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 >

**Acknowledgements**

<Optional. If you do not include a dedication, delete this entire page. **Do not** delete the section break on the Dedication page above. Press Ctrl+Shift+8 to see the section break.>

< 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.

**TABLE OF CONTENTS**

Level 1 Heading 1

Level 2 Heading 1

Level 3 Heading 1

Introduction 4

About the project 4

Application Features 4

Assumptions 4

Out-of-scope 4

Caveats 4

Feasibility study 5

Technical feasibulity 5

Market research and analysis 5

Unique preposition & value creation 5

Stakeholder Analysis 5

Identification of stakeholders 6

Persona identification 6

Stakeholder engagement 6

Stakeholder communication 6

Project Plan 7

Project Management Plan 7

Project Scope 7

Risk Management plan 7

Identification, analysis and mitigation plan 7

Risk response Strategies 7

Project Team structure, roles and responsibilites 8

Project Tracking, Scheduling 8

Product Backlog, Scrumm cycle, methodology 8

Requirement Analysis 8

Requirement collection methods used 8

Requirements finalisation process 9

Requirements documentation 9

User Stories 9

Detailed User stories description 9

Rquirements Estimation 9

Effort Estimation Report 10

Design 11

Software Development Methodology 11

Peer review process 11

Collaboration Methodology 11

Coding Guidelines 11

Source Code Repository 11

Technical stack details 12

Technologies used, their versions 12

Database Design 12

ER Diagrams 12

Flowcharts 12

Component Diagram 12

Sequence diagrams 13

Wireframes 13

Development 14

Application flow 14

Application features 14

Feature 1 14

Feature 2 14

Testing 15

Quality process 15

Test Cases identification 15

Test Cases execution 15

Defect Tracking and analysis 15

Implementation 16

Conclusion 17

References 18

Glossary: 19

Appendix A: 20

List of Tables

Table 2.1[press Shift + Enter] Sample Table Caption [Use “Table Caption” Style”] 3

List of Figures

Figure 2.1[press Shift + Enter] Sample Figure Caption [Use “Figure Caption” Style”]3

Level 1 Heading

Level 1 heading should always start on ODD PAGE number. If required include a blank page so that the heading starts on an odd page.

{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 }

Level 2 Heading

{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 }

Level 3 Heading

{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 }

* Sample bullet Text Sample bullet Text
* **Sample bullet Text**: Sample bullet Text   
  Sample bullet Text Sample bullet 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
  + Text Sample Text Sample Text Sample
  + Text Sample Text Sample Text Sample
* Text Sample Text Sample Text Sample

Sample numbered Text

* 1. Sample numbered Text
  2. Sample numbered Text

Table 2.1[press Shift + Enter]  
  
Sample Table Caption [Use “Table Caption” Style”]

|  |  |  |
| --- | --- | --- |
| **Variable** | **%** | ***N*** |
| Value | x | x |
| Value | x | x |
| Value | x | x |

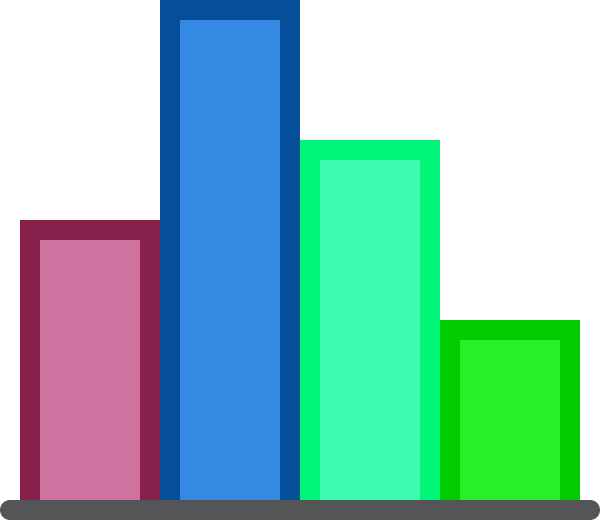


Figure 2.1[press Shift + Enter]  
Sample Figure Caption [Use “Figure Caption” Style”]

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

{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 }

Technical feasibulity

{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 }

Market research 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 }

Unique preposition & value creation

{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 }

Stakeholder 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 }

Identification of stakeholders

{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 }

Persona 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 }

Stakeholder engagement

{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 }

Stakeholder communication

{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 }

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 among 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

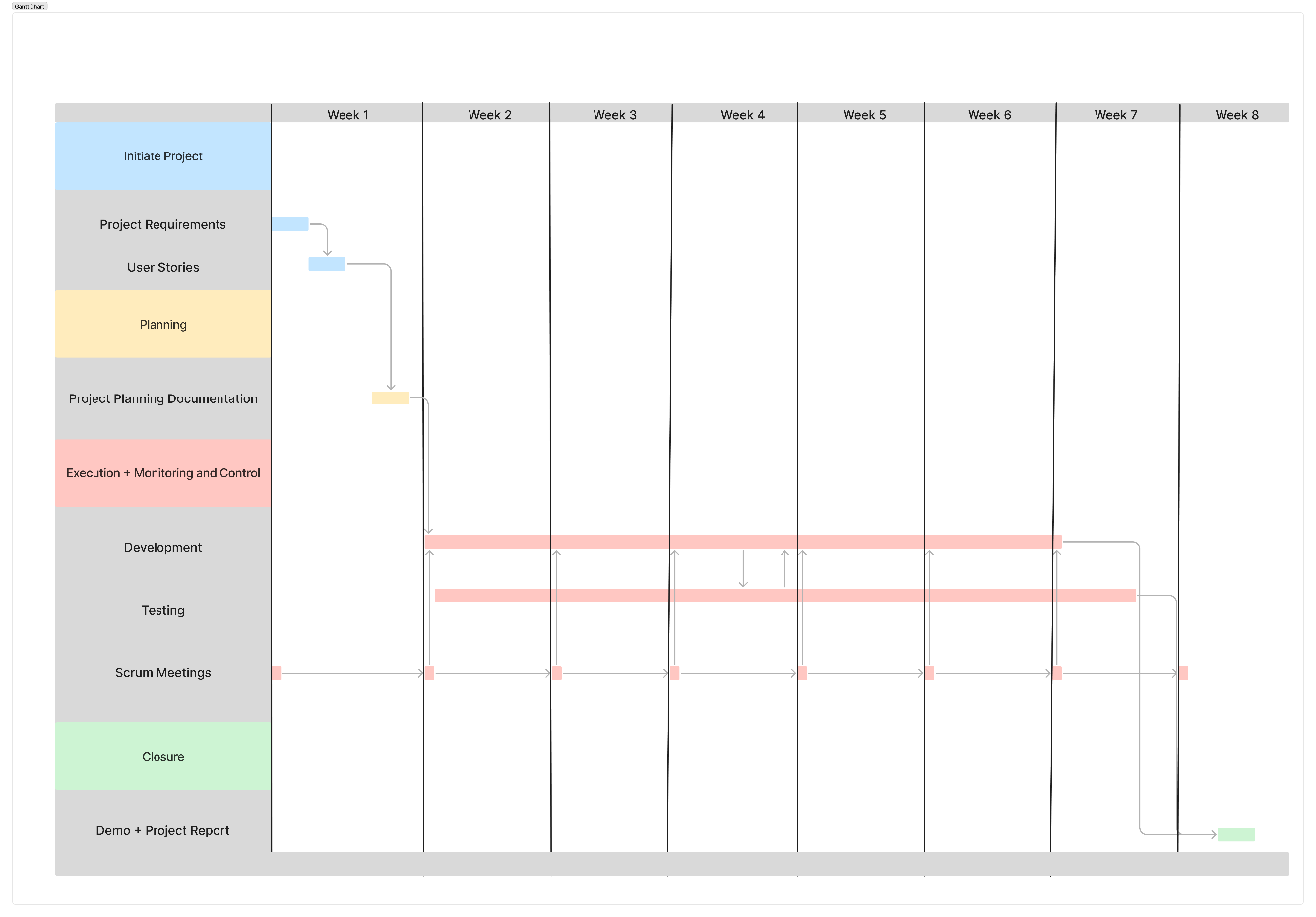
Project Tracking, Scheduling

fig 3.1 Project planning gantt chart

* Week 1-2: Requirement gathering and system design
* Week 3-4: Backend development (API, database setup)
* Week 5-6: Frontend development (React UI, integration)
* Week 7: System integration and testing
* Week 8: Documentation and final presentation

Product Backlog, Scrumm cycle, methodology

{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 }

Requirement 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 }

Requirement collection methods used

{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 }

Requirements finalisation 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 }

Requirements documentation

{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 }

User Stories

{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 }

Detailed User stories description

{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 }

Rquirements Estimation

{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 }

Effort Estimation Report  
Total Estimated Effort: 193 hrs

1. User story US-001:

* Est Dev Time: 1 hr
* Est QA Time: 0.5 hrs
* Total Est Time: 1.5 hrs

1. User story US-002:

* Est Dev Time: 1 hr
* Est QA Time: 0.5 hrs
* Total Est Time: 1.5 hrs

1. User story US-003:

* Est Dev Time: 1 hr
* Est QA Time: 0.5 hrs
* Total Est Time: 1.5 hrs

1. User story US-004:

* Est Dev Time: 1 hr
* Est QA Time: 0.5 hrs
* Total Est Time: 1.5 hrs

1. User story US-005:

* Est Dev Time: 5 hrs
* Est QA Time: 2 hrs
* Total Est Time: 7 hrs

1. User story US-006:

* Est Dev Time: 3 hrs
* Est QA Time: 1 hr
* Total Est Time: 4 hrs

1. User story US-007:

* Est Dev Time: 2 hrs
* Est QA Time: 1 hr
* Total Est Time: 3 hrs

1. User story US-008:

* Est Dev Time: 15 hrs
* Est QA Time: 4 hrs
* Total Est Time: 19 hrs

1. User story US-009:

* Est Dev Time: 16 hrs
* Est QA Time: 5 hrs
* Total Est Time: 21 hrs

1. User story US-010:

* Est Dev Time: 8 hrs
* Est QA Time: 2 hrs
* Total Est Time: 10 hrs

1. User story US-011:

* Est Dev Time: 12 hrs
* Est QA Time: 3 hrs
* Total Est Time: 15 hrs

1. User story US-012:

* Est Dev Time: 5 hrs
* Est QA Time: 1 hr
* Total Est Time: 6 hrs

1. User story US-013:

* Est Dev Time: 7 hrs
* Est QA Time: 2 hrs
* Total Est Time: 9 hrs

1. User story US-014:

* Est Dev Time: 4 hrs
* Est QA Time: 1 hr
* Total Est Time: 5 hrs

1. User story US-015:

* Est Dev Time: 10 hrs
* Est QA Time: 3 hrs
* Total Est Time: 13 hrs

1. User story US-016:

* Est Dev Time: 8 hrs
* Est QA Time: 2 hrs
* Total Est Time: 10 hrs

1. User story US-017:

* Est Dev Time: 6 hrs
* Est QA Time: 2 hrs
* Total Est Time: 8 hrs

1. User story US-018:

* Est Dev Time: 9 hrs
* Est QA Time: 2 hrs
* Total Est Time: 11 hrs

1. User story US-019:

* Est Dev Time: 7 hrs
* Est QA Time: 2 hrs
* Total Est Time: 9 hrs

1. User story US-020:

* Est Dev Time: 9 hrs
* Est QA Time: 3 hrs
* Total Est Time: 12 hrs

1. User story US-021:

* Est Dev Time: 11 hrs
* Est QA Time: 3 hrs
* Total Est Time: 14 hrs

1. User story US-022:

* Est Dev Time: 7 hrs
* Est QA Time: 2 hrs
* Total Est Time: 9 hrs

1. User story US-023:

* Est Dev Time: 5 hrs
* Est QA Time: 1 hr
* Total Est Time: 6 hrs

1. User story US-024:

* Est Dev Time: 6 hrs
* Est QA Time: 2 hrs
* Total Est Time: 8 hrs

1. User story US-025:

* Est Dev Time: 8 hrs
* Est QA Time: 2 hrs
* Total Est Time: 10 hrs

Design

{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 }

Software Development Methodology

{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 }

Peer review 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 }

Collaboration Methodology

Coding Guidelines

{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 }

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)

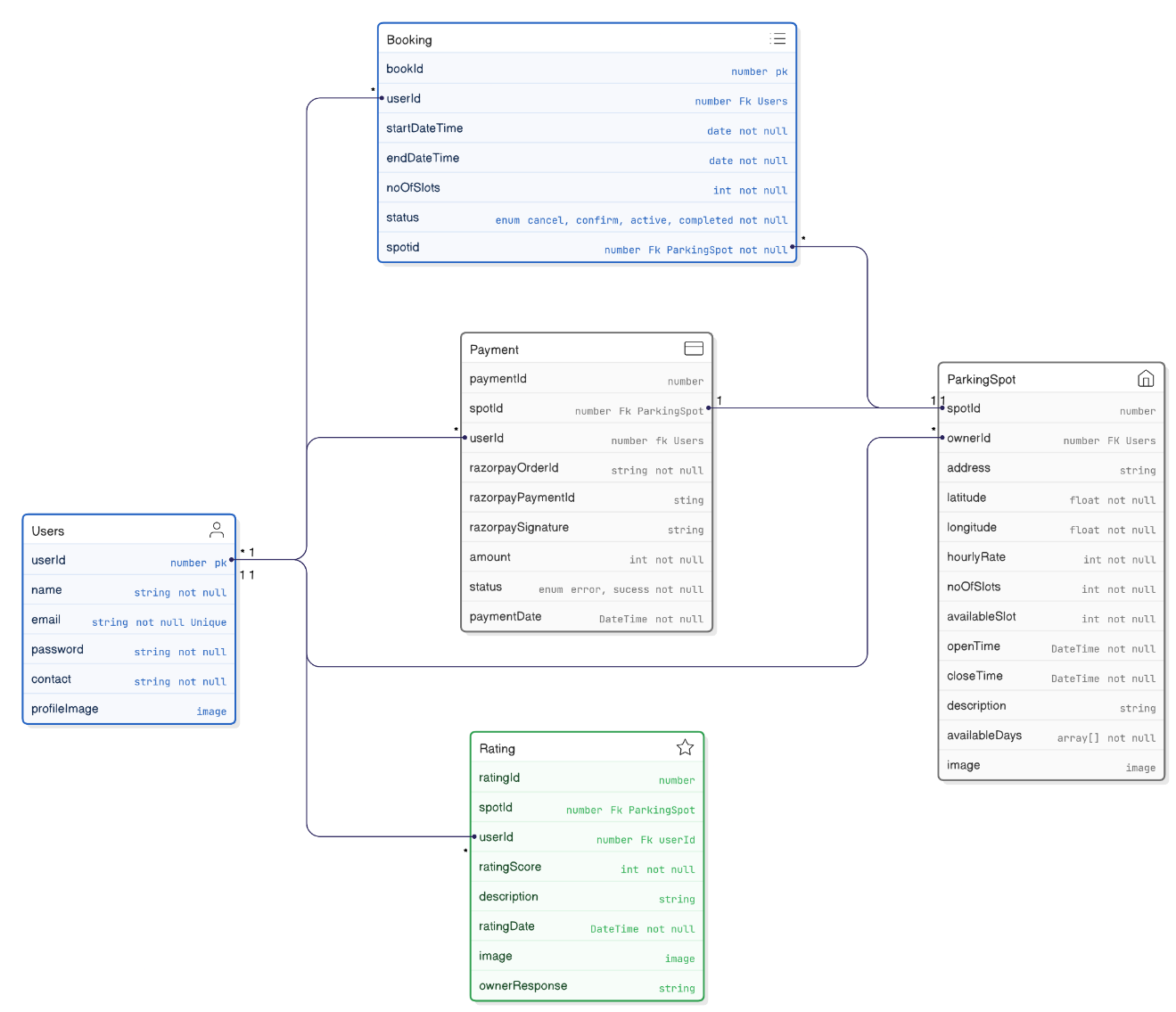
ratingDate (DATETIME, NOT NULL)

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

Flowcharts

Component Diagram

Sequence diagrams

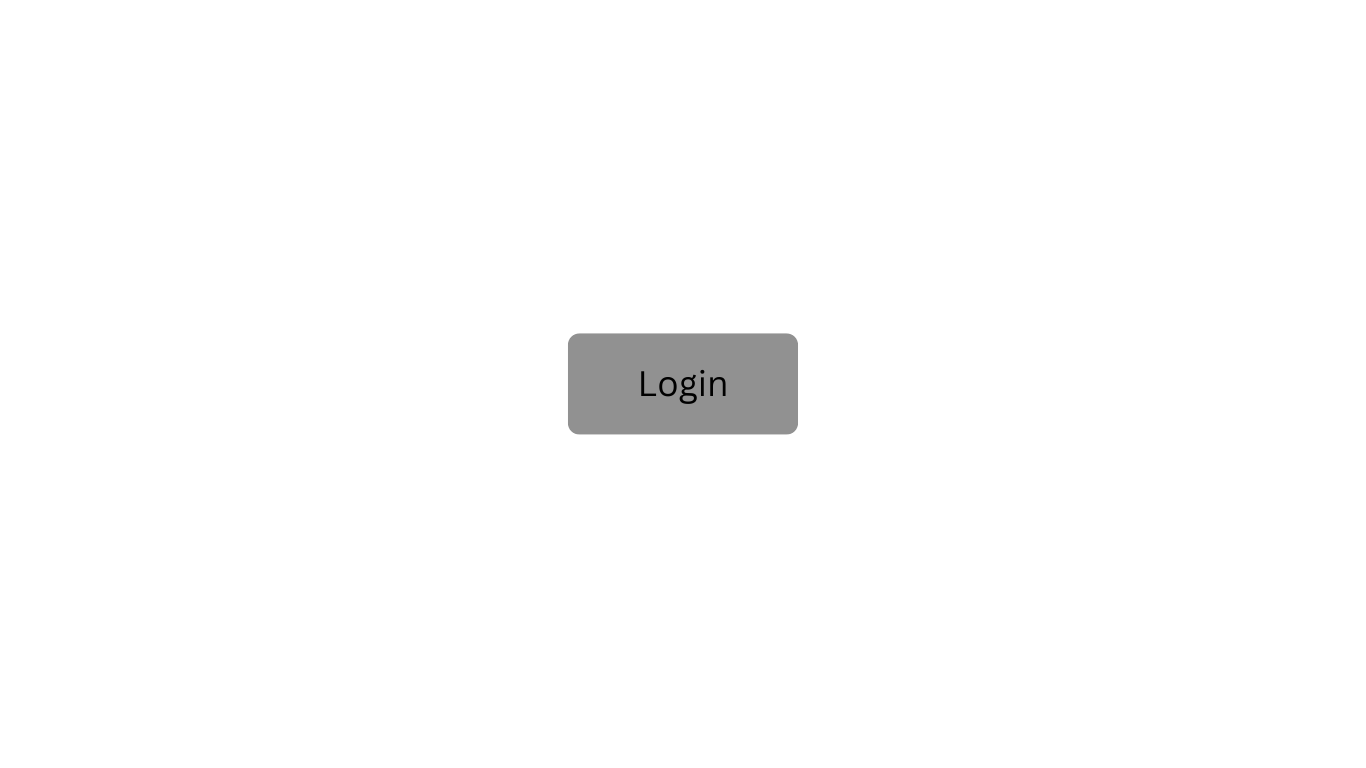
Wireframes

Fig 4.2 Login Screen

Fig 4.3 Home Screen

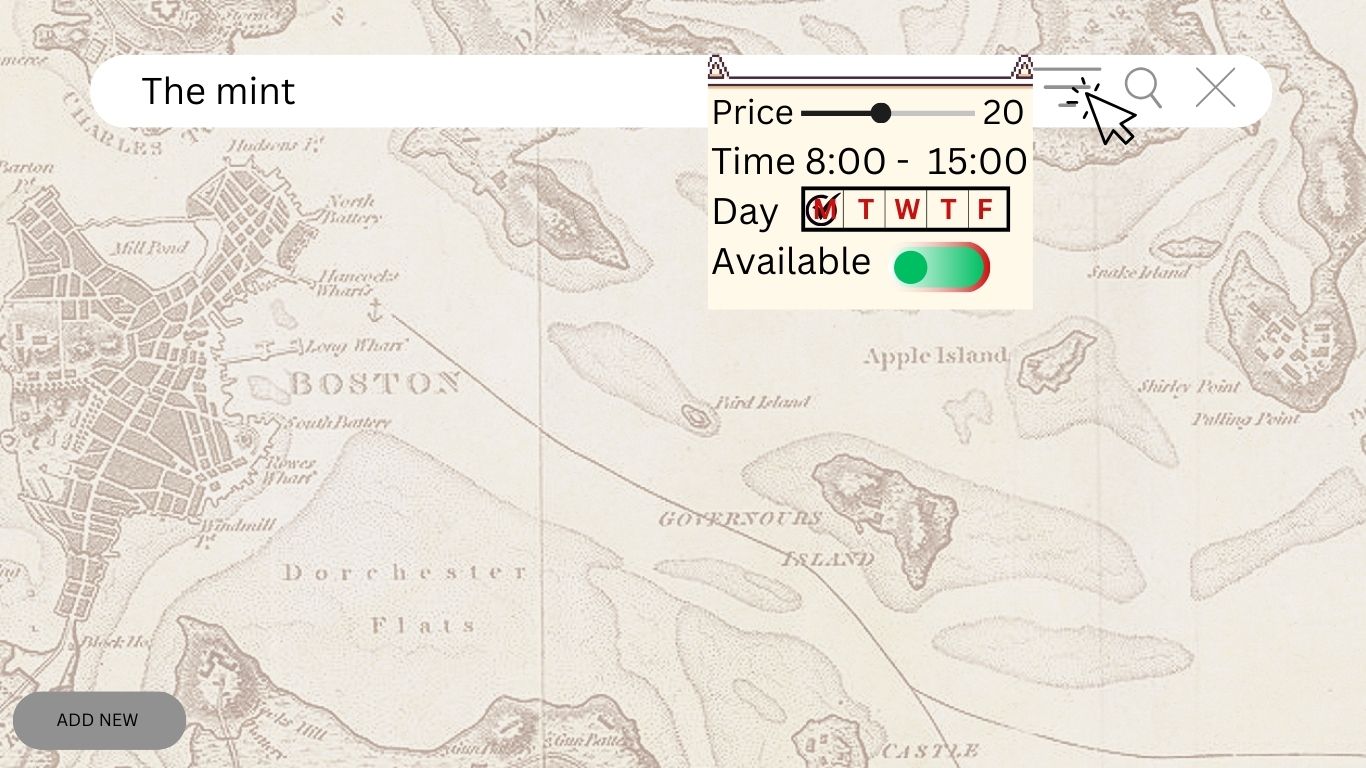
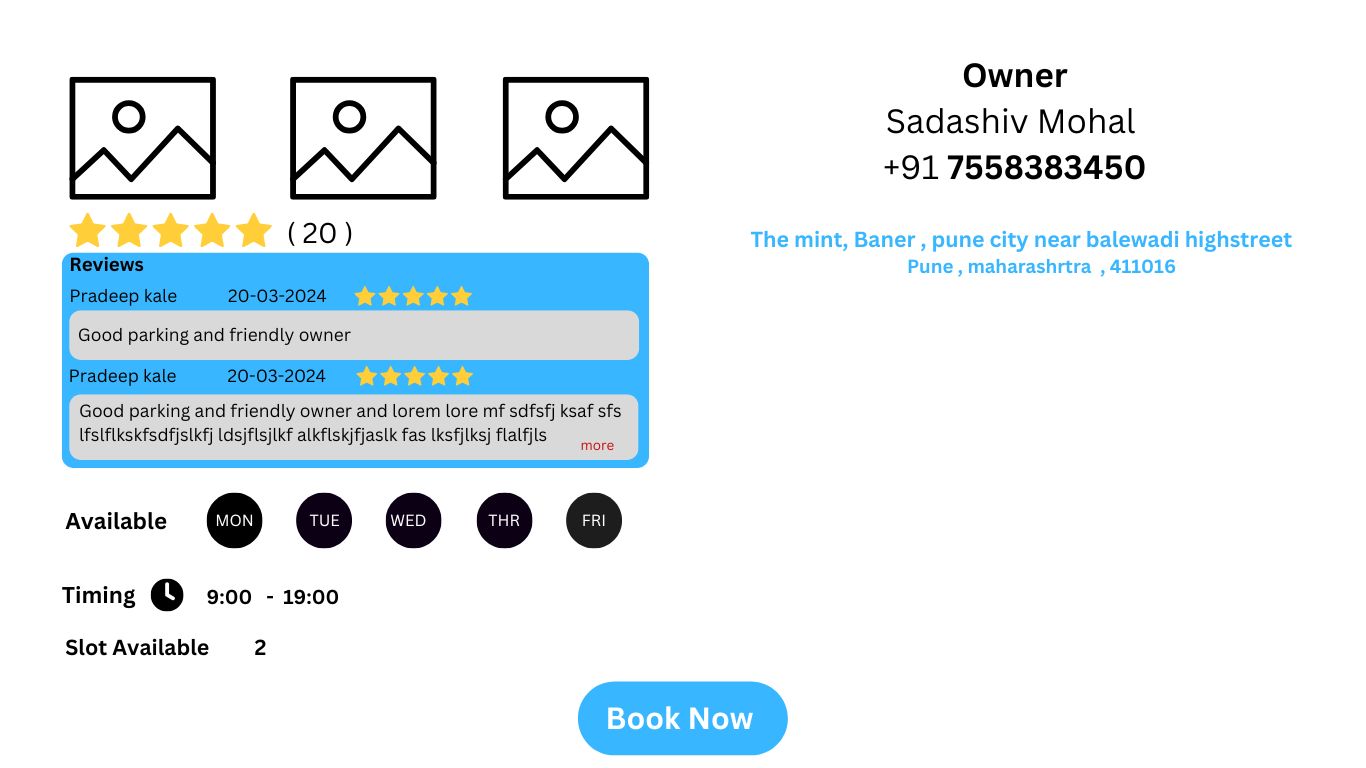
fig. 4.4 Home screen - search location

fig 4.5 navigation menu

fig 4.6 detailed pin screen

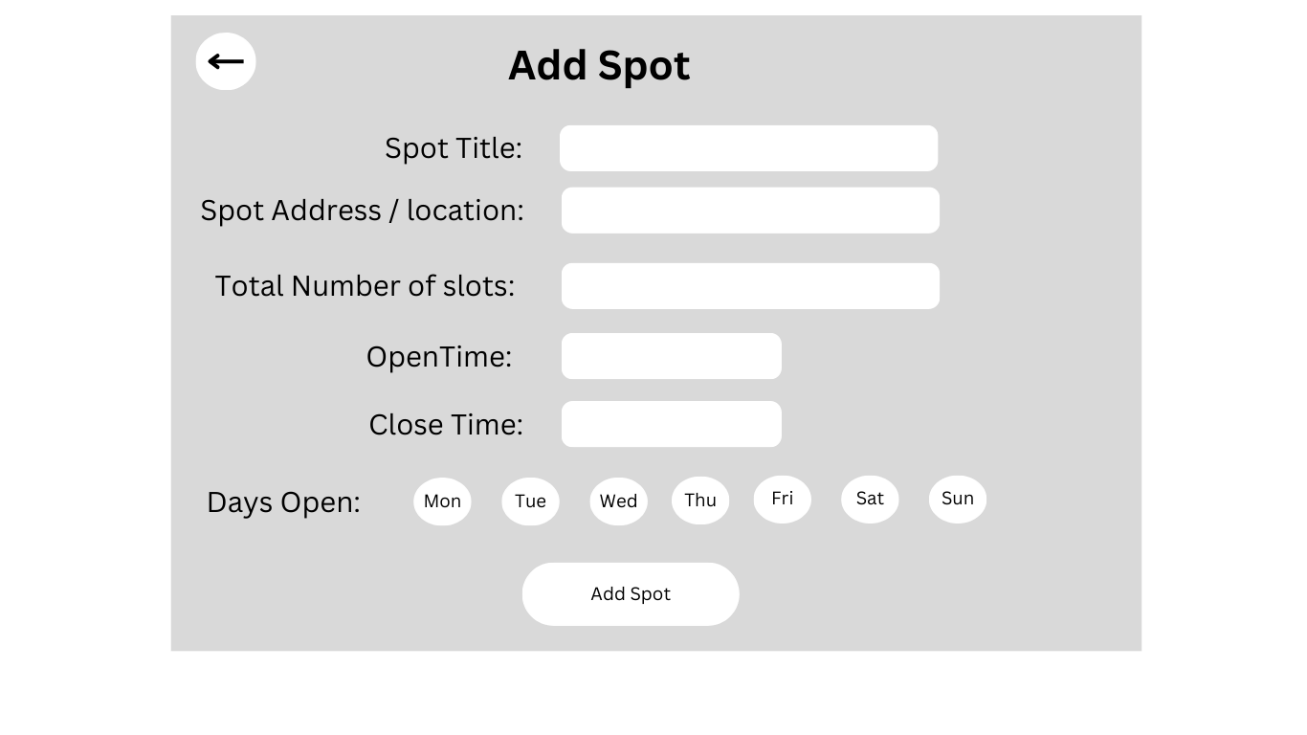


fig 4.7 add spot

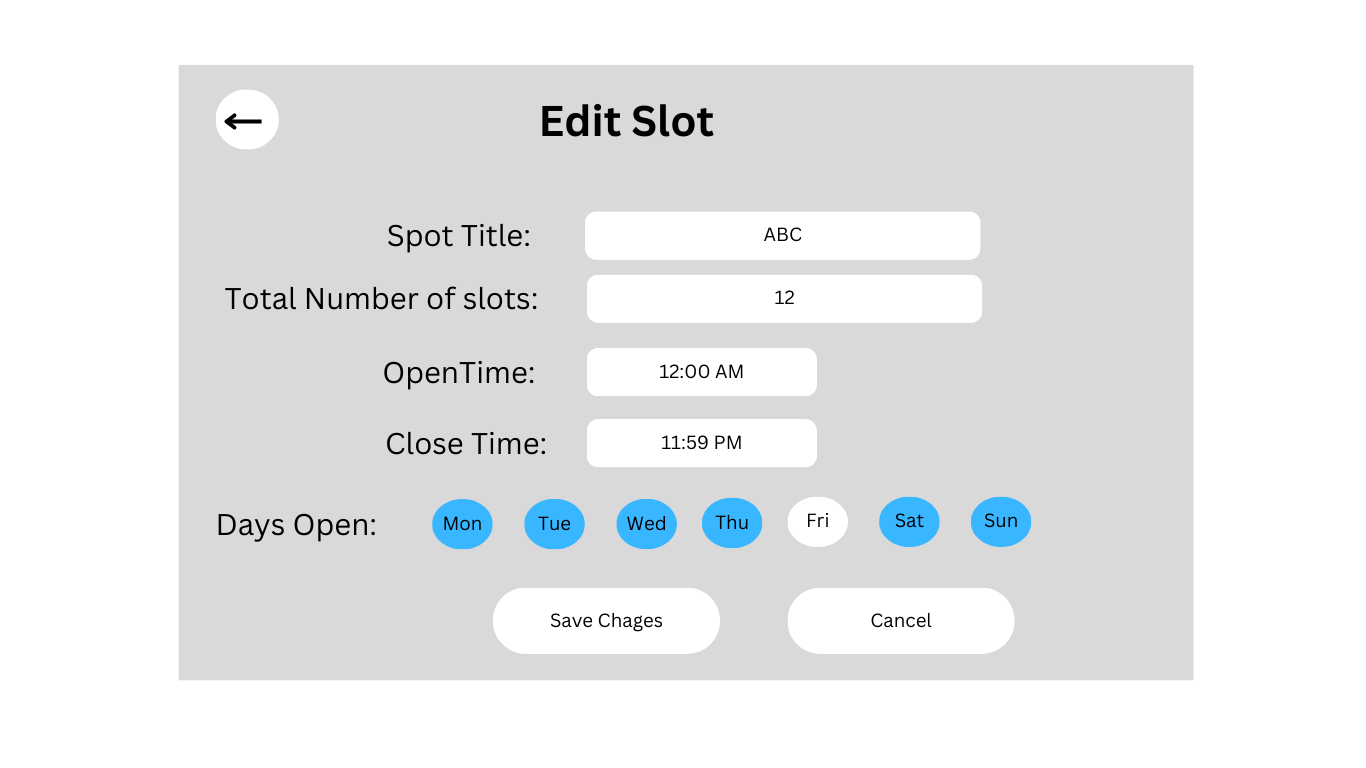


fig 4.8 edit spot

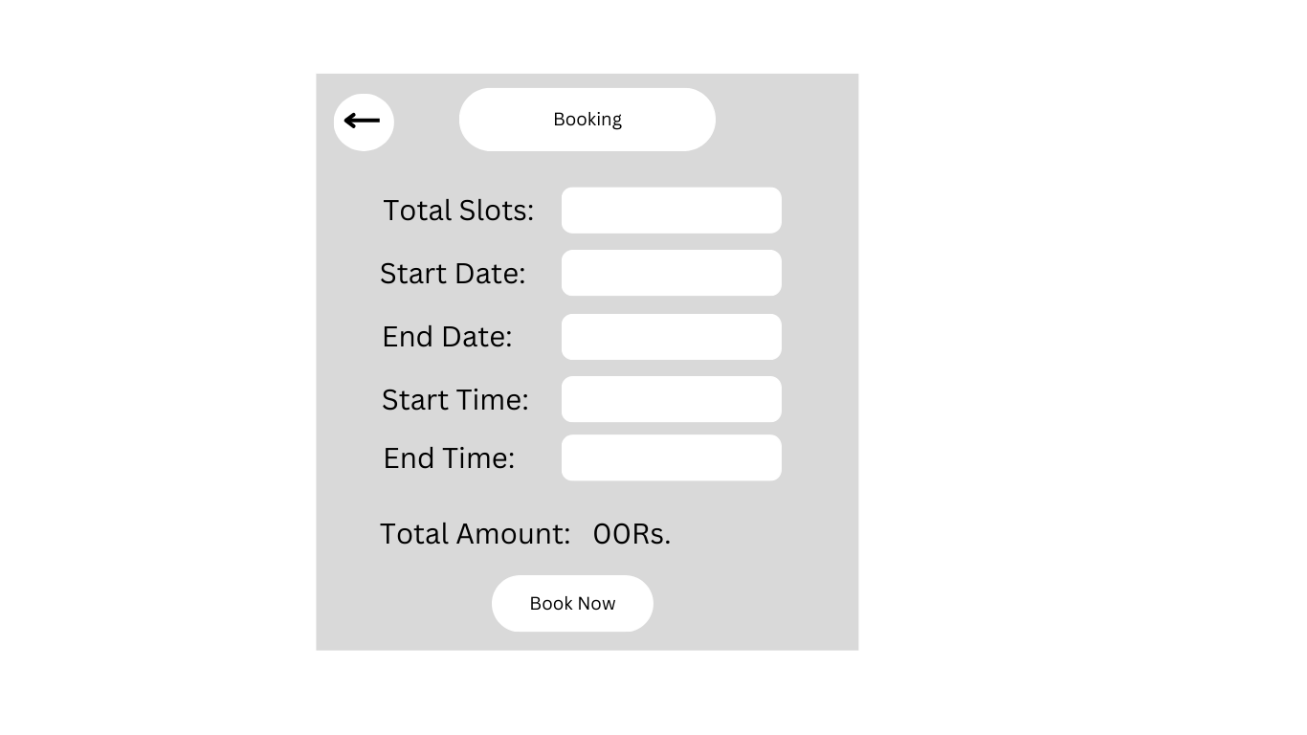


fig 4.9 book spot

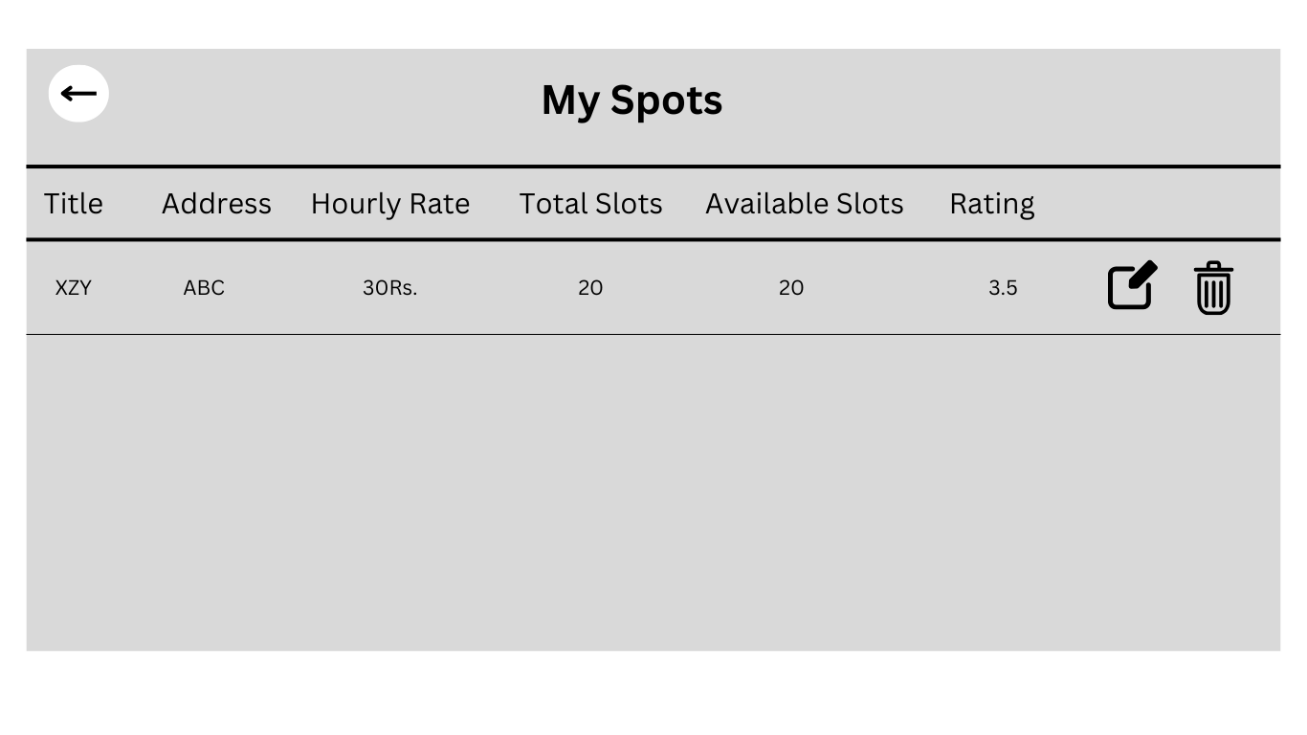


fig 4.10 my spots

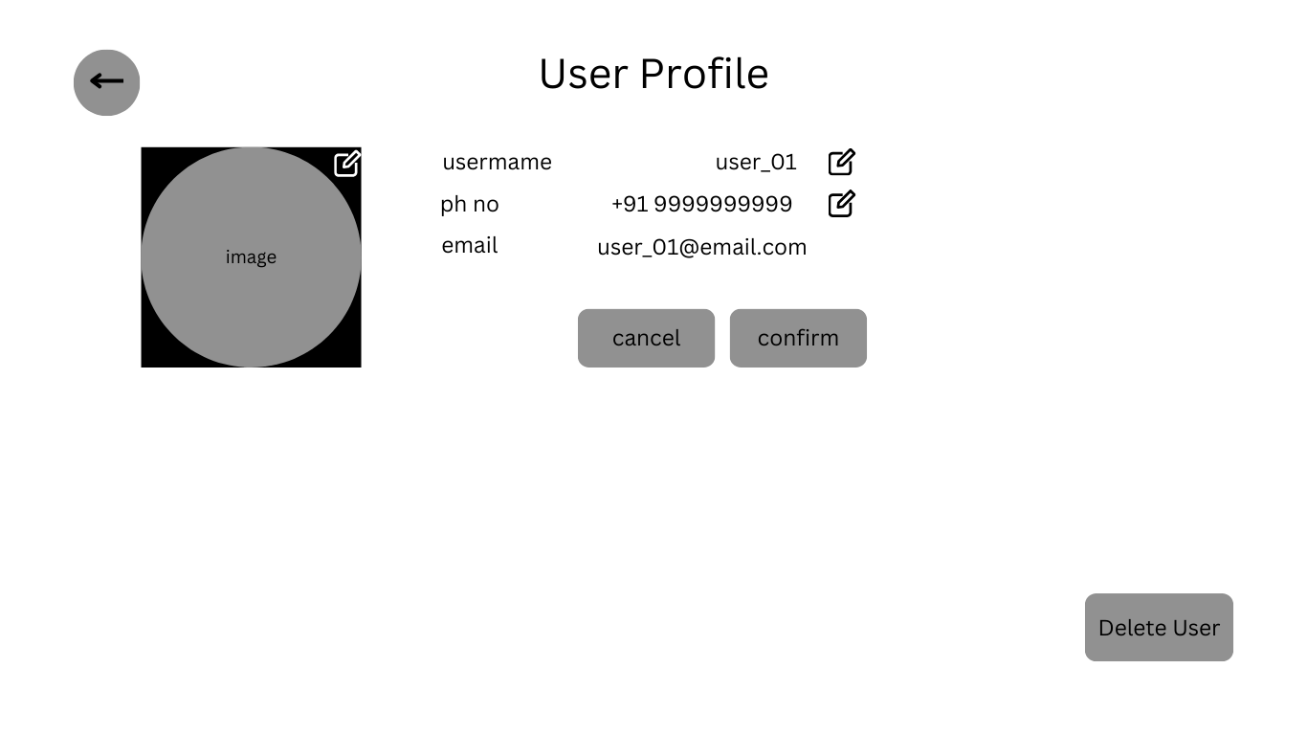


fig 4.11 user profile

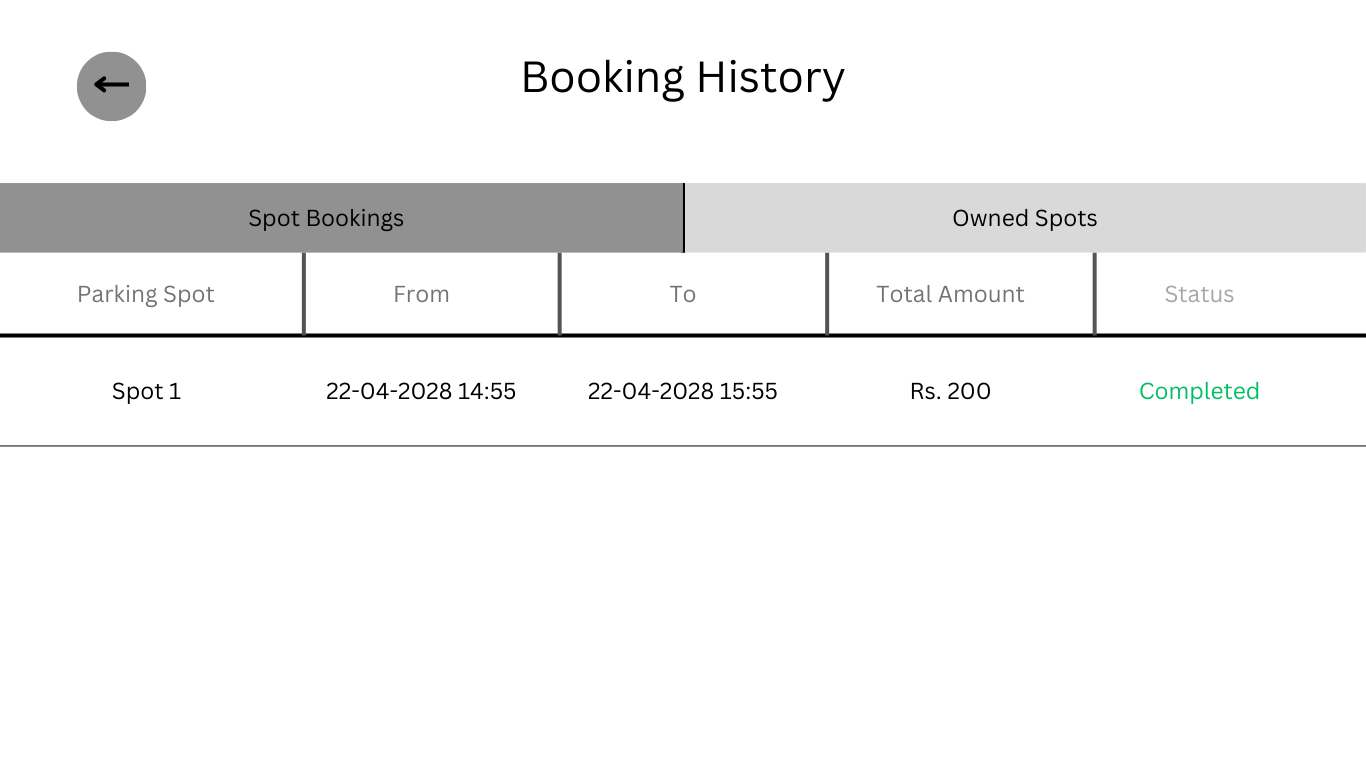


fig 4.12 booking history – user

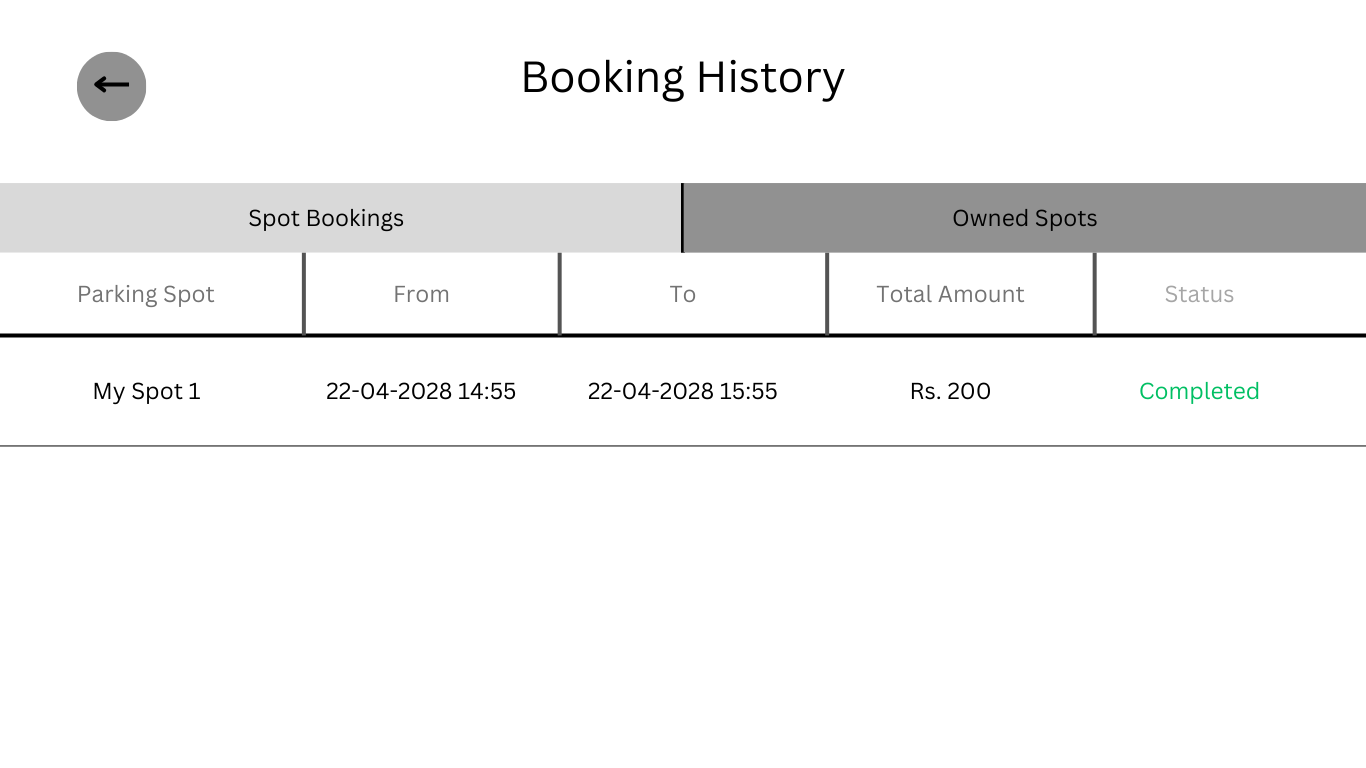


fig 4.13 booking history – owner

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

Description and actual screen shots of the application, pages, screens

{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 }

Feature 1

{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 }

Feature 2

{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 }

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 }