VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnanasangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V.Puram, Bengaluru-560 004

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DATABASE MANAGEMENT SYSTEM MINI PROJECT

18CSL58

"SERVICE PROVIDING DATABASE MANAGEMENT SYSTEM"

Submitted By

DHANANJAI NIRMAL
1BI18CS049

for the academic year 2020-21

Department of Computer Science & Engineering
Bangalore Institute of Technology
K.R. Road, V.V.Puram, Bengaluru-560 004

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnanasangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V.Puram, Bengaluru-560 004

Department of Computer Science & Engineering

Certificate

This is to certify that the implementation of **DBMS MINI PROJECT** entitled " SERVICE PROVIDING MANAGEMENT SYSTEM" has been successfully completed by

USN: 1BI18CS049 **NAME: DHANANJAI NIRMAL**

of V semester B.E. for the partial fulfilment of the requirements for the Bachelor's degree in Computer Science & Engineering of the Visvesvaraya Technological University during the academic year 2020-2021.

Lab Incharges :

Mrs. Kanchan A. Purohit
Assistant Professor
Dept. of CS&E
Bangalore Institute of Technology
Bangalore

Mrs. Archana A.
Assistant Professor
Dept. of CS&E
Bangalore Institute of Technology
Bangalore

Dr. Asha T .
Professor and Head
Department of CS&E
Bangalore Institute of Technology
Bangalore

Examiners: 1) 2)

ACKNOWLEDGEMENT

The knowledge & satisfaction that accompany the successful completion of any task would be incomplete without mention of people who made it possible, whose guidance and encouragement crowned my effort with success. I would like to thank all and acknowledge the help I have received to carry out this Mini Project.

I would like to convey my thanks to Head of Department **Dr. ASHA T** for being kind enough to provide the necessary support to carry out the mini project.

I am most humbled to mention the enthusiastic influence provided by the lab in-charges **Prof. Kanchan Purohit** and **Prof. Archana A.**, on the project for their ideas, time to time suggestions, for being a constant guide and co-operation showed during the venture and making this project a great success.

I would also take this opportunity to thank my friends and family for their constant support and help. I'm very much pleasured to express my sincere gratitude to the friendly co-operation showed by all the staff members of Computer Science Department, BIT.

DHANANJAI
NIRMAL
1BI18CS049

Contents

CHAPTER-1

INTRODUCTION

INTRODUCTION

1.1 Introduction

URBAN-SERVE provides a platform that allows skilled and experienced professionals to connect with users looking for specific services. All the professionals, though experienced and skilled, undergo intensive training modules before being allowed to list their services on the platform. Once on the platform, our match-making algorithm identifies professionals who are closest to the users' requirements and available at the requested time and date.

"Our Mission is to empower millions of service professionals by delivering services at-home in a way that has never been experienced before."

1.2 Problem Statement

To design and implement a database for the purpose of Service Booking Platform where the user has a provision of viewing the service and placing a request of a desired service which when accepted will be considered as a booking .

CHAPTER-2

BACK END DESIGN

BACK END DESIGN

2.1 Conceptual Design

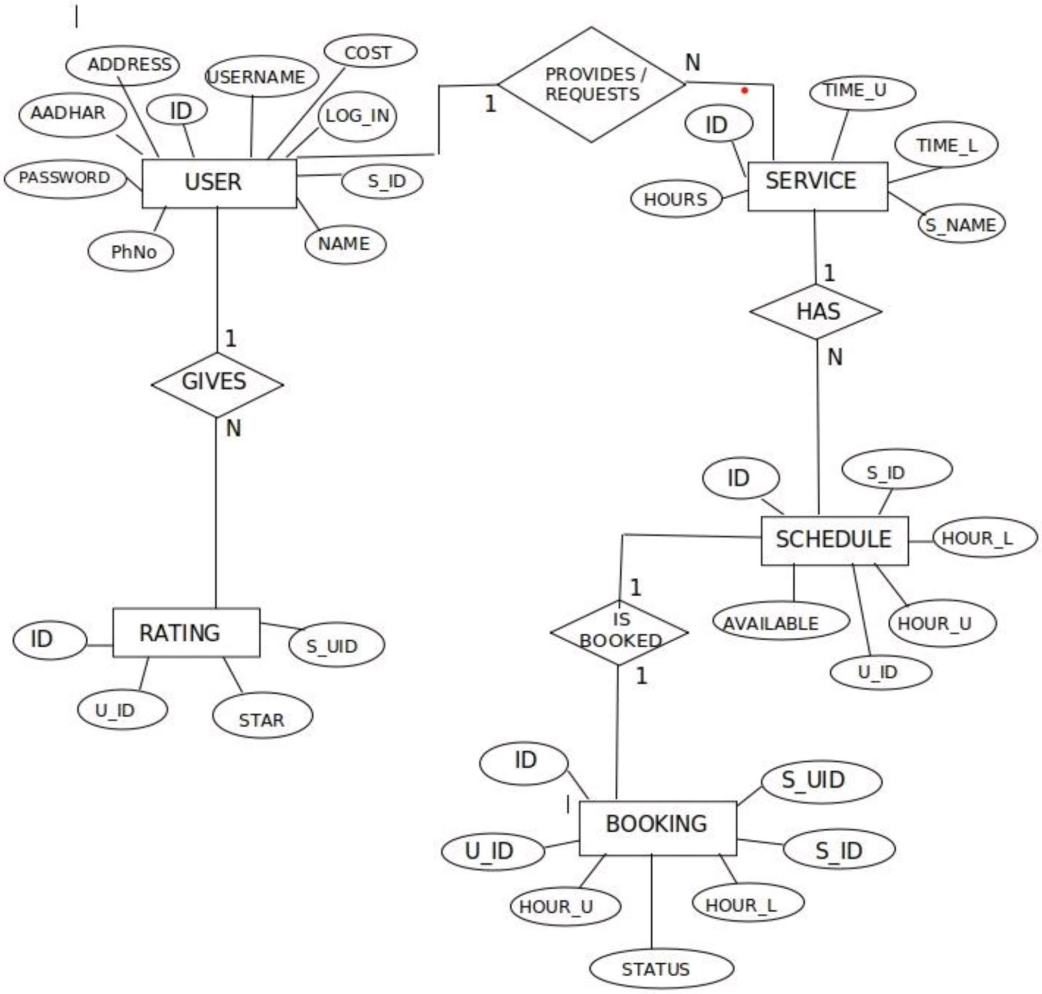


Figure 2.1.1: ER-Diagram-Timetable application

2.2 Logical Database Design

- USER : (id, name, username, aadhar, ph_no, password, logged_in,cost,address,s_id,cost)
- BOOKING : (id, u_id, s_id, s_uid, status, hour_l, hour_u)
- SCHEDULE : (id, u_id, s_id, hour_u, hour_l, available)
- RATING :(id, u_id, s_uid, star)
- SERVICE : (s_name, id, hours, time_u, time_l)

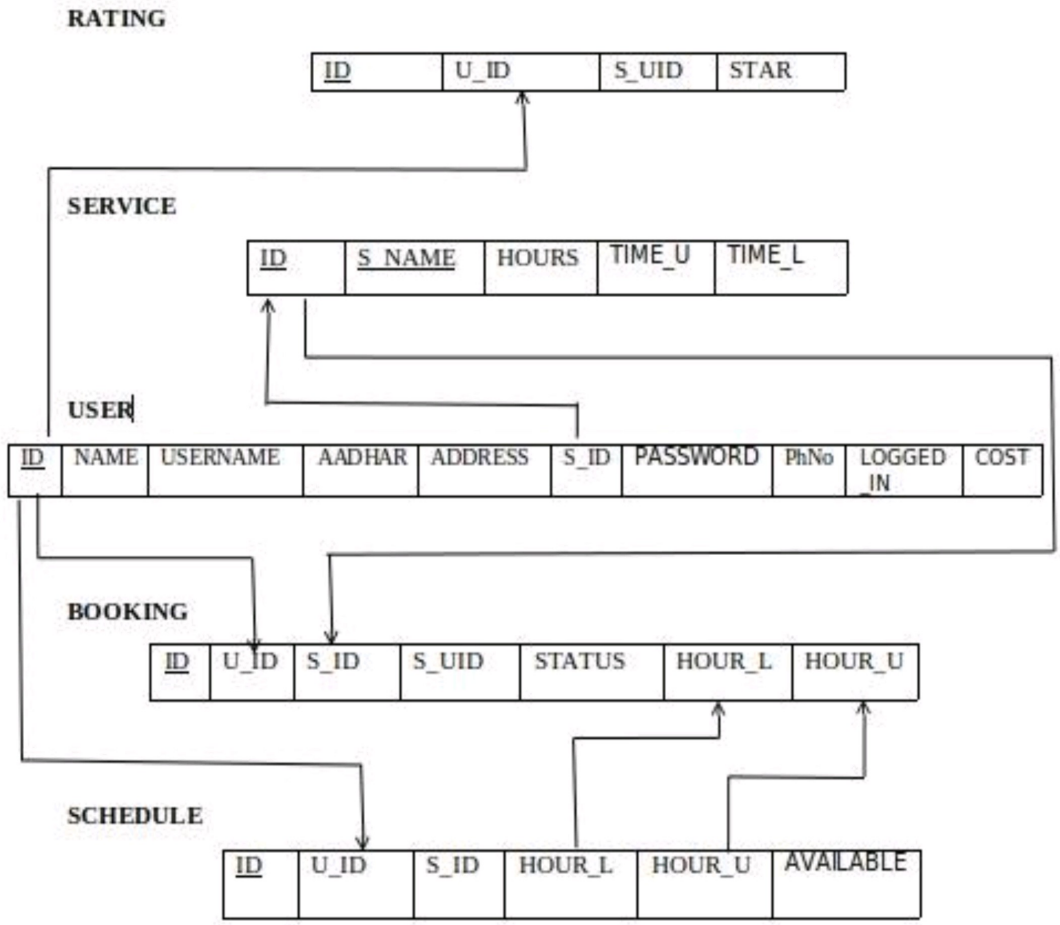


Figure 2.2.1: Relational Mapping-Timetable application

2.3 NORMALIZATION

Database Normalization is a technique of organizing the data in database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like avoid insertion anomaly, update anomaly & deletion anomaly. It is a multi step process that puts data into tabular form by removing duplicated data from the relation tables.

Normalization is used for mainly two purposes,

- Eliminating Redundant(useless) data
- Ensuring data dependencies make sense i.e data is logically stored

2.3.1: CONDITIONS FOR NORMALIZATION

First Normal Form(1NF):

As per First Normal Form, no two Rows of data must contain repeating group of information i.e. each set of columns must have a unique value, such that multiple columns cannot be used to fetch the same row. Each table should be organized into rows, and each row should have a primary key that distinguishes it as unique.

Second Normal Form (2NF):

As per the Second Normal Form there must not be any partial dependency of any column on primary key. It means that for a table that has concatenated primary key, each column in the table that is not part of the primary key must depend upon the entire concatenated key for its existence. If any column depends only on one part of the concatenated key, then the table fails **Second normal form**.

Third Normal Form (3NF):

Third Normal form applies that every non-prime attribute of table must be dependent on primary key, or we can say that, there should not be the case that a non-prime attribute is determined by another non-prime attribute. So this transitive functional dependency should be removed from the table and also the table must be in Second Normal form.

2.3.2 NORMALIZATION OF USER TABLE

USER : (id, name, username, aadhar, ph_no, password, logged_in,cost,address,s_id,cost)

Example:-

User Table-

id	name	username	aadhar	address	s_id	password	phone	logged_in	cost
1	xyx	xyz@	1212	home	2	*******	7878	1	300

Service Table-

ID	S_name	Hours	Time_l	Time_u
1	user	3	4	7
2	baby-sitter	3	4	7
3	dog_sitter	3	4	7
4	driver	3	4	7

The Primary Key is (id + s_id)

JUSTIFIICATION:

First Normal Form
In User, all the attributes are atomic and there cannot be duplicate rows.Hence it is in 1NF.

Second Normal Form

This is already in 2NF since every non key attribute is fully dependent on the primary key.

Third Normal Form

Since there is no transitive functional dependency, therefore table is already is 3NF

2.3.3 NORMALIZATION OF SERVICE TABLE

SERVICE : (s_name, id, hours, time_u, time_l)

ID	S_name	Hours	Time_l	Time_u
1	user	3	4	7
2	baby-sitter	3	4	7
3	dog_sitter	3	4	7
4	driver	3	4	7

The primary key is ID

JUSTIFIICATION:

First Normal Form

In Service, all the attributes are atomic and there cannot be duplicate rows.

Hence it is in 1NF.

Second Normal Form

This is already in 2NF since every non key attribute is fully dependent on the primary key.

Third Normal Form

Since there is no transitive functional dependency, therefore the table is already 3NF .

2.3.4 NORMALIZATION OF BOOKING TABLE

BOOKING : (id, u_id, s_id, s_uid, status, hour_l, hour_u)

Example-:

Booking

id	u_id	s_id	s_uid	status	hour_l	hour_u
1	2	2	1	R	4	5

User table

id	name	username	aadhar	address	s_id	password	phone	logged_in	cost
1	xyx	xyz@	1212	home	2	*******	7878	1	300
2	yyy	yyy@	2345	work	1	*******	9876	0	null

The Primary Key is (U_ID + S_UID)

JUSTIFIICATION:

First Normal Form
In Booking, all the attributes are atomic and there cannot be duplicate rows.

Hence it is in 1NF.

Second Normal Form

This is already in 2NF since every non key attribute is fully dependent on the primary key.

Third Normal Form

Since there is no transitive functional dependency, therefore the table is already 3NF .

2.3.5 NORMALIZATION OF SCHEDULE TABLE

SCHEDULE : (id, u_id, s_id, hour_u, hour_l, available)

Example:-

Schedule -

id	u_id	s_id	hour_l	hour_u	available
1	1	2	4	5	0

User table-

id	name	username	aadhar	address	s_id	password	phone	logged_in	cost
1	xyx	xyz@	1212	home	2	*******	7878	1	300

Service table-

ID	S_name	Hours	Time_l	Time_u
1	user	3	4	7
2	baby-sitter	3	4	7
3	dog_sitter	3	4	7
4	driver	3	4	7

The Primary Key is (U_ID + S_ID)

JUSTIFIICATION:

First Normal Form

In SCHEDULE, all the attributes are atomic and there cannot be duplicate rows.

Hence it is in 1NF.

Second Normal Form

This is already in 2NF since every non key attribute is fully dependent on the primary key.

Third Normal Form

Since there is no transitive functional dependency, therefore the table is already 3NF .

2.3.6 NORMALIZATION OF RATING TABLE

RATING :(id, u_id, s_uid, star)

Example:-

Rating

ID	U_ID	S_UID	STAR
1	2	1	4

User table

id	name	username	aadhar	address	s_id	password	phone	logged_in	cost
1	xyx	xyz@	1212	home	2	*******	7878	1	300
2	yyy	yyy@	2345	work	1	*******	9876	0	null

The Primary Key is (U_ID + S_UID)

JUSTIFIICATION:

First Normal Form

In Orders, all the attributes are atomic and there cannot be duplicate rows.

Hence it is in 1NF.

Second Normal Form

This is already in 2NF since every non key attribute is fully dependent on the primary key.

Third Normal Form

Since there is no transitive functional dependency, therefore the table is already 3NF .

CHAPTER-3

FRONT END DESIGN

3.1 FRONT END DESIGN

HTML (Hyper Text Markup Language)

HTML is a standard markup language for creating web pages and web applications with Cascading Style Sheet (CSS) and JavaScript, it forms a triad of corner stone technologies of the World Wide Web.

CSS (Cascading Style Sheet)

CSS is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

BOOTSTRAP

Bootstrap is a free front-end framework for faster and easier web development.Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins.Bootstrap also gives you the ability to easily create responsive designs.

3.1 Screen layout design for Web Pages and forms

HTML <p> TAG:

The HTML <p> element represents a paragraph of text. Paragraphs are usually represented in visual media as blocks of text that are separated from adjacent blocks by vertical blank space and/or first-line indentation. Paragraphs are block level elements.

HTML TAG:

The HTML element is the generic container for flow content and does not inherently represent anything. Use it to group elements for purposes such as styling (using the class or id attributes), marking a section of a document in a different language
(using the lang attribute), and so on\

.

**HTML
 TAG:**

The HTML
 element produces a line break in text (carriage-return). It is useful for writing a poem or an address, where the division of lines is significant.

HTML <input> TAG:

The HTML <input> element is used to create interactive controls for web-based forms in order to accept data from the user. An <input> works varies considerably depending on the value of its type attribute, hence the different types are covered in their own separate reference pages. If this attributes is not specified, the default type adopted is text.

<input> elements of type text create basic, single-line inputs.

The **HTML <head> element** defines a set of rows defining the head of the columns of the table.

The **HTML <body> element** groups one or more <tr> elements as the body of a <table> element.

HTML <form> TAG:

The HTML <form> element represents a document section that contains interactive controls to submit information to a web server. It is possible to use the :valid and :invalid CSS pseudo-classes to style a <form>element. The HTTP method that the browser uses to submit the form. Possible values are:

post: Corresponds to the HTTP POST method ; form data are included in the body of the form and sent to the server.

get: Corresponds to the HTTP GET method; form data are appended to the action attribute URI with a '?' as separator, and the resulting URI is sent to the server. Use this method when the form has no side-effects and contains only ASCII characters. This value can be overridden by a form method attribute on a <button> or <input>element.

action: The URI of a program that processes the form information. This value can be overridden by a form action attribute on a <button> or <input> element.

3.2 CONNECTIVITY

3.2.1 Connecting to a MySQL database

For all Spring applications, you should start with the Spring Initializr. The Initializr offers a fast way to pull in all the dependencies you need for an application and does a lot of the set up for you. This example needs the Spring Web, Spring Data JPA, and MySQL Driver dependencies.

<?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>2.3.2.RELEASE</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>accessing-data-mysql</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>accessing-data-mysql</name>
	<description>Demo project for Spring Boot</description>

	<properties>
		<java.version>1.8</java.version>
	</properties>

	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</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-test</artifactId>
			<scope>test</scope>
			<exclusions>
				<exclusion>
					<groupId>org.junit.vintage</groupId>
					<artifactId>junit-vintage-engine</artifactId>
				</exclusion>
			</exclusions>
		</dependency>
	</dependencies>

	<build>
		<plugins>
			<plugin>
```

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

</project>
```

TRIGGERS & STORED PROCEDURE

Trigger
A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server. DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view.

Stored Procedure
A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. ... You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

```
//TRIGGER

CREATE DEFINER=`root`@`localhost` TRIGGER `ecommerce`.`user_AFTER_INSERT` AFTER INSERT ON `user` FOR
EACH ROW
BEGIN



 call proc();



END


//STORED PROCEDURE

CREATE DEFINER=`root`@`localhost` PROCEDURE `proc`()
begin
 declare total int;

   set total:= (select count(*) from user where SERVICE_ID = 1 );
   update tally set SERVICE_SUBSCRIBERS = total;


    set total:= (select count(*) from user where SERVICE_ID != 1 );
   update tally set TOTAL_PROVIDERS = total;

end
```

CHAPTER-4

MAJOR MODULES

MAJOR MODULES

The functionalities of our Ecommerce website are as follows-

1. **Home**- This page is loaded as soon as the Application is launched,gives a brief description about our application and also a brief description of the services which can be booked using this application.

2. **User Sign-up**- You can register to access Services and request them. It contains a form to enter customer details like name, email, password, address and phone number fields and a submit button to submit the form. If your password and username field are unique , then you will be redirected to the login page.
3. **User Login**- The user who is registered into the database can login using his username and password into the website. This module contains an username and password field and a login button to login into his account. After logging in, the user will be redirected to the main index page.

4. **Service user module**- Through this module, we can select the service we need and then view all the options available for a particular service.Using the request a request for booking a service for a particular time can be requested. there will be a MY Booking tab displaying all the already booked services along with the details also the user is capable of rating the service provider based on the quality of service they provide.
5. **Service provider module**- Through this module we can accept/decline tout pending requests , see our declined and accepted requests.also there's a tab provided for the user to see all the ratings given to him/her by different users who have booked them in the past
6. **Logout**- By clicking on the logout button, the user will be logged out the site and redirected to the home page where in order to access products, he will have to login again.

CHAPTER-5

IMPLEMENTATION

IMPLEMENTATION

5.1 Create Statements

```
CREATE TABLE `user` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `NAME` varchar(255) NOT NULL,
  `USER_NAME` varchar(255) NOT NULL,
  `PH_NO` bigint(20) DEFAULT NULL,
  `PASSWORD` varchar(255) NOT NULL,
  `ADDRESS` varchar(255) NOT NULL,
  `AADHAR` bigint(20) DEFAULT NULL,
  `SERVICE_ID` int(11) NOT NULL,
  `LOGGED_IN` varchar(1) NOT NULL DEFAULT 'N',
  `COST` int(11) DEFAULT NULL,
  `TOTAL_USERS` int(11) DEFAULT '0',
  `TOTAL_PROVIDERS` int(11) DEFAULT '0',
  PRIMARY KEY (`ID`),
  UNIQUE KEY `USER_NAME` (`USER_NAME`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=latin1 |
```

```
CREATE TABLE `service` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `S_NAME` varchar(255) NOT NULL,
  `TIME_U` int(11) NOT NULL,
  `TIME_L` int(11) NOT NULL,
  `HOURS` int(11) NOT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1 |
```

```
CREATE TABLE `booking` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `U_ID` int(11) NOT NULL,
  `S_ID` int(11) NOT NULL,
  `S_UID` int(11) NOT NULL,
  `HOUR_L` int(11) NOT NULL,
  `HOUR_U` int(11) NOT NULL,
  `STATUS` varchar(1) NOT NULL DEFAULT 'R',
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=latin1 |

CREATE TABLE `schedule` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `U_ID` int(11) NOT NULL,
  `S_ID` int(11) NOT NULL,
  `HOUR_L` int(11) NOT NULL,
  `HOUR_U` int(11) NOT NULL,
  `AVAILABLE` int(11) NOT NULL DEFAULT '1',
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=22 DEFAULT CHARSET=latin1 |



CREATE TABLE `rating` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `U_ID` int(11) NOT NULL,
  `S_UID` int(11) NOT NULL,
  `STAR` int(11) NOT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1 |
```

JAVA , REACT & HTML CODE:

WEBPAGES:

HOMEPAGE

```
import React,{useState,setState} from 'react';
import { useHistory,Link} from "react-router-dom";
import "./css/style.css"
import "./lib/bootstrap/css/bootstrap.min.css"
import "./lib/font-awesome/css/font-awesome.min.css"
import "./lib/animate/animate.min.css"
import "./lib/owlcarousel/assets/owl.carousel.min.css"
import "./lib/lightbox/css/lightbox.min.css"
import Cookies from 'universal-cookie';
function Home() {
   return (<div>
       <header id="header" style={{background:"white"}}>
<div id="topbar">
  <div class="container">
   <div class="social-links">
     <a href="#" class="twitter"><i class="fa fa-twitter"></i></a>
     <a href="#" class="facebook"><i class="fa fa-facebook"></i></a>
     <a href="#" class="linkedin"><i class="fa fa-linkedin"></i></a>
     <a href="#" class="instagram"><i class="fa fa-instagram"></i></a>
   </div>
  </div>
</div>
<div class="container">
  <div class="logo float-left">
      <h1 class="text-light"><a href="#intro" class="scrollto"><span>URBAN-
SERvE</span></a></h1>
  </div>
<nav class="main-nav float-right d-none d-lg-block">
   <ul>
```

```
    <li class="active"><a href="#intro">Home</a></li>
    <li><a href="#about">About Us</a></li>
    <li><a href="#services">Services</a></li>
    <li><a href="#portfolio">Portfolio</a></li>
    <li><a href="#team">Team</a></li>
    <li><Link to="/login">LOGIN / SIGN-UP</Link></li>
    <li><a href="#footer">Contact Us</a></li>
   </ul>
  </nav>
  </div>
</header>
   <div class="container">
<div class="logo float-left">
 <h1 class="text-light"><a href="#intro" class="scrollto"><span>Rapid</span></a></h1>
  </div>
<nav class="main-nav float-right d-none d-lg-block">
  <ul>
    <li class="active"><a href="#intro">Home</a></li>
    <li><a href="#about">About Us</a></li>
    <li><a href="#services">Services</a></li>

    {/* <li class="drop-down"><a href="">Drop Down</a>
      <ul>
        <li><a href="#">Drop Down 1</a></li>
_       <li class="drop-down"><a href="#">Drop Down 2</a>
          <ul>
            <li><a href="#">Deep Drop Down 1</a></li>
            <li><a href="#">Deep Drop Down 2</a></li>
            <li><a href="#">Deep Drop Down 3</a></li>
            <li><a href="#">Deep Drop Down 4</a></li>
            <li><a href="#">Deep Drop Down 5</a></li>
          </ul>
        </li>
        <li><a href="#">Drop Down 3</a></li>
```

```
        <li><a href="#">Drop Down 4</a></li>
        <li><a href="#">Drop Down 5</a></li>
      </ul>
    </li> */}
    <li><a href="#footer">Contact Us</a></li>
  </ul>
</nav>
</div>
<section id="intro" class="clearfix">
    <div class="container d-flex h-100">
      <div class="row justify-content-center align-self-center">
        <div class="col-md-6 intro-info order-md-first order-last">
          <h2>Rapid Services<br/>for all Your <span>Needs!</span></h2>
          <div>
            <Link to="/login" class="btn-get-started scrollto">Get Started</Link>
          </div>
        </div>
         <div class="col-md-6 intro-img order-md-last order-first">
          <img src="intro-img.svg" alt="" class="img-fluid"/>
        </div>
      </div>
  </div>
  </section>
  <section id="about">
  <div class="container">cd lock
     <div class="row">
 <div class="col-lg-5 col-md-6">
           <div class="about-img">
             <img src="./img/about-img.jpg" alt=""/>
           </div>
         </div>

         <div class="col-lg-7 col-md-6">
           <div class="about-content">
```

```
            <h2>About Us</h2>
            <br></br>
            <h3>Urban serve  provides a platform that allows skilled, friendly and experienced professionals to connect with users looking for specific services. </h3>
            <br></br>
            <p>All the professionals, though experienced and skilled undergo intensive training modules before being allowed to list services on the platform. </p>
            <p>Once ton the platform, our match- making algorithm identifies professionals who are closest to the users' requirements and available at the requested time and date.</p>
            <ul>
              <li><i class="ion-android-checkmark-circle"></i> </li>
              <li><i class="ion-android-checkmark-circle"></i> </li>
              <li><i class="ion-android-checkmark-circle"></i></li>
            </ul>
          </div>
        </div>
      </div>
    </div>
</section>
<section id="services" class="section-bg">
    <div class="container">
  <header class="section-header">
        <h3>Services</h3>
        <p>If you can think of a better service tell me..</p>
      </header>

      <div class="row">

        <div class="col-md-6 col-lg-4 wow bounceInUp" data-wow-duration="1.4s">
          <div class="box">
            <div class="icon" style={{background:"#fceef3"}}><i class="ion-ios-analytics-outline" style={{color: "#ff689b"}}></i></div>
            <Link to="/login"><h4 class="title">Dog Keeping</h4></Link>
```

```
            <p class="description"> Dog walkers will walk dogs for clients and follow agreed-upon routes and schedules. They will be providing companionship during work hours or vacation absences as well.</p>
          </div>
        </div>
        <div class="col-md-6 col-lg-4 wow bounceInUp" data-wow-duration="1.4s">
          <div class="box">
            <div class="icon" style={{background: "#e6fdfc"}}><i class="ion-ios-bookmarks-outline" style={{color: "#e98e06"}}></i></div>
            <Link to="/login"><h4 class="title">Baby sitting</h4></Link>
            <p class="description">Baby sitting entails taking full responsibility for the health and welfare of children when their parents are away. Sitters are entrusted with every aspect of child care from meals to hygiene to educational activities.</p>
          </div>
        </div>

        <div class="col-md-6 col-lg-4 wow bounceInUp" data-wow-delay="0.1s" data-wow-duration="1.4s">
          <div class="box">
            <div class="icon" style={{background: "#fff0da"}}><i class="ion-ios-paper-outline" style={{color: "#3fcdc7"}}></i></div>
            <Link to="/login"><h4 class="title">Driver</h4></Link>
            <p class="description">Driver responsibilities include arranging cleaning and maintenance services for the vehicle, planning each route based on road and traffic conditions and managing payments.</p>
          </div>
        </div>
       </div> </div>
   </section>   </div>);
export default Home;
```

LOGIN PAGE

```
import React,{useState,setState} from 'react';
import { useHistory } from "react-router-dom";
import Button from 'react-bootstrap/Button';
import MyVerticallyCenteredModal from './Modal.js';
import './App.css';
import Cookies from 'universal-cookie';
const cookies = new Cookies();
function App() {
  let history = useHistory();
  const [name, setName] = useState("");
  const[username,setUsername]=useState("");
  const [address,setAddress]=useState("");
  const [password, setPassword] = useState("");
  const [phNo, setPhNo] = useState("");
  const [user, setUser] = useState("1");
  const [aadhar, setAadhar] = useState("")
  const [cost,setCost]=useState("0");
  const [modalShow, setModalShow] = React.useState(false);

  //const [visibility,setVisibility]=useState("1");


    const MyAlert=()=>{
 alert("SUCCESSFULLY LOGGED INN !
  const handleClickLogin = (e)=>{  console.log(username);
    const article = {userName:username,password:password};
    console.log(article);
    fetch('http://localhost:8080/user/login', {
    method: 'post',
    headers: {
      'Accept': 'application/json, text/plain, */*',
      'Content-Type': 'application/json',
```

```
    'Access-Control-Allow-Origin': '*',
   },
   body: JSON.stringify(article)
   })
   .then(res => res.json())
   .then(res => {
    //alert(res);
    MyAlert();
 if(res !=null ){
              console.log("modal");
 cookies.set('ID', res.id, { path: '/' });
      if(res.serviceId != 1 ){
       history.push("/loggedprovider");
}
      else{
       history.push("/loggeduser");
      }  }
   }   )
  .catch((err) => console.log(err));  }
 const handleClickSignUp=(e)=>{
     const article = {
name:name,userName:username,password:password,address:address,phNo:phNo,serviceId:u
ser,aadhar,cost:cost};
   console.log(article);
      fetch('http://localhost:8080/user/signup', {
   method: 'post',
   headers: {
    'Accept': 'application/json, text/plain, */*',
    'Content-Type': 'application/json',
    'Access-Control-Allow-Origin': '*',
   },
   body: JSON.stringify(article)
   })
   .then(res => res.json())
```

```
   .then(res => alert("SUCCESSFULLY SIGNED UP !!!"))
   .catch((err) => console.log(err));
   history.push("/");
  }
  return (
   <div className="marginn">

   {/* {modalShow && < MyVerticallyCenteredModal  show={modalShow}   onHide={()
=> setModalShow(false)} />} *  <div className="login-wrap">
  <div className="login-html">
   <input id="tab-1" type="radio" name="tab" className="sign-in"/><label for="tab-1"
className="tab">Sign In</label>
   <input id="tab-2" type="radio" name="tab" className="sign-up"/><label for="tab-2"
className="tab">Sign Up</label>
   <div className="login-form">
    <div className="sign-in-htm">
     <div className="group">
      <label for="user" className="label">Username</label>
      <input id="user" type="text"  value={username}
onChange={(e)=>setUsername(e.target.value)} className="input" />
     </div>
     <div className="group">
      <label for="pass" className="label">Password</label>
      <input id="password" type="password" className="input" value={password}
onChange={(e)=>setPassword(e.target.value)} data-type="password"/>
     </div>
      <div className="group">
      <input type="submit" onClick={handleClickLogin} className="button" value="Sign
In"/>
     </div>
     <div className="hr"></div>
      <div className="foot-lnk">
      <img src="https://img.icons8.com/cotton/100/000000/flip-flops--v3.png"/>
       </div>
```

```
    </div>
    <div className="sign-up-htm">
     <div className="group">
      <label for="user" className="label">Name</label>
      <input id="name" type="text" value={name} className="input"
onChange={(e)=>setName(e.target.value)}/>
     </div>
     <div className="group">
      <label for="pass" className="label">Username</label>
      <input id="username" className="input" type="text"  value={username}
onChange={(e)=>setUsername(e.target.value)}/>
     </div>
     <div className="group">
      <label for="pass" className="label">Password</label>
      <input id="password"  type="password" className="input" data-type="password"
value={password} onChange={(e)=>setPassword(e.target.value)}/>
     </div>
     <div className="group">
      <label for="pass" className="label">Address</label>
      <input id="address" type="text" value={address} className="input"
onChange={(e)=>setAddress(e.target.value)}/>
     </div>
     <div className="group">
      <label for="pass" className="label">Phone Number</label>
      <input id="phNo" type="number" value={phNo}
className="input"onChange={(e)=>setPhNo(e.target.value)}/>
     </div>
     <div className="group">
      <label for="pass" className="label">Aadhar No.</label>
      <input id="type" className="input" value={aadhar} type="number"
onChange={(e)=>setAadhar(e.target.value)}/>
     </div>
     <div className="group marg">
      <div className="side marg">
```

```
        <label for="pass" className="label">Type Of User</label >
        <select name="service" className="input" id="service"
onChange={(e)=>setUser(e.target.value)}>
                  <option value="1">Service User</option>
                  <option value="2">Dog Caretaker</option>
                  <option value="3">Baby Sitter</option>
                  <option value="4">Driver</option>
              </select>
        </div>
        { user != 1 &&  <div className="side1 marg">
          <label for="pass" className="label">Booking Cost (in Rs.)</label>
          <input id="type" className="input" value={cost} type="number"
onChange={(e)=>setCost(e.target.value)}/>

        </div>
        }
        </div>

        <div className="group siz">
        <center> <input type="submit" onClick={handleClickSignUp} className="button"
style={{marginLeft:"60px"}} value="Sign Up"/></center>
        </div>
        <div className="hr"></div>
        <div className="foot-lnk">
          <label for="tab-1">Already Member?</label>
        </div>
      </div>
    </div>
  </div>
</div>
</div>
     );}

export default App;
```

BOOKING PAGE:

```
import React,{useEffect,useState} from "react";
import Cookies from 'universal-cookie';
import RequestingComponent from "./RequestingComponent";
const cookies = new Cookies();
const Mybooking = () =>
{
   const [request, setRequest] = useState("");

   useEffect(() => {
      const article=cookies.get('ID');
      console.log("ksbkbjf");
      fetch(`http://localhost:8080/user/mybooking?id=${article}`, {
   method: 'get',
   headers: {
    'Accept': 'application/json, text/plain, */*',
    'Content-Type': 'application/json',
    'Access-Control-Allow-Origin': '*',
   },
   })
   .then(res => res.json())
   .then(res => {

       //  alert("jsijf");
         setRequest(res);
    })
      .catch((err) => console.log(err))}, [])

   // const trial=;

   return(<div>
      {Object.keys(request).map((item)=><RequestingComponent key={item}
req={request[item]} />)}
```

```
    </div>

    );

}
export default Mybooking;
```

RATING PAGE:

```
import React,{useState,useEffect} from "react"
import './RequestComponent.css';
import Card from 'react-bootstrap/Card';
import Button from 'react-bootstrap/Button';
import DropdownButton from 'react-bootstrap/DropdownButton';
import Dropdown from 'react-bootstrap/Dropdown';
import Image from 'react-bootstrap/Image';
import Container from 'react-bootstrap/Container';
import Col from 'react-bootstrap/Col';
import Row from 'react-bootstrap/Row';
import RatingComponent from './RatingComponent';
import Cookies from 'universal-cookie';
const cookies = new Cookies();
function Ratings(props) {
    const [request, setRequest] = useState("");
  console.log("hy",props.req);


  useEffect(() => {
    const article=cookies.get('ID');
    fetch(`http://localhost:8080/user/rating?suid=${article}`, {
    method: 'get',
    headers: {
        'Accept': 'application/json, text/plain, */*',
        'Content-Type': 'application/json',
        'Access-Control-Allow-Origin': '*',
```

```
    },
    })
    .then(res => res.json())
    .then(res => {

      //  alert("jsijf");
        setRequest(res);
    })
    .catch((err) => console.log(err))}, [props.status]);
return(<div>
    {/* array[0]={"id":"5"} key =0 req= {"id":"5"}*/}
    {Object.keys(request).map((item)=><RatingComponent key={item} req={request[item]}
/>)}
</div>);
}export default Ratings;
```

CODE IN THE BACK-

```
package com.shoes.ecom.controller;

import com.shoes.ecom.dto.BookingAlter;
import com.shoes.ecom.dto.BookingResponse;
import com.shoes.ecom.dto.Requestingdto;
import com.shoes.ecom.dto.ScheduleDto;
import com.shoes.ecom.po.Booking;
import com.shoes.ecom.po.Rating;
import com.shoes.ecom.po.Schedule;
import com.shoes.ecom.po.User;
import com.shoes.ecom.repo.UserRepo;
import com.shoes.ecom.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;
@RestController
@CrossOrigin(origins = "*", allowedHeaders = "*")
@RequestMapping("/user")
```

```
public class UserController {
  @Autowired
    UserService userService;

    @GetMapping("/all")
    public List<User> getAll() {
        return userService.getAll();
    }
   @PostMapping("/create")
    public User create(@RequestBody User userToCreate) {
        return userService.createUser(userToCreate);
    }
  @PostMapping("/login")
    public User login(@RequestBody User userToCreate) {

        return userService.login(userToCreate);
    }
 @PostMapping("/signup")
    public Boolean signup(@RequestBody User userToCreate) {
        return userService.signup(userToCreate);
    }
   @GetMapping("/requests")
    public List<BookingResponse> requests(@RequestParam Integer suid, @RequestParam
Character status) {
        return userService.requests(suid, status);
} ;
    @PostMapping("/acceptdecline")
    public Boolean signup(@RequestBody BookingAlter alters) {
        return userService.acceptdecline(alters);
    }
;
 @GetMapping("/availableslots")
    public List<ScheduleDto> availableSlots(@RequestParam String providerType ) {
        return userService.getAvailableSlots(providerType);
```

```
}
   @GetMapping("/mybooking")
   public List<ScheduleDto> myBooking(@RequestParam Integer id) {
      return userService.getBookings(id);   }

  @PostMapping("/requesting")
   public Boolean signup(@RequestBody Requestingdto Reqalters) {
      return userService.requesting(Reqalters);
   }
@PostMapping("/logout")
   public Boolean logout(@RequestBody User log) {
      return userService.getLogs(log);
   }
 @PostMapping("/rate")
   public Boolean rate(@RequestBody Rating rate) {
      return userService.getRate(rate);
   }
  @GetMapping("/rating")
   public List<ScheduleDto> rating(@RequestParam Integer suid) {
      return userService.getRating(suid);
}
}
```

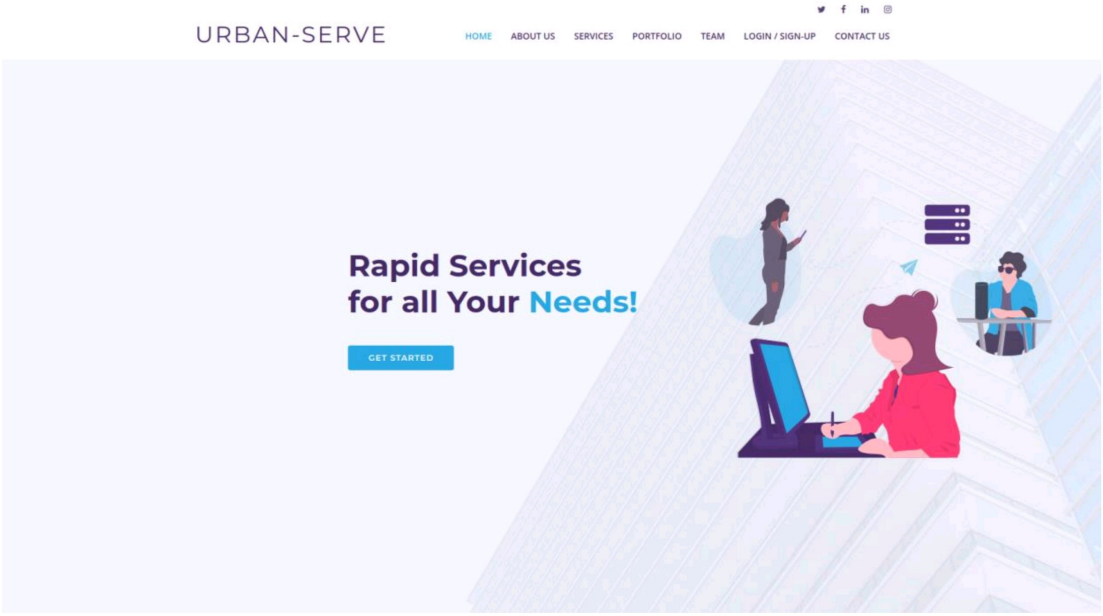
CHAPTER-6

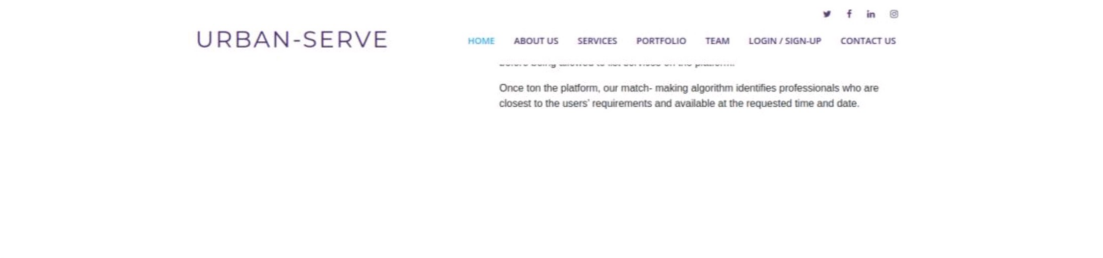
SNAPSHOTS

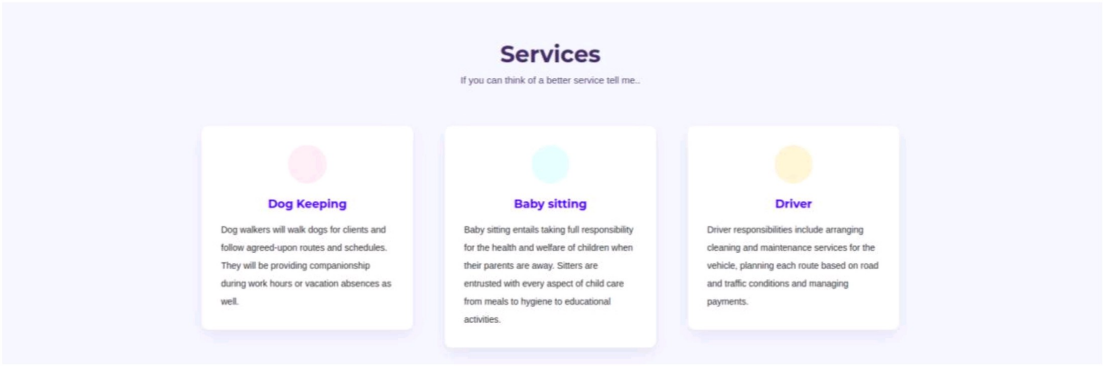
SNAPSHOTS

Home page

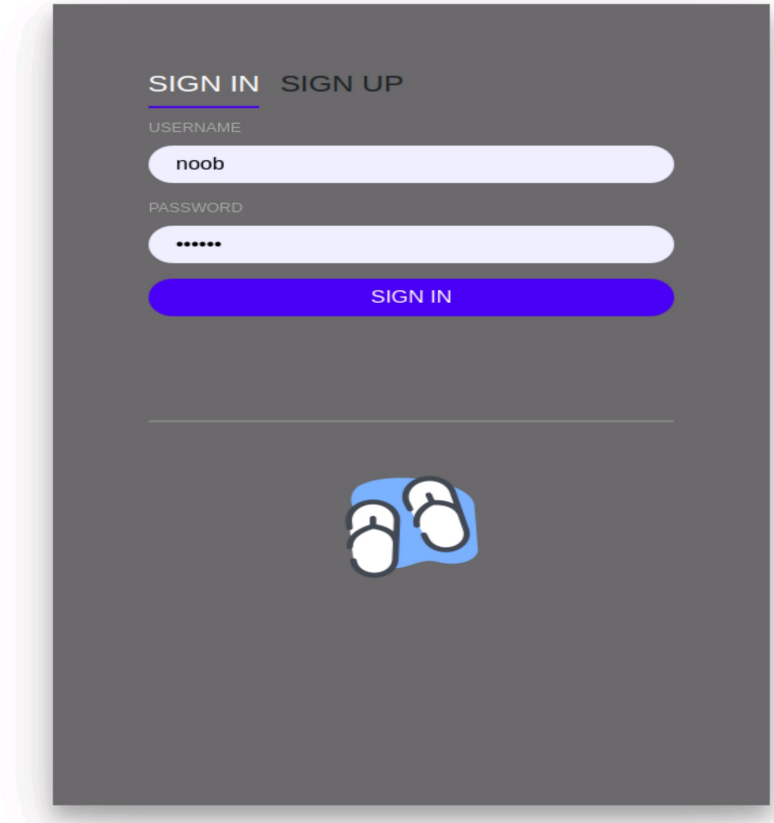
Introduction page which tells us about the web application



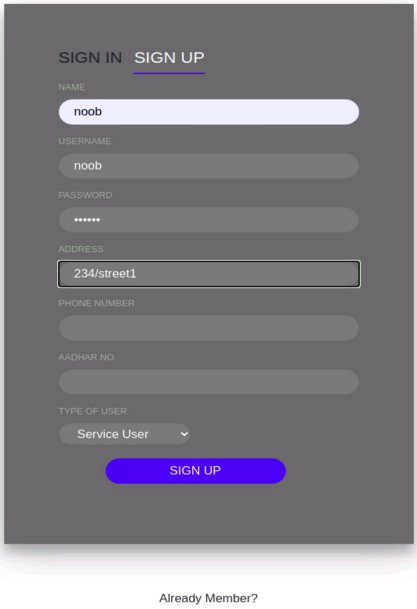




Login page



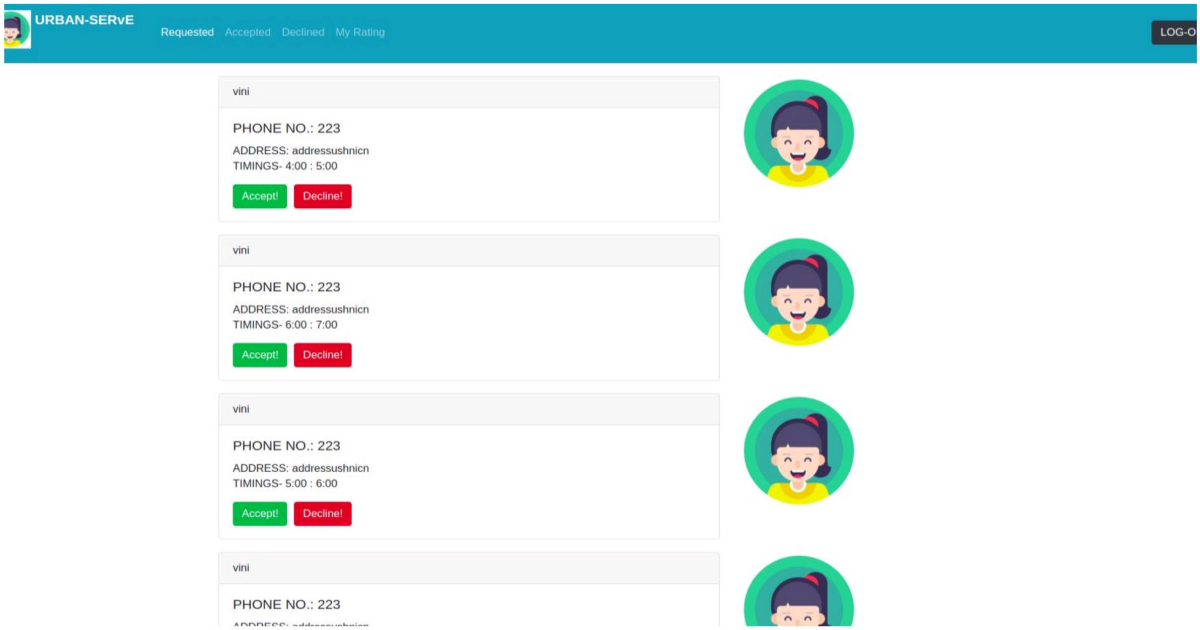
Sign-Up



Already Member?

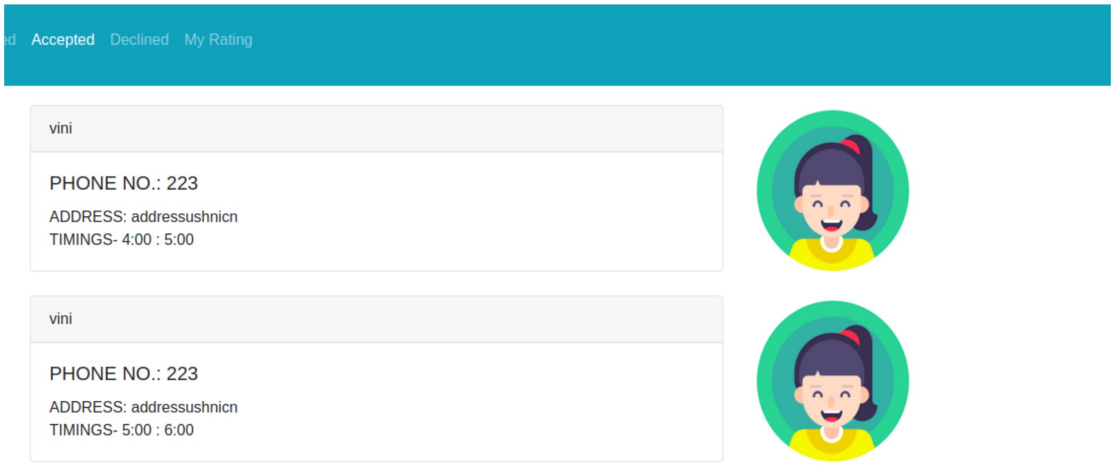
Service Provider Interface

A page displaying all requests.



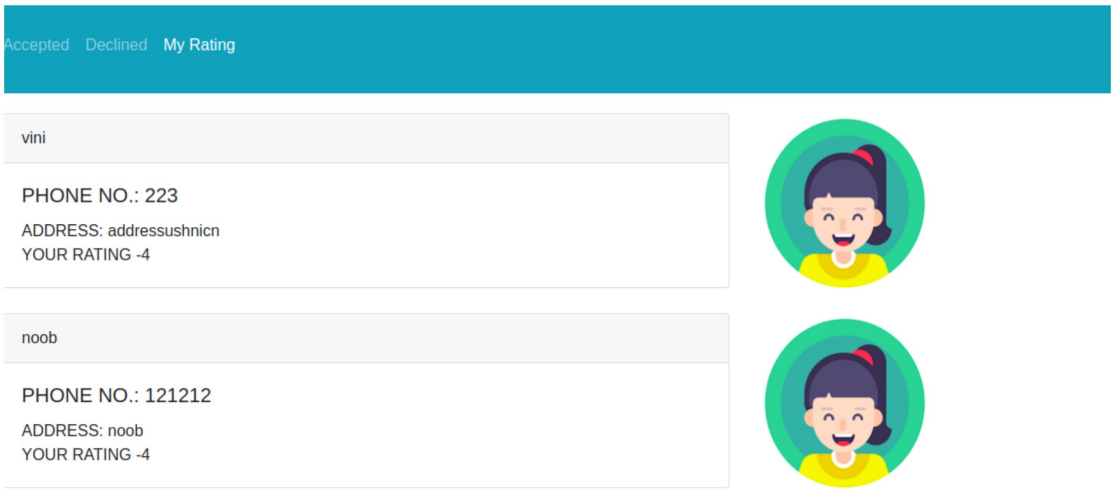
Accepted Requests

A page displaying all accepted requests.



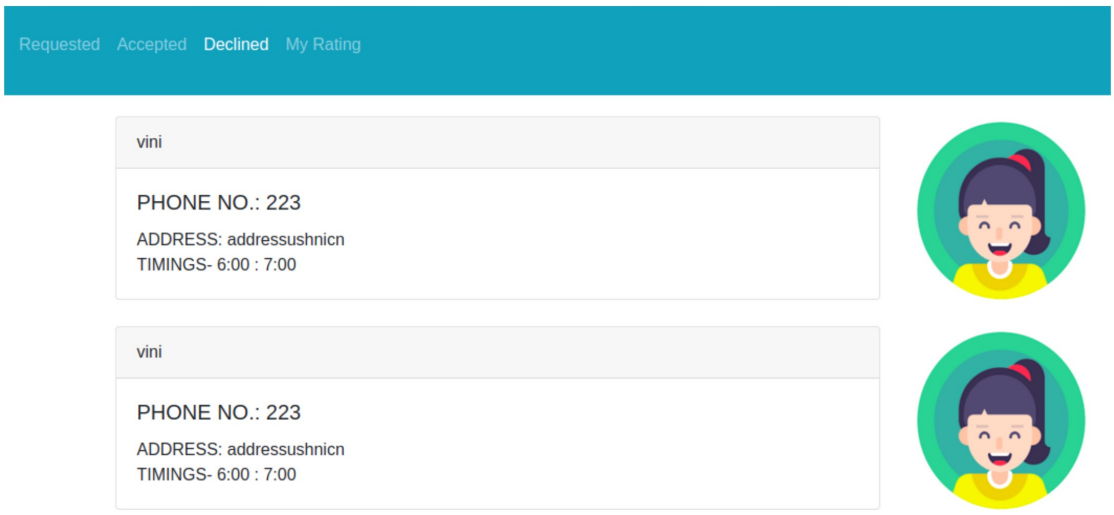
My Ratings

A page displaying all the ratings received by the user.



Declined Requests

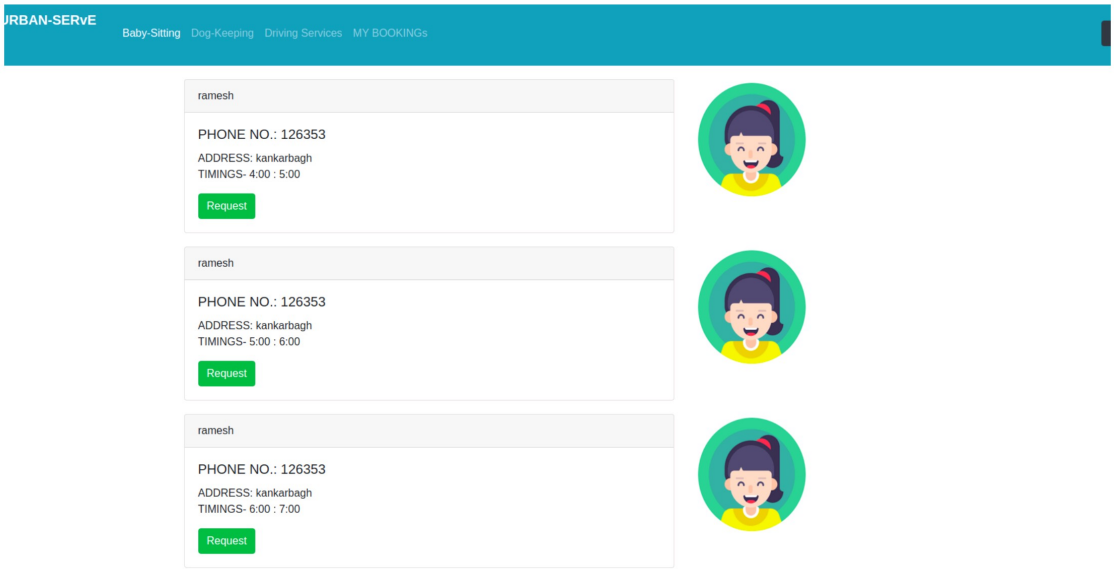
A page displaying all the requests declined by the service provider

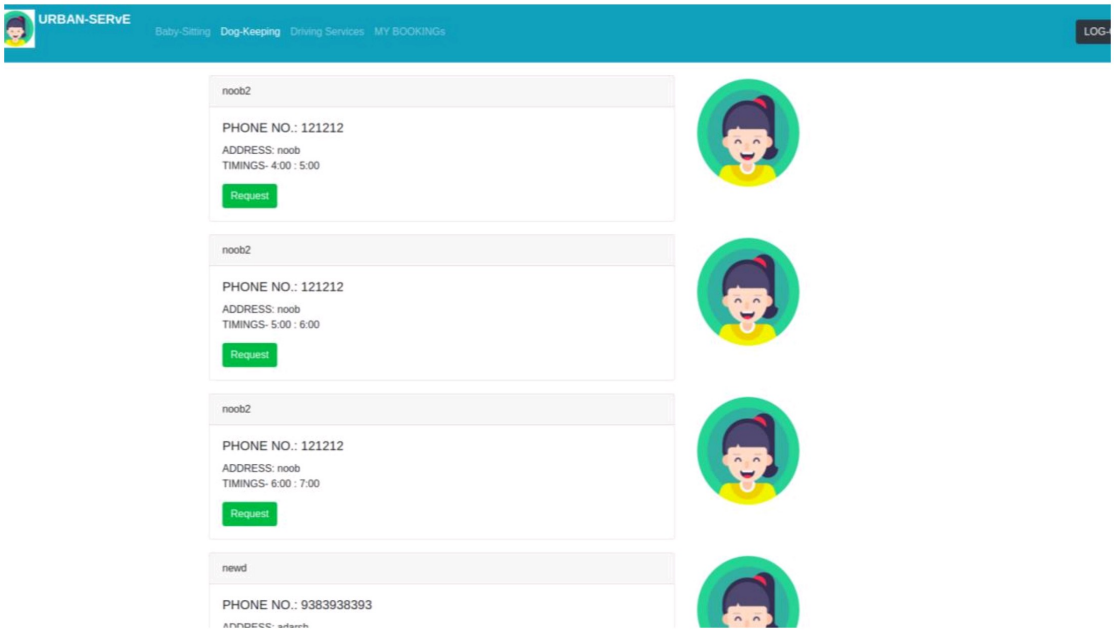


Service User

Requesting Page

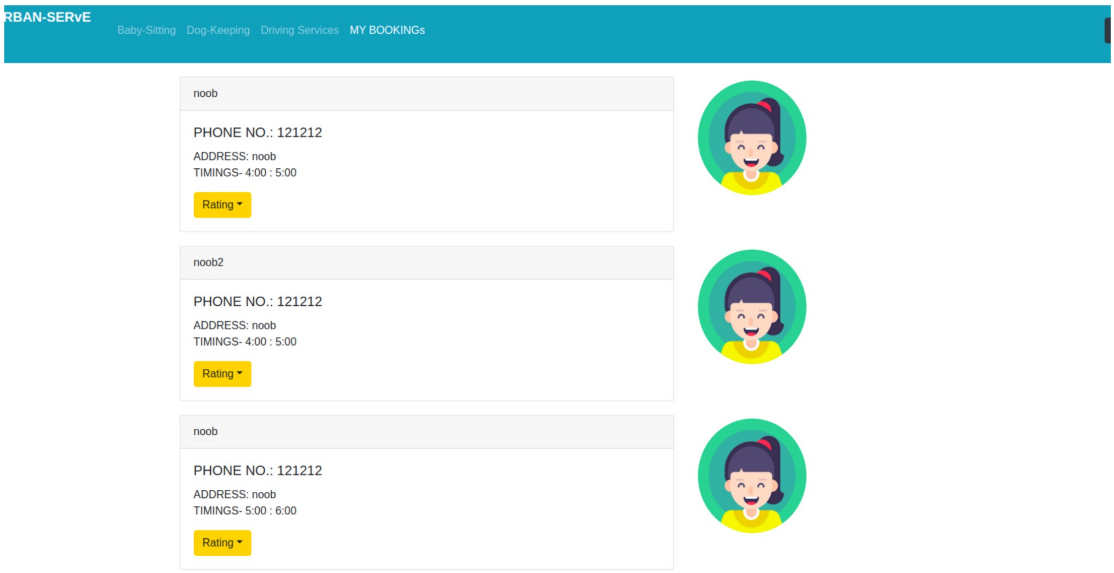
A page allows an user to requests from all available timings .





Rating Page

A page which allows the user to rate all the service providers based on their services.



CHAPTER-7

APPLICATIONS

APPLICATIONS

7.1 Applications

- There is a database to keep track of the information regarding the bookings that have been placed order of, customer details etc. Without the database systems it is hard to maintain that huge amount of data that keeps updating every millisecond.

- Service Booking Websites assist the customers to view the services and book the service online with their choice of timings and preferences method.

- It allows the Admin to keep track the record of the bookings made and schedule of the providers.

- Users have the option to sign up as a service provider and order their services to other people
- The customer can request any service with just a few clicks on a device acquainted with the Internet . Online services are a time efficient way of getting things done.

CHAPTER-8

CONCLUSION

CONCLUSION

8.1 Conclusion

In a traditional way to avail a service the customer needs to travel to a particular association and inquire about the services. This method we use currently offline services is very time consuming. Now-a-days everyone has a busy life. With such demand of time we are here to provide an Online platform where it is an easy, quick and safe way of availing services. This project is designed to meet the requirements of Online service booking windows. It has been developed in HTML, Cascading Style Sheets, Spring Boot JAVA and the Database has been built in MySQL server keeping in mind the specifications of the system.

With this platform we developed, we are hoping to reduce the time wasted, providing a variety of services, easy data flow, customer satisfaction and less hard work all at a single stop. We believe that we have accomplished our goals and are satisfied with the code we developed.