TITLE OF THE PROJECT:

UIGO (Mobile Phone Shopping System)

Student Details

Name of the Learner: Sorabh
Programme Code: MCA
Enrolment Number: 1XXXXXX98
Regional Centre Code: 39 (Noida)
Course Code: MCSP60
Mobile No.: XXXXXXXX50
E-mail ID: ssorabh.ssharma@hotmail.com

Mobile Phone Shopping System

Abstract

UIGO Mobile Phone Shopping System is designed to target to increase sales as well provide easy access to consumers for ordering. This website aims at creating a user friendly and secure user interface to selling various mobile phones at affordable price.

Online Stores have taken more and more place in our economic and social life and have also taken the place of traditional bazaars. Shopping markets are not only place to shop. They also have become places for having fun and spending time. Consumers spend a considerable time in shopping sites because they host a lot of items and activities that attract consumers attention and offer a lot of benefits. Shopping stores of Mobile phones have become the most attractive places to shop and hang out. Also Mobile phones consumers fill youngsters needs from supermarkets found in shopping stores. The aim of this research is to investigate the role of UIGO Mobile Phone shopping site in consumers life through factors affecting visits in local shops and to demonstrate how these factors differ according to consumers shopping habits in online shopping of mobile phone and consumers demographic characteristics.

In this context, a survey was conducted with Mobile phone consumers. The data obtained from 501 consumers via the internet survey method using convenience sample as a non-random sampling method were analysed with Independent Sample text and One-Way Communication. Also a principal factor analysis and reliability analysis has been performed. As a result of the research, six factors affecting the consumers visits in the shopping centers were obtained. These factors are named as aesthetics, entertainment, product variety, advantages, socialization and convenience. According to the results, consumer visits in shopping centers vary according to consumers shopping habits and demographic characteristics.

Consumers can come over and search mobile phones list based on different filters provided on website. For example price range, processor, battery, brand, etc. They can add them to his cart, and payment using different payment methods, for example payment gateway, card payment or cash on delivery.

Certificate of Originality

XI. CERTIFICATE OF ORIGINALITY

This is to certify that the project report entitled UIGO (Mobile Phone Shopping System) submitted to Indira Gandhi National Open University in partial fulfilment of the requirement for the award of the degree of MASTER OF COMPUTER APPLICATIONS (MCA) , is an authentic and original work carried out by Mr. / Ms. SORABH with enrolment no. 168615398 under my guidance.

The matter embodied in this project is genuine work done by the student and has not been submitted whether to this University or to any other University / Institute for the fulfilment of the requirements of any course of study.

Signature of the Student:

Date: 11-01-2022

Name and Address of the student

SORABH S/O SC SHARMA, D-23,
JANAKPURI, SHALIMAR
GARDEN, NEAR ESI HOSPITAL,
SAHIBABAD, GZB-201005

Signature of the Guide

Date: 11-01-2022

Name, Designation and Address of the Guide:

SANDEEP KUMAR GAUR,
SOFTWARE ENGINEER,
GURGAON SECTOR 18 NEAR
MG ROAD METRO STATION,
HARYANA

Enrolment No. 168615398

Table of Contents

1. Introduction & Ob0jectives

1.1. Introduction

Demands of mobile phones are high, there are many shops selling mobile phone locally. We are developing a mobile phone shopping system that enables online selling and purchase of products (in this case mobile phones). Company or individual have numerous models of smartphone to sell to the customer.

Here every shop selling mobile phones individually maintains various registers. Those register contains details of customer's details, inventory items, sales, payments, etc. and it is really a time consuming process. We are manually checking inventory for available handset and also manually attending each and every customers, describing details about item, and offering price. This interactive process required large man force to activities.

Online Shopping is getting popular now days. We are developing a Web application that depict online shopping for mobile phones and purchasing using Payment Gateway. Using this software, company can improve the efficiency of their services and manage records at one place.

1.2. Objectives

The main objective of our system is to provide user friendly e-commerce platform for selling and purchasing mobile phones online. The system helps entrepreneur to manage and track records of sales as well as this software helps customer to find different mobiles, their features, and new updates easily.

The Central objective is for increase selling as well making business much simple to manage, putting all records available on one place for authorized agents or partners of the business. System will help us to manage records of customers details, inventories records, sales records. These records could help us in near future to predict better investment plans in future, for examples we have a list of happy customers we can offer them better deals, we have previous year records of sales and remaining inventories those can help us in planning of supply and demands.

It is designed such a way that our customer's or someone interested in our mobile phone can view all the updates of the mobile from any place through online. The software will help in easy maintaining and updating products in the website for the management. Overall system will benefits us more in near future on both perspective, whether managing business or our customers for quickly purchase mobile phones from our website.

2. System Analysis

2.1. Identification of Need

Technologies are improving day by day, many businesses are now operating from some kind of online store where both retail price or wholesale price are offered to customers, if you are not upgrading yourself then other business may overtake you or they will have upper hand on you.

Due to Covid 19 pandemic local business are suffering. Most of customer's are vanished from local stores and interested for services, where they can order from their home and collect delivery at home, this is only possible by providing an online store.

Manufacture of mobile phones can collect records to plan and forecast future production quantity based on previous year sales records, also individual Shop owner also maintain their inventory based on previous year sales records.

Customer services can be improved to next level because authorised person have all details to contact them, conduct survey or review their ratings. These little direction will guide management to improve their services, it is said that better services always attract more customers. More customers means more sales and profit in business.

Advantages

- It enables easy and fast access to the website.
- This system efficiently store inventories.
- Ensure efficient and reliable communication with the customer.
- Security measures to avoid unauthorized access to customer records.
- Easy to use and user-friendly.
- It will attract more guests to site and increase our customer's.
- It will reduce data entry and processing errors.
- It will help us for future planning.
- It will improve our customer service due to better business management.

2.2. Preliminary Investigation

The preliminary investigation is the first step of the Software Development Life Cycle. The Preliminary investigation determine the feasibility of the system and investigation is to evaluate project requests.

UIGO Mobile Phone Shopping System is a web based online selling system which sell mobile phones to the customers. If a person want to purchase mobile phone then he can order it anywhere(home, office, etc), only requirement to access of internet connection on phone, laptop, etc. If he is confused or have questions, he can also get details like phone, email, address, etc. He can visit local store or communicate via phone, email, etc.

2.3. Feasibility Study

The feasibility study is the second step of the Software Development Life Cycle. For a new System, the requirements process must conduct a feasibility study about it. The input to the feasibility study is a set of preliminary business requirements, an outline description of the system and how the system is intended to support business processes.

A feasibility study is defined as an estimate or analysis of the possible impact of a proposed system. On the other hand we can say that whether decision makers can implement the project according to the customer's requirements or not. The feasibility study will include wide data associated to financial and operational impact. The feasibility study is conducted to support the decision-makers in creating the decisions:

- What will be the great importance of the UIGO System?
- How to improve the quality of developing system?

Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

2.3.1. Economical Feasibility

Economic analysis is used frequently for the evaluation of the effectiveness of the system. It is also know as cost/benefit analysis for specify possibility to deduce cost.

The procedure is to determine the benefit and saving that are considered from a system and compare them with costs, decisions is made to design and implement the system. This part of feasibility study gives the top management the economic justification for the new system.

This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.

- All hardware and software has to be conducted by the organization.
- What we have approximated will surely overcome the initial costs and later on running costs for the system.
- Using the minimum of time or resources necessary for effectiveness development.
- Specify to avoiding wastage of resources.

Every project is subjected to five different feasibility tests.

1. **Economic:**

   Is it possible to complete this project within the budget approved by upper management and stakeholders?

   UIGO Mobile Phone Shopping System will use existing framework(Spring-boot, Hibernate) and Design Patterns(MVC Pattern, Factory Pattern, etc) that boost development process and cut down so much time. Reduced time means reduced cost.

2. **Schedule:**

   Determine whether or not the project can be completed within the time-frame provided.

   We are using existing framework and famous design Patterns in our UIGO Mobile Phone Shopping System, this will also help us to complete within the timeline.

3. **Legal:**

   Can this project meet the requirements of cyberlaw as well as other regulatory compliances?

   There are no legal law that prohibited development of online shopping site, we will follow legal law of the land as per government guidance.

4. **Technical Feasibility:**

   Determine whether the software is compatible with the current computer system.

   This included the study of functions, performance, and constraints that may affect the ability to achieve an acceptable system. For this, we studied complete functionality to be provided in the UIGO Mobile Phone Shopping System and checked if all functions were working correctly using different types of frontend and back-end platforms.

5. **Operational Feasibility:**

   Are we able to create the operations that the client expects?

   The proposed system is web based system that is very user friendly. The activities of the system such as data entry, information retrieval, updating and deletion of records from various tables etc. are made easy. All the operators of this project are trained in this area. So this project is having operational feasibility.

2.4. Project Planning

Project planning is the process of defining your objectives, scope, goals and milestones (deliverables), and it assign specific piece of work required to be done as a duty or for a specific fee and budgetary resources for each step. A good plan is easy to share with peoples necessitated, it is most useful when it is revisited regularly. It defines the process by which the objectives will be achieved, and the responsibilities of carrying out each tasks.

There are some important aspects to be considered in project planning

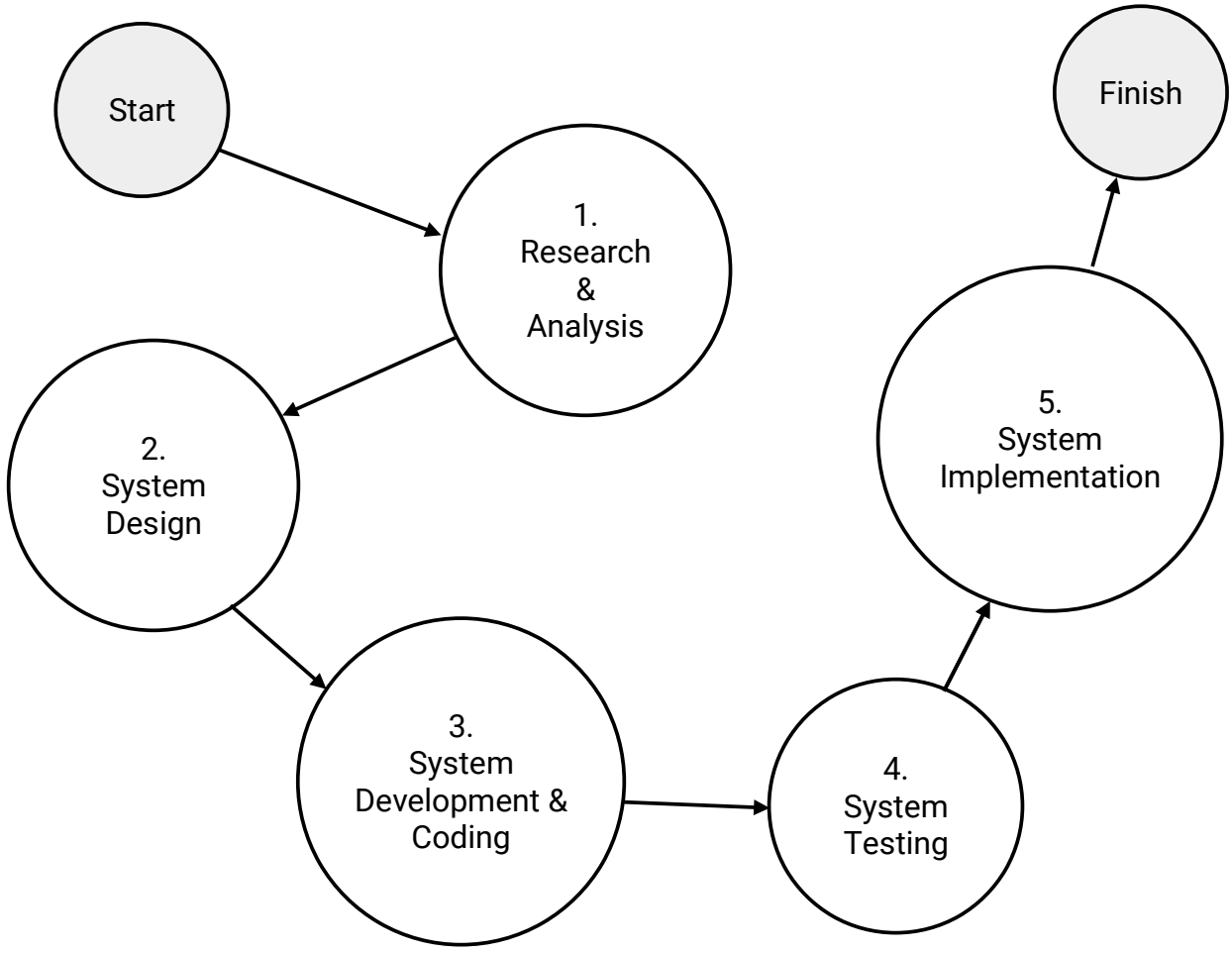
- **Involvement of Customers:** The customer on UIGO Mobile Phone Shopping site must be able to interact with the system in order to perform self related operations i.e. search mobile phones, order mobile phone, view their orders and cancel their orders.
- **Resources:** The resources are the important part of the project. Without resources the progress of the project is impossible. So, any resources related to the development of the system, if available, is to be considered prior to the development stage.
- **Project Phases:** Deciding project phases is a very important task as we can move forward only if we have the total estimates of the number of phases to be made.
- **Risk Management:** Each and every type of risk like hardware, information, technology, etc. It has to be identified through the risk assessment tools.
- **Requirement Analysis:** In the requirement analysis we have to identify different methods, techniques, resources and different tools that can be helpful and beneficial for the project management.

2.5. Project Scheduling

2.5.1. Gantt Chart

Tasks	Jan	Feb	Mar	Apr	May	June
Research & Analysis						
System Design						
System Coding						
System Testing						
System Implementation						

2.5.2. Pert Chart



2.6. Software Requirement Specifications(SRS)

2.6.1. Introduction

The following subsections of the Software Requirements Specifications (SRS) document provide an overview of the entire Software Requirement Specifications.

- **Purpose**

The Software Requirements Specification(SRS) will provide a detailed description of the requirements for the UIGO Mobile Phone Shopping System. This SRS will allow for a complete understanding of what is to be expected from the shopping system to be constructed. The clear understanding of this system and it's functionality will allow for the correct software to be developed for the end user and will be used for the development of the future stages of the project.

- **Scope**

Scope of the project is very broad as compared to manually selling Mobile Phones. Our project aims at automated mobile phone shopping process, i.e. we have tried to computerize various processes of e-commerce system.

  - The customers can fill the various forms and the number of copies of the forms can be easily generated at a time.
  - The system generates types of information that can be used for various purposes.
  - To utilize resources in an efficient manner by increasing their productivity through automation.
  - It satisfy the user requirements.
  - Have a good user interface.
  - Easy to understand and operate by the user.
  - Reduce the cost.
  - Delivery on schedule within the budget.

- **Definitions, Acronyms, and Abbreviations**
  - Software Requirements Specification
  - Mobile Phone Shopping System
  - End users – The people who will be actually using the system.

- **Overview**

The SRS is organized into two main sections. The first is the overall description and the second is the specific requirements. The overall description will describe the requirements of the mobile phone shopping system from a general high level perspective. The specific requirements section will describe in detailed requirements of the system.

2.6.2. The Overall Description

It describes the general factors that affect the product and its requirements. This section does not state specific requirements. Instead it provides a background for those requirements, and makes them easier to understand.

- **Product Perspective**

  The UIGO Mobile Phone Shopping System is an independent stand-alone e-commerce system. This system stores the following informations in the database:

  - **Software Interfaces:** All databases will be configured using MySQL. These databases include phone details, order details, payment details, user details, user chat messages and user feedback details. These can be modified by the end users.
  - **Phone Details:** The phone database will include the name and description about phone, it price, if they are in stock or available to order. After order is been placed, they can request for returning.
  - **Customer Description:** It includes customer id, name, address, phone number, order date, order return request, chats with manufacture, and customer feedback & rating. This information may be used for keeping the records of the customer for any emergency or for any other kind of information.

- **Product Functions**

  - Order Mobile Phones.
  - Search Mobile Phones.
  - Allowing typing of the customer information via form.
  - Adjust quantity of items in carts.
  - When a customer order a mobile phone, the inventory number will be changed to deduction of one inventory available in the database.
  - Ability to modify a Mobile phones specification can be next.
  - Displaying out of stock for those items have zero inventory holdings.
  - Display items in carts after add to cart button clicked on a product.
  - Allows for space to write customer's feedback about purchased items.
  - Allows addition, deletion and modification of information for phones and rates, menu items and prices, user profiles
  - Creation of users and assigning passwords, password recovery via otp email.

- **User Characteristics**

  1) Educational level for UIGO System -> Low,
  2) Experience required for UIGO System -> None,
  3) Technical Expertise -> Medium.

- **Assumptions and Dependencies**

  The system is not required to save generated reports.

2.6.3. Specific Requirements

2.6.3.1. Software Requirements

SL	Software	Minimum System Requirement
01	Operating System	Windows
02	Database	MySQL 8.0.13
03	Front-End	HTML5, CSS3, JQuery
04	Back-End	Spring Framework, Java Bean, Spring Context, Spring JDBC, Spring Web, Spring MVC, Spring Security, Thymeleaf.
05	Language	Java, JavaScript, XML
06	Designing Tool	VSCode, Gimp & Inkscape
07	IDE	STS4 & Intellij Community
08	Web Server	Apache Tomcat 9.0.11
09	Web Browser	IE 10 or later, Any Browser
10	Documentation Tool	LibreOffice

2.6.3.2 Hardware Requirements

SL	Hardware	Minimum System Requirements
01	Processor	Intel i5 processor
02	Memory	8 GB RAM
03	Hard Disk	256 GB SSD
04	Monitor	Color Monitor

2.6.3.3 Functional Requirements

- Customer can be able to create account.
- Customer can be able to update password.
- Customer can be able to reset password by email authentication.
- Customer can access list of product details without login.
- Customer must be login to account, before placing orders.
- Customer can be able to cancel order.
- Customer can be able to return any product after purchase.
- Customer can be able to chat with administrator.
- Customer can download invoice.
- System may have functionality to contact and query to customer.
- System must have cart feature to add multiple designs for order.
- System must have option to payment online.
- System must have Administrator to view orders.
- Admin can be able to manage products.
- Admin must have notification about order, he can approve or reject any order.
- Admin can be able to manage orders.
- Admin can be able to manage payments.

2.6.3.4. Non-Functional Requirements

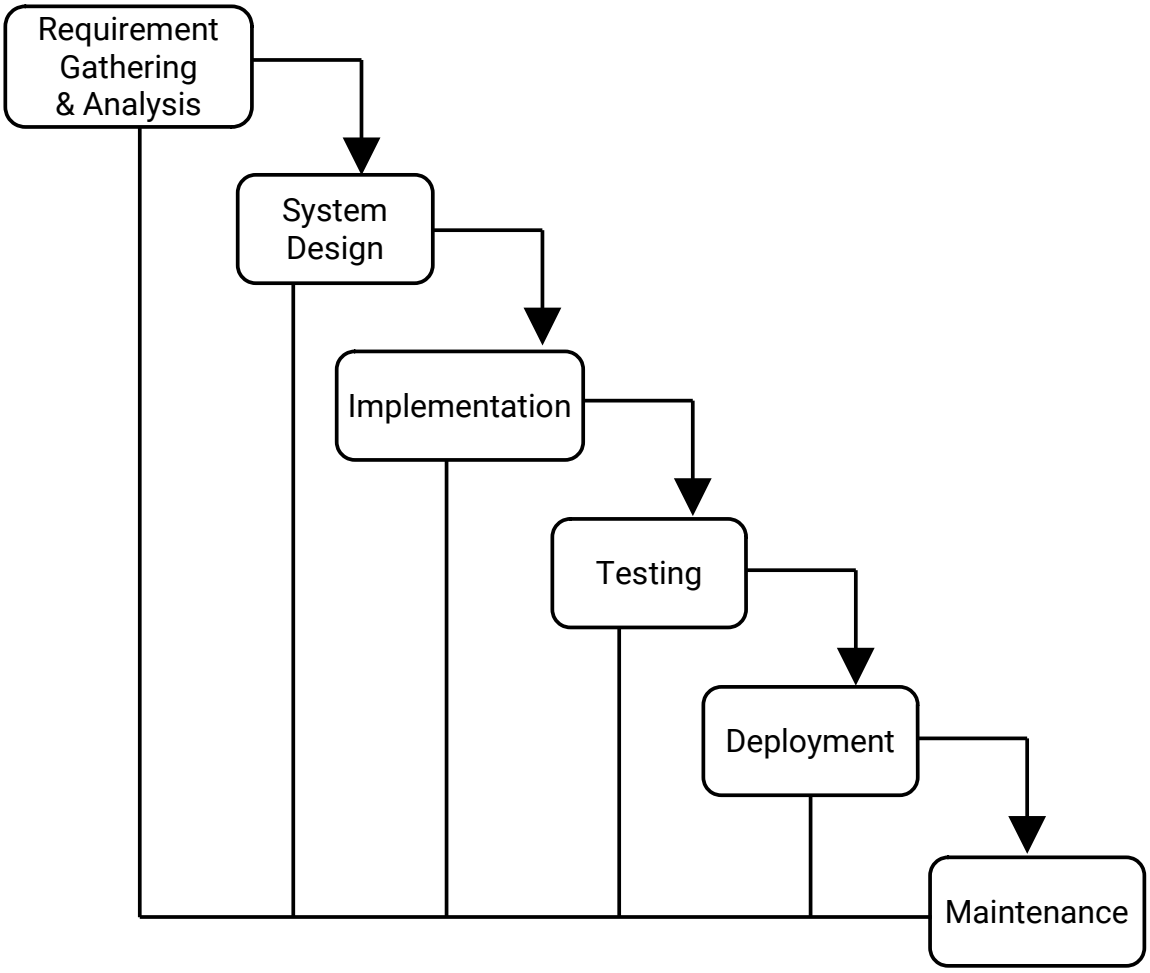
- The proposed system should be providing a good graphical user interface and it should be user friendly.
- The system should be reliable and robust.
- System should be providing the guaranteed security of their customer personal details and ensure its customer details will only retained for future communication only.
- System must be able to handle multiple orders and transaction at the same time.

2.7. Software Engineering Paradigm applied

- **Waterfall Model**

  Waterfall approach(also know as Traditional Approach) was the first Software Development Life Cycle Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

  The following illustration is a representation of the different phases of the Waterfall Model.



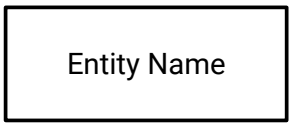
- The sequential phases in Waterfall model are –
  - **Requirement Gathering & Analysis–** All possible requirements of the system to be developed are captured in this phase and documented in a Requirement Specification Document.
  - **System Design–** The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
  - **Implementation–** With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
  - **Integration and Testing–** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
  - **Deployment of system–** Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
  - **Maintenance–** There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

2.7.1. Data Models

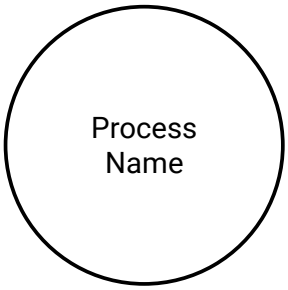
2.7.1.1. Data Flow Diagram (DFD)

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

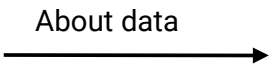
Important Symbols:



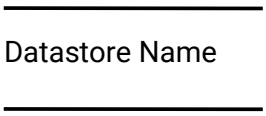
= External Entity, interact with system.



= Process or System/sub-system, it input some data and output response



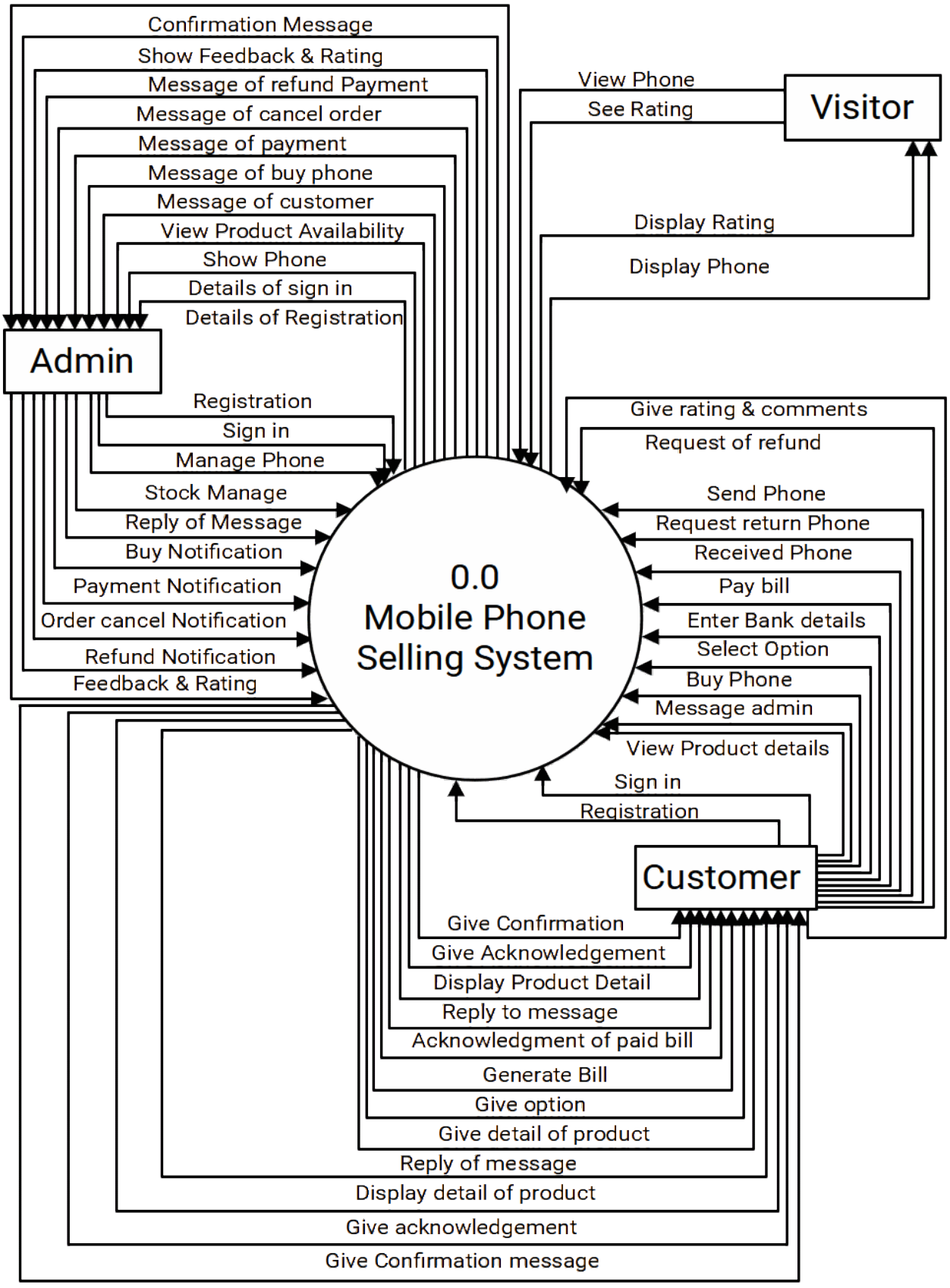
= Data Flow direction and information

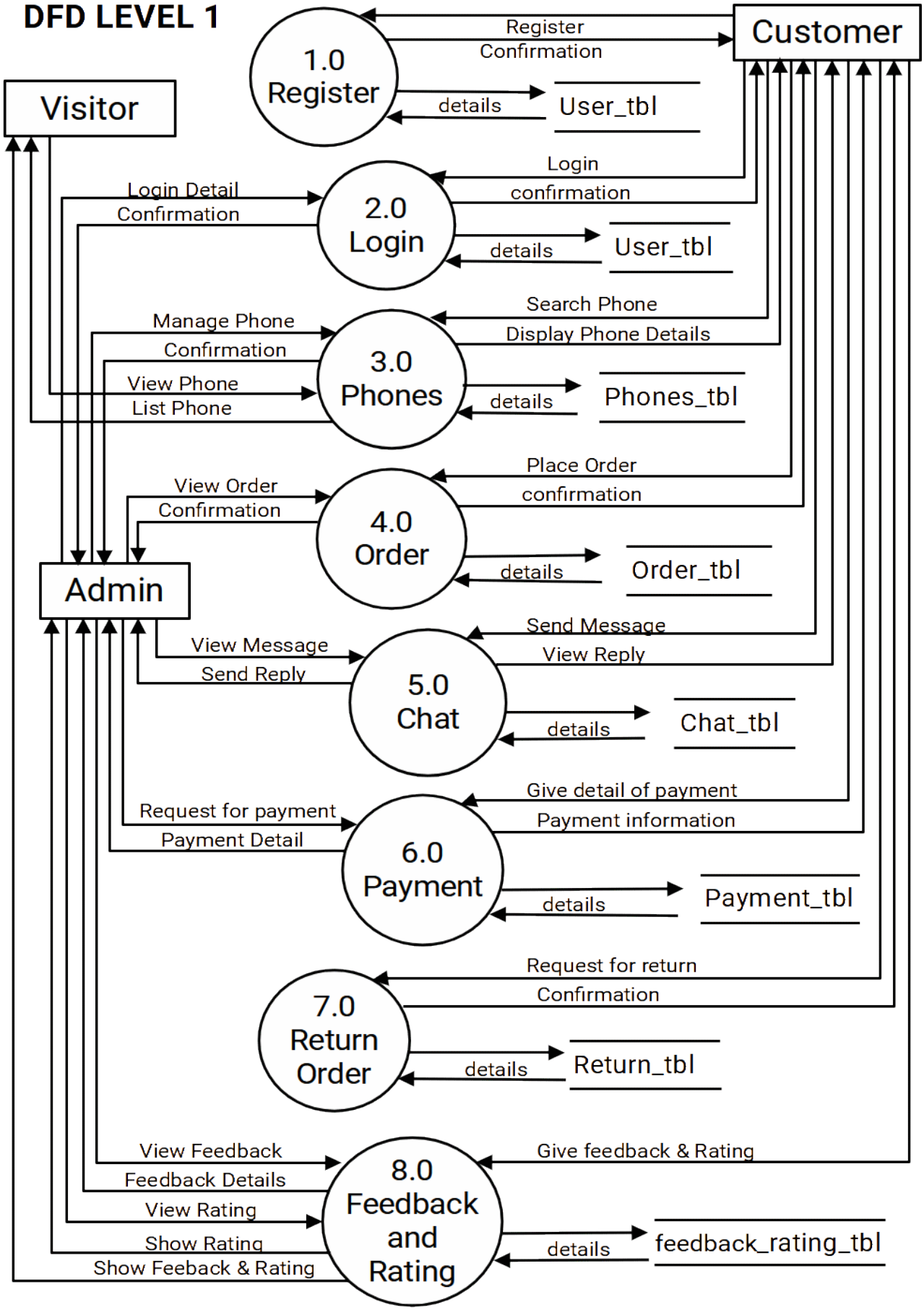


= Data Store, responsible for storing useful information

DFD LEVEL 0

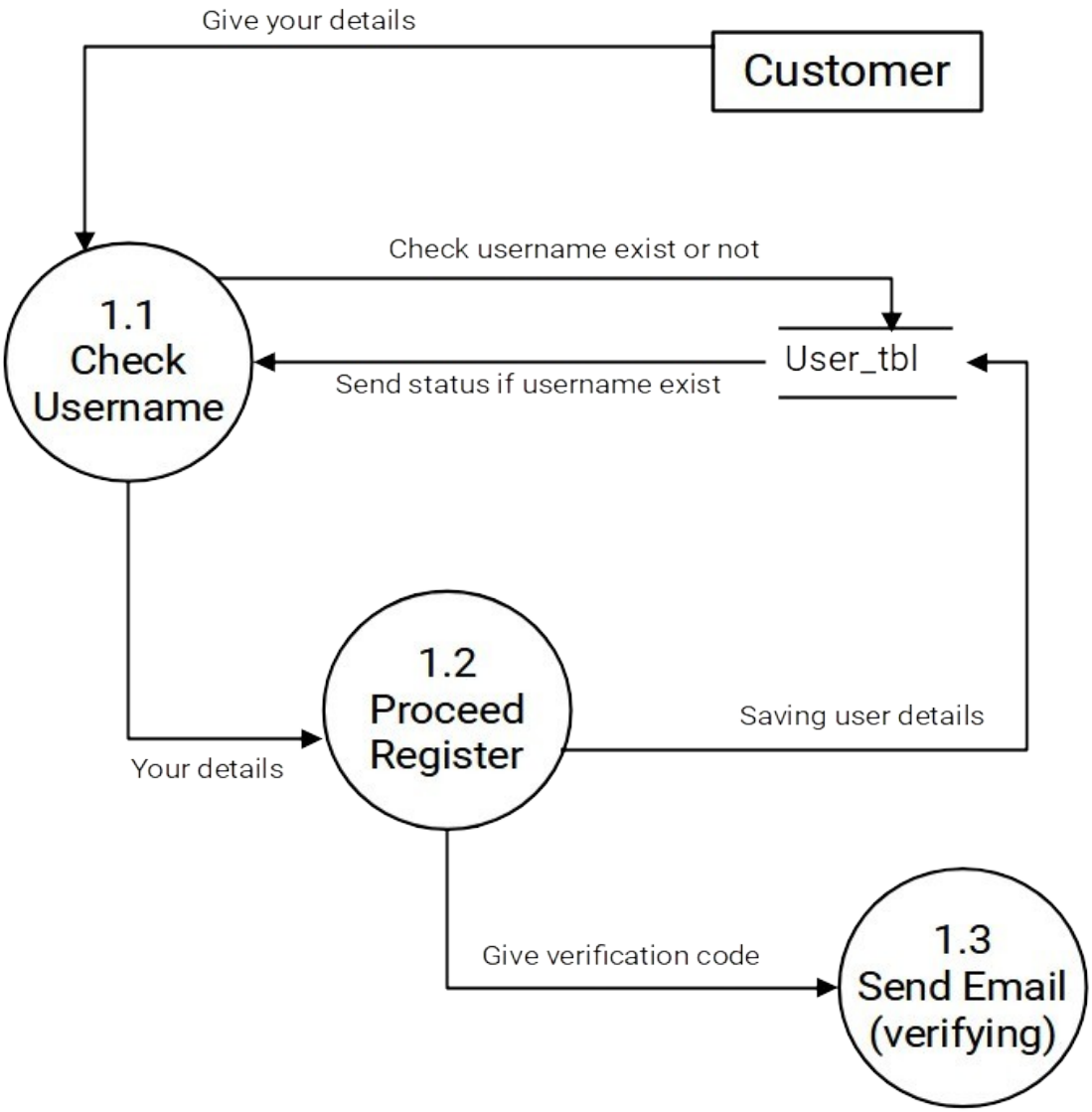
DFD Level 0 is also called a Context Diagram.
It's a basic overview of the whole system or process being analyzed or modeled.




DFD LEVEL 1
Customer
Visitor
Admin
1.0 Register
2.0 Login
3.0 Phones
4.0 Order
5.0 Chat
6.0 Payment
7.0 Return Order
8.0 Feedback and Rating
Register
Confirmation
details
User_tbl
Login
confirmation
Login Detail
Confirmation
details
User_tbl
Search Phone
Display Phone Details
Manage Phone
Confirmation
View Phone
List Phone
details
Phones_tbl
Place Order
confirmation
View Order
Confirmation
details
Order_tbl
Send Message
View Reply
View Message
Send Reply
details
Chat_tbl
Give detail of payment
Payment information
Request for payment
Payment Detail
details
Payment_tbl
Request for return
Confirmation
details
Return_tbl
View Feedback
Feedback Details
View Rating
Show Rating
Show Feeback & Rating
Give feedback & Rating
details
feedback_rating_tbl

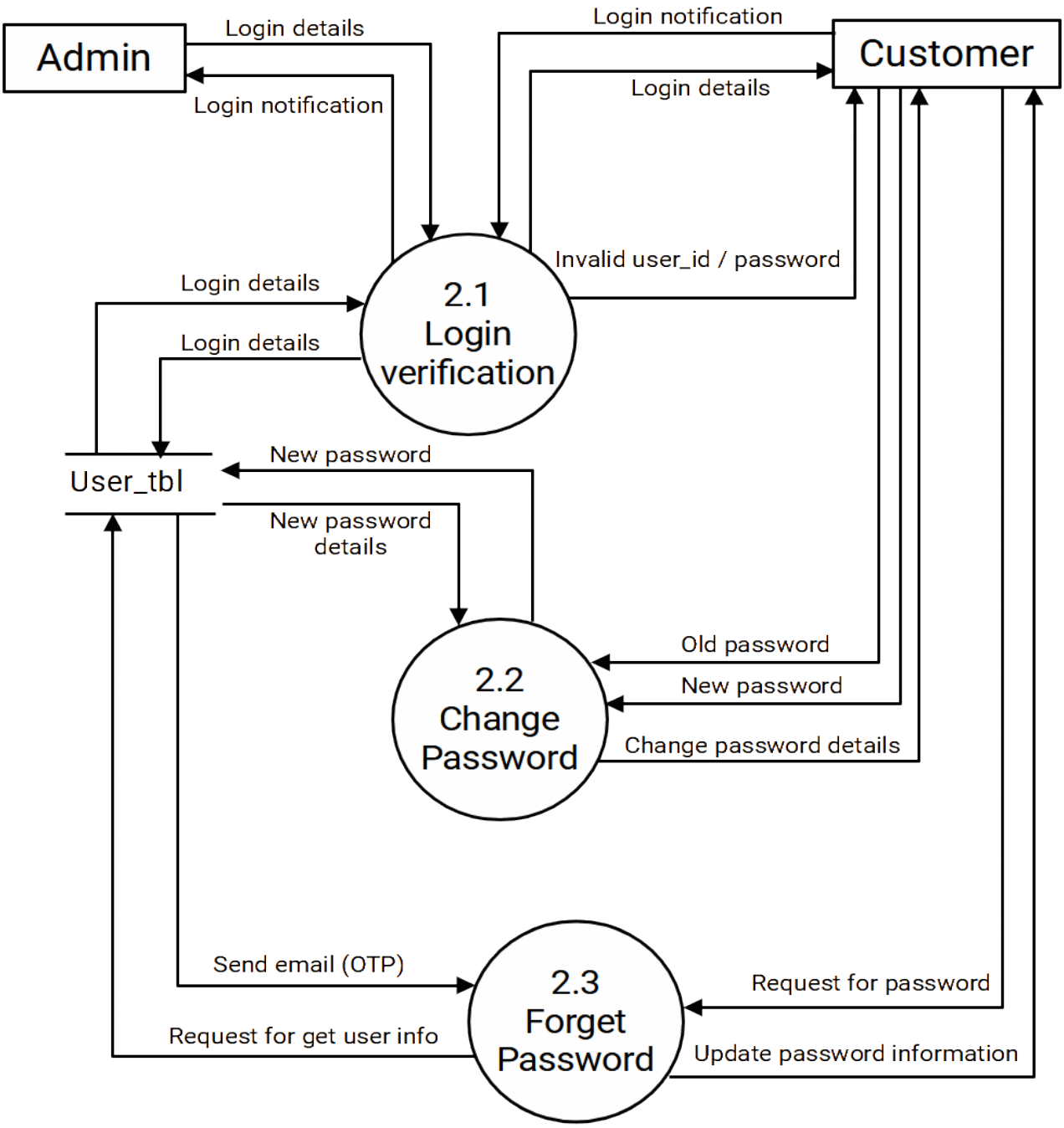
DFD LEVEL 2

REGISTER

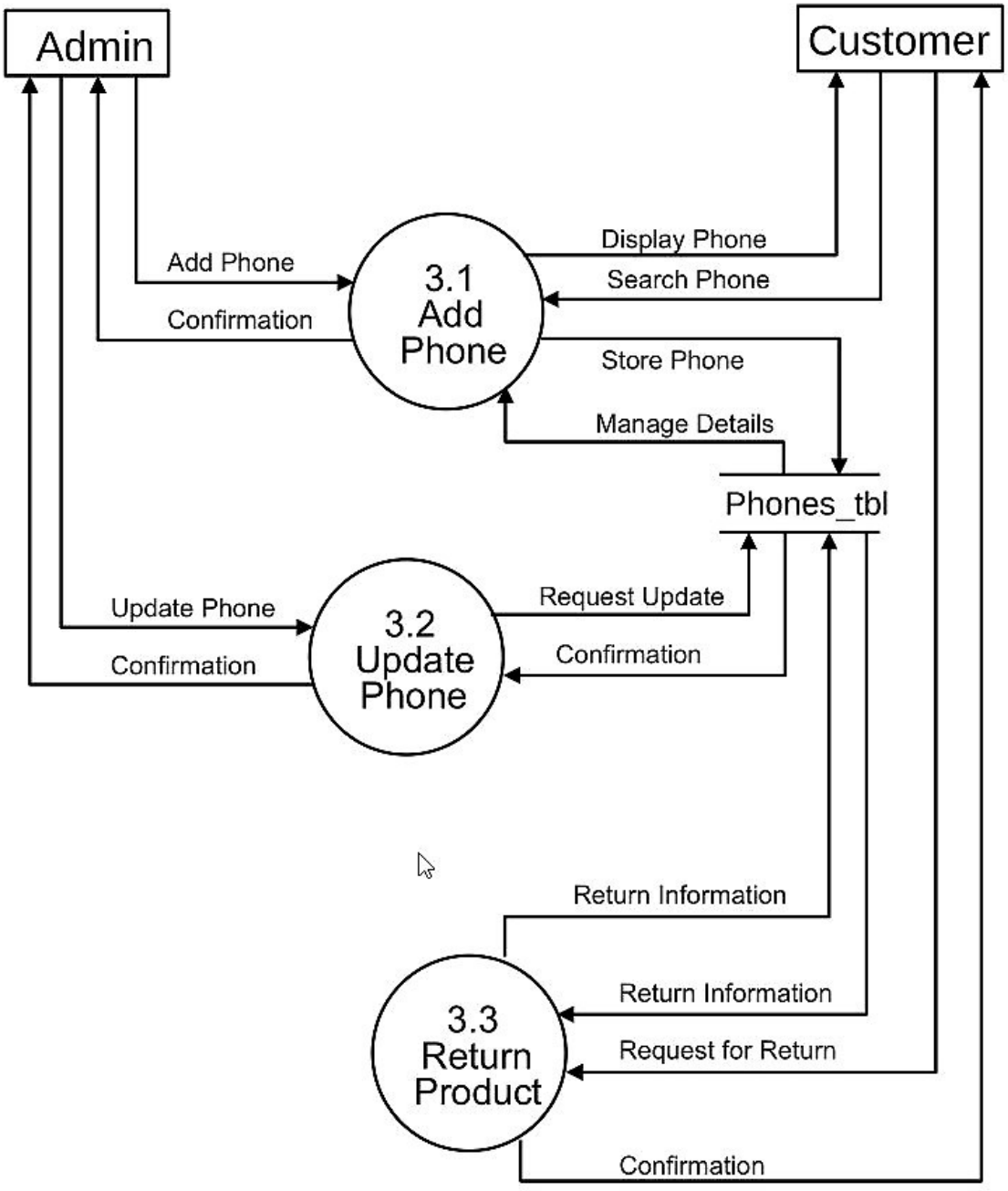


DFD LEVEL 2

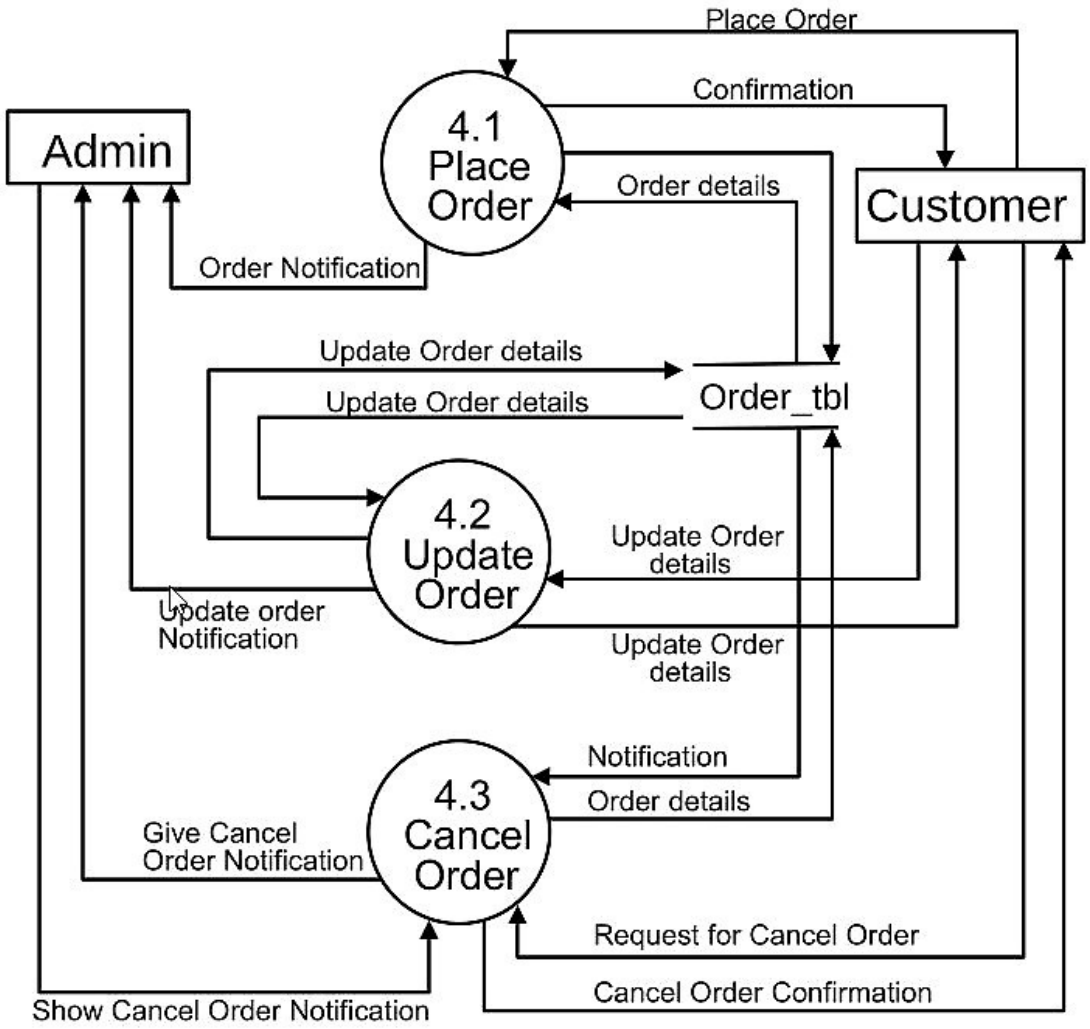
LOGIN



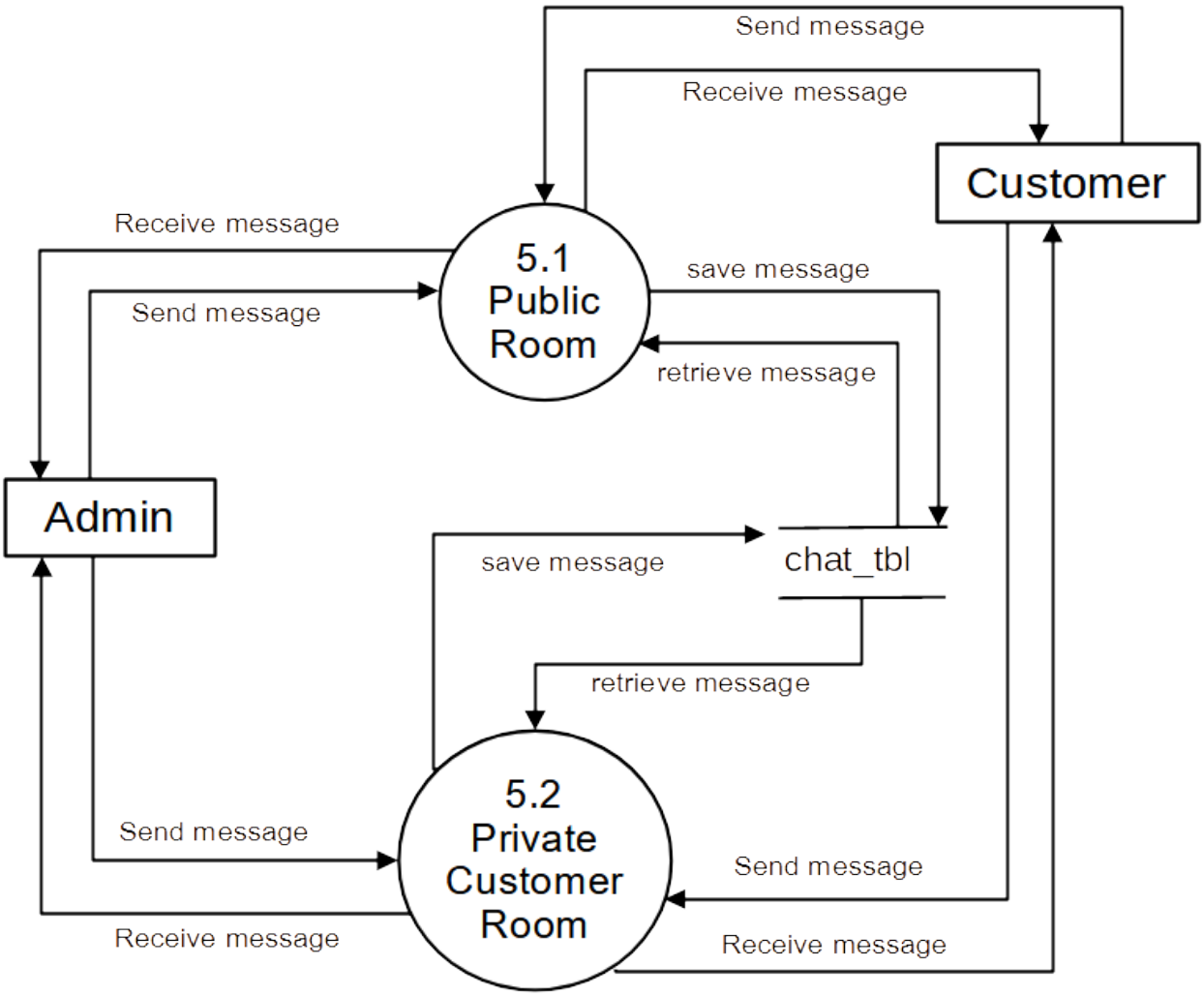
DFD LEVEL 2 PHONES



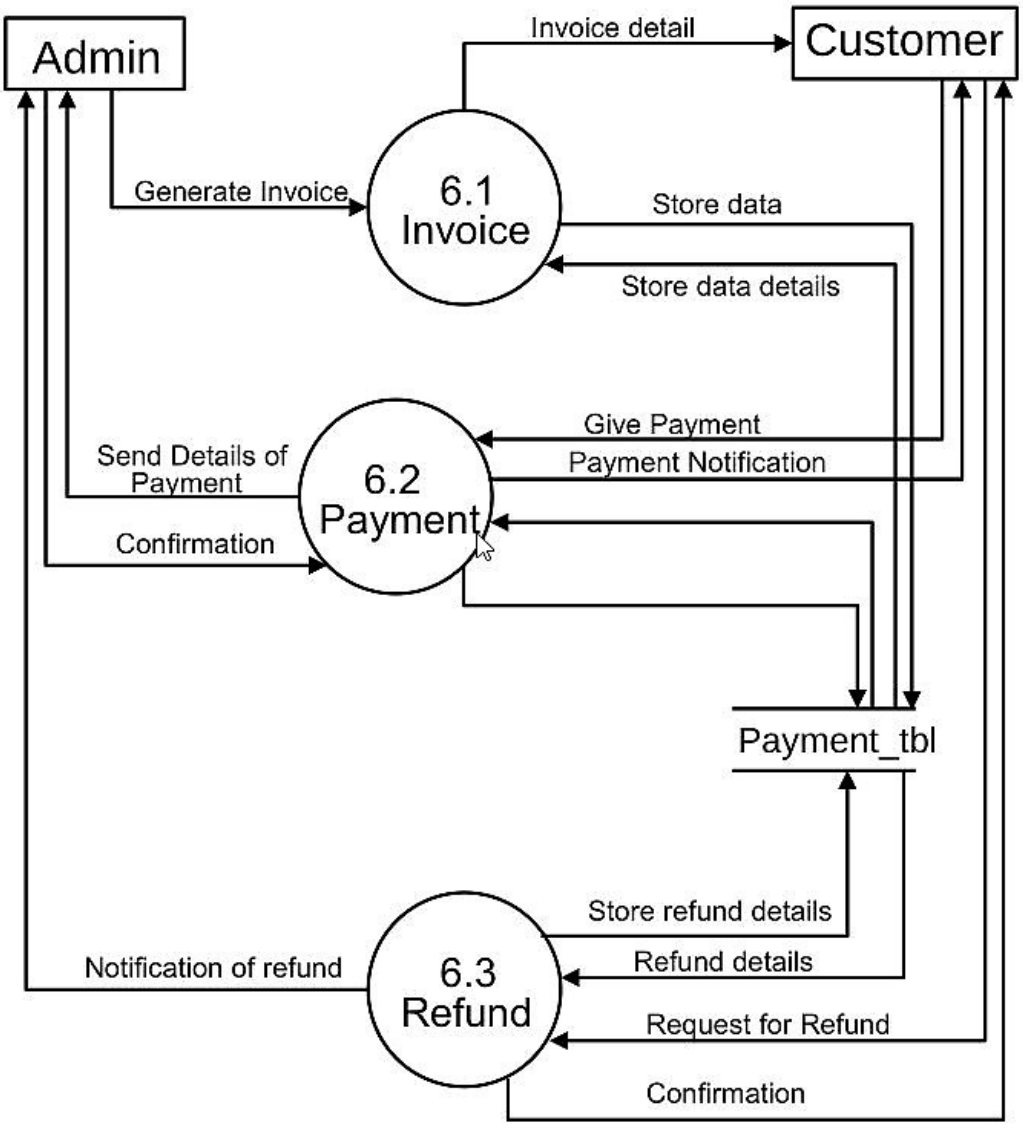
DFD LEVEL 2 ORDER



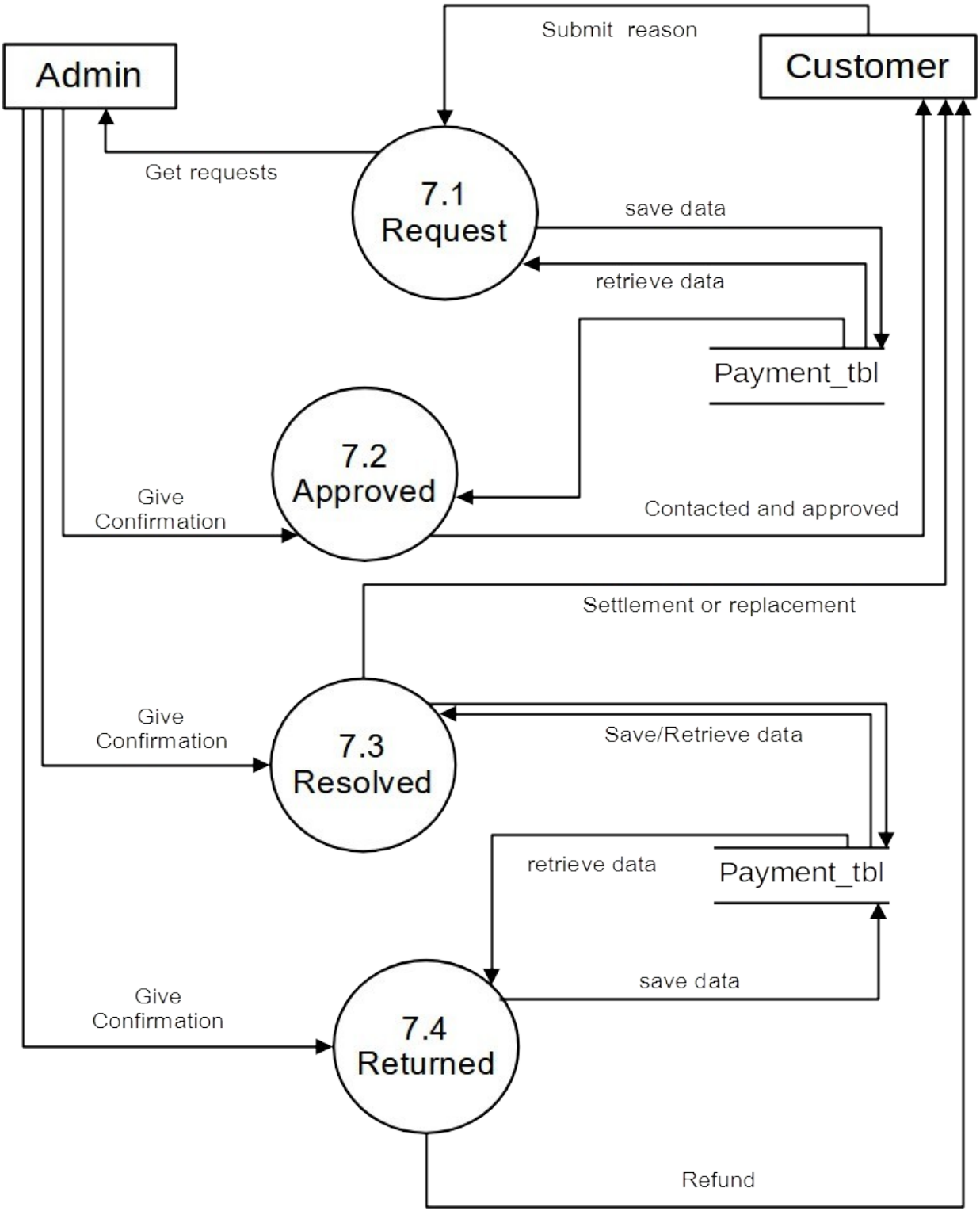
DFD LEVEL 2 **CHAT**



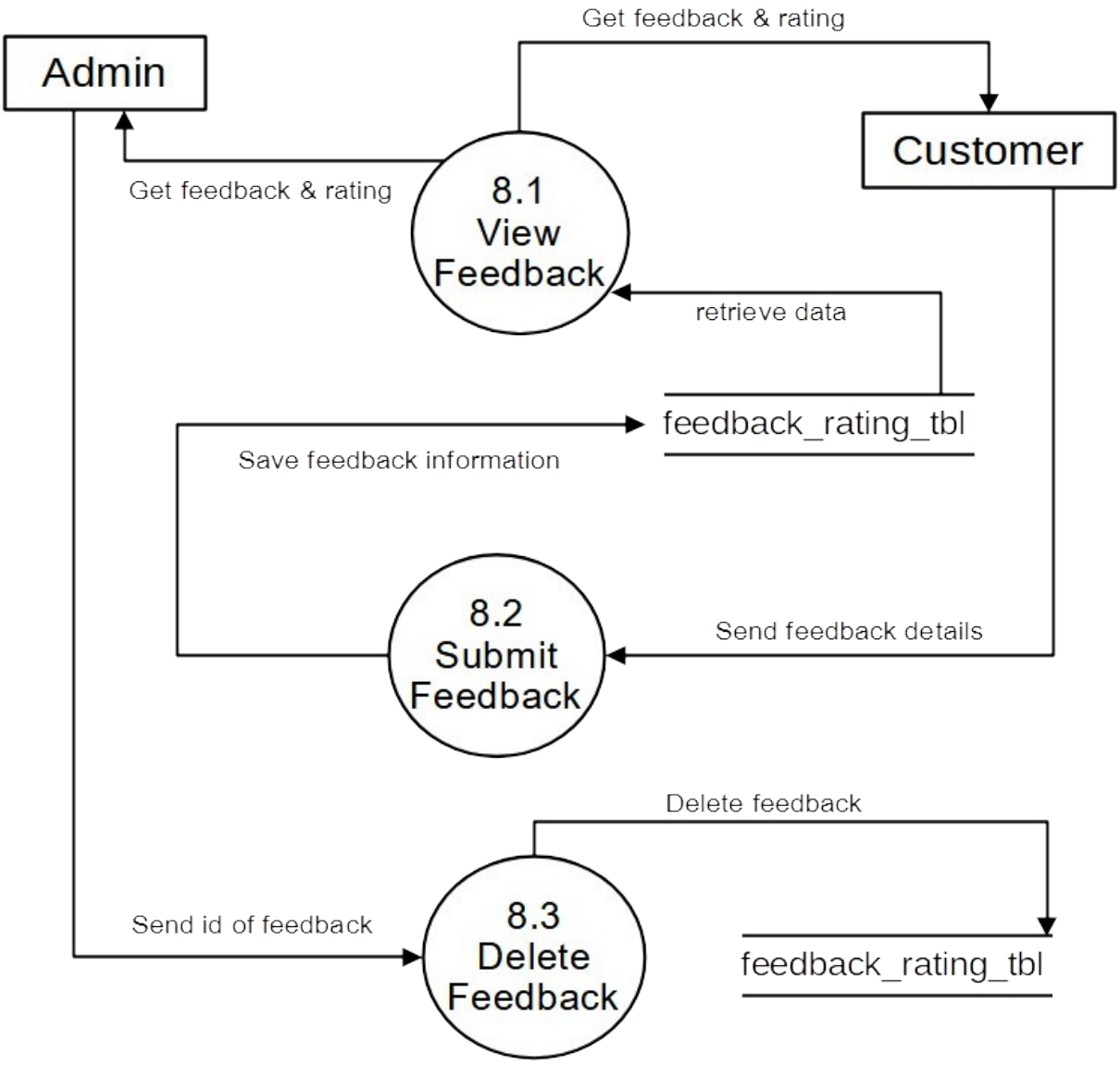
DFD LEVEL 2 PAYMENT



DFD LEVEL 2 **RETURN ORDER**

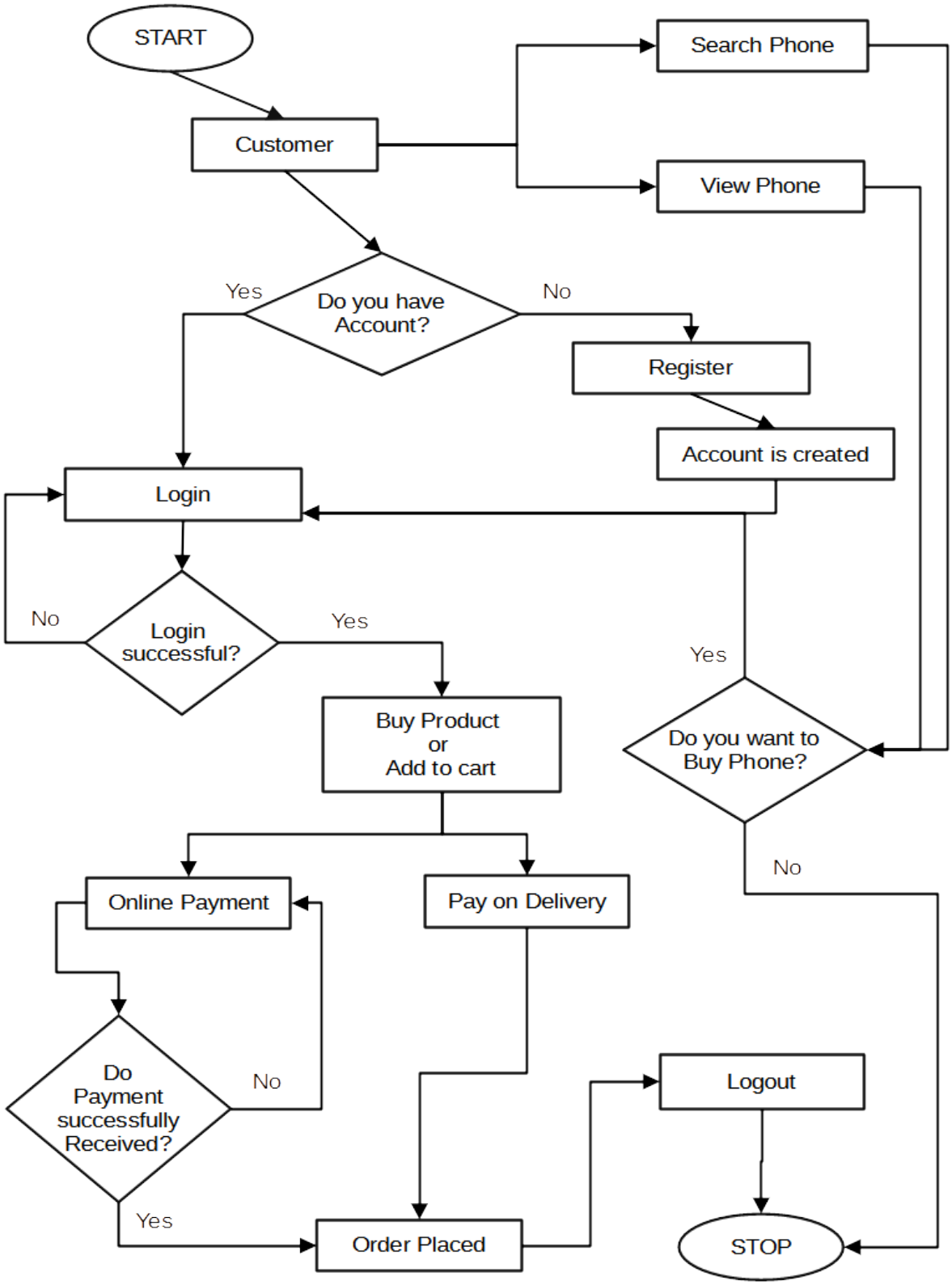


DFD LEVEL 2 FEEDBACK & RATING

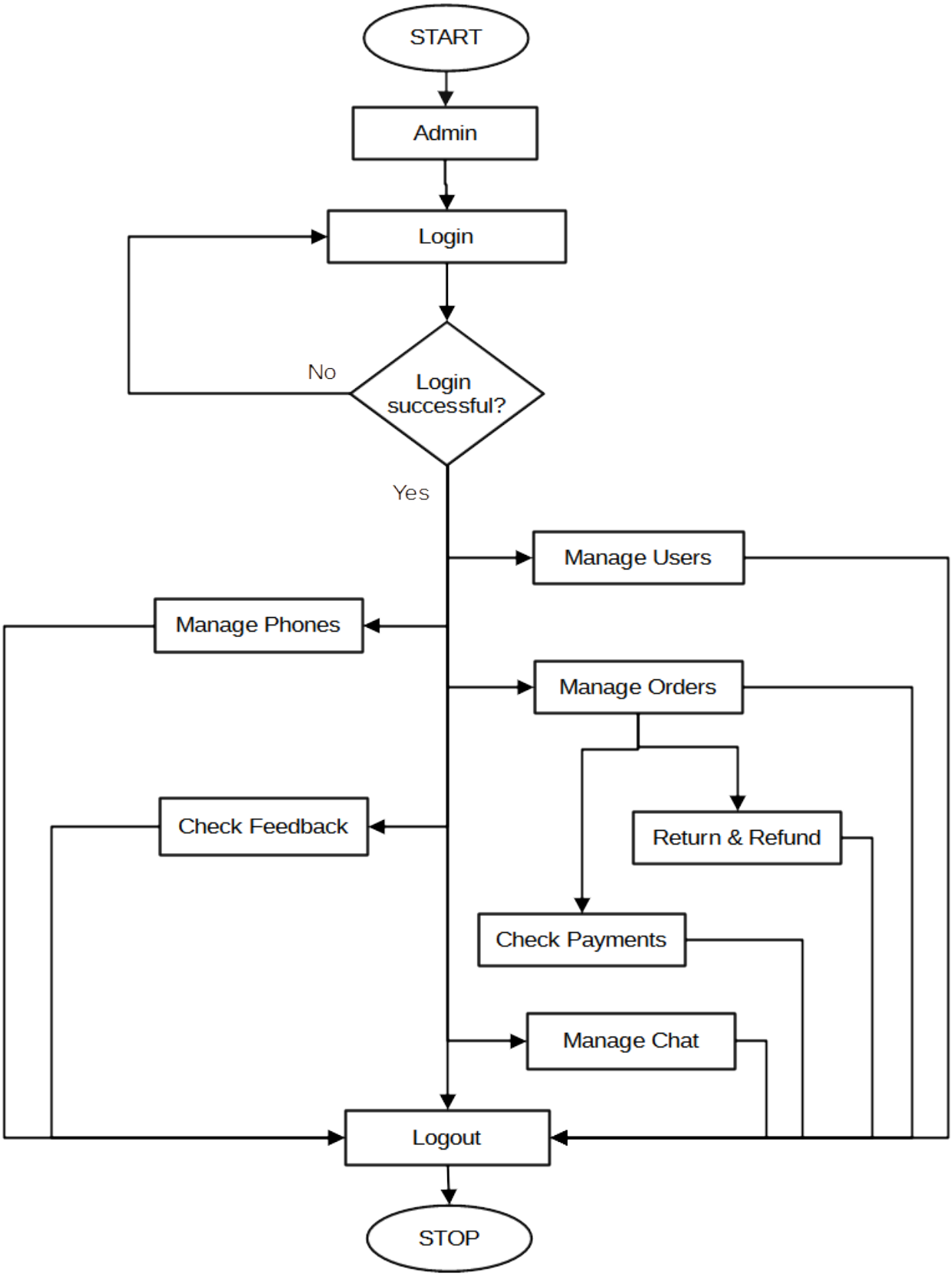


2.7.2. Control Flow Diagrams

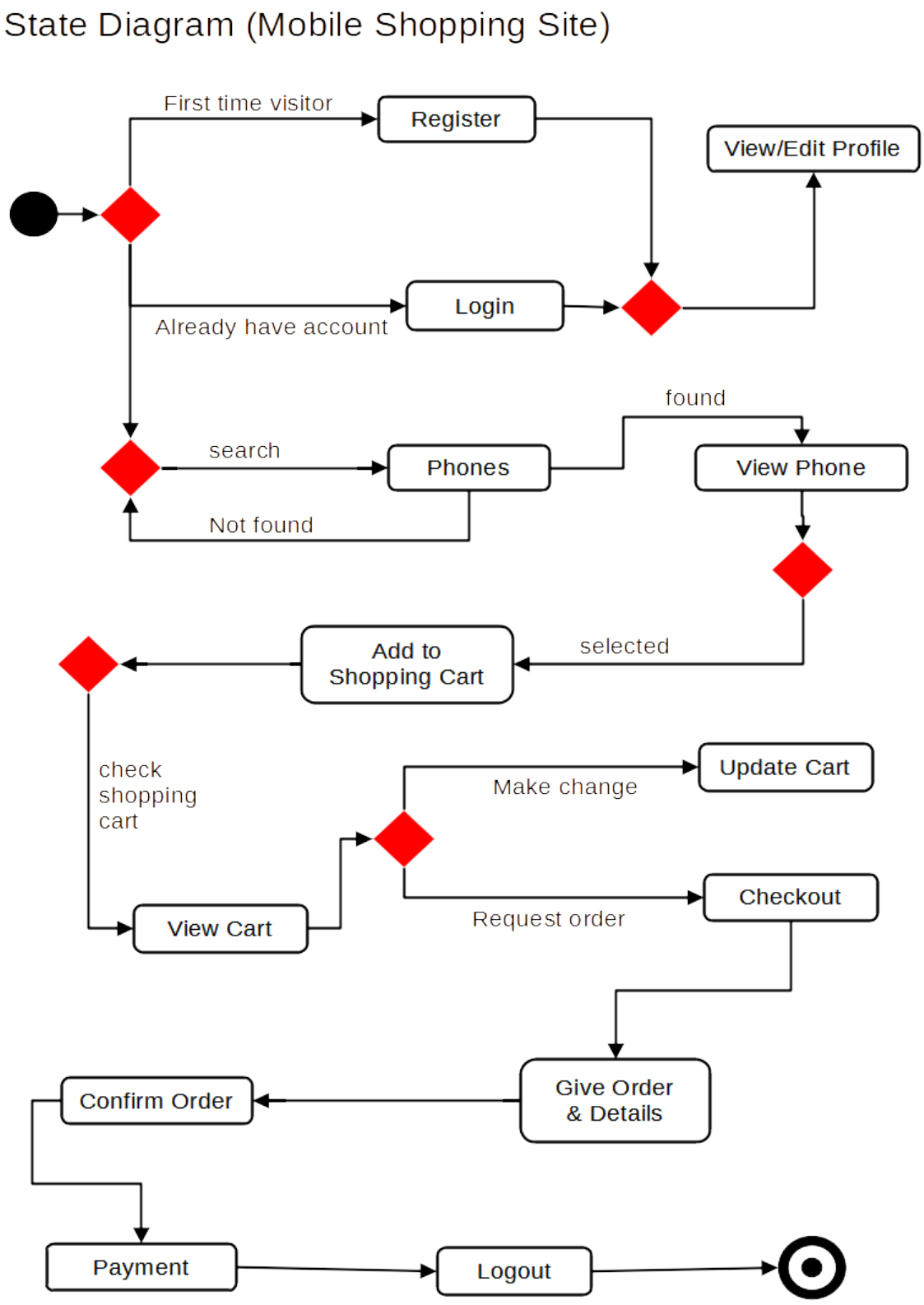
Control Flow Diagram (For Customer's)



Control Flow Diagram (For Admin's)



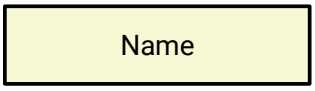
2.7.3. State Diagrams



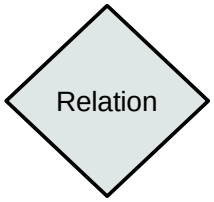
2.7.4. Entity Relationship Model,

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. In other words, ER diagrams illustrate the logical structure of databases.

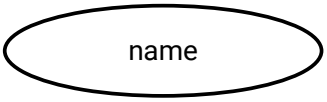
Important Symbols:



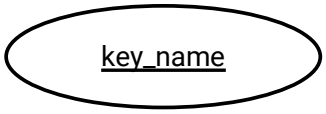
= Entity or Table Name



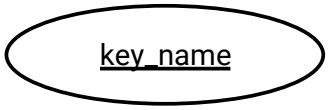
= Relationship between two Tables/Entites



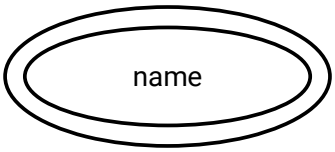
= Field/column of Table



= Primary Key of Table/Entity



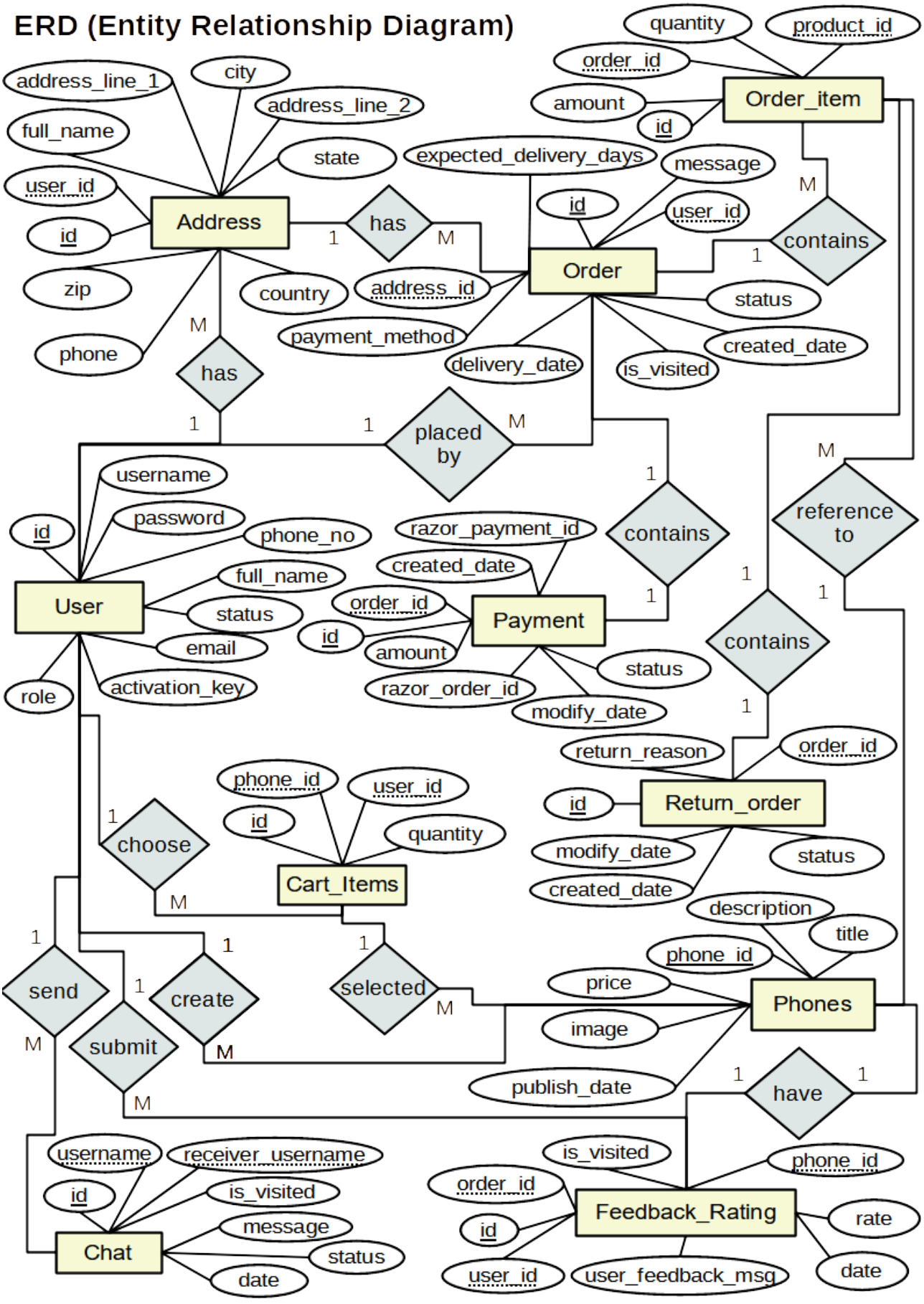
= Foreign Key



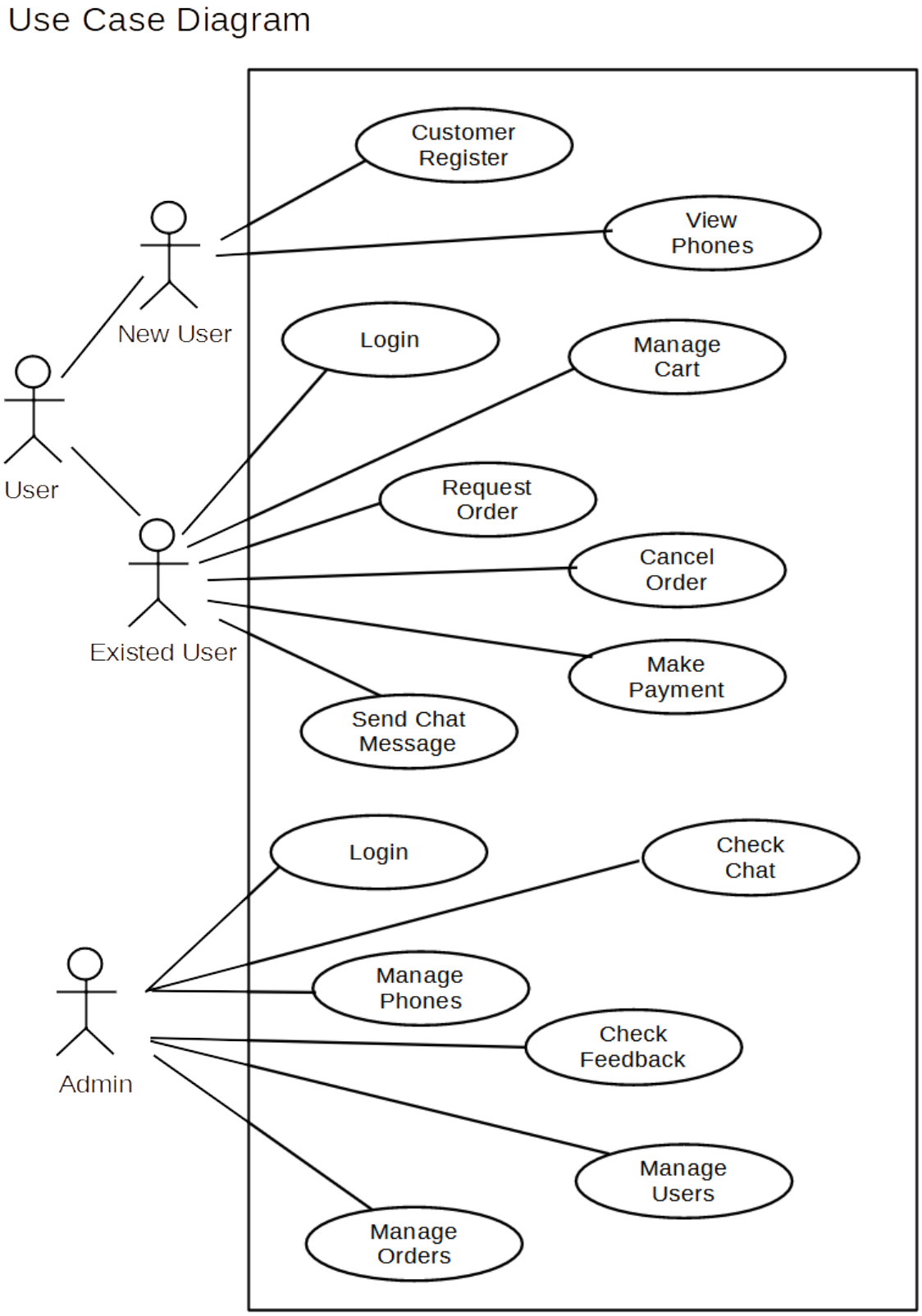
= Multi-value Key

= Derived Key

= Link


ERD (Entity Relationship Diagram)
quantity
product_id
order_id
Order_item
amount
id
address_line_1
city
address_line_2
full_name
state
expected_delivery_days
message
user_id
id
Address
has
Order
contains
user_id
id
zip
country
address_id
status
payment_method
created_date
phone
has
delivery_date
is_visited
placed by
username
password
phone_no
razor_payment_id
created_date
id
User
full_name
status
email
order_id
id
amount
Payment
contains
reference to
role
activation_key
razor_order_id
status
modify_date
return_reason
order_id
id
Return_order
modify_date
status
created_date
phone_id
user_id
id
quantity
Cart_Items
choose
description
title
phone_id
price
Phones
image
send
create
selected
submit
publish_date
have
username
receiver_username
id
is_visited
message
Chat
status
date
is_visited
phone_id
order_id
id
Feedback_Rating
rate
user_id
user_feedback_msg
date

2.7.5. User-case Diagrams



3. System Design

3.1. Modularisation Details

The modules used in this application are as follows:

3.1.1. Guest/Public Modules:

- **Home Page**: This page simply used to demonstrate our phones, and useful information about company and various helpful links all together with an attractive and beautiful design.
- **About Page**: This page show more useful information about company in a representative ways.
- **Contact Page:** This page where customer and send query and email to admin of site without registration.
- **Login Page**: This page provide feature to authenticated and authorised based on user role.
- **Registration Page**: This page is for the users who do not have their account. Customer is allowed to create an account for login.
- **List of Mobile Phone Page**: You can find all available phones in a presentable way, where customers can view list of phones and price, they can also add them to cart, it will notified for login or create your account and login.
- **Mobile Phone Details Page**: This page show you full details of Mobile phone with its name and picture, price information etc, also customer feedback and rating.

3.1.2. Customer Modules:

Customer have limited access to website, Customer can access Guest Modules as well as the list of modules below:

- **Profile Page**: This page show customer details and helpful links to other page where he can manage his account. Details like add card, add address, etc. After login as Customer he will be redirected to Customer Management System. He can also update his password here.
- **Address Page**: This page show list of address linked to his account. Customer can add, edit or delete from this list.
    1. Add New Address Page and Confirmation
    2. Edit Address Page and Confirmation
    3. Delete Address and Confirmation
- **Orders Page**: List of orders that customer requested, he can view Status, and retry for online payment if not paid, while submitting order.
- **Return Order Page**: List of orders those are delivered to customer are shown here, he can raise request to return his order and submit reason (if Damage Item Received or Package missing some item).
- **Download Invoice Page**: Customer can Download invoices of any delivered orders.
- **Feedback & Rating Page**: User can give feedback and rating on this page from list of purchased phone are listed here.
- **Chat Page**: Customer can participate in Public chat. Customer can also talk privately with any of

ADMIN users in database. If he is online then green circle appear, if he is offline then red circle appears.

- **Cart Page**: User can select any number of Mobile and add to the cart. He can also remove from the cart if he dislikes it later.
- **Order Page:** This module have a form to give details like name, address, payment options and generate invoice to be sent to his email address.
- **Payment Page**: This module describes the payment done by the customer. The payment information can include information like the model purchased, quantity, mode of payment (cash, loan) etc.

3.1.3. Admin Modules:

Administrator of System can access all modules mention above as well the list below:

- **Dashboard Page**: This page contains, overview information about system, for example total numbers of users, total number of orders, total phones, total feedbacks, etc. This page also contains links to navigate to other pages.
- **Users Page**: This page show list of all users (ADMIN | CUSTOMER | SUBSCRIBER). Admin can manage with follow features:
    1. **Add New User Page:** Adding new user by filling his information.
    2. **Edit User Page:** Editing existing user information.
    3. **Delete User**: Deleting any existing user after accepting confirmation.
- **Address Page**: List of all users address will shown here, admin can also manually manage them as follow:
    1. **Add New Address Page**: Adding new address for any of the user.
    2. **Edit Address Page**: Editing existing address for any of the user.
    3. **Delete Address**: Remove any address from database after accepting confirmation.
- **Phones Page**: List of all Mobile Phone available for sale. Admin can manage them as follow:
    1. **Add New Phone Page**: Adding new phone with informations.
    2. **Edit Phone Page**: Update or Editing existing phone information.
    3. **Delete Phone**: Removing any phone from database after accepting confirmation .
- **Orders Page**: All orders requested by customer will be shown in this page. Admin can view and manage as follow:
    1. **Order Details Page**: This page give more details and list of phones purchased in single order, admin can delete or update status of order in this page.
    2. **Delete Order**: Removing any order after accepting confirmation.
- **Payment Page**: Admin can view list of all Online payments placed by customers.
- **Return Order Page**: Admin will see any return request by customer in this page, he can contact and update status of individual request based on progress made during processs.
- **Feedback Page**: Admin can view list of feedback's on this page and removed any if he wanted.
- **Chat Page**: Admin can talk in public chat or private chat with any of customers.

3.2. Database Design

1. Table Name: user

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
username	VARCHAR	60	NOT NULL
password	VARCHAR	120	NULL
full_name	VARCHAR	60	NULL
email	VARCHAR	60	NULL
phone_no	VARCHAR	16	NULL
role	VARCHAR	20	NULL
activation_key	VARCHAR	255	NULL
status	Int	8	NULL

2. Table Name: address

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
user_id	INT	11	Foreign Key
full_name	VARCHAR	50	NULL
phone	VARCHAR	16	NULL
address_line_1	VARCHAR	255	NULL
address_line_2	VARCHAR	255	NULL
city	VARCHAR	50	NULL
state	VARCHAR	50	NULL
zip	VARCHAR	50	NULL
country	VARCHAR	50	NULL

3. Table Name: phone

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
title	VARCHAR	255	NO DEFAULT
description	VARCHAR	1024	NO DEFAULT
price	FLOAT		NO DEFAULT
image	VARCHAR	255	NULL
publish_date	DATETIME	6	CURRENT_TIMESTAMP

4. Table Name: orders

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
address_id	INT	11	Foreign Key
status	varchar	255	NULL
expected_delivery_days	INT	11	NULL
message	VARCHAR	255	NULL
payment_method	VARCHAR		NULL
user_id	INT	11	NULL
delivery_date	DATETIME	6	NULL
is_visited	BIT	1	NULL
created_date	DATETIME		CURRENT_TIMESTAMP

5. Table Name: order_item

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
order_id	INT	11	Foreign Key
phone_id	INT	11	Foreign Key
quantity	INT	11	NULL
amount	FLOAT		NO DEFAULT

6. Table Name: chat

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
username	VARCHAR	255	NULL
receiver_username	VARCHAR	255	NULL
message	VARCHAR	255	NULL
status	VARCHAR	50	NULL
date	DATETIME	6	CURRENT_TIMESTAMP
is_visited	BIT	1	FALSE

7. Table Name: payment

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
order_id	INT	11	Foreign Key
amount	FLOAT		NULL
status	VARCHAR	20	NULL
razor_order_id	VARCHAR	255	UNIQUE, NULL
razor_payment_id	VARCHAR	255	UNIQUE, NULL
modify_date	DATETIME	6	UPDATE_TIMESTAMP
created_date	DATETIME	6	CREATION_TIMESTAMP

8. Table Name: return_order

Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
order_id	INT	11	Foreign Key
return_reason	VARCHAR	255	NULL
status	VARCHAR	255	NULL
modify_date	DATETIME	6	UPDATE_TIMESTAMP
created_date	datetime	6	CREATION_TIMESTAMP

9. Table Name: feedback_rating

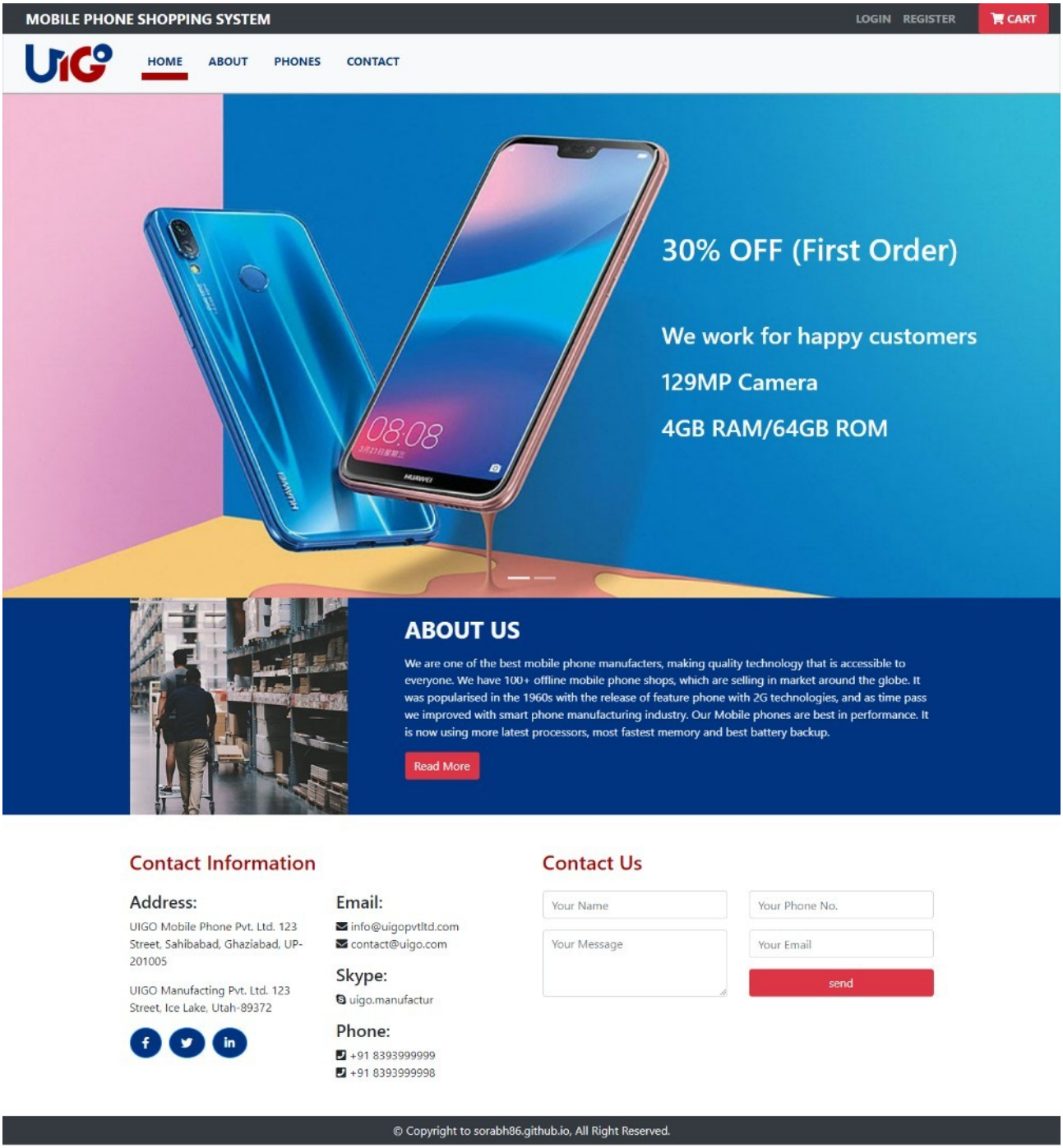
Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
phone_id	INT	11	Foreign Key
user_id	INT	11	Foreign Key
order_id	INT	11	Foreign Key
rate	INT	11	NULL
user_feedback_msg	VARCHAR	255	NULL
is_visited	BIT	1	NULL
date	DATETIME	6	CREATION_TIMESTAMP

10. Table Name: cart_items

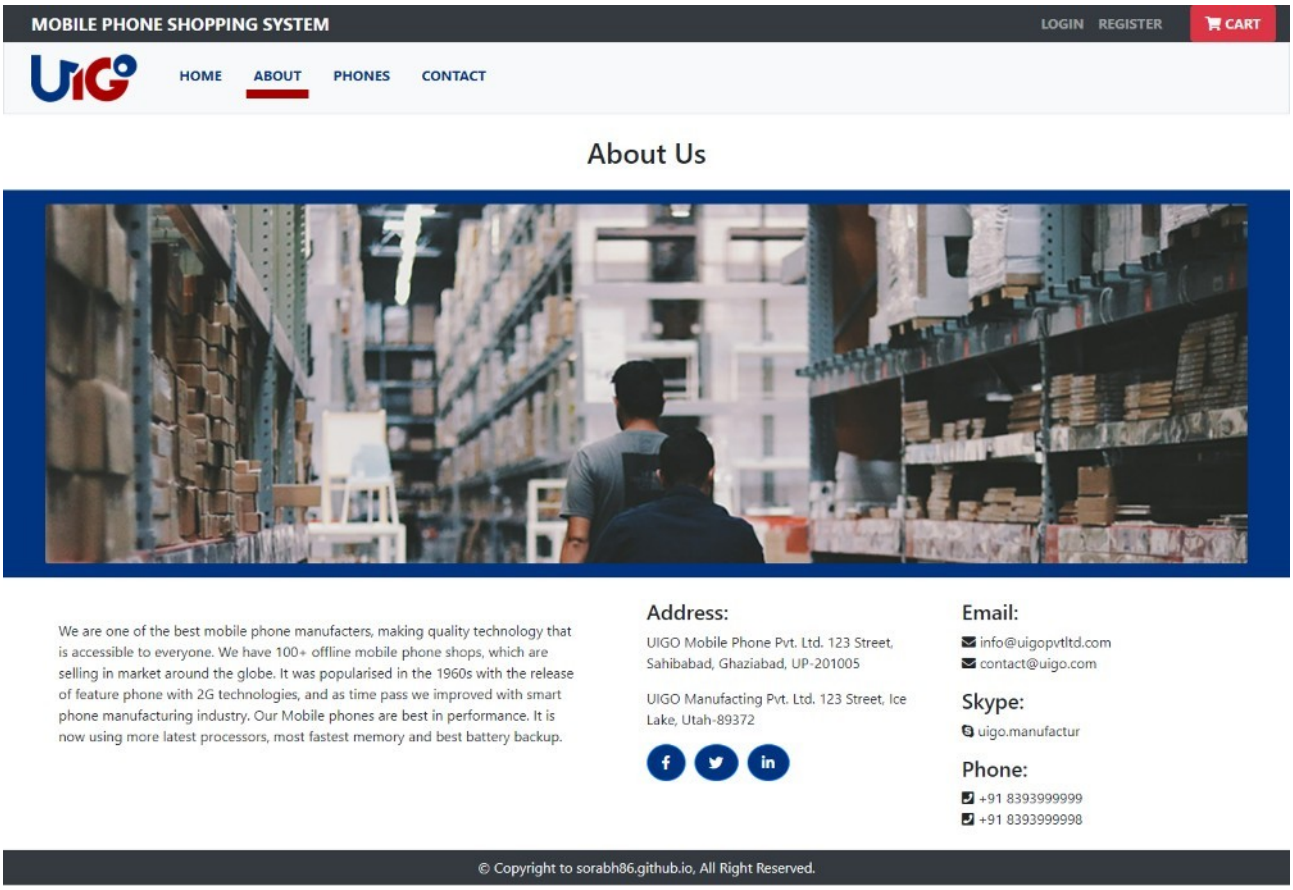
Field Name	Data Type	Length	Constraint & Description
id	INT	11	Primary Key, AUTO_INCREMENT
phone_id	INT	11	Foreign Key
user_id	INT	11	Foreign Key
quantity	INT	11	NULL

3.4. User Interface Design

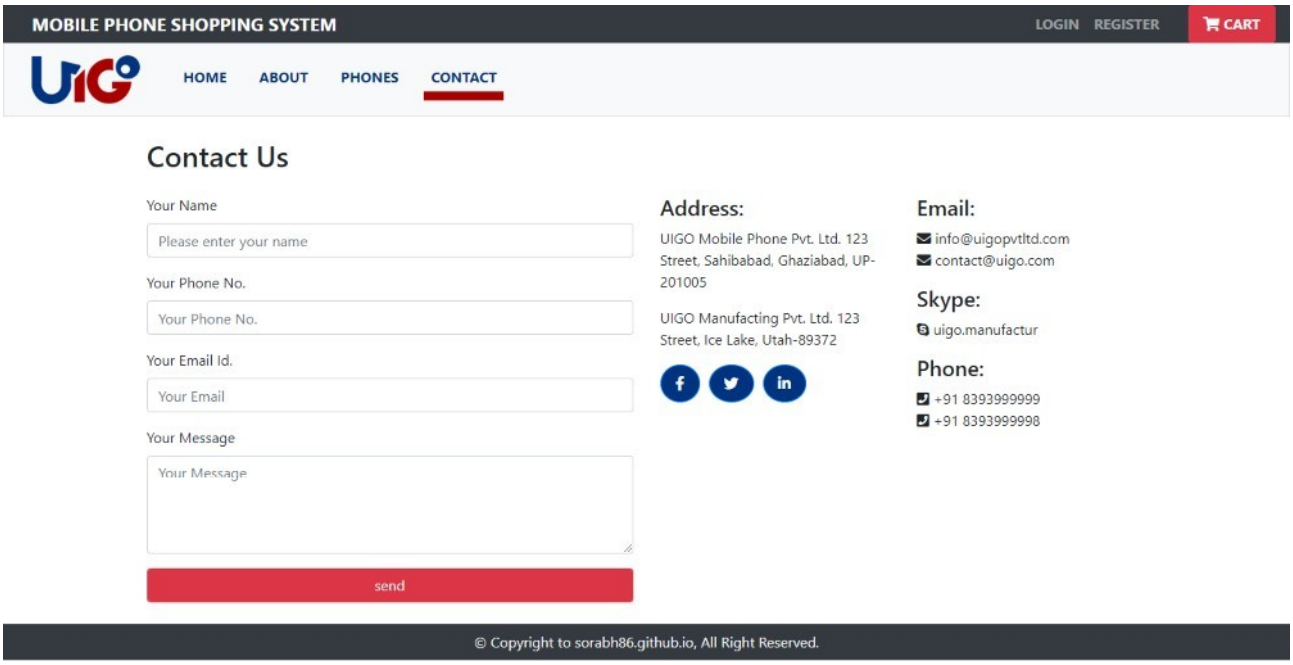
- **Home Page:**



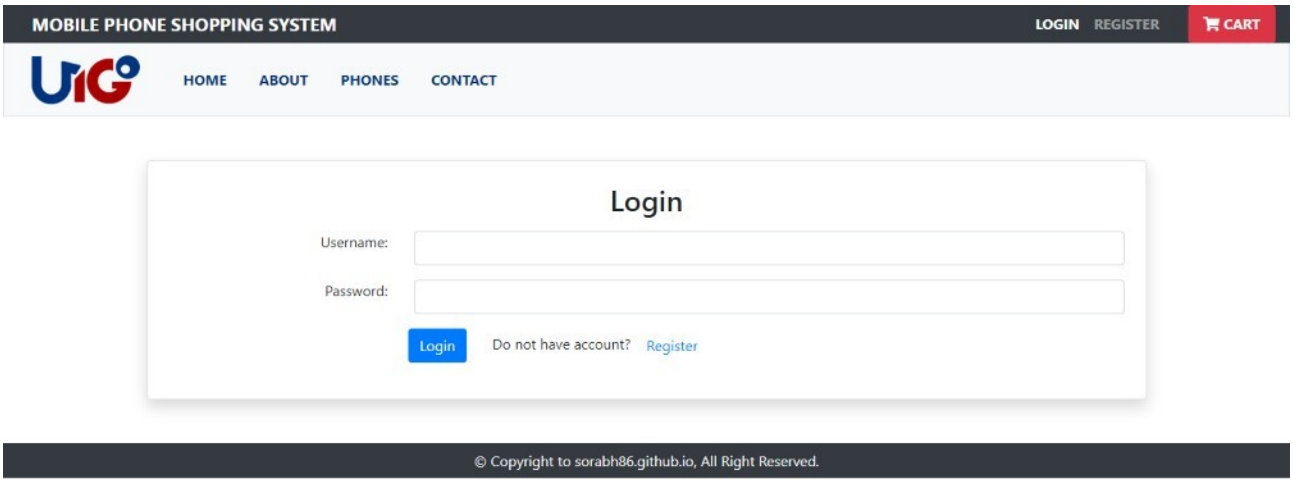
- **About Page:**



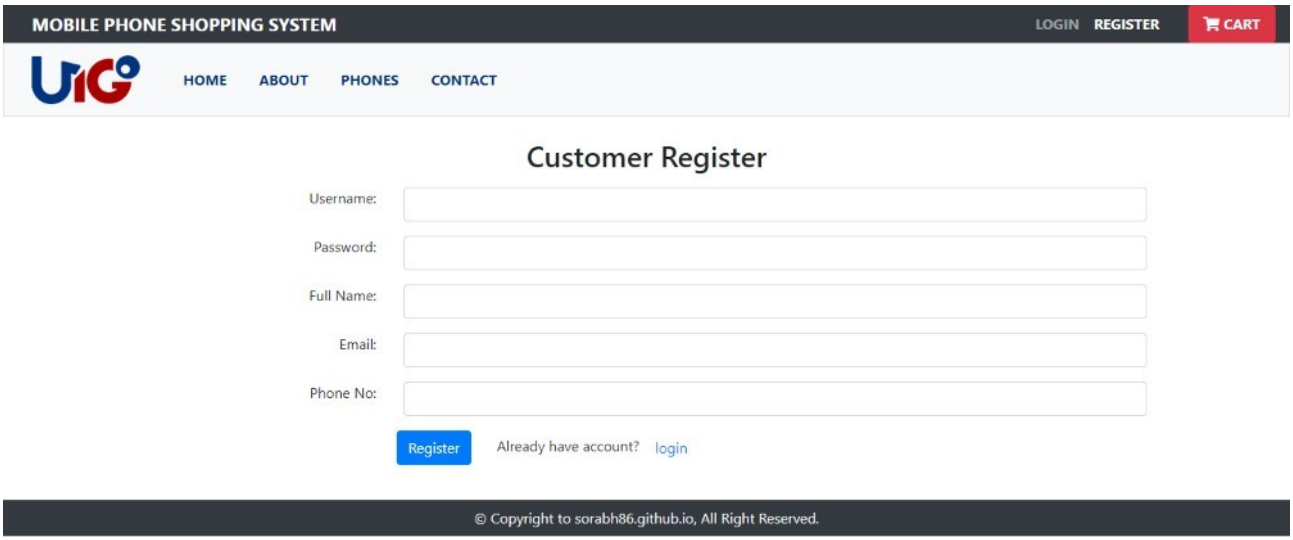
- **Contact Page:**



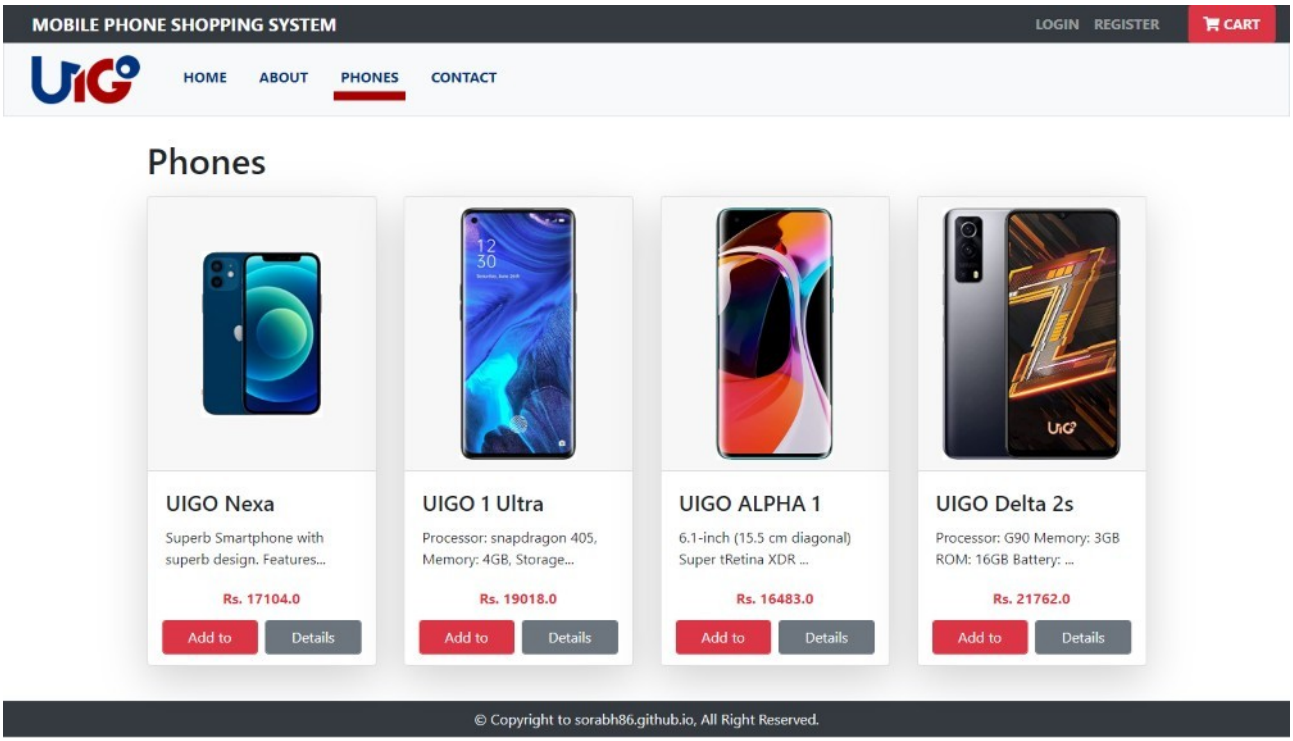
- **Login Page:**



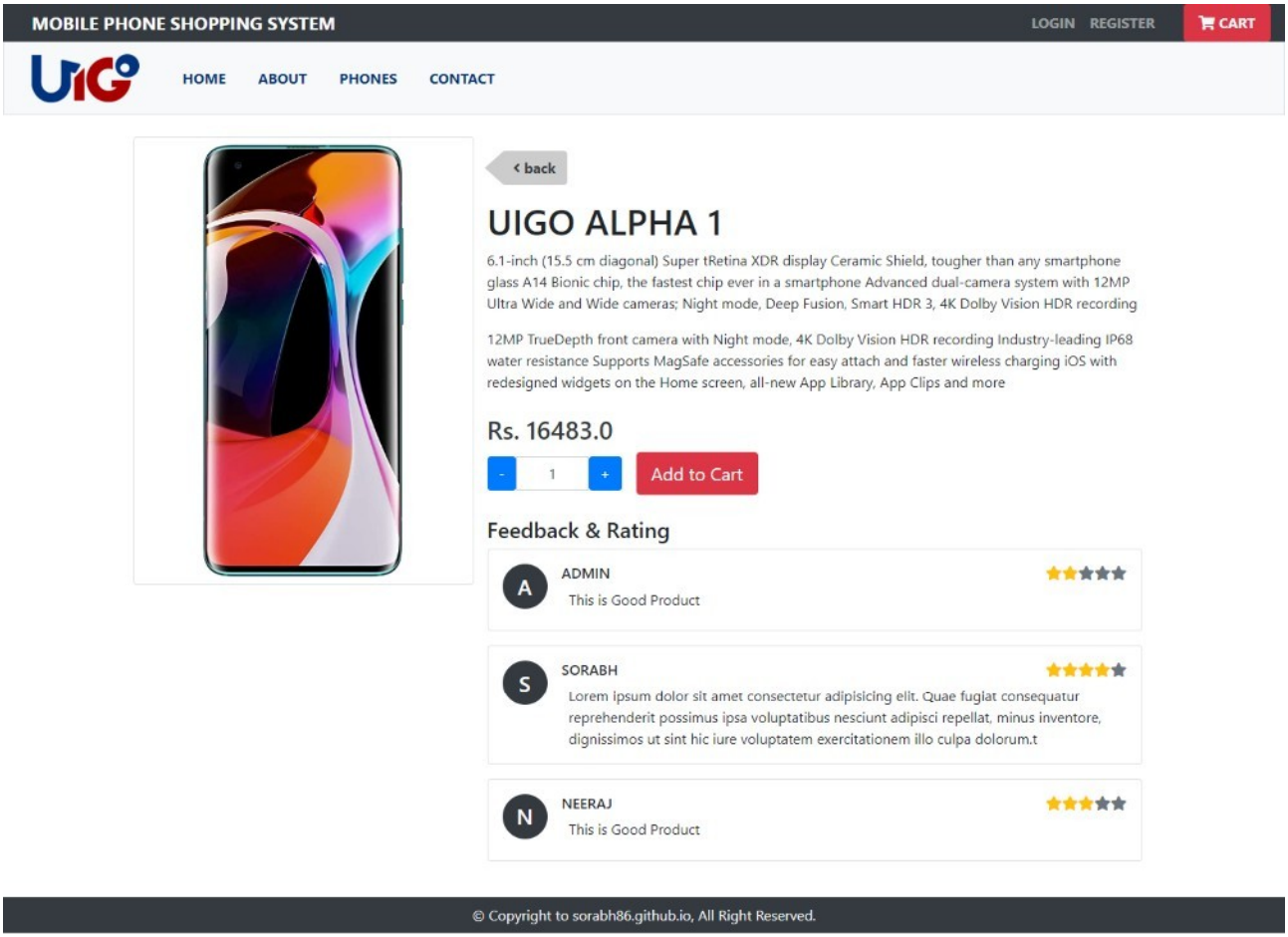
- **Registration Page:**



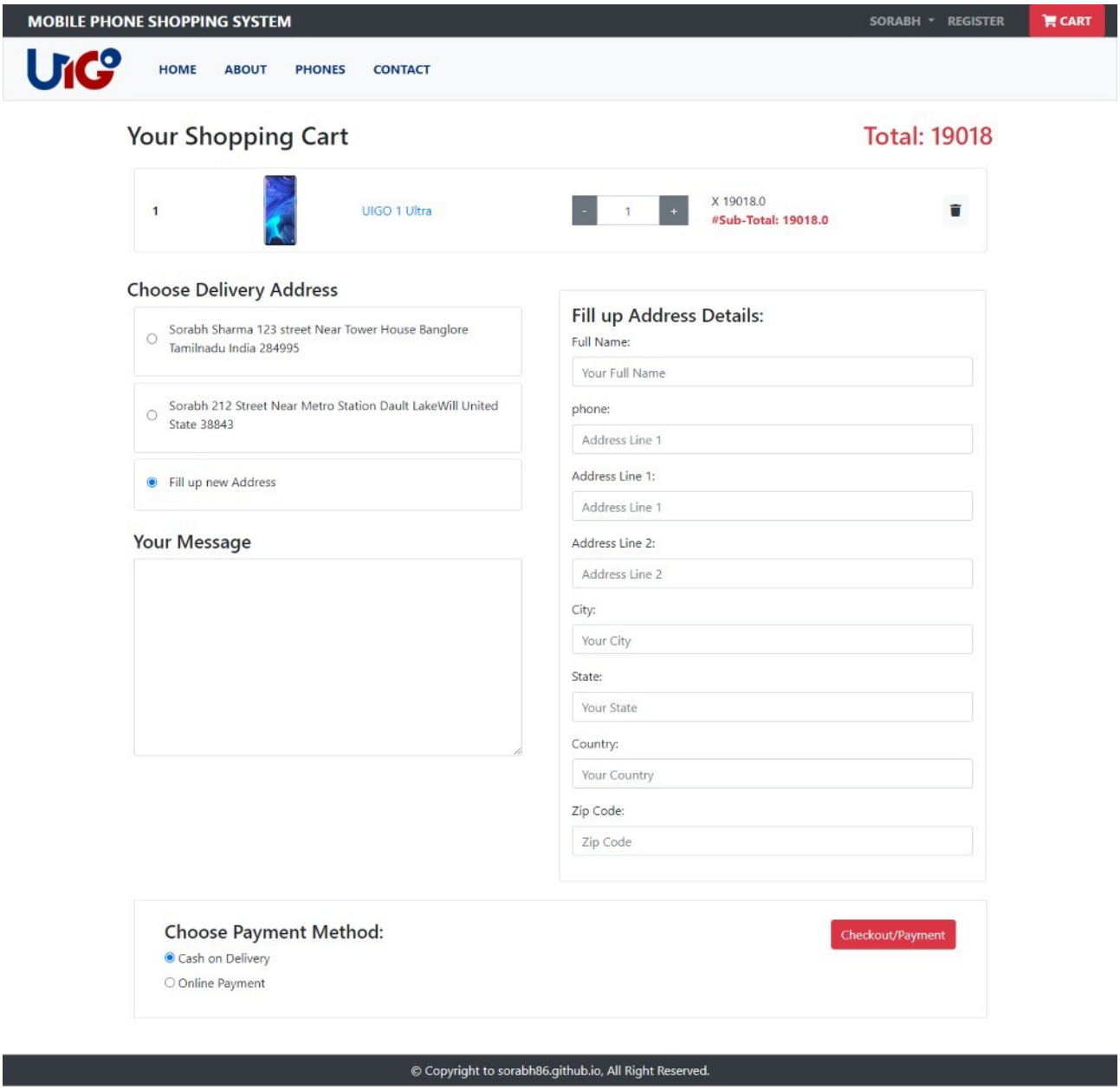
- **Phones Page:**



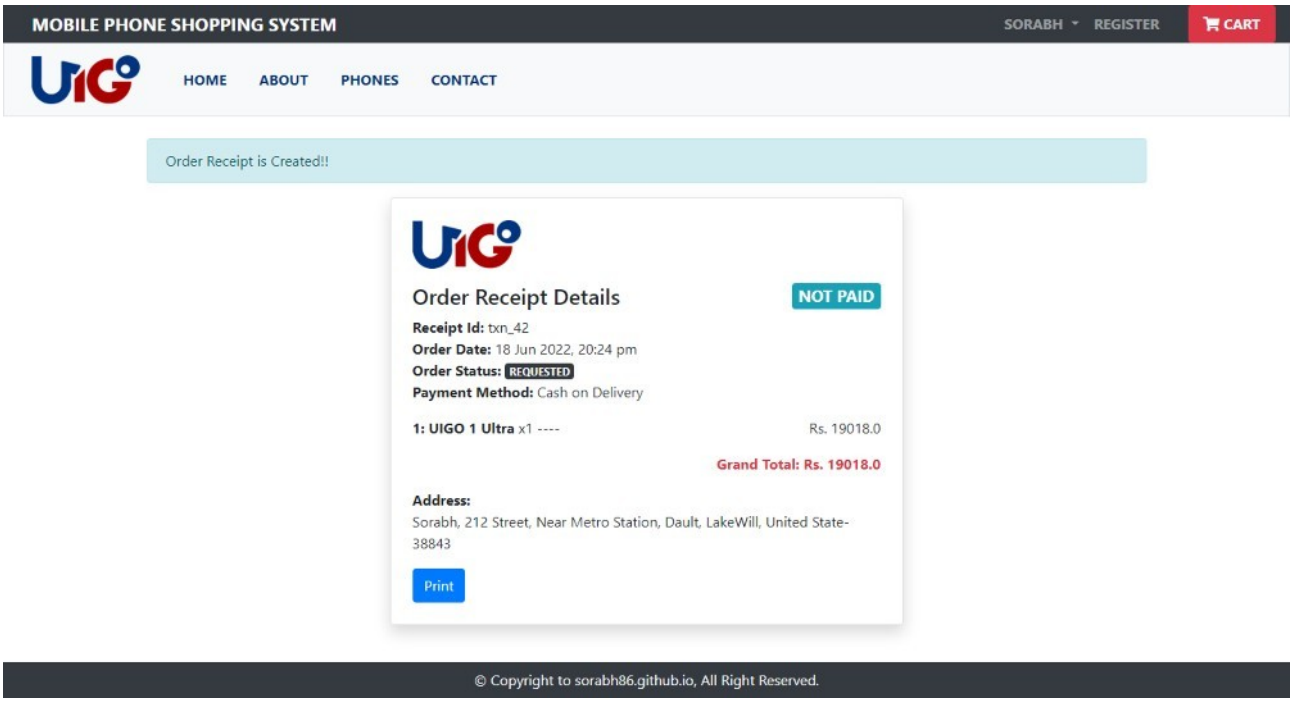
- **Phone detail Page:**



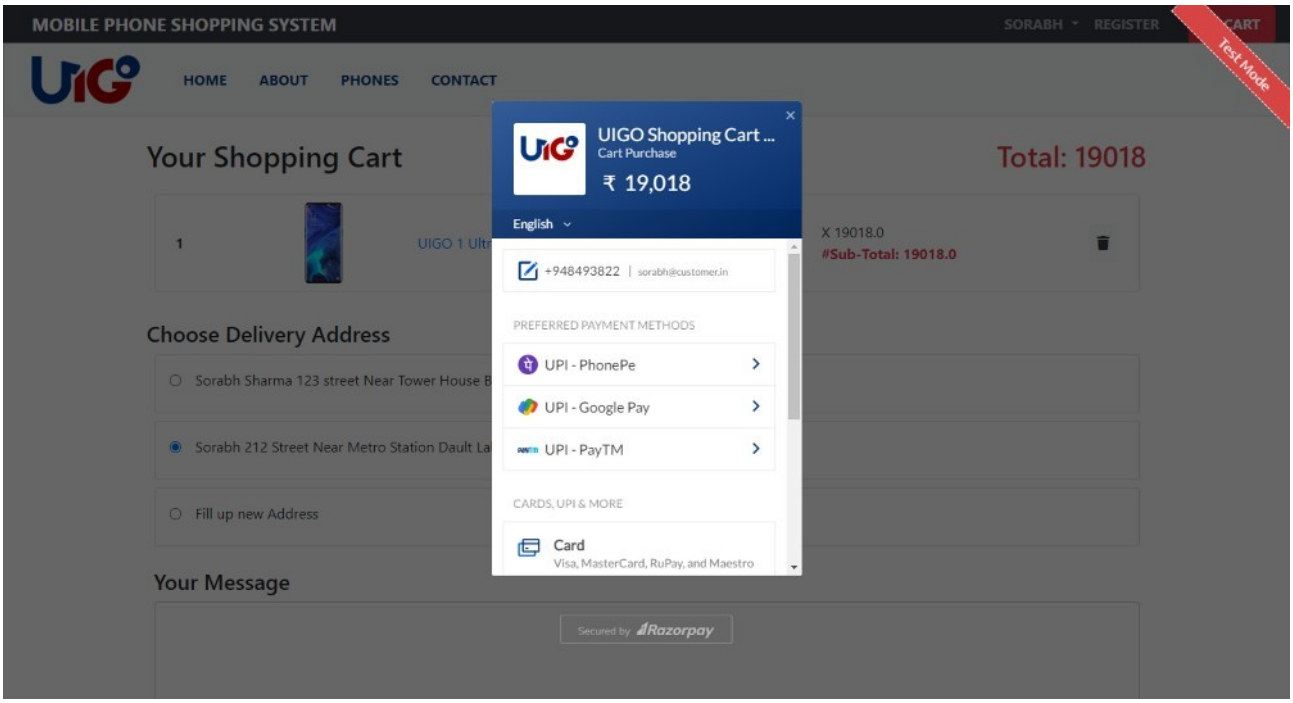
- **Cart Page:**


MOBILE PHONE SHOPPING SYSTEM
SORABH
REGISTER
CART
HOME
ABOUT
PHONES
CONTACT
Your Shopping Cart
Total: 19018
1
UIGO 1 Ultra
-
1
+
X 19018.0
#Sub-Total: 19018.0
Choose Delivery Address
Sorabh Sharma 123 street Near Tower House Banglore Tamilnadu India 284995
Sorabh 212 Street Near Metro Station Dault LakeWill United State 38843
Fill up new Address
Your Message
Fill up Address Details:
Full Name:
Your Full Name
phone:
Address Line 1
Address Line 1:
Address Line 1
Address Line 2:
Address Line 2
City:
Your City
State:
Your State
Country:
Your Country
Zip Code:
Zip Code
Choose Payment Method:
Cash on Delivery
Online Payment
Checkout/Payment
© Copyright to sorabh86.github.io, All Right Reserved.

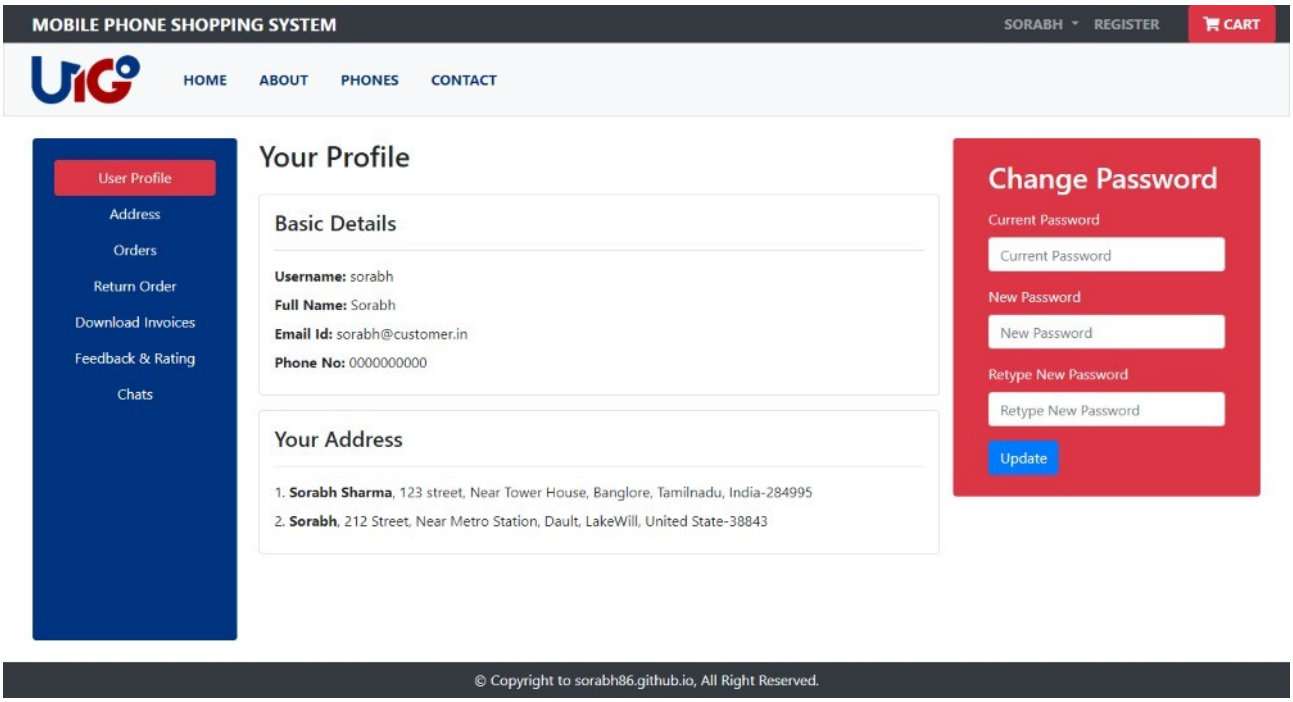
- **Order Receipt Page:**



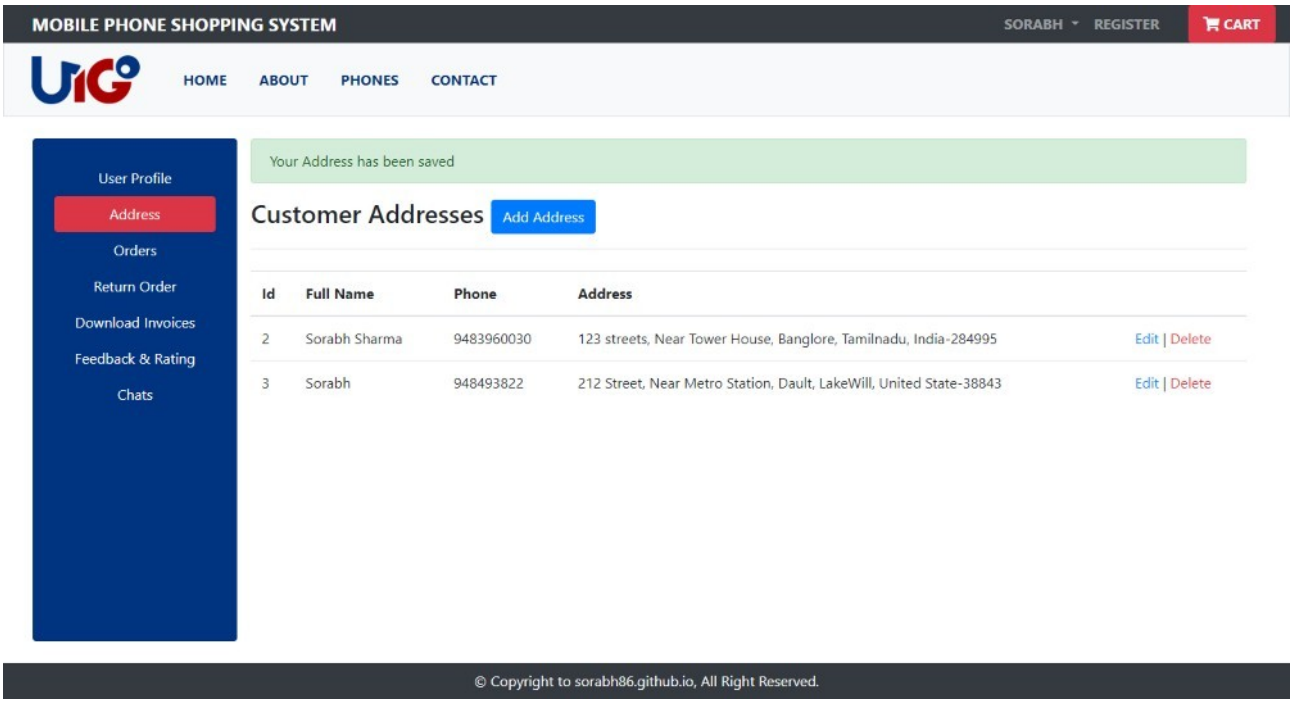
- **Razor Payment Page:**



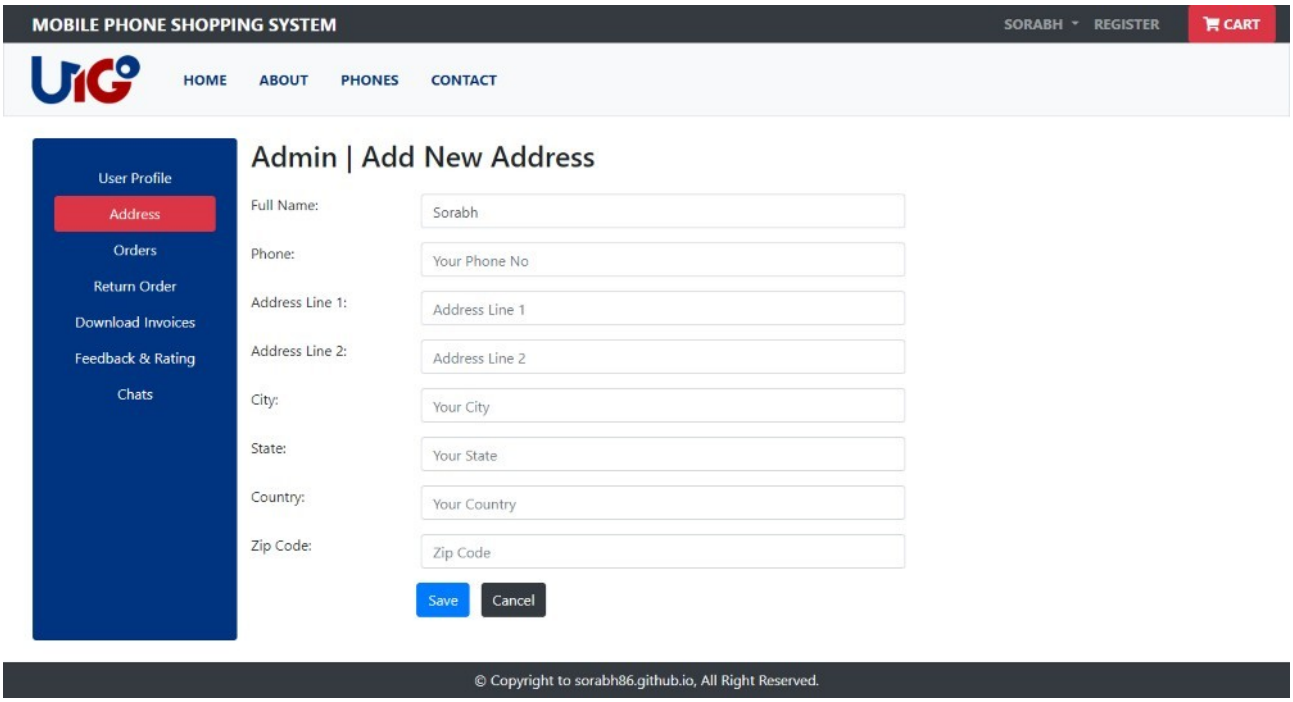
- **Customer Profile Page:**



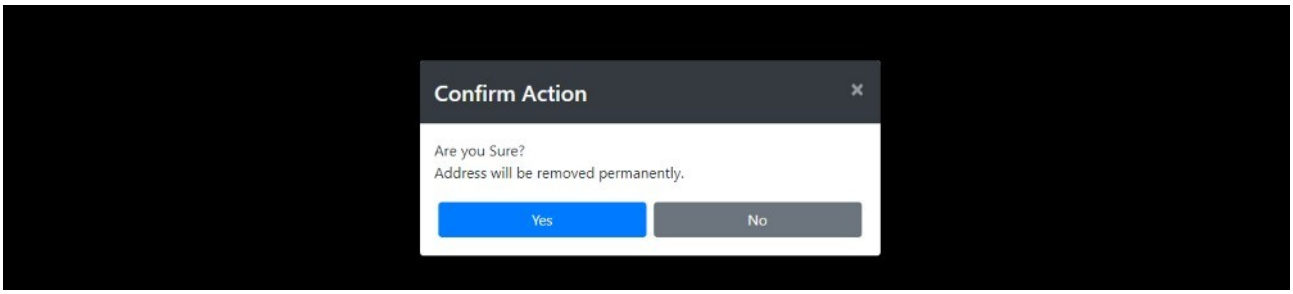
- **Customer Address Page:**



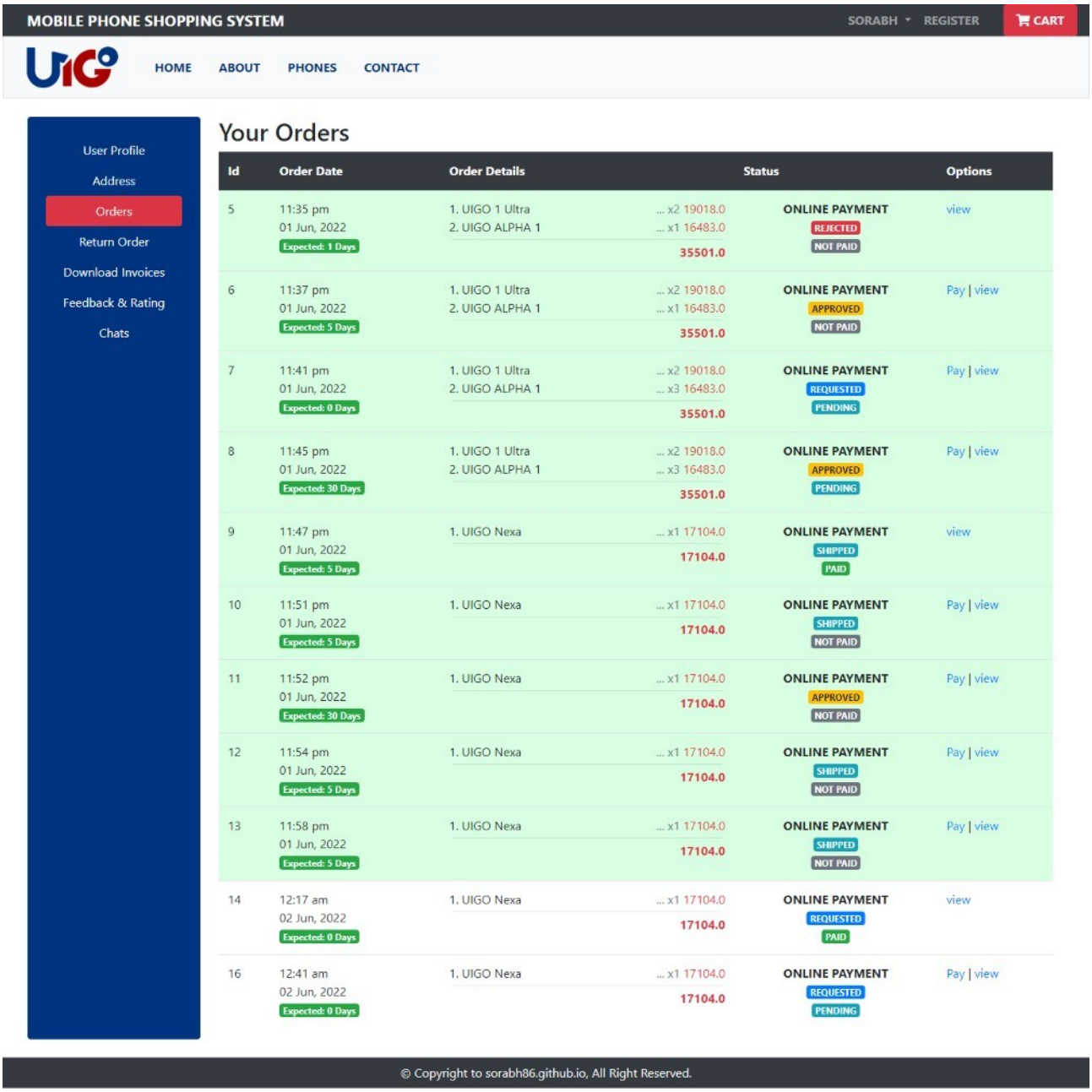
- **Customer Address Page (Add New/Edit Address):**



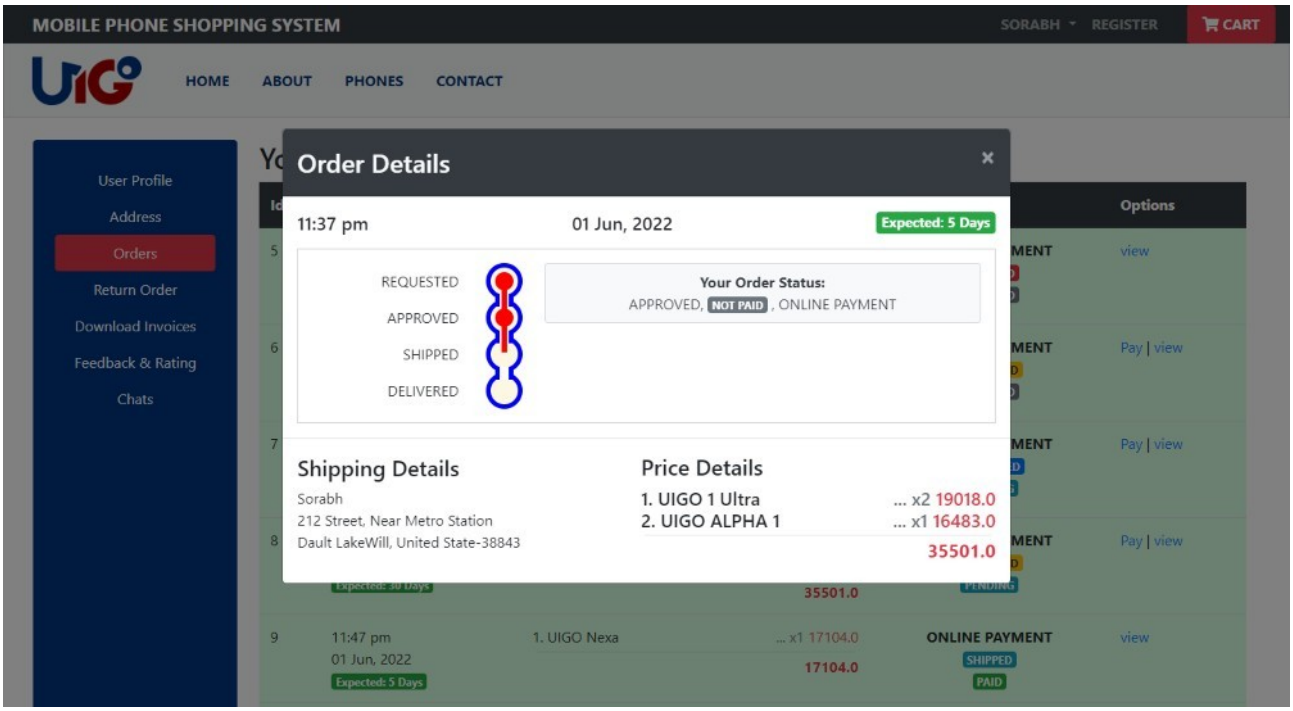
- **Customer Address Page (Delete Address):**



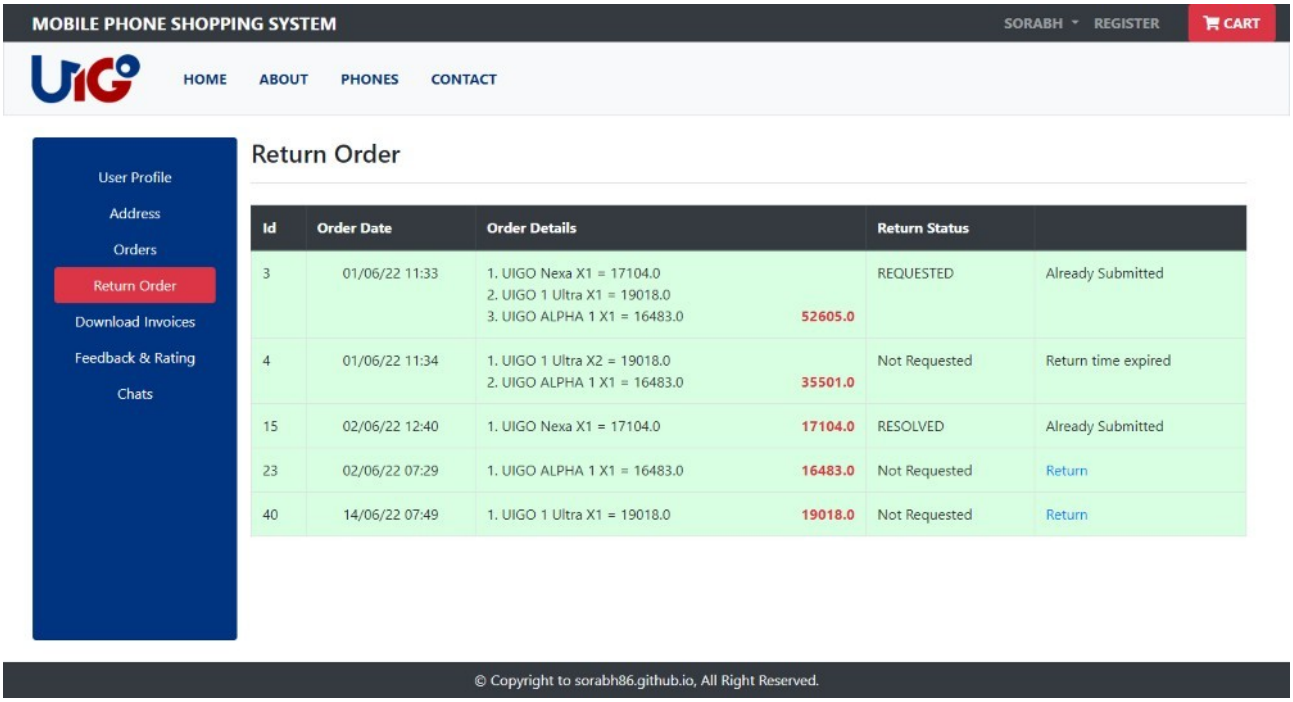
- **Customer Orders Page:**



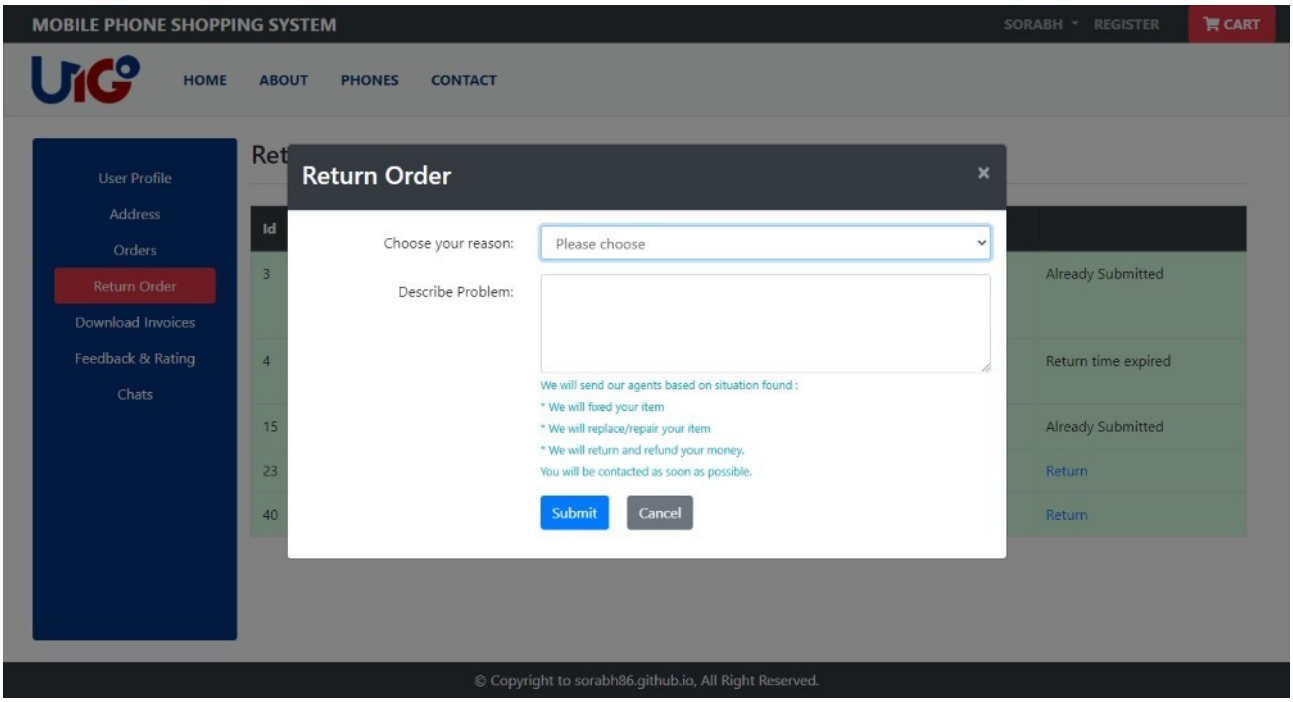
- **Customer Orders Page (view details):**



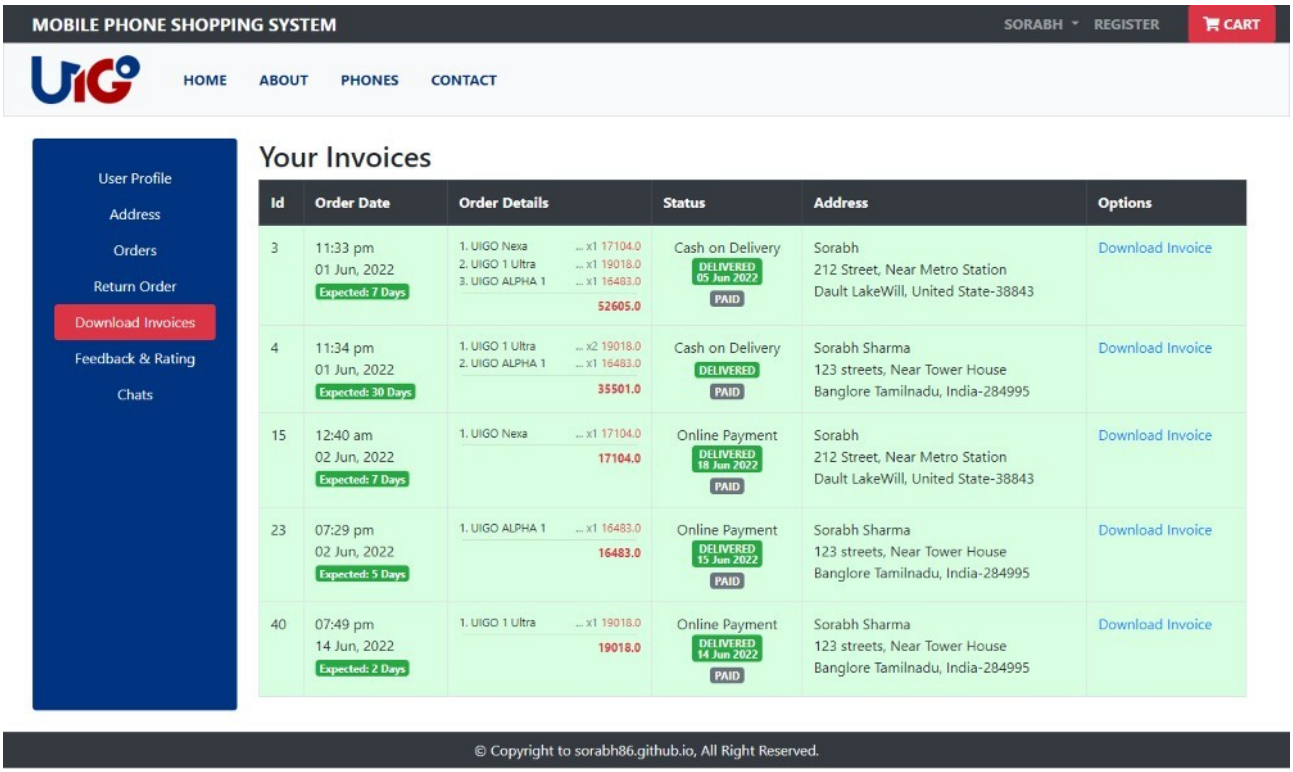
- **Customer Return Order Page:**



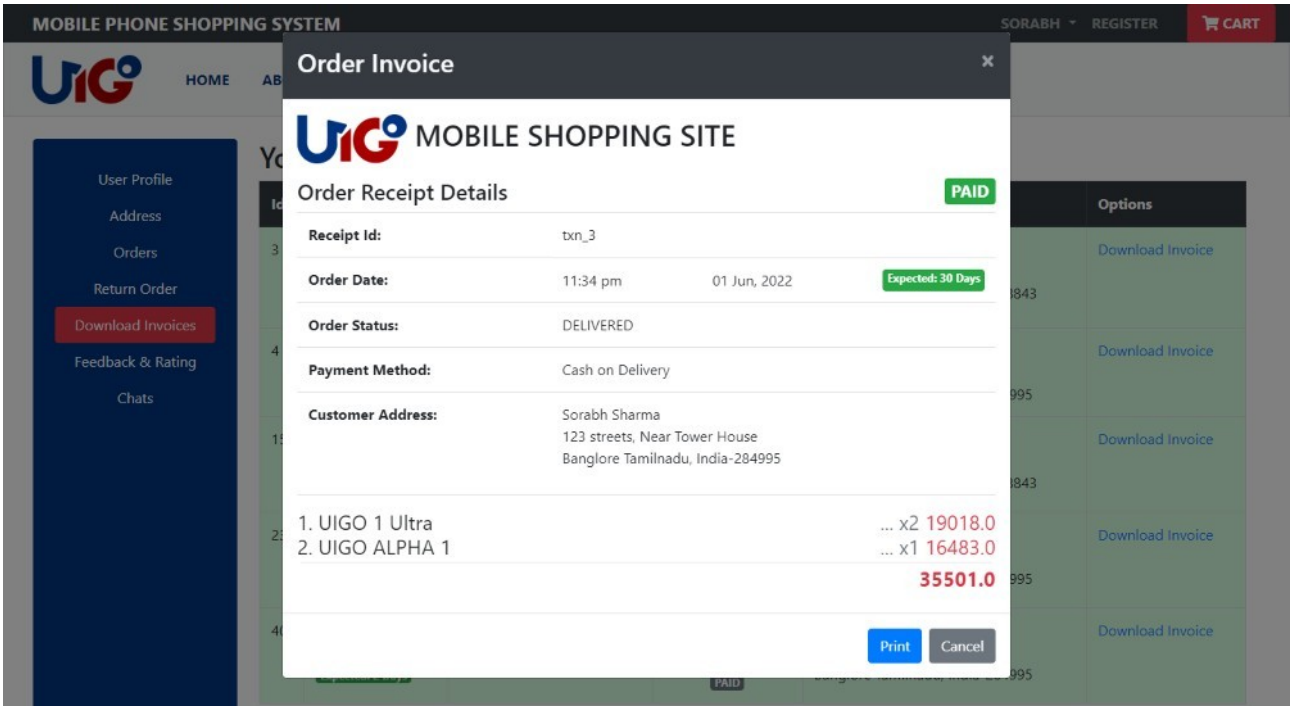
- **Customer Return Order Page (Return Request):**



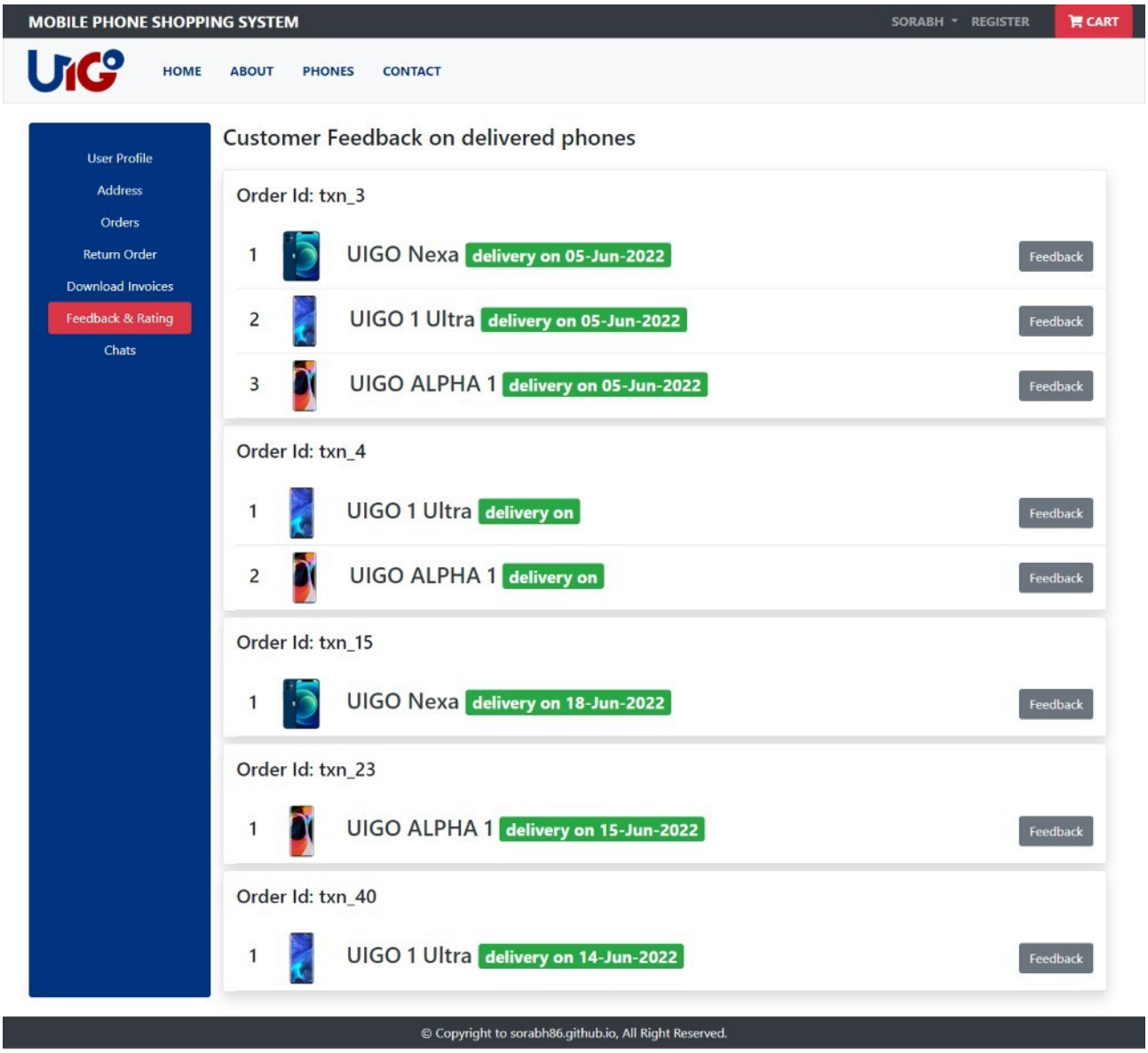
- **Customer Download Invoices Page:**



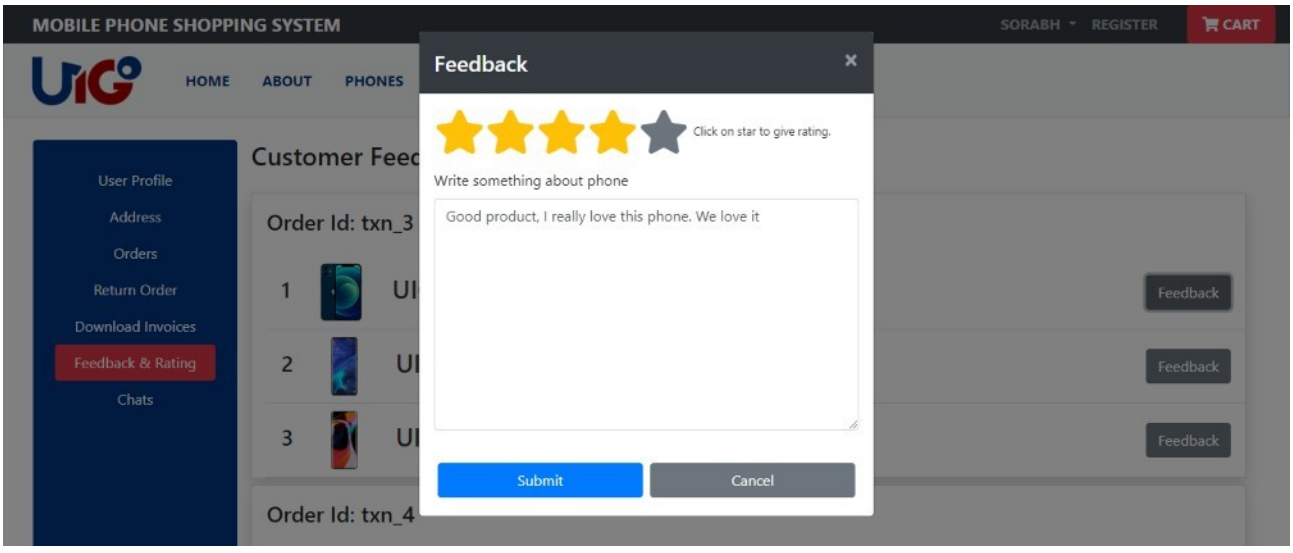
- **Customer Download Invoices Page (Download):**



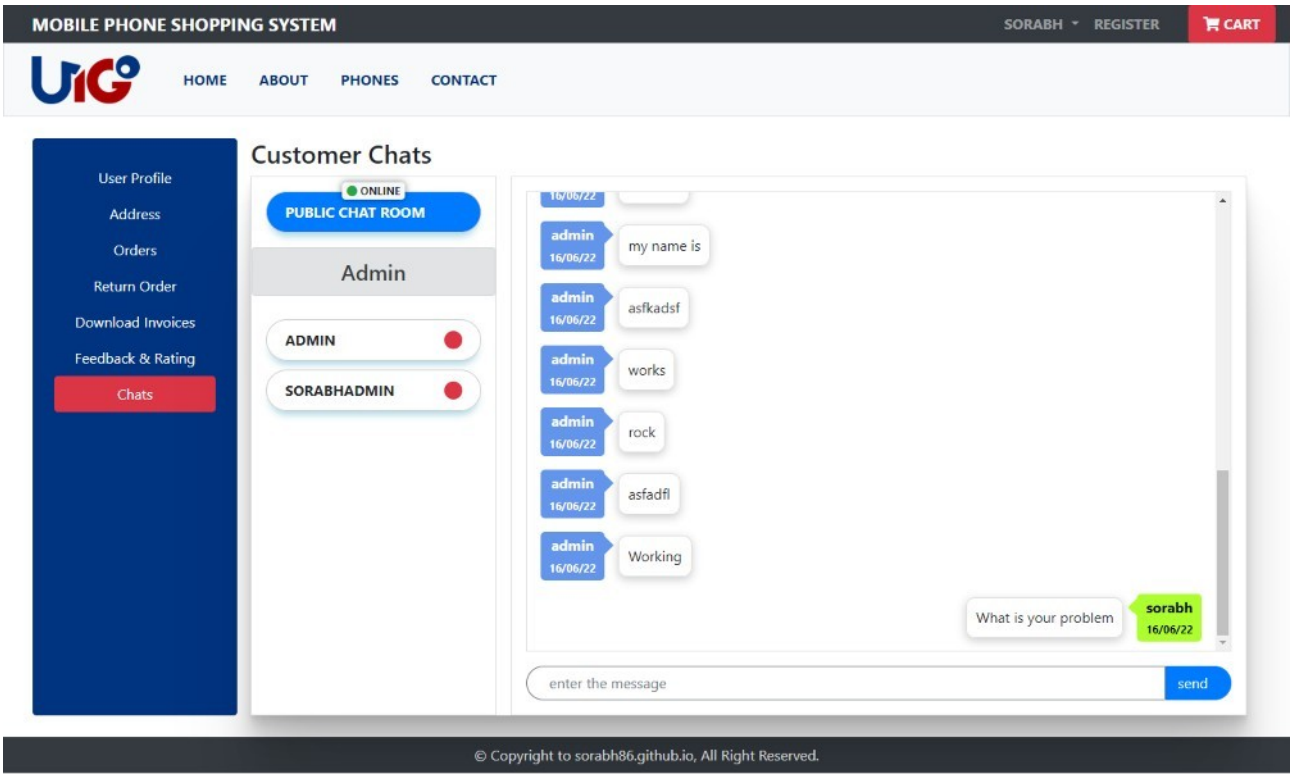
- **Customer Feedback & Rating Page:**



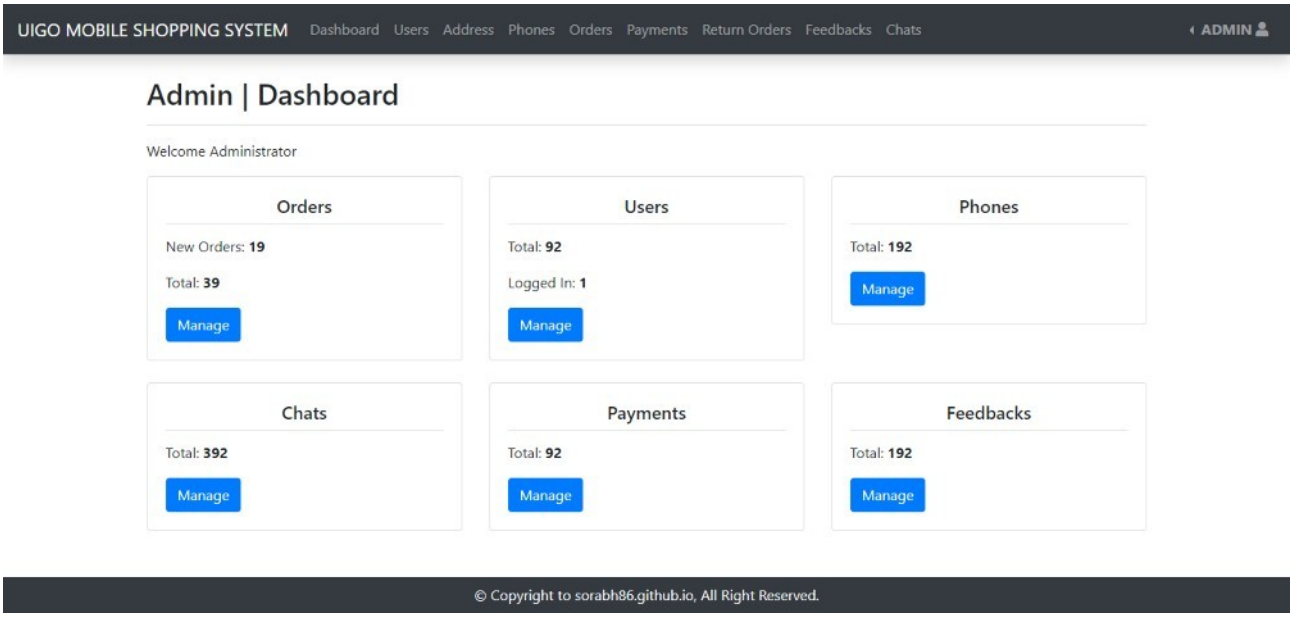
- **Customer Feedback & Rating Page (feedback):**



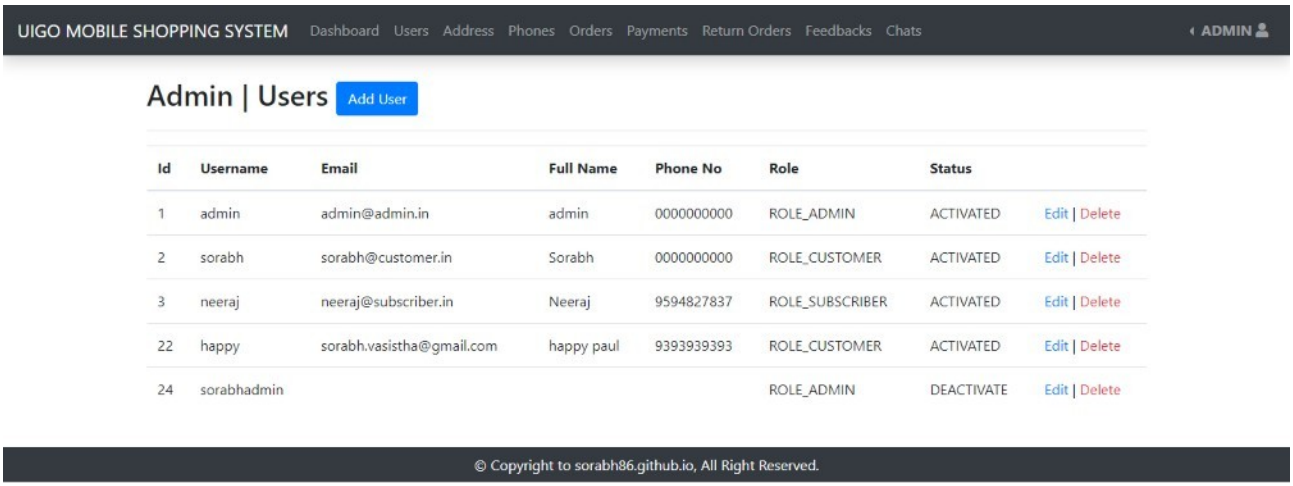
- **Customer Chat Page:**



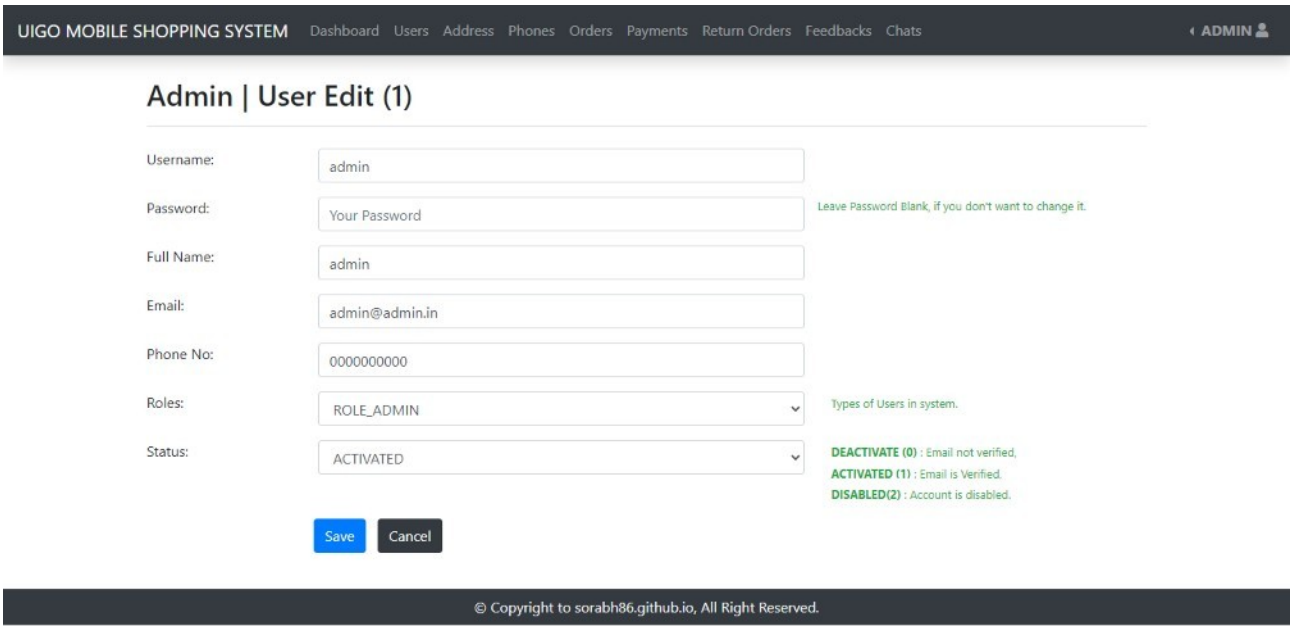
- **Admin Dashboard Page:**



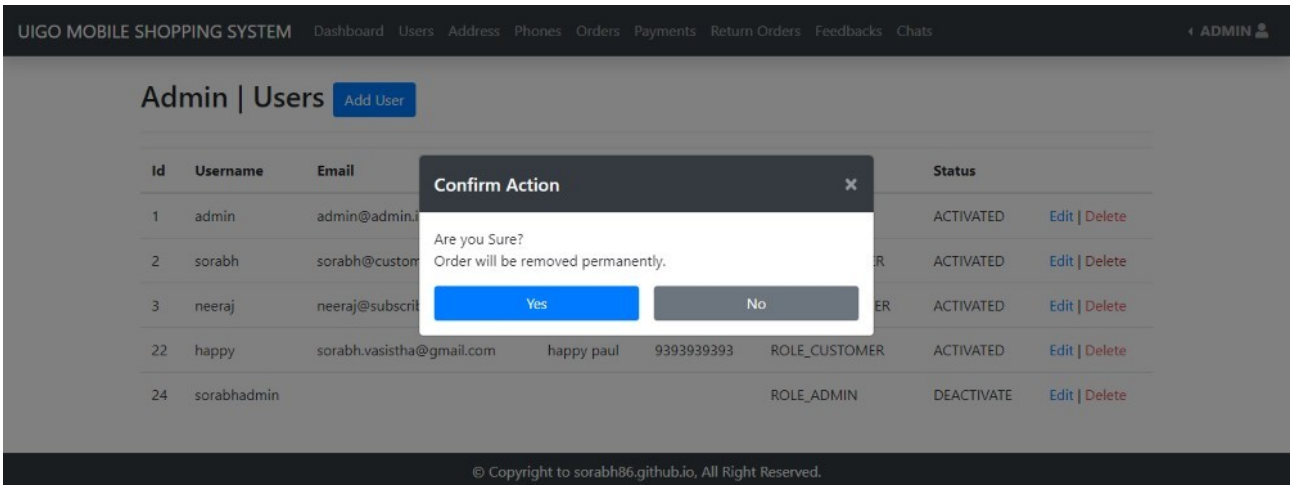
- **Admin Users Page:**



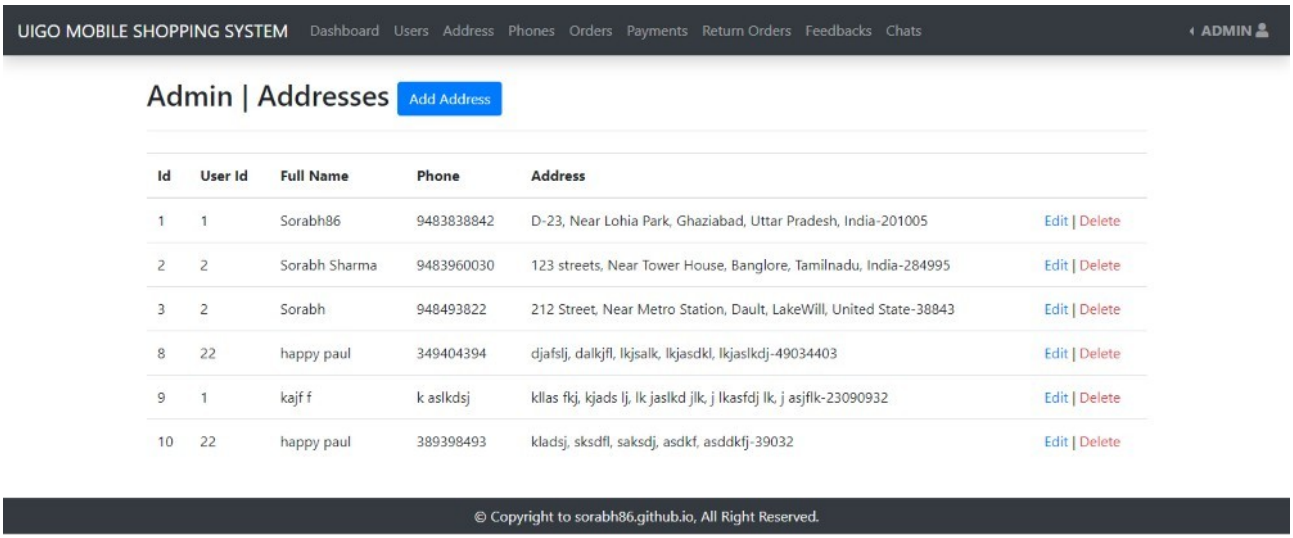
- **Admin Users Page (New/Edit):**



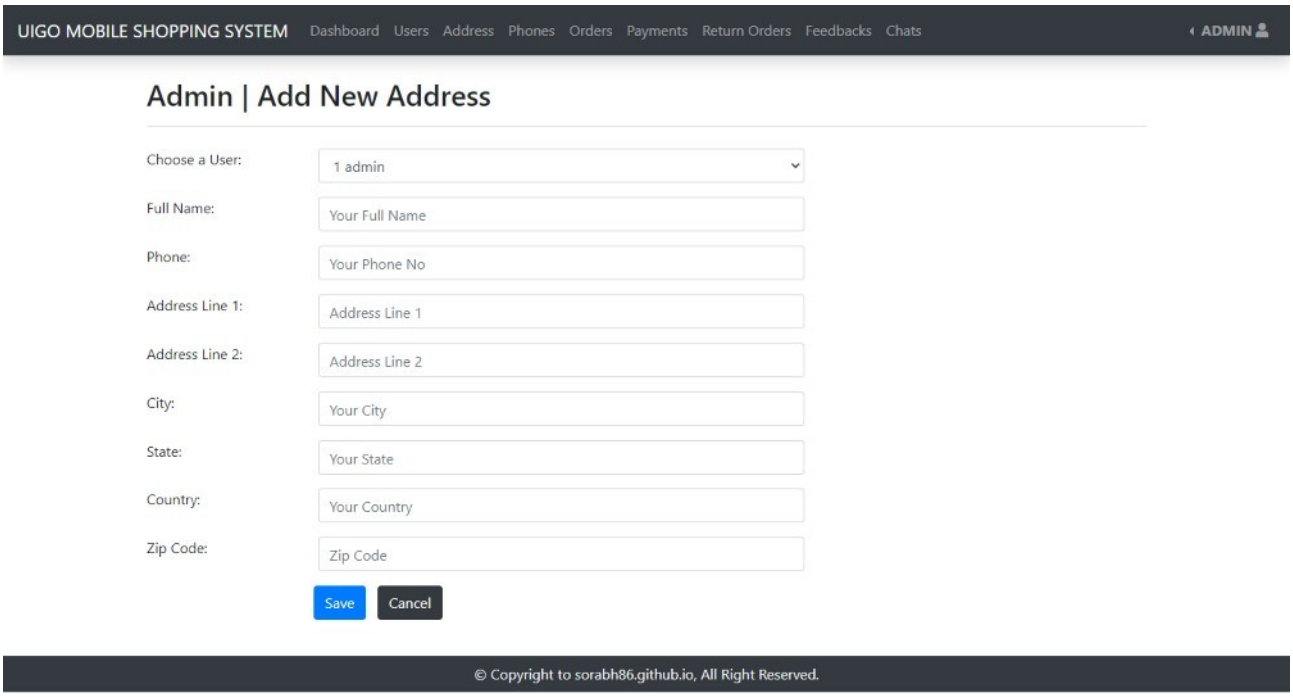
- **Admin Users Page (Delete):**



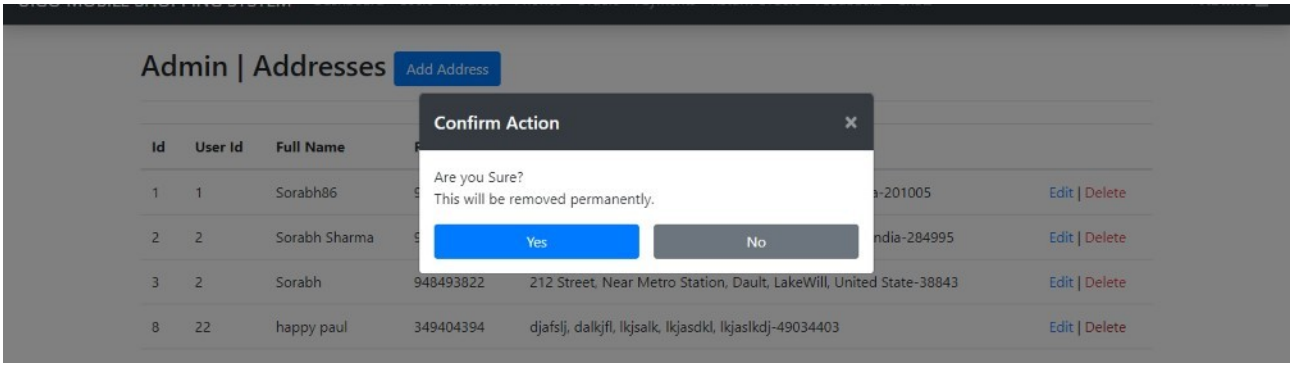
- **Admin Address Page:**



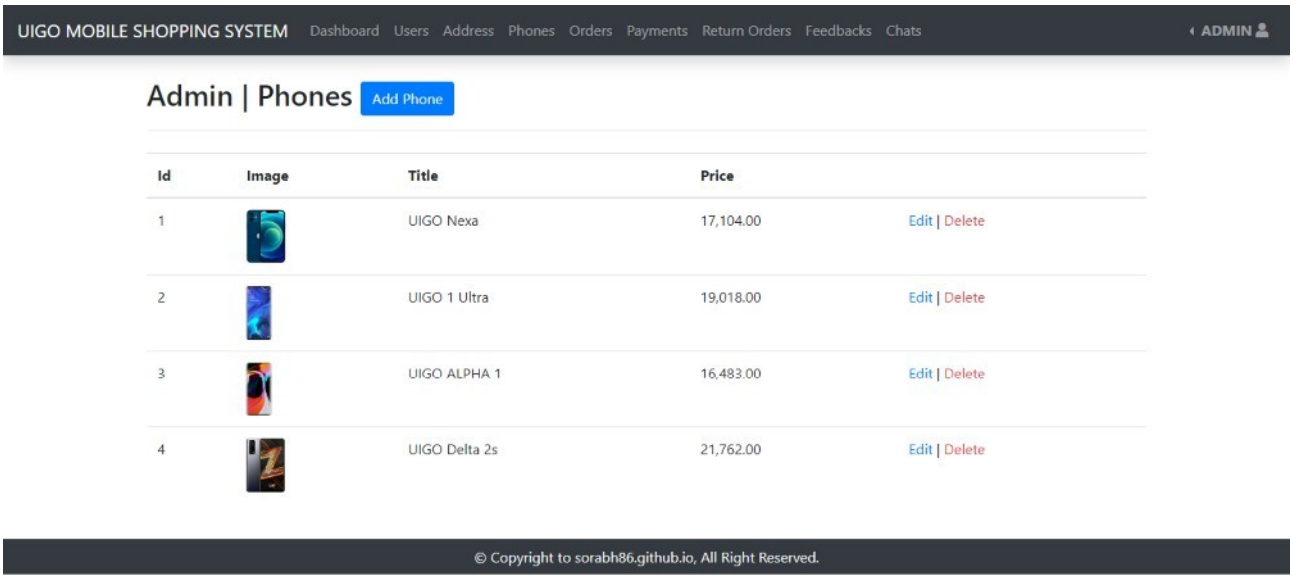
- **Admin Address Page (New/Edit):**



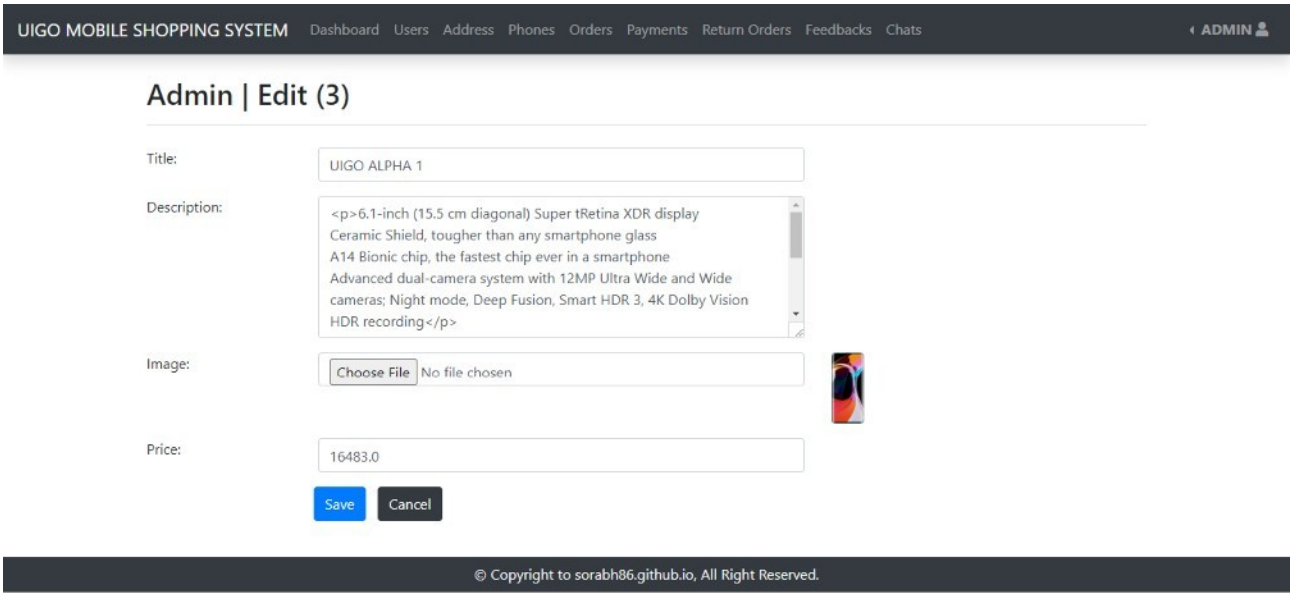
- **Admin Address Page (Delete):**



- **Admin Phones Page:**

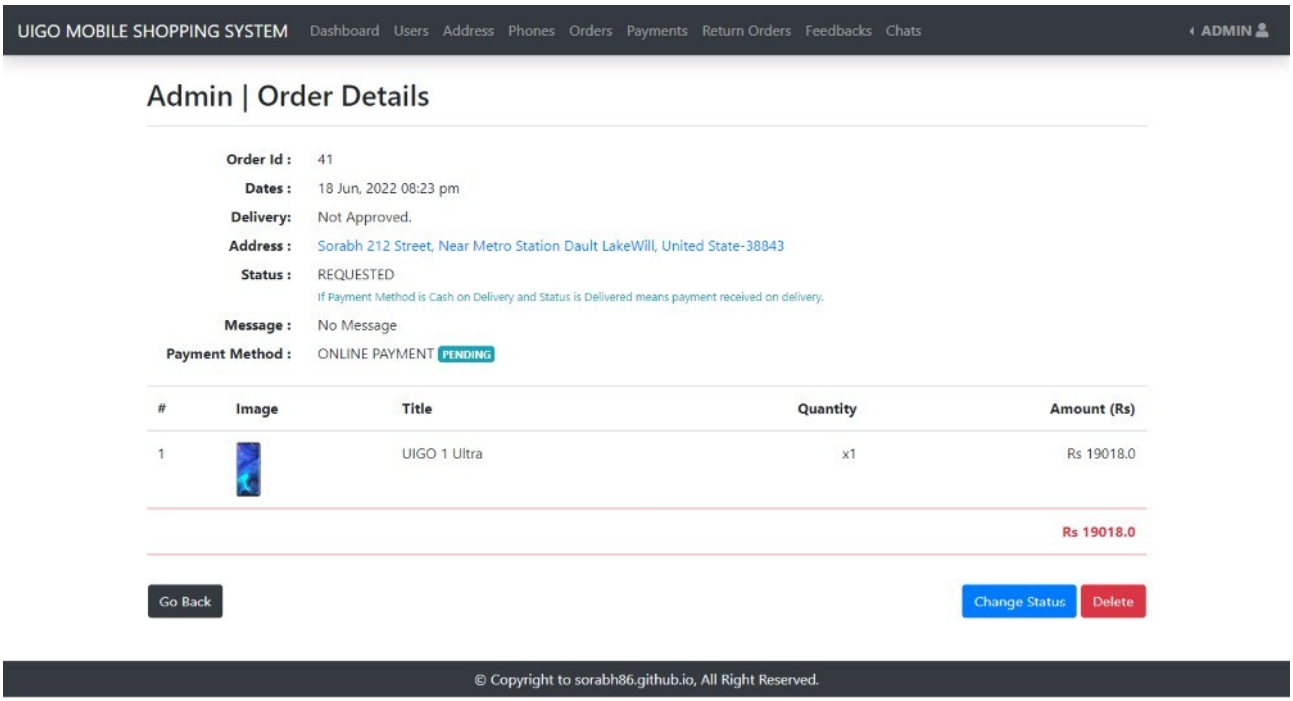


- **Admin Phones Page (Add/Edit):**

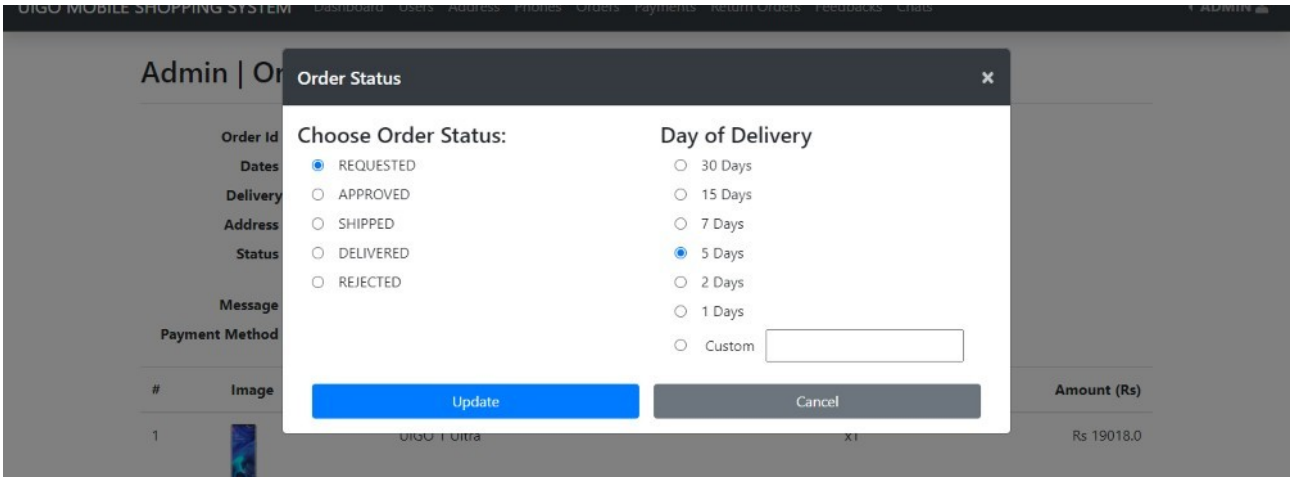


- **Admin Orders Page:**

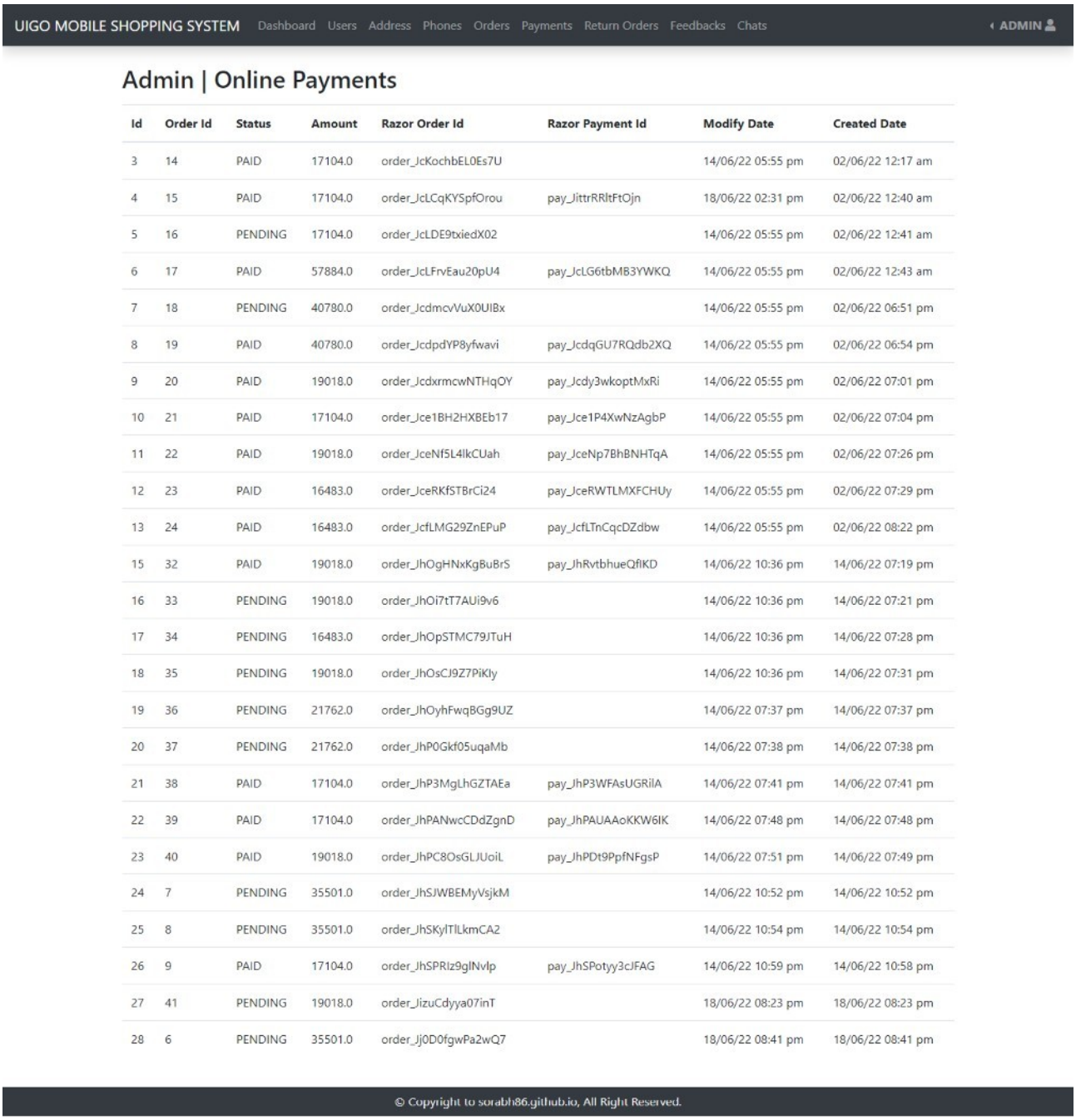
- **Admin Orders Page (View Details):**



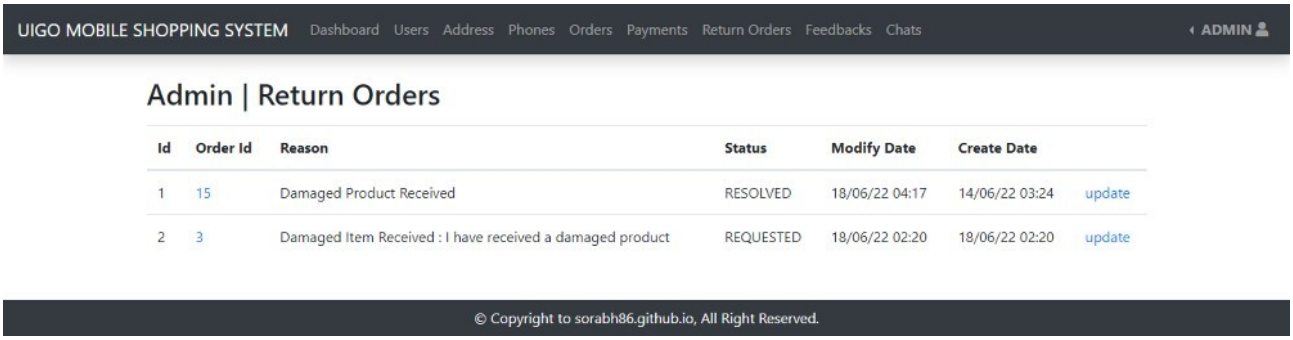
- **Admin Orders Page (Change Status):**



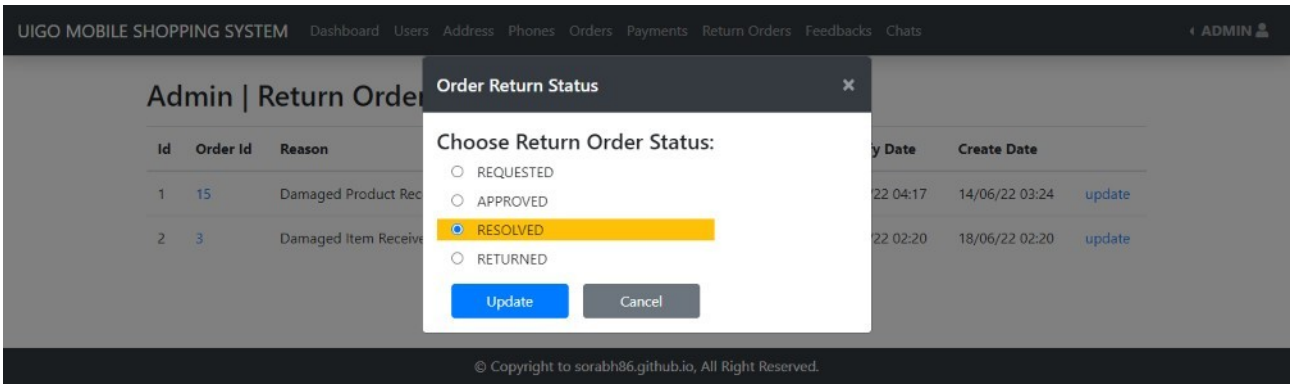
- **Admin Payments Page:**



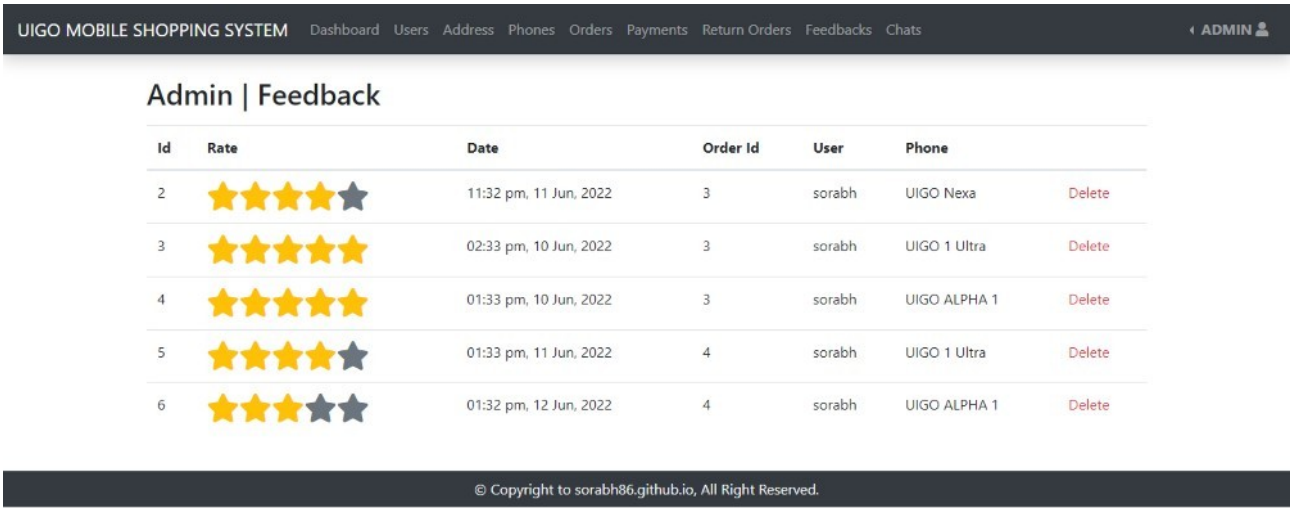
- **Admin Return Order Page:**



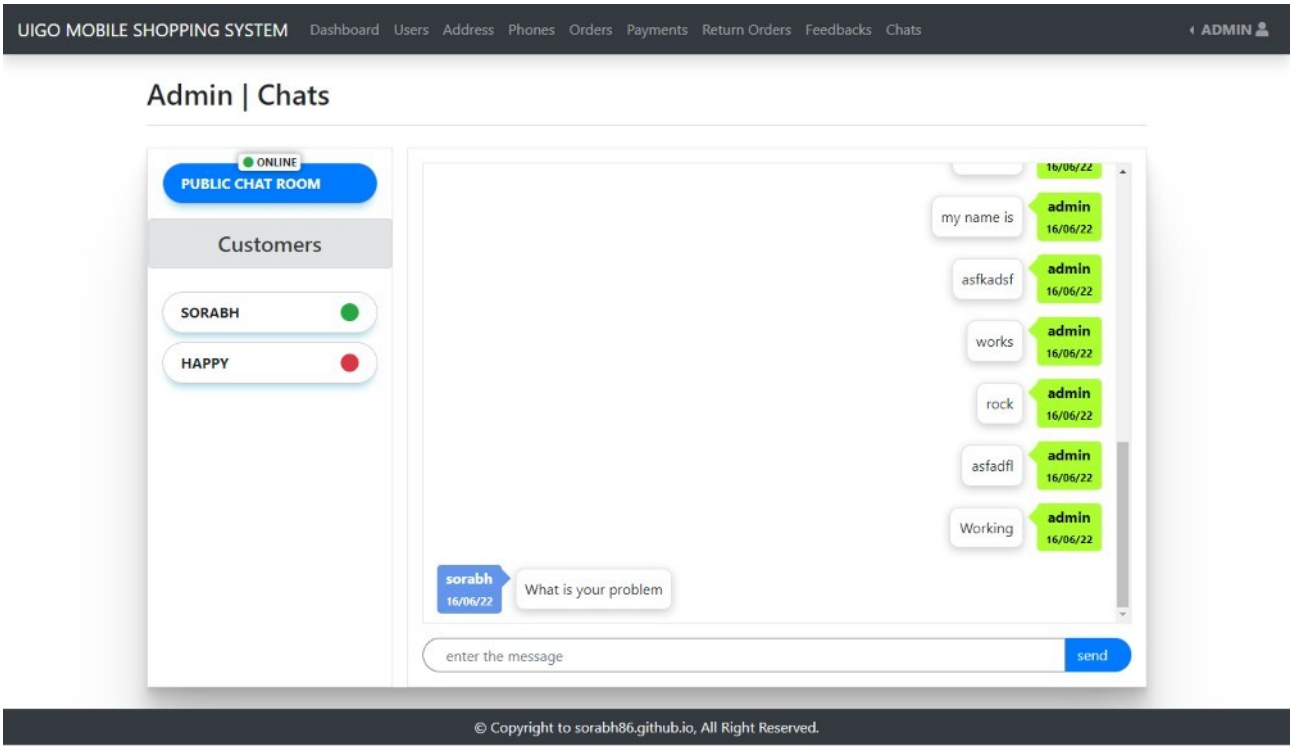
- **Admin Return Order Page (Update Status):**



- **Admin Feedback Page:**



- **Admin Chat Page:**



4. Coding

4.1. SQL:

4.1.1. database creating

```
--
-- Table structure for table `address`
--

CREATE TABLE `address` (
  `id` int(11) NOT NULL,
  `address_line_1` varchar(255) NOT NULL,
  `address_line_2` varchar(255) NOT NULL,
  `city` varchar(50) NOT NULL,
  `country` varchar(50) NOT NULL,
  `full_name` varchar(50) NOT NULL,
  `phone` varchar(16) DEFAULT NULL,
  `state` varchar(50) NOT NULL,
  `user_id` int(11) DEFAULT NULL,
  `zip` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `cart_items`
--

CREATE TABLE `cart_items` (
  `id` int(11) NOT NULL,
  `quantity` int(11) DEFAULT NULL,
  `phone_id` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `chat`
--

CREATE TABLE `chat` (
  `id` int(11) NOT NULL,
  `date` datetime(6) DEFAULT NULL,
  `message` varchar(255) DEFAULT NULL,
  `receiver_username` varchar(255) DEFAULT NULL,
  `status` varchar(50) DEFAULT NULL,
  `username` varchar(255) DEFAULT NULL,
  `is_visited` bit(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `feedback_rating`
--
```

```sql
CREATE TABLE `feedback_rating` (
  `id` int(11) NOT NULL,
  `rate` int(11) DEFAULT NULL,
  `user_feedback_msg` varchar(255) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  `order_id` int(11) DEFAULT NULL,
  `phone_id` int(11) DEFAULT NULL,
  `is_visted` bit(1) DEFAULT NULL,
  `date` datetime(6) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `orders`
--

CREATE TABLE `orders` (
  `id` int(11) NOT NULL,
  `created_date` datetime(6) DEFAULT NULL,
  `expected_delivery_days` int(11) DEFAULT NULL,
  `message` varchar(255) DEFAULT NULL,
  `payment_method` varchar(60) DEFAULT NULL,
  `status` varchar(50) DEFAULT NULL,
  `address_id` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  `delivery_date` datetime(6) DEFAULT NULL,
  `is_visited` bit(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `order_item`
--

CREATE TABLE `order_item` (
  `id` int(11) NOT NULL,
  `amount` float NOT NULL,
  `order_id` int(11) DEFAULT NULL,
  `quantity` int(11) DEFAULT NULL,
  `phone_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `payment`
--

CREATE TABLE `payment` (
  `id` int(11) NOT NULL,
  `order_id` int(11) DEFAULT NULL,
  `created_date` datetime(6) DEFAULT NULL,
  `modify_date` datetime(6) DEFAULT NULL,
  `status` varchar(20) DEFAULT NULL,
  `razor_order_id` varchar(255) DEFAULT NULL,
```

```
  `razor_payment_id` varchar(255) DEFAULT NULL,
  `amount` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `phone`
--

CREATE TABLE `phone` (
  `id` int(11) NOT NULL,
  `description` varchar(1024) NOT NULL,
  `image` varchar(255) DEFAULT NULL,
  `price` float NOT NULL,
  `publish_date` datetime(6) DEFAULT NULL,
  `title` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `return_order`
--

CREATE TABLE `return_order` (
  `id` int(11) NOT NULL,
  `created_date` datetime(6) DEFAULT NULL,
  `return_reason` varchar(255) DEFAULT NULL,
  `status` varchar(255) DEFAULT NULL,
  `order_id` int(11) DEFAULT NULL,
  `modify_date` datetime(6) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Table structure for table `user`
--

CREATE TABLE `user` (
  `id` int(11) NOT NULL,
  `activation_key` varchar(255) DEFAULT NULL,
  `email` varchar(60) NOT NULL,
  `full_name` varchar(60) DEFAULT NULL,
  `password` varchar(120) NOT NULL,
  `phone_no` varchar(16) DEFAULT NULL,
  `role` varchar(20) NOT NULL,
  `status` int(11) DEFAULT NULL,
  `username` varchar(60) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Indexes for table `address`
--
ALTER TABLE `address`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FKda8tuywtf0gb6sedwk7la1pgi` (`user_id`);
```

```sql
--
-- Indexes for table `cart_items`
--
ALTER TABLE `cart_items`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FKng28lgn25vvh6kwf3obtng6yx` (`phone_id`),
  ADD KEY `FKkjv4yjjdlt4hd9ayey6mti09m` (`user_id`);

--
-- Indexes for table `chat`
--
ALTER TABLE `chat`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `feedback_rating`
--
ALTER TABLE `feedback_rating`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FK40ujr2k539qaa8venvi92sunn` (`user_id`),
  ADD KEY `FKqbny93kowux5kfi5dqo9xp4s4` (`order_id`),
  ADD KEY `FKsmhibga1gq29pui42sco5vka8` (`phone_id`);

--
-- Indexes for table `orders`
--
ALTER TABLE `orders`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FKf5464gxwc32ongdvka2rtvw96` (`address_id`);

--
-- Indexes for table `order_item`
--
ALTER TABLE `order_item`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FKs7k485wawg3s5htgqj5fgalex` (`phone_id`),
  ADD KEY `FKt4dc2r9nbvbujrljv3e23iibt` (`order_id`);

--
-- Indexes for table `payment`
--
ALTER TABLE `payment`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `UK_jja7bpik8tl2prft0n9nknt5g` (`razor_order_id`),
  ADD UNIQUE KEY `UK_8mpqi5o0sg8x1xtvu2gvs65xe` (`razor_payment_id`),
  ADD KEY `FKlouu98csyullos9k25tbpk4va` (`order_id`);

--
-- Indexes for table `phone`
--
ALTER TABLE `phone`
```

```
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `return_order`
--
ALTER TABLE `return_order`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FKd2m1bv0p2swr9vqsmicjyou4o` (`order_id`);

--
-- Indexes for table `user`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `address`
--
ALTER TABLE `address`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `cart_items`
--
ALTER TABLE `cart_items`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=44;

--
-- AUTO_INCREMENT for table `chat`
--
ALTER TABLE `chat`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=686;

--
-- AUTO_INCREMENT for table `feedback_rating`
--
ALTER TABLE `feedback_rating`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;

--
-- AUTO_INCREMENT for table `orders`
--
ALTER TABLE `orders`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=43;

--
-- AUTO_INCREMENT for table `order_item`
```

```sql
--
ALTER TABLE `order_item`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=55;

--
-- AUTO_INCREMENT for table `payment`
--
ALTER TABLE `payment`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=29;

--
-- AUTO_INCREMENT for table `phone`
--
ALTER TABLE `phone`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;

--
-- AUTO_INCREMENT for table `return_order`
--
ALTER TABLE `return_order`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=25;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `address`
--
ALTER TABLE `address`
  ADD CONSTRAINT `FKda8tuywtf0gb6sedwk7la1pgi` FOREIGN KEY (`user_id`)
REFERENCES `user` (`id`);

--
-- Constraints for table `cart_items`
--
ALTER TABLE `cart_items`
  ADD CONSTRAINT `FKkjv4yjjdlt4hd9ayey6mti09m` FOREIGN KEY (`user_id`)
REFERENCES `user` (`id`),
  ADD CONSTRAINT `FKng28lgn25vvh6kwf3obtng6yx` FOREIGN KEY (`phone_id`)
REFERENCES `phone` (`id`);

--
-- Constraints for table `feedback_rating`
--
```

```
ALTER TABLE `feedback_rating`
  ADD CONSTRAINT `FK40ujr2k539qaa8venvi92sunn` FOREIGN KEY (`user_id`)
REFERENCES `user` (`id`),
  ADD CONSTRAINT `FKqbny93kowux5kfi5dqo9xp4s4` FOREIGN KEY (`order_id`)
REFERENCES `orders` (`id`),
  ADD CONSTRAINT `FKsmhibga1gq29pui42sco5vka8` FOREIGN KEY (`phone_id`)
REFERENCES `phone` (`id`);

--
-- Constraints for table `orders`
--
ALTER TABLE `orders`
  ADD CONSTRAINT `FKf5464gxwc32ongdvka2rtvw96` FOREIGN KEY (`address_id`)
REFERENCES `address` (`id`);

--
-- Constraints for table `order_item`
--
ALTER TABLE `order_item`
  ADD CONSTRAINT `FKs7k485wawg3s5htgqj5fgalex` FOREIGN KEY (`phone_id`)
REFERENCES `phone` (`id`),
  ADD CONSTRAINT `FKt4dc2r9nbvbujrljv3e23iibt` FOREIGN KEY (`order_id`)
REFERENCES `orders` (`id`);

--
-- Constraints for table `payment`
--
ALTER TABLE `payment`
  ADD CONSTRAINT `FKlouu98csyullos9k25tbpk4va` FOREIGN KEY (`order_id`)
REFERENCES `orders` (`id`);

--
-- Constraints for table `return_order`
--
ALTER TABLE `return_order`
  ADD CONSTRAINT `FKd2m1bv0p2swr9vqsmicjyou4o` FOREIGN KEY (`order_id`)
REFERENCES `orders` (`id`);
COMMIT;
```

4.1.2. data insertion in tables

```
--
-- Dumping data for table `address`
--

INSERT INTO `address` (`id`, `address_line_1`, `address_line_2`, `city`,
`country`, `full_name`, `phone`, `state`, `user_id`, `zip`) VALUES
(1, 'D-23', 'Near Lohia Park', 'Ghaziabad', 'India', 'Sorabh86', '9483838842',
'Uttar Pradesh', 1, '201005'),
(2, '123 streets', 'Near Tower House', 'Banglore', 'India', 'Sorabh Sharma',
```

```
'9483960030', 'Tamilnadu', 2, '284995'),
(3, '212 Street', 'Near Metro Station', 'Dault', 'United State', 'Sorabh',
'948493822', 'LakeWill', 2, '38843'),
(8, 'djafslj', 'dalkjfl', 'lkjsalk', 'lkjaslkdj', 'happy paul', '349404394',
'lkjasdkl', 22, '49034403'),
(9, 'kllas fkj', ' kjads lj', 'lk jaslkd jlk', 'j asjflk', 'kajf f', 'k aslkdsj
', 'j lkasfdj lk', 1, '23090932'),
(10, 'kladsj', 'sksdfl', 'saksdj', 'asddkfj', 'happy paul', '389398493',
'asdkf', 22, '39032');

--
-- Dumping data for table `cart_items`
--

INSERT INTO `cart_items` (`id`, `quantity`, `phone_id`, `user_id`) VALUES
(32, 1, 3, 22);

--
-- Dumping data for table `chat`
--

INSERT INTO `chat` (`id`, `date`, `message`, `receiver_username`, `status`,
`username`, `is_visited`) VALUES
(85, '2022-06-16 17:59:47.463000', 'hello man', NULL, 'MESSAGE', 'sorabh',
b'0'),
(86, '2022-06-16 18:00:10.052000', 'what do you want', 'sorabh', 'MESSAGE',
'admin', b'0'),
(152, '2022-06-16 18:40:52.721000', 'babby ii love it', NULL, 'MESSAGE',
'admin', b'0'),
(153, '2022-06-16 18:41:09.747000', 'what are you saying? are you mad?', NULL,
'MESSAGE', 'sorabh', b'0'),
(154, '2022-06-16 18:41:17.275000', 'baby i love you', NULL, 'MESSAGE',
'admin', b'0'),
(155, '2022-06-16 18:41:26.084000', 'so what?', NULL, 'MESSAGE', 'sorabh',
b'0'),
(156, '2022-06-16 18:41:30.305000', 'nothing', NULL, 'MESSAGE', 'admin', b'0'),
(161, '2022-06-16 18:41:42.289000', 'keep going', NULL, 'MESSAGE', 'admin',
b'0'),
(162, '2022-06-16 18:41:48.145000', 'where do go', NULL, 'MESSAGE', 'sorabh',
b'0'),
(163, '2022-06-16 18:42:05.227000', 'everywhere i go', NULL, 'MESSAGE',
'sorabh', b'0'),
(164, '2022-06-16 18:42:15.356000', 'shut up man', NULL, 'MESSAGE', 'admin',
b'0'),
(187, '2022-06-16 19:26:59.028000', 'hello', NULL, 'MESSAGE', 'sorabh', b'0'),
(207, '2022-06-16 19:39:46.728000', 'soarabh', NULL, 'MESSAGE', 'admin', b'0'),
(211, '2022-06-16 19:44:01.762000', 'my name is ', NULL, 'MESSAGE', 'admin',
b'0'),
(216, '2022-06-16 19:45:32.983000', 'asfkadsf', NULL, 'MESSAGE', 'admin',
b'0'),
(221, '2022-06-16 19:45:46.996000', 'works', NULL, 'MESSAGE', 'admin', b'0'),
(225, '2022-06-16 19:46:08.805000', 'rock', NULL, 'MESSAGE', 'admin', b'0'),
```

```
(228, '2022-06-16 19:47:05.726000', 'asfadfl', NULL, 'MESSAGE', 'admin', b'0'),
(232, '2022-06-16 20:15:12.163000', 'Working', NULL, 'MESSAGE', 'admin', b'0'),
(235, '2022-06-16 20:15:45.361000', 'working', 'admin', 'MESSAGE', 'sorabh',
b'0'),
(240, '2022-06-16 20:16:42.438000', 'good day sir', 'sorabh', 'MESSAGE',
'admin', b'0'),
(242, '2022-06-16 21:25:39.905000', 'i got to go', 'sorabh', 'MESSAGE',
'admin', b'0'),
(338, '2022-06-16 23:16:39.691000', 'What is your problem', NULL, 'MESSAGE',
'sorabh', b'0'),
(525, '2022-06-17 18:31:06.287000', 'skdfa', 'sorabh', 'MESSAGE', 'admin',
b'0'),
(528, '2022-06-17 18:37:02.746000', 'good morning', 'happy', 'MESSAGE',
'admin', b'0'),
(546, '2022-06-17 18:52:05.454000', 'kajdskfa', 'sorabh', 'MESSAGE', 'admin',
b'0'),
(548, '2022-06-17 19:00:18.035000', 'how are you? happy', 'happy', 'MESSAGE',
'admin', b'0'),
(551, '2022-06-17 20:05:02.361000', 'hello', 'happy', 'MESSAGE', 'admin',
b'0'),
(558, '2022-06-17 20:12:06.386000', 'sorabh', 'happy', 'MESSAGE', 'admin',
b'0'),
(564, '2022-06-17 20:13:01.436000', 'i am going to sleep now', 'happy',
'MESSAGE', 'admin', b'0'),
(569, '2022-06-17 20:13:49.774000', 'good day', 'happy', 'MESSAGE', 'admin',
b'0'),
(570, '2022-06-17 20:14:05.985000', 'good morning', 'admin', 'MESSAGE',
'happy', b'0'),
(571, '2022-06-17 20:15:04.803000', 'wola', 'admin', 'MESSAGE', 'happy', b'0'),
(575, '2022-06-17 20:15:49.950000', 'doma', 'admin', 'MESSAGE', 'happy', b'0'),
(579, '2022-06-17 20:18:22.559000', 'Hi, Sir how are you?', 'admin', 'MESSAGE',
'happy', b'0'),
(580, '2022-06-17 20:19:14.776000', 'i don\'t understand', 'happy', 'MESSAGE',
'admin', b'0'),
(583, '2022-06-17 20:21:30.609000', 'What are you saying?', 'happy', 'MESSAGE',
'admin', b'0'),
(611, '2022-06-17 22:40:45.763000', 'working', 'happy', 'MESSAGE', 'admin',
b'0'),
(613, '2022-06-17 22:42:24.958000', 'I wonder if we can talk', 'happy',
'MESSAGE', 'admin', b'0'),
(614, '2022-06-17 22:43:13.125000', 'can we talk', 'happy', 'MESSAGE', 'admin',
b'0'),
(621, '2022-06-17 22:48:00.021000', 'Yes, i want to talk with you', 'happy',
'MESSAGE', 'admin', b'0'),
(623, '2022-06-17 22:51:08.832000', 'Ok, fine. i will talk to you tomorrow',
'admin', 'MESSAGE', 'happy', b'0'),
(628, '2022-06-17 22:52:22.480000', 'I want to talk, today.', 'happy',
'MESSAGE', 'admin', b'0'),
(629, '2022-06-17 22:53:01.187000', 'no man, i am busy', 'admin', 'MESSAGE',
'happy', b'0'),
(650, '2022-06-17 23:03:55.238000', 'are you sure don\'t want to talk?',
```

```
'happy', 'MESSAGE', 'admin', b'0'),
(652, '2022-06-17 23:06:09.438000', 'Think again.', 'happy', 'MESSAGE',
'admin', b'0'),
(655, '2022-06-17 23:06:36.594000', 'many why are you messing with me?',
'admin', 'MESSAGE', 'happy', b'0'),
(661, '2022-06-17 23:08:39.870000', 'Can you please think it again?', 'happy',
'MESSAGE', 'admin', b'0'),
(662, '2022-06-17 23:09:39.638000', 'kick', 'happy', 'MESSAGE', 'admin', b'0'),
(663, '2022-06-17 23:11:42.949000', 'I don\'t understand what is wrong here',
'happy', 'MESSAGE', 'admin', b'0'),
(666, '2022-06-17 23:15:45.112000', 'leave it, man', 'admin', 'MESSAGE',
'happy', b'0'),
(667, '2022-06-17 23:16:34.574000', 'please, send me message, whenever online',
'sorabh', 'MESSAGE', 'admin', b'0'),
(675, '2022-06-17 23:26:08.501000', 'hello admin.', 'sorabhadmin', 'MESSAGE',
'sorabh', b'0');

--
-- Dumping data for table `feedback_rating`
--

INSERT INTO `feedback_rating` (`id`, `rate`, `user_feedback_msg`, `user_id`,
`order_id`, `phone_id`, `is_visted`, `date`) VALUES
(2, 3, 'Good product, I really love this phone. We love it', 2, 3, 1, b'0',
'2022-06-11 23:32:45.000000'),
(3, 4, 'This phone rocks', 2, 3, 2, b'0', '2022-06-10 14:33:27.000000'),
(4, 4, 'I purchase at 30% discount. i loved this product', 2, 3, 3, b'0',
'2022-06-10 13:33:41.000000'),
(5, 3, 'I purchase this phone for my friend, customer service was awesome.', 2,
4, 2, b'0', '2022-06-11 13:33:50.000000'),
(6, 2, 'Broken item received', 2, 4, 3, b'0', '2022-06-12 13:32:22.657000');

--
-- Dumping data for table `orders`
--

INSERT INTO `orders` (`id`, `created_date`, `expected_delivery_days`,
`message`, `payment_method`, `status`, `address_id`, `user_id`,
`delivery_date`, `is_visited`) VALUES
(3, '2022-06-01 23:33:59.647000', 7, 'something like it', 'COD', 'DELIVERED',
3, 2, '2022-06-05 19:18:10.000000', b'1'),
(4, '2022-06-01 23:34:49.274000', 30, 'Go for it instantly', 'COD',
'DELIVERED', 2, 2, NULL, b'1'),
(5, '2022-06-01 23:35:01.295000', 1, '', 'OP', 'REJECTED', 3, 2, NULL, b'1'),
(6, '2022-06-01 23:37:20.808000', 5, '', 'OP', 'APPROVED', 3, 2, '2022-06-13
15:46:23.293000', b'1'),
(7, '2022-06-01 23:41:54.070000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'1'),
(8, '2022-06-01 23:45:18.831000', 30, '', 'OP', 'APPROVED', 3, 2, NULL, b'1'),
(9, '2022-06-01 23:47:08.042000', 5, '', 'OP', 'SHIPPED', 3, 2, NULL, b'1'),
(10, '2022-06-01 23:51:19.501000', 5, '', 'OP', 'SHIPPED', 2, 2, NULL, b'1'),
```

```
(11, '2022-06-01 23:52:44.212000', 30, '', 'OP', 'APPROVED', 2, 2, NULL, b'1'),
(12, '2022-06-01 23:54:08.493000', 5, '', 'OP', 'SHIPPED', 3, 2, NULL, b'1'),
(13, '2022-06-01 23:58:27.121000', 5, '', 'OP', 'SHIPPED', 2, 2, NULL, b'1'),
(14, '2022-06-02 00:17:54.845000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(15, '2022-06-02 00:40:50.411000', 7, '', 'OP', 'DELIVERED', 3, 2, '2022-06-18
14:31:26.057000', b'1'),
(16, '2022-06-02 00:41:12.687000', NULL, 'going to pay you.', 'OP',
'REQUESTED', 3, 2, NULL, b'0'),
(17, '2022-06-02 00:43:42.750000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(18, '2022-06-02 18:51:09.949000', NULL, 'asdfklajsdfljad akf', 'COD',
'REQUESTED', 3, 2, NULL, b'0'),
(19, '2022-06-02 18:54:01.900000', NULL, 'asdfklajsdfljad akf', 'OP',
'REQUESTED', 3, 2, NULL, b'1'),
(20, '2022-06-02 19:01:49.056000', 1, '', 'OP', 'SHIPPED', 2, 2, NULL, b'1'),
(21, '2022-06-02 19:04:57.634000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(22, '2022-06-02 19:26:14.144000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(23, '2022-06-02 19:29:42.779000', 5, '', 'OP', 'DELIVERED', 2, 2, '2022-06-15
00:54:43.440000', b'1'),
(24, '2022-06-02 20:22:43.194000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(25, '2022-06-03 09:50:24.270000', NULL, '', 'OP', 'REQUESTED', 2, 2, NULL,
b'0'),
(26, '2022-06-03 09:50:34.582000', 1, '', 'OP', 'APPROVED', 2, 2, NULL, b'1'),
(27, '2022-06-03 10:02:22.286000', 7, '', 'OP', 'REJECTED', 3, 2, NULL, b'1'),
(28, '2022-06-03 10:02:46.886000', 5, '', 'OP', 'APPROVED', 3, 2, NULL, b'1'),
(29, '2022-06-13 00:31:04.398000', 5, 'Please send fast.', 'COD', 'APPROVED',
8, NULL, NULL, b'1'),
(31, '2022-06-14 19:04:15.726000', NULL, 'Last item worked well.', 'OP',
'REQUESTED', 3, 2, NULL, b'0'),
(32, '2022-06-14 19:19:44.550000', NULL, 'adfkasd fsaskdfa', 'OP', 'REQUESTED',
2, 2, NULL, b'0'),
(33, '2022-06-14 19:21:29.338000', NULL, '', 'OP', 'REQUESTED', 2, 2, NULL,
b'0'),
(34, '2022-06-14 19:28:25.972000', NULL, 'Send me the product.', 'OP',
'REQUESTED', 2, 2, NULL, b'0'),
(35, '2022-06-14 19:31:01.810000', NULL, 'Good product', 'OP', 'REQUESTED', 2,
2, NULL, b'0'),
(36, '2022-06-14 19:37:10.128000', NULL, 'asjlfkajslkd akldf', 'OP',
'REQUESTED', 2, 2, NULL, b'0'),
(37, '2022-06-14 19:38:40.447000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'0'),
(38, '2022-06-14 19:41:36.271000', NULL, 'Get me my item', 'OP', 'REQUESTED',
3, 2, NULL, b'0'),
(39, '2022-06-14 19:48:14.646000', NULL, '', 'OP', 'REQUESTED', 2, 2, NULL,
b'0'),
(40, '2022-06-14 19:49:54.459000', 2, '', 'OP', 'DELIVERED', 2, 2, '2022-06-14
23:01:51.070000', b'1'),
```

```
(41, '2022-06-18 20:23:36.771000', NULL, '', 'OP', 'REQUESTED', 3, 2, NULL,
b'1'),
(42, '2022-06-18 20:24:54.713000', NULL, '', 'COD', 'REQUESTED', 3, 2, NULL,
b'0');

--
-- Dumping data for table `order_item`
--

INSERT INTO `order_item` (`id`, `amount`, `order_id`, `quantity`, `phone_id`)
VALUES
(3, 17104, 3, 1, 1),
(4, 19018, 3, 1, 2),
(5, 16483, 3, 1, 3),
(6, 19018, 4, 2, 2),
(7, 16483, 4, 1, 3),
(8, 19018, 5, 2, 2),
(9, 16483, 5, 1, 3),
(10, 19018, 6, 2, 2),
(11, 16483, 6, 1, 3),
(12, 19018, 7, 2, 2),
(13, 16483, 7, 3, 3),
(14, 19018, 8, 2, 2),
(15, 16483, 8, 3, 3),
(16, 17104, 9, 1, 1),
(17, 17104, 10, 1, 1),
(18, 17104, 11, 1, 1),
(19, 17104, 12, 1, 1),
(20, 17104, 13, 1, 1),
(21, 17104, 14, 1, 1),
(22, 17104, 15, 1, 1),
(23, 17104, 16, 1, 1),
(24, 17104, 17, 1, 1),
(25, 19018, 17, 1, 2),
(26, 21762, 17, 1, 4),
(27, 19018, 18, 1, 2),
(28, 21762, 18, 1, 4),
(29, 19018, 19, 1, 2),
(30, 21762, 19, 1, 4),
(31, 19018, 20, 1, 2),
(32, 17104, 21, 1, 1),
(33, 19018, 22, 1, 2),
(34, 16483, 23, 1, 3),
(35, 16483, 24, 1, 3),
(36, 19018, 25, 1, 2),
(37, 19018, 26, 1, 2),
(38, 19018, 27, 1, 2),
(39, 16483, 28, 1, 3),
(40, 19018, 29, 1, 2),
(41, 16483, 29, 1, 3),
(43, 19018, 31, 1, 2),
```

```
(44, 19018, 32, 1, 2),
(45, 19018, 33, 1, 2),
(46, 16483, 34, 1, 3),
(47, 19018, 35, 1, 2),
(48, 21762, 36, 1, 4),
(49, 21762, 37, 1, 4),
(50, 17104, 38, 1, 1),
(51, 17104, 39, 1, 1),
(52, 19018, 40, 1, 2),
(53, 19018, 41, 1, 2),
(54, 19018, 42, 1, 2);

--
-- Dumping data for table `payment`
--

INSERT INTO `payment` (`id`, `order_id`, `created_date`, `modify_date`,
`status`, `razor_order_id`, `razor_payment_id`, `amount`) VALUES
(3, 14, '2022-06-02 00:17:55.871000', '2022-06-14 17:55:14.258000', 'PAID',
'order_JcKochbEL0Es7U', NULL, 17104),
(4, 15, '2022-06-02 00:40:51.456000', '2022-06-18 14:31:14.412000', 'PAID',
'order_JcLCqKYSpfOrou', 'pay_JittrRRltFtOjn', 17104),
(5, 16, '2022-06-02 00:41:13.413000', '2022-06-14 17:55:14.280000', 'PENDING',
'order_JcLDE9txiedX02', NULL, 17104),
(6, 17, '2022-06-02 00:43:43.405000', '2022-06-14 17:55:14.281000', 'PAID',
'order_JcLFrvEau20pU4', 'pay_JcLG6tbMB3YWKQ', 57884),
(7, 18, '2022-06-02 18:51:11.818000', '2022-06-14 17:55:14.283000', 'PENDING',
'order_JcdmcvVuX0UIBx', NULL, 40780),
(8, 19, '2022-06-02 18:54:02.693000', '2022-06-14 17:55:14.286000', 'PAID',
'order_JcdpdYP8yfwavi', 'pay_JcdqGU7RQdb2XQ', 40780),
(9, 20, '2022-06-02 19:01:50.256000', '2022-06-14 17:55:14.287000', 'PAID',
'order_JcdxrmcwNTHqOY', 'pay_Jcdy3wkoptMxRi', 19018),
(10, 21, '2022-06-02 19:04:58.359000', '2022-06-14 17:55:14.289000', 'PAID',
'order_Jce1BH2HXBEb17', 'pay_Jce1P4XwNzAgbP', 17104),
(11, 22, '2022-06-02 19:26:15.387000', '2022-06-14 17:55:14.290000', 'PAID',
'order_JceNf5L4lkCUah', 'pay_JceNp7BhBNHTqA', 19018),
(12, 23, '2022-06-02 19:29:43.883000', '2022-06-14 17:55:14.293000', 'PAID',
'order_JceRKfSTBrCi24', 'pay_JceRWTLMXFCHUy', 16483),
(13, 24, '2022-06-02 20:22:46.148000', '2022-06-14 17:55:14.295000', 'PAID',
'order_JcfLMG29ZnEPuP', 'pay_JcfLTnCqcDZdbw', 16483),
(15, 32, '2022-06-14 19:19:45.949000', '2022-06-14 22:36:58.954000', 'PAID',
'order_JhOgHNxKgBuBrS', 'pay_JhRvtbhueQfIKD', 19018),
(16, 33, '2022-06-14 19:21:30.616000', '2022-06-14 22:36:58.985000', 'PENDING',
'order_JhOi7tT7AUi9v6', NULL, 19018),
(17, 34, '2022-06-14 19:28:27.297000', '2022-06-14 22:36:58.988000', 'PENDING',
'order_JhOpSTMC79JTuH', NULL, 16483),
(18, 35, '2022-06-14 19:31:02.832000', '2022-06-14 22:36:58.991000', 'PENDING',
'order_JhOsCJ9Z7PiKIy', NULL, 19018),
(19, 36, '2022-06-14 19:37:11.801000', '2022-06-14 19:37:11.801000', 'PENDING',
'order_JhOyhFwqBGg9UZ', NULL, 21762),
(20, 37, '2022-06-14 19:38:41.260000', '2022-06-14 19:38:41.260000', 'PENDING',
```

```
'order_JhP0Gkf05uqaMb', NULL, 21762),
(21, 38, '2022-06-14 19:41:37.014000', '2022-06-14 19:41:50.715000', 'PAID',
'order_JhP3MgLhGZTAEa', 'pay_JhP3WFAsUGRilA', 17104),
(22, 39, '2022-06-14 19:48:15.718000', '2022-06-14 19:48:26.494000', 'PAID',
'order_JhPANwcCDdZgnD', 'pay_JhPAUAAoKKW6IK', 17104),
(23, 40, '2022-06-14 19:49:55.301000', '2022-06-14 19:51:39.681000', 'PAID',
'order_JhPC8OsGLJUoiL', 'pay_JhPDt9PpfNFgsP', 19018),
(24, 7, '2022-06-14 22:52:59.300000', '2022-06-14 22:52:59.300000', 'PENDING',
'order_JhSJWBEMyVsjkM', NULL, 35501),
(25, 8, '2022-06-14 22:54:22.247000', '2022-06-14 22:54:22.247000', 'PENDING',
'order_JhSKylTlLkmCA2', NULL, 35501),
(26, 9, '2022-06-14 22:58:35.616000', '2022-06-14 22:59:01.886000', 'PAID',
'order_JhSPRIz9glNvlp', 'pay_JhSPotyy3cJFAG', 17104),
(27, 41, '2022-06-18 20:23:38.581000', '2022-06-18 20:23:38.581000', 'PENDING',
'order_JizuCdyya07inT', NULL, 19018),
(28, 6, '2022-06-18 20:41:27.119000', '2022-06-18 20:41:27.119000', 'PENDING',
'order_Jj0D0fgwPa2wQ7', NULL, 35501);

--
-- Dumping data for table `phone`
--

INSERT INTO `phone` (`id`, `description`, `image`, `price`, `publish_date`,
`title`) VALUES
(1, 'Superb Smartphone with superb design.\r\n\r\nFeatures:\r\n4GB RAM, 64GB
ROM, 4G Lite supported, 6.6 inch display, 5000 Mah Battery.', 'phone-1.jpg',
17104, '2022-05-30 18:09:32.752000', 'UIGO Nexa'),
(2, 'Processor: snapdragon 405,\r\nMemory: 4GB,\r\nStorage: 32GB,\r\nBattery:
40Wh,\r\nGorilla Glass 4\r\n', 'UIGO1Ultra.jpg', 19018, '2022-05-30
18:09:32.770000', 'UIGO 1 Ultra'),
(3, '<p>6.1-inch (15.5 cm diagonal) Super tRetina XDR display\r\nCeramic
Shield, tougher than any smartphone glass\r\nA14 Bionic chip, the fastest chip
ever in a smartphone\r\nAdvanced dual-camera system with 12MP Ultra Wide and
Wide cameras; Night mode, Deep Fusion, Smart HDR 3, 4K Dolby Vision HDR
recording</p>\r\n<p>12MP TrueDepth front camera with Night mode, 4K Dolby
Vision HDR recording\r\nIndustry-leading IP68 water resistance\r\nSupports
MagSafe accessories for easy attach and faster wireless charging\r\niOS with
redesigned widgets on the Home screen, all-new App Library, App Clips and
more</p>', 'UIGOALPHA1.jpg', 16483, '2022-05-30 18:09:32.772000', 'UIGO ALPHA
1'),
(4, 'Processor: G90\r\nMemory: 3GB\r\nROM: 16GB\r\nBattery: 4080Mah',
'UIGODelta2s.jpg', 21762, '2022-05-30 18:09:32.775000', 'UIGO Delta 2s');

--
-- Dumping data for table `return_order`
--

INSERT INTO `return_order` (`id`, `created_date`, `return_reason`, `status`,
`order_id`, `modify_date`) VALUES
(1, '2022-06-14 15:24:05.293000', 'Damaged Product Received', 'RESOLVED', 15,
'2022-06-18 16:17:08.199000'),
(2, '2022-06-18 14:20:56.506000', 'Damaged Item Received : I have received a
```

```
damaged product', 'REQUESTED', 3, '2022-06-18 14:20:56.506000');

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`id`, `activation_key`, `email`, `full_name`, `password`,
`phone_no`, `role`, `status`, `username`) VALUES
(1, NULL, 'admin@admin.in', 'admin',
'$2a$10$Fo9V6p2SUiRMWHuD785ZTeHBavAmMH1IBS76/Jz/ZAgf9SR2klwVS', '0000000000',
'ROLE_ADMIN', 1, 'admin'),
(2, NULL, 'sorabh@customer.in', 'Sorabh',
'$2a$10$IKfsisH/xpGe9D0HbULtAOL0wxqplGIQ.QRWrd5qEtbEFVJZrvufi', '0000000000',
'ROLE_CUSTOMER', 1, 'sorabh'),
(3, '', 'neeraj@subscriber.in', 'Neeraj',
'$2a$10$FwIb7CGqoCFjSZOBZI8Vju4v8XnJ9hX9f0VpauFtmMlVo/VJ.Jnoi', '9594827837',
'ROLE_SUBSCRIBER', 1, 'neeraj'),
(22, 'SOS778486', 'sorabh.vasistha@gmail.com', 'happy paul',
'$2a$10$Z2c2mqcaFiX8qj./MMlK/ewz8SvilxCyYn81ySOsqP4R04aW/htIS', '9393939393',
'ROLE_CUSTOMER', 1, 'happy'),
(24, 'SOS746386', '', '',
'$2a$10$8.Pl4sHxjVkkLIbe2NxEle0L92kj9I23vdP31EMeHD9Ih26VPLdW.', '',
'ROLE_ADMIN', 0, 'sorabhadmin');
```

4.2. Complete Project Coding

This is a Maven based Spring boot project:

- **pom.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <parent>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-parent</artifactId>
                <version>2.6.8</version>
                <relativePath/> <!-- lookup parent from repository -->
        </parent>
        <groupId>com.github.sorabh86.mss</groupId>
        <artifactId>MobileShoppingSite</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <name>MobileShoppingSite</name>
        <description>UIGO Shopping Application</description>
        <properties>
                <java.version>17</java.version>
        </properties>
        <dependencies>
```

```
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-data-jpa</artifactId>
                </dependency>
                <dependency>
                        <groupId>mysql</groupId>
                        <artifactId>mysql-connector-java</artifactId>
                        <scope>runtime</scope>
                </dependency>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-security</artifactId>
                </dependency>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-thymeleaf</artifactId>
                </dependency>
                <dependency>
                    <groupId>org.thymeleaf.extras</groupId>
                    <artifactId>thymeleaf-extras-springsecurity5</artifactId>
                </dependency>

                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-web</artifactId>
                </dependency>
                <dependency>
                        <groupId>org.thymeleaf.extras</groupId>
                        <artifactId>thymeleaf-extras-
springsecurity5</artifactId>
                </dependency>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-mail</artifactId>
                </dependency>
                <dependency>
                        <groupId>com.razorpay</groupId>
                        <artifactId>razorpay-java</artifactId>
                        <version>1.3.9</version>
                </dependency>

                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-test</artifactId>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-test</artifactId>
                        <scope>test</scope>
                </dependency>
```

```
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-devtools</artifactId>
                </dependency>
                <!--
https://mvnrepository.com/artifact/com.google.code.gson/gson -->
                <dependency>
                    <groupId>com.google.code.gson</groupId>
                    <artifactId>gson</artifactId>
                    <version>2.8.9</version>
                </dependency>

                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-websocket</artifactId>
                </dependency>
        </dependencies>

        <build>
                <plugins>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-
plugin</artifactId>
                        </plugin>
                </plugins>
        </build>

</project>
```

- **application.properties**

```
#server.servlet.session.timeout=10s
#server.servlet.session.cookie.max-age=10s

#############################
## Database
#############################
spring.datasource.url=jdbc:mysql://localhost:3306/mobile_shopping_site
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.dbcp2.driver-class-name=com.mysql.cj.jdbc.Driver

###################################
## JPA
###################################
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
#spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
#spring.jpa.properties.hibernate.format_sql=true

#################################
```

```
# Mail server properties
##################################
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=uigo.mobile@gmail.com
# Please create your gmail account, turn on 2 way authentication and generate a
app password and replace your email and password
spring.mail.password=******** #
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.connectiontimeout=5000
spring.mail.properties.mail.smtp.timeout=5000
spring.mail.properties.mail.smtp.writetimeout=5000
spring.mail.properties.mail.smtp.starttls.enable=true

# fix file with space doesn't showing
spring.mvc.pathmatch.matching-strategy=ant-path-matcher
```

- **MobileShoppingSiteApplication.java**

```
package com.github.sorabh86.uigo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;

import com.github.sorabh86.uigo.admin.users.UserRepo;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.User;

@SpringBootApplication
@EntityScan({"com.github.sorabh86.uigo"})
public class MobileShoppingSiteApplication implements CommandLineRunner {

//	@Autowired
//	private PasswordEncoder passwordEncoder;
//
//	@Autowired
//	private UserRepo userRepo;

	public static void main(String[] args) {
		SpringApplication.run(MobileShoppingSiteApplication.class,
args);
	}

	@Override
	public void run(String... args) throws Exception {
//		User admin = new User();
```

```
//                admin.setId(1);
//                admin.setUsername("admin");
//                admin.setEmail("admin@admin.in");
//                admin.setPassword(passwordEncoder.encode("admin"));
//                admin.setFull_name("admin");
//                admin.setPhone_no("0000000000");
//                admin.setStatus(UserStatus.ACTIVATED);
//                admin.setRole(UserRoles.ROLE_ADMIN);
//                userRepo.save(admin);
//
//                User sorabh = new User();
//                sorabh.setId(2);
//                sorabh.setUsername("sorabh");
//                sorabh.setEmail("sorabh@customer.in");
//                sorabh.setPassword(passwordEncoder.encode("sorabh"));
//                sorabh.setFull_name("Sorabh");
//                sorabh.setPhone_no("0000000000");
//                sorabh.setStatus(UserStatus.ACTIVATED);
//                sorabh.setRole(UserRoles.ROLE_CUSTOMER);
//                userRepo.save(sorabh);
//
//                User neeraj = new User();
//                neeraj.setId(3);
//                neeraj.setUsername("neeraj");
//                neeraj.setEmail("neeraj@subscriber.in");
//                neeraj.setPassword(passwordEncoder.encode("neeraj"));
//                neeraj.setFull_name("Neeraj");
//                neeraj.setPhone_no("0000000000");
//                neeraj.setStatus(UserStatus.ACTIVATED);
//                neeraj.setRole(UserRoles.ROLE_SUBSCRIBER);
//                userRepo.save(neeraj);
        }

}
```

- **MainController.java**

```
package com.github.sorabh86.uigo;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
```

```java
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.github.sorabh86.uigo.admin.orders.OrderService;
import com.github.sorabh86.uigo.admin.phones.PhoneService;
import com.github.sorabh86.uigo.admin.users.UserService;
import com.github.sorabh86.uigo.constants.PhoneRates;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.FeedbackRating;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;
import com.razorpay.Order;
import com.razorpay.RazorpayClient;
import com.razorpay.RazorpayException;

@Controller
public class MainController {

	@Autowired
	private UserService userService;

	@Autowired
	private PhoneService phoneService;

	@GetMapping("/")
	public String homePage(Model m) {
		m.addAttribute("page", "home");
		return "home";
	}

	@GetMapping(value="/razorpay", produces = "application/json")
	@ResponseBody
	public String razorpay() throws RazorpayException {
		RazorpayClient client = new
RazorpayClient("rzp_test_nLIXAPQblFuAPV", "aCSDzmDy8ELRFtdOTUmaufzL");
//		List<Order> object = client.Orders.fetchAll();
		Order object = client.Orders.fetch("order_JcDPHqmvolZuql");
		return object.toString();
	}

	@GetMapping("/login")
	public String login(Model m) {
		m.addAttribute("page", "login");
		return "login";
	}

	@GetMapping("/register")
```

```
        public String register(Model m) {
                User user = new User();
                user.setRole(UserRoles.ROLE_CUSTOMER);
                user.setStatus(UserStatus.DEACTIVATE);
                m.addAttribute("user", user);
                m.addAttribute("page", "register");
                return "register";
        }

        @RequestMapping("/phones")
        public String phonePage(Model m) {
                List<Phone> phones = this.phoneService.getPhones();
                m.addAttribute("page", "phone");
                m.addAttribute("phones", phones);
                return "phones";
        }

        @GetMapping("/phones/{id}")
        public String phoneDetailPage(@PathVariable("id")Integer id, Model m) {
                List<FeedbackRating> feedbacks = new ArrayList<>();

                FeedbackRating feed1 = new FeedbackRating();
                feed1.setUser(userService.getUserByUsername("admin"));
                feed1.setRate(PhoneRates.Three);
                feed1.setUser_feedback_msg("This is Good Product");
                feedbacks.add(feed1);

                FeedbackRating feed2 = new FeedbackRating();
                feed2.setUser(userService.getUserByUsername("sorabh"));
                feed2.setRate(PhoneRates.Five);
                feed2.setUser_feedback_msg("Lorem ipsum dolor sit amet
consectetur adipisicing elit. Quae fugiat consequatur reprehenderit possimus
ipsa voluptatibus nesciunt adipisci repellat, minus inventore, dignissimos ut
sint hic iure voluptatem exercitationem illo culpa dolorum.t");
                feedbacks.add(feed2);

                FeedbackRating feed3 = new FeedbackRating();
                feed3.setUser(userService.getUserByUsername("neeraj"));
                feed3.setRate(PhoneRates.Four);
                feed3.setUser_feedback_msg("This is Good Product");
                feedbacks.add(feed3);

                Phone phone = this.phoneService.getPhone(id);
                m.addAttribute("id", id);
                m.addAttribute("page", "phone");
                m.addAttribute("phone", phone);
                m.addAttribute("redirect", "/phones/"+id);
                m.addAttribute("feedbacks", feedbacks);
                return "phone-details";
        }
```

```
	@GetMapping("/about")
	public String aboutPage(Model m) {
		m.addAttribute("page", "about");
		return "about";
	}
	@GetMapping("/contact")
	public String contactPage(Model m, @ModelAttribute("flash")String flash)
{
		m.addAttribute("page", "contact");
		m.addAttribute("flash", flash);
		System.out.println(flash);

		return "contact";
	}

	@PostMapping("/contact")
	public String doContact(RedirectAttributes ra, @RequestParam
HashMap<String, Object> data) {
		System.out.println(data);
		ra.addFlashAttribute("flash", "Your Message Submitted");
		return "redirect:/contact";
	}

	@GetMapping("/verification/{id}/{code}")
	public String verification(Model m,
		@PathVariable("id")Integer user_id,
		@PathVariable("code")String code
	) {

		User user = userService.getUserById(user_id);

		System.out.println(user.getId()+" "+user.getActivation_key()+"
"+code);
		if(user.getActivation_key().equals(code)) {
			if(user.getStatus()==UserStatus.DEACTIVATE) {
				m.addAttribute("message", "Your email is
verified for this Account");
				user.setStatus(UserStatus.ACTIVATED);
				userService.save(user);
			} else {
				m.addAttribute("message", "Your Account is
already verified");
			}
		} else {
			m.addAttribute("message", "We are failed to verify your
account");
		}

		return "verification";
	}
```

```
}
```

- **FileUpload.java**

```
package com.github.sorabh86.uigo.utils;

import java.io.IOException;
import java.io.InputStream;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardCopyOption;
import java.util.List;
import java.util.stream.Stream;

import org.springframework.web.multipart.MultipartFile;

public class FileUpload {

	public static void savefile(String uploadDir, String fileName,
MultipartFile file) throws IOException {
		Path uploadPath = Paths.get(uploadDir);

		if(!Files.exists(uploadPath)) {
			Files.createDirectories(uploadPath);
		}

		try(InputStream stream = file.getInputStream()) {
			Path filePath = uploadPath.resolve(fileName);
			Files.copy(stream, filePath,
StandardCopyOption.REPLACE_EXISTING);
		} catch (IOException e) {
			throw new IOException("File could not saved.");
		}
	}

	public static void deletefile(String directory) throws IOException {
		Path dir = Paths.get(directory);

		try {
			Stream<Path> paths = Files.list(dir);
			paths.forEach(file->{
				if(!Files.isDirectory(file)) {
					System.out.println(file.toString());
					try {
						Files.delete(file);
					} catch (IOException e) {
						System.out.println("Couldn't
delete file: "+file);
					}
				}
			});
```

```
                } catch (IOException e) {
//                      e.printStackTrace();
//                      throw new IOException("Could not list directory");
                        System.out.println("Directory doesn't exist at:
"+directory);
                }

        }
}
```

- **User.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.ArrayList;
import java.util.List;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.Transient;

import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;

@Entity
public class User {

        @Id
        @GeneratedValue(strategy = GenerationType.IDENTITY)
        private Integer id;

        @Column(length = 60, nullable = false, unique = true)
        private String username;

        @Column(length = 120, nullable = false)
        private String password;

        @Column(length = 60)
        private String full_name;

        @Column(length = 60, nullable = false)
```

```java
    private String email;

    @Column(length = 16)
    private String phone_no;

    @Enumerated(EnumType.STRING)
    @Column(length = 20, nullable = false)
    private UserRoles role;

    @Column(length = 255, nullable = true)
    private String activation_key;

    @Enumerated(EnumType.ORDINAL)
    @Column(length = 8)
    private UserStatus status;

    @OneToMany(
            cascade = CascadeType.ALL,
            fetch = FetchType.EAGER
    )
    @JoinColumn(name = "user_id")
    private List<Address> addresses;

    public User() {}

    public User(String username, String password,
                String full_name, String email,
                String phone_no, UserRoles role,
                String activation_key, UserStatus status) {
        this.username = username;
        this.password = password;
        this.full_name = full_name;
        this.email = email;
        this.phone_no = phone_no;
        this.role = role;
        this.activation_key = activation_key;
        this.status = status;
    }

    public void addAddress(Address address) {
        if(addresses==null) {
            addresses = new ArrayList<>();
        }
        addresses.add(address);
    }

    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
```

```
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getFull_name() {
        return full_name;
    }
    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public List<Address> getAddresses() {
        return addresses;
    }

    public void setAddresses(List<Address> addresses) {
        this.addresses = addresses;
    }

    public void setFull_name(String full_name) {
        this.full_name = full_name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPhone_no() {
        return phone_no;
    }
    public void setPhone_no(String phone_no) {
        this.phone_no = phone_no;
    }
    public UserRoles getRole() {
        return role;
    }
    public void setRole(UserRoles role) {
        this.role = role;
    }
    public String getActivation_key() {
        return activation_key;
    }
    public void setActivation_key(String activation_key) {
```

```
                this.activation_key = activation_key;
        }
        public UserStatus getStatus() {
                return status;
        }
        public void setStatus(UserStatus status) {
                this.status = status;
        }

        @Override
        public String toString() {
                return "User [id=" + id + ", username=" + username + ",
password=" + password + ", full_name=" + full_name
                                + ", email=" + email + ", phone_no=" + phone_no
+ ", role=" + role + ", activation_key="
                                + activation_key + ", status=" + status + ",
addresses=" + addresses + "]";
        }
}
```

- **ReturnOrder.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import org.hibernate.annotations.CreationTimestamp;
import org.hibernate.annotations.UpdateTimestamp;
import org.springframework.format.annotation.DateTimeFormat;

import com.github.sorabh86.uigo.constants.ReturnStatus;

@Entity
@Table(name = "return_order")
public class ReturnOrder {
        @Id
        @GeneratedValue(strategy = GenerationType.IDENTITY)
        private Integer id;

        @Column(name = "order_id")
        private Integer orderId;

        @Column(name = "return_reason")
```

```java
    private String returnReason;

    @Enumerated(EnumType.STRING)
    private ReturnStatus status;

    @UpdateTimestamp
    @DateTimeFormat
    private Date modify_date;

    @CreationTimestamp
    @DateTimeFormat
    @Column(updatable = false)
    private Date created_date;

    public Integer getId() {
            return id;
    }

    public void setId(Integer id) {
            this.id = id;
    }


    public Integer getOrderId() {
            return orderId;
    }

    public void setOrderId(Integer orderId) {
            this.orderId = orderId;
    }

    public String getReturnReason() {
            return returnReason;
    }

    public void setReturnReason(String return_reason) {
            this.returnReason = return_reason;
    }

    public ReturnStatus getStatus() {
            return status;
    }

    public void setStatus(ReturnStatus status) {
            this.status = status;
    }

    public Date getModify_date() {
            return modify_date;
    }
```

```
	public void setModify_date(Date modify_date) {
		this.modify_date = modify_date;
	}

	public Date getCreated_date() {
		return created_date;
	}

	public void setCreated_date(Date created_date) {
		this.created_date = created_date;
	}

	@Override
	public String toString() {
		return "ReturnOrder [id=" + id + ", orderId=" + orderId + ",
returnReason=" + returnReason + ", status="
				+ status + ", modify_date=" + modify_date + ",
created_date=" + created_date + ", getId()=" + getId()
				+ ", getOrderId()=" + getOrderId() + ",
getReturnReason()=" + getReturnReason() + ", getStatus()="
				+ getStatus() + ", getModify_date()=" +
getModify_date() + ", getCreated_date()=" + getCreated_date()
				+ ", getClass()=" + getClass() + ", hashCode()="
+ hashCode() + ", toString()=" + super.toString()
				+ "]";
	}
}
```

- **Phone.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.Date;
import java.util.Objects;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

import org.hibernate.annotations.CreationTimestamp;
import org.springframework.data.annotation.Transient;

import com.github.sorabh86.uigo.constants.Constants;

@Entity
public class Phone {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;
```

```java
        @Column(length = 255, nullable = false)
        private String title;

        @Column(length = 1024, nullable = false)
        private String description;

        @Column(precision = 2, length = 10, nullable = false)
        private float price;

        @Column(length = 255, nullable = true)
        private String image;

        @CreationTimestamp
        @Column(updatable = false)
        private Date publish_date;


        public Phone() {}

        public Phone(Integer id, String title, String description, float price)
{
                super();
                this.id = id;
                this.title = title;
                this.description = description;
                this.price = price;
        }

        public Integer getId() {
                return id;
        }

        public void setId(Integer id) {
                this.id = id;
        }

        public String getTitle() {
                return title;
        }

        public void setTitle(String title) {
                this.title = title;
        }

        public String getDescription() {
                return description;
        }

        public void setDescription(String description) {
                this.description = description;
        }
```

```
	public float getPrice() {
		return price;
	}

	public void setPrice(float price) {
		this.price = price;
	}

	public String getImage() {
		return image;
	}

	public void setImage(String image) {
		this.image = image;
	}

	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Phone other = (Phone) obj;

		return Objects.equals(id, other.id);
	}

	@Transient
	public String getImagePath() {
		if(id==null || image==null || image.isEmpty())
			return "/images/thumbnail-phone.png";
		return "/"+Constants.PHONE_DIR+this.id+"/"+this.image;
	}

	@Override
	public String toString() {
		return "Phone [id=" + id + ", title=" + title + ", price=" +
price + ", image=" + image + ", publish_date="
				+ publish_date + "]";
	}


}
```

- **Payment.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.Date;
```

```java
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

import org.hibernate.annotations.CreationTimestamp;
import org.hibernate.annotations.UpdateTimestamp;
import org.springframework.format.annotation.DateTimeFormat;

import com.github.sorabh86.uigo.constants.PaymentStatus;

@Entity
public class Payment {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;

	private Integer order_id;

	private float amount;

	@Column(name = "razor_order_id", unique = true)
	private String razorOrderId;


	@Column(name = "razor_payment_id", unique = true)
	private String razorPaymentId;

	@Enumerated(EnumType.STRING)
	@Column(length = 20)
	private PaymentStatus status;

	@UpdateTimestamp
	@DateTimeFormat
	private Date modify_date;

	@CreationTimestamp
	@DateTimeFormat
	@Column(updatable = false)
	private Date created_date;

	public Integer getId() {
		return id;
	}

	public void setId(Integer id) {
		this.id = id;
```

```
    }

    public Integer getOrder_id() {
        return order_id;
    }

    public void setOrder_id(Integer order_id) {
        this.order_id = order_id;
    }

    public float getAmount() {
        return amount;
    }

    public void setAmount(float amount) {
        this.amount = amount;
    }

    public String getRazorOrderId() {
        return razorOrderId;
    }

    public void setRazorOrderId(String razorOrderId) {
        this.razorOrderId = razorOrderId;
    }

    public String getRazorPaymentId() {
        return razorPaymentId;
    }

    public void setRazorPaymentId(String razorPaymentId) {
        this.razorPaymentId = razorPaymentId;
    }

    public PaymentStatus getStatus() {
        return status;
    }

    public void setStatus(PaymentStatus status) {
        this.status = status;
    }

    public Date getModify_date() {
        return modify_date;
    }

    public void setModify_date(Date modify_date) {
        this.modify_date = modify_date;
    }

    public Date getCreated_date() {
```

```
                return created_date;
        }

        public void setCreated_date(Date created_date) {
                this.created_date = created_date;
        }

        @Override
        public String toString() {
                return "Payment [id=" + id + ", order_id=" + order_id + ",
razorOrderId=" + razorOrderId + ", razorPaymentId="
                                + razorPaymentId + ", status=" + status + ",
modify_date=" + modify_date + ", created_date="
                                + created_date + "]";
        }

}
```

- **OrderItem.java**

```
package com.github.sorabh86.uigo.entity;

import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name = "order_item")
public class OrderItem {

        @Id
        @GeneratedValue(strategy = GenerationType.IDENTITY)
        private Integer id;

        @ManyToOne(fetch = FetchType.EAGER)
        private Phone phone;

        private float amount;
        private Integer quantity;

        private Integer order_id;


        public Integer getId() {
                return id;
        }

        public void setId(Integer id) {
```

```
                this.id = id;
        }

        public Phone getPhone() {
                return phone;
        }

        public void setPhone(Phone phone) {
                this.phone = phone;
        }

        public float getAmount() {
                return amount;
        }

        public void setAmount(float amount) {
                this.amount = amount;
        }

        public Integer getQuantity() {
                return quantity;
        }

        public void setQuantity(Integer quantity) {
                this.quantity = quantity;
        }

        public Integer getOrder_id() {
                return order_id;
        }

        public void setOrder_id(Integer order_id) {
                this.order_id = order_id;
        }

        @Override
        public String toString() {
                return "OrderItem [id=" + id + ", phone=" + phone + ", amount="
+ amount + ", quantity=" + quantity
                                + ", order_id=" + order_id + "]";
        }

}
```

- **Order.java**

```
package com.github.sorabh86.uigo.entity;

import java.time.Duration;
import java.time.Instant;
import java.util.ArrayList;
import java.util.Date;
```

```java
import java.util.Iterator;
import java.util.List;

import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
import javax.persistence.Table;

import org.hibernate.annotations.CreationTimestamp;
import org.springframework.data.annotation.Transient;
import org.springframework.format.annotation.DateTimeFormat;

import com.github.sorabh86.uigo.constants.Constants;
import com.github.sorabh86.uigo.constants.OrderStatus;

@Entity
@Table(name = "orders")
public class Order {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;

	private Integer expected_delivery_days;

	@Column(length = 50)
	@Enumerated(EnumType.STRING)
	private OrderStatus status;

	@DateTimeFormat
	private Date delivery_date;

	@CreationTimestamp
	@Column(updatable = false)
	@DateTimeFormat
	private Date created_date;

	@Column(length = 255)
	private String message;
```

```
	@ManyToOne(fetch = FetchType.EAGER)
	private Address address;

	@Column(name = "user_id")
	private Integer userId;

	@Column(length=60, name = "payment_method")
	private String paymentMethod;

	@OneToMany(
			cascade = CascadeType.ALL,
			fetch = FetchType.EAGER
	)
	@JoinColumn(name = "order_id")
	private List<OrderItem> orderItems;

	@OneToMany(
			cascade = CascadeType.ALL,
			fetch = FetchType.LAZY
	)
	@JoinColumn(name = "order_id")
	private List<Payment> payments;

	@OneToMany(
			cascade = CascadeType.ALL,
			fetch = FetchType.LAZY
	)
	@JoinColumn(name = "order_id")
	private List<ReturnOrder> returnOrders;

	private Boolean isVisited = false;

	public List<Payment> getPayments() {
			return payments;
	}

	public void setPayments(List<Payment> payments) {
			this.payments = payments;
	}

	public List<ReturnOrder> getReturnOrders() {
			return returnOrders;
	}

	public void setReturnOrders(List<ReturnOrder> returnOrders) {
			this.returnOrders = returnOrders;
	}

	public void addOrderItem(OrderItem orderItem) {
			if(this.orderItems==null) {
					this.orderItems = new ArrayList<>();
```

```
                }
                this.orderItems.add(orderItem);
        }

        public Integer getId() {
                return id;
        }

        public void setId(Integer id) {
                this.id = id;
        }
        public String getPaymentMethod() {
                return paymentMethod;
        }

        public void setPaymentMethod(String paymentMethod) {
                this.paymentMethod = paymentMethod;
        }

        public Integer getExpected_delivery_days() {
                return expected_delivery_days;
        }

        public void setExpected_delivery_days(Integer expected_delivery_days) {
                this.expected_delivery_days = expected_delivery_days;
        }

        public Address getAddress() {
                return address;
        }

        public void setAddress(Address address) {
                this.address = address;
        }

        public Integer getUserId() {
                return userId;
        }

        public void setUserId(Integer userId) {
                this.userId = userId;
        }

        public OrderStatus getStatus() {
                return status;
        }

        public void setStatus(OrderStatus status) {
                this.status = status;
        }
```

```java
	public Date getDelivery_date() {
		return delivery_date;
	}

	public void setDelivery_date(Date delivery_date) {
		this.delivery_date = delivery_date;
	}

	public Date getCreated_date() {
		return created_date;
	}

	public void setCreated_date(Date created_date) {
		this.created_date = created_date;
	}

		public List<OrderItem> getOrderItems() {
		return orderItems;
	}

	public void setOrderItems(List<OrderItem> orderItems) {
		this.orderItems = orderItems;
	}

	public String getMessage() {
		return message;
	}

	public void setMessage(String message) {
		this.message = message;
	}

	public Boolean getIsVisited() {
		return isVisited;
	}

	public void setIsVisited(Boolean isVisited) {
		this.isVisited = isVisited;
	}

	@Transient
	public float getGrandTotal() {
		float total = 0f;
		for(int i=0; i<orderItems.size(); i++) {
			total += orderItems.get(i).getAmount();
		}
		return total;
	}

	@Transient
	public boolean isReturnable() {
```

```
                if(delivery_date==null || status!=OrderStatus.DELIVERED) return
false;

                // check if you can return this order.
                Instant deliverDate = delivery_date.toInstant();
                Instant returnDate = deliverDate.plus(Duration.ofDays(15));
                Instant todayDate = Instant.now();

                if(todayDate.isBefore(returnDate)) return true;


                return false;
        }

        @Override
        public String toString() {
                return "Order [id=" + id + ", expected_delivery_days=" +
expected_delivery_days + ", status=" + status
                                + ", delivery_date=" + delivery_date + ",
created_date=" + created_date + ", message=" + message
                                + ", address=" + address + ", userId=" + userId
+ ", paymentMethod=" + paymentMethod + ", orderItems="
                                + orderItems + ", payments=" + payments + "]";
        }

}
```

- **FeedbackRating.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.persistence.OneToOne;
import javax.persistence.Table;

import org.hibernate.annotations.CreationTimestamp;
import org.springframework.data.annotation.CreatedDate;
import org.springframework.format.annotation.DateTimeFormat;

import com.github.sorabh86.uigo.constants.PhoneRates;

@Entity
```

```java
@Table(name = "feedback_rating")
public class FeedbackRating {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;

	@ManyToOne(fetch = FetchType.LAZY)
	private Order order;

	@ManyToOne(fetch = FetchType.LAZY)
	private Phone phone;

	@ManyToOne(fetch = FetchType.LAZY)
	private User user;

	private PhoneRates rate;

	private String user_feedback_msg;

	@CreationTimestamp
	@DateTimeFormat
	@Column(updatable = false, nullable = false)
	private Date date;

	@Column(columnDefinition = "boolean default false")
	private Boolean is_visted = false;

	public Integer getId() {
		return id;
	}

	public void setId(Integer id) {
		this.id = id;
	}

	public Order getOrder() {
		return order;
	}

	public void setOrder(Order order) {
		this.order = order;
	}

	public Phone getPhone() {
		return phone;
	}

	public void setPhone(Phone phone) {
		this.phone = phone;
	}
```

```
	public User getUser() {
		return user;
	}

	public void setUser(User user) {
		this.user = user;
	}

	public PhoneRates getRate() {
		return rate;
	}

	public void setRate(PhoneRates rate) {
		this.rate = rate;
	}

	public String getUser_feedback_msg() {
		return user_feedback_msg;
	}

	public void setUser_feedback_msg(String user_feedback_msg) {
		this.user_feedback_msg = user_feedback_msg;
	}

	public Boolean getIs_visted() {
		return is_visted;
	}

	public void setIs_visted(Boolean is_visted) {
		this.is_visted = is_visted;
	}

	public Date getDate() {
		return date;
	}

	public void setDate(Date date) {
		this.date = date;
	}

	@Override
	public String toString() {
		return "FeedbackRating [id=" + id + ", order=" + order + ",
phone=" + phone + ", user=" + user + ", rate="
				+ rate + ", user_feedback_msg=" +
user_feedback_msg + ", is_visted=" + is_visted + "]";
	}
}
```

- **Chat.java**

```
package com.github.sorabh86.uigo.entity;
```

```java
import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

import org.hibernate.annotations.ColumnDefault;
import org.hibernate.annotations.CreationTimestamp;
import org.springframework.format.annotation.DateTimeFormat;

import com.github.sorabh86.uigo.constants.ChatStatus;

@Entity
public class Chat {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private int id;

	private String username;

	private String receiverUsername;

	private String message;

	@CreationTimestamp
	@DateTimeFormat
	private Date date;

	@Column(length = 50)
	@Enumerated(EnumType.STRING)
	private ChatStatus status;

	@ColumnDefault("false")
	public Boolean isVisited = false;

	public Chat() {
	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}

	public String getUsername() {
```

```
                return username;
        }

        public void setUsername(String username) {
                this.username = username;
        }

        public String getReceiverUsername() {
                return receiverUsername;
        }

        public void setReceiverUsername(String receiverUsername) {
                this.receiverUsername = receiverUsername;
        }

        public Date getDate() {
                return date;
        }

        public void setDate(Date date) {
                this.date = date;
        }

        public ChatStatus getStatus() {
                return status;
        }

        public void setStatus(ChatStatus status) {
                this.status = status;
        }

        public String getMessage() {
                return message;
        }
        public void setMessage(String message) {
                this.message = message;
        }

        public Boolean getIsVisited() {
                return isVisited;
        }

        public void setIsVisited(Boolean isVisited) {
                this.isVisited = isVisited;
        }

        @Override
        public String toString() {
                return "Chat [id=" + id + ", username=" + username + ",
receiverUsername=" + receiverUsername + ", message="
                                + message + ", date=" + date + ", status=" +
```

```
status + "]";
	}

}
```

- **CartItem.java**

```
package com.github.sorabh86.uigo.entity;

import java.util.Objects;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name="cart_items")
public class CartItem {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;

	@ManyToOne
	@JoinColumn(name="phone_id")
	private Phone phone;

	@ManyToOne
	@JoinColumn(name="user_id")
	private User user;

	private Integer quantity;

	public Integer getId() {
		return id;
	}

	public void setId(Integer id) {
		this.id = id;
	}

	public Phone getPhone() {
		return phone;
	}

	public void setPhone(Phone phone) {
		this.phone = phone;
	}
```

```
	public User getUser() {
		return user;
	}

	public void setUser(User user) {
		this.user = user;
	}

	public Integer getQuantity() {
		return quantity;
	}

	public void setQuantity(Integer quantity) {
		this.quantity = quantity;
	}

	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		CartItem other = (CartItem) obj;

		return Objects.equals(id, other.id);
	}

	@Override
	public String toString() {
		return "CartItem [id=" + id + ", phone=[id="+(phone!=null?
phone.getId():"NULL")+
				"], user=[id="+(user!=null?user.getId():"NULL")
+"], quantity=" + quantity + "]";
	}


}
```

- **Address.java**

```
package com.github.sorabh86.uigo.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
```

```java
@Entity
public class Address {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer id;

	@Column(length = 50, nullable = false)
	private String full_name;

	@Column(length=16)
	private String phone;

	@Column(length = 255, nullable = false)
	private String address_line_1;

	@Column(length = 255, nullable = false)
	private String address_line_2;

	@Column(length = 50, nullable = false)
	private String city;

	@Column(length = 50, nullable = false)
	private String state;

	@Column(length = 50, nullable = false)
	private String zip;

	@Column(length = 50, nullable = false)
	private String country;

//	@ManyToOne
//	private User user;
	@Column(name = "user_id")
	private Integer userId;

	public Address() {}


	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}

	public String getFull_name() {
		return full_name;
	}

	public String getPhone() {
		return phone;
```

```
        }
        public void setPhone(String phone) {
                this.phone = phone;
        }

        public void setFull_name(String full_name) {
                this.full_name = full_name;
        }

        public String getAddress_line_1() {
                return address_line_1;
        }

        public void setAddress_line_1(String address_line_1) {
                this.address_line_1 = address_line_1;
        }

        public String getAddress_line_2() {
                return address_line_2;
        }

        public void setAddress_line_2(String address_line_2) {
                this.address_line_2 = address_line_2;
        }

        public String getCity() {
                return city;
        }

        public void setCity(String city) {
                this.city = city;
        }

        public String getState() {
                return state;
        }

        public void setState(String state) {
                this.state = state;
        }

        public String getZip() {
                return zip;
        }

        public void setZip(String zip) {
                this.zip = zip;
        }

        public String getCountry() {
                return country;
```

```
	}

	public void setCountry(String country) {
		this.country = country;
	}

	public Integer getUserId() {
		return userId;
	}

	public void setUserId(Integer userId) {
		this.userId = userId;
	}

	@Override
	public String toString() {
		return "Address [id=" + id + ", full_name=" + full_name + ",
address_line_1=" + address_line_1
				+ ", address_line_2=" + address_line_2 + ",
city=" + city + ", state=" + state + ", zip=" + zip
				+ ", country=" + country +
				", userId=" + userId +"]";
	}

}
```

- **EmailService.java**

```
package com.github.sorabh86.uigo.email;

import java.util.List;

import org.springframework.stereotype.Service;

@Service
public interface EmailService {
	void sendTextEmail(String to, String subject, String text);

	void sendEmailWithAttachment(String to, String subject, String text,
String attachment);

	void sendHTMLEmailAttach(String to, String subject, String text,String
attachments);

	void sendHTMLEmail(String to, String subject, String text);
}
```

- **EmailServiceImpl.java**

```
package com.github.sorabh86.uigo.email;

import java.io.File;
import java.nio.file.Files;
```

```java
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;

import javax.mail.MessagingException;
import javax.mail.internet.MimeMessage;
import javax.servlet.ServletContext;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.core.io.ClassPathResource;
import org.springframework.core.io.FileSystemResource;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.mail.javamail.MimeMessageHelper;
import org.springframework.stereotype.Service;

@Service
public class EmailServiceImpl implements EmailService {

	private static Logger logger =
LoggerFactory.getLogger(EmailServiceImpl.class);

	@Autowired
	private JavaMailSender javaMailSender;

	@Autowired
	private ServletContext context;

	@Override
	public void sendTextEmail(String to, String subject, String text) {
		logger.info("Simple Email sending start");

		SimpleMailMessage simpleMessage = new SimpleMailMessage();
		simpleMessage.setTo(to);
		simpleMessage.setSubject(subject);
		simpleMessage.setText(text);

		javaMailSender.send(simpleMessage);

		logger.info("Simple Email sent");
	}

	@Override
	public void sendEmailWithAttachment(String to, String subject, String
text, String imagePath) {
		logger.info("Sending email with attachment start");

		MimeMessage mimeMessage = javaMailSender.createMimeMessage();
```

```
                try {
                        // Set multipart mime message true
                        MimeMessageHelper mimeMessageHelper = new
MimeMessageHelper(mimeMessage, true);

                        mimeMessageHelper.setTo(to);
                        mimeMessageHelper.setSubject(subject);
                        mimeMessageHelper.setText(text);

                        imagePath = context.getRealPath(imagePath);
                        File file = new File(imagePath);
                        String fileName = file.getName();

                        try {
                                mimeMessageHelper.addAttachment(fileName, file);
                                logger.debug("Added a file atachment: {}",
fileName);
                        } catch (MessagingException ex) {
                                logger.error("Failed to add a file atachment:
{}", fileName, ex);
                        }

                        // Adding the attachment
//                      String absPath = context.getRealPath(imagePath);
//            FileSystemResource file = new FileSystemResource(new
File(absPath));
//                      mimeMessageHelper.addAttachment(file.getFilename(),
file);

//                      logger.info(imagePath+" : "+absPath);
                        javaMailSender.send(mimeMessage);

                } catch(MessagingException e) {
                        logger.error("Exception.sendEmailWithAttachment ", e);
                }

                logger.info("Email with attachment sent");
        }

        @Override
        public void sendHTMLEmailAttach(String to, String subject, String text,
String attachments) {
                logger.info("HTML email sending start");

                MimeMessage mimeMessage = javaMailSender.createMimeMessage();

                try {
                        // Set multipart mime message true
                        MimeMessageHelper mimeMessageHelper = new
MimeMessageHelper(mimeMessage, true);
```

```
                        mimeMessageHelper.setTo(to);
                        mimeMessageHelper.setSubject(subject);

                        if(attachments!=null) {
                                String absPath =
context.getRealPath(attachments);

                                mimeMessageHelper.setText(text, true);
                                File file = new File(absPath);
                                mimeMessageHelper.addInline(file.getName(),
file);
                                logger.info("PATH: "+file.getName());
                        }


                        javaMailSender.send(mimeMessage);
                } catch (MessagingException e) {
                        logger.error("Exception.sendHTMLEmail ", e);
                }

                logger.info("HTML email sent");
        }
        @Override
        public void sendHTMLEmail(String to, String subject, String text) {
                logger.info("HTML email sending start");

                MimeMessage mimeMessage = javaMailSender.createMimeMessage();

                try {
                        // Set multipart mime message true
                        MimeMessageHelper mimeMessageHelper = new
MimeMessageHelper(mimeMessage, true);

                        mimeMessageHelper.setTo(to);
                        mimeMessageHelper.setSubject(subject);
                        mimeMessageHelper.setText(text, true);

                        javaMailSender.send(mimeMessage);
                } catch (MessagingException e) {
                        logger.error("Exception.sendHTMLEmail ", e);
                }

                logger.info("HTML email sent");
        }

}
```

- **CustomerController.java**

```
package com.github.sorabh86.uigo.customers;

import java.util.List;
```

```java
import java.util.stream.Collectors;

import org.json.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.annotation.AuthenticationPrincipal;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.github.sorabh86.uigo.admin.address.AddressService;
import com.github.sorabh86.uigo.admin.chats.ChatRepo;
import com.github.sorabh86.uigo.admin.feedbacks.FeedbackService;
import com.github.sorabh86.uigo.admin.orders.OrderService;
import com.github.sorabh86.uigo.admin.returns.ReturnService;
import com.github.sorabh86.uigo.admin.users.UserService;
import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.constants.ChatStatus;
import com.github.sorabh86.uigo.constants.OrderStatus;
import com.github.sorabh86.uigo.constants.PhoneRates;
import com.github.sorabh86.uigo.constants.ReturnStatus;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.Chat;
import com.github.sorabh86.uigo.entity.FeedbackRating;
import com.github.sorabh86.uigo.entity.Order;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.ReturnOrder;
import com.github.sorabh86.uigo.entity.User;

@Controller
@RequestMapping(value = "/customer")
public class CustomerController {

	@Autowired
	private UserService userService;
	@Autowired
	private AddressService addressService;

	@Autowired
	private OrderService orderService;

	@Autowired
	private FeedbackService feedService;
```

```
	@Autowired
	private PasswordEncoder passwordEncoder;

	@Autowired
	private ReturnService returnSerivce;

	@Autowired
	private ChatRepo chatRepo;

	@GetMapping("")
	public String customer() {
		return "redirect:/customer/profile";
	}

	@PostMapping("/save")
	public String save(User user, RedirectAttributes ra) {

		User existingUser =
userService.getUserByUsername(user.getUsername());
		if(existingUser!=null) {
			ra.addFlashAttribute("error", "Username already existed,
please user different username");
			return "redirect:/register";
		}

		user.setUsername(user.getUsername().toLowerCase());
		user.setActivation_key(userService.getActivationKey());
		userService.save(user);

		ra.addFlashAttribute("message", "You are registered succesfully,
please login.");

		return "redirect:/login";
	}

	@GetMapping("/profile")
	public String profile(Model m) {
		UIGOUserDetails userDetails = (UIGOUserDetails)
SecurityContextHolder.getContext().getAuthentication()
				.getPrincipal();

//		m.addAttribute("message", "good job");
		m.addAttribute("customer", userDetails.getUser());
		return "customer/profile";
	}

	@PostMapping("/profile/password")
	public String profilePassword(
			Model m,
			@AuthenticationPrincipal UIGOUserDetails userdetails,
```

```
                        RedirectAttributes redirectAttributes,
                        @RequestParam("curpassword") String curPassword,
                        @RequestParam("newpassword") String newPassword
        ) {
                User user = userdetails.getUser();

                if(!passwordEncoder.matches(curPassword, user.getPassword())) {
                        redirectAttributes.addFlashAttribute("message", "Current
password is invalid!!");
                        return "redirect:/customer/profile";
                }
                System.out.println(curPassword+" "+newPassword);
                System.out.println(user);

                user.setPassword(newPassword);
                userService.save(user);
                redirectAttributes.addFlashAttribute("message", "Your Password
is Updated!!!");
                return "redirect:/customer/profile";
        }

        @GetMapping("/{id}/edit")
        public String edit(@PathVariable("id") String id) {
                return "customer/profile_form";
        }

        @GetMapping("/orders")
        public String ordersList(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {

                User user = userDetails.getUser();
                List<Order> orders =
orderService.getOrdersByUserIdAndStatusNot(user.getId(), OrderStatus.DELIVERED);

                m.addAttribute("orders", orders);
                m.addAttribute("user", user);

                return "customer/orders";
        }

        @GetMapping("/orders/{id}")
        public String orderDetail() {
                return "customer/order_detail";
        }
        @GetMapping("/orders/{id}/delete")
        public String removeOrder(@RequestParam("id")Integer id,
                        RedirectAttributes ra) {
                ra.addFlashAttribute("message", "Your order("+id+") has been
removed!!!");
                return "redirect:/customer/orders";
        }
```

```
        @GetMapping("/invoices")
        public String downloadOrderInvoice(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {
                User user = userDetails.getUser();
                List<Order> orders =
orderService.getOrdersByUserIdAndStatus(user.getId(), OrderStatus.DELIVERED);

                m.addAttribute("orders", orders);

                return "customer/invoices";
        }

        @GetMapping("/return")
        public String returnOrderItem(
                        Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {

                User user = userDetails.getUser();
                List<Order> orders =
orderService.getOrdersByUserIdAndStatus(user.getId(), OrderStatus.DELIVERED);

                m.addAttribute("orders", orders);
                return "customer/returnorder";
        }

        @PostMapping("/return/save")
        public String returnOrderSave(Model m,
                        RedirectAttributes ra,
                        @RequestParam("returnReason")String returnReason,
                        @RequestParam("status")ReturnStatus status,
                        @RequestParam("orderId")Integer orderId
                ) {

                ReturnOrder returnOrder = new ReturnOrder();
                returnOrder.setReturnReason(returnReason);
                returnOrder.setStatus(status);
                returnOrder.setOrderId(orderId);

                returnSerivce.save(returnOrder);
                ra.addFlashAttribute("message","Your request submitted!!!");
                return "redirect:/customer/return";
        }

        @GetMapping("/address")
        public String addressList(
                        @AuthenticationPrincipal UIGOUserDetails userDetails,
                        Model m) {

                List<Address> addresses =
addressService.getAddressByUserId(userDetails.getUser().getId());
```

```
                System.out.println(addresses);
                m.addAttribute("addresses", addresses);

                return "customer/address";
        }

        @GetMapping("/address/new")
        public String addAddress(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {
                User user = userDetails.getUser();

                Address address = new Address();
                address.setFull_name(user.getFull_name());
                address.setUserId(user.getId());

                m.addAttribute("address", address);
                m.addAttribute("page_title", "Admin | Add New Address");

                return "customer/address_form";
        }

        @PostMapping("/address/save")
        public String saveAddress(
                        Address address,
                        RedirectAttributes redirectAttribute
                        ) {

                addressService.save(address);
                redirectAttribute.addFlashAttribute("message", "Your Address has
been saved");

                return "redirect:/customer/address";
        }

        @GetMapping("/address/{id}/edit")
        public String editAddress(Model m,
                        @PathVariable("id")int id) {

                Address address = addressService.getAddress(id);
                m.addAttribute("address", address);
                System.out.println(address);

                m.addAttribute("page_title", "Admin | Edit Address ("+id+")");

                return "customer/address_form";
        }

        @GetMapping("/address/{id}/delete")
        public String deleteAddress() {
                return "customer/address";
        }
```

```
        @GetMapping("/feedback")
        public String feedback(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {
                User user = userDetails.getUser();
                List<Order> orders =
orderService.getOrdersByUserIdAndStatus(user.getId(), OrderStatus.DELIVERED);

                m.addAttribute("orders", orders);

                return "customer/feedback";
        }

        @PostMapping("/feedback/save")
        public String saveFeedback(
                @AuthenticationPrincipal UIGOUserDetails userDetails,
                @RequestParam(value = "id", required = false)Integer id,
                @RequestParam("phone_id")Integer phone_id,
                @RequestParam("order_id")Integer order_id,
                @RequestParam("rate") Integer rate,
                @RequestParam("feedback_msg")String feedback_msg,
                RedirectAttributes redirectAttributes
        ) {

                User user = userDetails.getUser();
                Order order = orderService.getOrderById(order_id);
                Phone phone = order.getOrderItems().stream().filter(orderitem->

orderitem.getPhone().getId()==phone_id).collect(Collectors.toList()).get(0)
                                .getPhone();

                FeedbackRating feed1;

                if(id!=null) {
                        feed1 = feedService.getFeedback(id);
                } else {
                 feed1 = new FeedbackRating();
                }
                switch (rate) {
                        case 1: feed1.setRate(PhoneRates.One); break;
                        case 2: feed1.setRate(PhoneRates.Two); break;
                        case 3: feed1.setRate(PhoneRates.Three); break;
                        case 4: feed1.setRate(PhoneRates.Four); break;
                        case 5: feed1.setRate(PhoneRates.Five);
                };
                feed1.setOrder(order);
                feed1.setPhone(phone);
                feed1.setUser(user);
                feed1.setUser_feedback_msg(feedback_msg);

                feedService.save(feed1);
```

```
                redirectAttributes.addFlashAttribute("message", "Your feedback
submitted");

                return "redirect:/customer/feedback";
        }
        @PostMapping("/feedback/{pid}/{oid}")
        @ResponseBody
        public String getFeedbackOfUserAndPhone(
                        @AuthenticationPrincipal UIGOUserDetails ud,
                        @PathVariable("pid")Integer phone_id,
                        @PathVariable("oid")Integer order_id
        ) {
                User user = ud.getUser();
//              Phone phone = phoneService.getPhone(phone_id);
//              FeedbackRating feed = feedService.getFeebackByUserAndPhone(user,
phone);
                FeedbackRating feed =
feedService.getFeedbackByUidPidOid(user.getId(), phone_id, order_id);

                var obj = new JSONObject();
                obj.put("status", false);
                obj.put("message", "No Feedback found on this phone");

                if(feed!= null) {
                        obj.put("id", feed.getId());
                        obj.put("msg", feed.getUser_feedback_msg());
                        switch(feed.getRate()) {
                                case One: obj.put("rate", 1); break;
                                case Two: obj.put("rate", 2); break;
                                case Three: obj.put("rate", 3); break;
                                case Four: obj.put("rate", 4); break;
                                case Five: obj.put("rate", 5); break;
                        }
                        obj.put("status", true);
                        obj.put("message", "User already submitted Feedback on
this phone");
                }
                return obj.toString();
        }

        @GetMapping("/chats")
        public String chat(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails) {
                User user = userDetails.getUser();
                List<User> users =
userService.getUsersByRole(UserRoles.ROLE_ADMIN);
                List<String> allUsers = users.stream().map(u ->
u.getUsername()).collect(Collectors.toList());
                System.out.println(allUsers);
                List<String> onlineUsers = chatRepo.findOnlineUsers();
```

```
                List<Chat> publicMessages =
chatRepo.findByStatusAndReceiverUsername(ChatStatus.MESSAGE, null);

                m.addAttribute("user", user);
                m.addAttribute("allUsers", allUsers);
                m.addAttribute("onlineUsers", onlineUsers);
                m.addAttribute("publicMessages", publicMessages);

                return "customer/chats";
        }


}
```

- **UserStatus.java**

```
package com.github.sorabh86.uigo.constants;

public enum UserStatus {

        DEACTIVATE(0),
        ACTIVATED(1),
        DISABLED(2);

        private final int value;

        UserStatus(final int newValue) {
                value = newValue;
        }
        public int getValue() {return value;}
}
```

- **UserRoles.java**

```
package com.github.sorabh86.uigo.constants;

public enum UserRoles {
        ROLE_ADMIN, ROLE_CUSTOMER, ROLE_SUBSCRIBER
}
```

- **ReturnStatus.java**

```
package com.github.sorabh86.uigo.constants;

public enum ReturnStatus {
        REQUESTED, APPROVED, RESOLVED, RETURNED;
}
```

- **PhoneRates.java**

```
package com.github.sorabh86.uigo.constants;

public enum PhoneRates {
        One(1),
```

```
        Two(2),
        Three(3),
        Four(4),
        Five(5);

        private final int value;

        PhoneRates(final int newValue) {
                value = newValue;
        }
        public int getValue() {return value;}
}
```

- **PaymentStatus.java**

```
package com.github.sorabh86.uigo.constants;

public enum PaymentStatus {
        PENDING, FAILED, PAID
}
```

- **OrderStatus.java**

```
package com.github.sorabh86.uigo.constants;

public enum OrderStatus {
        REQUESTED, APPROVED, SHIPPED, DELIVERED, REJECTED;
}
```

- **Constants.java**

```
package com.github.sorabh86.uigo.constants;

public class Constants {

        public static String UPLOAD_DIR = "upload";
        public static String PHONE_DIR = Constants.UPLOAD_DIR+"/phones/";
}
```

- **ChatStatus.java**

```
package com.github.sorabh86.uigo.constants;

public enum ChatStatus {
        JOIN, MESSAGE, LEAVE
}
```

- **WebSocketEventListener.java**

```
package com.github.sorabh86.uigo.config;

import java.security.Principal;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
```

```java
import javax.persistence.TransactionRequiredException;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.event.EventListener;
import org.springframework.messaging.simp.SimpMessageSendingOperations;
import org.springframework.messaging.simp.stomp.StompHeaderAccessor;
import org.springframework.security.core.annotation.AuthenticationPrincipal;
import org.springframework.stereotype.Component;
import org.springframework.web.socket.messaging.SessionConnectedEvent;
import org.springframework.web.socket.messaging.SessionDisconnectEvent;

import com.github.sorabh86.uigo.admin.chats.ChatRepo;
import com.github.sorabh86.uigo.constants.ChatStatus;
import com.github.sorabh86.uigo.entity.Chat;

@Component
public class WebSocketEventListener {
	private static final Logger logger =
LoggerFactory.getLogger(WebSocketEventListener.class);

	private List<Chat> onlineUsers = new ArrayList<>();

	@Autowired
	private ChatRepo chatRepo;

	@Autowired
	private SimpMessageSendingOperations messagingTemplate;

	@EventListener
	public void handleWebSocketConnectListener(SessionConnectedEvent event)
{
		logger.info("Received a new web socket connection");
	}

	@EventListener
	public void handleWebSocketDisconnectListener(SessionDisconnectEvent
event)
			throws InterruptedException,
TransactionRequiredException {
		StompHeaderAccessor headerAccessor =
StompHeaderAccessor.wrap(event.getMessage());

		Chat chat = (Chat)
headerAccessor.getSessionAttributes().get("chat");
		if (chat != null) {
			logger.info("User Disconnected : " + chat);

			chat.setStatus(ChatStatus.LEAVE);
```

```
chatRepo.deleteAllByUsernameAndStatus(chat.getUsername(), ChatStatus.JOIN);

                        Thread.sleep(100);

                        messagingTemplate.convertAndSend("/chatroom/public",
chat);
                }
        }
}
```

- **UIGOWebSocketConfig.java**

```
package com.github.sorabh86.uigo.config;

import org.springframework.context.annotation.Configuration;
import org.springframework.messaging.simp.config.MessageBrokerRegistry;
import
org.springframework.web.socket.config.annotation.EnableWebSocketMessageBroker;
import org.springframework.web.socket.config.annotation.StompEndpointRegistry;
import
org.springframework.web.socket.config.annotation.WebSocketMessageBrokerConfigure
r;

@Configuration
@EnableWebSocketMessageBroker
public class UIGOWebSocketConfig implements WebSocketMessageBrokerConfigurer {

        @Override
        public void configureMessageBroker(MessageBrokerRegistry registry) {
                registry.setApplicationDestinationPrefixes("/uigo");
                registry.enableSimpleBroker("/chatroom", "/user");
                registry.setUserDestinationPrefix("/user");
        }

        @Override
        public void registerStompEndpoints(StompEndpointRegistry registry) {

registry.addEndpoint("/ws").setAllowedOriginPatterns("*").withSockJS();
        }

}
```

- **UIGOWebSecurityConfig.java**

```
package com.github.sorabh86.uigo.config;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.builders.Authentic
```

```
ationManagerBuilder;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.builders.WebSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecuri
ty;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConf
igurerAdapter;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;

import com.github.sorabh86.uigo.admin.users.MyLoginSuccessHandler;
import com.github.sorabh86.uigo.admin.users.MyLogoutSuccessHandler;

@Configuration
@EnableWebSecurity
public class UIGOWebSecurityConfig extends WebSecurityConfigurerAdapter {

        @Bean
        public PasswordEncoder passwordEncoder() {
                return new BCryptPasswordEncoder();
        }

        @Autowired
        private UIGOUserDetailsService uigoUserDetailsService;

        @Autowired
        private MyLoginSuccessHandler myLoginSuccessHandler;
        @Autowired
        private MyLogoutSuccessHandler myLogoutSuccessHandler;

        private String[] PUBLIC_URLS = {
                        "/",
                        "/razorpay",
                        "/register",
                        "/about",
                        "/phones/**",
                        "/contact",
                        "/customer/save",
                        "/verification/**"
        };
        private String[] CUSTOMER_URLS = {
                        "/cart/**",
                        "/customer/**",
                        "/addtocart",
                        "/checkout",
                        "/removetocart/**",
                        "/orderstatus/**"
        };
        private String[] RESOURCE_URLS= {
```

```
                "/images/**",
                "/bootstrap/**",
                "/css/**",
                "/js/**",
                "/upload/**"
        };

        @Override
        protected void configure(HttpSecurity http) throws Exception {
//              http.authorizeHttpRequests()
//                      .antMatchers(RESOURCE_URLS).permitAll();

                http
                        .csrf()
                                .ignoringAntMatchers(
                                        "/cart/order",
                                        "/cart/payment",
                                        "/cart/payment/update"
//                                      "/customer/feedback/save"
                                )
//                              .disable()
                        .and()
                        .authorizeRequests()
//                      .anyRequest().permitAll();
                        .antMatchers(PUBLIC_URLS).permitAll()
                        .antMatchers(RESOURCE_URLS).permitAll()
                        .antMatchers
("/admin/chats/messages/**").hasAnyRole("CUSTOMER","ADMIN")
                        .antMatchers(CUSTOMER_URLS).hasAnyRole("CUSTOMER")
                        .antMatchers("/admin/**").hasAnyRole("ADMIN")
                        .anyRequest().authenticated()
                        .and()
                                .formLogin()
                                .loginPage("/login")
//                              .loginProcessingUrl("/login")
//                              .defaultSuccessUrl("/customer")
                                .successHandler(myLoginSuccessHandler)
                                .permitAll()
                        .and()
                                .logout()
                                .logoutSuccessHandler(myLogoutSuccessHandler)
                                .permitAll();
        }

        @Override
        protected void configure(AuthenticationManagerBuilder auth) throws
Exception {
                auth.userDetailsService(uigoUserDetailsService);
        }

        @Override
```

```
        public void configure(WebSecurity web) throws Exception {
                web.ignoring()
                        .antMatchers("images/**", "/css/**", "/js/**",
"/webfonts/**", "/upload/**");
        }

}
```

- **UIGOWebMvcConfig.java**

```
package com.github.sorabh86.uigo.config;

import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.concurrent.TimeUnit;

import org.springframework.context.annotation.Configuration;
import org.springframework.http.CacheControl;
import
org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
import com.github.sorabh86.uigo.constants.Constants;

@Configuration
public class UIGOWebMvcConfig implements WebMvcConfigurer {

        @Override
        public void addResourceHandlers(ResourceHandlerRegistry registry) {
                Path uploadPath = Paths.get(Constants.UPLOAD_DIR);
                String absolutePath = uploadPath.toFile().getAbsolutePath();

                System.out.println("upload directory: "+Constants.UPLOAD_DIR+"
"+absolutePath);

                registry.addResourceHandler("/"+Constants.UPLOAD_DIR+"/**")
                        .addResourceLocations("file:/"+absolutePath+"/")
                        .setCacheControl(CacheControl.maxAge(1,
TimeUnit.HOURS).cachePublic());
        }

}
```

- **UIGOUserDetailsService.java**

```
package com.github.sorabh86.uigo.config;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.admin.users.UserRepo;
```

```
import com.github.sorabh86.uigo.entity.User;

@Service
public class UIGOUserDetailsService implements UserDetailsService {

	@Autowired
	private UserRepo userRepo;

	@Override
	public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
		User user = userRepo.findByUsername(username);

		if(user == null)
			throw new UsernameNotFoundException("Username:
"+username+" not found");

		return new UIGOUserDetails(user);
	}

}
```

- **UIGOUserDetails.java**

```
package com.github.sorabh86.uigo.config;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.User;

public class UIGOUserDetails implements UserDetails {

	private User user;

	public UIGOUserDetails(User user) {
		this.user = user;
	}

	@Override
	public Collection<? extends GrantedAuthority> getAuthorities() {
		List<SimpleGrantedAuthority> authorities = new ArrayList<>();
		authorities.add(new
SimpleGrantedAuthority(user.getRole().toString()));
		return authorities;
	}
```

```
	public User getUser() {
		return user;
	}

	public void setUser(User user) {
		this.user = user;
	}

	@Override
	public String getPassword() {
		return this.user.getPassword();
	}

	@Override
	public String getUsername() {
		return this.user.getUsername();
	}

	@Override
	public boolean isAccountNonExpired() {
		return true;
	}

	@Override
	public boolean isAccountNonLocked() {
		return true;
	}

	@Override
	public boolean isCredentialsNonExpired() {
		return true;
	}

	@Override
	public boolean isEnabled() {
		return (user.getStatus() != UserStatus.DISABLED);
	}

}
```

- **CartItemsRepo.java**

```
package com.github.sorabh86.uigo.cart;

import java.util.List;

import javax.transaction.Transactional;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
```

```
import com.github.sorabh86.uigo.entity.CartItem;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;

@Repository
public interface CartItemsRepo extends JpaRepository<CartItem, Integer> {

	public List<CartItem> findByUser(User user);

	public CartItem findByUserAndPhone(User user, Phone phone);

	@Transactional
	public void deleteByPhone(Phone phone);

	@Transactional
	public void deleteByUser(User user);
}
```

- **CartItemService.java**

```
package com.github.sorabh86.uigo.cart;

import java.util.ArrayList;
import java.util.List;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.admin.phones.PhoneRepo;
import com.github.sorabh86.uigo.entity.CartItem;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;

@Service
public class CartItemService {

	@Autowired
	private CartItemsRepo cartRepo;

	@Autowired
	private PhoneRepo phoneRepo;

	public List<CartItem> getSessionCartItems(HttpSession session) {
		if(session!=null) {
			List<CartItem> shoppingCart = (List<CartItem>)
session.getAttribute("cart");

			if(shoppingCart==null) {
				shoppingCart = new ArrayList<CartItem>();
				session.setAttribute("cart", shoppingCart);
```

```
                }
                return shoppingCart;
            }
            return null;
        }

        public List<CartItem> getCartItemsByUser(User user){
            return cartRepo.findByUser(user);
        }

        public void save(CartItem cartItem) {
            cartRepo.save(cartItem);
        }

        public void saveAll(List<CartItem> cartItems) {
            cartRepo.saveAll(cartItems);
        }

        public void emptyCart(User user) {
            cartRepo.deleteByUser(user);
        }

        public CartItem getCartItemById(Integer cartItemId) {
            return cartRepo.findById(cartItemId).get();
        }

        public Integer addPhone(Integer phone_id, Integer quantity, User user) {
            Integer addedQuantity = quantity;

            Phone phone = phoneRepo.findById(phone_id).get();
            CartItem cartItem = cartRepo.findByUserAndPhone(user, phone);

            if(cartItem != null) {
                addedQuantity = cartItem.getQuantity() + quantity;
                cartItem.setQuantity(addedQuantity);
            } else {
                cartItem = new CartItem();
                cartItem.setQuantity(quantity);
                cartItem.setPhone(phone);
                cartItem.setUser(user);
            }

            cartRepo.save(cartItem);

            return addedQuantity;
        }
        public void removePhone(Integer phone_id) {
            Phone phone = phoneRepo.findById(phone_id).get();
            cartRepo.deleteByPhone(phone);
        }
```

```
        public void addToCart(HttpSession session, CartItem item) {
                List<CartItem> shoppingCart = getSessionCartItems(session);

                if(shoppingCart!=null) {

                        Boolean notExist = true;
                        //check if Item already exist then increment Quantity
                        for(int i=0; i< shoppingCart.size(); i++) {
                                CartItem citem = shoppingCart.get(i);
                                System.out.println(citem.equals(item)+" "+citem
+""+ item);
                                if(citem.getPhone().equals(item.getPhone())) {
                                        citem.setQuantity(citem.getQuantity()
+1);
                                        notExist = false;
                                }
                        }
                        if(notExist)
                                shoppingCart.add(item);
                } else {
                        System.out.println("Unable to find session");
                }
        }
}
```

- **CartController.java**

```
package com.github.sorabh86.uigo.cart;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

import javax.servlet.http.HttpServletRequest;

import org.json.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.security.core.annotation.AuthenticationPrincipal;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.github.sorabh86.uigo.admin.address.AddressService;
import com.github.sorabh86.uigo.admin.orders.OrderService;
```

```java
import com.github.sorabh86.uigo.admin.payments.PaymentService;
import com.github.sorabh86.uigo.admin.phones.PhoneService;
import com.github.sorabh86.uigo.admin.users.UserService;
import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.constants.OrderStatus;
import com.github.sorabh86.uigo.constants.PaymentStatus;
import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.CartItem;
import com.github.sorabh86.uigo.entity.Order;
import com.github.sorabh86.uigo.entity.OrderItem;
import com.github.sorabh86.uigo.entity.Payment;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;
import com.razorpay.RazorpayException;

@Controller
public class CartController {

	@Autowired
	private PhoneService phoneService;

	@Autowired
	private UserService userService;

	@Autowired
	private CartItemService cartService;

	@Autowired
	private AddressService addressService;

	@Autowired
	private OrderService orderService;

	@Autowired
	private PaymentService payService;

	@PostMapping("/addtocart")
	public String addToCart(
				@RequestParam("redirect") String redirectUrl,
				@RequestParam("phone_id")Integer phone_id,
				@RequestParam("quantity")Integer quantity,
				@AuthenticationPrincipal UIGOUserDetails userDetails,
				RedirectAttributes redirectAttributes,
				HttpServletRequest request
	) {

		User user = null;
		if(userDetails!=null) {
			String username = userDetails.getUsername();
			user = userService.getUserByUsername(username);
			cartService.addPhone(phone_id, quantity, user);
```

```
                } else {
                        Phone phone = this.phoneService.getPhone(phone_id);
                        CartItem newItem = new CartItem();
                        newItem.setUser(user);
                        newItem.setPhone(phone);
                        newItem.setQuantity(quantity);

                        cartService.addToCart(request.getSession(), newItem);
                }
                redirectAttributes.addFlashAttribute("message", "Item added
successfully, go to <a href=\"/cart\">cart</a>.");

                return "redirect:"+redirectUrl;
        }
        @GetMapping("/removetocart/{pid}")
        public String removeToCart(
                        @PathVariable("pid")Integer phone_id,
                        @AuthenticationPrincipal UIGOUserDetails userDetails,
                        HttpServletRequest request) {

                if(userDetails==null) {

                } else {
                        cartService.removePhone(phone_id);
                }

                return "redirect:/cart";
        }

        @GetMapping("/cart")
        public String cartPage(Model m,
                        @AuthenticationPrincipal UIGOUserDetails userDetails,
                        HttpServletRequest request) {
                List<CartItem> cartItems;
                List<Address> addresses;

                if(userDetails!=null) {
                        String username = userDetails.getUsername();
                        User user = userService.getUserByUsername(username);
                        cartItems = cartService.getCartItemsByUser(user);
                        addresses = user.getAddresses();
                } else {
                        cartItems =
cartService.getSessionCartItems(request.getSession());
                        addresses = new ArrayList<>();
                }

                float totalPrice = 0f;
                for(int i=0; i<cartItems.size(); i++) {
                        CartItem item = cartItems.get(i);
                        totalPrice +=
```

```
(item.getPhone().getPrice()*item.getQuantity());
                }

                m.addAttribute("page", "cart");
                m.addAttribute("cartItems", cartItems);
                m.addAttribute("totalPrice", Math.round(totalPrice));
                m.addAttribute("addresses",addresses);

                return "cart";
        }

        @PostMapping("/cart/order")
        @ResponseBody /* This will change behavior of method to return string
not matched template  */
        public String createOrder(
                @AuthenticationPrincipal UIGOUserDetails userDetails,
                @RequestBody Map<String, Object> data
        ) throws RazorpayException {

                User user = userDetails.getUser();
                String paymentMethod = (String)data.get("payment-method");
                Integer address_id =
Integer.parseInt(data.get("address_id").toString());
                String message = (String)data.get("message");
                Address address = addressService.getAddress(address_id);

                // check for address
                if(address_id==0) {
                        address.setFull_name((String)data.get("full_name"));

address.setAddress_line_1((String)data.get("address_line_1"));

address.setAddress_line_2((String)data.get("address_line_2"));
                        address.setPhone((String)data.get("phone"));
                        address.setCity((String)data.get("city"));
                        address.setState((String)data.get("state"));
                        address.setCountry((String)data.get("country"));
                        address.setZip((String)data.get("zip"));
                        address.setUserId(userDetails.getUser().getId());

                        address = addressService.save(address);
                }

                // Generate Order receipt
                Order newOrder = new Order();
                newOrder.setStatus(OrderStatus.REQUESTED);
                newOrder.setPaymentMethod(paymentMethod);
                newOrder.setAddress(address);
                newOrder.setMessage(message);

                newOrder.setOrderItems(this.getOrderItemsFromRequest(data));
```

```
                Order savedOrder = orderService.save(newOrder);

                // Empty cart saved to database
                cartService.emptyCart(user);

                return razorPaymentIntegration(newOrder.getGrandTotal(),
savedOrder.getId(), user.getEmail(), address);
        }

        @PostMapping("/cart/payment")
        @ResponseBody
        private String createRazorOrderAndPayment(@RequestParam("id")Integer
orderId,
                        @RequestParam("amount") float amount,
                        @AuthenticationPrincipal UIGOUserDetails userDetails)
throws RazorpayException {
                User user = userDetails.getUser();
                Order order = orderService.getOrderById(orderId);

                return razorPaymentIntegration(amount, orderId, user.getEmail(),
order.getAddress());
        }

        // razorpay payment integration
        private String razorPaymentIntegration(float amount, Integer order_id,
String email, Address address) throws RazorpayException {

                JSONObject ob = new JSONObject();
                ob.put("amount", amount*100);
                ob.put("currency", "INR");
                ob.put("receipt", "txn_"+order_id);

                // creating new order on razorpay server
                com.razorpay.Order order = payService.createRazorOrder(ob);
                System.out.println("RAZOR_ORDER: "+order);

                // SAVE CREATED ORDER TO LOCAL DATABASE
                Payment payment = new Payment();
                payment.setOrder_id(order_id);
                payment.setRazorOrderId(order.get("id"));
                payment.setStatus(PaymentStatus.PENDING);
                payment.setAmount(amount);
                payService.savePayment(payment);

                JSONObject returnObj = new JSONObject();
                returnObj.put("username", address.getFull_name());
                returnObj.put("email", email);
                returnObj.put("phone", address.getPhone());
                returnObj.put("id", order.get("id").toString());
                returnObj.put("amount",
Float.parseFloat(order.get("amount").toString()));
```

```
                return returnObj.toString();
        }

        @PostMapping("/cart/payment/update")
        @ResponseBody
        public String updatePaymentStatus(
                        @RequestParam("razorOrderId") String razorOrderId,
                        @RequestParam(name = "razorPaymentId", required=false)
String razorPaymentId,
                        @RequestParam("status") PaymentStatus status,
                        @AuthenticationPrincipal UIGOUserDetails userDetails
        ) {
                JSONObject json = new JSONObject();

                if(razorPaymentId!=null && !razorPaymentId.isEmpty()) {
                        Payment payment =
payService.getPaymentByRazorOrderId(razorOrderId);
                        if(payment!=null) {
                                json.put("paymentId", payment.getId());
                                json.put("orderId", payment.getOrder_id());
                                payment.setStatus(status);
                                if(razorPaymentId!=null)

payment.setRazorPaymentId(razorPaymentId);
                                payService.savePayment(payment);
                        }
                }

                json.put("status", status.name());
                json.put("razorOrderId", razorOrderId);

                return json.toString();
        }

        @PostMapping("/cart/update")
        public ResponseEntity create(@RequestParam("item_id")Integer item_id,
                        @RequestParam("quantity")Integer quantity) {
                CartItem cartItem = cartService.getCartItemById(item_id);
                cartItem.setQuantity(quantity);
                cartService.save(cartItem);
                return ResponseEntity.ok("OK");
        }

        @PostMapping("/checkout")
        public String checkout(
                        @AuthenticationPrincipal UIGOUserDetails userDetails,
                        @RequestParam Map<String, Object> data,
                        RedirectAttributes redirectAttributes
        ) {
                User user = userDetails.getUser();
```

```
                String paymentMethod = data.get("payment-method").toString();
                Integer address_id =
Integer.parseInt(data.get("address_id").toString());
                String message = data.get("message").toString();
                Address address = addressService.getAddress(address_id);

                if(address_id==0) {

address.setFull_name((String)data.get("full_name").toString());

address.setAddress_line_1((String)data.get("address_line_1").toString());

address.setAddress_line_2((String)data.get("address_line_2").toString());
                        address.setPhone((String)data.get("phone").toString());
                        address.setCity((String)data.get("city").toString());
                        address.setState((String)data.get("state").toString());

address.setCountry((String)data.get("country").toString());
                        address.setZip((String)data.get("zip").toString());

                        address.setUserId(userDetails.getUser().getId());

                        address = addressService.save(address);
                }

                Order newOrder = new Order();
                newOrder.setStatus(OrderStatus.REQUESTED);
                newOrder.setPaymentMethod(paymentMethod);
                newOrder.setAddress(address);
                newOrder.setMessage(message);

                // GET CARTITEMS
                List<OrderItem> orderItems =
this.getOrderItemsFromRequest(data);
                newOrder.setOrderItems(orderItems);

                System.out.println(newOrder);

                Order savedOrder = orderService.save(newOrder);

                // EMPTY CART
                cartService.emptyCart(user);

                redirectAttributes.addFlashAttribute("message", "Order Receipt
is Created!!");

                // check for payment options
                if(paymentMethod=="COD") {
                        // TODO: generate order straight;

                } else if(paymentMethod=="OP") {
```

```
                        // TODO: implement online payment system like payTM,
payu, etc.
                }

                return "redirect:/orderstatus/"+savedOrder.getId();
        }

        private List<OrderItem> getOrderItemsFromRequest(Map<String, Object>
data) {
                List<OrderItem> itemList = new ArrayList<>();
                int count = 1;
                while(data.get("cartItems["+count+"].item_id")!=null) {
                        int cart_item_id =
Integer.parseInt(data.get("cartItems["+count+"].item_id").toString());
                        int phone_id =
Integer.parseInt(data.get("cartItems["+count+"].phone_id").toString());
                        int quantity =
Integer.parseInt(data.get("cartItems["+count+"].quantity").toString());
                        float amount =
Float.parseFloat(data.get("cartItems["+count+"].amount").toString());

                        OrderItem item = new OrderItem();
                        item.setPhone(phoneService.getPhone(phone_id));
                        item.setAmount(amount);
                        item.setQuantity(quantity);
                        itemList.add(item);

                        count++;
                }
                return itemList;
        }

        @GetMapping(value={"/orderstatus/{order_id}","/orderstatus/{order_id}/
{payment_id}"})
        public String orderStatus(
                        Model m,
                        @PathVariable("order_id")Integer order_id,
                        @PathVariable(name = "payment_id", required =
false)Integer payment_id,
                        @ModelAttribute("message")String message) {

                Order order = orderService.getOrderById(order_id);
                PaymentStatus payStatus = PaymentStatus.PENDING;
                List<OrderItem> orderItems = order.getOrderItems();
                float totalPrice = 0f;
                for(int i=0; i<orderItems.size(); i++) {
                        OrderItem item = orderItems.get(i);
                        totalPrice += (item.getAmount()*item.getQuantity());
                }

                if(payment_id!=null) {
```

```
                        Payment payment = payService.getPayment(payment_id);
                        System.out.println(payment);
                        if(order.getId()==payment.getOrder_id() &&
payment.getId()!=null) {
                                payStatus = payment.getStatus();
                                m.addAttribute("payment", payment);
                        }
                }

                m.addAttribute("message", message);
                m.addAttribute("payStatus", payStatus);
                m.addAttribute("order", order);
                m.addAttribute("totalPrice", totalPrice);

                return "orderstatus";
        }


}
```

- **AdminController.java**

```
package com.github.sorabh86.uigo.admin;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import com.github.sorabh86.uigo.admin.orders.OrderService;
import com.github.sorabh86.uigo.admin.users.MyHttpSessionBindingListener;
import com.github.sorabh86.uigo.entity.User;

@Controller
@RequestMapping("/admin")
public class AdminController {

        @Autowired
        private OrderService orderService;

        @GetMapping("")
        public String dashboard(Model m, HttpSession session) {

                MyHttpSessionBindingListener listener =

(MyHttpSessionBindingListener)session.getAttribute("user");
```

```
                m.addAttribute("newOrderCount",
orderService.getNewOrderCount());
                m.addAttribute("totalOrders", orderService.getAllOrderCount());

                m.addAttribute("loggedInUsers",
listener.getLoggedInUsers().getUsers().size());

                return "admin/dashboard";
        }
}
```

- **UserService.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.text.DateFormat;
import java.util.Date;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.email.EmailService;
import com.github.sorabh86.uigo.entity.User;

import ch.qos.logback.classic.Logger;

@Service
public class UserService {
        @Autowired
        private UserRepo userRepo;

        @Autowired
        private PasswordEncoder passwordEncoder;

        @Autowired
        private EmailService emailService;

        public void save(User user) {

                if(user.getId()!=null) {
                        User dUser = userRepo.findById(user.getId()).get();

                        if(dUser!=null) {
                                if(user.getPassword().isEmpty()) {
                                        user.setPassword(dUser.getPassword());
                                } else {
```

```
                                        user.setPassword(
passwordEncoder.encode(user.getPassword())
                                        );
                                    }
                                }
                                userRepo.save(user);
                            } else {

                                String activeKey = getActivationKey();
                                user.setActivation_key(activeKey);

user.setPassword(passwordEncoder.encode(user.getPassword()));
                                User savedUser = userRepo.save(user);

                                if(user.getRole()==UserRoles.ROLE_CUSTOMER ||
user.getRole()==UserRoles.ROLE_SUBSCRIBER) {

                                    try {
                                        if(savedUser.getEmail()!=null && !
savedUser.getEmail().isEmpty()) {
                                            // sending email verification
link
                                            emailService.sendHTMLEmail(
savedUser.getEmail(),
                                                    "User Account
Verification",
                                                    "<h1>Your
Request is submitted</h1>"+
"<p>Please Verify your account by clicking the <a
href=\"http://localhost:8080/verification/"
+savedUser.getId()+"/"+activeKey+"\">link</a><br>"
+ "If above link didn't worked try copy pasting [
http://localhost:8080/verification/"
+savedUser.getId()+"/"+activeKey+" ]"
+ "<h4>Thanks & Regard <br> UIGO SHOPPING SYSTEM TEAM <br>"
+ "website:localhost:8080 <br>"
+ "<img src=\"http://localhost:8080/images/logo-sm.png\">"
+ "</h4>"
                                                    );
                                        } else {
```

```
                                                System.out.print("User Email is
not provided on save user and send email");
                                        }
                                } catch(Exception e) {
                                        System.out.println("Unable to Send
Eamil: "+e.getMessage());
                                }

                        }
                }
        }

        public User getUserByUsername(String username) {
                return userRepo.findByUsername(username);
        }

        public void delete(int id) {
                userRepo.deleteById(id);
        }

        public User getUserById(int id) {
                return userRepo.findById(id).get();
        }

        public List<User> getUsers() {
                return (List<User>) userRepo.findAll();
        }
        public List<User> getUsersByRole(UserRoles role) {
                return (List<User>) userRepo.findByRole(role);
        }

        public String getActivationKey() {
                return "SOS"+random(1000,10000)+"86";
        }

        private int random(int start, int end) {
                return (int)(start+Math.round(Math.random()*(end-start)));
        }

        public User getCurrentlyLoggedInUser(Authentication authentication) {
                if(authentication == null) return null;

                User user = null;
                Object principal = authentication.getPrincipal();

                if(principal instanceof UIGOUserDetails) {
                        user = ((UIGOUserDetails) principal).getUser();
                }
                System.out.println("User: "+user.toString());
//              else if(principal instanceof CustomOAuth2User) {
//                      String username = ((User)principal).getUsername();
```

```
//                      user = getUserByUsername(username);
//                }

                return user;
        }
}
```

- **UserRepo.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.util.List;

import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;

import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.entity.User;

public interface UserRepo extends CrudRepository<User, Integer> {

        public User findByUsername(String username);

        @Query("SELECT u FROM User u WHERE u.username= :username")
        public User getUserbyUsername(@Param("username") String username);

        public List<User> findByRole(UserRoles role);
}
```

- **UserController.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import org.springframework.web.servlet.view.RedirectView;

import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.User;

@Controller
```

```java
@RequestMapping("admin/users")
public class UserController {

	@Autowired
	private UserService userService;

	@GetMapping("")
	public String list(
			Model m
		) {
		List<User> users = userService.getUsers();
//		users.forEach(user->System.out.println(user));

		System.out.println("Message======== "+
m.getAttribute("message"));
		m.addAttribute("users", users);

		return "admin/users/users";
	}

	@GetMapping("/new")
	public String form(Model m) {
		User user = new User();
		System.out.println(user);
		m.addAttribute("user", user);
		m.addAttribute("page_title", "Admin | Add New User");
		return "admin/users/users_form";
	}
	@GetMapping("/{id}/edit")
	public String edit(Model m,
			@PathVariable("id")int id
	) {
		User user = userService.getUserById(id);
		System.out.println("user "+user);
		m.addAttribute("user", user);
		m.addAttribute("page_title", "Admin | User Edit ("+id+")");
		return "admin/users/users_form";
	}

	@PostMapping("/save")
	public String save(
			User user,
			RedirectAttributes ra
	) {
//		System.out.println(user.toString());
		User existingUser =
userService.getUserByUsername(user.getUsername());
		if(existingUser!=null) {
			ra.addFlashAttribute("error", "Username already existed,
use different username");
			return "redirect:/admin/users";
```

```
                }

                user.setUsername(user.getUsername().toLowerCase());
                userService.save(user);
                ra.addFlashAttribute("message", "User information has been
saved!!");
                return "redirect:/admin/users";
        }


        @GetMapping("/{id}/delete")
        public String delete(
                        RedirectAttributes redirectAttributes,
                        @PathVariable("id")int id
                ) {
                userService.delete(id);
                redirectAttributes.addFlashAttribute("message", "User is removed
from database!!!");
                return "redirect:/admin/users";
        }
}
```

- **MyLogoutSuccessHandler.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.io.IOException;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.springframework.security.core.Authentication;
import
org.springframework.security.web.authentication.logout.LogoutSuccessHandler;
import org.springframework.stereotype.Component;

import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.entity.User;

@Component
public class MyLogoutSuccessHandler implements LogoutSuccessHandler {

        @Override
        public void onLogoutSuccess(HttpServletRequest request,
                        HttpServletResponse response, Authentication
authentication)
                        throws IOException, ServletException {

//              UIGOUserDetails userDetails =
```

```
(UIGOUserDetails)authentication.getPrincipal();
//              if(userDetails==null) {
//                      System.out.println("USER NOT Fount authenticated");
//                      return;
//              }
//              User user = userDetails.getUser();

                System.out.println("_---------------- LOGOUT TRIGGERED
---------------------_");

                // Remove Logged out User list in memory
                HttpSession session = request.getSession();
        if (session != null){
                session.removeAttribute("user");
        }

        // redirect to other page
        response.sendRedirect("/login");
        }

}
```

- **MyLoginSuccessHandler.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import
org.springframework.security.web.authentication.AuthenticationSuccessHandler;
import org.springframework.stereotype.Component;

import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.entity.User;

@Component
public class MyLoginSuccessHandler implements AuthenticationSuccessHandler {

        @Autowired
    LoggedInUsers loggedInUsers;

        @Override
```

```
        public void onAuthenticationSuccess(HttpServletRequest request,
                        HttpServletResponse response,
                        Authentication authentication)
        throws IOException, ServletException {

                UIGOUserDetails userDetails =
(UIGOUserDetails)authentication.getPrincipal();
                if(userDetails==null) {
                        System.out.println("USER NOT Fount authenticated");
                        return;
                }

                User user = userDetails.getUser();

                // Add Logged in User list in memory
                HttpSession session = request.getSession(false);
        if (session != null) {
                MyHttpSessionBindingListener mhsb =
                                new MyHttpSessionBindingListener(user,
loggedInUsers);
             session.setAttribute("user", mhsb);
        }

        // code to redirect based on user role
                if(user.getRole().equals(UserRoles.ROLE_ADMIN)) {
                        response.sendRedirect("/admin");
                } else if (user.getRole().equals(UserRoles.ROLE_CUSTOMER)) {
                        response.sendRedirect("/customer");
                } else {
                        response.sendRedirect("/");
                }

        }

}
```

- **MyHttpSessionBindingListener.java**

```
package com.github.sorabh86.uigo.admin.users;

import java.util.List;

import javax.servlet.http.HttpSessionBindingEvent;
import javax.servlet.http.HttpSessionBindingListener;

import org.springframework.stereotype.Component;

import com.github.sorabh86.uigo.entity.User;

@Component
public class MyHttpSessionBindingListener implements HttpSessionBindingListener
{
```

```
    private User user;
    private LoggedInUsers loggedInUsers;

    public MyHttpSessionBindingListener(User user, LoggedInUsers loggedInUsers)
{
        this.user = user;
        this.loggedInUsers = loggedInUsers;
    }

    public MyHttpSessionBindingListener() {}

    @Override
    public void valueBound(HttpSessionBindingEvent event) {
        List<MyHttpSessionBindingListener> users = loggedInUsers.getUsers();
//        System.out.println(event.getValue());
        MyHttpSessionBindingListener user =
                        (MyHttpSessionBindingListener) event.getValue();
        if (!users.contains(user)) {
            users.add(user);
        }
    }

    @Override
    public void valueUnbound(HttpSessionBindingEvent event) {
        List<MyHttpSessionBindingListener> users = loggedInUsers.getUsers();
        MyHttpSessionBindingListener user =
                        (MyHttpSessionBindingListener) event.getValue();
        if (users.contains(user)) {
            users.remove(user);
        }
    }

        public User getUser() {
                return user;
        }

        public void setUsername(User user) {
                this.user = user;
        }

        public LoggedInUsers getLoggedInUsers() {
                return loggedInUsers;
        }

        public void setLoggedInUsers(LoggedInUsers loggedInUsers) {
                this.loggedInUsers = loggedInUsers;
        }
}
```

- **LoggedInUsers.java**

```java
package com.github.sorabh86.uigo.admin.users;

import java.util.ArrayList;
import java.util.List;

import org.springframework.stereotype.Component;

import com.github.sorabh86.uigo.entity.User;

@Component
public class LoggedInUsers {

	private List<MyHttpSessionBindingListener> users;

	public LoggedInUsers() {
	users = new ArrayList<>();
    }

	public List<MyHttpSessionBindingListener> getUsers() {
		return users;
	}

	public void setUsers(List<MyHttpSessionBindingListener> users) {
		this.users = users;
	}
}
```

- **ReturnService.java**

```java
package com.github.sorabh86.uigo.admin.returns;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.entity.ReturnOrder;

@Service
public class ReturnService {

	@Autowired
	private ReturnRepo retRepo;

	public List<ReturnOrder> getAllReturnOrders() {
		return retRepo.findAll();
	}

	public ReturnOrder getReturnOrderById(Integer id) {
		return retRepo.getById(id);
	}
```

```
        public void save(ReturnOrder returnOrder) {
                retRepo.save(returnOrder);

        }
}
```

- **ReturnRepo.java**

```
package com.github.sorabh86.uigo.admin.returns;

import org.springframework.data.jpa.repository.JpaRepository;

import com.github.sorabh86.uigo.entity.ReturnOrder;

public interface ReturnRepo extends JpaRepository<ReturnOrder, Integer> {

}
```

- **ReturnController.java**

```
package com.github.sorabh86.uigo.admin.returns;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.github.sorabh86.uigo.constants.ReturnStatus;
import com.github.sorabh86.uigo.entity.ReturnOrder;

@Controller
@RequestMapping("/admin/returns")
public class ReturnController {

        @Autowired
        private ReturnService retService;

        @GetMapping("")
        public String returnOrderList(Model m) {

                List<ReturnOrder> returnOrders =
retService.getAllReturnOrders();
                System.out.println(returnOrders);
                m.addAttribute("returnOrders", returnOrders);

                return "admin/return/returns";
```

```
        }

        @PostMapping("/status")
        public String changeStatus(Model m,
                                   RedirectAttributes ra,
                                   @RequestParam("id")Integer id,
                                   @RequestParam("returnStatus")ReturnStatus status) {

                ReturnOrder returnOrder = retService.getReturnOrderById(id);
                returnOrder.setStatus(status);
                retService.save(returnOrder);

                ra.addFlashAttribute("message", "Status Updated, Successfully");

                return "redirect:/admin/returns";
        }
}
```

- **PhoneService.java**

```
package com.github.sorabh86.uigo.admin.phones;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;

@Service
public class PhoneService {
        @Autowired
        private PhoneRepo phoneRepo;

        public Phone save(Phone phone) {
                return phoneRepo.save(phone);
        }

        public void delete(int id) {
                phoneRepo.deleteById(id);
        }

        public Phone getPhone(int id) {
                return phoneRepo.findById(id).get();
        }

        public List<Phone> getPhones() {
                return (List<Phone>)phoneRepo.findAll();
        }

        public String getActivationKey() {
```

```
                return "SXY"+Math.random();
        }
}
```

- **PhoneController.java**

```
package com.github.sorabh86.uigo.admin.phones;

import java.io.IOException;

import javax.mail.Multipart;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.util.StringUtils;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;

import com.github.sorabh86.uigo.constants.Constants;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.utils.FileUpload;

@Controller
@RequestMapping("/admin/phones")
public class PhonesController {

        @Autowired
        private PhoneService phoneService;

        @GetMapping("")
        public String listUsers(Model m) {
                m.addAttribute("phones", phoneService.getPhones());
                return "admin/phone/phone";
        }

        @GetMapping("/new")
        public String userForm(Model m) {
                Phone phone = new Phone();
                m.addAttribute("phone", phone);
                m.addAttribute("page_title", "Admin | Add New Phone");
                return "admin/phone/phone_form";
        }

        @PostMapping("/save")
        public String saveUser(Phone phone, @RequestParam("phone_image")
MultipartFile image) throws IOException {
                if(!image.isEmpty()) {
```

```
                        String fileName =
StringUtils.cleanPath(image.getOriginalFilename());
                        phone.setImage(fileName);
                        Phone save = phoneService.save(phone);
                        String uploadDir = Constants.PHONE_DIR+save.getId();

                        FileUpload.deletefile(uploadDir);
                        FileUpload.savefile(uploadDir, fileName, image);
                } else {
                        if(phone.getImage().isEmpty())
                                phone.setImage(null);
                        phoneService.save(phone);
                }
                return "redirect:/admin/phones";
        }

        @GetMapping("/{id}/edit")
        public String editUser(Model m, @PathVariable("id")int id) {
                Phone phone = phoneService.getPhone(id);
                m.addAttribute("phone", phone);
                m.addAttribute("page_title", "Admin | Edit ("+id+")");
                return "admin/phone/phone_form";
        }
        @GetMapping("/{id}/delete")
        public String deleteUser(@PathVariable("id")int id) {
                phoneService.delete(id);
                return "redirect:/admin/phones";
        }
}
```

- **PhoneRepo.java**

```
package com.github.sorabh86.uigo.admin.phones;

import org.springframework.data.repository.CrudRepository;

import com.github.sorabh86.uigo.entity.Phone;

public interface PhoneRepo extends CrudRepository<Phone, Integer> {

}
```

- **PaymentService.java**

```
package com.github.sorabh86.uigo.admin.payments;

import java.util.List;

import org.json.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.entity.Payment;
```

```java
import com.razorpay.Order;
import com.razorpay.RazorpayClient;
import com.razorpay.RazorpayException;

@Service
public class PaymentService {

	public static String RAZORPAY_KEY = "rzp_test_nLIXAPQblFuAPV";
	public static String RAZORPAY_SECRET = "aCSDzmDy8ELRFtdOTUmaufzL";

	private static RazorpayClient client;

	@Autowired
	private PaymentRepo payRepo;

	public List<Payment> getPaymentList() {
		return payRepo.findAll();
	}

	public Payment getPayment(Integer id) {
		if(id==null || id==0) return new Payment();
		return payRepo.findById(id).get();
	}

	public Payment getPaymentByRazorOrderId(String razorOrderId) {
		return payRepo.findByRazorOrderId(razorOrderId);
	}

	public Payment savePayment(Payment payment) {
		return payRepo.save(payment);
	}

	public void deletePayment(Payment payment) {
		payRepo.delete(payment);
	}
	public void deletePaymentById(Integer id) {
		payRepo.deleteById(id);
	}

	public Order createRazorOrder(JSONObject json) throws RazorpayException
{
		if(client==null)  client = new RazorpayClient(RAZORPAY_KEY,
RAZORPAY_SECRET);
		return client.Orders.create(json);
	}
	public List<Order> getRazorPaymentList() throws RazorpayException {
		if(client==null)  client = new RazorpayClient(RAZORPAY_KEY,
RAZORPAY_SECRET);
		return client.Orders.fetchAll();
	}
	public Order getRazorPaymentById(String id) throws RazorpayException {
```

```
                if(client==null)  client = new RazorpayClient(RAZORPAY_KEY,
RAZORPAY_SECRET);
                return client.Orders.fetch(id);
        }
        public void getRazorPaymentById(String id, JSONObject json) throws
RazorpayException {
                if(client==null)  client = new RazorpayClient(RAZORPAY_KEY,
RAZORPAY_SECRET);
                client.Orders.delete(id, json);
        }
}
```

- **PaymentRepo.java**

```
package com.github.sorabh86.uigo.admin.payments;

import org.springframework.data.jpa.repository.JpaRepository;

import com.github.sorabh86.uigo.entity.Payment;

public interface PaymentRepo extends JpaRepository<Payment, Integer> {

        public Payment findByRazorOrderId(String razorPaymentId);
}
```

- **PaymentController.java**

```
package com.github.sorabh86.uigo.admin.payments;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
@RequestMapping("/admin/payments")
public class PaymentController {

        @Autowired
        private PaymentService payService;

        @GetMapping("")
        public String listUsers(Model m) {
                m.addAttribute("payments", payService.getPaymentList());
                return "admin/payment/payment";
        }

}
```

- **OrderService.java**

```
package com.github.sorabh86.uigo.admin.orders;
```

```java
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Sort;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.constants.OrderStatus;
import com.github.sorabh86.uigo.entity.Order;

@Service
public class OrderService {

	@Autowired
	private OrderRepo orderRepo;

	public Order save(Order order) {
		order.setUserId(order.getAddress().getUserId());
		return orderRepo.save(order);
	}

	public Order getOrderById(Integer id) {
		return orderRepo.findById(id).get();
	}

	public List<Order> getOrderslist() {
		return orderRepo.findAll(Sort.by(Sort.Direction.DESC, "id"));
	}

	public List<Order> getOrdersByUserId(Integer user_id) {
		return orderRepo.findByUserId(user_id);
	}

	public List<Order> getOrdersByUserIdAndStatus(Integer userId,
OrderStatus status) {
		return orderRepo.findByUserIdAndStatus(userId, status);
	}
	public List<Order> getOrdersByUserIdAndStatusNot(Integer userId,
OrderStatus status) {
		return orderRepo.findByUserIdAndStatusNot(userId, status);
	}

	public Integer getAllOrderCount() {
		return orderRepo.getByTotalCount();
	}
	public Integer getNewOrderCount() {
		return orderRepo.getByIsVisited(false);
	}

}
```

- **OrderRepo.java**

```
package com.github.sorabh86.uigo.admin.orders;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.stereotype.Repository;

import com.github.sorabh86.uigo.constants.OrderStatus;
import com.github.sorabh86.uigo.entity.Order;

@Repository
public interface OrderRepo extends JpaRepository<Order, Integer> {

	public List<Order> findByUserId(Integer userId);
	public List<Order> findByUserIdAndStatus(Integer userId, OrderStatus
status);

	public List<Order> findByUserIdAndStatusNot(Integer userId, OrderStatus
status);

	@Query(value="SELECT COUNT(*) FROM orders",nativeQuery = true)
	public Integer getByTotalCount();

	@Query(value="SELECT COUNT(*) FROM orders WHERE
is_visited=?",nativeQuery = true)
	public Integer getByIsVisited(Boolean isVisited);
}
```

- **OrderItemRepo.java**

```
package com.github.sorabh86.uigo.admin.orders;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import com.github.sorabh86.uigo.entity.OrderItem;

@Repository
public interface OrderItemRepo extends JpaRepository<OrderItem, Integer> {

}
```

- **OrderController.java**

```
package com.github.sorabh86.uigo.admin.orders;

import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```java
import javax.websocket.server.PathParam;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import com.github.sorabh86.uigo.constants.OrderStatus;
import com.github.sorabh86.uigo.entity.Order;
import com.github.sorabh86.uigo.entity.User;

@Controller
@RequestMapping("admin/orders")
public class OrderController {

	@Autowired
	private OrderService orderService;

	@GetMapping("")
	public String list(Model m) {
		List<Order> orders = orderService.getOrderslist();
		m.addAttribute("orders", orders);
		return "admin/order/orders";
	}

	@GetMapping("/{id}/details")
	public String details(Model m, @PathVariable("id")Integer order_id) {
		Order order = orderService.getOrderById(order_id);

		if(!order.getIsVisited()) {
			order.setIsVisited(true);
			order = orderService.save(order);
		}

		m.addAttribute("order", order);
		return "admin/order/order-details";
	}

	@PostMapping("/{id}/status")
	public ResponseEntity changeStatus(
		@PathVariable("id")int id,
		@RequestParam("order_status")OrderStatus orderStatus,
		@RequestParam("delivery_days")Integer days
	) {
```

```
                Order order = orderService.getOrderById(id);
                order.setStatus(orderStatus);
                order.setExpected_delivery_days(days);
                if(orderStatus==OrderStatus.DELIVERED) {
                        order.setDelivery_date(new Date());
                }

                orderService.save(order);

                Map<String, Object> map = new HashMap<>();
                map.put("status", orderStatus);
                map.put("message", "OK");

                return ResponseEntity.ok(map);
        }

        @GetMapping("/{id}/delete")
        public String delete(@PathVariable("id")int id) {
                return "redirect:/admin/users";
        }
}
```

- **FeedbackService.java**

```
package com.github.sorabh86.uigo.admin.feedbacks;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.entity.FeedbackRating;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;

@Service
public class FeedbackService {
        @Autowired
        private FeedbackRepo feedRepo;

        public FeedbackRating save(FeedbackRating feed) {
                return feedRepo.save(feed);
        }

        public void delete(Integer id) {
                feedRepo.deleteById(id);
        }

        public FeedbackRating getFeedback(Integer id) {
                if(id==null || id==0) return new FeedbackRating();
                else return feedRepo.findById(id).get();
        }
```

```
        public List<FeedbackRating> getFeedbacks() {
                return (List<FeedbackRating>) feedRepo.findAll();
        }

        public FeedbackRating getFeebackByUserAndPhone(User user, Phone phone) {
                return feedRepo.findByUserAndPhone(user, phone);
        }

        public FeedbackRating getFeedbackByUidPidOid(Integer uid, Integer pid,
Integer oid) {
                return feedRepo.getFeedByUserAndPhoneAndOrderId(uid, pid, oid);
        }
}
```

- **FeedbackRepo.java**

```
package com.github.sorabh86.uigo.admin.feedbacks;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

import com.github.sorabh86.uigo.entity.FeedbackRating;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.entity.User;

public interface FeedbackRepo extends JpaRepository<FeedbackRating, Integer> {

        public FeedbackRating findByUserAndPhone(User user, Phone phone);

        @Query(value = "SELECT * FROM feedback_rating WHERE user_id=? AND
phone_id=? AND order_id=?", nativeQuery = true)
        public FeedbackRating getFeedByUserAndPhoneAndOrderId(Integer user_id,
Integer phone_id, Integer order_id);
}
```

- **FeedbackController.java**

```
package com.github.sorabh86.uigo.admin.feedbacks;

import java.io.IOException;

import javax.mail.Multipart;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.util.StringUtils;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
```

```java
import org.springframework.web.multipart.MultipartFile;

import com.github.sorabh86.uigo.constants.Constants;
import com.github.sorabh86.uigo.entity.FeedbackRating;
import com.github.sorabh86.uigo.entity.Phone;
import com.github.sorabh86.uigo.utils.FileUpload;

@Controller
@RequestMapping("/admin/feedbacks")
public class FeedbackController {

	@Autowired
	private FeedbackService feedService;

	@GetMapping("")
	public String listUsers(Model m) {
		m.addAttribute("feedbacks", feedService.getFeedbacks());
		return "admin/feedback/feedback";
	}

	@GetMapping("/new")
	public String userForm(Model m) {
		FeedbackRating feed = new FeedbackRating();
		m.addAttribute("feed", feed);
		m.addAttribute("page_title", "Admin | Add New Feedback");
		return "admin/feedback/feed_form";
	}

	@PostMapping("/save")
	public String saveUser(FeedbackRating feed) {
		feedService.save(feed);
		return "redirect:/admin/feedbacks";
	}

	@GetMapping("/{id}/edit")
	public String editUser(Model m, @PathVariable("id")int id) {
		FeedbackRating feed = feedService.getFeedback(id);
		m.addAttribute("feed", feed);
		m.addAttribute("page_title", "Admin | Edit ("+id+")");
		return "admin/feedback/feed_form";
	}
	@GetMapping("/{id}/delete")
	public String deleteUser(@PathVariable("id")int id) {
		feedService.delete(id);
		return "redirect:/admin/feedbacks";
	}
}
```

- **ChatRepo.java**

```java
package com.github.sorabh86.uigo.admin.chats;
```

```java
import java.util.List;

import javax.transaction.Transactional;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.stereotype.Repository;

import com.github.sorabh86.uigo.constants.ChatStatus;
import com.github.sorabh86.uigo.entity.Chat;

@Repository
@Transactional
public interface ChatRepo extends JpaRepository<Chat, Integer> {

	@Transactional
	public void deleteAllByUsernameAndStatus(String username, ChatStatus status);

	public List<Chat> findByStatusAndReceiverUsername(ChatStatus status, String receiverUsername);

	@Query(value = "SELECT * FROM chat WHERE ("
			+ "	((username=?1 OR username=?2) AND receiver_username IS NOT NULL ) AND "
			+ "	((receiver_username=?1 OR receiver_username=?2) AND username IS NOT NULL)"
			+ ") AND status='MESSAGE';", nativeQuery = true)
	public List<Chat> findGroupMessageByUsername(String username, String receiverUsername);

	@Query(value = "SELECT COUNT(*) FROM chat WHERE is_visited=0  AND receiver_username IS NULL AND status='MESSAGE';", nativeQuery = true)
	public Integer getPublicNotice();

	@Query("SELECT c.username FROM Chat c WHERE c.status='JOIN'")
	public List<String> findOnlineUsers();

	public List<Chat> findByStatus(ChatStatus status);

}
```

- **ChatController.java**

```java
package com.github.sorabh86.uigo.admin.chats;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

import org.json.JSONObject;
```

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.messaging.handler.annotation.MessageMapping;
import org.springframework.messaging.handler.annotation.Payload;
import org.springframework.messaging.handler.annotation.SendTo;
import org.springframework.messaging.simp.SimpMessageHeaderAccessor;
import org.springframework.messaging.simp.SimpMessagingTemplate;
import org.springframework.messaging.simp.annotation.SendToUser;
import org.springframework.security.core.annotation.AuthenticationPrincipal;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import com.github.sorabh86.uigo.admin.users.UserService;
import com.github.sorabh86.uigo.config.UIGOUserDetails;
import com.github.sorabh86.uigo.constants.ChatStatus;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.entity.Chat;
import com.github.sorabh86.uigo.entity.User;
import com.google.gson.Gson;

@Controller
@RequestMapping("/admin/chats")
public class ChatController {

	@Autowired
	private SimpMessagingTemplate messagingTemplate;

	@Autowired
	private UserService userService;
	@Autowired
	private ChatRepo chatRepo;

	@GetMapping("")
	public String chat(Model m,
			@AuthenticationPrincipal UIGOUserDetails ud) {
		User user = ud.getUser();
		List<User> users =
userService.getUsersByRole(UserRoles.ROLE_CUSTOMER);
		List<String> allUsers = users.stream().map(u ->
u.getUsername()).collect(Collectors.toList());
		List<String> onlineUsers = chatRepo.findOnlineUsers();
		List<Chat> publicMessages =
chatRepo.findByStatusAndReceiverUsername(ChatStatus.MESSAGE, null);

		m.addAttribute("user", user);
		m.addAttribute("allUsers", allUsers);
		m.addAttribute("onlineUsers", onlineUsers);
		m.addAttribute("publicMessages", publicMessages);
```

```
                return "admin/chat/chats";
        }

        @GetMapping("/messages/{username}/{receiver}")
        @ResponseBody
        public String getPrivateMessages(@PathVariable("username")String
username, @PathVariable("receiver")String receiver) {
                List<Chat> chats = chatRepo.findGroupMessageByUsername(username,
receiver);
                Gson gson = new Gson();
                String json = gson.toJson(chats);
                return json;
        }

        /** MESSAGING MAPPINGS **/
        // /uigo/message
        @MessageMapping("/message")
        @SendTo("/chatroom/public")
        public Chat receivePublicMessage(@Payload Chat chat) throws
InterruptedException {
                Chat saved = chatRepo.save(chat);
                Thread.sleep(300);
                return saved;
        }

        /** adding users to session to detect offline **/
        @MessageMapping("/message.add")
    @SendTo("/chatroom/public")
    public Chat addUser(@Payload Chat chat, SimpMessageHeaderAccessor
headerAccessor) throws InterruptedException {

                Chat saved = chatRepo.save(chat);
                Thread.sleep(300);

        // Add username in web socket session
        headerAccessor.getSessionAttributes().put("chat", chat);

        return saved;
    }

        /** private message sending and receiving
         * @throws InterruptedException **/
        @MessageMapping("/private")
        public Chat receivePrivateMessage(@Payload Chat chat) throws
InterruptedException {
                Chat saved = chatRepo.save(chat);
                Thread.sleep(300);
                System.out.println(chat.getReceiverUsername());
                // /user/{name}/private

messagingTemplate.convertAndSendToUser(chat.getReceiverUsername(), "/private",
```

```
chat);
                return saved;
        }

}
```

- **AddressService.java**

```
package com.github.sorabh86.uigo.admin.address;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.User;

@Service
public class AddressService {
        @Autowired
        private AddressRepo addressRepo;

        public Address save(Address address) {
                return addressRepo.save(address);
        }

        public void delete(Integer id) {
                addressRepo.deleteById(id);
        }

        public Address getAddress(Integer id) {
                if(id==null || id==0) return new Address();
                else return addressRepo.findById(id).get();
        }

        public List<Address> getAddresses() {
                return (List<Address>) addressRepo.findAll();
        }

        public List<Address> getAddressByUserId(int id) {
                return  addressRepo.findByUserId(id);
        }

}
```

- **AddressRepo.java**

```
package com.github.sorabh86.uigo.admin.address;

import java.util.List;

import org.springframework.data.repository.CrudRepository;
```

```java
import com.github.sorabh86.uigo.entity.Address;

public interface AddressRepo extends CrudRepository<Address, Integer> {

	public List<Address> findByUserId(Integer user_id);
}
```

- **AddressController.java**

```java
package com.github.sorabh86.uigo.admin.address;

import java.lang.reflect.Array;
import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.github.sorabh86.uigo.admin.users.UserService;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.User;

@Controller
@RequestMapping("admin/address")
public class AddressController {

	@Autowired
	private AddressService addressService;

	@Autowired
	private UserService userService;

	@GetMapping("")
	public String list(Model m) {
		List<Address> addresses = addressService.getAddresses();

		m.addAttribute("addresses", addresses);
		return "admin/address/address";
	}
```

```
        @GetMapping("/{user_id}")
        public String list(Model m, @PathVariable("user_id")int user_id) {
                User user = userService.getUserById(user_id);
                List<Address> addresses = user.getAddresses();

                m.addAttribute("addresses", addresses);

                return "admin/address/address";
        }

        @GetMapping("/new")
        public String form(Model m) {
                List<User> users = userService.getUsers();
                Address address = new Address();

                m.addAttribute("users", users);
                m.addAttribute("address", address);
                m.addAttribute("page_title", "Admin | Add New Address");

                return "admin/address/address_form";
        }

        @GetMapping("/new/{user_id}")
        public String form(Model m, @PathVariable("user_id") int id) {
                Address address = new Address();
                User user = userService.getUserById(id);
                if(user==null) {
                        System.out.println("User Not found
exception.............................");
                } else {
                        address.setUserId(user.getId());
//                      address.setUser(user);
                        address.setFull_name(user.getFull_name());
                }

                m.addAttribute("address", address);
                m.addAttribute("page_title", "Admin | New Address");

                return "admin/address/address_form";
        }

        @PostMapping("/save")
        public String save(
                        Address address,
                        RedirectAttributes redirectAttribute
        ) {

                String redirect = "redirect:/admin/address";
//              String redirect =
"redirect:/admin/address/"+address.getUserId();
```

```
//                System.out.println(address);

                addressService.save(address);
                redirectAttribute.addFlashAttribute("message", "User has been
Saved!!!" );

                return redirect;
        }

        @GetMapping("/{id}/edit")
        public String edit(Model m,
                           @PathVariable("id")int id) {

                Address address = addressService.getAddress(id);
                m.addAttribute("address", address);

                m.addAttribute("page_title", "Admin | Edit Address ("+id+")");

                return "admin/address/address_form";
        }

        @GetMapping("/{id}/delete")
        public String delete(
                RedirectAttributes redirectAttributes,
                @PathVariable("id")int id
        ) {
                addressService.delete(id);
                redirectAttributes.addFlashAttribute("message", "Address has
been deleted");
                return "redirect:/admin/address";
        }
}
```

- ***/resources*/static/bootstrap/4.6.1 (download online: and put here)**
- **/resources/css/all.css (download online: *Font Awesome Free 5.15.4*)**
- **/resources/css/jquery.bootstrap-touchspin.min.css (download online: *Bootstrap TouchSpin – v3.0.1*)**
- **/resources/css/chat.css**

```
.chat-box{
  box-shadow:0 2.8px 2.2px rgba(0, 0, 0, 0.034),0 6.7px 5.3px rgba(0, 0, 0,
0.048),0 12.5px 10px rgba(0, 0, 0, 0.06),0 22.3px 17.9px rgba(0, 0, 0, 0.072),0
41.8px 33.4px rgba(0, 0, 0, 0.086),0 100px 80px rgba(0, 0, 0, 0.12);
  height: 600px;
}

.member-list{
  width: 20%;
}
```

```css
ul {
  padding: 0;
  list-style-type: none;
}
.member-content{overflow-y: auto;}
.member{
        display: flex;
    align-items: center;
        position:relative;
        text-transform: uppercase;
    font-weight: bold;
    margin: 10px 0;
    border: 1px solid #ccc;
    border-radius: 40px;
    padding: 10px 20px;
    cursor: pointer;
    box-shadow: 0 8px 8px -4px lightblue;
}
.member.active{
  background: #007bff;
  color:#fff;
}
.member:hover{
  background: #0069d9;
  color:#fff;
}
.chat-status {
        position:absolute;
        left: 50%;
    top: 8px;
    transform: translate(-50%, -100%);
    box-shadow: 0 0 8px rgb(0 0 0 / 50%);
}
.user-status {
    display: inline-block;
    margin-left: auto;
    width: 20px;
    height: 20px;
}

.chat-content {
        display:none;
}
.chat-content.active{
        display:flex;
        flex-direction:column;
        flex-grow:1;
}
.chat-messages{overflow-y: scroll;}

.avatar{
```

```
        position: relative;
        text-align:center;
        font-weight: bold;
        background-color: cornflowerblue;
        padding: 3px 5px;
        border-radius: 5px;
        color:#fff;
}
.avatar:after {
    content: "";
    position: absolute;
    border-right: none;
    border-left: 10px solid cornflowerblue;
    border-top: 10px solid transparent;
    border-bottom: 10px solid transparent;
    left: 100%;
    top: 5px;
}
.avatar.self{
  color:#000;
  background-color: greenyellow;
}
.avatar.self:after {
        content: "";
    position: absolute;
    border-left: none;
    border-right: 10px solid greenyellow;
    left: 0;
    top: 5px;
    transform: translate(-100%, 0);
}
.input-message {border-radius: 50px 0 0 50px!important;padding-left:25px}
.send-button{border-radius: 0 50px 50px 0!important;padding-right:25px}
.message{
        padding: 5px;
        width: auto;
        display: flex;
        flex-direction: row;
        align-items: center;
        margin: 5px 10px;
}
.message-data{
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 10px;
    margin: 0 15px;
    box-shadow: 0 3px 10px rgb(0 0 0 / 20%);
}
.message.self{
  justify-content: end;
}
```

- **/resources/css/style.css**

```
#nav-main a,
#nav-top a.nav-link,
#nav-top a.nav-brand,
.about-us h2 {
    text-transform: uppercase;
    font-weight: bold;
}

.img-responsive {max-width: 100%;
    max-height: 100%;}
.phone-detail-img {height:500px;}
.phone-img {height:280px}
.cart-item-img {height: 90px;}

#nav-main .home {
    background-color: rgba(255,255,255,0.8)!important;
    position: absolute;
    top: 43px;
    left: 0;
    width: 100%;
    z-index: 1;
}

#nav-main .collapse .navbar-item:first-child {
    /* margin-left: 40px; */
}
#nav-main .collapse .navbar-item {
    margin-right: 20px;
}
#nav-main .collapse .navbar-item a.nav-link {
    color:#003380;
    font-weight: bold;
}
#nav-main .collapse .navbar-item.active {
    border-bottom: 10px solid #aa0000;
}
#nav-main .collapse .navbar-item.active .nav-link {
    padding-bottom: 4px;
}
#slider {
    position: relative;
}
#slider .carousel-item  h2,
#slider .carousel-item  h3{
    color: white;
    position: absolute;
}
#slider .banner-1 h2 {
    top: 190px;
    left: 910px;
```

```
    font-size: 42px;
}
#slider .banner-1 h3 {
    top: 302px;
    left: 910px;
    font-size: 32px;
    line-height: 2em;
}
#slider .banner-2 h2 {
    top: 190px;
    left: 200px;
    font-size: 42px;
}
#slider .banner-2 h3 {
    top: 302px;
    left: 200px;
    font-size: 32px;
    line-height: 2em;
}

.bg-blue {
    background: #003380;
}
.bg-black {
    background: #010101;
}
.text-orange {
    color: #aa0000;
}

#contact-form textarea {height: 100%;}

.order-mark,
.order-mark-join {
    width: 40px;
    height: 40px;
    border-radius: 100%;
    border: 5px solid blue;
    position: relative;
}
.order-mark-join .cborder {
        position: absolute;
    border-left: 5px solid blue;
    border-right: 5px solid blue;
    width: 20px;
    height: 14px;
    left: 50%;
    top: 96%;
    background: #fff;
    z-index: 1;
    transform: translate(-50%, 0);
```

```css
}
.order-mark-join.active:before {
    content: "";
    width: 6px;
    height: 38px;
    background: red;
    position: absolute;
    /* border: 2px solid white; */
    top: 50%;
    left: 50%;
    transform: translate(-50%, 0);
    z-index: 2;
}
.order-mark.active:after,
.order-mark-join.active:after {
    content: "";
    width: 20px;
    height: 20px;
    border-radius: 50%;
    position: absolute;
    background: red;
    left: 50%;
    top: 50%;
    transform: translate(-50%, -50%);
    z-index: 4;
}
.order-item {
    border: 1px solid #ccc;
    padding: 15px;
}
.tracking {
    line-height: 2.5em;
    text-align: right;
}

.social-links {
        display: flex;
    column-gap: 10px;
}
.btn-back {
    position: relative;
    background: #ccc;
    font-weight: bold;
    transition: .5s all;
}
.btn-back:hover {
        background: #fff;
}
.btn-back:after {
    content: "";
    border-right: 20px solid #ccc;
```

```
    border-top: 18px solid transparent;
    border-bottom: 18px solid transparent;
    position: absolute;
    left: 0;
    top: 0;
    transform: translate(-100%, 0);
    transition: .5s all;
}
.btn-back:hover:after {
    border-right-color: #fff;
}

#loading {
        display:none;
        position: fixed;
    width: 100%;
    height: 100vh;
    top: 0;
    left: 0;
    background: rgba(0,0,0,0.4);
}

.bg-visted {
        background-color:#d8ffe1!important
}
.border-amt {
        border-top: 3px solid #f2d4d7!important;
        border-bottom: 3px solid #f2d4d7!important;
}
```

- **/resources/js/1.1.4/sockjs.min.js (download online: *sockjs-client v1.1.4*)**
- **/resources/js/2.3.3/stomp.min.js (download online: stomp-websocket *v2.3.3*)**
- **/resources/js/jquery.bootstrap-touchspin.min.js (download online: *Bootstrap TouchSpin – v3.0.1*)**
- **/resources/js/jquery.js (download online: *jQuery JavaScript Library v3.1.0*)**
- **/resources/js/jQuery.print.js (download online: jQuery.print, version 1.6.0)**
- **/resources/js/payment-razorpay.js**

```
var $paymentStatusWindow = $("#paymentStatusWindow");

// checkout payment using ajax
function startPayment(formData) {
        console.log("payment started... ", formData);

        if(window.hasOwnProperty("RazorPay")) {
                alert("Unable to initilise online payment, Please check network
connection.")
                return;
        }
```

```
        if(!formData && !formData.amount && formData.amount=="") {
                return;
        } else {
//              formData.amount = formData.amount*100;
        }

        // request server to create order -- jquery
//      var reqData = {
//              address_id: formData["address_id"],
//              "payment-method": formData['payment-method'],
//              "message": formData['message'],
//              amount: amount,
//              _csrf: formData[name='_csrf']
//      };

        var reqData = JSON.stringify(formData);

        $.ajax({
                url: "/cart/order",
                data: reqData,
                contentType: "application/json; charset=utf-8",
                type: 'POST',
                dataType: "json",
                beforeSend: function(){
                $('#loading').show();
            }, success: function(res) {
                        console.log(res);
                        createRazorPaymentWindow(res);
                }, error: function(error) {
                        console.error(error);
                        alert("Something went wrong");
                }, complete: function(){
                $('#loading').hide();
            }
        });
};

/**
 *      res = {amount, razor_order_id, username, email, phone}
 */
function createRazorPaymentWindow(res) {
        // Open payment form after order is created on razorpay server
        var options = {
                key: "rzp_test_nLIXAPQblFuAPV",
                amount: res.amount,
                currency: "INR",
                name: "UIGO Shopping Cart Payment",
                description: "Cart Purchase",
                image: "/images/logo-sm.png",
                order_id: res.id,
```

```
                handler: function(response) {
                        console.log("Payment Success:",response);
                        updatePaymentStatus({
                                razorOrderId:response.razorpay_order_id,
                                razorPaymentId:response.razorpay_payment_id,
                                status:"PAID"
                        });
                },
                // auto filling razorpay form
                prefill: {
                        name: res.username,
                        email: res.email,
                        "contact": res.phone
                },
                notes: {
                        address: "UIGO Shopping Site"
                },
                theme: {
                        color: "#003380"
                }
        };
        // Initiate Payment
        var rzp = new Razorpay(options);

        rzp.on("payment.fail", function(r) {
                console.log("Payment Failed:", r.error);
                updatePaymentStatus({
                        razorOrderId:response.razorpay_order_id,
                        razorPaymentId:response.razorpay_payment_id,
                        status:"FAILED"
                });
        });

        rzp.open();
}

function updatePaymentStatus(payment) {
        $.post({
                url:"/cart/payment/update",
                method:"POST",
                data:payment,
                dataType:"json",
                beforeSend: function(){
                $('#loading').show();
            }, success:function(data) {
                        console.log("payment-status", data);
                        $paymentStatusWindow.data("status", data.status);
                        var $modalBody = $paymentStatusWindow.find(".modal-
body");
                        $modalBody.find(".icon").hide();
                        if(data.status=="FAILED") {
```

```
                                $modalBody.find(".icon.failure").show();
                                $modalBody.find(".message").html("YOUR PAYMENT
UNSUCCSESSFUL!!!");
                        } else if(data.status=="PAID") {
                                $modalBody.find(".icon.success").show();
                                $modalBody.find(".message").html("YOUR PAYMENT
IS SUCCESSFUL!!!, ");
                        }
                        $paymentStatusWindow.data("data", data);
                        $paymentStatusWindow.modal();
                },
                error: function(err) {
                        console.log("payment-error", status);
                        var $modalBody = $paymentStatusWindow.find(".modal-
body");
                        $modalBody.find(".icon").hide();
                        $modalBody.find(".message").html("Your payment is
successful, but unable to save to our server. We will update information
manually if something didn't matched.'");
                        $paymentStatusWindow.modal();

                }, complete: function(){
                $('#loading').hide();
            }
        });
};

$paymentStatusWindow.find(".ok-btn").on("click", function(){
        var data = $paymentStatusWindow.data("data");
        var url = "/orderstatus/"+data.orderId+"/"+data.paymentId;
        window.location.href = url;
});
```

- **/resources/js/chat.js**

```
;(function(so, $) {
        so.URL = 'http://localhost:8080/ws';
        so.MURL = "/uigo/message";
        so.PMURL = "/uigo/private";
        so.AURL = "/uigo/message.add";
        so.$memberList = $(".member-list");
        so.$chatRooms = $(".chat-rooms");

        var p = so.Chat = function(username) {
                this.stompClient = null;
                this.username = username;
                this.publicChats = [];
                this.privateChats = [];

                console.log("Try to connect with server ", username);
                this.connect();
        };
```

```
        var p = so.Chat.prototype = {};

        p.connect = function() {
                this.socket = new SockJS(so.URL);
                this.stompClient = Stomp.over(this.socket);
                this.stompClient.connect({}, this.onConnected.bind(this),
this.onError.bind(this));
        };

        p.onConnected = function() {
                console.log("onConnected", this.stompClient);

                // Subscribe to the Public Topic
            this.stompClient.subscribe('/chatroom/public',
this.onMessageReceived.bind(this));
                // Subscribe to the Private Topic
            this.stompClient.subscribe('/user/'+this.username+'/private',
                this.onPrivateMessageReceived.bind(this));

            var users = $(".member-list ul .member");
            for(var i = 0; i<users.length; i++) {
                        var username = $(users[i]).data("href");
                this.stompClient.subscribe('/user/'+username+'/private',
                        this.onPrivateMessageReceived.bind(this));
                }

            // User Join
            this.stompClient.send(so.AURL, {}, JSON.stringify({
            username: this.username,
            status:"JOIN"
        }));

            if(this.connectedHandler) this.connectedHandler();
        };
        p.onError = function (error) {
                console.log('Could not connect to WebSocket server. Please
refresh this page to try again!');
        };
        p.onMessageReceived = function(payload){
                console.log("onMesssageReceived:", payload);
                var payloadData = JSON.parse(payload.body);

                if(payloadData.status === "JOIN") {
                        if(this.username!=payloadData.username) {
                                var $userStatus = so.
$memberList.find("ul ."+payloadData.username+" .user-status");
                                $userStatus.removeClass("bg-danger");
                                $userStatus.addClass("bg-success");
//                              if(so.
$memberList.find("ul ."+payloadData.username).length<=0){
```

```
//                                          so.
$memberList.find("ul").append(this.getUserTab(payloadData.username));
//                                          so.
$chatRooms.append(this.getUserContent(payloadData.username));
//                                  }
                        }
                } else if(payloadData.status === "LEAVE") {
                        var $userStatus = so.
$memberList.find("ul ."+payloadData.username+" .user-status");
                        $userStatus.removeClass("bg-success");
                        $userStatus.addClass("bg-danger");
//                      so.$memberList.find("ul li[data-
href='."+payloadData.username+"']").remove();
//                      so.$chatRooms.find("."+payloadData.username).remove();
                } else {
                        so.$chatRooms.find(".chatroom .chat-
messages").append(this.getChatMessage(payloadData));
                }

                if(this.updateHandler) this.updateHandler(payloadData);
        };
        p.onPrivateMessageReceived = function(payload) {
                console.log("onPrivateMessageReceived:", payload);
                var payloadData = JSON.parse(payload.body);

                if(payloadData.status === "MESSAGE") {
                        if(this.username==payloadData.username) {
                                so.
$chatRooms.find("."+payloadData.receiverUsername+" .chat-
messages").append(this.getChatMessage(payloadData));
                        } else {
                                so.
$chatRooms.find("."+payloadData.username+" .chat-
messages").append(this.getChatMessage(payloadData));
                        }
                }

        if(this.updateHandler) this.updateHandler(payloadData);
        };
        p.sendMessage = function(message){
        if (this.stompClient) {
          console.log("sendValue:", message);
          this.stompClient.send(so.MURL, {}, JSON.stringify({
            username: this.username,
            message: message,
            status:"MESSAGE"
          }));
        }
    };
        p.sendPrivateMessage = function(message, receiverUsername){
                if (this.stompClient) {
```

```
            console.log("sendValue:", message);
            this.stompClient.send(so.PMURL, {}, JSON.stringify({
              username: this.username,
                  receiverUsername: receiverUsername,
              message: message,
              status:"MESSAGE"
            }));
          }
      };

          p.getChatMessage = function(chat) {
                  var dateParts = chat.date.split("-");
                  var dateStr =
dateParts[2].substr(0,2)+"/"+dateParts[1]+"/"+dateParts[0].substr(2,2);
                  console.log(dateStr);
                  var str = '';
                  if(chat.username == this.username) {
                          str = '<li class="message self">\
                                  <div class="message-
data">'+chat.message+'</div>\
                                  <div class="avatar self">'+chat.username+'<br
/>\
                                  <small class="badge">'+dateStr+'</small>\
                                  </div>\
                          </li>';
                  } else {
                          str = '<li class="message">\
                                  <div class="avatar">'+chat.username+'<br />\
                                  <small class="badge">'+dateStr+'</small>\
                                  </div>\
                                  <div class="message-
data">'+chat.message+'</div>\
                          </li>';
                  }
                  return str;
          }
})(window.so = window.so || {}, jQuery);

/**** DOM SCRIPTURES **/
var $body = $(document.body);

var $startBtn = $("#start-btn");
$startBtn.click(loginHandler);
function loginHandler(){
          if(window.chat) {
                  console.log("already connected");
          } else {
                  var username = $startBtn.data("name");

                  window.chat = new so.Chat(username);
                  window.chat.connectedHandler=function(){
```

```
                        $(".chat-status").html('<span class="fa fa-circle text-
success" ></span> Online');
                };
                window.chat.updateHandler = function(){
                        scrollBottom();
                };
        }
        return false;
}

function scrollBottom(){
        // scroll active chat box to bottom
        $chatMessages = $(".chat-content.active .chat-messages")
        $chatMessages.scrollTop($chatMessages[0].scrollHeight);
}

function getGroupChat(username, $element) {
        $.ajax({
                url:"/admin/chats/messages/"+username+"/"+window.chat.username,
                dataType:"json",
                success:function(s){
                        var htmlStr = '';
                        for(var i in s) {
                                var chat = s[i], str='', date = (new
Date(chat.date)).toJSON();
                                chat.date = date;
                                htmlStr += window.chat.getChatMessage(chat);
                        }
                        $element.html(htmlStr);
                        console.log('get-private-chat'+s);
                }, error:function(e){
                        console.log(e)
                }
        });
}
loginHandler();

// Switch between different rooms
$body.on("click", ".member-list .member", function(){
        var $this = $(this),
                href = $this.data("href");

        $(".member-list .member").removeClass("active");
        $this.addClass("active");

        var $chatContent = $('.chat-content');
        $chatContent.removeClass("active");
        var $element = $chatContent.filter('.'+href).addClass("active");

        if(href!='chatroom')
                getGroupChat(href, $element.find('.chat-messages'));
```

```
        setTimeout(scrollBottom, 100);
        return false;
});

// Sending messages to different rooms
$body.on("keyup", ".input-message", function(e){
        if(e.keyCode == 13) { // ENTER KEY
                var $this = $(this),
                        $parent = $this.parent().parent(),
                        message = $(this).val(),
                        receiverUsername = $(".member.active").data("href");

                if($parent.hasClass('chatroom')) {
                        window.chat.sendMessage(message);
                } else {
                        window.chat.sendPrivateMessage(message,
receiverUsername);
                }
                $this.val("");
        }
        return false;
})
$body.on("click", ".send-button", function() {
        var $this = $(this),
                $parent = $this.parent(),
                $parent2 = $parent.parent(),
                $input = $parent.find(".input-message"),
                message = $input.val(),
                receiverUsername = $(".member.active").data("href");

        if($parent.hasClass('chatroom')) {
                window.chat.sendMessage(message);
        } else {
                window.chat.sendPrivateMessage(message, receiverUsername);
        }
        $input.val('');
        return false;
});
```

- **/resources/templates/verification.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Verification')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />
        <div class="container">
                <h2 class="border-bottom mt-4 pb-3 mb-3">Account
Verification</h2>
```

```
                <p>[[${message}]]</p>
                <p>You access your account, here <a
th:href="@{/login}">login</a></p>
        </div>

        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
</body>
</html>
```

- **/resources/templates/register.html**

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Register Page')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />
        <div class="container">
                <p class="alert alert-success mt-4" th:if="${message}">[[$
{message}]]</p>
                <p class="alert alert-danger mt-4" th:if="${error}">[[$
{error}]]</p>
                <h2 class="text-center mb-3 mt-4">Customer Register</h2>
                <form class="form" name="customer_form"
th:action="@{/customer/save}" method="post" th:object="${user}">
                        <div class="form-group row">
                                <label class="col-md-3 text-right"
for="username">Username: </label>
                                <div class="col-md-9">
                                        <input class="form-control" type="text"
name="username" th:field="*{username}"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-3 text-right"
for="password">Password:</label>
                                <div class="col-md-9">
                                        <input class="form-control"
type="password" name="password" th:field="*{password}" />
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-3 text-right"
for="full_name">Full Name:</label>
                                <div class="col-md-9">
                                        <input class="form-control" type="text"
name="full_name" th:field="*{full_name}"/>
                                </div>
```

```
                        </div>
                        <div class="form-group row">
                                <label class="col-md-3 text-right"
for="email">Email:</label>
                                <div class="col-md-9">
                                        <input class="form-control" type="email"
name="email"  th:field="*{email}"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-3 text-right"
for="phone_no">Phone No:</label>
                                <div class="col-md-9">
                                        <input class="form-control" type="text"
name="phone_no" th:field="*{phone_no}"/>
                                </div>
                        </div>
                        <input type="hidden" name="role" th:field="*{role}" />
                        <input type="hidden" name="status"
th:field="*{status}"/>
                        <div class="form-group">
                                <div class="offset-md-3">
                                <input class="btn btn-primary text-center"
type="submit" value="Register" />
                                <span class="ml-4">Already have account? <a
class="btn btn-link" th:href="@{/login}">login</a></span>
                                </div>
                        </div>
                </form>
        </div>

        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
</body>
</html>
```

- **/resources/templates/phones.html**

```
<!doctype html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Phones Listing Page')" />
<body>
<style>
.card-body {min-height:88px}
</style>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />
        <div class="container mt-4">
                <h1 class="mb-3">Phones</h1>
```

```
                <div th:if="${message}" class="alert alert-success">
                        <th:block th:utext="${message}"></th:block>
                </div>

                <div class="phone_list row">
                        <div th:each="phone : ${phones}" class="col-lg-3 col-md-
4 mb-3">
                                <div class="card shadow-lg">
                                        <div class="card-header">
                                                <a class="d-flex phone-img"
th:href="@{'/phones/'+${phone.id}}">
                                                        <img th:src="$
{phone.imagePath}" alt="Phone Image" class="img-responsive m-auto" />
                                                </a>
                                        </div>
                                        <div class="card-body">
                                                <h4 class="card-title">[[$
{phone.title}]]</h4>
                                                <span class="desc" th:if="${!
phone.description.isEmpty()}">
                                                        <th:block th:text="$
{#strings.abbreviate(phone.description,52).replaceAll('<.*?>', '')}"></th:block>
                                                </span>
                                                <span class="no-desc" th:if="$
{phone.description.isEmpty()}">
                                                        <th:block >No
Description Found</th:block>
                                                </span>
                                        </div>
                                        <div class="">
                                                <p class="text-danger mb-1 text-
center"><strong>Rs. <span class="price">[[${phone.price}]]</span></strong></p>
                                                <form class="form pl-1 pr-1"
th:action="@{/addtocart}" method="post">
                                                        <input type="hidden"
name="redirect" value="/phones"/>
                                                        <input type="hidden"
name="quantity" value="1" />
                                                        <input type="hidden"
name="phone_id" th:value="${phone.id}" />
                                                        <div class="form-group
row m-2">
                                                                <button
class="btn btn-danger col form-control m-1" type="submit">
                                                                <!--
<span
class="fa fa-cart-plus"></span>  -->
                                                                <span
class=""> Add to Cart</span>
                                                                </button>
                                                                <a
```

```
th:href="@{'/phones/'+${phone.id}}" class="btn btn-secondary col form-control m-
1">Details</a>
                                                                    </div>
                                                                </form>
                                                            </div>
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                            <div class="mt-auto">
                                            <footer th:replace="fragments::footer()" />
                                        </div>
                                    </div>
                                    </body>
                                    </html>
```

- **/resources/templates/phone-details.html**

```
<!DOCTYPE html>
<html>
<head th:replace="fragments::page_head('Phone Detail Page')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('')" />
        <div th:replace="fragments::main_navigation('')" />

        <div class="container mt-4">
                <div th:if="${message}" class="alert alert-success">
                        <th:block th:utext="${message}"></th:block>
                </div>

                <div class="row">
                        <div class="col-lg-4 d-flex border p-2 rounded phone-
detail-img">
                                <img class="img-responsive m-auto" th:src="$
{phone.imagePath}" alt="Phone Photo" />
                        </div>
                        <div class="col">
                                <a class="m-3 btn btn-link text-dark btn-back"
th:href="@{/phones}"><span class="fa fa-angle-left"></span> back</a>
                                <h1>[[${phone.title}]] </h1>
                                <div class="phone_desc mb-4">
                                        <th:block th:utext="$
{phone.description}"></th:block>
                                </div>
                                <h3 class="text-dark text-bold">Rs. [[$
{phone.price}]]</h3>
                                <form class="form d-flex align-items-center"
th:action="@{/addtocart}" method="post">
                                        <input type="hidden" name="redirect"
th:value="${redirect}" />
                                        <input type="hidden" name="phone_id"
```

```
th:value="${phone.id}" />
                                        <div class="mr-3" style="width: 150px;">
                                            <input type="text"
name="quantity" value="1" class="form-control text-center"
                                                   data-bts-button-
down-class="btn btn-primary"
                                                   data-bts-button-up-
class="btn btn-primary"/>
                                        </div>
                                        <button class="btn btn-danger btn-lg"
type="submit">Add to Cart</button>
                                    </form>

                                    <div class="feedback-ratings mt-4">
                                        <h4>Feedback & Rating</h4>
                                        <div class="alert alert-info" th:if="$
{(feedbacks==null || feedbacks.size()<1)}">No feedbacks found for this Phone!!!
</div>
                                        <div th:if="${feedbacks!=null}"
th:each="feedback : ${feedbacks}" class="feedback card p-3 mb-3 d-flex flex-
row">
                                            <div class="bg-dark text-light
rounded-circle mr-3 h4" style="display:flex;justify-content:center;align-
items:center;min-width:50px;height:50px">
                                                [[$
{#strings.toUpperCase(#strings.substring(feedback.user.username,0,1))}]]
                                            </div>
                                            <div class="flex-grow-1">
                                                <div
class="h6"><spanclass =" username " >[[$
{#strings.toUpperCase(feedback.user.username)}]]</span>
                                                    <span
class="float-right rate">
                                                        <span
th:class="${(index<feedback.rate.value)?'text-warning fa fa-star':'fa fa-star
text-secondary'}" th:each="index: ${#numbers.sequence(1,5)}" ></span>
                                                    </span>
                                                </div>
                                                <div
class="user_feedback_msg pl-2">[[${feedback.user_feedback_msg}]]</div>
                                            </div>

                                        </div>
                                    </div>
                                </div>
                            </div>
        </div>
        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
```

```
        <link rel="stylesheet" href="/css/jquery.bootstrap-touchspin.min.css" />
        <script src="/js/jquery.bootstrap-touchspin.min.js"></script>
        <script>
            $("input[name='quantity']").TouchSpin();
        </script>
</body>
</html>
```

- **/resources/templates/orderstatus.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Order Status')" />
<body class="d-flex flex-column">
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />

        <div class="container mt-4">
                <div th:if="${!message.isEmpty()}" class="alert alert-info">
                        [[${message}]]
                </div>

                <div class="row">
                        <div id="order-details" class="offset-lg-3 border p-4
col-lg-6 shadow rounded" th:with="address = ${order.address}">
                                <img th:src="@{/images/logo-sm.png}"/>
                                <h4 class="mt-3 d-flex">Order Receipt Details
                                        <span th:if="$
{payStatus.toString()=='PAID'}" class="badge badge-success ml-auto">PAID</span>
                                        <span th:if="$
{payStatus.toString()=='PENDING'}" class="badge badge-info ml-auto">NOT
PAID</span>
                                        <span th:if="$
{payStatus.toString()=='FAILED'}" class="badge badge-danger ml-auto">FAIL</span>
                                </h4>
                                <p style="position:relative">
                                        <strong>Receipt Id: </strong>txn_[[$
{order.id}]] <br />
                                        <strong>Order Date: </strong>[[$
{#dates.format(order.created_date,'dd MMM yyyy, HH:mm aa')}]] <br />
                                        <strong>Order Status: </strong><span
class="badge badge-dark">[[${order.status}]]</span> <br />
                                        <strong>Payment Method: </strong>[[$
{(order.paymentMethod=="OP")?"Online Payment":"Cash on Delivery"}]] <br />
                                </p>
                                <p th:each="orderItem,status : $
{order.orderItems}" class="item" style="position:relative">
                                        <strong>[[${status.count}]]: [[$
{orderItem.phone.title}]] </strong>
                                        <span>x[[${orderItem.quantity}]] ----
```

```
</span> <span style="float:right">Rs. [[${orderItem.amount}]]</span>
                                    </p>
                                    <p class="text-danger text-right">
                                            <strong>Grand Total: Rs. [[$
{totalPrice}]]</strong>
                                    </p>
                                    <p>
                                            <strong>Address: </strong><br />
                                            [[${
                                                    address.full_name+", "
                                                    +address.address_line_1+", "
                                                    +address.address_line_2+", "
                                                    +address.city+", "
                                                    +address.state+", "
                                                    +address.country+"-"
                                                    +address.zip
                                            }]]
                                    </p>

                                    <a id="print-btn" href="#" class="btn btn-
primary">Print</a>
                            </div>
                    </div>
            </div>
            <div class="mt-auto">
            <footer th:replace="fragments::footer()" />
        </div>
</div>

<script type="text/JavaScript" th:src="@{/js/jQuery.print.js}"></script>
<script>
        $("#print-btn").on("click", function(){
                $("#order-details").print();
                return false;
        });
</script>
</body>
</html>
```

- **/resources/templates/login.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Login Page')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />
        <div class="container mb-4">
                <div class="card shadow mt-5 ml-auto mr-auto p-4">
                        <h2 class="text-center mb-3">Login</h2>
```

```
                        <p class="alert alert-success mt-4" th:if="$
{message}">[[${message}]]</p>
                        <div th:if="${param.error}" class="alert alert-danger">
                                Invalid Username and Password.
                        </div>
                        <div th:if="${param.logout}" class="alert alert-
success">
                                You are logged out.
                        </div>
                        <form class="form" th:action="@{/login}" method="post">

                                <div class="form-group row">
                                        <label class="col-md-3 text-right"
for="username">Username:</label>
                                        <div class="col-md-9">
                                                <input class="form-control"
type="text" name="username" required/>
                                        </div>
                                </div>
                                <div class="form-group row">
                                        <label class="col-md-3 text-right"
for="password">Password:</label>
                                        <div class="col-md-9">
                                                <input class="form-control"
type="password" name="password" required/>
                                        </div>
                                </div>
                                <div class="form-group">
                                        <div class="offset-md-3">
                                                <input class="btn btn-primary"
type="submit" value="Login" />
                                                <span class="ml-4">Do not have
account? <a th:href="@{/register}" class="btn btn-link">Register</a></span>
                                        </div>
                                </div>
                        </form>
                </div>
        </div>

        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
<script>
        $("input[name=username]").focus();
</script>
</body>
</html>
```

- **/resources/templates/home.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Home Page')" />
<body>
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />
        <div id="slider" class="carousel slide" data-ride="carousel">
        <ol class="carousel-indicators">
            <li data-target="#slider" data-slide-to="0" class="active"></li>
            <li data-target="#slider" data-slide-to="1"></li>
        </ol>

        <div class="carousel-inner">
            <div class="carousel-item active banner-1">
                <img class="d-block w-100" src="images/banner.jpg" alt="">
                <h2>30% OFF (First Order)</h2>
                <h3>We work for happy customers <br>
                    129MP Camera<br>
                    4GB RAM/64GB ROM</h3>
            </div>
            <div class="carousel-item banner-2">
                <img class="d-block w-100" src="images/banner-2.jpg" alt="">
                <h2>SHOPPING WITH US</h2>
                <h3>Best Mobile Phones <br>
                    6GB RAM/128 ROM <br>
                    4080MAH Battery</h3>
            </div>
        </div>
    </div>
    <div class="about-us bg-blue mb-5 text-light">
        <div class="container">
            <div class="row">
                <div class="col-4">
                    <img src="images/about-us-banner.jpg" alt="about-us-banner"
class="img-fluid">
                </div>
                <div class="col-8">
                    <h2 class="mt-4 mb-3">About Us</h2>
                    <p>We are one of the best mobile phone manufacters, making
quality technology that is accessible to everyone. We have 100+ offline mobile
phone shops, which are selling in market around the globe. It was popularised in
the 1960s with the release of feature phone with 2G technologies, and as time
pass we improved with smart phone manufacturing industry. Our Mobile phones are
best in performance. It is now using more latest processors, most fastest memory
and best battery backup.</p>
                    <a th:href="@{/about}" class="btn btn-danger">Read More</a>
                </div>
            </div>
        </div>
    </div>
    <div class="contact-info container mb-5">
```

```
        <div class="row">
            <div class="col-6">
                <h3 class="text-orange mb-4">Contact Information</h3>
                <div class="row">
                    <div class="col-6">
                        <h4>Address:</h4>
                        <address>
                            UIGO Mobile Phone Pvt. Ltd.
                            123 Street, Sahibabad, Ghaziabad, UP-201005
                        </address>
                        <address>
                            UIGO Manufacting Pvt. Ltd.
                            123 Street, Ice Lake, Utah-89372
                        </address>
                        <nav class="social-links">
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-facebook-f"></span></a>
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-twitter"></span></a>
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-linkedin-in"></span></a>
                        </nav>
                        </Address>
                    </div>
                    <div class="col-6">
                        <h4>Email:</h4>
                        <span class="fa fa-envelope"></span> info@uigopvtltd.com
<br>
                        <span class="fa fa-envelope"></span> contact@uigo.com

                        <h4 class="mt-3">Skype:</h4>
                        <span class="fab fa-skype"></span> uigo.manufactur

                        <h4 class="mt-3">Phone:</h4>
                        <span class="fa fa-phone-square"></span> +91 8393999999
<br>
                        <span class="fa fa-phone-square"></span> +91 8393999998
                    </div>
                </div>
            </div>
            <div class="col-6">
                <h3 class="text-orange mb-4">Contact Us</h3>
                <form id="contact-form" class="">
                    <div class="form-group row">
                        <div class="col-md-6">
                            <input class="form-control" type="text" name="name"
placeholder="Your Name" required>
                        </div>
                        <div class="col-md-6">
                            <input class="form-control" type="text" name="phone"
placeholder="Your Phone No." required>
```

```
                        </div>
                    </div>
                    <div class="form-group row">
                        <div class="col-md-6">
                            <textarea class="form-control" name="message"
placeholder="Your Message" required></textarea>
                        </div>
                        <div class="col-md-6">
                            <input class="form-control mb-3" type="text"
name="email" placeholder="Your Email" required>
                            <button class="btn btn-danger w-100"
type="submit">send</button>
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>

    <footer th:replace="fragments::footer()" />
</body>
</html>
```

- **/resources/templates/fragments.html**

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
        xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity5">
<head th:fragment="page_head(title)">
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta name="viewport" content="width=device-width,initial-
scale=1.0,minimum-scale=1.0">
        <title>[[${title}]]</title>

        <link rel="stylesheet"
th:href="@{/bootstrap/4.6.1/css/bootstrap.min.css}" />
        <link rel="stylesheet" th:href="@{/css/all.css}" />
        <link rel="stylesheet" th:href="@{/css/style.css}" />

        <script th:src="@{/js/jquery.js}" /></script>
        <script th:src="@{/bootstrap/4.6.1/js/bootstrap.bundle.min.js}"
/></script>
</head>
<body>
        <div th:fragment="top_navigation(page)" id="nav-top" class="navbar
navbar-dark bg-dark navbar-expand-lg text-light p-0">
        <!-- <div class="container"> -->
            <a class="navbar-brand ml-3 mr-auto nav-link">MOBILE PHONE SHOPPING
SYSTEM</a>
            <div class="navbar-nav mr-4">
                    <button class="navbar-toggler m-2" type="button" data-
toggle="collapse" data-target="#auth_div" aria-controls="auth_div" aria-
```

```
expanded="false" aria-label="Toggle navigation">
                                    <span class="fa fa-user"></span>
                                </button>
                    <div class="collapse navbar-collapse" id="auth_div">
                        <li sec:authorize="isAnonymous()" th:class="$
{(page=='login')?'navbar-item active':'navbar-item'}">
                                <a class="nav-link"
th:href="@{/login}">login</a>
                        </li>
                        <li sec:authorize="isAuthenticated()" class="navbar-item
dropdown">
                                <a class="nav-link dropdown-toggle" href="#"
role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                                    <span sec:authentication="name"></span>
                                </a>
                                <div class="dropdown-menu">
                                    <a sec:authorize="hasRole('ADMIN')"
class="dropdown-item" th:href="@{/admin}">Admin Dashboard</a>
                                    <a sec:authorize="hasRole('CUSTOMER')"
class="dropdown-item" th:href="@{/customer}">Profile</a>
                                    <form th:action="@{/logout}"
method="post">
                                        <button class="dropdown-
item" type="submit"> Logout</button>
                                    </form>
                                </div>
                        </li>
                        <li th:class="${(page=='register')?'navbar-item
active':'navbar-item'}">
                                <a class="nav-link"
th:href="@{/register}">register</a>
                        </li>
                    </div>
            </div>

            <a class="nav-link text-light btn btn-danger mr-3"
th:href="@{/cart}">
                <span class="fas fa-shopping-cart"></span> cart
            </a>
        <!-- </div> -->
    </div>
]
        <nav th:fragment="main_navigation(page)" id="nav-main" class="navbar
navbar-expand-lg navbar-light bg-light border">
                <div class="container-fluid">
                        <a th:href="@{/}" class="navbar-brand">
                                <img th:src="@{/images/logo-sm.png}"
alt="logo" />
                        </a>
                        <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
```

```
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
                            <span class="navbar-toggler-icon"></span>
                        </button>

                        <div class="collapse navbar-collapse"
id="navbarSupportedContent">
                            <ul class="navbar-nav ml-4 mr-auto">
                                    <li th:class="${(page=='home')?'navbar-
item active':'navbar-item'}">
                        <a class="nav-link" th:href="@{/}">home</a>
                    </li>
                    <li th:class="${(page=='about')?'navbar-item
active':'navbar-item'}">
                        <a class="nav-link" th:href="@{/about}">about</a>
                        </li>
                    <li th:class="${(page=='phone')?'navbar-item
active':'navbar-item'}">
                        <a class="nav-link" th:href="@{/phones}">phones</a>
                        </li>
                    <li th:class="${(page=='contact')?'navbar-item
active':'navbar-item'}">
                        <a class="nav-link" th:href="@{/contact}">contact</a>
                        </li>
                            </ul>
                            <!-- <form class="form-inline text-right"
th:action="@{/<phones></phones>}" method="post">
                    <div class="form-group ml-auto m-2">
                        <input type="text" class="form-control mr-2"
name="search-btn" placeholder="Search Phone..." required />
                        <button class="btn btn-danger" type="submit">
                            <span class="fas fa-search"></span>
                        </button>
                    </div>
                </form> -->
                        </div>
                </div>
        </nav>

        <div th:fragment="customer_menu(page)" class="col-lg-2 text-center bg-
blue rounded p-4">
                <a th:class="${page=='profile'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}" th:href="@{/customer/profile}">User
Profile</a>
                <a th:class="${page=='address'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}"
th:href="@{/customer/address}">Address</a>
                <a th:class="${page=='orders'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}"
th:href="@{/customer/orders}">Orders</a>
                <a th:class="${page=='return'?'nav-link text-white bg-danger
```

```
rounded':'nav-link text-white rounded'}" th:href="@{/customer/return}">Return
Order</a>
                <a th:class="${page=='invoices'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}"
th:href="@{/customer/invoices}">Download Invoices</a>
                <a th:class="${page=='feedback'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}"
th:href="@{/customer/feedback}">Feedback &amp; Rating</a>
                <a th:class="${page=='chats'?'nav-link text-white bg-danger
rounded':'nav-link text-white rounded'}" th:href="@{/customer/chats}">Chats</a>
        </div>

        <footer th:fragment="footer()" class="footer mt-4 bg-dark text-light p-
2">
        <div class="container text-center">
            <p class="m-0">&copy; Copyright to sorabh86.github.io, All Right
Reserved.</p>
        </div>
    </footer>
</body>
</html>
```

- **/resources/templates/contact.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Home Page')" />
<body class="d-flex flex-column">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />

        <div class="container mt-4">
                <h2 class="mb-4">Contact Us</h2>
                <div th:if="${!flash.isEmpty()}" class="alert alert-info">
                        [[${flash}]]
                </div>
                 <div class="row">
                         <div class="col-6">
                <form id="contact-form" class="form" th:action="@{/contact}"
method="post">
                    <div class="form-group">
                                                <label for="name">Your
Name</label>
                        <input class="form-control" type="text" name="name"
placeholder="Please enter your name" required>
                    </div>
                    <div class="form-group">
                        <label for="phone">Your Phone No.</label>
                        <input class="form-control" type="text" name="phone"
placeholder="Your Phone No." required>
                    </div>
                    <div class="form-group">
```

```
                        <label for="email">Your Email Id.</label>
                        <input class="form-control" type="text" name="email"
placeholder="Your Email" required>
                    </div>
                    <div class="form-group">
                        <label for="message">Your Message</label>
                        <textarea class="form-control" rows="4" name="message"
placeholder="Your Message" required></textarea>
                    </div>
                    <button class="btn btn-danger form-control"
type="submit">send</button>
                </form>
            </div>
            <div class="col-6">
                <div class="row">
                    <div class="col-6">
                        <h4>Address:</h4>
                        <address>
                            UIGO Mobile Phone Pvt. Ltd.
                            123 Street, Sahibabad, Ghaziabad, UP-201005
                        </address>
                        <address>
                            UIGO Manufacting Pvt. Ltd.
                            123 Street, Ice Lake, Utah-89372
                        </address>
                        <nav class="social-links">
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-facebook-f"></span></a>
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-twitter"></span></a>
                            <a class="btn btn-primary bg-blue rounded-circle pl-
3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-linkedin-in"></span></a>
                        </nav>
                        </Address>
                    </div>
                    <div class="col-6">
                        <h4>Email:</h4>
                        <span class="fa fa-envelope"></span> info@uigopvtltd.com
<br>
                        <span class="fa fa-envelope"></span> contact@uigo.com

                        <h4 class="mt-3">Skype:</h4>
                        <span class="fab fa-skype"></span> uigo.manufactur

                        <h4 class="mt-3">Phone:</h4>
                        <span class="fa fa-phone-square"></span> +91 8393999999
<br>
                        <span class="fa fa-phone-square"></span> +91 8393999998
                    </div>
                </div>
            </div>
```

```
        </div>
        </div>
        <footer th:replace="fragments::footer()" />
</body>
</html>
```

- **/resources/templates/checkout.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Checkout')" />
<body>
        <div th:replace="fragments::top_navigation('')" />
        <div th:replace="fragments::main_navigation('')" />
        <div class="container mt-4">

                <h2 class="mb-3">Checkout</h2>
                <div class="row">
                        [[${data.toString()}]]
                </div>
        </div>
        <link rel="stylesheet" href="/css/jquery.bootstrap-touchspin.min.css" />
        <script src="/js/jquery.bootstrap-touchspin.min.js"></script>
        <script>
                $("input[name='quantity']").TouchSpin();
        </script>
</body>
</html>
```

- **/resources/templates/cart.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Shopping Cart')" />
<body>
        <div class="d-flex flex-column" style="min-height: 100vh">
                <div th:replace="fragments::top_navigation(${page})" />
                <div th:replace="fragments::main_navigation(${page})" />

                <div class="container mt-4 mb-4">
                        <h2 class="d-flex mb-3">
                                Your Shopping Cart <span th:if="${!
cartItems.isEmpty()}"
                                        class="text-danger ml-auto">Total: <span
class="total-price">[[${totalPrice}]]</span>
                                </span>
                        </h2>
                        <div class="row">
                                <form id="checkout_form" name="checkout_form"
                                        th:action="@{/checkout}" method="post"
class="col">
```

```
                                        <!--
<input type="hidden" th:name="${_csrf.parameterName}" th:value="$
{_csrf.token}"/> -->
                                        <div th:if="${cartItems.isEmpty()}"
class="alert alert-warning">
                                            Your Shopping Cart is Empty!!!
<a th:href="@{/phones}">Go for
                                                Shopping</a>
                                        </div>

                                        <div th:if="${!cartItems.isEmpty()}"
class="d-flex flex-column">
                                            <div class="cart-list mb-4">
                                                <th:block th:each="item,
status : ${cartItems}">
                                                    <div
class="cart-item row border rounded m-2 p-2">
                                                        <div
class="d-sm-none d-md-flex col-1 m-auto">

<strong class="text-dark">[[${status.count}]]</strong>
                                                        </div>
                                                        <div
class="col-2 m-auto d-flex cart-item-img">

<img th:src="@{${item.phone.imagePath}}"

class="img-responsive m-auto" alt="Phone Image" />
                                                        </div>
                                                        <div
class="col mt-auto mb-auto">

<a th:href="@{${'/phones/'+item.phone.id}}">[[${item.phone.title}]]</a>
                                                        </div>
                                                        <div
class="col-lg-2 m-auto">

<div style="max-width: 180px">

<input class="item_id" type="hidden"

th:name="${'cartItems['+status.count+'].item_id'}"

th:value="${item.id}" /> <input class="phone_id"

type="hidden"

th:name="${'cartItems['+status.count+'].phone_id'}"

th:value="${item.phone.id}" /> <input class="amount"
```

```
type="hidden"

th:name="${'cartItems['+status.count+'].amount'}"

th:value="${item.phone.price}" /> <input

class="text-center touch-spin quantity" type="text"

th:name="${'cartItems['+status.count+'].quantity'}"

th:value="${item.quantity}"

data-bts-button-down-class="btn btn-secondary rounded-0"

data-bts-button-up-class="btn btn-secondary rounded-0" />

</div>
                                                                    </div>
                                                                    <div
class="col m-auto">

X <span class="price">[[${item.phone.price}]]</span> <strong

class="d-block text-danger">#Sub-Total: <span

class="subtotal-price">[[${item.phone.price*item.quantity}]]</span>

</strong>
                                                                    </div>
                                                                    <div
class="col-lg-1 m-auto text-right">

<a th:href="@{${'/removetocart/'+item.phone.id}}"

class="remove btn btn-light text-dark p-2"> <span

class="fa fa-trash"></span>

</a>
                                                                    </div>
                                                                </div>
                                                            </th:block>
                                                        </div>

                                                        <div class="address-list row">
                                                            <div class="col">
                                                                <h4>Choose
Delivery Address</h4>
                                                                <th:block
th:each="address, status : ${addresses}">
                                                                    <div
```

```
class="card d-block m-2 p-3">

<label class="d-flex"> <input class="mt-auto mb-auto"

type="radio" name="address_id" th:value="${address.id}" />

</td> <span class="ml-3"> <span class="full_name">[[$
{address.full_name}]]</span>

<span class="address_line_1">[[${address.address_line_1}]]</span>

<span class="address_line_2">[[${address.address_line_2}]]</span>

<span class="city">[[${address.city}]]</span> <span

class="state">[[${address.state}]]</span> <span

class="country">[[${address.country}]]</span> <span

class="zip">[[${address.zip}]]</span>

</span>

</label>
                                                                    </div>
                                                                </th:block>
                                                                <div class="card
d-block m-2 p-3">
                                                                    <label
class="d-flex"> <input

class="mt-auto mb-auto new_address" type="radio"

name="address_id" value="0" checked />
                                                                    </td>
<span class="ml-3 d-inline-block">Fill up new Address</span>
                                                                    </label>
                                                                </div>

                                                                <div
class="your-message ml-2 mr-2 mt-4">
                                                                    <h4>Your
Message</h4>

<textarea name="message" id="" cols="30" rows="10"

class="form-control"></textarea>
                                                                </div>
                                                            </div>
                                                            <div id="address-form"
                                                                class="col-lg
```

```
border p-3 rounded mt-3 ml-4 mr-4">
                                                    <h4>Fill up
Address Details:</h4>
                                                    <div
class="form-group">

<label>Full Name: </label> <input name="full_name"

class="form-control" type="text" placeholder="Your Full Name"

required />
                                                    </div>
                                                    <div
class="form-group">

<label>phone:</label> <input name="phone" class="form-control"

type="text" placeholder="Address Line 1" required />
                                                    </div>
                                                    <div
class="form-group">

<label>Address Line 1:</label> <input name="address_line_1"

class="form-control" type="text" placeholder="Address Line 1"

required />
                                                    </div>
                                                    <div
class="form-group">

<label>Address Line 2:</label> <input name="address_line_2"

class="form-control" type="text" placeholder="Address Line 2" />
                                                    </div>
                                                    <div
class="form-group">

<label>City:</label> <input name="city" class="form-control"

type="text" placeholder="Your City" required />
                                                    </div>
                                                    <div
class="form-group">

<label>State:</label> <input name="state" class="form-control"

type="text" placeholder="Your State" required />
                                                    </div>
                                                    <div
class="form-group">
```

```
<label>Country:</label> <input name="country"

class="form-control" type="text" placeholder="Your Country"

required />
                                                                    </div>
                                                                    <div
class="form-group">

<label>Zip Code:</label> <input name="zip" class="form-control"

type="text" placeholder="Zip Code" required />
                                                                    </div>
                                                                </div>
                                                            </div>

                                                            <div class="payment-options ml-2
mr-2 mt-4 border p-4 row">
                                                                <div class="col">
                                                                    <h4>Choose
Payment Method:</h4>
                                                                    <label class="d-
block"> <input type="radio"

name="payment-method" value="COD" checked /> Cash on Delivery
                                                                    </label> <label
class="d-block"> <input type="radio"

name="payment-method" value="OP" /> Online Payment
                                                                    </label>
                                                                </div>
                                                                <div class="col text-
right">
                                                                    <button
type="submit" class="btn btn-danger ml-auto">Checkout/Payment</button>
                                                                </div>
                                                            </div>

                                                        </div>
                                                    </form>
                                                </div>
                                            </div>
                                            <div class="mt-auto">
                                                <footer th:replace="fragments::footer()" />
                                            </div>
                                        </div>

                                        <div id="paymentStatusWindow" class="modal fade show" aria-modal="true"
role="dialog">
                                            <div class="modal-dialog modal-dialog-centered">
```

```
                        <div class="modal-content">
                                <div class="modal-header text-light bg-dark">
                                        <h5 class="modal-title">Order
Status</h5>
                                        <button type="button" class="close text-
light" data-dismiss="modal" aria-label="Close">
                                          <span aria-hidden="true">&times;</span>
                                </button>
                                </div>
                                <div class="modal-body">
                                        <div class="success icon text-center">
                                                <span class="fa fa-check-circle
fa-3x text-success"></span>
                                        </div>
                                        <div class="failure icon text-center">
                                                <span class="fa fa-times-circle
fa-3x text-danger"></span>
                                        </div>
                                        <div class="message text-center h4 mt-3
font-weight-bold border border-success p-3"></div>
                                </div>
                                <div class="modal-footer border-0 text-center
pt-0 pb-4">
                                        <button class="ok-btn btn btn-primary
pl-4 pr-4 font-weight-bold m-auto">OK</button>
                                </div>
                        </div>
                </div>
        </div>

        <div id="loading">
                <div class="spinner-border text-light"
                        style="position: fixed; left: 50%; top: 50%">
                        <span class="sr-only">Loading...</span>
                </div>
        </div>

        <link rel="stylesheet" href="/css/jquery.bootstrap-touchspin.min.css" />
        <script src="/js/jquery.bootstrap-touchspin.min.js"></script>
        <script src="https://checkout.razorpay.com/v1/checkout.js"></script>
        <script src="/js/payment-razorpay.js"></script>

        <script>
                /** UPDATE TOTAL PRICE **/
                var $cartItems = $(".cart-item"), $totalPrice = $(".total-
price");
                function updatePrice() {
                        var totalPrice = 0;
                        for (var i = 0; i < $cartItems.length; i++) {
                                var $cartItem = $($cartItems[i]);
```

```
                                var amount = $cartItem.find(".amount").val();
                                var quantity =
$ cartItem.find(".quantity").val();
                                totalPrice += (amount * quantity);
                        }
                        $totalPrice.html(totalPrice);
                }

                var $touchInput = $("input.touch-spin");
                $touchInput.TouchSpin({
                        min : 1,
                        max : 10
                });

                $touchInput.change(function() {
                        var $this = $(this), $parent = $this.parent().parent()
                        item_id = $parent.find(".item_id").val(), quantity = $
(this).val();

                        $.post({
                                url : "/cart/update",
                                method : "post",
                                data : {
                                        _csrf : $("input[name='_csrf']").val(),
                                        item_id : item_id,
                                        quantity : quantity
                                },
                                success : function(data) {
                                        console.log(data);
                                }.bind(this),
                                error : function(error) {
                                        console.error(error);
                                }.bind(this)
                        })
                        updatePrice();
                });

                $(function() {
                        var $addressForm = $("#address-form");
                        var addressFormChild = $addressForm.html();

                        $("input[name='address_id']").on("change", function() {
                                var $this = $(this), value = $this.val();

                                if (value == "0") {
                                        $addressForm.show();
                                        $addressForm.html(addressFormChild);
```

```
                                } else {
                                        $addressForm.hide();
                                        $addressForm.empty();
                                }
                        });
                });

                // IF Online payment mode selected then start ajax payment
                var $checkoutForm = $("#checkout_form");
                $checkoutForm.on("submit", function() {
                        var paymentMethod =
document.forms.checkout_form['payment-method'].value;

                        if (paymentMethod == 'OP') {
                                // convert form data to json
                                var checkoutForm = document.forms.checkout_form;
                                var formData = new FormData(checkoutForm);
                                var jsonData =
Object.fromEntries(formData.entries());
                                jsonData.amount = $totalPrice.html();
                                console.log("form data:", jsonData);
                                startPayment(jsonData);

                                return false;
                        }
                });
        </script>
</body>
</html>
```

- **/resources/templates/about.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Home Page')" />
<body>
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />

        <h2 class="mt-4 mb-4 text-center">About Us</h2>
        <div class="bg-blue pt-3 pb-3 pl-5 pr-5">
                <img class="card-img" th:src="@{/images/warehouse-banner.jpg}"
alt="About Banner" />
        </div>

        <div class="container-fluid row">
                <div class="col-lg-6">
                        <p class="m-5">We are one of the best mobile phone
manufacters, making quality technology that is accessible to everyone.
                        We have 100+ offline mobile phone shops, which are
selling in market around the globe.
```

```
                        It was popularised in the 1960s with the release of
feature phone with 2G technologies,
                        and as time pass we improved with smart phone
manufacturing industry. Our Mobile phones are
                        best in performance. It is now using more latest
processors, most fastest memory and best battery backup.</p>
                </div>
                <div class="col-lg-6">
                        <div class="row mt-4">
                            <div class="col-6">
                                <h4>Address:</h4>
                                <address>
                                    UIGO Mobile Phone Pvt. Ltd.
                                    123 Street, Sahibabad, Ghaziabad, UP-201005
                                </address>
                                <address>
                                    UIGO Manufacting Pvt. Ltd.
                                    123 Street, Ice Lake, Utah-89372
                                </address>
                                <nav class="social-links">
                                    <a class="btn btn-primary bg-blue rounded-
circle pl-3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-facebook-f"></span></a>
                                    <a class="btn btn-primary bg-blue rounded-
circle pl-3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-twitter"></span></a>
                                    <a class="btn btn-primary bg-blue rounded-
circle pl-3 pr-3 pt-2 pb-2" href="#"><span class="fab fa-linkedin-
in"></span></a>
                                </nav>
                                </Address>
                            </div>
                            <div class="col-6">
                                <h4>Email:</h4>
                                <span class="fa fa-envelope"></span>
info@uigopvtltd.com <br>
                                <span class="fa fa-envelope"></span>
contact@uigo.com

                                <h4 class="mt-3">Skype:</h4>
                                <span class="fab fa-skype"></span>
uigo.manufactur

                                <h4 class="mt-3">Phone:</h4>
                                <span class="fa fa-phone-square"></span> +91
8393999999 <br>
                                <span class="fa fa-phone-square"></span> +91
8393999998
                            </div>
                        </div>
                </div>
        </div>
```

```
        <footer th:replace="fragments::footer()" />
</body>
</html>
```

- **/resources/templates/error/500.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('404 Error')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation(${page})" />
        <div th:replace="fragments::main_navigation(${page})" />


    <div class="container mt-4">
        <h1>500 Server Error</h1>
        <p>Something Went wrong</p>
        <h3 th:text="${errorCode}">500</h3>
        <p th:utext="${errorMessage}">Error java.lang.NullPointerException</p>
        <a href="/" th:href="@{/}">Back to Home Page</a>
    </div>

    <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
</body>
</html>
```

- **/resources/templates/error/404.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('404 Error')" />
<body>
        <div class="d-flex flex-column" style="min-height:100vh">
                <div th:replace="fragments::top_navigation(${page})" />
                <div th:replace="fragments::main_navigation(${page})" />


            <div class="container mt-4">
                <h1>404 Not found</h1>
                <h3 th:text="${errorCode}">404</h3>
                <p th:utext="${errorMessage}">Error
java.lang.NullPointerException</p>
                <a href="/" th:href="@{/}">Back to Home Page</a>
            </div>

            <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
            </div>
        </div>
```

```
</body>
</html>
```

- **/resources/templates/error/403.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('404 Error')" />
<body>
        <div class="d-flex flex-column" style="min-height:100vh">
                <div th:replace="fragments::top_navigation(${page})" />
                <div th:replace="fragments::main_navigation(${page})" />


            <div class="container mt-4">
                <h1>403 Forbidden</h1>
                <h3 th:text="${errorCode}">404</h3>
                <p th:utext="${errorMessage}">Error
java.lang.NullPointerException</p>
                <a href="/" th:href="@{/}">Back to Home Page</a>
            </div>

            <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
            </div>
        </div>
</body>
</html>
```

- **/resources/templates/customer/returnorder.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments :: page_head('Customer | Return Order')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('customer')" />
        <div th:replace="fragments::main_navigation('customer')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('return')"></div>
                        <div class="col">
                                <div th:if="${message!=null}" class="alert
alert-success">
                                        [[${message}]]
                                </div>

                                <h3 class="border-bottom pb-3 mb-4">Return
Order</h3>

                                <div class="alert alert-success" th:if="$
{message}">[[${message}]]</div>
```

```
                                    <table class="table table-striped table-
bordered">
                                        <thead class="thead-dark">
                                            <tr>
                                                <th>Id</th>
                                                <th>Order Date</th>
                                                <th>Order Details</th>
                                                <th>Return Status</th>
                                                <th></th>
                                            </tr>
                                        </thead>
                                        <tbody>
                                            <tr th:each="order: ${orders}"
th:class="${order.isVisited?'bg-visted order order-'+order.id:'order
order-'+order.id}" >
                                                <td>[[${order.id}]]</td>
                                                <td class="order-date
text-center">
                                                    <div
class="date-date">[[${#dates.format(order.created_date,"dd/MM/yy
hh:mm")}]]</div>
                                                </td>
                                                <td class="order-items"
th:with="orderItems = ${order.orderItems}">
                                                    <span class="m-0
item" style="position:relative" th:each="orderItem, status : ${orderItems}">
                                                        [[$
{status.count+".   "+orderItem.phone.title+"    X"+orderItem.quantity+" =
"+orderItem.amount}]]
                                                        <br
th:if="${status.count!=order.orderItems.size()}" />
                                                    </span>
                                                    <strong
class="total-amount text-danger float-right d-block">[[$
{order.grandTotal}]]</strong>
                                                </td>
                                                <td>
                                                    <span th:if="$
{order.returnOrders.isEmpty}">Not Requested</span>
                                                    <span th:if="${!
order.returnOrders.isEmpty}">[[${order.returnOrders.get(0).status}]]</span>
                                                </td>
                                                <td>
                                                    <a th:if="$
{order.isReturnable && order.returnOrders.isEmpty}" class="return-btn text-
primary" th:data-id="${order.id}" href="#">Return</a>
                                                    <span th:if="${!
order.isReturnable}">Return time expired</span>
                                                    <span th:if="${!
order.returnOrders.isEmpty}">Already Submitted</span>
                                                </td>
```

```
                                                    </tr>
                                                </tbody>
                                            </table>
                                        </div>
                                    </div>
                                </div>

                                <div class="mt-auto">
                                    <footer th:replace="fragments::footer()" />
                                </div>
                            </div>
                            <div id="orderreturnwindow" class="modal" tabindex="-1" role="dialog">
                                <div class="modal-dialog modal-lg modal-dialog-centered"
                                    role="document">
                                    <div class="modal-content">
                                        <div class="modal-header bg-dark text-light">
                                            <h3>Return Order</h3>
                                            <button type="button" class="close text-light"
                                                data-dismiss="modal"
                                                    aria-label="Close">
                                                    <span aria-hidden="true">&times;</span>
                                            </button>
                                        </div>
                                        <div class="modal-body">
                                            <form name="returnForm" method="post"
                                                th:action="@{/customer/return/save}">
                                                    <input type="hidden" name="orderId" />
                                                    <input type="hidden" name="status"
                                                        value="REQUESTED" />
                                                    <input type="hidden"
                                                        name="returnReason"/>
                                                    <div class="form-group row">
                                                        <label class="col-4 col-form-
                                                            label text-right">Choose your reason:</label>
                                                        <div class="col-8">
                                                            <select
                                                                name="complaintype" class="form-control" required>
                                                                <option
                                                                    value="">Please choose</option>
                                                                <option>Damaged
                                                                    Item Received</option>
                                                                <option>Missing
                                                                    Component in package</option>
                                                            </select>
                                                        </div>
                                                    </div>
                                                    <div class="form-group row">
                                                        <label class="col-4 col-form-
                                                            label text-right">Describe Problem:</label>
                                                        <div class="col-8">
                                                            <textarea name="reason"
```

```
class="form-control" rows="4" required></textarea>
                                                        <small class="text-
info">We will send our agents based on situation found : <br />
                                                        * We will fixed your
item <br />
                                                        * We will replace/repair
your item <br />
                                                        * We will return and
refund your money. <br />
                                                        You will be contacted as
soon as possible.</small>
                                                    </div>
                                                </div>
                                                <div class="form-group row">
                                                    <label class="col-4 col-form-
label"></label>
                                                    <div class="col-8">
                                                        <button type="submit"
class="btn btn-primary mr-3">Submit</button>
                                                        <button type="button"
data-dismiss="modal" class="btn btn-secondary">Cancel</button>
                                                    </div>
                                                </div>
                                            </form>
                                        </div>
                                    </div>
                                </div>
                            </div>

                            <div id="loading">
                                <div class="spinner-border text-light"
                                    style="position: fixed; left: 50%; top: 50%">
                                    <span class="sr-only">Loading...</span>
                                </div>
                            </div>

                            <script>
                                var $orderreturnwindow = $("#orderreturnwindow");
                                var returnForm = document.forms.returnForm;

                                // show return modal
                                $(".return-btn").on("click", function(){
                                    returnForm['orderId'].value = $(this).data("id");
                                    $orderreturnwindow.modal();
                                    return false;
                                });

                                // submit return form
                                $(returnForm).on("submit", function(){
                                    if(returnForm.complaintype.value!='' && returnForm.reason.value!
```

```
='') {
                        returnForm['returnReason'].value =
returnForm.complaintype.value+" : "+returnForm.reason.value;
                        return true;
                }
                return false;
        });
</script>
</body>
</html>
```

- **/resources/templates/customer/profile.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Customer | Profile')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('customer')" />
        <div th:replace="fragments::main_navigation('customer')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('profile')"></div>
                        <div class="col pl-4">
                                <h2 class="pb-3">Your Profile</h2>
                                <div class="border p-3 mb-3 rounded">
                                        <h4 class="border-bottom pb-3 mb-
3">Basic Details</h4>
                                        <p class="mb-2"><strong>Username:
</strong>[[${customer.username}]]</p>
                                        <p class="mb-2"><strong>Full Name:
</strong>[[${customer.full_name}]]</p>
                                        <p class="mb-2"><strong>Email Id:
</strong>[[${customer.email}]]</p>
                                        <p class="mb-2"><strong>Phone No:
</strong>[[${customer.phone_no}]]</p>
                                </div>
                                <div th:if="${customer.addresses.size()<=0}"
class="address-list border p-3 mb-3 rounded">
                                        No address found, Please <a
th:href="@{/customer/address}">add address</a>
                                </div>
                                <div th:if="${customer.addresses.size()>0}"
class="address-list border p-3 mb-3 rounded">
                                        <h4 class="border-bottom pb-3 mb-3">Your
Address</h4>
                                        <div class="address pb-2"
th:each="address, status : ${customer.addresses}">
                                                [[${status.count}]]. <strong>[[$
{address.full_name}]]</strong>, [[${address.address_line_1}]], [[$
{address.address_line_2}]],
```

```
                                                    [[${address.city}]], [[$
{address.state}]], [[${address.country}]]-[[${address.zip}]]
                                            </div>
                                    </div>
                            </div>
                            <div class="col-lg-3 bg-danger rounded mb-auto">
                                    <div th:if="${message}" class="mt-3 alert alert-
success">
                                            [[${message}]]
                                    </div>
                                    <form name="passwordForm" class="p-4"
th:action="@{/customer/profile/password}" method="post" novalidate>
                                            <h2 class="text-white mb-3">Change
Password</h2>
                                            <div class="form-group">
                                                    <label class="text-
light">Current Password</label>
                                                    <input name="curpassword"
type="password" class="form-control" placeholder="Current Password" required />
                                                    <div class="invalid-feedback
alert-danger pl-2 pr-2">
                                                    Please type your current
password
                                            </div>
                                            </div>
                                            <div class="form-group">
                                                    <label class="text-light">New
Password</label>
                                                    <input name="newpassword"
type="password" class="form-control" placeholder="New Password" required />
                                                    <div class="invalid-feedback
alert-danger pl-2 pr-2">
                                                    <strong>Please type your new
password</strong> <br /> * Your Current password must not same to New password.
                                            </div>
                                            </div>
                                            <div class="form-group">
                                                    <label class="text-light">Retype
New Password</label>
                                                    <input name="newpassword2"
type="password" class="form-control" placeholder="Retype New Password" />
                                                    <div class="invalid-feedback
alert-danger pl-2 pr-2">
                                                    Your password didn't match
                                            </div>
                                            </div>
                                            <input type="submit" class="btn btn-
primary" value="Update" />
                                    </form>
                            </div>
                    </div>
```

```
        </div>

        <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
        </div>
</div>

<script>
var passwordForm = document.forms.passwordForm,
        $passwordForm = $(passwordForm);

$passwordForm.find("input[type=password]").on("keyup", function(){
        $(this).removeClass('is-invalid');
});
$passwordForm.on("submit", function() {
        var curpassword = passwordForm['curpassword'].value,
                newpassword = passwordForm['newpassword'].value,
                newpassword2 = passwordForm['newpassword2'].value;

        if(curpassword=='' || newpassword=='' || newpassword2=='' ||
curpassword==newpassword || newpassword!=newpassword2) {

                if(curpassword=='') {
                        $(passwordForm['curpassword']).addClass('is-invalid');
                }
                if(newpassword=='' || curpassword==newpassword) {
                        $(passwordForm['newpassword']).addClass('is-invalid');
                }
                if(newpassword2=='' || newpassword!=newpassword2) {
                        $(passwordForm['newpassword2']).addClass('is-invalid');
                }

                console.log(passwordForm['curpassword'].value,
passwordForm['newpassword'].value);
                return false;
        }
});
</script>
</body>
</html>
```

- **/resources/templates/customer/orders.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Customer | Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('orders')" />
        <div th:replace="fragments::main_navigation('orders')" />
```

```
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('orders')"></div>
                        <div class="col pl-4">
                                <h2>Your Orders</h2>
                                <div class="alert alert-success" th:if="${message}">[[${message}]]</div>
                                <table class="table">
                                        <thead class="thead-dark">
                                                <tr>
                                                        <th>Id</th>
                                                        <th>Order Date</th>
                                                        <th>Order Details</th>
                                                        <th>Status</th>
                                                        <th class="d-none">Address</th>
                                                        <th>Options</th>
                                                </tr>
                                        </thead>
                                        <tbody>
                                                <tr th:each="order: ${orders}" th:class="${order.isVisited?'bg-visted order order-'+order.id:'order order-'+order.id}" >
                                                        <td>[[${order.id}]]</td>
                                                        <td class="order-date">
                                                                <div class="date-time">[[${#dates.format(order.created_date,"hh:mm a")}]]</div>
                                                                <div class="date-date">[[${#dates.format(order.created_date,"dd MMM, yyyy")}]]</div>
                                                                <small class="date-delivery badge badge-success">Expected: <span class="days">[[${order.expected_delivery_days==null?0:order.expected_delivery_days}]]</span> Days</small>
                                                        </td>
                                                        <td class="order-items" th:with="orderItems = ${order.orderItems}">
                                                                <p class="m-0 item" style="position:relative" th:each="orderItem, status : ${orderItems}">

<span>[[${status.count}]].</span>

<span>[[${orderItem.phone.title}]]</span>
                                                                        <span style="float:right" class="text-secondary">

... x<span class="item-quantity">[[${orderItem.quantity}]]</span>

<span class="item-amount text-danger">[[${orderItem.amount}]]</span>
                                                                        </span>
                                                                </p>
```

```
                                                    <hr class="m-
1" />
                                                    <strong
class="total-amount text-danger text-right d-block">[[$
{order.grandTotal}]]</strong>
                                                </td>
                                                <td class="text-center">
                                                    <!-- Payment
Method -->
                                                    <div th:if="$
{order.paymentMethod=='COD'}" class="payment-method font-weight-bold">CASH ON
DELIVERY</div>
                                                    <div th:if="$
{order.paymentMethod=='OP'}" class="payment-method font-weight-bold">ONLINE
PAYMENT</div>

                                                    <!-- Order
Status -->
                                                    <div th:if="$
{order.status.toString()=='REQUESTED'}" class="status badge badge-primary">[[$
{order.status}]]</div>
                                                    <div th:if="$
{order.status.toString()=='APPROVED'}" class="status badge badge-warning">[[$
{order.status}]]</div>
                                                    <div th:if="$
{order.status.toString()=='SHIPPED'}" class="status badge badge-info">[[$
{order.status}]]</div>
                                                    <div th:if="$
{order.status.toString()=='REJECTED'}" class="status badge badge-danger">[[$
{order.status}]]</div>
                                                    <div th:if="$
{order.status.toString()=='DELIVERED'}" class="status badge badge-success">[[$
{order.status}]]</div>

                                                    <div
class="payment-status">
                                                        <!--
CASH ON DELIVERY -->
                                                        <span
th:if="${(order.paymentMethod=='COD' && order.status.toString()=='DELIVERED')}"
class="pay-status badge badge-success">PAID</span>
                                                        <span
th:if="${order.paymentMethod=='COD' && order.status.toString()!='DELIVERED'}"
class="pay-status badge badge-secondary">NOT PAID</span>

                                                        <!--
ONLINE -->
                                                        <span
th:if="${(order.paymentMethod=='OP' && order.payments.size()>0)

&& order.payments.get(0).status.toString=='PENDING'}" class="pay-status badge
```

```
badge-info">PENDING</span>
                                                                    <span
th:if="${(order.paymentMethod=='OP' && order.payments.size()>0)

&& order.payments.get(0).status.toString=='PAID'}" class="pay-status badge
badge-success">PAID</span>
                                                                    <span
th:if="${order.paymentMethod=='OP' && order.payments.isEmpty}" class="badge
badge-secondary">NOT PAID</span>
                                                                </div>
                                                            </td>
                                                            <td class="address d-
none" th:with="adr = ${order.address}" th:utext="$
{adr.full_name+'<br>'+adr.address_line_1+
                                                                ',
'+adr.address_line_2+'<br>'+adr.city+' '+adr.state+',
'+adr.country+'-'+adr.zip}"></td>
                                                            <td>
                                                                <span th:if="$
{order.status.toString != 'REJECTED' &&
                                                                    !
(order.paymentMethod=='OP' && !order.payments.isEmpty &&
order.payments.get(0).status.toString == 'PAID')}">
                                                                    <a
class="pay-btn" th:data-orderid="${order.id}"

th:data-id="${(!order.payments.isEmpty && order.payments.size()&gt;=0)?
order.payments.get(0).razorOrderId:''}"

th:data-amount="${order.grandTotal}"

th:data-username="${order.address.full_name}"

th:data-email="${user.email}"

th:data-phone="${order.address.phone}"

href="#">Pay</a> |
                                                                </span>
                                                                <a class="view-
btn" th:data-id="${order.id}" href="#">view</a>
                                                            </td>
                                                        </tr>
                                                    </tbody>
                                                </table>

                                            </div>
                                        </div>
                                    </div>
                                    <div class="mt-auto">
                                        <footer th:replace="fragments::footer()" />
```

```
        </div>
</div>

<div id="orderdetailwindow" class="modal" tabindex="-1" role="dialog">
        <div class="modal-dialog modal-lg modal-dialog-centered"
role="document">
                <div class="modal-content">
                        <div class="modal-header bg-dark text-light">
                                <h3 class="modal-title">Order Details</h3>
                                <button type="button" class="close text-light"
data-dismiss="modal"
                                        aria-label="Close">
                                        <span aria-hidden="true">&times;</span>
                                </button>
                        </div>
                        <div class="modal-body">
                                <div class="order-date d-flex mb-3 justify-
content-between h5"></div>
                                <div class="order-items">
                                        <div class="order-item">
                                                <div class="row">
                                                        <div class="col-md-3">
                                                                <div
class="confirm process">
                                                                        <div
class="tracking">REQUESTED</div>
                                                                        <div
class="tracking">APPROVED</div>
                                                                        <div
class="tracking">SHIPPED</div>
                                                                        <div
class="tracking">DELIVERED</div>
                                                                </div>
                                                                <div
class="confirm reject">
                                                                        <div
class="tracking">REJECTED</div>
                                                                </div>
                                                        </div>
                                                        <div class="col-md-1">
                                                                <div
class="order-mark-join odr-1 mark"><span class="cborder"></span></div>
                                                                <div
class="order-mark-join odr-2 mark"><span class="cborder"></span></div>
                                                                <div
class="order-mark-join odr-3 mark"><span class="cborder"></span></div>
                                                                <div
class="order-mark odr-4 mark" ></div>
                                                        </div>
                                                        <div class="col">
                                                                <div class="bg-
```

```
light text-dark card mb-2 p-2 text-center">

<strong>Your Order Status:</strong>
                                                                    <span
class="status"></span>
                                                                    <span
class="pay-status"></span>
                                                                </div>
                                                                <div class="mb-3
order-cancel">
                                                                    <a
class="btn btn-primary" href="#">Cancel Order</a>
                                                                    <small
class="text-primary">You can cancel the order before approval.</small>
                                                                </div>
                                                            </div>
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                            <div class="modal-footer d-flex">
                                                <div class="flex-grow-1">
                                                    <h4>Shipping Details</h4>
                                                    <address class="address">
                                                        Sorabh <br />
                                                        D23D, D block Janakpuri,
Shalimar Garden Sahibabad Ghaziabad Uttar Pradesh-201005, India
                                                    </address>
                                                </div>
                                                <div class="flex-grow-1">
                                                    <h4>Price Details</h4>
                                                    <div class="items h5">
                                                        <div class="item">
                                                            1. UIGO Mobile ... x1
<span class="ml-auto" style="float:right">Rs 24,000</span> <br />
                                                            Base price ... x1   Rs
20,000 <br /> Tax Rs 4,000
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>

<div id="ordercancelwindow" class="modal" tabindex="-1" role="dialog">
                            <div class="modal-dialog modal-dialog-centered" role="document">
                                <div class="modal-content">
                                    <div class="modal-header bg-dark text-light">
                                        <h3>Cancel Order</h3>
                                        <button type="button" class="close text-light"
```

```
data-dismiss="modal"
                                        aria-label="Close">
                                        <span aria-hidden="true">&times;</span>
                                </button>
                        </div>
                        <div class="modal-body">
                                Are you sure? <br> Your Order will be deleted
permanently.
                        </div>
                        <div class="modal-footer">
                                <a class="delete-btn btn btn-primary mr-
3">Yes</a>
                                <button type="button" data-dismiss="modal"
class="btn btn-secondary">No</button>
                        </div>
                </div>
        </div>
</div>

<div id="paymentStatusWindow" class="modal fade show" aria-modal="true"
role="dialog">
        <div class="modal-dialog modal-dialog-centered">
                <div class="modal-content">
                        <div class="modal-header text-light bg-dark">
                                <h5 class="modal-title">Order Status</h5>
                                <button type="button" class="close text-light"
data-dismiss="modal" aria-label="Close">
                                  <span aria-hidden="true">&times;</span>
                        </button>
                        </div>
                        <div class="modal-body">
                                <div class="success icon text-center">
                                        <span class="fa fa-check-circle fa-3x
text-success"></span>
                                </div>
                                <div class="failure icon text-center">
                                        <span class="fa fa-times-circle fa-3x
text-danger"></span>
                                </div>
                                <div class="message text-center h4 mt-3 font-
weight-bold border border-success p-3"></div>
                        </div>
                        <div class="modal-footer border-0 text-center pt-0 pb-
4">
                                <button class="ok-btn btn btn-primary pl-4 pr-4
font-weight-bold m-auto">OK</button>
                        </div>
                </div>
        </div>
</div>
```

```
<div id="loading">
        <div class="spinner-border text-light"
                style="position: fixed; left: 50%; top: 50%">
                <span class="sr-only">Loading...</span>
        </div>
</div>

<script src="https://checkout.razorpay.com/v1/checkout.js"></script>
<script src="/js/payment-razorpay.js"></script>
<script type="text/JavaScript" th:src="@{/js/jQuery.print.js}"></script>
<script>
// RETRY RAZOR PAYMENT
$(".pay-btn").on("click", function(){
        var $this = $(this),
                orderId = $this.data("orderid"),
                razorOrderId = $this.data("id"),
                amount = $this.data("amount"),
                username = $this.data("username"),
                email = $this.data("email"),
                phone = $this.data("phone");

        console.log(orderId, razorOrderId, amount, username, email, phone);
        if(!razorOrderId || razorOrderId == '') {
                // create razor order from server
                $.ajax({
                        url: '/cart/payment',
                        type: 'post',
                        data:{
                                id: orderId,
                                amount: amount
                        },
                        dataType:'json',
                        beforeSend: function(){
                        $('#loading').show();
                    }, success: function(d) {
                        console.log('success', d);
                        // proced payment to razor order
                                createRazorPaymentWindow({
                                        id: d.id,
                                        amount: d.amount,
                                        username: d.username,
                                        email: d.email,
                                        phone: d.phone
                                });
                        }, error: function(e) {
                                console.log('error', e)
                        }, complete: function(){
                        $('#loading').hide();
```

```
                    }
                })
                console.log("create razor order based on order");
        } else {
                // proced payment to razor order
                createRazorPaymentWindow({
                        id: razorOrderId,
                        amount: amount,
                        username: username,
                        email: email,
                        phone: phone
                });
        }
        return false;
});


//Show order details
var $orderdetailwindow = $("#orderdetailwindow"),
        $ordercancelwindow = $("#ordercancelwindow"),
        $orderinvoicewindow = $("#orderinvoicewindow");

window.so = window.so || {};

$(".view-btn").on("click", function() {
        var $this = $(this),
                $parent = $this.parent().parent(),
                id = $this.data('id');

        so.address = $parent.find(".address").html();
        so.status = $parent.find(".status").html();
        so.paymentMethod = $parent.find(".payment-method").html();
        so.paymentStatus = $parent.find(".payment-status").html();
        so.orderItems = $parent.find(".order-items").html();
        so.orderDate = $parent.find(".order-date").html();

        // get current date compare with calculated date after 15 days on order;
        var todayDate = new Date();
        var orderDate = new Date($parent.find(".order-date .date-
date").html());
        orderDate.setDate(orderDate.getDate()+15);

        $orderdetailwindow.find(".confirm").hide();

        var $orderCancel = $orderdetailwindow.find(".order-cancel").hide(),
                $orderInvoice = $orderdetailwindow.find(".order-
invoice").hide();
```

```
        var $mark = $orderdetailwindow.find(".mark");
        if(so.status.trim() == "REJECTED") {
                $orderdetailwindow.find(".confirm.process").hide();
                $orderdetailwindow.find(".confirm.reject").show();
                $mark.hide();
                $orderdetailwindow.find(".odr-4").addClass("active").show();
        } else {
                $orderdetailwindow.find(".confirm.process").show();
                $orderdetailwindow.find(".confirm.reject").hide();
                $orderdetailwindow.find(".mark").removeClass("active");
                $mark.show();

                if(so.status.trim() == "REQUESTED") {
                        $orderdetailwindow.find(".odr-1").addClass("active");
                        $orderCancel.show();
                } else if(so.status.trim() == "APPROVED") {
                        $orderdetailwindow.find(".odr-1").addClass("active");
                        $orderdetailwindow.find(".odr-2").addClass("active");
                } else if(so.status.trim() == "SHIPPED") {
                        $orderdetailwindow.find(".odr-1").addClass("active");
                        $orderdetailwindow.find(".odr-2").addClass("active");
                        $orderdetailwindow.find(".odr-3").addClass("active");

                } else if(so.status.trim().search("DELIVERED") != -1) {
                        $orderdetailwindow.find(".odr-1").addClass("active");
                        $orderdetailwindow.find(".odr-2").addClass("active");
                        $orderdetailwindow.find(".odr-3").addClass("active");
                        $orderdetailwindow.find(".odr-4").addClass("active");

                        $orderInvoice.show();
                }
        }

        $orderdetailwindow.find(".order-date").html(so.orderDate);
        $orderdetailwindow.find(".address").html(so.address);
        $orderdetailwindow.find(".items").html(so.orderItems);
        $orderdetailwindow.find(".status").html(so.status+",
"+so.paymentStatus+", "+so.paymentMethod);
        $orderdetailwindow.data("id", id);
        $orderdetailwindow.modal();

        if($orderCancel.is(":visible")) {
                $orderCancel.find(".btn").one('click', function(){
```

```
                        $orderdetailwindow.modal("hide");

                        $ordercancelwindow.data("id", id);
                        $ordercancelwindow.modal();
                        console.log("clicked cancel order");
                        return false;
                    });
            }

            return false;
});

/* CANCEL ORDER REQUESTED */
$ordercancelwindow.find(".delete-btn").on("click", function(){
            var id = $ordercancelwindow.data("id");
            window.location.href = "/customer/orders/{id}/delete";
            console.log("order_id "+id);
            return false;
});


</script>
</body>
</html>
```

- **/resources/templates/customer/order_detail.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Profile Page')" />
<body>
<div class="container">
            <div th:replace="fragments::main_navigation()" />

            <h2>Customer Order Details</h2>
</div>
</body>
</html>
```

- **/resources/templates/customer/invoices.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments::page_head('Customer | Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
            <div th:replace="fragments::top_navigation('orders')" />
            <div th:replace="fragments::main_navigation('orders')" />
            <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                    <div class="row flex-grow-1">
                            <div
```

```
th:replace="fragments::customer_menu('invoices')"></div>
                        <div class="col pl-4">
                                <h2>Your Invoices</h2>
                                <div class="alert alert-success" th:if="${message}">[[${message}]]</div>
                                <table class="table table-striped table-bordered">
                                        <thead class="thead-dark">
                                                <tr>
                                                        <th>Id</th>
                                                        <th>Order Date</th>
                                                        <th>Order Details</th>
                                                        <th>Status</th>
                                                        <th>Address</th>
                                                        <th>Options</th>
                                                </tr>
                                        </thead>
                                        <tbody>
                                                <tr th:each="order: ${orders}" th:class="${order.isVisited?'bg-visted order order-'+order.id:'order order-'+order.id}" >
                                                        <td>[[${order.id}]]</td>
                                                        <td class="order-date">
                                                                <div class="date-time">[[${#dates.format(order.created_date,"hh:mm a")}]]</div>
                                                                <div class="date-date">[[${#dates.format(order.created_date,"dd MMM, yyyy")}]]</div>
                                                                <small class="date-delivery badge badge-success">Expected: <span class="days">[[${order.expected_delivery_days==null?0:order.expected_delivery_days}]]</span> Days</small>
                                                        </td>
                                                        <td class="order-items" th:with="orderItems = ${order.orderItems}">
                                                                <small>
                                                                        <p class="m-0 item" style="position:relative" th:each="orderItem, status : ${orderItems}">

<span>[[${status.count}]].</span>

<span>[[${orderItem.phone.title}]]</span>

<span style="float:right" class="text-secondary">

... x<span class="item-quantity">[[${orderItem.quantity}]]</span>

<span class="item-amount text-danger">[[${orderItem.amount}]]</span>

</span>
                                                                        </p>
```

```
                                                                    <hr
class="m-1" />
                                                                    <strong
class="total-amount text-danger text-right d-block">[[$
{order.grandTotal}]]</strong>
                                                                </small>
                                                            </td>
                                                            <td class="text-center">
                                                                <div
class="payment-method">[[${order.paymentMethod=='OP'?'Online Payment':'Cash on
Delivery'}]]</div>
                                                                <span th:if="$
{order.status.toString()=='REQUESTED'}" class="status badge badge-primary">[[$
{order.status}]]</span>
                                                                <span th:if="$
{order.status.toString()=='APPROVED'}" class="status badge badge-warning">[[$
{order.status}]]</span>
                                                                <span th:if="$
{order.status.toString()=='SHIPPED'}" class="status badge badge-info">[[$
{order.status}]]</span>
                                                                <span th:if="$
{order.status.toString()=='REJECTED'}" class="status badge badge-danger">[[$
{order.status}]]</span>
                                                                <span th:if="$
{order.status.toString()=='DELIVERED'}" class="status badge badge-success">[[$
{order.status}]]
                                                                    <span
th:if="${order.delivery_date}" class="delivery-date">
                                                                    <br
/>[[${#dates.format(order.delivery_date, "dd MMM yyyy")}]]</span>
                                                                </span>
                                                                <span th:if="$
{order.status.toString()=='CANCELLED'}" class="status badge badge-danger">[[$
{order.status}]]</span>
                                                                <div
class="payment-status">
                                                                    <span
class="badge badge-secondary">[[${

(order.paymentMethod=='COD' && order.status.toString()=='DELIVERED')?

'PAID':(

(order.paymentMethod=='OP' && order.payments!=null && order.payments.size()>0)?

(order.payments.get(0).status):

'NOT PAID'

)
                                                                    }]]<
```

```
/span>
                                                                    </div>
                                                                </td>
                                                                <td class="address"
th:with="adr = ${order.address}" th:utext="$
{adr.full_name+'<br>'+adr.address_line_1+
                                                                    ',
'+adr.address_line_2+'<br>'+adr.city+' '+adr.state+',
'+adr.country+'-'+adr.zip}"></td>
                                                                <td>
                                                                    <a
class="invoice-btn text-primary" href="#">Download Invoice</a>
                                                                </td>
                                                            </tr>
                                                        </tbody>
                                                    </table>

                                                </div>
                                            </div>
                                        </div>
                                        <div class="mt-auto">
                                            <footer th:replace="fragments::footer()" />
                                        </div>
                                    </div>

                                    <div id="orderinvoicewindow" class="modal" tabindex="-1" role="dialog">
                                        <div class="modal-dialog modal-lg modal-dialog-centered"
role="document">
                                            <div class="modal-content">
                                                <div class="modal-header bg-dark text-light">
                                                    <h3>Order Invoice</h3>
                                                    <button type="button" class="close text-light"
data-dismiss="modal"
                                                        aria-label="Close">
                                                        <span aria-hidden="true">&times;</span>
                                                    </button>
                                                </div>
                                                <div class="modal-body">
                                                    <div id="order-details">
                                                        <h2>
                                                            <img th:src="@{/images/logo-
sm.png}"/> MOBILE SHOPPING SITE
                                                        </h2>
                                                        <h4 class="mt-3 d-flex">Order Receipt
Details
                                                            <span class="invoice-pay-status
badge badge-success ml-auto">PAID</span>
                                                        </h4>
                                                        <table class="table">
                                                            <tr>
                                                                <td><strong>Receipt Id:
```

```
</strong></td>
                                                    <td>txn_<span
class="invoice-id">3</span></td>
                                                </tr>
                                                <tr>
                                                    <td><strong>Order Date:
</strong></td>
                                                    <td class="invoice-date
d-flex justify-content-between">dd MMM yyyy, HH:mm aa</td>
                                                </tr>
                                                <tr>
                                                    <td><strong>Order
Status: </strong></td>
                                                    <td><span
class="invoice-status">PAID</span></td>
                                                </tr>
                                                <tr>
                                                    <td><strong>Payment
Method: </strong></td>
                                                    <td><span
class="invoice-pay-method">CASH ON Delivery</span></td>
                                                </tr>
                                                <tr>
                                                    <td><strong>Customer
Address: </strong></td>
                                                    <td><span
class="invoice-address">CASH ON Delivery</span></td>
                                                </tr>
                                            </table>
                                            <hr />
                                            <p class="invoice-items h3">
                                                <strong>1: Phone Name </strong>
                                                <span>x1 ---- </span> <span
class="float-left">Rs. 0</span>
                                            </p>
                                        </div>
                                    </div>
                                    <div class="modal-footer">
                                        <a id="print-btn" href="#" class="btn btn-
primary">Print</a>
                                        <button type="button" data-dismiss="modal"
class="btn btn-secondary">Cancel</button>
                                    </div>
                                </div>
                            </div>
                        </div>

<script type="text/JavaScript" th:src="@{/js/jQuery.print.js}"></script>
<script>

//Show order details
```

```
var $orderdetailwindow = $("#orderdetailwindow"),
        $ordercancelwindow = $("#ordercancelwindow"),
        $orderinvoicewindow = $("#orderinvoicewindow");

window.so = window.so || {};

function setSoObj($parent) {
        so.address = $parent.find(".address").html();
        so.status = $parent.find(".status").html();
        so.paymentMethod = $parent.find(".payment-method").html();
        so.paymentStatus = $parent.find(".payment-status").html();
        so.orderItems = $parent.find(".order-items").html();
        so.orderDate = $parent.find(".order-date").html();
}

// ORDER INVOICE GENERATION CODE
$(".invoice-btn").on("click", function() {
        setSoObj($(this).parent().parent());
        var id = $orderdetailwindow.data("id");

        var $paymentStatus = $(so.paymentStatus);
        var payStatus = $paymentStatus.html();
        var $invoicePayStatus = $orderinvoicewindow.find(".invoice-pay-
status");
        console.log(payStatus);
        if(payStatus == 'PAID') {
                $invoicePayStatus.removeClass("badge-danger");
                $invoicePayStatus.addClass("badge-success");
        } else {
                $invoicePayStatus.addClass("badge-danger");
                $invoicePayStatus.removeClass("badge-success");
        }

        $orderinvoicewindow.find(".invoice-id").html(id);
        $invoicePayStatus.html(payStatus);
        $orderinvoicewindow.find(".invoice-
status").html(so.status.replace("<br>", " on "));
        $orderinvoicewindow.find(".invoice-address").html(so.address);
        $orderinvoicewindow.find(".invoice-pay-method").html(so.paymentMethod);
        $orderinvoicewindow.find(".invoice-items").html(so.orderItems);
        $orderinvoicewindow.find(".invoice-date").html(so.orderDate);
        $orderinvoicewindow.modal();
});
```

```
$("#print-btn").on("click", function(){
        $("#order-details").print();
        return false;
});


</script>
</body>
</html>
```

- **/resources/templates/customer/feedback.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments :: page_head('Customer | Feedback')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('customer')" />
        <div th:replace="fragments::main_navigation('customer')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('feedback')"></div>
                        <div class="col">
                                <div th:if="${message!=null}" class="alert
alert-success">
                                        [[${message}]]
                                </div>
                                <div class="mb-3 d-flex">
                                        <h3 class="mr-3">Customer Feedback on
delivered phones</h3>
                                </div>

                                <div class="feedback-content">
                                        <div class="alert alert-warning"
th:if="${orders==null || orders.size()<1}">No Feedback Available.</div>

                                        <div class="orders">
                                                <div class="order card mb-2
border-bottom-0 shadow"
                                                        th:each="order:$
{orders}">
                                                        <h4 class="p-3">Order
Id: txn_[[${order.id}]]</h4>
                                                        <div class="ml-3
orderitem d-flex align-items-center border-bottom"

th:each="orderitem, status:${order.orderItems}">
                                                                <span class="h4
p-3">[[${status.count}]]</span>
                                                                <a th:href="@{$
```

```
{'/phones/'+orderitem.phone.id}}" style="min-width:80px;height:80px" class="d-
flex mr-3 p-2">
                                                                    <img
class="img-responsive m-auto" th:src="@{${orderitem.phone.imagePath}}"
alt="Phone Image" />
                                                                </a>
                                                                <div class="h3
flex-grow-1">
                                                                    <a
class="text-dark" th:href="@{${'/phones/'+orderitem.phone.id}}" >
                                                                    <span
class="title">[[${orderitem.phone.title}]]</span>
                                                                    <small
class="date badge badge-success"> delivery on [[$
{#dates.format(order.delivery_date, "dd-MMM-yyyy")}]]</small>
                                                                    </a>
                                                                </div>
                                                                <a
class="feedback-btn btn btn-secondary mr-3" th:data-oid="${order.id}"
                                                                    th:data-
pid="${orderitem.phone.id}" href="#">Feedback</a>
                                                            </div>
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>

                <div class="mt-auto">
                        <footer th:replace="fragments::footer()" />
                </div>
</div>

                <div id="feedback-window" class="modal fade">
                  <div class="modal-dialog">
                <form name="feedbackForm" th:action="@{/customer/feedback/save}"
method="post" novalidate>
<!--                            <input type="hidden" th:name="${_csrf.parameterName}"
th:value="${_csrf.token}" /> -->
                            <input type="hidden" name="phone_id"  />
                            <input type="hidden" name="order_id"  />
                            <input type="hidden" name="id" />
                        <div class="modal-content">
                            <div class="modal-header bg-dark text-light">
                                    <h4 class="modal-title">Feedback</h4>
                                    <button type="button" class="close text-light"
data-dismiss="modal">
                                                <span aria-
hidden="true">&times;</span><span class="sr-only">Close</span>
                                    </button>
```

```
                        </div>
                        <div class="modal-body">
                                <div class="rate form-group">
                                        <div class="warning">
                                                <span th:class="${'p-0
btn fa fa-star fa-3x text-secondary star-'+index}"
                                                                th:data-
index="${index}"

th:each="index: ${#numbers.sequence(1,5)}" ></span>
                                                <small>Click on star to
give rating.</small>
                                        </div>
                                                <input type="hidden" name="rate"
/>
                                                <div class="invalid-feedback">
                                                Please rate this phone.
                                            </div>
                                        </div>
                                        <div class="form-group">
                                                <label>Write something about
phone</label>
                                                <textarea name="feedback_msg"
id="feedback_msg" class="form-control"
                                                                placeholder="Please
write your review" required
                                                                cols="30"
rows="10"></textarea>
                                                <div class="invalid-feedback">
                                                Please provide your feedback.
                                        </div>
                                        </div>
                                </div>
                        <div class="modal-footer border-0 row p-0 pb-3 ml-3 mr-3
">
                                <button id="submit-btn" type="submit" class="btn
btn-primary col">Submit</button>
                                <button type="button" class="btn btn-secondary
col" data-dismiss="modal">Cancel</button>
                        </div>
                    </div><!-- /.modal-content -->
            </form>
          </div><!-- /.modal-dialog -->
        </div><!-- /.modal -->

        <div id="loading">
                <div class="spinner-border text-light"
                        style="position: fixed; left: 50%; top: 50%">
                        <span class="sr-only">Loading...</span>
                </div>
        </div>
```

```
        <script>
                function checkIfAlreadySubmitted(pid, oid) {
                        $.post({
                                url:"/customer/feedback/"+pid+"/"+oid,
                                type:"POST",
                                data:{_csrf:$('input[name=_csrf]').val()},
                                dataType:"json",
                                beforeSend: function(){
                                $('#loading').show();
                            },success:function(s){
                                if(s.status) {
                                        document.forms.feedbackForm['id'].value
= s.id;

document.forms.feedbackForm['feedback_msg'].value = s.msg;

document.forms.feedbackForm['rate'].value = s.rate;
                                        $(".star-"+s.rate).click();
                                }
                                }, error:function(e){
                                        $feedbackWindow.modal("hide");
                                        console.error("Server Error", e);
                                }, complete: function(){
                                $('#loading').hide();
                            }
                        })
                }

                // feedback icon click feature
                var $feedbackWindow = $("#feedback-window"),
                        $stars = $feedbackWindow.find(".fa-star");

                $(".feedback-btn").on("click", function() {
                        var $this = $(this),
                                orderId = $this.data("oid"),
                                phoneId = $this.data("pid");

                        // reset feedbackFrom
                        document.forms.feedbackForm['id'].value = '';
                document.forms.feedbackForm['feedback_msg'].value = '';
                document.forms.feedbackForm['rate'].value = '';
                $stars.removeClass("text-warning");

                        // check if feedback submitted
                        checkIfAlreadySubmitted(phoneId, orderId);

                        $feedbackWindow.data("order_id", orderId);
```

```
                        $feedbackWindow.data("phone_id", phoneId);


$ feedbackWindow.find("input[name=order_id]").val(orderId);

$ feedbackWindow.find("input[name=phone_id]").val(phoneId);

                        $feedbackWindow.modal();
                        return false;
                });

                $stars.on("click", function(){
                        var $this = $(this),
                                $parent = $(this).parent().parent(),
                                rate = $this.data("index");

                        $stars.removeClass("text-warning");
                        for(var i=1; i<=rate; i++) {
                                var $temp = $parent.find(".star-"+i);
                                $temp.addClass("text-warning");
                        }

                        var $rate = $parent.find("input[name=rate]");
                        $rate.val(rate);
                        $rate.parent().find(".warning").removeClass("rounded
border border-danger");
                        $rate.removeClass('is-invalid');
                });

                /** FEEDBACK FORM VALIDATION **/
                var $feedbackMsg =
$ feedbackWindow.find("textarea[name='feedback_msg']");
                $feedbackMsg.on("input",function(){
                        if(this.value=="") $feedbackMsg.addClass('is-invalid');
                        else $feedbackMsg.removeClass('is-invalid');
                });

                $(document.forms.feedbackForm).on("submit", function(){
                        var form = document.forms.feedbackForm,
                                $rate = $(form['rate']),
                                $feedbackMsg = $(form['feedback_msg']),
                                rate = form['rate'].value,
                                msg = form['feedback_msg'].value,
                                orderId = form['order_id'].value,
                                phoneId = form['phone_id'].value;
```

```
                        console.log($rate, $feedbackMsg, rate, msg, orderId,
phoneId);

                        if(rate=="" || msg=="" || orderId=="" || phoneId=="") {
                                if(rate=="") {

$rate.parent().find(".warning").addClass("rounded border border-danger");
                                        $rate.addClass('is-invalid');
                                }
                                if(msg==""){
                                        $feedbackMsg.addClass('is-invalid');
                                }

                                return false;
                        }
                });

//              $("#submit-btn").on("click", function(){
//                      var $rate = $feedbackWindow.find("input[name=rate]"),
//                              orderId = $feedbackWindow.data("order_id"),
//                              phoneId = $feedbackWindow.data("phone_id"),
//                              _csrf =
$ feedbackWindow.find("input[name=_csrf]"),
//                              rate = $rate.val(),
//                              msg = $feedbackMsg.val();

//                      if(rate=="" || msg=="" || !orderId || orderId=="" || !
phoneId || phoneId=="") {
//                              if(rate=="") {
//
$rate.parent().find(".warning").addClass("rounded border border-danger");
//                                      $rate.addClass('is-invalid');
//                              }
//                              if(msg==""){
//                                      $feedbackMsg.addClass('is-invalid');
//                              }
//                      } else  {
//                              var reqData = {
//                                      "order_id":orderId,
//                                      "phone_id":phoneId,
//                                      "rate":rate,
//                                      "feedback_msg":msg,
//                                      "_csrf": _csrf
//                              };

//                              reqData = JSON.stringify(reqData);
```

```
//                                  $.ajax({
//                                      url:"/customer/feedback/save",
//                                      type:'post',
// //                                   header:{
// //                                       "X-CSRFToken": _csrf,
// // //                                                "Content-Type":
"application/json"
// //                                   },
// //                                   contentType: "application/json; charset=utf-
8",
// //                                   crossDomain: true,
//                                      dataType: "json",
// //                                   contentType: false,
// //                                   processData: false,
//                                      data: reqData,
//                                      success:function(s){
//                                          console.log("success", s)
//                                      }, error:function(e){
//                                          console.log("error", e)
//                                      }
//                                  });
//                              }
//                              return false;
//                          });
        </script>
</body>
</html>
```

- **/resources/templates/customer/chats.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Customer | Chats')" />
<body>
<link rel="stylesheet" th:href="@{/css/chat.css}" />
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('chat')" />
        <div th:replace="fragments::main_navigation('chat')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('chats')"></div>
                        <div class="col">
                                <div th:if="${message!=null}" class="alert
alert-success">
                                        [[${message}]]
                                </div>
                                <div class="d-flex flex-column">
                                        <h3 class="mr-3">Customer Chats</h3>
                                        <div class="d-none">
                                                <a th:data-name="$
```

```
{user.username}"
                                                    href="#" id="start-btn"
class="btn btn-primary">Start Chatting</a>
                                            </div>
                                            <div class="chat-box d-flex">
                                        <div class="member-list w-25 border d-flex
flex-column">
                                            <div class="member m-3 active" data-
href="chatroom">Public Chat Room
                                                    <div class="chat-status
badge badge-light"><span class="fa fa-circle text-danger"> Offline</span></div>
                                            </div>
                                            <h4 class="text-center m-0 alert alert-
secondary">Admin</h4>
                                            <ul class="member-content p-3 flex-grow-
1">
                                                <th:block th:each="username : $
{allUsers}">
                                                    <li th:if="${user.username!
=username}" th:class="${'member '+username}" th:data-href="${username}">
                                                        [[${username}]]
                                                        <span th:class="$
{#lists.contains(onlineUsers, username)?
                                                            'user-status
rounded-circle bg-success':'user-status rounded-circle bg-danger'}"></span>
                                                    </li>
                                                </th:block>
                                            </ul>
                                        </div>
                                        <div class="chat-rooms w-75 ml-3 border p-3
d-flex">
                                            <!-- Public Chat window -->
                                            <div class="chat-content chatroom
active">
                                                <ul class="chat-messages border
flex-grow-1">
                                                    <th:block
th:each="message : ${publicMessages}">
                                                        <li th:class="$
{user.username==message.username? 'message self':'message'}">
                                                            <div
th:if="${user.username} ne ${message.username}" class="avatar">

[[${message.username}]] <br />

<small class="badge date">[[${#dates.format(message.date, "dd/MM/YY")}]]</small>
                                                            </div>
                                                            <div
class="message-data">[[${message.message}]]</div>
                                                            <div
th:if="${user.username} eq ${message.username}" class="avatar self">
```

```
[[${message.username}]] <br />

<small class="badge">[[${#dates.format(message.date, "dd/MM/YY")}]]</small>
                                                                                </div>
                                                                            </li>
                                                                        </th:block>
                                                                    </ul>
                                                                    <div class="send-message d-
flex">
                                                                        <input type="text"
class="input-message form-control border-secondary rounded-0 flex-grow-1"
                                                                               placeholder="enter the
message" value="" />
                                                                        <button type="button"
class="send-button btn btn-primary rounded-0">send</button>
                                                                    </div>
                                                                </div>

                                                                <th:block th:each="username : $
{allUsers}">
                                                                    <div th:class="${'chat-content
'+username}">
                                                                    <ul class="chat-messages border
flex-grow-1">
                                                                    </ul>
                                                                    <div class="send-message d-
flex">
                                                                        <input type="text"
class="input-message form-control border-secondary rounded-0 flex-grow-1"
                                                                               placeholder="enter the
message" value="" />
                                                                        <button type="button"
class="send-button btn btn-primary rounded-0">send</button>
                                                                    </div>
                                                                </div>
                                                                </th:block>
                                                            </div>
                                                        </div>
                                                        </div>

                                                    </div>
                                                </div>
                                            </div>

                                            <div class="mt-auto">
                                                <footer th:replace="fragments::footer()" />
                                            </div>
                                        </div>

                                        <script th:src="@{/js/1.1.4/sockjs.min.js}"></script>
```

```
<script th:src="@{/js/2.3.3/stomp.min.js}"></script>
<script th:src="@{/js/chat.js}"></script>

</body>
</html>
```

- **/resources/templates/customer/address_form.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments :: page_head('Customer | Address')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('customer')" />
        <div th:replace="fragments::main_navigation('customer')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
th:replace="fragments::customer_menu('address')"></div>
                        <div class="col">
                                <div th:if="${message!=null}" class="alert
alert-success">
                                        [[${message}]]
                                </div>
                                <div th:if="${error!=null}" class="alert alert-
success">
                                        [[${error}]]
                                </div>
                                <h2 class="mb-4">[[${page_title}]]</h2>

                                <form class="form" name="customer_form"
th:action="@{/customer/address/save}" method="post"
                                        th:object="${address}">

                                        <input type="hidden" th:field="*{id}" />
                                        <input type="hidden"
th:field="*{userId}" />

                                        <div class="form-group row">
                                                <label class="col-md-2">Full
Name: </label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{full_name}"

placeholder="Your Full Name"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-2">Phone:
```

```
</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{phone}"

placeholder="Your Phone No"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-2">Address
Line 1:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{address_line_1}" placeholder="Address Line 1"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-2">Address
Line 2:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{address_line_2}" placeholder="Address Line 2"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-
2">City:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{city}" placeholder="Your City"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-
2">State:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{state}" placeholder="Your State"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
```

```
                                                <label class="col-md-
2">Country:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{country}" placeholder="Your Country"/>
                                                </div>
                                        </div>
                                        <div class="form-group row">
                                                <label class="col-md-2">Zip
Code:</label>
                                                <div class="col-md-6">
                                                        <input class="form-
control" type="text"

th:field="*{zip}" placeholder="Zip Code"/>
                                                </div>
                                        </div>

                                        <div class="form-group">
                                                <div class="offset-md-2">
                                                        <input class="btn btn-
primary mr-2" type="submit" value="Save" />
                                                        <a
th:href="@{/customer/address}" class="btn btn-dark" >Cancel</a>
                                                </div>
                                        </div>
                                </form>
                        </div>
                </div>
        </div>

        <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
        </div>
</div>
</body>
</html>
```

- **/resources/templates/customer/address.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments :: page_head('Customer | Address')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="fragments::top_navigation('customer')" />
        <div th:replace="fragments::main_navigation('customer')" />
        <div class="ml-5 mr-5 mt-4 d-flex flex-grow-1">
                <div class="row flex-grow-1">
                        <div
```

```
th:replace="fragments::customer_menu('address')"></div>
                        <div class="col">
                                <div th:if="${message!=null}" class="alert
alert-success">
                                        [[${message}]]
                                </div>
                                <div th:if="${error!=null}" class="alert alert-
success">
                                        [[${error}]]
                                </div>
                                <h3 class="border-bottom mb-4 pb-3">Customer
Addresses
                                        <a class="btn btn-primary" th:href="@{$
{'/customer/address/new'}}">Add Address</a>
                                </h3>
                                <div th:if="${addresses==null}" class="alert
alert-info">[[${addresses}]] No Record found.</div>
                                <table th:if="${addresses}" class="table">
                                        <thead>
                                                <tr>
                                                        <th>Id</th>
                                                        <th>Full Name</th>
                                                        <th>Phone</th>
                                                        <th>Address</th>
                                                        <th></th>
                                                </tr>
                                        </thead>
                                        <tbody>
                                                <tr th:each="address : $
{addresses}">
                                                        <td>[[$
{address.id}]]</td>
                                                        <td>[[$
{address.full_name}]]</td>
                                                        <td>[[$
{address.phone}]]</td>
                                                        <td>[[$
{address.address_line_1}]], [[${address.address_line_2}]], [[${address.city}]],
[[${address.state}]], [[${address.country}]]-[[${address.zip}]]</td>
                                                        <td>
                                                                <a
th:href="@{'/customer/address/'+${address.id}+'/edit'}">Edit</a> |
                                                                <a
class="delete-btn text-danger" th:data-href="@{'/customer/address/'+$
{address.id}+'/delete'}" href="#">Delete</a>
                                                        </td>
                                                </tr>
                                        </tbody>
                                </table>
                        </div>
                </div>
```

```
        </div>

        <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
        </div>
</div>

        <div id="confirm-window" class="modal fade">
          <div class="modal-dialog modal-dialog-centered">
            <div class="modal-content">
              <div class="modal-header bg-dark text-light">
                <h4 class="modal-title">Confirm Action</h4>
                <button type="button" class="close text-light" data-
dismiss="modal">
                        <span aria-hidden="true">&times;</span><span class="sr-
only">Close</span>
                </button>
              </div>
              <div class="modal-body">
                                Are you Sure? <br /> Address will be removed
permanently.
              </div>
              <div class="modal-footer border-0 row p-0 pb-3 ml-3 mr-3 ">
                <button type="button" class="yes-btn btn btn-primary
col">Yes</button>
                <button type="button" class="btn btn-secondary col" data-
dismiss="modal">No</button>
              </div>
            </div><!-- /.modal-content -->
          </div><!-- /.modal-dialog -->
        </div><!-- /.modal -->

        <script>
                var $confirmWindow = $("#confirm-window");
                $(".delete-btn").on("click", function() {
                        var href = $(this).data("href");
                        $confirmWindow.data("href", href);
                        $confirmWindow.modal();
                        return false;
                });
                $confirmWindow.find(".yes-btn").on("click", function(){
                        var href = $confirmWindow.data("href");
                        window.location.href = href;
                });
        </script>
</body>
</html>
```

- **/resources/templates/admin/fragments.html**

```
<!doctype html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
        xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity5">
<head th:fragment="admin_head(title)">
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta name="viewport" content="width=device-width,initial-
scale=1.0,minimum-scale=1.0">
        <title>[[${title}]]</title>

        <link rel="stylesheet"
th:href="@{/bootstrap/4.6.1/css/bootstrap.min.css}" />
        <link rel="stylesheet" th:href="@{/css/all.css}" />
        <link rel="stylesheet" th:href="@{/css/style.css}" />

        <script th:src="@{/js/jquery.js}" /></script>
        <script th:src="@{/bootstrap/4.6.1/js/bootstrap.bundle.min.js}"
/></script>
</head>
<body>
        <div class="navbar navbar-expand-lg bg-dark navbar-dark shadow mb-4"
th:fragment="admin_navigation()">
                <a th:href="@{/}" class="navbar-brand">UIGO MOBILE SHOPPING
SYSTEM</a>
                <div class="navbar-collapse">
                        <ul class="navbar-nav">
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin}">Dashboard</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/users}">Users</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/address}">Address</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/phones}">Phones</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/orders}">Orders</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/payments}">Payments</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/returns}">Return Orders</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
th:href="@{/admin/feedbacks}">Feedbacks</a></li>
                                <li class="navbar-item">
                                        <a class="nav-link"
```

```
th:href="@{/admin/chats}">Chats</a></li>
                        </ul>
                        <ul sec:authorize="isAuthenticated()" class="navbar-nav
ml-auto">
                            <li class="navbar-item dropleft">
                                <a class="nav-link dropdown-toggle"
href="#" id="navbarDropdown" role="button" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
                                    <!-- [[$
{#authentication.getPrincipal().getUsername()}]] -->
                                    <span class="text-uppercase font-weight-bold"
sec:authentication="name"></span> <span class="fa fa-user"></span>
                                </a>
                                <div class="dropdown-menu">
                                    <a class="dropdown-item"
href="#">Profile</a>
                                    <div class="dropdown-divider"></div>
                                <form th:action="@{/logout}"
method="post">
                                        <!-- <input
type="hidden" th:name="${_csrf.parameterName}" th:value="${_csrf.token}" /> -->
                                    <button class="dropdown-item"
type="submit"> Logout</button>
                                    </form>
                                </div>
                            </li>
                        </ul>
                    </div>
                </div>

                <div th:fragment="confirmWindow()" id="confirm-window" class="modal
fade">
                  <div class="modal-dialog modal-dialog-centered">
                    <div class="modal-content">
                      <div class="modal-header text-light bg-dark">
                        <h5 class="modal-title">Confirm Action</h5>
                        <button type="button" class="close text-light" data-
dismiss="modal">
                            <span aria-hidden="true">&times;</span><span class="sr-
only">Close</span>
                        </button>
                      </div>
                      <div class="modal-body">
                        <p class="m-0">Are you Sure? <br /> This will be removed
permanently.</p>
                      </div>
                      <div class="row ml-3 mr-3 mb-3">
                        <button type="button" class="yes-btn btn btn-primary
col">Yes</button>
                        <button type="button" class="ml-3 btn btn-secondary col" data-
dismiss="modal">No</button>
```

```
            </div>
          </div><!-- /.modal-content -->
        </div><!-- /.modal-dialog -->
      </div><!-- /.modal -->
</body>
</html>
```

- **/resources/templates/admin/dashboard.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Dashboard')" />
<body>
      <div class="d-flex flex-column" style="min-height:100vh">
            <div th:replace="admin/fragments :: admin_navigation()" />
            <div class="container">
                  <h2 class="border-bottom pb-3 mb-3">Admin |
Dashboard</h2>
                  <p>Welcome Administrator</p>

                  <div class="row">
                        <div class="col">
                              <div class="card">
                                    <div class="card-body">
                                          <h5 class="card-title
border-bottom pb-2 text-center">Orders</h5>
                                          <p class="card-text">New
Orders: <strong>[[${newOrderCount}]]</strong></p>
                                          <p class="card-
text">Total: <strong>[[${totalOrders}]]</strong></p>
                                          <a
th:href="@{/admin/orders}" class="btn btn-primary">Manage</a>
                                    </div>
                              </div>
                        </div>
                        <div class="col">
                              <div class="card">
                                    <div class="card-body">
                                          <h5 class="card-title
border-bottom pb-2 text-center">Users</h5>
                                          <p class="card-
text">Total: <strong>92</strong></p>
                                          <p class="card-
text">Logged In: <strong>[[${loggedInUsers}]]</strong></p>
                                          <a
th:href="@{/admin/users}" class="btn btn-primary">Manage</a>
                                    </div>
                              </div>
                        </div>
                        <div class="col">
                              <div class="card">
                                    <div class="card-body">
```

```
                                                    <h5 class="card-title
border-bottom pb-2 text-center">Phones</h5>
                                                    <p class="card-
text">Total: <strong>192</strong></p>
                                                    <a
th:href="@{/admin/users}" class="btn btn-primary">Manage</a>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                    <div class="row mt-4">
                                        <div class="col">
                                            <div class="card">
                                                <div class="card-body">
                                                    <h5 class="card-title
border-bottom pb-2 text-center">Chats</h5>
                                                    <p class="card-
text">Total: <strong>392</strong></p>
                                                    <a
th:href="@{/admin/chats}" class="btn btn-primary">Manage</a>
                                                </div>
                                            </div>
                                        </div>
                                        <div class="col">
                                            <div class="card">
                                                <div class="card-body">
                                                    <h5 class="card-title
border-bottom pb-2 text-center">Payments</h5>
                                                    <p class="card-
text">Total: <strong>92</strong></p>
                                                    <a
th:href="@{/admin/payments}" class="btn btn-primary">Manage</a>
                                                </div>
                                            </div>
                                        </div>
                                        <div class="col">
                                            <div class="card">
                                                <div class="card-body">
                                                    <h5 class="card-title
border-bottom pb-2 text-center">Feedbacks</h5>
                                                    <p class="card-
text">Total: <strong>192</strong></p>
                                                    <a
th:href="@{/admin/feedbacks}" class="btn btn-primary">Manage</a>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>

                                <div class="mt-auto">
```

```
                        <footer th:replace="fragments::footer()" />
                </div>
        </div>
</body>
</html>
```

- **/resources/templates/admin/users/users_form.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | User Form')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">

                <h2 class="border-bottom pb-3 mb-4">[[${page_title}]]</h2>

                <form class="form" name="customer_form"
th:action="@{/admin/users/save}" method="post"
                        th:object="${user}">
                        <input type="hidden" th:field="*{id}" />
                        <div class="form-group row">
                                <label class="col-md-2" for="username">Username:
</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
th:field="*{username}"
                                                placeholder="Your Username" required/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2"
for="password">Password:</label>
                                <div class="col-md-6">
                                        <input class="form-control"
type="password" th:field="*{password}"
                                                placeholder="Your Password"/>
                                </div>
                                <small class="text-success">Leave Password
Blank, if you don't want to change it.</small>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2" for="full_name">Full
Name:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
th:field="*{full_name}" placeholder="Your Full Name"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2"
```

```
for="email">Email:</label>
                                    <div class="col-md-6">
                                            <input class="form-control" type="email"
th:field="*{email}" placeholder="Your email Id"/>
                                    </div>
                            </div>
                            <div class="form-group row">
                                    <label class="col-md-2" for="phone_no">Phone
No:</label>
                                    <div class="col-md-6">
                                            <input class="form-control" type="text"
th:field="*{phone_no}" placeholder="Your contact no."/>
                                    </div>
                            </div>
                            <div class="form-group row">
                                    <label class="col-md-2"
for="phone_no">Roles:</label>
                                    <div class="col-md-6">
                                            <select class="form-control"
th:field="*{role}" >
                                                    <option th:each="userrole : $
{T(com.github.sorabh86.uigo.constants.UserRoles).values()}"
                                                            th:value="${userrole}"
                                                            th:text="${userrole}" />
                                            </select>
                                    </div>
                                    <div class="col-md-4 desc">
                                            <small class="text-success">Types of
Users in system.</small>
                                    </div>
                            </div>
                            <div class="form-group row">
                                    <label class="col-md-2"
for="phone_no">Status:</label>
                                    <div class="col-md-6">
                                            <select class="form-control"
th:field="*{status}" >
                                                    <option th:each="userstatus : $
{T(com.github.sorabh86.uigo.constants.UserStatus).values()}"
                                                            th:value="${userstatus}"
                                                            th:text="$
{userstatus}" />
                                            </select>
                                    </div>
                                    <div class="col-md-4 desc">
                                            <small class="text-
success"><strong>DEACTIVATE (0)</strong> : Email not verified,<br />
<strong>ACTIVATED (1)</strong> : Email is Verified,<br />
<strong>DISABLED(2)</strong> : Account is disabled. </small>
                                    </div>
                            </div>
```

```
                        <input type="hidden" th:field="*{activation_key}" />
                        <div class="form-group">
                                <div class="offset-md-2">
                                        <input class="btn btn-primary mr-2"
type="submit" value="Save" />
                                        <a th:href="@{/admin/users}" class="btn
btn-dark" >Cancel</a>
                                </div>
                        </div>
                </form>


        </div>
        <div class="mt-auto">
                <footer th:replace="fragments::footer()" />
        </div>
</div>
</body>
</html>
```

- **/resources/templates/admin/users/users.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Users')" />
<body>
        <div class="d-flex flex-column" style="min-height:100vh">
                <div th:replace="admin/fragments::admin_navigation()" />
                <div class="container">

                        <h2 class="border-bottom pb-3 mb-3">Admin | Users <a
class="btn btn-primary" th:href="@{/admin/users/new}">Add User</a></h2>

                        <div th:if="${message!=null}" class="alert alert-
success">[[${message}]]</div>
                        <div th:if="${error!=null}" class="alert alert-
danger">[[${error}]]</div>

                        <table class="table ">
                                <thead>
                                        <tr>
                                                <th>Id</th>
                                                <th>Username</th>
                                                <th>Email</th>
                                                <th>Full Name</th>
                                                <th>Phone No</th>
                                                <th>Role</th>
                                                <th>Status</th>
                                                <th></th>
                                        </tr>
                                </thead>
                                <tbody>
```

```
                                        <tr th:each="user : ${users}">
                                                <td>[[${user.id}]]</td>
                                                <td>[[${user.username}]]</td>
                                                <td>[[${user.email}]]</td>
                                                <td>[[${user.full_name}]]</td>
                                                <td>[[${user.phone_no}]]</td>
                                                <td>[[${user.role}]]</td>
                                                <td>[[${user.status}]]</td>
                                                <td>
                                                        <a
th:href="@{'/admin/users/'+${user.id}+'/edit'}">Edit</a> |
                                                        <a class="delete-btn
text-danger" th:data-href="@{'/admin/users/'+${user.id}+'/delete'}"
href="#">Delete</a>
                                                </td>
                                        </tr>
                                </tbody>
                        </table>
                </div>

                <div class="mt-auto">
                        <footer th:replace="fragments::footer()" />
                </div>
        </div>

        <div th:replace="admin/fragments::confirmWindow()"></div>

        <script>
                var $confirmWindow = $("#confirm-window");
                $(".delete-btn").on("click", function() {
                        var href = $(this).data("href");
                        $confirmWindow.data("href", href);
                        $confirmWindow.modal();
                        return false;
                });
                $confirmWindow.find(".yes-btn").on("click", function(){
                        var href = $confirmWindow.data("href");
                        window.location.href = href;
                });
        </script>
</body>
</html>
```

- **/resources/templates/admin/return/returns.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Return Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
```

```
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">
                <h2 class="mb-3">Admin | Return Orders</h2>
                <div th:if="${message!=null}" class="alert alert-success">[[${message}]]</div>
                <div th:if="${error!=null}" class="alert alert-danger">[[${error}]]</div>

                <div th:if="${returnOrders==null}" class="alert alert-warning">No Order Found!!!<br> No Order had been requested by customers.</div>
                <table th:if="${returnOrders}" class="table">
                        <thead>
                                <tr>
                                        <th>Id</th>
                                        <th>Order Id</th>
                                        <th>Reason</th>
                                        <th>Status</th>
                                        <th>Modify Date</th>
                                        <th>Create Date</th>
                                        <th></th>
                                </tr>
                        </thead>
                        <tbody>
                                <tr th:each="return : ${returnOrders}">
                                        <td>[[${return.id}]]</td>
                                        <td><a th:href="@{${'/admin/orders/'+return.orderId+'/details'}}">[[${return.orderId}]]</a></td>
                                        <td>[[${return.returnReason}]]</td>
                                        <td class="return-status">[[${return.status}]]</td>
                                        <td class="text-uppercase">
                                                [[${#dates.format(return.modify_date,"dd/MM/yy hh:mm")}]]
                                        </td>
                                        <td class="text-uppercase">
                                                [[${#dates.format(return.created_date,"dd/MM/yy hh:mm")}]]
                                        </td>
                                        <td>
                                                <a class="return-btn" th:data-id="${return.id}" href="#">update</a>
                                        </td>
                                </tr>
                        </tbody>
                </table>
        </div>
        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>
```

```
<div th:replace="admin/fragments::confirmWindow()"></div>

<div id="returnorder-window" class="modal fade">
  <div class="modal-dialog modal-dialog-centered">
    <div class="modal-content">
      <div class="modal-header text-light bg-dark">
        <h5 class="modal-title">Order Return Status</h5>
        <button type="button" class="close text-light" data-dismiss="modal">
                <span aria-hidden="true">&times;</span><span class="sr-
only">Close</span>
        </button>
      </div>
      <div class="modal-body">
        <div id="order-status-form" class="row">
                <form name="returnStatusForm"
th:action="@{/admin/returns/status}" method="post">
                        <input type="hidden" name="id" />
                        <input th:if="${_csrf!=null}" type="hidden" th:name="$
{_csrf.parameterName}" th:value="${_csrf.token}" />
                        <div class="col-12">
                                <h4>Choose Return Order Status:</h4>
                                <label class="d-flex align-items-center"
th:each="returnStatus : $
{T(com.github.sorabh86.uigo.constants.ReturnStatus).values()}">
                                        <input type="radio" class="mr-3 ml-3"
name="returnStatus" th:value="${returnStatus}" th:text="${returnStatus}" />
                                </label>
                        </div>
                        <div class="col-12 mt-3">
                                <div class="row ml-3 mr-3">
                                        <button type="submit" class="update-btn
btn btn-primary col">Update</button>
                                        <button type="button" class="ml-3 btn
btn-secondary col" data-dismiss="modal">Cancel</button>
                                </div>
                        </div>
                </form>
        </div>
      </div>
    </div>
  </div>
</div>

<script>
        // CHANGE RETURN ORDER STATUS
        var $returnOrderWindow = $('#returnorder-window');
        var $returnStatus =
$ returnOrderWindow.find("input[name=returnStatus]");
        var $returnBtn = $(".return-btn");
        var returnStatusForm = document.forms['returnStatusForm'];
```

```
        $returnBtn.on("click", function(){
                var $this = $(this),
                        id = $this.data('id'),
                        $parent = $this.parent().parent(),
                        returnStatus = $parent.find(".return-status").html();
                console.log(returnStatus);

                // remove all previous selection
                $returnStatus.parent().removeClass("bg-warning");
                $returnStatus.removeAttr('checked');

                // select current selection
                var $Ele = $returnStatus.filter("[value="+returnStatus+"]");
                $Ele.parent().addClass("bg-warning");
                $Ele.attr('checked', 'checked');
                $Ele[0].checked = true;

                // show popup window
                $returnOrderWindow.modal();
                $returnOrderWindow.find('input[name=id]').val(id);

                return false;
        });


        // Add Delete Confirmation Logic
        var $confirmWindow = $("#confirm-window");
        $(".delete-btn").on("click", function() {
                var href = $(this).data("href");
                $confirmWindow.data("href", href);
                $confirmWindow.modal();
                return false;
        });
        $confirmWindow.find(".yes-btn").on("click", function(){
                var href = $confirmWindow.data("href");
                window.location.href = href;
        });
</script>

</body>
</html>
```

- **/resources/templates/admin/phone/phone_form.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
```

```
<head th:replace="admin/fragments :: admin_head('Admin | Phone Form')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">

                <h2 class="border-bottom pb-3 mb-4">[[${page_title}]]</h2>

                <form class="form" name="customer_form"
th:action="@{/admin/phones/save}" method="post"
                        enctype="multipart/form-data" th:object="${phone}">
                        <input type="hidden" th:field="*{id}" />
                        <div class="form-group row">
                                <label class="col-md-2">Title: </label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
th:field="*{title}" placeholder="Title of Phone"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Description:</label>
                                <div class="col-md-6">
                                        <textarea class="form-control"
th:field="*{description}" rows="6"
                                                placeholder="Description of
Phone"></textarea>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Image:</label>
                                <div class="col-md-6">
                                        <input type="hidden" th:field="*{image}"
/>
                                        <input class="form-control" type="file"
id="phone_image" name="phone_image"  accept="image/png, image/jpeg"/>
                                </div>
                                <div class="col-md-2">
                                        <img id="preview" th:src="$
{phone.imagePath}" alt="Phone Photo"  height="80" />
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Price:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
th:field="*{price}"/>
                                </div>
                        </div>

                        <div class="form-group">
                                <div class="offset-md-2">
```

```
                                        <input class="btn btn-primary mr-2"
type="submit" value="Save" />
                                        <a th:href="@{/admin/phones}" class="btn
btn-dark" >Cancel</a>
                                </div>
                        </div>
                </form>
        </div>
        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>

        <script>
        (function($){

                /* show preview of image */
                $("#phone_image").on("change", function() {
                        var file = this.files[0],
                                $preview = $("#preview");

                        var fileReader = new FileReader();
                        fileReader.readAsDataURL(file);
                        fileReader.onload = function(e) {
                                $preview.attr("src", e.target.result);
                        }

                });
        })(jQuery);
        </script>
</body>
</html>
```

- **/resources/templates/admin/phone/phone.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Phones')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">

                <h2 class="border-bottom pb-3 mb-4">Admin | Phones <a class="btn
btn-primary" th:href="@{/admin/phones/new}">Add Phone</a></h2>
                <table class="table">
                        <thead>
                                <tr>
                                        <th>Id</th>
                                        <th>Image</th>
                                        <th>Title</th>
```

```
                                        <th>Price</th>
                                        <th></th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr th:each="phone : ${phones}">
                                        <td>[[${phone.id}]]</td>
                                        <td>
                                            <img th:src="${phone.imagePath}"
height="60" />
                                        </td>
                                        <td>[[${phone.title}]]</td>
                                        <td>[[$
{#numbers.formatDecimal(phone.price, 0, 'COMMA', 2, 'POINT')}]]</td>
                                        <td>
                                            <a th:href="@{'/admin/phones/'+$
{phone.id}+'/edit'}">Edit</a> |
                                            <a class="delete-btn text-
danger" th:data-href="@{'/admin/phones/'+${phone.id}+'/delete'}"
href="#">Delete</a>
                                        </td>
                                    </tr>
                                </tbody>
                            </table>
                        </div>
                        <div class="mt-auto">
                            <footer th:replace="fragments::footer()" />
                        </div>
                    </div>

                    <div th:replace="admin/fragments::confirmWindow()"></div>
                    <script>
                        var $confirmWindow = $("#confirm-window");
                        $(".delete-btn").on("click", function() {
                            var href = $(this).data("href");
                            $confirmWindow.data("href", href);
                            $confirmWindow.modal();
                            return false;
                        });
                        $confirmWindow.find(".yes-btn").on("click", function(){
                            var href = $confirmWindow.data("href");
                            window.location.href = href;
                        });
                    </script>


                </body>
                </html>
```

- **/resources/templates/admin/payment/payment.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">
                <h2 class="mb-3">Admin | Online Payments</h2>
                <div th:if="${payments==null}" class="alert alert-warning">No
Payment Records Found!!!</div>
                <table th:if="${payments}" class="table">
                        <thead>
                                <tr>
                                        <th>Id</th>
                                        <th>Order Id</th>
                                        <th>Status</th>
                                        <th>Amount</th>
                                        <th>Razor Order Id</th>
                                        <th>Razor Payment Id</th>
                                        <th>Modify Date</th>
                                        <th>Created Date</th>
                                </tr>
                        </thead>
                        <tbody>
                                <tr th:each="payment : ${payments}">
                                        <td>[[${payment.id}]]</td>
                                        <td>
                                                [[${payment.order_id}]]
                                        </td>
                                        <td>
                                                [[${payment.status}]]
                                        </td>
                                        <td>
                                                [[${payment.amount}]]
                                        </td>
                                        <td>
                                                [[${payment.razorOrderId}]]
                                        </td>
                                        <td>
                                                [[${payment.razorPaymentId}]]
                                        </td>
                                        <td>
                                                [[$
{#dates.format(payment.modify_date,"dd/MM/yy hh:mm a")}]]
                                        </td>
                                        <td>
                                                [[$
{#dates.format(payment.created_date,"dd/MM/yy hh:mm a")}]]
                                        </td>
                                </tr>
```

```
                        </tbody>
                </table>
        </div>
        <div class="mt-auto">
        <footer th:replace="fragments::footer()" />
    </div>
</div>

<div th:replace="admin/fragments::confirmWindow()"></div>


<script>
        // Add Delete Confirmation Logic
        var $confirmWindow = $("#confirm-window");
        $(".delete-btn").on("click", function() {
                var href = $(this).data("href");
                $confirmWindow.data("href", href);
                $confirmWindow.modal();
                return false;
        });
        $confirmWindow.find(".yes-btn").on("click", function(){
                var href = $confirmWindow.data("href");
                window.location.href = href;
        });
</script>

</body>
</html>
```

- **/resources/templates/admin/order/orders.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">
                <h2 class="mb-3">Admin | Orders</h2>
                <div th:if="${orders==null}" class="alert alert-warning">No
Order Found!!!<br> No Order had been requested by customers.</div>
                <table th:if="${orders}" class="table">
                        <thead>
                                <tr>
                                        <th>Id</th>
                                        <th>Order Date</th>
                                        <th>Status</th>
<!--                                    <th>Customer Message</th> -->
                                        <th>Delivery</th>
                                        <th>Grand Total (Rs)</th>
```

```
                                        <th>Payment</th>
                                        <th>Option</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr th:each="order : ${orders}" th:class="${!
order.isVisited?'bg-visted order order-'+order.id:'order order-'+order.id}">
                                        <td>[[${order.id}]]</td>
                                        <td class="text-uppercase">
                                            <small>[[$
{#dates.format(order.created_date,"dd/MM/yy")}]]</small>
                                        </td>
                                        <td>
                                            <small th:class="$
{order.status.toString=='REJECTED'?'badge badge-danger':

(order.status.toString=='DELIVERED')?'badge badge-success':

(order.status.toString=='APPROVED')?'badge badge-warning':

(order.status.toString=='SHIPPED')?'badge badge-info':''}">[[$
{order.status}]]</small>
                                        </td>
<!--                                        <td>[[${order.message}]]</td> -->
                                        <td>
                                            <small class="text-uppercase
badge badge-success">[[${order.paymentMethod=='OP'?'ONLINE':'CASH ON
DELIVERY'}]]</small><br />
                                            <small>[[$
{order.expected_delivery_days==null?'Not
Approved.':order.expected_delivery_days+' Days'}]]</small>
                                            <small th:if="$
{order.delivery_date}" class="text-uppercase"><br />[[$
{#dates.format(order.delivery_date,"dd/MM/yy hh:mm a")}]]</small>
                                        </td>
                                        <td>[[${order.grandTotal}]]</td>
                                        <td>
                                            <span th:if="$
{(order.paymentMethod=='COD' && order.status.toString()=='DELIVERED')}"
class="badge badge-success">PAID</span>
                                            <span th:if="$
{order.paymentMethod=='COD' && order.status.toString()!='DELIVERED'}"
class="badge badge-secondary">NOT PAID</span>

                                            <span th:if="$
{(order.paymentMethod=='OP' && order.payments.size()>0)
                                                &&
order.payments.get(0).status.toString=='PENDING'}" class="badge badge-
info">PENDING</span>
```

```
                                                    <span th:if="${(order.paymentMethod=='OP' && order.payments.size()>0)
                                                                    && order.payments.get(0).status.toString=='PAID'}" class="badge badge-success">PAID</span>
                                                    <span th:if="${order.paymentMethod=='OP' && order.payments.isEmpty}" class="badge badge-secondary">NOT PAID</span>
                                                </td>
                                                <td>
                                                    <a th:href="@{'/admin/orders/'+${order.id}+'/details'}">View</a> |
                                                    <a class="delete-btn text-danger" th:data-href="@{'/admin/orders/'+${order.id}+'/delete'}" href="#">Delete</a>
                                                </td>
                                            </tr>
                                        </tbody>
                                    </table>
                                </div>
                                <div class="mt-auto">
                                <footer th:replace="fragments::footer()" />
                            </div>
                        </div>

                        <div th:replace="admin/fragments::confirmWindow()"></div>

                        <script>
                                // Add Delete Confirmation Logic
                                var $confirmWindow = $("#confirm-window");
                                $(".delete-btn").on("click", function() {
                                        var href = $(this).data("href");
                                        $confirmWindow.data("href", href);
                                        $confirmWindow.modal();
                                        return false;
                                });
                                $confirmWindow.find(".yes-btn").on("click", function(){
                                        var href = $confirmWindow.data("href");
                                        window.location.href = href;
                                });
                        </script>

                </body>
                </html>
```

- **/resources/templates/admin/order/order-details.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Orders')" />
<body>
```

```
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">
                <h2 class="border-bottom pb-3 mb-4">Admin | Order Details</h2>

                <div class="order-details">

                        <div class="row mb-2">
                                <strong class="col-2 text-right">Order
Id :</strong>
                                <span class="col-10">[[${order.id}]]</span>
                        </div>
                        <div class="row mb-2">
                                <strong class="col-2 text-
right">Dates :</strong>
                                <span class="col-10">[[$
{#dates.format(order.created_date,"dd MMM, yyyy hh:mm a")}]]</span>
                        </div>
                        <div class="row mb-2">
                                <strong class="col-2 text-
right">Delivery:</strong>
                                <span class="col-10" id="days" th:data-day="$
{order.expected_delivery_days}">[[${order.expected_delivery_days==null?'Not
Approved.':order.expected_delivery_days+' Days'}]]</span>
                        </div>
                        <div class="row mb-2" th:with="adr = ${order.address}" >
                                <strong class="col-2 text-
right">Address :</strong>
                                <a class="col-10" th:href="@{'/admin/address/'+$
{adr.userId}}">
                                        <th:block class="pl-5" th:utext="$
{adr.full_name+' '+adr.address_line_1+
                                        ', '+adr.address_line_2+' '+adr.city+'
'+adr.state+', '+adr.country+'-'+adr.zip}"></th:block>
                                </a>
                        </div>
                        <div class="row mb-2">
                                <strong class="col-2 text-
right">Status :</strong>
                                <div class="col-10"><div id="status">[[$
{order.status}]]</div>
                                        <small class="text-info">If Payment
Method is Cash on Delivery and Status is Delivered means payment received on
delivery.</small>
                                </div>
                        </div>
                        <div class="row mb-2">
                                <strong class="col-2 text-
right">Message :</strong>
                                <span class="col-10 status">[[$
{order.message.isEmpty?'No Message':order.message}]]</span>
```

```
                        </div>
                        <div class="row mb-2">
                                <strong class="col-2 text-right">Payment
Method :</strong>
                                <div class="col-10 status text-uppercase">
                                        [[${order.paymentMethod=="OP"?'online
payment':'cash on delivery'}]]

                                        <!-- CASH ON DELIVERY -->
                                        <span th:if="$
{(order.paymentMethod=='COD' && order.status.toString()=='DELIVERED')}"
class="pay-status badge badge-success">PAID</span>
                                        <span th:if="$
{order.paymentMethod=='COD' && order.status.toString()!='DELIVERED'}"
class="pay-status badge badge-secondary">NOT PAID</span>

                                        <!-- ONLINE -->
                                        <span th:if="$
{(order.paymentMethod=='OP' && order.payments.size()>0)
                                                &&
order.payments.get(0).status.toString=='PENDING'}" class="pay-status badge
badge-info">PENDING</span>
                                        <span th:if="$
{(order.paymentMethod=='OP' && order.payments.size()>0)
                                                &&
order.payments.get(0).status.toString=='PAID'}" class="pay-status badge badge-
success">PAID</span>
                                        <span th:if="${order.paymentMethod=='OP'
&& order.payments.isEmpty}" class="badge badge-secondary">NOT PAID</span>
                                </div>
                        </div>
                        <table class="table mt-4 order-items"
th:with="orderItems = ${order.orderItems}">
                                <thead>
                                        <tr>
                                                <th>#</th>
                                                <th>Image</th>
                                                <th>Title</th>
                                                <th class="text-
right">Quantity</th>
                                                <th class="text-right">Amount
(Rs)</th>
                                        </tr>
                                </thead>
                                <tbody>
                                        <tr class="item" th:each="orderItem,
status : ${orderItems}">
                                                <td>[[${status.count}]]</td>
                                                <td><img class="max-width:60px"
height="60" th:src="${orderItem.phone.imagePath}" alt="Phone Image" /></td>
                                                <td>[[$
```

```
{orderItem.phone.title}]]</td>
                                                <td class="text-right">x[[${orderItem.quantity}]]</td>
                                                <td class="text-right">Rs [[${orderItem.amount}]]</td>
                                            </tr>
                                            <tr class="border-amt">
                                                <td></td>
                                                <td></td>
                                                <td></td>
                                                <td></td>
                                                <td class="total-amount text-right font-weight-bold text-danger">Rs [[${order.grandTotal}]]</td>
                                            </tr>
                                        </tbody>
                                    </table>
                                    <div class="row p-3 mb-2">
                                        <a class="btn btn-dark mr-auto" th:href="@{/admin/orders}" href="#">Go Back</a>
                                        <div class="offset-2">
                                            <a class="status-btn btn btn-primary" th:data-href="${'/admin/orders/'+order.id+'/status'}" href="#">Change Status</a>
                                            <a class="delete-btn btn btn-danger" th:data-href="@{'/admin/orders/'+${order.id}+'/delete'}" href="#">Delete</a>
                                        </div>
                                    </div>
                                </div>
                            </div>
                            <div class="mt-auto">
                            <footer th:replace="fragments::footer()" />
                        </div>
                    </div>

                    <div th:replace="admin/fragments::confirmWindow()"></div>

                    <div id="order-status-window" class="modal fade">
                      <div class="modal-dialog modal-lg modal-dialog-centered">
                        <div class="modal-content">
                          <div class="modal-header text-light bg-dark">
                            <h5 class="modal-title">Order Status</h5>
                            <button type="button" class="close text-light" data-dismiss="modal">
                                    <span aria-hidden="true">&times;</span><span class="sr-only">Close</span>
                            </button>
                          </div>
                          <div class="modal-body">
                            <div id="order-status-form" class="row">
                                    <input th:if="${_csrf!=null}" type="hidden" th:name="${_csrf.parameterName}" th:value="${_csrf.token}" />
                                    <div class="col-6">
                                            <h4>Choose Order Status:</h4>
```

```
                        <label class="d-flex align-items-center"
th:each="orderStatus : $
{T(com.github.sorabh86.uigo.constants.OrderStatus).values()}">
                                <input type="radio" class="mr-3 ml-3"
name="order_status" th:value="${orderStatus}" th:text="${orderStatus}" />
                        </label>
                </div>
                <div class="col-6">
                        <h4>Day of Delivery</h4>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="30"/> 30 Days</label>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="15"/> 15 Days</label>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="7"/> 7 Days</label>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="5" checked/> 5
Days</label>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="2"/> 2 Days</label>
                        <label class="d-flex align-items-center"><input
class="mr-3 ml-3" type="radio" name="delivery_days" value="1"/> 1 Days</label>
                        <label>
                                <input id="custom-day" class="mr-3 ml-3"
type="radio" name="delivery_days" value="custom"/> Custom
                                <input class="ml-2 p-1 pl-3 pr-3"
type="number" />
                        </label>
                </div>
                <div class="col mt-3">
                        <div class="row ml-3 mr-3">
                                <button type="button" class="update-btn btn btn-
primary col">Update</button>
                                <button type="button" class="ml-3 btn btn-
secondary col" data-dismiss="modal">Cancel</button>
                            </div>
                </div>
        </div>
      </div>
    </div>
  </div>
</div>

<script>
        // Add Delete Confirmation Logic
        var $confirmWindow = $("#confirm-window");
        $(".delete-btn").on("click", function() {
                var href = $(this).data("href");
                $confirmWindow.data("href", href);
                $confirmWindow.modal();
```

```
                return false;
        });
        $confirmWindow.find(".yes-btn").on("click", function(){
                var href = $confirmWindow.data("href");
                window.location.href = href;
        });

        // Update Total Amount for Order Items
        var $orderItems = $(".order-items");
        for(var i=0; i< $orderItems.length; i++) {
                var $orderItem = $($orderItems[i]).find(".item");
                var totalAmount = 0;
                for(var j=0; j < $orderItem.length; j++) {
                        var $subItem = $($orderItem[j]);
                        var qty = $subItem.find(".item-quantity").html();
                        var amt = $subItem.find(".item-amount").html();
                        totalAmount += (amt*qty);
                }
                $orderItem.find(".total-amount").html("Total: "+totalAmount);
        }

        // Change status use AJAX call
        var $orderForm = $("#order-status-form");
        var $orderWindow = $("#order-status-window");
        var $orderStatus = $("input[name=order_status]");
        var $deliverDays = $("input[name=delivery_days]");
        var $customDay = $deliverDays.filter("#custom-day"),
                $customDayInput =
$ customDay.parent().find("input[type=number]");

        $(".status-btn").on("click", function() {
                var day = $('#days').data('day');
                var status = $('#status').html();
                var $this = $(this);

                $orderStatus.removeAttr("checked", "");
                $orderStatus.filter("[value="+status+"]").attr('checked',
'checked');
                if(day) {
                        var $deldays = $deliverDays.filter("[value="+day+"]");
                        if($deldays.length>0) {
                                $deldays.attr('checked', 'checked');
                                $deldays[0].checked = true;
                        } else {
```

```
                                $customDay.attr("checked", true);
                                $customDay[0].checked = true;
                                $customDayInput.val(day);
                        }
                }

                $orderWindow.data("href", $this.data("href"));
                $orderWindow.data("parent", $this.parent());
                $orderWindow.modal({
                        backdrop: 'static',
                    keyboard: false
                });
                return false;
        });

        $customDay.on("click", function(){
                $customDayInput.focus();
        });
        $customDayInput.on("click", function(){
                $deliverDays.attr("checked", false);
                $customDay.attr("checked", true);
                $customDay[0].checked = true;
                return false;
        });

        $orderForm.find(".update-btn").on("click", function(){
                var order_status = $orderStatus.filter(":checked").val();
                var $deliveryDays = $deliverDays.filter(":checked");
                var delivery_days = $deliveryDays.val();
                var _csrf = $orderForm.find("input[name=_csrf]").val();
                var href = $orderWindow.data("href");

                console.log(order_status, delivery_days, href, _csrf);

                if(delivery_days =='custom') {
                        delivery_days = parseInt($customDayInput.val());
                        if(delivery_days == '') {
                                alert("Please Input Custom Day");
                                return false;
                        }
                }
                if(!order_status || order_status=='' || !delivery_days ||
delivery_days=='') {
                        alert("Choose Status and days");
                        return false;
                }
```

```
                $.post({
                        url:href,
                        data:{
                                order_status: order_status,
                                delivery_days: delivery_days,
                                _csrf: _csrf
                        },
                        success:function(d) {
                                if(d.message=='OK') {
                                        $orderWindow.modal("hide");
                                        $('#days').data('day',delivery_days);
                                        $('#days').html(delivery_days+' Days')
                                        $('#status').html(d.status);
                                }
                                console.log("Successful: ",d);
                        },
                        error:function(e) {
                                console.error(e.responseText);
                                var error = JSON.parse(e.responseText);
                                alert("Error: "+error.message);
                        }
                });

        });
</script>

</body>
</html>
```

- **/resources/templates/admin/feedback/feedback.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Orders')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">
                <h2 class="mb-3">Admin | Feedback</h2>
                <div th:if="${feedbacks==null}" class="alert alert-warning">No
Feedback Found!!!<br> No Feedback submitted by customers.</div>
                <table th:if="${feedbacks}" class="table">
                        <thead>
                                <tr>
                                        <th>Id</th>
                                        <th>Rate</th>
                                        <th>Date</th>
                                        <th>Order Id</th>
                                        <th>User</th>
                                        <th>Phone</th>
```

```
                                        <th></th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr th:each="feedback : ${feedbacks}">
                                        <td>[[${feedback.id}]]</td>
                                        <td>
                                            <span th:each="index: ${#numbers.sequence(1,5)}"
                                                  th:data-index="${index}"
                                                  th:class="${index<=feedback.rate.value?'p-0 fa fa-star fa-2x text-warning':'p-0 fa fa-star fa-2x text-secondary'}"
                                                  ></span>
                                        </td>
                                        <td>
                                            [[${#dates.format(feedback.date,"hh:mm a")}]],
                                            [[${#dates.format(feedback.date,"dd MMM, yyyy")}]]
                                        </td>
                                        <td>
                                            [[${feedback.order.id}]]
                                        </td>
                                        <td>
                                            [[${feedback.user.username}]]
                                        </td>
                                        <td>
                                            [[${feedback.phone.title}]]
                                        </td>
                                        <td>
                                            <a class="delete-btn text-danger" href="#"
                                               th:data-href="@{'/admin/feedbacks/'+${feedback.id}+'/delete'}" >Delete</a>
                                        </td>
                                    </tr>
                                </tbody>
                            </table>
                        </div>
                        <div class="mt-auto">
                        <footer th:replace="fragments::footer()" />
                    </div>
                </div>

                <div th:replace="admin/fragments::confirmWindow()"></div>

                <script>
                    // Add Delete Confirmation Logic
                    var $confirmWindow = $("#confirm-window");
                    $(".delete-btn").on("click", function() {
```

```
                var href = $(this).data("href");
                $confirmWindow.data("href", href);
                $confirmWindow.modal();
                return false;
            });
            $confirmWindow.find(".yes-btn").on("click", function(){
                var href = $confirmWindow.data("href");
                window.location.href = href;
            });
        </script>

</body>
</html>
```

- **/resources/templates/admin/chat/chats.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Chats')" />
<body>
<link rel="stylesheet" th:href="@{/css/chat.css}" />
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container d-flex flex-column flex-grow-1">
            <h2 class="border-bottom pb-3 mb-4">Admin | Chats</h2>
            <div class="d-none">
                <a th:data-name="${user.username}"
                    href="#" id="start-btn" class="btn btn-
primary">Start Chatting</a>
            </div>
            <div class="chat-box d-flex">
          <div class="member-list w-25 border d-flex flex-column">
            <div class="member m-3 active" data-href="chatroom">Public Chat
Room
                    <div class="chat-status badge badge-light"><span
class="fa fa-circle text-danger"> Offline</span></div>
            </div>
            <h4 class="text-center m-0 alert alert-secondary">Customers</h4>
            <ul class="member-content p-3 flex-grow-1">
                <th:block th:each="username : ${allUsers}">
                    <li th:if="${user.username!=username}" th:class="$
{'member '+username}" th:data-href="${username}">
                    [[${username}]]
                    <span th:class="${#lists.contains(onlineUsers,
username)?
                        'user-status rounded-circle bg-
success':'user-status rounded-circle bg-danger'}"></span>
                    </li>
                </th:block>
            </ul>
          </div>
```

```
            <div class="chat-rooms w-75 ml-3 border p-3 d-flex">
                    <!-- Public Chat window -->
                    <div class="chat-content chatroom active">
                        <ul class="chat-messages border flex-grow-1">
                                <th:block th:each="message : ${publicMessages}">
                                        <li th:class="${user.username==message.username? 'message self':'message'}">
                                                <div th:if="${user.username} ne ${message.username}" class="avatar">
                                                        [[${message.username}]]<br />
                                                        <small class="badge date">[[${#dates.format(message.date, "dd/MM/YY")}]]</small>
                                                </div>
                                                <div class="message-data">[[${message.message}]]</div>
                                                <div th:if="${user.username} eq ${message.username}" class="avatar self">
                                                        [[${message.username}]]<br />
                                                        <small class="badge">[[${#dates.format(message.date, "dd/MM/YY")}]]</small>
                                                </div>
                                        </li>
                                </th:block>
                        </ul>
                        <div class="send-message d-flex">
                            <input type="text" class="input-message form-control border-secondary rounded-0 flex-grow-1"
                                    placeholder="enter the message" value="" />
                            <button type="button" class="send-button btn btn-primary rounded-0">send</button>
                        </div>
                    </div>

                    <th:block th:each="username : ${allUsers}">
                        <div th:class="${'chat-content '+username}">
                        <ul class="chat-messages border flex-grow-1">
                        </ul>
                        <div class="send-message d-flex">
                            <input type="text" class="input-message form-control border-secondary rounded-0 flex-grow-1"
                                    placeholder="enter the message" value="" />
                            <button type="button" class="send-button btn btn-primary rounded-0">send</button>
                        </div>
                    </div>
                    </th:block>
            </div>
        </div>
        </div>
```

```
        <div class="">
        <footer th:replace="fragments::footer()" />
    </div>
</div>

<script th:src="@{/js/1.1.4/sockjs.min.js}"></script>
<script th:src="@{/js/2.3.3/stomp.min.js}"></script>
<script th:src="@{/js/chat.js}"></script>

</body>
</html>
```

- **/resources/templates/admin/address/address_form.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | User Form')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
        <div th:replace="admin/fragments::admin_navigation()" />
        <div class="container">

                <h2 class="border-bottom pb-3 mb-4">[[${page_title}]] <span
th:if="${user_id}"> [ <small>User([[${user_id}]])</small> ] </span></h2>

                <form class="form" name="customer_form"
th:action="@{/admin/address/save}" method="post"
                        th:object="${address}">

                        <input type="hidden" th:field="*{id}" />

                        <div th:if="${users}" class="form-group row">
                                <label class="col-md-2">Choose a User: </label>
                                <div class="col-md-6">
                                        <select class="form-control" id="userId"
name="userId">
                                                <option th:each="user1 : $
{users}"
                                                        th:value="${user1.id}"
th:text="${user1.id+' '+user1.full_name}" />
                                        </select>
                                </div>
                        </div>
                        <input th:if="${users==null}" type="hidden"
th:field="*{userId}" />

                        <div class="form-group row">
                                <label class="col-md-2">Full Name: </label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{full_name}"
                                                placeholder="Your Full Name"/>
```

```
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Phone: </label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{phone}"
                                                placeholder="Your Phone No"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Address Line 1:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{address_line_1}"
placeholder="Address Line 1"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Address Line 2:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{address_line_2}"
placeholder="Address Line 2"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">City:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{city}"
placeholder="Your City"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">State:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{state}"
placeholder="Your State"/>
                                </div>
                        </div>
                        <div class="form-group row">
                                <label class="col-md-2">Country:</label>
                                <div class="col-md-6">
                                        <input class="form-control" type="text"
                                                th:field="*{country}"
placeholder="Your Country"/>
                                </div>
                        </div>
                        <div class="form-group row">
```

```
                                    <label class="col-md-2">Zip Code:</label>
                                    <div class="col-md-6">
                                            <input class="form-control" type="text"
                                                   th:field="*{zip}"
placeholder="Zip Code"/>
                                    </div>
                            </div>

                            <div class="form-group">
                                    <div class="offset-md-2">
                                            <input class="btn btn-primary mr-2"
type="submit" value="Save" />
                                            <a th:href="@{/admin/address}"
class="btn btn-dark" >Cancel</a>
                                    </div>
                            </div>
                    </form>
            </div>

            <div class="mt-auto">
                    <footer th:replace="fragments::footer()" />
            </div>
</div>
</body>
</html>
```

- **/resources/templates/admin/address/address.html**

```
<!doctype html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="admin/fragments :: admin_head('Admin | Address')" />
<body>
<div class="d-flex flex-column" style="min-height:100vh">
            <div th:replace="admin/fragments::admin_navigation()" />
            <div class="container">

                    <h2 class="border-bottom pb-3 mb-4">Admin | Addresses <span
th:if="${user_id}"> [ <small>User([[${user_id}]])</small> ] </span>
                            <a th:if="${user_id==null}" class="btn btn-primary"
th:href="@{${'/admin/address/new'}}">Add Address</a>
                            <a th:if="${user_id}" class="btn btn-primary"
th:href="@{${'/admin/address/new/'+user_id}}">Add Address</a>
                    </h2>

                    <div th:if="${message!=null}" class="alert alert-success">
                            [[${message}]]
                    </div>

                    <table class="table">
                            <thead>
                                    <tr>
                                            <th>Id</th>
```

```
                                        <th>User Id</th>
                                        <th>Full Name</th>
                                        <th>Phone</th>
                                        <th>Address</th>
                                        <th></th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr th:each="address : ${addresses}">
                                        <td>[[${address.id}]]</td>
                                        <td>[[${address.userId}]]</td>
                                        <td>[[${address.full_name}]]</td>
                                        <td>[[${address.phone}]]</td>
                                        <td>[[${address.address_line_1}]], [[${address.address_line_2}]], [[${address.city}]], [[${address.state}]], [[${address.country}]]-[[${address.zip}]]</td>
                                        <td>
                                            <a th:href="@{'/admin/address/'+${address.id}+'/edit'}">Edit</a> |
                                            <a class="delete-btn text-danger" th:data-href="@{'/admin/address/'+${address.id}+'/delete'}" href="#">Delete</a>
                                        </td>
                                    </tr>
                                </tbody>
                            </table>
                        </div>

                        <div class="mt-auto">
                            <footer th:replace="fragments::footer()" />
                        </div>
                    </div>

                    <div th:replace="admin/fragments::confirmWindow()"></div>

                    <script>
                        var $confirmWindow = $("#confirm-window");
                        $(".delete-btn").on("click", function() {
                            var href = $(this).data("href");
                            $confirmWindow.data("href", href);
                            $confirmWindow.modal();
                            return false;
                        });
                        $confirmWindow.find(".yes-btn").on("click", function(){
                            var href = $confirmWindow.data("href");
                            window.location.href = href;
                        });
                    </script>
                </body>
```

```
</html>
```

4.3. Comments and Description of Coding segments

- **HTML Tags:**
  - <html></html>: This is main tag of any html document.
  - <head></head>: This tag containers header information about page.
  - <title></title>: Title of the page.
  - <body></body>: This tag contains visual layout of any html document.
  - <h1>,<h2>,<h3>,<h4>,<h5>,<h6> : These tags represent page headings.
  - <p> : This tag represent paragraph.
  - <header> : This tag define a visualise page header.
  - <nav> : This tag tell us about navigation links.
  - <section> : This tag is used to group content into sub category.
  - <footer> : This tag define a visualise page footer, basically where you put all copyright and terms and condition informations.
  - <form> : Form are used for sending information to other page.
  - <input> : You can set many type of input tags, those are used to carry values in form elements.
  - <script></script>: All your inline JavaScript code go in between this tag.
- **CSS:**
  - **# (Hash)** : Here # mean select element by id attribute.
  - **. (dot)** : Here . Is used to select element and apply styles based on class attribute of html tags.
  - **, (comma)** : comma is used to assign two target element to one style block.
  - **{} (Curly Brackets)** : These brackets are used to group styles together.
  - **: (Colon)** : This is used to represent (key : value).
  - **; (Semi-colon)** : This is used to complete statements
  - **Example** :

```
#someid,
.someclass {
        width: 500px;
        /* ...more styles... */
}
```

- **Spring Boot Annotations:**
  - @SpringBootApplication : Marking a class as main application.
  - @EntityScan : Scanning in package for auto creation of classes.
  - @Configuration : Class where all beans and configuration are stored.

- @Controller : This declares a class to be map routes and reload html templates based on requested path.
- @Service : This declare a class as service, it means all your server logic will stored inside that class.
- @Repository : This annotation declared a class to connect to database.
- @Entity: This annotation declared a class as database persistence object.
- @Autowired: This annotation auto inject required class using Spring boot IOC (Inversion of Control) Container.
- @EnableWebSecurity : This annotation enable Spring security, but you will need to further override various methods to get custom security.
- @EnableWebSocketMessageBroker: This annotation enable Websocket in your sprint boot application, you will need to further declare endpoints where connection could established for two way communication or message pushing to all connected clients.

- **ThymeLeaf directives:**
  - <html xmlns:th="http://www.thymeleaf.org"> : Enable thymeleaf template to html template to use various useful tags.
  - th:fragment & th:replace : th:fragment declared a reusable html template that can be include as many time as required using th:replace attribute.
  - th:object : this attribute map a java object to a form fields.
  - th:field : bind a java object property to a html input element.
  - th:if : If value passed inside is true only this element will be created.
  - th:each : This attribute used to iterate in any type of Java List implementations.

4.4. Standardization of the coding

Good software development organizations develop their own coding standards and guidelines depending on what suits their organization best and based on the specific types of product they develop.

1. Rules for limiting the use of global: These rules list what types of data can be declared global and what cannot.
2. Naming conventions for global variables, local variables and constant identifiers: A popular naming convention is that variables are named using mixed case lettering.
3. Do not use a coding style that is too clever or too difficult to understand: code should be easy to understand. Clever coding can obscure meaning of the code and hamper understanding.
4. The length of any function should not exceed 10 source lines: a lengthy function is usually very difficult to understand as it probably has a large number of variables.

4.5. Code Efficiency

Code efficiency plays a significant role in applications in a high-execution-speed environment where performance and scalability are uppermost. One of the recommended bes practices in coding is to ensure good quality of code.

- Your project doesn't have any unnecessary code or code that goes to redundant processing.
- Your project's coding makes use of optimal memory and non volatile storage.
- Use reusable components wherever possible.
- Alway ensures data integrity and consistency of your code.
- Functions and variables are meaningful variable names to increase the readability and avoided abbreviations that may confuse the users.

4.6. Error Handling

Error Handling is an important step, if you are not handling errors on your website, then it may have bad impact on your customers.

1. Setup error Template Pages: Spring boot provide better error handling by setup thymeleaf error templates, Those are useful feature that you can customize further more or implement your own on top of that.

2. Error handling using try & catch statements: When you write code and many times you face issues that could be happened due to many reasons like may be your remote connection dropped, or your database connectivity broken, or cases where you didn't get expected results. For those custom code, you can use try & catch blocks.

4.7. Parameters Calling/Passing

There are two basic types of method types for parameters passing, Those are:

1. **GET Method :** The GET method send the parameters via URL address which could be seen by anyone in search box of the browser.

   e.g.

   ```
   <a href="/contact?key=value"></a>
   ```

   OR

   ```
   <form action="/contact" method="GET">
           <input type="text" name="name" value="sorabh">
           <input type="text" name="age" value="30">
    </form>
   ```

2. **POST Method :** The POST method send the parameters to request data, which is completely hidden from user and can only be seen by some special developer tools.

   e.g.

   ```
   <form action="/contact" method="POST">
           <input type="text" name="name" value="sorabh">
           <input type="text" name="age" value="30">
    </form>
   ```

4.8. Validation checks

HTML 5 language itself provide some of basic validation check features for form and input fields, also we can implement our own complex validation via JavaScript.

We can divide our code in following categories:

1. **HTML**: We can user attribute called required for any html form elements, That automatically enforced user to type something before submitted any form.

   e.g.

```
<form name="ourform" action="/contact" method="POST">
        <input type="text" name="name" value="sorabh" requried>
        <input type="text" name="age" value="30" requried>
        <button type="submit">Submit</button>
</form>
```

2. **JavaScript (jQuery Library)**:

   e.g.

```
<form name="ourForm" action="/contact" method="POST">
        <input type="text" name="name" value="sorabh">
        <input type="text" name="age" value="30">
        <button type="submit">Submit</button>
</form>

<script>
var ourForm = document.forms.ourForm;

$(ourForm).on("submit", function(){
        if(ourForm.name.value=='') {
                alert("You didn't fillup name");

                // stop form to submit to url specified
                return false;
        }
        if(ourForm.age.value=='') {
                alert("You didn't fillup your age");

                return false;
        } else if(typeof parseInt(ourForm.age.value)!='number') {
                 alert("Please Type age to Number");
                 return false;
        }

        // allow form to send data to url
        return true;
});
</script>
```

5. Testing

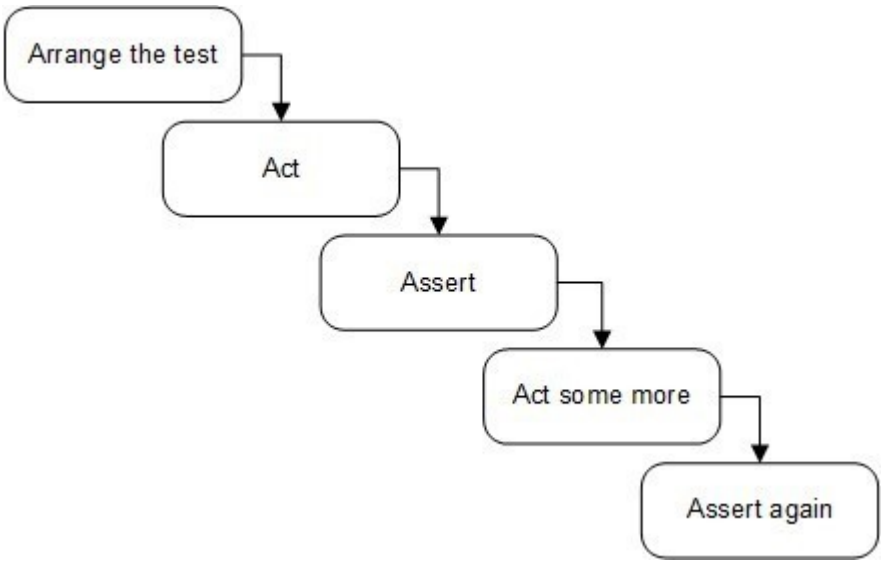
We test every line of code to check, verify and evaluate the system that comes up after going through the demanding process. It is one of the important process in any software development life cycle. We mostly divide system into 3 various categories (Unit Test, Integration Test and System Test). Each category contains test cases to be defined, and generate Test Report to ensure functionality.

5.1. Testing techniques and Testing strategies used

AAA-Pattern (Arrange, Act, Assert Pattern)

Arrange-Act-Assert is a great way to structure test cases. It prescribes an order of operations:

1. **Arrange inputs and targets:** Arrange steps should set up the test case. Does the test require any objects or special settings? Does it need to prep a database? Does it need to log into a web app? Handle all of these operations at the start of the test.
2. **Act on the target behaviour:** Act steps should cover the main thing to be tested. This could be calling a function or method, calling a REST API, or interacting with a web page. Keep actions focused on the target behaviour.
3. **Assert expected outcomes:** Act steps should elicit some sort of response. Assert steps verify the goodness or badness of that response. Sometimes, assertions are as simple as checking numeric or string values. Other times, they may require checking multiple facets of a system. Assertions will ultimately determine if the test passes or fails.



Advantages:

1. Powerful because it is simple
2. Focus on independent, individual behaviours.
3. It separates setup actions from the main actions.
4. It can be used for both Unit or Integration Test.

5.2. Test Cases

5.2.1. Unit Test Cases

Test Case No.	UTC-001
Functionality	Add New User to database
Environment	Spring Boot Testing Environment (@DataJpaTest,@Rollback(false))
Procedure	1. Create a User Object. 2. Save User Object using UserRepo 3. Assert that it returned a positive number or greater then zero.
Expected Outcome	User is successfully added to database.

Test Case No.	UTC-002
Functionality	Remove Existing User to database
Environment	Spring Boot Testing Environment (@DataJpaTest,@Rollback(false))
Procedure	1. Create a User Object. 2. Set User Id to existing user in database. 3. Assert that it returned positive number or greater then zero.
Expected Outcome	User is removed from database.

Test Case No.	UTC-003
Functionality	Getting user by username
Environment	Spring Boot Testing Environment (@DataJpaTest,@Rollback(false))
Procedure	1. Initialize variable with existing username in database. 2. Call method findByUsername. 3. Assert that return value is not null.
Expected Outcome	Get a new User Object with all details.

Test Case No.	UTC-004
Functionality	Sending Text Email
Environment	Spring Boot Testing Environment(@SpringBootTest)

Procedure	1. Create SimpleMailMessage Object. 2. Set attribute values 3. Send email by calling send method, No Exceptions means successful.
Expected Outcome	Your text email received to receiver email id.

Test Case No.	UTC-005
Functionality	Sending Attachment Email
Environment	Spring Boot Testing Environment(@SpringBootTest)
Procedure	1. Create SimpleMailMessage Object. 2. Set attribute values and attached media file(image, pdf, etc) 3. Send email by calling send method, No Exceptions means successful.
Expected Outcome	Your email with attachment received to receiver email id.

Test Case No.	UTC-006
Functionality	Sending HTML Email
Environment	Spring Boot Testing Environment (@SpringBootTest)
Procedure	1. Create SimpleMailMessage Object. 2. Set attribute values and attached media file(image, pdf, etc) 3. Send email by calling send method, No Exceptions means successful.
Expected Outcome	Your html email received to receiver email id.

Test Case No.	UTC-007
Functionality	Test Saving file on server
Environment	Spring Boot Testing Environment
Procedure	1. Create folder if not existing. 2. Save file on that folder. 3. Check/Assert if file is existed in directory.
Expected Outcome	File uploaded is saved to the server

Test Case No.	UTC-008
Functionality	Test delete file on server
Environment	Spring Boot Testing Environment

Procedure	1. Check if path is qualified. 2. Open File IO location and unlink that file. 3. Check/Assert if file is not existed in directory.
Expected Outcome	Existing file is removed from server

5.2.2. Integration Test Cases

Test Case No.	ITC-001
Functionality	Registration and Login
Environment	Manually
Procedure	1. Open Registration Page. 2. Register for account. 3. Check if it automatically open your login page.
Expected Outcome	Prompt user for login after registration.

5.2.3. System Test Cases

Test Case No.	STC-001
Functionality	Login with account
Environment	Manually
Procedure	1. Open Login Page 2. Fill up username and password, then click login button. 3. Check if it display your username and logout button.
Expected Outcome	User successfully logged into his account.

Test Case No.	STC-002
Functionality	Register for account
Environment	Manually
Procedure	1. Open Register Page 2. Fill up your details, then click register button. 3. Check if it display successful message.
Expected Outcome	User successfully register for an account.

Test Case No.	STC-003

Functionality	Search for Phone
Environment	Manually
Procedure	1. Open website home page, you will see a search form. 2. Type your search keyword, then click search button. 3. You will redirected to list of phones page with matching phones are filter.
Expected Outcome	Showing Phone list based on search keyword.

Test Case No.	STC-004
Functionality	Add Phone to Cart
Environment	Manually
Procedure	1. Open phone list page. 2. Check details, set quantity click on addtocart button. 3. Go to Cart page check if item is added in cart list.
Expected Outcome	Added Phone to Cart page.

Test Case No.	STC-005
Functionality	Order a form (Cash on delivery)
Environment	Manually
Procedure	1. Go to phone page, choose phone and add it to cart. 2. Go to cart page, choose option cash on delivery and click checkout button. 3. Display a page with your order details.
Expected Outcome	Your Order is generated and email sent with details

Test Case No.	STC-006
Functionality	Order a form (Online Payment)
Environment	Manually
Procedure	1. Go to phone page, choose phone and add it to cart. 2. Go to cart page, choose option online payment and click checkout button. 3. Display a page with your order details if successful, otherwise display payment failures page with retry button.
Expected Outcome	Your Order is generated and email sent with details.

5.2. Test Reports

Test Case No.	Date	Pass/Fail	Comments
UTC-001	20/05/22	Pass	New entry for user has been created in database. e.g. username=sorabh inserted.
UTC-002	20/05/22	Pass	Existing is deleted from database. e.g. username=sorabh removed.
UTC-003	20/05/22	Pass	Getting back user details by calling method. e.g. getByUsername("admin").
UTC-004	20/05/22	Pass	Text Email received on my email.
UTC-005	20/05/22	Pass	Email with attachment received on my email.
UTC-006	20/05/22	Pass	Html Email received on my email.
UTC-007	20/05/22	Pass	Uploading Phone photo is saved on server.
UTC-008	20/05/22	Pass	Deleted one of existing image file on server.
ITC-001	20/05/22	Pass	After Registration, we are redirected to login page.
STC-001	20/05/22	Pass	Login with 3 type of ROLES, that is ROLE_ADMIN, ROLE_CUSTOMER, ROLE_SUBSCRIBER. Link for logout under username drop-down after login based on authorisation.
STC-002	20/05/22	Pass	Register with new user without any problem.
STC-003	20/05/22	Pass	Search Phone return list of matching items.
STC-004	20/05/22	Pass	Add Phone to Cart listed correctly.
STC-005	20/05/22	Pass	Order Id generated straight after choose Cash on delivery.
STC-006	20/05/22	Pass	Order Id generated after payment success.

5.2. Automate Testing Code

We are using Spring Boot Test Classes for automate Testing here are some testing classes:

- **MobileShoppingSiteApplicationTests.java**

```
package com.github.sorabh86.uigo;

import java.time.Duration;
import java.time.Instant;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
import java.util.List;
```

```java
import java.util.Map;
import java.util.stream.Collectors;

import org.junit.jupiter.api.Test;

import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.CartItem;
import com.github.sorabh86.uigo.entity.User;

class MobileShoppingSiteApplicationTests {

	@Test
	public void getArrayFromListKeyValue() {
		List<User> users = new ArrayList<>();
		for(int i = 0; i<10; i++) {
			User u1 = new User();
			u1.setId(i);
			u1.setUsername("sorab"+i+Math.random());
			users.add(u1);
		}
		var array = users.stream().map(u->
u.getUsername()).collect(Collectors.toList());
		System.out.println(array);
	}

	@Test
	public void checkDefaultValueOfId() {
		Address address = new Address();
		System.out.println(address);
	}

	@Test
	public void getDateAfterSomeDays() {
		Date date = new Date();
		System.out.println("before date: "+date);

		Instant now = date.toInstant();
		System.out.println("now: "+now);
		Instant plus = now.plus(Duration.ofDays(7));
		System.out.println("plus: "+plus);

		Date newDate = Date.from(plus);
		System.out.println("after date: "+newDate);
	}

	@Test
	public void convertRequestKeyIntoArrayList() {
		Map<String, Object> map = new HashMap<>();
		map.put("cartItems[1].phone_id", 2);
		map.put("cartItems[1].price", 12903);
```

```
                map.put("cartItems[1].quantity", 1);
                map.put("cartItems[1].phone_id", 3);
                map.put("cartItems[1].price", 28422);
                map.put("cartItems[1].quantity", 1);
                map.put("cartItems[1].phone_id", 6);
                map.put("cartItems[1].price", 11000);
                map.put("cartItems[1].quantity", 1);
                map.put("cartItems[1].phone_id", 8);
                map.put("cartItems[1].price", 16000);
                map.put("cartItems[1].quantity", 1);

                List<Map<String, Object>> filterList = new ArrayList<>();
                map.forEach((key, value) -> {
                        if(key.contains("cartItems")) {
                                Map<String, Object> item = new HashMap<>();
                                Integer index = key.indexOf(10);
                                System.out.println(index);
                        }
                });

                boolean hasCartKey = "[1].phone_id".contains("cartItems");
                System.out.println(hasCartKey);
        }

        @Test
        public void testEnumString() {
                UserRoles role = UserRoles.ROLE_ADMIN;
                System.out.println(role.toString());
        }

        @Test
        public void testCartItemEqual() {
                CartItem item1 = new CartItem();
                item1.setId(2);
                CartItem item2 = new CartItem();
                item2.setId(2);
                System.out.println(item1.equals(item2));
        }

        @Test
        public void testRandomRange() {
                for(int i=0; i<20; i++)
                        System.out.print(getRandomNumber(1, 10)+ "  =  ");

        }

        public int getRandomNumber(int start, int end) {
                return (int)(start+(Math.round(Math.random()*(end-start))));
        }
}
```

- **TestUser.java**

```
package com.github.sorabh86.uigo.users;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
import
org.springframework.boot.test.autoconfigure.jdbc.AutoConfigureTestDatabase;
import
org.springframework.boot.test.autoconfigure.jdbc.AutoConfigureTestDatabase.Repla
ce;
import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.test.annotation.Rollback;
import org.springframework.test.context.ContextConfiguration;

import com.github.sorabh86.uigo.admin.users.UserRepo;
import com.github.sorabh86.uigo.config.UIGOWebSecurityConfig;
import com.github.sorabh86.uigo.constants.UserRoles;
import com.github.sorabh86.uigo.constants.UserStatus;
import com.github.sorabh86.uigo.entity.Address;
import com.github.sorabh86.uigo.entity.User;

@DataJpaTest(/*properties = {"spring.jpa.hibernate.ddl-auto=create-drop"}*/)
@AutoConfigureTestDatabase(replace = Replace.NONE)
@Rollback(false)
public class TestUser {

	@Autowired
	private UserRepo userRepo;

	private PasswordEncoder passwordEncoder = new BCryptPasswordEncoder();

	@Test
	public void createUsers() {
		addAdminUser();
		addCustomerUser();
		addSubscriberUser();
	}

	public void addAdminUser() {
		Address address1 = new Address();
		address1.setFull_name("Sorabh86");
		address1.setPhone("9483838842");
		address1.setAddress_line_1("D-23");
		address1.setAddress_line_2("Near Lohia Park");
		address1.setCity("Ghaziabad");
		address1.setState("Uttar Pradesh");
		address1.setCountry("India");
```

```
                address1.setZip("201005");

                User admin = new User();
                admin.setId(1);
                admin.setUsername("admin");
                admin.setEmail("admin@admin.in");
                admin.setPassword(passwordEncoder.encode("admin"));
                admin.setFull_name("admin");
                admin.setPhone_no("0000000000");
                admin.setStatus(UserStatus.ACTIVATED);
                admin.setRole(UserRoles.ROLE_ADMIN);

                admin.addAddress(address1);

                userRepo.save(admin);
        }

        public void addCustomerUser() {
                Address address1 = new Address();
                address1.setFull_name("Sorabh Sharma");
                address1.setPhone("9483960030");
                address1.setAddress_line_1("123 street");
                address1.setAddress_line_2("Near Tower House");
                address1.setCity("Banglore");
                address1.setState("Tamilnadu");
                address1.setCountry("India");
                address1.setZip("284995");

                User sorabh = new User();
                sorabh.setId(2);
                sorabh.setUsername("sorabh");
                sorabh.setEmail("sorabh@customer.in");
                sorabh.setPassword(passwordEncoder.encode("sorabh"));
                sorabh.setFull_name("Sorabh");
                sorabh.setPhone_no("0000000000");
                sorabh.setStatus(UserStatus.ACTIVATED);
                sorabh.setRole(UserRoles.ROLE_CUSTOMER);

                sorabh.addAddress(address1);
                userRepo.save(sorabh);
        }

        public void addSubscriberUser() {
                User neeraj = new User();
                neeraj.setId(3);
                neeraj.setUsername("neeraj");
                neeraj.setEmail("neeraj@subscriber.in");
                neeraj.setPassword(passwordEncoder.encode("neeraj"));
                neeraj.setFull_name("Neeraj");
                neeraj.setPhone_no("0000000000");
                neeraj.setStatus(UserStatus.ACTIVATED);
```

```
                neeraj.setRole(UserRoles.ROLE_SUBSCRIBER);

                userRepo.save(neeraj);
        }

        @Test
        public void getAllUsersTest() {
                Iterable<User> users = userRepo.findAll();
                users.forEach(user->System.out.println(user));
        }
        @Test
        public void getUserByUsernameTest() {
                User user = userRepo.getUserbyUsername("admin");
                System.out.println(user);
        }

        @Test
        public void updatePassword() {
                User sorabh = userRepo.getUserbyUsername("happy");
                System.out.println(sorabh);
                sorabh.setPassword(passwordEncoder.encode("happy"));
                userRepo.save(sorabh);
        }
}
```

6. System Security Measures

6.1. Database/Data Security

Data security or Data security refereed to the process of protecting data from unauthorized access and protect the data corruption during the lifecycle of the system. Data security includes data encryption, tokenization and key management practises that protect data across all application.

Our system has the following security features:-

- System manages who has access of data, unauthorized access will be denied.
- System will use encryption code (BcyrptEncryption) for hashing password saved in database, All Password saved in database are encrypted, and salted with random key pairs.
- Protected from SQL Injection Attacks, code are secured and used layers to pass SQL queries.
- System will be protected by Cross-Site Request Forgery (CSRF) Token enabled to protect it from injection of malware during form submission.

6.2. Creation of User profiles and access rights

- This system is provide authentication, without correct username and password system will not give access to sensitive data.
- User Role will handle and restrict access of data.

7. Cost Estimation/Cost Estimation Model

Software cost estimation is the process of predicting the effort required to develop a software system. If you are developing a system or software, that required you estimate the total cost, this is a basic estimation about forecasting the resources and work hours, which helps ensure you achieve project objectives within the approved timeline and budget. The cost of a project is derived not only from the estimates of effort and size but from other parameters such as hardware, travel expenses, telecommunication costs, training cost, etc. You should also be taken into account.

7.1. Cost Estimation Model

For cost estimation of UIGO Mobile Shopping System, we use **Cost of Quality** Cost Estimation Model.

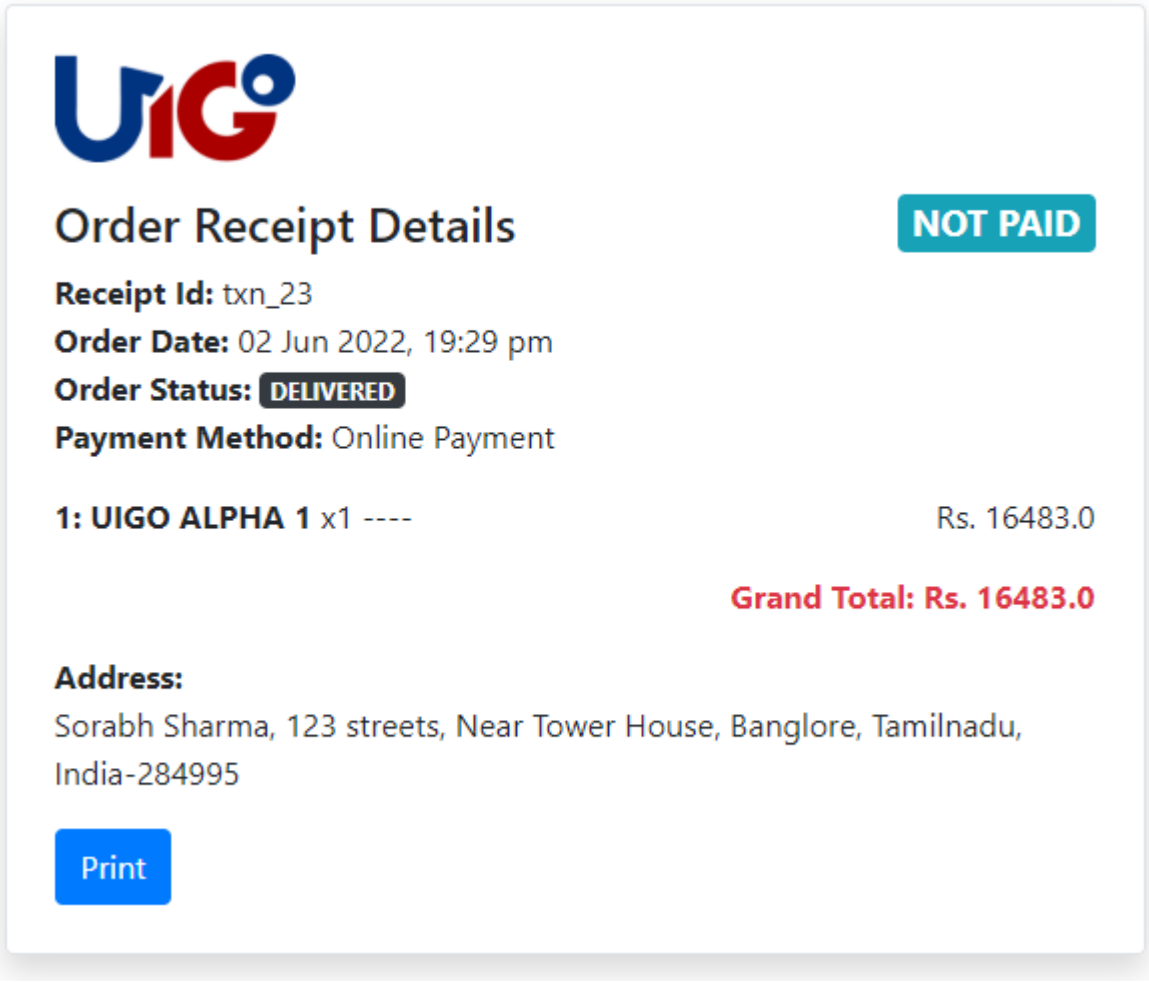
Cost of Quality (COQ) is defined as a methodology that allows an organization to determine the extent to which its resources are used for activities that prevent poor quality, that appraise the quality of the organization's products or services, and that result from internal and external failures.

7.2. Cost Estimation

Cost Estimation Process	Cost
Development Cost (Analysis Efforts, Development Efforts, Testing Efforts)	Rs. 34,951
Hardware Cost (Motherboard, RAM, Hard Disk, Processor, Keyboard, Mouse, Cabinet) per Computer.	Rs. 24,451
Internet Connection (per month)	Rs. 1,001
Training Cost (Basic web designing, Basic Database Management, Basic Administrator Management.	Rs. 11,001
Web Hosting (Rented cloud server, include public IP Address) (per month)	Rs. 801
Domain Name (Purchase & Rented domain address) (per month)	Rs. 801
Total Cost	Rs. 73,006

8. Reports

In this system after submitting order following reports is generated for customer to download.



UIGO

Order Receipt Details

NOT PAID

Receipt Id: txn_23

Order Date: 02 Jun 2022, 19:29 pm

Order Status: DELIVERED

Payment Method: Online Payment

1: UIGO ALPHA 1 x1 ---- Rs. 16483.0

Grand Total: Rs. 16483.0

Address:

Sorabh Sharma, 123 streets, Near Tower House, Banglore, Tamilnadu, India-284995

Print

9. Future scope and Further enhancement of the Project

Our system have further scopes as business grow further, some of those are below:

- Implement pagination for various listing pages.
- We can introduce more product categories, like phone cover, screen guard, mobile charger, battery, PC, laptop, etc.
- We can introduce seasonal discounts on mobile phones.
- We can add services and appoint engineers for repairing to our system.
- We can provide seasonal discounts on cards after succeeds in agreements with various banks.
- We can introduce membership subscription for fast delivery.
- We can integrate social media platform for quick registration, and other share product informations.
- We can integrate google maps to for tracking delivery.
- We can add users to send media files or video meetings of sales executives to customers to get product reviewed and share videos to convenience customer to increase sales.
- We can add more payment options for payments.
- We can also provide 3D models of our phones to provide 360 degree view of selling phones.

10. Bibliography

- Essentials of Software Engineering by Tsui
- Web Design & Development by Nicolae Sfetcu
- https://www.yaantra.com/
- https://budli.in/
- https://www.mi.com/in/list
- https://www.w3schools.com/
- https://docs.spring.io/
- https://www.w3schools.com/
- https://getbootstrap.com/
- https://jquery.com/
- https://www.oracle.com/java/technologies/downloads/
- https://www.mysql.com/
- https://nodejs.org/en/
- https://www.jetbrains.com/idea/

11. Git Repository for Complete code

You can access complete code from following link:

https://github.com/sorabh86/MobileShoppingSite