A

Project Report titled

# "E-Store for Om Sai Enterprises"

Submitted to

Department of Information Technology

of

[illegible]

S. M. Shetty College of Science, Commerce and Management Studies, Powai

For Partial Fulfillment for Degree of

Bachelor of Science (Information Technology)

2021 – 2022

In the Subject Head

Project (VI Semester)

Submitted by

**MANDAR RAVI JUNNARKAR**

A

Project Report titled

# "E-Store for Om Sai Enterprises"

Submitted to

Department of Information Technology

of

**Bunts Sangha's**

**S. M. Shetty College of Science, Commerce and Management Studies, Powai**

For Partial Fulfillment for Degree of

Bachelor of Science (Information Technology)

2021 – 2022

In the Subject Head

Project (VI Semester)

Submitted by

**MANDAR RAVI JUNNARKAR**

**Bunts Sangha's**

**S. M. Shetty College of Science, Commerce and Management Studies, Powai**

# *CERTIFICATE*

This is to certify that **Mandar Ravi Junnarkar** ( Exam Seat No: 2017539 ), a final year student of **Bachelor of Information Technology** (B.Sc.IT) from University of Mumbai has successfully completed the project entitled **"E-store for Om Sai Enterprises"** as a part of academic in the subject head Project (VI Semester) which is approved for degree of Bachelor of Information Technology (B.Sc.IT) a under-graduate course of **Mumbai University** during academic year **2021-22.**

**Project Guide**

21/04/22

**Examiner**

**Coordinator**

BUNTS SANGHA'S
**S.M. SHETTY COLLEGE OF SCIENCE, COMMERCE & MANAGEMENT STUDIES**
**POWAI, MUMBAI-76**

PERMANENTLY AFFILIATED TO THE UNIVERSITY OF MUMBAI
ACCREDITED BY N.A.A.C. – 'A' GRADE
IMC RBNQ Certificate of Merit 2019
ISO 21001:2018 Certified

# ***PLAGIARISM CERTIFICATE***

This is to certify that **Mr. Mandar Junnarkar (Seat No: 2017539)**, third year (Semester VI) student of **Bachelor of Information Technology** from University of Mumbai has submitted the project book content for plagiarism check. It was found that his content is good, as per format and the duplication is under UGC Recommended rules given in UGC (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018 [D.O.No.F.1-18/2010 (CPP-II)].

| | | |
|---|---|---|
| **In-Charge**<br>**Plagiarism Committee** | **Project Guide** | **Coordinator** |

**Stamp:** Librarian

**Date: 05.04.2022**

**Bunts Sangha's**

**S. M. Shetty College of Science, Commerce and Management Studies, Powai**

# PREFACE

It gives me enormous pleasure to present this report of project documentation / black book and the application I developed as a project of my last year. This project idea came into existence when the owner of Om Sai Enterprises asked me if I can create online store for their shop.

Then, I started looking for ideas and after having skipped several ideas during the Semester 5 and finished partially or almost on other topics, I thought to continue with this.

And after much research, once I got the basic configuration, I discovered other utilities essentially needed to implement my application.

Once, I completed the processing of my modules, I started the research and added additional structures and functionality to make this application healthy and harmonious.

So, in conclusion, I tried to form a dedicated application and this black book is maintained for the same purpose.

Therefore, I hope you adore analysis of the book and that the user appreciates the use of this website and because it will be satisfactory for my efforts.

# ACKNOWLEDGEMENT

I would like to use this Chance to convey my Gratitude to Management of S.M.SHETTY College for generous me this chance to accomplish this project. I am very thankful to

Dr. Sridhara Shetty, The Principal of S. M. Shetty College for his co-operation in the successful accomplishment of our project.

A special thanks to our project guide and also our Coordinator Dr.Tushar Sambare for his most sincere efforts, support and encouraging contribution throughout the project.

I would like to express thanks all our teachers, friends & our family for their support, motivation and encouragement.

- Mandar Junnarkar.

# INDEX

# CHAPTER 1

# INTRODUCTION

## 1.1 BACKGROUND

- Om Sai Enterprises is a store in Andheri, Mumbai. They and are known for dealing with products like name plates, number plates, rubber stamps, etc.
- Their business has grown rapidly after registering on Google business and sites like Justdial and they wish to expand further with the help of a website.

## 1.2 OBJECTIVES

- Creating an e-store for Om Sai Enterprises.
- To provide an interface for customers to surf, select and purchase their products online.
- To provide an interface for the owner to manage the point of sales as well as the user side elements.

## 1.3 PURPOSE

- Providing a platform for selling the store's products online.
- Let people to buy products sold buy Om Sai Enterprises online.

## 1.4 SCOPE

- The project is aimed at ensuring compatibility with latest browsers. Anyone on the internet with latest browser support will be able to order products online.
- **DELIVERABLES**
  - An e-store website for customers to shop online
  - An application for the owner to manage point of sales.

- **OBJECTIVES**
  - To increase profit by increasing point of sales. (online)
  - Allow customers to buy products from anywhere.
  - Increase customer satisfaction.
  - Increase efficiency of business by managing transactions online.

- **ASSUMPTIONS**
  - Requirements will not change in future.
  - There will not be any compatibility issue in future.
  - There will be a backup plan in case of failure or shutdown.
  - Proper security will be provided for user accounts and payments.
  - Budget will not be exceeded.

## 1.5 APPLICABILITY

- Anyone who wants to buy custom name plates, stamps, etc which are manufactured by Om Sai Enterprises.
- For anyone surfing on the internet who wants to buy a particular product which is sold by Om Sai Enterprises.

# CHAPTER 2

# SURVEY OF TECHNOLOGIES

## 2.1 DESCRIPTION OF PROJECT

The aim is to make a website through which customers can buy the products sold by Om Sai Enterprises and also to make an application for the owner to manage the sales. The customer side application will include all features which are supposed to be included in an e-store such as searching for their desired products, adding them in cart, making payments, etc. The admin side will include all features to manage the sales as well as some parts of the user side.

## 2.2 AVAILABLE TECHNOLOGIES

### 1. FRONT END TECHNOLOGIES

- **HTML5**

It is a markup language used to create the structure of a web page. It is a foundational technology and is very popular for making modern websites.

- **Bootstrap**

It is an open source CSS framework which is used to make responsive front end websites. It provides CSS and JavaScript templates for different attributes of HTML.

- **Vanilla JS**

Vanilla is the name for plain JavaScript without any additional library. It is the lightest weight framework ever. It helps to improve user experience and make web pages faster.

- **React JS**

It is a popular, open-source JavaScript library which helps to build highly interactive user interfaces and components. It is maintained by Facebook.

## 2. BACK END TECHNOLOGIES

### i. BUSINESS LOGIC

- **JSP**

It is used to insert Java code into HTML pages. It is a web based technology that helps to make dynamic and platform independent web pages.

- **Node JS**

It is an open-source, back end JavaScript run time environment which uses Google's V8 engine. It executes JavaScript code outside a web browser.

- **Python**

Python is an ideal back end language because of its libraries and frameworks. Django and Flask are two web development frameworks for python.

- **PHP**

It is a scripting language developed specially for web development. It is used for making fast, flexible, dynamic and interactive web pages.

### ii. DATABASE

- **SQL**

It is a query language which is used for managing relational databases. Some popular systems based on SQL are MySQL, SQLite, SQL server and Maria DB.

- **No SQL**

They are non-relational databases and they do not store data in relational tables. Some popular examples of No SQL databases are Mongo DB and Firebase.

## 2.3 JUSTIFICATION OF PLATFORM

### 2.3.1 THE FOLLOWING TECHNOLOGIES WILL BE USED IN THE FRONT END.

- **React JS**
  - It is a popular, open-source JavaScript library which helps to build highly interactive user interfaces and components. It is maintained by Facebook
  - It overcomes the struggles of HTML by being able to deliver amazing graphics and high definition videos.
  - It is free, lightweight, fast, works across all devices and is also supported in all modern browsers.
  - It follows the component architecture where in we can create reusable components which we can import and use in any other component as well.
  - It is used to create Single Page Applications(SPAs) because it just manipulates the DOM using JavaScript and displays different pages inside a single HTML page.
  - React uses a special syntax called JSX(JavaScript XML) which looks same as HTML but not actually HTML.
  - Making complex and seamless work flows is very easy in React JS.
- **Tailwind CSS**
  - Tailwind CSS is an highly customizable utility-first CSS framework.
  - It provides utility classes for different CSS properties which we can use directly in our HTML class.
  - It reduces time used for writing CSS.
  - It is a low-level CSS framework because it has all the things which are used for creating a custom designed application without having any default styles for elements which we have to fight to override.
  - Tailwind also has its UI components which we can copy and use it directly in our projects.

- **Vanilla JS**
  - JavaScript is a text-based programming language and is one of the popular technologies alongside HTML and CSS.
  - It gives the pages interactive elements that engage a user. It plays a very important role along CSS to create beautiful designs and is highly responsible to enhance the user's experience.
  - It was initially created to make web pages alive.
  - It has full integration with HTML and CSS.
  - It is supported by all major browsers and is enabled by default.

### 2.3.2 THE FOLLOWING TECHNOLOGIES WILL BE USED IN THE BACK END.

- **Firebase**
  - Firebase is a tool by Google which provides Software Development Kits for building applications on Android, IOS, Web, C++ and Unity.
  - It is Baas(Backend-as-a-Service).
  - For developers, it provides services like authentication, firestore(database), storage(cloud),etc which are very easy to implement using the provided documentation.

# CHAPTER 3

# REQUIREMENT AND ANALYSIS

## 3.1 PROBLEM DEFINITION

As technology advances, many new things have been developed and are helping us in a big manner. One of them are e-commerce websites through which we can buy things sitting in our homes.

Om Sai Enterprises wants to expand their business online as well. So, with the help of a website, the store can be made available online and people will be able to buy products from their homes.

**Advantages of an e-store:**

- Builds a brand by reaching more people.
- Saves time and money to visit the store.
- Detailed information about products and customer reviews to consider while purchasing.
- Getting customer data and making analysis for marketing more effectively and accurately.

## 3.2 HARDWARE AND SOFTWARE REQUIREMENTS

Following are the hardware and software requirements for making the website:

i. **Hardware requirements for development -**

- Ram: minimum 2GB
- Processor: minimum 1GHz
- Storage: 2GB or more

ii. **Software requirements for development -**

- Operating System: Windows
- Front end: React JS, Tailwind CSS, Bootstrap Icons.
- Back end:
  - Firebase
- Software tool: Visual Studio Code, Node JS, browser with developer tools, Firebase, GitHub, Heroku.

iii. **Requirements for user**

- User devices: Smartphone or computer
  - Windows XP/7/8/10/11
  - Android 4.1 or higher
  - Ios 6 or higher
  - Mac OS 10.5 or higher
- User software: Latest version of browsers

## 3.3 FUNCTIONAL REQUIREMENTS

These are the requirements which the end user demands from the system.

### 3.3.1 Search and visit the website easily

- Users must be able to search for the website easily using the URL.

### 3.3.2 Have a secured account

- Users can register and login with their email or phone number without any fear of data piracy and have a secured account.
- Option of two factor authentication for better security.

### 3.3.3 Retrieve account

- The users should be able to retrieve their account with their phone number if they have forgotten their email or password.

### 3.3.4 Option to Search

- There should be a field for the users to search for their desired items.
- Filter products based on categories.

### 3.3.5 Contact for help

- If the customers have any query regarding anything about the store, there will be an option to contact via email and phone.

### 3.3.6 Give and view feedback.

- Customers can view the reviews given by other customers.
- Customers can share their feedback about a product only after they have purchased it.

### 3.3.7 Add to Cart

- Customers can add products in a cart and then checkout all the desired products at once.

### 3.3.8 Checkout and Payment

- Check order details and final amount to be paid.
- Customers will be able to make secured transactions online using their debit card, credit card and UPI.

### 3.3.9 Track the order

- Allow customers to know where their order is by having an option to track the order.

### 3.3.10 Personalize information

- Users can change their password, add multiple addresses and add alternate phone numbers.

### 3.3.11 Cancel orders

- Customers can cancel their order if they do not wish have the product any more.

### 3.3.12 Logout

- After using the website, users should have an option to logout from their accounts for safety.

## 3.4 NON-FUNCTIONAL REQUIREMENTS

### 3.4.1 Performance

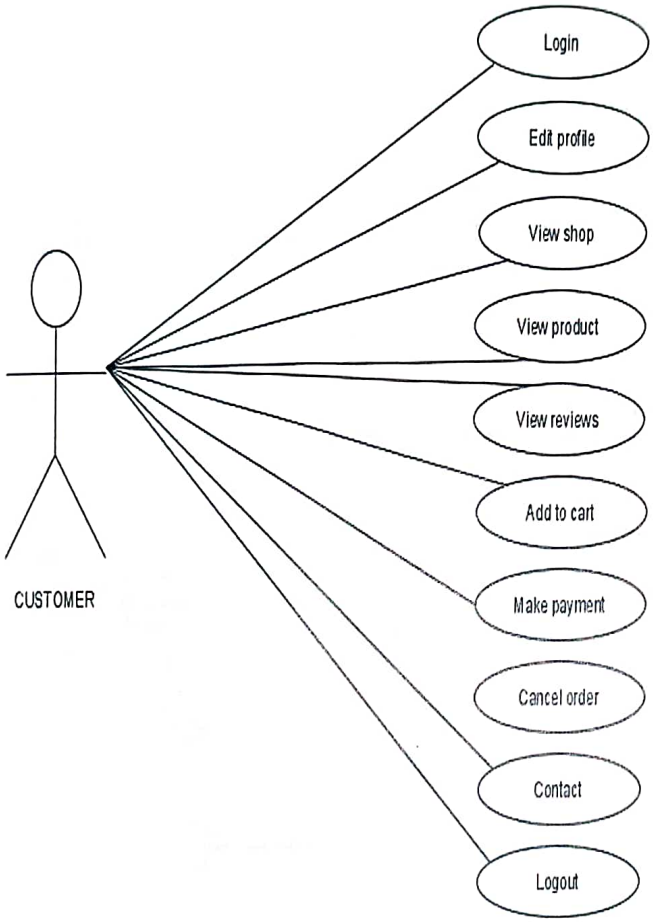
- The website shall be based on the web and has to run on all latest web browsers.
- The website design should be responsive to run on all devices.
- The performance should be such that the website runs smoothly on a user's system with minimal configuration.

### 3.4.2 Security

- The website should have a good database and shall secure the data of users.
- In case of failure,data backup should be available.

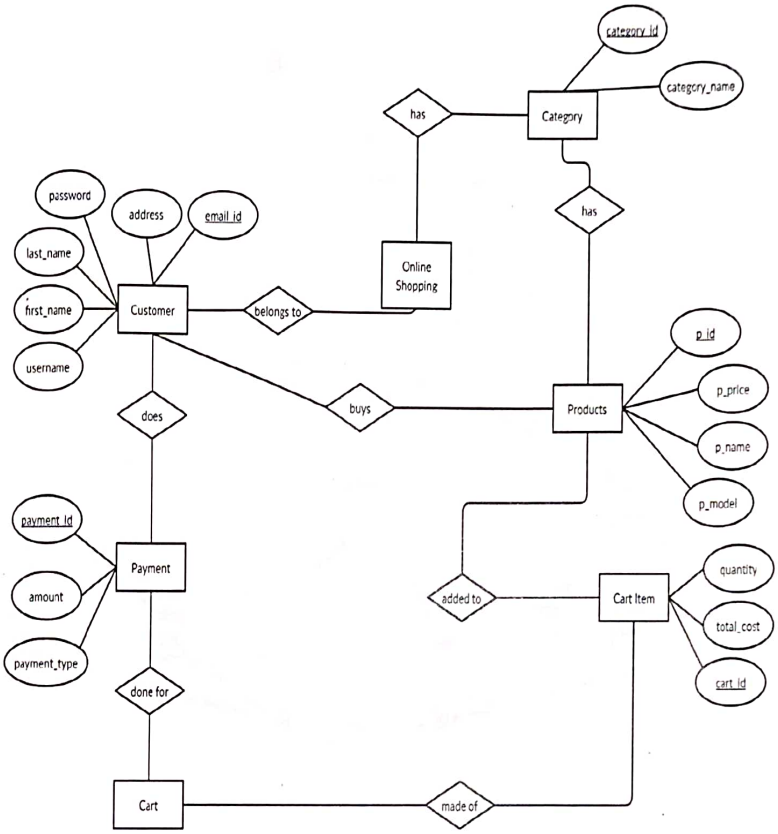
## 3.5 CONCEPTUAL DIAGRAMS

### i. Use Case diagram



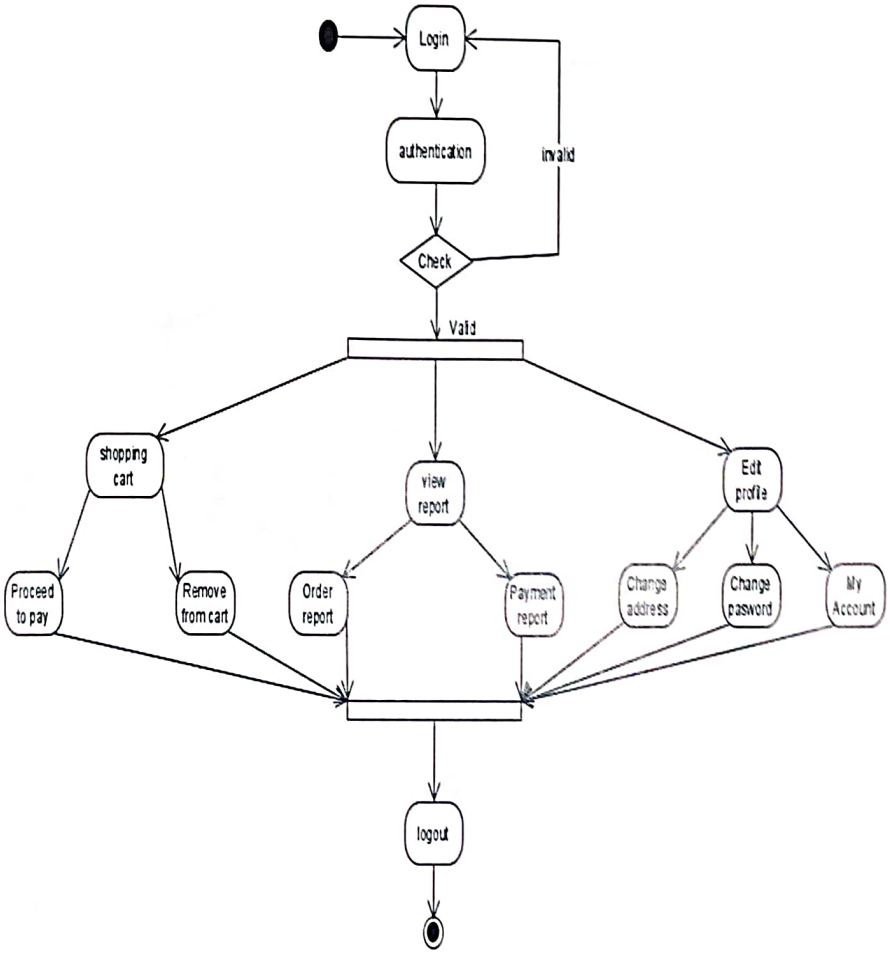
## ii. ER diagram

Entity Relationship diagram for user interface

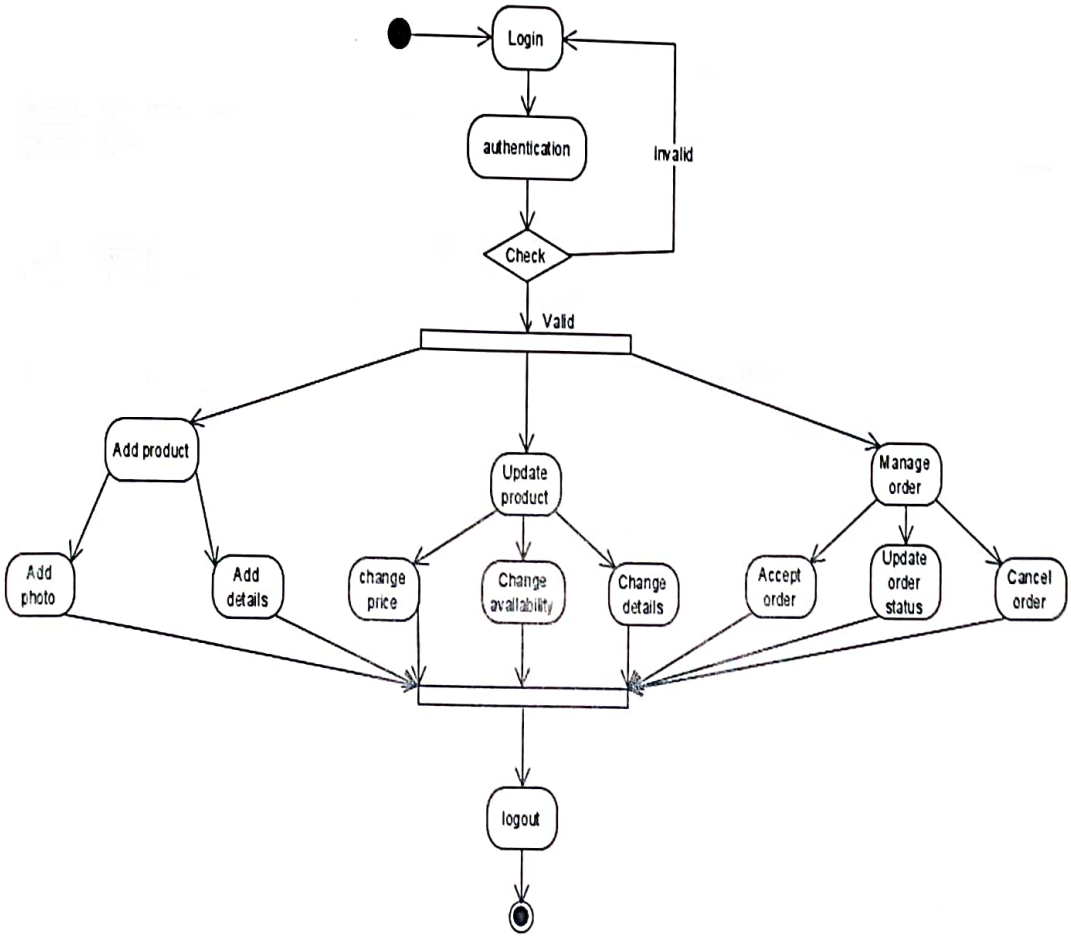


### iii. Activity diagram

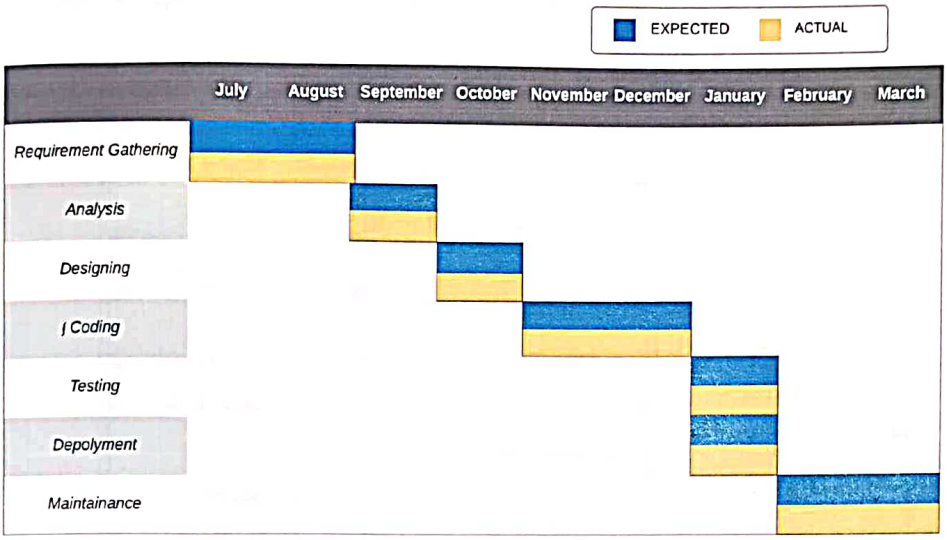
**Activity diagram for customer interface**



**Activity diagram for admin interface**



iv. **Planning and scheduling using Gantt chart**

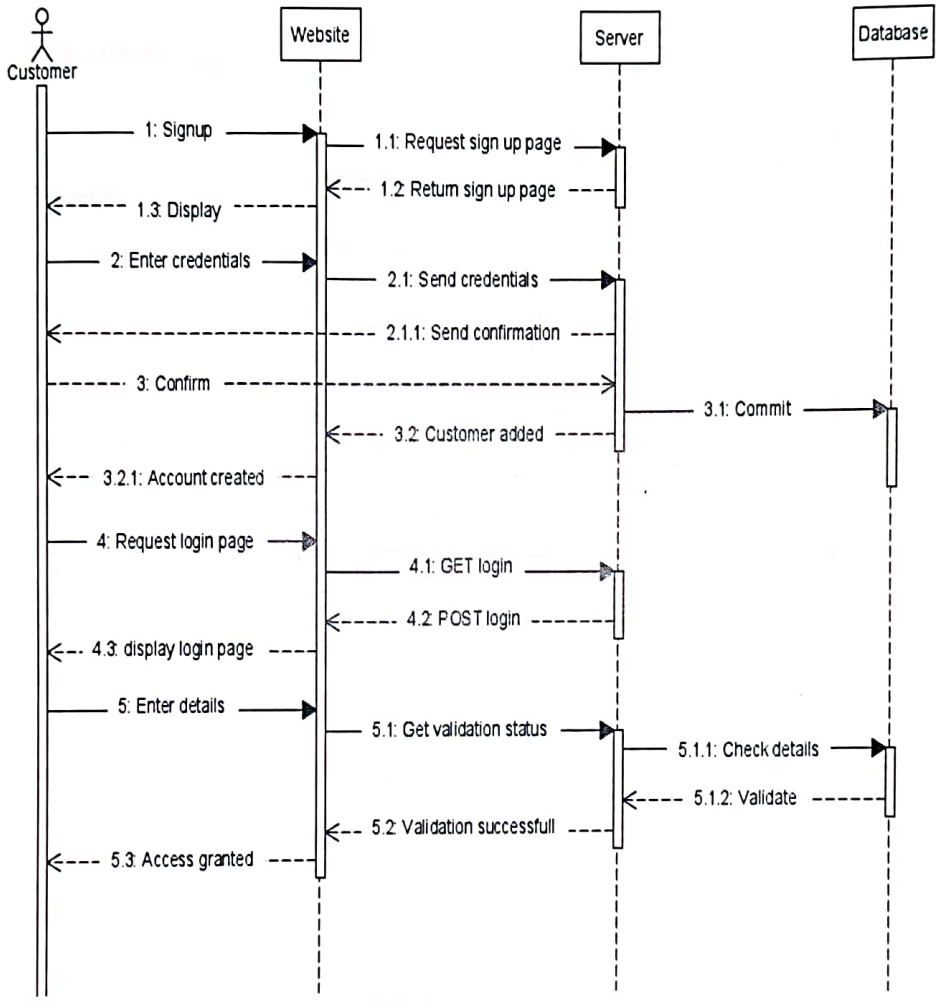


# CHAPTER 4

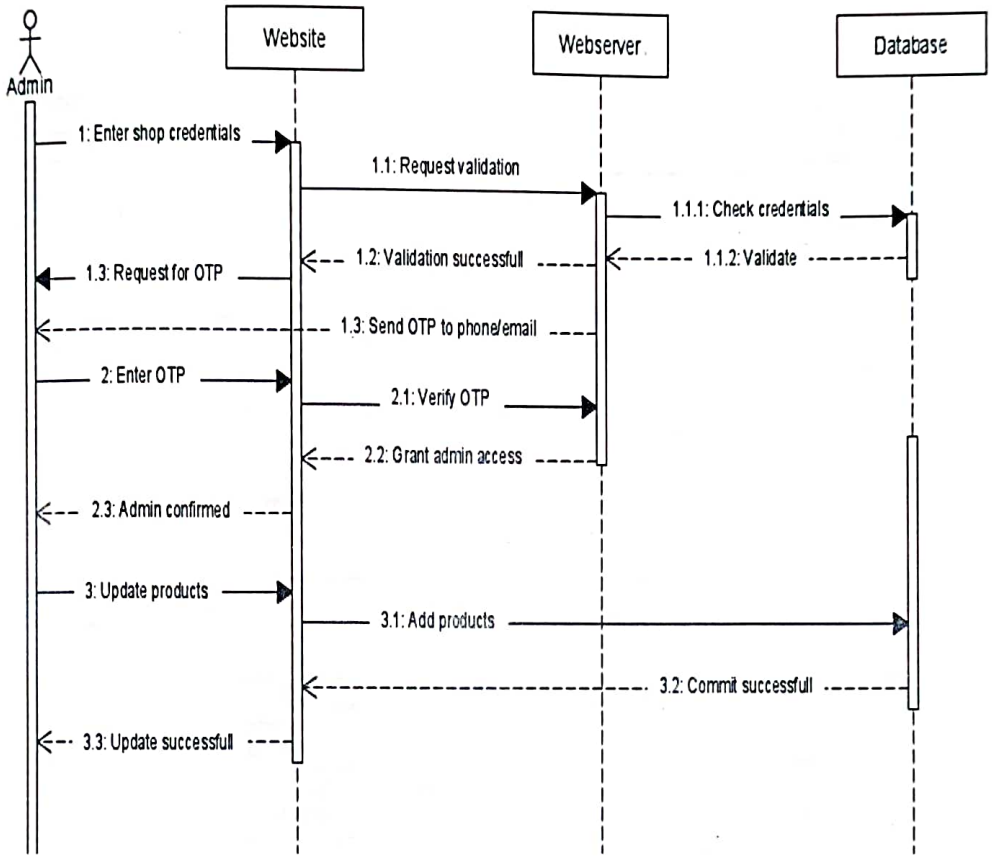
# SYSTEM DESIGN

## 4.1 PROCEDURAL DIAGRAMS

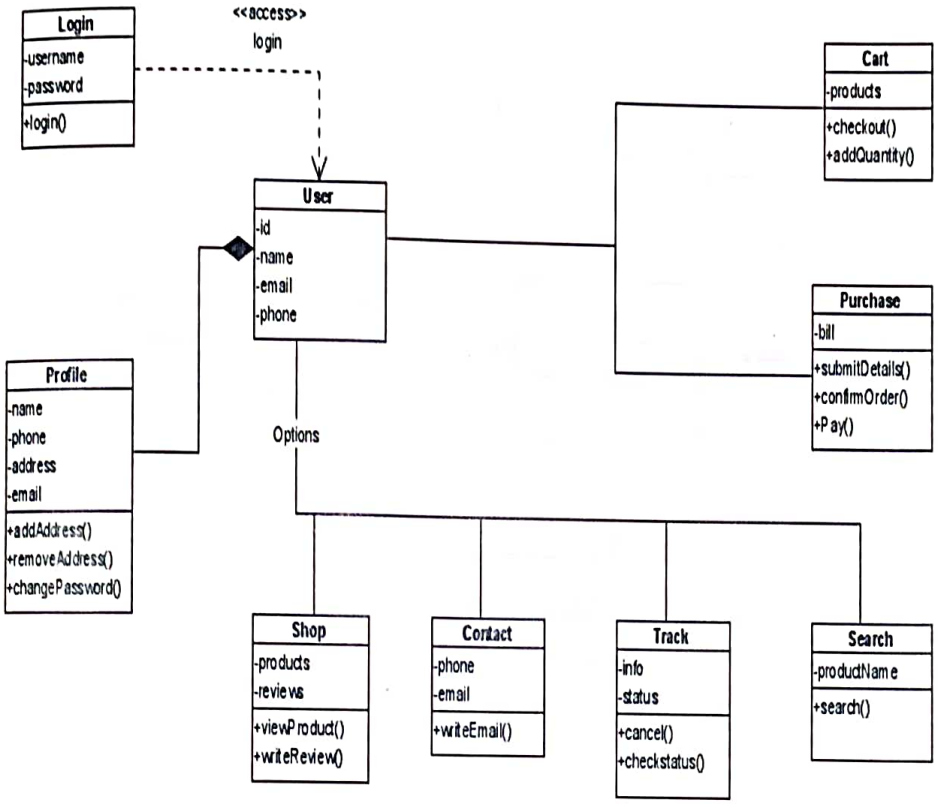
1. **Sequence diagram for user sign registration and login.**



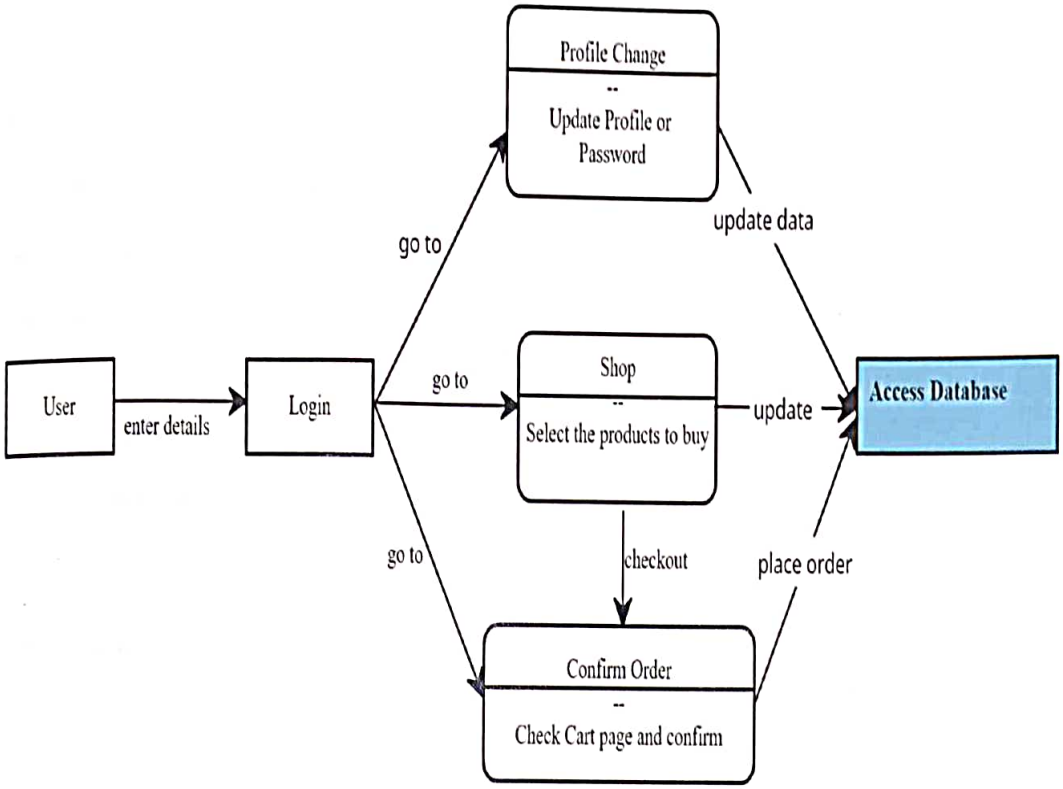
2. **Sequence diagram for admin login.**



3. **Class diagram for the client side.**



## 4. Data Flow diagram



## 4.2 DATA DESIGN

**User data**

| Keys |
|---|
| user_id |
| user_name |
| email_address |
| contact_no |
| password |
| address |

**Product data**

| Keys |
|---|
| Product_id |
| Product_name |
| Product_availability |
| Product_image |
| Product_price |
| Product_description |
| Product_category |

**Cart data**

| Keys |
| --- |
| Product_id |
| Product_name |
| Total_products |
| Total_price |

**Order data**

| Keys |
| --- |
| Order_no |
| Product_id |
| User_address |
| Total_products |

**Payment data**

| Keys |
| --- |
| payment_id |
| product_id |
| user_id |
| order_no |
| payment_method |
| Total_price |

# CHAPTER 5

# IMPLEMENTATION

## 5.1 MODEL USED - INCREMENTAL MODEL

Incremental model is a software development model used for delivering the system in small increments to the customer until the best or desired system is not implemented. It is also known as successive version model.

For Om Sai enterprises, initially a system with only the minimal requirements is developed and delivered to the customer and then, as per requirements, the required iterations/ versions will be implemented and delivered to the customer until the desired system is delivered.

## 5.2 DETAILS OF THE TOOLS AND TECHNOLOGIES USED

### 5.2.1 Tools Used

- **Visual Studio Code**

  Visual Studio code is a strong code editor which is available for Windows, Linux and Mac. It is lightweight but powerful and has built-in support for JavaScript, TypeScript and Node JS. It is an Integrated Development environment which can be used for developing a whole project because it has many extensions and support for different languages and technologies.

- **Firebase**

  Firebase is a tool by Google which provides Software Development Kits for building applications on Android, IOS, Web, C++ and Unity. It is Baas(Backend-as-a-Service). For developers, it provides services like authentication, firestore(database), storage(cloud),etc which are very easy to implement using the provided documentation.

### 5.2.2 Technologies Used

- **React JS**

React JS is one of the most popular front-end JavaScript libraries used for creating user interfaces. It has the component architecture which allows developers to create reusable components which can be imported and implemented where ever required.

It does not follow the regular HTML and Vanilla JavaScript syntax, in-place it has a special syntax called as JSX(JavaScript XML) which allows us to write HTML code inside a JavaScript file. It makes development process easy and also helps in building flexible and fast performance applications. It is easy to import and use different packages in React JS.

- **Tailwind CSS**

Tailwind CSS is an highly customizable utility-first CSS framework. It provides utility classes for different CSS properties which we can use directly in our HTML class. Thus, it reduces time used for writing CSS.

It is a low-level CSS framework because it has all the things which are used for creating a custom designed application without having any default styles for elements which we have to fight to override.

- **npm**

npm or Node Package Manager is the Node JS package manager. It is a repository for all the available free and paid-private packages used for development called as the npm registry. It is one of the largest software registries which has over 8 lac packages. To use these packages we have to just install them through command line interface using an npm command and import them in our project.

- **Chart JS**

Chart JS is a JavaScript run-time package which helps in creating charts and graphs very easy. One can simply install it via npm or download it from GitHub and use it in their project.

- **React Router**

React Router is a routing library for React JS. It is used to create multi-page single page application. It helps to render different JavaScript components based on different URLs on a single page.

- **Razor Pay**

Razor pay is an API for payments. It offers payment gateway for digital-transactions.

## 5.3 DETAILS OF HARDWARE USED

| Components | Hardware Used |
|---|---|
| Processor | AMD Athlon 3000G with Radeon Vega Graphics 3.50 GHz. |
| RAM | 4.00 GB |
| System type | 64-bit operating system, x64-based processor |

## 5.4 SOURCE CODE SNIPPETS

**Main App.js file**

```
JS App.j...  ×
src > JS ... > ...
import React, { useSta... ...ateContext } from "react";
import "./index.css";
import "bootstrap-ico... ...ootstrap-icons.css";
import { Route, Routes, ...ate } from "react-router-dom";
import AllProducts fr... ...ut/AllProducts";
import Home from "./pa... ";
import AdminDashboard ... ...ages/AdminDashboard";
import MyCart from "./... ...s/User/MyCart";
import TodoOrderList f... ...mponents/Admin/TodoOrderList";
import EditContent fro... ...ponents/Admin/EditContent";
import EditCarousel fr... ...ponents/Admin/EditCarousel";
import EditProducts fr... ...omponents/Admin/EditProducts";
import { onAuthStateChanged } from "firebase/auth";
import { auth } from "./components/Authentication/firebase-config";
import Login from "./components/Authentication/Login";
import Register from "./components/Authentication/Register";
import AddCarouselSlide from "./components/Admin/AddCarouselSlide";
import AddNewProduct from "./components/Admin/AddNewProduct";
import ProductDetails from "./components/User/ProductDetails";
import ExistingProducts from "./components/Admin/ExistingProducts";
import Analytics from "./components/Admin/Analytics/Analytics";
import Orders from "./components/Admin/Orders";
import EditExistingProducts from "./components/Admin/EditExistingProducts";

const UserContext = createContext(null);

function App() {
  const [user, setUser] = useState(null);
  const [cartItems, setCartItems] = useState([]);

  onAuthStateChanged(auth, (currentUser) => {
    if (currentUser) {
      setUser(currentUser);
    }
  });
```

```
JS App.js M ×
src > JS App.js > ...
    return (
      <>
        <UserContext.Provider value={user}>
          <Routes>
            <Route path="/" element={<Navigate to="/home" />} />
            <Route
              path="/home"
              element={<Home cartItems={cartItems} setCartItems={setCartItems} />}
            />

            <Route path="/shop" element={<AllProducts />} />
            <Route path="/product/:id" element={<ProductDetails />} />

            <Route path="/ose_admin" element={<AdminDashboard />}>
              <Route path="" element={<Analytics />} />
              <Route path="edit" element={<EditContent />}>
                <Route path="carousel" element={<EditCarousel />}>
                  <Route path="addSlide" element={<AddCarouselSlide />}></Route>
                </Route>

                <Route path="products" element={<EditProducts />}>
                  <Route path="" element={<ExistingProducts />} />
                  <Route
                    path="editExisting/:id"
                    element={<EditExistingProducts />}
                  />
                  <Route path="addProduct" element={<AddNewProduct />} />
                </Route>
              </Route>
              <Route path="orders" element={<Orders />} />
              <Route path="todo" element={<TodoOrderList />} />
            </Route>

            <Route path="/cart" element={<MyCart cartItems={cartItems} />} />

            <Route path="/register" element={<Register user={user} />} />

            <Route path="/login" element={<Login user={user} />} />
          </Routes>
        </UserContext.Provider>

      )
    }

    export { App as default, UserContext };
```

## Registration page

Register.jsx

src > components > Authentication > Register.jsx > Register

```
import React, { useState } from "react";
import { Link, useNavigate } from "react-router-dom";
import AuthNavbar from "../User/AuthNavbar";
import { auth } from "./firebase-config";
import { createUserWithEmailAndPassword, updateProfile } from "firebase/auth";

const Register = (props) => {
  const [registerEmail, setRegisterEmail] = useState("");
  const [registerPassword, setRegisterPassword] = useState("");
  const [displayName, setDisplayName] = useState("");
  const [error, setError] = useState({ isError: false, msg: "" });

  const navigate = useNavigate();

  const handleRegister = async (e) => {
    e.preventDefault();
    try {
      await createUserWithEmailAndPassword(
        auth,
        registerEmail,
        registerPassword
      );
      await updateProfile(auth.currentUser, { displayName: displayName });
      navigate("/home");
    } catch (error) {
      setError({ isError: true, msg: error.code });
      setTimeout(() => {
        setError({ isError: false, msg: "" });
      }, 5000);
    }
  };
```

## Login page

Login.jsx

src > components > Authentication > Login.jsx > Login

```
import React, { useState } from "react";
import { Link, useNavigate } from "react-router-dom";
import AuthNavbar from "../User/AuthNavbar";
import { auth } from "./firebase-config";
import { signInWithEmailAndPassword } from "firebase/auth";

const Login = (props) => {
  const [loginEmail, setLoginEmail] = useState("");
  const [loginPassword, setLoginPassword] = useState("");
  const [error, setError] = useState({ isError: false, msg: "" });

  const navigate = useNavigate();

  const login = async (e) => {
    e.preventDefault();

    try {
      await signInWithEmailAndPassword(auth, loginEmail, loginPassword);
      navigate("/home");
    } catch (error) {
      setError({ isError: true, msg: error.code });
      setTimeout(()=>{
        setError({ isError: false, msg: "" });
      }, 5000)
    }
  };
```

## Home page

```
Home.jsx ×
src > pages > Home.jsx > [e] Home
import React from "react";
import Navbar from "../components/User/Navbar";
import HomeSlider from "../components/User/HomeSlider";
import HomeProducts from "../components/User/HomeProducts";
import Footer from "../components/User/Footer";

const Home = (props) => {
  return (
    <>
      <div className="bg-white pb-10">
        <Navbar items={props.cartItems} logout={props.logout}/>
        <HomeSlider />
      </div>
      <HomeProducts cartItems={props.cartItems} setCartItems={props.setCartItems}
      setProductDetails={props.setProductDetails} />
      <Footer />
    </>
  );
};

export default Home;
```

## Admin dashboard

```
AdminDashboard.jsx
src > pages > Admin[illegible]nboard.jsx > ...
import [illegible] from "reac[illegible]
import [illegible]inNavbar fro[illegible]components/Admin/AdminNavbar";
import [illegible]inSidebar fr[illegible]/components/Admin/AdminSidebar";
import {Outlet} from [illegible]router-dom'

const AdminDashboard = [illegible] {
  return (
    <>
      <AdminNavbar />
      <div class[illegible]="flex">
        <AdminSidebar />
      <Outlet />
      </div>

    </>
  );
};

expor[illegible]fault AdminDashboard;
```

*Products page*

```
AllProducts.jsx ×
src > pages > AllProducts.jsx > AllProducts > handleSearch
import React, { useState } from 'react'
import FilterProducts from '../components/User/FilterProducts';
import Footer from '../components/User/Footer';
import Navbar from '../components/User/Navbar';
import ShopProducts from '../components/User/ShopProducts';
import ProductData from "../product-data.json";

const AllProducts = () => {
  const [productData, setProductData] = useState(ProductData.homeCardProductData)

  const handleSearch = (searchText) =>{
    let newProductData = [...productData];
    newProductData = newProductData.filter(item => item.productName.toLowerCase().includes
    (searchText.toLowerCase()));
    setProductData(newProductData);
  }

  return (
    <>
    <Navbar onSearch={handleSearch} />
      <div className='flex bg-neutral-800 '>
      <FilterProducts />
      <ShopProducts productData={productData} />
      </div>
    <Footer />
    </>
```

# CHAPTER 6

# TESTING

## 6.1 TESTING APPROACH

Software testing is used for assessing a software item to find the discrepancies between the specified input and the expected output. It assesses the quality consistency of the product made. The testing is done through all the stage of the development life cycle. It is a verification and validation approach (V&V Model).

- **Verification**

  Verification is the method to check if the product is made according to the requirements given by the customer at the start of the development cycle. To put in another way, to check if the product performs the way we want it to.

- **Validation**

  Validation is the procedure to ensure the product satisfies the stated requirements at the end of the phase growth. That is to say, to ensure the product is made as per customer requirements.

Testing goes hand in hand with execution to ensure that system works accurately and efficiently. The most common type of testing is done by the user who executed the program. The application that is developed should be valid. It is in this phase, all the bugs and errors made in the previous step of the SDLC are solved.

System testing helps in uncovering the weaknesses that were not detected in the earlier stages of the development life cycle. The program is executed to find errors intentionally. This includes stress testing, boundary value analysis etc. For the purpose of testing, various test cases are generated.

There are many approaches to test the software:

- Unit Testing
- Integration Testing

## 6.2 UNIT TESTING

It is the smallest executable component of the software including codes, classes and methods. They're individually checked for correctness. It is validation technique which uses the methods used in black box where the internal coding is not a concern. The individual components and the units are tested to check that they work individually as defined.

It requires drivers and stubs as individual units may or may not be executable. It may be performed in debugger mode to find how the variables behave during the execution. Gray box testing examines the code in detail for functioning.

Types of unit testing:

- Black Box Testing – Tests the user interface, inputs and outputs.
- White Box Testing - Tests the function's behavior.
- Gray Box Testing - Executes tests, risks and assessment methods.

The techniques of unit testing applied in the Om Sai Enterprises are as follows:

- All the modules were tested individually.
- All the errors were found and corrected.
- Gray box method of testing was applied.
- It was seen that the buttons were properly functioning.
- It was checked that devices would run when the input is passed.

## 6.3 INTEGRATION TESTING

This Testing of consolidated pieces of an application to decide whether they work together accurately. The parts can be code modules, person's application, customer and server applications on a system, and so forth this sort of testing is particularly pertinent to customer/server and conveyed frameworks.

There are two approaches in Integration Testing:

- **Bottom-up approach**: It focuses on testing the bottom part individual units and modules. It goes upward by integrating the units.
- **Top-Down approach**: The highest degree of the software is tested first, and goes downward until the final component is reached.

## 6.4 APPLICATION OF THE TECHNIQUES IN THE PROJECT

### 6.4.1 Test Plan

A test plan is an itemized report that diagrams the test system, testing destinations, assets (labour, programming, equipment) required for testing, test plan, Test Estimation and test expectations.

The test plan fills in as an outline to lead programming testing exercises as a characterized procedure which is minutely observed and constrained by the test director and check various levels of tests.

Follow the steps below to create a test plan:

- Analyse the product
- Design the Test Strategy
- Define the Test Objectives
- Define Test Criteria
- Resource Planning
- Plan Test Environment
- Schedule & Estimation
- Determine Test Deliverables

**Step 1: Analyze the product**

Is it possible to assess a commodity without any information of it?

The answer is no. A product should be understood properly before being tested. The developer must also understand the end user's needs and their expectations from the product.

Following is the test of the product: Om Sai Enterprises

a) Who will use this application?

The application is applicable for anyone who wants to buy a product from Om Sai Enterprises.

b) What is the use of it?

It allows the user to sit at home, surf the products from the store and then choose and order whatever they want. The ordered product will be delivered directly to their address via courier.

c) What are the development and deployment environment requirements?

i. **Development environment requirements:**

Language: React JS.

Operating system: Windows 7 and above

Code editor: Any code editor.

Packages used: React, Create React app, React Router, Chart JS, Firebase and other npm packages.

ii. **Deployment environment requirements**: Any system with an internet and latest web browser.

**Step 2: Develop Test Strategy**

Test Strategy is an important step in the development of a test plan. An evaluation strategy document is established by the Test Manager. This document defines:

- The testing objectives and the ways to achieve them.
- It is also determining the testing effort and costs.

### Step 3: Define Test Objective

Test aim is the goal and accomplishment of carrying out the check. The aim of testing is to find as many software flaws as possible; make sure that the program test is bug free before it releases.

Following are the test objectives of Om Sai enterprises - e-store:

- The principal target of the software is the proper functioning of all the .
- The end users must be satisfied with the final output of the software.

### Step 4: Define Test Criteria

Test Criteria is the condition on the basis of which the test process can be constructed.

There are 2 types of test criteria as following:

- **Suspension Criteria**: Decide the critical suspension criteria for test. When the suspension criterion is met during testing, the current active test cycle is suspended until the criterion is solved.
- **Exit Criteria**: This is the criterion which signifies the completion of a test phase. They are the targeted outcomes of a test.

### Step 5: Resource planning

It is a summary of all the resources which are required for an active completion of the project. The resources can be human, equipment or materials. It helps in understanding the mass of capital required. Hence it helps the test manager to make an accurate schedule.

### Step 6: Plan Test Environment

The testing workbench consists of the software, hardware on which the test team performs the test cases and scenarios. It consists of business and user environment, and the physical environments.

**Step 7: Test Deliverables**

It is the listing of documents, tools and the components that has to be stored for further reference.

Test deliverables to be provided before the testing phase:

- Test plans, Test cases and Test design document.

Test deliverables that are to be provided during the testing:

- Test Scripts, Test Data and Simulator.
- Error logs and execution logs.

Test deliverables that are provided subsequently after the testing cycles are over:

- Test Results/reports.
- Defect Report.

### 6.4.2 Test Cases

The test case represents the collection of acts that are to be executed to check a particular feature or functionality of the software application.

Typical Parameters of Test case:

i. Test Case ID

ii. Test Case Description

iii. Input Given

iv. Condition

v. Expected Result

vi. Actual Result

vii. Status (Fail/Pass)

viii. Note

| Project Name: E-store for Om Sai Enterprises | |
|---|---|
| Test Case | |
| Test Case ID: 1 | Test Designed By: Mandar Junnarkar |
| Test Priority: High | Test Design Date: 20/2/22 |
| Test Executed By: Mandar Junnarkar | Test Execution Date: 20/2/22 |

**Test Cases**

| Sr. No | Page/Form | Input | Action | Expected Output | Actual Output | Test case | Test Comment |
|---|---|---|---|---|---|---|---|
| 1. | Registration page | Email Id: mandar@gmail.com Password: 123456 | Click on submit button | user should redirect on login page | user will redirect on login page | Pass. | Email id & password id valid. |
| 2. | Registration page | Email Id: mandar@gmail.com Password: 1234abc | Click on submit button | User should not get access | System will display invalid username or password | Pass. | Password is invalid. |
| 3. | Login page | Email: mandar@gmail.com Password: 123456 | Click on submit button | user should redirect on homepage | user will redirect on homepage | Pass. | Username & password is valid. |
| 4. | Login page | Username: mandar@gmail.com Password: Abcd1234 | Click on submit button | User should not get access | System will display invalid username or password | Pass. | Password is invalid. |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5. | Login page | Email: abc@gmail.com Password: ab123456 | Click on submit button | User should not get access | System will display invalid username or password | Pass. | Invalid username or password. |
| 6. | Home page | Search media content by title: Round rubber stamp | Click on search icon | Searched product should be fetched from the database and displayed | Searched product will be fetched from the database and displayed | Pass. | Valid Title |
| 7. | Home page | Search media content by title: xyz | Click on search icon | No such media file exist | No such media file exist | Pass. | Invalid Title |
| 8. | Logout | Click on logout button | Click on logout button | User should be logged out | User is logged out | Pass | Successfully logged out. |

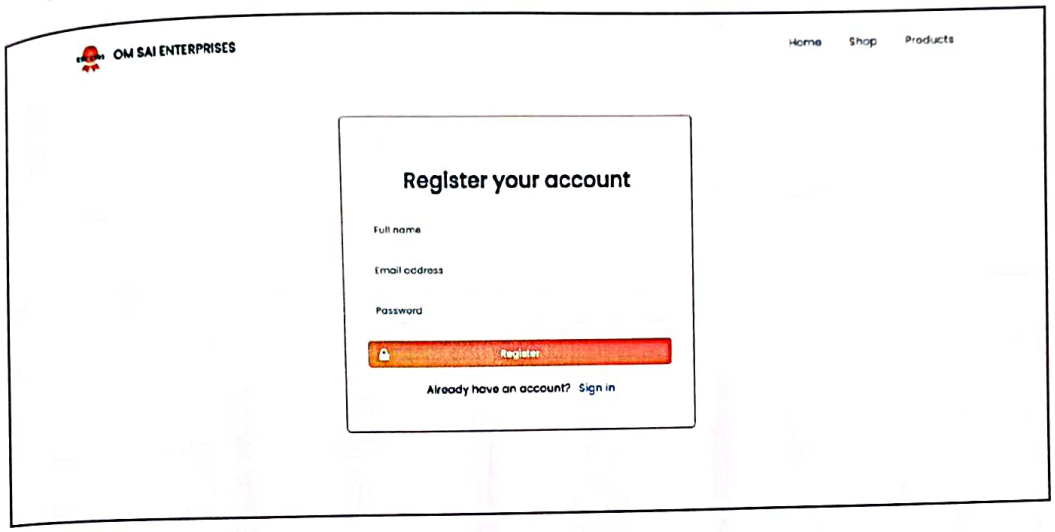
# CHAPTER 7

## UI SCREENSHOTS

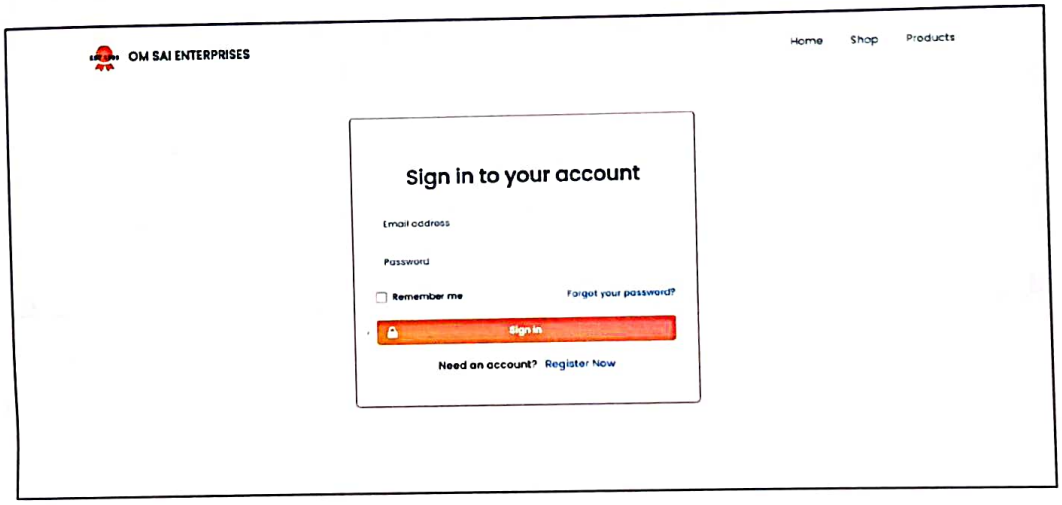
**Following are the screen-shots of the website's UI.**

### 7.1 Authentication

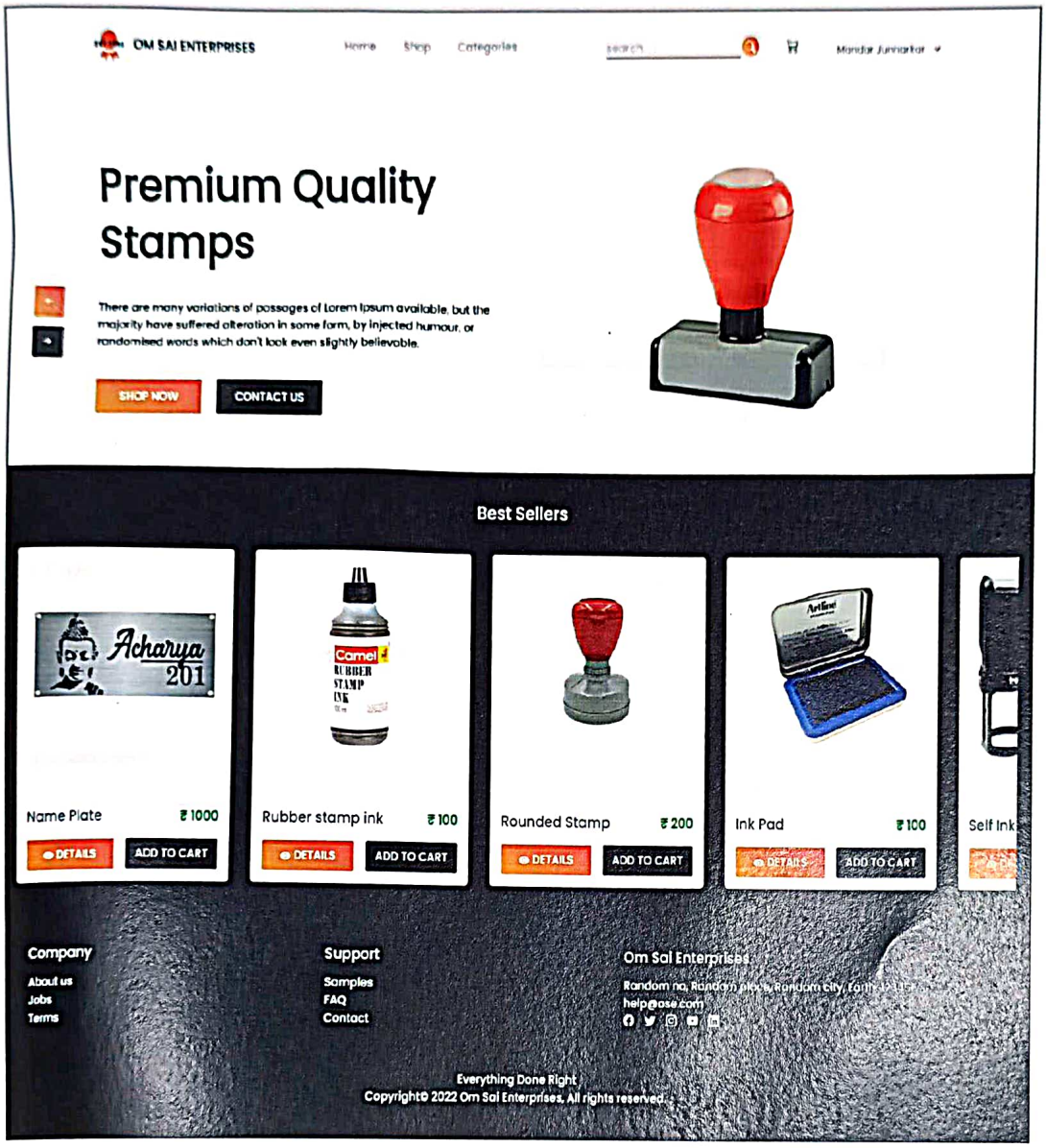
**a) Registration**



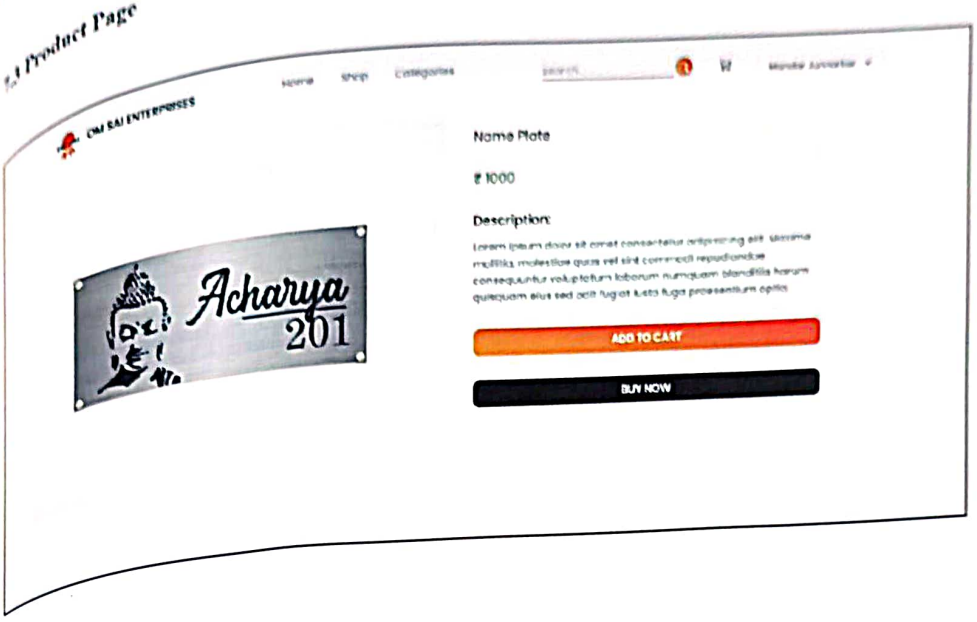

**b) Login**



## 7.2 Home page

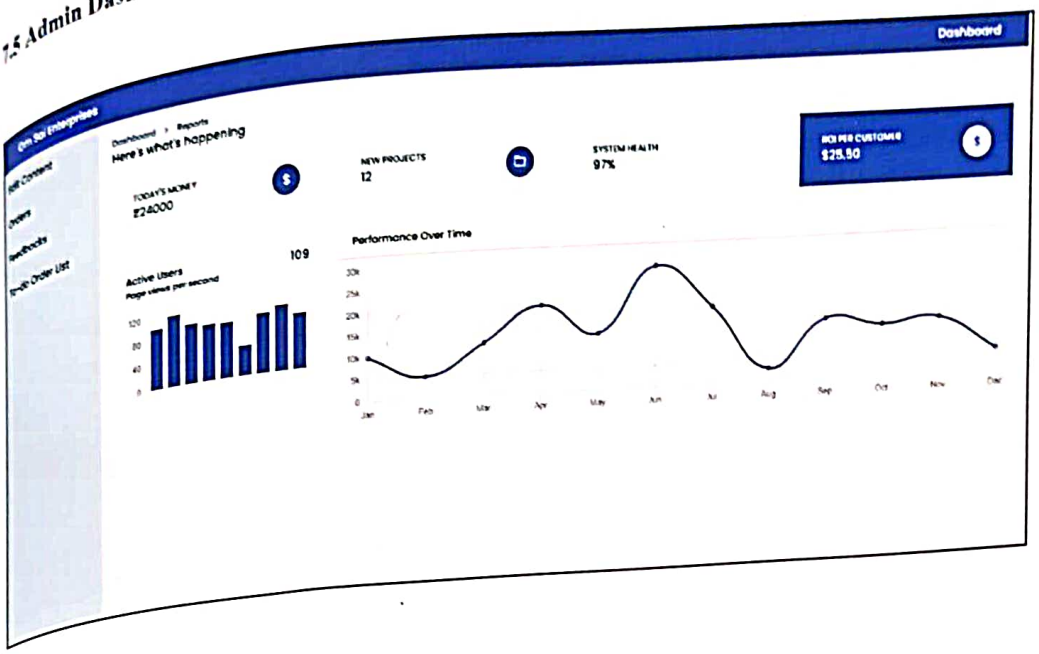

OM SAI ENTERPRISES
Home
Shop
Categories
Premium Quality Stamps
There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.
SHOP NOW
CONTACT US
Best Sellers
Name Plate
₹ 1000
DETAILS
ADD TO CART
Rubber stamp ink
₹ 100
DETAILS
ADD TO CART
Rounded Stamp
₹ 200
DETAILS
ADD TO CART
Ink Pad
₹ 100
DETAILS
ADD TO CART
Self Ink
Company
About us
Jobs
Terms
Support
Samples
FAQ
Contact
Om Sai Enterprises
help@ose.com
Everything Done Right
Copyright© 2022 Om Sai Enterprises, All rights reserved.

## 7.3 Product Page



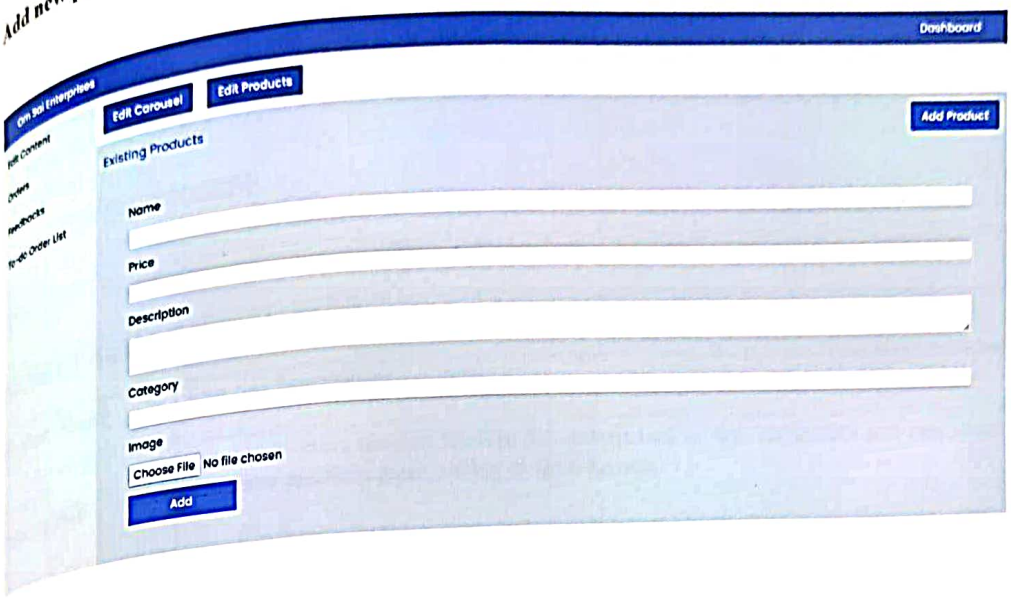

## 7.4 Cart

## 7.5 Admin Dashboard




## Edit existing products

Add new product



# CHAPTER 8

## RESULTS AND DISCUSSION

The test results are promising as all the test cases are passed and the proposed system can go through all the conditions set while testing it. It can face problematic situations and can work fine in different & difficult conditions.

### 8.1 PROJECT INFORMATION

| Project Overview | | | | |
|---|---|---|---|---|
| **PROJECT BASIC INFORMATION** | | | | |
| **Project Name** | Om Sai Enterprises - e store | | | |
| **Project Description** | An online store created for Om Sai enterprises so that customers can surf their products and buy them sitting at their homes. | | | |
| | **Project Type** | | Web Application | |
| **Project Duration** | **Start date** | 1st July 2021 | **End date** | 29th March 2022 |

### 8.2 TEST SUMMARY

| | | | | | |
|---|---|---|---|---|---|
| EXECUTED | PASSED | | | | 8 |
| | FAILED | | | | 0 |
| | *(Total) TEST EXECUTED (PASSED + FAILED)* | | | | 8 |
| PENDING | | | | | 0 |
| IN PROGRESS | | | | | 0 |
| BLOCKED | | | | | 0 |
| *(Sub- Total) TEST PLANNED* | | | | | 8 |
| *(PENDING + IN PROGRESS + BLOCKED + TEST EXECUTES)* | | | | | |

# CHAPTER 9

# CONCLUSION

## 9.1 PROJECT CONCLUSION

**Om Sai Enterprises - e store** is a live project for a shop in Andheri, Mumbai called Om Sai Enterprises. The main aim behind developing the web application is to provide the customers to surf and buy products from the store online.

The web application has successfully been developed using the latest web technologies. Upgrading or making changes in the current version is easy and can be done as per requirement. Incremental model will be followed for further development.

The project is ready and all the functions are working as per need. All the workflows are managed properly and the user experience is designed properly. Once the shop owner is ready with all the product photographs and details and sets up the courier delivery service, the project will be deployed on a secured web domain.

## 9.2 LIMITATIONS

There are many possible limitations which can cause the application to not function as designed.

Following are some of them:

- Use of an older version browser which does not support the used technologies might cause the application to crash or not load as designed.
- Poor internet connection might cause failure in online payments and money can get stuck in the gateway.
- Also, because of poor internet, data from the database might take time to load.
- As firebase is used, any problem in their online server might cause failure.

## 9.3 FUTURE SCOPE

As mentioned before, the project uses incremental approach and will always be up for further version increments for new features so as to enhance the user's experience.

One of the examples for development in future is to create 3D models of the products and let the user view the product in a 3D view instead on just an image. So, for buying a customized product the customer will be able to see the exact look of the product.

## 9.4 ACHIEVEMENTS

- Learned latest web development tools like React JS, Tailwind CSS, Vanilla JS and Firebase.
- Learned UI/UX design for enhancing user's experience.
- Successful in deploying the website on the internet.
- Made an online e-store where customers can buy products online and the admin can manage the POS(point of sales).

## CHATPER 10

## BIBLIOGRAPHY

**Title of the project: E store for Om Sai Enterprises**

**Project type: Web application**

**Project Developed by: Mandar Ravi Junnarkar**

### REFERENCES


**React JS: https://reactjs.org/**

**https://create-react-app.dev/**

**React router: https://reactrouter.com/**

**Tailwind CSS: https://tailwindcss.com/**

**Chart JS: https://www.chartjs.org/**

**Firebase: https://firebase.google.com/**

**Icons: https://getbootstrap.com/docs/5.0/extend/icons/**

**Temporary hosting: https://www.heroku.com/**

**Packages: https://www.npmjs.com/**

**https://stackoverflow.com/**