A PROJECT REPORT

ON

"TISTICS DELIVERY SYSTEM"

Submitted to

UNIVERSITY OF MUMBAI

BACHELOR'S DEGREE IN
COMPUTER ENGINEERING

BY

ANSARI SANA SABIR AMINA 16DCO49
PAL PRITESH ROOPLAL SUMAN 15CO29

UNDER THE GUIDANCE OF
PROF. JAVED KHAN SHEIKH

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
2018-2019

AFFILIATED TO

UNIVERSITY OF MUMBAI

A PROJECT II REPORT
ON

"TISTICS DELIVERY SYSTEM"

Submitted to
UNIVERSITY OF MUMBAI

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER ENGINEERING

BY

ANSARI SANA SABIR AMINA 16DCO49
PAL PRITESH ROOPLAL SUMAN 15CO29

UNDER THE GUIDANCE OF
PROF. JAVED KHAN SHEIKH

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

2018-2019
AFFILIATED TO

UNIVERSITY OF MUMBAI

Anjuman-i-Islam's Kalsekar Technical Campus

Department of Computer Engineering
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

CERTIFICATE

This is certify that the project entitled

"Tistics Delivery System"

submitted by

ANSARI SANA SABIR AMINA 16DCO49
PAL PRITESH ROOPLAL SUMAN 15CO29

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Engineering) at ***Anjuman-I-Islam's Kalsekar Technical Campus, Navi Mumbai*** under the University of MUMBAI. This work is done during year 2018-2019, under our guidance.

Date: / /

Prof. JAVED KHAN SHEIKH
Project Supervisor

Prof. KALPANA BODKE
Project Coordinator

Prof. TABREZ KHAN
HOD, Computer Department

DR. ABDUL RAZAK HONNUTAGI
Director

External Examiner

Acknowledgements

We would like to take the opportunity to express our sincere thanks to our guide **Prof. Javed Khan Sheikh**, Assistant Professor, Department of Computer Engineering, AIKTC, School of Engineering, Panvel for his invaluable support and guidance throughout our project research work. Without his kind guidance & support this was not possible.

We are grateful to him/her for his timely feedback which helped us track and schedule the process effectively. His/her time, ideas and encouragement that he gave is help us to complete our project efficiently.

We would like to express deepest appreciation towards **DR. ABDUL RAZAK HONNUTAGI**, Director, AIKTC, Navi Mumbai, **Prof.Tabrez Khan**, Head of Department of Computer Engineering and **Prof. Kalpana Bodke**, Project Coordinator whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped us directly or indirectly during this course of work.

Ansari Sana Sabir Amina

Pal Pritesh Rooplal Suman

Project I Approval for Bachelor of Engineering

This project entitled ***"Tistics Delivery System"*** by ***Ansari Sana Sabir Amina, Pal Pritesh Rooplal Suman*** is approved for the degree of ***Bachelor of Engineering in Department of Computer Engineering.***

Examiners

1. .....

2. .....

Supervisors

1. .....

2. .....

Chairman

.....

Declaration

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Pal Pritesh Rooplal Suman
15CO29

Ansari Sana Sabir Amina
16DCO49

ABSTRACT

In today's world of mobile phones and tablets humans want to conceal everything into their very own devices. Many manual and labour inclusive tasks have been replaced by machines and various interactive devices and so is shopping.when it comes to shopping, a customer is spoiled by many options available in the market. Some of the popular examples include Amazon, Flipkart, Snapdeal etc. A wholesale dealer on the other has has very limited options. Although there are platforms available for wholesale dealers are not as flexible as the ones available for the customers. In this modern era why not give wholesalers the privilege that a customer gets.Tistics Delivery System (TDS) provides a very concise and protected app-based environment to carry out their transactions. And as TDS grows and make its place in the market it will create job opportunities for many people . TDS is comprised of three apps viz. 1. The Customer App(For Customers) 2. The Vendor App(for vendors) 3. The Delivery Boy App(for delivery personnel) and one web-based admin panel to overview the entire system.

Keywords: TDS- Tistics Delivery System

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The wholesale market circuit is quite huge. Such wholesale transactions are mostly carried out independently without any support of external system. TDS will provide a platform wherein we wholesale dealers will be provided a platform where they can perform their transaction in a protected and reliable environnment.There are many people that rely on such wholesale dealers for delivery jobs. However there is no job security for such people. TDS will create a system where such jobless people will get a permanent jobs with a monthly basis salary. TDS will provide a structure to the wholesale dealing process.

1.1 Purpose

Our TDS will reduce the chaos in WD's life and provide a structure to their way of business. The conventional way in which the WD(Wholesale Dealer) and vendors carry out their transactions takes in a lot of time and human efforts. The livelihood of many labours depends on the WD as they offer temporary tasks to people such as transportation, loading and unloading of goods etc. TDS aims to remove the manual efforts done by WD and vendors and provide a permanent and decent source of income to such labours.

1.2 Project Scope

The app will be launched in a particular region so the vendors and the customers residing in area will making use of the vendor app and customer respectively. The delivery boy in the respective area will be appointed by us. For a kick start TDS will only trade Curtain fabrics gradually over the time other products will be added as the TDS stabilizes with the previous working of the systems. As TDS will be stable in a particular area the scope will increase and regular upgrades will be made to make the system better.

1.3 Project Goals and Objectives

1.3.1 Goals

To Provide an application based platform to the wholesale market in order to eradicate the traditional way of doing business which is based on trust, public relations and is too slow for 21 century.

1.3.2 Objectives

1. To replace the current way of wholesale marketing with an online marketing system which is based on an android application based interface. 2. To make the wholesale business more simple and transparent 3. To give permanent employments to labourers who work along side whole-sale dealers. 4. To provide smart delivery services to the wholesale dealers using travelling salesman problem to increase the overall pace of the business.

1.4 Organization of Report

In chapter 1: We have considered Project overview under which we have explained various important terminologies like Introduction of the project, Motivation for the project Problem definition, About the current system, Advantages over the current system, Goals, Objectives, Scope Application.

In chapter 2: We have discussed about various paper that we have referred for our project. We have mentioned the description, pros and cons and how to overcome the problem under every paper. Total of three paper have been referred.

In chapter 3: We have discussed about the requirement analysis under it we have consider about the requirement for the system, the requirement supporting for the OS of the software nd hardware.

In chapter 4: We can see the system design and architecture various diagram can be seen in this chapter which represents the software, diagrams including our system architecture, use case diagram, activity diagram, DFD diagram, data flow diagram,sequence diagram, etc.

In chapter 5 : We have seen the methodology, here we have explain the project in detail by dividing into modules. Various modules or priority based cab search are explained with the help of few diagram.

In chapter 6: We have discussed about the implementation details, the assumptions and dependencies, this part contains details of the implementation of methodology that we have discuss earlier.

In chapter 7: We have show the test cases and the result along with analytic discussion, this part consist of the result of the output of the project.

In chapter 8: We have concluded the whole project and future scope along with the limitation followed up by reference and chapter 9 consist of Appendix.

Chapter 2

Literature Survey

2.1 Food Ordering System

Food Ordering System with Delivery Routing Opti-mization Using Global Positioning System (GPS) Tech-nology and Google Maps

2.1.1 Advantages of Paper

a. The capability of a consumer to purchase products online or via telephone, and then has the purchased items to be delivered safely to her/his address is a value added by sellers.

b. Moreover, the customers usually do not have much information about the status of their order nor the position of the delivery staff.

c. The delivery service, one of the main problems is to find the shortest path between customers' addresses in order to deliver the product in reasonably short time, to save fuel-usage and to optimize the utilization of the vehicles and delivery personnel.(using Google Maps).

d. The routing problem that is related to the condition of food delivery service is called Traveling Salesman Problem (TSP). In TSP, the seller starts moving from his/her home-town and is required to visit several cities exactly one time before going back to his/her hometown with minimum total distance.

2.1.2 Disadvantages of Paper

a. There is no route optimization between multiple nodes.

2.1.3 How to overcome the problems mentioned in Paper

a. We are going to provide map with route optimization between multiple nodes.

2.2 Technical Review

The project is based on Data analytic. It consists of an app based interface for users and an web-based panel for admin to facilitate the maintenance of the system.

For Front-end design the following technologies were used
1. Marvel App
2.Android Studio

For Back-end design the following technologies were used
1.Node-js
2.MySql Database

2.2.1 Advantages of Technology

a. Node-js : Node is similar in design to, and influenced by, systems like Ruby's Event Machine or Python's Twisted. Node takes the event model a bit further. It presents an event loop as a runtime construct instead of as a library. In other systems there is always a blocking call to start the event-loop. Typically behavior is defined through callbacks at the beginning of a script and at the end starts a server through a blocking call like EventMachine::run(). In Node there is no such start-the-event-loop call. Node simply enters the event loop after executing the input script. Node exits the event loop when there are no more callbacks to perform. This behavior is like browser JavaScript — the event loop is hidden from the user.

b. MySql : MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo! and many more.

c. Android : Android is a mobile operating system developed by Google. It is based on a modified version of the Linux kernel and other open source software, and is designed primarily for touchscreen mobile devices such as smart-phones and tablets. In addition, Google has further developed Android TV for televisions, Android Auto for cars, and Wear OS for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.

2.2.2 Reasons to use this Technology

a. Node.js : The Node.js is more advantageous to the developers in comparison to its disadvantages. What's more important is the fact that it has extended the

area of JavaScript application and can be evidently used for both front-end as well as back-end servers. With the progress of time, more and more business organizations have adopted Node.js and have ended getting positive results.

b. Android : We have decided to use android studio cause it is better and more convienient than other softwares available in the market and of course better than eclispse. Android Studio is certainly a step ahead of Eclipse, which lost its position in less than a year as the main IDE for android application development and became died out. There has been a huge publicity around it among android app developers ever since Android Studio was announced in 2013, and without doubt AS meets up to nearly all expectations.

Chapter 3

Project Planning

3.1 Members and Capabilities

Table 3.1: Table of Capabilities

SR. No	Name of Member	Capabilities
1	Pritesh Pal	Problem Solving , Project Management,Risk Management
2	Sana Ansari	Problem Solving, Customer and Client Management,Risk Management

3.2 Roles and Responsibilities

Table 3.2: Table of Responsibilities

SR. No	Name of Member	Role	Responsibilities
1	Pritesh Pal	Team Leader	API, Database
2	Sana Ansari	Motivator	Front-end Developer, Database

3.3 Assumptions and Constraints

If the database crashes or if the server crashes then the system might gave to be shutdown for some time until the repair is done. The users should provide proper results in order to get accurate results. The algorithm to find the shortest path uses google maps api so if the routes displayed are not updated regularly the algorithm will work accordingly and provide a wrong route.

3.4 Project Management Approach

The scrum approach includes assembling the project's requirements and using them to define the project. You then plan the necessary sprints, and divide each sprint into its own list of requirements. Daily scrum meetings help keep the project on target as do regular inspections and reviews. At the end of every sprint, you hold a sprint retrospective to look for ways to improve the next sprint. The process looks something like this:

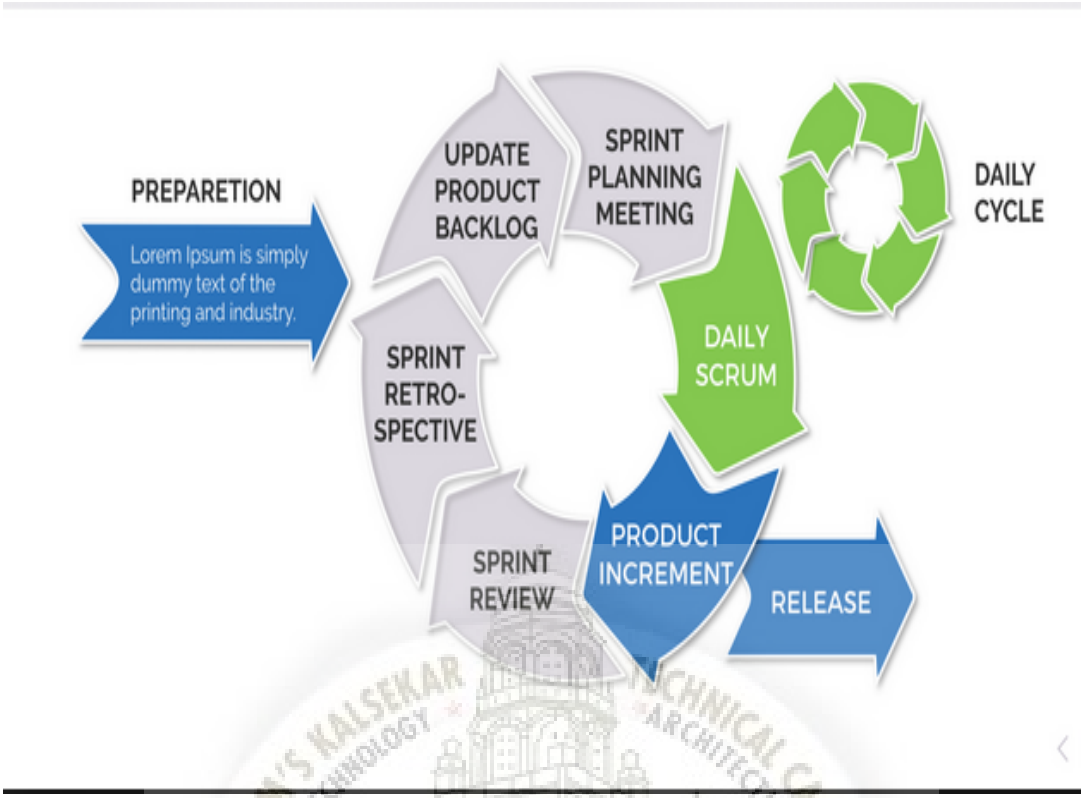


Figure 3.1: Scrum Model

Within each sprint, the development team builds and tests a functional part of the product until the product owner accepts it and the functionality becomes a potentially shippable product. When one sprint finishes, another sprint starts. Scrum teams deliver product features in increments at the end of each sprint. A product release occurs at the end of a sprint or after several sprints

You use the tenets of inspection and adaptation on a daily basis as part of a scrum project:

1.During a sprint, you conduct constant inspections to assess progress toward the sprint goal, and consequentially, toward the release goal.

2.You hold a daily scrum meeting to organize the day by reviewing what the team completed yesterday and what it will work on today. Essentially, the scrum team inspects its progress toward the sprint goal.

3.At the end of the sprint, you use a retrospective meeting to assess performance and plan necessary adaptations.

3.5 Ground Rules for the Project

1.We treat each other in the team with respect and dignity.

2.We intend to develop better relationships to enhance our trust and have a better communication.

3. We value constructive feedback. We will avoid being defensive and give feedback in a constructive manner.

4. As a team members, we will pitch in to help wherever necessary to help solve the problems.

5. Additional meetings can be scheduled to discuss critical issues or tabled we should create and adopt written notes for help. No responsibilities be assigned unless the person who is being assigned the job accepts it.

3.6 Project Budget

NodeJs : Free Open Source

Android : Free Open Source

Firebase SDK : Free Open Source

3.7 Project Timeline

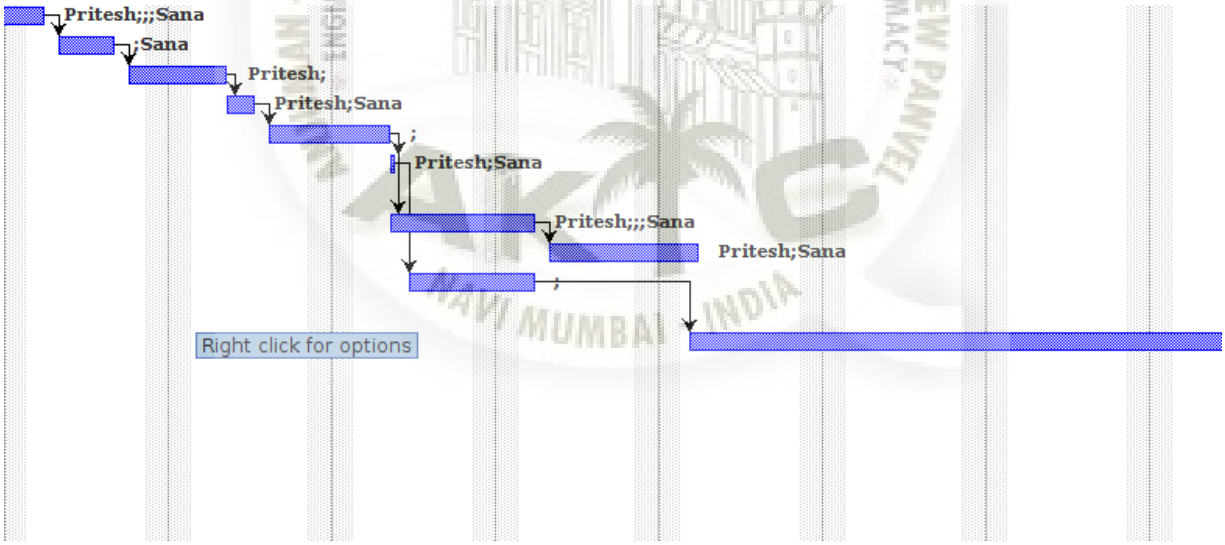


Figure 3.2: Project Timeline

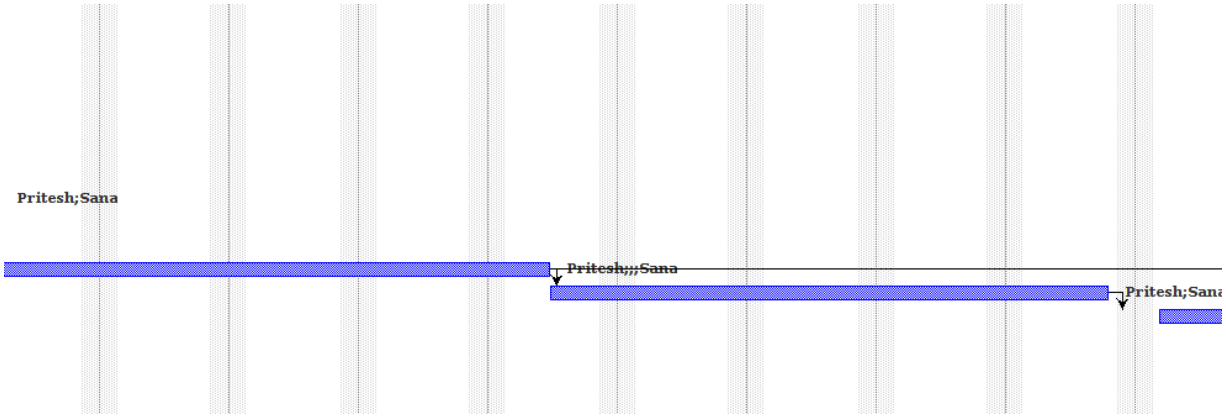


Figure 3.3: Project Timeline

Chapter 4

Software Requirements Specification

4.1 Overall Description

4.1.1 Product Perspective

In today's world of mobile phones and tablets humans want to conceal everything into their very own devices. Many manual and labour inclusive tasks have been replaced by machines and various interactive devices and so is shopping.when it comes to shopping, a customer is spoiled by many options available in the market. Some of the popular examples include Amazon, Flipkart, Snapdeal etc. A wholesale dealer on the other has has very limited options. Although there are platforms available for wholesale dealers are not as flexible as the ones available for the customers. In this modern era why not give wholesalers the privilege that a customer gets.Tistics Delivery System (TDS) provides a very concise and protected app-based environment to carry out their transactions. And as TDS grows and make its place in the market it will create job opportunities for many people .

4.1.2 Product Features

The apps involved in TDS System will operate on Android Operating System.Using the apps , the WD can get a list of all the available vendors to him and can have a look at their repositories. A WD can create one or multiple orders according to his needs whenever he wishes to. Vendors on the other hand will be notified on their apps about the orders.They can Update their repositories on to their profiles.When a Vendor accepts a WD's order then the transaction is completed with the help of the Delivery boys provided by the TDS thus reducing the burden of transportation on the shoulders of both vendors and WD's. The delivery boy's provided with a delivery app will be assigned a task on a daily basis.So the delivery personnel working with TDS will be assured with a permanent source of income. A lot of manual work is terminated and a huge amount of time will be saved using TDS.

4.1.3 User Classes and Characteristics

There are basically 3 apps in TDS which will operation android Operating System

1. Vendor App
2. Delivery Person App
3. Customer App

Vendor App:-

1. Create Order: Needed in case customer calls them to place an order. Specifies the quantity of cloth from different types, choose which customer and location to deliver, and then submit their request.
2. View Order: See all the order, and filter by status.
3. Accepting/Declining Order Request: A vendor can do two things: Accept order, Decline order(needs to provide reason).
4. Generate Invoice: A vendor can generate an invoice for the created order with GST.

Customer App: List products: Customers can see all the products with options to filter(basedon vendor, price, quality, etc.).

1. Create Order: Customer can create an order with quantity and type of cloth.
2. View Order: See all the order, and filter by status (Accepted, Preparing, Delivered).
3. Download Invoice: A vendor can download aninvoice for an order.
4. Live Tracking: Customer should ableto live track the order(like swiggy).
5. Online Payment : This will allow customer to pay for his order(Note customer can select Cash on delivery option also) 3.Delivery Person App:

4.1.4 Operating Environment

The entire system is Based on three apps which works on android operating system. The working of the apps and the data sharing between them is handled with the help of a web-based panel.

4.1.5 Design and Implementation Constraints

For Mobile Applications
Operating System : Android(Kitkat bean and above)
Ram: 300MB
Storage:800MB
For Web-pannel Opearting System : Windows/Linux Ram: 2GB Storage: minimum of 20GB available on hard drive Internet : Upto 4mb/sec Monitor Resolution : 1024*768

4.2 System Features

Our System Creates an online app-based interface for customers who wish to purchase and sell items in wholesale. We also provide delivery services to our customers which makes the wholesale online marketing a lot easier for both buyer and seller.

4.2.1 System Feature

Description and Priority

1.The Vendor App: The wholesale vendor will upload the images and details of the products which hes selling .ie he will provide a brief details of his/her repository which will be visible to the customer. The vendor gets to accepts of reject the order of the customer.

2.The Customer App: The profiles of vendors which includes their respective repositories are visible to the customers.The customer will then make an order according to his/her need.The customer has the power to cancel the order at any point of time.System also provides live tracking of orders.

3.The Delivery Boy App : As soon as the customer makes an order and that order is accepted by the vendor, the app will notify the delivery boy about the transaction. The delivery personnel will then be provided with optimal path from source to destination which cover all the orders.

4. The Master Web-panel: The panel is responsible for monitoring the smooth working of the three apps. if any kind of mishandling happens it will be taken care of with the help of this panel

Stimulus/Response Sequences

1. The Delivery personnel appointed by the system will log-in to the system using the id's and password provided to them. Tasks will be provided to them accordingly.

2.The Vendors will also be verified and id's will be given to them using which they will log-in themselves and upload their repositories.

3. The customers are free to make their accounts and buy anything that pops on screen.They are even provided with a live tracking feature of their orders.

Functional Requirements

These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary.

4.3 External Interface Requirements

4.3.1 User Interfaces

The systems consists of three apps each of which provides a very simple and interactive User-Interface. The customer is allowed to create a basic profile of himself/herself.Customer enjoys features like cart, sorting and filtering, live tracking and is also provided with a search bar. The vendor on his app can create an order, add items to his repository.Vendor's also have live tracking feature and profile facilities. The delivery boy on the other hand has his profile already created. He gets an optimal path which covers all the orders assigned to him and by a single touch of finger he can get all the details of the particular consignment like name,address,phone number of the person the consignment belongs to.

4.3.2 Hardware Interfaces

No hardware interfaces required.

4.3.3 Software Interfaces

Operating systems : Android

4.3.4 Communications Interfaces

Not Required.

4.4 Nonfunctional Requirements

4.4.1 Performance Requirements

The System must be interactive and the delays involved must be less. When we connecting to the server the delay is because the data is stored or manage online very securely and safely. the data is reliable to the user to see this data very correctly.

4.4.2 Safety Requirements

The data that use for implementation which is concerned with the possible loss or harmful use of the data. The data stored online is very secure because these data is acess by only authorised user by providing username and password to the webapp. The external policies and safety isuue that the product design must be satisfied.

4.4.3 Security Requirements

The server on which the online data is stored will have its own security to prevent unauthorized write/delete access. The Server will be handled by an authorized Tistics member. The PC on which the database resides will have its own security.

Chapter 5

System design

5.1 System Requirements Definition

In this project we are trying to ease the business done by the wholesale sealers and the workers who are depended on them for daily wages. Our system will provide an app-based platform where vendor will upload all of its product on the app. The customer can view different vendors and their products and purchase according to his/her needs in bulk. Our systems also provides transport services which will use an optimal path from source to destination saving resources and an ample amount of time.

5.1.1 Functional requirements

Apps: They will serve as an interface for the customer,vendor and the delivery boy. Using these apps the vendor will create their repositories which will be visible to the customer. When the customer confirms an order then the delivery bot will be notified about the delivery and using the optimal path visible on the app the delivery will be completed.

Use-case Diagram

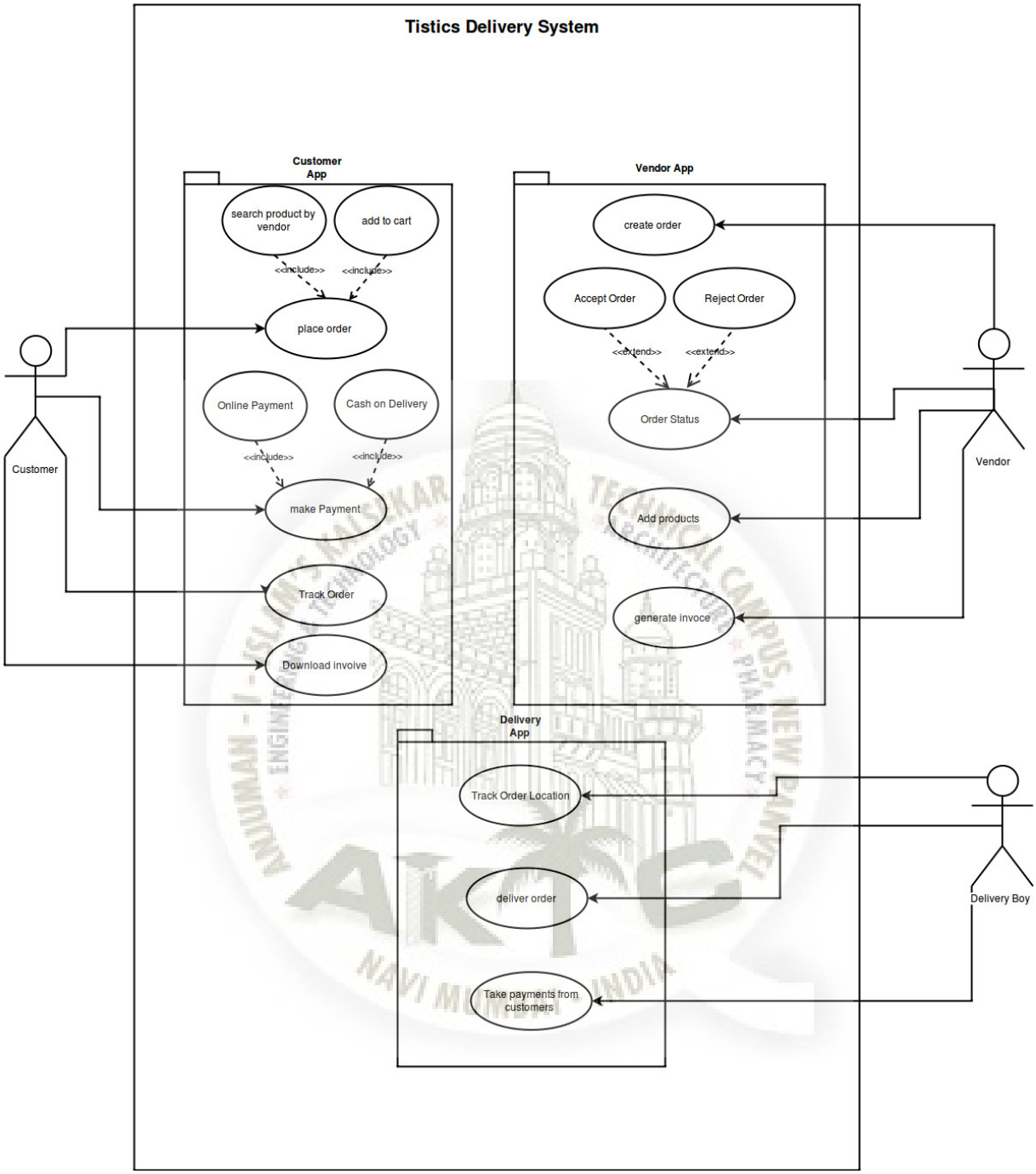


Figure 5.1: Use-Case Diagram

There are three users Customer, Vendor and Delivery Boy.
Customer can login with their respective credentials and can create order and can live track their order.
Vendor can upload their respective products and can accept or reject orders requested

by customer.
Delivery boy can access Map where he have to deliver orders with shortest path.

Data-flow Diagram

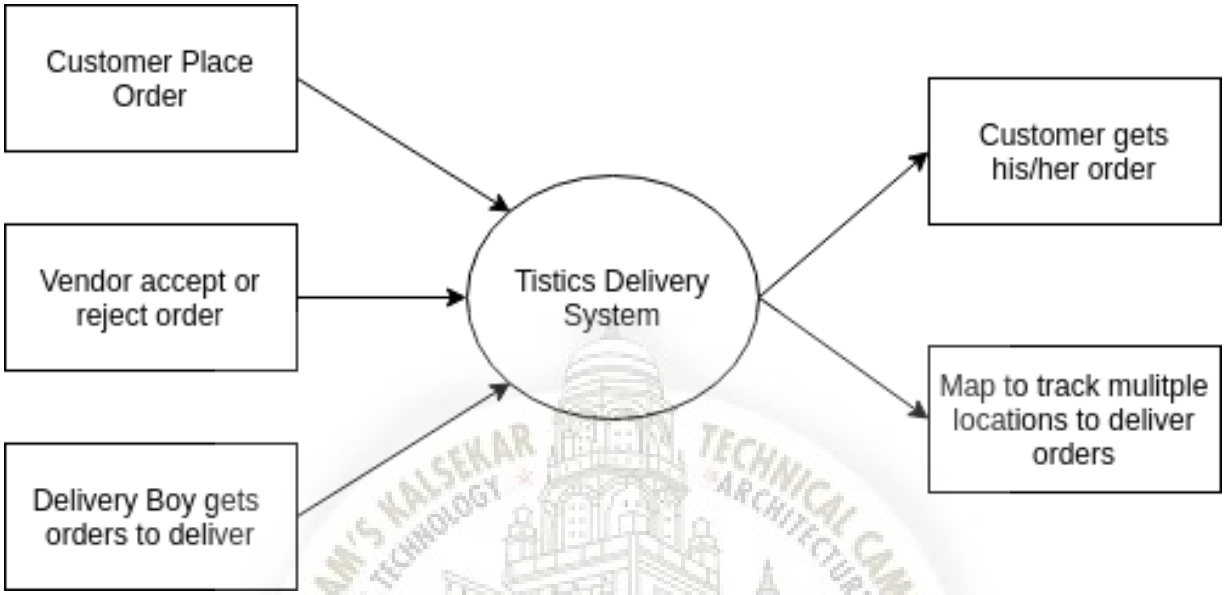


Figure 5.2: Level 0 DFD

In DFD Level 0 entire process is specified with its input and respective output. There are three inputs can be given from Customer, Vendor and Delivery Boy.

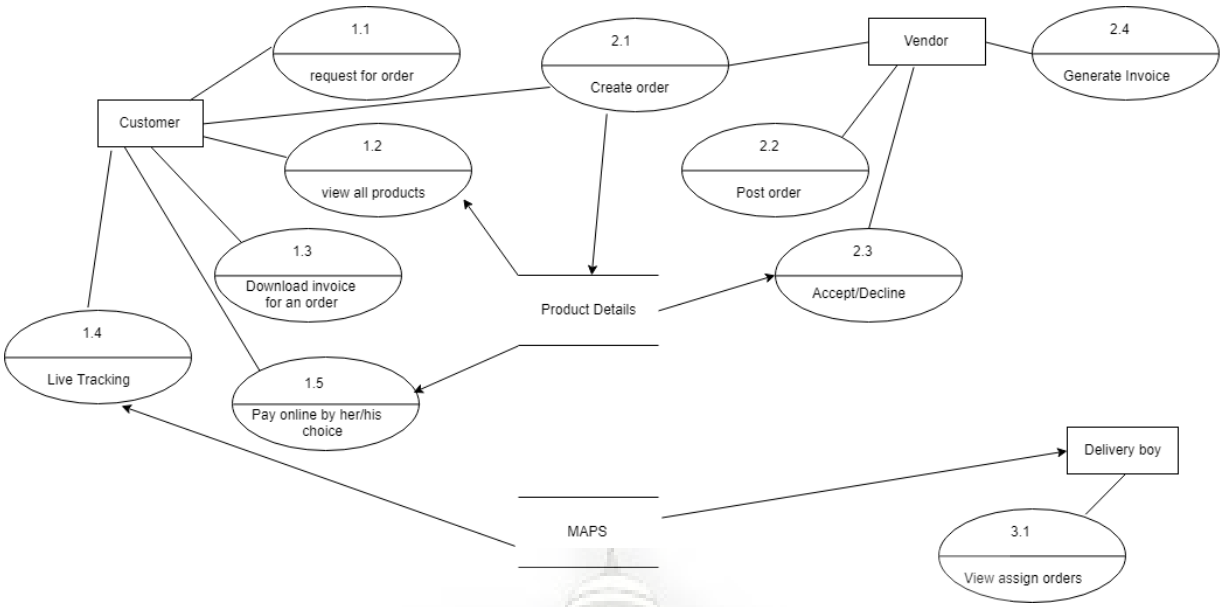


Figure 5.3: Level 1 Data-flow Diagram

In Level 1 DFD level 0 DFD is expanded, the different entities like Customer, Vendor and Delivery Boy tasks and their respective datastore.Customer can request for order, view his/her orders and live track their orders.
Vendor can upload products, create orders and accept and decline order request.
Delivery Boy can view maps for shortest distance between multiple locations.

5.1.2 System requirements (non-functional requirements)

These are non-functional system properties such as availability, performance and safety etc. They define functions of a system, services and operational constraints in detail.

5.2 System Architecture Design

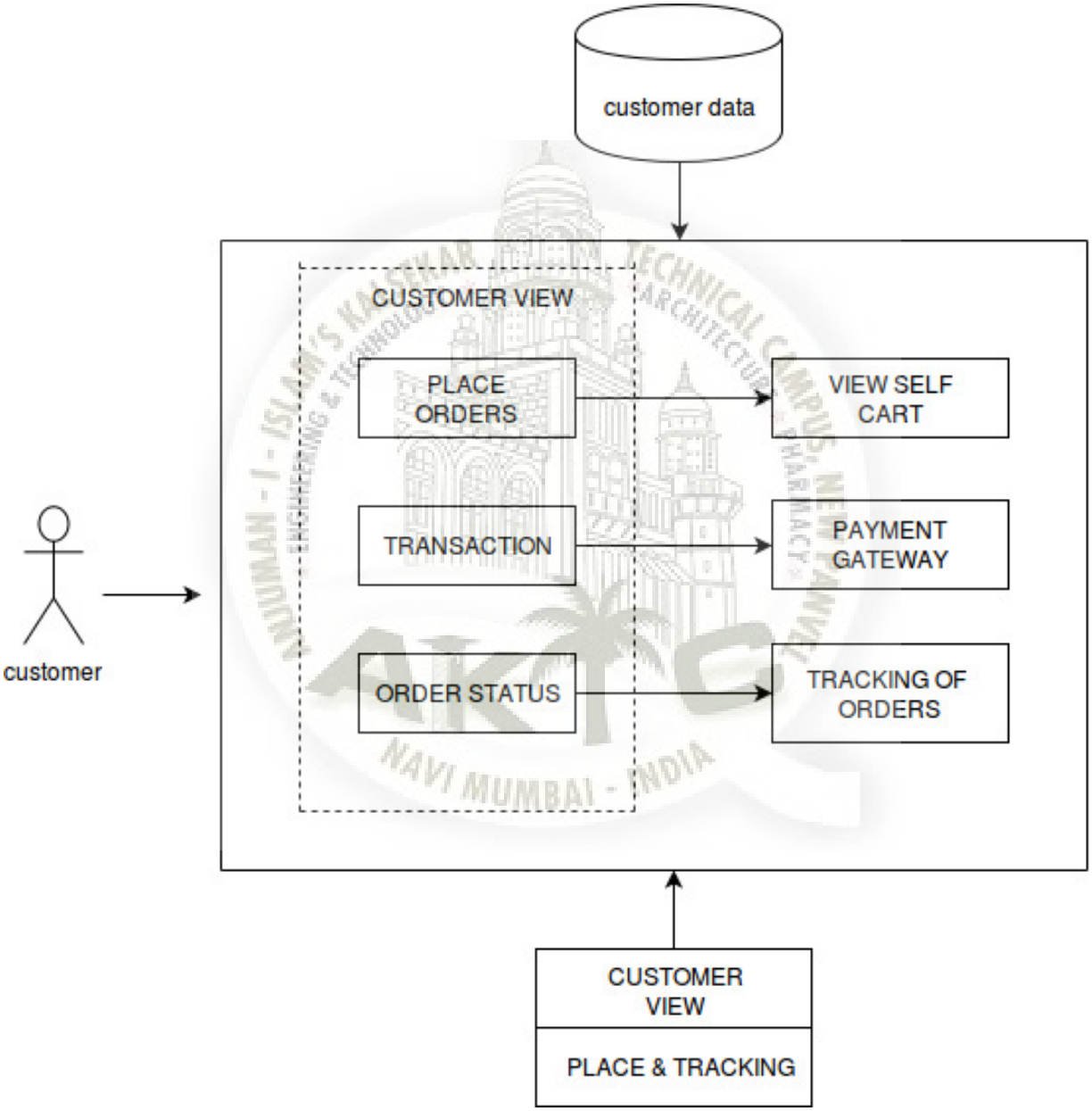


Figure 5.4: Customer App Architecture

Customer App:

List products: Customers can see all the products with options to filter(based on vendor, price, quality, etc.).
Create Order: Customer can create an order with quantity and type of cloth. View Order: See all the order, and filter by status (Accepted, Preparing, Delivered).
Download Invoice: A vendor can download an invoice for an order.
Live Tracking : Customer should able to live track the order(like swiggy)
Online Payment : This will allow customer to pay for his order(Note customer can select Cash on delivery option also)

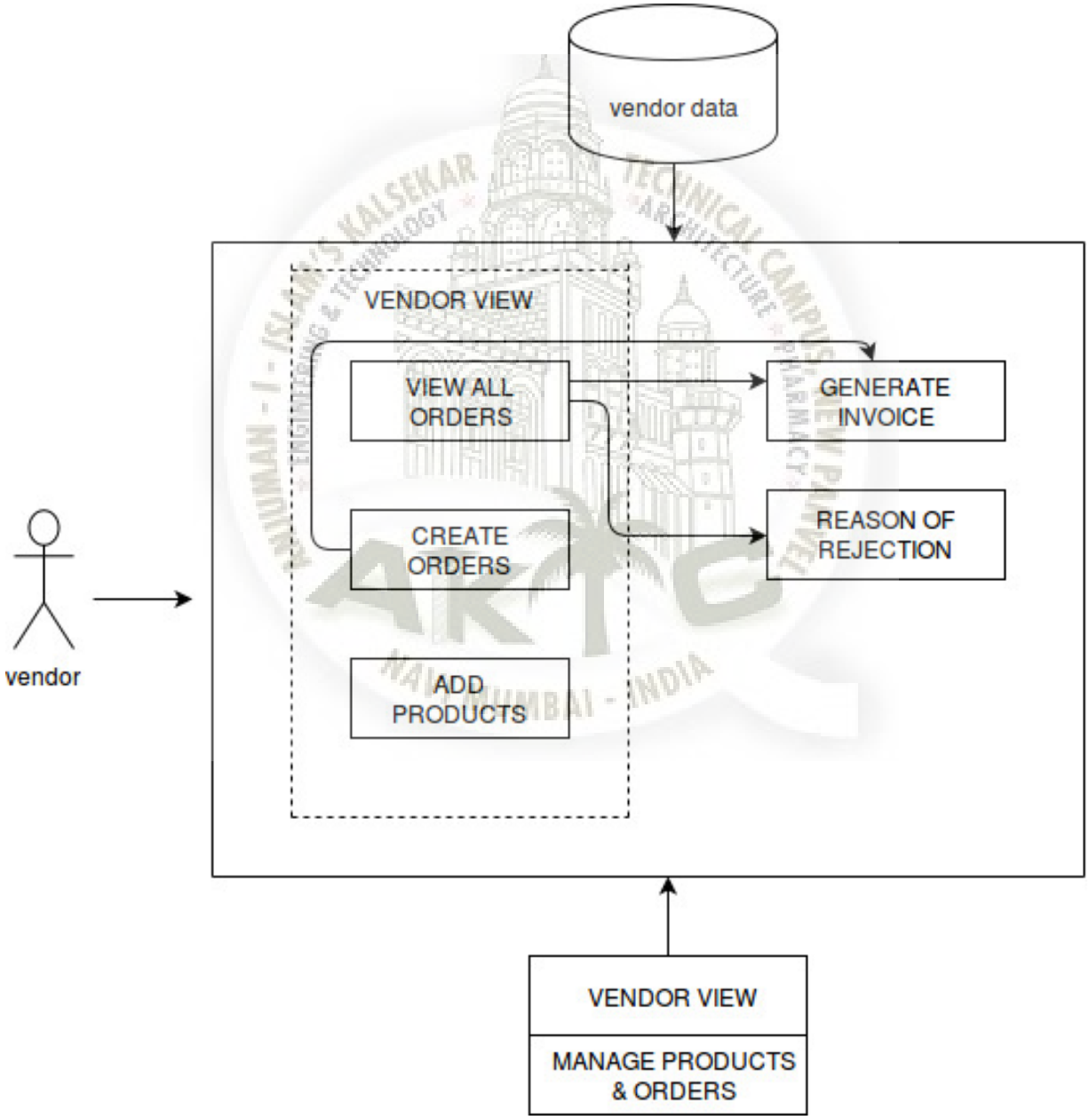


Figure 5.5: Vendor App Architecture

Vendor App:

Create Order : Needed in case customer calls them to place an order. Specifies the quantity of cloth from different types, choose which customer and location to deliver, and then submit the request.
View Order: See all the order, and filter by status.
Accepting/Declining Order Request: A vendor can do two things: Accept order, Decline order(needs to provide reason).
Generate Invoice: A vendor can generate an invoice for the created order with GST.

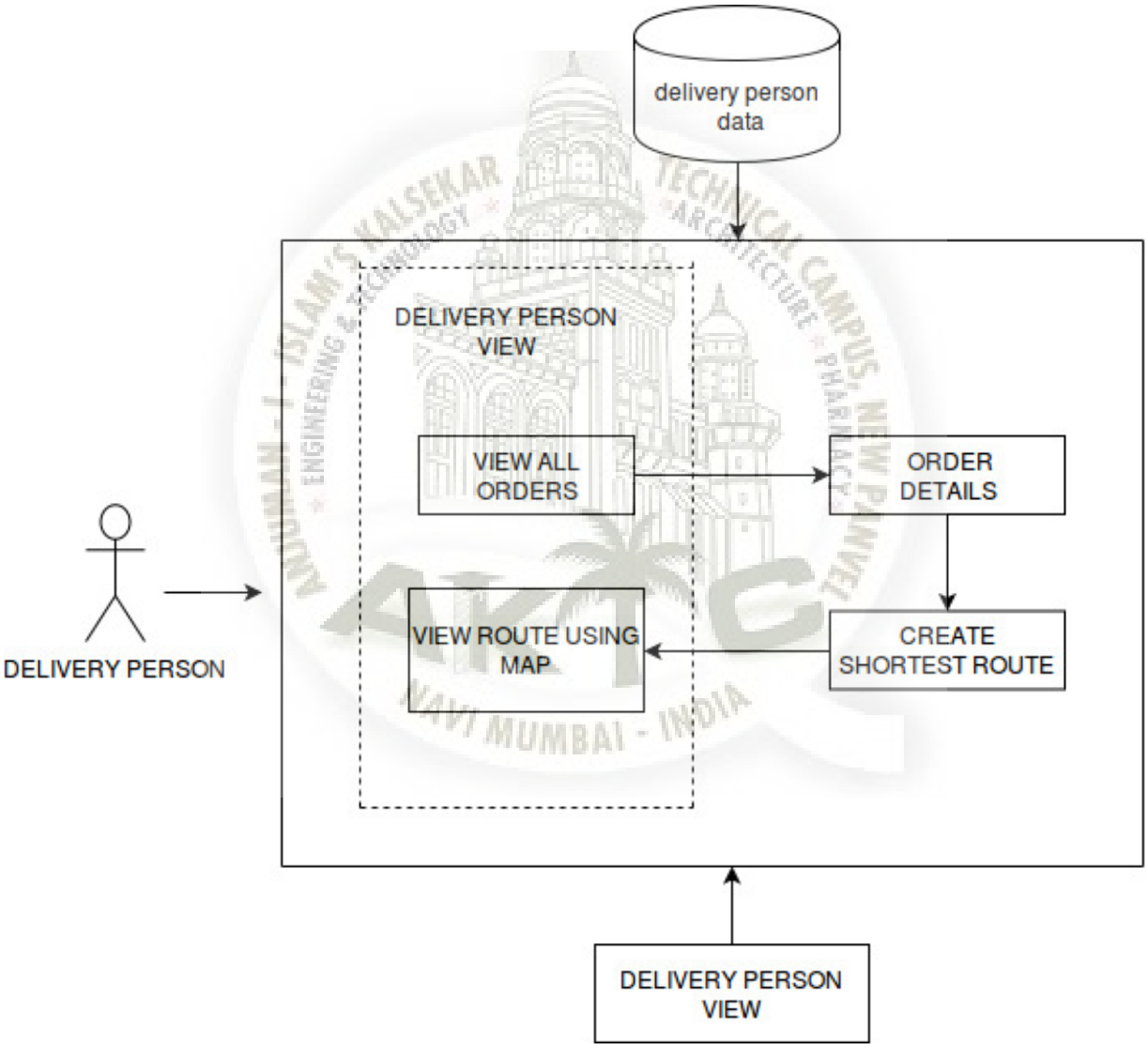


Figure 5.6: Delivery App Architecture

Delivery Boy App:

View the assign order to that delivery boy.
View in the maps direction to that shop(like ola/uber).

5.3 Sub-system Development

There are total three main modules in system architecture namely Customer, Vendor and Deliver Boy. All the modules will be briefly described further.

5.3.1 Customer Module

List products: Customers can see all the products with options to lter(based on vendor, price, quality, etc.). Create Order: Customer can create an order with quantity and type of cloth. View Order: See all the order, and lter by status (Accepted, Preparing, Delivered). Download Invoice: A vendor can download an invoice for an order. Live Tracking : Customer should able to live track the order(like swiggy) Online Payment : This will allow customer to pay for his order.

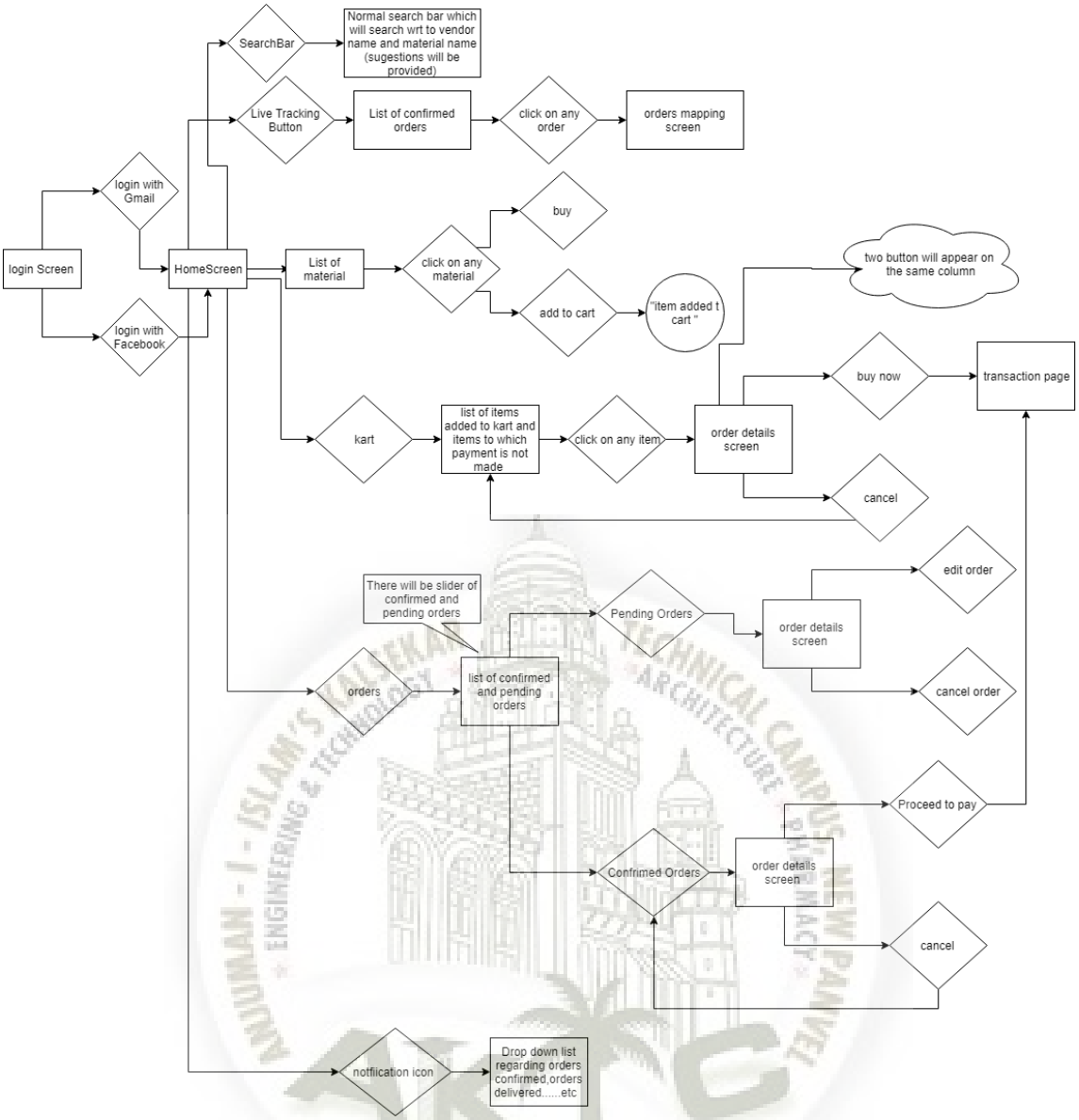


Figure 5.7: The Customer App Flow Diagram

5.3.2 Vendor Module

Create Order : Needed in case customer calls them to place an order. Species the quantity of cloth from different types, choose which customer and location to deliver, and then submit the request. View Order: See all the order, and lter by status. Accepting/Declining Order Request: A vendor can do two things: Accept order, Decline order(needs to provide reason). Generate Invoice: A vendor can generate an invoice for the created order with GST.

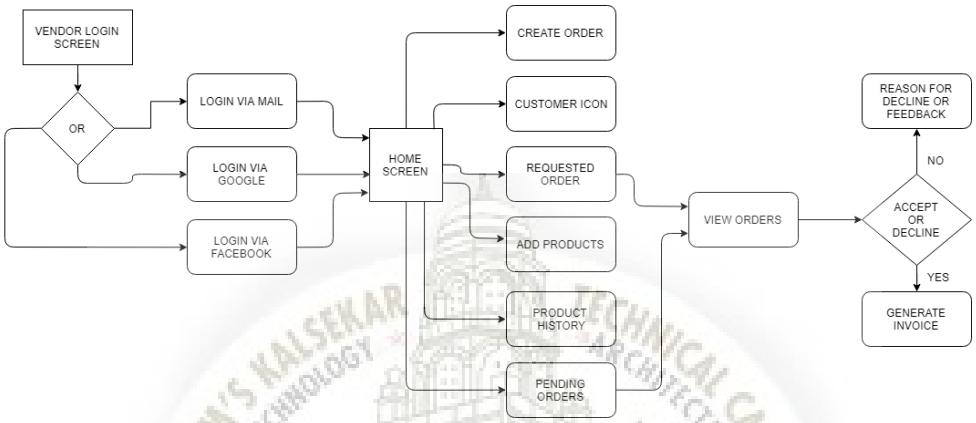


Figure 5.8: The Vendor App Flow Diagram

5.3.3 Delivery Boy Module

View the assign order to that delivery boy. View in the maps direction to that shop(like ola/uber).

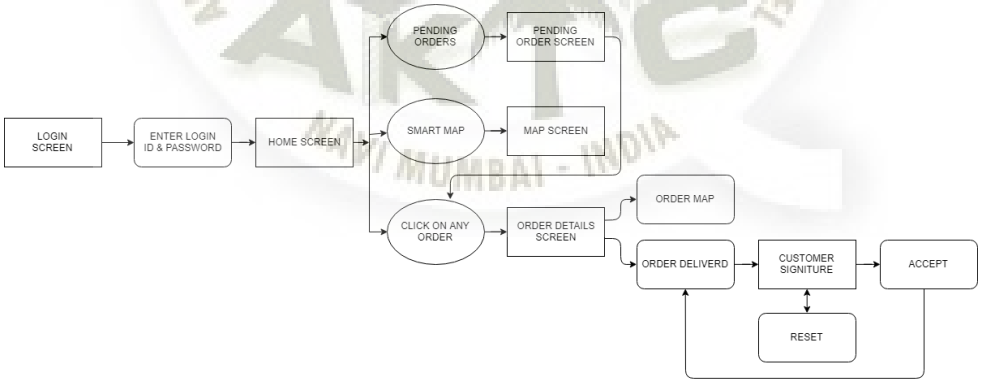


Figure 5.9: The Delivery-Boy App Flow Diagram

5.4 Systems Integration

Inorder to ensure the smooth working of the apps we must make sure that all the required modules in the app should interact with the web panel to mange the registration and authentication tokens.Google api should be integrated with the google maps api.

5.4.1 Class Diagram

There are basically four main classes viz. customer,product, vendor, delivery Boy and Maps API. The attributes of these classes and functions performed by them are as follows

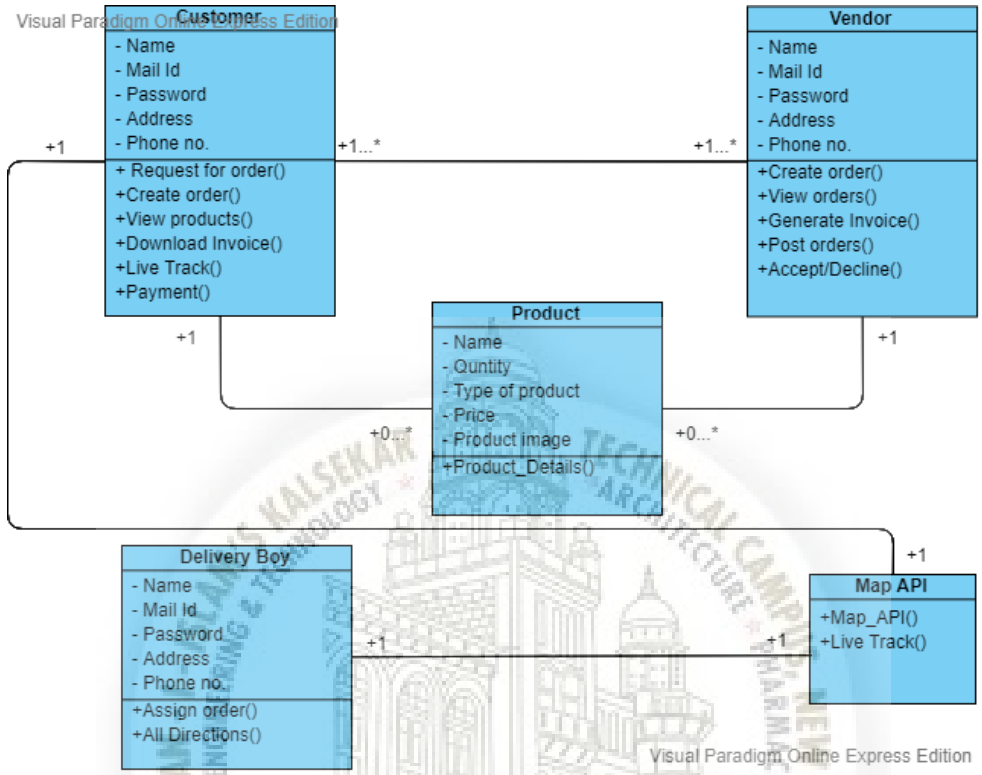


Figure 5.10: Class Diagram

5.4.2 Sequence Diagram

The customer first requests for an order. The Order is accepted or rejected by vendor. The vendor then proceeds by accepting the request and then the order details is forwarded to delivery boy. The detailed sequence is given in the diagram below :

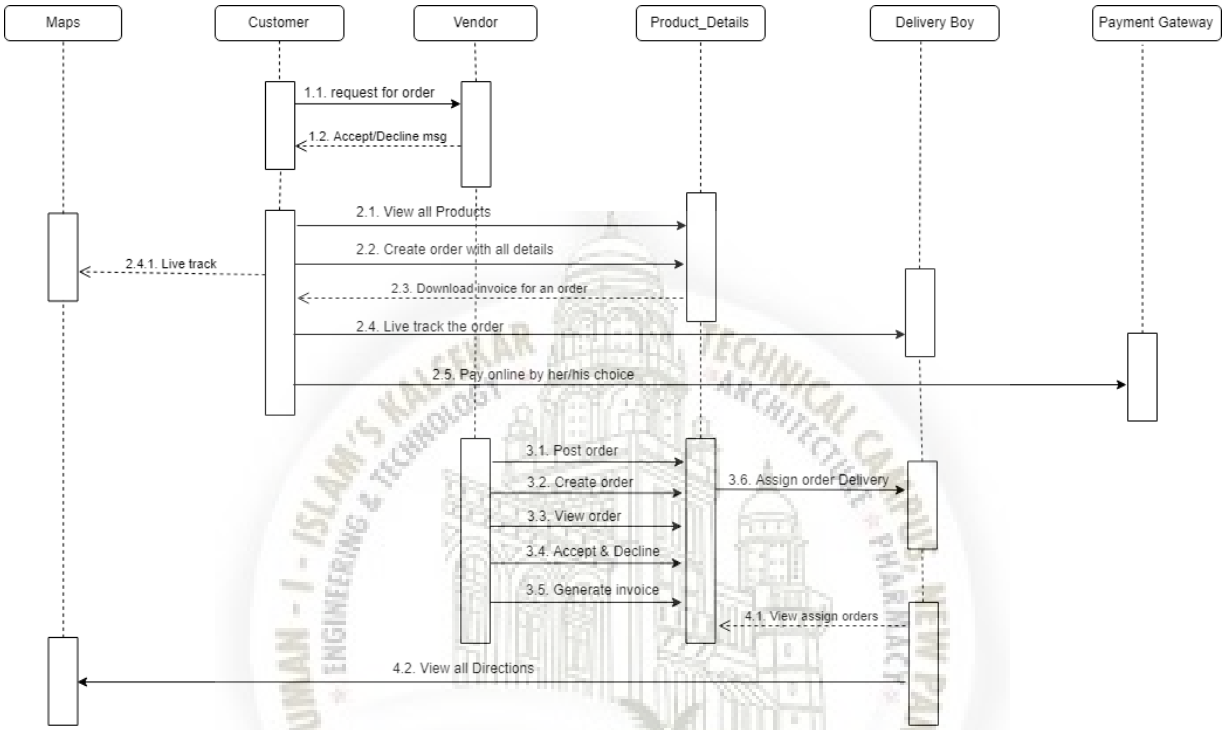


Figure 5.11: Sequence Diagram

5.4.3 Component Diagram

In component diagram it helps to understand the each module in better way forming its integrated sub module likewise shown in the diagram.The major components of the projects are Customer,Vendor,Product List,Payment Gateway,Google-map API,Product,Delivery Boy and some other which are mentioned in-detail in the figure below along with the interaction between them.

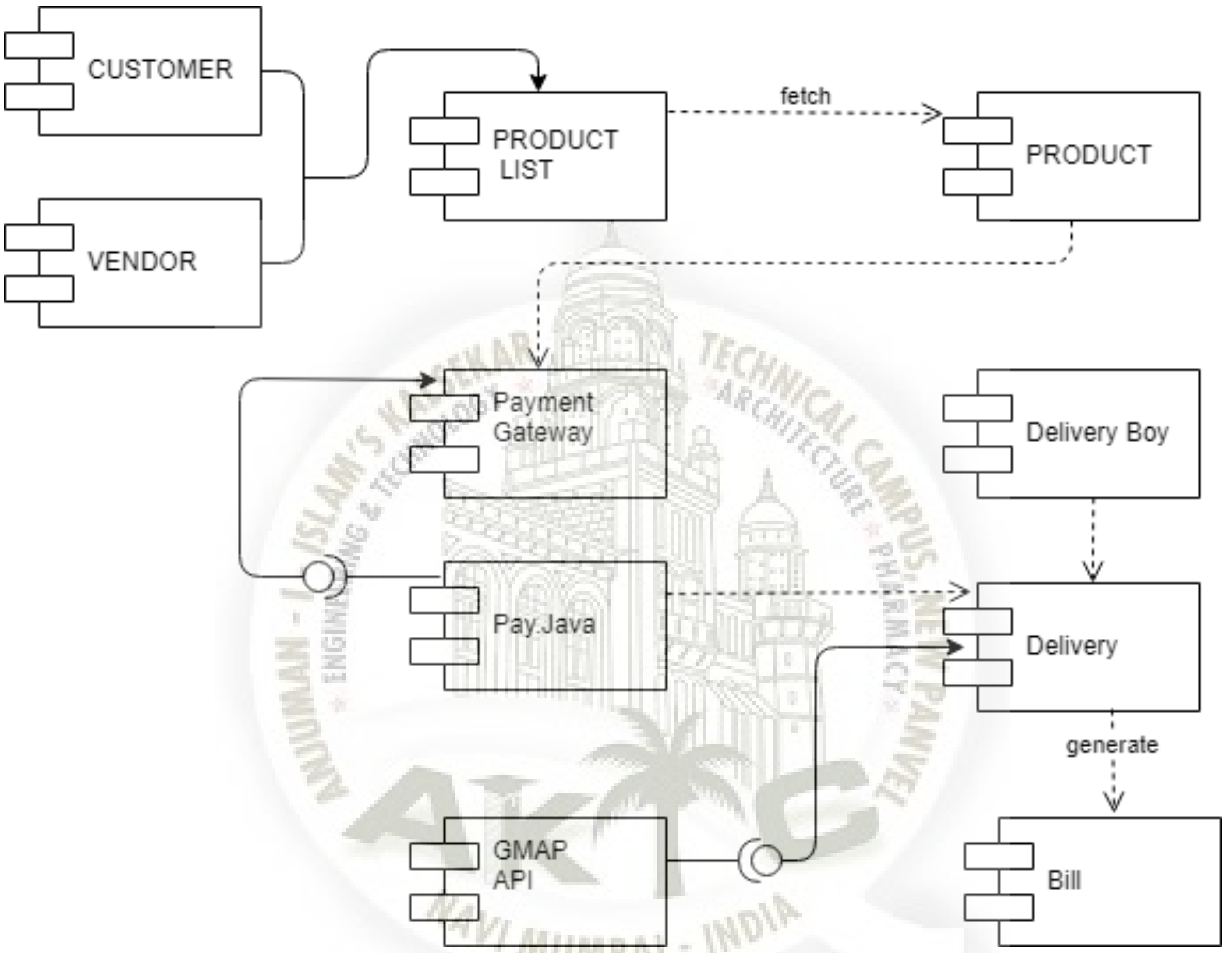


Figure 5.12: Component Diagram

5.4.4 Deployment Diagram

The Deployment diagram highlights the modules of the products and explains the interaction of the applications with the application server, database server and payment gateway.

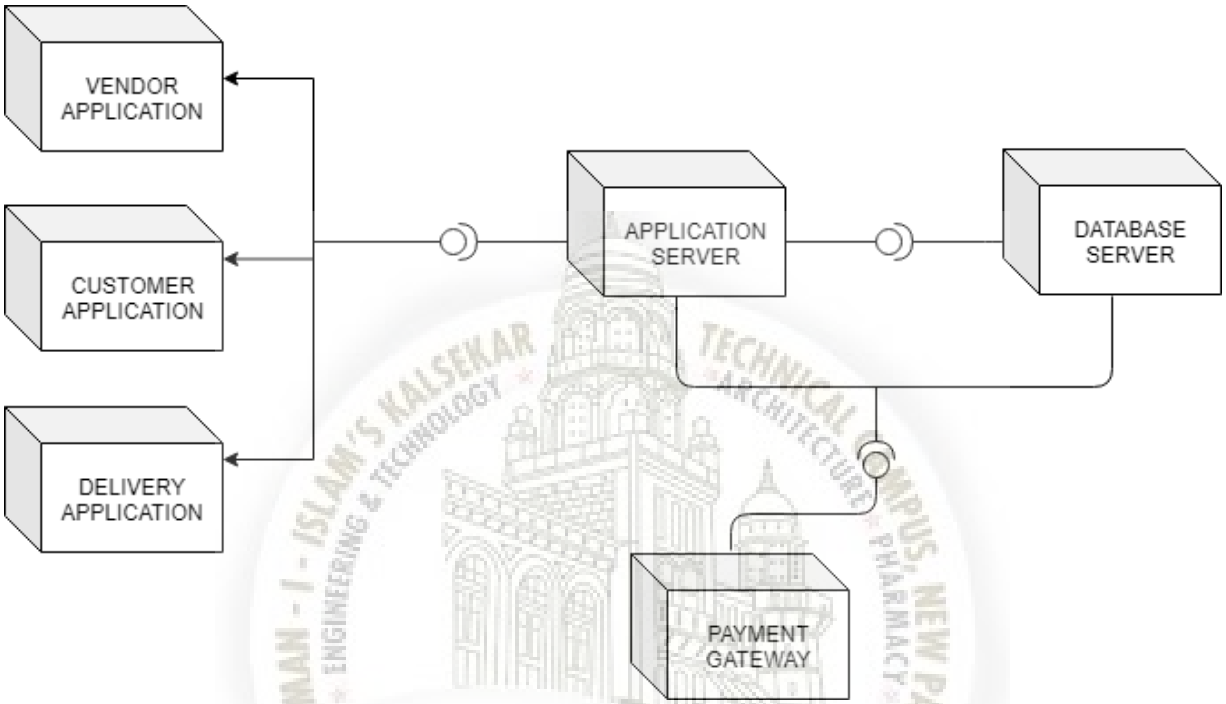


Figure 5.13: Deployment Diagram

Chapter 6

Implementation

6.1 Sign Up

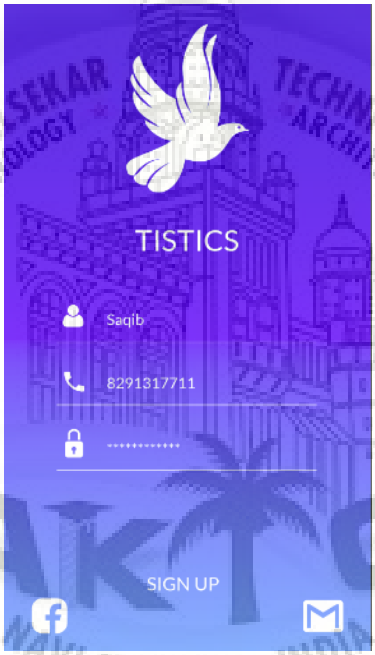


Figure 6.1: Sign Up

```
package com.tisticswebcos.tistics_signup;

import ...;

public class SignupActivity extends AppCompatActivity {

    private TextInputLayout user_name;
    private TextInputLayout phone_no;
    private TextInputLayout user_password;
    private TextView login_link;
    private Button signup_btn;
    String name,phone,password,token,  recvdToken;
    DataBaseHelper dataBaseHelper;
    RequestQueue requestQueue;
    StringRequest signupRequest;
    static int count = 0;

```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        FirebaseApp.initializeApp(this);
        dataBaseHelper = new DataBaseHelper(this);
        setContentView(R.layout.activity_signup);
        requestQueue = Volley.newRequestQueue(this);
        user_name = findViewById(R.id.text_input_name);
        phone_no  = findViewById(R.id.text_input_phone);
        user_password = findViewById(R.id.text_input_password);
        signup_btn = findViewById(R.id.btn_signup);
        login_link = findViewById(R.id.link_login);

        FirebaseInstanceId.getInstance().getInstanceId().addOnCompleteListener(
            new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if(task.isSuccessful()){
                    token = task.getResult().getToken();
                    Log.d("Token",token);

                }
                else{
                    Log.d("Token",task.getException().getMessage());
                }
            }
        });

        signup_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                validate();

            }
        });
        login_link.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getApplicationContext(),User_Profile.
                    class);
                startActivity(intent);
                overridePendingTransition(R.anim.slide_in_right,R.anim.
                    slide_out_left);

            }
        });




    }

    public void validate() {
        if(!validate_UserName() || !validate_Phone() || !validate_Password()) {
            AlertDialog alertDialog = new AlertDialog.Builder(SignupActivity.
                this).create();
            alertDialog.setTitle("Incorrect Values");
            alertDialog.setMessage("enter correct values");

            alertDialog.setButton("OK", new DialogInterface.OnClickListener() {
                @Override
```

```
                public void onClick(DialogInterface dialogInterface, int i) {

                }
            });
            alertDialog.show();
            Toast.makeText(getApplicationContext(),"Enter correct values ",Toast
                .LENGTH_SHORT).show();

        }
        else {


            doSignUp();


            Toast.makeText(getApplicationContext(), "Successfully Registered ",
                Toast.LENGTH_SHORT).show();

        }
    }
    public void doSignUp() {

        Toast.makeText(getApplicationContext(), "In Do Signup ", Toast.
            LENGTH_SHORT).show();

        signupRequest = new StringRequest(Request.Method.POST, URLHelper.signUp,
            new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {

                Log.d("Signup Response", response);
                if (response.equals("true")) {

                    if (dataBaseHelper.insertData(recvdToken)){
                        Toast.makeText(getApplicationContext(), dataBaseHelper.
                            getAllData(), Toast.LENGTH_SHORT).show();

                    }
                    //dataBaseHelper.deleteAll();
                    Intent intent = new Intent(getApplicationContext(),Home.
                        class);
                    startActivity(intent);
                    overridePendingTransition(R.anim.slide_in_left,R.anim.
                        slide_out_right);

                } else if (response.equals("false")) {

                }

            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                if (error instanceof TimeoutError) {
                    if (count < 3) {
                        count++;
                        doSignUp();
                    } else {
                        Toast.makeText(SignupActivity.this, "You have Weak
                            Internet Connecton", Toast.LENGTH_SHORT).show();
                    }
```

```
                }

            }
        })

        {
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                HashMap<String, String> params = new HashMap<>();
                params.put("name", name);
                params.put("phone", phone);
                params.put("password", password);
                params.put("type","customer");
                Log.d("Login Params", "" + params);
                return params;
            }
            @Override
            public Map<String, String> getHeaders() throws AuthFailureError{
                HashMap<String, String> headers = new HashMap<>();
                headers.put("device-token",token) ;
                return headers;
            }

            @Override
            protected Response<String> parseNetworkResponse(NetworkResponse
                response) {

                recvdToken = response.headers.get("x-auth");
                Log.d("Rcvd Token",recvdToken);
                return super.parseNetworkResponse(response);
            }
        };
        requestQueue.add(signupRequest);

    }
    public boolean validate_UserName() {
        name = user_name.getEditText().getText().toString();
        if (name.isEmpty() || name.length() < 3) {
            user_name.setError("at least 3 characters");
            return false;
        } else {
            user_name.setError(null);
            return true;
        }

    }

    public boolean validate_Phone() {
        phone = phone_no.getEditText().getText().toString();

        if (phone.isEmpty() || phone.length() <9 ) {
            phone_no.setError("enter a valid phone number");
            return false;
        } else {
            phone_no.setError(null);
            return true;
        }
    }

    public boolean validate_Password() {
        password = user_password.getEditText().getText().toString();
```

```
188             if (password.isEmpty() || password.length() < 4 || password.length() >
                    10) {
189                 user_password.setError("between 4 and 10 alphanumeric characters");
190                 return false;
191             } else {
192                 user_password.setError(null);
193                 return true;
194             }
195         }
196 }
```

6.2 Product List

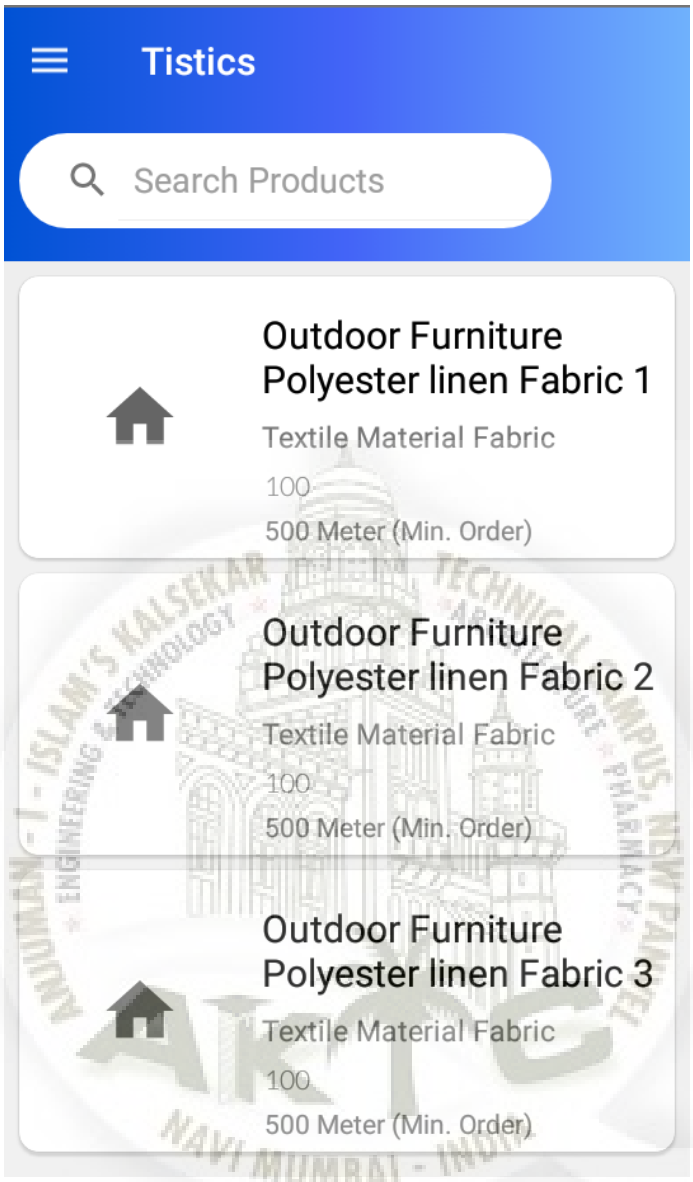


Figure 6.2: Home

```
package com.tisticswebcos.tistics_signup;

import ...;

public class Home extends BaseActivity {

    ItemInitializor list;
    SearchView searchView;
    String queryProduct,d_token;
    private ArrayList<ItemInitializor> marrayList = new ArrayList<>();
    private RecyclerView mrecyclerView;
    private RecyclerAdapter mAdapter;
    RequestQueue requestQueue;
//    JsonObjectRequest orderRequest;
    StringRequest orderRequest;
    DataBaseHelper dataBaseHelper;
```

```
17      SearchManager searchManager;
18      @Override
19      protected void onCreate(Bundle savedInstanceState) {
20          super.onCreate(savedInstanceState);
21          //setContentView(R.layout.activity_home);
22         requestQueue = Volley.newRequestQueue(this);
23          dataBaseHelper = new DataBaseHelper(this);
24          d_token = dataBaseHelper.getAllData();
25         ConstraintLayout contentFrameLayout = (ConstraintLayout) findViewById(R.
               id.content_frame); //Remember this is the FrameLayout area within
               your activity_main.xml
26          getLayoutInflater().inflate(R.layout.activity_home, contentFrameLayout)
                ;
27          NavigationView navigationView = (NavigationView) findViewById(R.id.
               nav_view);
28          navigationView.getMenu().getItem(0).setChecked(true);
29
30          mrecyclerView = findViewById(R.id.recyclerview);
31          mAdapter = new RecyclerAdapter(marrayList, this);
32          mrecyclerView.setLayoutManager(new LinearLayoutManager(
               getApplicationContext()));
33          mrecyclerView.setItemAnimator(new DefaultItemAnimator());
34         // mrecyclerView.addItemDecoration(new DividerItemDecoration(this,
               LinearLayoutManager.VERTICAL));
35          mrecyclerView.setAdapter(mAdapter);
36          System.out.print("above prepare data");
37         // prepareData();
38
39
40      }
41
42      public boolean onCreateOptionsMenu(Menu menu) {
43          // Inflate the options menu from XML
44           searchManager = (SearchManager) getSystemService(Context.SEARCH_SERVICE
                );
45
46          searchView = (SearchView) findViewById(R.id.searchview);
47
48          searchView.setQueryHint("Search Products");
49           //Assumes current activity is the searchable activity
50          searchView.setSearchableInfo(searchManager.getSearchableInfo(
               getComponentName()));
51          searchView.setIconifiedByDefault(false); //Do not iconify the widget;
               expand it by default
52          searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
53              @Override
54              public boolean onQueryTextSubmit(String query) {
55                  Toast.makeText(getApplicationContext(), "Query submitted",Toast.
                       LENGTH_SHORT).show();
56                  Toast.makeText(getApplicationContext(), searchView.getQuery(),
                       Toast.LENGTH_SHORT).show();
57                  queryProduct = searchView.getQuery().toString();
58                  prepareData();
59                  return false;
60              }
61
62              @Override
63              public boolean onQueryTextChange(String newText) {
64                  Toast.makeText(getApplicationContext(),"query changed",Toast.
                       LENGTH_SHORT).show();
65
```

```
                return false;
            }
        });


        return true;



    }

    private void prepareData()
    {

            Toast.makeText(Home.this,"Searched is "+queryProduct,Toast.
                LENGTH_SHORT).show();
        list = null;
        orderRequest = new StringRequest(Request.Method.POST,
                URLHelper.orders,
                new Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {
                        Toast.makeText(Home.this,"Response is "+response,Toast.
                            LENGTH_SHORT).show();
                        try
                        {
                            JSONArray jsonArray = new JSONArray(response);
                            if(jsonArray.length()>0)
                            {
                                Log.d("JSOn Response",jsonArray.
                                    getJSONObject(0).toString());
                                for(int i=0;i<jsonArray.length();i++)
                                {
                                    JSONObject jsonObject = jsonArray.
                                        getJSONObject(i);
                                    String productName = jsonObject.
                                        getString("product_name");
                                    String productType = jsonObject.
                                        getString("product_type");
                                    int price = jsonObject.getInt("price");
                                    Log.d("Product Name",productName);
                                    Log.d("Product Type",productType);
                                    Log.d("Price",""+price);
                                    list = new ItemInitializor(productName,
                                        productType,price);
                                    marrayList.add(list);
                                }
                                mAdapter.notifyDataSetChanged();
                            }
                            else
                            {
                                Toast.makeText(Home.this,"Empty JSON ARRAY",
                                    Toast.LENGTH_SHORT).show();
                            }

                        }
                        catch(Exception e){
                            System.out.print(e.toString());
                        }
```

```
                    }
                },
                new Response.ErrorListener() {
                    @Override
                    public void onErrorResponse(VolleyError error) {
                        Toast.makeText(Home.this,"Error is"+error.toString(),
                            Toast.LENGTH_SHORT).show();
                    }
                })
        {
            @Override
            protected Map<String, String> getParams() throws com.android.volley.
                AuthFailureError {
                Map<String, String> params = new HashMap<>();
                params.put("product_name", queryProduct);
                Log.d("LoginParams", "" + params);
                return params;
            }

            @Override
            public Map<String, String> getHeaders() throws AuthFailureError{
                HashMap<String, String> headers = new HashMap<>();
                headers.put("x-auth",d_token);
                // Toast.makeText(getApplicationContext(),dataBaseHelper.
                    getAllData(),Toast.LENGTH_SHORT).show();
                //Toast.makeText(getApplicationContext(),"getheaders",Toast.
                    LENGTH_SHORT).show();

                return headers;
            }
        };

        requestQueue.add(orderRequest);
    }
}
```

6.3 Order Tracking

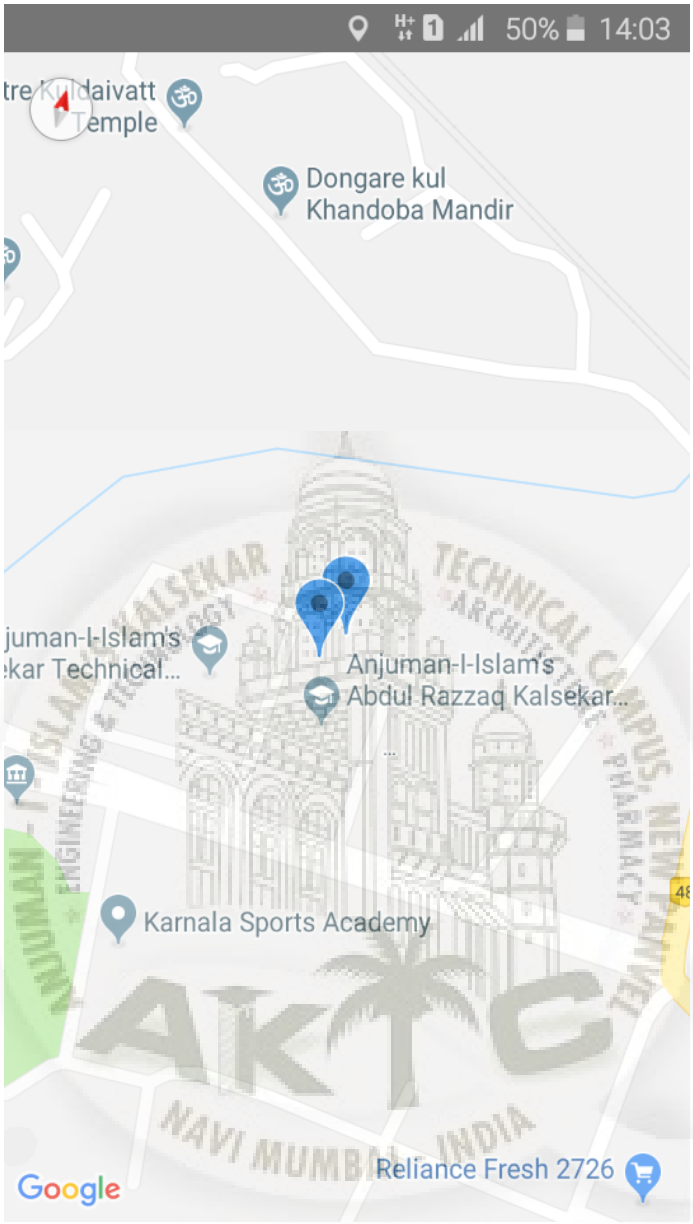


Figure 6.3: Order Tracking

```
package com.tisticswebcos.tistics_signup;

import ...;
public class OrderLocationMap extends FragmentActivity implements
    OnMapReadyCallback {

    private GoogleMap mMap;
    LocationManager locationManager;
    LocationListener locationListener;


    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
        permissions, @NonNull int[] grantResults) {
```

```
13          super.onRequestPermissionsResult(requestCode, permissions, grantResults)
                ;
14          if(requestCode == 1){
15              if(grantResults.length > 0 && grantResults[0] == PackageManager.
                    PERMISSION_GRANTED){
16
17                  if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                        ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED)
                        {
18
19                      locationManager.requestLocationUpdates(LocationManager.
                            GPS_PROVIDER, 0, 0, locationListener);
20                  }
21              }
22
23          }
24      }
25
26      @Override
27      protected void onCreate(Bundle savedInstanceState) {
28          super.onCreate(savedInstanceState);
29          setContentView(R.layout.activity_order_location_map);
30          // Obtain the SupportMapFragment and get notified when the map is ready
                to be used.
31          SupportMapFragment mapFragment = (SupportMapFragment)
                getSupportFragmentManager()
32                  .findFragmentById(R.id.map);
33          mapFragment.getMapAsync(this);
34      }
35      @Override
36      public void onMapReady(GoogleMap googleMap) {
37          mMap = googleMap;
38
39          locationManager = (LocationManager)this.getSystemService(Context.
                LOCATION_SERVICE);
40
41          locationListener = new LocationListener() {
42              @Override
43              public void onLocationChanged(Location location) {
44
45                  LatLng userLocation = new LatLng(location.getLatitude(),
                        location.getLongitude());
46                  mMap.addMarker(new MarkerOptions().position(userLocation).title(
                        "Your Location").icon(BitmapDescriptorFactory.defaultMarker(
                        BitmapDescriptorFactory.HUE_AZURE)));
47                  mMap.moveCamera(CameraUpdateFactory.newLatLng(userLocation));
48              }
49
50              @Override
51              public void onStatusChanged(String s, int i, Bundle bundle) {
52
53              }
54
55              @Override
56              public void onProviderEnabled(String s) {
57
58              }
59
60              @Override
61              public void onProviderDisabled(String s) {
62
```

```
            }
        };

        if(Build.VERSION.SDK_INT < 23){

            if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {

                locationManager.requestLocationUpdates(LocationManager.
                    GPS_PROVIDER, 0, 0, locationListener);
            }
        }
        else{


            if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED){

                ActivityCompat.requestPermissions(this, new String[]{Manifest.
                    permission.ACCESS_FINE_LOCATION},1);

            }
            else{

                locationManager.requestLocationUpdates(LocationManager.
                    GPS_PROVIDER,0,0,locationListener);

            }
        }
        mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
      //  LatLng userLocation = new LatLng(-34, 151);
       // mMap.addMarker(new MarkerOptions().position(userLocation).title("Your
           Location").icon(BitmapDescriptorFactory.defaultMarker(
           BitmapDescriptorFactory.HUE_AZURE)));

        // Add a marker in Sydney and move the camera

    }
}
```

6.4 Product Details

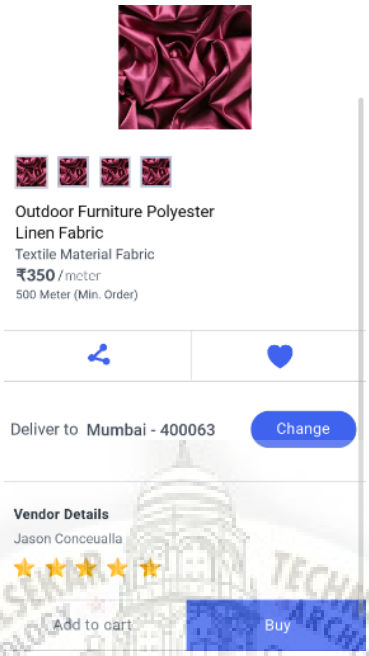


Figure 6.4: Product Details

```
<?xml version="1.0" encoding="utf-8"?>

  <android.support.v4.widget.NestedScrollView
       android:id="@+id/proddetnestedScrollView"
       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=".ProductDetails"
       >
  <android.support.constraint.ConstraintLayout
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       tools:context=".ProductDetails">

       <ImageView
            android:id="@+id/imageView8"
            android:layout_width="50dp"
            android:layout_height="50dp"
            android:layout_marginStart="8dp"
            app:layout_constraintBottom_toBottomOf="@+id/imageView7"
            app:layout_constraintStart_toEndOf="@+id/imageView7"
            app:srcCompat="@drawable/nature" />

       <ImageView
            android:id="@+id/imageView7"
            android:layout_width="50dp"
            android:layout_height="50dp"
            android:layout_marginStart="8dp"
            app:layout_constraintBottom_toBottomOf="@+id/imageView6"
            app:layout_constraintStart_toEndOf="@+id/imageView6"
            app:srcCompat="@drawable/nature" />
```

```
34
35          <ImageView
36              android:id="@+id/imageView4"
37              android:layout_width="155dp"
38              android:layout_height="108dp"
39              android:layout_marginTop="8dp"
40              app:layout_constraintEnd_toEndOf="parent"
41              app:layout_constraintStart_toStartOf="parent"
42              app:layout_constraintTop_toBottomOf="@+id/appBarLayout"
43              app:srcCompat="@drawable/nature" />
44
45          <ImageView
46              android:id="@+id/imageView5"
47              android:layout_width="50dp"
48              android:layout_height="50dp"
49              android:layout_marginStart="16dp"
50              android:layout_marginTop="24dp"
51              app:layout_constraintStart_toStartOf="parent"
52              app:layout_constraintTop_toBottomOf="@+id/imageView4"
53              app:srcCompat="@drawable/nature" />
54
55          <ImageView
56              android:id="@+id/imageView6"
57              android:layout_width="50dp"
58              android:layout_height="50dp"
59              android:layout_marginStart="8dp"
60              app:layout_constraintBottom_toBottomOf="@+id/imageView5"
61              app:layout_constraintStart_toEndOf="@+id/imageView5"
62              app:srcCompat="@drawable/nature" />
63
64          <TextView
65              android:id="@+id/productDetName"
66              android:layout_width="wrap_content"
67              android:layout_height="wrap_content"
68              android:layout_marginStart="16dp"
69              android:layout_marginTop="17dp"
70              android:fontFamily="@font/roboto"
71              android:textColor="#0C0C0C"
72              android:textSize="20sp"
73              android:text="TextView"
74              app:layout_constraintStart_toStartOf="parent"
75              app:layout_constraintTop_toBottomOf="@+id/imageView5" />
76
77          <TextView
78              android:id="@+id/productDetType"
79              android:layout_width="wrap_content"
80              android:layout_height="wrap_content"
81              android:layout_marginStart="16dp"
82              android:layout_marginTop="13dp"
83              android:text="TextView"
84              android:fontFamily="@font/roboto"
85              android:textColor="#47525E"
86              android:textSize="17sp"
87              app:layout_constraintStart_toStartOf="parent"
88              app:layout_constraintTop_toBottomOf="@+id/productDetName" />
89
90          <TextView
91              android:id="@+id/prodDetPrice"
92              android:layout_width="wrap_content"
93              android:layout_height="wrap_content"
94              android:layout_marginStart="16dp"
```

```
95              android:layout_marginTop="8dp"
96              android:text="TextView"
97              android:textColor="#0C0C0C"
98              android:textSize="20sp"
99              android:fontFamily="@font/roboto"
100             app:layout_constraintStart_toStartOf="parent"
101             app:layout_constraintTop_toBottomOf="@+id/productDetType" />
102
103         <TextView
104             android:id="@+id/prodDetpermeter"
105             android:layout_width="wrap_content"
106             android:layout_height="wrap_content"
107             android:layout_marginTop="8dp"
108             android:text="/meter"
109             android:fontFamily="@font/lato_light"
110             android:textColor="#47525E"
111             android:textSize="17sp"
112             app:layout_constraintStart_toEndOf="@+id/prodDetPrice"
113             app:layout_constraintTop_toBottomOf="@+id/productDetType" />
114
115         <TextView
116             android:id="@+id/prodDetMinOrder"
117             android:layout_width="wrap_content"
118             android:layout_height="wrap_content"
119             android:layout_marginStart="16dp"
120             android:layout_marginTop="10dp"
121             android:text="Min. Order(500)"
122             android:textColor="#47525E"
123             android:textSize="17sp"
124             android:fontFamily="@font/roboto"
125             app:layout_constraintStart_toStartOf="parent"
126             app:layout_constraintTop_toBottomOf="@+id/prodDetPrice" />
127
128         <ImageButton
129             android:id="@+id/shareButton"
130             android:layout_width="205dp"
131             android:layout_height="wrap_content"
132             android:layout_marginTop="16dp"
133             android:background="@drawable/transparent_bg_bordered"
134             android:paddingTop="10dp"
135             android:paddingBottom="10dp"
136             app:layout_constraintEnd_toStartOf="@+id/likeButton"
137             app:layout_constraintStart_toStartOf="parent"
138             app:layout_constraintTop_toBottomOf="@+id/prodDetMinOrder"
139             app:srcCompat="@drawable/ic_share_black_24dp" />
140
141         <ImageButton
142             android:id="@+id/likeButton"
143             android:layout_width="205dp"
144             android:layout_height="wrap_content"
145             android:layout_marginEnd="1dp"
146             android:background="@drawable/transparent_bg_bordered"
147             android:paddingTop="10dp"
148             android:paddingBottom="10dp"
149             app:layout_constraintBottom_toBottomOf="@+id/shareButton"
150             app:layout_constraintEnd_toEndOf="parent"
151             app:layout_constraintStart_toEndOf="@+id/shareButton"
152             app:layout_constraintTop_toTopOf="@+id/shareButton"
153             app:layout_constraintVertical_bias="0.0"
154             app:srcCompat="@drawable/ic_favorite_24dp" />
155
```

```
156         <TextView
157             android:id="@+id/textView2"
158             android:layout_width="wrap_content"
159             android:layout_height="wrap_content"
160             android:layout_marginStart="16dp"
161             android:layout_marginTop="36dp"
162             android:layout_marginEnd="20dp"
163             android:fontFamily="@font/roboto"
164             android:text="Deliver to"
165             android:textColor="#47525E"
166             android:textSize="17sp"
167             app:layout_constraintEnd_toStartOf="@+id/custAdd"
168             app:layout_constraintStart_toStartOf="parent"
169             app:layout_constraintTop_toBottomOf="@+id/shareButton" />
170
171         <TextView
172             android:id="@+id/custAdd"
173             android:layout_width="wrap_content"
174             android:layout_height="wrap_content"
175             android:layout_marginStart="20dp"
176             android:layout_marginEnd="10dp"
177             android:fontFamily="@font/roboto"
178             android:text="Mumbai - 400063"
179             android:textColor="#1D1F22"
180             android:textSize="18sp"
181             app:layout_constraintBaseline_toBaselineOf="@+id/textView2"
182             app:layout_constraintEnd_toStartOf="@+id/addChange"
183             app:layout_constraintStart_toEndOf="@+id/textView2" />
184
185         <TextView
186             android:id="@+id/textView5"
187             android:layout_width="wrap_content"
188             android:layout_height="wrap_content"
189             android:layout_marginStart="16dp"
190             android:layout_marginTop="20dp"
191             android:fontFamily="@font/roboto"
192             android:text="Vendor Details"
193             android:textColor="#0C0C0C"
194             android:textSize="18sp"
195             android:textStyle="bold"
196             app:layout_constraintStart_toStartOf="parent"
197             app:layout_constraintTop_toBottomOf="@+id/divider7" />
198
199         <View
200             android:id="@+id/divider7"
201             android:layout_width="412dp"
202             android:layout_height="1dp"
203             android:layout_marginTop="36dp"
204             android:background="?android:attr/listDivider"
205             app:layout_constraintEnd_toEndOf="parent"
206             app:layout_constraintHorizontal_bias="0.0"
207             app:layout_constraintStart_toStartOf="parent"
208             app:layout_constraintTop_toBottomOf="@+id/custAdd" />
209
210         <TextView
211             android:id="@+id/vendorName"
212             android:layout_width="wrap_content"
213             android:layout_height="wrap_content"
214             android:layout_marginStart="16dp"
215             android:layout_marginTop="15dp"
216             android:text="Jason Conceualla"
```

```
217             android:textColor="#47525E"
218             android:textSize="18sp"
219             android:fontFamily="@font/roboto"
220             app:layout_constraintStart_toStartOf="parent"
221             app:layout_constraintTop_toBottomOf="@+id/textView5" />
222
223         <RatingBar
224             android:id="@+id/ratingBar"
225             android:layout_width="wrap_content"
226             android:layout_height="wrap_content"
227             android:layout_marginStart="16dp"
228             android:layout_marginTop="19dp"
229             android:numStars="5"
230             app:layout_constraintStart_toStartOf="parent"
231             app:layout_constraintTop_toBottomOf="@+id/vendorName" />
232
233         <Button
234             android:id="@+id/addChange"
235             android:layout_width="wrap_content"
236             android:layout_height="wrap_content"
237             android:layout_marginStart="20dp"
238             android:layout_marginEnd="10dp"
239             android:background="@drawable/button_round"
240             android:fontFamily="@font/roboto"
241             android:paddingLeft="30dp"
242             android:paddingRight="30dp"
243             android:text="Change"
244             android:textColor="#FFFFFF"
245             app:layout_constraintBaseline_toBaselineOf="@+id/custAdd"
246             app:layout_constraintEnd_toEndOf="parent"
247             app:layout_constraintStart_toEndOf="@+id/custAdd" />
248
249         <Button
250             android:id="@+id/addToCart"
251             android:layout_width="205dp"
252             android:layout_height="wrap_content"
253             android:layout_marginTop="48dp"
254             android:layout_marginEnd="1dp"
255             android:background="@drawable/transparent_bg_bordered"
256             android:fontFamily="@font/roboto"
257             android:text="Add to Cart"
258             android:textColor="#5A6978"
259             app:layout_constraintEnd_toStartOf="@+id/buyProduct"
260             app:layout_constraintStart_toStartOf="parent"
261             app:layout_constraintTop_toBottomOf="@+id/ratingBar" />
262
263         <Button
264             android:id="@+id/buyProduct"
265             android:layout_width="205dp"
266             android:layout_height="wrap_content"
267             android:background="#2F80ED"
268             android:fontFamily="@font/roboto"
269             android:text="Buy"
270             android:textColor="#FFFFFF"
271             app:layout_constraintBaseline_toBaselineOf="@+id/addToCart"
272             app:layout_constraintEnd_toEndOf="parent"
273             app:layout_constraintStart_toEndOf="@+id/addToCart" />
274
275         <android.support.design.widget.AppBarLayout
276             android:id="@+id/appBarLayout"
277             android:layout_width="match_parent"
```

```
278             android:layout_height="wrap_content"
279             android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
280             app:layout_constraintEnd_toEndOf="parent"
281             app:layout_constraintStart_toStartOf="parent"
282             app:layout_constraintTop_toTopOf="parent">
283
284             <android.support.v7.widget.Toolbar
285
286                 android:id="@+id/productdettoolbar"
287
288                 android:layout_width="match_parent"
289
290                 android:layout_height="?attr/actionBarSize"
291
292                 android:background="@drawable/imageview_background"
293
294                 app:layout_constraintEnd_toEndOf="parent"
295
296                 app:layout_constraintStart_toStartOf="parent"
297
298                 app:layout_constraintTop_toTopOf="parent"
299                 app:layout_scrollFlags="enterAlways"
300                 app:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">
301
302
303             </android.support.v7.widget.Toolbar>
304         </android.support.design.widget.AppBarLayout>
305
306
307
308     </android.support.constraint.ConstraintLayout>
309     </android.support.v4.widget.NestedScrollView>
```

6.5 Location Map

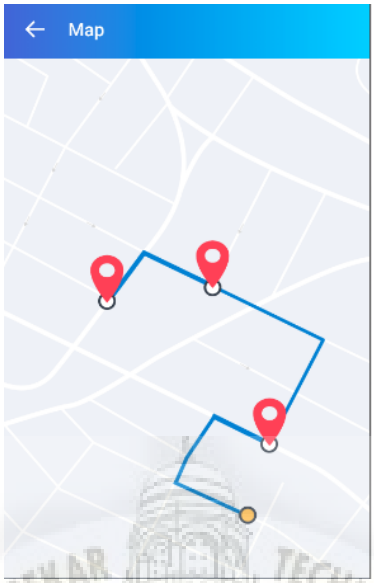


Figure 6.5: Location Map

```
package com.tisticswebcos.tistics_signup;

import ...

public class OrderLocationMap extends FragmentActivity implements
    OnMapReadyCallback {

    private GoogleMap mMap;
    LocationManager locationManager;
    LocationListener locationListener;


    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
        permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
            ;
        if(requestCode == 1){
            if(grantResults.length > 0 && grantResults[0] == PackageManager.
                PERMISSION_GRANTED){

                if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                    ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED)
                    {

                    locationManager.requestLocationUpdates(LocationManager.
                        GPS_PROVIDER, 0, 0, locationListener);
                }
            }

        }
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
30          setContentView(R.layout.activity_order_location_map);
31          // Obtain the SupportMapFragment and get notified when the map is ready
                to be used.
32          SupportMapFragment mapFragment = (SupportMapFragment)
                getSupportFragmentManager()
33                      .findFragmentById(R.id.map);
34          mapFragment.getMapAsync(this);
35      }
36
37      @Override
38      public void onMapReady(GoogleMap googleMap) {
39          mMap = googleMap;
40
41          locationManager = (LocationManager)this.getSystemService(Context.
                LOCATION_SERVICE);
42
43          locationListener = new LocationListener() {
44              @Override
45              public void onLocationChanged(Location location) {
46
47                  LatLng userLocation = new LatLng(location.getLatitude(),
                        location.getLongitude());
48                  mMap.addMarker(new MarkerOptions().position(userLocation).title(
                        "Your Location").icon(BitmapDescriptorFactory.defaultMarker(
                        BitmapDescriptorFactory.HUE_AZURE)));
49                  mMap.moveCamera(CameraUpdateFactory.newLatLng(userLocation));
50              }
51
52              @Override
53              public void onStatusChanged(String s, int i, Bundle bundle) {
54
55              }
56
57              @Override
58              public void onProviderEnabled(String s) {
59
60              }
61
62              @Override
63              public void onProviderDisabled(String s) {
64
65              }
66          };
67
68          if(Build.VERSION.SDK_INT < 23){
69
70              if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                    ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
71
72                  locationManager.requestLocationUpdates(LocationManager.
                        GPS_PROVIDER, 0, 0, locationListener);
73              }
74          }
75          else{
76
77
78              if(ContextCompat.checkSelfPermission(this, Manifest.permission.
                    ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED){
79
80                  ActivityCompat.requestPermissions(this, new String[]{Manifest.
                        permission.ACCESS_FINE_LOCATION},1);
```

```
81
82                  }
83                  else{
84
85                      locationManager.requestLocationUpdates(LocationManager.
                            GPS_PROVIDER,0,0,locationListener);
86
87                  }
88              }
89              mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
90            //   LatLng userLocation = new LatLng(-34, 151);
91             // mMap.addMarker(new MarkerOptions().position(userLocation).title("Your
                    Location").icon(BitmapDescriptorFactory.defaultMarker(
                    BitmapDescriptorFactory.HUE_AZURE)));
92
93              // Add a marker in Sydney and move the camera
94
95      }
96  }
```

Chapter 7

System Testing

After the completion of application we created a hypothetical situation to check our system.After the individual construction of the apps it was integrated with the web based panel which act as a connecting element between them.

7.1 Test Cases and Test Results

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Log In And Registration	Registration and log in activities should be successful in the app and the server should return a token	web server responded and the tokens were generated sucessfully	works fine
T02	order creation,acceptance, rejection and notifications	Order should be created in customer app whereas the vendor should be able to accept and reject the orders. accordingly notification should be provided to the users	The users were able to create , accept and reject orders and notifications were received accordingly to the users	works fine
T03	Google Maps Api	The Google maps should provide appropriate keys and tokens and generate the shortest path available	The tokens and keys were properly generated to provide the shortest path available	works fine

7.2 Sample of a Test Case

Title: Log-in and Registration Page – Authenticate Successfully on the applications of the system

Description: A registered user should be able to successfully log-in or register. The user should be able to register itself on the registration page of the apps.

Test Steps:

1. Navigate to log-in
2. Enter user-id and password to log-in the system
3. Click on the registration button to enter the registration page
4. Enter user-name, password and mobile in the appropriate spaces.
5. Click on the register button to register yourself

Expected Result: The home screen of the app should be displayed on the screen whereas after clicking on the registration button the app should redirect you to the log-in page

Actual Result: Expected results were achieved

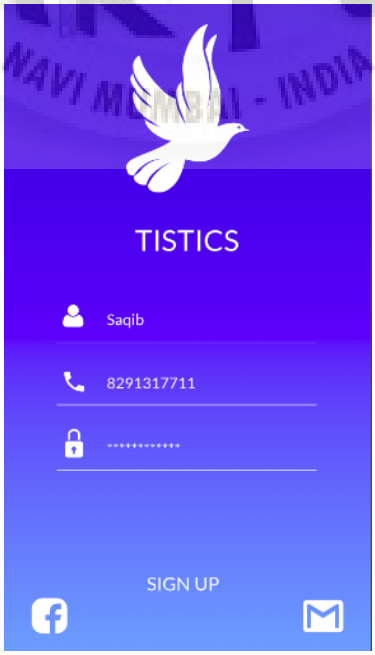


Figure 7.1: Sign Up

Title: creating accepting and rejecting the order and displaying appropriate notification

Description: A customer should be able to successfully create an order and should be notified about it . The vendor should be able to accept and reject the order.

Test Steps:

1. log- in to the customer app.
2. Select the item you wish to buy and then click on buy button.
3. log-in to the vendor app.
4. The home screen will display the list of orders given to the vendor from which the vendor should click on accept or reject.
5. Whether clicked on Accepted or rejected the customer will be notified Accordingly.

Expected Result: The home screen of the app should be displayed on the screen whereas after clicking on the registration button the app should redirect you to the log-in page

Actual Result: Expected results were achieved

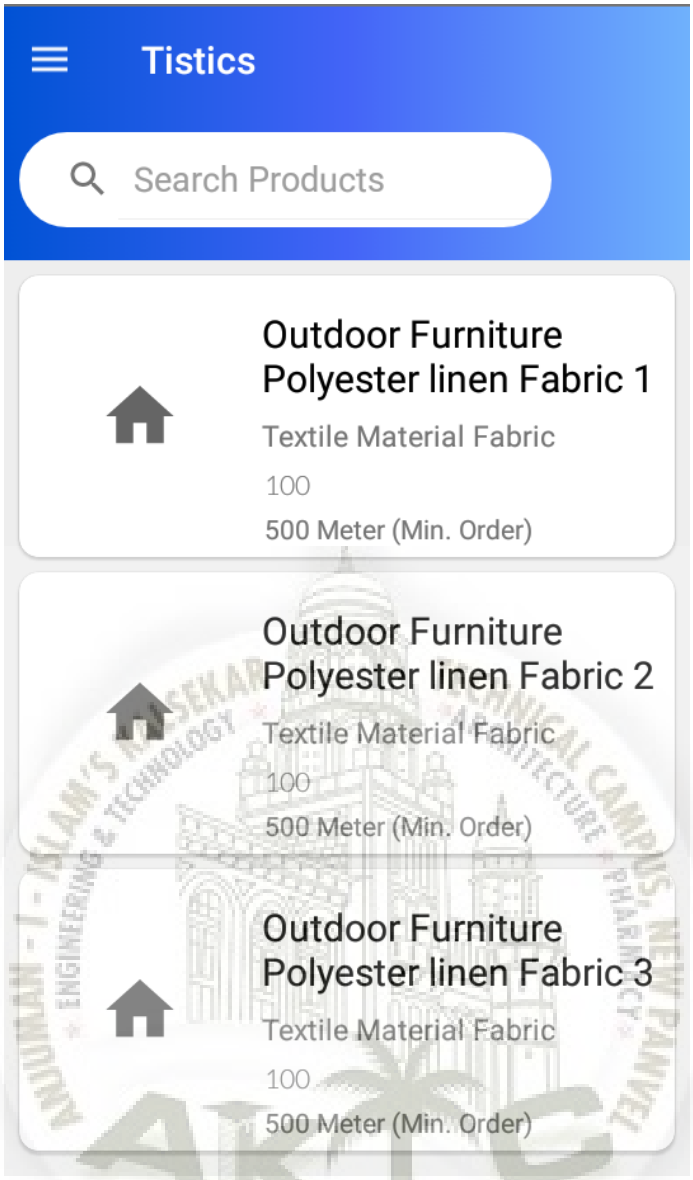


Figure 7.2: List of Products

Title: Google Maps API

Description: The Google maps api should generate appropriate tokens and api keys to provide the accurate paths in the maps.

Test Steps:

1. log- in to the Delivery boy app .
2. Click on location icon to view the maps
3. OR Click on the orders available on the screen.The order info page appears
4. Click on the address column which gives you the maps screen

Expected Result: After entering the maps screen the apps should show us the map with the shortest available path marked on them.

Actual Result: Expected results were achieved

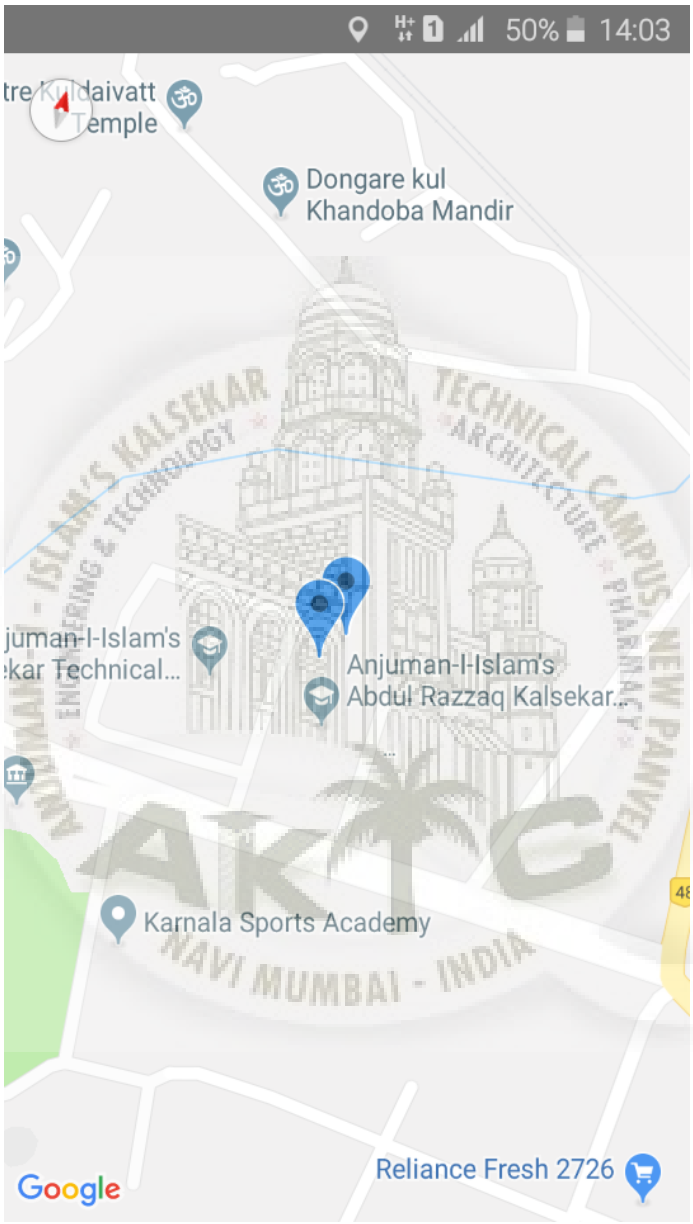


Figure 7.3: List of Products

7.2.1 Software Quality Attributes

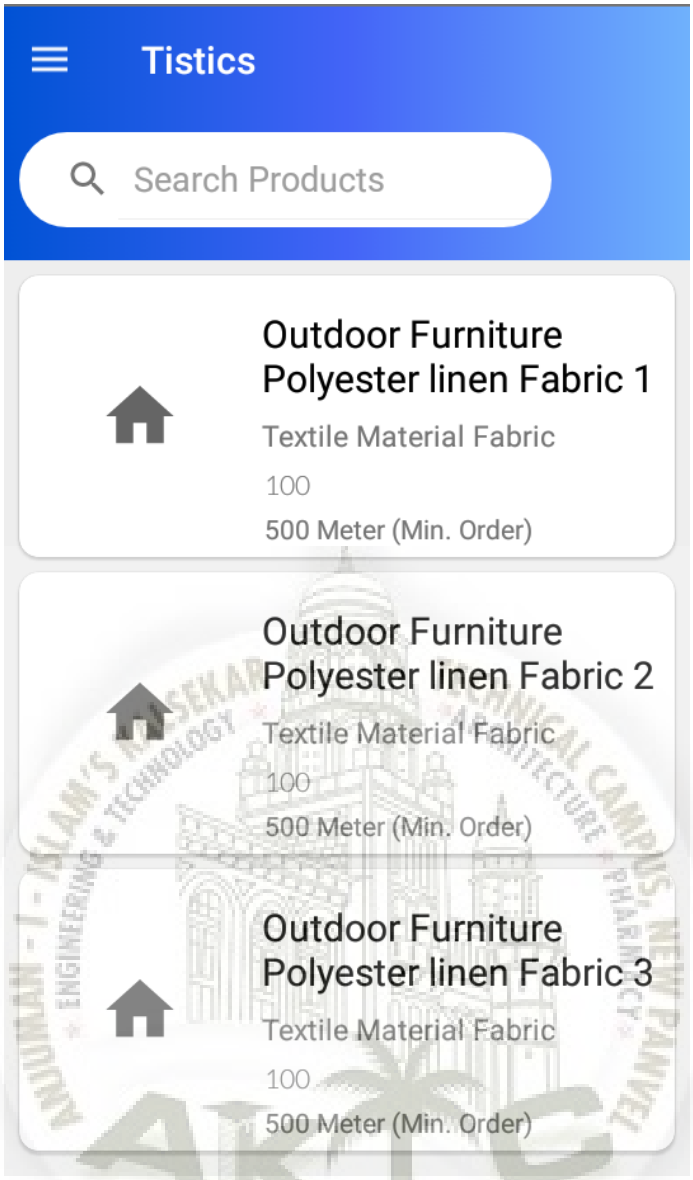
1.Availability: The Services provided by the apps should be available at all times. The server should not crash.

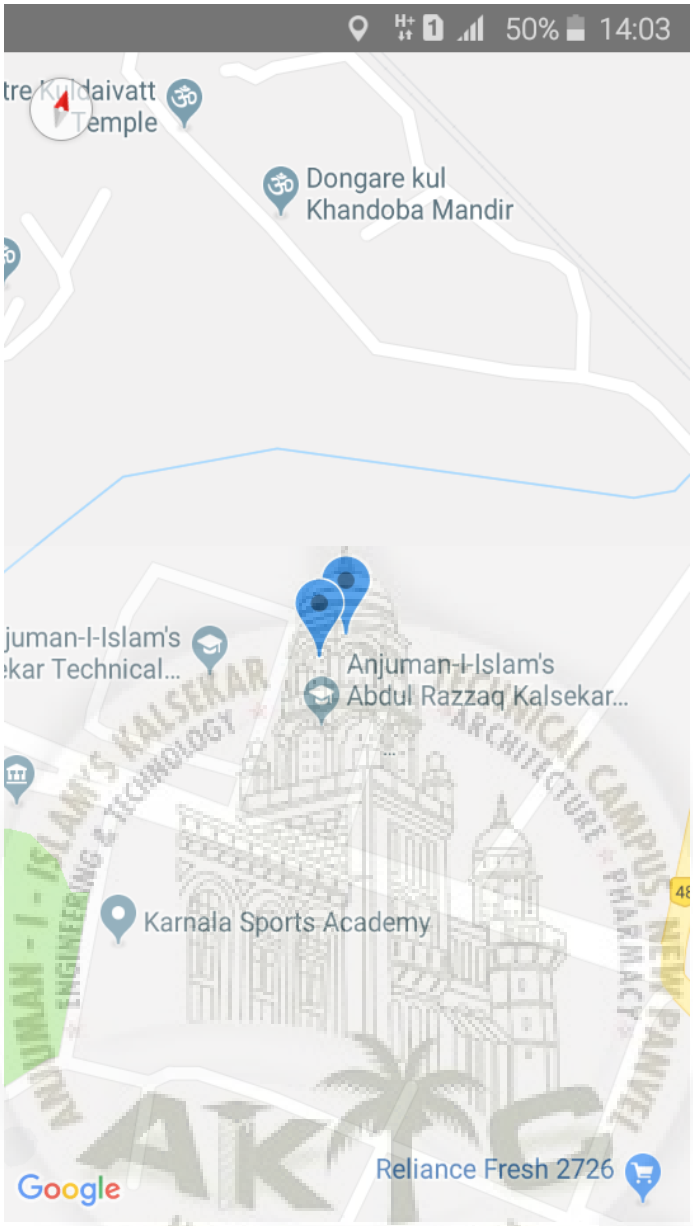
2. Correctness : The routes provided by the delivery boy app should be optimal.

3. Maintainability : The web-panel should be able to repair the damage done by apps if in-case they malfunction.

Chapter 8

Screenshots of Project





Chapter 9

Conclusion and Future Scope

9.1 Conclusion

Our System provides many features which will remove all the anomalies present in the current system of doing business and will give dynamic platform to the indian wholesale business

9.2 Future Scope

For a kick start Tistics has decided to target wholesalers who are trading cloth required in furniture. After covering this objective our Future scope is to cover as many product as we can which are ofcourse a part of wholesale market and to increase the reach of our business as wide as we can to match or maybe overpower the tycoons of the market.

References

[1] *Food ordering system with delivery routing optimization using globalpositioning system (gps) technology and google maps*; R. D. H. Tobing,Internetworking Indonesia Journal, 2016

[2] *New method for finding optimal path in dynamicnetworks*;M. Alivand, A. Alesheikh, and M. Malek, ,World Applied Sciences Journal, 2008

[3] *Commerce order batching algorithm based on association rule mining inthe era of big data*;Y. Li , IEEE, 2018

Achievements

1. Paper Presentation

   (a) *Tistics Delivery System*;Pal Pritesh Rooplal Suman,Ansari Sana Sabir Amina ,Avalon 2019 at Terna Engineering College on 5th March)

TERNA PUBLIC CHARITABLE TRUST'S
TERNA ENGINEERING COLLEGE
AN ISO 9001 : 2000 | NBA ACCREDITATION

terna
Knowledge to Empower

CERTIFICATE OF PARTICIPATION

This is to certify that

Pritesh Pal

of

A.I. Kalsekar Technical Campus

has participated in

Avalon 2019, A National Level
(Technical Paper Presentation / Project Competition)
conducted on 5th & 6th March, 2019
at **Terna Engineering College, Nerul**

Prof. D.M. Bavkar
Avalon co-ordinator

Dr. L.K. Ragha
Principal

TERNA PUBLIC CHARITABLE TRUST'S
TERNA ENGINEERING COLLEGE
AN ISO 9001 : 2000 | NBA ACCREDITATION

terna
Knowledge to Empower

CERTIFICATE OF PARTICIPATION

This is to certify that

Sana Ansari

of

A.I. Kalsekar Technical Campus

has participated in

Avalon 2019, A National Level
(Technical Paper Presentation / Project Competition)
conducted on 5th & 6th March, 2019
at **Terna Engineering College, Nerul**

AVALON

Prof. D.M. Bavkar
Avalon co-ordinator

Dr. L.K. Ragha
Principal