smart parking MANAGEMENT APPLICATION

by

<Student’s full official name, previous degree(s) abbreviated>

Presented to the Faculty of

<Name of University>

In Partial Fulfillment

Of the Requirements

For the Degree

<NAME OF DEGREE>

THE UNIVERSITY OF <UNIV Name>

<MONTH OF GRADUATION, YEAR>

< Leave this page blank >

SMART PARKING MANAGEMENT APPLICATION

by

<Student’s full official name>

APPROVED BY

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Chair’s Name, Degree>, Chair

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Member’s Name, Degree >, Committee Member

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Member’s Name, Degree >, Committee Member

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Member’s Name, Degree >, Committee Member

RECEIVED/APPROVED BY THE COLLEGE OF <COLLEGE NAME>:

<Associate Dean’s Name, Degree>, Associate Dean

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Dean’s Name, Degree>, Dean

< Leave this page blank >

ABSTRACT

SMART PARKING MANAGEMENT APPLICATION

<Student’s official name >

<University of Name of Univ, <year>

< Project> Chair: <Chair’s Name>

Co-Chair: <If applicable. Co-Chair’s Name>

Urban Areas in modern day society are dense and space is at a premium, and as such cannot be allotted as a parking space easily. But private parking spaces remain open for a variety of reasons. If these spaces were open for use by the public on payed basis, there would be fulfillment of a huge demand while also netting the owners of said parking spaces extra disposable income. A platform that could manage the listings, hourly rate, and allotment of parking spaces is currently not present. Smart Parking aims to fill in this specific vacuum in the market. Smart Parking would allow owners of parking spaces to list their owned parking spaces for an hourly rate, and other users could pay to make use of or reserve these parking spaces. Users would also be able to provide feedback and ratings on parking spaces after using them.

< Leave this page blank >

**TABLE OF CONTENTS**

Introduction 1

About the project 1

Application Features 1

Assumptions 1

Out-of-scope 2

Caveats 2

Feasibility study 2

Technical feasibulity 2

Market research and analysis 3

Unique preposition & value creation 3

Stakeholder Analysis 3

Identification of stakeholders 3

Persona identification 3

Stakeholder engagement 4

Project Plan 5

Project Management Plan 5

Project Scope 5

Risk Management plan 6

Identification, analysis and mitigation plan 6

Risk response Strategies 6

Project Team structure, roles and responsibilites 7

Project Tracking, Scheduling 8

Product Backlog, Scrum cycle, methodology 8

Requirement Analysis 8

Requirement collection methods used 8

Requirements finalisation process 9

Requirements documentation 9

User Stories 9

Effort Estimation Report 9

Design 11

Software Development Methodology 11

Peer review process 11

Collaboration Methodology 11

Coding Guidelines 11

Source Code Repository 12

Technical stack details 12

Database Design: 12

ER Diagram 15

Flowcharts 15

Component Diagram 15

Sequence diagrams 15

Wireframes 16

Application flow 22

Application features 23

Feature 1: Displaying Parking Spots on Map of searched location 23

Feature 2: Parking Spot Search with Filtering 24

Feature 3: Viewing Parking Spot brief Details on marker 24

Feature 4: Viewing Parking Spot Details 24

Feature 5 : Adding Parking spot & Management 25

Feature 6 : Booking a Parking Spot 25

Feature 7 : Booking History & Management 25

Feature 8: Third party payment integration 25

Feature 9 : User Authentication & Profile Management 26

Feature 10: Ratings & Reviews System 26

Testing 27

Quality process 27

Test Cases identification 27

Test Cases execution 27

Defect Tracking and analysis 28

Implementation 29

Conclusion 30

References 31

Glossary: 32

Appendix A: 33

List of Tables

Table 2.1 User Stories 10

List of Figures

Fig. 2.1 Project Planning Gantt Chart 8

FIG. 3.1 ER Diagram 15

FIG. 3.2 FlowChart 15

FIG. 3.3 COMPONENT DIAGRAM 15

FIG. 3.4 SEQUENCE DIAGRAM 15

Fig 3.5 Login Screen 16

Fig 3.6 Home Screen 17

Fig. 3.7 Home screen - Search Location 17

Fig 3.8 Navigation Menu 18

Fig 3.9 Detailed Pin Screen 18

Fig 3.10 Add Spot 19

Fig 3.11 Edit Spot 19

Fig 3.12 Book Spot 20

Fig 3.13 My Spots 20

Fig 3.14 User Profile 21

Fig 3.15 Booking History – User 21

Fig 3.16 Booking History – Owner 22

Introduction

Finding a parking spot in busy urban areas or large venues is a daily struggle. Many people drive around endlessly, wasting time and fuel. This web application aims to solve these challenges by helping users find and book parking spaces in real-time.

**Key Goals:**

Make it easy for users to find and reserve parking spaces before arriving.

About the project

The motivation is to address the daily struggle of finding parking spots in busy areas, saving users time and fuel.

The software developed is a web application that allows users to search for and reserve parking spaces in real-time.

Application Features

* A real-time parking space finder to check available spots in private lots.
* A booking system so users can reserve a space before they arrive.
* A payment system that supports digital transactions for paid parking areas.
* A dashboard for parking lot owners to manage bookings and track earnings.

Assumptions

* The system will be built as a web application.
* The frontend interface will use React.js Native.
* The backend system will use FastAPI (Python).
* A database (PostgreSQL) will be used.
* Maps API integration will be included.
* Payment gateway integration (Stripe/Razorpay) will be implemented.

Out-of-scope

* Physical sensors and hardware-based parking solutions.
* Automated enforcement or towing services for unauthorized parking.
* Integration with government or municipal parking databases.

Caveats

Most parking apps only focus on finding paid spots, but this system adds more flexibility with a seamless reservation system.

Feasibility study

The project is feasible by addressing a common problem (finding parking) with a technology-based solution (web application). It outlines a development plan and identifies the necessary technologies and components. The project's feasibility is also supported by the plan to include features that add value for both users and parking lot owners.

Technical feasibulity

The project leverages several established technologies, indicating technical feasibility:

* Frontend: React.js
* Backend: FastAPI (Python)
* Database: PostgreSQL
* Google Maps API integration
* Razorpay Payment API integration

Market research and analysis

Currently there are no available solutions or competitors that provide the services the app is trying to provide. The closest would be different map apps which include if a location has a parking available or not.

Unique preposition & value creation

As there is no other competitor currently in the market, this itself creates a USP for the app. But adding on an easy to use UI and reservation system would also be preferred.

Stakeholder Analysis

Identification of stakeholders

* Primary Stakeholders: Users of the application.
* Secondary Stakeholders: The application developers and the organization
* Tertairy Stakeholders: Payment gateway providers (Stripe/Razorpay) and Map API providers (Google Maps)

Persona identification

There are two probable personas identified.

1. User: A user is an individual that uses the app to find and book a parking spot.
2. Owner: A user is an individual that uses the app to list their owned parking spots on the app.

Stakeholder engagement

Research & Requirement Gathering: This involves identifying parking challenges and user needs, implying engagement with potential users and parking lot owners to understand their requirements.

The system includes features catering to both users (parking finder, reservation, digital payments) and parking lot owners (dashboard for managing bookings and tracking earnings), demonstrating an intent to engage and provide value to both key stakeholder groups.

***Stakeholder communication***

The development process includes documentation and presentation preparation, suggesting that project progress and outcomes will be communicated to stakeholders.

The community ratings & reviews feature allows users to communicate their experiences with parking spots, facilitating communication and information sharing among users.

Project Plan

The Smart Parking project follows an **TEAMFIT methodology**, which is an in house methododlogy in Bluepineapple. This approach enables the team to build and refine the system based on user needs while also trying to find the root cause for the need, and make changes or improvements accordingly.

Project Management Plan

The project will be managed using Scrum methodology, ensuring efficient development cycles. The team will hold sprint planning meetings, daily standups, and sprint reviews to track progress.

Project Scope

The Smart Parking web application aims to address urban parking issues by allowing users to find and reserve parking spots in real time.

**The system includes:**

* A real-time parking space finder
* A reservation system for advance bookings
* Digital payment integration
* A dashboard for parking lot owners

**Exclusions:**

* No physical sensors or hardware-based solutions
* No automated enforcement or towing services
* No integration with government parking databases

Risk Management plan

* **Technical challenges**: Unforeseen bugs in API integration, New API knowledge acquisition.
* **Security vulnerabilities**: Risks associated with payment processing, data storage and data handling.

Identification, analysis and mitigation plan

* **Technical issues:** Regular code reviews and testing. Regular research and documentation review.
* **Security threats:** Data encryption and compliance with payment security standards, OAuth Implementation.

Risk response Strategies

* **Avoidance**: Proactively addressing security vulnerabilities before launch
* **Mitigation**: Implementing robust API testing procedures
* **Transfer**: Using third-party payment gateways for security compliance
* **Acceptance**: Some minor bugs may exist but will be resolved in updates

Project Team structure, roles and responsibilites

The team is given equal resposibilities and load. The project is subdivided into modules and assigned to the team accordingly. The team is given responsibility of all the parts of the module including Frontend, Backend, Database, Third Party API integration etc.

Any module not listed is to be considered as common amongst all.

**Abhishek:** User Profile, OAuth implementation, Booking History(For both user and owner)

**Pradeep:** Map/Home Screen, Filter Functionality, Search Functionality, Detailed Pin Information Page

**Arjun:** Parking Spot Booking Form, Add Spot Form, Payment Integration

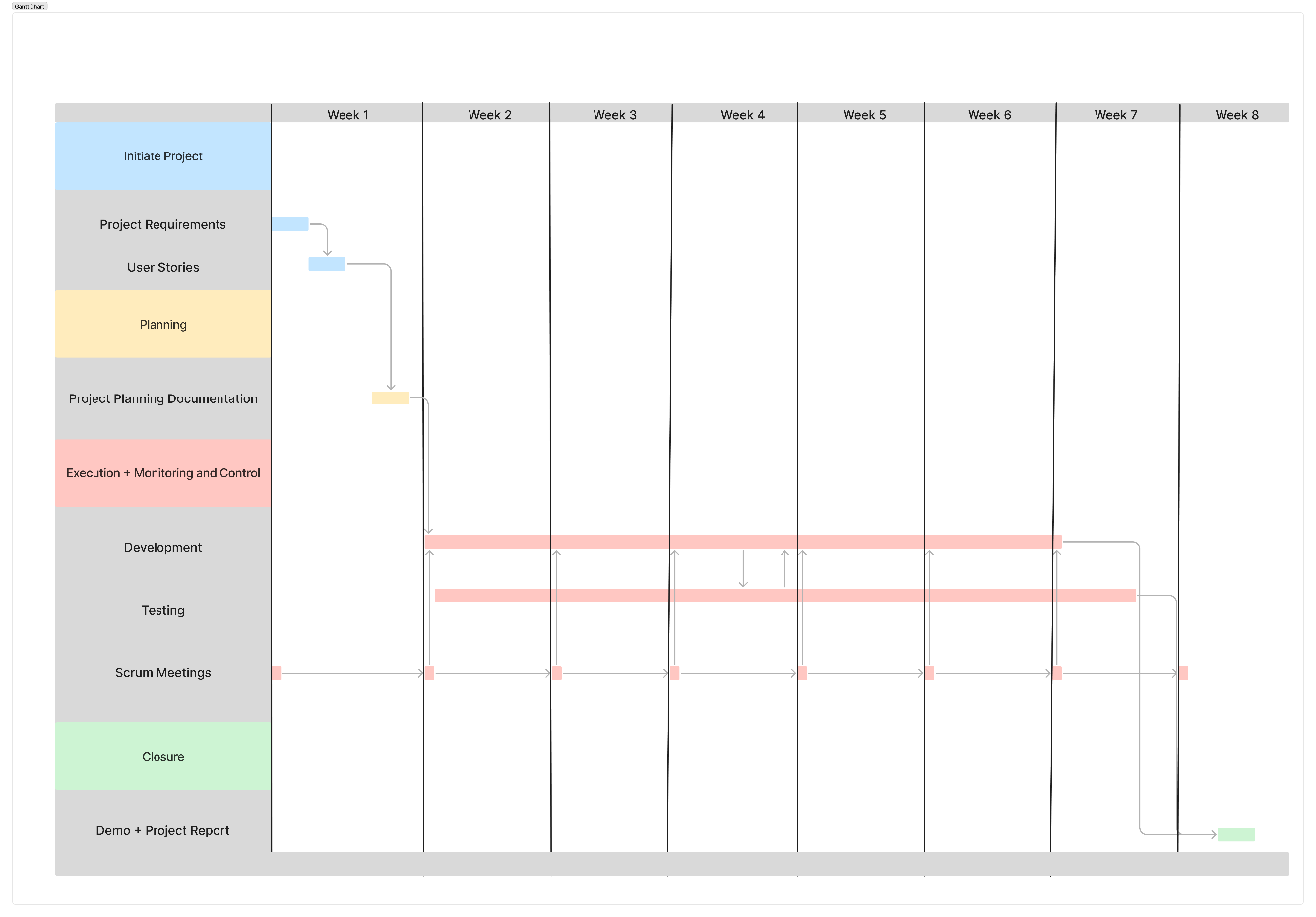


Fig. 2.1 Project Planning Gantt Chart

Project Tracking, Scheduling

* Week 1-2: Requirement gathering and system design
* Week 3-6: Project Development and Unit Testing
* Week 7: System integration and Integration Testing
* Week 8: Documentation and Final Presentation

Product Backlog, Scrum cycle, methodology

The project would be divided into a number of sprints. Each sprint meeting would then discuss the backlog of the previous sprint cycle, decide the time frame to fullfill the backlog, and then decide the features to be developed in the new sprint cycle. A sprint cycle is approximately 15 days.

Requirement Analysis

The project revolves around the need to solve the problem of finding parking spaces in busy areas. This involves understanding the needs of both users (drivers seeking parking) and parking lot owners (providing the parking spaces). The analysis leads to the identification of key functional requirements such as real-time parking availability, booking/reservation capabilities, digital payment processing, and a management dashboard for parking owners.

Requirement collection methods used

Potential cutomers were approached and asked questions to finalise on user stories for the project requirements.

Requirements finalisation process

* Customer survey: Asking potential customer on requirements and pain points.
* User Story inference: This stage helps to solidify the understanding of requirements and translate them into user stories and respective test cases.
* Wireframing: These phases involve validation of the user stories into mock components/modules.
* Finalisation: This suggests a point where the requirements are considered to be fully understood and documented

Requirements documentation

User Stories

[User Stories](https://bluepineapple.sharepoint.com/:x:/r/sites/Internship2025-Project02-SmartParking/Shared%20Documents/Project%2002%20-%20Smart%20Parking/Project%2002%20-%20Smart%20Parking%20-%20Project%20Planning.xlsx?d=w49119c119c864c048621f5406354cbbf&csf=1&web=1&e=ipBmsf&nav=MTVfezAwMDAwMDAwLTAwMDEtMDAwMC0wMDAwLTAwMDAwMDAwMDAwMH0)

Table 2.1 User Stories

Effort Estimation Report

Total Estimated Effort: 193 hrs

< Leave this page blank >

Design

Software Development Methodology

Peer review process

The team would gather every day to discuss issues faced and completed modules. The team then cross checks the code of the other members.

Collaboration Methodology

The team is using GitHub as a version control system, are given separate branches to work on independently. Every week, there would be a merge of the branches into the main branch and resolve the conflicts that might occur in the code. Then every branch is synced to the main branch.

Coding Guidelines

The team will be following general clean cloding practices and conventions.

Source Code Repository

<https://github.com/Abhishek-Rajopadhye/internship-parking-project>

Technical stack details

1. Frontend:
   1. Vite + NodeJS
   2. ReactJS + React Router + Material UI
2. Backend:
   1. FastAPI
   2. SQLAlchemy
   3. OAuth
3. Database:
   1. PostgreSQL

Database Design:

1. Users Table:

userId (INT, Primary Key)

name (VARCHAR, NOT NULL)

email (VARCHAR, NOT NULL, UNIQUE)

password (VARCHAR, NOT NULL)

contact (VARCHAR, NOT NULL)

profileImage (BLOB or VARCHAR for image path)

1. Booking Table:

bookId (INT, Primary Key)

userId (INT, Foreign Key referencing Users.userId)

startTime (DATETIME, NOT NULL)

endTime (DATETIME, NOT NULL)

noOfSlots (INT, NOT NULL)

status (ENUM('cancel', 'confirm', 'active', 'completed'), NOT NULL)

spotId (INT, Foreign Key referencing ParkingSpot.spotId)

1. Payment Table:

paymentId (INT, Primary Key)

spotId (INT, Foreign Key referencing ParkingSpot.spotId)

userId (INT, Foreign Key referencing Users.userId)

razorpayOrderId (VARCHAR, NOT NULL)

razorpayPaymentId (VARCHAR)

razorpaySignature (VARCHAR)

amount (INT, NOT NULL)

status (ENUM('error', 'success'), NOT NULL)

paymentDate (DATETIME, NOT NULL)

1. ParkingSpot Table:

spotId (INT, Primary Key)

ownerId (INT, Foreign Key referencing Users.userId)

address (VARCHAR)

latitude (FLOAT, NOT NULL)

longitude (FLOAT, NOT NULL)

hourlyRate (INT, NOT NULL)

noOfSlots (INT, NOT NULL)

availableSlot (INT, NOT NULL)

openTime (DATETIME, NOT NULL)

closeTime (DATETIME, NOT NULL)

description (VARCHAR)

availableDays (VARCHAR or JSON array, NOT NULL - e.g., "Mon,Tue,Wed")

image (BLOB or VARCHAR for image path)

1. Rating Table:

ratingId (INT, Primary Key)

spotId (INT, Foreign Key referencing ParkingSpot.spotId)

userId (INT, Foreign Key referencing Users.userId)

ratingScore (INT, NOT NULL)

description (VARCHAR)

ratingDate (DATETIME, NOT NULL)

image (BLOB or VARCHAR for image path)

ownerResponse (VARCHAR)

**Explanation of Data Types and Constraints:**

* 1. INT: Integer numbers.
  2. VARCHAR: Variable-length character strings.
  3. DATETIME: Date and time values.
  4. FLOAT: Floating-point numbers (for latitude and longitude).
  5. BLOB: Binary Large Object (for storing images directly in the database). Alternatively, you can store image paths as VARCHAR.
  6. ENUM: A data type that allows you to define a list of possible values.
  7. Primary Key (PK): Uniquely identifies each record in a table.
  8. Foreign Key (FK): Establishes a relationship between tables by referencing the primary key of another table.
  9. NOT NULL: Ensures that a column cannot have a null value.
  10. UNIQUE: Ensures that all values in a column are different.

ER Diagram

[ER Diagram](https://bluepineapple.sharepoint.com/:i:/r/sites/Internship2025-Project02-SmartParking/Shared%20Documents/Project%2002%20-%20Smart%20Parking/ER%20Diagram.png?csf=1&web=1&e=ahcTer)

FIG. 3.1 ER Diagram

Flowcharts

[Flowchart](https://bluepineapple.sharepoint.com/:i:/r/sites/Internship2025-Project02-SmartParking/Shared%20Documents/Project%2002%20-%20Smart%20Parking/Flowchart.jpg?csf=1&web=1&e=HjN9kv)

FIG. 3.2 FlowChart

Component Diagram

[Component Diagram](https://bluepineapple.sharepoint.com/:i:/r/sites/Internship2025-Project02-SmartParking/Shared%20Documents/Project%2002%20-%20Smart%20Parking/component_diagram.png?csf=1&web=1&e=ILcRom)

FIG. 3.3 COMPONENT DIAGRAM

Sequence diagrams

[Sequence Diagram](https://bluepineapple.sharepoint.com/:i:/r/sites/Internship2025-Project02-SmartParking/Shared%20Documents/Project%2002%20-%20Smart%20Parking/Sequence%20Diagram.png?csf=1&web=1&e=m1EPez)

FIG. 3.4 SEQUENCE DIAGRAM

Wireframes

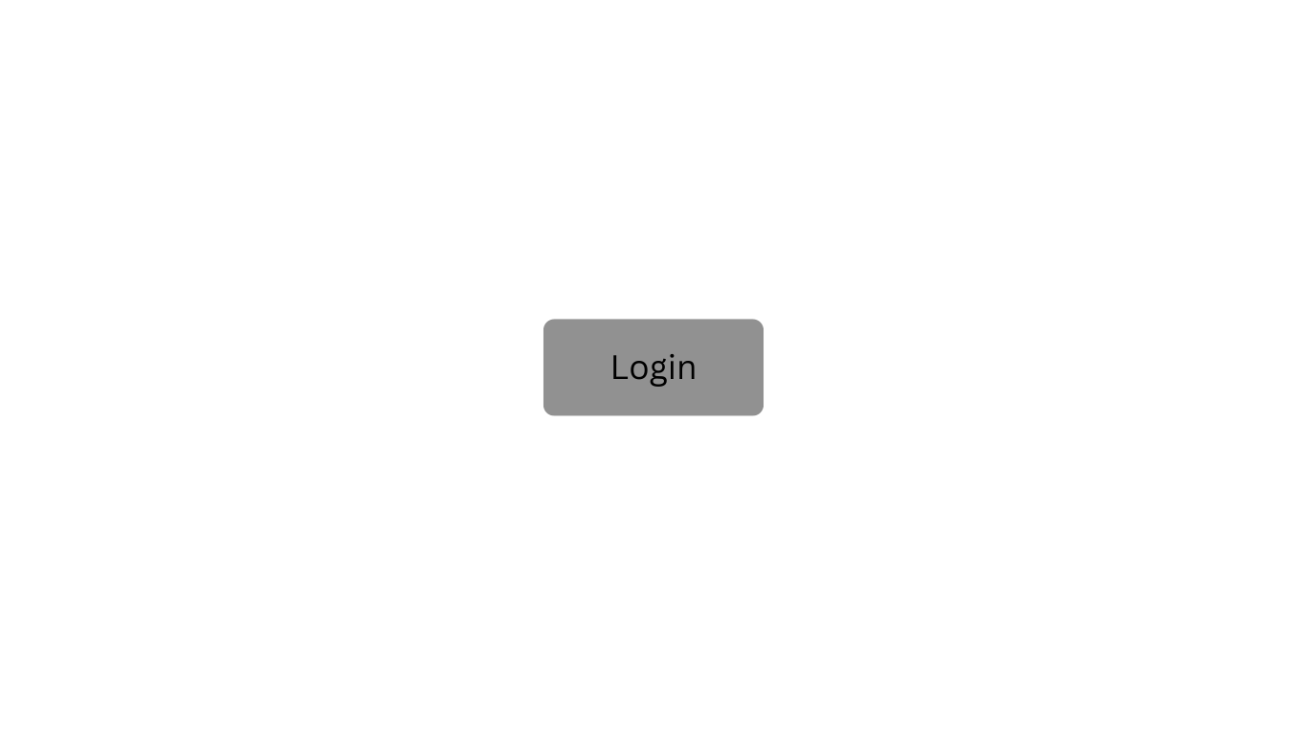


Fig 3.5 Login Screen



Fig 3.6 Home Screen

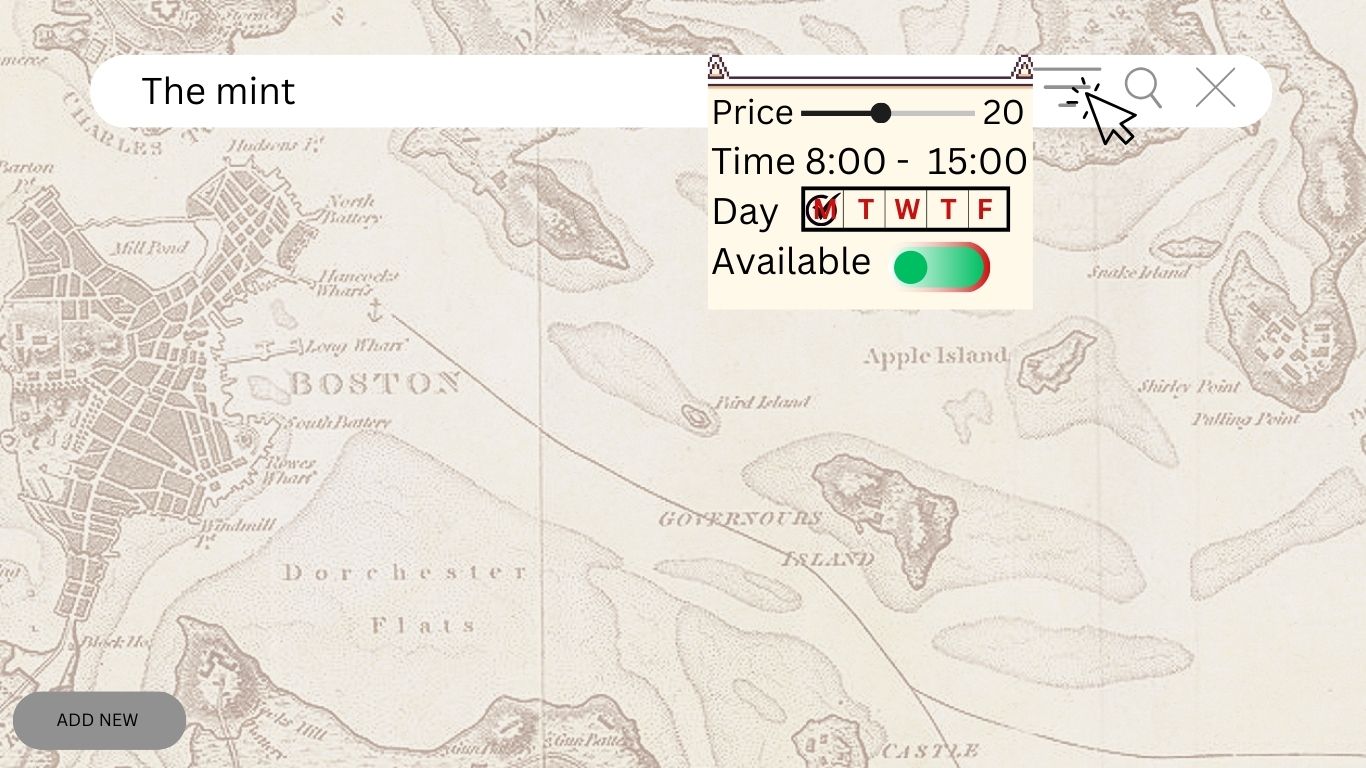


Fig. 3.7 Home screen - Search Location



Fig 3.8 Navigation Menu

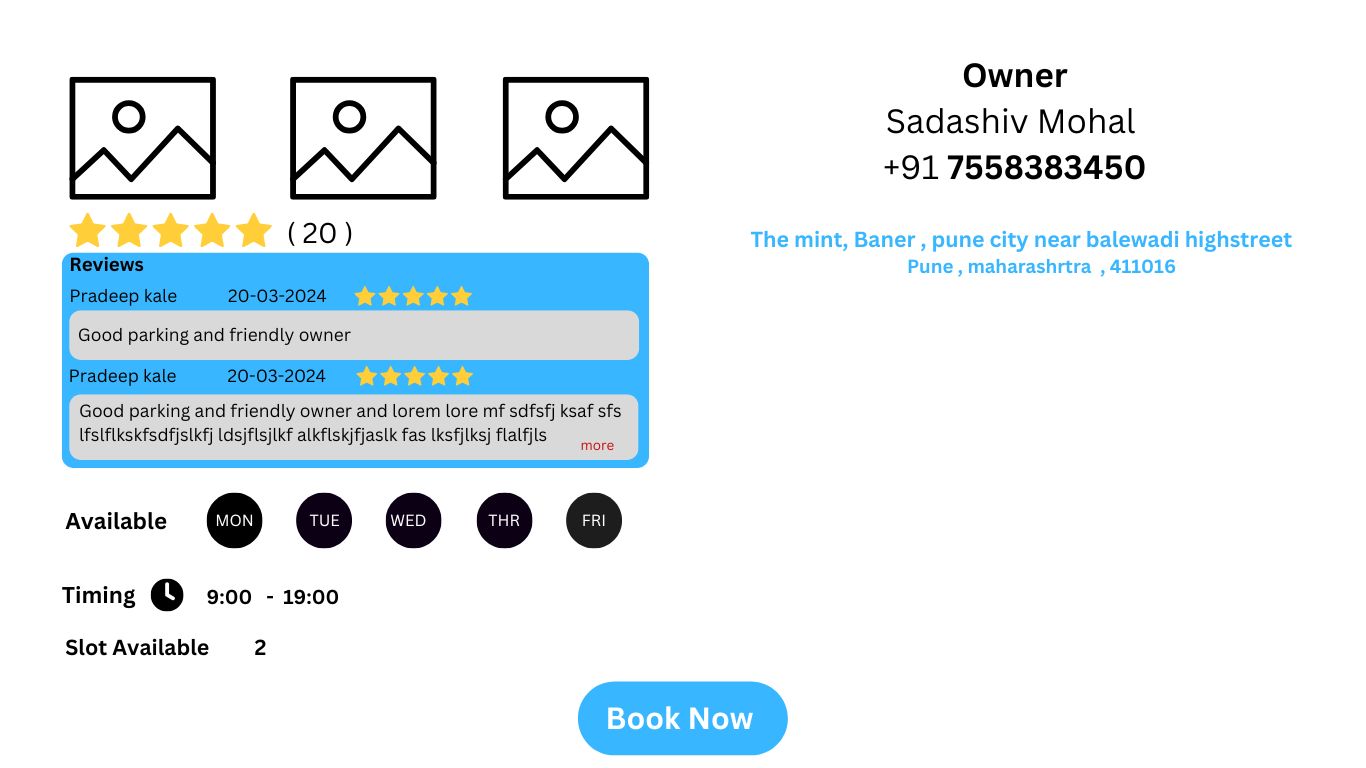


Fig 3.9 Detailed Pin Screen

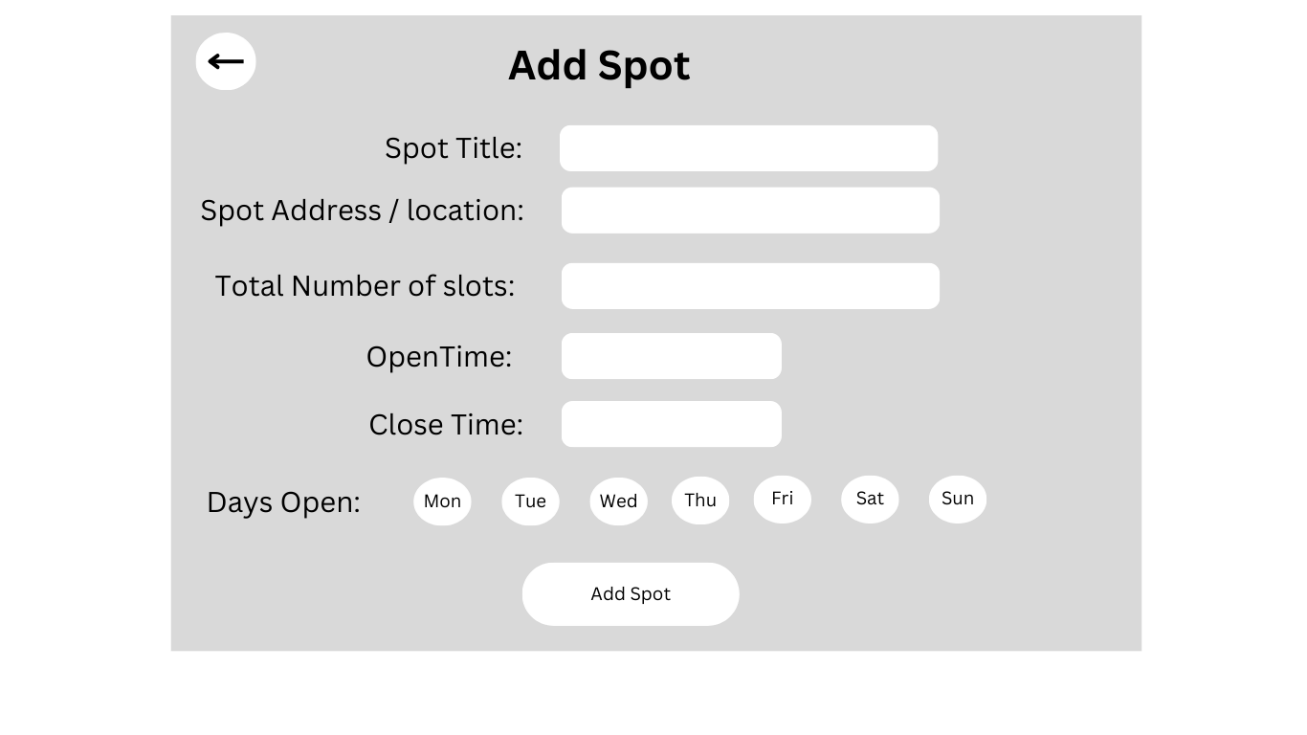


Fig 3.10 Add Spot

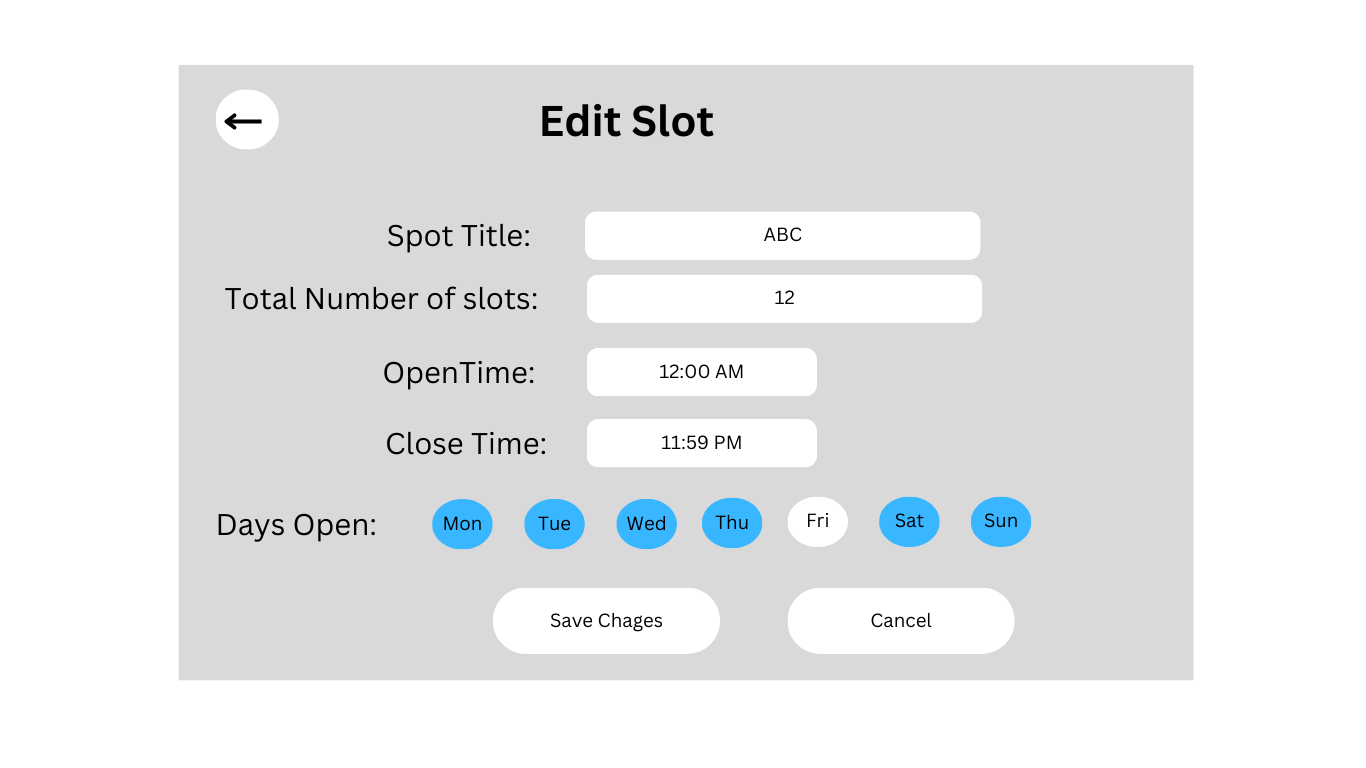


Fig 3.11 Edit Spot

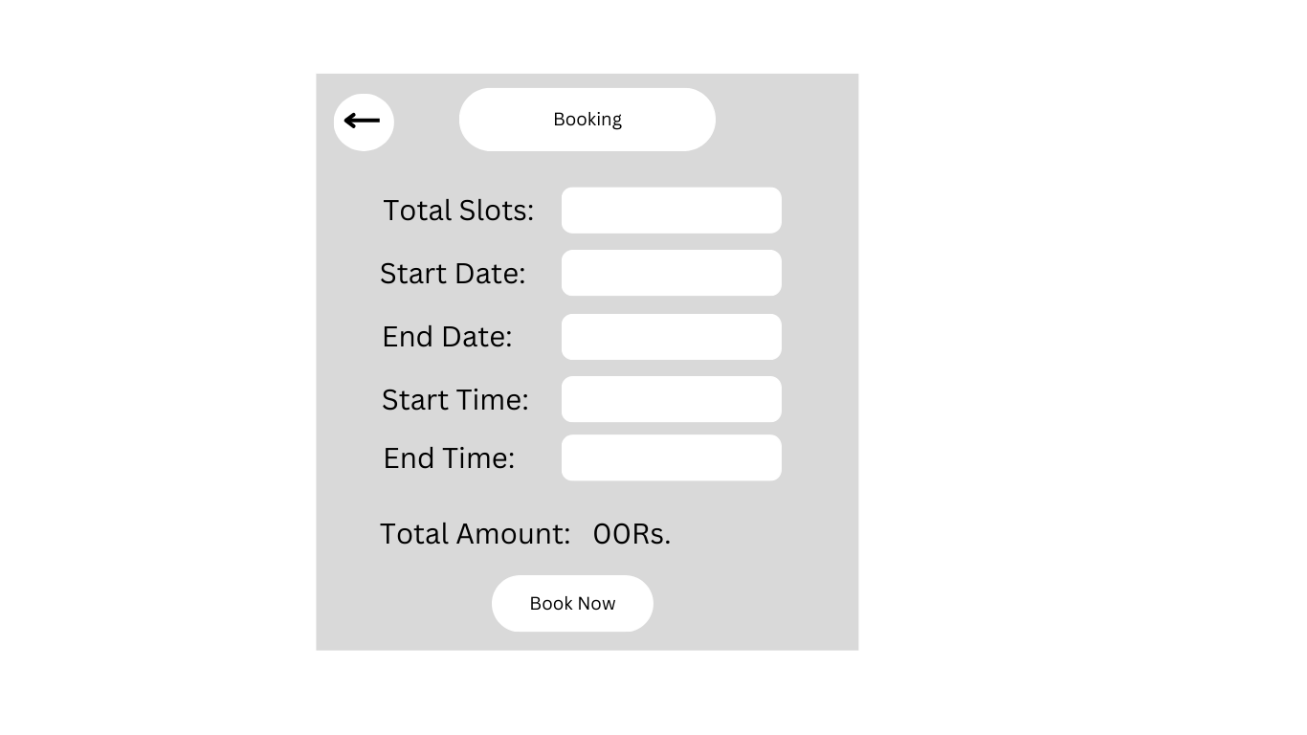


Fig 3.12 Book Spot

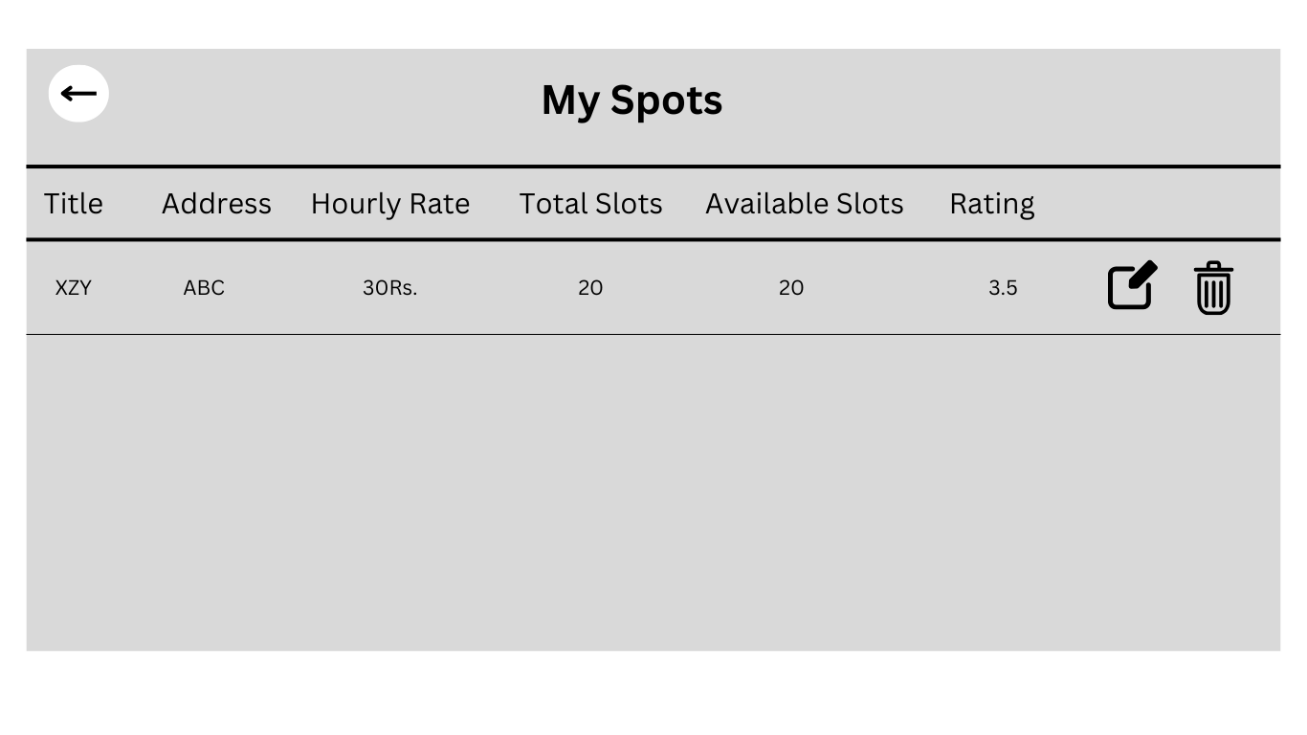


Fig 3.13 My Spots

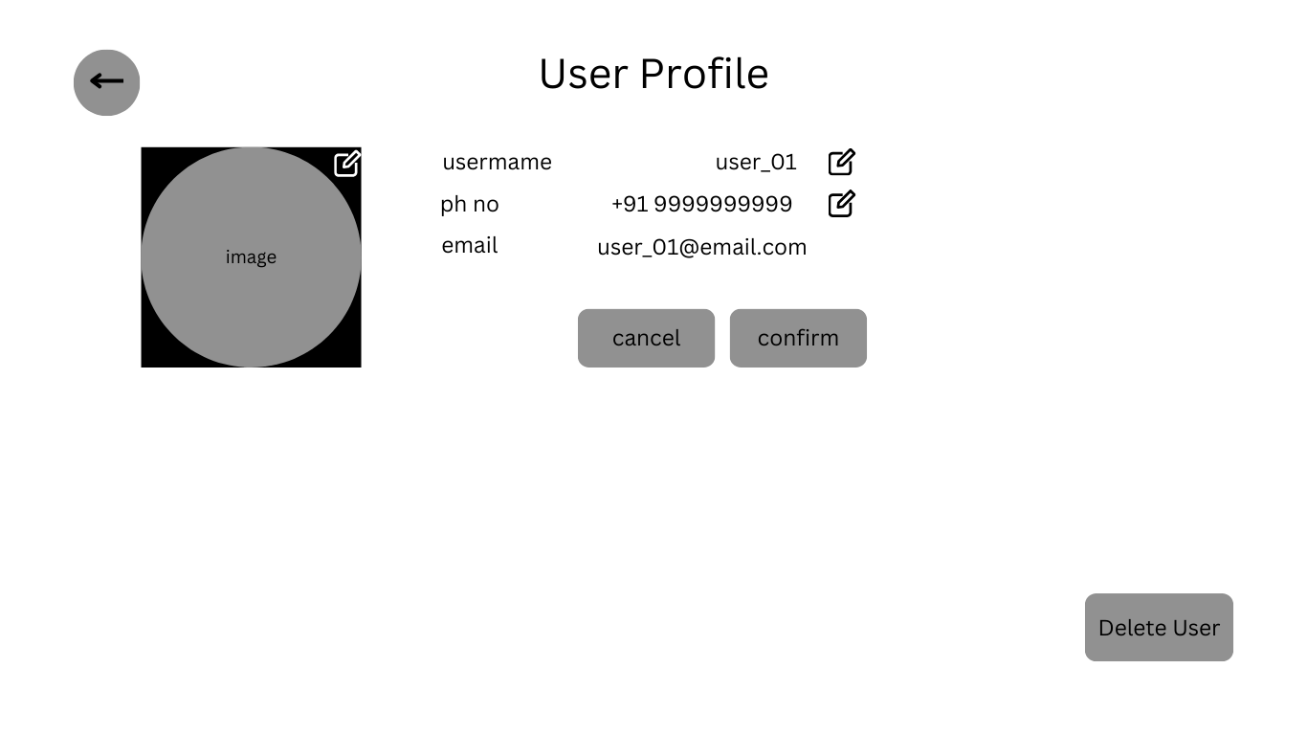


Fig 3.14 User Profile

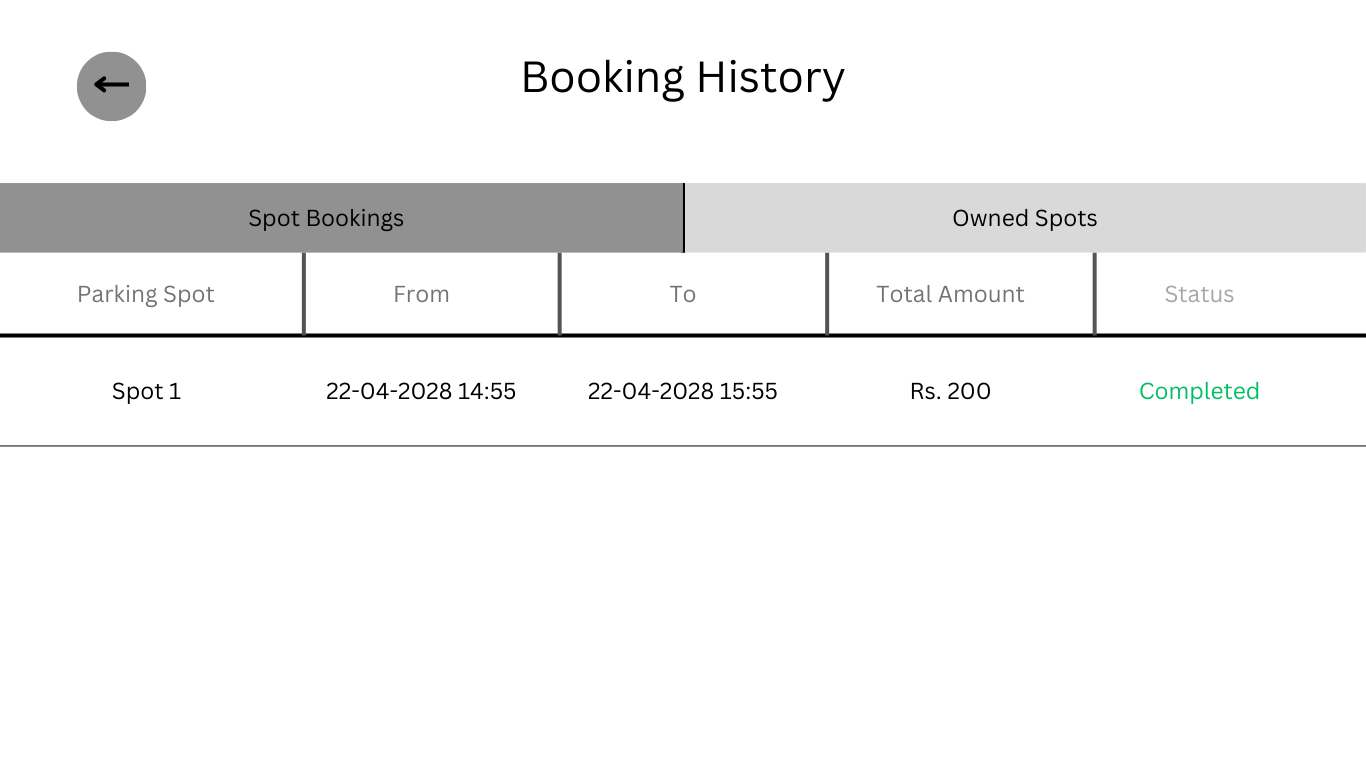


Fig 3.15 Booking History – User

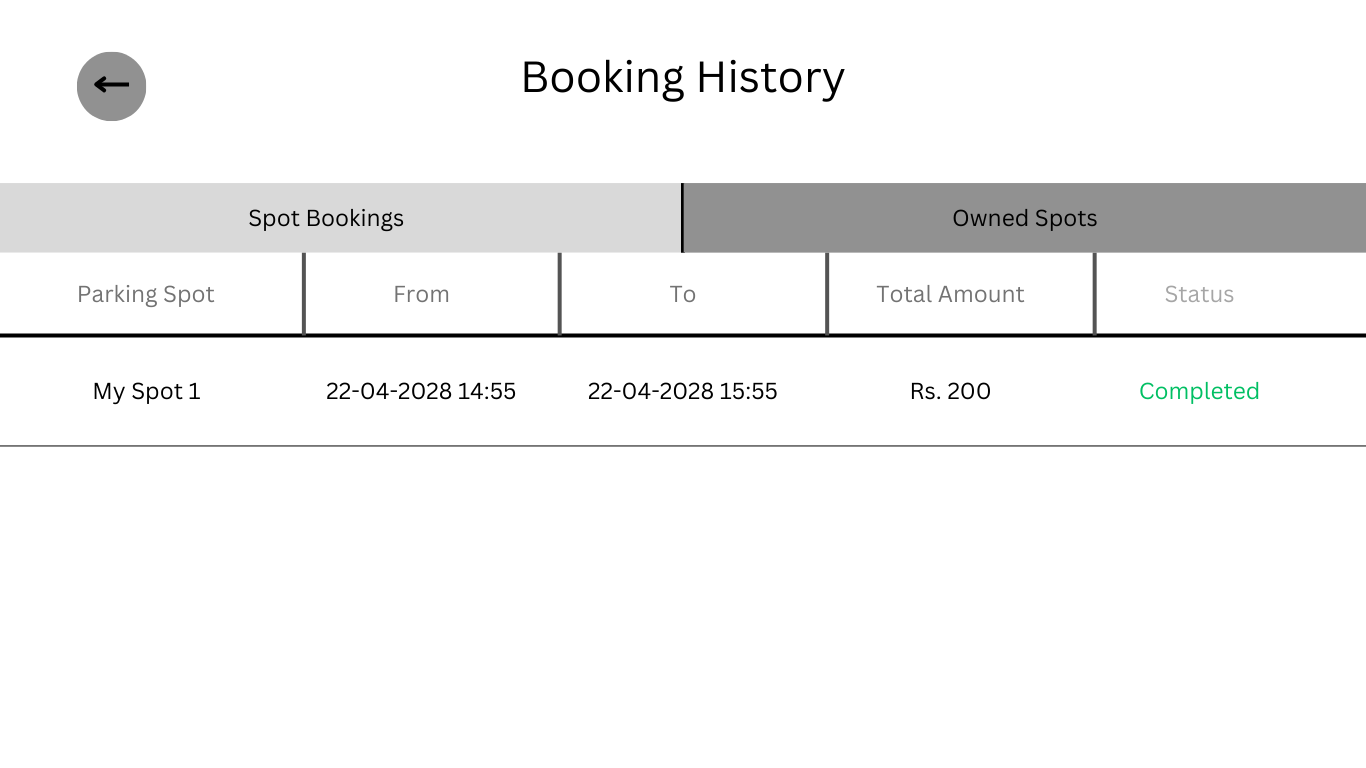


Fig 3.16 Booking History – Owner

Application flow

1. Login
2. Home Screen
   1. Map with search and filter location functionality
   2. Select Spot for brief details
   3. Click on Info Button for Detailed Spot details
      1. Book Spot
   4. Add new/my spot
3. NavBar
   1. Link to User Profile
   2. Link to My Booking History
   3. Link to My Parking Spots
   4. Link to Home
   5. Link to Logout
4. User Profile
   1. Edit User Profile
   2. Delete User
   3. NavBar Navigation
5. My Booking History
   1. My bookings
   2. My spots’ bookings
   3. Navbar navigation
6. My Parking Spots
   1. Individual Spot Details with Earnings
   2. Edit Spot Details
   3. Delete Spot
   4. Navbar Navigation

Application features

The Smart Parking is designed to simplify the process of finding and reserving parking spots. The application allows users to search for parking spots near specific locations, filter results based on various criteria, view details about parking spots, and make reservations for specific dates and times. The application is built using React for the frontend and FastAPI with PostgreSQL for the backend.

Feature 1: Displaying Parking Spots on Map of searched location

This feature allows users to search for a specific location and view all available parking spots marked on an interactive map. Users can zoom, pan, and interact with the map to explore various parking options.

[Insert Screenshot of Map Display Here]

Feature 2: Parking Spot Search with Filtering

Users can search for parking spots by applying various filters, such as days available, time, price range, availability, and distance. Clicking on the filter icon beside the search bar opens a detailed filter panel. Once the desired filters are selected, clicking the 'Apply' button updates the search results to display only relevant parking spots.

[Insert Screenshot of Search Bar and Filter Panel Here]

Feature 3: Viewing Parking Spot brief Details on marker

Clicking on a map marker shows a brief description of the parking spot, including its name, availability, and price. Users can click “INFO ICON” for complete details.

[Insert Screenshot of Marker InfoWindow]

Feature 4: Viewing Parking Spot Details

Users can view detailed information about a parking spot by clicking on a “INFO ICON” on the marker Info Window . The details include address, available slots, hourly rate, open hours,open days, instructions, reviews,rating,owner details and photos.

[Insert Screenshot of Parking Spot Details Page Here]

Feature 5 : Adding Parking spot & Management

Parking spot owners can add new spots, provide descriptions, and manage existing listings. They can also edit or delete their listings as needed.

[Insert Screenshot of Adding Parking Spot Form ]

Feature 6 : Booking a Parking Spot

Users can book a parking spot for a specific date and time by filling out a booking form. After confirming the booking, users receive a confirmation message or notification.

[Insert Screenshot of Booking Form ]

Feature 7 : Booking History & Management

Users can access their past and upcoming bookings. They can view details, cancel bookings if needed.

[Insert Screenshot of Booking History ]

Feature 8: Third party payment integration

Users can securely make payments for their bookings through integrated payment gateways, ensuring a smooth transaction process.

[Insert Screenshot of Payment Gateway Page ]

Feature 9 : User Authentication & Profile Management

This feature allows users to register, log in, and manage their profiles. Users can view and edit their personal details and view their booking history.

[Insert Screenshot of Profile Page ]

Feature 10: Ratings & Reviews System

Users can submit ratings and reviews based on their parking experience. They can also view ratings and reviews submitted by others .

[Insert Screenshot of Ratings & Reviews Section ]

Testing

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Quality process

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Test Cases identification

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Test Cases execution

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Defect Tracking and analysis

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Implementation

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Conclusion

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

References

<Use the citation style for your discipline. Indicate materials referenced in the project and from sources you have actually used.>

{Author, A. A., Author, B. B., & Author, C. C. (Year). Title of article. Title of Periodical, volume number(issue number), pages. http://dx.doi.org/xx.xxx/yyyyy}

Glossary:

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }

Appendix A:

{Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text Sample Text }