A MAJOR PROJECT REPORT

ON

URBAN CITY EXPLORER

Submitted in partial fulfillment of the Requirements for the award of the degree of

Bachelor of Technology

Computer Science and Engineering

By

J. Charan Reddy	**O180668**
K. Venu Gopal Reddy	**O180669**
M. Anil Kumar	**O180677**

Under the supervision of

Mr. Mallikarjuna Nandi M.E.,

Assistant professor

Department of Computer Science Engineering

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

ONGOLE CAMPUS, PRAKASAM (Dt)

ANDHRA PRADESH -523225

APRIL 2023-24.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

ONGOLE CAMPUS 2024

CERTIFICATE

This is to certify that the report of "**URBAN CITY EXPLORER**" submitted by **J. Charan Reddy** (o180668), **M. Anil Kumar** (o180677), **K. Venu Gopal Reddy** (o180669) is work done by them and during the academic year 2023-24 is a partial fulfilment for the award Undergraduate degree of Bachelor of technology in Computer Science and Engineering, is a bonafide record carried out by them under my supervision. The project has fulfilled all the requirements of this institute and in my opinion, reached the standard of submission.

(Supervisor)	**(Head of the Department)**
Mr. Mallikarjuna Nandi M.E.,	Mr. Mallikarjuna Nandi M.E.,
Assistant Professor	Assistant Professor
Department of CSE	Department of CSE
RGUKT ONGOLE.	**RGUKT ONGOLE.**

APPROVAL SHEET

This report entitled "**URBAN CITY EXPLORER**" by **J. Charan Reddy**(o180668), **M. Anil Kumar** (o180677), **K. Venu Gopal Reddy** (o180669) is **Mr. Mallikarjuna Nandi** approved for the degree of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING.**

Examiner

____

____

____

Supervisor

____

____

____

Date:

Place:

DECLARATION

We hereby declare that this written submission represents our ideas in our own words, and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented, fabricated, or falsified any idea, data, fact, or source in our submission. We understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1)Signature: ___________________

Name: J. Charan Reddy

ID: O180668

2)Signature: ____________________

Name: M. Anil Kumar

ID: O180677

3)Signature: ___________________

Name: K. Venu Gopal Reddy

ID: O180669

ACKNOWLEDGEMENT

First and foremost, we express our sincere gratitude to our project guide, **Mr. Mallikarjuna Nandi,** Department of Computer Science and Engineering, for the constant encouragement during the project. The blessings, help, and guidance provided by them from time to time shall accompany us along the journey of life on which we are about to embark.

We would like to extend our thanks to our Head of the Department, **Mr. Mallikarjuna Nandi,** & for the constructive encouragement during the project.

We are highly indebted to our honorable Director of Ongole RGUKT, **Prof. Bhaskar Patel,** for accepting and providing us with the best opportunity to work.

We are highly indebted to our honorable Dean of Academics RGUKT Ongole, **M. Rupas Kumar** for his encouragement and constant help.

Efforts have been taken in this project. However, it could not have been possible without the kind support and help of many individuals and **RGUKT**. We would like to extend sincere thanks to all of them. Our thanks and appreciation also go to our colleagues in developing the project and people who have willingly helped us out with their abilities.

With sincere regards

J. Charan Reddy

M. Anil Kumar

K. Venu Gopal Reddy

ABSTRACT

"**Urban City Explorer**" is a cutting-edge platform designed to revolutionize the way travelers explore and engage with urban environments. By amalgamating advanced technologies such as React, Spring Boot, and PostgreSQL, the application provides users with a seamless experience in discovering and booking tickets for various attractions within cities globally. Through an intuitive interface and robust search functionality, users can effortlessly navigate through a plethora of options, including museums, monuments, temples, and more.

At the heart of **Urban City Explorer** lies a commitment to user satisfaction and convenience. Leveraging the power of React for the frontend and Spring Boot for the backend, the platform ensures high performance and reliability, catering to the diverse needs of travelers seeking to immerse themselves in urban experiences. With features like advanced search filters, detailed attraction descriptions, and real-time booking capabilities, Urban City Explorer empowers users to make informed decisions and optimize their exploration journeys.

Moreover, **Urban City Explorer** fosters community engagement by allowing users to contribute reviews and recommendations for attractions they have visited. By harnessing the collective wisdom of its user base, the platform creates a dynamic ecosystem where travelers can share insights, discover hidden gems, and connect with like-minded explorers. With its innovative approach and commitment to enhancing the urban exploration experience, Urban City Explorer emerges as a quintessential companion for modern-day travelers seeking to unlock the wonders of urban landscapes.

TABLE OF CONTENTS

CHAPTER 1:

INTRODUCTION

1.1 PURPOSE

Urban City Explorer is designed to address the evolving needs of modern-day travelers by providing a comprehensive platform for exploring urban environments. The purpose of the application is threefold:

1. **Facilitating Exploration:** Urban City Explorer aims to streamline the process of discovering and exploring attractions within cities. By offering a user-friendly interface and advanced search functionalities, the platform empowers travelers to efficiently navigate through a vast array of options, including museums, monuments, temples, and more. Whether users are planning a trip or seeking spontaneous adventures, Urban City Explorer offers the tools and resources to facilitate seamless exploration.

2. **Enabling Informed Decision-Making:** In addition to facilitating discovery, Urban City Explorer serves as a valuable resource for travelers seeking to make informed decisions. Through detailed attraction descriptions, user reviews, and real-time booking capabilities, the platform equips users with the information they need to plan and optimize their urban exploration experiences. By providing transparent and reliable information, Urban City Explorer empowers users to make decisions that align with their preferences and interests.

3. **Fostering Community Engagement:** Beyond its utility as a discovery and planning tool, Urban City Explorer fosters community engagement among travelers. By enabling users to contribute reviews, recommendations, and insights about attractions they have visited, the platform creates a vibrant ecosystem where travelers can share experiences, exchange tips, and connect with fellow explorers. Through community-driven interactions, Urban City Explorer enriches the exploration journey and cultivates a sense of camaraderie among users.

1.2 OBJECTIVE

The purpose of Urban City Explorer is to provide travelers with a sophisticated yet user-friendly platform to explore various urban attractions seamlessly. With an emphasis on simplifying the discovery process, the platform aggregates comprehensive information about museums, monuments, temples, and other points of interest in cities worldwide. By offering detailed descriptions, images, and user reviews, Urban City Explorer empowers travelers to make informed decisions about their exploration activities, ensuring they can maximize their experiences in any city they visit.

The primary objective of Urban City Explorer is to enhance the travel experience by offering a centralized hub for accessing accurate and up-to-date information about urban attractions. Through advanced search functionalities and intuitive navigation features, the platform enables users to effortlessly browse through a diverse array of attractions, discover hidden gems, and plan their itineraries accordingly. Moreover, Urban City Explorer aims to foster a sense of community among travelers by providing a platform for sharing insights, recommendations, and personal experiences, thereby facilitating meaningful interactions and connections between like-minded individuals.

By combining comprehensive data, intuitive design, and community engagement features, Urban City Explorer strives to become the go-to resource for travelers seeking to explore urban destinations worldwide. Whether users are planning a weekend getaway or embarking on a long-term adventure, the platform offers valuable tools and resources to enhance their exploration journey. From discovering iconic landmarks to uncovering off-the-beaten-path treasures, Urban City Explorer empowers travelers to embark on unforgettable urban adventures with confidence and ease.

1.3 MOTIVATION

The motivation behind the creation of Urban City Explorer stemmed from a recognition of the challenges that modern travelers face when exploring urban destinations. Traditional travel guides often provide limited information and can be cumbersome to carry around, while online resources may lack comprehensiveness or reliability. Additionally, the sheer volume of information available on the internet can be overwhelming, making it difficult for travelers to sift through and find relevant details about the places they wish to visit.

Moreover, the increasing popularity of urban travel among tourists and the growing trend of experiential tourism highlighted the demand for a more sophisticated and interactive tool for discovering urban attractions. As travelers seek more immersive and authentic experiences, there is a growing appetite for platforms that go beyond basic listings and provide insights into the cultural significance, historical context, and local flavor of urban destinations. Urban City Explorer aims to meet this demand by offering not only practical information about attractions but also rich storytelling, user-generated content, and community engagement features to enhance the overall travel experience.

1.4 DEFINITION AND OVERVIEW

Urban City Explorer is a web-based application designed to assist travelers in exploring urban destinations efficiently and effectively. It serves as a digital companion for tourists seeking to discover and navigate various attractions, landmarks, and points of interest within cities. The platform offers a curated selection of places to visit, ranging from museums and monuments to parks, markets, and cultural sites, providing users with comprehensive information and insights to enhance their travel experience.

At its core, Urban City Explorer functions as a comprehensive directory and guidebook for urban travelers, offering detailed profiles of each destination, including descriptions, images, location details, operating hours, admission prices, and user reviews. Users can browse through different categories of attractions, filter results based on their preferences, and access personalized recommendations tailored to their interests and travel preferences. Additionally, the platform facilitates seamless trip planning and booking, allowing users to reserve tickets, explore nearby amenities, and create custom itineraries to make the most of their visit.

CHAPTER 2:

ANALYSIS

2.1 Existed System

The Existed System provides an overview of the traditional methods and systems used by travelers to explore urban destinations before the advent of Urban City Explorer. This section highlights the limitations and challenges associated with conventional travel planning approaches, emphasizing the need for a more efficient and user-friendly solution.

In the past, travelers relied heavily on guidebooks, brochures, and maps to navigate cities and discover points of interest. While these resources offered valuable insights into popular attractions and landmarks, they often lacked real-time updates, personalized recommendations, and interactive features. As a result, travelers faced difficulties in accessing up-to-date information, planning customized itineraries, and discovering off-the-beaten-path destinations.

Furthermore, traditional travel planning methods were time-consuming and labor-intensive, requiring travelers to conduct extensive research, cross-reference multiple sources, and manually organize their trip details. This fragmented approach often led to information overload, decision fatigue, and missed opportunities to explore lesser-known attractions. Additionally, travelers encountered challenges in booking tickets, finding accommodation, and coordinating transportation, further complicating the trip planning process.

Overall, the existing system lacked the convenience, accessibility, and interactivity needed to support modern travelers' needs and preferences. With the emergence of Urban City Explorer, there is an opportunity to revolutionize the way travelers explore urban destinations, offering a comprehensive and user-centric platform that addresses the shortcomings of traditional travel planning methods. By leveraging technology to provide real-time updates, personalized recommendations, and seamless booking functionalities, Urban City Explorer aims to enhance the travel experience and empower users to make the most of their urban adventures.

2.2 Proposed System

The Proposed System outlines the key features, functionalities, and benefits of Urban City Explorer, the proposed solution for modern urban exploration. This section highlights how the platform aims to address the limitations of traditional travel planning methods and improve the overall travel experience for users.

Urban City Explorer is a comprehensive and user-friendly platform designed to facilitate urban exploration and enhance the travel planning process. The proposed system leverages advanced technologies such as data analytics, machine learning, and geolocation services to provide users with personalized recommendations, real-time updates, and interactive mapping features.

One of the main features of Urban City Explorer is its extensive database of points of interest, including museums, monuments, landmarks, restaurants, and more. Users can browse through curated lists, explore interactive maps, and discover hidden gems in their chosen city. The platform also offers detailed information about each location, including descriptions, reviews, ratings, and photographs, allowing users to make informed decisions about their itinerary.

Another key aspect of the proposed system is its booking functionality, which enables users to seamlessly reserve tickets, tours, and activities directly through the platform. By partnering with local vendors and attractions, Urban City Explorer offers exclusive deals, discounts, and special offers to enhance the value proposition for users.

Overall, Urban City Explorer aims to revolutionize the way travelers explore urban destinations by providing a one-stop solution for trip planning, booking, and exploration. By combining comprehensive data, intuitive design, and seamless integration with third-party services, the proposed system offers a convenient and efficient platform for modern travelers to discover and experience the best that cities have to offer.

2.3 Future Scope

The Future Scope outlines potential enhancements, expansions, and future development paths for Urban City Explorer. It provides insights into how the platform can evolve over time to meet changing user needs, technological advancements, and market demands.

1. **Enhanced Personalization:** In the future, Urban City Explorer could incorporate advanced personalization features to tailor recommendations and suggestions even more closely to each user's preferences, interests, and past behaviors. This could include machine learning algorithms that analyze user interactions, feedback, and historical data to provide increasingly relevant and customized recommendations.
2. **Integration with Augmented Reality (AR):** As AR technology continues to mature, integrating AR functionalities into Urban City Explorer could enhance the user experience by overlaying digital information, directions, and interactive elements onto the physical environment. This could enable users to visualize points of interest, navigate city streets, and access additional contextual information in real-time.
3. **Community Engagement and User-Generated Content:** To foster community engagement and enhance the platform's content richness, future iterations of Urban City Explorer could incorporate features that allow users to contribute their own reviews, recommendations, and photos. User-generated content could complement existing data sources and provide valuable insights for other travelers seeking authentic experiences and insider tips.
4. **Expansion to New Cities and Regions:** As Urban City Explorer gains popularity and traction, there is a significant opportunity to expand its coverage to new cities, regions, and destinations around the world. By continuously adding new locations and updating existing content, the platform can become an indispensable tool for travelers seeking to explore urban areas globally.
5. **Integration with Smart City Initiatives:** As cities become increasingly connected and technologically advanced, Urban City Explorer could integrate with smart city initiatives to provide users with real-time information on public transportation, traffic conditions, event schedules, and safety alerts. By leveraging IoT sensors, open data platforms, and municipal APIs, the platform can offer valuable insights and services that enhance the overall urban experience.

2.4 Project Requirements

2.4.1. Software Requirements

- JDK
- IntelliJ
- Spring Boot
- Postgres SQL
- Microservices
- React Js

2.4.2 Hardware Requirements

- CPU: Intel Core i3 or higher
- RAM: 4GB or higher
- Storage: 10GB or higher
- Operating system: Windows 10, macOS 10.15, or Linux

CHAPTER 3:
LITERATURE REVIEW

The literature review provides a comprehensive understanding of the existing body of knowledge related to the domain of the current project. It involves examining relevant research papers, articles, books, and other scholarly sources to identify key concepts, theories, methodologies, and findings that contribute to the project's context and objectives.

In the context of this project, the literature review focuses on several key areas:

- Tourism Management Systems: Previous studies and research papers related to tourism management systems offer insights into the features, functionalities, and challenges associated with such systems. Understanding existing solutions in this domain helps in identifying common requirements, user expectations, and industry standards.
- Online Ticket Booking Platforms: Literature on online ticket booking platforms provides valuable insights into the design, implementation, and user experience aspects of ticket reservation systems. By reviewing existing platforms, the project can learn from best practices, user interface designs, and backend architectures to enhance its own booking functionalities.
- Location-Based Services (LBS): Research on location-based services explores the use of geographic information systems (GIS), global positioning systems (GPS), and mobile technologies in delivering personalized and context-aware services to users. Understanding LBS principles and applications is crucial for developing location-based features in the project, such as geo-tagging, navigation, and proximity-based recommendations.
- Technological Frameworks and Tools: Literature review also includes an examination of relevant technological frameworks, programming languages, development tools, and platforms used in similar projects. This helps in selecting appropriate technologies for implementing the project, considering factors such as scalability, performance, security, and ease of maintenance.
- User Experience (UX) Design: Studies on user experience design principles provide guidance on creating intuitive, user-friendly interfaces for the project's web and mobile applications. By incorporating UX best practices, the project aims to enhance user satisfaction, engagement, and retention.

CHAPTER 4:

DESIGN

4.1 UML Diagrams

In software engineering, a class diagram in the Unified Modeling Language is **a type of static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

Sequence Diagram

A sequence diagram is a type of interaction diagram that shows how processes operate with one another and the order in which they occur. It illustrates the dynamic behavior of the system, emphasizing the sequence of messages exchanged between the different components or objects within the system.

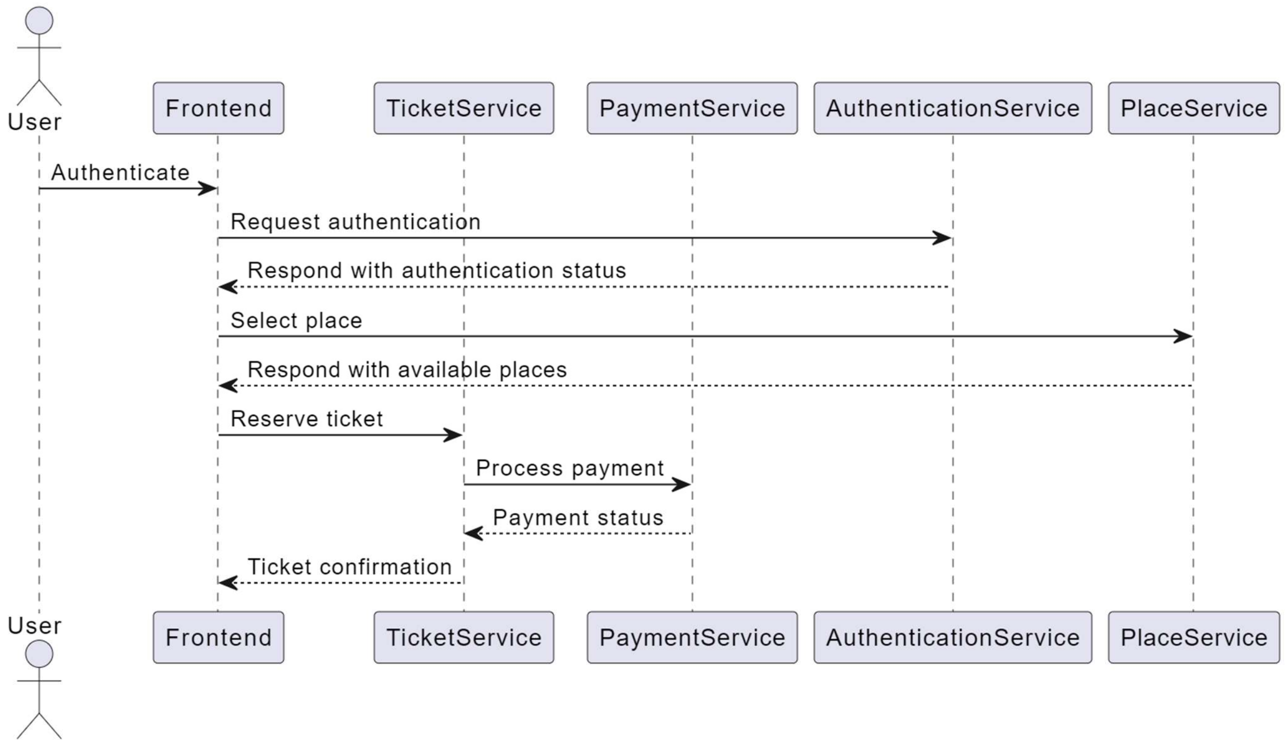


Fig.4.1 Sequence Diagram

Use Case Diagram:

Use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior. Use cases once specified can be denoted both textual and visual representation. A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

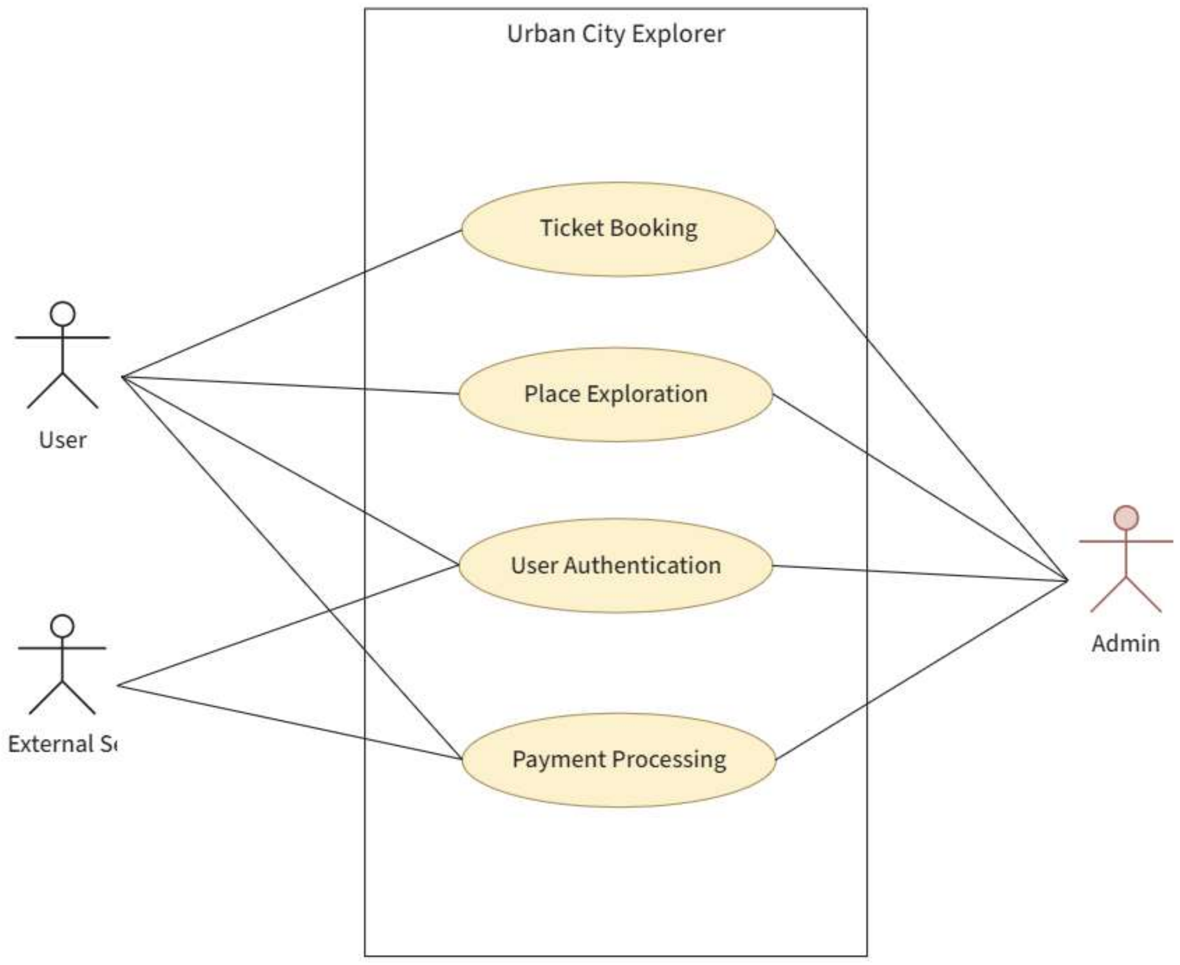


Fig.4.2 Use Case Diagram

Activity Diagram

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.

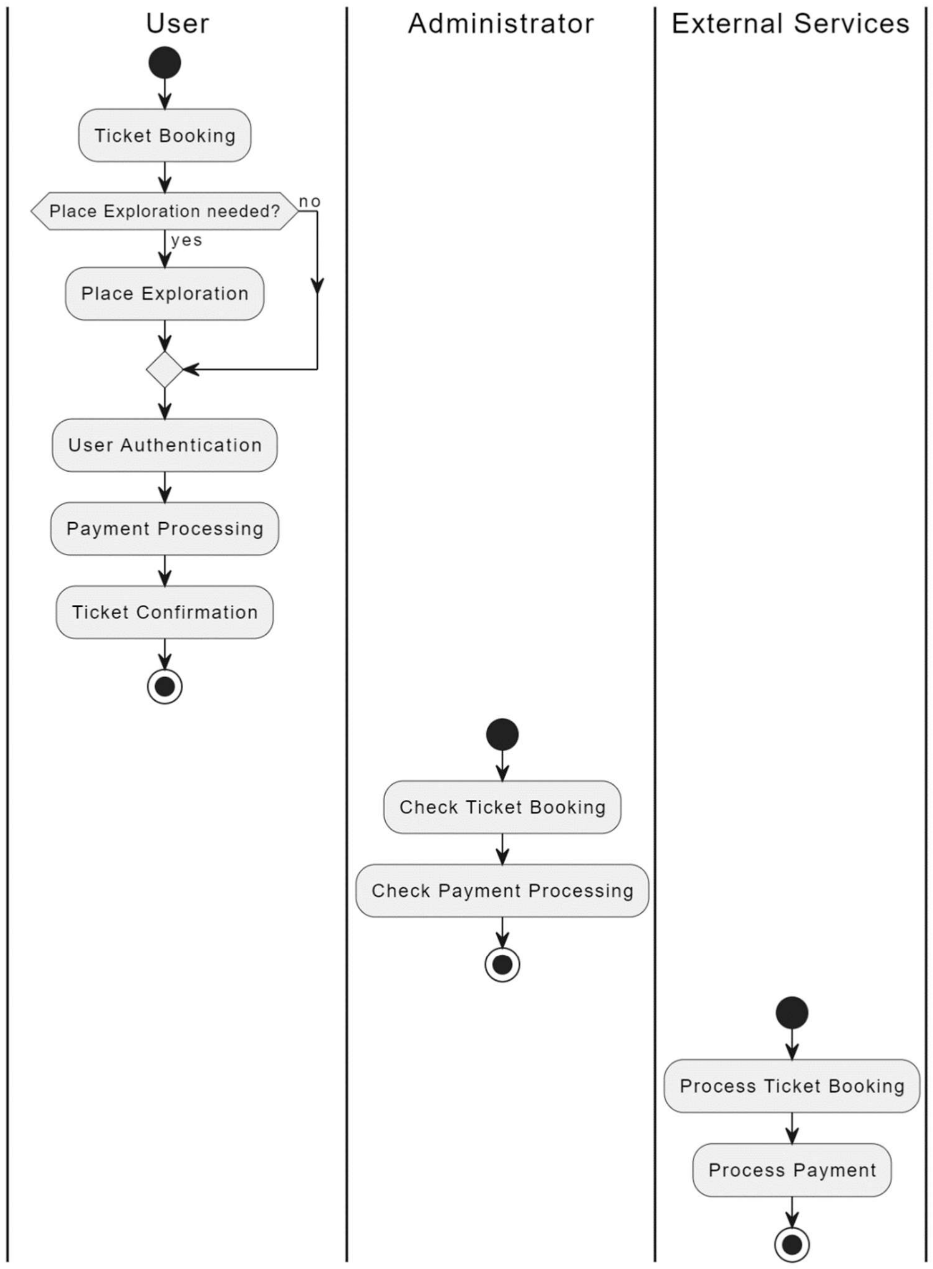


Fig.4.3 Activity Diagram

Class Diagram

Class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

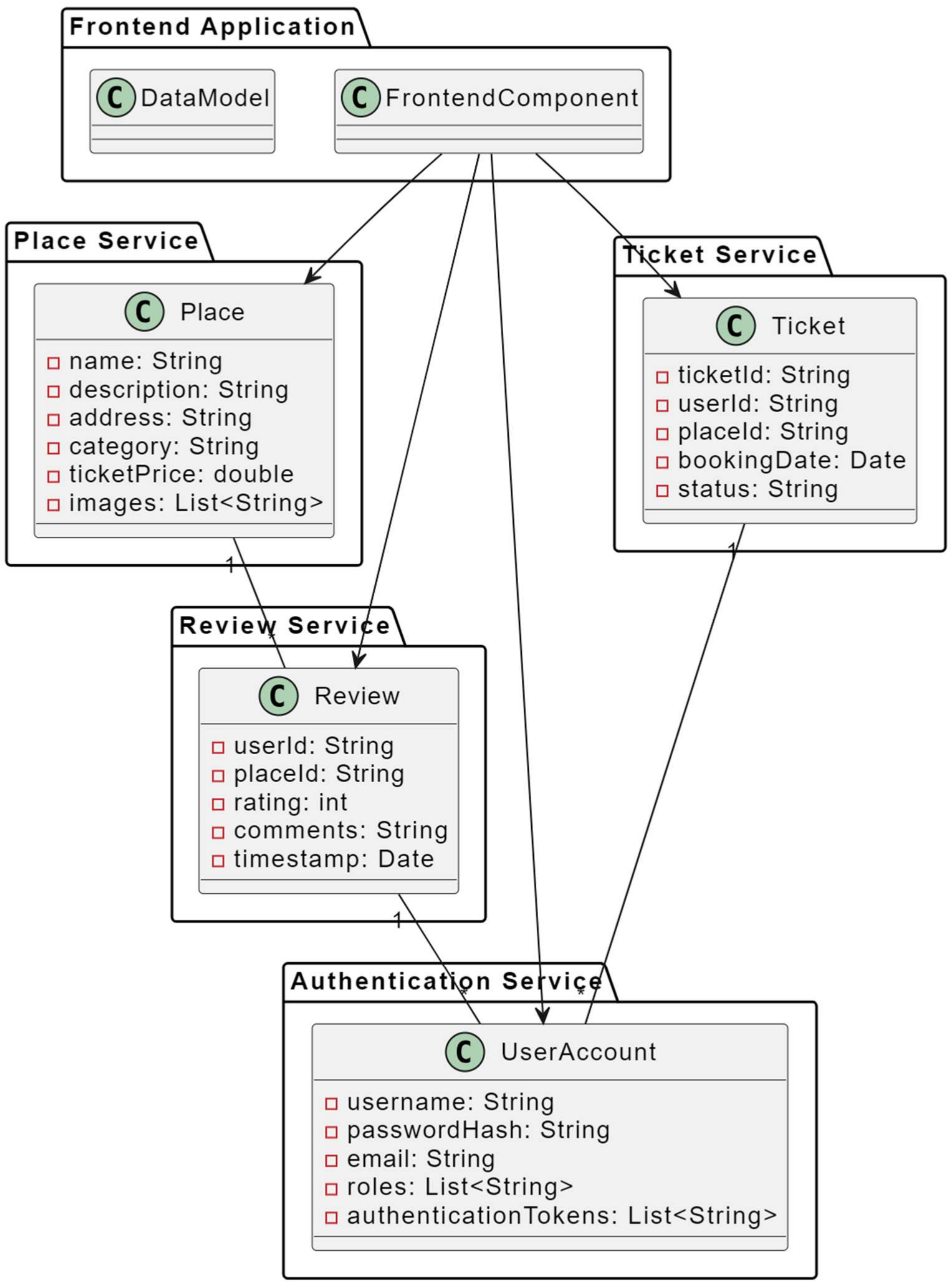


Fig.4.4 Class Diagram

CHAPTER 5:

IMPLEMENTATION

5.1 Modules

- Api Gateway
- Auth Service
- Email Service
- Frontend
- Payment Service
- Place Service
- Question and Answer Service
- Review Service
- Service Registry
- Ticket Service

5.2 Module Description

1. **API Gateway**:
   The API Gateway serves as the single entry point for all client requests to the backend microservices. It handles request routing, authentication, and authorization, ensuring secure and controlled access to the microservices. Additionally, it often provides features like rate limiting, request/response transformation, and logging for monitoring and analytics purposes.

2. **Auth Service**:
   The Auth Service is responsible for user authentication and authorization within the system. It manages user registration, login, logout, and password management functionalities. This module generates and verifies authentication tokens, ensuring that only authenticated users can access protected resources across the application.

3. **Email Service**:
   The Email Service handles the sending of email notifications to users for various events and actions within the application. It integrates with email delivery services or SMTP servers to send account-related emails such as account verification, password reset, account activation, and notification alerts, enhancing user engagement and communication.

4. **Frontend:**
   The Frontend module encompasses the user interface components of the application, built using React.js or similar frontend frameworks. It interacts with users, presenting the application's features and functionalities in an intuitive and user-friendly manner. The frontend consumes backend APIs provided by the microservices to fetch and display data, enabling seamless user interaction and experience.

5. **Payment Service**:
   The Payment Service manages payment processing and transactions within the application. It integrates with third-party payment gateways or payment processors to securely handle payments for various services offered by the application. This module ensures smooth and secure payment processing, enhancing user satisfaction and facilitating monetization strategies.

6. **Place Service**:
   The Place Service handles the management of places of interest, such as museums, monuments, temples, and other attractions within the application. It provides CRUD (Create, Read, Update, Delete) operations for managing place information, including descriptions, images, reviews, and pricing. Additionally, it may offer search functionalities to enable users to discover and explore different places.

7. **Question and Answer Service**:
   The Question and Answer Service implements a platform where users can ask questions, provide answers, and engage in discussions related to various topics within the application. It facilitates community-driven interactions, fostering knowledge sharing and collaboration

among users. This module may include features such as upvoting/downvoting, comments, and moderation tools to maintain quality content.

8. **Review Service**:
   The Review Service manages user reviews and ratings for places, events, or other entities within the application. It allows users to submit reviews, rate their experiences, and read reviews from other users. This module plays a crucial role in influencing user decisions, providing valuable feedback, and maintaining transparency and trust within the community.
9. **Service Registry**:
   The Service Registry acts as a centralized registry or directory of all microservices deployed in the application architecture. It keeps track of service instances, their locations, and endpoints, enabling dynamic service discovery and load balancing. This module enhances scalability, fault tolerance, and resilience by facilitating efficient communication and coordination among microservices.
10. **Ticket Service:**
    The Ticket Service handles the booking and management of tickets for events, places, or activities offered by the application. It allows users to browse available tickets, select desired options, make reservations, and receive confirmation details. This module may include features such as seat selection, pricing tiers, and ticket validation, ensuring a seamless booking experience for users.

5.3 Introduction to Technologies Used:

In the development of our project, we have employed a range of modern technologies and frameworks to build a robust and scalable system. Each technology was carefully selected to address specific requirements and to ensure efficient implementation of the project's features. Here's an overview of the key technologies used:

1. **React.js:**
   We chose React.js as the frontend framework for building our user interface. React.js offers a component-based architecture, making it easy to create reusable UI components and manage state efficiently. Its virtual DOM implementation ensures fast rendering, resulting in a responsive and interactive user experience.

2. **Spring Boot:**

   Spring Boot is the backbone of our backend services, providing a powerful and opinionated framework for building Java-based microservices. With Spring Boot, we benefit from auto-configuration, dependency injection, and seamless integration with other Spring projects, allowing us to rapidly develop and deploy RESTful APIs.

3. **PostgreSQL:**

   PostgreSQL serves as the relational database management system (RDBMS) for storing and managing our application's data. Known for its robustness, extensibility, and compliance with SQL standards, PostgreSQL offers advanced features such as ACID transactions, JSONB data type support, and full-text search capabilities, ensuring data integrity and flexibility.

4. **Resilience4j:**

   Resilience4j is a lightweight fault tolerance library for Java-based applications, offering resilience patterns such as circuit breakers, retries, and rate limiters. We use Resilience4j to enhance the resilience and stability of our microservices, ensuring graceful degradation and fault tolerance in the face of network failures or system disruptions.

5. **Spring Cloud Netflix Eureka:**

   Spring Cloud Netflix Eureka provides service discovery and registration capabilities for our microservices architecture. By leveraging Eureka's server-client model, we achieve dynamic service discovery, load balancing, and failover, enabling seamless communication and coordination among distributed services.

6. **JWT (JSON Web Tokens):**

   JWT is a compact and self-contained mechanism for securely transmitting information between parties as a JSON object. We utilize JWT as the authentication mechanism in our system, enabling stateless and token-based authentication for user access to protected resources.

7. **Axios**:

   Axios is a popular HTTP client library for making asynchronous HTTP requests in JavaScript applications. We use Axios in our React.js frontend to communicate with backend APIs, fetching and sending data between the client and server asynchronously, and handling HTTP response handling and error management.

8. **AWS (Amazon Web Services):**

   AWS provides a reliable and scalable cloud computing platform, offering a wide range of services such as compute, storage, database, and networking. We leverage AWS services, such as EC2 (Elastic Compute Cloud) for hosting, RDS (Relational Database Service) for database management, and S3 (Simple Storage Service) for storing static assets and media files.

By leveraging these technologies and frameworks, we have built a modern and resilient application architecture that delivers a seamless and engaging user experience while ensuring reliability, scalability, and security.

5.4 Sample Code

```
import React from 'react'
import './Advertisement.css'

function Advertisement() {
  return (
    <div className='adContainer'>
        <img className='adImage' src='https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcRRxLIYALSkHRwmyC7MzogMfroysOXaaxhK5w&usqp=C
AU' />
        <span className='adText'>ENDLESS TICKET ANYTIME. ANYWHERE</span>
    </div>
  )
}

export default Advertisement
```

```
function BookingForm() {

  const location = useLocation();
  const [date, setDate] = useState(new Date());
  const [price, setPrice] = useState(location.state["price"]);
  const [quantity, setQuantity] = useState(1);
  const [email, setEmail] = useState(localStorage.getItem('email'));
  const {placeName} = useParams();
  const navigate = useNavigate();

  const decrease = () => {
    if(quantity > 1){
        setQuantity(quantity-1);
    }else{
        toast.error("Reached Minimum Quantity");
    }
  }

  const increase = () =>{
    setQuantity(quantity+1);
  }

  const createOrder = (e) =>{
    e.preventDefault();
    console.log("order clicked");
    const token = localStorage.getItem('token')
    if(token != null){
        const obj = {};
        obj["email"] = email;
        obj["price"] = price;
        obj["quantity"] = quantity;
        if(location.state["type"] === "place"){
            obj["placeId"] = location.state["id"];
            obj["bookedDates"] = [date];
        }else{
            obj["eventId"] = location.state["id"];
            obj["bookedDates"] = [location.state["date"]]
        }
        obj["placeName"] = placeName

        const url = process.env.REACT_APP_ROOT_URL + "/ticket/buy";
        axios.post(url,obj,{
            headers: {
                'Content-Type' : 'application/json',
                'Authorization' : `Bearer ${token}`
            }
        }
        ).then((res) => {
            console.log(res);
            if(res.status === 200){
                navigate(`/${res.data.ticketId}/payment`);
            }
        }).catch((error) => {
            console.log(error);
            toast.error(error.response.data.message);
            if(error.response.data.status == 500){
                navigate("/sign-in")
            }
        })
    }else{
        navigate("/sign-in")
    }
  }

  return (
    <div className='bookingContainer'>
        <ToastContainer/>
        <form className="beautiful-form">
            <h2>Order Form</h2>
            <div className="form-group">
                <label for="email">Email:</label>
                <input disabled value={email} className="bookingFormInput" type="text" id="quantity"
name="quantity" required/>
            </div>
            <div className="form-group">
                <label for="quantity">Quantity:</label>
                <div className='quantityItem'>
                    <div><FontAwesomeIcon style={{height:"25px", width:"25px",
marginRight:"10px",cursor:"pointer"}} onClick={() => decrease()} icon={faCircleMinus} /></div>
                    <div>{quantity}</div>
                    <div><FontAwesomeIcon style={{cursor:"pointer", marginLeft:"10px", height:"25px",
width:"25px"}} onClick={() => {increase()}} icon={faCirclePlus} /></div>
```

```
            </div>
          </div>
          <div className="form-group">
            <label for="price">Price:</label>
            <input disabled value={price*quantity} className="bookingFormInput" type="number"
id="price" name="price" step="0.01" required/>
          </div>
          <div className="form-group">
            <label for="date">Date:</label>
            {location.state["type"] == "place" ? <DatePicker minDate={new Date()}
className='bookingFormInput' selected={date} onChange={(currentDate) => setDate(currentDate)}
/> : <DatePicker className='bookingFormInput' selected={location.state["date"]} disabled/>}
          </div>
          <button className="submitButton" onClick={(e) => {createOrder(e)}}>Submit</button>
        </form>
    </div>
  )
}

export default BookingForm


function Card({place}) {

  const navigate = useNavigate();

  return (
    <div onClick={() => {navigate(`/${place.city}/place/${place.placeName}`,{state:place.id})}}
className='card'>
        <img className="cardImage" src={place.mainImage}/>
        <div className='cardContent'>
          <span className='cardRating'>{place.rating==null ? 4.5 :
Math.round(place.rating*10)/10}</span>
          <span className='cardPlace'>{place.placeName}</span>
          <span className='cardCategory'>{place.category}</span>
        </div>
    </div>
  )
}

export default Card
```

```
function MyCarousel() {
  return (
    <Carousel className="carousel" autoPlay={true}
      infiniteLoop={true}
      interval={3000}
      stopOnHover={true}
      dynamicHeight={false}
      width="100%"
      centerSlidePercentage={80}
```

CHAPTER 6:
SYSTEM FEATURES

1. **User Authentication and Authorization:**
   The system allows users to securely authenticate themselves using credentials and provides role-based access control to authorize access to different features based on user roles such as admin, customer, and guest.

2. **Explore Places**:
   Users can explore various places such as museums, monuments, temples, and other attractions within a city. They can view detailed information about each place, including descriptions, images, reviews, and ratings, to make informed decisions about visiting.

3. **Booking Tickets**:
   The system enables users to book tickets for visiting places directly through the application. Users can select a place, choose the desired date and time, and purchase tickets securely online, eliminating the need to wait in queues or visit multiple websites.

4. **Booking Tickets**:
   The system enables users to book tickets for visiting places directly through the application. Users can select a place, choose the desired date and time, and purchase tickets securely online, eliminating the need to wait in queues or visit multiple websites.

5. **Booking Tickets**:
   The system enables users to book tickets for visiting places directly through the application. Users can select a place, choose the desired date and time, and purchase tickets securely online, eliminating the need to wait in queues or visit multiple websites.

6. **Review and Rating**:
   Users can leave reviews and ratings for places they have visited, sharing their experiences and feedback with other users. The review feature helps users make better decisions and provides valuable insights for improving the quality of services at each place.

7. **Search Functionality**:
   The system offers a robust search functionality that allows users to search for specific places based on keywords, categories, or locations. Users can quickly find places of interest using filters and sorting options, enhancing their browsing experience.

8. **Event Management**:
   The system includes a feature for managing events such as exhibitions, concerts, and festivals happening in the city. Users can view event details, purchase tickets, and stay updated on upcoming events, enriching their cultural and social experiences.

9. **Secure Payment Gateway**:
   To facilitate online transactions, the system integrates with a secure payment gateway that supports various payment methods such as credit/debit cards, net banking, and digital wallets. Users can complete transactions confidently, knowing that their payment information is protected.

10. **Responsive Design:**
    The system is designed with a responsive and user-friendly interface that adapts seamlessly to different devices and screen sizes. Whether users access the application from a desktop, tablet, or mobile device, they can enjoy a consistent and intuitive user experience.

CHAPTER 7:

NON-FUNCTIONAL FEATURES

A careful specification and adherence of non-functional requirements such as performance, security, privacy and availability are crucial to the success or failure of any software system.

7.1 Performance Requirements

- The capability of the application depends on the performance of the servers. Anyone can use the application easily because of good GUI.
- On mobile devices and laptops, the battery is a scarce and valuable resource. The battery should remain maximally available for the application to perform well. Your application may therefore fall by the wayside or even get uninstalled by the user, if it drains too much battery.
- The text font size may need to be adjusted up (for high resolution screens) or down (for low resolution screens) so as to keep the text readable.

7.2 Safety Requirements

- The layout may need to be taken care of and adjusted to increase or decrease the spacing between and around labels and widgets shown on the screen so as to prevent them from getting clustered together on high-res screens or spaced apart too much on low-res screens.
- Any images used in the project must be provided in two different versions: a large size/high resolution version and a small size/low resolution version so that it properly fills the amount of physical space available on the screen.

7.3 Security Requirements

- Although security is the utmost priority and has been taken care of the most, care must be taken against virus and malware threats.
- This application will be available for all the users of the Internet. The system server should be up for 365 days (about 12 months), and the downtime should be minimized in case of any attack or difficulties.
- Firewall should be used on the user's system to prevent any suspicious activity.

7.4 Software Quality Attributes

- 24x7 availability of the system with suitable updating at regular intervals of time. To maintain integrity of the data and to ensure the security of the database by asking them to sign up for the application.
- Form validation so that only real users access the system. An error message should be displayed in case of improper working of the application.
- Email -ID entered should be valid as OTP is sent to that Email ID.
- The application can be accessed at any place that has Internet connectivity.
- Always save the data before closing the website.
- An error message should be displayed in case of improper working of the application.
- 24-hour availability of internet connection is required.

CHAPTER 8:

TEST CASES

1. **User Authentication:**
    a. Test Case 1: Verify that a registered user can log in successfully with valid credentials.
    b. Test Case 2: Verify that a user cannot log in with invalid credentials and receives an appropriate error message.
    c. Test Case 3: Verify that a user can reset their password using the forgot password functionality.

2. **Explore Places:**
    a. Test Case 4: Verify that the user can view a list of places in the city with relevant details such as name, description, and images.
    b. Test Case 5: Verify that the user can filter places by categories such as museums, monuments, and temples.
    c. Test Case 6: Verify that the user can click on a place to view detailed information including reviews and ratings.

3. **Booking Tickets:**
    a. Test Case 7: Verify that the user can select a place and book tickets for a specific date and time.
    b. Test Case 8: Verify that the user receives a booking confirmation after successfully completing a ticket purchase.
    c. Test Case 9: Verify that the user can cancel a ticket booking before the scheduled date and time.

4. **Review and Rating:**
    a. Test Case 10: Verify that the user can leave a review and rating for a place after visiting it.
    b. Test Case 11: Verify that the average rating for a place is calculated correctly based on user reviews.
    c. Test Case 12: Verify that the user can edit or delete their own reviews, but not those of other users.

5. **Search Functionality:**
    a. Test Case 13: Verify that the user can search for places by entering keywords in the search bar.
    b. Test Case 14: Verify that the user can filter search results by categories, locations, or other parameters.
    c. Test Case 15: Verify that the search results are displayed accurately and relevant to the user's query.

6. **Event Management:**
    a. Test Case 16: Verify that the user can view a list of upcoming events with details such as name, description, and date.
    b. Test Case 17: Verify that the user can purchase tickets for an event and receives a booking confirmation.
    c. Test Case 18: Verify that the user can view past events and access details such as attendance and feedback.

7. **Secure Payment Gateway:**
    a. Test Case 19: Verify that the user can select a payment method and securely enter payment details.

b. Test Case 20: Verify that the payment gateway processes transactions accurately and sends payment confirmations to users.

8. **Responsive Design:**
    a. Test Case 21: Verify that the application's interface adapts smoothly to different screen sizes and devices.
    b. Test Case 22: Verify that all features and functionalities are accessible and usable on desktop, tablet, and mobile platforms.

9. **Admin Dashboard:**
    a. Test Case 23: Verify that administrators can log in to the admin dashboard and access administrative functionalities.
    b. Test Case 24: Verify that administrators can add, edit, or delete places, events, and user accounts as needed.
    c. Test Case 25: Verify that administrators can generate reports, monitor system activities, and perform other administrative tasks efficiently.

10. **Notification System:**
    a. Test Case 26: Verify that users receive timely notifications about upcoming events, booking confirmations, and important updates.
    b. Test Case 27: Verify that notifications are delivered reliably through various channels such as email, SMS, and in-app notifications.

CHAPTER 9: RESULTS

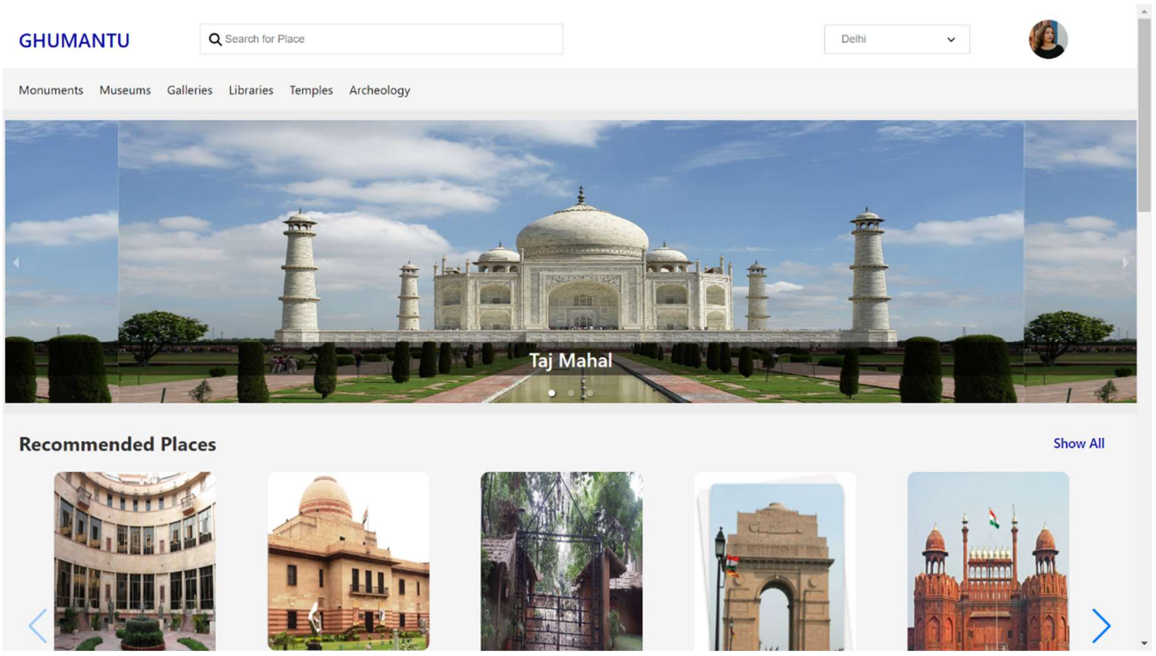


Fig 9.1 Home Page

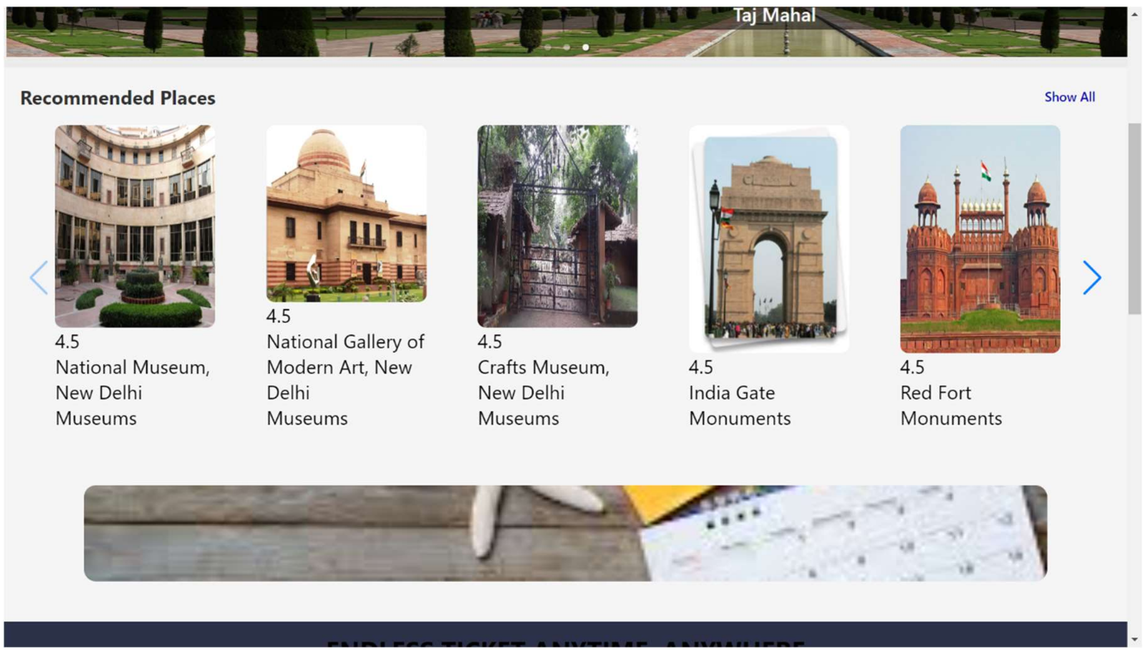
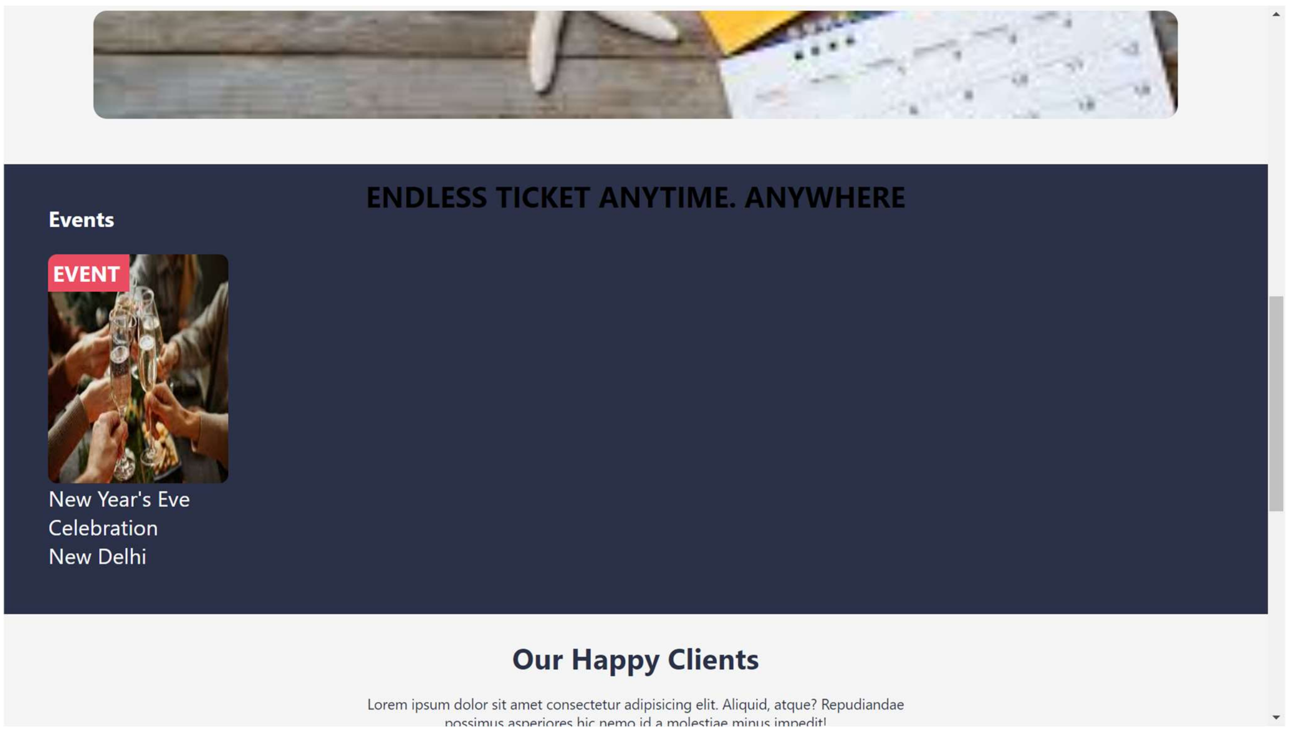


Fig 9.2 Home Page


ENDLESS TICKET ANYTIME. ANYWHERE

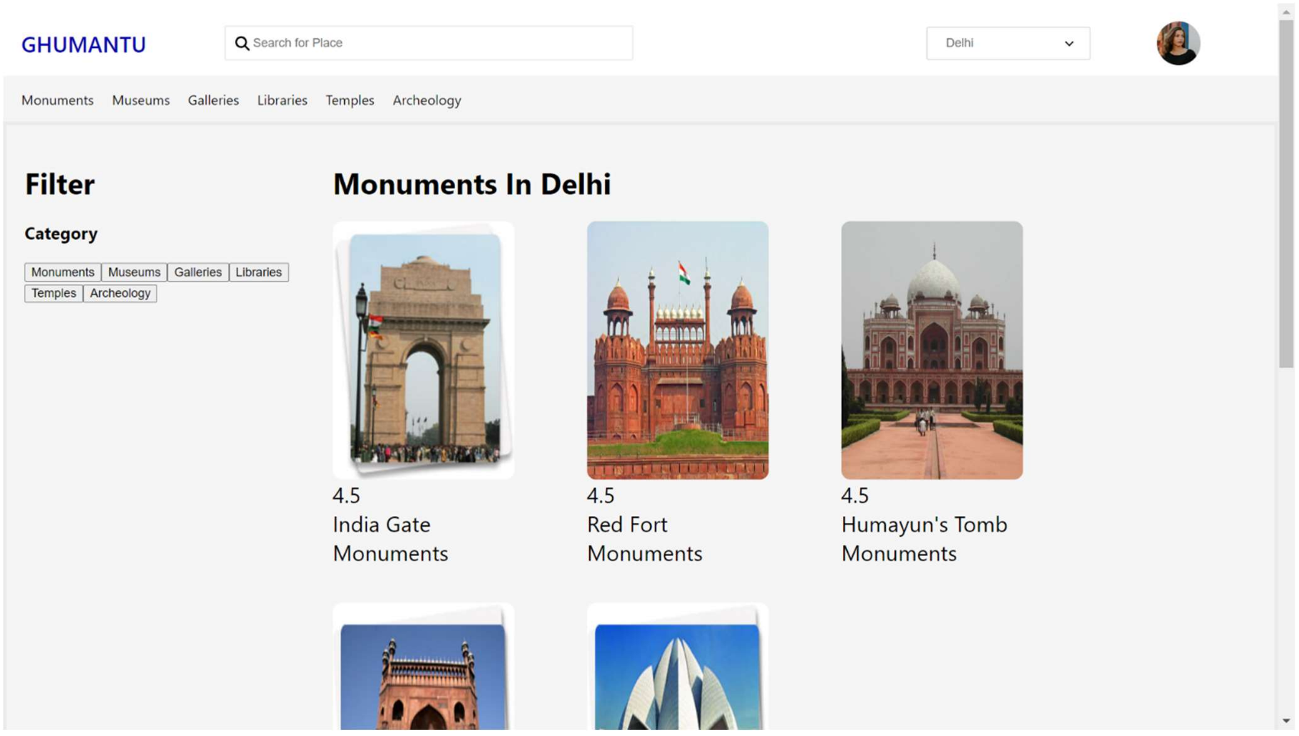
Events

New Year's Eve Celebration
New Delhi

Our Happy Clients

Lorem ipsum dolor sit amet consectetur adipisicing elit. Aliquid, atque? Repudiandae possimus asperiores hic nemo id a molestiae minus impedit!

Fig 9.3 Events



Filter

Category

Monuments In Delhi

4.5
India Gate
Monuments

4.5
Red Fort
Monuments

4.5
Humayun's Tomb
Monuments

Fig 9.4 Places

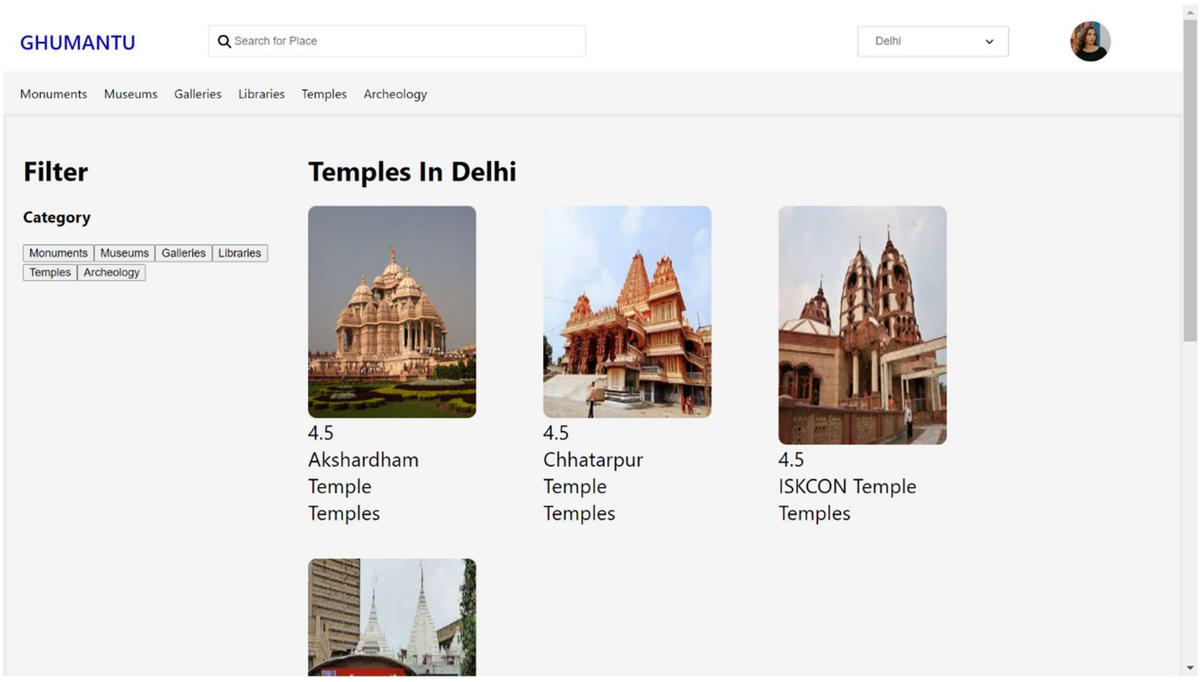
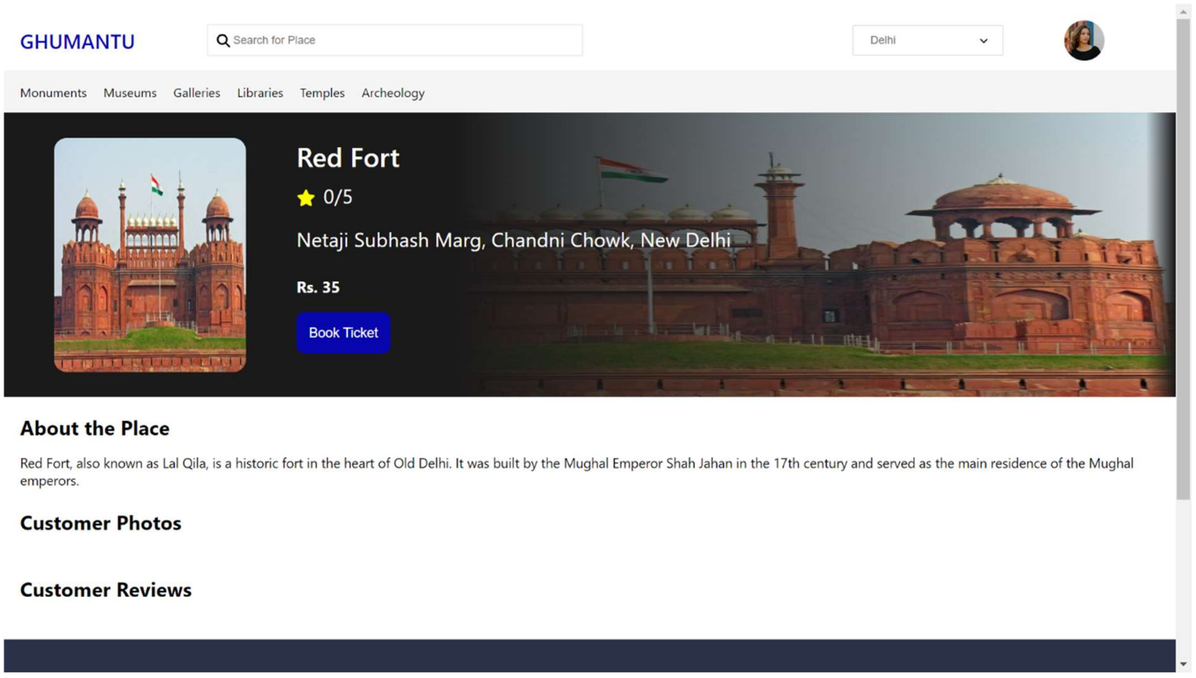
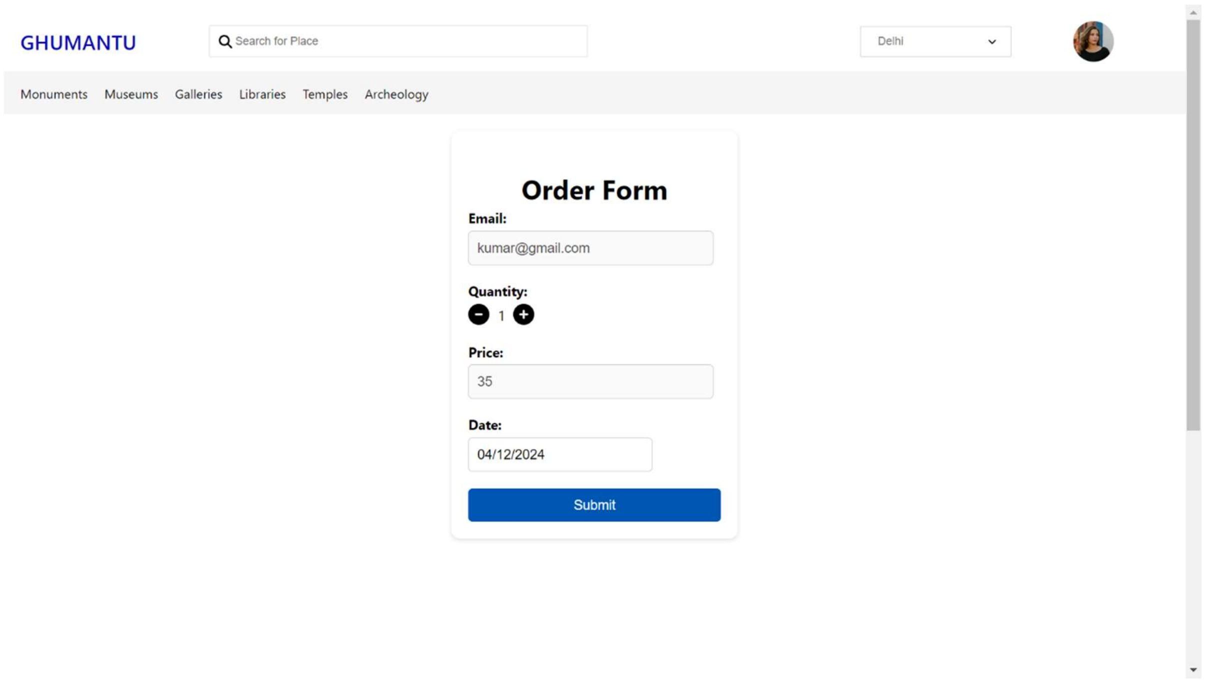


Fig 9.5 Temples



Red Fort, also known as Lal Qila, is a historic fort in the heart of Old Delhi. It was built by the Mughal Emperor Shah Jahan in the 17th century and served as the main residence of the Mughal emperors.

Fig 9.6 Red Fort


GHUMANTU

Search for Place

Delhi

Monuments Museums Galleries Libraries Temples Archeology

Order Form

Email: kumar@gmail.com

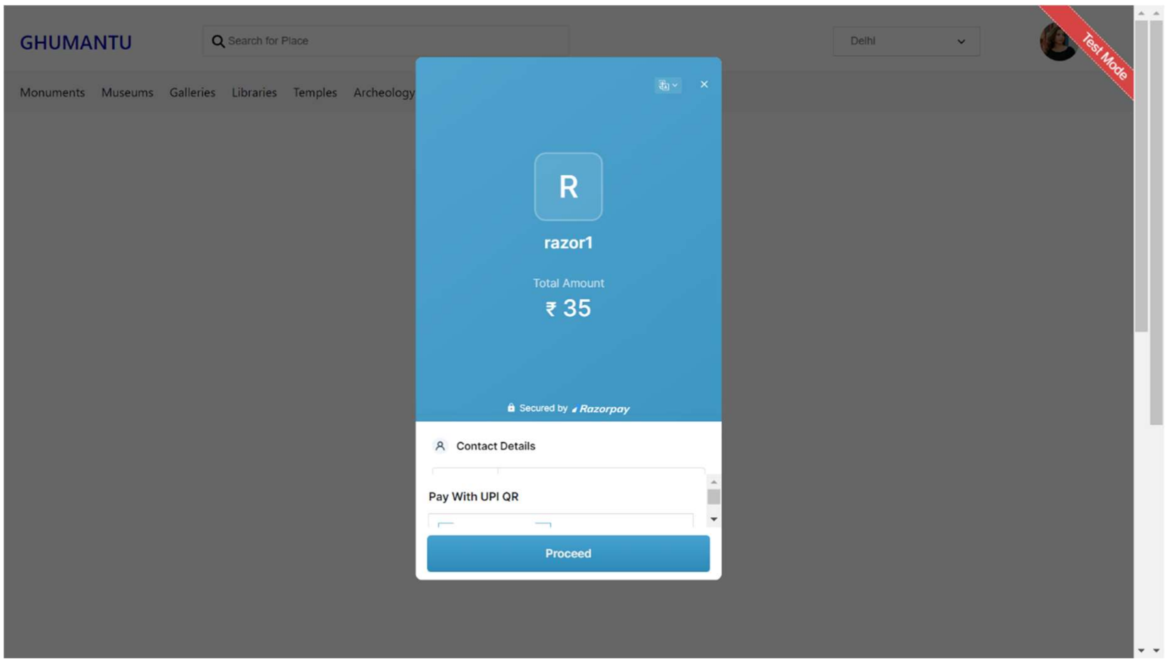
Quantity: 1

Price: 35

Date: 04/12/2024

Submit

Fig 9.7 Order Form


GHUMANTU

Search for Place

Delhi

Monuments Museums Galleries Libraries Temples Archeology

Test Mode

R

razor1

Total Amount

₹ 35

Secured by Razorpay

Contact Details

Pay With UPI QR

Proceed

Fig 9.8 Payment section

CHAPTER 10:

CONCLUSION

CONCLUSION

In conclusion, this project aims to provide users with a comprehensive platform for exploring and experiencing the rich cultural heritage and attractions of cities. By leveraging modern technologies such as microservices architecture, cloud computing, and secure payment gateways, we have developed a robust and user-friendly application that facilitates seamless exploration, booking, and engagement with various places of interest and events.

Through extensive research, analysis, and collaboration, we have successfully implemented a range of features and functionalities that cater to the diverse needs and preferences of users, including user authentication, place exploration, ticket booking, review and rating, search functionality, event management, secure payment processing, and responsive design.

Overall, this project represents a significant step towards enhancing the tourism and cultural experience for travelers and locals alike, promoting community engagement, economic growth, and cultural exchange. We remain committed to continuously improving and expanding our platform to deliver greater value and satisfaction to our users and stakeholders, ensuring that it remains a top choice for urban exploration and discovery.

REFERENCES

[1] Flask Documentation. (https://flask.palletsprojects.com/en/2.0.x/)

[2] Spring Boot Documentation. (https://docs.spring.io/spring-boot/docs/current/reference/html/)

[3] React Documentation. (https://reactjs.org/docs/getting-started.html)

[4] PostgreSQL Documentation. (https://www.postgresql.org/docs/)

[5] Resilience4j Documentation. (https://resilience4j.readme.io/docs)

[6] AWS Documentation. (https://docs.aws.amazon.com/index.html)

[7] JSON Web Token (JWT) Documentation. (https://jwt.io/introduction/)

[8] OpenAPI Specification (OAS) Documentation. (https://swagger.io/docs/specification/about/)

[9] Spring Cloud Netflix Documentation. (https://spring.io/projects/spring-cloud-netflix)

[10] Microservices Architecture: Patterns and Practices for Building Microservices. (https://microservices.io/index.html)

These references were instrumental in the development of the project, providing guidance, documentation, and best practices for various technologies and architectural patterns utilized in the system.