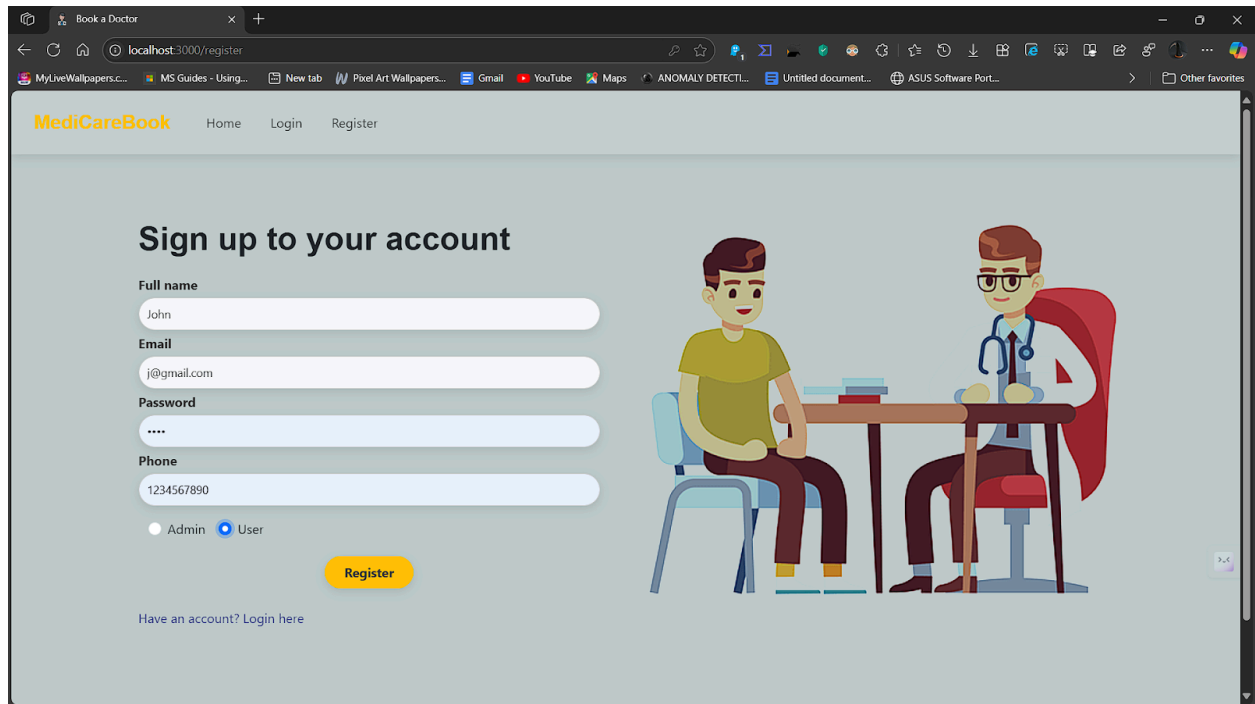


Front End

Pictures on Front page

Register page of DOCS@POT



The screenshot shows a web browser window with the address bar displaying 'localhost:3000/register'. The browser's tab is titled 'Book a Doctor'. The website's header includes the logo 'MediCareBook' and navigation links for 'Home', 'Login', and 'Register'. The main content area is titled 'Sign up to your account' and contains a registration form with the following fields: 'Full name' (with the value 'John'), 'Email' (with the value 'j@gmail.com'), 'Password' (with masked characters '....'), and 'Phone' (with the value '1234567890'). Below these fields are radio buttons for 'Admin' and 'User', with 'User' selected. A yellow 'Register' button is positioned to the right of the form. To the right of the form is an illustration of a doctor in a white coat and stethoscope sitting at a desk, talking to a patient. At the bottom left of the form area, there is a link that says 'Have an account? Login here'.

Book a Doctor

localhost:3000/register

MyLiveWallpapers.c... MS Guides - Using... New tab Pixel Art Wallpapers... Gmail YouTube Maps ANOMALY DETECTL... Untitled document... ASUS Software Port...

MediCareBook Home Login Register

Sign up to your account

Full name
John

Email
j@gmail.com

Password
....

Phone
1234567890

☐ Admin ☒ User

Register

Have an account? Login here


Login Page of DOCS POT

Book a Doctor

localhost:3000/login

MyLiveWallpapers.c... MS Guides - Using... New tab Pixel Art Wallpapers... Gmail YouTube Maps ANOMALY DETECTL... Untitled document... ASUS Software Port... Other favorites

MediCareBook Home Login Register



Sign in to your account

Email

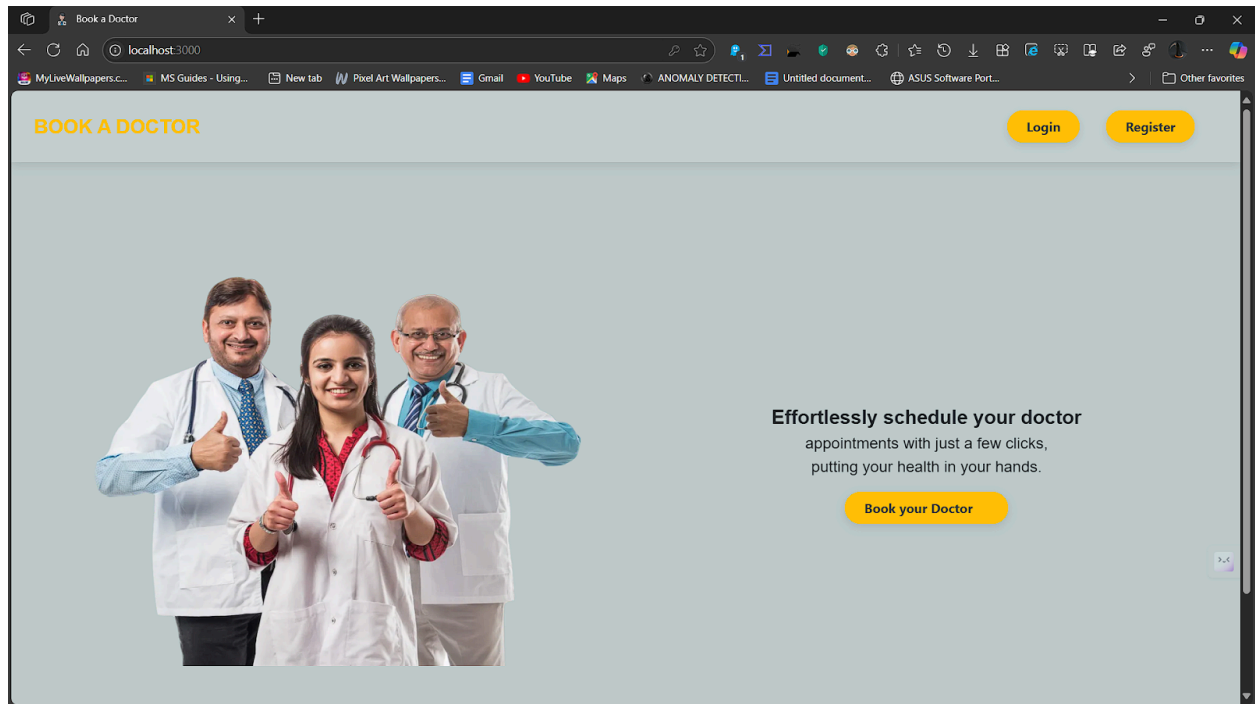
j@gmail.com

Password

Let's Enter

Don't have an account? [Register here](#)

Landing page of DOCS POT



Admin Dashboard DOCS POT

MediCareBook

📅 Users

👤 Doctor

🚪 Logout

🔔 Hi..Admin

All Appointments for Admin Panel

Appointment ID	User Name	Doctor Name	Date	Status
672f7a9b4c8952b18190cb7b	User	Koushick	2024-11-09 20:36	approved
672f7ce54c8952b18190cba5	User	Koushick	2024-11-09 20:46	approved
67303fb33ae507476ffb12d7	User	Koushick	2024-11-10 10:37	approved
6730423aaa10078f304cce6e	User	Koushick	Sun Nov 10 2024 10:48:00 GMT+0530 (India Standard Time)	approved
6730a3e26adc4e5cc1d89d4e	User	Koushick	Sun Nov 10 2024 17:45:00 GMT+0530 (India Standard Time)	approved
6730a3e36adc4e5cc1d89d52	User	Koushick	Sun Nov 10 2024 17:45:00 GMT+0530 (India Standard Time)	approved
6730a73a6adc4e5cc1d89db3	User	Koushick	Sun Nov 10 2024 17:59:00 GMT+0530 (India Standard Time)	approved

© 2023 Copyright: MediCareBook

User Dashboard DOCS@POT

MediCareBook

Users

Doctor

Logout

🔔 Hi..Admin

All Appointments for Admin Panel

Appointment ID	User Name	Doctor Name	Date	Status
672f7a9b4c8952b18190cb7b	User	Koushick	2024-11-09 20:36	approved
672f7ce54c8952b18190cba5	User	Koushick	2024-11-09 20:46	approved
67303fb33ae507476ffb12d7	User	Koushick	2024-11-10 10:37	approved
6730423aaa10078f304cce6e	User	Koushick	Sun Nov 10 2024 10:48:00 GMT+0530 (India Standard Time)	approved
6730a3e26adc4e5cc1d89d4e	User	Koushick	Sun Nov 10 2024 17:45:00 GMT+0530 (India Standard Time)	approved
6730a3e36adc4e5cc1d89d52	User	Koushick	Sun Nov 10 2024 17:45:00 GMT+0530 (India Standard Time)	approved
6730a73a6adc4e5cc1d89db3	User	Koushick	Sun Nov 10 2024 17:59:00 GMT+0530 (India Standard Time)	approved

© 2023 Copyright: MediCareBook

Doctor Dashboard DOCS@POT

Book A Doctor

📅

Appointments

👤

Apply doctor

🚪

Logout

🔔 User

🟢 Doctor Registration request sent successfully

Apply for Doctor

Personal Details:

* Full Name:

* Phone:

* Email:

* Address:

Professional Details:

* Specialization:

* Experience:

* Fees:

* Timings: → ⌚

Submit

© 2023 Copyright: MediCareBook

src => Components

From command prompt, We have to Install Node Modules. Then we use src from Node Modules .
From the src, We have to create the components one by One.

To see the components click on the sub topics present in "src =>Components.

Common

[Register.jsx](#)
[Notification.jsx](#)
[Login.jsx](#)
[Home.jsx](#)

Register.jsx

```
import React, { useState } from 'react'
import Container from 'react-bootstrap/Container';
import Nav from 'react-bootstrap/Nav';
import Navbar from 'react-bootstrap/Navbar';
import { message } from 'antd'
import p2 from '../images/p2.png'
import { Button, Form } from 'react-bootstrap';
import { Link, useNavigate } from 'react-router-dom';
import {
  MDBContainer,
  MDBCard,
  MDBCardBody,
  MDBCardImage,
  MDBRow,
  MDBCol,
  MDBInput,
  MDBRadio,
}
  from 'mdb-react-ui-kit';
import axios from 'axios';

const Register = () => {

  const navigate = useNavigate()
  const [user, setUser] = useState({
    fullName: '', email: '', password: '', phone: '', type: ''
  })

  const handleChange = (e) => {
    setUser({ ...user, [e.target.name]: e.target.value })
  }

  const handleSubmit = async (e) => {
    e.preventDefault()
    try {
      const res = await
    axios.post('http://localhost:8001/api/user/register', user)
      if (res.data.success) {
```

```

        message.success('Registered Successfully')
        navigate('/login')
    }
    else {
        message.error(res.data.message)
    }
} catch (error) {
    console.log(error)
    message.error('Something went wrong')
}
}

return (
    <>
    <Navbar expand="lg" className="bg-body-tertiary">
        <Container fluid>
            <Navbar.Brand>
                <Link to={'/'}>MediCareBook</Link>
            </Navbar.Brand>
            <Navbar.Toggle aria-controls="navbarScroll" />
            <Navbar.Collapse id="navbarScroll">
                <Nav
                    className="me-auto my-2 my-lg-0"
                    style={{ maxHeight: '100px' }}
                    navbarScroll
                >
                </Nav>
                <Nav>
                    <Link to={'/'}>Home</Link>
                    <Link to={'/login'}>Login</Link>
                    <Link to={'/register'}>Register</Link>
                </Nav>

            </Navbar.Collapse>
        </Container>
    </Navbar>

    <MDBContainer className="my-5">

        <MDBCard style={{ border: 'none' }}>
            <MDBRow style={{ background: 'rgb(190, 203, 203)' }}
                className='g-0 border-none p-3'>

                <MDBCol md='6'>
                    <MDBCardBody className='d-flex mx-3 flex-column'>

```

```

        <div className='d-flex flex-row mb-2'>
            <span className="h1 text-center fw-bold">Sign up to your
account</span>
        </div>
        <div className="p-2">
            <Form onSubmit={handleSubmit} >
                <label class="my-1 form-label"
for="formControlLg">Full name</label>
                <MDBInput style={{ height: '40px' }} name='fullName'
value={user.fullName} onChange={handleChange} id='formControlLg'
type='text' size="sm" />

                <label class="my-1 form-label"
for="formControlLg">Email</label>
                <MDBInput style={{ height: '40px' }} name='email'
value={user.email} onChange={handleChange} id='formControlLg' type='email'
size="sm" />

                <label class="my-1 form-label"
for="formControlLg">Password</label>
                <MDBInput style={{ height: '40px' }} name='password'
value={user.password} onChange={handleChange} id='formControlLg'
type='password' size="sm" />

                <label class="my-1 form-label"
for="formControlLg">Phone</label>
                <MDBInput style={{ height: '40px' }} name='phone'
value={user.phone} onChange={handleChange} id='formControlLg' type='phone'
size="sm" />

            <Container className='my-3'>
                <MDBRadio
                    name='type'
                    id='inlineRadio1'
                    checked={user.type === 'admin'}
                    value='admin'
                    onChange={handleChange}
                    label='Admin'
                    inline
                />
                <MDBRadio
                    name='type'
                    id='inlineRadio2'
                    checked={user.type === 'user'}
                    value='user'
                    onChange={handleChange}
                    label='User'

```

```

        inline
      />
    </Container>

    <Button style={{marginTop: '20px'}} className="mb-4
bg-dark" variant='dark' size='lg' type="submit">Register</Button>
  </Form>
  <p className="mb-5 pb-md-2" style={{ color: '#393f81'
}}>Have an account? <Link to={'/login'} style={{ color: '#393f81' }}>Login
here</Link></p>

</div>

  </MDBCardBody>
</MDBCol>

  <MDBCol md='6'>
    <MDBCardImage style={{ mixBlendMode: 'darken' }} src={p2}
alt="login form" className='rounded-start w-100' />
  </MDBCol>

  </MDBRow>
</MDBCard>

  </MDBContainer>
</>
)
}

export default Register

```

Notification.jsx

```
import { Tabs, message } from 'antd'
import axios from 'axios'
import React, { useState, useEffect } from 'react'
import { useNavigate } from 'react-router-dom'

const Notification = () => {
  const [user, setUser] = useState()
  const navigate = useNavigate()
  const getUser = () => {
    const userdata = JSON.parse(localStorage.getItem('userData'))
    if (userdata) {
      setUser(userdata)
    }
  }

  const handleAllMarkRead = async () => {
    try {
      const res = await
    axios.post('http://localhost:8001/api/user/getallnotification', { userId:
    user._id }, {
      headers: {
        Authorization: `Bearer ${localStorage.getItem('token')}`,
      },
    })
    if (res.data.success) {
      const updatedUser = { ...user, notification: [],
    seennotification: [...user.seennotification, ...user.notification] };
      localStorage.setItem('userData', JSON.stringify(updatedUser));

      message.success(res.data.message)
      setUser({ ...user, notification: [] })
    }
    else {
      message.error(res.data.message)
    }
  } catch (error) {
    console.log(error)
    message.error("something went wrong")
  }
}

const handledelateAllMark = async () => {
  try {
```

```

        const res = await
        axios.post('http://localhost:8001/api/user/deleteallnotification', {
        userId: user._id }, {
            headers: {
                Authorization: `Bearer ${localStorage.getItem('token')}`,
            },
        })
        if (res.data.success) {
            setUser({ ...user, seennotification: [] });
            message.success(res.data.message)
        }
        else {
            message.error(res.data.message)
        }
    } catch (error) {
        console.log(error)
        message.error("something went wrong")
    }
}

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

return (
    <div>
        <h2 className='p-3 text-center'>Notification</h2>
        <Tabs>
            <Tabs.TabPane tab="unread" key={0}>
                <div className="d-flex justify-content-end">
                    <h4 style={{ cursor: 'pointer', }}
onClick={handleAllMarkRead} className="p-2">Mark all read</h4>
                </div>
                {user?.notification.map((notificationMsg) => (
                    <div onClick={notificationMsg.onClickPath}
className="card">
                        <div className="card-text">
                            {notificationMsg.message}
                        </div>
                    </div>
                ))}
            </Tabs.TabPane>
            <Tabs.TabPane tab="Read" key={1}>
                <div className="d-flex justify-content-end">
                    <h4 style={{ cursor: 'pointer' }}
onClick={handledeleteAllMark} className="p-2">delete all read</h4>

```



```

        </div>
        {user?.seennotification.map((notificationMsg) => (
          <div style={{ cursor: 'pointer' }} className="card" >
            <div className="card-text" onClick={() =>
              navigate(notificationMsg.onClickPath)}>
              {notificationMsg.message}
            </div>
          </div>
        ))}
      </Tabs.TabPane>
    </Tabs>
  </div>
)
}

export default Notification

```

Login.jsx

```

import React, { useState } from 'react'
import Container from 'react-bootstrap/Container';
import Nav from 'react-bootstrap/Nav';
import Navbar from 'react-bootstrap/Navbar';
import { message } from 'antd';
import { Button, Form } from 'react-bootstrap';
import photo1 from '../images/photo1.png'
import axios from 'axios';
import { Link, useNavigate } from 'react-router-dom';
import {
  MDBContainer,
  MDBCard,
  MDBCardBody,
  MDBCardImage,
  MDBRow,
  MDBCol,
  MDBInput
}
from 'mdb-react-ui-kit';

const Login = () => {
  const navigate = useNavigate()
  const [user, setUser] = useState({
    email: '', password: ''
  })

```

```

const handleChange = (e) => {
  setUser({ ...user, [e.target.name]: e.target.value })
}

const handleSubmit = async (e) => {
  e.preventDefault();
  try {
    const res = await axios.post("http://localhost:8001/api/user/login",
user);
    if (res.data.success) {
      localStorage.setItem('token', res.data.token);
      localStorage.setItem('userData',
JSON.stringify(res.data.userData));
      message.success('Login successfully');
      const isLoggedIn = JSON.parse(localStorage.getItem("userData"));
      const { type } = isLoggedIn

      switch (type) {
        case "admin":
          navigate("/adminHome")
          break;
        case "user":
          navigate("/userhome")
          break;

        default:
          navigate("/Login")
          break;
      }
    }
    else{
      message.error(res.data.message)
    }
  } catch (error) {
    console.log(error);
    message.error('Something went wrong')
  }
};

```

```

return (
  <>
    <Navbar expand="lg" className="bg-body-tertiary">
      <Container fluid>
        <Navbar.Brand>

```

```

        <Link to={'/'}>MediCareBook</Link>
    </Navbar.Brand>
    <Navbar.Toggle aria-controls="navbarScroll" />
    <Navbar.Collapse id="navbarScroll">
        <Nav
            className="me-auto my-2 my-lg-0"
            style={{ maxHeight: '100px' }}
            navbarScroll
        >
    </Nav>
    <Nav>
        <Link to={'/'}>Home</Link>
        <Link to={'/login'}>Login</Link>
        <Link to={'/register'}>Register</Link>
    </Nav>

    </Navbar.Collapse>
</Container>
</Navbar>

<MDBContainer className="my-5">

    <MDBCard style={{ border: 'none' }}>
        <MDBRow style={{ background: 'rgb(190, 203, 203)' }}
            className='g-0 border-none p-3'>

            <MDBCol md='6'>
                <MDBCardImage src={photo1} alt="login form"
                    className='rounded-start w-100' />
            </MDBCol>

            <MDBCol md='6'>
                <MDBCardBody className='d-flex mx-5 flex-column'>

                    <div className='d-flex flex-row mt-2 mb-5'>
                        <span className="h1 fw-bold mb-0">Sign in to your
account</span>
                    </div>

                    <Form onSubmit={handleSubmit}>
                        <label class="form-label"
                            for="formControlLgEmail">Email</label>
                        <MDBInput
                            style={{ margin: '5px auto' }}
                            name="email"
                            value={user.email}

```

```

        onChange={handleChange}
        id="formControlLgEmail"
        type="email"
        size="md"
        autoComplete='off'
      />
      <label class="form-label"
for="formControlLgPassword">Password</label>
      <MDBInput
        style={{ margin: '5px auto' }}
        name="password"
        value={user.password}
        onChange={handleChange}
        id="formControlLgPassword"
        type="password"
        size="md"
        autoComplete='off'
      />
      <Button className="mb-4 px-5 bg-dark" size='lg'
type='submit'>Login</Button>
    </Form>
    <p className="mb-5 pb-lg-2" style={{ color: '#393f81'
}}>Don't have an account? <Link to={'/register'} style={{ color: '#393f81'
}}>Register here</Link></p>

    </MDBCardBody>
  </MDBCol>

  </MDBRow>
</MDBCard>

  </MDBContainer>
</>
);
}

export default Login;

```

Home.jsx

```
import React from 'react'
import Nav from 'react-bootstrap/Nav';
import Navbar from 'react-bootstrap/Navbar';
import { Link } from 'react-router-dom';
import { Container, Button } from 'react-bootstrap';

import p3 from '../images/p3.webp'

const Home = () => {
  return (
    <>
      <Navbar expand="lg" className="bg-body-tertiary">
        <Container fluid>
          <Navbar.Brand>
            <Link to={'/'}>MediCareBook</Link>
          </Navbar.Brand>
          <Navbar.Toggle aria-controls="navbarScroll" />
          <Navbar.Collapse id="navbarScroll">
            <Nav
              className="me-auto my-2 my-lg-0"
              style={{ maxHeight: '100px' }}
              navbarScroll
            >
              </Nav>
            <Nav>
              <Link to={'/'}>Home</Link>
              <Link to={'/login'}>Login</Link>
              <Link to={'/register'}>Register</Link>
            </Nav>
          </Navbar.Collapse>
        </Container>
      </Navbar>

      <div className='home-container'>
        <div className="left-side">
          <img alt="" src={p3} />
        </div>
        <div className="right-side">
          <p>
            <span className='f-letter'>Effortlessly schedule your
            doctor</span><br />
            <span className='s-letter'>appointments with just a few
            clicks,</span> <br />

```

```
        <span className='t-letter'>putting your health in your
hands.</span><br />
        <Button color='info' className='mt-3 register'><Link
to={'/Login'}>Book your Doctor</Link></Button>
    </p>
</div>
</div>
```

```
<Container>
  <h1 className='text-center mb-4'>About Us</h1>
  <div className="right-side">
    <p>
```

Booking a doctor appointment has never been easier. With our convenient online platform, you can quickly and effortlessly schedule your appointments from the comfort of your own home. No more waiting on hold or playing phone tag with busy receptionists.

Our user-friendly interface allows you to browse through a wide range of doctors and healthcare providers, making it simple to find the perfect match for your needs. Whether you require a routine check-up, specialist consultation, or urgent care, we have a diverse network of medical professionals ready to serve you.

Gone are the days of flipping through phone directories or relying on word-of-mouth recommendations. Our comprehensive database provides detailed profiles of each doctor, including their specialties, qualifications, and availability. You can read reviews from other patients to gain insights into their experiences and make an informed decision.

Once you've found the ideal doctor, booking an appointment is just a few clicks away. Select a convenient date and time slot, and our system will handle the rest. You'll receive instant confirmation, along with reminders leading up to your appointment, ensuring you never miss a crucial healthcare visit.

Take control of your health and experience the convenience of online doctor appointment booking. Say goodbye to long waits and hello to seamless scheduling. Join our platform today and prioritize your well-being with ease and efficiency.

With our advanced booking system, you can say goodbye to the hassle of traditional appointment booking. Our platform offers real-time availability, allowing you to choose from a range of open slots that fit your schedule. Whether you prefer early morning, evening, or weekend appointments, we have options to accommodate your needs.

We understand that emergencies can arise unexpectedly. That's why we offer same-day and next-day appointment options for urgent cases. No more waiting weeks for an available slot. We prioritize your health and ensure prompt access to medical care when you need it most.

Our platform also provides convenient features such as online payment options and the ability to securely store your medical history and insurance information. This streamlines the check-in process, saving you valuable time during your visit.

In addition, our dedicated support team is available to assist you every step of the way. If you have any questions or need assistance with booking, our friendly representatives are just a call or message away. We strive to provide exceptional customer service and ensure a seamless experience for our users.

Experience the convenience and efficiency of our doctor appointment booking platform. Take charge of your health and prioritize your well-being with ease. Join our growing community of satisfied users and discover the future of healthcare scheduling.

</p>
</div>
</Container>

</>
)
}

export default Home

Admin

[AdminUsers.jsx](#)
[AdminHome.jsx](#)
[AdminDoctors.jsx](#)
[AdminAppointments.jsx](#)

AdminUsers.jsx

```
import React, { useEffect, useState } from 'react';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container } from 'react-bootstrap';
import axios from 'axios';

const AdminAppointments = () => {

  const [allAppointments, setAllAppointments] = useState([])

  const getAppointments = async () => {
    try {
      const res = await
    axios.get('http://localhost:8001/api/admin/getallAppointmentsAdmin', {
      headers: {
        Authorization: `Bearer ${localStorage.getItem('token')}`
      },
    })
    if (res.data.success) {
      setAllAppointments(res.data.data)
    }

    } catch (error) {
      console.log(error)
    }
  }

  useEffect(() => {
    getAppointments();
  }, [])
  return (
    <div>
      <h2 className='p-3 text-center'>All Appointments for Admin
Panel</h2>
      <Container>
        <Table className='my-3' striped bordered hover>
          <thead>
            <tr>
```

```

        <th>Appointment ID</th>
        <th>User Name</th>
        <th>Doctor Name</th>
        <th>Date</th>
        <th>Status</th>

    </tr>
</thead>
<tbody>
    {allAppointments.length > 0 ? (
      allAppointments.map((appointment) => {
        return (
          <tr key={appointment._id}>
            <td>{appointment._id}</td>
            <td>{appointment.userInfo.fullName}</td>
            <td>{appointment.doctorInfo.fullName}</td>
            <td>{appointment.date}</td>
            <td>{appointment.status}</td>
          </tr>
        )
      })
    ) : (
      <tr>
        <td colspan={6}>
          <Alert variant="info">
            <Alert.Heading>No Appointments to
show</Alert.Heading>
          </Alert>
        </td>
      </tr>
    )}
</tbody>
</Table>
</Container>
</div>
)
}

export default AdminAppointments

```

AdminHome.jsx

```
import React, { useEffect, useState } from 'react'

import axios from 'axios';
import { Link } from 'react-router-dom';
import CalendarMonthIcon from '@mui/icons-material/CalendarMonth';
import MedicationIcon from '@mui/icons-material/Medication';
import LogoutIcon from '@mui/icons-material/Logout';
import NotificationsIcon from '@mui/icons-material/Notifications';
import { Badge } from 'antd';
import Notification from '../common/Notification';
import AdminUsers from '../AdminUsers';
import AdminDoctors from '../AdminDoctors';
import AdminAppointments from '../AdminAppointments';

const AdminHome = () => {
  const [userdata, setUserData] = useState({})
  const [activeMenuItem, setActiveMenuItem] = useState('');

  const getUserData = async () => {
    try {
      await axios.post('http://localhost:8001/api/user/getuserdata',
        {}, {
          headers: {
            Authorization: "Bearer " + localStorage.getItem('token')
          },
        });
    } catch (error) {
      console.log(error);
    }
  };

  const getUser = () => {
    const user = JSON.parse(localStorage.getItem('userData'))
    if (user) {
      setUserData(user)
    }
  }

  useEffect(() => {
    getUserData();
    getUser()
  }, []);

  const logout = () => {
    localStorage.removeItem("token")
  }
}
```

```

    localStorage.removeItem("userData")
    window.location.href = "/"

  }

  const handleMenuItemClick = (menuItem) => {
    setActiveMenuItem(menuItem);
  };
  return (
    <>

      <div className='main'>
        <div className="layout">
          <div className="sidebar">
            <div className="logo">
              <h2>MediCareBook</h2>
            </div>
            <div className="menu">
              <div className={`menu-items ${activeMenuItem ===
'adminusers' ? 'active' : ''}`} onClick={() =>
handleMenuItemClick('adminusers')}>
                <CalendarMonthIcon className='icon'
/><Link>Users</Link>
              </div>
              <div className={`menu-items ${activeMenuItem ===
'admindocctors' ? 'active' : ''}`} onClick={() =>
handleMenuItemClick('admindocctors')}>
                <MedicationIcon className='icon'
/><Link>Doctor</Link>
              </div>
              <div className="menu-items">
                <LogoutIcon className='icon' /><Link
onClick={logout}>Logout</Link>
              </div>
            </div>
          </div>
          <div className="content">
            <div className="header">
              <div className="header-content" style={{ cursor:
'pointer' }}>
                <Badge className={`notify ${activeMenuItem ===
'notification' ? 'active' : ''}`} onClick={() =>
handleMenuItemClick('notification')} count={userdata?.notification ?
userdata.notification.length : 0}>
                  <NotificationsIcon className='icon' />
                </Badge>

```

```

        <h3>Hi..{userdata.fullName}</h3>
      </div>
    </div>
    <div className="body">
      {activeMenuItem === 'notification' && <Notification
/>>
      {activeMenuItem === 'adminusers' && <AdminUsers />}
      {activeMenuItem === 'admindocctors' && <AdminDoctors
/>>
      {activeMenuItem !== 'notification' && activeMenuItem
!== 'adminusers' && activeMenuItem !== 'admindocctors' &&
<AdminAppointments />}
    </div>
  </div>
</div>
</>
);
};

export default AdminHome;

```

AdminDoctors.jsx

```

import React, { useEffect, useState } from 'react';
import { Button } from 'react-bootstrap';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container } from 'react-bootstrap';
import axios from 'axios';
import { message } from 'antd';

const AdminDoctors = () => {

  const [doctors, setDoctors] = useState([])

  const getDoctors = async () => {
    try {
      const res = await
    axios.get('http://localhost:8001/api/admin/getalldoctors', {
      headers: {
        Authorization: `Bearer ${localStorage.getItem("token")}`,
      },
    })
  }
}

```

```

        if (res.data.success) {
            setDoctors(res.data.data)
        }
    } catch (error) {
        console.log(error)
        message.error('something went wrong')
    }
}

const handleApprove = async (doctorId, status, userid) => {
    console.log(doctorId, status, userid)
    try {
        const res = await
    axios.post('http://localhost:8001/api/admin/getapprove', { doctorId,
    status, userid }, {
        headers: {
            Authorization: `Bearer ${localStorage.getItem("token")}`,
        },
    })

        if (res.data.success) {
            message.success(res.data.message)
        }
        console.log(res)
    } catch (error) {
        console.log(error)
        message.error('something went wrong')
    }
}

const handleReject = async (doctorId, status, userid) => {
    console.log(doctorId, status, userid)
    try {
        const res = await
    axios.post('http://localhost:8001/api/admin/getreject', { doctorId,
    status, userid }, {
        headers: {
            Authorization: `Bearer ${localStorage.getItem("token")}`,
        },
    })

        if (res.data.success) {
            message.success(res.data.message)
        }
        console.log(res)
    } catch (error) {
        console.log(error)
    }
}

```

```

        message.error('something went wrong')
    }
}

useEffect(() => {
    getDoctors()

}, [])

return (
    <div>
        <h2 className='p-3 text-center'>All Doctors</h2>

        <Container>
            <Table striped bordered hover>
                <thead>
                    <tr>
                        <th>Key</th>
                        <th>Name</th>
                        <th>Email</th>
                        <th>Phone</th>
                        <th>Action</th>
                    </tr>
                </thead>
                <tbody>
                    {doctors.length > 0 ? (
                        doctors.map((user) => {
                            return (
                                <tr key={user._id}>
                                    <td>{user._id}</td>
                                    <td>{user.fullName}</td>
                                    <td>{user.email}</td>
                                    <td>{user.phone}</td>
                                    <td>{user.status === 'pending' ?
                                        <Button onClick={() =>
handleApprove(user._id, 'approved', user.userId)} className='mx-2'
size='sm' variant="outline-success">
                                            Approve
                                        </Button>
                                        :
                                        <Button onClick={() =>
handleReject(user._id, 'rejected', user.userId)} className='mx-2'
size='sm' variant="outline-danger">
                                            Reject
                                        </Button>}</td>
                                </tr>
                            )
                        })
                    ) : null}
                </tbody>
            </Table>
        </Container>
    </div>
)

```

```

        </tr>
    )
    })
    ) : (
        <tr>
            <td colSpan={5}>
                <Alert variant="info">
                    <Alert.Heading>No Doctors to
show</Alert.Heading>
                </Alert>
            </td>
        </tr>

    )}
</tbody>
</Table>
</Container>
</div>
)
}

export default AdminDoctors;

```


AdminAppointments.jsx

```
import React, { useEffect, useState } from 'react';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container } from 'react-bootstrap';
import axios from 'axios';

const AdminAppointments = () => {

  const [allAppointments, setAllAppointments] = useState([])

  const getAppointments = async () => {
    try {
      const res = await
    axios.get('http://localhost:8001/api/admin/getallAppointmentsAdmin', {
      headers: {
        Authorization: `Bearer ${localStorage.getItem('token')}`
      },
    })
    if (res.data.success) {
      setAllAppointments(res.data.data)
    }

    } catch (error) {
      console.log(error)
    }
  }

  useEffect(() => {
    getAppointments();
  }, [])
  return (
    <div>
      <h2 className='p-3 text-center'>All Appointments for Admin
Panel</h2>
      <Container>
        <Table className='my-3' striped bordered hover>
          <thead>
            <tr>
              <th>Appointment ID</th>
              <th>User Name</th>
              <th>Doctor Name</th>
              <th>Date</th>
              <th>Status</th>
            </tr>
          </thead>
          <tbody>
            {allAppointments.map((appointment) => (
              <tr>
                <td>{appointment.id}</td>
                <td>{appointment.user}</td>
                <td>{appointment.doctor}</td>
                <td>{appointment.date}</td>
                <td>{appointment.status}</td>
              </tr>
            ))}
          </tbody>
        </Table>
      </Container>
    </div>
  )
}
```

```

        </tr>
      </thead>
      <tbody>
        {allAppointments.length > 0 ? (
          allAppointments.map((appointment) => {
            return (
              <tr key={appointment._id}>
                <td>{appointment._id}</td>
                <td>{appointment.userInfo.fullName}</td>
                <td>{appointment.doctorInfo.fullName}</td>
                <td>{appointment.date}</td>
                <td>{appointment.status}</td>
              </tr>
            )
          })
        ) : (
          <tr>
            <td colspan={6}>
              <Alert variant="info">
                <Alert.Heading>No Appointments to
show</Alert.Heading>
              </Alert>
            </td>
          </tr>
        )}
      </tbody>
    </Table>
  </Container>
</div>
)
}

export default AdminAppointments

```

User

UserHome.jsx
UserAppointments.jsx
DoctorList.jsx
ApplyDoctor.jsx
AddDoctor.jsx

UserHome.jsx

```
import React, { useEffect, useState } from 'react'
import { Badge, Row } from 'antd';
import Notification from '../common/Notification';
import axios from 'axios';
import { Link } from 'react-router-dom';
import CalendarMonthIcon from '@mui/icons-material/CalendarMonth';
import MedicationIcon from '@mui/icons-material/Medication';
import LogoutIcon from '@mui/icons-material/Logout';
import NotificationsIcon from '@mui/icons-material/Notifications';
import { Container } from 'react-bootstrap';

import ApplyDoctor from '../ApplyDoctor';
import UserAppointments from '../UserAppointments';
import DoctorList from '../DoctorList';

const UserHome = () => {
  const [doctors, setDoctors] = useState([])
  const [userdata, setUserData] = useState({})
  const [activeMenuItem, setActiveMenuItem] = useState('');

  const getUser = () => {
    const user = JSON.parse(localStorage.getItem('userData'))
    if (user) {
      setUserData(user)
    }
  }

  const getUserData = async () => {
    try {
      await axios.post('http://localhost:8001/api/user/getuserdata',
        {}, {
          headers: {
            Authorization: "Bearer " + localStorage.getItem('token')
          }
        }
      )
    } catch (error) {
      console.log(error)
    }
  }
}
```

```

        },
    });
} catch (error) {
    console.log(error);
}
};

const getDoctorData = async () => {
    try {
        const res = await
axios.get('http://localhost:8001/api/user/getalldoctorsu', {
            headers: {
                Authorization: "Bearer " + localStorage.getItem('token')
            },
        });
        if (res.data.success) {
            setDoctors(res.data.data)
        }
    } catch (error) {
        console.log(error);
    }
};

useEffect(() => {
    getUser();
    getUserData();
    getDoctorData()
}, []);

const logout = () => {
    localStorage.removeItem("token");
    localStorage.removeItem("userData");
    window.location.href = "/";
};

const handleMenuItemClick = (menuItem) => {
    setActiveMenuItem(menuItem);
};

return (
    <>
        <div className='main'>
            <div className="layout">
                <div className="sidebar">
                    <div className="logo">
                        <h2>MediCareBook</h2>
                    </div>

```

```

        <div className="menu">
            {/* <div className="menu-items">
                <CalendarMonthIcon className='icon'
/><Link>Home</Link>
            </div> */}
            <div className={`menu-items ${activeMenuItem ===
'userappointments' ? 'active' : ''}`} onClick={() =>
handleMenuItemClick('userappointments')}>
                <CalendarMonthIcon className='icon'
/><Link>Appointments</Link>
            </div>
            {userdata.isdoctor === true ? <></> : <div
className={`menu-items ${activeMenuItem === 'applyDoctor' ? 'active' :
''}`} onClick={() => handleMenuItemClick('applyDoctor')}>
                <MedicationIcon className='icon' /><Link>Apply
doctor</Link>
            </div>}

            <div className="menu-items" onClick={logout}>
                <LogoutIcon className='icon' /><Link>Logout</Link>
            </div>
        </div>
    </div>
    <div className="content">
        <div className="header">
            <div className="header-content">

                <Badge className={`notify ${activeMenuItem ===
'notification' ? 'active' : ''}`} onClick={() =>
handleMenuItemClick('notification')} count={userdata?.notification ?
userdata.notification.length : 0}>
                    <NotificationsIcon className="icon" />
                </Badge>

                {userdata.isdoctor === true && <h3>Dr. </h3>}
                <h3>{userdata.fullName}</h3>
            </div>
        </div>
        <div className="body">
            {activeMenuItem === 'applyDoctor' && <ApplyDoctor
userId={userdata._id} />}
            {activeMenuItem === 'notification' && <Notification
/>}
            {activeMenuItem === 'userappointments' &&
<UserAppointments />}

```

```

        {activeMenuItem !== 'applyDoctor' && activeMenuItem
!== 'notification' && activeMenuItem !== 'userappointments' && <Container>
        <h2 className="text-center p-2">Home</h2>

        {userdata.isdoctor === true ? <></> : <Row>
        {doctors && doctors.map((doctor, i) => {
        let notifyDoc = doctor.userId
        return (
        <DoctorList userDoctorId={notifyDoc}
doctor={doctor} userdata={userdata} key={i} />
        )
        }}}
        </Row>}
        </Container>}
    </div>
</div>
</div>
</div>
</>
);
};

export default UserHome;

```

UserAppointments.jsx

```
import React, { useEffect, useState } from 'react';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container, Button } from 'react-bootstrap';
import axios from 'axios';
import { message } from 'antd';

const UserAppointments = () => {
  const [userid, setUserId] = useState();
  const [type, setType] = useState(false);
  const [userAppointments, setUserAppointments] = useState([]);
  const [doctorAppointments, setDoctorAppointments] = useState([]);

  const getUser = () => {
    const user = JSON.parse(localStorage.getItem('userData'));
    if (user) {
      const { _id, isdoctor } = user;
      setUserId(_id);
      setType(isdoctor);
    } else {
      alert('No user to show');
    }
  };

  const getUserAppointment = async () => {
    console.log(userid)
    try {
      const res = await
    axios.get('http://localhost:8001/api/user/getuserappointments', {
      headers: {
        Authorization: `Bearer ${localStorage.getItem("token")}`,
      },
      params: {
        userId: userid,
      },
    });
    if (res.data.success) {
      message.success(res.data.message);
      setUserAppointments(res.data.data);
    }
  }
}
```



```

    } catch (error) {
      console.log(error);
      message.error('Something went wrong');
    }
  };

const getDoctorAppointment = async () => {
  console.log(userid)
  try {
    const res = await
    axios.get('http://localhost:8001/api/doctor/getdoctorappointments', {
      headers: {
        Authorization: `Bearer ${localStorage.getItem("token")}`,
      },
      params: {
        userId: userid,
      },
    });
    if (res.data.success) {
      message.success(res.data.message);
      setDoctorAppointments(res.data.data);
    }
  } catch (error) {
    console.log(error);
    message.error('Something went wrong');
  }
};

const handleStatus = async (userid, appointmentId, status) => {
  try {
    const res = await
    axios.post('http://localhost:8001/api/doctor/handlestatus', {
      userid,
      appointmentId,
      status,
    },
    {
      headers: {
        Authorization: `Bearer ${localStorage.getItem("token")}`,
      },
    })
    if (res.data.success) {
      message.success(res.data.message)
      getDoctorAppointment();
      getUserAppointment();
    }
  } catch (error) {

```

```

        console.log(error);
        message.error('Something went wrong');
    }
};

useEffect(() => {
    getUser();
}, [userid]);

useEffect(() => {
    if (type === true) {
        getDoctorAppointment();
    } else {
        getUserAppointment();
    }
}, [type])

const handleDownload = async (url, appointId) => {
    try {
        const res = await
axios.get('http://localhost:8001/api/doctor/getdocumentdownload', {
        headers: {
            'Authorization': `Bearer ${localStorage.getItem("token")}`,
        },
        params: { appointId },
        responseType: 'blob'
    });
        console.log(res.data)
        if (res.data) {
            const fileUrl = window.URL.createObjectURL(new Blob([res.data], {
"type": "application/pdf" }));
            const downloadLink = document.createElement("a");
            document.body.appendChild(downloadLink);
            downloadLink.setAttribute("href", fileUrl);

            // Extract the file name from the url parameter
            const fileName = url.split("/").pop(); // Assuming the URL is in
the format "uploads/document.pdf"

            console.log(fileUrl, downloadLink, fileName)
            // Set the file name for the download
            downloadLink.setAttribute("download", fileName);
            downloadLink.style.display = "none";
            downloadLink.click();
        } else {
            message.error(res.data.error);
        }
    }
};

```

```

    } catch (error) {
      console.log(error);
      message.error('Something went wrong');
    }
  };
  return (
    <div>
      <h2 className='p-3 text-center'>All Appointments</h2>
      <Container>

        {type === true ? (
          <Table striped bordered hover>
            <thead>
              <tr>
                <th>Name</th>
                <th>Date of Appointment</th>
                <th>Phone</th>
                <th>Document</th>
                <th>Status</th>
                <th>Action</th>
              </tr>
            </thead>
            <tbody>
              {doctorAppointments.length > 0 ? (
                doctorAppointments.map((appointment) => {
                  return (
                    <tr key={appointment._id}>
                      <td>{appointment.userInfo.fullName}</td>
                      <td>{appointment.date}</td>
                      <td>{appointment.userInfo.phone}</td>
                      <td><Button variant='link' onClick={() =>
handleDownload(appointment.document.path,
appointment._id)}>{appointment.document.filename}</Button></td>
                      <td>{appointment.status}</td>
                      <td>{appointment.status === 'approved' ? <></> :
<Button onClick={() => handleStatus(appointment.userInfo._id,
appointment._id, 'approved')}>Approve</Button>}
                    </tr>
                  );
                })
              ) : (
                <tr>
                  <td colspan={6}>
                    <Alert variant="info">
                      <Alert.Heading>No Appointments to
show</Alert.Heading>
                    </Alert>

```

```

        </td>
      </tr>
    )}
  </tbody>
</Table>
) : (
  <Table striped bordered hover>
    <thead>
      <tr>
        <th>Doctor Name</th>
        <th>Date of Appointment</th>
        <th>Status</th>
      </tr>
    </thead>
    <tbody>
      {userAppointments.length > 0 ? (
        userAppointments.map((appointment) => {
          return (
            <tr key={appointment._id}>
              <td>{appointment.docName}</td>
              <td>{appointment.date}</td>
              <td>{appointment.status}</td>
            </tr>
          );
        })
      ) : (
        <tr>
          <td colspan={3}>
            <Alert variant="info">
              <Alert.Heading>No Appointments to
show</Alert.Heading>
            </Alert>
          </td>
        </tr>
      )}
    </tbody>
  </Table>
)}
</Container>
</div>
);
};

export default UserAppointments;

```

DoctorList.jsx

```
import { message } from 'antd';
import axios from 'axios';
import React, { useState } from 'react'
import { Form, Row, Col } from 'react-bootstrap';
import Button from 'react-bootstrap/Button';
import Card from 'react-bootstrap/Card';
import Modal from 'react-bootstrap/Modal';

const DoctorList = ({ userDoctorId, doctor, userdata }) => {

  const [dateTime, setDateTime] = useState('');
  const [documentFile, setDocumentFile] = useState(null);
  const [show, setShow] = useState(false);

  const currentDate = new Date().toISOString().slice(0, 16);

  const handleClose = () => setShow(false);
  const handleShow = () => setShow(true);

  const handleChange = (event) => {
    setDateTime(event.target.value);
  };

  const handleDocumentChange = (event) => {
    setDocumentFile(event.target.files[0]);
  };

  console.log(doctor._id)
  const handleBook = async (e) => {
    e.preventDefault()
    try {
      const formattedDateTime = dateTime.replace('T', ' ');
      const formData = new FormData();
      formData.append('image', documentFile); // Make sure the field
name matches the one on the server side
      formData.append('date', formattedDateTime);
      formData.append('userId', userDoctorId);
      formData.append('doctorId', doctor._id);
      formData.append('userInfo', JSON.stringify(userdata));
      formData.append('doctorInfo', JSON.stringify(doctor));
```

```

    const res = await
    axios.post('http://localhost:8001/api/user/getappointment', formData, {
      headers: {
        Authorization: `Bearer ${localStorage.getItem("token")}`,
        'Content-Type': 'multipart/form-data',
      },
    });

    if (res.data.success) {
      message.success(res.data.message)
    }
    else {
      message.error(res.data.success)
    }
  } catch (error) {
    console.log(error)
  }
}
return (
  <>
  <Card style={{ width: '18rem' }}>
    <Card.Body>
      <Card.Title>Dr. {doctor.fullName}</Card.Title>
      <Card.Text>
        <p>Phone: <b>{doctor.phone}</b></p>
      </Card.Text>
      <Card.Text>
        <p>Address: <b>{doctor.address}</b></p>
      </Card.Text>
      <Card.Text>
        <p>Specialization: <b>{doctor.specialization}</b></p>
      </Card.Text>
      <Card.Text>
        <p>Experience: <b>{doctor.experience} Yrs</b></p>
      </Card.Text>
      <Card.Text>
        <p>Fees: <b>{doctor.fees}</b></p>
      </Card.Text>
      <Card.Text>
        <p>Timing: <b>{doctor.timings[0]} :
{doctor.timings[1]}</b></p>
      </Card.Text>
      <Button variant="primary" onClick={handleShow}>
        Book Now
      </Button>
      <Modal show={show} onHide={handleClose}>
        <Modal.Header closeButton>

```

[illegible]

```

        </Modal>
      </Card.Body>
    </Card>
  </>
)
}

export default DoctorList

```

ApplyDoctor.jsx

```

import { Col, Form, Input, Row, TimePicker, message } from 'antd';
import { Container } from 'react-bootstrap';
import React, { useState } from 'react';
import axios from 'axios';

function ApplyDoctor({ userId }) {
  const [doctor, setDoctor] = useState({
    fullName: '',
    email: '',
    phone: '',
    address: '',
    specialization: '',
    experience: '',
    fees: '',
    timings: '',
  });

  const handleTimingChange = (_, timings) => {
    setDoctor({ ...doctor, timings });
  };

  const handleChange = (e) => {
    setDoctor({ ...doctor, [e.target.name]: e.target.value })
  }
  const handleSubmit = async () => {

    try {
      const res = await
    axios.post('http://localhost:8001/api/user/registerdoc', { doctor, userId:
    userId }, {
      headers: {
        Authorization: `Bearer ${localStorage.getItem('token')}`

```



```

    }
  })
  if (res.data.success) {
    message.success(res.data.message)
  }
  else {
    message.error(res.data.success)
  }
} catch (error) {
  console.log(error)
  message.error('Something went wrong')
}
};

return (
  <Container>
    <h2 className='text-center p-3'>Apply for Doctor</h2>
    <Form onFinish={handleSubmit} className='m-3'>
      <h4>Personal Details:</h4>
      <Row gutter={20}>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Full Name" required>
            <Input name='fullName' value={doctor.fullName}
onChange={handleChange} placeholder='Enter name' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Phone" required>
            <Input value={doctor.phone} onChange={handleChange}
name='phone' type='number' placeholder='Your phone' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Email" required>
            <Input value={doctor.email} onChange={handleChange}
name='email' type='email' placeholder='Your email' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Address" required>
            <Input value={doctor.address} onChange={handleChange}
name='address' type='text' placeholder='Your address' />
          </Form.Item>
        </Col>
      </Row>
      <h4>Professional Details:</h4>
      <Row gutter={20}>

```

```

        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Specialization" required>
            <Input value={doctor.specialization}
onChange={handleChange} type='text' name='specialization'
placeholder='Your specialization' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Experience" required>
            <Input value={doctor.experience}
onChange={handleChange} type='number' name='experience' placeholder='Your
experience' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Fees" required>
            <Input value={doctor.fees} onChange={handleChange}
name='fees' type='number' placeholder='Your fees' />
          </Form.Item>
        </Col>
        <Col xs={24} md={12} lg={8}>
          <Form.Item label="Timings" name="timings" required>
            <TimePicker.RangePicker format="HH:mm"
onChange={handleTimingChange} />
          </Form.Item>
        </Col>
      </Row>
      <div className="d-flex justify-content-end">
        <button className="btn btn-primary"
type="submit">Submit</button>
      </div>
    </Form>
  </Container>
);
}

export default ApplyDoctor;

```

App.css

```
@import "../node_modules/bootstrap/dist/css/bootstrap.min.css";
```

```
* {  
  padding: 0;  
  margin: 0;  
  box-sizing: border-box;  
}
```

```
body {  
  background-color: rgb(190, 203, 203);  
}
```

```
a {  
  color: black;  
  text-decoration: none;  
  margin-right: 20px;  
}
```

```
.App {  
  display: flex;  
  flex-direction: column;  
  min-height: 100vh;  
}
```

```
.content {  
  flex: 1;  
}
```

```
.home-container {  
  max-width: 100vw;  
  height: 100vh;  
  display: flex;  
}
```

```
.left-side,  
.right-side {  
  flex: 1;  
  height: 100%;  
  display: flex;  
  flex-direction: column;  
  justify-content: center;  
  align-items: center;  
}
```

```
.left-side img {
```

```

    max-width: 100%;
    height: 70%;

}

.right-side {
    display: flex;
    align-items: center;
    justify-content: center;
}

span {
    font-family: Arial, Helvetica, sans-serif;
}

.f-letter {
    font-size: 3em;
}

.s-letter {
    font-size: 2.2em;
}

.t-letter {
    font-size: 1.5em;
}

/*****/
.main {
    padding: 10px;
    height: 92.1vh;
    /* background-color: aqua; */
}

.layout {
    display: flex;
}

.sidebar {
    min-height: 100%;
    height: 88vh;
    width: 17vw;
    border-radius: 5px;
    background-color: rgb(153, 145, 145);
    box-shadow: 0 0 2px grey;
    margin-right: 20px;
    color: white;
}

```

```
}

.content {
  width: 100%;
  height: 100%;
}

.header {
  height: 10vh;
  margin-bottom: 20px;
  box-shadow: 0 0 2px grey;
  background-color: whitesmoke;
  border-radius: 5px;
}

.body {
  height: 75vh;
  margin-bottom: 20px;
  box-shadow: 0 0 2px grey;
  background-color: whitesmoke;
  border-radius: 5px;
}

.logo h2 {
  text-align: center;
  margin: 20px 0px;
}

.menu {
  margin-top: 100px;
  display: flex;
  flex-direction: column;
  align-items: center;
}

.menu-items {
  margin-top: 20px;
}

.menu-items p {
  text-align: center;
  color: white;
  text-decoration: none;
  font-size: 1.3rem;
  cursor: pointer;
}
```

```
.menu-items .icons {  
  font-size: 1.2rem;  
  margin: 0 15px;  
}
```

```
.header-content {  
  display: flex;  
  align-items: center;  
  height: 70px;  
  justify-content: flex-start;  
  margin: auto 15px;  
}
```

```
.notify {  
  position: relative;  
  display: inline-block;  
  cursor: pointer;  
}
```

```
sup{  
  margin-right: 15px;  
}
```

```
.notify .icon {  
  margin-right: 10px;  
  font-size: 1.5rem;  
}
```

```
/*****/
```

```
.box {  
  display: flex;  
  flex-direction: column;  
  justify-content: center;  
  align-items: center;  
  padding: 20px;  
  border-radius: 10px;  
  box-shadow: 0 0 2px grey;  
  background-color: whitesmoke;  
  max-width: 700px;  
  width: 100%;  
}
```

```
.component {  
  display: flex;  
  flex-direction: column;  
  justify-content: center;
```

```
    align-items: center;
    margin: 20px;
    padding: 20px;
    border: 2px dashed grey;
    border-radius: 10px;
    width: 100%;
    height: 200px;
    text-align: center;
    cursor: pointer;
}

.component p {
    margin: 10px 0;
    color: grey;
}

.btn-progress {
    display: flex;
    justify-content: center;
    align-items: center;
    margin-top: 20px;
    width: 100%;
}

.btn-progress button {
    color: grey;
    padding: 10px 20px;
    margin-right: 10px;
}

.btn-progress button:hover {
    box-shadow: 0 0 10px grey;
}

progress {
    width: 100%;
    height: 10px;
    appearance: none;
    background-color: #f0f0f0;
    border-radius: 10px;
}

progress::-webkit-progress-bar {
    background-color: #f0f0f0;
    border-radius: 10px;
}
```



```
progress::-webkit-progress-value {  
  background-color: grey;  
  border-radius: 10px;  
}
```

App.js

```

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import "./App.css";
import Home from "./components/common/Home";
import Login from "./components/common/Login";
import Register from "./components/common/Register";
import UserHome from "./components/user/UserHome";
import AdminHome from "./components/admin/AdminHome";
import UserAppointments from "./components/user/UserAppointments";

function App() {
  const userLoggedIn = !!localStorage.getItem("userData");
  return (
    <div className="App">
      <Router>
        <div className="content">
          <Routes>
            <Route exact path="/" element={<Home/>} />
            <Route path="/login" element={<Login/>} />
            <Route path="/register" element={<Register/>} />
            {userLoggedIn ? (
              <>
                <Route path="/adminhome" element={<AdminHome />} />
                <Route path="/userhome" element={<UserHome />} />
                <Route path="/userhome/userappointments/:doctorId"
element={<UserAppointments />} />
              </>
            ) : (
              <Route path="/login" element={<Login />} />
            )}
          </Routes>
        </div>
        <footer className="bg-light text-center text-lg-start">
          <div className="text-center p-3">© 2023 Copyright:
MediCareBook</div>
        </footer>
      </Router>
    </div>
  );
}

export default App;

```

Index.js

```
import React from 'react';

import ReactDOM from 'react-dom/client';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Images

Photo 1



Photo 2



Photo 3



Public

Index.Html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />

    <title>Book a Doctor</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>

  </body>
</html>
```

Back End

Config

Controllers

adminC.js

```

const docSchema = require("../schemas/docModel");
const userSchema = require("../schemas/userModel");
const appointmentSchema = require("../schemas/appointmentModel");
const getAllUsersControllers = async (req, res) => {
  try {
    const users = await userSchema.find({});
    return res.status(200).send({
      message: "Users data list",
      success: true,
      data: users,
    });
  } catch (error) {
    console.log(error);
    return res.status(500).send({ message: "something went wrong",
success: false });
  }
};

const getAllDoctorsControllers = async (req, res) => {
  try {
    const docUsers = await docSchema.find({});

    return res.status(200).send({
      message: "doctor Users data list",
      success: true,
      data: docUsers,
    });
  } catch (error) {
    console
      .log(error)
      .status(500)
      .send({ message: "something went wrong", success: false });
  }
};

const getStatusApproveController = async (req, res) => {
  try {
    const { doctorId, status, userid } = req.body;
    const doctor = await docSchema.findOneAndUpdate(
      { _id: doctorId },
      { status }
    );

    const user = await userSchema.findOne({ _id: userid });

```

```

const notification = user.notification;
notification.push({
  type: "doctor-account-approved",
  message: `Your Doctor account has ${status}`,
  onClickPath: "/notification",
});

user.isdoctor = status === "approved" ? true : false;
await user.save();
await doctor.save();

return res.status(201).send({
  message: "Successfully update approve status of the doctor!",
  success: true,
  data: doctor,
});
} catch (error) {
  console.log(error);
  return res.status(500).send({ message: "something went wrong",
success: false });
}
};

const getStatusRejectController = async (req, res) => {
  try {
    const { doctorId, status, userid } = req.body;
    const doctor = await docSchema.findOneAndUpdate(
      { _id: doctorId },
      { status }
    );

    const user = await userSchema.findOne({ _id: userid });

    const notification = user.notification;
    notification.push({
      type: "doctor-account-approved",
      message: `Your Doctor account has ${status}`,
      onClickPath: "/notification",
    });

    await user.save();
    await doctor.save();

    console.log(user);
    console.log(doctor);

    return res.status(201).send({

```

```

        message: "Successfully updated Rejected status of the doctor!",
        success: true,
        data: doctor,
    });
} catch (error) {
    console.log(error);
    return res.status(500).send({ message: "something went wrong",
success: false });
}
};

const displayAllAppointmentController = async (req, res) => {
    try {
        const allAppointments = await appointmentSchema.find();
        return res.status(200).send({
            success: true,
            message: "successfully fetched All Appointments ",
            data: allAppointments,
        });

    } catch (error) {
        console.log(error);
        return res.status(500).send({ message: "something went wrong",
success: false });
    }
};

module.exports = {
    getAllDoctorsControllers,
    getAllUsersControllers,
    getStatusApproveController,
    getStatusRejectController,
    displayAllAppointmentController,
};

```

doctorC.js


```

const docSchema = require("../schemas/docModel");
const appointmentSchema = require("../schemas/appointmentModel");
const userSchema = require("../schemas/userModel");
const fs = require("fs");
const path = require('path');

const updateDoctorProfileController = async (req, res) => {
  console.log(req.body);
  try {
    const doctor = await docSchema.findOneAndUpdate(
      { userId: req.body.userId },
      req.body
    );
    await doctor.save();
    return res.status(200).send({
      success: true,
      data: doctor,
      message: "successfully updated profile",
    });
  } catch (error) {
    console.log(error);
    return res
      .status(500)
      .send({ message: "something went wrong", success: false });
  }
};

const getAllDoctorAppointmentsController = async (req, res) => {
  try {
    const doctor = await docSchema.findOne({ userId: req.body.userId });

    const allAppointments = await appointmentSchema.find({
      doctorId: doctor._id,
    });

    return res.status(200).send({
      message: "All the appointments are listed below.",
      success: true,
      data: allAppointments,
    });
  } catch (error) {
    console.log(error);
    return res
      .status(500)
      .send({ message: "something went wrong", success: false });
  }
};

```

```

const handleStatusController = async (req, res) => {
  try {
    const { userid, appointmentId, status } = req.body;

    // Make sure to add a query condition to find the specific appointment
    const appointment = await appointmentSchema.findOneAndUpdate(
      { _id: appointmentId }, // Use _id to uniquely identify the
appointment
      { status: status }, // Update the status field
      { new: true } // Set { new: true } to get the updated document as a
result
    );

    const user = await userSchema.findOne({ _id: userid });

    const notification = user.notification;

    notification.push({
      type: "status-updated",
      message: `your appointment get ${status}`,
    });

    await user.save();
    await appointment.save();

    return res.status(200).send({
      success: true,
      message: "successfully updated",
    });
  } catch (error) {
    console.log(error);
    return res
      .status(500)
      .send({ message: "something went wrong", success: false });
  }
};

const documentDownloadController = async (req, res) => {
  const appointId = req.query.appointId;
  try {
    const appointment = await appointmentSchema.findById(appointId);

    if (!appointment) {
      return res.status(404).send({ message: "Appointment not found" });
    }
  }
};

```

```

    // Assuming that the document URL is stored in the "document" field of
the appointment
    const documentUrl = appointment.document?.path; // Use optional
chaining to safely access the property

    if (!documentUrl || typeof documentUrl !== "string") {
        return res.status(404).send({ message: "Document URL is invalid",
success: false });
    }

    // Construct the absolute file path
    const absoluteFilePath = path.join(__dirname, "..", documentUrl);

    // Check if the file exists before initiating the download
    fs.access(absoluteFilePath, fs.constants.F_OK, (err) => {
        if (err) {
            return res.status(404).send({ message: "File not found", success:
false, error: err });
        }

        // Set appropriate headers for the download response
        res.setHeader("Content-Disposition", `attachment;
filename="${path.basename(absoluteFilePath)}"`);
        res.setHeader("Content-Type", "application/octet-stream");

        // Stream the file data to the response
        const fileStream = fs.createReadStream(absoluteFilePath);
        fileStream.on('error', (error) => {
            console.log(error);
            return res.status(500).send({ message: "Error reading the
document", success: false, error: error });
        });
        // Pipe the fileStream to the response
        fileStream.pipe(res);

        // Send the response after the file stream ends (file download is
completed)
        fileStream.on('end', () => {
            console.log('File download completed.');
```

```
};

module.exports = {
  updateDoctorProfileController,
  getAllDoctorAppointmentsController,
  handleStatusController,
  documentDownloadController,
};
```

userC.js

```

const bcrypt = require("bcryptjs");
const jwt = require("jsonwebtoken");
const path = require("path");
const fs = require("fs");

const userSchema = require("../schemas/userModel");
const docSchema = require("../schemas/docModel");
const appointmentSchema = require("../schemas/appointmentModel");

/// for registering the user
const registerController = async (req, res) => {
  try {
    const existsUser = await userSchema.findOne({ email: req.body.email });
  }
  if (existsUser) {
    return res
      .status(200)
      .send({ message: "User already exists", success: false });
  }
  const password = req.body.password;
  const salt = await bcrypt.genSalt(10);
  const hashedPassword = await bcrypt.hash(password, salt);
  req.body.password = hashedPassword;

  const newUser = new userSchema(req.body);
  await newUser.save();

  return res.status(201).send({ message: "Register Success", success:
true });
  } catch (error) {
    console.log(error);
    return res
      .status(500)
      .send({ success: false, message: `${error.message}` });
  }
};

```

```

/////for the login
const loginController = async (req, res) => {
  try {
    const user = await userSchema.findOne({ email: req.body.email });
    if (!user) {
      return res
        .status(200)
        .send({ message: "User not found", success: false });
    }
  }
};

```

```

    }
    const isMatch = await bcrypt.compare(req.body.password,
user.password);
    if (!isMatch) {
        return res
            .status(200)
            .send({ message: "Invalid email or password", success: false });
    }
    const token = jwt.sign({ id: user._id }, process.env.JWT_KEY, {
        expiresIn: "1d",
    });
    user.password = undefined;
    return res.status(200).send({
        message: "Login success successfully",
        success: true,
        token,
        userData: user,
    });
} catch (error) {
    console.log(error);
    return res
        .status(500)
        .send({ success: false, message: `${error.message}` });
}
};

```

```

/////auth controller
const authController = async (req, res) => {
    try {
        const user = await userSchema.findOne({ _id: req.body.userId });

        if (!user) {
            return res
                .status(200)
                .send({ message: "user not found", success: false });
        } else {
            return res.status(200).send({
                success: true,
                data: user,
            });
        }
    } catch (error) {
        console.log(error);
        return res
            .status(500)
            .send({ message: "auth error", success: false, error });
    }
}

```

```

    }
  };

  /////for the doctor registration of user
  const docController = async (req, res) => {
    try {
      const { doctor, userId } = req.body;

      const newDoctor = new docSchema({
        ...doctor,
        userId: userId.toString(),
        status: "pending",
      });
      await newDoctor.save();

      const adminUser = await userSchema.findOne({ type: "admin" });
      const notification = adminUser.notification;
      notification.push({
        type: "apply-doctor-request",
        message: `${newDoctor.fullName} has applied for doctor
registration`,
        data: {
          userId: newDoctor._id,
          fullName: newDoctor.fullName,
          onClickPath: "/admin/doctors",
        },
      });

      await userSchema.findByIdAndUpdate(adminUser._id, { notification });

      return res.status(201).send({
        success: true,
        message: "Doctor Registration request sent successfully",
      });
    } catch (error) {
      console.log(error);
      res
        .status(500)
        .send({ message: "error while applying", success: false, error });
    }
  };
};

```

```

  /////for the notification
  const getAllNotificationController = async (req, res) => {
    try {
      const user = await userSchema.findOne({ _id: req.body.userId });

```



```

    const seennotification = user.seennotification;
    const notification = user.notification;

    seennotification.push(...notification);
    user.notification = [];
    user.seennotification = seennotification;

    const updatedUser = await user.save();
    return res.status(200).send({
      success: true,
      message: "All notification marked as read",
      data: updatedUser,
    });
  } catch (error) {
    console.log(error);
    return res
      .status(500)
      .send({ message: "unable to fetch", success: false, error });
  }
};

```

```

/////for deleting the notification
const deleteallnotificationController = async (req, res) => {
  try {
    const user = await userSchema.findOne({ _id: req.body.userId });
    user.notification = [];
    user.seennotification = [];

    const updatedUser = await user.save();
    updatedUser.password = undefined;
    return res.status(200).send({
      success: true,
      message: "notification deleted",
      data: updatedUser,
    });
  } catch (error) {
    console.log(error);
    res
      .status(500)
      .send({ message: "unable to delete", success: false, error });
  }
};

```

```

/////displaying all doctors in user profile
const getAllDoctorsControllers = async (req, res) => {
  try {

```

```

const docUsers = await docSchema.find({ status: "approved" });
return res.status(200).send({
  message: "doctor Users data list",
  success: true,
  data: docUsers,
});
} catch (error) {
  console
    .log(error)
    .status(500)
    .send({ message: "something went wrong", success: false, error });
}
};

```

```

/////getting appointments done in user
const appointmentController = async (req, res) => {
  try {
    let { userInfo, doctorInfo } = req.body;
    userInfo = JSON.parse(userInfo)
    doctorInfo = JSON.parse(doctorInfo)

    let documentData = null;
    if (req.file) {
      documentData = {
        filename: req.file.filename,
        path: `/uploads/${req.file.filename}`,
      };
    }

    req.body.status = "pending";

    const newAppointment = new appointmentSchema({
      userId: req.body.userId,
      doctorId: req.body.doctorId,
      userInfo: userInfo,
      doctorInfo: doctorInfo,
      date: req.body.date,
      document: documentData,
      status: req.body.status,
    });

    await newAppointment.save();

    const user = await userSchema.findOne({ _id: doctorInfo.userId });

    if (user) {
      user.notification.push({

```

```

        type: "New Appointment",
        message: `New Appointment request from ${userInfo.fullName}`,
    });

    await user.save();
}

return res.status(200).send({
    message: "Appointment book successfully",
    success: true,
});
} catch (error) {
    console.log(error);
    res
        .status(500)
        .send({ message: "something went wrong", success: false, error });
}
};

const getAllUserAppointments = async (req, res) => {
    try {
        const allAppointments = await appointmentSchema.find({
            userId: req.body.userId,
        });

        const doctorIds = allAppointments.map(
            (appointment) => appointment.doctorId
        );

        const doctors = await docSchema.find({
            _id: { $in: doctorIds },
        });

        const appointmentsWithDoctor = allAppointments.map((appointment) => {
            const doctor = doctors.find(
                (doc) => doc._id.toString() === appointment.doctorId.toString()
            );
            const docName = doctor ? doctor.fullName : "";
            return {
                ...appointment.toObject(),
                docName,
            };
        });

        return res.status(200).send({
            message: "All the appointments are listed below.",
            success: true,
            data: appointmentsWithDoctor,
        });
    }
};

```

```

    });
  } catch (error) {
    console.log(error);
    res
      .status(500)
      .send({ message: "something went wrong", success: false, error });
  }
};

```

```

const getDocsController = async (req, res) => {
  try {
    const user = await userSchema.findOne({ _id: req.body.userId });
    const allDocs = user.documents;
    if (!allDocs) {
      return res.status(201).send({
        message: "No documnets",
        success: true,
      });
    }
    return res.status(200).send({
      message: "All the appointments are listed below.",
      success: true,
      data: allDocs,
    });
  } catch (error) {
    console.log(error);
    res
      .status(500)
      .send({ message: "something went wrong", success: false, error });
  }
};

```

```

module.exports = {
  registerController,
  loginController,
  authController,
  docController,
  getallnotificationController,
  deleteallnotificationController,
  getAllDoctorsControllers,
  appointmentController,
  getAllUserAppointments,
  getDocsController,
};

```

Middleware

authMiddleware.js

```

const jwt = require("jsonwebtoken");

module.exports = async (req, res, next) => {
  try {
    const authorizationHeader = req.headers["authorization"];
    if (!authorizationHeader) {
      return res
        .status(401)
        .send({ message: "Authorization header missing", success: false
});
    }

    const token = req.headers["authorization"].split(" ")[1];
    jwt.verify(token, process.env.JWT_KEY, (err, decode) => {
      if (err) {
        return res
          .status(200)
          .send({ message: "Token is not valid", success: false });
      } else {
        req.body.userId = decode.id;
        next();
      }
    });
  } catch (error) {
    console.error(error); // Handle or log the error appropriately
    res.status(500).send({ message: "Internal server error", success:
false });
  }
};

```


Routes

userRoutes.js

```
const multer = require("multer");
const express = require("express");

const {
  registerController,
  loginController,
  authController,
  docController,
  deleteallnotificationController,
  getallnotificationController,
  getAllDoctorsControllers,
  appointmentController,
  getAllUserAppointments,
  getDocsController,
  downloadDocController,
} = require("../controllers/userC");
const authMiddleware = require("../middlewares/authMiddleware");

const router = express.Router();

const upload = multer({ dest: 'uploads/' })

router.post("/register", registerController);

router.post("/login", loginController);

router.post("/getuserdata", authMiddleware, authController);

router.post("/registerdoc", authMiddleware, docController);

router.get("/getalldoctorsu", authMiddleware, getAllDoctorsControllers);

router.post("/getappointment",upload.single("image"), authMiddleware,
appointmentController);

router.post(
  "/getallnotification",
  authMiddleware,
  getallnotificationController
);

router.post(
  "/deleteallnotification",
  authMiddleware,
  deleteallnotificationController
);
```

```
router.get("/getuserappointments", authMiddleware,  
getAllUserAppointments);  
  
router.get("/getDocsforuser", authMiddleware, getDocsController)  
  
module.exports = router;
```

doctorRoutes.js

```
const express = require("express");
const multer = require("multer");
const authMiddleware = require("../middlewares/authMiddleware");
const {
  updateDoctorProfileController,
  getAllDoctorAppointmentsController,
  handleStatusController,
  documentDownloadController,
} = require("../controllers/doctorC");

const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, "./uploads/");
  },
  filename: function (req, file, cb) {
    const uniqueSuffix = Date.now();
    cb(null, uniqueSuffix + "-" + file.originalname);
  },
});

const upload = multer({ storage: storage });

const router = express.Router();

router.post("/updateprofile", authMiddleware,
updateDoctorProfileController);

router.get(
  "/getdoctorappointments",
  authMiddleware,
  getAllDoctorAppointmentsController
);

router.post("/handlestatus", authMiddleware, handleStatusController);

router.get(
  "/getdocumentdownload",
  authMiddleware,
  documentDownloadController
);

module.exports = router;
```

adminRoutes.js


```
const multer = require("multer");
const express = require("express");

const {
  registerController,
  loginController,
  authController,
  docController,
  deleteallnotificationController,
  getallnotificationController,
  getAllDoctorsControllers,
  appointmentController,
  getAllUserAppointments,
  getDocsController,
  downloadDocController,
} = require("../controllers/userC");
const authMiddleware = require("../middlewares/authMiddleware");

const router = express.Router();

const upload = multer({ dest: 'uploads/' })

router.post("/register", registerController);

router.post("/login", loginController);

router.post("/getuserdata", authMiddleware, authController);

router.post("/registerdoc", authMiddleware, docController);

router.get("/getalldoctorsu", authMiddleware, getAllDoctorsControllers);

router.post("/getappointment",upload.single("image"), authMiddleware,
appointmentController);

router.post(
  "/getallnotification",
  authMiddleware,
  getallnotificationController
);

router.post(
  "/deleteallnotification",
  authMiddleware,
  deleteallnotificationController
);
```

```
router.get("/getuserappointments", authMiddleware,  
getAllUserAppointments);  
  
router.get("/getDocsforuser", authMiddleware, getDocsController)  
  
module.exports = router;
```

Schemas

userModel.js

```

const mongoose = require("mongoose");

const userModel = mongoose.Schema({
  fullName: {
    type: String,
    required: [true, "Name is required"],
    set: function (value) {
      return value.charAt(0).toUpperCase() + value.slice(1);
    },
  },
  email: {
    type: String,
    required: [true, "email is required"],
  },
  password: {
    type: String,
    required: [true, "password is required"],
  },
  phone: {
    type: String,
    required: [true, "phone is required"],
  },
  type: {
    type: String,
    required: [true, "type is required"],
  },
  notification: {
    type: Array,
    default: [],
  },
  seennotification: {
    type: Array,
    default: [],
  },
  isdoctor: {
    type: Boolean,
    default: false,
  }
});

const userSchema = mongoose.model("user", userModel);

module.exports = userSchema;

```

docModel.js

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

const docModel = mongoose.Schema(
  {
    userId: {
      type: mongoose.Schema.Types.ObjectId,
      ref: 'users'
    },
    fullName: {
      type: String,
      required: [true, "full Name is required"],
      set: function (value) {
        return value.charAt(0).toUpperCase() + value.slice(1);
      },
    },
    email: {
      type: String,
      required: [true, "email is required"],
    },
    phone: {
      type: String,
      required: [true, "phone is required"],
    },
    address: {
      type: String,
      require: [true, "address required"],
    },
    specialization: {
      type: String,
      required: [true, "specialization is required"],
    },
    experience: {
      type: String,
      required: [true, "experience is required"],
    },
    fees: {
      type: Number,
      required: [true, "fees is required"],
    },
    status: {
      type: String,
      default: 'pending'
    },
    timings: {
      type: Object,
      required: [true, "work time required"],
    },
  },
```



```
    },  
    {  
      timestamps: true,  
    }  
  );  
  
  const docSchema = mongoose.model("doctor", docModel);  
  
  module.exports = docSchema;
```

appointmentModel.js

```

const mongoose = require("mongoose");

const appointmentModel = mongoose.Schema(
  {
    userId: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "users",
      required: true,
    },
    doctorId: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "doctor",
      required: true,
    },
    userInfo: {
      type: Object,
      default: {},
      required: true,
    },
    doctorInfo: {
      type: Object,
      default: {},
      required: true,
    },
    date: {
      type: String,
      required: true,
    },
    document: {
      type: Object,
    },
    status: {
      type: String,
      require: true,
      default: "pending",
    },
  },
  {
    timestamps: true,
  }
);

const appointmentSchema = mongoose.model("appointment", appointmentModel);

module.exports = appointmentSchema;

```

Picture



gitignore

```
# See https://help.github.com/articles/ignoring-files/ for more about  
ignoring files.
```

```
# dependencies  
/node_modules  
/.pnp  
.pnp.js
```

```
# testing  
/coverage
```

```
# production  
/build
```

```
# misc  
.DS_Store  
.env.local  
.env.development.local  
.env.test.local  
.env.production.local
```

```
npm-debug.log*  
yarn-debug.log*  
yarn-error.log*
```

package-lock.json

ReadMe.Md

Getting Started with Create React App

This project was bootstrapped with [Create React App] (<https://github.com/facebook/create-react-app>).

Available Scripts

In the project directory, you can run:

`npm start`

Runs the app in the development mode.\nOpen [<http://localhost:3000>] (<http://localhost:3000>) to view it in your browser.

The page will reload when you make changes.\nYou may also see any lint errors in the console.

`npm test`

Launches the test runner in the interactive watch mode.\nSee the section about [running tests] (<https://facebook.github.io/create-react-app/docs/running-tests>) for more information.

`npm run build`

Builds the app for production to the `build` folder.\nIt correctly bundles React in production mode and optimizes the build for the best performance.

The build is minified and the filenames include the hashes.\nYour app is ready to be deployed!

See the section about [deployment] (<https://facebook.github.io/create-react-app/docs/deployment>) for more information.

`npm run eject`

****Note:** this is a one-way operation. Once you `eject`, you can't go back!**

If you aren't satisfied with the build tool and configuration choices, you can `eject` at any time. This command will remove the single build dependency from your project.

Instead, it will copy all the configuration files and the transitive dependencies (webpack, Babel, ESLint, etc) right into your project so you have full control over them. All of the commands except `eject` will still work, but they will point to the copied scripts so you can tweak them. At this point you're on your own.

You don't have to ever use `eject`. The curated feature set is suitable for small and middle deployments, and you shouldn't feel obligated to use this feature. However we understand that this tool wouldn't be useful if you couldn't customize it when you are ready for it.

Learn More

You can learn more in the [Create React App documentation](<https://facebook.github.io/create-react-app/docs/getting-started>).

To learn React, check out the [React documentation](<https://reactjs.org/>).

Code Splitting

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/code-splitting>] (<https://facebook.github.io/create-react-app/docs/code-splitting>)

Analyzing the Bundle Size

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/analyzing-the-bundle-size>] (<https://facebook.github.io/create-react-app/docs/analyzing-the-bundle-size>)

Making a Progressive Web App

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/making-a-progressive-web-app>] (<https://facebook.github.io/create-react-app/docs/making-a-progressive-web-app>)

Advanced Configuration

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/advanced-configuration>] (<https://facebook.github.io/create-react-app/docs/advanced-configuration>)

Deployment

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/deployment>] (<https://facebook.github.io/create-react-app/docs/deployment>)

`npm run build` fails to minify

This section has moved here:

[<https://facebook.github.io/create-react-app/docs/troubleshooting#npm-run-build-fails-to-minify>] (<https://facebook.github.io/create-react-app/docs/troubleshooting#npm-run-build-fails-to-minify>)