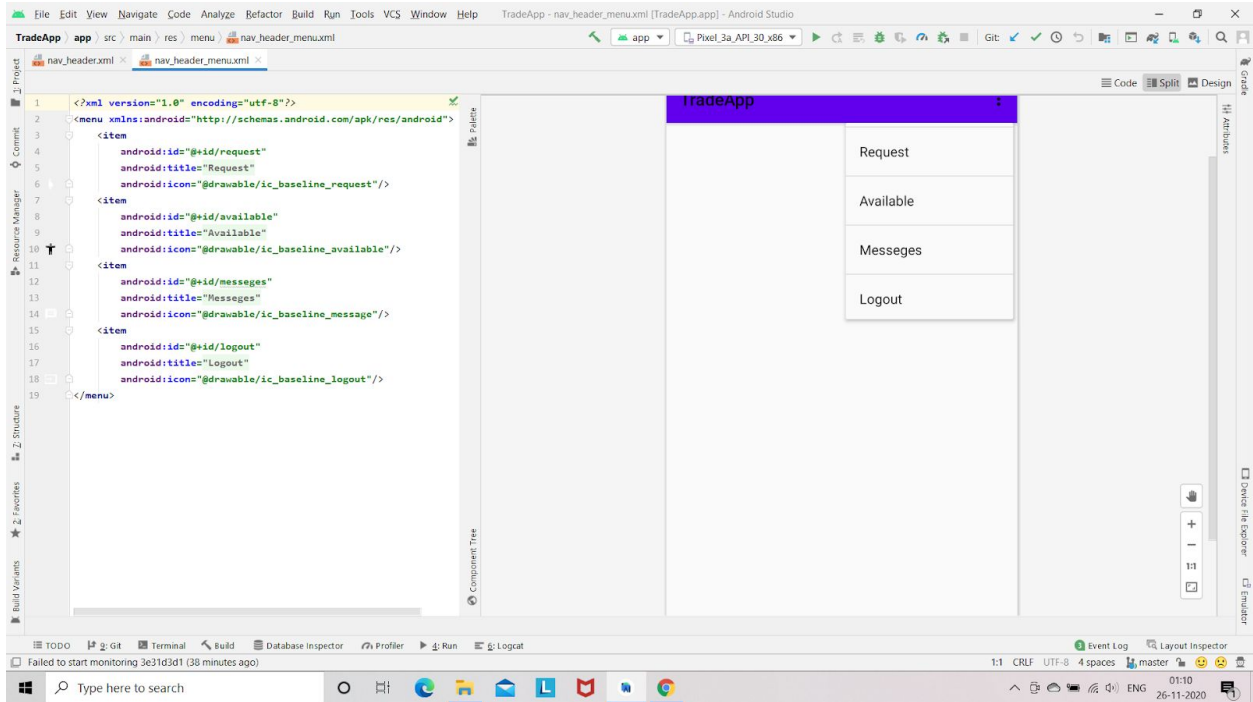


### 3. SignUp Page

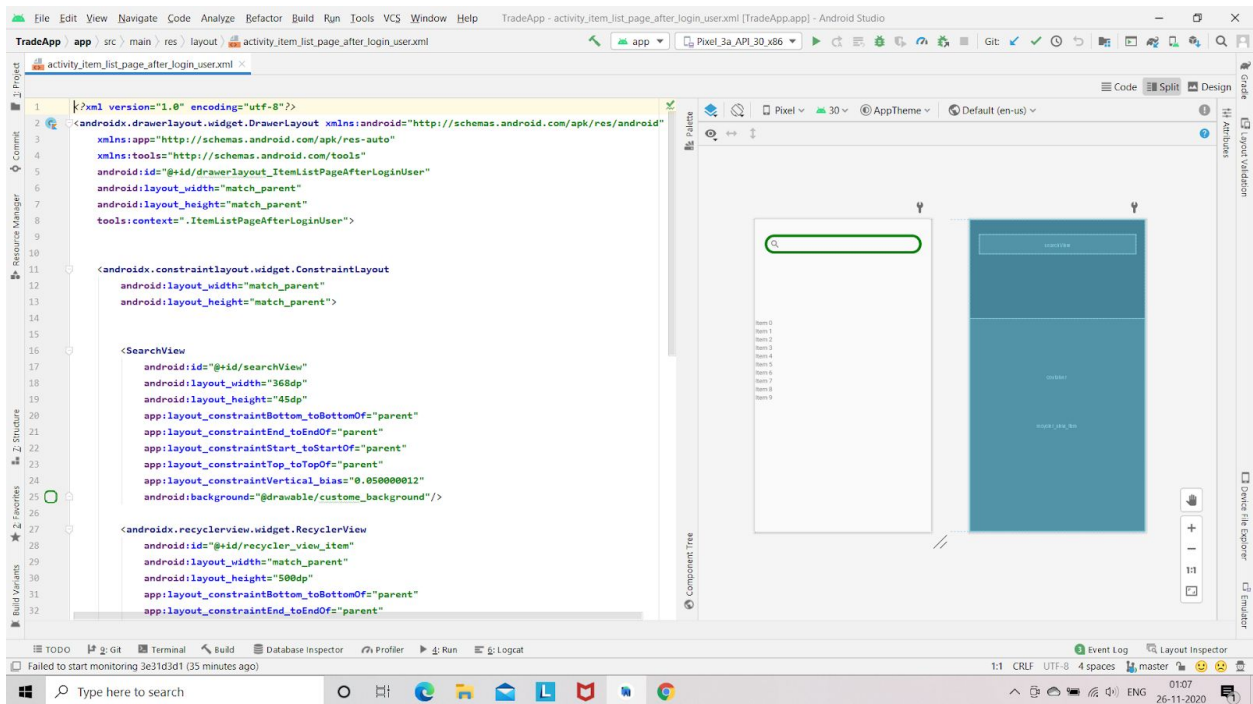


### 3.Menu Items:

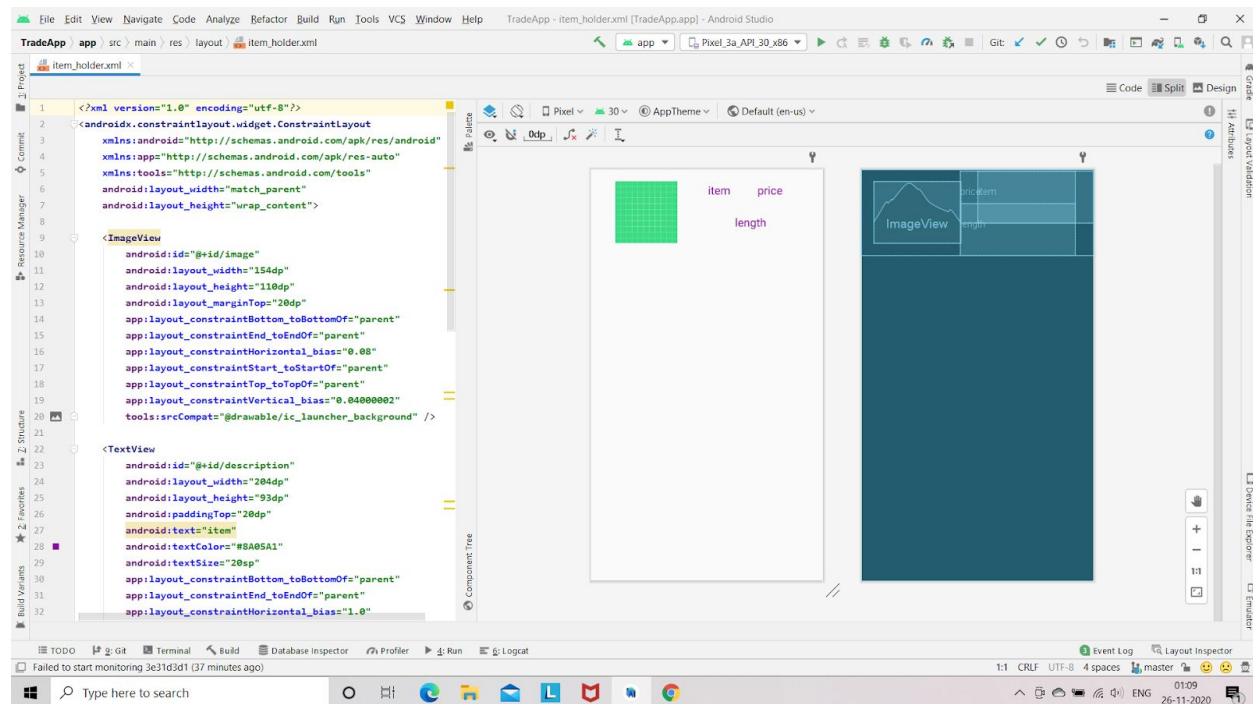
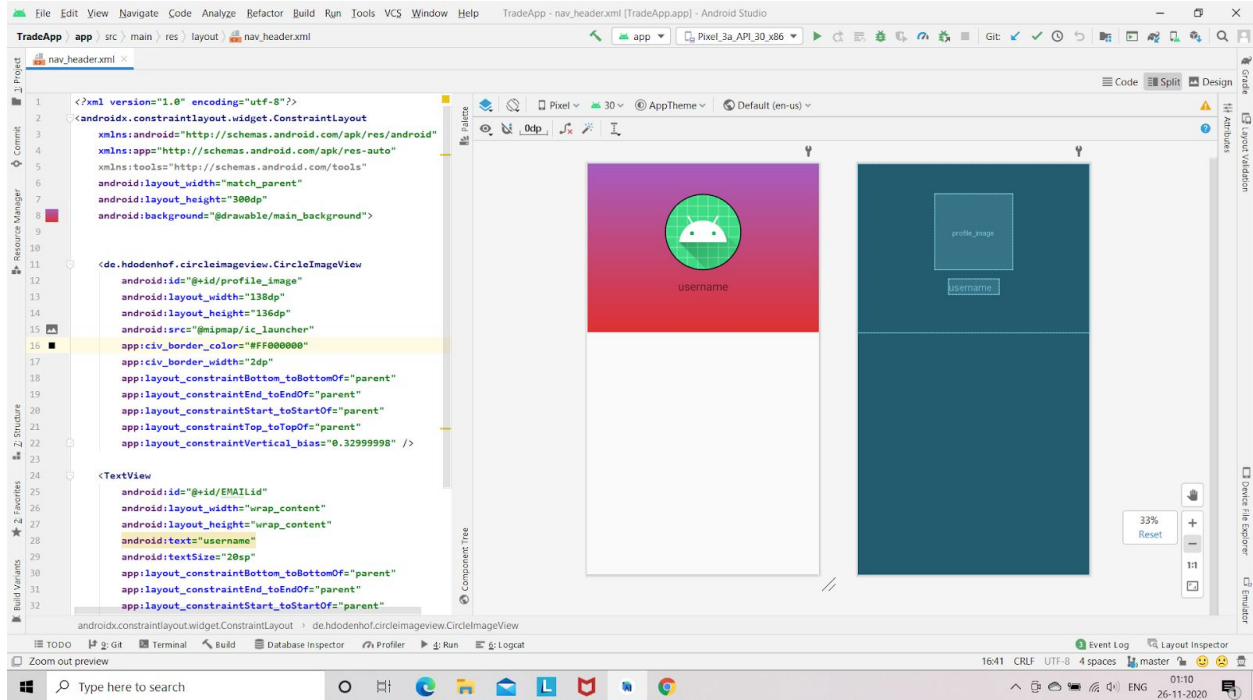




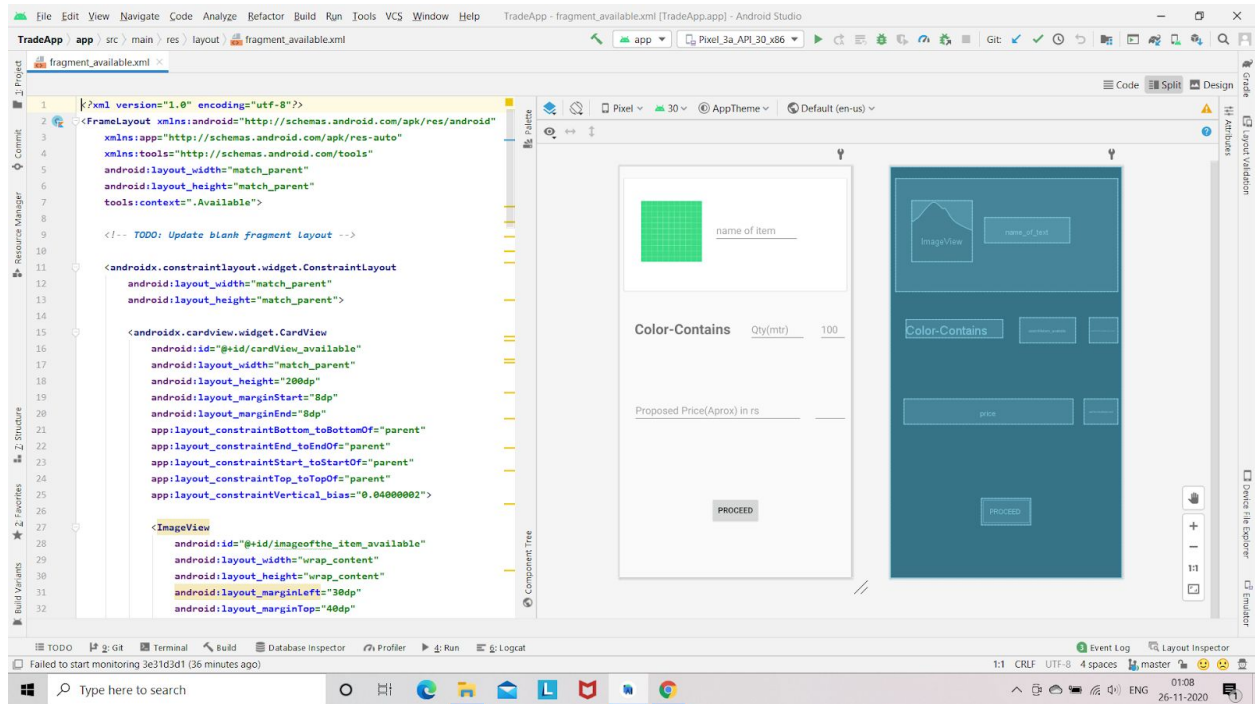
## 4.Item list page after login



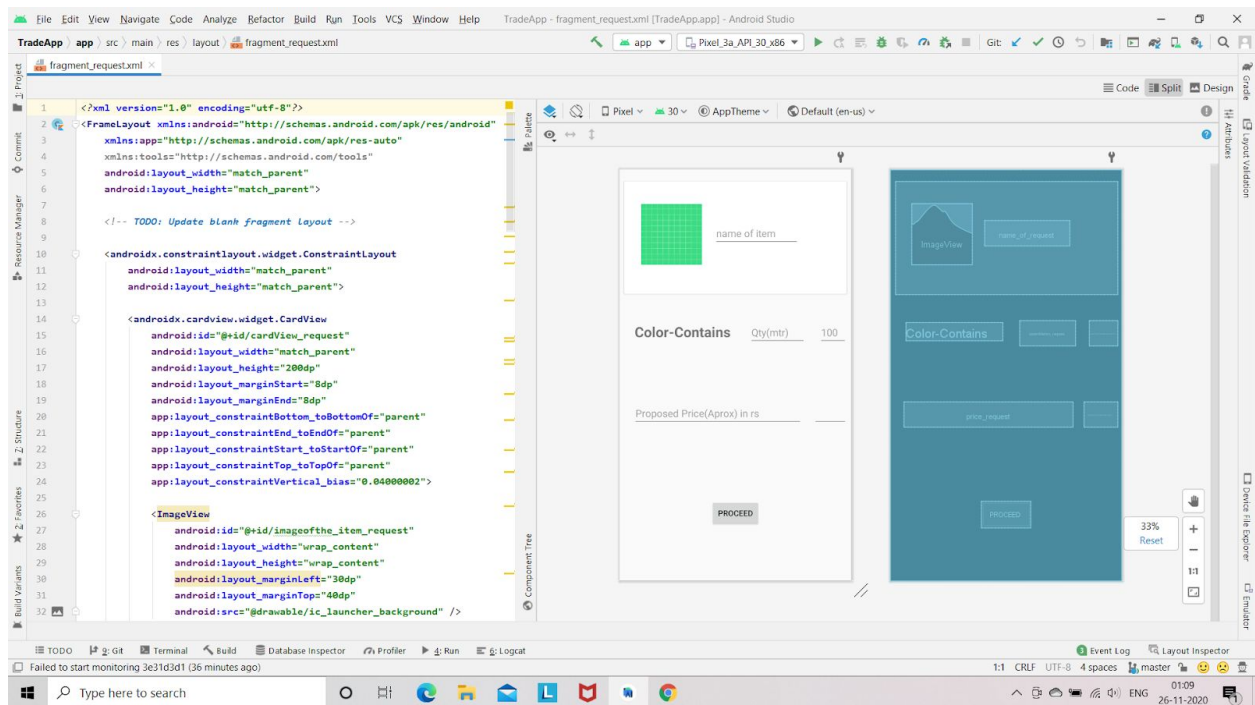
## 5.user info for navigation bar



## 6.Addition of items page



## 7. Request of items page



## **Scope:**

Online shopping is rising day by day in India. Because India is the country where computer user's are increasing day by day so the online shopping trends are also increasing. This project contains selling of online fabrics by various merchants for buyers by showing users different items and categories for fabrics which they may like and buy. As there are huge no of people who are merchants who first buys fabrics and converts them to wearables and sells them, this app is basically targeted for them only. As for now there is no such app and can be of huge success.

## **CONCLUSION**

After completing this project, I concluded that this project was the good opportunity to implement my information that I have learnt during my PROJECT. This project is more informative and more helpful for understanding the concept of the android app development. This project is enough to implement my concept. I can further try much harder to make much more efficient and useful app that can benefit to other.

## **Bibliography**

Following are the links from which all the information have been taken :

1. <https://developer.android.com/training/basics/index.html>

2. <https://www.youtube.com/>

3. <https://www.tutorialspoint.com/android/>

---

**YASH KUMAR SINGH**

**181500824**

**B.tech(CSE)**

**BALARAM SAMANTA**

**181500188**

**B.Tech(CSE)**