EventSphereManagement

TheEvent

GROUP MEMBERS

AMMAR IRFAN KOTWAL

Student1411204

MUHAMMAD SUBHAN RAEES

Student1411211

MUHAMMAD ADEEL KHAN

Student1405822

OWAIS ALAM

Student1417919

MUHAMMAD HAMMAD ALI

Student1427563

SHAHMEER KHAN

Student1392219

CURRICULUM

7062

BATCH

2211C1

FACULTY

MS. SAMREEN RAFIQ

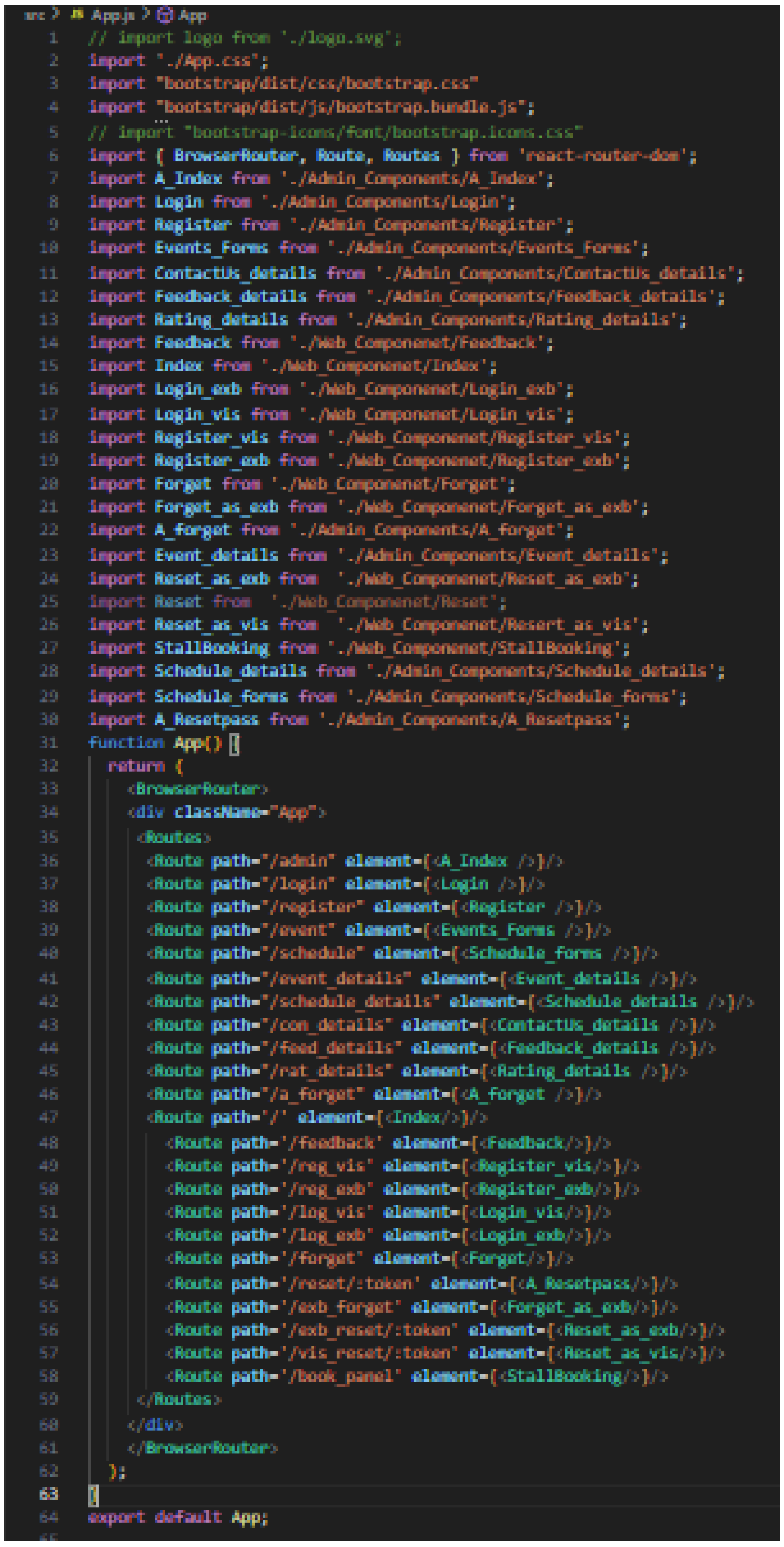
CO-ORDINATOR

MS. SANA YOUSUF

SCREEN SHOTS

Frontend

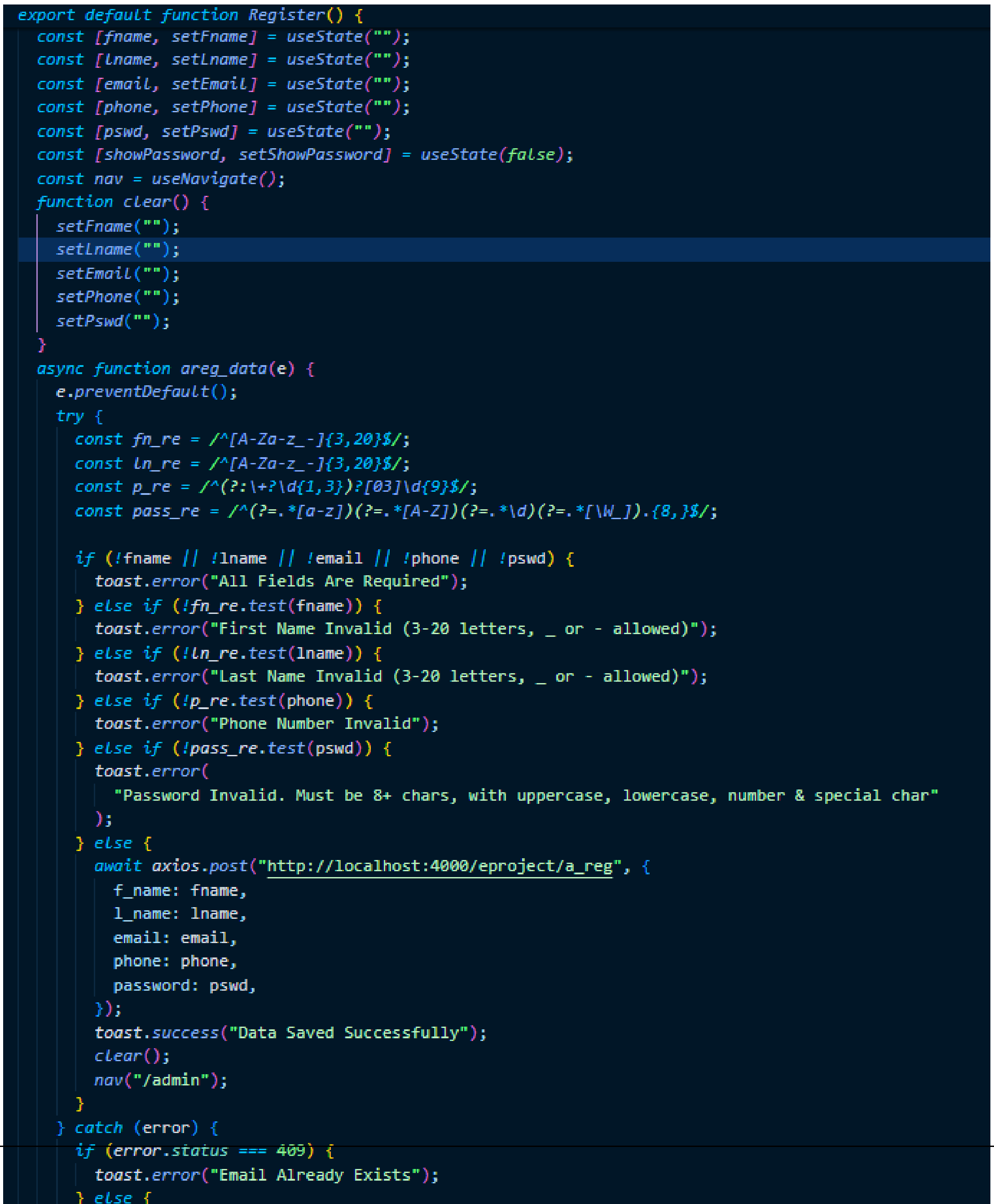
App.js File’s Code



src > App.js > App

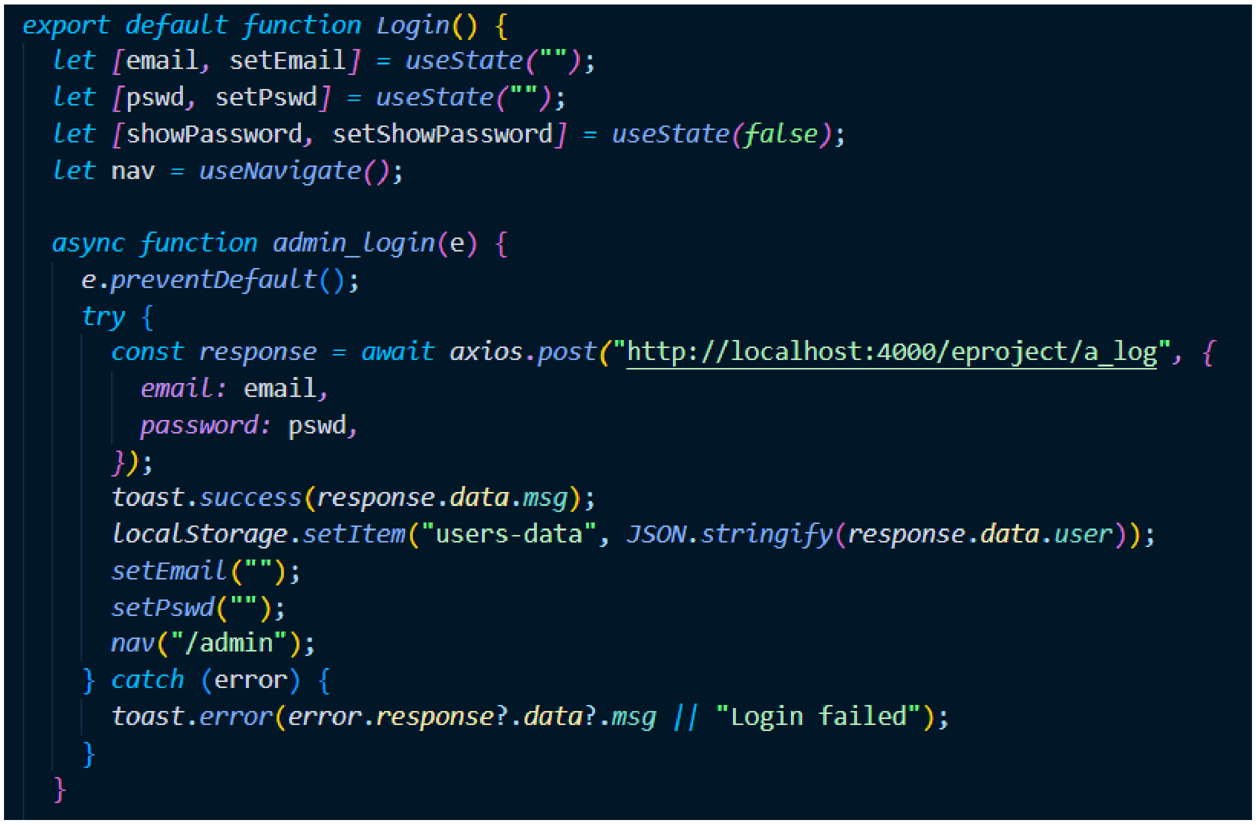
```javascript
// import logo from './logo.svg';
import './App.css';
import "bootstrap/dist/css/bootstrap.css"
import "bootstrap/dist/js/bootstrap.bundle.js";
// import "bootstrap-icons/font/bootstrap.icons.css"
import { BrowserRouter, Route, Routes } from 'react-router-dom';
import A_Index from './Admin_Components/A_Index';
import Login from './Admin_Components/Login';
import Register from './Admin_Components/Register';
import Events_Forms from './Admin_Components/Events_Forms';
import ContactUs_details from './Admin_Components/ContactUs_details';
import Feedback_details from './Admin_Components/Feedback_details';
import Rating_details from './Admin_Components/Rating_details';
import Feedback from './Web_Componenet/Feedback';
import Index from './Web_Componenet/Index';
import Login_exb from './Web_Componenet/Login_exb';
import Login_vis from './Web_Componenet/Login_vis';
import Register_vis from './Web_Componenet/Register_vis';
import Register_exb from './Web_Componenet/Register_exb';
import Forget from './Web_Componenet/Forget';
import Forget_as_exb from './Web_Componenet/Forget_as_exb';
import A_forget from './Admin_Components/A_forget';
import Event_details from './Admin_Components/Event_details';
import Reset_as_exb from  './Web_Componenet/Reset_as_exb';
import Reset from  './Web_Componenet/Reset';
import Reset_as_vis from  './Web_Componenet/Resert_as_vis';
import StallBooking from './Web_Componenet/StallBooking';
import Schedule_details from './Admin_Components/Schedule_details';
import Schedule_forms from './Admin_Components/Schedule_forms';
import A_Resetpass from './Admin_Components/A_Resetpass';
function App() {
  return (
    <BrowserRouter>
    <div className="App">
     <Routes>
      <Route path="/admin" element={<A_Index />}/>
      <Route path="/login" element={<Login />}/>
      <Route path="/register" element={<Register />}/>
      <Route path="/event" element={<Events_Forms />}/>
      <Route path="/schedule" element={<Schedule_forms />}/>
      <Route path="/event_details" element={<Event_details />}/>
      <Route path="/schedule_details" element={<Schedule_details />}/>
      <Route path="/con_details" element={<ContactUs_details />}/>
      <Route path="/feed_details" element={<Feedback_details />}/>
      <Route path="/rat_details" element={<Rating_details />}/>
      <Route path="/a_forget" element={<A_forget />}/>
      <Route path='/' element={<Index/>}/>
        <Route path='/feedback' element={<Feedback/>}/>
        <Route path='/reg_vis' element={<Register_vis/>}/>
        <Route path='/reg_exb' element={<Register_exb/>}/>
        <Route path='/log_vis' element={<Login_vis/>}/>
        <Route path='/log_exb' element={<Login_exb/>}/>
        <Route path='/forget' element={<Forget/>}/>
        <Route path='/reset/:token' element={<A_Resetpass/>}/>
        <Route path='/exb_forget' element={<Forget_as_exb/>}/>
        <Route path='/exb_reset/:token' element={<Reset_as_exb/>}/>
        <Route path='/vis_reset/:token' element={<Reset_as_vis/>}/>
        <Route path='/book_panel' element={<StallBooking/>}/>
     </Routes>
    </div>
    </BrowserRouter>
  );
}
export default App;
```

Admin Registration Code



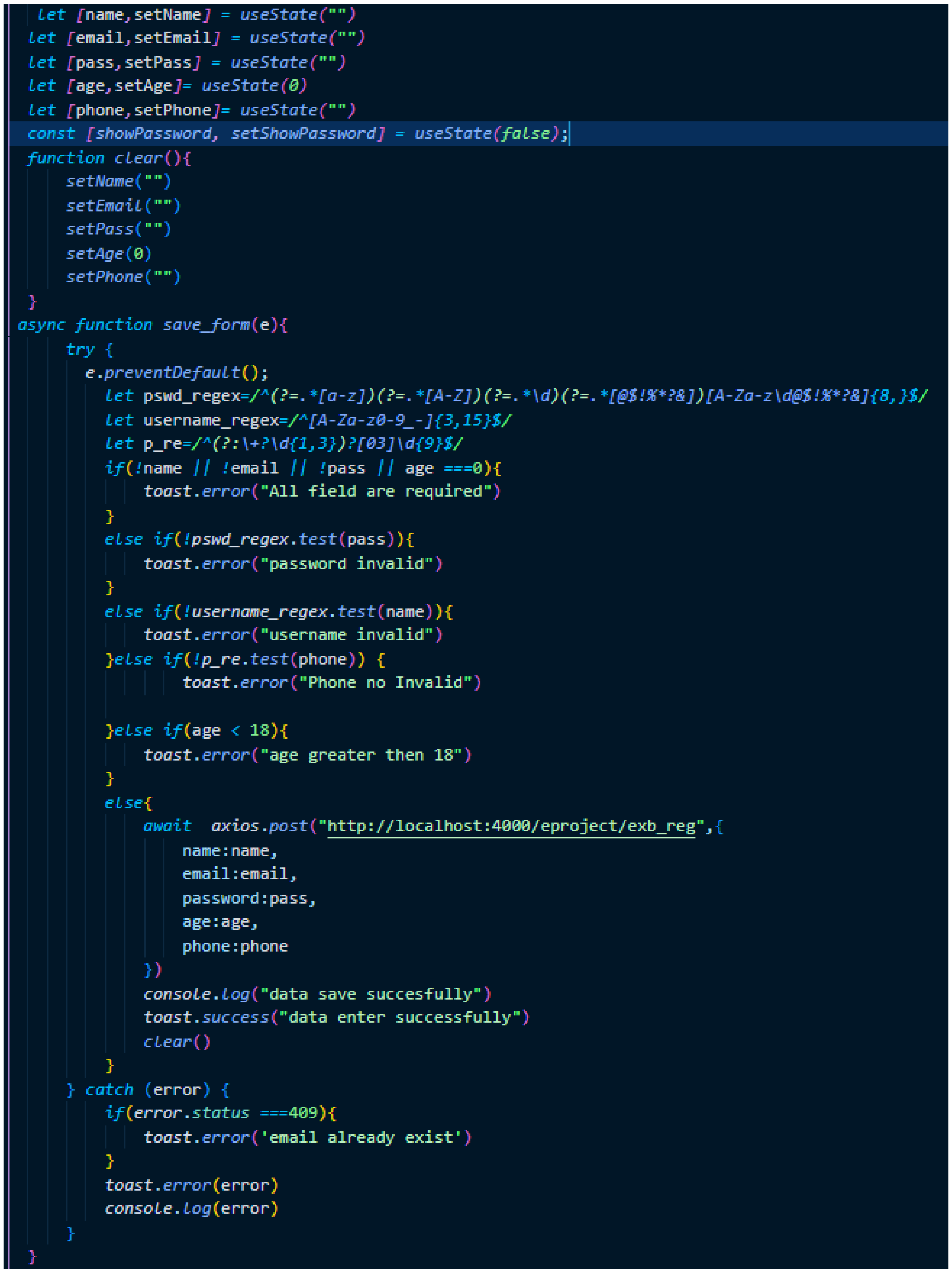
```
export default function Register() {
  const [fname, setFname] = useState("");
  const [lname, setLname] = useState("");
  const [email, setEmail] = useState("");
  const [phone, setPhone] = useState("");
  const [pswd, setPswd] = useState("");
  const [showPassword, setShowPassword] = useState(false);
  const nav = useNavigate();
  function clear() {
    setFname("");
    setLname("");
    setEmail("");
    setPhone("");
    setPswd("");
  }
  async function areg_data(e) {
    e.preventDefault();
    try {
      const fn_re = /^[A-Za-z_-]{3,20}$/;
      const ln_re = /^[A-Za-z_-]{3,20}$/;
      const p_re = /^(?:\+?\d{1,3})?[03]\d{9}$/;
      const pass_re = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[\W_]).{8,}$/;

      if (!fname || !lname || !email || !phone || !pswd) {
        toast.error("All Fields Are Required");
      } else if (!fn_re.test(fname)) {
        toast.error("First Name Invalid (3-20 letters, _ or - allowed)");
      } else if (!ln_re.test(lname)) {
        toast.error("Last Name Invalid (3-20 letters, _ or - allowed)");
      } else if (!p_re.test(phone)) {
        toast.error("Phone Number Invalid");
      } else if (!pass_re.test(pswd)) {
        toast.error(
          "Password Invalid. Must be 8+ chars, with uppercase, lowercase, number & special char"
        );
      } else {
        await axios.post("http://localhost:4000/eproject/a_reg", {
          f_name: fname,
          l_name: lname,
          email: email,
          phone: phone,
          password: pswd,
        });
        toast.success("Data Saved Successfully");
        clear();
        nav("/admin");
      }
    } catch (error) {
      if (error.status === 409) {
        toast.error("Email Already Exists");
      } else {
```

Admin Login Code



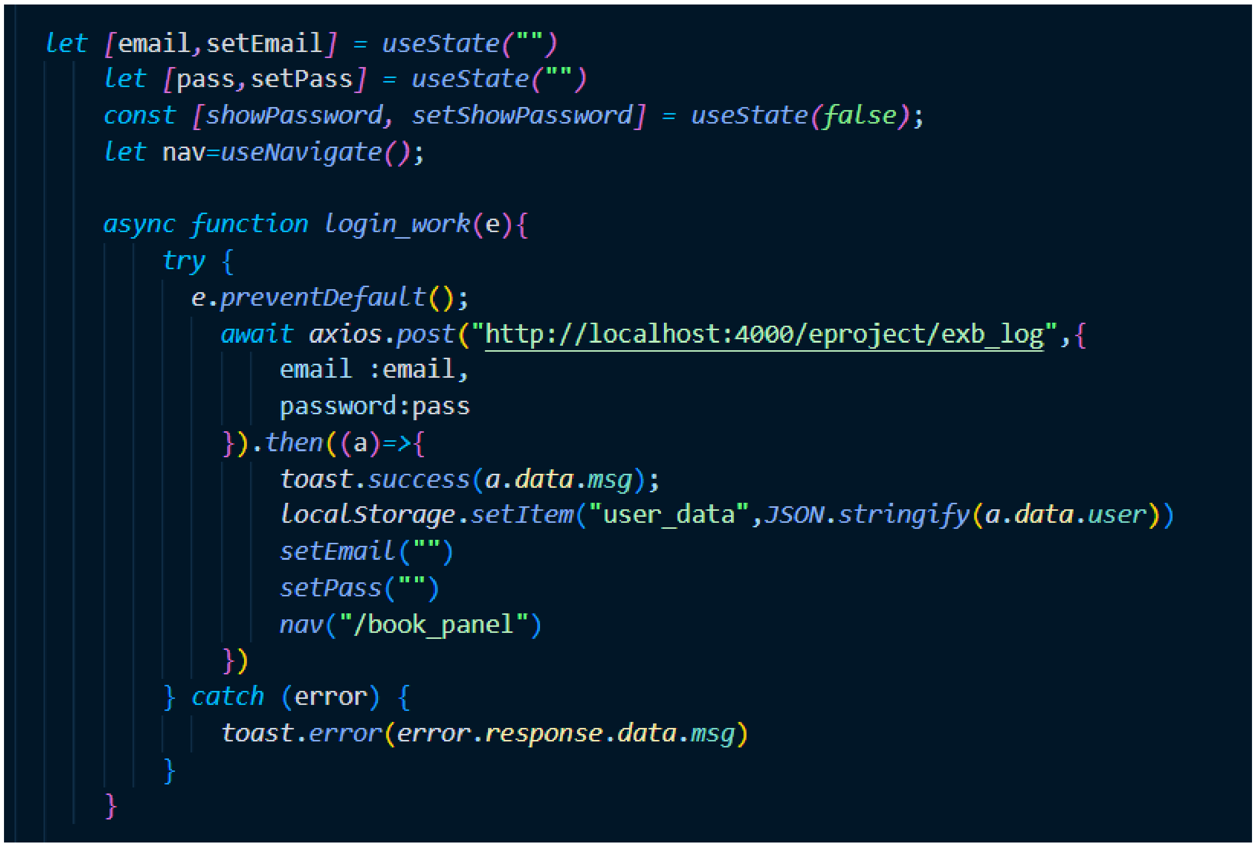
```
export default function Login() {
  let [email, setEmail] = useState("");
  let [pswd, setPswd] = useState("");
  let [showPassword, setShowPassword] = useState(false);
  let nav = useNavigate();

  async function admin_login(e) {
    e.preventDefault();
    try {
      const response = await axios.post("http://localhost:4000/eproject/a_log", {
        email: email,
        password: pswd,
      });
      toast.success(response.data.msg);
      localStorage.setItem("users-data", JSON.stringify(response.data.user));
      setEmail("");
      setPswd("");
      nav("/admin");
    } catch (error) {
      toast.error(error.response?.data?.msg || "Login failed");
    }
  }
```

Register As Exhibitor Code



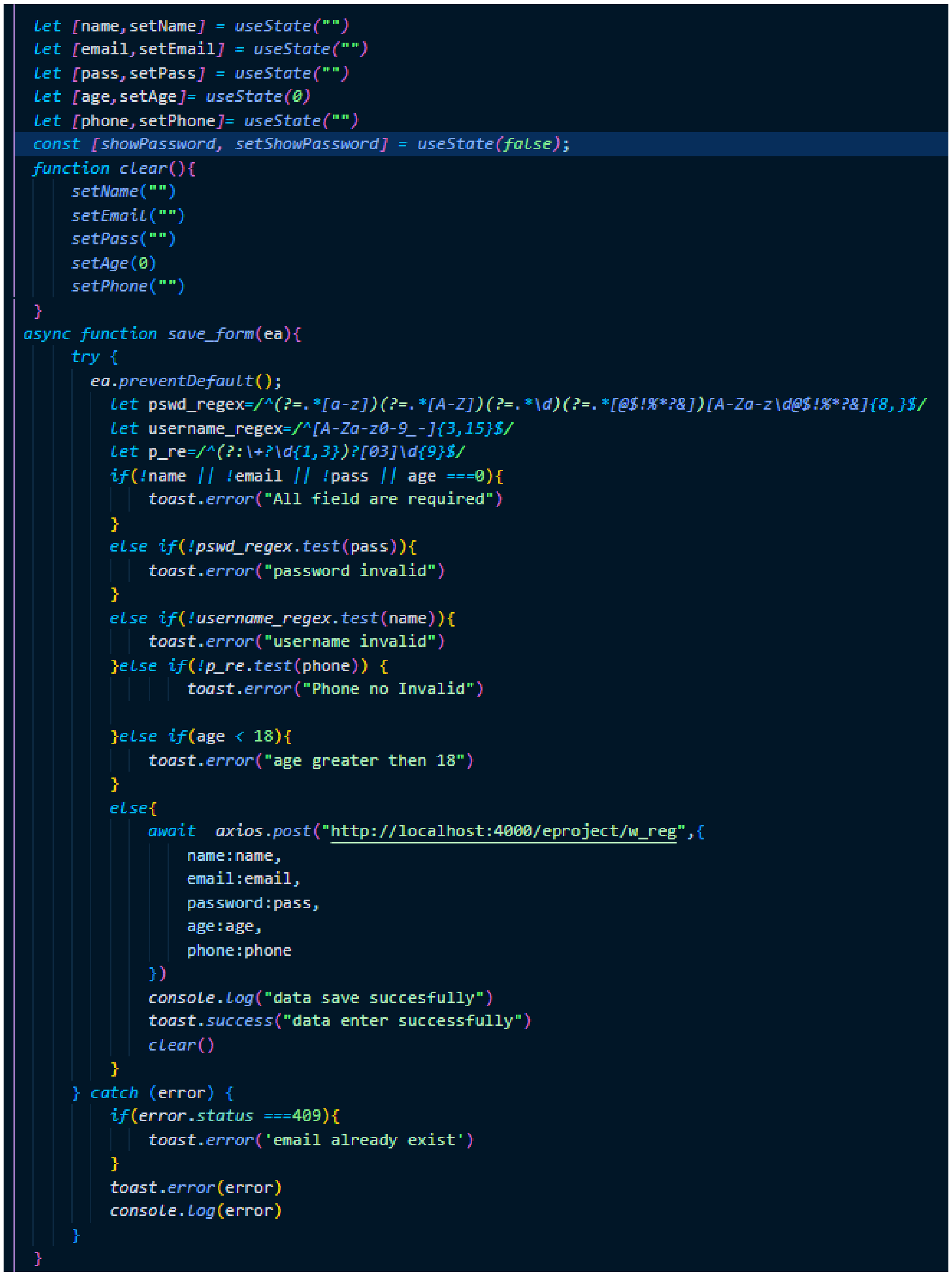
```
  let [name,setName] = useState("")
 let [email,setEmail] = useState("")
 let [pass,setPass] = useState("")
 let [age,setAge]= useState(0)
 let [phone,setPhone]= useState("")
 const [showPassword, setShowPassword] = useState(false);
 function clear(){
     setName("")
     setEmail("")
     setPass("")
     setAge(0)
     setPhone("")
 }
async function save_form(e){
     try {
       e.preventDefault();
         let pswd_regex=/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/
         let username_regex=/^[A-Za-z0-9_-]{3,15}$/
         let p_re=/^(?:\+?\d{1,3})?[03]\d{9}$/
         if(!name || !email || !pass || age ===0){
             toast.error("All field are required")
         }
         else if(!pswd_regex.test(pass)){
             toast.error("password invalid")
         }
         else if(!username_regex.test(name)){
             toast.error("username invalid")
         }else if(!p_re.test(phone)) {
                 toast.error("Phone no Invalid")

         }else if(age < 18){
             toast.error("age greater then 18")
         }
         else{
             await  axios.post("http://localhost:4000/eproject/exb_reg",{
                 name:name,
                 email:email,
                 password:pass,
                 age:age,
                 phone:phone
             })
             console.log("data save succesfully")
             toast.success("data enter successfully")
             clear()
         }
     } catch (error) {
         if(error.status ===409){
             toast.error('email already exist')
         }
         toast.error(error)
         console.log(error)
     }
 }
```

Login As Exhibitor Code



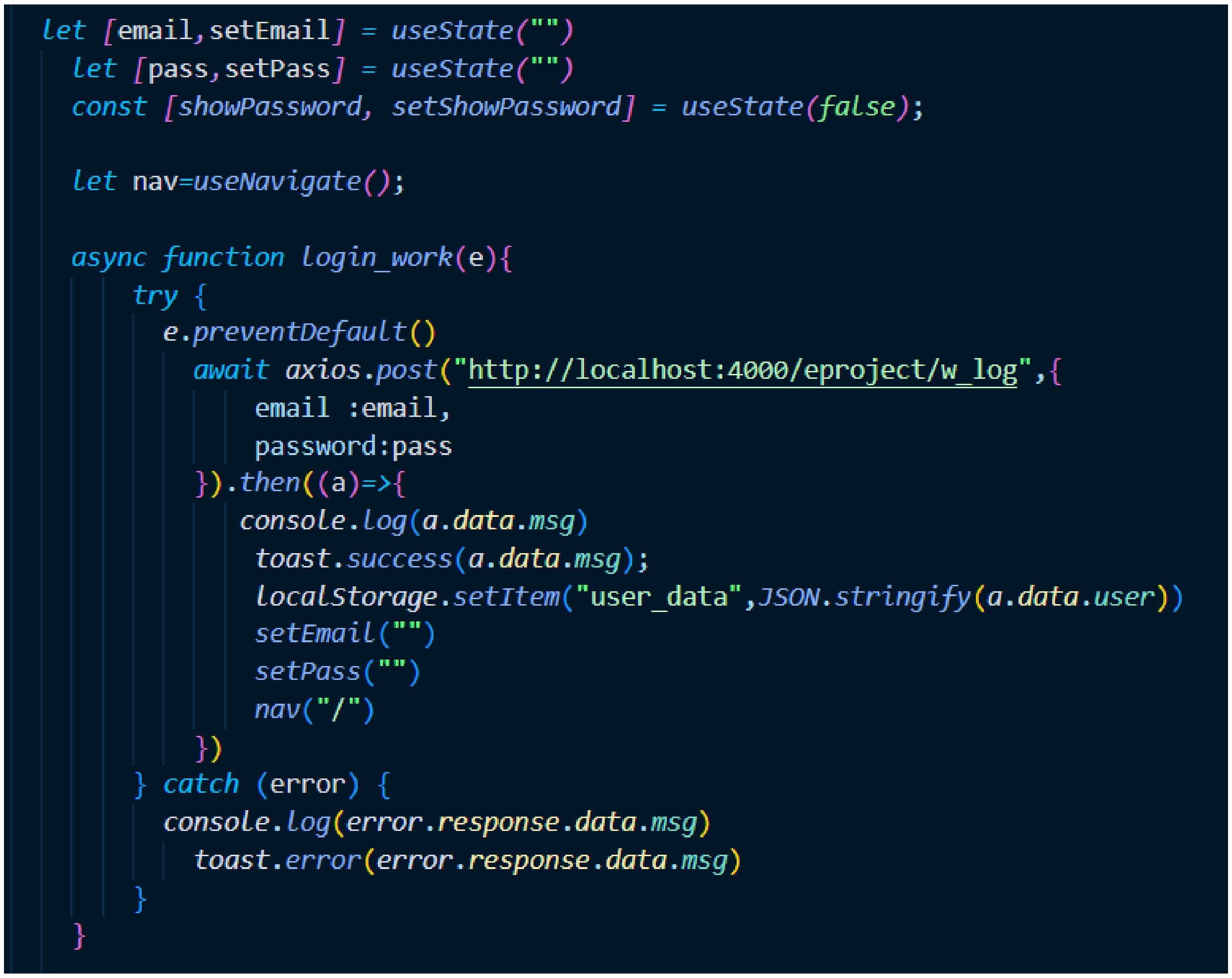
```
let [email,setEmail] = useState("")
    let [pass,setPass] = useState("")
    const [showPassword, setShowPassword] = useState(false);
    let nav=useNavigate();

    async function login_work(e){
        try {
          e.preventDefault();
            await axios.post("http://localhost:4000/eproject/exb_log",{
                email :email,
                password:pass
            }).then((a)=>{
                toast.success(a.data.msg);
                localStorage.setItem("user_data",JSON.stringify(a.data.user))
                setEmail("")
                setPass("")
                nav("/book_panel")
            })
        } catch (error) {
            toast.error(error.response.data.msg)
        }
    }
```

SignUp As Visitor Code



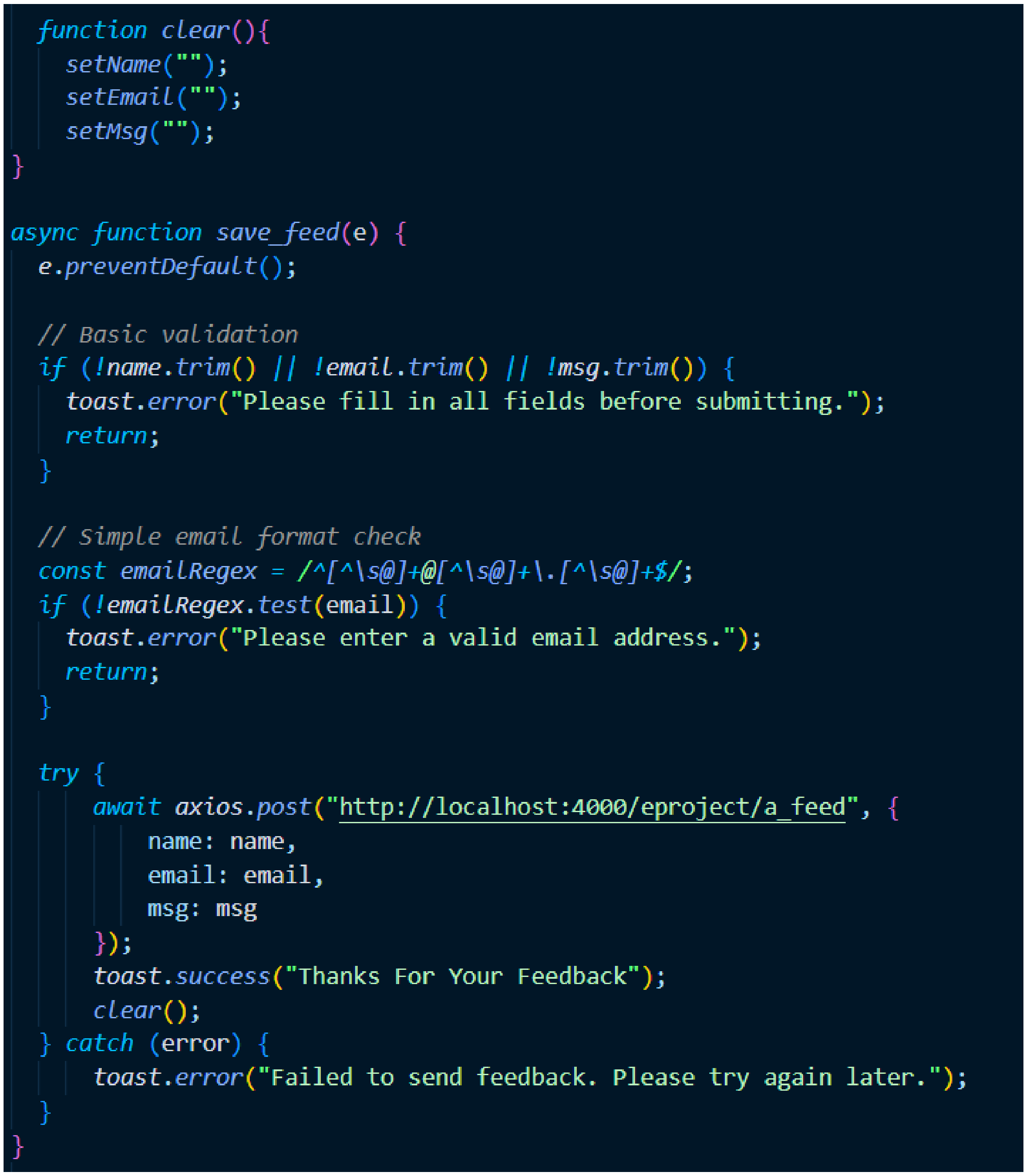
```
let [name,setName] = useState("")
let [email,setEmail] = useState("")
let [pass,setPass] = useState("")
let [age,setAge]= useState(0)
let [phone,setPhone]= useState("")
const [showPassword, setShowPassword] = useState(false);
function clear(){
    setName("")
    setEmail("")
    setPass("")
    setAge(0)
    setPhone("")
}
async function save_form(ea){
    try {
      ea.preventDefault();
        let pswd_regex=/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/
        let username_regex=/^[A-Za-z0-9_-]{3,15}$/
        let p_re=/^(?:\+?\d{1,3})?[03]\d{9}$/
        if(!name || !email || !pass || age ===0){
            toast.error("All field are required")
        }
        else if(!pswd_regex.test(pass)){
            toast.error("password invalid")
        }
        else if(!username_regex.test(name)){
            toast.error("username invalid")
        }else if(!p_re.test(phone)) {
                toast.error("Phone no Invalid")

        }else if(age < 18){
            toast.error("age greater then 18")
        }
        else{
            await  axios.post("http://localhost:4000/eproject/w_reg",{
                name:name,
                email:email,
                password:pass,
                age:age,
                phone:phone
            })
            console.log("data save succesfully")
            toast.success("data enter successfully")
            clear()
        }
    } catch (error) {
        if(error.status ===409){
            toast.error('email already exist')
        }
        toast.error(error)
        console.log(error)
    }
}
```

Login As Visitor Code



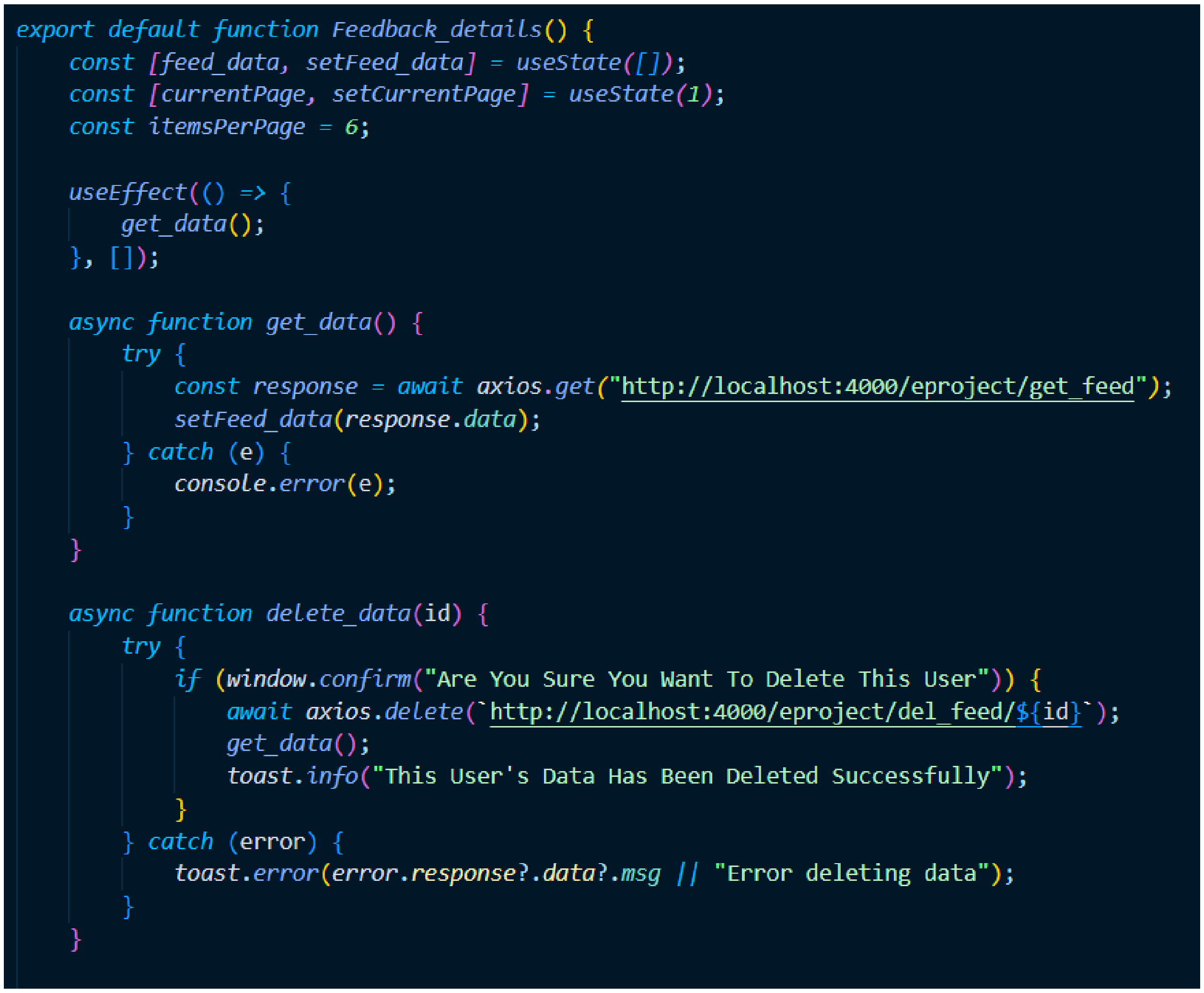
```
let [email,setEmail] = useState("")
  let [pass,setPass] = useState("")
  const [showPassword, setShowPassword] = useState(false);

  let nav=useNavigate();

  async function login_work(e){
      try {
        e.preventDefault()
          await axios.post("http://localhost:4000/eproject/w_log",{
              email :email,
              password:pass
          }).then((a)=>{
             console.log(a.data.msg)
              toast.success(a.data.msg);
              localStorage.setItem("user_data",JSON.stringify(a.data.user))
              setEmail("")
              setPass("")
              nav("/")
          })
      } catch (error) {
        console.log(error.response.data.msg)
          toast.error(error.response.data.msg)
      }
  }
```

Feedback Form’s Code



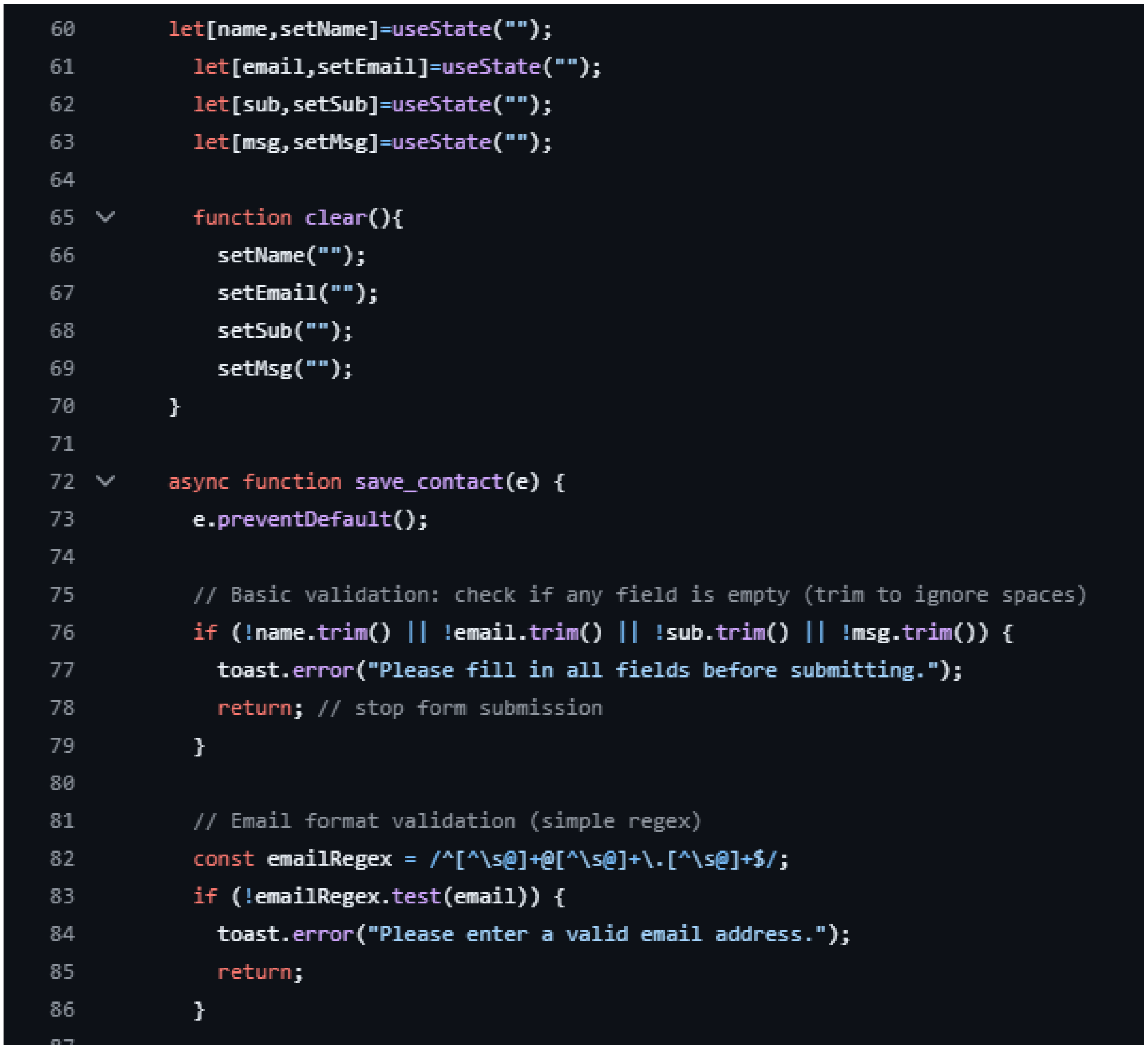
```javascript
  function clear(){
    setName("");
    setEmail("");
    setMsg("");
}

async function save_feed(e) {
  e.preventDefault();

  // Basic validation
  if (!name.trim() || !email.trim() || !msg.trim()) {
    toast.error("Please fill in all fields before submitting.");
    return;
  }

  // Simple email format check
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  if (!emailRegex.test(email)) {
    toast.error("Please enter a valid email address.");
    return;
  }

  try {
      await axios.post("http://localhost:4000/eproject/a_feed", {
        name: name,
        email: email,
        msg: msg
      });
      toast.success("Thanks For Your Feedback");
      clear();
  } catch (error) {
      toast.error("Failed to send feedback. Please try again later.");
  }
}
```

Feedback Details Code



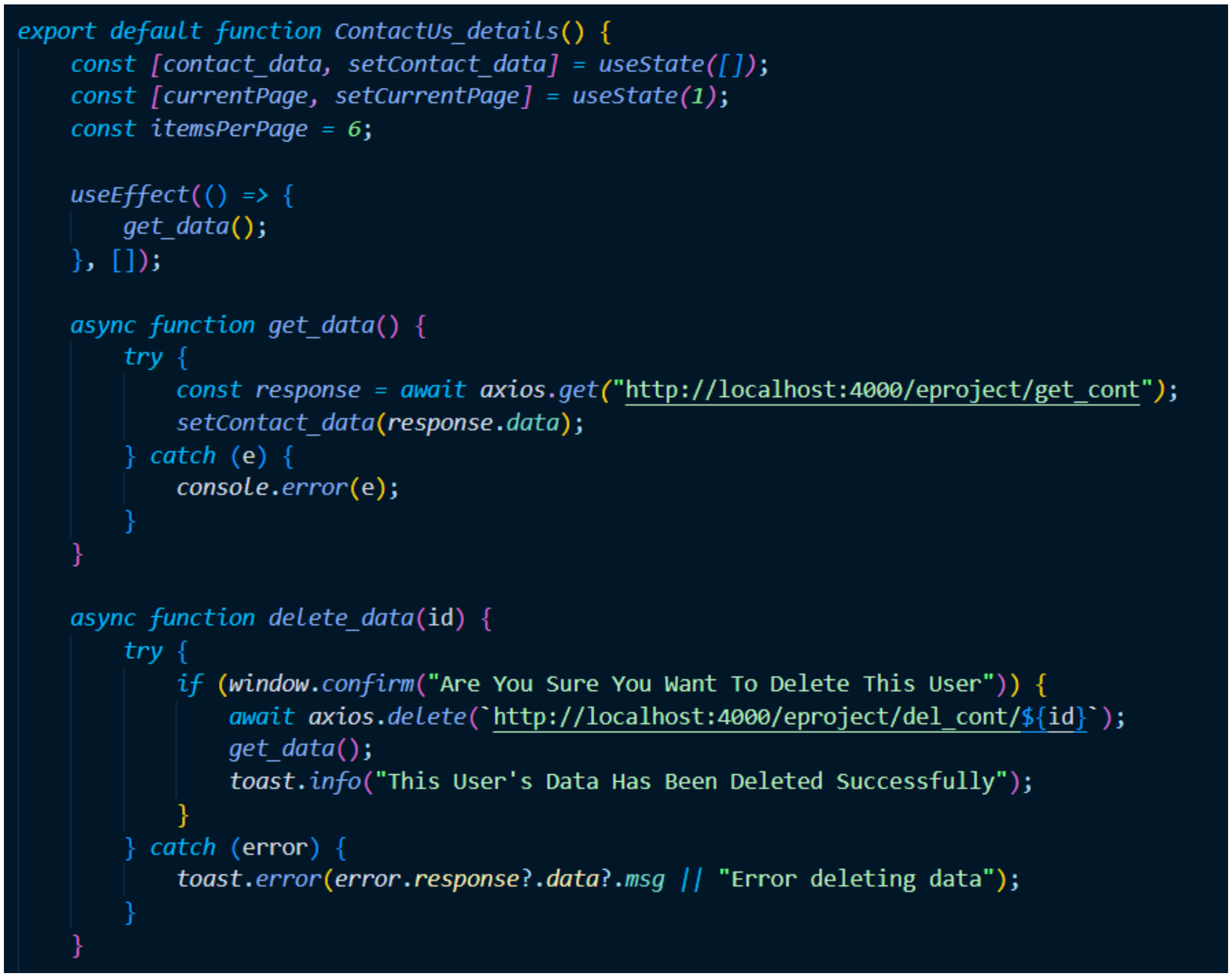
```
export default function Feedback_details() {
    const [feed_data, setFeed_data] = useState([]);
    const [currentPage, setCurrentPage] = useState(1);
    const itemsPerPage = 6;

    useEffect(() => {
        get_data();
    }, []);

    async function get_data() {
        try {
            const response = await axios.get("http://localhost:4000/eproject/get_feed");
            setFeed_data(response.data);
        } catch (e) {
            console.error(e);
        }
    }

    async function delete_data(id) {
        try {
            if (window.confirm("Are You Sure You Want To Delete This User")) {
                await axios.delete(`http://localhost:4000/eproject/del_feed/${id}`);
                get_data();
                toast.info("This User's Data Has Been Deleted Successfully");
            }
        } catch (error) {
            toast.error(error.response?.data?.msg || "Error deleting data");
        }
    }
```

Contact Us Form’s Code



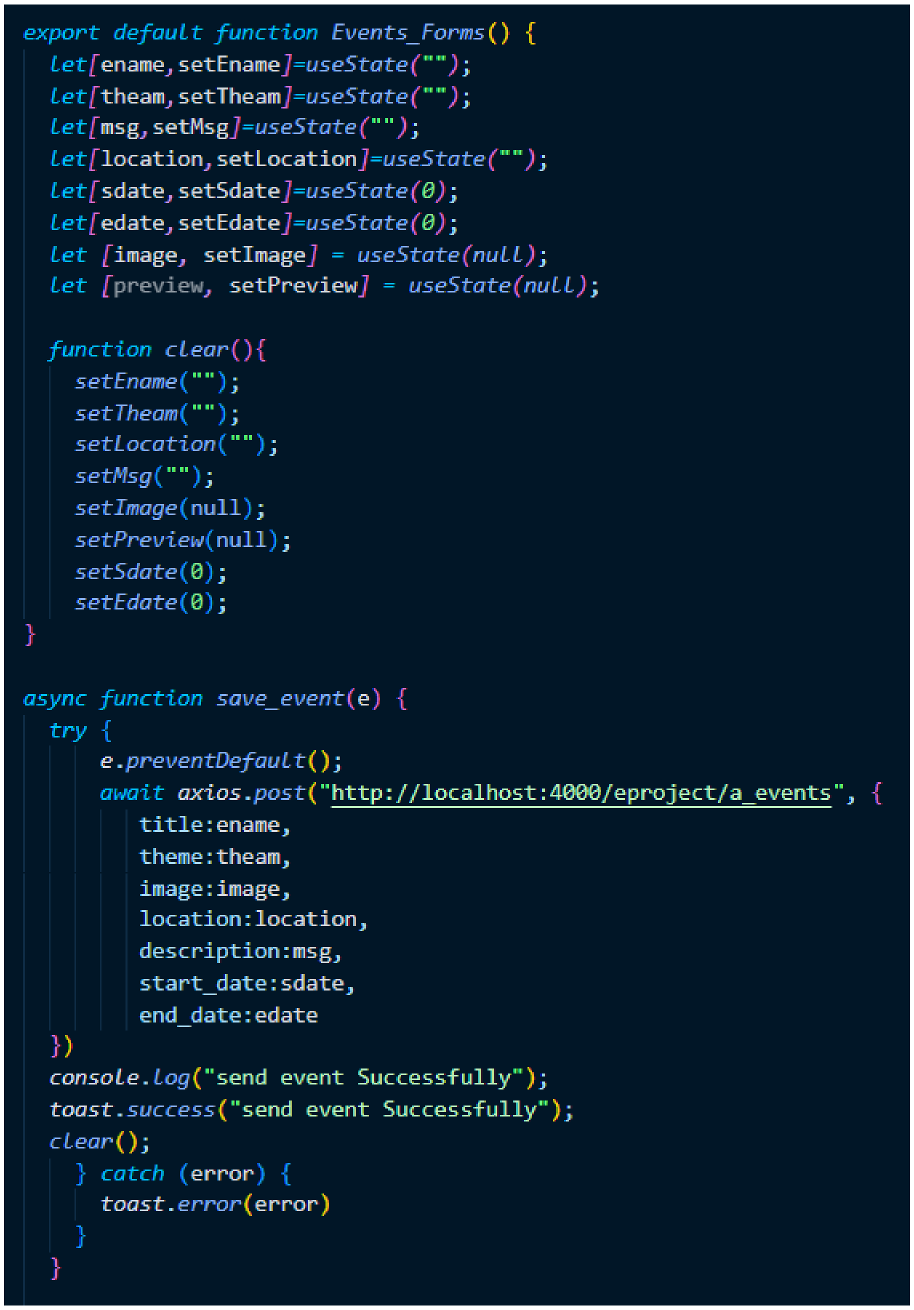
```
    let[name,setName]=useState("");
      let[email,setEmail]=useState("");
      let[sub,setSub]=useState("");
      let[msg,setMsg]=useState("");

      function clear(){
        setName("");
        setEmail("");
        setSub("");
        setMsg("");
    }

    async function save_contact(e) {
      e.preventDefault();

      // Basic validation: check if any field is empty (trim to ignore spaces)
      if (!name.trim() || !email.trim() || !sub.trim() || !msg.trim()) {
        toast.error("Please fill in all fields before submitting.");
        return; // stop form submission
      }

      // Email format validation (simple regex)
      const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
      if (!emailRegex.test(email)) {
        toast.error("Please enter a valid email address.");
        return;
      }
```

Contact Details Code



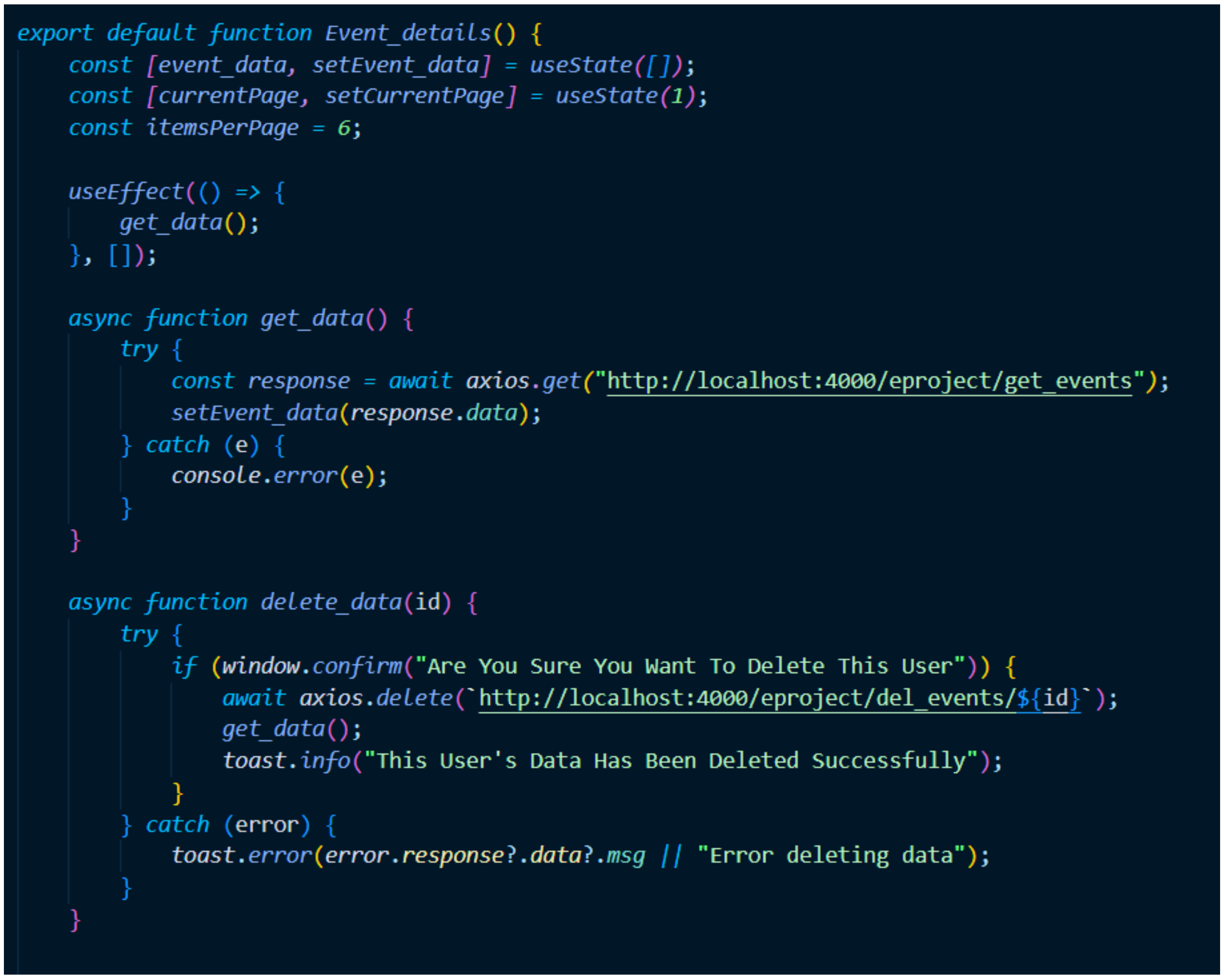
```
export default function ContactUs_details() {
    const [contact_data, setContact_data] = useState([]);
    const [currentPage, setCurrentPage] = useState(1);
    const itemsPerPage = 6;

    useEffect(() => {
        get_data();
    }, []);

    async function get_data() {
        try {
            const response = await axios.get("http://localhost:4000/eproject/get_cont");
            setContact_data(response.data);
        } catch (e) {
            console.error(e);
        }
    }

    async function delete_data(id) {
        try {
            if (window.confirm("Are You Sure You Want To Delete This User")) {
                await axios.delete(`http://localhost:4000/eproject/del_cont/${id}`);
                get_data();
                toast.info("This User's Data Has Been Deleted Successfully");
            }
        } catch (error) {
            toast.error(error.response?.data?.msg || "Error deleting data");
        }
    }
```

Event's Form's Code



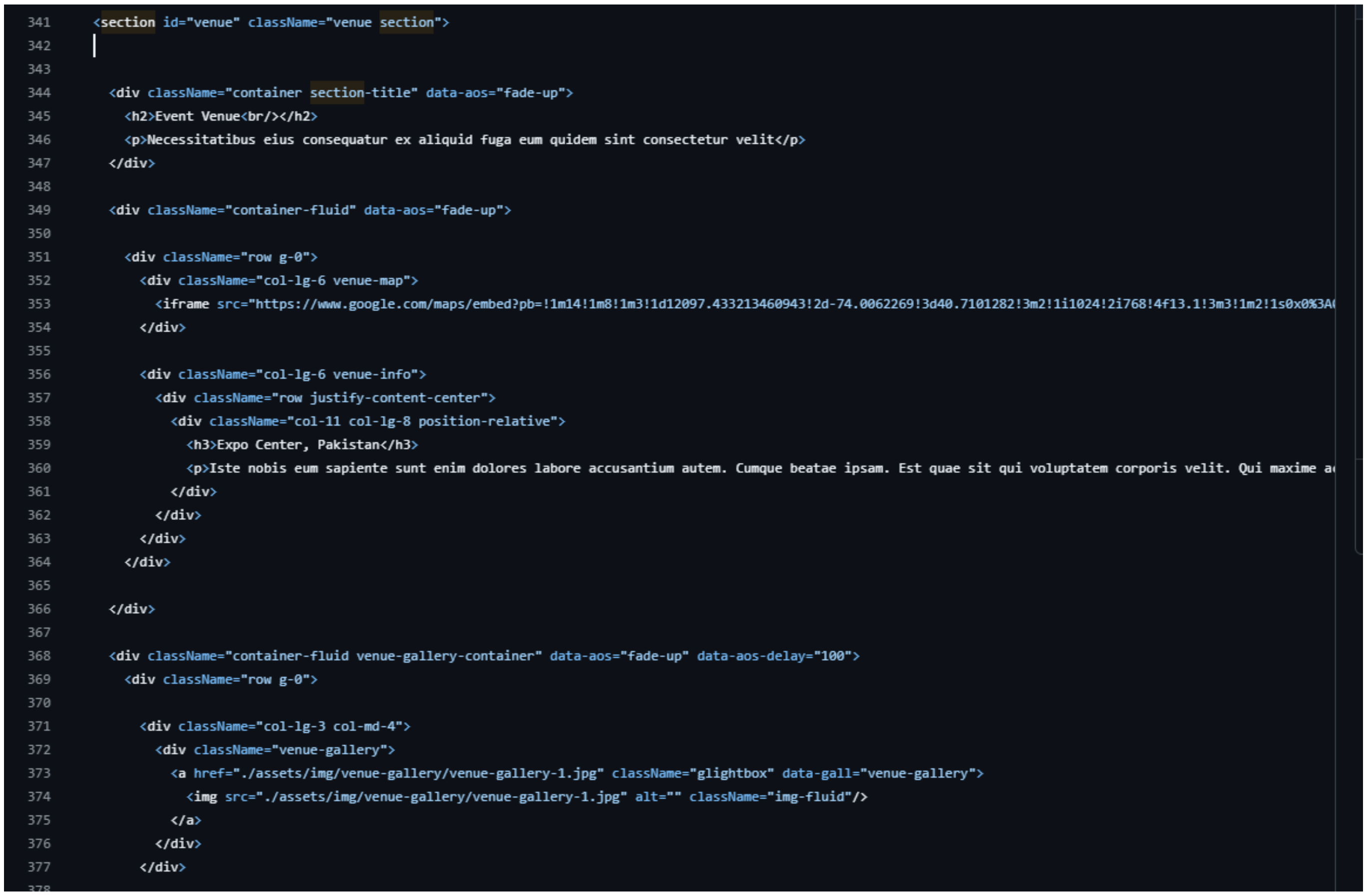
```
export default function Events_Forms() {
  let[ename,setEname]=useState("");
  let[theam,setTheam]=useState("");
  let[msg,setMsg]=useState("");
  let[location,setLocation]=useState("");
  let[sdate,setSdate]=useState(0);
  let[edate,setEdate]=useState(0);
  let [image, setImage] = useState(null);
  let [preview, setPreview] = useState(null);

  function clear(){
    setEname("");
    setTheam("");
    setLocation("");
    setMsg("");
    setImage(null);
    setPreview(null);
    setSdate(0);
    setEdate(0);
}

async function save_event(e) {
  try {
      e.preventDefault();
      await axios.post("http://localhost:4000/eproject/a_events", {
         title:ename,
         theme:theam,
         image:image,
         location:location,
         description:msg,
         start_date:sdate,
         end_date:edate
  })
  console.log("send event Successfully");
  toast.success("send event Successfully");
  clear();
    } catch (error) {
      toast.error(error)
    }
  }
```

Event’s Details Code



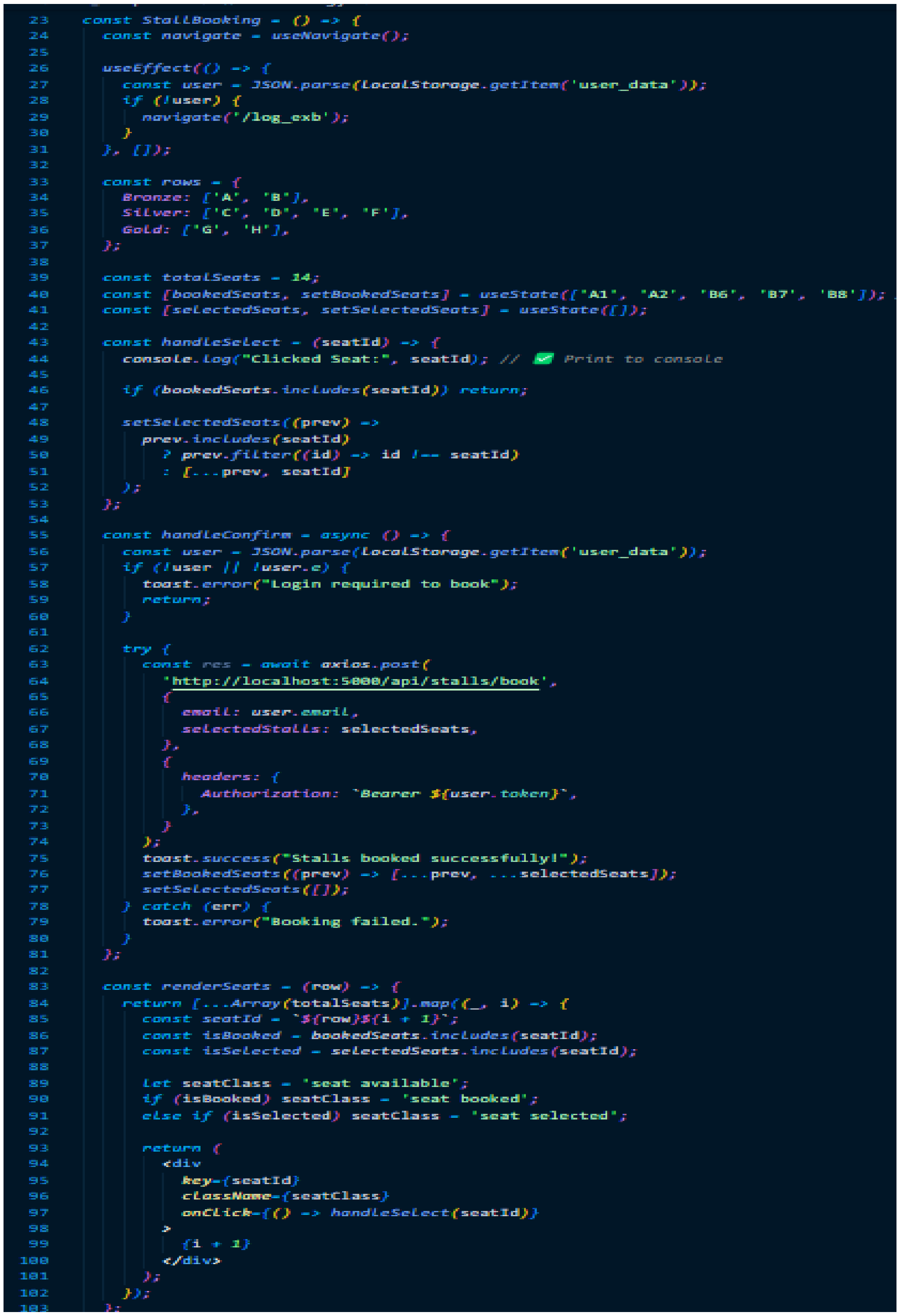
```
export default function Event_details() {
    const [event_data, setEvent_data] = useState([]);
    const [currentPage, setCurrentPage] = useState(1);
    const itemsPerPage = 6;

    useEffect(() => {
        get_data();
    }, []);

    async function get_data() {
        try {
            const response = await axios.get("http://localhost:4000/eproject/get_events");
            setEvent_data(response.data);
        } catch (e) {
            console.error(e);
        }
    }

    async function delete_data(id) {
        try {
            if (window.confirm("Are You Sure You Want To Delete This User")) {
                await axios.delete(`http://localhost:4000/eproject/del_events/${id}`);
                get_data();
                toast.info("This User's Data Has Been Deleted Successfully");
            }
        } catch (error) {
            toast.error(error.response?.data?.msg || "Error deleting data");
        }
    }
```

Event’s Venue Code



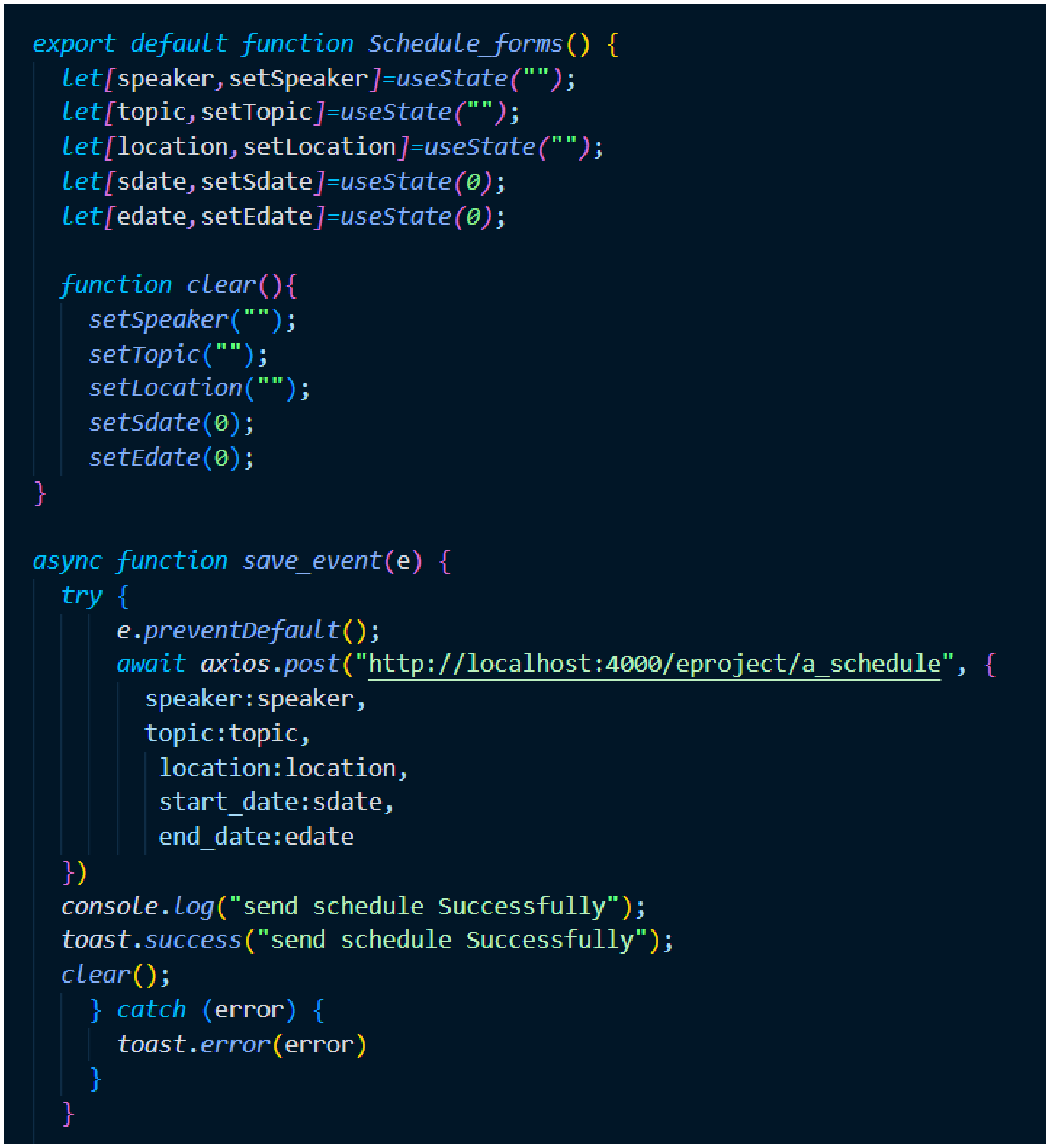
```
<section id="venue" className="venue section">


  <div className="container section-title" data-aos="fade-up">
    <h2>Event Venue<br/></h2>
    <p>Necessitatibus eius consequatur ex aliquid fuga eum quidem sint consectetur velit</p>
  </div>

  <div className="container-fluid" data-aos="fade-up">

    <div className="row g-0">
      <div className="col-lg-6 venue-map">
        <iframe src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d12097.433213460943!2d-74.0062269!3d40.7101282!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A
      </div>

      <div className="col-lg-6 venue-info">
        <div className="row justify-content-center">
          <div className="col-11 col-lg-8 position-relative">
            <h3>Expo Center, Pakistan</h3>
            <p>Iste nobis eum sapiente sunt enim dolores labore accusantium autem. Cumque beatae ipsam. Est quae sit qui voluptatem corporis velit. Qui maxime a
          </div>
        </div>
      </div>
    </div>

  </div>

  <div className="container-fluid venue-gallery-container" data-aos="fade-up" data-aos-delay="100">
    <div className="row g-0">

      <div className="col-lg-3 col-md-4">
        <div className="venue-gallery">
          <a href="./assets/img/venue-gallery/venue-gallery-1.jpg" className="glightbox" data-gall="venue-gallery">
            <img src="./assets/img/venue-gallery/venue-gallery-1.jpg" alt="" className="img-fluid"/>
          </a>
        </div>
      </div>
```

Stall’s Booking Code



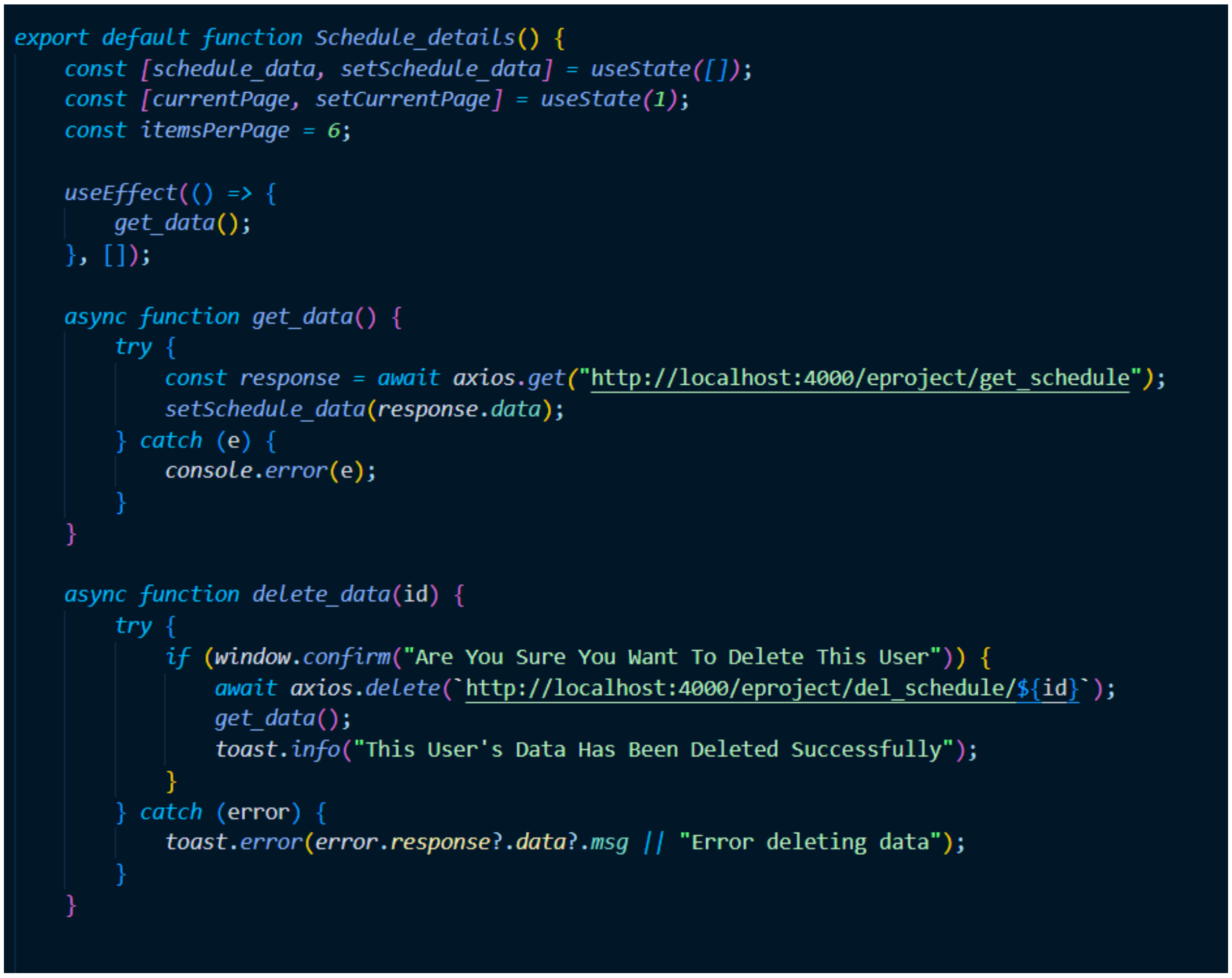
```javascript
const StallBooking = () => {
  const navigate = useNavigate();

  useEffect(() => {
    const user = JSON.parse(localStorage.getItem('user_data'));
    if (!user) {
      navigate('/log_exb');
    }
  }, []);

  const rows = {
    Bronze: ['A', 'B'],
    Silver: ['C', 'D', 'E', 'F'],
    Gold: ['G', 'H'],
  };

  const totalSeats = 14;
  const [bookedSeats, setBookedSeats] = useState(['A1', 'A2', 'B6', 'B7', 'B8']);
  const [selectedSeats, setSelectedSeats] = useState([]);

  const handleSelect = (seatId) => {
    console.log("Clicked Seat:", seatId); // ✅ Print to console

    if (bookedSeats.includes(seatId)) return;

    setSelectedSeats((prev) =>
      prev.includes(seatId)
        ? prev.filter((id) => id !== seatId)
        : [...prev, seatId]
    );
  };

  const handleConfirm = async () => {
    const user = JSON.parse(localStorage.getItem('user_data'));
    if (!user || !user.e) {
      toast.error("Login required to book");
      return;
    }

    try {
      const res = await axios.post(
        'http://localhost:5000/api/stalls/book',
        {
          email: user.email,
          selectedStalls: selectedSeats,
        },
        {
          headers: {
            Authorization: `Bearer ${user.token}`,
          },
        }
      );
      toast.success("Stalls booked successfully!");
      setBookedSeats((prev) => [...prev, ...selectedSeats]);
      setSelectedSeats([]);
    } catch (err) {
      toast.error("Booking failed.");
    }
  };

  const renderSeats = (row) => {
    return [...Array(totalSeats)].map((_, i) => {
      const seatId = `${row}${i + 1}`;
      const isBooked = bookedSeats.includes(seatId);
      const isSelected = selectedSeats.includes(seatId);

      let seatClass = 'seat available';
      if (isBooked) seatClass = 'seat booked';
      else if (isSelected) seatClass = 'seat selected';

      return (
        <div
          key={seatId}
          className={seatClass}
          onClick={() => handleSelect(seatId)}
        >
          {i + 1}
        </div>
      );
    });
  };
```

Schedule Management Code



```
export default function Schedule_forms() {
  let[speaker,setSpeaker]=useState("");
  let[topic,setTopic]=useState("");
  let[location,setLocation]=useState("");
  let[sdate,setSdate]=useState(0);
  let[edate,setEdate]=useState(0);

  function clear(){
    setSpeaker("");
    setTopic("");
    setLocation("");
    setSdate(0);
    setEdate(0);
}

async function save_event(e) {
  try {
      e.preventDefault();
      await axios.post("http://localhost:4000/eproject/a_schedule", {
        speaker:speaker,
        topic:topic,
          location:location,
          start_date:sdate,
          end_date:edate
  })
  console.log("send schedule Successfully");
  toast.success("send schedule Successfully");
  clear();
    } catch (error) {
      toast.error(error)
    }
  }
```

Schedule's Details Code

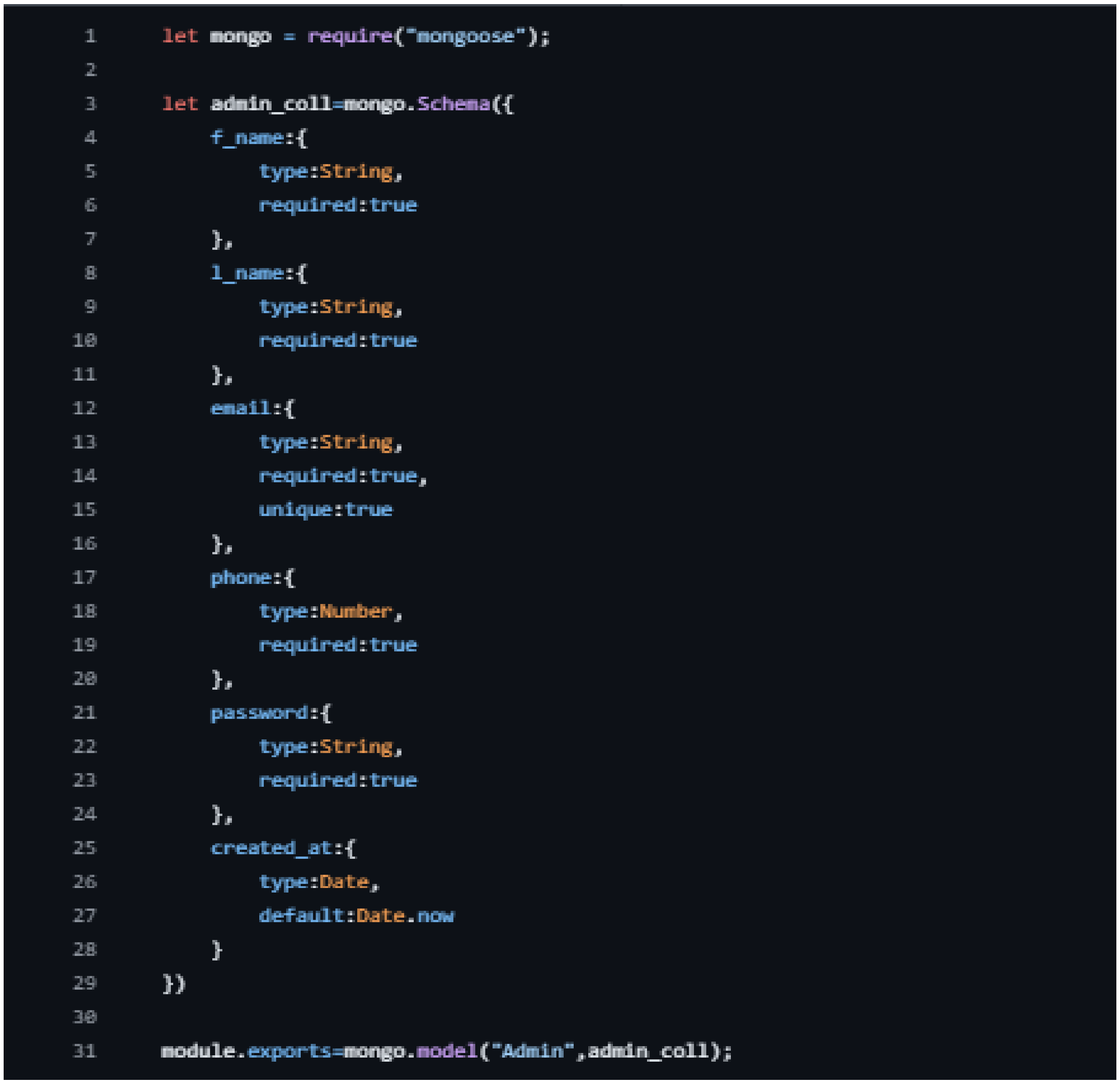


```
export default function Schedule_details() {
    const [schedule_data, setSchedule_data] = useState([]);
    const [currentPage, setCurrentPage] = useState(1);
    const itemsPerPage = 6;

    useEffect(() => {
        get_data();
    }, []);

    async function get_data() {
        try {
            const response = await axios.get("http://localhost:4000/eproject/get_schedule");
            setSchedule_data(response.data);
        } catch (e) {
            console.error(e);
        }
    }

    async function delete_data(id) {
        try {
            if (window.confirm("Are You Sure You Want To Delete This User")) {
                await axios.delete(`http://localhost:4000/eproject/del_schedule/${id}`);
                get_data();
                toast.info("This User's Data Has Been Deleted Successfully");
            }
        } catch (error) {
            toast.error(error.response?.data?.msg || "Error deleting data");
        }
    }
```

Backend

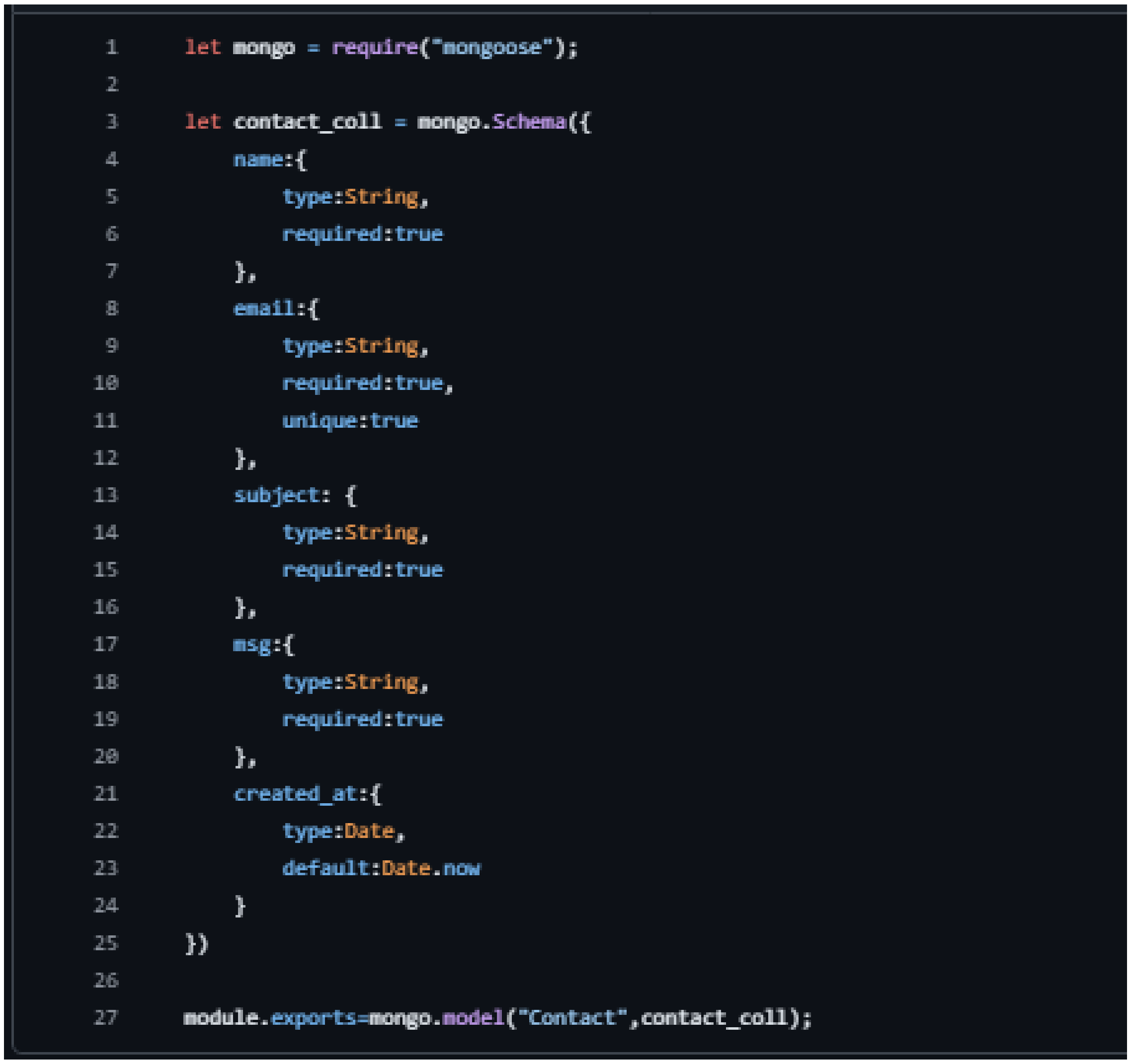
Collections

Admin Collection Code



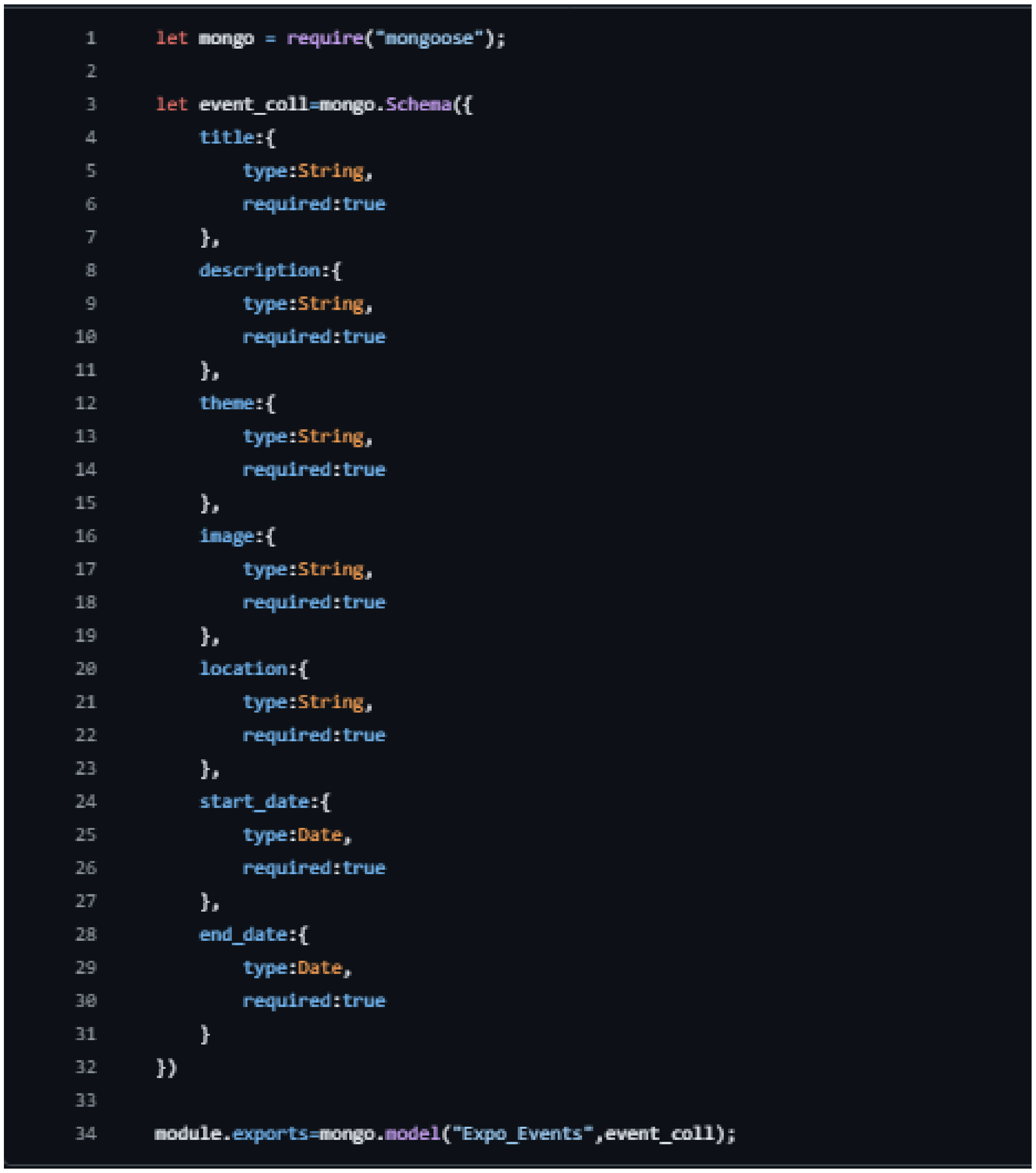
```
let mongo = require("mongoose");

let admin_coll=mongo.Schema({
    f_name:{
        type:String,
        required:true
    },
    l_name:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true,
        unique:true
    },
    phone:{
        type:Number,
        required:true
    },
    password:{
        type:String,
        required:true
    },
    created_at:{
        type:Date,
        default:Date.now
    }
})

module.exports=mongo.model("Admin",admin_coll);
```

Contact Collection Code



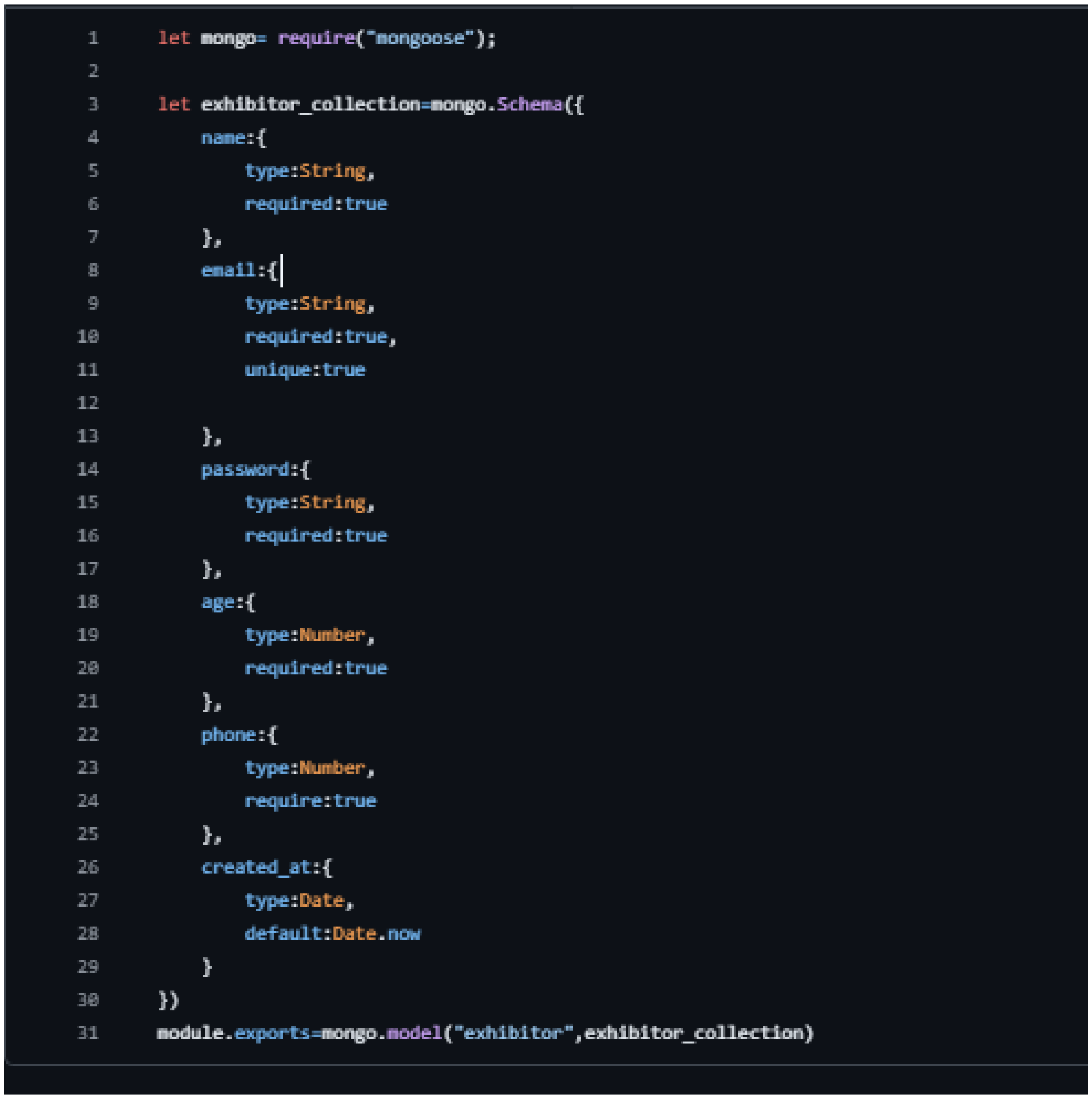
```
let mongo = require("mongoose");

let contact_coll = mongo.Schema({
    name:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true,
        unique:true
    },
    subject: {
        type:String,
        required:true
    },
    msg:{
        type:String,
        required:true
    },
    created_at:{
        type:Date,
        default:Date.now
    }
})

module.exports=mongo.model("Contact",contact_coll);
```

Event Collection Code



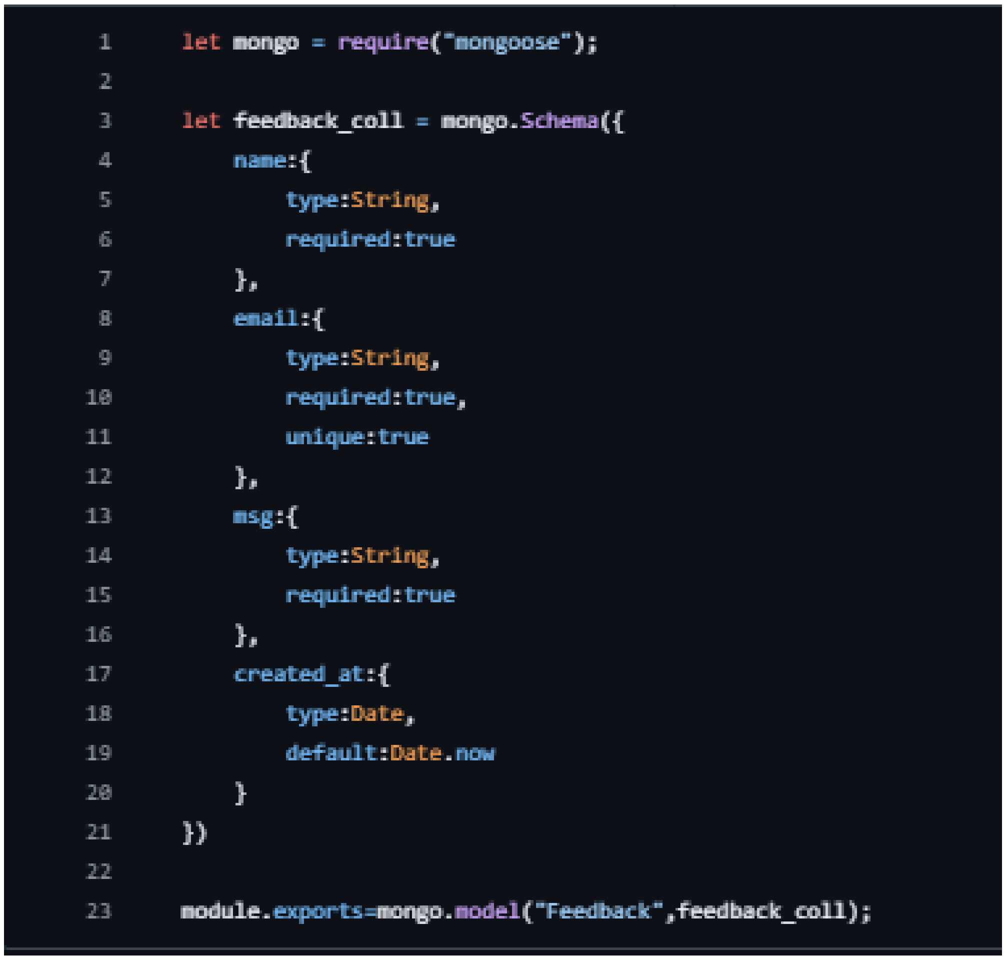
```
let mongo = require("mongoose");

let event_coll=mongo.Schema({
    title:{
        type:String,
        required:true
    },
    description:{
        type:String,
        required:true
    },
    theme:{
        type:String,
        required:true
    },
    image:{
        type:String,
        required:true
    },
    location:{
        type:String,
        required:true
    },
    start_date:{
        type:Date,
        required:true
    },
    end_date:{
        type:Date,
        required:true
    }
})

module.exports=mongo.model("Expo_Events",event_coll);
```

Exhibitor Collection Code



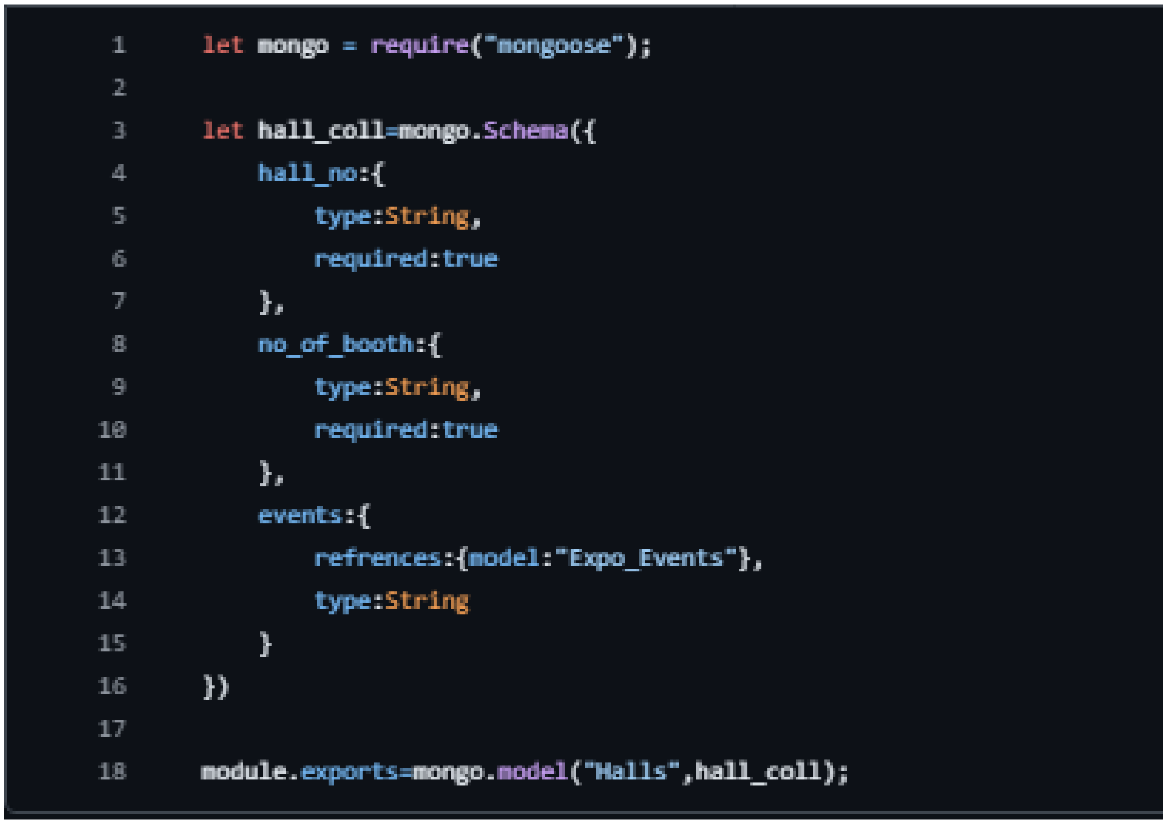
```
let mongo= require("mongoose");

let exhibitor_collection=mongo.Schema({
    name:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true,
        unique:true

    },
    password:{
        type:String,
        required:true
    },
    age:{
        type:Number,
        required:true
    },
    phone:{
        type:Number,
        require:true
    },
    created_at:{
        type:Date,
        default:Date.now
    }
})
module.exports=mongo.model("exhibitor",exhibitor_collection)
```

Feedback Collection Code



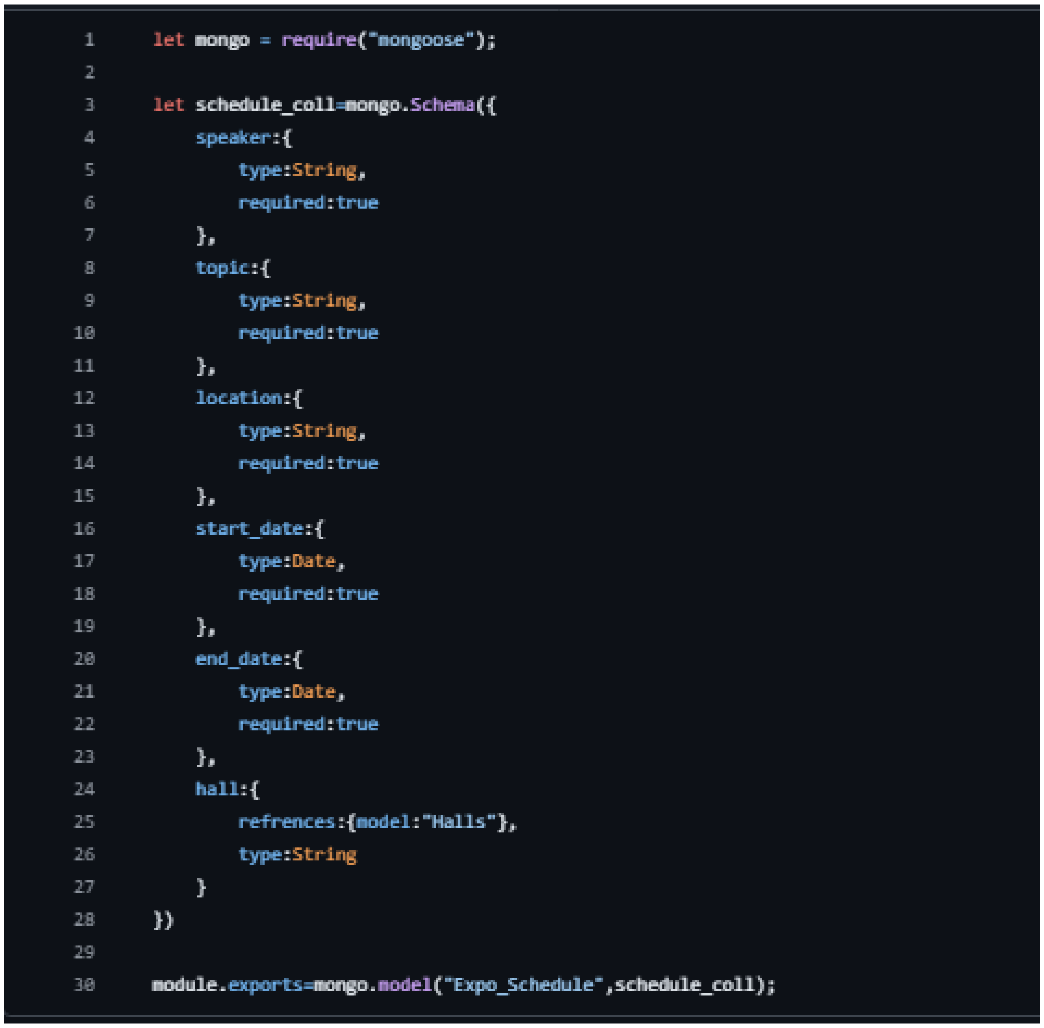
```
let mongo = require("mongoose");

let feedback_coll = mongo.Schema({
    name:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true,
        unique:true
    },
    msg:{
        type:String,
        required:true
    },
    created_at:{
        type:Date,
        default:Date.now
    }
})

module.exports=mongo.model("Feedback",feedback_coll);
```

Hall Collection Code



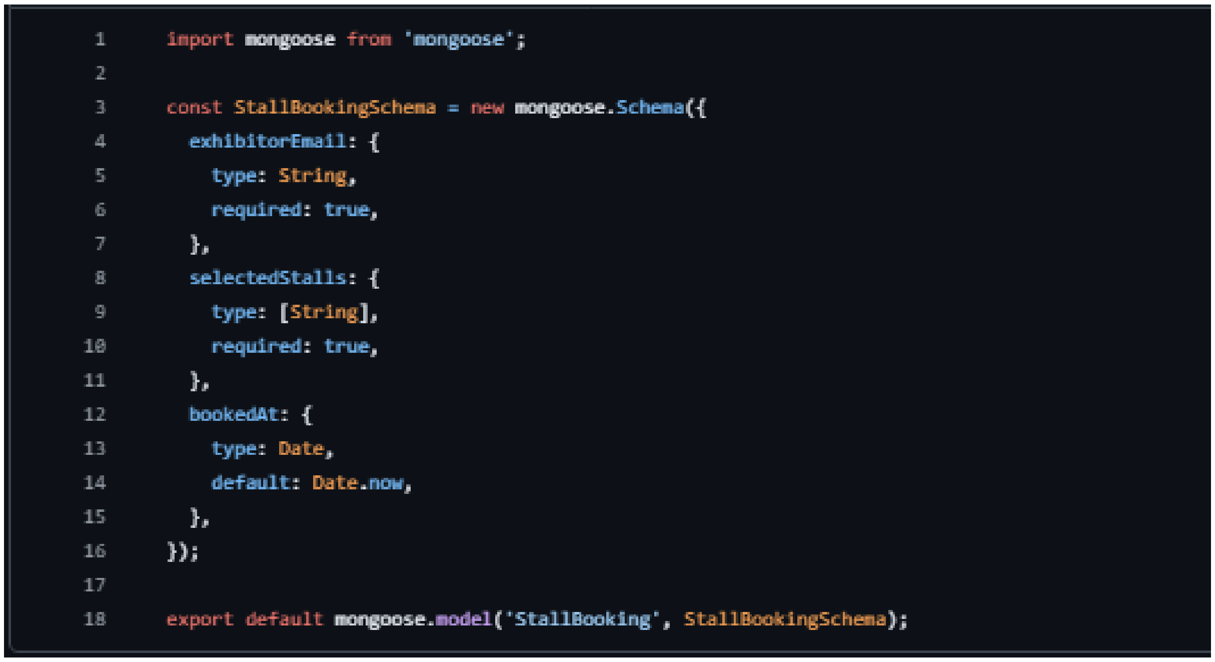
```
let mongo = require("mongoose");

let hall_coll=mongo.Schema({
    hall_no:{
        type:String,
        required:true
    },
    no_of_booth:{
        type:String,
        required:true
    },
    events:{
        refrences:{model:"Expo_Events"},
        type:String
    }
})

module.exports=mongo.model("Halls",hall_coll);
```

Schedule Collection Code



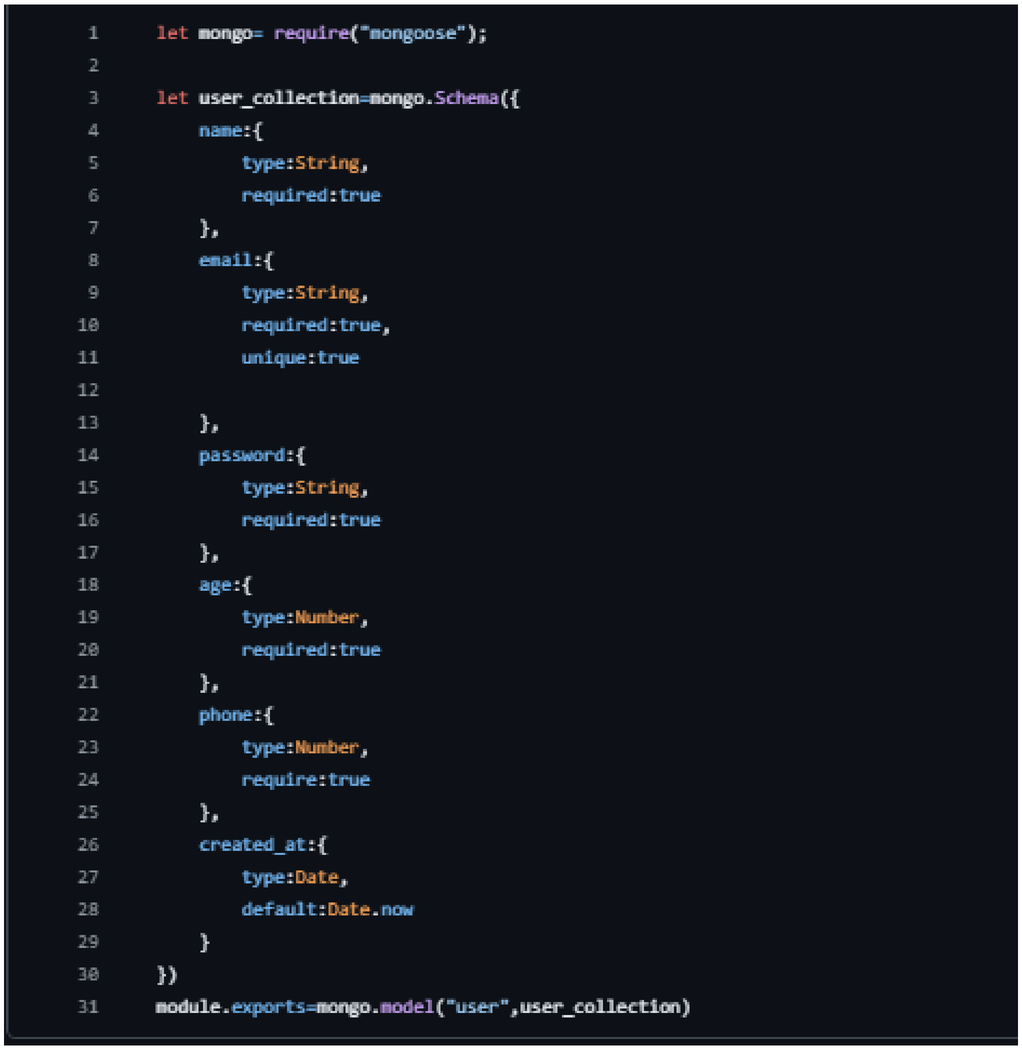
```
let mongo = require("mongoose");

let schedule_coll=mongo.Schema({
    speaker:{
        type:String,
        required:true
    },
    topic:{
        type:String,
        required:true
    },
    location:{
        type:String,
        required:true
    },
    start_date:{
        type:Date,
        required:true
    },
    end_date:{
        type:Date,
        required:true
    },
    hall:{
        refrences:{model:"Halls"},
        type:String
    }
})

module.exports=mongo.model("Expo_Schedule",schedule_coll);
```

Stall’s Booking Collection Code



```
import mongoose from 'mongoose';

const StallBookingSchema = new mongoose.Schema({
  exhibitorEmail: {
    type: String,
    required: true,
  },
  selectedStalls: {
    type: [String],
    required: true,
  },
  bookedAt: {
    type: Date,
    default: Date.now,
  },
});

export default mongoose.model('StallBooking', StallBookingSchema);
```

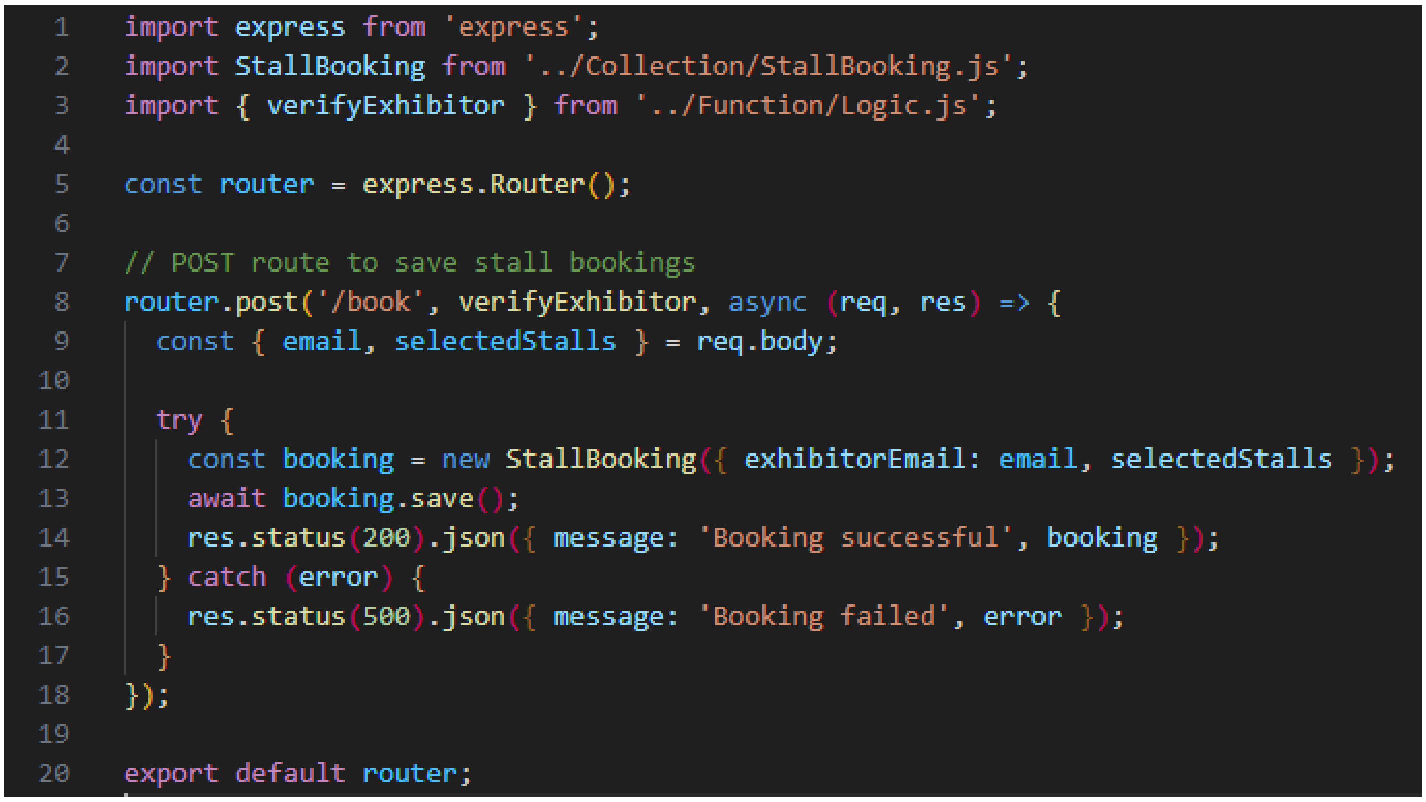
User Collection Code



```
let mongo= require("mongoose");

let user_collection=mongo.Schema({
    name:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true,
        unique:true

    },
    password:{
        type:String,
        required:true
    },
    age:{
        type:Number,
        required:true
    },
    phone:{
        type:Number,
        require:true
    },
    created_at:{
        type:Date,
        default:Date.now
    }
})
module.exports=mongo.model("user",user_collection)
```

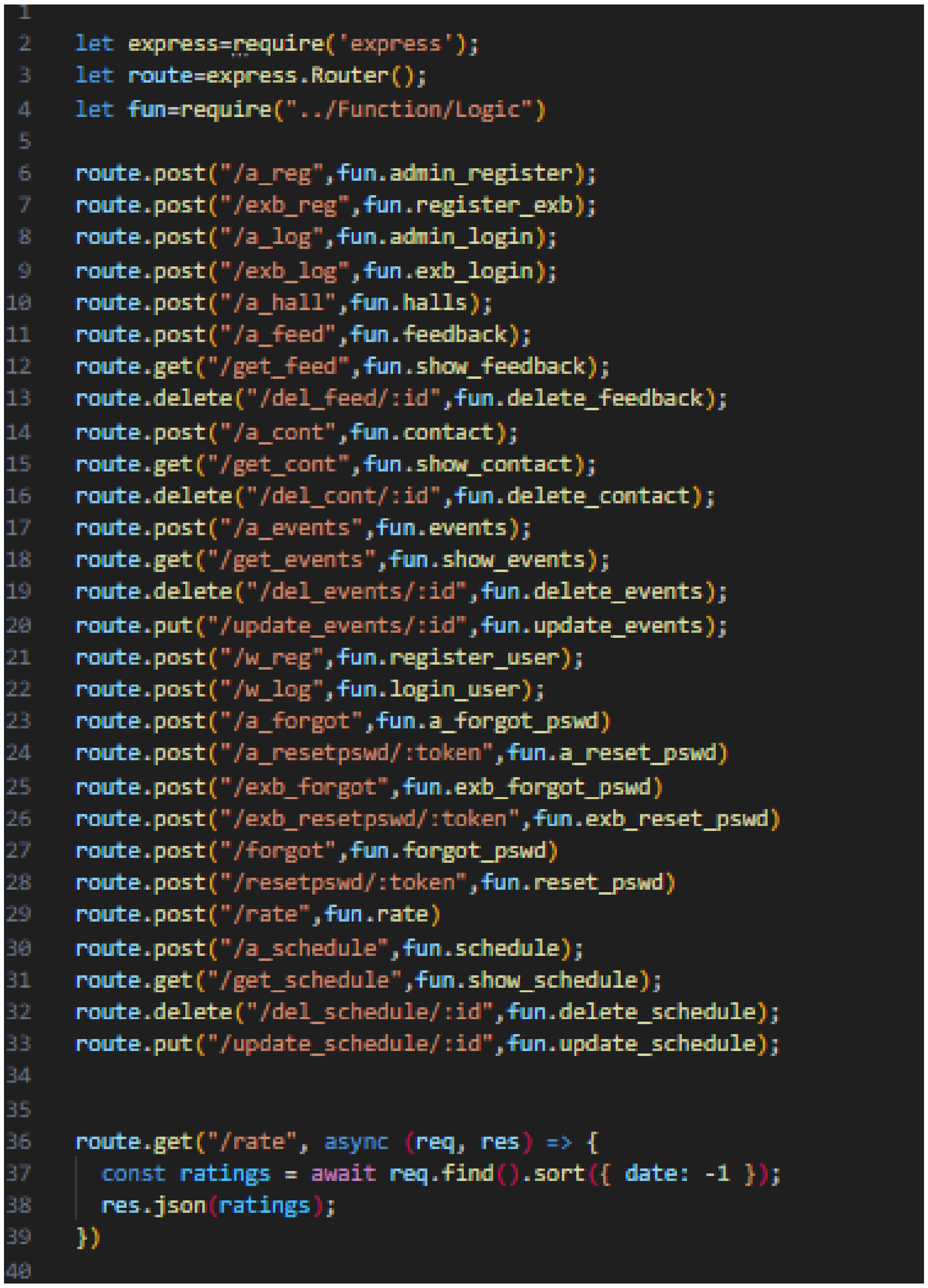
Routes

Stall's Booking Route



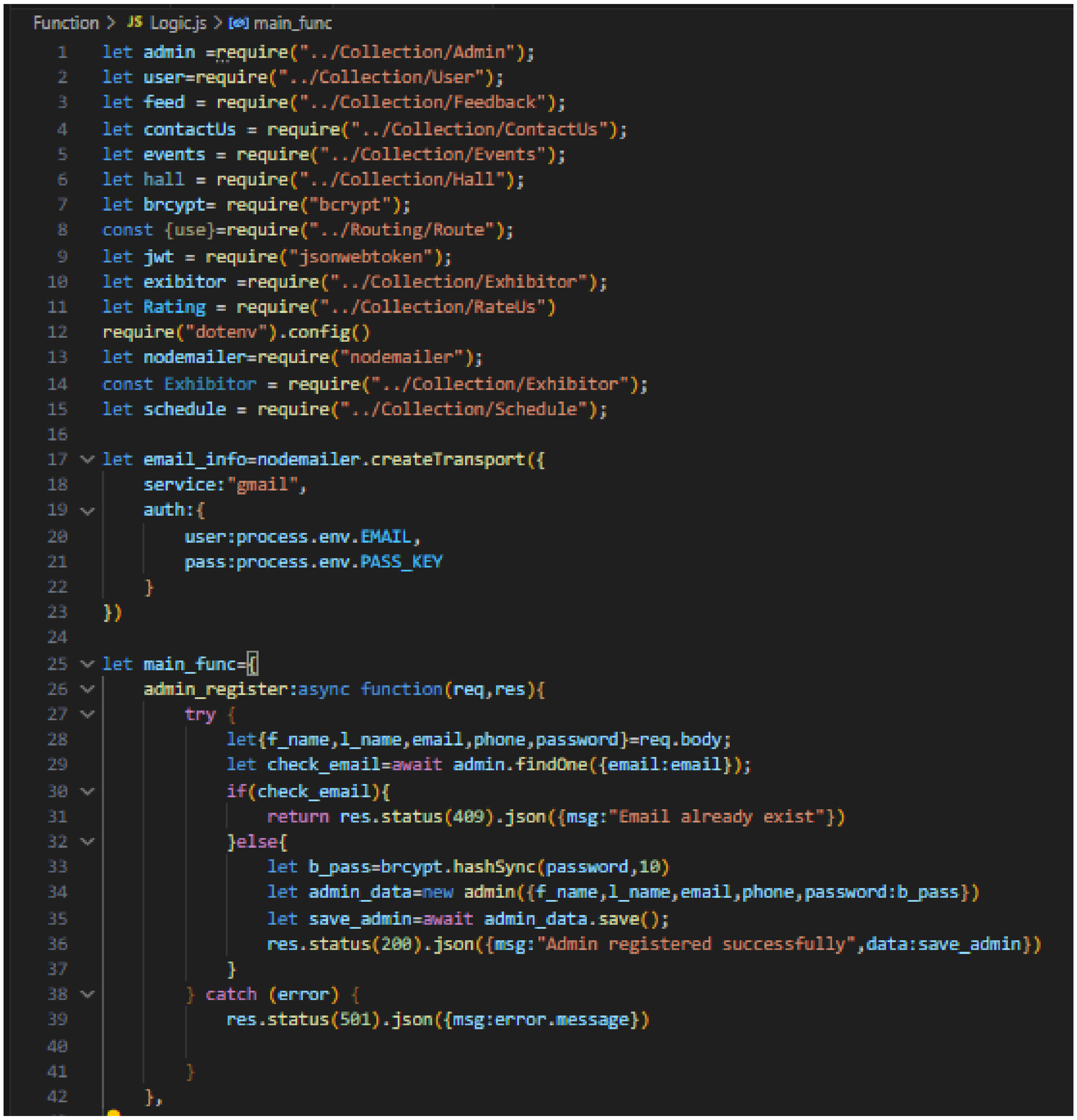
```javascript
import express from 'express';
import StallBooking from '../Collection/StallBooking.js';
import { verifyExhibitor } from '../Function/Logic.js';

const router = express.Router();

// POST route to save stall bookings
router.post('/book', verifyExhibitor, async (req, res) => {
  const { email, selectedStalls } = req.body;

  try {
    const booking = new StallBooking({ exhibitorEmail: email, selectedStalls });
    await booking.save();
    res.status(200).json({ message: 'Booking successful', booking });
  } catch (error) {
    res.status(500).json({ message: 'Booking failed', error });
  }
});

export default router;
```

All Routes



```javascript

let express=require('express');
let route=express.Router();
let fun=require("../Function/Logic")

route.post("/a_reg",fun.admin_register);
route.post("/exb_reg",fun.register_exb);
route.post("/a_log",fun.admin_login);
route.post("/exb_log",fun.exb_login);
route.post("/a_hall",fun.halls);
route.post("/a_feed",fun.feedback);
route.get("/get_feed",fun.show_feedback);
route.delete("/del_feed/:id",fun.delete_feedback);
route.post("/a_cont",fun.contact);
route.get("/get_cont",fun.show_contact);
route.delete("/del_cont/:id",fun.delete_contact);
route.post("/a_events",fun.events);
route.get("/get_events",fun.show_events);
route.delete("/del_events/:id",fun.delete_events);
route.put("/update_events/:id",fun.update_events);
route.post("/w_reg",fun.register_user);
route.post("/w_log",fun.login_user);
route.post("/a_forgot",fun.a_forgot_pswd)
route.post("/a_resetpswd/:token",fun.a_reset_pswd)
route.post("/exb_forgot",fun.exb_forgot_pswd)
route.post("/exb_resetpswd/:token",fun.exb_reset_pswd)
route.post("/forgot",fun.forgot_pswd)
route.post("/resetpswd/:token",fun.reset_pswd)
route.post("/rate",fun.rate)
route.post("/a_schedule",fun.schedule);
route.get("/get_schedule",fun.show_schedule);
route.delete("/del_schedule/:id",fun.delete_schedule);
route.put("/update_schedule/:id",fun.update_schedule);


route.get("/rate", async (req, res) => {
  const ratings = await req.find().sort({ date: -1 });
  res.json(ratings);
})

```

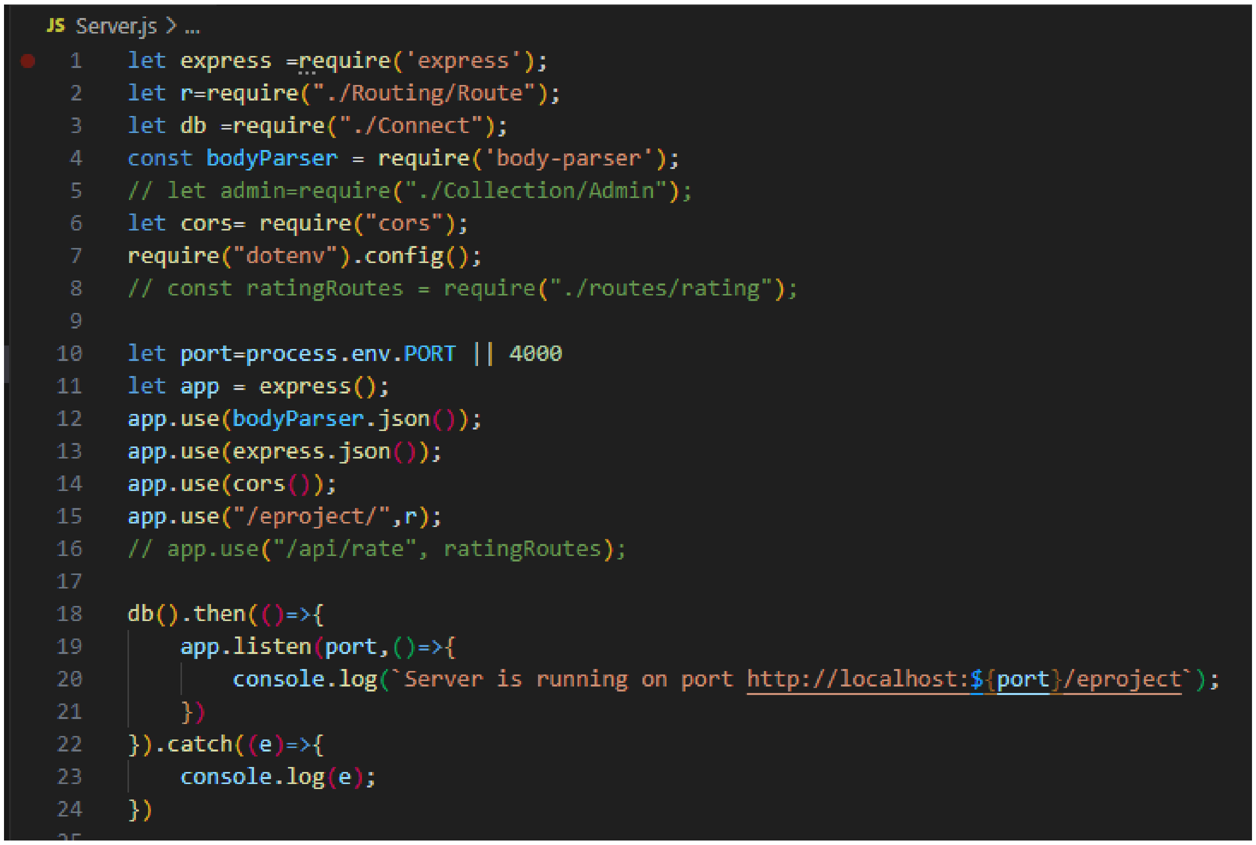
All Funtion's Logics



Function > JS Logic.js > main_func

```javascript
let admin =require("../Collection/Admin");
let user=require("../Collection/User");
let feed = require("../Collection/Feedback");
let contactUs = require("../Collection/ContactUs");
let events = require("../Collection/Events");
let hall = require("../Collection/Hall");
let brcypt= require("bcrypt");
const {use}=require("../Routing/Route");
let jwt = require("jsonwebtoken");
let exibitor =require("../Collection/Exhibitor");
let Rating = require("../Collection/RateUs")
require("dotenv").config()
let nodemailer=require("nodemailer");
const Exhibitor = require("../Collection/Exhibitor");
let schedule = require("../Collection/Schedule");

let email_info=nodemailer.createTransport({
    service:"gmail",
    auth:{
        user:process.env.EMAIL,
        pass:process.env.PASS_KEY
    }
})

let main_func={
    admin_register:async function(req,res){
        try {
            let{f_name,l_name,email,phone,password}=req.body;
            let check_email=await admin.findOne({email:email});
            if(check_email){
                return res.status(409).json({msg:"Email already exist"})
            }else{
                let b_pass=brcypt.hashSync(password,10)
                let admin_data=new admin({f_name,l_name,email,phone,password:b_pass})
                let save_admin=await admin_data.save();
                res.status(200).json({msg:"Admin registered successfully",data:save_admin})
            }
        } catch (error) {
            res.status(501).json({msg:error.message})

        }
    },
```

Server.js File’s Code



JS Server.js > ...

```javascript
let express =require('express');
let r=require("./Routing/Route");
let db =require("./Connect");
const bodyParser = require('body-parser');
// let admin=require("./Collection/Admin");
let cors= require("cors");
require("dotenv").config();
// const ratingRoutes = require("./routes/rating");

let port=process.env.PORT || 4000
let app = express();
app.use(bodyParser.json());
app.use(express.json());
app.use(cors());
app.use("/eproject/",r);
// app.use("/api/rate", ratingRoutes);

db().then(()=>{
    app.listen(port,()=>{
        console.log(`Server is running on port http://localhost:${port}/eproject`);
    })
}).catch((e)=>{
    console.log(e);
})
```