Rapid Cars

-Easy Book and Comfortable Ride

Submitted To,
Faculty of Technology,
Dept. of Computer Engineering,
Dharmsinh Desai University.

DEVELOPED BY: Sayanish Chakraborty, Divyesh Patel & Bharg Trivedi

A

PROJECT REPORT ON

RAPID CARS-

Easy Book and Comfortable Ride

By

DIVYESH PATEL	SAYANISH CHAKRABORTY	BHARG TRIVEDI
(CE-114) (19CEUXS020)	(CE-135) (19CEUON136)	(CE-165) (19CEUON110)

B.Tech CE Semester-VI

Subject: Advanced Technologies

Guided by:

Prof. Shailja H. Patel

Assistant Professor

Dept. of Comp. Engg.

Department of Computer Engineering

Faculty of Technology

Dharmsinh Desai University

Nadiad, Gujarat

Department of Computer Engineering
Faculty of Technology
Dharmsinh Desai University
Nadiad, Gujarat

CERTIFICATE

This is to certify that the practical / term work carried out in the subject of **Advanced Technologies**
And recorded in this journal is the

Bonafide work of

DIVYESH PATEL	**SAYANISH CHAKRABORTY**	**BHARG TRIVEDI**
(CE-114) (19CEUXS020)	**(CE-135) (19CEUON136)**	**(CE-165) (19CEUON110)**

Of B.Tech semester **VI** in the branch of **Computer Engineering**

During the academic year **2021-2022**.

Guided by:
Prof.Shailja H. Patel
Assistant Professor,
Dept. of Computer Engg.
Faculty of Technology
Dharmsinh Desai University, Nadiad

Head of Department:
Dr. C. K. Bhensdadia,
Head of Dept,
Dept. of Computer Engg.
Faculty of Technology
Dharmsinh Desai University, Nadiad

Table of Contents:

1 Abstract.....4

2 Introduction

2.1 Project Details: Brief Introduction 5

2.2 Technology and Tools Used 6

3 Software Requirement Specifications

3.1 System Functional Requirement 7

3.2 System Non-Functional Requirement.....11

4 Design

4.1 Use Case Diagram.....12

4.2 State Diagram.....13

4.3 E-R Diagram 14

4.4 Class Diagram 15

4.5 Sequence Diagram 16

4.6 Activity Diagram 17

4.7 Data-Dictionary Diagram

4.7.1 User Data Record.....18

4.7.2 Car Data Record.....18

4.7.3 Booking Data Record.....19

5. Implementation Details

5.1 Component for User

5.1.1 User Registration and Login.....20

5.1.2 Navbar.....22

5.1.3 Home.....23

5.1.4 Booking.....24

5.1.5 My Booking.....25

5.2 Components for Admin

5.2.1 Car Details.....26

5.2.2 Register Car.....27

5.2.3 Booked Car.....28

6 Testing

6.1 User Pages.....29

6.2 Admin Pages.....30

7 Screenshots of the System

7.1 User View.....34

7.2 Admin View.....40

7.3 Mobile View.....43

8 Conclusion 44

9 Limitations and Future Extensions of System 45

10 Bibliography 46

1. Abstract

"RAPID CARS" is an online Car Rental website platform which is a business website which set for renting cars for the Customers to different locations on the basis of their interest and desire.

Here, Customers can register themselves and then book their fares, can go through their reservations and also can cancel their rides and fares if necessary. Also have the Customer Care Services as in case of any inquiry or inconvenience.

Hereby, Enhancing Business Processes by incorporating Internet Technology with Rental Company to take business online and make it available to many places instead of limiting their services to their local domain alone, thus increases their Return of Investment (ROI).

2. Introduction

2.1 Brief Introduction

Our aim in this project is to produce a web-based system that allows customers to register and reserve Self-Drive Cars online to effectively manage their Car Rental Business.

Also specializing it with automating the work performed in the car rental management system like generating daily bookings, records of car or cab available for booking, record of routes, rental charges for cars for every rout, store record of the customer, etc.

This Website will be user-friendly, and will be made with a responsive layout so that it can be used on any type of device and ease customer's task whenever they need to rent a car.

2.2 Tools/Technologies Used:

Technologies:

- React
- Node.js
- MongoDB
- Bootstrap
- HTML
- CSS

Tools:

- Visual Studio Code
- Post man
- Git and GitHub

Platform:

- Local development server

3. Software Requirement Specifications

R1. Registration:

R.1.1. Registration for Users:

Description: Users who are renting car can sign in for the first time. For Signup user needs set that he/she is a customer and then enter their following details for this website.

Input: Enter required details

Output: User's data is stored in database

R.1.2. Confirmation Message:

Description: When user is registered a conformation message is send to them

Output: Conformation message is received by user in form of Email

R2. Login & Logout:

R.2.1. Login:

Input: Enter Username and Password

Output: If correct then Redirected to Home page if User/ Redirected to Admin Page if an Admin and if incorrect then invalid message is shown.

R.2.1.1. Forget Password:

Input: Recovery Email

Output: OTP Generated and sent via Email, which needs to be entered in the OTP box provided on our Webpage.

R.2.2. Logout:

Input: User Input

Output: Logout from website

R3. Reservation:

R.3.1. Booking Car/s:

Description: Users can book their fares as per their desire.

Input: Choose car from the given options

Output: Redirected to booking confirmation

R.3.2. Cancel Booking:

Description: Users can cancel booking before 2 hr of the journey to begin if they want to.

Input: Cancel the booking by giving booking_id.

Output: Cancel booking message shown

R.3.3. Confirmation Message:

Description: Users get a confirmation message after their booking is successful through Email.

Output: Booking Confirm message sent to users.

R4. Manage Cars:

R.4.1. Adding Cars Details:

Description: Admin can add car to the website

Input: User Input

Output: Car Added in Database and will be displayed on the List of Cars in Webpage.

R.4.2. Updating Car Status:

Description: Car status is updated by the admin or system if it is under repair or being already booked

Output: Car Details will not be displayed to the user for booking.

R5. Explore your Class:

R.5.1. Sort for Ride:

Description: Sort results based on Customer Interest.

Input: User Selection

Output: Your Car details will be displayed as per the order selected by the Customer.

R.5.2. Filter Results:-

Description: Filter your choice based on keys specified.

Output: Display results based on Customer Feedbacks and Filters applied by User.

R6. Bill Generation:

Description: After confirmation, Token will be generated for payment according to the reserved fare and will be sent including confirmation booking message.

Input: User Input

Output: Bill Receipt

3.2 Other Nonfunctional Requirements:

1. Performance

This Website will be user-friendly, easy to use and will be made with responsive layout so that it can be used in any type of device.

2. Safety

Only Authorized Admins have the access and can add some data and details of some new cars, update the status of cars and can also maintain history, etc.

3. Database

System requires to access user versions, reviews and profile data fast to maintain the performance.

4. Design

4.1 Use Case Diagram:

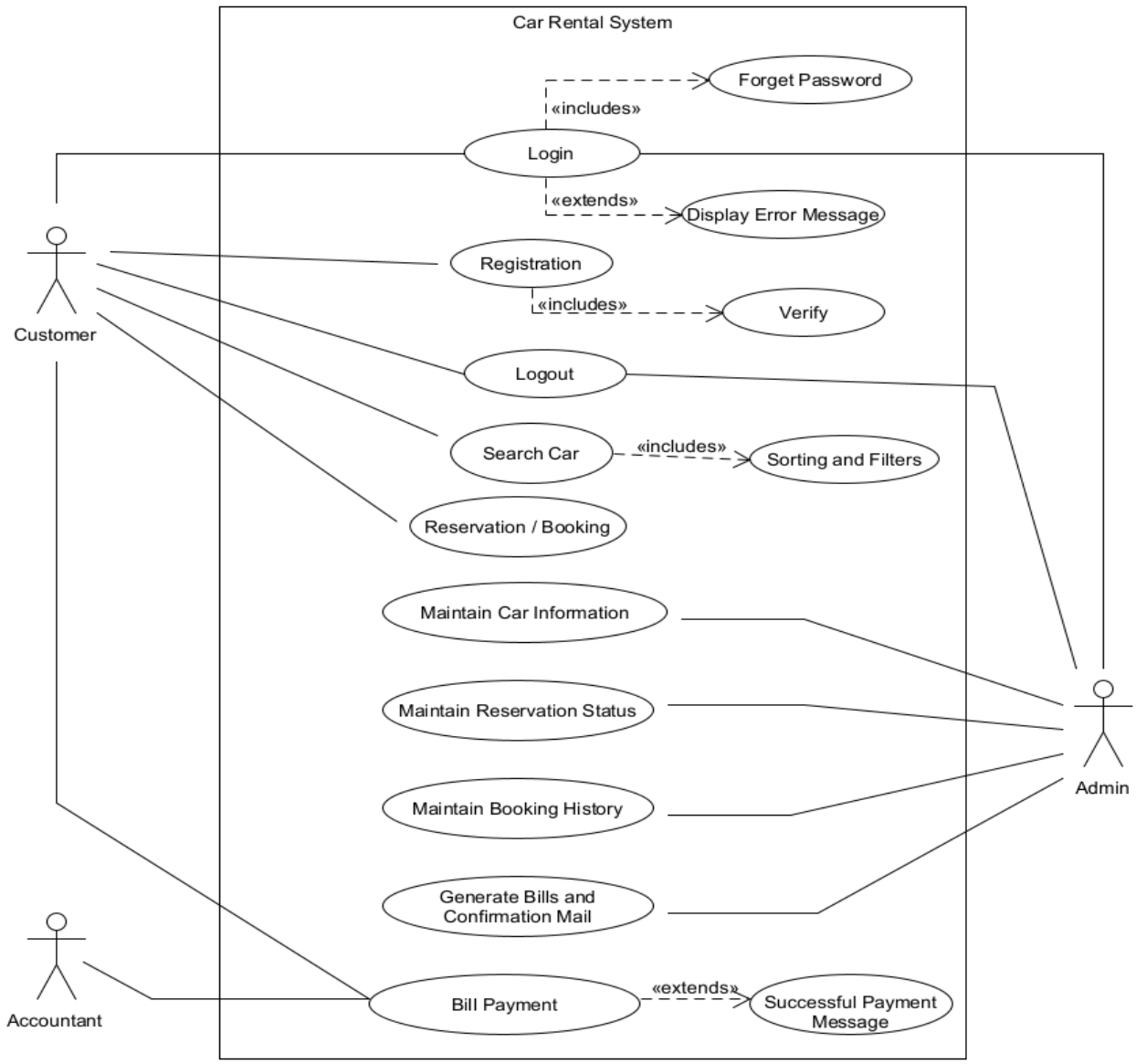


Fig. 4.1
Use Case Diagram

4.2 State Diagram:

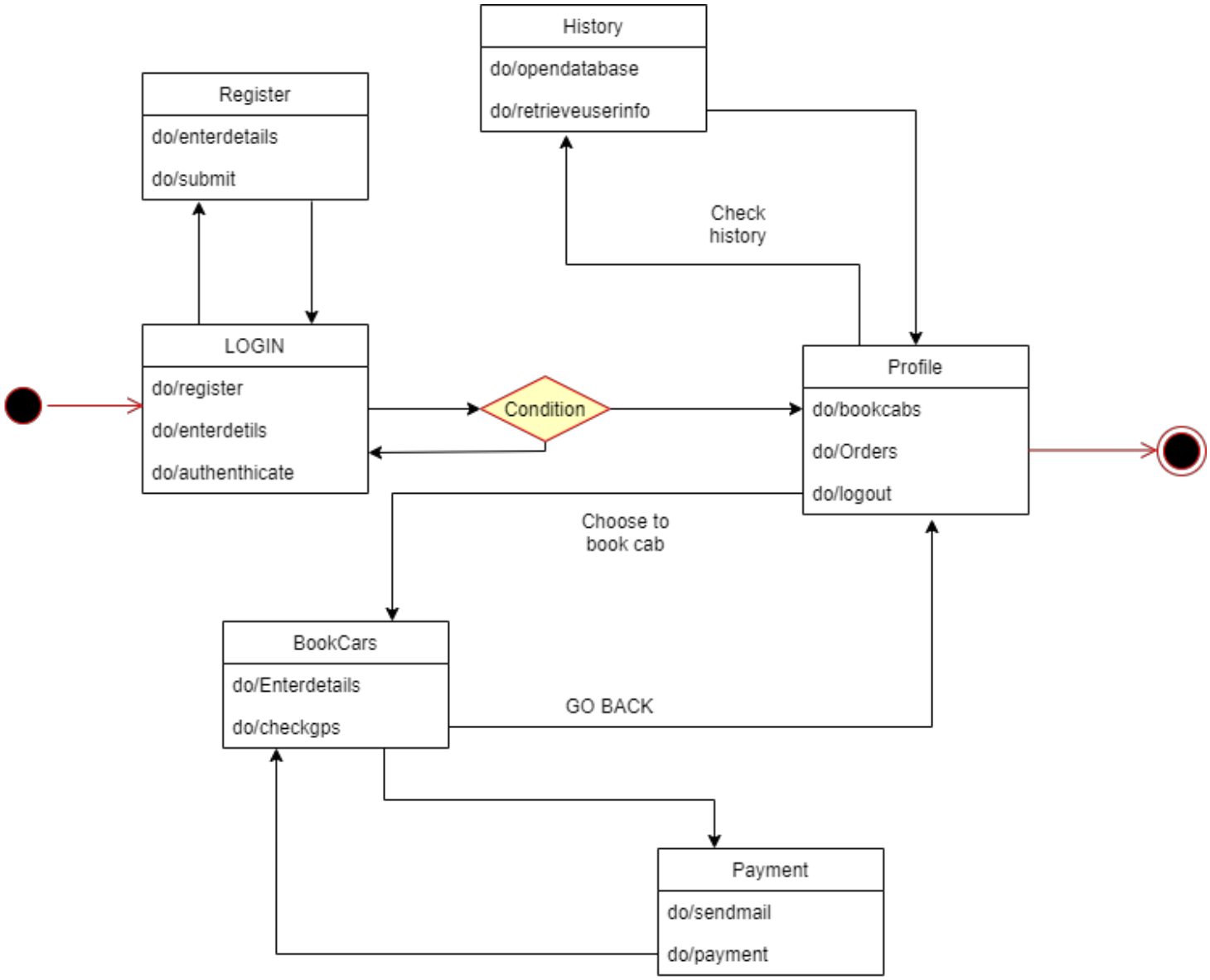


Fig. 4.2
State Diagram

4.3 E-R Diagram:

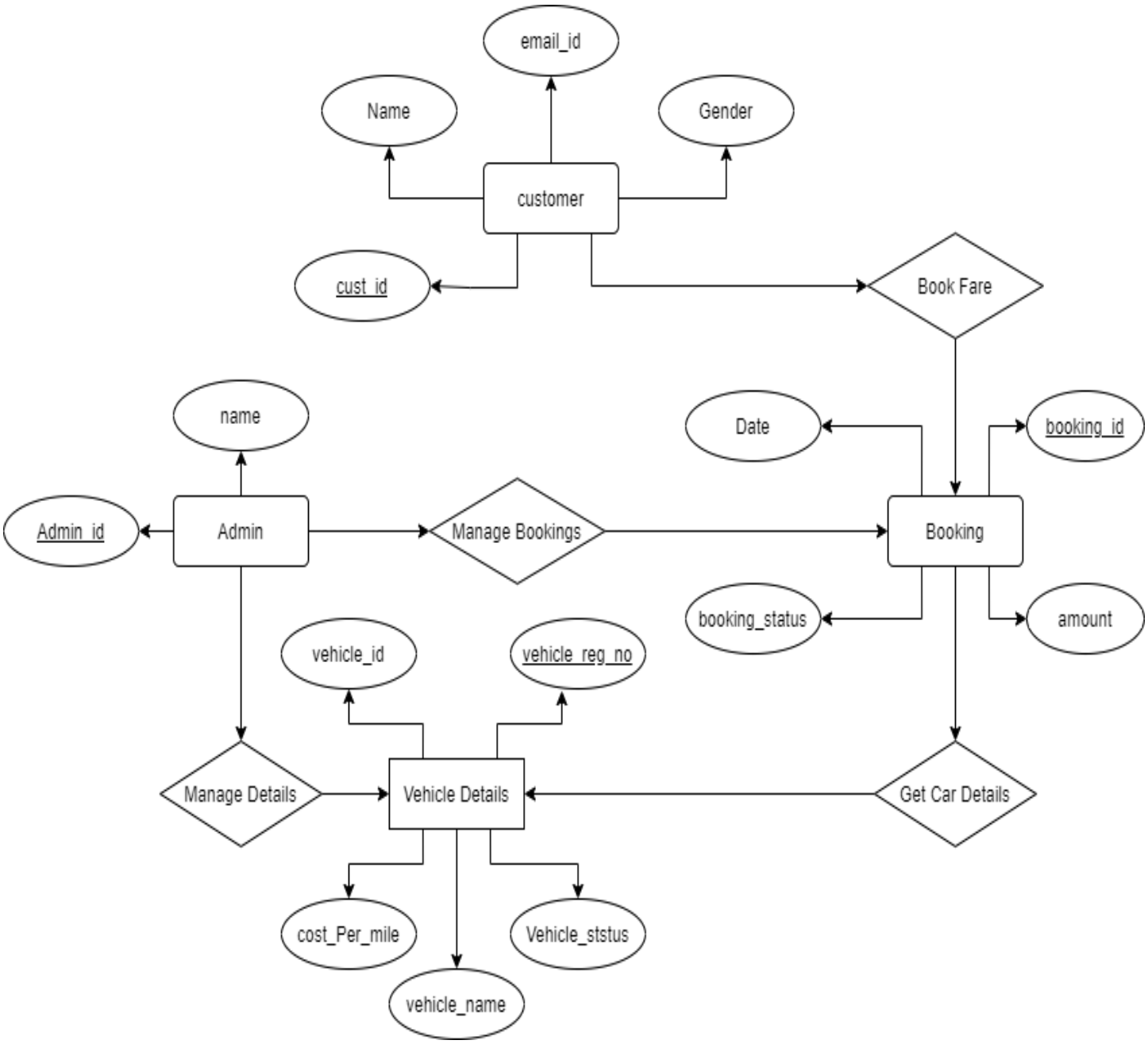


Fig. 4.3
ER Diagram

4.4 Class Diagram:

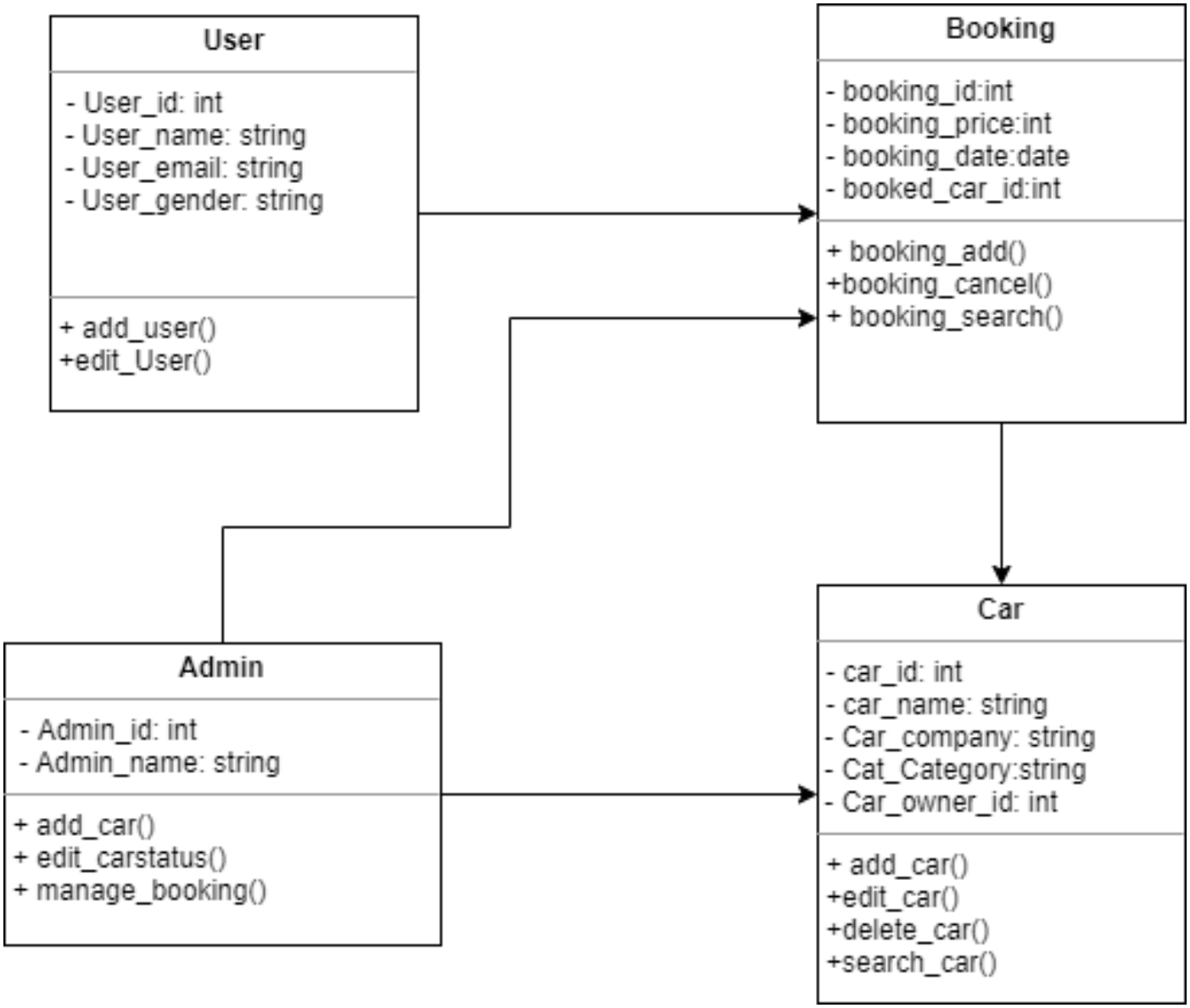


Fig. 4.4
Class Diagram

4.5 Sequence Diagram:

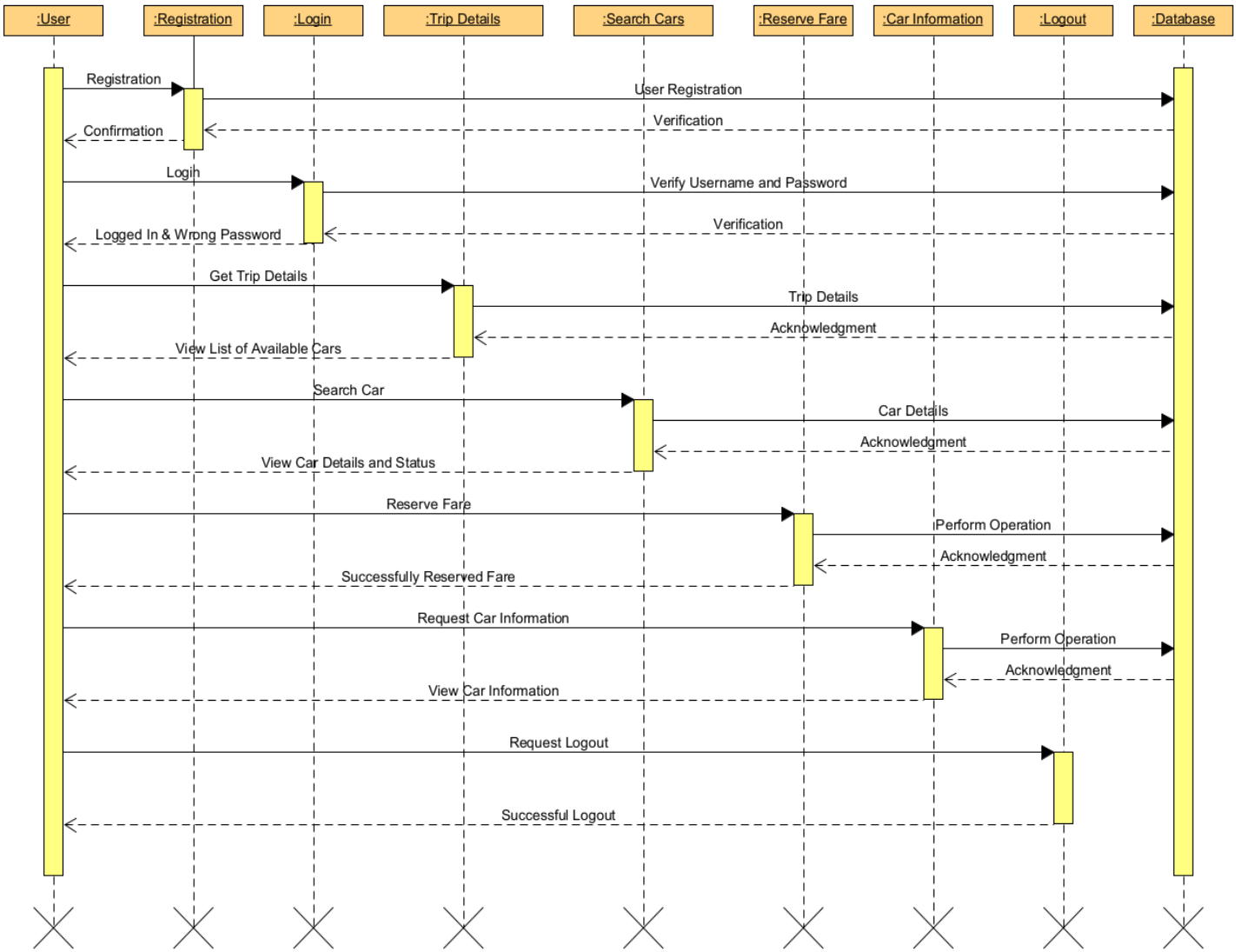


Fig. 4.5
Sequence Diagram

4.6 Activity Diagram:

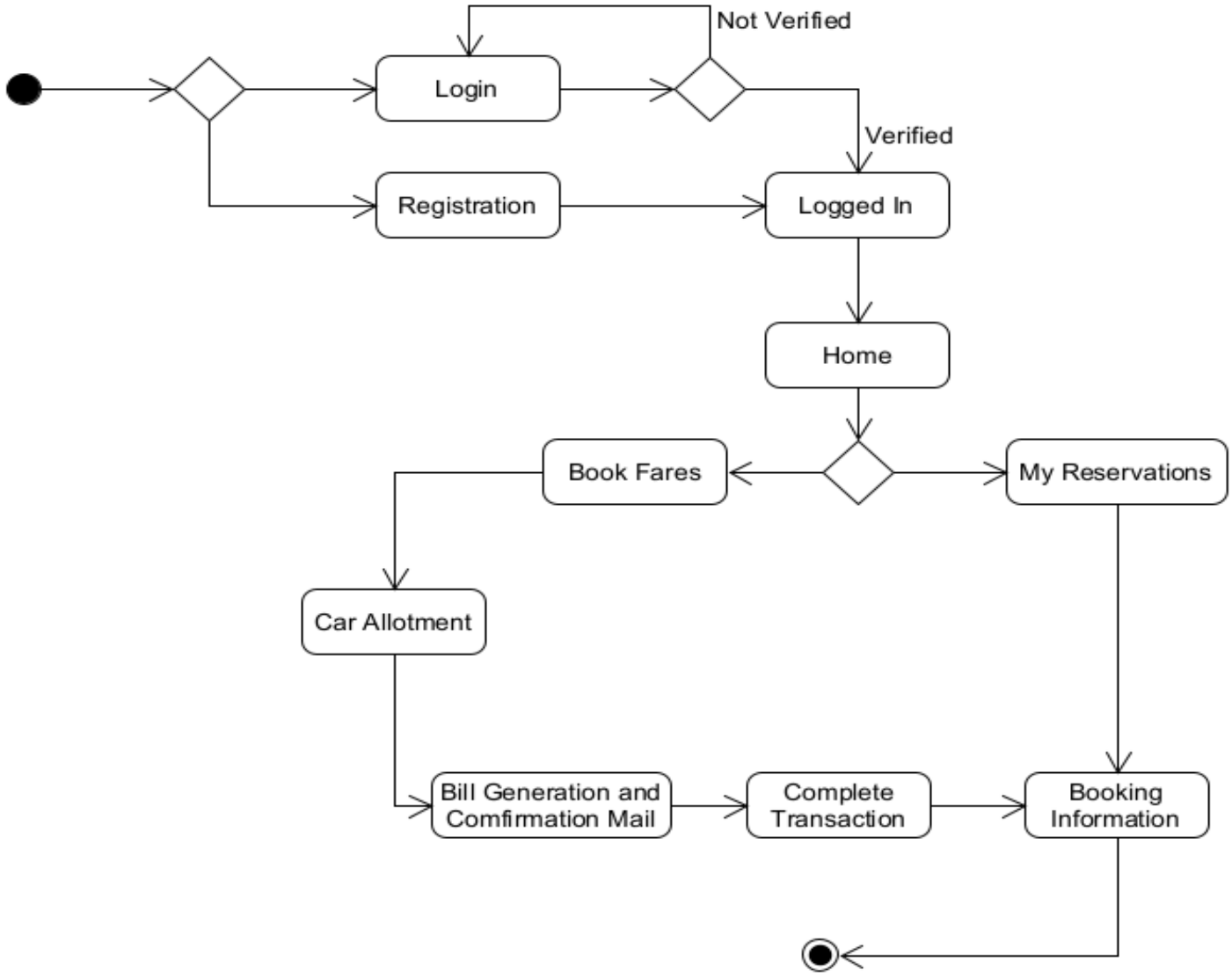


Fig. 4.6
Activity Diagram

4.7 Data Dictionary Diagram:

4.7.1 Data-Dictionary -1

User Data Record								
Sr No.	**Field Name**	**Data Type**	**Field Size**	**Required**	**Unique**	**PF/FK**	**Referred Table**	**Description**
1	ID	number	-	True	Yes	Yes	-	Auto Increment
2	Username	varchar	20	True	Yes	No	-	Username of User
3	Email	varchar	20	True	Yes	No	-	Email Address of User
4	Mobile	varchar	10	True	Yes	No	-	Contact Info. Of User
5	Password	varchar	50	True	No	No	-	Key for Authentication

Table 4.1
Data Dictionary 1

4.7.2 Data-Dictionary -2

Car Data Record								
Sr No.	**Field Name**	**Data Type**	**Field Size**	**Required**	**Unique**	**PF/FK**	**Referred Table**	**Description**
1	ID	number	-	True	Yes	Yes	-	Auto Increment
2	Cname	varchar	20	True	No	No	-	Name of Car
3	Carno	varchar	20	True	Yes	No	-	Car Registration Number
4	Cartype	varchar	10	True	No	No	-	Type of Car
5	Fuel	varchar	50	True	No	No	-	Type of Fuel
6	Mileage	number	5	True	No	No	-	Mileage of Car
7	Gear Type	varchar	10	True	No	No	-	Type of Gear Car uses
8	Company	varchar	10	True	No	No	-	Company of the Car
9	Seat	number	5	True	No	No	-	Number of Seats
10	Cpurchase	date	-	True	No	No	-	Date of Car Purchase
11	Cpirce	number	10	True	No	No	-	Price of Car
12	Cimage	varchar	-	True	No	No	-	Image of the Car
13	Isbook	Boolean	1	True	No	No	-	Check Car book status
14	Location	varchar	20	True	No	No	-	Location of the car where it's Available

Table 4.2
Data Dictionary 2

4.7.3 Data-Dictionary -3

Booking Data Record								
Sr No.	**Field Name**	**Data Type**	**Field Size**	**Required**	**Unique**	**PF/FK**	**Referred Table**	**Description**
1	ID	number	-	True	Yes	Yes	-	Booking ID Auto Increment
2	User_ID	varchar	20	True	Yes	Yes	-	User ID who books his/her Trip
3	Car_ID	varchar	20	True	Yes	Yes	-	Car ID of the car booked by the User
4	User_db	varchar	20	True	Yes	Yes	User Data Record	Specifies Name of Database to which it belongs
5	Car_db	varchar	20	True	Yes	Yes	Car Data Record	Specifies Name of Database to which it belongs
6	From Location	varchar	20	True	No	No	-	Pickup Point
7	To Location	varchar	20	True	No	No	-	Destination
8	From Date	date	-	True	No	No	-	Pickup Date
9	ToDate	date	-	True	No	No	-	Return Date
10	Bill Amount	number	10	True	No	No	-	Amount of the Complete Trip
11	IsRunning	Boolean	1	True	No	No	-	Check whether trip is ongoing or has ended
12	Hours	varchar	5	True	No	No	-	Total hours of the Trip

Table 4.3
Data Dictionary 3

5. Implementation Details

5.1 Component for User:

5.1.1 User Registration and Login:

Information of new user is taken and stored to the user's database. Whenever that user will login then fetch data from database and compare for login.

⇨ **Login Component**

```
8   export default function Login( {setloginuser } ) {
9
10      const history = useHistory()
11      const [user, setUser] = useState({
12          email: "",
13          password: ""
14      })
15
16
17      function login() {
18
19          if (validate_input(user) == true) {
20
21              console.log("in posrt")
22
23              axios.post("http://localhost:8000/api/signin", user)
24                  .then(res => {
25
26                      console.log(res.data.message.email);
27                      setloginuser(res.data.message);
28                      window.localStorage.setItem("user_login", "yes")
29                      window.localStorage.setItem("user" , JSON.stringify(res.data.message))
30
31                      if (res.data.message.email === "rapidcars123@gmail.com"){
32
33                          history.push({
34                              pathname :'/cardetails',
35                          });
36                      }
37                      else{
38
39                          console.log("ddfsd")
40
41                          history.push('/')
42
43                      }
44
45                  })
46                  .catch(err => {
47
48                      console.log(err.data);
```

⇨ Sign-Up Component

```
35        e.preventDefault();
36
37        if( validate_input(car) == true)
38        {
39
40          var Cardata = new FormData();
41          Cardata.append("file", file);
42          for (var key in car) {
43            Cardata.append(key, car[key]);
44          }
45
46          const config = {
47            Headers: {
48              "content-type": "multipart/form-data",
49            },
50          };
51
52
53
54            axios.post("http://localhost:8000/api/carregister", Cardata , config)
55            .then( res => {
56                console.log("valid input  1233")
57                console.log(res.data.result)
58                history.push('/cardetails');
59
60            })
61            .catch(err => {
62
63                console.log(err);
64            })
65        }
66        else
67        {
68            console.log("invalid input");
69        }
70
71      }
72
```

5.1.2 Navbar:

Navigation bar can redirect one to another page and it has different view for user and admin.

```
import React from 'react'
import './navbar.css'
import { Link } from 'react-router-dom'
import { Redirect } from 'react-router-dom'
import { useHistory } from 'react-router-dom'


export default function Navbar( props ) {



const history = useHistory()

  function myFunction() {


    var x = document.getElementById("myTopnav");
    if (x.className === "topnav") {
      x.className += " responsive";
    } else {
      x.className = "topnav";
    }
  }

  function redirect(){

    var auth = document.getElementById('logout').innerHTML;
    console.log(auth)
    document.getElementById("remove-button").innerHTML = ""
    if(auth == "Login")
        history.push('/login');
    else{
      window.localStorage.setItem('user_login', 'No')
      window.localStorage.removeItem('user')
      props.setloginuser({});
      history.push('/');
    }

```

5.1.3 Home:

Home is user friendly and responsive home page which gives our users a brief introduction on our website and how it works.

```
</section>

<section id="featured-services" class="featured-services">
    <div className="container" data-aos="fade-up">
        <div className="section-title">
            <h2>How we work</h2>
            {/* <h3>Find Out More <span>About Us</span></h3> */}
            {/* <p>Below we have shown the small .</p> */}
        </div>
        <div className="row">
            <div className="col-md-6 col-lg-3 d-flex align-items-stretch mb-5 mb-lg-0">
                <div class="icon-box" data-aos="fade-up" data-aos-delay="100">
                    <div class="icon"><i class="bi bi-search"></i></div>
                    <h4 class="title"><a href="">Find your car</a></h4>
                    <p class="description">Search the car which you want for booking and select book button</p>
                </div>
            </div>

            <div class="col-md-6 col-lg-3 d-flex align-items-stretch mb-5 mb-lg-0">
                <div class="icon-box" data-aos="fade-up" data-aos-delay="200">
                    <div class="icon"><i class="bi bi-box-arrow-in-right"></i></div>
                    <h4 class="title"><a href="">Enter your route</a></h4>
                    <p class="description">Enter your starting and ending location with journey time and then you can book the
                </div>
            </div>

            <div class="col-md-6 col-lg-3 d-flex align-items-stretch mb-5 mb-lg-0">
                <div class="icon-box" data-aos="fade-up" data-aos-delay="300">
                    <div class="icon"><i class='fas fa-car'></i></div>
                    <h4 class="title"><a href=""> Get your car</a></h4>
                    <p class="description">After booking get the car from our store for the ride.</p>
                </div>
            </div>

            <div class="col-md-6 col-lg-3 d-flex align-items-stretch mb-5 mb-lg-0">
                <div class="icon-box" data-aos="fade-up" data-aos-delay="400">
                    <div class="icon"><i class="bi bi-geo-alt"></i></div>
                    <h4 class="title"><a href="">Return car</a></h4>
                    <p class="description">Return the car after your journey is over after your specified time.</p>
                </div>
```

5.1.4 Booking:

Users are allowed to book cars with their own choice .They can sort and filter the available cars and after entering date and location they can book the car of their own choice.

⇨ Booking Component

```
import React from 'react'
import './booking.css'
import axios from 'axios';

import { useHistory } from 'react-router-dom';

import { useState } from 'react/cjs/react.production.min';
// import image from './../../public/img/download.jpg';

export default function Booking(props) {

    const history = useHistory();

    const [cardata, setData] = React.useState(null);
    const [car, setcar] = React.useState(null);


    React.useEffect(() => {

        axios.get("http://localhost:8000/api/cars")
            .then(res => {
                console.log("valid input  1233")
                // setData(res.data);
                setData(res.data.filter((val) => {
                    return val.IsBook === false
                }))

                setcar(res.data.filter((val) => {
                    return val.IsBook === false
                }))


                console.log(res.data.errors);
            })
            .catch(err => {
                console.log("hello======")
                console.log(err.data);
            })

```

5.1.5 My Booking:

User can see his ongoing booking history in this page .They can also cancel their ongoing booking.

⇨ My Booking Component

```
import React from 'react'
import './mybooking.css'
import axios from 'axios';


import { useHistory } from 'react-router-dom';

import { useState } from 'react/cjs/react.production.min';
// import image from './../../public/img/download.jpg';

export default function Booking(props) {

   const history = useHistory();

    const [mybooking, setData] = React.useState(null);
    var TotalPrice = 0;

    React.useEffect(() => {
        var user_id = props.user_id
        //console.log(user_id)
        axios.post("http://localhost:8000/api/bookingdata" , {user_id})
            .then(res => {
                console.log("valid input  1233")
                setData( res.data.filter( (val) => {
                    return val.booking.IsRuning == true
                } ) )

                console.log(res.data);
            })
            .catch(err => {
                console.log("hello======")
                console.log(err.data);
            })


    }, []);
```

5.2 Components for Admin:

5.2.1 Car Details:

Admin can view details of all cars registered in the website after logging in and can also filter their result according to his/her choice.

```
import React from 'react'
import axios from 'axios';
import { useHistory } from 'react-router-dom';
import { useEffect } from 'react';
import { useState } from 'react';
import './CarsDetails.css'
export default function CarsDetails() {

    const history = useHistory();
    const [cardetails, setData] = React.useState(null);
    const [cars, setcar] = React.useState(null);
    React.useEffect(() => {

        axios.get("http://localhost:8000/api/cars")
            .then(res => {
                console.log("valid input  1233")
                setcar(res.data);
                setData(res.data);

                console.log(res.data);
            })
            .catch(err => {
                console.log("hello======")
                console.log(err.data);
            })
        console.log('karm patel')


    }, []);


```

```
            var html = <a className="delete" title="Delete" data-toggle="tooltip" onClick={() => {

                const confirmBox = window.confirm(
                    "Do you really want to Remove!  " + car.cname
                )
                if (confirmBox === true) {
                    RemoveCar(car._id )
                }
               }
                }><i class="fa fa-trash"></i></a>
        }
```

5.2.2 Register Car:

Admin can register a new car by entering the required information about the car. And the car will be shown in the in car details page and booking page of user side for booking.

```
});

const history = useHistory();

const [file, setFile] = useState(null);

const handlefile = (e) => {
  setFile(e.target.files[0]);
};


const register = (e) => {

  e.preventDefault();

  if( validate_input(car) == true)
  {

    var Cardata = new FormData();
    Cardata.append("file", file);
    for (var key in car) {
      Cardata.append(key, car[key]);
    }

    const config = {
      Headers: {
        "content-type": "multipart/form-data",
      },
    };



      axios.post("http://localhost:8000/api/carregister", Cardata , config)
      .then( res => {
          console.log("valid input  1233")
          console.log(res.data.result)
          history.push('/cardetails');

      })
      .catch(err => {
```

5.2.3 Booked Car:

Admin can see the booking of all the cars until now and also the ongoing booking. Only Admin can complete the booking after the user return the car. Admin can also filter the booked car into ongoing booking and previous booking.

```
export default function BookingCars() {

    const history = useHistory();

    const [booking, setbooking] = React.useState(null);
    const [bookingcar, setData] = React.useState(null);


    React.useEffect(() => {

        axios.post("http://localhost:8000/api/systembookingdata")
            .then(res => {
                console.log("valid input  1233")
                console.log(res.data)
                setbooking(res.data)
                setData(res.data);


                console.log(res.data);
            })
            .catch(err => {
                console.log("error")

            })
        console.log('karm patel')


    }, []);



    const booking_history_sort = (isbook) => {
        console.log("in booking" + isbook)

        if (isbook == true) {
            var el = document.getElementById("reserve")
            el.style.color = "white";
            el.style.background = "#299be4";
```

6. Testing

Manual testing was performed in order to find and fix the bugs in development process.

6.1 User Pages:

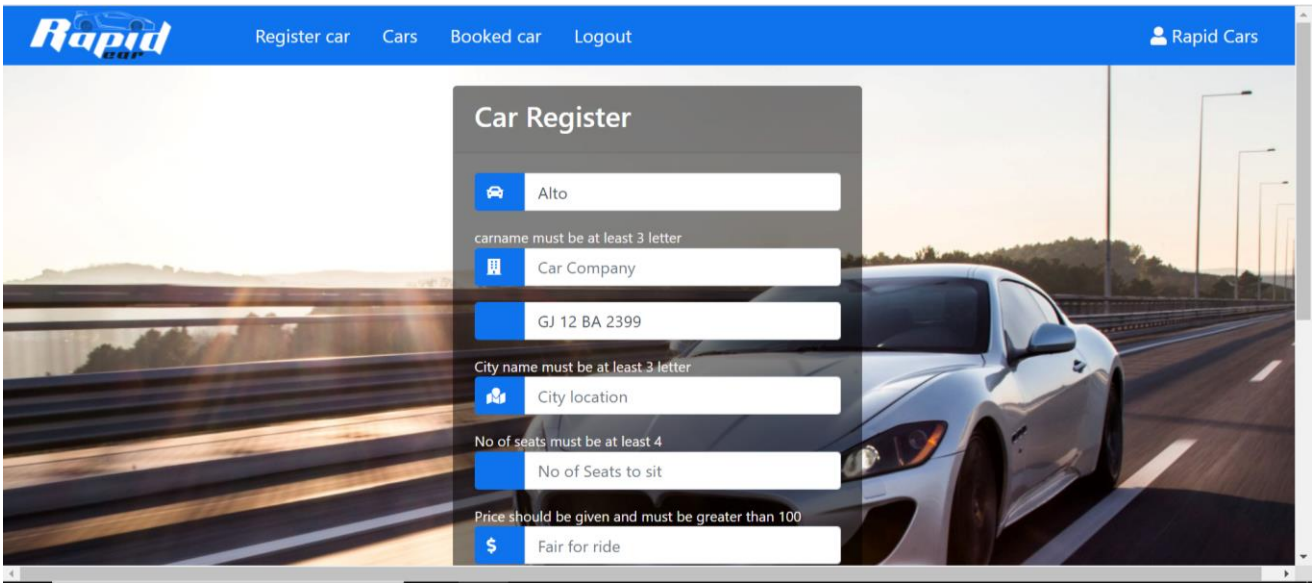
Sr. No.	Test Scenario	Expected Result	Actual Result	Status
1	Open home page.	Information shown.	Proper information is seen.	Success
2	After Sign In.	The user name is shown on the header.	The user name is shown on the header.	Success
3	Open Booking Page.	All unreserved cars are shown.	All unreserved cars are shown and booking can be done.	Success
4	Open My Booking page.	Information about the ongoing booking.	Information about the ongoing booking and cancel booking can be done.	Success

Table 6.1
User Page Testing

6.2 Admin Pages:

Sr. No.	Test Scenario	Expected Result	Actual Result	Status
1	After Sign In.	User name is shown on the header	User name is shown on the header and car details page will be shown.	Success
2	Open Car Details Page.	. Information about all car will be shown	Information about all car will be shown with necessary filter options	Success
3	Open Booked Car Page.	All booked cars shown.	All booked car cars and ongoing book car will be shown with necessary filter options.	Success
4	Open Register Car Page.	Car Register Page is shown.	Car Register Page is shown and admin can register a new car.	Success

Table 6.2
Admin Page Testing



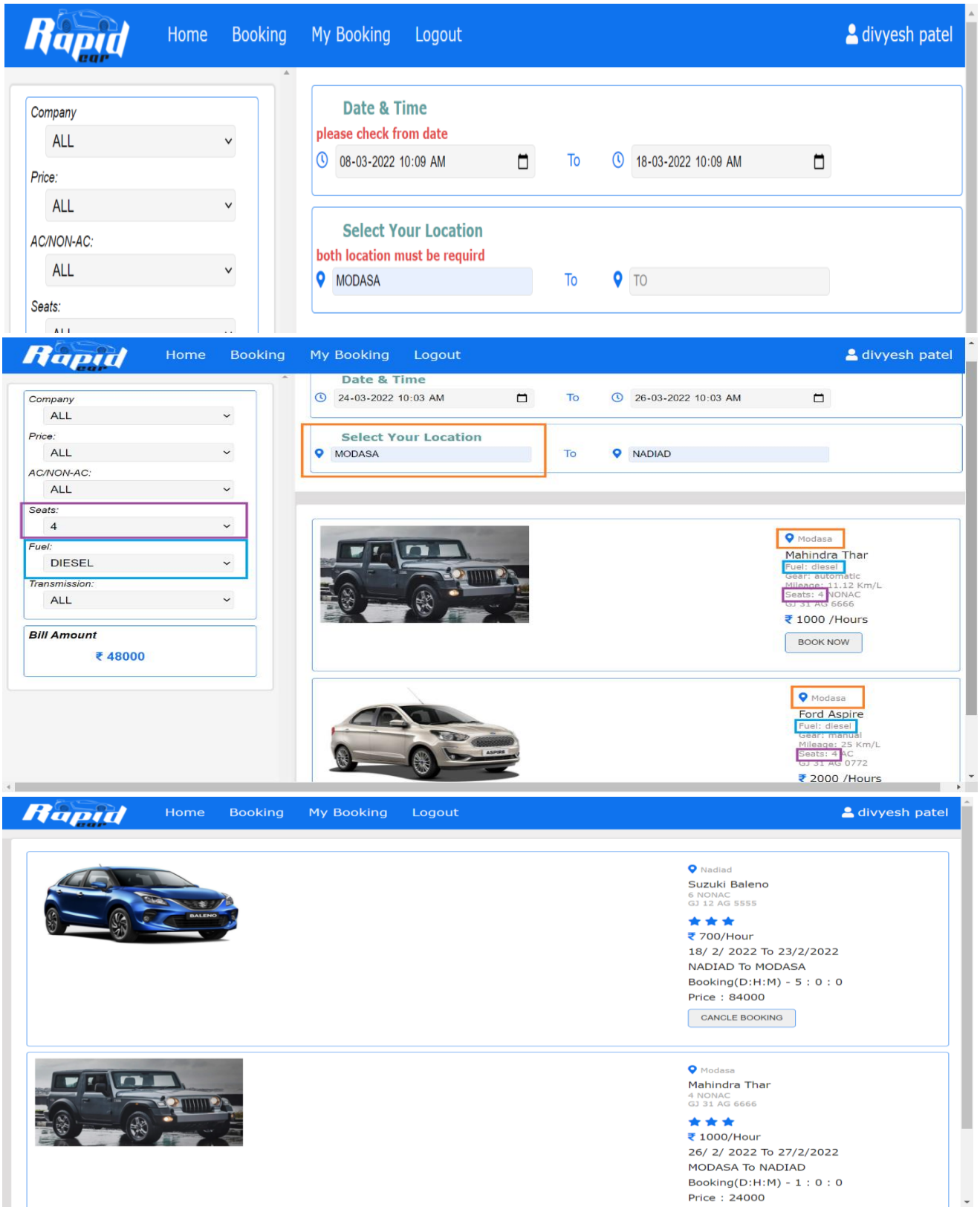


Fig. 6.1
User Page

Rapid car | Register car | Cars | Booked car | Logout | Rapid Cars

Car Details

Running | History | All Car

No	Car	Company	Car No	Cprice	Location	From	To	From Date	To Date	Price	Day:Hrs:Min
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	NADIAd	MODASa	18/ 2/ 2022	23/2/2022	84000	5 : 0 : 0
2	Thar	Mahindra	GJ 31 AG 6666	1000	Modasa	MODASA	NADIAD	26/ 2/ 2022	27/2/2022	24000	1 : 0 : 0

Rapid car | Register car | Cars | Booked car | Logout | Rapid Cars

Car Details

Reserved | Unreserved | All Car

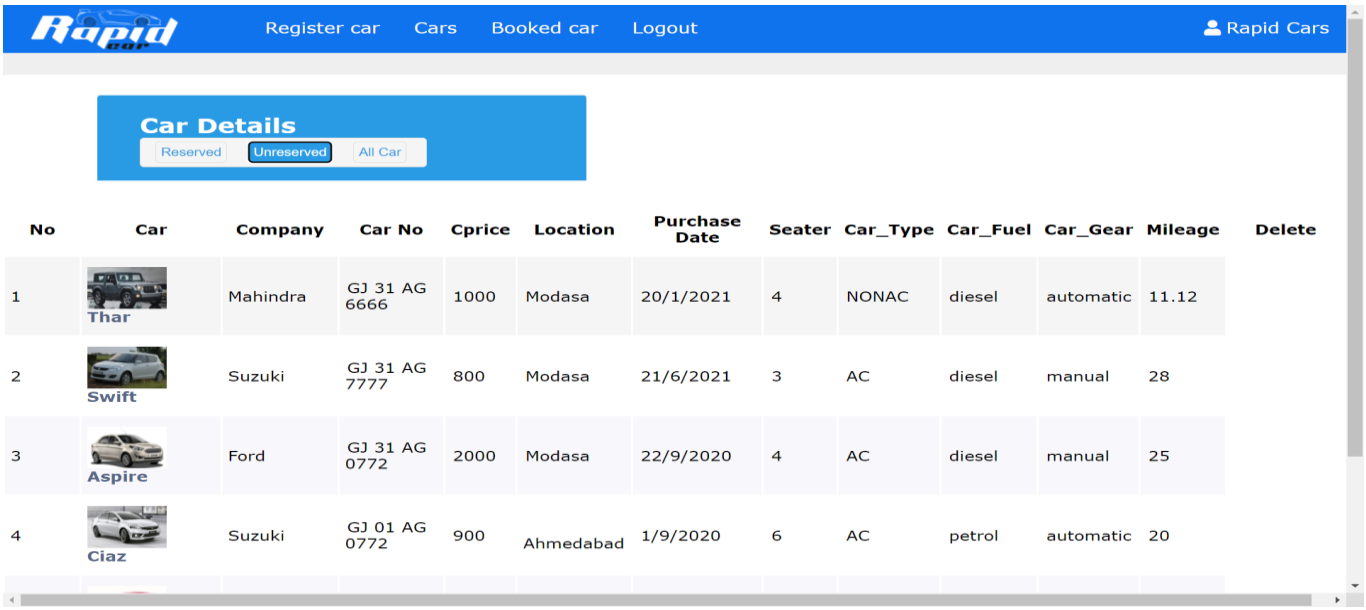
No	Car	Company	Car No	Cprice	Location	Purchase Date	Seater	Car_Type	Car_Fuel	Car_Gear	Mileage	Delete
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	3/1/2021	6	NONAC	petrol	manual	22	
2	Thar	Mahindra	GJ 31 AG 6666	1000	Modasa	20/1/2021	4	NONAC	diesel	automatic	11.12	

Rapid car | Register car | Cars | Booked car | Logout | Rapid Cars

Car Details

Running | History | All Car

No	Car	Company	Car No	Cprice	Location	From	To	From Date	To Date	Price	Day:Hrs:Min
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	bayad	ahmedabad	5/ 2/ 2022	7/2/2022	33600	2 : 0 : 0
2	Ciaz	Suzuki	GJ 01 AG 0772	900	Ahmedabad	ajabpura	modasa	25/ 1/ 2022	5/2/2022	172800	8 : 0 : 0
3	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	ajabpura	unjha	25/ 1/ 2022	5/2/2022	134400	8 : 0 : 0
4	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	bayad	ahmedabad	26/ 1/ 2022	3/2/2022	84000	5 : 0 : 0
5	Swift	Suzuki	GJ 31 AG 7777	800	Modasa	bayad	ahmedabad	26/ 1/ 2022	2/2/2022	76813	4 : 0 : 1
6	Thar	Mahindra	GJ 31 AG 6666	1000	Modasa	ajabpura	unjha	26/ 1/ 2022	27/1/2022	24000	1 : 0 : 0



No	Car	Company	Car No	Cprice	Location	Purchase Date	Seater	Car_Type	Car_Fuel	Car_Gear	Mileage	Delete
1	Thar	Mahindra	GJ 31 AG 6666	1000	Modasa	20/1/2021	4	NONAC	diesel	automatic	11.12	
2	Swift	Suzuki	GJ 31 AG 7777	800	Modasa	21/6/2021	3	AC	diesel	manual	28	
3	Aspire	Ford	GJ 31 AG 0772	2000	Modasa	22/9/2020	4	AC	diesel	manual	25	
4	Ciaz	Suzuki	GJ 01 AG 0772	900	Ahmedabad	1/9/2020	6	AC	petrol	automatic	20	

Fig. 6.2
Admin Page

7. Screenshots

7.1 User View

7.1.1 Home:

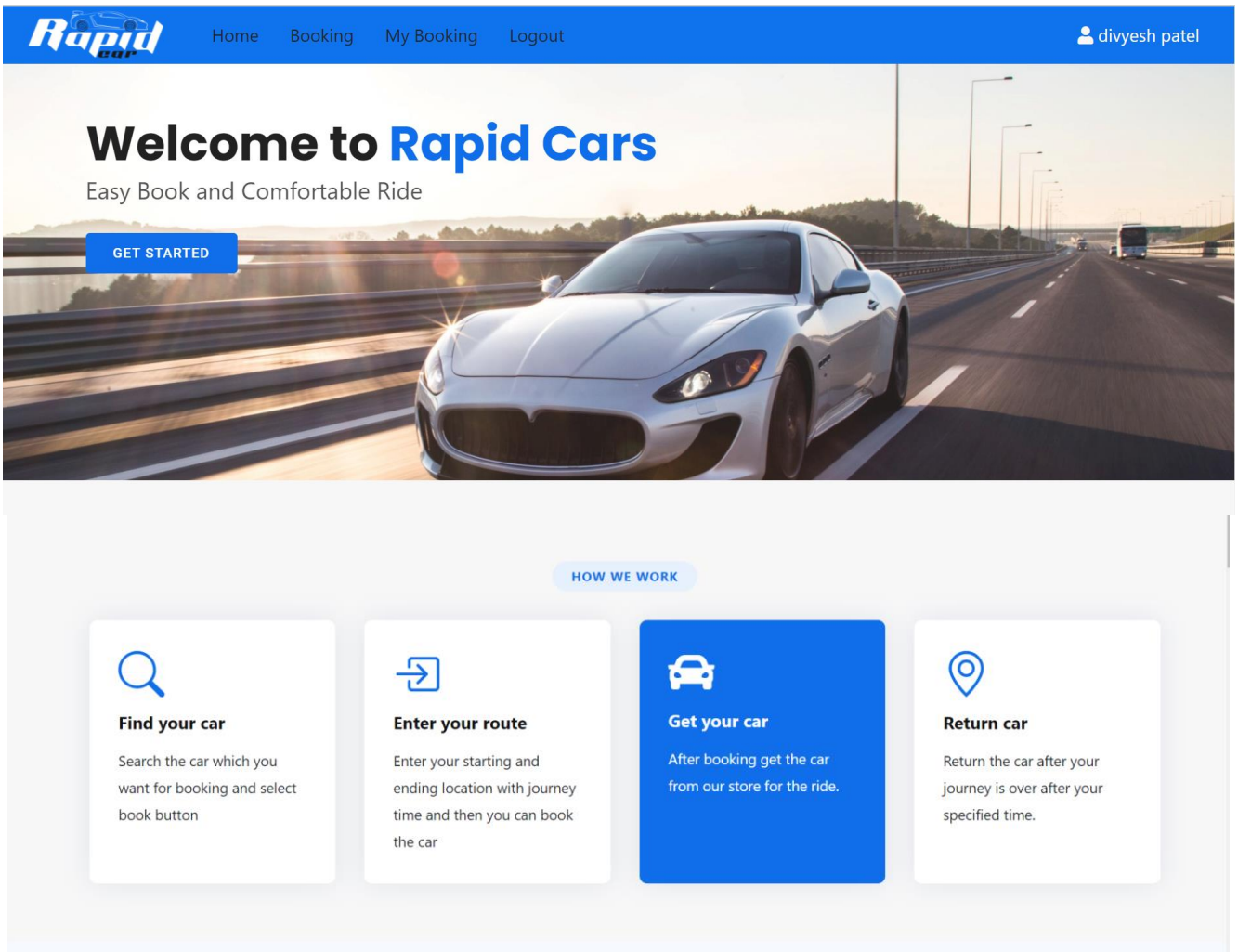


Fig. 7.1
Home Page (i)

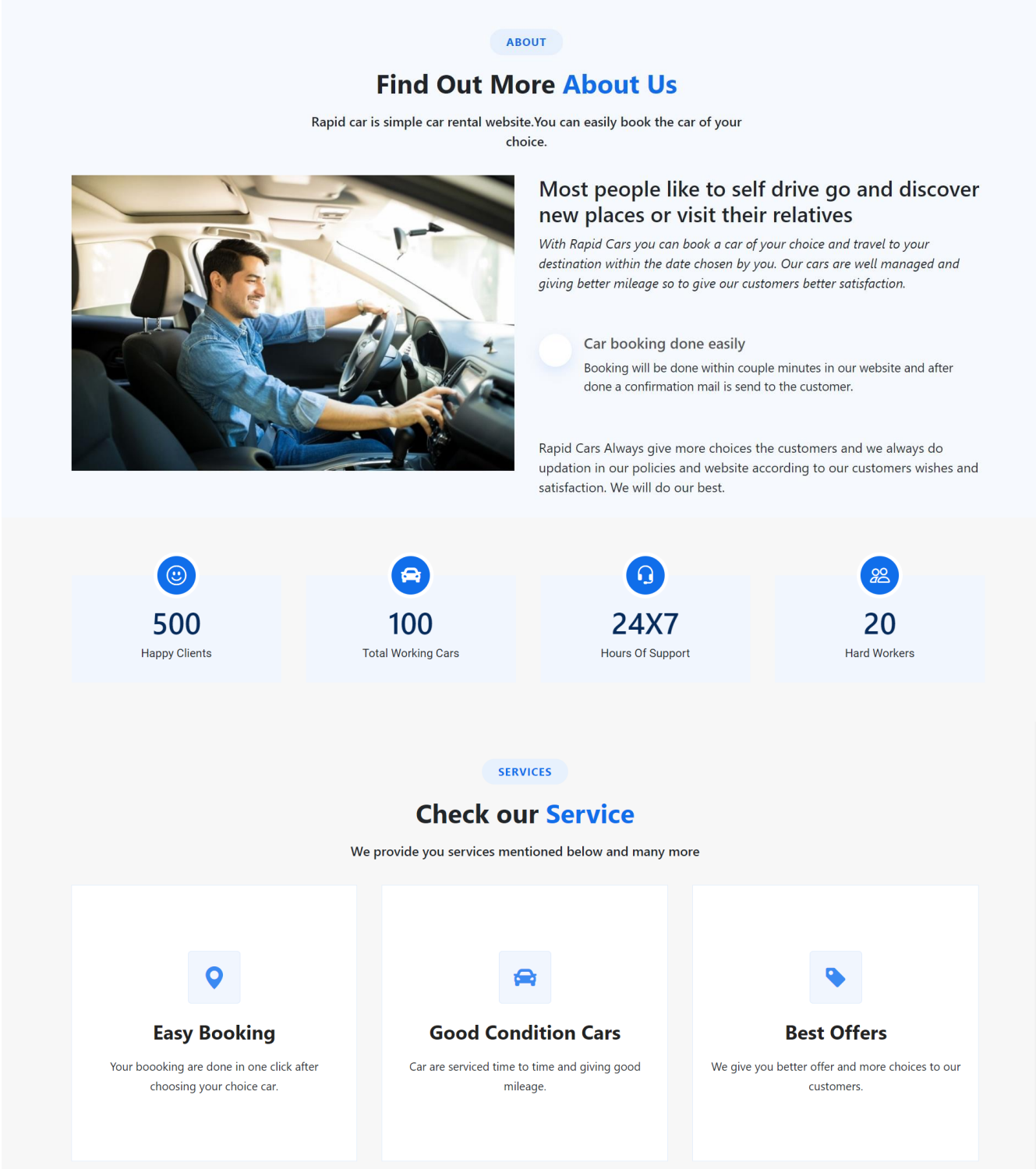


Fig. 7.2
Home Page (ii)

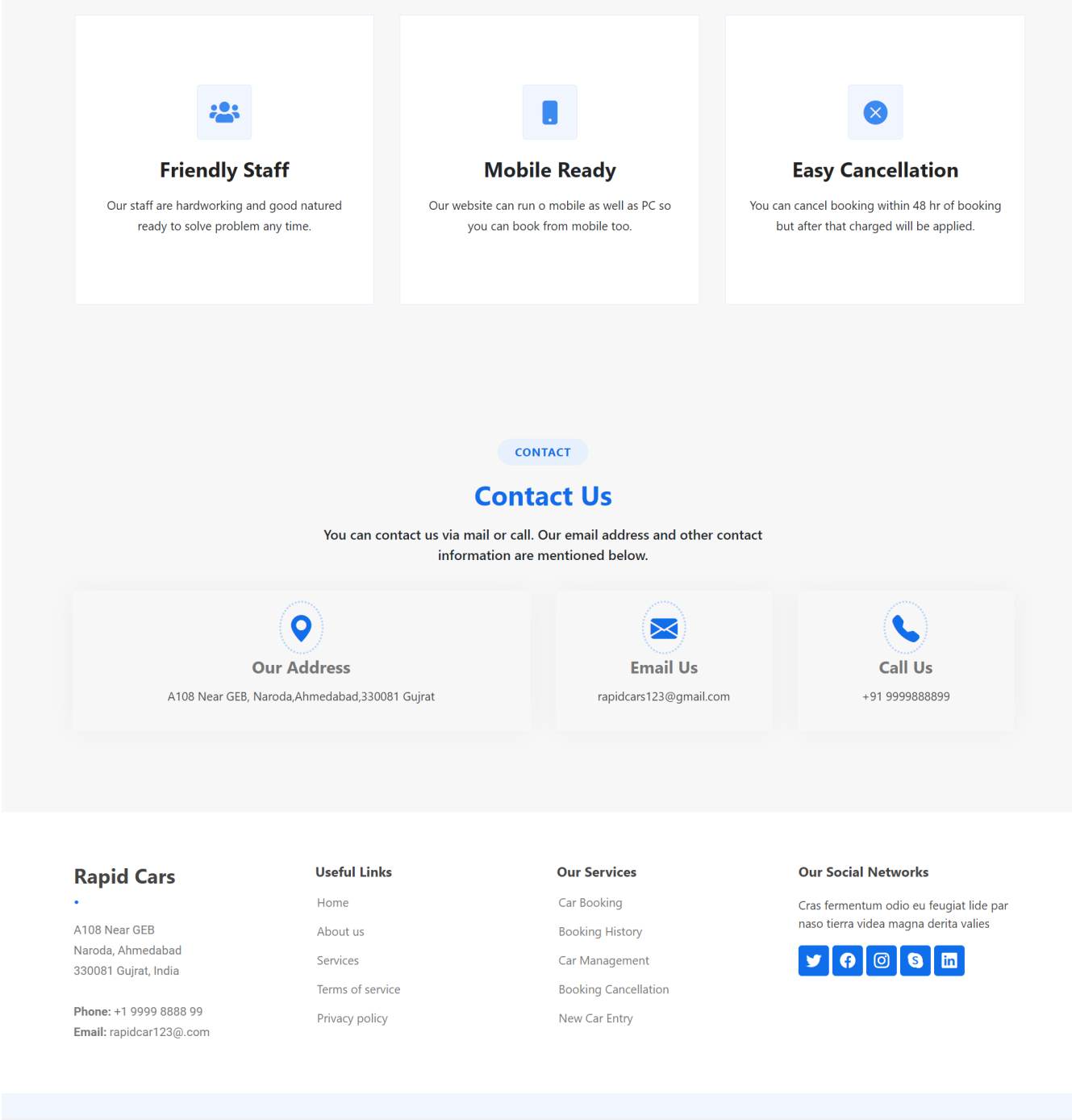


Fig. 7.3
Home Page (iii)

7.1.2 Navigation:

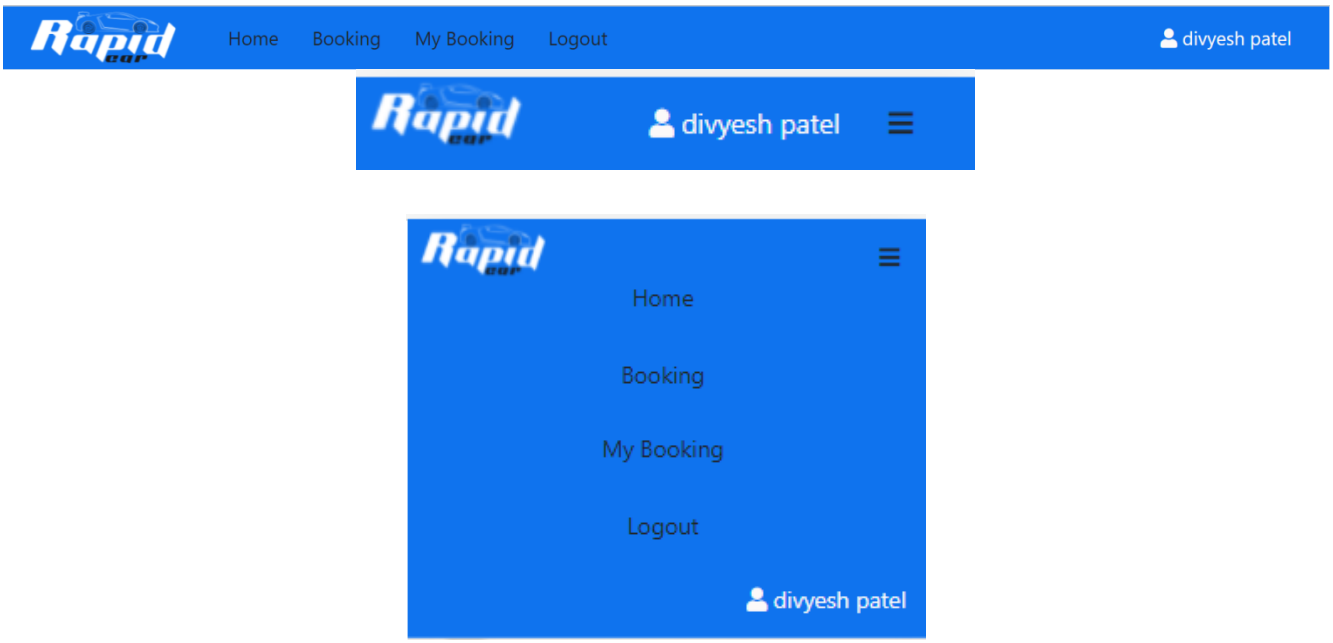


Fig. 7.4
Navigation

7.1.3 Registration:

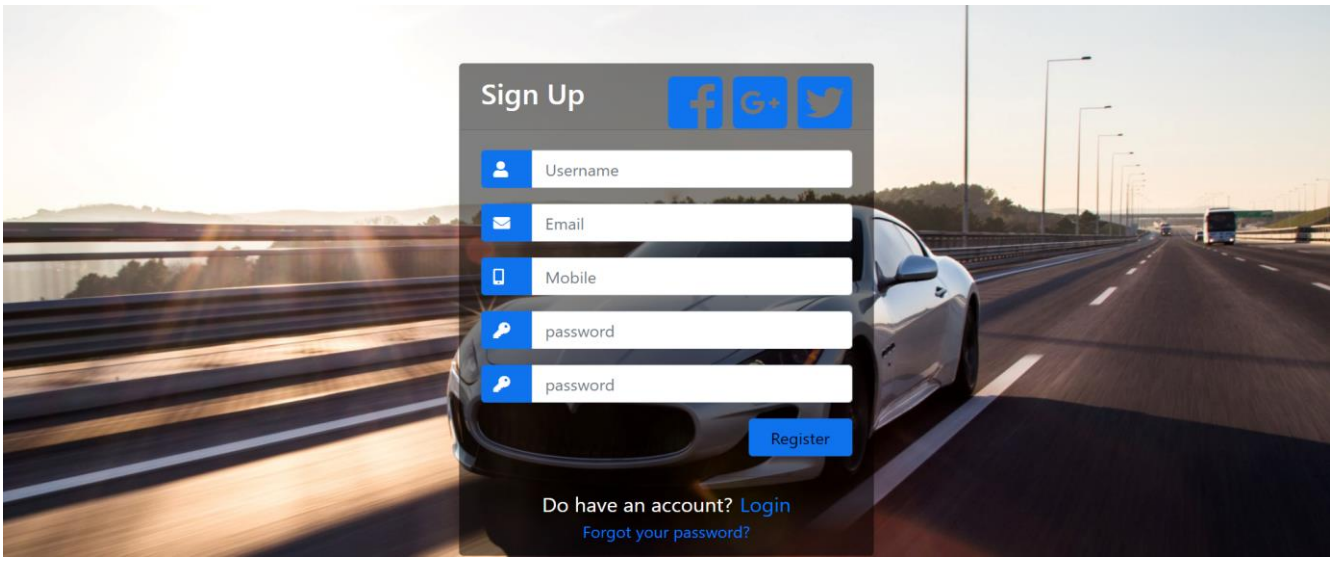


Fig. 7.5
Registration

7.1.4 Sign In:

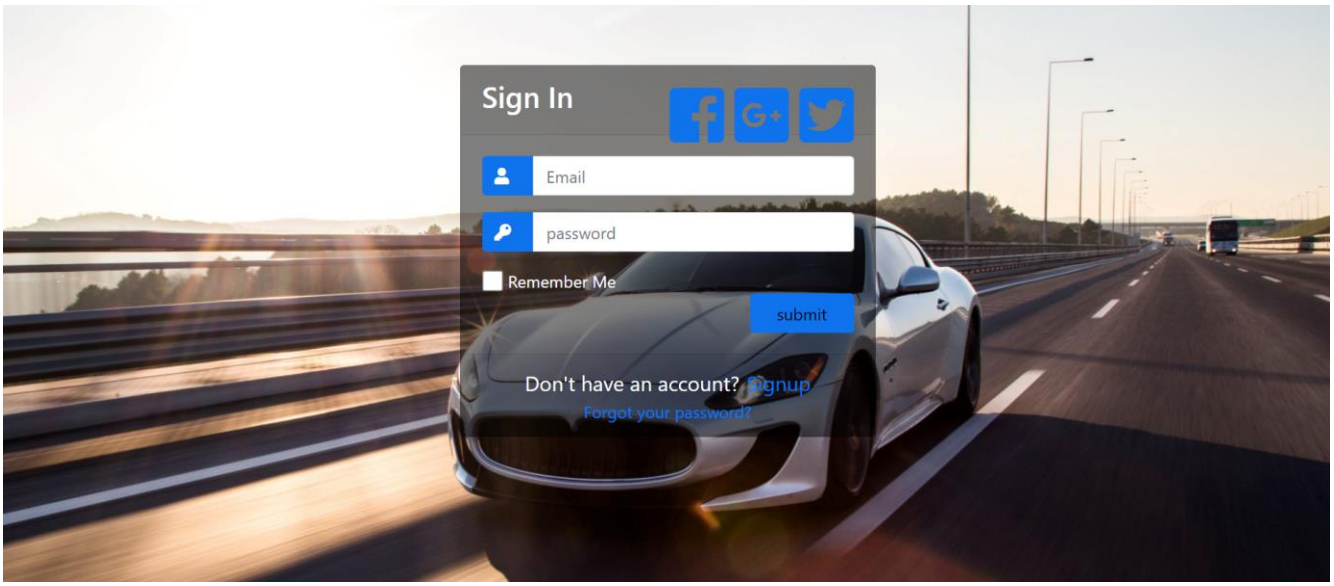
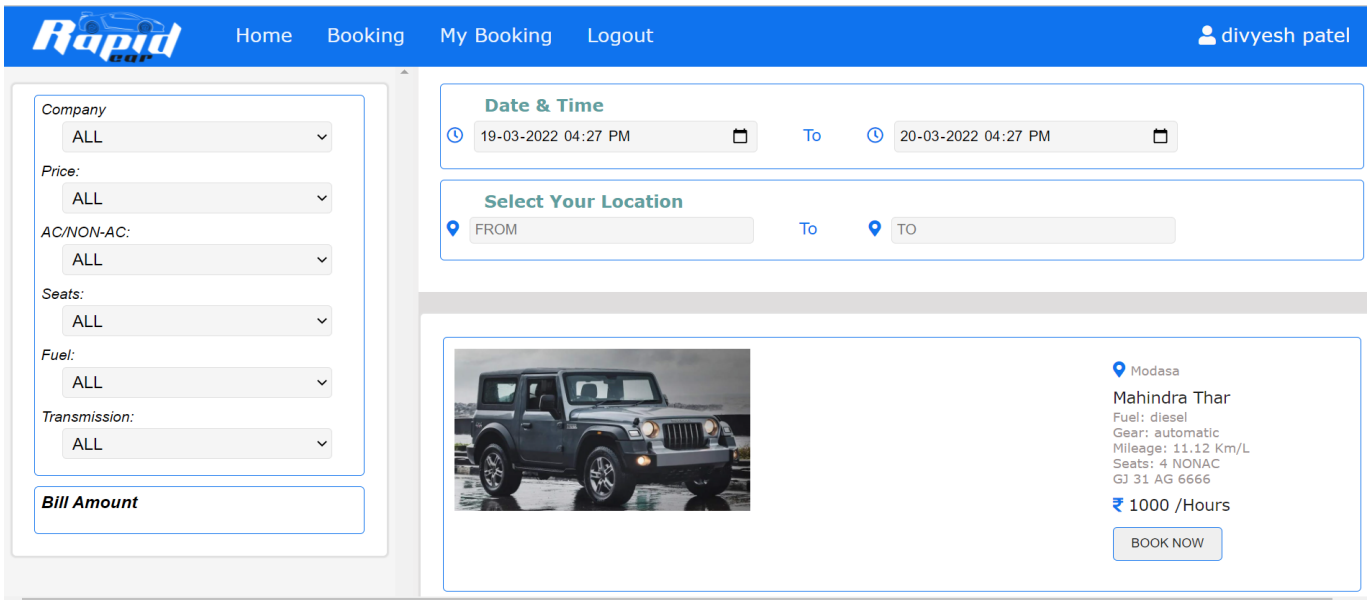


Fig. 7.6
Sign In

7.1.5 Booking Page:



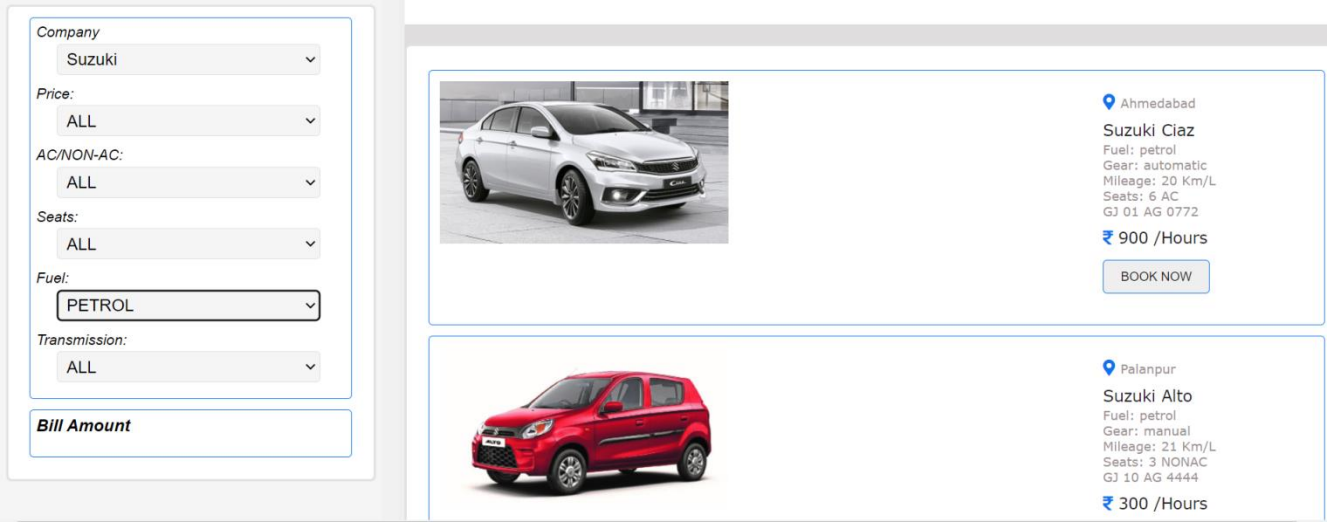


Fig. 7.7
Booking Page

7.1.6 My booking:

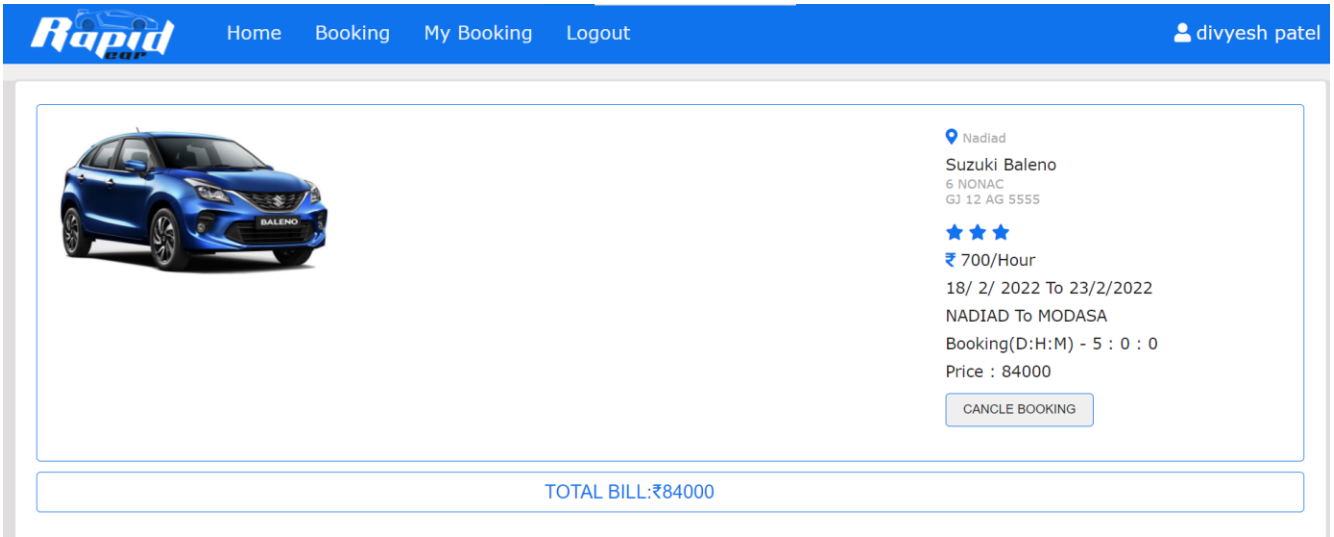


Fig. 7.8
My Booking Page

7.2 Admin View

7.2.1 Register car:

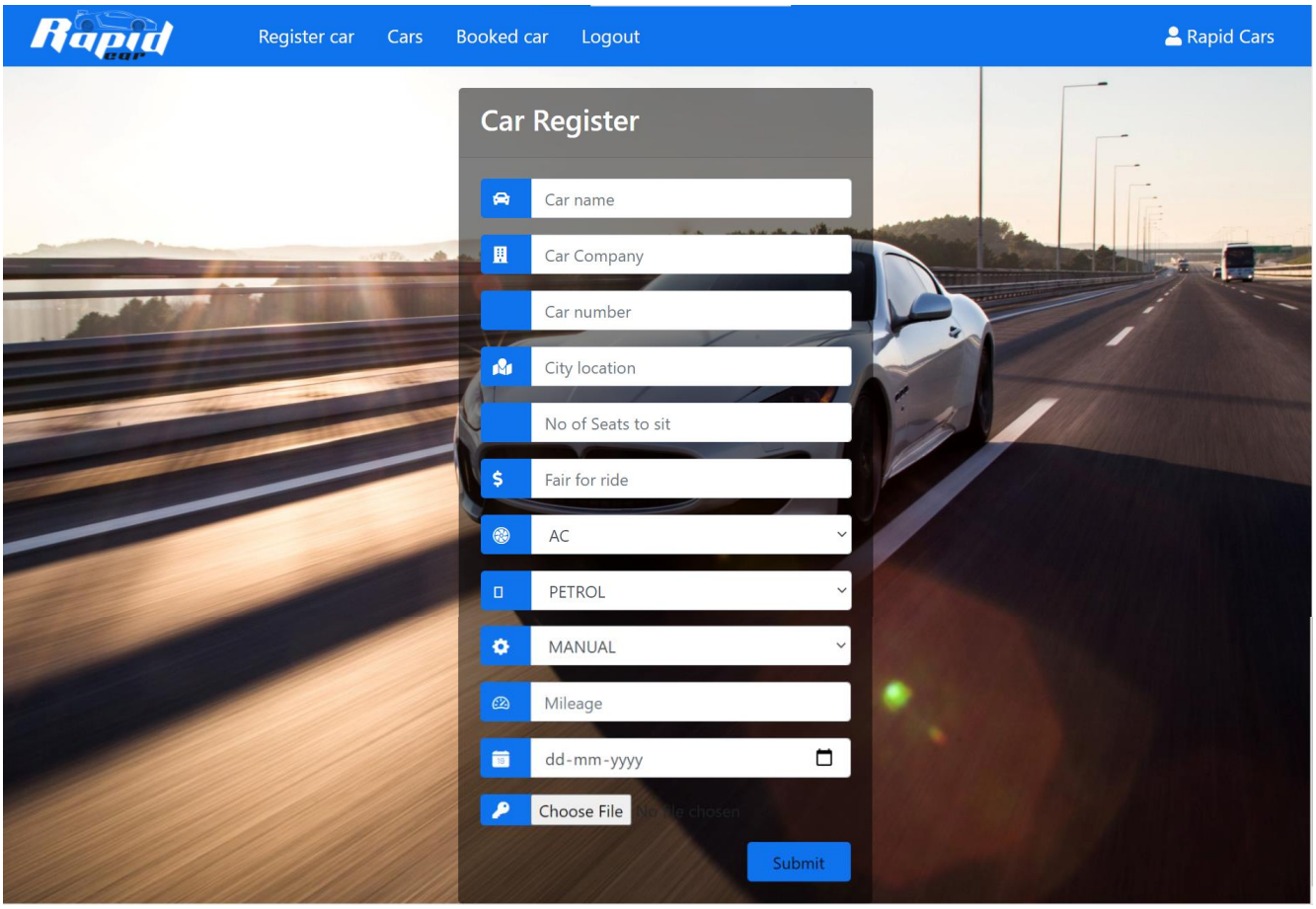
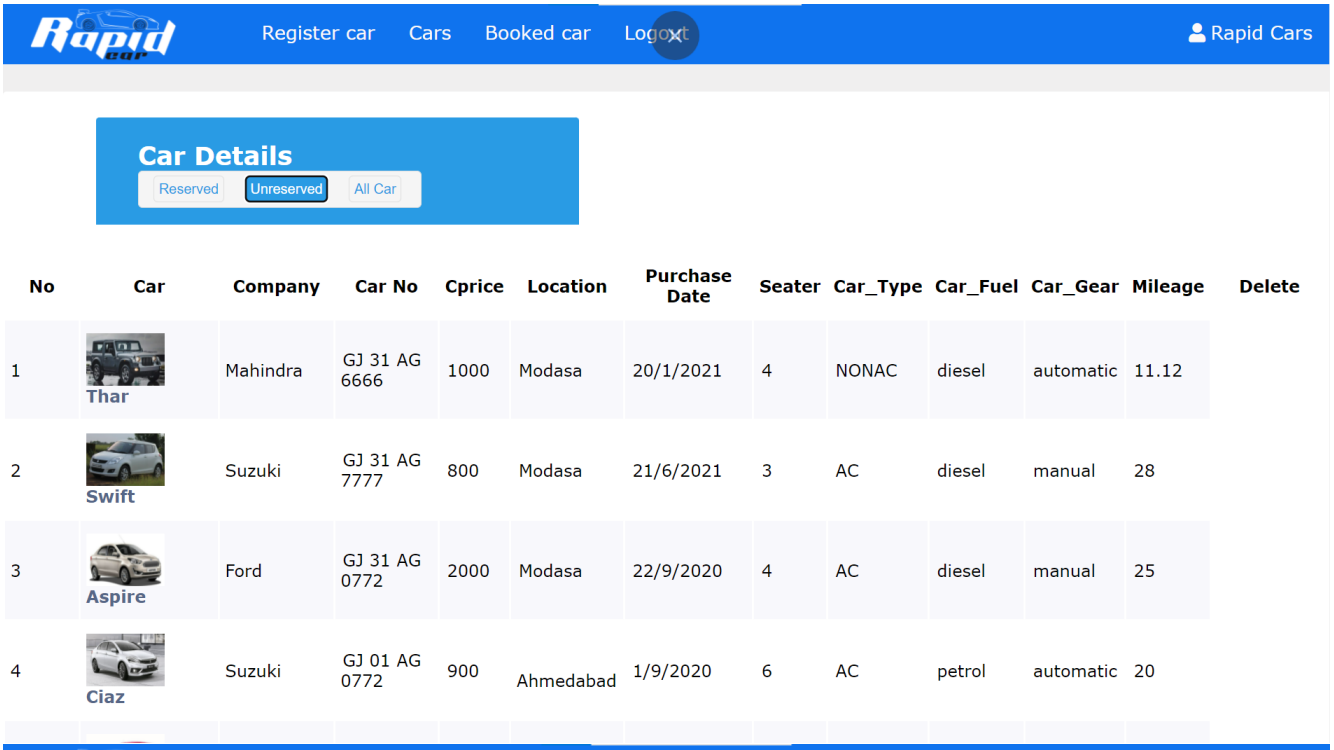
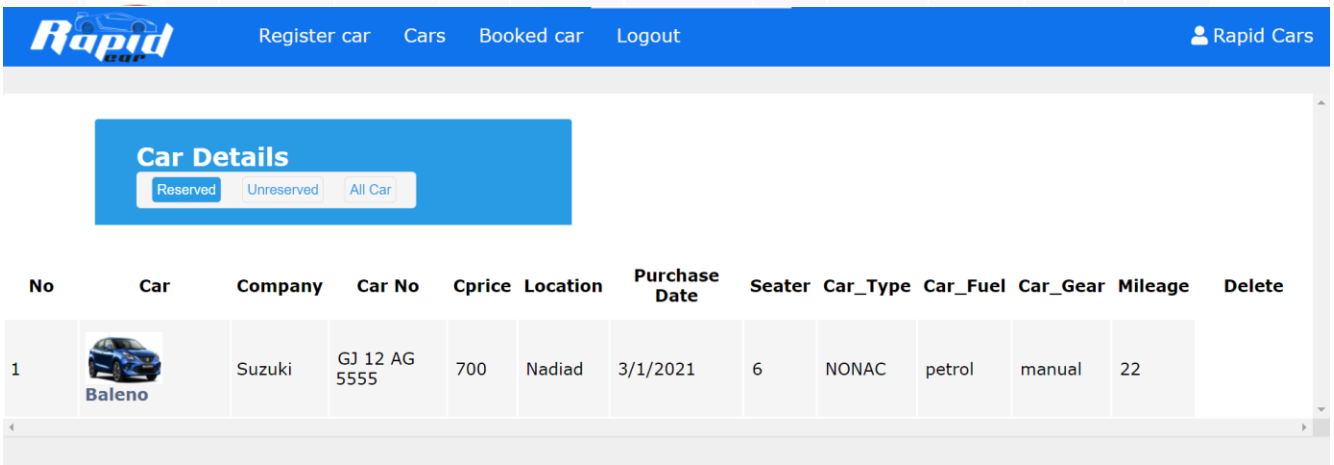


Fig. 7.9
Register Car Page

7.2.1 Car Details:



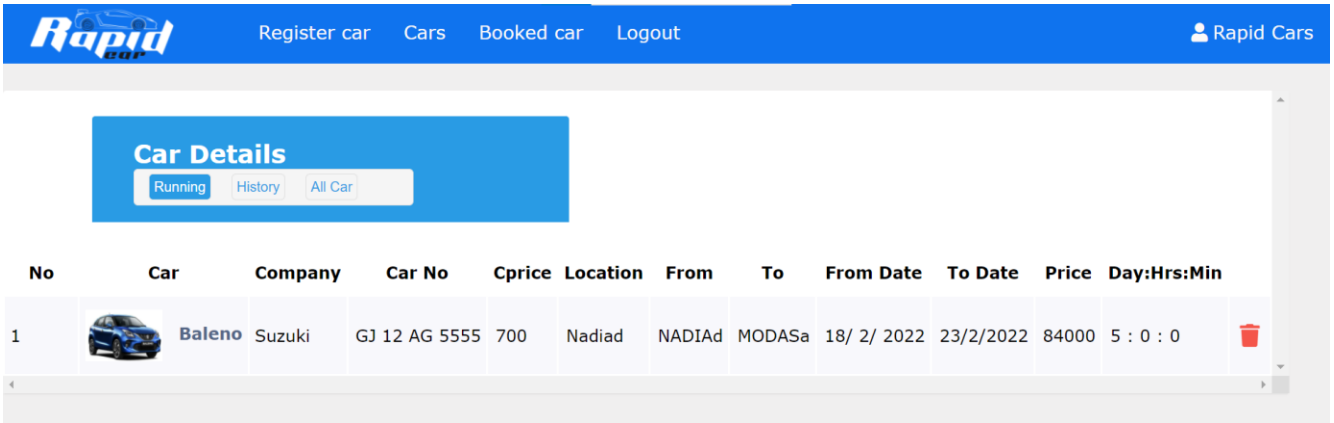
No	Car	Company	Car No	Cprice	Location	Purchase Date	Seater	Car_Type	Car_Fuel	Car_Gear	Mileage	Delete
1	Thar	Mahindra	GJ 31 AG 6666	1000	Modasa	20/1/2021	4	NONAC	diesel	automatic	11.12	
2	Swift	Suzuki	GJ 31 AG 7777	800	Modasa	21/6/2021	3	AC	diesel	manual	28	
3	Aspire	Ford	GJ 31 AG 0772	2000	Modasa	22/9/2020	4	AC	diesel	manual	25	
4	Ciaz	Suzuki	GJ 01 AG 0772	900	Ahmedabad	1/9/2020	6	AC	petrol	automatic	20	



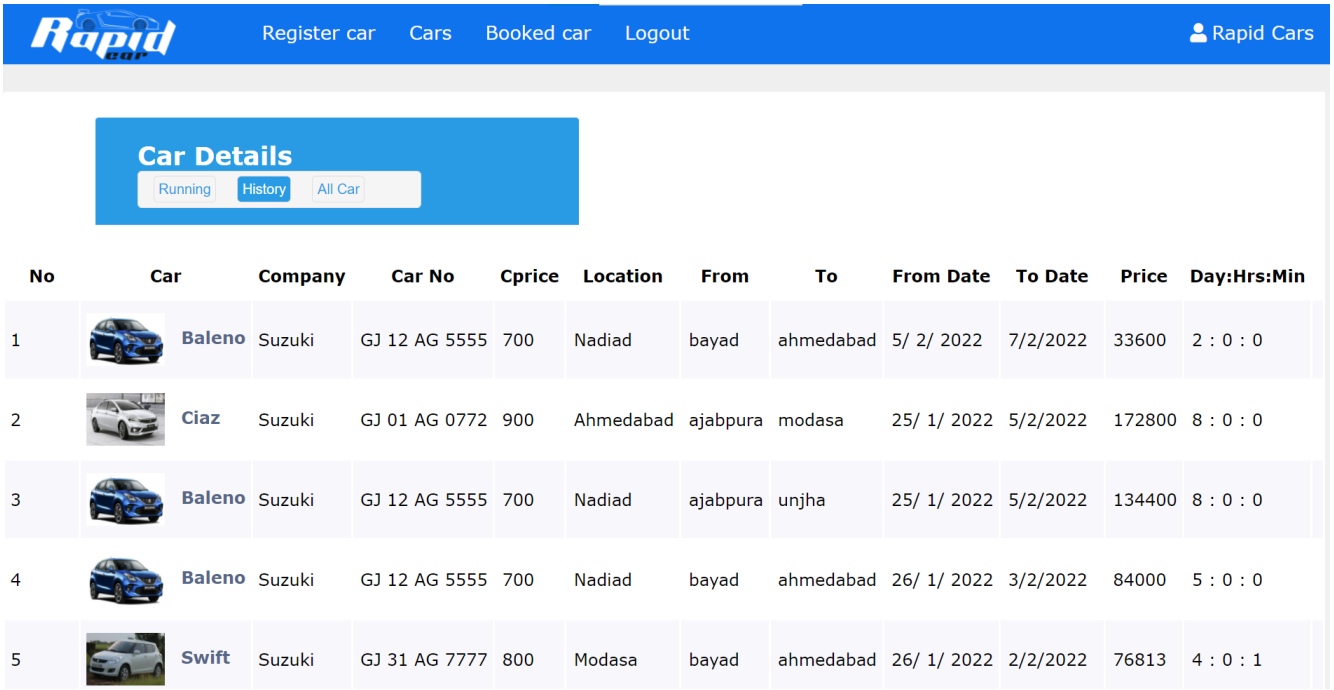
No	Car	Company	Car No	Cprice	Location	Purchase Date	Seater	Car_Type	Car_Fuel	Car_Gear	Mileage	Delete
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	3/1/2021	6	NONAC	petrol	manual	22	

Fig. 7.10
Car Details Page

7.2.3 Booking Details:



No	Car	Company	Car No	Cprice	Location	From	To	From Date	To Date	Price	Day:Hrs:Min
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	NADIAd	MODASa	18/ 2/ 2022	23/2/2022	84000	5 : 0 : 0



No	Car	Company	Car No	Cprice	Location	From	To	From Date	To Date	Price	Day:Hrs:Min
1	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	bayad	ahmedabad	5/ 2/ 2022	7/2/2022	33600	2 : 0 : 0
2	Ciaz	Suzuki	GJ 01 AG 0772	900	Ahmedabad	ajabpura	modasa	25/ 1/ 2022	5/2/2022	172800	8 : 0 : 0
3	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	ajabpura	unjha	25/ 1/ 2022	5/2/2022	134400	8 : 0 : 0
4	Baleno	Suzuki	GJ 12 AG 5555	700	Nadiad	bayad	ahmedabad	26/ 1/ 2022	3/2/2022	84000	5 : 0 : 0
5	Swift	Suzuki	GJ 31 AG 7777	800	Modasa	bayad	ahmedabad	26/ 1/ 2022	2/2/2022	76813	4 : 0 : 1

Fig. 7.11
Booking Details Page

7.3 Mobile view:

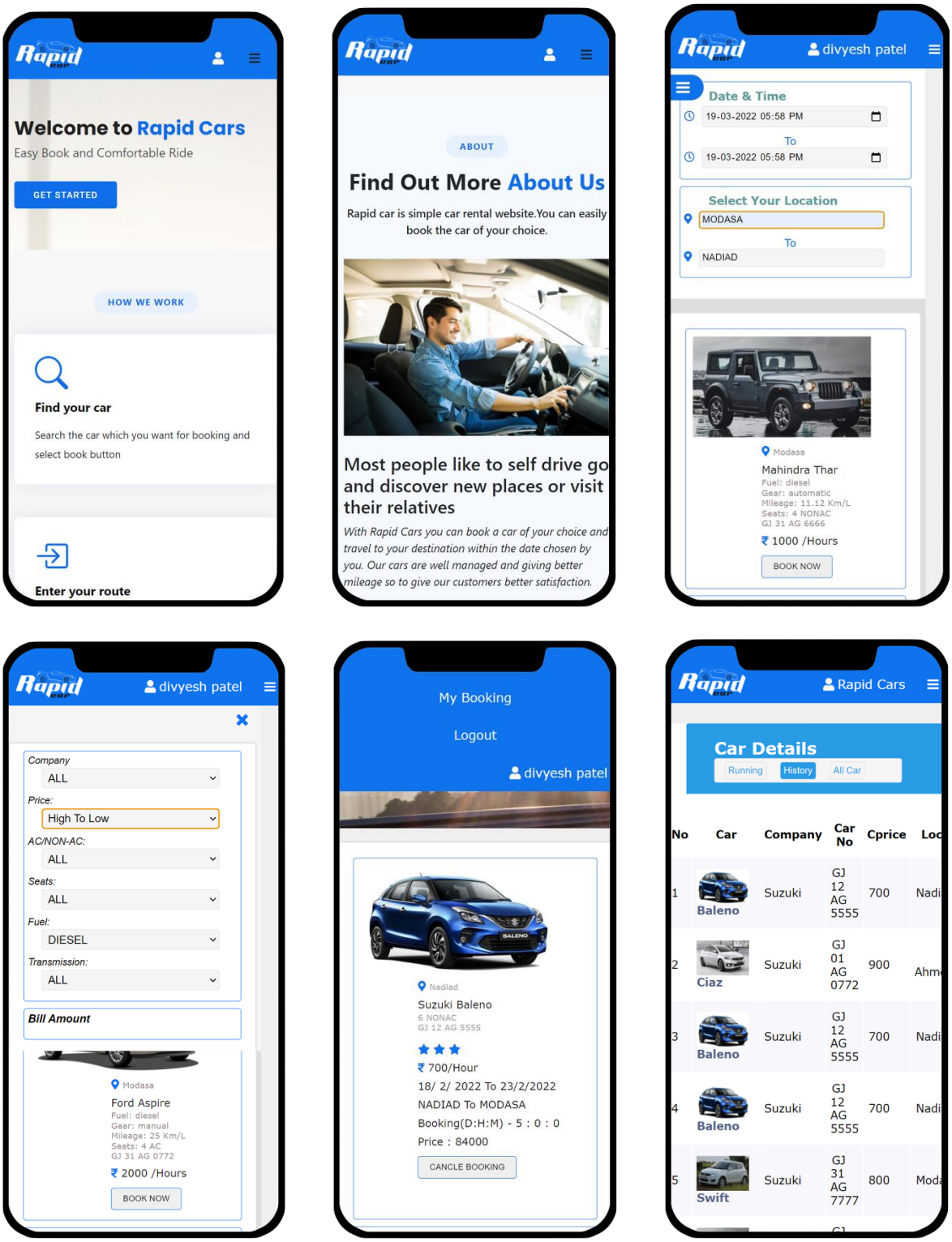


Fig. 7.12
Mobile View

8. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements. Functionalities that are successfully implemented in the system are:

- Authentication of User details.
- Send email after successful registering.
- Display Un-reserved car on booking page.
- Filter and sort according to user's choice.
- Book Car after entering location and date.
- Send conformation email after successful booking
- Cancel Booking.
- Display ongoing booking history of the user.
- Display car details.
- Display booking of all cars.
- Complete booking of the car.
- Registering a new car.

After the implementation and coding of system, comprehensive testing was performed on the system to determine the errors and possible flaws in the system.

9. Limitations and Future Enhancements

- We are able to implement the functionality model of the "Rapid Cars". We aim to make this product ready to be used in small practical business use cases. Currently, the project supports user to enter data.

- The project can be extended with the support of Feedback System for more interactions and to provide more information, about their experiences on their trip with us, to customers.

- Further extensions involve integrating it with Web API and integrating Machine Learning recommendation models for suggesting best cars on the basis of customer feedbacks for review.

10. Reference / Bibliography

Following links and websites were referred during the development of this project:

- "JavaScript Tutorial." W3Schools Online Web Tutorials, 16 Jan 2022 https://www.w3schools.com/js/.
- Programming. React JS - React Tutorial for Beginners. YouTube, 29 Jan 2022 https://www.youtube.com/watch?v=Ke90Tje7VS0.
- "What Is MongoDB Atlas? — MongoDB Atlas, 14 Feb 2022 https://docs.atlas.mongodb.com/.
- "Reactjs - How to Upload Image File in MERN Stack Application with Multer." Stack Overflow, 16 Feb 2022 https://stackoverflow.com/questions/67904317/how-to-upload-image-file-in-mern-stack-application-with-multer.
- "User Authentication in MERN Stack" Medium, 1 Mar 2022 https://medium.com/swlh/user-authentication-using-mern-stack-part-1-backend-cd4d193f15b1.
- "Self Drive Car Rentals | Rent a Car | Car Hire @ Zoomcar." Zoomcar - India, 20 Jan 2022 https://www.zoomcar.com/.