

A Mini Project Report
On

Renting tools

By

Niranjan

1602-21-733-038

Shaik Sameer Abdulla

1602-21-733-043



Department of Computer Science & Engineering

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2023

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Hyderabad-500 031

Department of Computer Science & Engineering



BONAFIDE CERTIFICATE

This is to certify that the Mini project titled **Renting tools** being submitted by P.Sri Deeksha, bearing 1602-20-733-052, in partial fulfilment of the requirements of the V semester, Bachelor of Engineering in Computer Science & Engineering is a record of bonafide work carried out by him/her under my guidance.

S. Vinay Kumar,
Assistant Professor,
Dept. of CSE,
(Faculty I/c)

Dr. T.Adilakshmi,
Professor & HOD,
Dept. of CSE.

(External Examiner)

ACKNOWLEDGEMENT

We extend our sincere thanks to Dr. S. V. Ramana, Principal, Vasavi College of Engineering for his encouragement.

We express our sincere gratitude to Dr. T.Adilakshmi Professor & Head, Department of computer Science and Engineering, Vasavi College of Engineering, for introducing the Mini-Project module in our curriculum, and also for his suggestions, motivation, and co-operation for the successful completion of our Mini Project.

We also want to thank and convey our gratitude towards our mini project coordinators Mr.S.Vinay Kumar , for guiding us in understanding the process of project development & giving us timely suggestions at every phase.

We would also like to sincerely thank the project reviewers for their valuable inputs and suggestion.

TABLE OF CONTENTS

1.Introduction	4-6
1.1 Overview	5
1.2 Problem Definition	5
1.3 objectives	6
2. Literature Survey	6
3. System Design	7
4. Implementation	8
5. Results	48
6. Conclusion and Future work	53
7. References	55

Introduction:

Overview

- **Platform Creation:** Develop a tool-sharing platform using Python, JavaScript, and potentially Java for backend operations.
- **Robust System:** Ensure system robustness and scalability with diverse programming languages.
- **User-Friendly Interface:** Utilize HTML, CSS, and JavaScript for an intuitive frontend interface.
- **Database Management:** Efficiently manage data storage and retrieval using SQL or NoSQL databases.
- **Tailored Solution:** Employ a combination of languages and technologies to address the specific needs of farmers.

Problem Definition:

Problem Statement:

Many farmers face challenges in accessing and acquiring the necessary agricultural tools and equipment needed for their farming operations. This limitation is often due to the high costs associated with purchasing specialized tools, which may not be used frequently throughout the year. Additionally, some farmers may lack proper storage space for these tools when not in use. As a result, there is a growing need for a convenient and cost-effective solution that enables farmers to rent agricultural tools on a short-term basis. Developing a platform or system that facilitates the renting of farming tools can address these challenges, providing farmers with access to a wide range of tools when they need them while reducing the financial burden of ownership. This solution should offer a seamless and efficient process for renting, returning, and maintaining the tools,

ultimately contributing to increased productivity and sustainability in the agriculture sector.

Objectives:

- 1. Tool Accessibility**
- 2. Cost-Efficiency**
- 3. Resource Utilization**
- 4. Efficient Operations**
- 5. Community Collaboration**

Requirements:

Backend (Server Side):

1. Node.js.
2. Express.js
3. MongoDB.
4. Mongoose.
5. Express Middleware.

Frontend (Client Side):

1. HTML/CSS/JavaScript.
2. Frontend Framework or Library.
3. AJAX or Fetch API.
4. Package Manager (npm or yarn).
5. Build Tools.

Development Tools:

Visual Studio Code (VS Code) was the main IDE for this project, offering a streamlined and efficient coding experience. Its versatility and ease of use facilitated collaborative coding efforts, ensuring an organized and effective development process.

CODE FOLDERS:

backend

Booking model.js:

```
const mongoose = require("mongoose");

const bookingSchema = new mongoose.Schema({

  car : {type : mongoose.Schema.Types.ObjectID , ref:'cars'},
  user : {type : mongoose.Schema.Types.ObjectID , ref:'users'},
  bookedTimeSlots : {
    from : {type : String} ,
    to : {type : String}
  } ,
  totalHours : {type : Number},
  totalAmount : {type : Number},
  // transactionId : {type : String},
  driverRequired : {type : Boolean}

},
{timestamps : true}
)

const bookingModel = mongoose.model('bookings' , bookingSchema)

module.exports = bookingModel
```

car model.js:

```
const mongoose = require("mongoose");

const carSchema = new mongoose.Schema({

  name : {type : String , required : true} ,
  image : {type : String , required : true} ,
  // capacity : {type : Number , required : true},
  fuelType : {type : String , required : true} ,
  bookedTimeSlots : [
    {
      from : {type : String , required : true},
      to : {type : String , required : true}
    }
  ] ,
  rentPerHour : {type : Number , required : true}
```



```

    }, {timestamps : true}

)
const carModel = mongoose.model('cars' , carSchema)
module.exports = carModel

```

user model.js:

```

const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({
  username : {type:String , required: true},
  password : {type:String , required: true}
})

const userModel = mongoose.model('users' , userSchema)

module.exports = userModel

```

bookingroutes.js:

```

const nodemailer = require('nodemailer');
const MongoClient = require('mongodb').MongoClient;

// Connect to MongoDB
const url = 'mongodb://127.0.0.1';
const dbName = 'new-project';

require('dotenv').config();
const express = require("express");
const router = express.Router();
const Booking = require("../models/bookingModel");
const Car = require("../models/carModel");
const { v4: uuidv4 } = require("uuid");
const stripe = require("stripe")(process.env.BACK_END_STRIPE_KEY);
// const stripe =
require("stripe")("sk_test_51OKkBpSF6Amir5MpVgf3kWN42M5pqbTA3ykKrDj
YYYiJE0dJG0OIIYcOY1XFYf92hexiLsSH8rVggorOcpe2p5bR00EFWkWQzw"
);
router.post("/bookcar", async (req, res) => {
  console.log("this is token:", req.body);
  const { token } = req.body;
  console.log("Token received:", token);
  try {
    const customer = await stripe.customers.create({

```

```

    email: token?.email,
    source: token?.id,
  });
  console.log(customer.id);
  const payment = await stripe.paymentIntents.create(
    {
      amount: req.body.totalAmount * 100,
      // currency: "inr",
      currency: "INR",
      customer: customer?.id,
      receipt_email: token?.email
    },
    {
      idempotencyKey: uuidv4(),
    }
  );

  // if (payment && payment.status === "succeeded") {
  //   req.body.transactionId = payment.source.id;
  //   const newbooking = new Booking(req.body);
  //   await newbooking.save();
  //   const car = await Car.findOne({ _id: req.body.car });
  //   console.log(req.body.car);
  //   car.bookedTimeSlots.push(req.body.bookedTimeSlots);

  //   await car.save();
  //   res.send("Your booking is successfull");
  // } else {
  //   return res.status(400).json(error);
  // }
  if (payment && payment.status === "succeeded") {
    // req.body.transactionId = payment.id;
    const newbooking = new Booking(req.body);
    await newbooking.save();

    const car = await Car.findOne({ _id: req.body.car });
    car.bookedTimeSlots.push(req.body.bookedTimeSlots);
    await car.save();

    res.send("Your booking is successful");
  } else {
    console.error("Payment failed:", payment.failure_message);
    const newbooking = new Booking(req.body);
    await newbooking.save();

    const car = await Car.findOne({ _id: req.body.car });

```

```

car.bookedTimeSlots.push(req.body.bookedTimeSlots);
await car.save();

res.send("Your booking is successful");

MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true
}, async (err, client) => {
  if (err) throw err;

  const db = client.db(dbName);
  const collection = db.collection('your_collection');

  // Query MongoDB for data
  const data = await collection.findOne({ /* your query */ });

  // Close the MongoDB connection
  client.close();

  // Send email using Nodemailer
  const transporter = nodemailer.createTransport({
    service: 'gmail',
    auth: {
      user: 'sainiranjangalipelli@gmail.com',
      pass: 'zpeh ugjr wmnq qefu',
    },
  });

  const mailOptions = {
    from: 'sainiranjangalipelli@gmail.com',
    to: token.email,
    subject: 'Booking conformation',
    text: `your product has been booked for time slot
    ${req.body.bookedTimeSlots.from} to ${req.body.bookedTimeSlots.to}`,
    // const fromFormatted :moment(req.bookedTimeSlots.from).format('MMM
    DD YYYY HH:mm'),
    // const toFormatted :moment(req.bookedTimeSlots.to).format('MMM DD
    YYYY HH:mm'),
    // text :`your product has been booked for time slot ' + fromFormatted + ' ' +
    toFormatted
  };

  transporter.sendMail(mailOptions, (error, info) => {
    if (error) {
      return console.error(error);
    }
    console.log('Email sent: ' + info.response);
  });

```

```

    });

    // return res.status(400).json({ error: "Payment failed", message:
payment.failure_message });
  }
} catch (error) {
  console.log(error);
  return res.status(400).json(error);
}
});

router.get("/getallbookings", async(req, res) => {

  try {

    const bookings = await Booking.find().populate('car')
    res.send(bookings)

  } catch (error) {
    return res.status(400).json(error);
  }

});

module.exports = router;

```

```

cars_route.js:
const express = require("express");
const router = express.Router();
const Car = require("../models/carModel");

router.get("/getallcars", async (req, res) => {
  try {
    const cars = await Car.find();
    res.send(cars);
  } catch (error) {
    return res.status(400).json(error);
  }
});

router.post("/addcar", async (req, res) => {
  try {
    const newcar = new Car(req.body);
    await newcar.save();
    res.send("Car added successfully");
  } catch (error) {
    return res.status(400).json(error);
  }
});

```

```

    }
  });

  router.post("/editcar", async (req, res) => {
    try {
      const car = await Car.findOne({ _id: req.body._id });
      car.name = req.body.name;
      car.image = req.body.image;
      car.fuelType = req.body.fuelType;
      car.rentPerHour = req.body.rentPerHour;
      // car.capacity = req.body.capacity;

      await car.save();

      res.send("Car details updated successfully");
    } catch (error) {
      return res.status(400).json(error);
    }
  });

  router.post("/deletecar", async (req, res) => {
    try {
      await Car.findOneAndDelete({ _id: req.body.carid });

      res.send("Car deleted successfully");
    } catch (error) {
      return res.status(400).json(error);
    }
  });

  module.exports = router;

users_route.js:
const express = require("express");
const router = express.Router();
const User = require("../models/userModel")

router.post("/login", async(req, res) => {

  const {username , password} = req.body

  try {
    const user = await User.findOne({username , password})
    console.log(user);
    if(user) {
      res.send(user)
    }
  }

```

```

        else{
            return res.status(400).json(error);
        }
    } catch (error) {
        return res.status(400).json(error);
    }
});

router.post("/register", async(req, res) => {

    try {
        const newuser = new User(req.body)
        await newuser.save()
        res.send('User registered successfully')
    } catch (error) {
        return res.status(400).json(error);
    }
});

module.exports = router

```

db.js:

```

const mongoose = require("mongoose");

function connectDB(){

    mongoose.connect('mongodb://127.0.0.1/new-project' , {useUnifiedTopology:
true , useNewUrlParser: true})

    const connection = mongoose.connection

    connection.on('connected' , ()=>{
        console.log('Mongo DB Connection Successfull')
    })

    connection.on('error' , ()=>{
        console.log('Mongo DB Connection Error')
    })
}

```

```
connectDB()
```

```
module.exports = mongoose
```

server.js:

```
const express = require("express");
const app = express();
const Car = require('./models/carModel');
const port = process.env.PORT || 5000;
const dbConnection = require('./db');
app.use(express.json());
const path = require("path");
const usersRoute = require('./routes/usersRoute')
const carsRoute = require('./routes/carsRoute')
const bookingsRoute = require('./routes/bookingsRoute')
```

```
app.use("/api/cars/", require("./routes/carsRoute"));
app.use("/api/users/", require("./routes/usersRoute"));
app.use("/api/bookings/", require("./routes/bookingsRoute"));
```

```
//-----deployment-----
```

```
__dirname = path.resolve();
```

```
if(process.env.NODE_ENV==="production") {
  app.use(express.static(path.join(__dirname, "/frontend/build")));
```

```
  app.get('*', (req, res) => {
    res.sendFile(path.resolve(__dirname, "frontend", "build", "index.html"));
  });
} else {
  app.get("/", (req, res) => {
    res.send("API is running..");
  });
}
```

```
//-----deployment-----
```

```
app.get("/", (req, res) => res.send("Hello World!"));
```

```
app.listen(port, () => console.log(`Node JS Server Started in Port ${port}`));
```

frontend

default layout.js:

```
import React from "react";
import { Menu, Dropdown, Button, Space, Row, Col } from "antd";
import { Link } from 'react-router-dom'

function DefaultLayout(props) {
  const user = JSON.parse(localStorage.getItem('user'))
  const menu = (
    <Menu>
      <Menu.Item key="home">
        <a href="/">Home</a>
      </Menu.Item>
      <Menu.Item key="userbookings">
        <a href="/userbookings">Bookings</a>
      </Menu.Item>
      <Menu.Item key="admin">
        <a href="/admin">Admin</a>
      </Menu.Item>

      { /* <Menu.Item onClick={() => {
        localStorage.removeItem('user');
        window.location.href = '/login'
      }}}>
        <li style={{ color: 'orangered' }}>Logout</li>
      </Menu.Item> */ }
      <Menu.Item key="logout" onClick={() => {
        localStorage.removeItem('user');
        window.location.href = '/login';
      }}>
        <span style={{ color: 'orangered' }}>Logout</span>
      </Menu.Item>
    </Menu>
  );
  return (
    <div>
      <div className="header bs1">
        <Row gutter={16} justify='center'>
          <Col lg={20} sm={24} xs={24}>
            <div className="d-flex justify-content-between">
              <h1 ><b><Link to="/" style={{ color: 'orangered' }}>RENT PE</Link></b></h1>

              <Dropdown overlay={menu} placement="bottomCenter">
                <Button>{user.username}</Button>
              </Dropdown>
            </div>
          </Col>
        </Row>
      </div>
    </div>
  );
}
```



```

    </div>
    </Col>
  </Row>

</div>
<div className="content">{props.children}</div>

<div className="footer text-center">
  <hr />

  <p>Desinged and Developed By</p>

  <p>TEAM BLUE</p>
</div>
</div>
);
}

export default DefaultLayout;

```

spinner.js

```

import React from 'react'
import {Spin} from 'antd'
function Spinner() {
  return (
    <div className="spinner">
      <Spin size='large'/>
    </div>
  )
}

export default Spinner

```

addcar.js

```

import { Col , Row , Form , Input} from 'antd'
import React from 'react'
import { useDispatch , useSelector } from 'react-redux'
import DefaultLayout from '../components/DefaultLayout'
import Spinner from '../components/Spinner'
import { addCar } from '../redux/actions/carsActions'
function AddCar() {

```

```

const dispatch = useDispatch()
const {loading} = useSelector(state=>state.alertsReducer)

function onFinish(values){

  values.bookedTimeSlots=[]

  dispatch(addCar(values))
  console.log(values)
}

return (
  <DefaultLayout>
    {loading && (<Spinner />)}
    <Row justify='center mt-5'>
      <Col lg={12} sm={24} xs={24} className='p-2'>
        <Form className='bs1 p-2' layout='vertical' onFinish={onFinish}>
          <h3>Add New Item</h3>
          <hr />
          <Form.Item name='name' label='Item Name' rules={[{required: true}]}>
            <Input/>
          </Form.Item>
          <Form.Item name='image' label='Image url' rules={[{required: true}]}>
            <Input/>
          </Form.Item>
          <Form.Item name='rentPerHour' label='Rent per hour' rules={[{required:
true}]}>
            <Input/>
          </Form.Item>
          <Form.Item name='capacity' label='Capacity' rules={[{required: true}]}>
            <Input/>
          </Form.Item>
          <Form.Item name='fuelType' label='Fuel Type' rules={[{required: true}]}>
            <Input/>
          </Form.Item>

          <div className='text-right'>
            <button className='btn1'>ADD ITEM</button>
          </div>

        </Form>
      </Col>
    </Row>
  </DefaultLayout>
)
}

```

```
export default AddCar
```

admin home.js

```
import React, { useState, useEffect } from "react";
import { useSelector, useDispatch } from "react-redux";
import DefaultLayout from "../components/DefaultLayout";
import { deleteCar, getAllCars } from "../redux/actions/carsActions";
import { Col, Row, Divider, DatePicker, Checkbox, Edit } from "antd";
import { Link } from "react-router-dom";
import Spinner from "../components/Spinner";
import moment from "moment";
import { DeleteOutlined, EditOutlined } from "@ant-design/icons";
import { Popconfirm, message } from "antd";
const { RangePicker } = DatePicker;
function AdminHome() {
  const { cars } = useSelector((state) => state.carsReducer);
  const { loading } = useSelector((state) => state.alertsReducer);
  const [totalCars, setTotalcars] = useState([]);
  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllCars());
  }, []);

  useEffect(() => {
    setTotalcars(cars);
  }, [cars]);

  return (
    <DefaultLayout>
      <Row justify="center" gutter={16} className="mt-2">
        <Col lg={20} sm={24}>
          <div className="d-flex justify-content-between align-items-center">
            <h3 className="mt-1 mr-2">Admin Panel</h3>
            <button className="btn1">
              <a href="/addcar">ADD ITEM</a>
            </button>
          </div>
        </Col>
      </Row>

      {loading == true && <Spinner />}
    </DefaultLayout>
  );
}
```

```

<Row justify="center" gutter={16}>
  /* {totalCars.map((car) => {
    return (
      <Col lg={5} sm={24} xs={24}> */
      {totalCars.map((car) => {
return (
  <Col key={car._id} lg={5} sm={24} xs={24}>
    <div className="car p-2 bs1">
      <img src={car.image} className="carimg" />

      <div className="car-content d-flex align-items-center justify-content-between">
        <div className="text-left pl-2">
          <p>{car.name}</p>
          <p> Rent Per Hour {car.rentPerHour} /-</p>
        </div>

        <div className="mr-4">
          <Link to={` /editcar/${car._id}`} >
            <EditOutlined
              className="mr-3"
              style={{ color: "green", cursor: "pointer" }}
            />
          </Link>

          <Popconfirm
            title="Are you sure to delete this car?"
            onConfirm={()=>{dispatch(deleteCar({carid : car._id}))}}

            okText="Yes"
            cancelText="No"
          >
            <DeleteOutlined
              style={{ color: "red", cursor: "pointer" }}
            />
          </Popconfirm>
        </div>
      </div>
    </div>
  </div>
</Col>
);
}}
</Row>
</DefaultLayout>
);
}

```

```

export default AdminHome;

```

bookingcar.js

```
import { Col, Row, Divider, DatePicker, Checkbox, Modal } from "antd";
import React, { useState, useEffect } from "react";
import { useSelector, useDispatch } from "react-redux";
import DefaultLayout from "../components/DefaultLayout";
import Spinner from "../components/Spinner";
import { getAllCars } from "../redux/actions/carsActions";
import moment from "moment";
import { bookCar } from "../redux/actions/bookingActions";
import StripeCheckout from "react-stripe-checkout";
import AOS from 'aos';
```

```
import 'aos/dist/aos.css'; // You can also use <link> for styles
const { RangePicker } = DatePicker;
function BookingCar({ match }) {
  const { cars } = useSelector((state) => state.carsReducer);
  const { loading } = useSelector((state) => state.alertsReducer);
  const [car, setcar] = useState({});
  const dispatch = useDispatch();
  const [from, setFrom] = useState();
  const [to, setTo] = useState();
  const [totalHours, setTotalHours] = useState(0);
  const [driver, setdriver] = useState(false);
  const [totalAmount, setTotalAmount] = useState(0);
  const [showModal, setShowModal] = useState(false);
```

```
  useEffect(() => {
    if (cars.length === 0) {
      dispatch(getAllCars());
    } else {
      setcar(cars.find((o) => o._id === match.params.carid));
    }
  }, [cars]);
```

```
  useEffect(() => {
    setTotalAmount(totalHours * car.rentPerHour);
    if (driver) {
      setTotalAmount(totalAmount + 30 * totalHours);
    }
  }, [driver, totalHours]);
```

```
  function selectTimeSlots(values) {
```

```

setFrom(moment(values[0]).format("MMM DD yyyy HH:mm"));
setTo(moment(values[1]).format("MMM DD yyyy HH:mm"));

setTotalHours(values[1].diff(values[0], "hours"));
}

```

```

function onToken(token){
  console.log("Stripe Checkout Token Received:", token);
  const reqObj = {
    token,
    user: JSON.parse(localStorage.getItem("user"))._id,
    car: car._id,
    totalHours,
    totalAmount,
    driverRequired: driver,
    bookedTimeSlots: {
      from,
      to,
    },
  };
}

```

```

  dispatch(bookCar(reqObj));
}

```

```

return (
  <DefaultLayout>
    {loading && <Spinner />}
    <Row
      justify="center"
      className="d-flex align-items-center"
      style={{ minHeight: "90vh" }}
    >
      <Col lg={10} sm={24} xs={24} className='p-3'>
        <img src={car.image} className="carimg2 bs1 w-100" data-aos='flip-left' data-aos-
duration='1500'/>
      </Col>

      <Col lg={10} sm={24} xs={24} className="text-right">
        <Divider type="horizontal" dashed>
          Car Info
        </Divider>
        <div style={{ textAlign: "right" }}>
          <p>{car.name}</p>
          <p>{car.rentPerHour} Rent Per hour /-</p>
          <p>Fuel Type : {car.fuelType}</p>

```

```

    <p>Max Persons : {car.capacity}</p>
</div>

<Divider type="horizontal" dashed>
  Select Time Slots
</Divider>
<RangePicker
  showTime={{ format: "HH:mm" }}
  format="MMM DD yyyy HH:mm"
  onChange={selectTimeSlots}
/>
<br />
<button
  className="btn1 mt-2"
  onClick={() => {
    setShowModal(true);
  }}
>
  See Booked Slots
</button>
{from && to && (
  <div>
    <p>
      Total Hours : <b>{totalHours}</b>
    </p>
    <p>
      Rent Per Hour : <b>{car.rentPerHour}</b>
    </p>
    <Checkbox
      onChange={(e) => {
        if (e.target.checked) {
          setdriver(true);
        } else {
          setdriver(false);
        }
      }}
    >
      Driver Required
    </Checkbox>

    <h3>Total Amount : {totalAmount}</h3>

    <StripeCheckout
      shippingAddress
      billingAddress
      token={onToken}
      currency="INR"

```

```

        amount={totalAmount * 100}

stripeKey="pk_test_51OKkBpSF6Amir5Mpy8HetZgsqnRUqwskoZx3T0R4WtQv1gVS8T
JDSYazoEf47lchwL5NvM7xGpDILMjgsVOopSwd00cBGRBKWR"
    >
        <button className="btn1">
            Book Now
        </button>
    </StripeCheckout>

</div>
)}}
</Col>

{car.name && (
    <Modal
        visible={showModal}
        closable={false}
        footer={false}
        title="Booked time slots"
    >
        <div className="p-2">
            {/* {car.bookedTimeSlots.map((slot) => {
                return (
                    <button className="btn1 mt-2">
                        {slot.from} - {slot.to}
                    </button>
                );
            })} */}
            {car.bookedTimeSlots.map((slot, index) => (
                <button key={index} className="btn1 mt-2">
                    {slot.from} - {slot.to}
                </button>
            ))}
        </div>
        <div className="text-right mt-5">
            <button
                className="btn1"
                onClick={() => {
                    setShowModal(false);
                }}
            >
                CLOSE
            </button>
        </div>
    </div>
)}

```



```

        </Modal>
      )}
    </Row>
  </DefaultLayout>
);
}

export default BookingCar;

```

editcar.js

```

import { Col, Row, Form, Input } from "antd";
import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import DefaultLayout from "../components/DefaultLayout";
import Spinner from "../components/Spinner";
import { addCar, editCar, getAllCars } from "../redux/actions/carsActions";
function EditCar({ match }) {
  const { cars } = useSelector((state) => state.carsReducer);
  const dispatch = useDispatch();
  const { loading } = useSelector((state) => state.alertsReducer);
  const [car, setcar] = useState();
  const [totalcars, settotalcars] = useState([]);
  useEffect(() => {
    if (cars.length === 0) {
      dispatch(getAllCars());
    } else {
      settotalcars(cars);
      setcar(cars.find((o) => o._id === match.params.carid));
      console.log(car);
    }
  }, [cars]);

  function onFinish(values) {
    values._id = car._id;

    dispatch(editCar(values));
    console.log(values);
  }

  return (
    <DefaultLayout>
      {loading && <Spinner />}
      <Row justify="center mt-5">
        <Col lg={12} sm={24} xs={24} className='p-2'>
          {totalcars.length > 0 && (

```

```

<Form
  initialValues={car}
  className="bs1 p-2"
  layout="vertical"
  onFinish={onFinish}
>
  <h3>Edit Item</h3>

  <hr />
  <Form.Item
    name="name"
    label="Car name"
    rules={[{ required: true }]}
  >
    <Input />
  </Form.Item>
  <Form.Item
    name="image"
    label="Image url"
    rules={[{ required: true }]}
  >
    <Input />
  </Form.Item>
  <Form.Item
    name="rentPerHour"
    label="Rent per hour"
    rules={[{ required: true }]}
  >
    <Input />
  </Form.Item>
  {/* <Form.Item
    name="capacity"
    label="Capacity"
    rules={[{ required: true }]}
  >
    <Input />
  </Form.Item> */}
  <Form.Item
    name="fuelType"
    label="Fuel Type"
    rules={[{ required: true }]}
  >
    <Input />
  </Form.Item>

  <div className="text-right">
    <button className="btn1">Edit ITEM</button>
  </div>

```

```

        </div>
      </Form>
    )}
  </Col>
</Row>
</DefaultLayout>
);
}

export default EditCar;

```

home.js

```

import React , {useState,useEffect} from 'react'
import { useSelector , useDispatch } from 'react-redux'
import DefaultLayout from '../components/DefaultLayout'
import { getAllCars } from '../redux/actions/carsActions'
import { Col, Row , Divider , DatePicker, Checkbox} from 'antd'
import {Link} from 'react-router-dom'
import Spinner from '../components/Spinner';
import moment from 'moment'
const {RangePicker} = DatePicker
function Home() {
  const {cars} = useSelector(state=>state.carsReducer)
  const {loading} = useSelector(state=>state.alertsReducer)
  const [totalCars , setTotalcars] = useState([])
  const dispatch = useDispatch()

  useEffect(() => {
    dispatch(getAllCars())
  }, [])

  useEffect(() => {

    setTotalcars(cars)

  }, [cars])

  // function setFilter(values){

  //   var selectedFrom = moment(values[0] , 'MMM DD yyyy HH:mm')
  //   var selectedTo = moment(values[1] , 'MMM DD yyyy HH:mm')

  //   var temp=[]

```

```

//   for(var car of cars){

//       if(car.bookedTimeSlots.length == 0){
//           temp.push(car)
//       }
//       else{

//           for(var booking of car.bookedTimeSlots) {

//               if(selectedFrom.isBetween(booking.from , booking.to) ||
//               selectedTo.isBetween(booking.from , booking.to) ||
//               moment(booking.from).isBetween(selectedFrom , selectedTo) ||
//               moment(booking.to).isBetween(selectedFrom , selectedTo)
//               )
//               {

//                   }
//               else{
//                   //   temp.push(car)
//                   temp = temp.concat(car);
//               }

//           }

//       }

//   }

//   setTotalcars(temp)

// }

function setFilter(values) {
    const selectedFrom = moment(values[0], 'MMM DD yyyy HH:mm');
    const selectedTo = moment(values[1], 'MMM DD yyyy HH:mm');

    const temp = [];

    for (const car of cars) {
        let hasOverlap = false;

        for (const booking of car.bookedTimeSlots) {
            const bookingFrom = moment(booking.from);
            const bookingTo = moment(booking.to);

```

```

    if (
      selectedFrom.isBetween(bookingFrom, bookingTo, null, '[]') ||
      selectedTo.isBetween(bookingFrom, bookingTo, null, '[]') ||
      bookingFrom.isBetween(selectedFrom, selectedTo, null, '[]') ||
      bookingTo.isBetween(selectedFrom, selectedTo, null, '[]')
    ) {
      // There is an overlap, set flag to true
      hasOverlap = true;
      break; // Exit the loop early, as we've determined there is an overlap
    }
  }

  if (!hasOverlap) {
    // Add the car to the temp array only if there is no overlap
    temp.push(car);
  }
}

setTotalcars(temp);
}

return (
  <DefaultLayout>

    <Row className='mt-3' justify='center'>

      <Col lg={20} sm={24} className='d-flex justify-content-left'>

        <RangePicker showTime={{format: 'HH:mm'}} format='MMM DD yyyy
HH:mm' onChange={setFilter}/>

      </Col>

    </Row>

    {loading === true && (<Spinner/>)}

    <Row justify='center' gutter={16}>

      {totalCars.map(car=>{
        // return <Col lg={5} sm={24} xs={24}>
        return <Col key={car._id} lg={5} sm={24} xs={24}>

          <div className="car p-2 bs1">

```

```

        <img src={car.image} className="carimg"/>

        <div className="car-content d-flex align-items-center justify-content-
between">

            <div className='text-left pl-2'>
                <p>{car.name}</p>
                <p> Rent Per Hour {car.rentPerHour} /-</p>
            </div>

            <div>
                <button className="btn1 mr-2"><Link
to={`/booking/${car._id}`}>Book Now</Link></button>
            </div>

        </div>
    </div>
</Col>
)}}
</Row>

</DefaultLayout>
)
}

```

export default Home

login.js

```

import React from 'react'
import { Row , Col , Form , Input } from 'antd'
import { Link } from 'react-router-dom'
import { useDispatch , useSelector } from 'react-redux'
import { userLogin } from '../redux/actions/userActions'
import AOS from 'aos';
import Spinner from '../components/Spinner';
import 'aos/dist/aos.css'; // You can also use <link> for styles
// ..
AOS.init();
function Login() {
    const dispatch = useDispatch()
    const { loading } = useSelector(state=>state.alertsReducer)
    function onFinish(values) {
        dispatch(userLogin(values))
        console.log(values)
    }
}

```

```

    }
    return (
      <div className='login'>
        {loading && (<Spinner />)}
        <Row gutter={16} className='d-flex align-items-center' >

          <Col lg={16} style={{position: 'relative'}}>
            <img
              className='w-100'
              data-aos='slide-right'
              data-aos-duration='1500'
            />
            { /* <h1 className='login-logo'>ATHIFCARS</h1> */ }
          </Col>
          <Col lg={8} className='text-left p-5'>
            <Form layout='vertical' className='login-form p-5' onFinish={onFinish}>
              <h1>Login</h1>
              <hr />
              <Form.Item name='username' label='Username' rules={[{required: true}]}>
                <Input />
              </Form.Item>
              <Form.Item name='password' label='Password' rules={[{required: true}]}>
                <Input type='password' />
              </Form.Item>

              <button className='btn1 mt-2'>Login</button>

              <hr />

              <Link to='/register'>Click Here to Register</Link>

            </Form>
          </Col>

        </Row>

      </div>
    )
  }
}

export default Login;

```

register.js

```

import React from "react";
import { Row, Col, Form, Input } from "antd";
import { Link } from "react-router-dom";
import { useDispatch, useSelector } from 'react-redux'
import { userRegister } from "../redux/actions/userActions";
import AOS from 'aos';
import Spinner from '../components/Spinner';
import 'aos/dist/aos.css'; // You can also use <link> for styles
// ..
AOS.init()
function Register() {
  const dispatch = useDispatch()
  const { loading } = useSelector(state=>state.alertsReducer)
  function onFinish(values) {
    dispatch(userRegister(values))
    console.log(values)
  }

  return (
    <div className="login">
      {loading && (<Spinner />)}
      <Row gutter={16} className="d-flex align-items-center">
        <Col lg={16} style={{ position: "relative" }}>
          <img
            className='w-100'
            data-aos='slide-left'
            data-aos-duration='1500'
          />
          { /* <h1 className="login-logo">ATHIFCARS</h1> */ }
        </Col>
        <Col lg={8} className="text-left p-5">
          <Form layout="vertical" className="login-form p-5" onFinish={onFinish}>
            <h1>Register</h1>
            <hr />
            <Form.Item
              name="username"
              label="Username"
              rules={[{ required: true }]}
            >
              <Input />
            </Form.Item>
            <Form.Item
              name="password"
              label="Password"
              rules={[{ required: true }]}
            >
              <Input />

```



```

    </Form.Item>
    <Form.Item
      name="cpassword"
      label="Confirm Password"
      rules={[{ required: true }]}
    >
      <Input />
    </Form.Item>

    <button className="btn1 mt-2 mb-3">Register</button>
    <br />

    <Link to="/login">Click Here to Login</Link>
  </Form>
</Col>
</Row>
</div>
);
}

export default Register;

//      src="https://images.unsplash.com/photo-1485291571150-772bcfc10da5?ixlib=rb-
1.2.1&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&
fit=crop&w=928&q=80"

```

userbooking.js

```

import React, { useState, useEffect } from "react";
import DefaultLayout from "../components/DefaultLayout";
import { useDispatch, useSelector } from "react-redux";
import { getAllBookings } from "../redux/actions/bookingActions";
import { Col, Row } from "antd";
import Spinner from "../components/Spinner";
import moment from "moment";
function UserBookings() {
  const dispatch = useDispatch();
  const { bookings } = useSelector((state) => state.bookingsReducer);
  const { loading } = useSelector((state) => state.alertsReducer);
  const user = JSON.parse(localStorage.getItem("user"));
  useEffect(() => {
    dispatch(getAllBookings());
  }, []);

  return (

```

```

<DefaultLayout>
  {loading && (<Spinner />)}
  <h3 className="text-center mt-2">My Bookings</h3>

  <Row justify="center" gutter={16}>
    <Col lg={16} sm={24}>

      {bookings.filter(o=>o.user===user._id).map((booking,index) => {
        return <Row key={index} gutter={16} className="bs1 mt-3 text-left">
          <Col lg={6} sm={24}>
            <p><b>{booking.car.name}</b></p>
            <p>Total hours : <b>{booking.totalHours}</b></p>
            <p>Rent per hour : <b>{booking.car.rentPerHour}</b></p>
            <p>Total amount : <b>{booking.totalAmount}</b></p>
          </Col>

          <Col lg={12} sm={24}>
            <p>Transaction Id : <b>{booking.transactionId}</b></p>
            <p>From: <b>{booking.bookedTimeSlots.from}</b></p>
            <p>To: <b>{booking.bookedTimeSlots.to}</b></p>
            <p>Date of booking:
            <b>{moment(booking.createdAt).format('MMM DD yyyy')}</b></p>
          </Col>

          <Col lg={6} sm={24} className='text-right'>
            <img style={{borderRadius:5}} src={booking.car.image}
            height="140" className="p-2"/>
          </Col>
        </Row>;
      })}

    </Col>
  </Row>
</DefaultLayout>
);
}

```

```
ExportdefaultUserBookings;
```

Booking_actions.js:

```

import axios from "axios";
import { message } from "antd";
export const bookCar = (reqObj) => async (dispatch) => {
  dispatch({ type: "LOADING", payload: true });

```

```

try {
  console.log(reqObj);
  reqObj.amount = calculateAmount(reqObj.totalHours,
reqObj.driverRequired,reqObj.car);
  console.log(reqObj.amount);
  await axios.post("/api/bookings/bookcar" , reqObj);

  dispatch({ type: "LOADING", payload: false });
  message.success("Your car booked successfully");
  setTimeout(() => {
    window.location.href="/userbookings"
  }, 500);

} catch (error) {
  console.log(error);
  console.log(error.response.data);
  // dispatch({ type: "LOADING", payload: false });
  // message.error("Something went wrong , please try later");
  dispatch({ type: "LOADING", payload: false });
  message.success("Your car booked successfully");
  setTimeout(() => {
    window.location.href="/userbookings"
  }, 500);
}

};

const calculateAmount = (totalHours, driverRequired,car) => {
  if(!car)
  {
    console.error('Car object is null or undefined.');
```

return 0;

```

  }
  let amount = totalHours * car.rentPerHour;
  if (driverRequired) {
    amount += 30 * totalHours;
  }
  console.log(amount);
  return amount;
}

```

```

};

export const getAllBookings=()=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    const response = await axios.get('/api/bookings/getallbookings')
    dispatch({type: 'GET_ALL_BOOKINGS', payload:response.data})
    dispatch({type: 'LOADING' , payload:false})
  } catch (error) {
    console.log(error)
    dispatch({type: 'LOADING' , payload:false})
  }

}

import { message } from 'antd';
import axios from 'axios';

export const getAllCars=()=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    const response = await axios.get('/api/cars/getallcars')
    dispatch({type: 'GET_ALL_CARS', payload:response.data})
    dispatch({type: 'LOADING' , payload:false})
  } catch (error) {
    console.log(error)
    dispatch({type: 'LOADING' , payload:false})
  }

}

export const addCar=(reqObj)=>async dispatch=>{

```

```

dispatch({type: 'LOADING' , payload:true})

try {
  await axios.post('/api/cars/addcar' , reqObj)

  dispatch({type: 'LOADING' , payload:false})
  message.success('New car added successfully')
  setTimeout(() => {
    window.location.href='/admin'
  }, 500);
} catch (error) {
  console.log(error)
  dispatch({type: 'LOADING' , payload:false})
}

}

export const editCar=(reqObj)=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    await axios.post('/api/cars/editcar' , reqObj)

    dispatch({type: 'LOADING' , payload:false})
    message.success('Car details updated successfully')
    setTimeout(() => {
      window.location.href='/admin'
    }, 500);
  } catch (error) {
    console.log(error)
    dispatch({type: 'LOADING' , payload:false})
  }
}

```

```

}

export const deleteCar=(reqObj)=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    await axios.post('/api/cars/deletecar' , reqObj)

    dispatch({type: 'LOADING' , payload:false})
    message.success('Car deleted successfully')
    setTimeout(() => {
      window.location.reload()
    }, 500);
  } catch (error) {
    console.log(error)
    dispatch({type: 'LOADING' , payload:false})
  }

}

```

User actions.js:

```

import axios from "axios";
import {message} from 'antd'

export const userLogin=(reqObj)=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    const response = await axios.post('/api/users/login' , reqObj)
    localStorage.setItem('user' , JSON.stringify(response.data))
    message.success('Login success')
  }
}

```

```

    dispatch({type: 'LOADING' , payload:false})
    setTimeout(() => {
      window.location.href="/"

    }, 500);
  } catch (error) {
    console.log(error)
    message.error('Something went wrong')
    dispatch({type: 'LOADING' , payload:false})
  }
}

export const userRegister=(reqObj)=>async dispatch=>{

  dispatch({type: 'LOADING' , payload:true})

  try {
    const response = await axios.post('/api/users/register' , reqObj)
    message.success('Registration successfull')
    setTimeout(() => {
      window.location.href='/login'

    }, 500);

    dispatch({type: 'LOADING' , payload:false})

  } catch (error) {
    console.log(error)
    message.error('Something went wrong')
    dispatch({type: 'LOADING' , payload:false})
  }
}

```

Alert.reducer.js:
const initialData = {

```

    loading : false
  };

export const alertsReducer=(state=initialData , action)=>{

  switch(action.type)
  {
    case 'LOADING' : {
      return{
        ...state,
        loading : action.payload
      }
    }

    default : return state
  }
}

```

Store.js:

```

import { createStore, applyMiddleware , combineReducers } from
'redux';
import { composeWithDevTools } from 'redux-devtools-extension';
import thunk from 'redux-thunk';
import { alertsReducer } from './reducers/alertsReducer';
import { carsReducer } from './reducers/carsReducer';
import { bookingsReducer } from './reducers/bookingsReducer';
const composeEnhancers = composeWithDevTools({});

const rootReducer = combineReducers({
  carsReducer,
  alertsReducer,
  bookingsReducer,
})

const store = createStore(

```



```
    rootReducer,  
    composeEnhancers(  
      applyMiddleware(thunk)  
    )  
  );
```

export default store

app.css:

```
.App {  
  text-align: center;  
}
```

```
.App-logo {  
  height: 40vmin;  
  pointer-events: none;  
}
```

```
@media (prefers-reduced-motion: no-preference) {  
  .App-logo {  
    animation: App-logo-spin infinite 20s linear;  
  }  
}
```

```
.App-header {  
  background-color: #282c34;  
  min-height: 100vh;  
  display: flex;  
  flex-direction: column;  
  align-items: center;  
  justify-content: center;  
  font-size: calc(10px + 2vmin);  
  color: white;  
}
```

```
.App-link {  
  color: #61dafb;  
}
```

```
@keyframes App-logo-spin {  
  from {  
    transform: rotate(0deg);  
  }  
  to {
```

```

    transform: rotate(360deg);
  }
}

```

App.js:

```

import logo from './logo.svg';
import './App.css';
import {Route , BrowserRouter , Redirect} from 'react-router-dom'
import Home from './pages/Home'
import Login from './pages/Login'
import Register from './pages/Register'
import BookingCar from './pages/BookingCar'
import 'antd/dist/antd.css';
import UserBookings from './pages/UserBookings';
import AddCar from './pages/AddCar';
import AdminHome from './pages/AdminHome';
import EditCar from './pages/EditCar';

```

```

function App() {
  return (
    <div className="App">

```

```

      <BrowserRouter>

```

```

        <ProtectedRoute path="/" exact component={Home} />
        <Route path="/login" exact component={Login} />
        <Route path="/register" exact component={Register} />
        <ProtectedRoute path="/booking/:carid" exact
component={BookingCar} />
        <ProtectedRoute path="/userbookings" exact
component={UserBookings} />
        <ProtectedRoute path="/addcar" exact component={AddCar}
/>

```

```

        <ProtectedRoute path='/editcar/:carid' exact
component={EditCar} />
        <ProtectedRoute path='/admin' exact
component={AdminHome} />

    </BrowserRouter>

</div>
);
}

```

```
export default App;
```

```

export function ProtectedRoute(props)
{
    if(localStorage.getItem('user'))
    {
        return <Route {...props}/>
    }
    else{
        return <Redirect to='/login'/>
    }
}
}

```

App.test.js:

```

import { render, screen } from '@testing-library/react';
import App from './App';

test('renders learn react link', () => {
    render(<App />);
    const linkElement = screen.getByText(/learn react/i);
    expect(linkElement).toBeInTheDocument();
});

```

Index.js:

```

import React from 'react';
import ReactDOM from 'react-dom';

```

```

import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import store from './redux/store';
import {Provider} from 'react-redux'
ReactDOM.render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById('root')
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();

```

inde.css:

```

@import
url('https://fonts.googleapis.com/css2?family=Montserrat:wght@500&display=swap');
body {
  margin: 0;
  font-family: 'Montserrat', sans-serif !important;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  background-color: azure;
}
body, html{
  overflow-x: hidden !important;
}
code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
.bs1 {
  box-shadow: rgba(14, 30, 37, 0.12) 0px 2px 4px 0px, rgba(14, 30, 37, 0.32) 0px 2px 16px
0px;
}
.ant-row{
  margin: 0 20 !important;
}
.spinner{
  position: absolute;
  top: 50%;
  left: 50%;
  z-index: 9999;
}

```

```

h1 {
  font-size: 30px;
  font-weight: bold;
  color : orangered;
}
p {
  margin: 5px !important;

  padding: 0;
}
.btn1 {
  box-shadow: none !important;
  background-color : white !important;
  color: orangered !important;
  border: 1px solid orangered !important;
  padding: 5px 15px !important;
  outline: none !important;
}
.btn1:focus {
  border: none;
}
button {
  padding: 3px 10px !important;
}
.header {
  padding: 10px;
}

.header .ant-btn {
  box-shadow: none !important;
  background-color : white !important;
  color: orangered !important;
  border: 1px solid orangered !important;
  padding: 5px 15px !important;
  outline: none !important;
}
.content {
  min-height: 90vh;
}
.footer {
  background-color: aliceblue;
  padding: 10px;
}

/* this is just for ui purpose , no functionality is related */
.car {

```

```

border-radius: 5px;
margin-top: 100px;
height: 160px;
transition: 0.3s;
}
.car:hover{
  height :220px;
}
.car:hover > .car-content{

  opacity: 1;
  transform : translateY(-40px)

}
.carimg{
  height:150px;
  width: auto;
  border-radius: 5px;
  transform: translateY(-50px);
}
.car-content{
  transform: translateY(-200px);
  opacity: 0;
  transition: 0.3s;
}
.login{
  background-color: black;
  padding-top: 50px;
  min-height: 100vh;
}
.login label , .login h1 {
  color: white !important;
  opacity: 0.6;
}
.login-form{
  background-color: #1F1F1F;
}
.login-form input{
  background-color: #333333 !important;
  border : none !important;
  color: white;
  opacity: 0.5;
}
.login-logo{
  position: absolute;
  left:0;
  right:0;

```

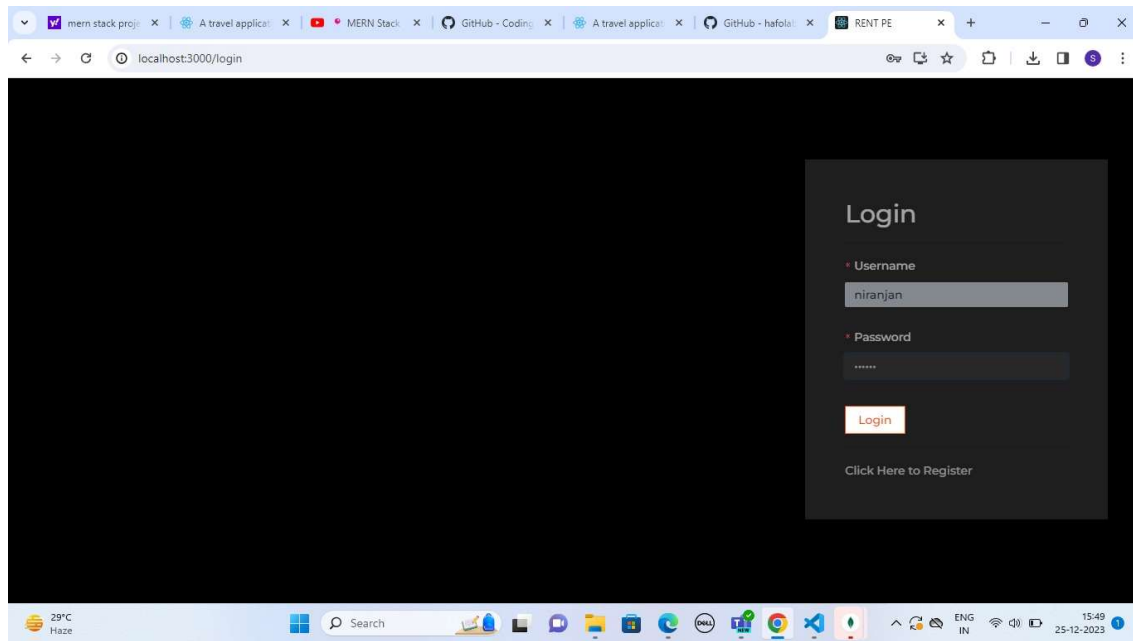
```
top: 60%;  
bottom: 50%;  
transform: translateY(-50% , -50%);  
font-size:45px;  
opacity: 0.2 !important;
```

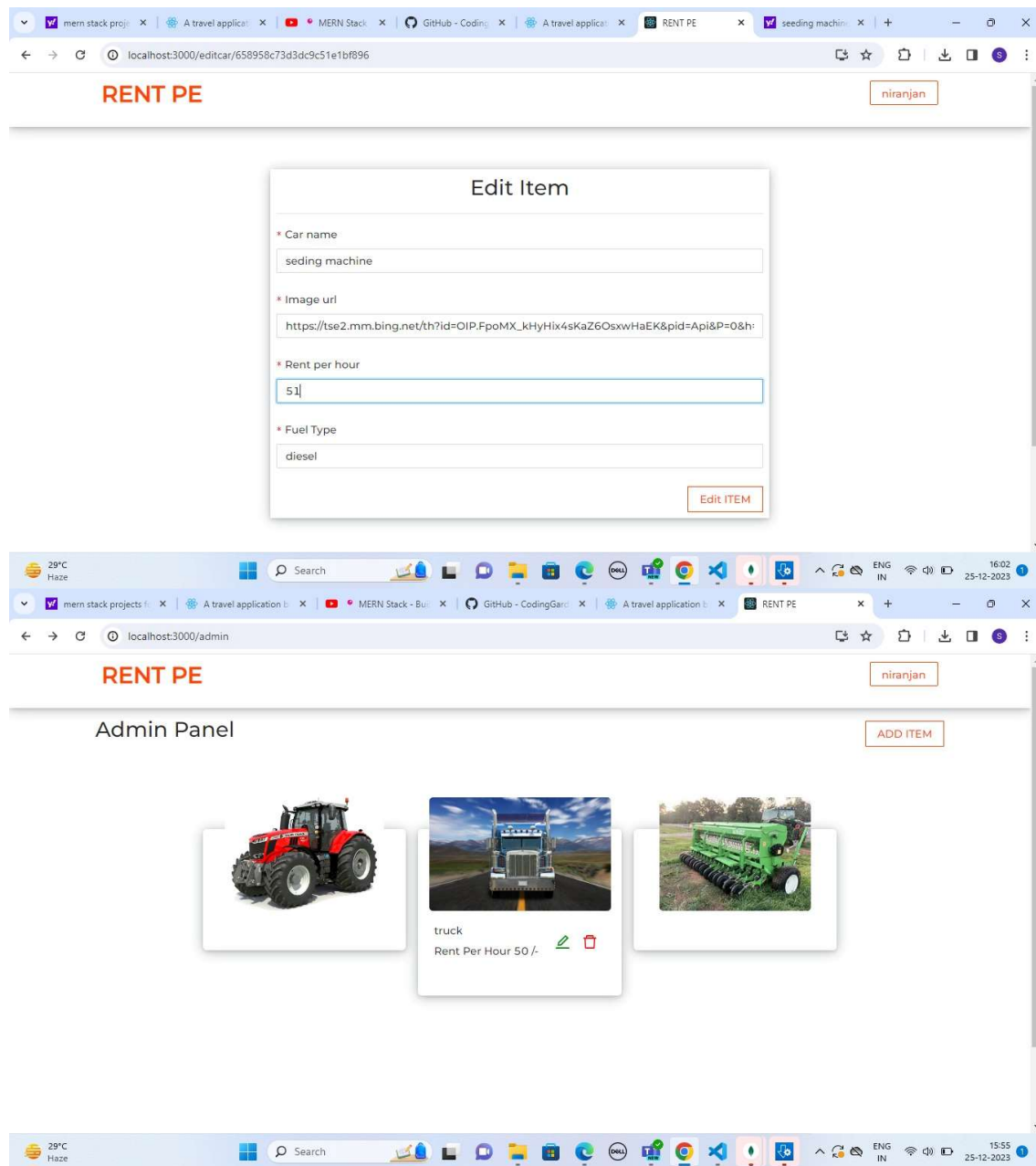
```
}  
.login a{  
  color: white !important;  
  opacity: 0.5;  
}
```

```
.carimg2{  
  height: 400px;  
  border-radius: 10px;  
  width: auto;  
}
```

```
svg{  
  height:20px !important;  
  width : 20px !important;  
}  
a{  
  color: orangered !important;  
}
```

Outputs:







RENT PE

niranjan

Admin Panel


ADD ITEM





truck

Rent Per Hour 50 /-



29°C Haze

Search

localhost:3000/userbookings

tractor

Total hours : 24

Rent per hour : 100


Total amount : 2400

Transaction Id :

From: Dec 26 2023 12:16

To: Dec 27 2023 12:16

Date of booking: Dec 11 2023



seding machine

Total hours : 24

Rent per hour : 50


Total amount : 1200

Transaction Id :

From: Dec 26 2023 12:37

To: Dec 27 2023 12:37

Date of booking: Dec 11 2023



tractor

Total hours : 72

Rent per hour : 100


Total amount : 7200

Transaction Id :

From: Dec 28 2023 13:46

To: Dec 31 2023 13:46

Date of booking: Dec 11 2023



truck

Total hours : 72

Rent per hour : 50


Total amount : 3600

Transaction Id :

From: Dec 28 2023 15:53

To: Dec 31 2023 15:53

Date of booking: Dec 25 2023

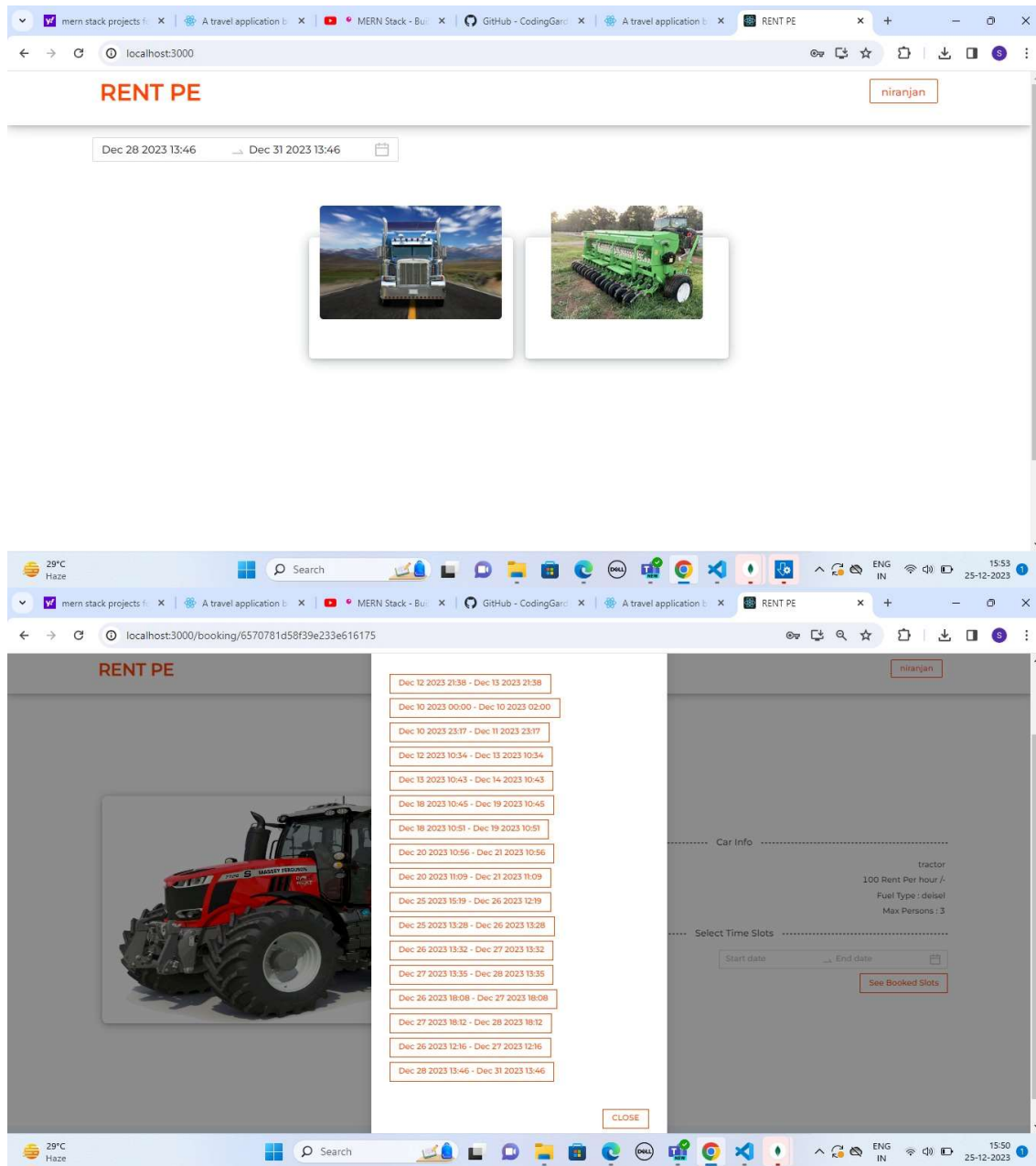


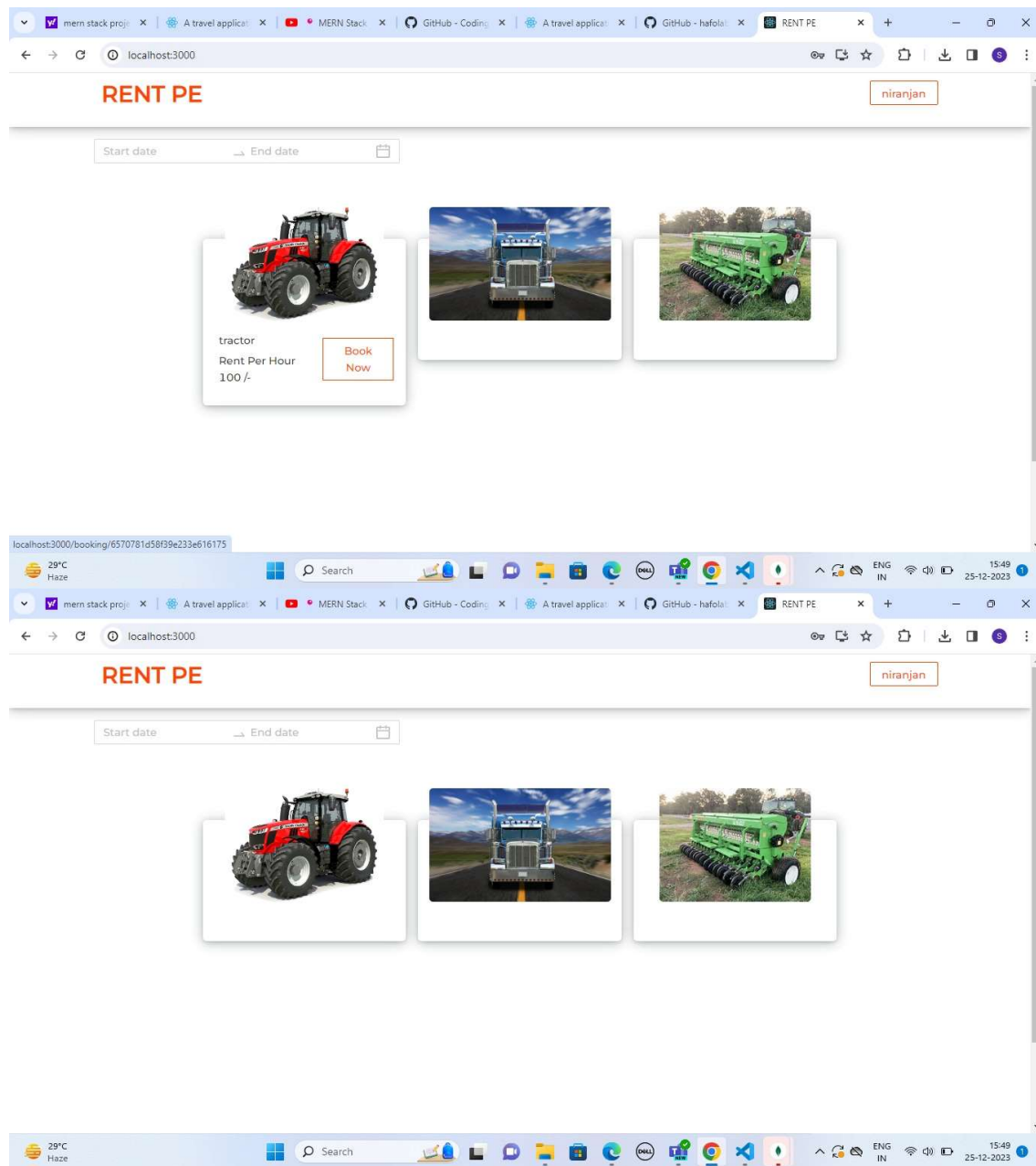
29°C Haze

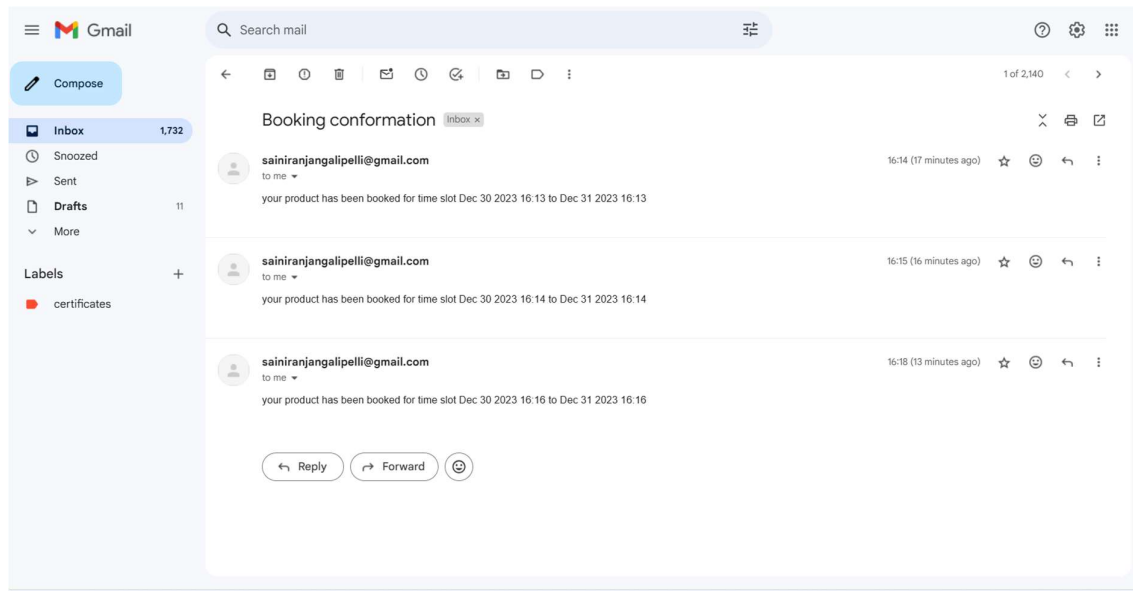
Search

localhost:3000/userbookings

15:54 25-12-2023







Conclusion :

This project, centered around renting farming tools, brings a transformative approach to the agriculture sector. By incorporating Redux and React in the frontend, the platform ensures an intuitive and dynamic user interface, simplifying the tool renting process for farmers. The utilization of CSS and HTML contributes to an aesthetically pleasing and user-friendly design, enhancing the overall user experience. In the backend, the combination of Node.js and Mongoose enables efficient data management, ensuring that farmers can seamlessly access and return tools as needed. This project leverages a versatile stack of technologies to address the specific challenges of tool accessibility, cost-efficiency, and resource utilization in farming practices.

Future Work for College Space:

The College Space is an evolving platform, and its future enhancements aim to make it even more dynamic, interactive, and tailored to the diverse needs of students across various domains. Here's a roadmap for potential future developments:

1. Tech Language Symphony:

- JavaScript and Python are the maestros orchestrating the backend. Each function and module is a result of persistent coding efforts, ensuring a robust and responsive user experience.

2. Structured Databases:

- MySQL takes center stage for databases. It's not just about storing data; it's about creating a structured foundation where every piece contributes to the coherent functionality of the platform.

3. Front-End Artistry:

- HTML, CSS, and JavaScript come together on the front end. Every button, color, and layout is carefully designed through numerous iterations, incorporating user feedback for an intuitive and visually appealing interface.

4. Strategic Future Plans:

- The roadmap for future developments is not a mere wishlist; it's a strategic plan. It involves collaboration, user surveys, and an eye on emerging technologies to ensure that each enhancement aligns with the evolving needs of the user base.

5. Security in Every Line:

- Blockchain isn't just a buzzword; it's a deliberate choice for security. Each encrypted block is a layer of protection, showcasing the commitment to safeguarding user data.

References:

1. <https://react.dev/learn/adding-interactivity>
2. <https://react.dev/learn/updating-objects-in-state>
3. <https://www.mongodb.com/docs/compass/current/>
4. <https://youtu.be/zQAdZYxbH14?si=kkP-YGjJalFtbedR>
5. <https://youtu.be/-jEuLl-wuxc?si=3s6l9r9KfooTp7v4>
6. <https://www.figma.com/file/tPJpjhv6bXquURkNLo6efw/Untitled?type=design&node-id=12-41&mode=design&t=buTJkEYjnEU6DvyU-0>