MODERN HOTEL BOOKING SYSTEM POWERED BY REACT AND FIREBASE

IT3711 - SUMMER INTERNSHIP REPORT

Submitted by

DIVYA DHARSHINI R 311521205015

HAIFA ZULAIKA Z 311521205016

submitted to the Faculty of

INFORMATION TECHNOLOGY

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE, KODAMBAKKAM

(An Autonomous Institution)

ANNA UNIVERSITY::CHENNAI 600025

NOVEMBER 2024

ANNA UNIVERSITY::CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report titled **MODERN HOTEL BOOKING SYSTEM POWERED BY REACT AND FIREBASE** is the bonafide work of **DIVYA DHARSHINI R (311521205015) and HAIFA ZULAIKA Z(311521205016)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

MR. MOHAN RAJ VIJAYAN M.E (CSE), (Ph.D.) M.B.A (PM), M.Music, M.Sc. (Psy), M.A (Philo), M.A (French), Dip.(Arch Epi)
INTERNSHIP CO-ORDINATOR
ASSISTANT PROFESSOR
DEPARTMENT OF IT, MSEC
(An Autonomous Institution)
KODAMBAKKAM
CHENNAI 600024

SIGNATURE

DR. A. KANIMOZHI M.E., Ph.D.
HEAD OF THE DEPARTMENT
DEPARTMENT OF IT, MSEC
(An Autonomous Institution)
KODAMBAKKAM
CHENNAI 600024

Submitted for the project viva voce held at Meenakshi Sundararajan Engineering College on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

This project presents modern room-booking system in a hotel powered by React and Firebase.The efficiency and friendliness while the customers surf through the rooms available for reservation have been the very aspects along with real-time easy manageability of bookings. It utilizes the back end for database maintenance and authentication even of live updates to ensure hassle-free interaction with the application by its users.The database has user details and room status, even the history of bookings, all managed on Firebase's No SQL database, which helps in fast access and updating. Customer management features have been incorporated into the application; be it account registration, logging, or history of bookings. This concludes that using React and Firebase is an efficient way in which scalable and responsive booking systems for the hospitality industry can be developed. The project symbolizes the capability of combining different technologies involved in the process of creating fluid and interactive applications. Future upgrades most probably would be done by integrating payment gateways and making the application mobile-friendly for easier access for all users.

ACKNOWLEDGEMENT

We are immensely grateful for the opportunity to express my heartfelt thanks to the **LORD ALMIGHTY** for showering His infinite blessings, wisdom, and guidance throughout this project. His unwavering support has been a constant source of strength, helping us overcome all challenges.

We extend our deepest gratitude to our President, **Dr. K.S. Lakshmi,** and my **Secretary Shri. N. Sreekanth** of the **IIET (Indian Institute of Engineering Technology) Society and Management Trust,** for their visionary leadership, providing the facilities and a conducive environment that greatly facilitated the successful completion of my project as part of my internship.

We sincerely thankful to our **Principal, Prof. Dr. S.V. Saravanan, of Meenakshi Sundararajan Engineering College,** for his constant encouragement, support, and guidance. We also express my profound thanks to **Prof. Dr. A. Kanimozhi**, **Head of the Information Technology Department,** for her invaluable guidance, insightful feedback, encouragement, and support throughout the completion of this internship project.

Our heartfelt thanks go to my **Assistant Professor and Internship Coordinator, Mr. Mohan Raj Vijayan,** for his invaluable suggestions, patience, guidance, and unwavering support throughout the duration of the project.

We would also like to express my deep appreciation to **CODSOFT** for providing me with the opportunity to carry out my internship, which played a significant role in the completion of this project.

Above all, We express my deepest sense of gratitude, respect, and reverence to my beloved **PARENTS** for their constant motivation, moral support, and encouragement throughout my lives.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The hospitality industry faces significant challenges in efficiently managing hotel bookings. This creates a significant challenge in the effective management of the hotel booking mainly because of increased demands from customers for seamless, user-friendly digital experiences. Traditional techniques of booking hotels involve manual operations that translate to time in reservations and error in respect to availability between systems.

The Modern Hotel Booking System powered by React and Firebase addresses these issues with this reason in serving online hotel reservation as an integral solution.The No SQL Firebase database updates room availability very fast, while React's component-based structure gives a fast and responsive user interface that will enhance the overall customer experience and make the management of hotels easier.

The primary purpose for which this study was undertaken is the development of a contemporary hotel room booking application with efficiency in reservation, reduced error cases,and customer satisfaction. This system provides real-time room checking,instant booking for rooms, and even management for bookings. This relies on the feature; functionality, usability and scope for further development and enhancement such as inclusion of payment gateway and mobile-friendliness advanced further to a system that shall be even more accessible and user-friendly,evaluates the study.

1.1 PROBLEM STATEMENT

In this modern travel industry context, lies the challenge where individuals and families are unable to find suitable accommodations that meet their needs and match their budget. Most existing hotel booking platforms are found not to have an easy interface coupled with powerful backend systems. These lead to inefficiencies as comparing options is difficult, and the system provides limited real-time updates and also less relevant customer engagement. A hotel reservation system of this kind creates a necessity for an efficient, fully integrated system that streamlines the booking process for the user while providing real-time information, recommending to individual customers, and management capabilities for the hotel owners. It would improve the customer experience and empower the hotelier in optimizing their operations to make online hotel reservations more sustainable and user-centric.

1.2 OBJECTIVE

The modern hotel booking system using react and firebase to be developed in the direction of building an online booking platform which can suitably answer the conventional hotel reservation systems.

- To improve the experience of customers through the easy process of searching, booking, and managing hotel reservations.
- To enable hotel owners to keep better track of room status, monitor bookings, and update customer information easily.
- To make scalable features for payment gateway options and advance filters that could work on different user needs and geographical locations.

1.3 SCOPE

During the development of a fully operable web application using React and Firebase within the context of an application booking system at a hotel, the back and front will be taken care of accordingly to ensure seamless interaction. The whole system will be mobile responsive; hence, they will be able to access it using a variety of devices. Basic functions such as user registration, login and tracking of booking history will be included in the platform. Their future release and feature might be to add support for the payment gateway, enhance the advanced filters for search, and integrate more languages to have expansive reach.

1.4 MOTIVATION

The motivation behind this project is the necessity for efficient, user-friendly, and scalable hotel booking systems in the hospitality industry. Customers also get fast technology-based in their pursuit of frictionless online experiences that enable them to easily book rooms with real-time updates and security. Traditional ways of booking people into hotels often characterize inefficiencies, errors, and delays that impact customer satisfaction and hotel operations. However, with all these modern technologies in place, which are React and Firebase, this project is going to bring the entire process of making reservations down to the bare minimum intervention and ensure that with any experience with a reservation there is smoothness without any errors. Thus, the solution has to be user-centric yet management-efficient for hotels.

CHAPTER 2

LITERATURE SURVEY

In the traditional hotel booking systems, reservations are mainly accessed manually or through basic websites, which lack real-time data synchronization and smooth two-way interaction between the user and the system. Users mostly have to contact the hotel for the status of rooms available,and this makes it all delayed and inconsistent. Even though online systems may have simple booking functionalities,they are mostly incomplete about real-time updates, authorized users, and proper data handling.There is, hence, a growing need for an infinitely more dynamic system that should be responsive and automated.

Oliveira A, Silva B, Souza C (2024) VEGAS EXPERIENCE: Application for the Assistance of Tourists in Las Vegas. EnGeTec em Revista,1:63–75

The VEGAS EXPERIENCE app is a specially created tourist assistant for Brazilian visitors to Las Vegas. It will provide local data on available tourist attractions, accommodation, and transportation through an intuitive and user-friendly interface. The software was built using React Native and Firebase and uses the cloud feature to get efficient data storage and cross-platform compatibility as well as enhanced usability through the development tool of Expo. It caters to the Portuguese-speaking user, with the app bringing in relevant out-of-the-box information to ease travel and address key challenges such users face. However, the app has some notable limitations. It is limited to Portuguese-speaking users only, thus limiting its appeal to a broader audience. Its dependency on internet connectivity will also make its usability an issue in unconnected areas. However, these shortcomings are at a

minimum in most productive regions where connectivity is stable. Generally speaking, the app seems to have enhanced the traveling experience of its users, thus making it worthwhile despite some drawbacks.

Borse S (2022) Cloud-based Hotel Management System. International Research Journal of Engineering and Technology, 9:1187–1191

This is an Internet-based hotel management system for paper-based operations substitution in the hospitality industry. The core inputs of the system are customer reservations, room services, and confirmations. In a bid to increase operations efficiency, it allows the customers to interface using a digital platform with the concept of improved satisfaction.It has used modern technologies in building the user interface with ReactJS, for server-side operations with Express JS, and Firebase as a No SQL database for real-time data storage. This enables instant updates and full transparency, enhancing decision-making by the managers of hotels while making sure the operation runs perfectly. However, it also presents several limits difficult learning curve for new users and a reliance on sustained internet connection, which may impose particular challenges in areas of locality. However, still the system presents the important point regarding modernization in hotel management practices.

Azmie IA, Jayathilaka ADN, Paboda PDK, Nawarathna MTI, De Silva DI, Cooray D (2022) Smart Tourist Assistance System for Sri Lanka. International Journal of Computing and Digital Systems,11:567–579

Travel Go is a tourist management system aspires to improve the travel experience and ensure safety to tourists visiting Sri Lanka. It's introducing authentication for user and hotel booking data to make travel planning easier, increase access for customers for travel and accommodation services, and promote the development of Sri Lanka's tourism industry. Its concept revolves

around ensuring information availability in a manner that makes planning journeys for tourists easy while ensuring an hassle-free travel experience. However the system has many weaknesses-the system heavily depends upon uninterrupted internet connectivity for potential use in inaccessible regions and creates security concerns since it might well store and guard user data. Future developments will consider solving these bugs by creating a mobile application to make things more convenient. Despite its flaws, Travel Go is one giant leap in terms of modernization within Sri Lanka's tourism infrastructure, a reliable and user-friendly platform for the tourist and service provider alike.

Rahimi A, Ahmad B, Hassan C, Ismail D (2022) Hotel Booking System for Ridel Hotel Kota Bharu. Applied Information Technology and Computer Science,3:912–929.

It is a live application hotel room booking made in React and Firebase because it is targeting the easing of the management of hotels and also the user experience. An application that is a living interface for the users so that they can intuitively see what is available, make bookings, and get more detailed information about hotels in real time. During user interaction with the front end interface, the back-end side of Firebase maintains security features related to data storage and authentication by the user. The No SQL structure of Firebase supports real-time update of user details, booking records, and hotel information where data access and management happen relatively efficiently and easily. It has dramatically reduced the booking time, improved data accuracy, and allowed seamless access to room availability and booking details. As far as efficiency and response from the system are concerned, the testing results proved to have high user satisfaction. The project is very important because it assumes the job set by technology in the hospitality industry to build scalable applications with React and Firebase. Further developments may include making the application mobile-friendly and integrating a payment gateway to make the application even

more versatile and convenient for users. Thus, this system is a beginning for modernization in hotel management processes because it attains this standard within the demand of an industry toward efficiency and reliability.

Bemile R, Achampong A, Danquah E (2014) Online Hotel Reservation System. International Journal of Innovative Science, Engineering and Technology 1:583–588.

It is a live application hotel room booking made in React and Firebase because it is targeting the easing of the management of hotels and also the user experience. An application that is a living interface for the users so that they can intuitively see what is available, make bookings, and get more detailed information about hotels in real time. During user interaction with the front end interface, the back-end side of Firebase maintains security features related to data storage and authentication by the user. The NoSQL structure of Firebase supports real-time updatism of user details, booking records, and hotel information where data access and management happen relatively efficiently and easily. It has dramatically reduced the booking time, improved data accuracy, and allowed seamless access to room availability and booking details. Further developments may include making the application mobile-friendly and integrating a payment gateway to make the application even more versatile and convenient for users. Thus, this system is a beginning for modernization in hotel management processes because it attains this standard within the demand of an industry toward efficiency and reliability.

CHAPTER 3

SYSTEM REQUIREMENTS

The robust configurations of its hardware and software ensure the attainment of High-Performance Functionality for the Modern Hotel Reservation System. Such configurations are very important in processing real-time data to ensure that user interactions are handled smoothly, safety and efficiency are guaranteed in the operations of hotel reservations.

3.1 SOFTWARE REQUIREMENTS

- **Operating System**: The cross-platform application should work without any issues on multiple operating systems, granting freedom to developers as well as end-use.
- **React**: The component-based structure ensures that reusable and scalable UI elements, thereby making the development process more efficient. Fast rendering comes with having the virtual DOM by React thus maximizing user experience.
- **Firebase**: It is a highly potent BaaS platform. It supports Realtime Database for real time update on booking status, availability, customer information,etc.
- **Node.js**: Node.js is pretty handy to apply in API development, server-side rendering, or involving all other features and advanced analytics.

3.2 LIBRARIES USED

Modern Hotel Booking System utilizes several libraries and frameworks to make development easier, functionality better, and to generally enrich the user experience. Each of the library has its role in either contributing towards the aspect of the front-end or the back-end of the application.

- **React**: This is the major library used in the creation of the front-end interface.Dynamically and responsively uses reusable components to create the most appealing user interface.It has Virtual DOM for best rendering.
- **React Router**: This library will handle the routing inside your application.It allows navigation between pages - home, hotel listings, booking history. Declarative routing, which is easier to read Dynamic parameters to the URL for better, personalized content.
- **Tailwind CSS**: User Interface Component Library.It makes your styling easier, with a modern consistent look over your application.

3.3 HARDWARE REQUIREMENTS

- **Processor** It is prudent to use a multi-core processor like the Intel Core i5 or AMD Ryzen 5 so that data operations and interaction with the database can be executed without hassles within the application.
- **Memory (RAM)** Use at least 8 GB RAM for the Java application and database processes so that they run very smoothly. The more RAM is allocated in the system, preferably 16 GB RAM, the higher the likelihood of handling larger student datasets when multitasking.

- **Storage** There should also be sufficient storage space to accommodate files of the project and records of all the students in the database. Installation of a Solid State Drive would improve read/write times. This would result in faster data access and retrieval in the application.
- **Internet Connection** Startup Apps may require internet to download the necessary libraries, dependencies or receive updates.

3.4 GRAPHICAL USER INTERFACE

It is intuitive and user-friendly. With such a large library of widgets and such flexible layout options on React, it should all flow along-seamlessly from browsing through hotels to booking your stay to managing a hotel.

- **Navigation Bar, Sidebar**: It should be easy to navigate to the core pages-Home, Search, Bookings, Profile.
- **Input Fields, Dropdowns, and Buttons**: Search hotels by name or via date range; fill in your preferred dates; room selection; book initiation
- **Calendar Picker and List View**: Calendar to select the check-in date and check-out date, List View for displaying hotel options by filtering out location and cost.
- **Responsive Elements** All-in-one cohesive experience for mobile as well as desktop. Interactive elements placed in such a way so there is no room for clutter and accessibility turns out uncomplicated.

CHAPTER 4

SYSTEM DESIGN

The Modern Hotel Booking System aims to enhance the booking functionality will help of real-time available data and smoothen the booking process with safe management of the customer s' data in a friendly interface. The front here is React for an interactive and responsive user interface to make a excellent user experience across platforms. It uses Firebase as the back-end with real-time updates in user authentication and booking information storage. The system reduces all the work that a hotel management has to perform, starting from how many rooms are available up to customer details, making work ease and minimizing manual errors.It integrates functionalities like booking, live room availability, and secured login systems for clients and the managers of the hotel. This application ensures immediate updating of data hence accurate and up-to-date information for all the clients.

4.1 ARCHITECTURE DIAGRAM

The Modern Hotel Booking System's architecture is provided for proper user flow. It starts with the user interface, which leads the user to a list of hotels and thus enables the selection of preferred options, after which comes the login page so that only secured access is provided to customers. Once the user logs in correctly, he can go ahead with the booking process where room availability merged with real-time booking information can be fetched from the backend powered with Firebase. Finally when booking is confirmed, on generating the bill, the system provides a tabulation of itemized summary of the reservation.

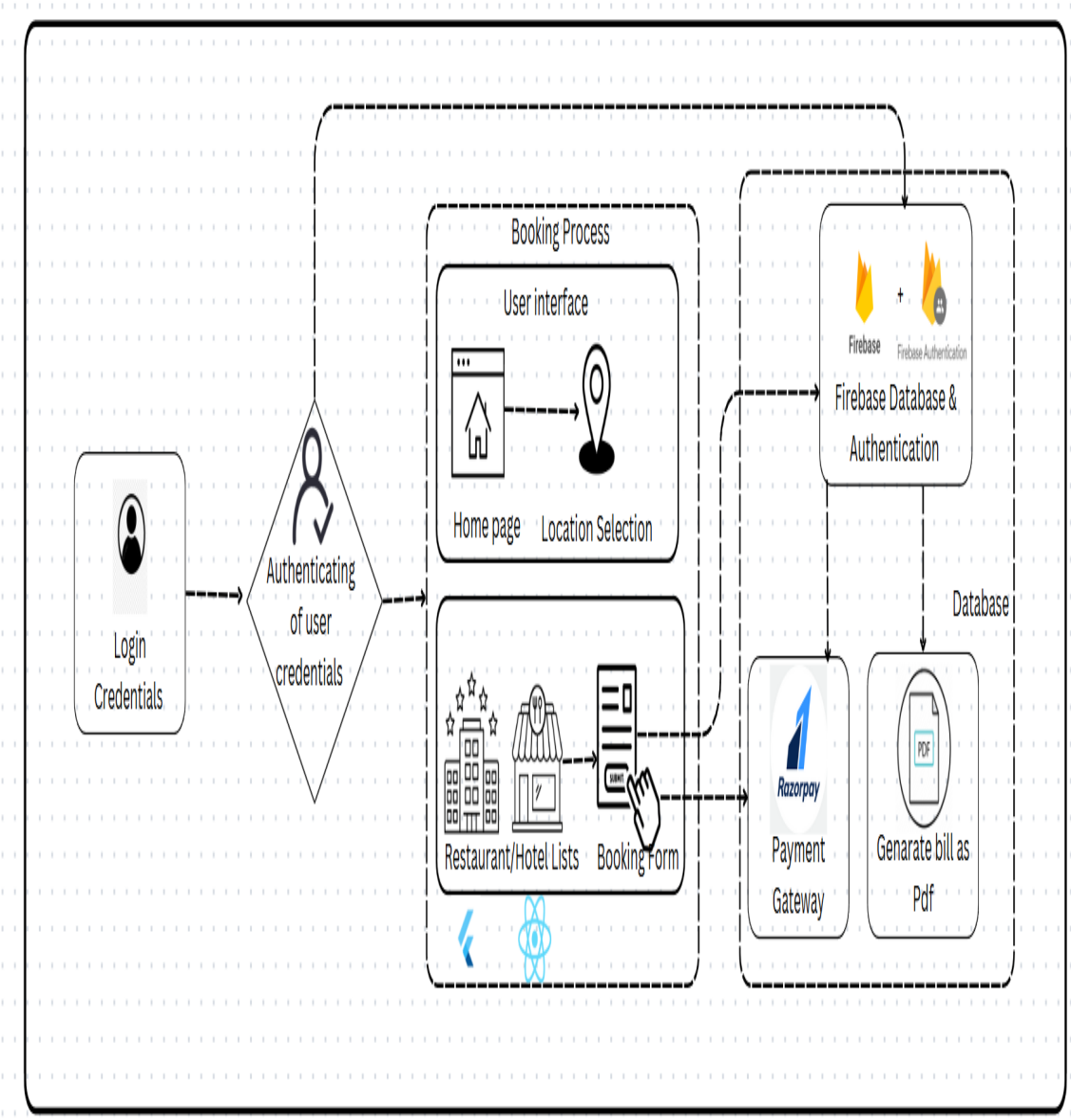

Figure 4.1 Architecture Diagram

4.2 LIST OF MODULES

The System is designed to have utmost usability, integrity of data, and efficiency for a seamless reservation through various modules.These modules are listed below, each of which will be discussed in detail.

1. User Input Module
2. Booking Management Module
3. Payment Module
4. Backend Storage Module

4.2.1 User Input Module

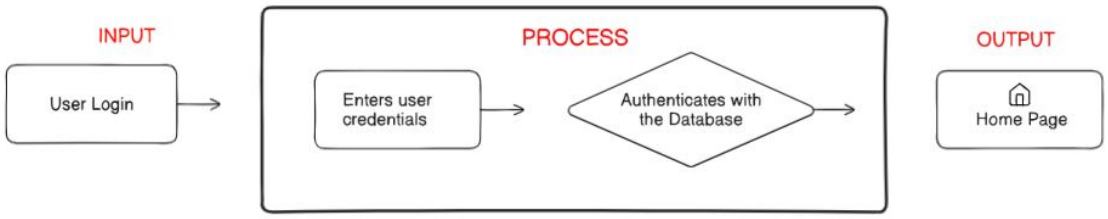

Figure 4.2 User Module

The User Input Module is the point of interaction of the system. It collects all relevant information of users, including their personal data and login details, as well as preference on reservation. All these inputs submitted by the user are passed on to the Firebase for authentication for further processing.The inclusion of real-time data validation and error handling modules ensures a seamless user experience in the search process of the application.

4.2.2 Booking Management Module

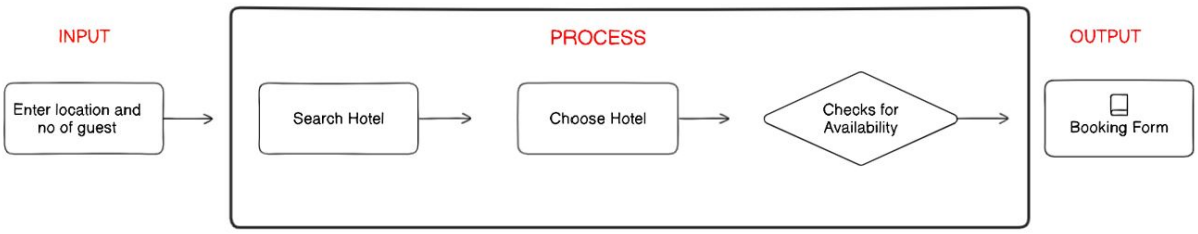

Figure 4.3 Booking Module

This module takes care of Availability of rooms and canceling user reservations. After validating that a room is available, the module processes the booking request a user has made, creates a reservation unique ID and saves the reservation to the back-end database,Firebase.

4.2.3 Payment Module

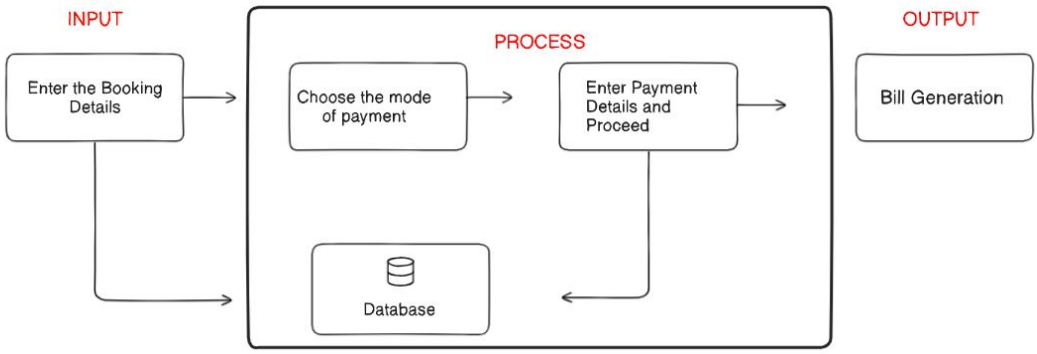

Figure 4.4 Payment Module

The Payment Module is going to be one of the new features that will eventually be included in the latter versions, allowing users to generate a form of online payment functionality for reservation purposes.This module integrates with external payment gateways like Razor pay or credit and debit card processing services to make secure payments through the hotel booking.

4.2.4 Backend Storage Module

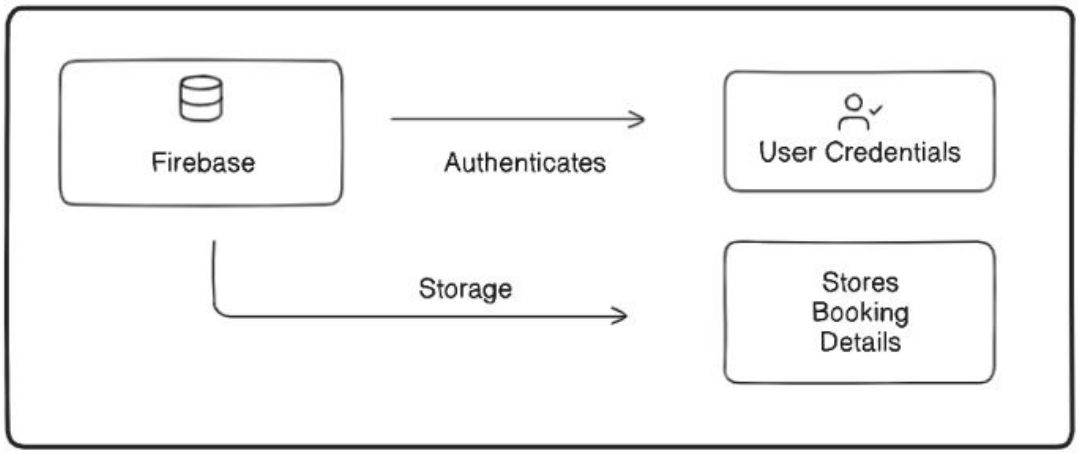

Figure 4.5 Backend Storage Module

The Data Management Module of Modern Hotel Booking System is implemented on Firebase. It thus manages all data within real-time and keeps it updated. For example, if new bookings or cancellations are entered into the system, that immediately reflects in the module as well. All authentication information will be dealt with securely by Firebase Authentication for each user. Reservations accommodation is normally logged together with a booking ID, dates, details of the hotels involved and payment status; additionally, it is possible for users to view and edit their reservation in real time. Firebase, therefore does manage huge amounts of growing data and user demands with its scalable, high-performance, cloud-based infrastructure and optimized queries as well as indexing to enable very fast retrieval of data.

CHAPTER 5

IMPLEMENTATION AND RESULT

5.1 IMPLEMENTATION

1. Login

The login page of the restaurant project contains a default email ID and password so that users can check how the application works as a guest or for initial testing. A Logo of the website is given nicely at the top right section of the page to strengthen the recognition and professional approach of the brand. It creates a pleasant visual appearance along with building trust for the brand. Using the credentials provided, users can log in and access the home page with ease while exploring all available features.

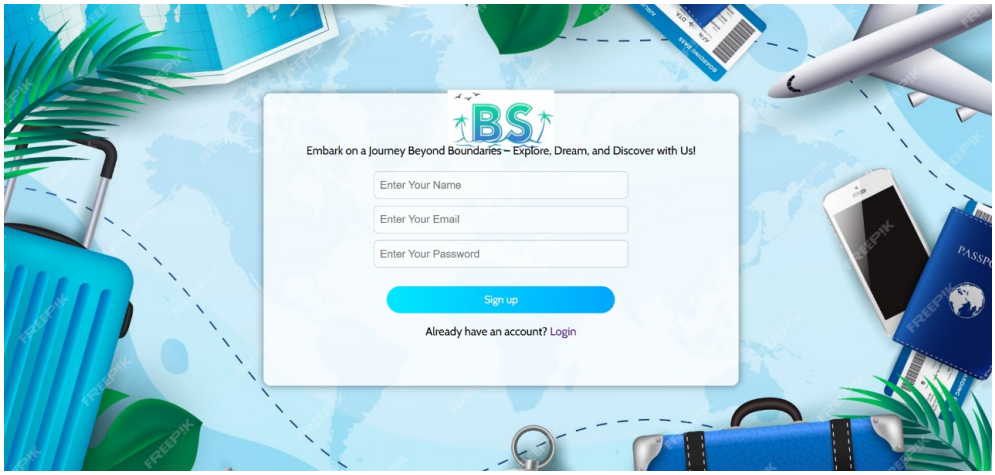

Figure 5.1 Login

2. Home Page

The Home Page is the core dashboard of the application in which the users find an overview of restaurants available. It has a clean and user-friendly interface with a title banner at the top. Below the banner, a dynamic restaurant list will be displayed with respect to the selected location including essential details on restaurant names, types of cuisine, ratings, etc., and hours of operation. The page also contains a search tab so that visitors may filter restaurants according to location and desires, which enhances user experience and guides the visitor for quicker selection of desired dining options.

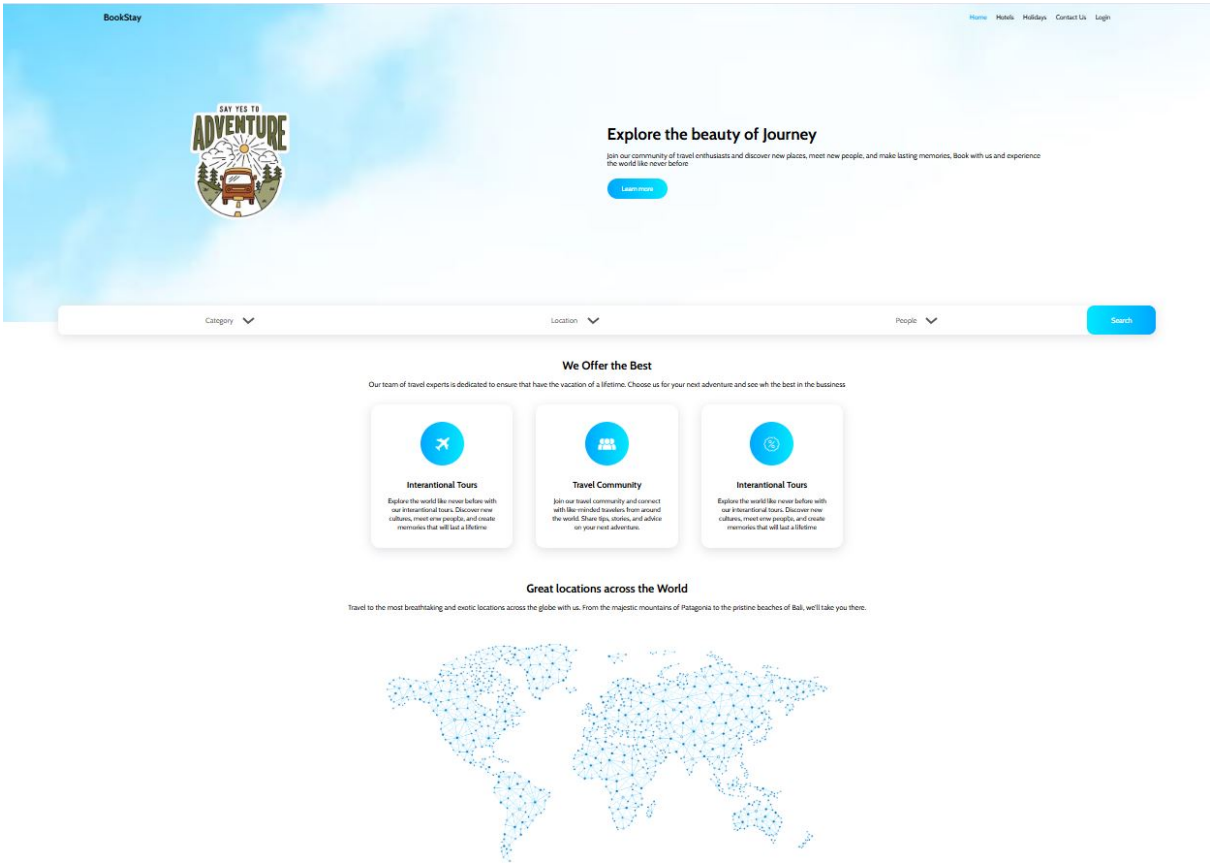

Figure 5.2 Home Page

3. Booking Page

The important feature regarding the restaurant reservation system is the Booking Form. This booking form allows the users to easily book a table at their desired location's restaurant. After choosing a restaurant, the user will be required to fill up different boxes for personal details, such as name, contact number, and the number of people for the reservation and slots . The form also calls for the most preferred date and time of dining. It covers both lunch and dinner time slots.

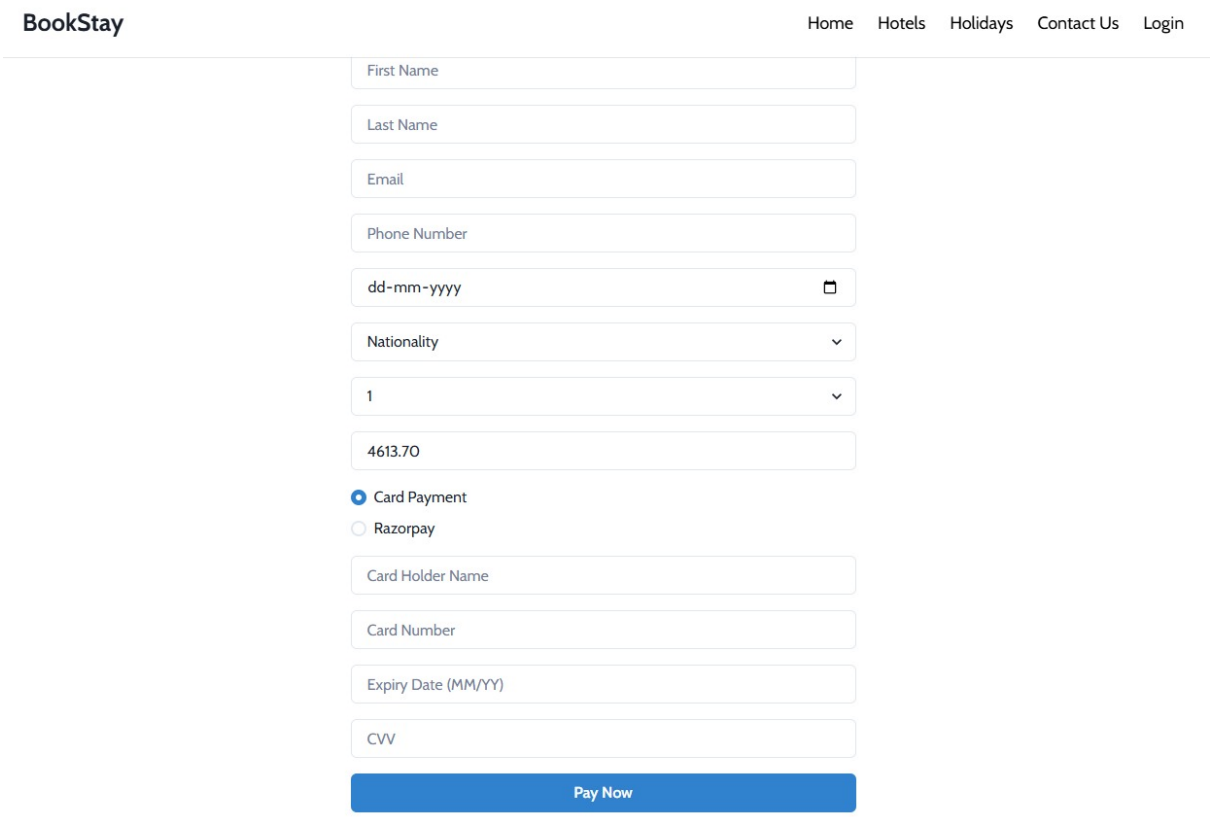

Figure 5.3 Booking Page

4. Payment Options

The smooth payment processing is interfaced with a popularly used and a secure payment gateway: Razorpay. If the Users have filled their booking details and confirmed their reservation, they will be taken through to the payment completion. The Razorpay payment gateway allows the user to choose any of the several available payment options that includes credit/debit cards, UPI, or wallets. Once a payment goes through successfully, it sends a confirmation that the payment indeed does go through securely. It sends a successful transaction message to the user and marks the reservation as confirmed.

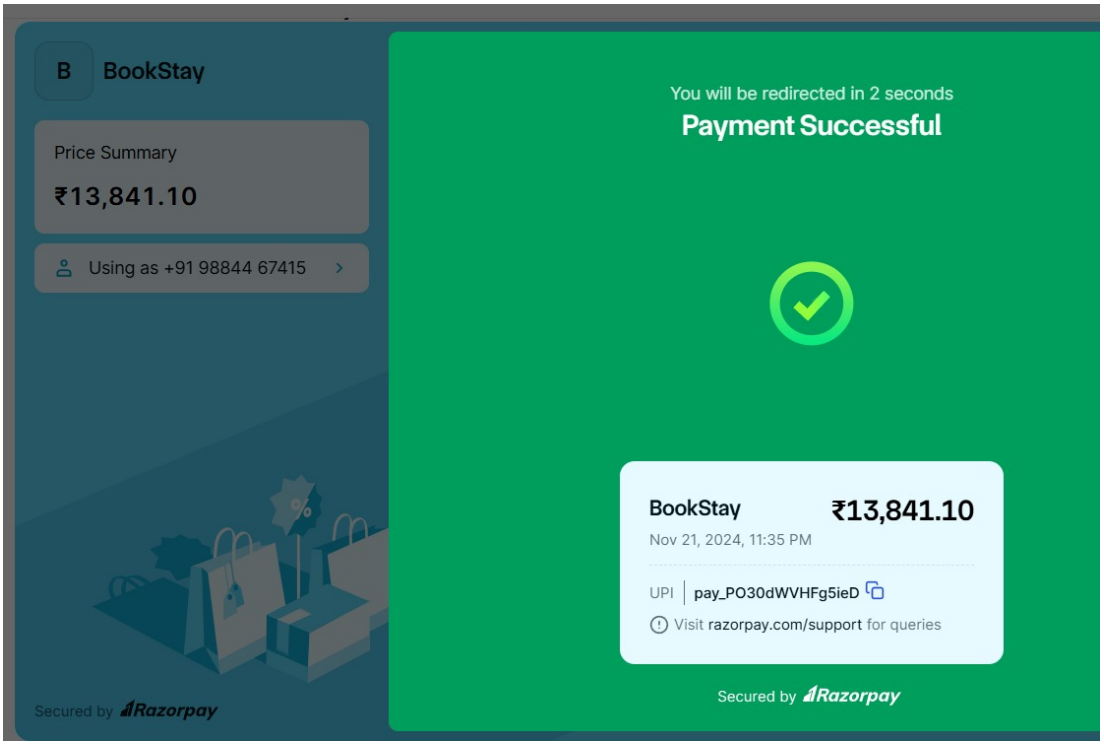

Figure 5.4 Confirmation

5. Database

The Firebase Realtime Database stores and handles all the information relevant to the restaurant reservation system. It ensures smooth storage of the data regarding user information, hotel details, room availability, and booking status. hotel names, locations, and operational hours are kept as structured data and hence are easy to query and come up on the front-end for display. Reservation details of users, such as the date, time slot, and how many people, are all safely saved on the Firebase end. In case any data is altered, it will immediately reflect on all devices, thereby giving the correct information to the person.

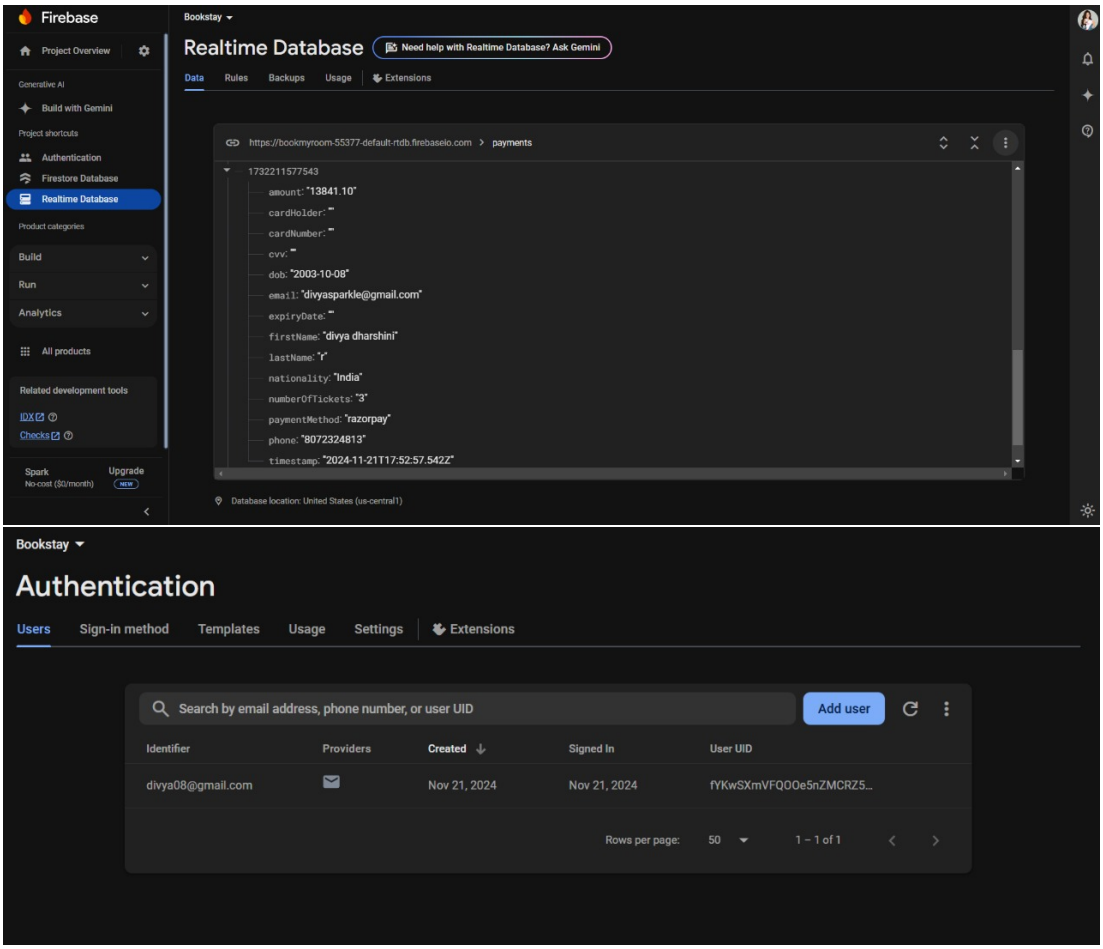

Figure 5.5 Confirmation

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The Modern Hotel Reservation System is a simple, powerful web application meant to simplify the hotel booking process as much as possible. It uses modern technology and tools. React on the front end and Firebase on the back end to realize a flawless interaction between the user interface and the database with minimum to no disconnections.The very key features in this application are an attraction-filled room carousel to browse through, and the intuitive booking management system. It ensures security for authentication, real-time data feeds and central storage for user and hotel data by integrating it with Firebase.The system is capable of serving all the different needs of its users in finding a hotel close to them, sorting results according to varied preferences or even managing bookings.In the Modern Hotel Reservation System, however,it not only indicates the current challenges in hotel booking but also provides a basis for future development, such as payment gateway integration, advanced filters, the application of support in many languages and cross-platform accessibility. For all the above considerations, the project under offer gives a well-supported and interactive interface to the users as well as the administrators, thus entailing efficiency, precision and smoothness in bookings.

6.2 FUTURE ENHANCEMENT

Moving forward, many enhancements can be made to the Modern Hotel Reservation System toward better functionality and also in user experience. First of all, an improvement would be advanced analytics added to the database so that it would then enable dynamic report generation, through which administrators will monitor trending like customer preference, booking patterns, and seasonal demand. Additional benefits would be that room availability, pricing, and offers can change instantly across the system in real-time. More, user profiles could be extended to include history, like previous bookings and preferred room types for more customized recommendations and offers. A secure payment gateway will deploy ease and convenience for users in having smooth transaction thus increasing its trust-building aspect. Apart from that, mobile applications might also be developed to expand the reach of the system as it provides on-the-go booking options and real-time notifications. Many features of accessibility can be extended through multi-language support and also through the easy-to-use interfaces. In the long run, it needs to be expanded with other modules like other attractions, dining recommendations, and travel services that may position it more as a holistic travel solution. It may turn out to be a well-rounded, scalable, and impactful platform in hospitality.

CHAPTER 7

APPENDIX I - PROGRAM CODE

MODEL TRAINING CODE

HOMEPAGE

```
import React from "react";
import styles from "./HomePage.module.css";
import roundImage from "./images/roundtwoimage.png";
import mountains from "./images/mountains.png";
import Filter from "./filter/Filter";
import ContactUs from "../contact-us/ContactUs";
import BorderBox from "../common-styles/BorderBox";
import { useDispatch } from "react-redux";
import { updateSingleProduct } from "../../Redux/payment/action-creator";
import { useNavigate } from "react-router";
import { setSingleProduct } from "../../Redux/singleproduct/action-creator

function HomePage() {
  const dispatch = useDispatch();
  const nagivate = useNavigate();
  return (
    <BorderBox>
      {/* first section  */}
      <div className={styles.first_section}>
        <div className={styles.left_side_first}>
```

```
          <img src={roundImage} className={styles.rounded_image_first} />
          {/* <img src="..." className={styles.rounded_image_first} /> */}
        </div>

        <div className={styles.right_side_first}>
          <h1>Explore the beauty of Journey</h1>
          <p>
            Join our community of travel enthusiasts and discover new plac
            meet new people, and make lasting memories. Book with us and
            experience the world like never before.
          </p>
          <button>Learn more</button>
        </div>
      </div>

      {/* Additional sections */}
      <Filter />
      <ContactUs />
    </BorderBox>
  );
}
```

APP.CSS

```
import "./App.css";
import Footer from "./Components/footer/Footer";
import "./Components/common-styles/index.css";
import AllRoutes from "./Routes/AllRoutes";
import { useLocation } from "react-router-dom";
import Navbar from "./Components/navbar/Navbar";
```

```
function App() {
  const location = useLocation();
  const isWhite = location.pathname !== "/" ? true : false;
  return (
    <>
      <div className="App">

        )}
        <AllRoutes />
      </div>
      "/signup" && (
        <Footer />
      )}
    </>
  );
}
```

ROUTER.JS

```
export default App;
export default HomePage;
import React from "react";
import ReactDOM from "react-dom/client";
import "./index.css";
import App from "./App";
import reportWebVitals from "./reportWebVitals";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import { store } from "./Redux/store";
```

```
import { HolidayContextProvider } from "./Holiday/HolidayContext";
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <Provider store={store}>
    <BrowserRouter>
      <HolidayContextProvider>
        <App />
      </HolidayContextProvider>
    </BrowserRouter>
  </Provider>
);
```

FIREBASE.JS

```
import { initializeApp } from "firebase/app";
import { getAuth, GoogleAuthProvider } from "firebase/auth";
import { getFirestore } from "firebase/firestore"; // Import Firestore
import { getAnalytics } from "firebase/analytics";
const firebaseConfig = {
  apiKey: "AIzaSyDMMTBW_qqMFwigQbZxLn7eyIC1L6FTscI",
  authDomain: "bookmyroom-55377.firebaseapp.com",
  projectId: "bookmyroom-55377",
  storageBucket: "react-auth-e033d.appspot.com",
  messagingSenderId: "998920818864",
  appId: "1:998920818864:web:f993ddb02db9994f6deb75",
  measurementId: "G-DYHFKQ3XE5",
};

const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);
```

```
const auth = getAuth();
const db = getFirestore(app); // Initialize Firestore
const provider = new GoogleAuthProvider();

export { app, auth, db, provider };
import React from "react";
import { Routes, Route } from "react-router-dom";
import HomePage from "../Components/homepage/HomePage";
import ContactUs from "../Components/contact-us/ContactUs";
import Holidays from "../Holiday/HolidayPage/Holidays";
import { Login } from "../Components/Login/Login";
import { Signup } from "../Components/Signup/Signup";
import Payment from "../Components/Payment/Payment";
import Thankyou from "./../Components/Payment/Thankyou";
import Hotel from "../Hotels/Hotel";
import { ChakraProvider } from "@chakra-ui/react";
import HotelSingleInfo from "../Hotels/HotelSingleInfo";



function AllRoutes() {
  return (
    <Routes>
      <Route path="/" element={<HomePage />}></Route>
      <Route path="/contactus" element={<ContactUs />}></Route>
      <Route path="/hotel" element={<Hotel />}></Route>
      <Route path="/singlepage" element={<HotelSingleInfo />}></Route>
      <Route path="/holidays" element={<Holidays />}></Route>
      <Route path="/login" element={<Login />}></Route>
      <Route path="/Signup" element={<Signup />}></Route>
```

```
      <Route
        path="/Payment"
        element={
          <ChakraProvider>
            <Payment />
          </ChakraProvider>
        }></Route>
      <Route
        path="/Payment-Success"
        element={
          <ChakraProvider>
            <Thankyou />
          </ChakraProvider>
        }></Route>
    </Routes>
  );
}

export default AllRoutes;
```

Payment.js

```
import React, { useState, useEffect } from "react";
import {
  Box,
  Button,
  FormControl,
  Input,
  Select,
  VStack,
```

```
  Text,
  useToast,
  RadioGroup,
  Radio,
  HStack,
  Divider,
} from "@chakra-ui/react";
import { db, ref, set } from "D:\\projects\\bookstay\\src\\firebase.js";

const Payment = () => {
  const [formData, setFormData] = useState({
    firstName: "",
    lastName: "",
    email: "",
    phone: "",
    dob: "",
    nationality: "",
    cardHolder: "",
    cardNumber: "",
    expiryDate: "",
    cvv: "",
    numberOfTickets: 1,
    amount: "",
  });

  const [paymentMethod, setPaymentMethod] = useState("card");
  const toast = useToast();

  const roomRate = 6591; // Base rate per ticket
  const discountRate = 0.3; // 30% discount
```

```
useEffect(() => {
  const script = document.createElement("script");
  script.src = "https://checkout.razorpay.com/v1/checkout.js";
  script.async = true;
  document.body.appendChild(script);
}, []);

const handleChange = (e) => {
  const { name, value } = e.target;
  setFormData((prev) => ({
    ...prev,
    [name]: value,
    amount:
      name === "numberOfTickets"
        ? ((roomRate * value) * (1 - discountRate)).toFixed(2)
        : prev.amount,
  }));
};

const handlePaymentMethodChange = (value) => {
  setPaymentMethod(value);
};

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

  // Validate required fields
  if (
    !formData.firstName ||
```

```
  !formData.lastName ||
  !formData.email ||
  !formData.phone ||
  !formData.dob ||
  !formData.nationality ||
  !formData.numberOfTickets ||
  !formData.amount
) {
  toast({
    title: "Please fill all required fields.",
    status: "warning",
    position: "top",
    duration: 3000,
    isClosable: true,
  });
  return;
}

if (paymentMethod === "card" && (
  !formData.cardHolder ||
  !formData.cardNumber ||
  !formData.expiryDate ||
  !formData.cvv
)) {
  toast({
    title: "Please fill all card details.",
    status: "warning",
    position: "top",
    duration: 3000,
    isClosable: true,
```

```
  });
  return;
}

// If Razorpay is selected
if (paymentMethod === "razorpay") {
  const options = {
  key: "rzp_test_dnv3nQiWbqzTGt",
    amount: formData.amount * 100, // Amount in paise
    currency: "INR",
    name: "BookStay",
    description: "Payment for services",
    handler: async function (response) {
      try {
        const dataToWrite = {
          ...formData,
          paymentMethod,
          razorpayPaymentId: response.razorpay_payment_id,
          timestamp: new Date().toISOString(),
        };

        const newPaymentRef = ref(db, payments/${Date.now()});
        await set(newPaymentRef, dataToWrite);
        toast({
          title: "Payment successful",
          status: "success",
          position: "top",
          duration: 3000,
          isClosable: true,
        });
```

```
      } catch (e) {
        console.error("Error adding document: ", e);
        toast({
          title: "Payment failed",
          status: "error",
          position: "top",
          duration: 3000,
          isClosable: true,
        });
      }
    },
    prefill: {
      name: ${formData.firstName} ${formData.lastName},
      email: formData.email,
      contact: formData.phone,
    },
    notes: {
      address: "Corporate Office",
    },
    theme: {
      color: "#61dafb",
    },
  };

  const paymentObject = new window.Razorpay(options);
  paymentObject.on("payment.failed", function (response) {
    toast({
      title: "Payment failed",
      description: response.error.description,
      status: "error",
```

```
      position: "top",
      duration: 3000,
      isClosable: true,
    });
  });
  paymentObject.open();
  return;
}

// For card payment
try {
  const dataToWrite = {
    ...formData,
    paymentMethod,
    timestamp: new Date().toISOString(),
  };

  const newPaymentRef = ref(db, payments/${Date.now()});
  await set(newPaymentRef, dataToWrite);
  toast({
    title: "Payment successful",
    status: "success",
    position: "top",
    duration: 3000,
    isClosable: true,
  });
} catch (e) {
  console.error("Error adding document: ", e);
  toast({
    title: "Payment failed",
```

```
      status: "error",
      position: "top",
      duration: 3000,
      isClosable: true,
    });
  }
};

return (
  <Box p={10} maxW="600px" mx="auto">
    <Text fontSize="2xl" fontWeight="bold" mb={5}>
      Payment Information
    </Text>

    {/* Price Breakdown Section */}
    <Box mb={5}>
      <Text fontSize="xl" fontWeight="semibold">Price Breakdown</Text>
      <HStack justifyContent="space-between">
        <Text>Base price</Text>
        <Text>1 Traveller x 6,591</Text>
      </HStack>
      <HStack justifyContent="space-between">
        <Text>Discount</Text>
        <Text>- 1,977.3</Text>
      </HStack>
      <Divider />
      <HStack justifyContent="space-between" fontWeight="bold">
        <Text>Total due</Text>
        <Text>4,613.7</Text>
      </HStack>
```

```
  <HStack justifyContent="space-between">
    <Text>Due today</Text>
    <Text>2,306.85</Text>
  </HStack>
</Box>

<form onSubmit={handleSubmit}>
  <VStack spacing={4}>
    <FormControl isRequired>
      <Input
        name="firstName"
        placeholder="First Name"
        value={formData.firstName}
        onChange={handleChange}
      />
    </FormControl>
    <FormControl isRequired>
      <Input
        name="lastName"
        placeholder="Last Name"
        value={formData.lastName}
        onChange={handleChange}
      />
    </FormControl>
    <FormControl isRequired>
      <Input
        name="email"
        type="email"
        placeholder="Email"
        value={formData.email}
```

```
    onChange={handleChange}
  />
</FormControl>
<FormControl isRequired>
  <Input
    name="phone"
    type="tel"
    placeholder="Phone Number"
    value={formData.phone}
    onChange={handleChange}
  />
</FormControl>
<FormControl isRequired>
  <Input
    name="dob"
    type="date"
    placeholder="Date of Birth"
    value={formData.dob}
    onChange={handleChange}
  />
</FormControl>
<FormControl isRequired>
  <Select
    name="nationality"
    placeholder="Nationality"
    value={formData.nationality}
    onChange={handleChange}
  >
    <option value="India">India</option>
    <option value="USA">USA</option>
```

```
    <option value="UK">UK</option>
    <option value="Canada">Canada</option>
  </Select>
</FormControl>
<FormControl isRequired>
  <Select
    name="numberOfTickets"
    placeholder="Number of Tickets"
    value={formData.numberOfTickets}
    onChange={handleChange}
  >
    {[1, 2, 3, 4, 5].map((num) => (
      <option key={num} value={num}>
        {num}
      </option>
    ))}
  </Select>
</FormControl>
<FormControl isRequired>
  <Input
    name="amount"
    type="number"
    placeholder="Amount (INR)"
    value={formData.amount}
    readOnly
  />
</FormControl>
<FormControl isRequired>
    <VStack align="start">
      <Radio value="card">Card Payment</Radio>
```

```
      <Radio value="razorpay">Razorpay</Radio>
    </VStack>
  </RadioGroup>
</FormControl>
{paymentMethod === "card" && (
  <>
    <FormControl isRequired>
      <Input
        name="cardHolder"
        placeholder="Card Holder Name"
        value={formData.cardHolder}
        onChange={handleChange}
      />
    </FormControl>
    <FormControl isRequired>
      <Input
        name="cardNumber"
        placeholder="Card Number"
        value={formData.cardNumber}
        onChange={handleChange}
      />
    </FormControl>
    <FormControl isRequired>
      <Input
        name="expiryDate"
        placeholder="Expiry Date (MM/YY)"
        value={formData.expiryDate}
        onChange={handleChange}
      />
    </FormControl>
```

```
              <FormControl isRequired>
                <Input
                  name="cvv"
                  placeholder="CVV"
                  value={formData.cvv}
                  onChange={handleChange}
                />
              </FormControl>
            </>
          )}
          <Button type="submit" colorScheme="blue" width="full">
            Pay Now
          </Button>
        </VStack>
      </form>
    </Box>
  );
};

export default Payment;
```

CHAPTER 8

REFERENCES

5

[1] Azmie I, Jayathilaka ea ADN (2022) Smart tourist assistance system for sri lanka. International Journal of Computing and Digital Systems 11:567–579

[2] Bemile R, Achampong A, Danquah E (2014) Online hotel reservation system. International Journal of Innovative Science, Engineering and Technology 1:583–588

[3] Borse S (2022) Cloud-based hotel management system. International Research Journal of Engineering and Technology 9:1187–1191

[4] Oliveira A, Silva B, Souza C (2024) Vegas experience: Application for the assistance of tourists in las vegas. EnGeTec em Revista 1:63–75

[5] Rahimi A, Ahmad ea B (2022) Hotel booking system for ridel hotel kota bharu. Applied Information Technology and Computer Science 3:912–929