HOTEL VISTA & ROSA

FSWD-INNOVATIVE ASSIGNMENT

22BCE037-BHAYANI SAHI 22BCE039-YATRA BHIMANI DIVISION-A

About the project

In this project we have developed a Hotel and restaurant website named as Hotel Vista and Rosa Restaurant. This project helps to book rooms online in the hotel and to order food online from the restaurant.

Technologies Used

Frontend: HTML, CSS, Bootstrap, Tailwind, JS.

HTML: Hyper-text Markup Language is used to create the structure and content of the website.

CSS: Cascading Style Sheet is used to give the styling to the structure of the website.

Bootstrap: It is a popular front-end framework for developing responsive websites and web applications. It provides pre-designed templates, CSS, and JavaScript components to streamline responsiveness across different devices and screen sizes. In this project Bootstrap is used for the creation of the menu card of restaurant.

Tailwind: Tailwind CSS is a utility-first CSS framework that provides low-level utility classes to build user interfaces. In this project it is used to create the booking form for room reservation.

JS: JavaScript (JS) is commonly used in frontend web development to add interactivity and dynamic behaviour to web pages. In this project it is used in

- Booking form to check the validation of the data and to calculate the total amount of payment.
- In order form of restaurant to make the orders and to check the validity of the entered user details for the order.

Backend: Node.js, Express.js

Node.js: Node.js is an open-source, server-side JavaScript runtime environment built on Chrome's V8 JavaScript engine. It allows developers to run JavaScript code on the server, enabling the development of scalable, networked applications.

Using Node.js in a hotel and restaurant website has the following benefits:

1. ***Real-Time Updates***: Node.js excels in handling real-time interactions, making it suitable for features like real-time booking updates, and dynamic menu changes.

2***. Scalability:*** Node.js's non-blocking, event-driven architecture allows for efficient handling of concurrent connections, ensuring that the website remains responsive and scalable even under heavy loads.

3. ***Single Language Stack***: With Node.js, you can use JavaScript both on the frontend and backend, providing a unified language stack.

4.***Rich Ecosystem***: Node.js has a vast ecosystem of libraries, frameworks, and tools that can streamline development and add functionality to the website. For example, frameworks like Express.js provide a lightweight and flexible platform for building web servers.

5***. Integration with Third-Party Services:*** Hotels and restaurants often rely on various third-party services such as payment gateways, booking engines, or CRM systems. Node.js's asynchronous nature makes it easy to integrate with external APIs and services, allowing for seamless communication and data exchange between different systems.

Express.JS: Express.js is a web application framework for Node.js, designed to simplify the process of building web applications and APIs. It provides a robust set of features for handling HTTP requests, routing and more, making it a popular choice for building backend services in Node.js applications.

Express.js can be beneficial for a hotel and restaurant website built on Node.js:

1***. Routing:*** Express.js simplifies the process of defining routes for handling HTTP requests. In a hotel and restaurant website, you can define routes for different pages and functionalities such as home page, menu display, reservation booking, contact form submission, etc.

2. ***Database Integration:*** Express.js can be seamlessly integrated with various databases, including SQL databases like MySQL and PostgreSQL, as well as NoSQL databases like MongoDB.

3***. Scalability and Performance:*** Node.js and Express.js are known for their scalability and performance, making them suitable for handling high traffic and concurrent connections. This is particularly important for a hotel and restaurant website that may experience spikes in traffic during peak times or special events.

Database: MySQL

MySQL is a popular open-source relational database management system (RDBMS) that uses structured query language (SQL) to manage and manipulate data. It is widely used for building web applications, including hotel and restaurant websites, due to its reliability, scalability, and robust feature set.

Benefits of using MySQL instead of a NoSQL database like MongoDB for a hotel and restaurant website:

1. ***Data Integrity***: MySQL enforces data integrity through features like foreign key constraints, transactions, and ACID (Atomicity, Consistency, Isolation, Durability) properties. This ensures that data remains consistent and accurate, which is crucial for managing reservations, orders and customer information.

2. ***No Schema Flexibility***: While NoSQL databases offer schema flexibility, relational databases like MySQL provide a predefined schema that can enforce data consistency and relationships between different entities. In a hotel and restaurant website, having a structured schema can be advantageous for organizing data related to menus, room bookings, customer profiles, etc.

3. ***Community Support and Tools:*** MySQL has a large and active community of developers, administrators, and contributors who provide support, documentation, and a wide range of tools and utilities for database management, monitoring, and optimization. This makes it easier to troubleshoot issues, optimize performance, and scale the database as the website grows.

4. ***Integration with Other Technologies:*** MySQL integrates seamlessly with popular programming languages, frameworks, and web servers commonly used in web development, such as PHP, Python, Node.js. This facilitates the development of dynamic and interactive features for the hotel and restaurant website, such as online booking forms, interactive menus, and real-time updates.

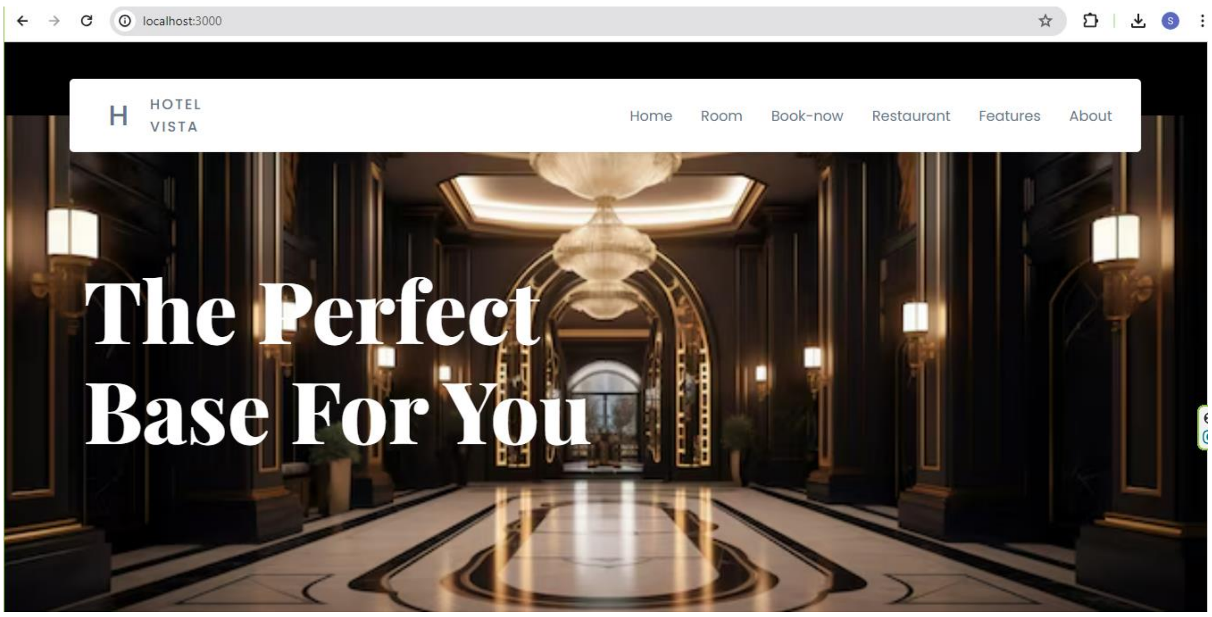
This is the schema of the database created in MySQL:

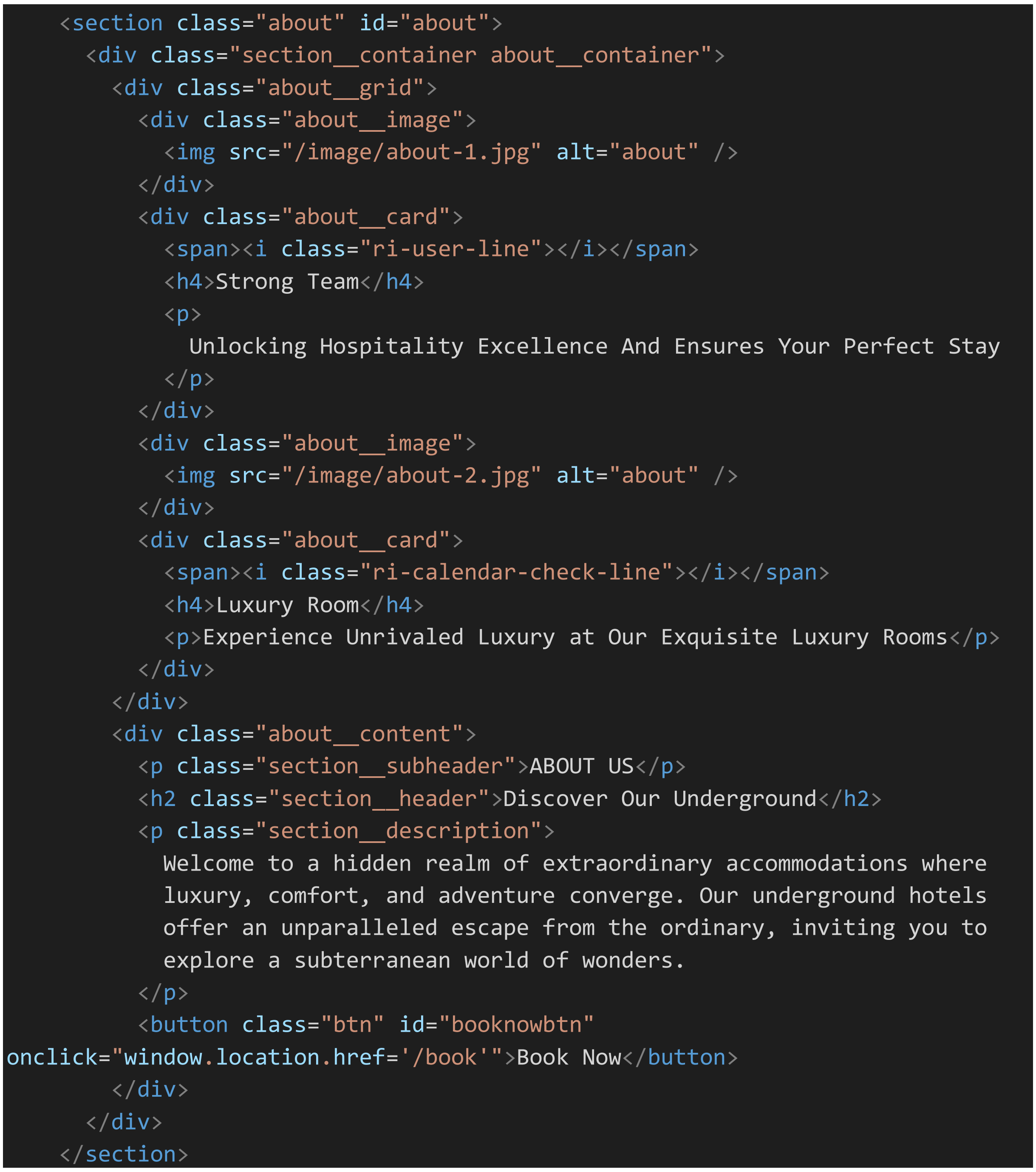
```
use hotel_management;
create table Reservation(
        Room_id    varchar(20),
        Guest_id     varchar(20),
        Payment_id   varchar(20),
        Check_In_Date date,
        Check_Out_Date date,
    Room_type varchar(20)
);

create table Transaction(
        Payment_id varchar(20),
        amount numeric(10),
         Room_id    varchar(20),
        Guest_id     varchar(20)
);

create table if not exists Guest(
        Guest_id varchar(10),
        FirstName varchar(50),
        MiddleName varchar(50),
        LastName varchar(50),
        Aadhar_no numeric(12),
        Email varchar(50),
        PhoneNo numeric(10)
);
```

Home Page:

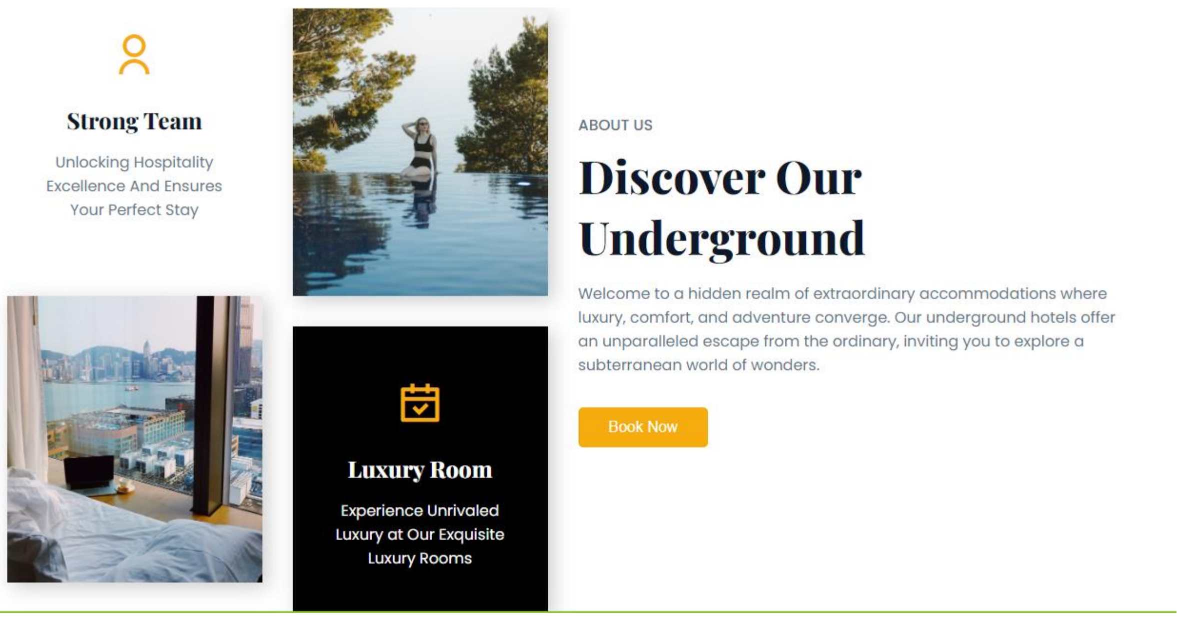
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link
      href="https://cdn.jsdelivr.net/npm/remixicon@3.5.0/fonts/remixicon.css"
      rel="stylesheet"
    />
    <link rel="stylesheet" href="/styles/hotel.css" />
    <title>Web Design Mastery | Vista</title>
  </head>
  <body>
    <script src="/hotel.js"></script>
    <nav>
      <div class="nav__bar">
        <div class="nav__header">
          <div class="logo nav__logo">
            <div>H</div>
            <span>HOTEL<br/>VISTA</span>
          </div>
          <div class="nav__menu__btn" id="menu-btn">
            <i class="ri-menu-line"></i>
          </div>
        </div>
        <ul class="nav__links" id="nav-links">
          <li><a href="#home">Home</a></li>
          <li><a href="#room">Room</a></li>
          <li><a href="/book">Book-now</a></li>
          <li><a href="/restaurant">Restaurant</a></li>
          <li><a href="#feature">Features</a></li>
          <li><a href="#about">About</a></li>

          <!--<li><a href="/signup">Log-in/Sign-up</a></li>-->
        </ul>
      </div>
    </nav>

    <header class="header" id="home">
      <div class="section__container header__container">
        <!--<p class="section__subheader">ABOUT US</p>-->
        <h1>The Perfect<br />Base For You</h1>
      </div>
    </header>

    <section class="booking">
      <div class="section__container booking__container">
```

```
        <form action="/">
          <div class="input__group">
            <label for="arrival">Arrival Date</label>
            <input type="date" placeholder="Your Arrival Date" />
          </div>
          <div class="input__group">
            <label for="departure">Departure Date</label>
            <input type="date" placeholder="Your Departure Date" />
          </div>
          <div class="input__group">
            <label for="roomtype">Room-type</label>
            <select id="roomtype" name="roomtype" aria-placeholder="select
room-type">
              <option value="Deluxe">Deluxe - $3000 per night</option>
              <option value="Family Suite">Family Suite - $5000 per
night</option>
            </select>
          </div>
          <div class="input__group">
            <label for="totalrooms">Total Rooms</label>
            <input type="number" placeholder="Total rooms" />
          </div class="flex items-center justify-center">
          <button class="btn">Check Availability</button>
        </form>
      </div>
    </section>
        <script src="https://unpkg.com/scrollreveal"></script>
    <script src="/main.js"></script>
  </body>
</html>
```





```html
    <section class="about" id="about">
      <div class="section__container about__container">
        <div class="about__grid">
          <div class="about__image">
            <img src="/image/about-1.jpg" alt="about" />
          </div>
          <div class="about__card">
            <span><i class="ri-user-line"></i></span>
            <h4>Strong Team</h4>
            <p>
              Unlocking Hospitality Excellence And Ensures Your Perfect Stay
            </p>
          </div>
          <div class="about__image">
            <img src="/image/about-2.jpg" alt="about" />
          </div>
          <div class="about__card">
            <span><i class="ri-calendar-check-line"></i></span>
            <h4>Luxury Room</h4>
            <p>Experience Unrivaled Luxury at Our Exquisite Luxury Rooms</p>
          </div>
        </div>
        <div class="about__content">
          <p class="section__subheader">ABOUT US</p>
          <h2 class="section__header">Discover Our Underground</h2>
          <p class="section__description">
            Welcome to a hidden realm of extraordinary accommodations where
            luxury, comfort, and adventure converge. Our underground hotels
            offer an unparalleled escape from the ordinary, inviting you to
            explore a subterranean world of wonders.
          </p>
          <button class="btn" id="booknowbtn"
onclick="window.location.href='/book'">Book Now</button>
        </div>
      </div>
    </section>
```



Strong Team

Unlocking Hospitality Excellence And Ensures Your Perfect Stay

Luxury Room

Experience Unrivaled Luxury at Our Exquisite Luxury Rooms

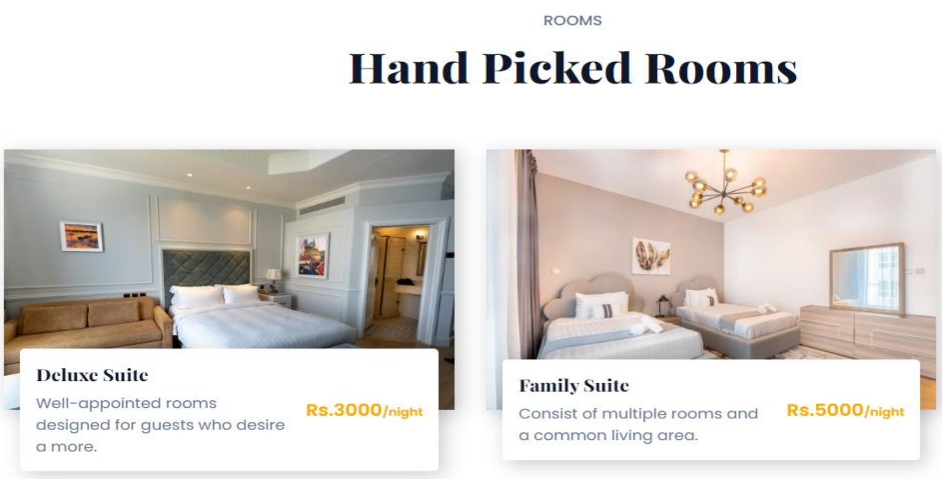
ABOUT US

Discover Our Underground

Welcome to a hidden realm of extraordinary accommodations where luxury, comfort, and adventure converge. Our underground hotels offer an unparalleled escape from the ordinary, inviting you to explore a subterranean world of wonders.

Book Now

```
<section class="room__container" id="room">
      <p class="section__subheader">ROOMS</p>
      <h2 class="section__header">Hand Picked Rooms</h2>
      <div class="room__grid">
        <div class="room__card">
          <img src="/image/room-1.jpg" alt="room"/>
          <div class="room__card__details">
            <div>
              <h4>Deluxe Suite</h4>
              <p>Well-appointed rooms designed for guests who desire a
more.</p>
            </div>
            <h3>Rs.3000<span>/night</span></h3>
          </div>
        </div>
        <div class="room__card">
          <img src="/image/room-2.jpg" alt="room" />
          <div class="room__card__details">
            <div>
              <h4>Family Suite</h4>
              <p>Consist of multiple rooms and a common living area.</p>
            </div>
            <h3>Rs.5000<span>/night</span></h3>
          </div>
        </div>
    </section>
```



ROOMS

Hand Picked Rooms

Deluxe Suite

Well-appointed rooms designed for guests who desire a more.

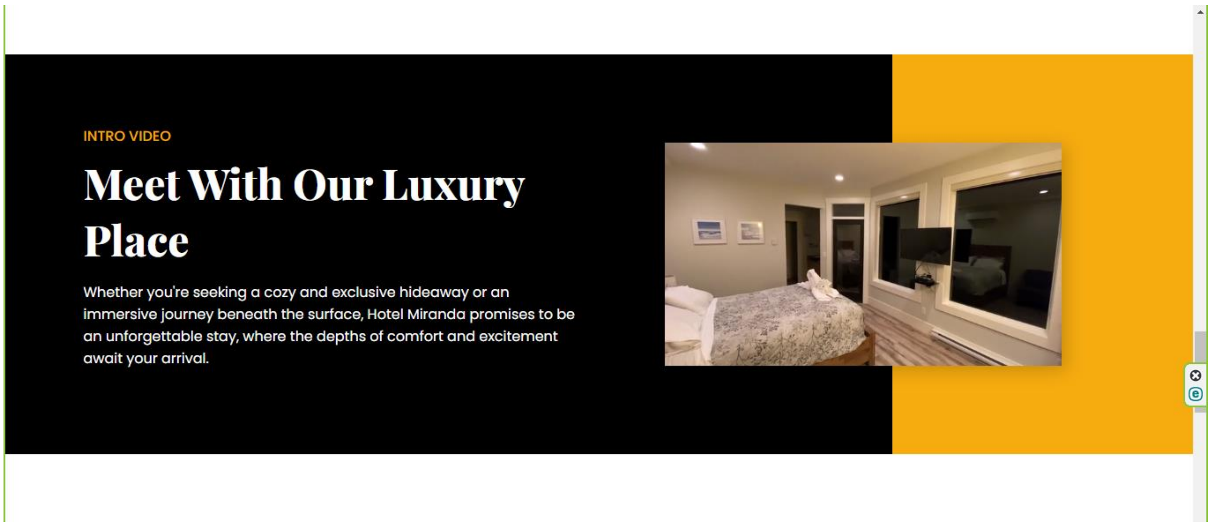
Rs.3000/night

Family Suite

Consist of multiple rooms and a common living area.

Rs.5000/night

```
<section class="intro">
      <div class="section__container intro__container">
        <div class="intro__cotent">
          <p class="section__subheader">INTRO VIDEO</p>
          <h2 class="section__header">Meet With Our Luxury Place</h2>
          <p class="section__description">
            Whether you're seeking a cozy and exclusive hideaway or an
immersive
            journey beneath the surface, Hotel Miranda promises to be an
            unforgettable stay, where the depths of comfort and excitement
await
            your arrival.
          </p>
        </div>
        <div class="intro__video">
          <video src="/image/luxury.mp4" autoplay muted loop></video>
        </div>
      </div>
    </section>
```

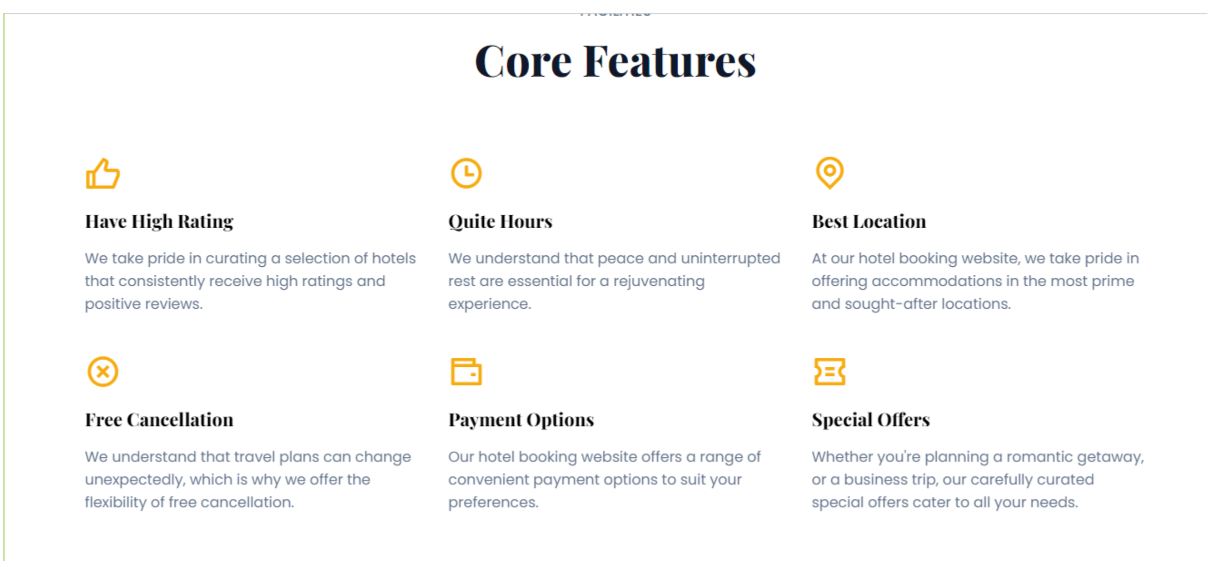


INTRO VIDEO

Meet With Our Luxury Place

Whether you're seeking a cozy and exclusive hideaway or an immersive journey beneath the surface, Hotel Miranda promises to be an unforgettable stay, where the depths of comfort and excitement await your arrival.

```html
    <section class="section__container feature__container" id="feature">
      <p class="section__subheader">FACILITIES</p>
      <h2 class="section__header">Core Features</h2>
      <div class="feature__grid">
        <div class="feature__card">
          <span><i class="ri-thumb-up-line"></i></span>
          <h4>Have High Rating</h4>
          <p>
            We take pride in curating a selection of hotels that consistently
            receive high ratings and positive reviews.
          </p>
        </div>
        <div class="feature__card">
          <span><i class="ri-time-line"></i></span>
          <h4>Quite Hours</h4>
          <p>
            We understand that peace and uninterrupted rest are essential for
a

            rejuvenating experience.
          </p>
        </div>
        <div class="feature__card">
          <span><i class="ri-map-pin-line"></i></span>
          <h4>Best Location</h4>
          <p>
            At our hotel booking website, we take pride in offering
            accommodations in the most prime and sought-after locations.
          </p>
        </div>
        <div class="feature__card">
          <span><i class="ri-close-circle-line"></i></span>
          <h4>Free Cancellation</h4>
          <p>
            We understand that travel plans can change unexpectedly, which is
            why we offer the flexibility of free cancellation.
          </p>
        </div>
        <div class="feature__card">
          <span><i class="ri-wallet-line"></i></span>
          <h4>Payment Options</h4>
          <p>
            Our hotel booking website offers a range of convenient payment
            options to suit your preferences.
          </p>
        </div>
        <div class="feature__card">
          <span><i class="ri-coupon-line"></i></span>
          <h4>Special Offers</h4>
          <p>
            Whether you're planning a romantic getaway, or a business trip,
our
            carefully curated special offers cater to all your needs.
          </p>
        </div>
      </div>
    </section>
```



Core Features

Have High Rating

We take pride in curating a selection of hotels that consistently receive high ratings and positive reviews.

Quite Hours

We understand that peace and uninterrupted rest are essential for a rejuvenating experience.

Best Location

At our hotel booking website, we take pride in offering accommodations in the most prime and sought-after locations.

Free Cancellation

We understand that travel plans can change unexpectedly, which is why we offer the flexibility of free cancellation.

Payment Options

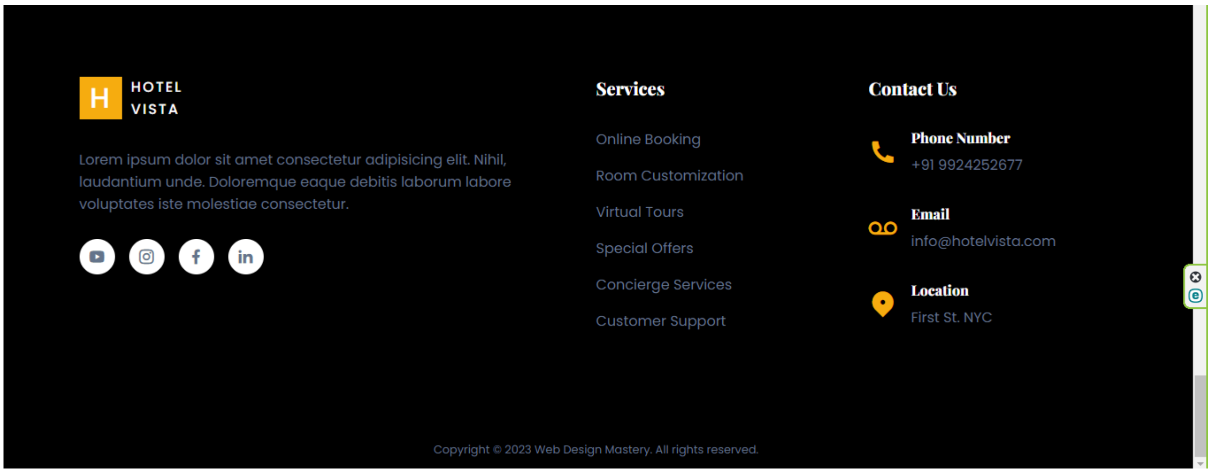
Our hotel booking website offers a range of convenient payment options to suit your preferences.

Special Offers

Whether you're planning a romantic getaway, or a business trip, our carefully curated special offers cater to all your needs.

```
<footer class="footer">
      <div class="section__container footer__container">
        <div class="footer__col">
          <div class="logo footer__logo">
            <div>H</div>
            <span>HOTEL<br />VISTA</span>
          </div>
          <p class="section__description">
            Lorem ipsum dolor sit amet consectetur adipisicing elit. Nihil,
            laudantium unde. Doloremque eaque debitis laborum labore
voluptates
            iste molestiae consectetur.
          </p>
          <ul class="footer__socials">
            <li>
              <a href="#"><i class="ri-youtube-fill"></i></a>
            </li>
            <li>
              <a href="#"><i class="ri-instagram-line"></i></a>
            </li>
            <li>
              <a href="#"><i class="ri-facebook-fill"></i></a>
            </li>
            <li>
```

```html
          <a href="#"><i class="ri-linkedin-fill"></i></a>
        </li>
      </ul>
    </div>
    <div class="footer__col">
      <h4>Services</h4>
      <div class="footer__links">
        <li><a href="#">Online Booking</a></li>
        <li><a href="#">Room Customization</a></li>
        <li><a href="#">Virtual Tours</a></li>
        <li><a href="#">Special Offers</a></li>
        <li><a href="#">Concierge Services</a></li>
        <li><a href="#">Customer Support</a></li>
      </div>
    </div>
    <div class="footer__col">
      <h4>Contact Us</h4>
      <div class="footer__links">
        <li>
          <span><i class="ri-phone-fill"></i></span>
          <div>
            <h5>Phone Number</h5>
            <p>+91 9924252677</p>
          </div>
        </li>
        <li>
          <span><i class="ri-record-mail-line"></i></span>
          <div>
            <h5>Email</h5>
            <p>info@hotelvista.com</p>
          </div>
        </li>
        <li>
          <span><i class="ri-map-pin-2-fill"></i></span>
          <div>
            <h5>Location</h5>
            <p>First St. NYC</p>
          </div>
        </li>
      </div>
    </div>
  </div>
  <div class="footer__bar">
    Copyright © 2023 Web Design Mastery. All rights reserved.
  </div>
</footer>
```



Lorem ipsum dolor sit amet consectetur adipisicing elit. Nihil, laudantium unde. Doloremque eaque debitis laborum labore voluptates iste molestiae consectetur.

Services

Online Booking
Room Customization
Virtual Tours
Special Offers
Concierge Services
Customer Support

Contact Us

Phone Number
+91 9924252677

Email
info@hotelvista.com

Location
First St. NYC

Copyright © 2023 Web Design Mastery. All rights reserved.

Book-now page:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Booking Form</title>
<script src="https://cdn.tailwindcss.com"></script>
<script src="/hotel.js"></script>
<link rel="stylesheet" href="/styles/book.css">
</head>
<body class="bg-cover bg-center flex justify-center items-center h-screen">
<form action="/submit_booking" method="post" onsubmit="return validateForm()"
class="bg-white shadow-md rounded px-8 pt-6 pb-8 mb-3 max-w-md w-full">
<!-- Customer Details -->
<div class="mb-3 flex justify-between">
<div class="w-1/3 mr-2">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="FirstName">First Name:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="name"
type="text" placeholder="First Name" name="FirstName" required=""
autocomplete="off">
</div>
<div class="w-1/3 mr-2">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="MiddleName">Middle Name:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline"
id="middleName" type="text" placeholder="Middle Name" name="MiddleName"
required="" autocomplete="off">
</div>
```

```html
<div class="w-1/3">
<label class="block text-gray-700 text-sm font-bold mb-2" for="LastName">Last
Name:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="lastName"
type="text" placeholder="Last Name" name="LastName" required=""
autocomplete="off">
</div>
</div>
<!-- Email and Phone Number -->
<div class="mb-3 flex justify-between">
<div class="w-1/2 mr-2">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="Email">Email:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="email"
type="email" placeholder="Email" name="Email" required="" autocomplete="off">
</div>
<div class="w-1/2">
<label class="block text-gray-700 text-sm font-bold mb-2" for="PhoneNo">Phone
Number:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="phone"
type="tel" placeholder="Phone Number" name="PhoneNo" required=""
autocomplete="off">
</div>
</div>
<!-- Room Type -->
<div class="mb-3 flex items-center">
<label class="block text-gray-700 text-sm font-bold mr-2"
for="Room_type">Type:</label>
<select onchange="calculateTotal()" class="shadow appearance-none border
rounded py-2 px-1 text-sm text-gray-700 leading-tight focus:outline-none
focus:shadow-outline" id="Room_type" name="Room_type" required="">
<option value="" disabled="" selected="">Select</option>
<option value="3000">Deluxe Room - Rs3000/night</option>
<option value="5000">Family Suite - Rs5000/night </option>
</select>
<label class="block text-gray-700 text-sm font-bold ml-2"
for="num_rooms">Rooms:</label>
<input onchange="calculateTotal()" class="shadow appearance-none border
rounded py-2 px-1 text-sm text-gray-700 leading-tight focus:outline-none
focus:shadow-outline w-10 text-center ml-1" type="number" id="num_rooms"
name="num_rooms" min="1" value="1">
</div>
<!-- Check-in and Check-out Dates -->
<div class="mb-3 flex justify-between">
  <div class="w-1/2 pr-2">
```

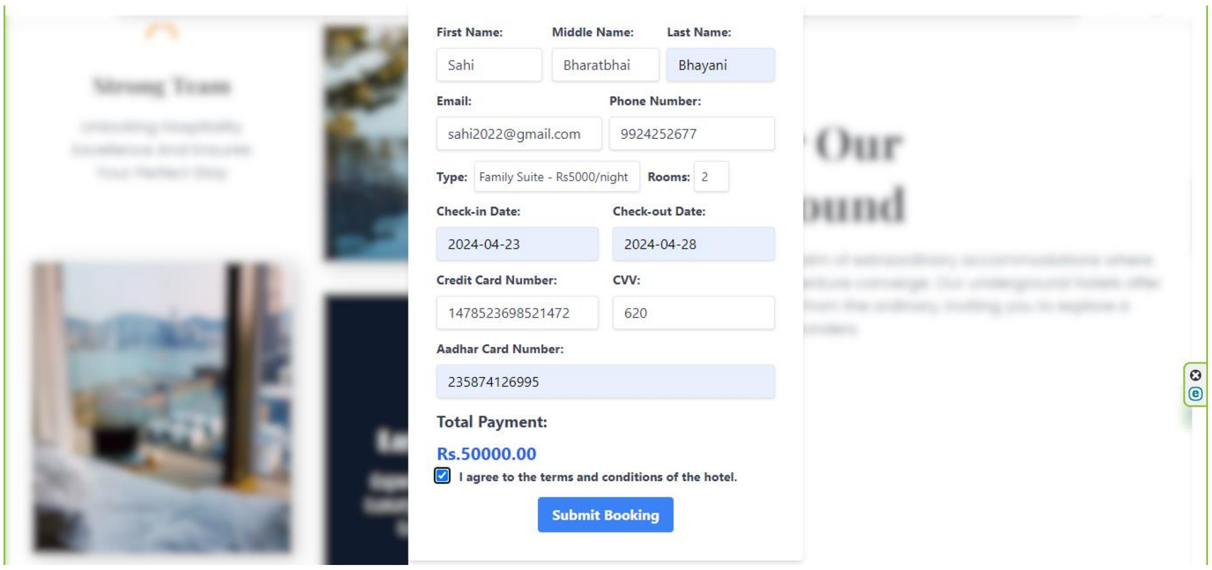
```
      <label class="block text-gray-700 text-sm font-bold mb-2"
for="Check_In_Date">Check-in Date:</label>
      <input onchange="calculateTotal()" class="shadow appearance-none border
rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none
focus:shadow-outline" id="Check_In_Date" type="text" name="Check_In_Date"
pattern="\d{4}-\d{2}-\d{2}" placeholder="YYYY-MM-DD" required="">
  </div>
  <div class="w-1/2 pl-2">
      <label class="block text-gray-700 text-sm font-bold mb-2"
for="Check_Out_Date">Check-out Date:</label>
      <input  onchange="calculateTotal()" class="shadow appearance-none border
rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none
focus:shadow-outline" id="Check_Out_Date" type="text" name="Check_Out_Date"
pattern="\d{4}-\d{2}-\d{2}" placeholder="YYYY-MM-DD" required="">
  </div>
</div>
<!-- Credit Card Number and CVV -->
<div class="mb-3 flex justify-between">
<div class="w-1/2 pr-2">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="credit_card">Credit Card Number:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline"
id="credit_card" type="text" name="credit_card" required=""
autocomplete="off">
</div>
<div class="w-1/2 pl-2">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="cvv">CVV:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="cvv"
type="text" name="cvv" required="" autocomplete="off">
</div>
</div>
<!-- Aadhar Card Number -->
<div class="mb-3">
<label class="block text-gray-700 text-sm font-bold mb-2"
for="Aadhar_no">Aadhar Card Number:</label>
<input class="shadow appearance-none border rounded w-full py-2 px-3 text-
gray-700 leading-tight focus:outline-none focus:shadow-outline" id="Aadhar_no"
type="text" name="Aadhar_no" required="" pattern="[1-9]{12}" title="Aadhar
card number must be 12 digits long">
</div>
<!-- Total Amount -->
<h3 class="text-lg font-bold text-gray-700 mb-2">Total Payment:</h3>
<p id="amount" class="text-xl font-bold text-blue-600">Rs.0</p>
<!-- Terms and Conditions -->
<div class="mb-3">
```

```
<input type="checkbox" class="mr-2 leading-tight" id="terms" name="terms"
required="">
<label class="text-gray-700 text-sm font-bold" for="terms">I agree to the
terms and conditions of the hotel.</label>
</div>
<!-- Submit Button -->
<div class="flex items-center justify-center">
<button class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4
rounded focus:outline-none focus:shadow-outline" type="submit">Submit
Booking</button>
</div>
</form>
</body>
</html>
```

Function to calculate the total payment:

```
function calculateTotal() {
    console.log("Calculating total...");
    var roomTypeSelect = document.getElementById("Room_type");
    var numRoomsInput = document.getElementById("num_rooms");
    var checkInDateInput = document.getElementById("Check_In_Date");
    var checkOutDateInput = document.getElementById("Check_Out_Date");
    var totalPaymentParagraph = document.getElementById("amount");

    var selectedOption = roomTypeSelect.options[roomTypeSelect.selectedIndex];
    var roomPrice = parseFloat(selectedOption.value); // Parse room price as a
float
    var numRooms = parseInt(numRoomsInput.value); // Parse number of rooms as
an integer

    var checkInDate = new Date(checkInDateInput.value);
    var checkOutDate = new Date(checkOutDateInput.value);

    // Calculate the number of nights the guest will stay
    var oneDay = 24 * 60 * 60 * 1000; // hours*minutes*seconds*milliseconds
    var numberOfNights = Math.round(Math.abs((checkOutDate - checkInDate) /
oneDay));

    // Check if the dates are valid and the number of nights is positive
    if (isNaN(numberOfNights) || numberOfNights <= 0) {
        totalPaymentParagraph.textContent = "Rs.0";
    } else {
        // Calculate the total price based on room price, number of rooms, and
number of nights
        var totalPrice = roomPrice * numRooms * numberOfNights;
        // Update the total payment displayed on the form with the currency
symbol "Rs."
        totalPaymentParagraph.textContent = "Rs." + totalPrice.toFixed(2);
    }
}
```

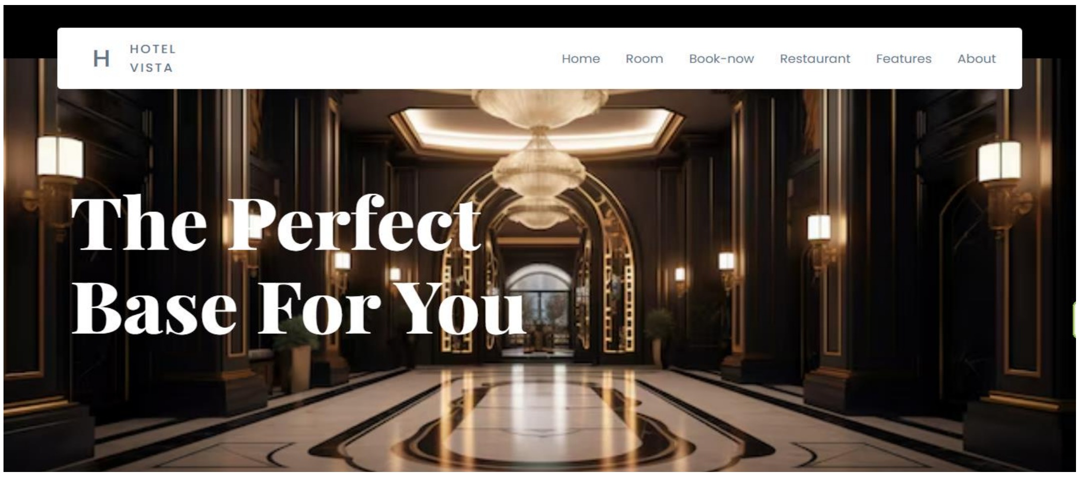
Data Inserted in Form:



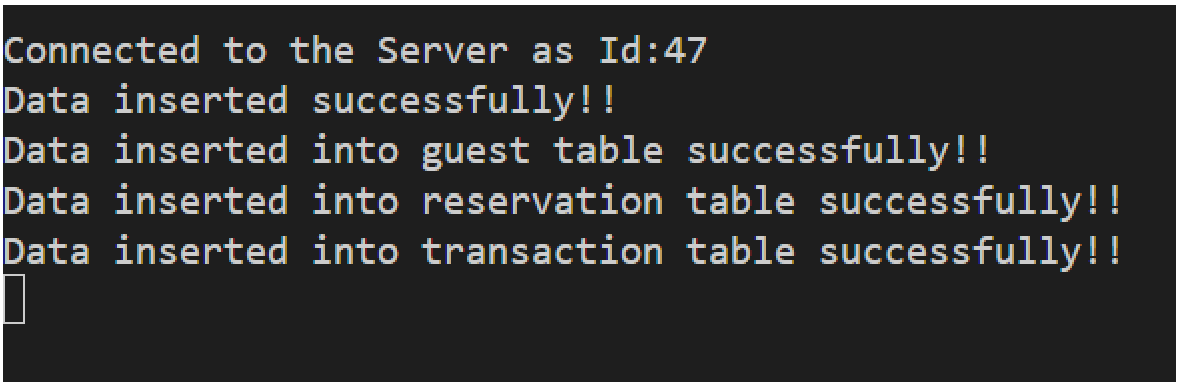
First Name: Sahi
Middle Name: Bharatbhai
Last Name: Bhayani
Email: sahi2022@gmail.com
Phone Number: 9924252677
Type: Family Suite - Rs5000/night
Rooms: 2
Check-in Date: 2024-04-23
Check-out Date: 2024-04-28
Credit Card Number: 1478523698521472
CVV: 620
Aadhar Card Number: 235874126995
Total Payment:
Rs.50000.00
✓ I agree to the terms and conditions of the hotel.
Submit Booking

On-Submit action

1.Renders back to home page



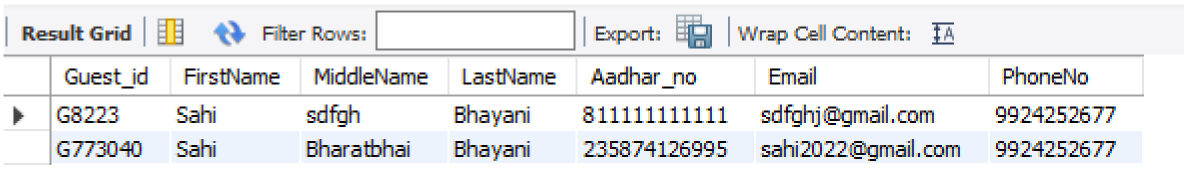
2. Shows data inserted successfully.



```
Connected to the Server as Id:47
Data inserted successfully!!
Data inserted into guest table successfully!!
Data inserted into reservation table successfully!!
Data inserted into transaction table successfully!!
```

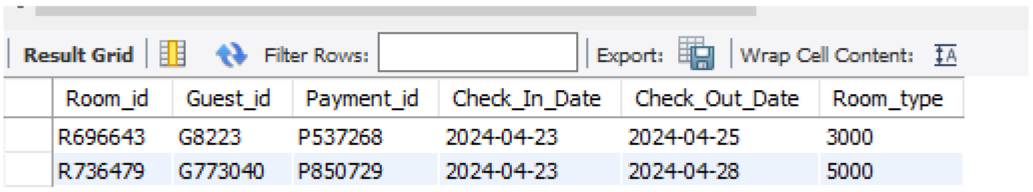
3. Inserted data in the Tables.

For guest table:



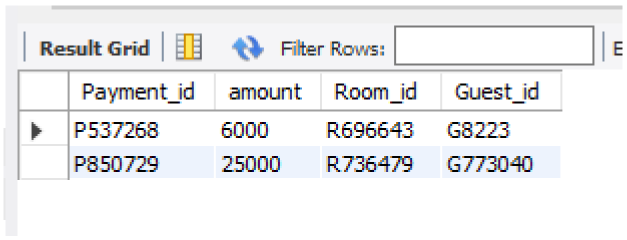
Guest_id	FirstName	MiddleName	LastName	Aadhar_no	Email	PhoneNo
G8223	Sahi	sdfgh	Bhayani	811111111111	sdfghj@gmail.com	9924252677
G773040	Sahi	Bharatbhai	Bhayani	235874126995	sahi2022@gmail.com	9924252677

For Reservation Table:



Room_id	Guest_id	Payment_id	Check_In_Date	Check_Out_Date	Room_type
R696643	G8223	P537268	2024-04-23	2024-04-25	3000
R736479	G773040	P850729	2024-04-23	2024-04-28	5000

For Transaction Table:



Payment_id	amount	Room_id	Guest_id
P537268	6000	R696643	G8223
P850729	25000	R736479	G773040

Index.js file for data connectivity, insertion and get request for rendering

- **Database connectivity**

  1. Importing Dependencies:

  This section imports necessary modules and libraries for the application. It includes `mysql2` for MySQL database connectivity, `body-parser` for parsing incoming request bodies, `express` for creating the web server, and `path` and `url` modules for working with file and directory paths.

```
import mysql from "mysql2";
import bodyParser from "body-parser"
import express, { query, response } from "express";
import path from "path";
import { fileURLToPath } from "url";
import { dirname } from "path";
```

2. Defining File and Directory Paths:

These lines define the current file and directory paths using the `fileURLToPath` and `dirname` functions from the `url` and `path` modules, respectively.

```
const __filename = fileURLToPath(import.meta.url);
const __dirname = dirname(__filename);
```

3. Setting up Express Application:

Here, an Express application is created, and the port number `3000` is specified for the server to listen on.

```
const app = express();
const port = 3000;
```

4. Middleware Setup:

Middleware functions are added to the Express application using app.use().

express.static("public"): This middleware serves static files from the `public` directory.

bodyParser.urlencoded({ extended: true }): This middleware parses incoming request bodies with URL-encoded data.

app.set("view engine", "ejs") and `app.set("views", path.join(__dirname, "views"))`: These lines configure the application to use the EJS templating engine and set the views directory to the `views` folder in the current directory.

```
app.use(express.static("public"));//static files
app.use(bodyParser.urlencoded({ extended: true }));
app.set("view engine", "ejs");
app.set("views", path.join(__dirname, "views"));
```

5. MySQL Connection Configuration:

This section sets up the configuration for connecting to the MySQL database. It specifies the host, user, password, and database name.

```
const connection = mysql.createConnection({
    host:"localhost",
    user: "root",
    password:"s@#!B1005",
    database:"hotel_management"
});

connection.connect((err)=>{
    if(err){
        console.error("Error Connecting to Server:" + err.stack);
    }
    else{
        console.log('Connected to the Server as Id:' + connection.threadId);
    }
});
```

```
app.listen(port,()=>{
    try{
        console.log(`Listening to port ${port}`);
    }
    catch(error){
        console.log(error);
    }

})
```

- For every .ejs file a get request is made that helps to render to that website. Below is get request for all files.

```
•  app.get("/", (req,res)=>{
•      try{
•          res.render("hotel.ejs");
•      }
•      catch(error){
•          res.send(error);
•      }
•  });
•
•  app.get("/hotel",(req,res)=>{
•      try{
        res.render("hotel.ejs");
    }
    catch(error){
        res.send(error);
    }
});

app.get("/book", (req,res)=>{
    try{
        res.render("book.ejs");
    }
    catch(error){
        res.send(error);
    }
});

app.get("/restaurant", (req,res)=>{
    try{
        res.render("restaurant.ejs");
    }
    catch(error){
        res.send(error);
    }
});

app.get("/menu1", (req,res)=>{
    try{
        res.render("menu1.ejs");
    }
    catch(error){
        res.send(error);
    }
});

app.get("/menu", (req,res)=>{
    try{
        res.render("menu.ejs");
    }
    catch(error){
        res.send(error);
    }
});

app.get("/signup", (req,res)=>{
    try{
        res.render("signup.ejs");
    }
    catch(error){
```

```
•           res.send(error);
•       }
•   });
```

App.post request created for the /submit booking action of the form:

```
app.post("/submit_booking", (req, res) => {
    try {
        // Extract form data from request body
        const { FirstName, MiddleName, LastName, Aadhar_no, Email, PhoneNo,
Check_In_Date, Check_Out_Date, Room_type } = req.body;

        // Calculate the number of days the guest will stay
        const checkInDate = new Date(Check_In_Date);
        const checkOutDate = new Date(Check_Out_Date);
        const numberOfDays = Math.ceil((checkOutDate - checkInDate) / (1000 *
60 * 60 * 24));

        // Determine the room price based on the room type
        let roomPrice = 0;
        if (Room_type === "3000") {
            roomPrice = 3000;
        } else if (Room_type === "5000") {
            roomPrice = 5000;
        }

        // Calculate the total amount
        const amount = roomPrice * numberOfDays;

        // Generate unique IDs for guest, room, and payment
        const Guest_id = generateGuestID();
        const Room_id = generateRoomID();
        const Payment_id = generatePaymentID();

        // Insert data into 'guest' table
        const guestInsertionResult = inserted(Guest_id, FirstName, MiddleName,
LastName, Email, PhoneNo, Aadhar_no);

        // Insert data into 'reservation' table
        const roomInsertionResult = inserted1(Room_id, Guest_id, Payment_id,
Check_In_Date, Check_Out_Date, Room_type);

        // Insert data into 'transaction' table
        const paymentInsertionResult = inserted2(Payment_id, amount, Room_id,
Guest_id);

        // Check if all insertions were successful
        if (guestInsertionResult && roomInsertionResult &&
paymentInsertionResult) {
            console.log("Data inserted successfully!!");
            res.redirect("/");
        } else {
            console.log("Data not inserted");
            res.redirect("/book");
        }
    } catch (error) {
        console.error("Error handling form submission:", error);
        res.status(500).send("Internal Server Error");
    }
});
```

- This request first extracts the data from the request body using body-parser.
- Then it inserts the data into respective tables and if the data gets inserted successfully in all the 3 tables then it renders back to home page.

  Functions created to generate unique GuestID, RoomID and PaymentID.

```
function generateGuestID() {
    // Generate a random number between 1000 and 9999
    const randomNum = Math.floor(Math.random() * 1000000);

    // Combine timestamp and random number to create a unique ID
    const guestID = `G${randomNum}`;

    return guestID;
}

function generateRoomID() {
    // Generate a random number between 1000 and 9999
    const randomNum = Math.floor(Math.random() * 1000000);

    // Combine timestamp and random number to create a unique ID
    const roomID = `R${randomNum}`;

    return roomID;
}

function generatePaymentID() {
    // Generate a random number between 1000 and 9999
    const randomNum = Math.floor(Math.random() * 1000000);

    // Combine timestamp and random number to create a unique ID
    const paymentID = `P${randomNum}`;

    return paymentID;
}
```

Inserted Function created to Insert data in guest table:

```
// Function to insert guest data into the 'guest' table
const inserted = (Guest_id, FirstName, MiddleName, LastName, Email, PhoneNo,
Aadhar_no) => {
    return new Promise((resolve, reject) => {
        connection.query("INSERT INTO guest (Guest_id, FirstName, MiddleName,
LastName, Email, PhoneNo, Aadhar_no) VALUES (?, ?, ?, ?, ?, ?, ?)",
            [Guest_id, FirstName, MiddleName, LastName, Email, PhoneNo,
Aadhar_no],
            (err, results) => {
                if (err) {
                    console.error("Error inserting data into guest table:",
err);
                    reject(err);
                } else {
                    console.log("Data inserted into guest table
successfully!!");
                    resolve(true);
                }
            });
    });
};
```

Inserted1 Function created to Insert data in reservation table:

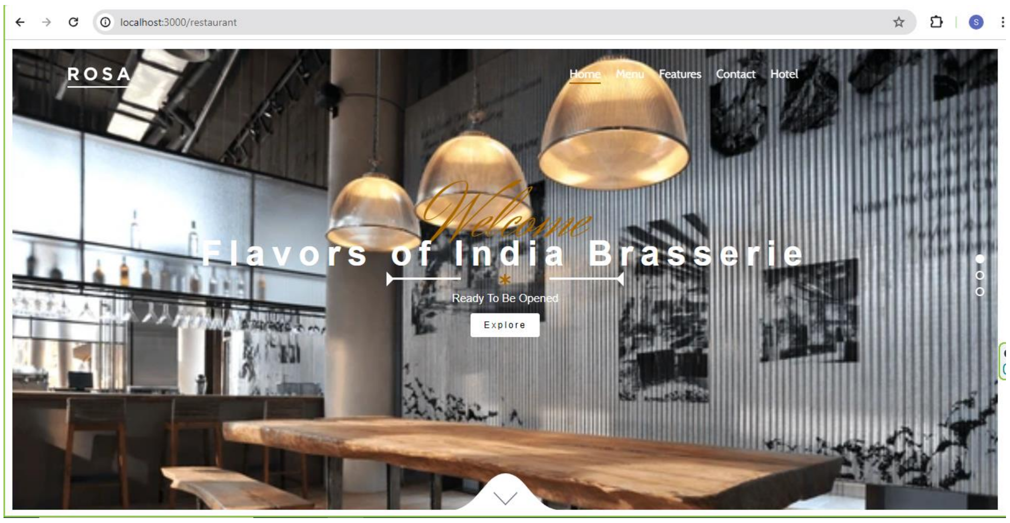
```
// Function to insert reservation data into the 'reservation' table
const inserted1 = (Room_id, Guest_id, Payment_id, Check_in_date,
Check_out_date, room_type) => {
    return new Promise((resolve, reject) => {
        connection.query("INSERT INTO reservation (Room_id, Guest_id,
Payment_id, Check_in_date, Check_out_date, room_type) VALUES (?, ?, ?, ?, ?,
?)",
            [Room_id, Guest_id, Payment_id, Check_in_date, Check_out_date,
room_type],
            (err, results) => {
                if (err) {
                    console.error("Error inserting data into reservation
table:", err);
                    reject(err);
                } else {
                    console.log("Data inserted into reservation table 
successfully!!");
                    resolve(true);
                }
            });
    });
};
```

Inserted2 Function created to Insert data in transaction table:

```
// Function to insert payment data into the 'transaction' table
const inserted2 = (Payment_id, amount, Room_id, Guest_id) => {
    return new Promise((resolve, reject) => {
        connection.query("INSERT INTO transaction (Payment_id, amount, 
Room_id, Guest_id) VALUES (?, ?, ?, ?)",
            [Payment_id, amount, Room_id, Guest_id],
            (err, results) => {
                if (err) {
                    console.error("Error inserting data into transaction 
table:", err);
                    reject(err);
                } else {
                    console.log("Data inserted into transaction table 
successfully!!");
                    resolve(true);
                }
            });
    });
};
```

Now rendering to the restaurant.ejs from hotel page:

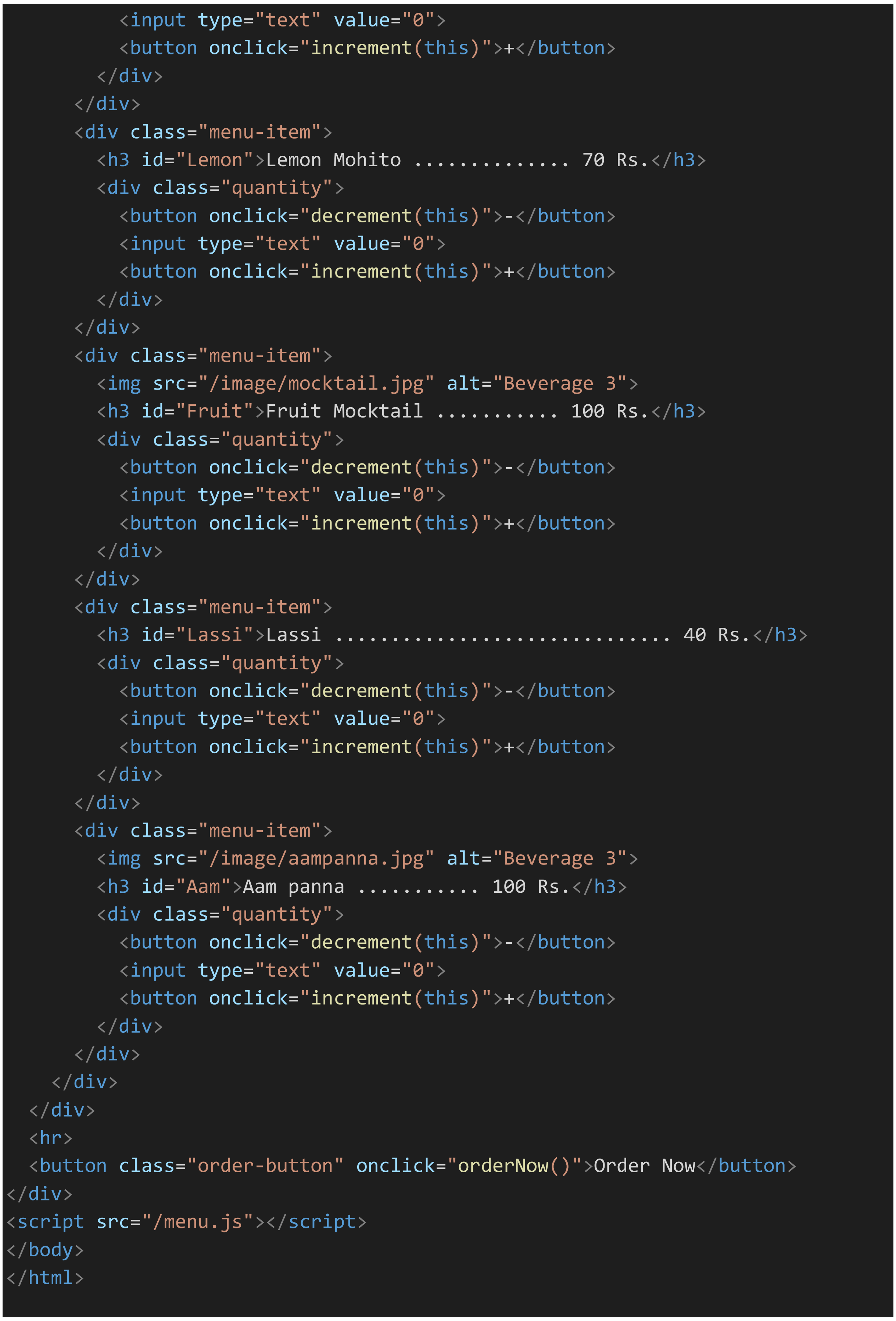
Main page of restaurant:



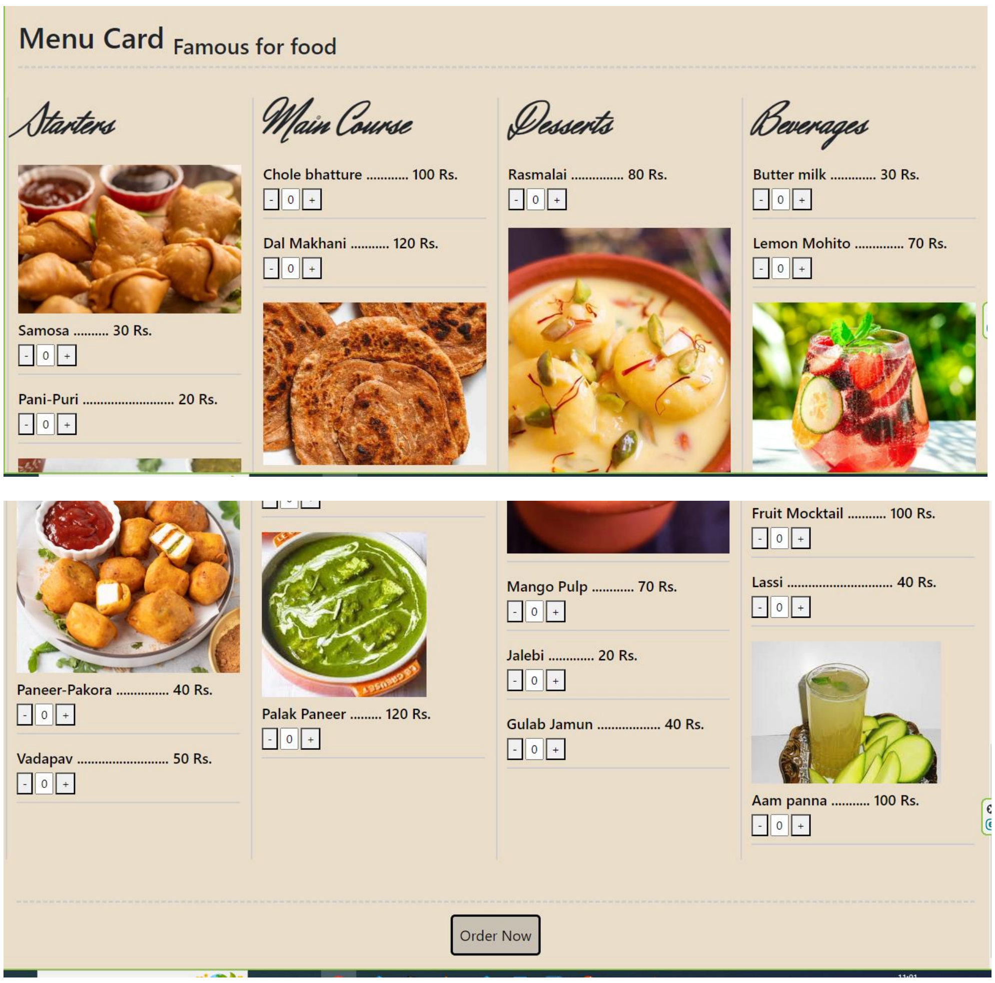
Menu.ejs file

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Restaurant Menu</title>
  <!-- Bootstrap CSS -->
  <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css
" rel="stylesheet">
  <!-- Custom CSS -->
  <link
href="https://fonts.googleapis.com/css?family=Cabin|Herr+Von+Muellerhoff|Sourc
e+Sans+Pro" rel="stylesheet">
  <link rel="stylesheet" href="/styles/menu.css">
</head>
<body>
  <div class="container-fluid">
  <h1>Menu Card <sub>Famous for food</sub></h1>
  <hr>
  <br>
  <div class="row">
    <div class="col-md-3 category" style>
      <h2 style="font-family: 'Herr Von Muellerhoff', cursive; font-weight:
bolder;">Starters</h2>
      <div class="menu-item">
        <img src="/image/Punjabi samosa.jpg" alt="Starter 1">
        <h3 id="Samosa">Samosa .......... 30 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Pani-Puri">Pani-Puri .......................... 20 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <img src="/image/Paneer pakora.jpg" alt="Starter 3">
        <h3 id="Paneer-Pakora">Paneer-Pakora ............... 40 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Vadapav">Vadapav .......................... 50 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
    </div>
    <div class="col-md-3 category">
      <h2 style="font-family: 'Herr Von Muellerhoff', cursive; font-weight:
bolder;">Main Course</h2>
      <div class="menu-item">
        <h3 id="Chole">Chole bhatture ............ 100 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Dal">Dal Makhani ........... 120 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <img src="/image/laccha paratha.jpg" alt="Main Course 3">
        <h3 id="Lachcha">Lachcha Paratha .......... 110 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <img src="/image/palak.jpg" alt="Main Course 4">
        <h3 id="Palak">Palak Paneer ......... 120 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
```

```html
        </div>
      </div>
    </div>
    <div class="col-md-3 category">
      <h2 style="font-family: 'Herr Von Muellerhoff', cursive; font-weight:
bolder;">Desserts</h2>
      <div class="menu-item">
        <h3 id="Rasmalai">Rasmalai ............... 80 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
        <br>
        <img src="/image/Rasmalai-recipe-2.jpg" alt="Dessert 1">
      </div>
      <div class="menu-item">
        <h3 id="Mango">Mango Pulp ............ 70 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Jalebi">Jalebi .............. 20 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Gulab">Gulab Jamun .................. 40 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
    </div>
    <div class="col-md-3 category">
      <h2 style="font-family: 'Herr Von Muellerhoff', cursive; font-weight:
bolder;">Beverages</h2>
      <div class="menu-item">
        <h3 id="Butter">Butter milk ............. 30 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
```



```html
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Lemon">Lemon Mohito .............. 70 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <img src="/image/mocktail.jpg" alt="Beverage 3">
        <h3 id="Fruit">Fruit Mocktail ........... 100 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <h3 id="Lassi">Lassi .............................. 40 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
      <div class="menu-item">
        <img src="/image/aampanna.jpg" alt="Beverage 3">
        <h3 id="Aam">Aam panna ........... 100 Rs.</h3>
        <div class="quantity">
          <button onclick="decrement(this)">-</button>
          <input type="text" value="0">
          <button onclick="increment(this)">+</button>
        </div>
      </div>
    </div>
  </div>
  <hr>
  <button class="order-button" onclick="orderNow()">Order Now</button>
</div>
<script src="/menu.js"></script>
</body>
</html>
```



Menu Card Famous for food

Starters

Samosa .... 30 Rs.

Pani-Puri .... 20 Rs.

Paneer-Pakora .... 40 Rs.

Vadapav .... 50 Rs.

Main Course

Chole bhatture .... 100 Rs.

Dal Makhani .... 120 Rs.

Palak Paneer .... 120 Rs.

Desserts

Rasmalai .... 80 Rs.

Mango Pulp .... 70 Rs.

Jalebi .... 20 Rs.

Gulab Jamun .... 40 Rs.

Beverages

Butter milk .... 30 Rs.

Lemon Mohito .... 70 Rs.

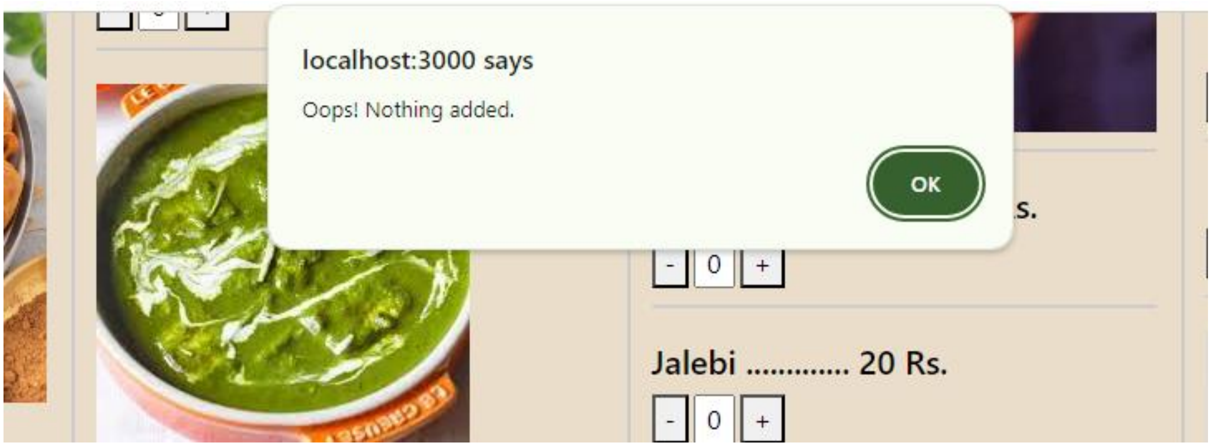
Fruit Mocktail .... 100 Rs.

Lassi .... 40 Rs.

Aam panna .... 100 Rs.

Order now button action:

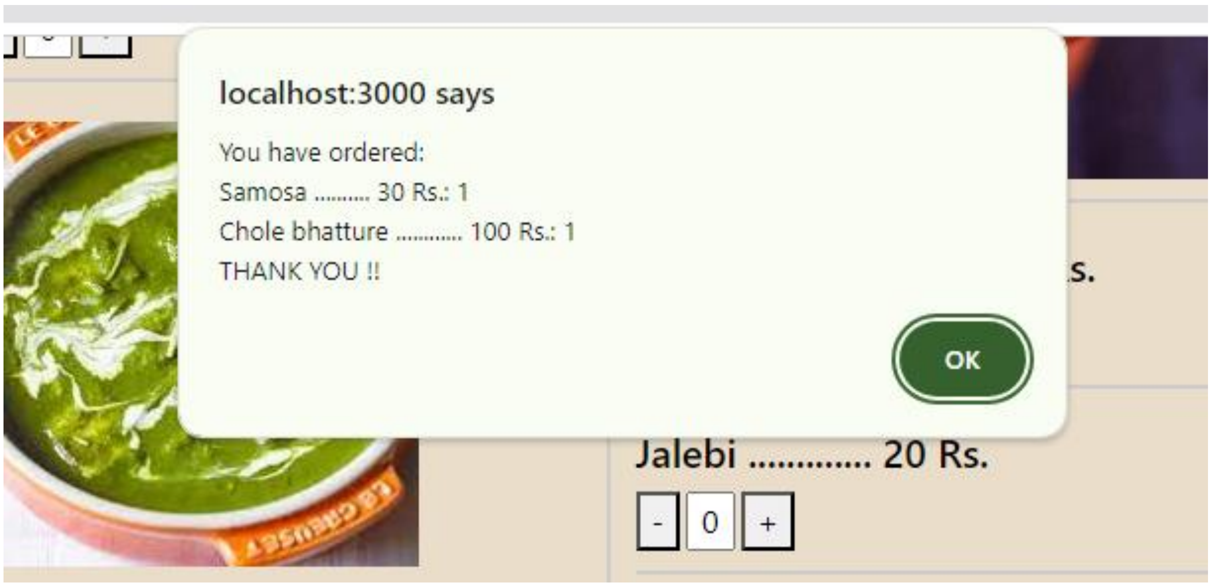
If nothing is added to the cart:



localhost:3000 says

Oops! Nothing added.

After adding to the cart:



localhost:3000 says

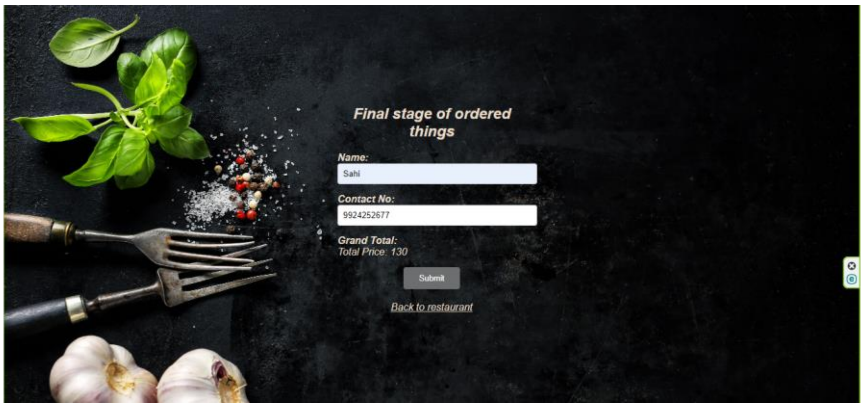
You have ordered:

Samosa .... 30 Rs.: 1

Chole bhatture .... 100 Rs.: 1

THANK YOU !!

Clicking on ok opens the menu1.ejs file where details of customer are taken and total payment is displayed.



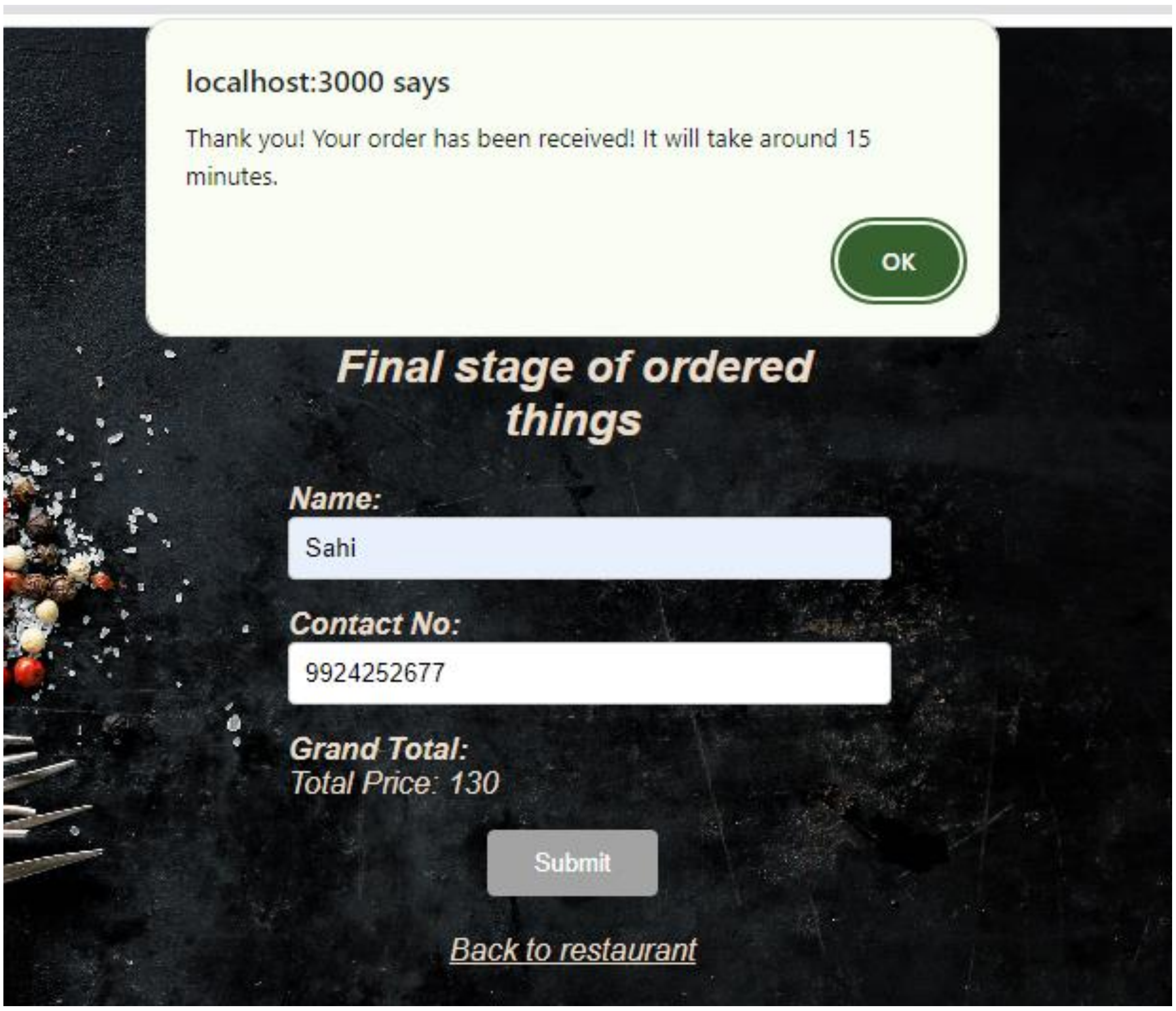
Final stage of ordered things

Name: Sahi

Contact No: 9924252677

Grand Total:
Total Price: 130

On clicking the submit button and alert will be shown:


localhost:3000 says

Thank you! Your order has been received! It will take around 15 minutes.

Final stage of ordered things

Name:
Sahi

Contact No:
9924252677

Grand Total:
Total Price: 130

Menu.js file which handles the functionality to order food from menu :

```
function increment(button) {
    var input = button.parentNode.querySelector('input');
    input.value = parseInt(input.value) + 1;
  }

  function decrement(button) {
    var input = button.parentNode.querySelector('input');
    input.value = parseInt(input.value) - 1 >= 0 ? parseInt(input.value) - 1 :
0;
  }

  function orderNow() {
    var quantities = document.querySelectorAll('.quantity input');
    var totalItems = 0;

    quantities.forEach(function(quantity) {
      totalItems += parseInt(quantity.value);
    });
    if (totalItems === 0) {
      alert("Oops! Nothing added.");
    } else {
      var message = "You have ordered:\n";
      quantities.forEach(function(quantity, index) {
        if (parseInt(quantity.value) > 0) {
          message +=
quantity.parentNode.previousElementSibling.textContent.trim() + ": " +
quantity.value + "\n";
        }
      });
      alert(message+"THANK YOU !!");
      function extractQuantitiesById() {
        // Initialize an empty object to store quantities
        var quantitiesById = {};

        // Select all menu items using a query selector
        var menuItems = document.querySelectorAll('.menu-item');

        // Loop through each menu item
        menuItems.forEach(function(item) {
          // Get the ID of the menu item by accessing its h3 tag and
retrieving the id attribute
          var itemId = item.querySelector('h3').id;

          // Get the quantity input value
          var quantity = parseInt(item.querySelector('.quantity
input').value);

          // Store the quantity in the object with the item ID as the key
          quantitiesById[itemId] = quantity;
        });

        return quantitiesById;
      }

      // Example usage
      var quantities = extractQuantitiesById();
      const table=[
        {item:'Samosa',Qty:30*quantities['Samosa']},
        {item:'Pani-Puri',Qty:20*quantities['Pani-Puri']},
        {item:'Paneer-Pakora',Qty:40*quantities['Paneer-Pakora']},
        {item:'Vadapav',Qty:50*quantities['Vadapav']},
        {item:'Chole',Qty:100*quantities['Chole']},
        {item:'Dal',Qty:120*quantities['Dal']},
        {item:'Lachcha',Qty:110*quantities['Lachcha']},
        {item:'Palak',Qty:120*quantities['Palak']},
        {item:'Rasmalai',Qty:80*quantities['Rasmalai']},
        {item:'Mango',Qty:70*quantities['Mango']},
        {item:'Jalebi',Qty:20*quantities['Jalebi']},
        {item:'Gulab',Qty:40*quantities['Gulab']},
        {item:'Butter',Qty:30*quantities['Butter']},
        {item:'Lemon',Qty:70*quantities['Lemon']},
        {item:'Fruit',Qty:100*quantities['Fruit']},
        {item:'Lassi',Qty:40*quantities['Lassi']},
        {item:'Aam',Qty:100*quantities['Aam']}
      ]
      let totalQuantity = 0;
      table.forEach((item) => {
          totalQuantity += item.Qty;
      });
      // console.log("Total Price:", totalQuantity);
      const queryString = 'totalQuantity=' +
encodeURIComponent(totalQuantity);
      const newUrl = '/menu1' + '?' + queryString;
      window.location.href = newUrl;
    }
  }
```

Menu1.js file which takes care about the validation of the details entered in the order form.

```
// Function to get the value of a query parameter from the URL
function getParameterByName(name, url) {
    if (!url) url = window.location.href;
    name = name.replace(/[\[\]]/g, '\\$&');
    var regex = new RegExp('[?&]' + name + '(=([^&#]*)|&|#|$)'),
        results = regex.exec(url);
    if (!results) return null;
    if (!results[2]) return '';
    return decodeURIComponent(results[2].replace(/\+/g, ' '));
}

// Get the value of the 'totalQuantity' query parameter
const totalQuantity = getParameterByName('totalQuantity');

// Now you can use the totalQuantity variable to access the total quantity
value
document.getElementById('grandTotal').textContent = 'Total Price: ' +
totalQuantity;

// Submit button click event listener
document.getElementById('submitBtn').addEventListener('click', function(event)
{
    event.preventDefault(); // Prevent default form submission behavior
    // Validate form fields
    const name = document.getElementById('name').value.trim();
    const contact = document.getElementById('contact').value.trim();
    let valid = true;

    // Validate name
    if (name === '') {
        document.getElementById('nameError').textContent = 'Name is required';
        valid = false;
    } else {
        document.getElementById('nameError').textContent = '';
    }

    // Validate contact number
    if (contact === '') {
        document.getElementById('contactError').textContent = 'Contact number
is required';
        valid = false;
    } else {
        document.getElementById('contactError').textContent = '';
    }

    // If form is valid, submit the form
    if (valid) {
        alert('Thank you! Your order has been received! It will take around 15
minutes.');
        // Optionally, you can submit the form here using AJAX or redirect to
another page
        // document.getElementById('orderForm').submit();
    }
});
```

Conclusion:

The hotel and restaurant website, employing HTML, CSS, and JavaScript for frontend development, alongside Node.js and Express.js for backend functionality, and MySQL for database management, presents a comprehensive digital solution tailored to the hospitality industry. By leveraging modern frontend technologies for intuitive user interfaces and dynamic interactions, coupled with a scalable backend infrastructure and robust database management, the website ensures seamless integration, optimal performance, and enhanced security. Through careful consideration of user experience, functionality, security, and scalability, businesses can create a compelling online platform that drives customer engagement, facilitates bookings, and fosters long-term success in the competitive hospitality market.

Future Expansion:

- Log in /Sign up can be inbuilt so that user can create an account on the site and can have a record of his bookings.
- Check availability functionality is lacking which can be done in further work.
- If the website is used in real-life, a payment API can be linked for transaction.
- Security features can be added mor to protect the customer details.
- This project can be expanded for booking rooms in different hotels from the same site.
- The restaurant website can be made to do delivery at home just like resembling Zomato.

References:

1. https://youtu.be/AYaWP-VOe-c?si=qHMumToyO6I1Rz0d
2. https://chat.openai.com
3. https://youtu.be/YSOY_NyOg40?si=P7H71o-HgSZW_RsR
4. https://youtu.be/eIjbSH3Imb8?si=zJfZxtnjNPKmQ3Vf