REALTIME PETROL PRICE STREAMER

TEAM MEMBERS:

|  |  |
| --- | --- |
| **ROLLNO** | **NAME** |
| 22ALR050 | MITHUN KUMAR B |
| 22ALR017 | ELAKKIYAN M |
| 22ALL117 | AJAY P S |

**INTRODUCTION:**

Welcome to Realtime Petrol price Streamer, the premier platform for real-time price streaming of petrol, diesel, and oil. Our website is designed to provide users with instant updates on fuel prices, catering to commuters, businesses, and industry analysts alike. With a user-friendly interface and customizable features, you can set alerts based on price thresholds and receive notifications in real-time, ensuring you never miss a crucial update. Realtime Petrol price Streamer 's live streaming capabilities allow you to monitor fluctuations and trends as they happen, empowering you to make informed decisions swiftly and effectively. In addition to live prices, our platform offers extensive historical data and in-depth market analysis, providing valuable insights into the factors driving fuel price changes. We prioritize accuracy and reliability, sourcing data from trusted providers and continuously updating information to maintain precision. Join our community today and experience the convenience and efficiency of real-time price streaming with Realtime Petrol price Streamer.

**Key features:**

1.A Login and Registration page was given for the user to connect with their account.

2.User data are stored in mongodb database.

3.We came to see the home page of the website which will have the link to connect with other streaming page.

4.There are three pages in the application with which we can find the price of the product.

5.The three pages have been connected with api to get price of the product.

6.Given with the dates selected with input we get the json string with we can find the price of the product.

**REQUIREMENTS:**

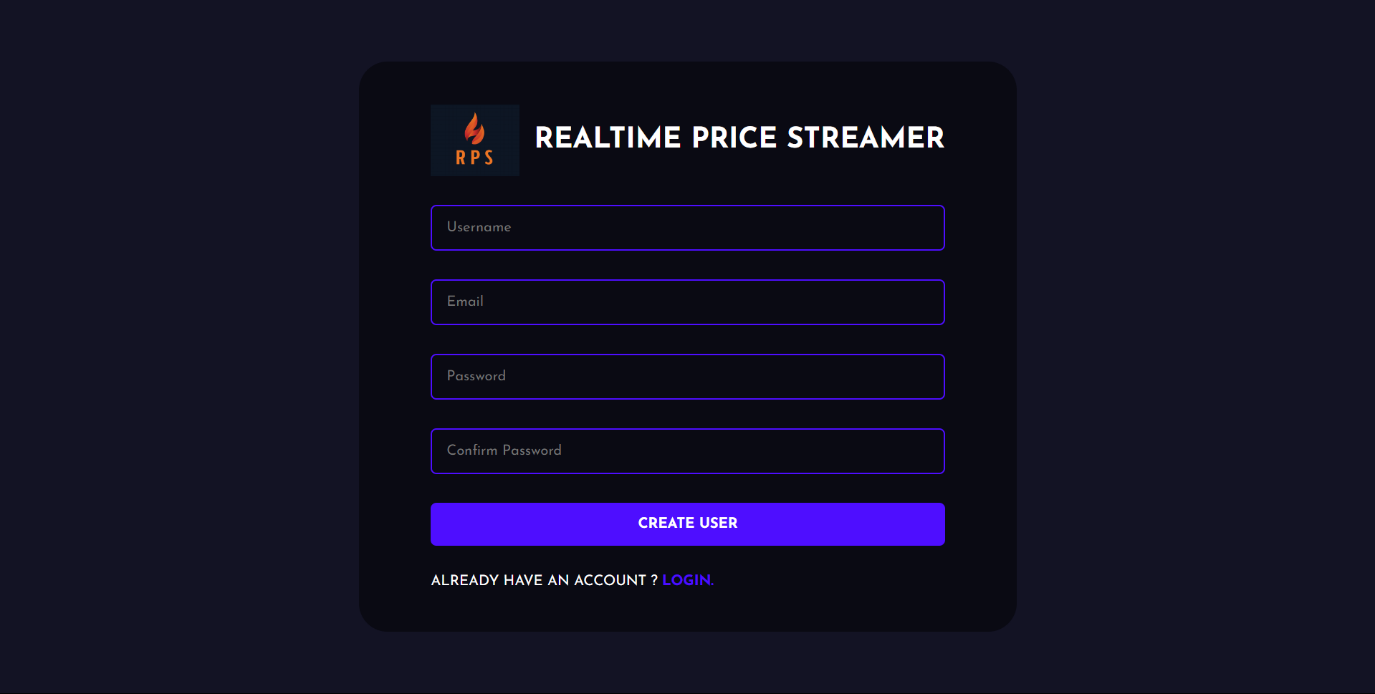
**Front-End Programming**

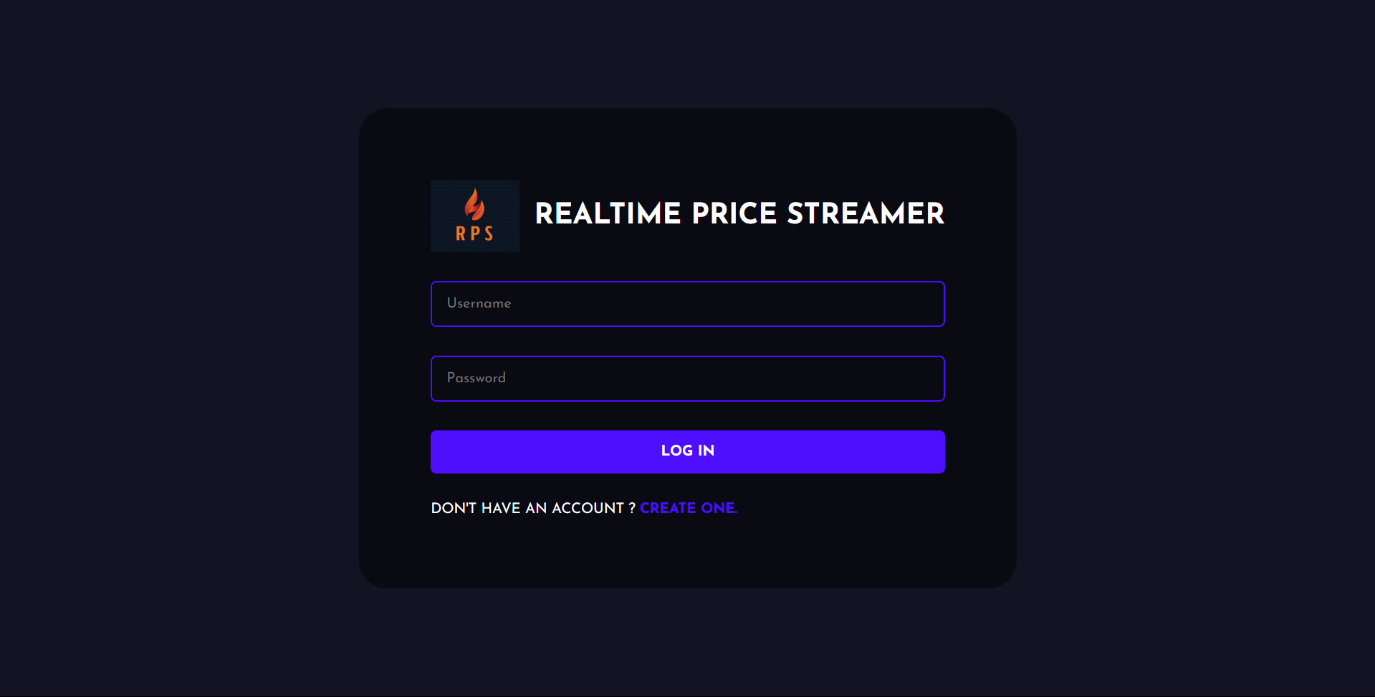
1. **CSS (Cascading Style Sheets):** CSS is a stylesheet language used to define the visual presentation of HTML documents, controlling layout, colors, fonts, and more. It enables developers to create visually appealing and consistent designs across multiple web pages. With CSS, you can implement responsive design techniques to ensure web pages look great on various devices and screen sizes. CSS offers a range of styling capabilities, from basic text formatting to complex animations and grid layouts. Its separation from HTML enhances maintainability and flexibility in web development.
2. **JavaScript:** JavaScript is a versatile programming language widely used for creating interactive web applications. It allows developers to manipulate HTML and CSS dynamically, enabling real-time updates without page reloads. As an event-driven language, JavaScript responds to user actions like clicks and key presses, enhancing user experience. Its support for asynchronous programming helps manage tasks such as data fetching efficiently. JavaScript's extensive ecosystem includes numerous libraries and frameworks like React and Angular, which simplify and accelerate development.
3. **React:** React is a popular JavaScript library for building user interfaces, particularly single-page applications. Developed by Facebook, it allows developers to create reusable UI components that manage their own state. React's virtual DOM improves performance by efficiently updating and rendering components only when data changes. It supports server-side rendering, enhancing SEO and initial load times. With a strong ecosystem and community, React integrates well with other libraries and frameworks, making it a versatile choice for modern web development.

**Back-End Programming**

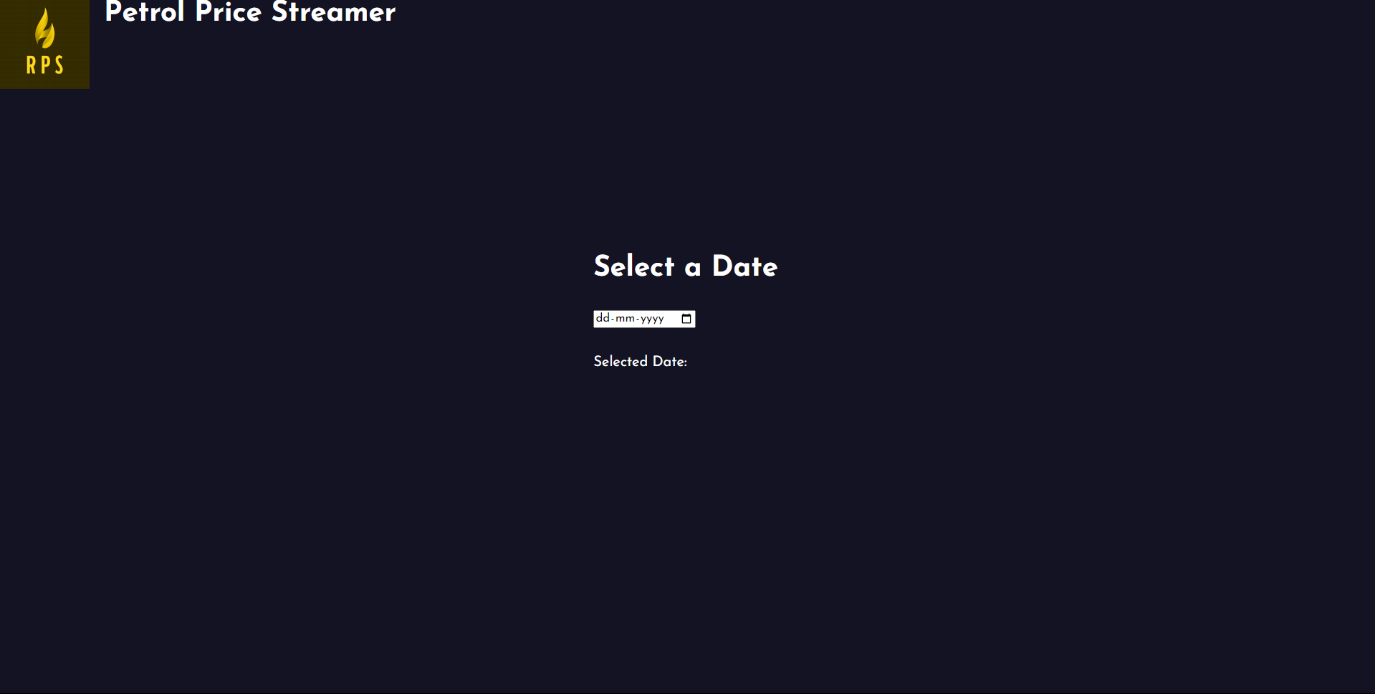
1. **MongoDB:** MongoDB is a NoSQL database that stores data in JSON-like documents, offering flexibility and scalability. It's widely used for handling large volumes of unstructured data and is known for its high performance and availability. MongoDB supports features like sharding for horizontal scaling and replication for fault tolerance. Its query language allows for complex queries and aggregation operations, making it suitable for diverse applications. With built-in support for indexing and full-text search, MongoDB streamlines data retrieval and analysis. Its robust community and extensive documentation ensure comprehensive support for developers.
2. **APIs (Application Programming Interfaces):** An API, or Application Programming Interface, acts as a bridge between different software applications, allowing them to communicate and interact seamlessly. APIs define the methods and protocols that applications can use to request and exchange data, enabling developers to integrate functionalities from one system into another. They are essential for building modern web and mobile applications, enabling them to access and utilize external services and resources efficiently. APIs can be public, private, or partner-specific, depending on the intended audience and use case.

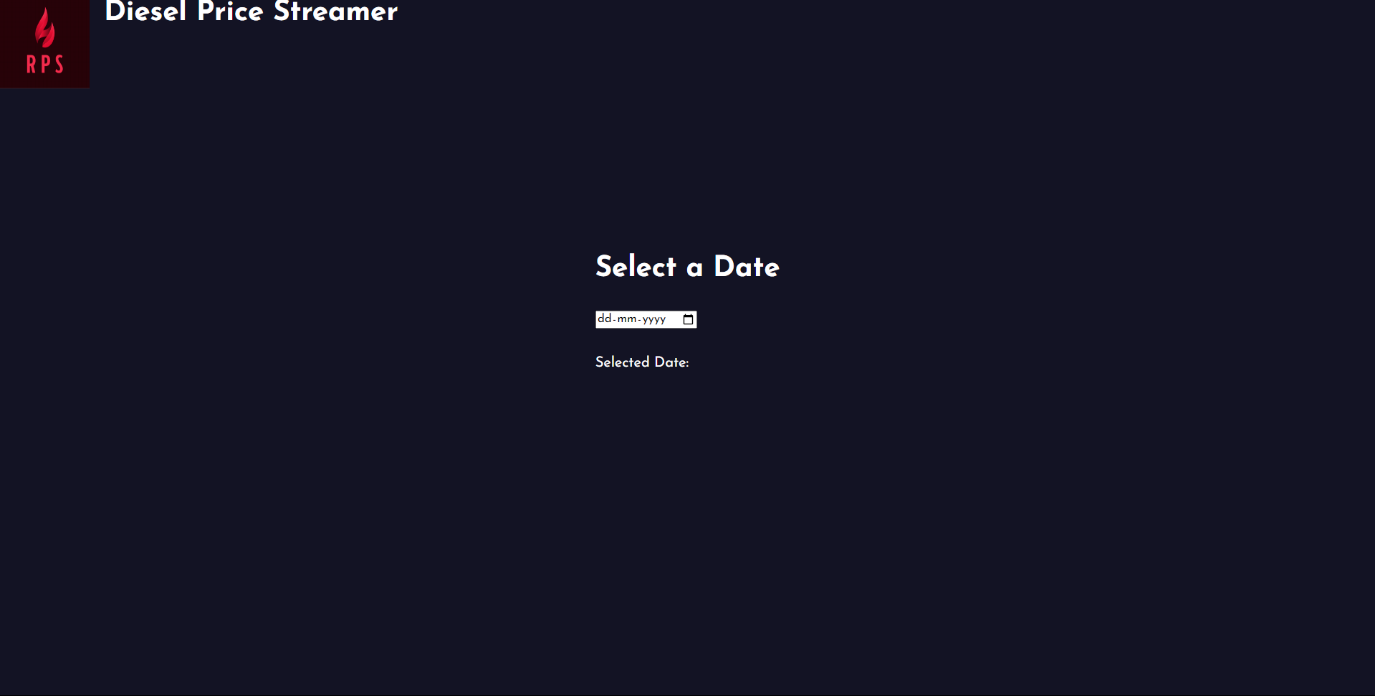
**UI DESIGN:**

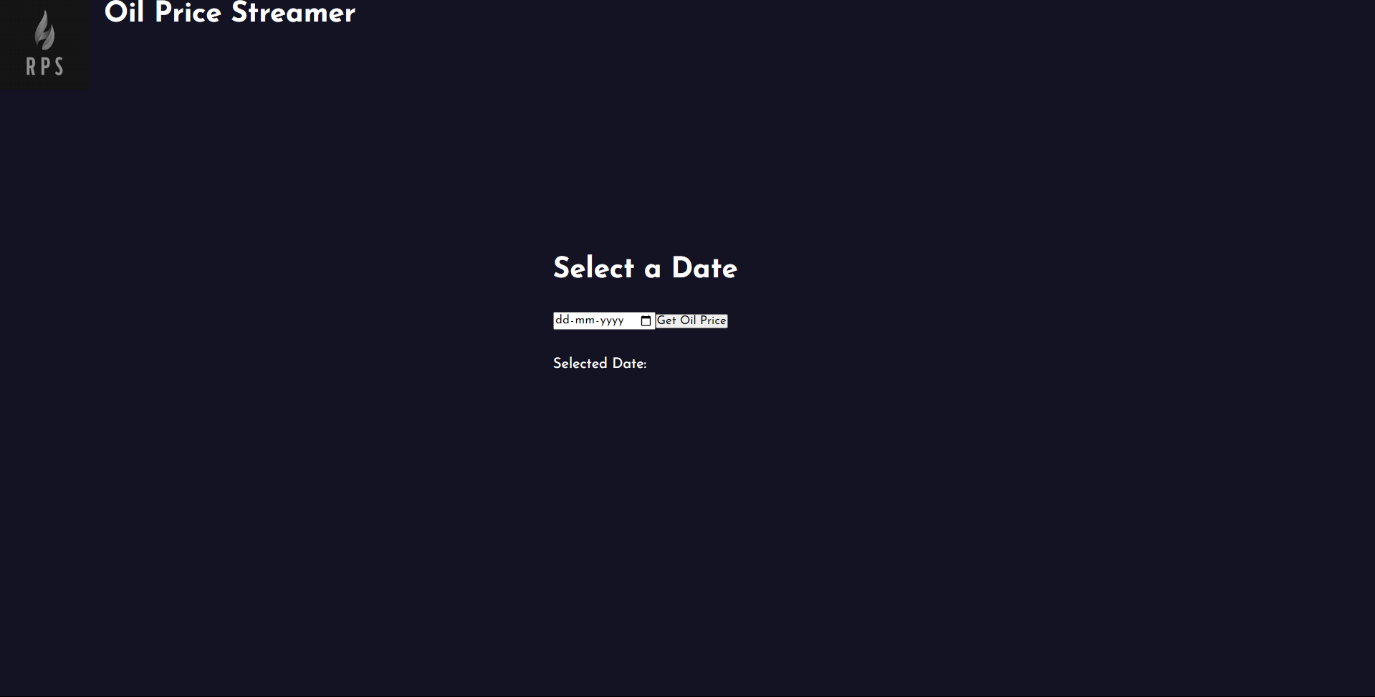
****

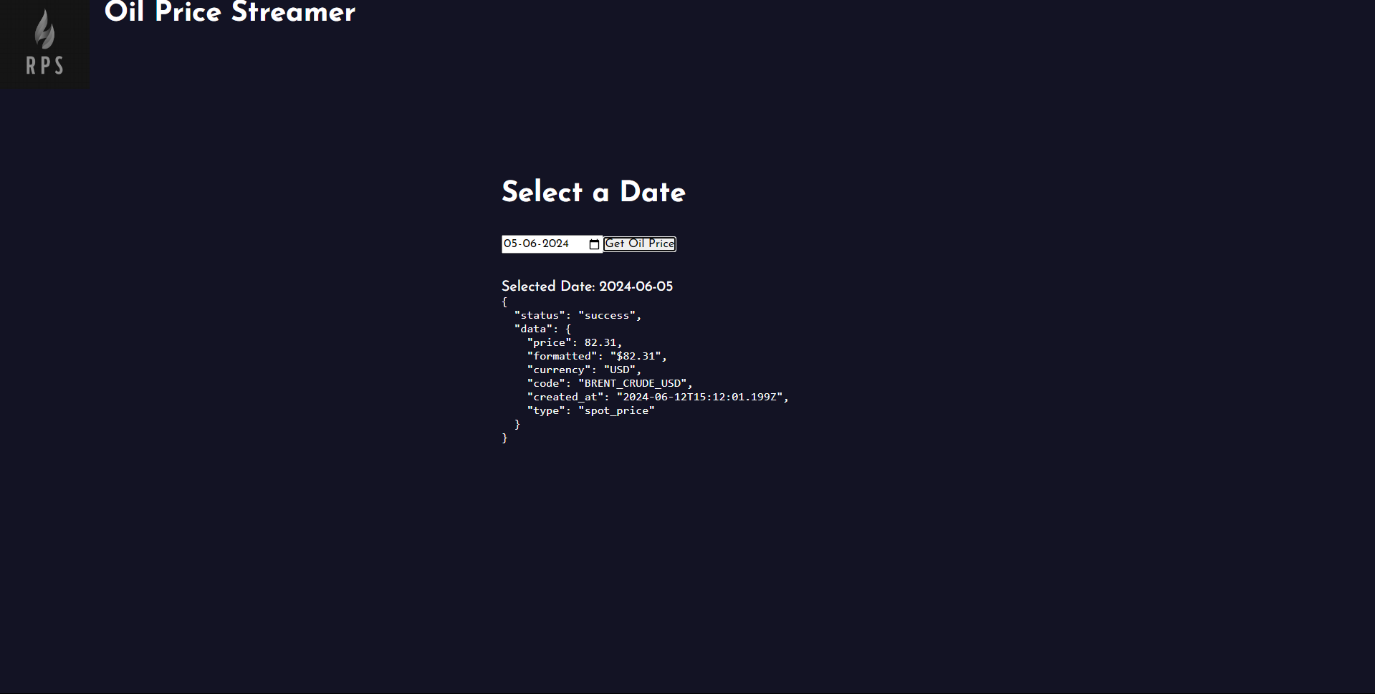












**SAMPLE CODING:**

**BACKEND:**

INDEX.JS

const express = require("express");

const cors = require("cors");

const mongoose = require("mongoose");

const authRoutes = require("./routes/auth");

const messageRoutes = require("./routes/messages");

const app = express();

const socket = require("socket.io");

require("dotenv").config();

app.use(cors());

app.use(express.json());

mongoose

.connect("mongodb+srv://mithunkumarb22aim:JtuZm3ohnLL0g11C@cluster0.huuomm5.mongodb.net/loginpage?retryWrites=true&w=majority&appName=Cluster0")

.then(() => {console.log("DB Connetion Successfull");})

.catch((err) => {console.log(err.message);});

app.use("/api/auth", authRoutes);

app.use("/api/messages", messageRoutes);

const server = app.listen(process.env.PORT, () =>console.log(`Server started on ${process.env.PORT}`));

const io = socket(server, {cors: {origin: "http://127.0.0.1:3000",credentials: true,},});

global.onlineUsers = new Map();

io.on("connection", (socket) => {global.chatSocket = socket;socket.on("add-user", (userId) => {onlineUsers.set(userId, socket.id); });

socket.on("send-msg", (data) => {const sendUserSocket = onlineUsers.get(data.to);

if (sendUserSocket) {socket.to(sendUserSocket).emit("msg-recieve", data.msg);}

});

});

**FRONTEND:**

LOGIN.JS

import React, { useState, useEffect } from "react";

import axios from "axios";

import styled from "styled-components";

import { useNavigate, Link } from "react-router-dom";

import { ToastContainer, toast } from "react-toastify";

import "react-toastify/dist/ReactToastify.css";

import { loginRoute } from "../utils/APIRoutes";

import image1 from './image4.png';

export default function Login() {const navigate = useNavigate();const [values, setValues] = useState({ username: "", password: "" });

const toastOptions = {position: "bottom-right",autoClose: 8000,pauseOnHover: true,draggable: true,theme: "dark",};

useEffect(() => {if (localStorage.getItem(process.env.REACT\_APP\_LOCALHOST\_KEY)) {navigate("/");}}, []);

const handleChange = (event) => {setValues({ ...values, [event.target.name]: event.target.value });};

const validateForm = () => {const { username, password } = values;

if (username === "") {toast.error("Email and Password is required.", toastOptions);return false;}

else if (password === "") {toast.error("Email and Password is required.", toastOptions);return false;}

return true;

};

const handleSubmit = async (event) => {event.preventDefault();

if (validateForm()) {const { username, password } = values;const { data } = await axios.post(loginRoute, {username,password,});

if (data.status === false) {toast.error(data.msg, toastOptions);}

if (data.status === true) {localStorage.setItem(process.env.REACT\_APP\_LOCALHOST\_KEY,JSON.stringify(data.user));navigate("/");}}

};

return (

<>

<FormContainer>

<form action="" onSubmit={(event) => handleSubmit(event)}>

<div className="brand">

<img src={image1} alt="Image 1" className="button-image" />

<h1>REALTIME PRICE STREAMER</h1>

</div>

<input type="text" placeholder="Username" name="username" onChange={(e) => handleChange(e)} min="3"/>

<input type="password" placeholder="Password" name="password" onChange={(e) => handleChange(e)}/>

<button type="submit">Log In</button>

<span> Don't have an account ? <Link to="/register">Create One.</Link></span>

</form>

</FormContainer>

<ToastContainer />

</>

);

}

const FormContainer = styled.div`

height: 100vh;width: 100vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;align-items: center;background-color: #131324;

.brand {display: flex;align-items: center;gap: 1rem;justify-content: center;img {height: 5rem;}

h1 {color: white;text-transform: uppercase;}}

form {display: flex; flex-direction: column;gap: 2rem;background-color: #00000076;border-radius: 2rem;padding: 5rem;}

input {background-color: transparent;padding: 1rem;border: 0.1rem solid #4e0eff;border-radius: 0.4rem;color: white;width: 100%;font-size: 1rem;

&:focus {border: 0.1rem solid #997af0;outline: none;}}

button {background-color: #4e0eff;color: white;padding: 1rem 2rem;border: none;font-weight: bold;

cursor: pointer;border-radius: 0.4rem;font-size: 1rem;text-transform: uppercase;

&:hover {background-color: #4e0eff;}}

span {color: white;text-transform: uppercase;a {color: #4e0eff;text-decoration: none;font-weight: bold;}}

`;

REGISTER.JS

import React, { useState, useEffect } from "react";

import axios from "axios";

import styled from "styled-components";

import { useNavigate, Link } from "react-router-dom";

import { ToastContainer, toast } from "react-toastify";

import "react-toastify/dist/ReactToastify.css";

import { registerRoute } from "../utils/APIRoutes";

import image1 from './image4.png';

export default function Register() { const navigate = useNavigate();

const toastOptions = { position: "bottom-right",autoClose: 8000,pauseOnHover: true,draggable: true,theme: "dark",};

const [values, setValues] = useState({username: "",email: "",password: "",confirmPassword: "",});

useEffect(() => {if (localStorage.getItem(process.env.REACT\_APP\_LOCALHOST\_KEY)) {navigate("/");}}, []);

const handleChange = (event) => {setValues({ ...values, [event.target.name]: event.target.value });};

const handleValidation = () => {const { password, confirmPassword, username, email } = values;

if (password !== confirmPassword) {toast.error("Password and confirm password should be same.",toastOptions);return false;}

else if (username.length < 3) {toast.error("Username should be greater than 3 characters.",toastOptions);return false;}

else if (password.length < 8) {toast.error("Password should be equal or greater than 8 characters.",toastOptions);return false;}

else if (email === "") {toast.error("Email is required.", toastOptions);return false;}return true;};

const handleSubmit = async (event) => {event.preventDefault();

if (handleValidation()) {const { email, username, password } = values;

const { data } = await axios.post(registerRoute, {username,email,password,});

if (data.status === false) { toast.error(data.msg, toastOptions);}

if (data.status === true) {localStorage.setItem(process.env.REACT\_APP\_LOCALHOST\_KEY,JSON.stringify(data.user)); navigate("/");}}

};

return (

<>

<FormContainer>

<form action="" onSubmit={(event) => handleSubmit(event)}>

<div className="brand">

<img src={image1} alt="Image 1" className="button-image" />

<h1>REALTIME PRICE STREAMER</h1>

</div>

<input type="text" placeholder="Username"name="username"onChange={(e) => handleChange(e)}/>

<input type="email"placeholder="Email"name="email"onChange={(e) => handleChange(e)}/>

<input type="password"placeholder="Password"name="password"onChange={(e) => handleChange(e)}/>

<input type="password"placeholder="Confirm Password"name="confirmPassword"onChange={(e) => handleChange(e)}/>

<button type="submit">Create User</button>

<span>Already have an account ? <Link to="/login">Login.</Link></span>

</form>

</FormContainer>

<ToastContainer />

</>

);

}

const FormContainer = styled.div`

height: 100vh;width: 100vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;align-items: center;background-color: #131324;

.brand {display: flex;align-items: center;gap: 1rem;justify-content: center;img {height: 5rem;}

h1 {color: white;text-transform: uppercase;}}

form {display: flex; flex-direction: column;gap: 2rem;background-color: #00000076;border-radius: 2rem;padding: 5rem;}

input {background-color: transparent;padding: 1rem;border: 0.1rem solid #4e0eff;border-radius: 0.4rem;color: white;width: 100%;font-size: 1rem;

&:focus {border: 0.1rem solid #997af0;outline: none;}}

button {background-color: #4e0eff;color: white;padding: 1rem 2rem;border: none;font-weight: bold;

cursor: pointer;border-radius: 0.4rem;font-size: 1rem;text-transform: uppercase;

&:hover {background-color: #4e0eff;}}

span {color: white;text-transform: uppercase;a {color: #4e0eff;text-decoration: none;font-weight: bold;}}

`;

PAGE1.JS

import React, { useState } from 'react';

import styled from "styled-components";

import image1 from './image1.png';

const Page1 = () => {const [selectedDate, setSelectedDate] = useState('');

const handleDateChange = (event) => {setSelectedDate(event.target.value);}

return (

<>

<Container1>

<div><img style={{height:"100px",width:"100px"}}src={image1} alt="Image 1" className="button-image" /></div>

<div><h1 style={{ marginBottom: '30px' }}>Petrol Price Streamer</h1></div>

<Container>

<div ><h1 style={{ marginBottom: '30px' }}>Select a Date</h1>

<input style={{ marginBottom: '30px' }}type="date" value={selectedDate} onChange={handleDateChange} />

<p>Selected Date: {selectedDate}</p>

</div>

</Container>

</Container1>

</>

);

};

const Container1 = styled.div`height: 100vh;width: 100vw;display: flex;gap: 1rem;background-color: #131324;color: white`;

const Container = styled.div`height: 90vh;width: 40vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;

align-items: center;background-color: #131324;color: white;`;

export default Page1;

PAGE2.JS

import React, { useState } from 'react';

import styled from "styled-components";

import image1 from './image2.png';

const Page2 = () => {const [selectedDate, setSelectedDate] = useState('');

const handleDateChange = (event) => {setSelectedDate(event.target.value);}

return (

<>

<Container1>

<div><img style={{height:"100px",width:"100px"}}src={image1} alt="Image 1" className="button-image" /></div>

<div><h1 style={{ marginBottom: '30px' }}>Diesel Price Streamer</h1></div>

<Container>

<div > <h1 style={{ marginBottom: '30px' }}>Select a Date</h1>

<input style={{ marginBottom: '30px' }}type="date" value={selectedDate} onChange={handleDateChange} />

<p>Selected Date: {selectedDate}</p>

</div>

</Container>

</Container1>

</>

);

};

const Container1 = styled.div`height: 100vh;width: 100vw;display: flex;gap: 1rem;background-color: #131324;color: white`;

const Container = styled.div`height: 90vh;width: 40vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;

align-items: center;background-color: #131324;color: white;`;

export default Page2;

PAGE3.JS

import React, { useState } from 'react';

import styled from 'styled-components';

import image1 from './image3.png';

const Page3 = () => { const [selectedDate, setSelectedDate] = useState('');

const [price, setPrice] = useState(null);const [error, setError] = useState(null);

const apiKey = '5171f7b9fd3a2dafbbf4a3e011929917';const baseUrl = 'https://api.oilpriceapi.com/v1/prices/latest';

const handleDateChange = (event) => {setSelectedDate(event.target.value);};

const fetchOilPrice = async () => {

try {const response = await fetch(`${baseUrl}?date=${selectedDate}`, {headers: {'Authorization': `Token ${apiKey}`,'Content-Type': 'application/json',},});

if (!response.ok) {throw new Error('Network response was not ok ' + response.statusText);}

const data = await response.json();setPrice(data);setError(null);

} catch (error) {setError(error.message); setPrice(null);}

};

const handleSubmit = (event) => {event.preventDefault(); fetchOilPrice();};

return (

<>

<Container1>

<div><img style={{ height: '100px', width: '100px' }}src={image1}alt="Image 1"className="button-image"/></div>

<div><h1 style={{ marginBottom: '30px' }}>Oil Price Streamer</h1></div>

<Container>

<div><h1 style={{ marginBottom: '30px' }}>Select a Date</h1><form onSubmit={handleSubmit}>

<input style={{ marginBottom: '30px' }}type="date" value={selectedDate}onChange={handleDateChange}/>

<button type="submit">Get Oil Price</button>

</form><p>Selected Date: {selectedDate}</p>{price && <pre>{JSON.stringify(price, null, 2)}</pre>}{error && <p style={{ color: 'red' }}>Error: {error}</p>}

</div>

</Container>

</Container1>

</>

);

};

const Container1 = styled.div`height: 100vh;width: 100vw;display: flex;gap: 1rem;background-color: #131324;color: white`;

const Container = styled.div`height: 90vh;width: 40vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;`

export default Page3;

HOME.JS

import React, { useEffect, useState, useRef } from "react";

import axios from "axios";

import { useNavigate } from "react-router-dom";

import { io } from "socket.io-client";

import styled from "styled-components";

import { allUsersRoute, host } from "../utils/APIRoutes";

import ChatContainer from "../components/ChatContainer";

import Contacts from "../components/Contacts";

import Welcome from "../components/Welcome";

import Page1 from "./page1"

import Page2 from "./page2"

import Page3 from "./page3"

export default function Chat() {const navigate = useNavigate();const socket = useRef();const [contacts, setContacts] = useState([]);const [currentChat, setCurrentChat] = useState(undefined);const [currentUser, setCurrentUser] = useState(undefined);

useEffect(async () => {if (!localStorage.getItem(process.env.REACT\_APP\_LOCALHOST\_KEY)) {navigate("/login");}

else {setCurrentUser(await JSON.parse(localStorage.getItem(process.env.REACT\_APP\_LOCALHOST\_KEY)));}}, []);

useEffect(() => {if (currentUser) {socket.current = io(host);socket.current.emit("add-user", currentUser.\_id);}}, [currentUser]);

useEffect(async () => {if (currentUser) {

if (currentUser.isAvatarImageSet) {const data = await axios.get(`${allUsersRoute}/${currentUser.\_id}`);setContacts(data.data);}

else {navigate("/setAvatar");}}}, [currentUser]);

const handleChatChange = (chat) => {setCurrentChat(chat);};

const [currentPage, setCurrentPage] = useState(null);

const nextPage = (page) => {switch (page) {

case 1:setCurrentPage('/Page1');break;

case 2:setCurrentPage('/Page2');break;

case 3:setCurrentPage('/Page3');break;

default:break;}};

useEffect(() => { if (currentPage) {window.location.href = currentPage;}}, [currentPage]);

return (

<>

<Container>

<div style={{ color: 'White' }} display="flex"><div><h1>Welcome to Realtime Price Streamer</h1></div>

<Container1>

<div className="button-container"><div style={{height:"100px",width:"100px"}} className="button" onClick={() => nextPage(1)}><span>Petrol Price</span></div>

<div style={{height:"100px",width:"100px"}} className="button" onClick={() => nextPage(2)}><span>Diesel Price</span></div>

<div style={{height:"100px",width:"100px"}} className="button" onClick={() => nextPage(3)}><span>Oil Price</span></div></div>

</Container1></div>

</Container>

</>

);

}

const Container = styled.div`height: 100vh;width: 100vw;display: flex;flex-direction: column;justify-content: center;gap: 1rem;align-items: center;background-color: #131324;`;

const Container1 = styled.div`height: 90vh;width: 100vw;display: flex;flex-direction: row;justify-content: center;gap: 1rem;align-items: center;background-color: #131324;color: white;`;

APP.JS

import React from "react";

import { BrowserRouter, Routes, Route , Link, Switch } from "react-router-dom";

import SetAvatar from "./components/SetAvatar";

import Chat from "./pages/Chat";

import Login from "./pages/Login";

import Register from "./pages/Register";

import Page1 from "./pages/page1";

import Page2 from "./pages/page2";

import Page3 from "./pages/page3";

export default function App() {

return (

<BrowserRouter>

<Routes>

<Route path="/register" element={<Register />} />

<Route path="/login" element={<Login />} />

<Route path="/setAvatar" element={<SetAvatar />} />

<Route path="/" element={<Chat />} />

<Route path="/page1" element={<Page1 />} />

<Route path="/page2" element={<Page2 />} />

<Route path="/page3" element={<Page3 />} />

</Routes>

</BrowserRouter>

);

}

**CONCLUSION:**

In conclusion, Realtime Petrol Price Streamer emerges as an essential tool for individuals and businesses seeking to stay informed and make informed decisions regarding fuel expenses. Its real-time streaming capabilities, coupled with customizable alerts and user-friendly interface, ensure that users are always up-to-date with the latest petrol prices. The platform's commitment to accuracy and reliability, backed by trusted data sources and continuous updates, installs confidence in users as they navigate the dynamic landscape of fuel pricing.

Realtime Petrol Price Streamer not only provides live price updates but also offers valuable historical data and market analysis, empowering users with insights into price trends and market dynamics. Whether you're a daily commuter, a fleet manager, or an industry professional, Realtime Petrol Price Streamer provides the tools and information needed to optimize fuel expenses and make strategic decisions. Join us today and experience the convenience and efficiency of real-time petrol price tracking with Realtime Petrol Price Streamer.