

```
1 MIT License
2
3 Copyright (c) 2021 Nirav Koley
4
5 Permission is hereby granted, free of charge, to any person obtaining
6 a copy
7 of this software and associated documentation files (the "Software"),
8 to deal
9 in the Software without restriction, including without limitation the
10 rights
11 to use, copy, modify, merge, publish, distribute, sublicense, and/or
12 sell
13 copies of the Software, and to permit persons to whom the Software is
14 furnished to do so, subject to the following conditions:
15
16 The above copyright notice and this permission notice shall be
17 included in all
18 copies or substantial portions of the Software.
19
20 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
21 EXPRESS OR
22 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
23 MERCHANTABILITY,
24 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT
25 SHALL THE
26 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
27 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
28 ARISING FROM,
29 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
30 IN THE
31 SOFTWARE.
32
```

```
1 # Logs
2 logs
3 *.log
4 npm-debug.log*
5 yarn-debug.log*
6 yarn-error.log*
7
8 # Runtime data
9 pids
10 *.pid
11 *.seed
12 *.pid.lock
13
14 # Directory for instrumented libs generated by jscoverage/JSCover
15 lib-cov
16
17 # Coverage directory used by tools like istanbul
18 coverage
19
20 # nyc test coverage
21 .nyc_output
22
23 # Grunt intermediate storage (https://gruntjs.com/creating-plugins#storing-task-files)
24 .grunt
25
26 # Bower dependency directory (https://bower.io/)
27 bower_components
28
29 # node-waf configuration
30 .lock-wscript
31
32 # Compiled binary addons (https://nodejs.org/api/addons.html)
33 build/Release
34
35 # Dependency directories
36 node_modules/
37 jspm_packages/
38
39 # TypeScript v1 declaration files
40 typings/
41
42 # Optional npm cache directory
43 .npm
44
45 # Optional eslint cache
46 .eslintcache
47
48 # Optional REPL history
49 .node_repl_history
50
51 # Output of 'npm pack'
52 *.tgz
53
54 # Yarn Integrity file
55 .yarn-integrity
56
```

```
57 # dotenv environment variables file
58 .env
59
60 # parcel-bundler cache (https://parceljs.org/)
61 .cache
62
63 # next.js build output
64 .next
65
66 # nuxt.js build output
67 .nuxt
68
69 # vuepress build output
70 .vuepress/dist
71
72 # Serverless directories
73 .serverless
74
75 # FuseBox cache
76 .fusebox/
77
```

```
1 # Auto detect text files and perform LF normalization
2 * text=auto
3
```

```
1 # Compose the client and server containers into a single docker
  instance
2 version: '3'
3
4 services:
5   backend:
6     env_file:
7       "./server/.env"
8     build:
9       context: ./server
10      dockerfile: ./Dockerfile
11      ports:
12        - "5000:5000"
13   frontend:
14     build:
15       context: ./client
16       dockerfile: ./Dockerfile
17     ports:
18       - "3000:3000"
19     links:
20       - "localhost:server"
```

```
1 # Getting Started with Create React App
2
3 This project was bootstrapped with [Create React App](https://github.
4 com/facebook/create-react-app).
5
6 ## Available Scripts
7
8 In the project directory, you can run:
9
10 #### `npm start`
11 Runs the app in the development mode.\n
12 Open [http://localhost:3000](http://localhost:3000) to view it in the
13 browser.
14 The page will reload if you make edits.\n
15 You will also see any lint errors in the console.
16
17 #### `npm test`
18
19 Launches the test runner in the interactive watch mode.\n
20 See the section about [running tests](https://facebook.github.io/
21 create-react-app/docs/running-tests) for more information.
22 #### `npm run build`
23
24 Builds the app for production to the `build` folder.\n
25 It correctly bundles React in production mode and optimizes the build
26 for the best performance.
27 The build is minified and the filenames include the hashes.\n
28 Your app is ready to be deployed!
29
30 See the section about [deployment](https://facebook.github.io/create-
31 react-app/docs/deployment) for more information.
32 #### `npm run eject`
33
34 **Note: this is a one-way operation. Once you `eject`, you can't go
35 back!**
36 If you aren't satisfied with the build tool and configuration choices
37 , you can `eject` at any time. This command will remove the single
38 build dependency from your project.
39 Instead, it will copy all the configuration files and the transitive
40 dependencies (webpack, Babel, ESLint, etc) right into your project so
41 you have full control over them. All of the commands except `eject`
42 will still work, but they will point to the copied scripts so you can
43 tweak them. At this point you're on your own.
44 You don't have to ever use `eject`. The curated feature set is
45 suitable for small and middle deployments, and you shouldn't feel
46 obligated to use this feature. However we understand that this tool
47 wouldn't be useful if you couldn't customize it when you are ready for
48 it.
49
```

```
42 ## Learn More
43
44 You can learn more in the [Create React App documentation](https://facebook.github.io/create-react-app/docs/getting-started).
45
46 To learn React, check out the [React documentation](https://reactjs.org/).
47
48 ### Code Splitting
49
50 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)
51
52 ### Analyzing the Bundle Size
53
54 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)
55
56 ### Making a Progressive Web App
57
58 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)
59
60 ### Advanced Configuration
61
62 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)
63
64 ### Deployment
65
66 This section has moved here: [https://facebook.github.io/create-react-app/docs/deployment](https://facebook.github.io/create-react-app/docs/deployment)
67
68 ### `npm run build` fails to minify
69
70 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)
71
```

```
1 # Docker file for deploying front end
2
3 FROM node:14.17.4
4
5 WORKDIR /app
6
7 COPY package*.json ./
8
9 RUN npm i
10
11 COPY . .
12
13 EXPOSE 3000
14
15 CMD ["npm", "start"]
16
```

```
1 {
2   "name": "react-fe-counsweb",
3   "version": "0.1.0",
4   "private": true,
5   "dependencies": {
6     "@testing-library/jest-dom": "^5.14.1",
7     "@testing-library/react": "^11.2.7",
8     "@testing-library/user-event": "^12.8.3",
9     "axios": "^0.21.4",
10    "bootstrap": "^4.6.0",
11    "dayjs": "^1.10.6",
12    "react": "^17.0.2",
13    "react-bootstrap": "^1.6.1",
14    "react-calendar": "^3.4.0",
15    "react-dom": "^17.0.2",
16    "react-icons": "^4.2.0",
17    "react-router-dom": "^5.2.0",
18    "react-scripts": "4.0.3",
19    "web-vitals": "^1.1.2"
20  },
21  "scripts": {
22    "start": "react-scripts start",
23    "build": "react-scripts build",
24    "test": "react-scripts test",
25    "eject": "react-scripts eject"
26  },
27  "eslintConfig": {
28    "extends": [
29      "react-app",
30      "react-app/jest"
31    ]
32  },
33  "browserslist": {
34    "production": [
35      ">0.2%",
36      "not dead",
37      "not op_mini all"
38    ],
39    "development": [
40      "last 1 chrome version",
41      "last 1 firefox version",
42      "last 1 safari version"
43    ]
44  }
45 }
46
```

```
1 node_modules
2
```

```
1 // Client side-routing for serving web pages by rendering components
2
3 import React, { useState } from 'react';
4 import 'bootstrap/dist/css/bootstrap.min.css';
5 import FormPage from './components/FormPage/FormPage';
6 import EditBookings from './components/AdminPage/bookings/EditBookings';
7 import EditCalendar from './components/AdminPage/calendar/EditCalendar';
8 import EditEmail from './components/AdminPage/emails/EditEmail';
9 import Support from './components/Support';
10 import { BrowserRouter, Route, Switch } from 'react-router-dom';
11 import Sidebar2 from './components/Sidebar2';
12 import PrivateRoute from './PrivateRoute';
13
14 function App() {
15     const [admin, setAdmin] = useState(false)
16     return (
17         <>
18             <BrowserRouter>
19                 <Sidebar2 admin={admin} setAdmin={setAdmin}/>
20                 <Switch>
21                     <Route path='/' exact component={FormPage} />
22                     <Route path='/support' exact component={Support}>
23                 />
24                     {/*Private routes are blocked until admin access
25                     is granted*/}
26                     <PrivateRoute admin={admin} path='/admin/booking'
27                         exact component={EditBookings} />
28                     <PrivateRoute admin={admin} path='/admin/calendar'
29                         exact component={EditCalendar} />
30                     <PrivateRoute admin={admin} path='/admin/emails'
31                         exact component={EditEmail} />
32                 </Switch>
33             </BrowserRouter>
34     );
35 }
36
37 export default App;
38
39
```

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './App';
5
6 ReactDOM.render(
7     <React.StrictMode>
8         <App />
9     </React.StrictMode>,
10    document.getElementById('root')
11 );
12
```

```
1 .formAlign {
2     padding: 2em 1em 2em 1em;
3     width: 40rem;
4     max-width: 100%;
5     margin: auto;
6 }
7
8 .calendarAlign {
9     padding: 1em 1em 3em 1em;
10    width: 38rem;
11    max-width: 100%;
12    margin: auto;
13 }
14
15 .timeBox {
16     text-align: center;
17     position: relative;
18     width: 38rem;
19     max-width: 100%;
20     padding: 1rem;
21     margin: auto;
22
23     border: 0.3rem solid #ececfc;
24     border-radius: 8px;
25     color: #212529;
26 }
27
28 .submitAlign {
29     text-align: center;
30     position: relative;
31     width: 38rem;
32     max-width: 100%;
33     padding: 1rem;
34     margin: auto;
35 }
36
37 .timeTitle {
38     /* text-decoration: underline; */
39     font-size: 1.5em;
40     font-weight: normal;
41     text-decoration-thickness: 0.1rem;
42 }
43
44 .localTime {
45     padding-top: 0.5rem;
46 }
47
48 .localTimeTitle {
49     text-decoration: underline;
50     font-size: 1.3em;
51     font-weight: lighter;
52     text-decoration-thickness: 0.1rem;
53 }
54
55 .ourTime {
56     padding-top: 0.5rem;
57 }
```

```
58
59 .ourTimeTitle {
60     text-decoration: underline;
61     font-size: 1.3em;
62     font-weight: lighter;
63     text-decoration-thickness: 0.1rem;
64 }
65
66 .alert-success {
67     color: #0f5132;
68     background-color: #d1e7dd;
69     border-color: #badbcc;
70     position: relative;
71     padding: 1rem;
72     margin: 2.5rem;
73     border: 1px solid transparent;
74     border-radius: 0.25rem;
75     font-size: 16px;
76     font-weight: normal;
77     text-align: center;
78     visibility: visible !important;
79 }
80
81 .hidden {
82     visibility: hidden;
83 }
84
85 .button {
86     padding: 2em 1em 2.5em 1em;
87     width: 30rem;
88     height: 4rem;
89     max-width: 100%;
90     margin: auto;
91 }
92
93 .center {
94     align-items: center;
95 }
96
97 .disableCalendar {
98     cursor: default !important;
99 }
100
101 .container {
102     overflow: hidden;
103 }
104
105 .unavailable {
106     color: #000 !important;
107     background-color: rgba(242, 242, 240, 0.45) !important;
108 }
109
110 .past {
111     color: #000 !important;
112     background-color: rgba(232, 232, 225, 0.6) !important;
113 }
114
```

```
115 .booked:hover {  
116     color: hsla(3, 100%, 20%, 1) !important;  
117     background-color: rgba(255, 65, 54, 0.55) !important;  
118 }  
119  
120 .booked {  
121     color: hsla(3, 100%, 20%, 1) !important;  
122     background-color: rgba(255, 65, 54, 0.6) !important;  
123 }  
124  
125 .regionBooked {  
126     color: hsla(350, 65%, 15%) !important;  
127     background-color: rgba(194, 43, 65, 0.6) !important;  
128 }  
129  
130 .regionBooked:hover {  
131     color: hsla(350, 65%, 15%) !important;  
132     background-color: rgba(194, 43, 65, 0.55) !important;  
133 }  
134  
135 .available {  
136     color: hsla(127, 63%, 15%, 1) !important;  
137     background-color: rgba(1, 255, 112, 0.7) !important;  
138 }  
139  
140 .available:hover {  
141     color: hsla(127, 63%, 15%, 1) !important;  
142     background-color: rgba(1, 255, 112, 0.5) !important;  
143 }  
144  
145 .selected {  
146     color: #111111 !important;  
147     background-color: rgba(13, 110, 253 0.85) !important;  
148 }  
149  
150 .selected:hover {  
151     color: #111111 !important;  
152     background-color: rgba(13, 110, 253 0.8) !important;  
153 }  
154  
155 .invalidCalendar {  
156     transition: border-color 0.15s ease-in-out, box-shadow 0.15s ease  
     -in-out;  
157     border-radius: 0.25rem;  
158     border: 1.5px solid #dc3545;  
159     padding: 0.5em;  
160 }  
161  
162 .invalidText {  
163     width: 100%;  
164     margin-top: 0.3rem;  
165     margin-left: 0;  
166     margin-right: 0;  
167     font-size: 0.875em;  
168     color: #dc3545;  
169     display: block;  
170     text-align: center;
```

```
171 }
172
173 .invalidText:hover {
174     cursor: auto;
175 }
176
177 .support {
178     text-align: center;
179     position: relative;
180     width: 38rem;
181     max-width: 100%;
182     padding: 15vh 1rem 1rem 1rem;
183     margin: auto;
184 }
```

```
1 // Private route component
2
3 import {Route, Redirect} from 'react-router-dom'
4
5 const PrivateRoute = ({component, path, admin}) => {
6     if(admin){
7         return(<Route path={path} exact component={component}/>)
8     } else {
9         return(<Redirect to='/' />)
10    }
11 }
12
13 export default PrivateRoute
14
```

```

1 import { useState } from 'react'
2 import Form from 'react-bootstrap/Form'
3 import Modal from 'react-bootstrap/Modal'
4 import Button from 'react-bootstrap/Button'
5 import axios from 'axios';
6
7 const Login = ({show, cancel, setShow, error, setError, setAdmin}) => {
8     const [loginfo, setLoginfo] = useState({email:'',password:''})
9
10
11     async function handleSubmit(){
12         await axios.post('http://localhost:5000/login/login', loginfo)
13         .then((res)=>{
14             if(res.data === "FALSE"){
15                 setError(true)
16
17             } else{
18                 setAdmin(true)
19                 localStorage.setItem('adminToken', res.data.adminToken
20             ) // Store admin token in local storage
21                 setShow(false)
22             }
23         })
24         .catch(err =>{
25             console.log(err)
26         });
27     }
28
29     return (
30         <Modal show={show} onHide={cancel}>
31             <Modal.Header closeButton>
32                 <Modal.Title>Login</Modal.Title>
33             </Modal.Header>
34             <Modal.Body>
35                 <Form>
36                     <Form.Group>
37                         <Form.Label> Email </Form.Label>
38                         <Form.Control type='text' onChange={(e) => {
39                             setLoginfo({...loginfo,email:e.target.value})}}/>
40                     </Form.Group>
41                     <Form.Group>
42                         <Form.Label> Password </Form.Label>
43                         <Form.Control type='password' onChange={(e) => {
44                             setLoginfo({...loginfo,password:e.target.value})}}/>
45                     </Form.Group>
46                     {error?
47                         <div className="incorrect">
48                             Incorrect Password/Email
49                         </div>
50                     :<div></div>
51                 }
52             </Form>
53             <Modal.Footer>
54                 <Button onClick={()=>{handleSubmit()}}>Submit</Button>
55             </Modal.Footer>
56         </Modal>
57     )
58 }

```

```
55      )
56  }
57
58 export default Login
59
```

```
1 import "../index.css"
2
3 const Support = () => {
4     return (
5         <div className="support">
6             If you have any questions please call or text:
7             <h6>+1 713 290 9025</h6>
8             <br />
9             or email:
10            <h6>Paula.Cooper@houston.nae.school </h6>
11        </div>
12    )
13 }
14
15 export default Support
16
```

```

1 // Sidebar for client side routing
2
3 import React, { useEffect, useState } from 'react';
4 import { BsPencilSquare } from 'react-icons/bs';
5 import { FaCalendarPlus, FaCalendarAlt, } from 'react-icons/fa'
6 import { AiFillLock, AiFillUnlock, AiOutlineQuestionCircle } from 'react-icons/ai';
7 import { MdEmail } from 'react-icons/md'
8 import { Link, NavLink } from 'react-router-dom';
9 import './sidebar2.css';
10 import Navbar from 'react-bootstrap/Navbar'
11 import {Nav, NavDropdown} from 'react-bootstrap';
12 import Login from './Login';
13 import axios from 'axios';
14
15 const Sidebar2 = ({admin, setAdmin}) => {
16     const [sidebar, setSidebar] = useState(false); // Toggle sidebar visibility
17     const toggleSidebar = () => setSidebar(!sidebar); // Separate function as bugfix
18
19     const [show, setShow] = useState(false);
20     const [error, setError] = useState(false);
21
22     useEffect(async () => {
23         const token = localStorage.getItem('adminToken')
24         if(!token){
25             const verify = await axios.post('http://localhost:5000/login/verify', {adminToken:token})
26             if(verify){
27                 setAdmin(true)
28             }
29         }
30     }, [])
31
32     function isAdmin(){
33         if(admin){
34             return(
35                 <NavDropdown title={<span><AiFillUnlock size={22}>/>
36 Admin</span>} className="nav-text">
37                     <NavDropdown.Item href="#edit">
38                         <Nav.Link as={NavLink} to='/admin/booking'
39 className='nopad'>
40                             <span><FaCalendarPlus size={20}>/> Bookings
41                         </Nav.Link>
42                     </NavDropdown.Item>
43                     <NavDropdown.Item href="#dates">
44                         <Nav.Link as={NavLink} to='/admin/calendar'
45 className='nopad'>
46                             <span><FaCalendarAlt size={20}>/> Calendar
47                         </Nav.Link>
48                     </NavDropdown.Item>
49                     <NavDropdown.Item href="#email">
50                         <span><AiOutlineQuestionCircle size={20}>/> Email
51                     </NavDropdown.Item>
52                 </NavDropdown>
53             )
54         }
55     }
56
57     return(
58         <div>
59             <Navbar>
60                 <Nav>
61                     <Nav.Link as={NavLink} to='/admin/booking'>
62                         <span><FaCalendarPlus size={20}>/> Bookings
63                     </Nav.Link>
64                 </Nav>
65             </Navbar>
66             {show ? <div>Show</div> : <div>Hide</div>}
67             {error ? <div>Error</div> : <div>Success</div>}
68         </div>
69     )
70 }

```

```

49          <Nav.Link as={NavLink} to='/admin/emails'
50          className='nopad'>
51          <span><MdEmail size={20}/> Emails</span>
52          </Nav.Link>
53          </NavDropdown.Item>
54        )
55      } else {
56        return(
57          <Nav.Link onClick={()=>{setShow(true);setError(false)
58}}>
59          <span><AiFillLock size={22}/> Admin</span>
60          </Nav.Link>
61        )
62      }
63
64    function cancel(){
65      setShow(false)
66    }
67
68    return (
69      <>
70      <div className='wCube'></div>
71      <Login show={show} cancel={cancel} setShow={setShow} error={error} setError={setError} setAdmin={setAdmin}/>
72      <div>
73        <Navbar bg="light">
74          <Navbar.Brand className='bars' as={Link} onClick
75            ={()=>{toggleSidebar()}}>
76            <div className='burgerpad'>
77              <div className={!sidebar?'burger':'burger-
78                close'}></div>
79            </div>
80          </Navbar.Brand>
81          <Nav className={sidebar?"me-auto sidebar-active":
82            "me-auto sidebar-inactive"}>
83            <Nav.Link as={NavLink} to='/' className="nav-
84              text">
85              <span className='nav-icon'><
86                BsPencilSquare size={22}/> Form</span>
87              </Nav.Link>
88              {isAdmin()}
89              <Nav.Link as={NavLink} to='/support'
90                className="nav-text">
91                <span><AiOutlineQuestionCircle size={22
92 }/>> Help</span>
93              </Nav.Link>
94            </Nav>
95          </Navbar>
96        </div>
97      </>
98    )
99  }
100
101  export default Sidebar2
102

```

```
1 .nopad{  
2     padding: 0 0 0 0 !important;  
3 }  
4  
5 .wCube{  
6     position: absolute;  
7     top:0;  
8     left:0;  
9     height:64px;  
10    width:64px;  
11    background-color: #f8f9fa;  
12    z-index: 1;  
13 }  
14  
15 .bars{  
16     z-index: 2;  
17     margin-right: 0;  
18 }  
19  
20 .sidebar-active{  
21     left: 0rem;  
22     position: relative;  
23     transition: 500ms;  
24 }  
25  
26 .sidebar-inactive{  
27     left: -20rem;  
28     position: absolute;  
29     transition: 500ms;  
30 }  
31  
32 .burger{  
33     width:40px;  
34     height: 6px;  
35     background: rgba(0,0,0,0.8);  
36     border-radius: 3px;  
37     box-shadow: 0 2px 2px rgba(255,101,47,.2);  
38     transition: all .5s ease-in-out;  
39 }  
40  
41 .burger::before,.burger::after{  
42     content:'';  
43     position: absolute;  
44     width:40px;  
45     height: 6px;  
46     background: rgba(0,0,0,0.8);  
47     border-radius: 3px;  
48     box-shadow: 0 2px 2px rgba(255,101,47,.2);  
49     transition: all .5s ease-in-out;  
50 }  
51  
52 .burger::before {  
53     transform: translateY(-12px);  
54 }  
55  
56 .burger::after {  
57     transform: translateY(12px);
```

```
58 }
59
60 .burger-close::before,.burger-close::after{
61     content:'';
62     position: absolute;
63     width: 40px;
64     height: 6px;
65     background: rgba(0,0,0,0.8);
66     border-radius: 3px;
67     box-shadow: 0 2px 2px rgba(255,101,47,.2);
68     transition: all .5s ease-in-out;
69 }
70
71 .burger-close{
72     transform: translateX(-50px);
73     background: transparent;
74     box-shadow: none;
75     width: 40px;
76     height: 6px;
77     border-radius: 3px;
78     transition: all .5s ease-in-out;
79 }
80
81 .burger-close::before{
82     transform: translateY(-12px);
83     transform: rotate(45deg) translate(35px, -35px);
84 }
85 .burger-close::after{
86     transform: translateY(12px);
87     transform: rotate(-45deg) translate(35px, 35px);
88 }
89
90 .burgerpad{
91     margin: 1rem;
92     margin-left: 0.5rem !important;
93 }
94
95 .nav-text{
96     font-size: 1.3rem;
97     margin-right: 1rem;
98     margin-left: 0.35rem;
99 }
100
101 .nav-div{
102     display: flex;
103     justify-content: center;
104 }
105
106 .incorrect{
107     color: #dc3545;
108     font-size: 80%;
109 }
110
```

```

1 // Main form page, a sort of hub for many components to mount to and
  render from
2
3 import React, { useState, useEffect } from 'react';
4 import 'bootstrap/dist/css/bootstrap.min.css';
5 import Button from 'react-bootstrap/Button';
6 import TextForm from './TextForm';
7 import TimeDisplay from './TimeDisplay';
8 import DateCalendar from './DateCalendar';
9 import axios from 'axios';
10 import dayjs from 'dayjs';
11 import isBetween from 'dayjs/plugin/isBetween';
12 import validateInfo from './validateForm';
13
14 dayjs.extend(isBetween);
15
16 // Initialize calendar states
17 let available = [];
18 let booked = [];
19 let regionBooked = [];
20
21 function FormPage() {
22   let calendarData;
23
24   const [checkDate, setCheckDate] = useState(false); // Used to
  manually re-render after validation
25   const [key, setKey] = useState(false); // Used to manually re-
  render after validation
26
27   const [validated, setValidated] = useState(false); // Used to
  prevent validation tooltips from appearing before submission
28   const [error, setError] = useState({}); // Array of validation
  errors
29   const [success, setSuccess] = useState(false);
30   const [dateData, setDateData] = useState(new Date()); // Store
  appointment date
31   const [formData, setFormData] = useState({ // Store booking
  information
32     uniName: '',
33     uniRepName: '',
34     uniRepJobTitle: '',
35     uniRepEmail: '',
36     uniRegion: '',
37   });
38
39
40   // Submission handler
41   const handleSubmit = (e) => {
42     e.preventDefault();
43
44     const bookingDateInfoInstance = {
45       aptDate: dateData,
46       status: 'Booked',
47       booking: {
48         uniName: formData.uniName,
49         uniRepName: formData.uniRepName,
50         uniRepJobTitle: formData.uniRepJobTitle,

```



```

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
        // Upper bound of the week
        let uppB = dayjs(instance.aptDate)
            .subtract(7, 'day')
            .format('DD/MM/YYYY');

        if (index - 1 >= 0) {
            // Checks last available day
            if (calendarData[index - 1].status === 'Available' && dayjs(calendarData[index - 1])
                .isBetween(lowB, uppB, null, '[]')) {
                regionBooked.push(dayjs(calendarData[index - 1].aptDate).format('DD/MM/YYYY'));
                calendarData[index - 1].status = 'Region booked'; // Prevent further formatting
            }
        }

        if (index + 1 <= calendarData.length - 1) {
            // Checks next available day
            if (calendarData[index + 1].status === 'Available' && dayjs(calendarData[index + 1])
                .isBetween(lowB, uppB, null, '[]')) {
                regionBooked.push(dayjs(calendarData[index + 1].aptDate).format('DD/MM/YYYY'));
                calendarData[index + 1].status = 'Region booked';
            }
        }
    });

    calendarData.forEach((instance) => {
        if (instance.status === 'Available') {
            available.push(dayjs(instance.aptDate).format('DD/MM/YYYY'));
        } else if (instance.status === 'Booked') {
            booked.push(dayjs(instance.aptDate).format('DD/MM/YYYY'));
        }
    });

    // Deselect current date if it became booked/region booked during selection
    if (booked.includes(dayjs(dateData).format('DD/MM/YYYY')) || regionBooked.includes(dayjs(dateData).format('DD/MM/YYYY'))) {
        setDateData(new Date());
    }

    setKey(!key); // Manually re-render
},
[formData.uniRegion, checkDate]); // Dependencies which, after change, will trigger the useEffect
return (

```

```

146      <div className='container'>
147          <div className='row align-items-center'>
148              <div className='col-6'>
149                  <TextForm
150                      validated={validated}
151                      errors={error}
152                      isReadOnly={success}
153                      formData={formData}
154                      setFormData={setFormData}
155                  />
156              </div>
157              <div className='col-6'>
158                  <DateCalendar
159                      validated={validated}
160                      errors={error}
161                      available={available}
162                      booked={booked}
163                      regionBooked={regionBooked}
164                      success={success}
165                      selectedDate={dateData}
166                      setSelectedDate={setDateData}
167                  />
168              </div>
169          </div>
170          <div className='row align-items-center'>
171              <div className='col-6'>
172                  <TimeDisplay date={dateData} />
173              </div>
174
175              <div className='col-6'>
176                  <div className='submitAlign'>
177                      <Button
178                          className={success ? 'disabled button' :
179 'button'}
180                          as='input'
181                          type='submit'
182                          value='Submit'
183                          onClick={handleSubmit}
184                      />
185                  </div>
186              </div>
187              <h5
188                  id='successSubmit'
189                  className={success ? 'alert-success' : 'hidden'}>
190                  Thank you for signing up for the BISH virtual
191                  presentation, you
192                  will receive an email shortly with the details of
193                  your booking.
194              </h5>
195          </div>
196      );
197  export default FormPage;
198

```

```

1 // All the text fields for the form page
2
3 import React from 'react';
4 import Form from 'react-bootstrap/Form';
5
6 const TextForm = ({
7   validated,
8   errors,
9   isReadOnly,
10  formData,
11  setFormData
12 }) => {
13   return (
14     <div className='formAlign'>
15       <Form validated={validated}>
16         <Form.Group controlId='form.
University'>
17           <Form.Label>University</Form.Label>
18           <Form.Control
19             required
20             readOnly={isReadOnly ? true : false}
21             as='input'
22             type='text'
23             defaultValue={formData.uniName}
24             onChange={(e) => {
25               setFormData({...formData, uniName: e.target
26                 .value});
27               delete errors.uniRepName;
28             }}
29           />
30           <Form.Control.Feedback type='invalid'>
31             {errors.uniName}
32           </Form.Control.Feedback>
33         </Form.Group>
34         <Form.Group controlId='form.FullName'>
35           <Form.Label>Full name</Form.Label>
36           <Form.Control
37             required
38             readOnly={isReadOnly ? true : false}
39             as='input'
40             type='text'
41             defaultValue={formData.uniRepName}
42             onChange={(e) => {
43               setFormData({...formData, uniRepName: e.
44                 target.value});
45               delete errors.uniRepName;
46             }}
47           />
48           <Form.Control.Feedback type='invalid'>
49             {errors.uniRepName}
50           </Form.Control.Feedback>
51         </Form.Group>
52         <Form.Group controlId='form.JobTitle'>
53

```

```

53                  <Form.Label>Job title</Form.Label>
54                  <Form.Control
55                      required
56                      isValid={false}
57                      readOnly={isReadOnly ? true : false}
58                      as='input'
59                      type='text'
60                      defaultValue={formData.uniRepJobTitle}
61                      onChange={(e) =>
62                          setFormData({...formData, uniRepJobTitle:
63                              e.target.value,});
64                          delete errors.uniRepJobTitle;
65                      }
66                  />
67                  <Form.Control.Feedback type='invalid'>
68                      {errors.uniRepJobTitle}
69                  </Form.Control.Feedback>
70              </Form.Group>
71
72              <Form.Group className='mb-3' controlId='form.
73                  EmailAddress'>
74                  <Form.Label>Email address</Form.Label>
75                  <Form.Control
76                      required
77                      readOnly={isReadOnly ? true : false}
78                      as='input'
79                      type='email'
80                      defaultValue={formData.uniRepEmail}
81                      onChange={(e) =>
82                          setFormData({...formData, uniRepEmail: e.
83                              target.value,});
84                          delete errors.uniRepEmail;
85                      }
86                  />
87                  <Form.Control.Feedback type='invalid'>
88                      {errors.uniRepEmail}
89                  </Form.Control.Feedback>
90              </Form.Group>
91
92              <Form.Group className='mb-3' controlId='form.Region'>
93                  <Form.Label>Region</Form.Label>
94                  <Form.Control
95                      required
96                      disabled={isReadOnly ? true : false}
97                      as='select'
98                      defaultValue={formData.uniRegion}
99                      onChange={(e) =>
100                          setFormData({...formData, uniRegion: e.
101                              target.value,})
102                          }
103                  <option hidden={formData.uniRegion !== '' ?
104                      true : false} selected='selected'></option>
105                      <option value='USA'>USA</option>
106                      <option value='CANADA'>Canada</option>
107                      <option value='EUROPE'>Europe</option>
108                      <option value='UNITED KINGDOM'>United Kingdom
109                  </option>

```

```
104          <option value='OTHER'>Other</option>
105      </Form.Control>
106      <Form.Control.Feedback type='invalid'>
107          {errors.uniRegion}
108      </Form.Control.Feedback>
109      </Form.Group>
110  </Form>
111  </div>
112  );
113 };
114
115 export default TextForm;
116
```

```
1 abbr[title] {
2     border-bottom: none !important;
3     cursor: inherit !important;
4     text-decoration: none !important;
5 }
6 .react-calendar {
7     width: 38rem;
8     max-width: 100%;
9     background: white;
10    /* border: 1px solid #a0a096; */
11    font-family: 'system-ui';
12    line-height: 1.125em;
13 }
14 .react-calendar--doubleView {
15     width: 700px;
16 }
17 .react-calendar--doubleView .react-calendar__viewContainer {
18     display: flex;
19     margin: -0.5em;
20 }
21 .react-calendar--doubleView .react-calendar__viewContainer > * {
22     width: 50%;
23     margin: 0.5em;
24 }
25 .react-calendar,
26 .react-calendar *,
27 .react-calendar *:before,
28 .react-calendar *:after {
29     -moz-box-sizing: border-box;
30     -webkit-box-sizing: border-box;
31     box-sizing: border-box;
32 }
33 .react-calendar button {
34     margin: 0;
35     border: 0;
36     outline: none;
37 }
38 .react-calendar button:enabled:hover {
39     cursor: pointer;
40 }
41 .react-calendar__navigation {
42     height: 44px;
43     margin-bottom: 1em;
44 }
45 .react-calendar__navigation button {
46     min-width: 44px;
47     background: none;
48 }
49 .react-calendar__navigation button:enabled:hover,
50 .react-calendar__navigation button:enabled:active {
51     background-color: #e6e6e6;
52 }
53 /* .react-calendar__navigation button[disabled] {
54     background-color: #f0f0f0; */
55 /*} */
56 .react-calendar__month-view__weekdays {
57     text-align: center;
```

```
58     text-transform: uppercase;
59     font-weight: bold;
60     font-size: 0.75em;
61 }
62 .react-calendar__month-view__weekdays__weekday {
63     padding: 0.5em;
64 }
65 .react-calendar__month-view__weekNumbers {
66     font-weight: bold;
67 }
68 .react-calendar__month-view__weekNumbers .react-calendar__tile {
69     display: flex;
70     align-items: center;
71     justify-content: center;
72     font-size: 0.75em;
73     padding: calc(0.75em / 0.75) calc(0.5em / 0.75);
74 }
75 .react-calendar__month-view__days__day--weekend {
76     color: #d10000;
77 }
78 .react-calendar__month-view__days__day--neighboringMonth {
79     color: #757575;
80 }
81 .react-calendar__year-view .react-calendar__tile,
82 .react-calendar__decade-view .react-calendar__tile,
83 .react-calendar__century-view .react-calendar__tile {
84     padding: 2em 0.5em;
85 }
86 .react-calendar__tile {
87     max-width: 100%;
88     text-align: center;
89     padding: 0.75em 0.5em;
90     background: none;
91 }
92 .react-calendar__tile:disabled {
93     background-color: #f0f0f0;
94 }
95 .react-calendar__tile:enabled:hover,
96 .react-calendar__tile:enabled:focus {
97     background-color: #e6e6e6;
98 }
99 .react-calendar__tile--now {
100     background: #ffff76;
101 }
102 .react-calendar__tile--now:enabled:hover,
103 .react-calendar__tile--now:enabled:focus {
104     background: #ffffa9;
105 }
106 .react-calendar__tile--hasActive {
107     background: #76baff;
108 }
109 .react-calendar__tile--hasActive:enabled:hover,
110 .react-calendar__tile--hasActive:enabled:focus {
111     background: #a9d4ff;
112 }
113 .react-calendar__tile--active {
114     background: #006edc;
```

```
115     color: white;
116 }
117 .react-calendar__tile--active:enabled:hover,
118 .react-calendar__tile--active:enabled:focus {
119     background: #1087ff;
120 }
121 .react-calendar--selectRange .react-calendar__tile--hover {
122     background-color: #e6e6e6;
123 }
124
```

```

1 // Simple time display component
2
3 import React from 'react';
4 import dayjs from 'dayjs';
5 import utc from 'dayjs/plugin/utc';
6 import isToday from 'dayjs/plugin/isToday';
7 import advancedFormat from 'dayjs/plugin/advancedFormat';
8 import timezone from 'dayjs/plugin/timezone';
9
10 dayjs.extend(isToday);
11 dayjs.extend(utc);
12 dayjs.extend(timezone);
13 dayjs.extend(advancedFormat);
14
15 const TimeDisplay = ({ date }) => {
16     let aptDate = dayjs(date).utc().hour(13).minute(30).second(0); // Sets date to predetermined appointment time
17     let visible = dayjs(date).format('DD/MM/YYYY') === dayjs(new Date()).format('DD/MM/YYYY'); // Checks if a date has been selected
18
19     return ( // A lot of the code here is just formatting
20         <div className='timeBox'>
21             <h1 className='timeTitle'>Your appointment is at</h1>
22             <div className='localTime'>
23                 <h1 className='localTimeTitle'>Local time</h1>
24                 {visible ? dayjs(aptDate).local().format('h:mm A') : dayjs(aptDate).local().format('MMMM D, h:mm A')}
25                 <br />
26                 {'GMT' + dayjs(aptDate).local().format('z (z)')}
27             </div>
28             <br />
29
30             <div className='ourTime'>
31                 <h1 className='ourTimeTitle'>Our time</h1>
32                 {visible ? dayjs(aptDate).tz('America/Chicago').format('h:mm A') : dayjs(aptDate).tz('America/Chicago').format('MMMM D, h:mm A')}
33                 <br />
34                 {'GMT' + dayjs(aptDate).tz('America/Chicago').format('z (z)')}
35             </div>
36         </div>
37     );
38 };
39
40 export default TimeDisplay;
41

```

```

1 // Dynamic calendar
2
3 import React from 'react';
4 import Calendar from 'react-calendar';
5 import './Calendar.css'
6 // import 'react-calendar/dist/Calendar.css'; <- Old CSS file
7 import dayjs from 'dayjs';
8
9 const DateCalendar = ({
10     errors,
11     validated,
12     available,
13     booked,
14     regionBooked,
15     success,
16     selectedDate,
17     setSelectedDate,
18 }) => {
19     return (
20         <div className='calendarAlign disableCalendar'>
21             <div className={validated && !errors.date ? 'invalidCalendar' : ''}> {/*Control validation formatting*/}
22                 <Calendar
23                     onChange={setSelectedDate}
24                     value={selectedDate}
25                     maxDate={new Date(dayjs().add(11, 'month').endOf('month').toDate())}
26                     minDate={new Date()}
27                     next2Label=''
28                     prev2Label=''
29                     minDetail='year'
30                     showNeighboringMonth='false'
31                     tileClassName={({ date, view }) => { /*Conditional formatting*/
32                         if (dayjs(date).format('DD/MM/YYYY') === dayjs(selectedDate).format('DD/MM/YYYY')) {
33                             return 'selected';
34                         } else if (available.includes(dayjs(date).format('DD/MM/YYYY'))) {
35                             return 'available';
36                         } else if (booked.includes(dayjs(date).format('DD/MM/YYYY'))) {
37                             return 'booked';
38                         } else if (regionBooked.includes(dayjs(date).format('DD/MM/YYYY'))) {
39                             return 'regionBooked';
40                         } else if (date > new Date()) {
41                             return 'unavailable';
42                         } else {
43                             return 'past';
44                         }
45                     }
46                 }
47             </div>
48         </div>
49     )
50 }

```

```
51          tileDisabled={({ date, view }) => { /*  
52              Conditional disabling of dates*/  
53              if (success && dayjs(date).format('DD/MM/YYYY')  
54                  !== dayjs(selectedDate).format('DD/MM/YYYY')) {  
55                  return true;  
56              } else {  
57                  if (available.includes(dayjs(date).format  
58                      ('DD/MM/YYYY')))) {  
59                      return false;  
60                  } else {  
61                      return true;  
62                  }  
63              }  
64          }  
65      }  
66      <span className='invalidText'>{errors.date}</span>  
67  </div>  
68 );  
69 };  
70  
71 export default DateCalendar;  
72
```

```
1 // Validation code
2
3 import dayjs from 'dayjs';
4
5 export default function validateInfo(info) {
6     let errors = {};
7     const regex =
8         /^(([^\<><()[]\\.,;:\\s@"]+(\.[^\<><()[]\\.,;:\\s@"]+)*|(.+))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z]-0-9]+\.)+[a-zA-Z]{2,})$/;
9
10    if (dayjs(info.aptDate).format('DD/MM/YYYY') === dayjs(new Date()).format('DD/MM/YYYY')) {
11        errors.date = 'Date required';
12    }
13    if (info.booking.uniName === '') {
14        errors.uniName = 'University name required';
15    }
16    if (info.booking.uniRepName === '') {
17        errors.uniRepName = 'Name required';
18    }
19    if (info.booking.uniRepJobTitle === '') {
20        errors.uniRepJobTitle = 'Job title required';
21    }
22    if (info.booking.uniRepEmail === '') {
23        errors.uniRepEmail = 'Email required';
24
25    } else if (!regex.test(info.booking.uniRepEmail)) {
26        errors.uniRepEmail = 'Invalid email';
27    }
28    if (info.booking.uniRegion === '') {
29        errors.uniRegion = 'Region required';
30    }
31
32    return errors;
33 }
34
```

```
1 .cardUni{  
2     font-size: 1.5rem;  
3     font-weight: 300;  
4 }  
5  
6 .cardRegion{  
7     font-size: 1rem;  
8     font-weight: 500;  
9     color: #6c757d;  
10 }  
11  
12 .cardImage{  
13     width: calc(100% + 2px);  
14     border-radius: 0;  
15     margin-left: -1px;  
16     border-left: 1px solid #DFDFDF;  
17     border-right: 1px solid #DFDFDF;  
18 }  
19  
20 .cardDate{  
21 }  
22 }  
23  
24 .cardTitle{  
25 }  
26 }  
27  
28 .cardEmail{  
29 }  
30 }  
31  
32 .pad{  
33     margin-top: 1rem;  
34     margin-bottom: 1rem;  
35 }  
36  
37 .cardButton{  
38     width: 100%;  
39 }  
40  
41 .login-container{  
42     display: flex;  
43     justify-content: center;  
44     align-items: center;  
45 }  
46  
47 .ccontainer {  
48     margin-left: calc(50vw - 20rem);  
49     margin-top: calc(50vh - 15rem);  
50 }  
51  
52 .cbooked {  
53     color: hsla(3, 100%, 20%, 1) !important;  
54     background-color: rgba(255, 65, 54, 0.6) !important;  
55 }  
56  
57 .cbooked:hover {
```

```
58     color: hsla(3, 100%, 20%, 1) !important;
59     background-color: rgba(255, 65, 54, 0.55) !important;
60 }
61
62 .cavailable {
63     color: hsla(127, 63%, 15%, 1) !important;
64     background-color: rgba(1, 255, 112, 0.7) !important;
65 }
66
67 .cavailable:hover {
68     color: hsla(127, 63%, 15%, 1) !important;
69     background-color: rgba(1, 255, 112, 0.5) !important;
70 }
```

```
1 // Same algorithm as used in AdminPage/bookings
2
3 import {useState} from "react";
4 import Card from 'react-bootstrap/Card'
5 import '../admin.css'
6 import Button from 'react-bootstrap/Button';
7 import Modal from 'react-bootstrap/Modal';
8 import Form from 'react-bootstrap/Form'
9 import axios from 'axios';
10 import validateCouns from "./validateCouns";
11
12 const CCard = ({instance, setEmails}) => {
13     const [show, setShow] = useState(false);
14     const [validated, setValidated] = useState(false);
15     const [error, setError] = useState({});
16
17     const [counsInfo, setCounsInfo] = useState({
18         name: instance.name,
19         counsEmail: instance.counsEmail,
20         receiveEmail: instance.receiveEmail,
21     })
22
23     const [tempCounsInfo, setTempCounsInfo] = useState({
24         name: instance.name,
25         counsEmail: instance.counsEmail,
26         receiveEmail: instance.receiveEmail,
27     })
28
29     function cancel(){
30         setCounsInfo(tempCounsInfo)
31         setShow(false)
32     }
33
34     function handleOpen(){
35         setTempCounsInfo(counsInfo)
36         setShow(true);
37         setValidated(false);
38     }
39
40     async function getEmail(){
41         let res = await axios.get('http://localhost:5000/couns/read')
42         return res
43     }
44
45     async function edit(){
46         if(!Object.keys(validateCouns(counsInfo)).length === 0)){
47             setError(validateCouns(counsInfo));
48             setValidated(true);
49         } else{
50             const token = localStorage.getItem('adminToken')
51             let verify;
52
53             if (!!token) {
54                 verify = await axios.post('http://localhost:5000/login/verify', {adminToken: token});
55             } else {
56                 verify = false;
57             }
58
59             if(verify){
60                 setShow(false);
61                 setValidated(true);
62             } else {
63                 setError("Verification failed");
64             }
65         }
66     }
67
68     return (
69         <Card>
70             <Card.Body>
71                 <Form.Group controlId="formBasicEmail">
72                     <Form.Label>Email address</Form.Label>
73                     <Form.Control type="email" value={counsInfo.receiveEmail} />
74                     <Form.Text>We'll never share your email with anyone else.</Form.Text>
75                 </Form.Group>
76                 <Form.Group controlId="formBasicPassword">
77                     <Form.Label>Password</Form.Label>
78                     <Form.Control type="password" value={counsInfo.receiveEmail} />
79                 </Form.Group>
80                 <Form.Group controlId="formBasicName">
81                     <Form.Label>Name</Form.Label>
82                     <Form.Control type="text" value={counsInfo.name} />
83                 </Form.Group>
84                 <Form.Group controlId="formBasicCounsEmail">
85                     <Form.Label>Couns Email</Form.Label>
86                     <Form.Control type="text" value={counsInfo.counsEmail} />
87                 </Form.Group>
88                 <Form.Group controlId="formBasicReceiveEmail">
89                     <Form.Label>Receive Email</Form.Label>
90                     <Form.Control type="text" value={counsInfo.receiveEmail} />
91                 </Form.Group>
92                 <Form.Group controlId="formBasicValidated">
93                     <Form.Label>Validated</Form.Label>
94                     <Form.Control type="checkbox" checked={validated} />
95                 </Form.Group>
96                 <Form.Group controlId="formBasicError">
97                     <Form.Label>Error</Form.Label>
98                     <Form.Control type="text" value={error} />
99                 </Form.Group>
100                 <Form.Group controlId="formBasicCancel">
101                     <Form.Label>Cancel</Form.Label>
102                     <Form.Control type="button" value="Cancel" onClick={cancel} />
103                 </Form.Group>
104                 <Form.Group controlId="formBasicHandleOpen">
105                     <Form.Label>Handle Open</Form.Label>
106                     <Form.Control type="button" value="Handle Open" onClick={handleOpen} />
107                 </Form.Group>
108                 <Form.Group controlId="formBasicEdit">
109                     <Form.Label>Edit</Form.Label>
110                     <Form.Control type="button" value="Edit" onClick={edit} />
111                 </Form.Group>
112             </Card.Body>
113         </Card>
114     );
115 }
116
117 export default CCard;
```

```

57         }
58
59         if(verify){
60             setValidated(false);
61
62             counsInfo.id = instance._id
63             counsInfo.adminToken = token
64
65             axios.patch('http://localhost:5000/couns/edit',
66             counsInfo);
67             setShow(false);
68         } else{
69             alert('Invalid authentication token')
70         }
71
72     }
73 }
74
75     async function remove(){
76         const token = localStorage.getItem('adminToken')
77         let verify;
78
79         if (!!token) {
80             verify = await axios.post('http://localhost:5000/login/
80 verify', {adminToken:token});
81         } else {
82             verify = false;
83         }
84
85         if(verify){
86             axios.delete('http://localhost:5000/couns/remove', {data
86 : {id:instance._id, adminToken:token}})
87             .then(()=>{
88                 // eslint-disable-next-line
89                 getEmail()
90                 .then((res)=>{
91                     setEmails(res.data);
92                 })
93                 .then(setShow(false));
94             } else{
95                 alert('Invalid authentication token')
96             }
97
98         }
99
100     return (
101         <>
102             <Modal show={show} onHide={cancel} backdrop="static">
103                 <Modal.Header closeButton>
104                     <Modal.Title>{tempCounsInfo.name}</Modal.Title>
105                 </Modal.Header>
106                 <Modal.Body>
107                     <Form>
108                         <Form.Group>
109                             <Form.Label> ID </Form.Label>
110                             <Form.Control>

```

```

111                      type='text'
112                      value={instance._id}
113                      readOnly={true}
114                  />
115              </Form.Group>
116              <Form.Group>
117                  <Form.Label> Counsellor Name </Form.Label>
118              >
119                  <Form.Control
120                      type='text'
121                      defaultValue={tempCounsInfo.name}
122                      onChange={(e) => {
123                          setCounsInfo({...counsInfo, name:
124                              e.target.value,})
125                      }}
126                      isValid={validated?!error.name:
127                      false}
128                  >
129                  <Form.Group>
130                      <Form.Label> Email </Form.Label>
131                      <Form.Control
132                          type='text'
133                          defaultValue={tempCounsInfo.
134                          counsEmail}
135                          onChange={(e) => {
136                              setCounsInfo({...counsInfo,
137                              counsEmail: e.target.value,})
138                          }}
139                          isValid={validated?!error.counsEmail:
140                          false}
141                      >
142                  </Form.Group>
143                  <Form.Group>
144                      <Form.Label> Receive Emails? </Form.Label>
145                  >
146                      <Form.Control
147                          type='checkbox'
148                          defaultChecked={tempCounsInfo.
149                          receiveEmail}
150                          onChange={(e)=> {
151                              setCounsInfo({...counsInfo,
152                              receiveEmail: e.target.value,})
153                          }}
154                          isValid={validated?!error.
155                          receiveEmail: false}
156                      >
157                  </Form.Group>
158              </Form>
159          </Modal.Body>
160          <Modal.Footer>
161              <Button variant="danger" onClick={()=>{remove

```

```
155 ()} style={{'margin-right':'auto'}}>
156             Delete Booking
157             </Button>
158             <Button variant="primary" onClick={()=>{edit
159                 ()}} >
160                 Save Changes
161             </Button>
162             </Modal.Footer>
163         </Modal>
164         <div className='col-6 pad'>
165             <Card style={{ width: '100%' }}>
166                 <Card.Body>
167                     <Card.Title className='cardUni'>{counsInfo.
168                         name}</Card.Title>
169                     <Card.Text className='cardEmail'>{counsInfo.
170                         counsEmail}</Card.Text>
171                     <Card.Text className='cardTitle'>{'Receive
172                         Emails?: '+counsInfo.receiveEmail}</Card.Text>
173                     <Button variant="primary" className="
174                         cardButton" onClick={() => {handleOpen()}}>Edit</Button>
175                 </Card.Body>
176             </Card>
177             </div>
178         </>
179     )
176 }
177
178 export default CCard
179
```

```
1 // Same algorithm as used in AdminPage/bookings
2
3 import {useState} from "react";
4 import Card from 'react-bootstrap/Card'
5 import '../admin.css'
6 import Button from 'react-bootstrap/Button';
7 import Modal from 'react-bootstrap/Modal';
8 import Form from 'react-bootstrap/Form'
9 import axios from 'axios';
10 import validateCouns from "./validateCouns";
11
12 const CCardAdd = ({setEmails}) => {
13     const [show, setShow] = useState(false);
14     const [validated, setValidated] = useState(false);
15     const [error, setError] = useState({});
16
17     const [counsInfo, setCounsInfo] = useState({
18         name: "Name",
19         counsEmail: "Email",
20         receiveEmail:true,
21     })
22
23     const [tempCounsInfo, setTempCounsInfo] = useState({
24         name: "Name",
25         counsEmail: "Email",
26         receiveEmail:true,
27     })
28
29     function cancel(){
30         setCounsInfo(tempCounsInfo)
31         setShow(false)
32     }
33
34     function handleOpen(){
35         setTempCounsInfo(counsInfo)
36         setShow(true);
37         setValidated(false);
38     }
39
40     async function getEmail(){
41         let res = await axios.get('http://localhost:5000/couns/read')
42         return res
43     }
44
45     async function create(){
46         if(!(Object.keys(validateCouns(counsInfo)).length === 0)){
47             setError(validateCouns(counsInfo));
48             setValidated(true);
49         } else{
50             const token = localStorage.getItem('adminToken')
51             let verify;
52
53             if(!token) {
54                 verify = await axios.post('http://localhost:5000/login
55 /verify', {adminToken:token});
56             } else {
57                 verify = false;
58             }
59             if(verify) {
60                 localStorage.setItem('adminToken', verify.token);
61                 setShow(false);
62             } else {
63                 setError("Email already exists");
64             }
65         }
66     }
67
68     function handleEmail(e){
69         e.preventDefault();
70         const name = e.target[0].value;
71         const email = e.target[1].value;
72
73         if(name === "" || email === ""){
74             setError("Please fill all fields");
75             return;
76         }
77
78         const counsInfo = {
79             name: name,
80             counsEmail: email,
81             receiveEmail: true
82         }
83
84         if(show){
85             setTempCounsInfo(counsInfo);
86             setShow(false);
87         } else {
88             setCounsInfo(counsInfo);
89             setShow(true);
90         }
91
92         handleOpen();
93     }
94
95     function handleCancel(e){
96         e.preventDefault();
97         cancel();
98     }
99
100     function handleCreate(e){
101         e.preventDefault();
102         create();
103     }
104
105     function handleVerify(e){
106         e.preventDefault();
107         verify();
108     }
109
110     function handleLogout(e){
111         e.preventDefault();
112         localStorage.removeItem('adminToken');
113         window.location.href = '/login';
114     }
115
116     return (
117         <Card>
118             <Card.Body>
119                 <Form>
120                     <Form.Group>
121                         <Form.Label>Name</Form.Label>
122                         <Form.Control type="text" value={counsInfo.name} onChange={handleEmail}>
123                     </Form.Group>
124                     <Form.Group>
125                         <Form.Label>Email</Form.Label>
126                         <Form.Control type="text" value={counsInfo.counsEmail} onChange={handleEmail}>
127                     </Form.Group>
128                     <Form.Group>
129                         <Form.Check type="checkbox" checked={counsInfo.receiveEmail} onChange={handleEmail}>
130                             Receive Email
131                         </Form.Check>
132                     </Form.Group>
133                 </Form>
134             </Card.Body>
135         </Card>
136     );
137 }
138
139 export default CCardAdd;
```

```

57         }
58
59         if(verify){
60             setValidated(false);
61
62             counsInfo.adminToken = token
63
64             axios.post('http://localhost:5000/couns/create',
65             counsInfo)
66                 .then(()=>{
67                     getEmail()
68                     .then((res)=>{
69                         setEmails(res.data)
70                     })
71                     .catch((err)=>{
72                         console.log(err)
73                         alert(err)
74                     })
75                     setShow(false);
76                 } else{
77                     alert('Invalid authentication token')
78                 }
79             }
80
81         return (
82             <>
83                 <Modal show={show} onHide={cancel} backdrop="static">
84                     <Modal.Header closeButton>
85                         <Modal.Title>Name</Modal.Title>
86                     </Modal.Header>
87                     <Modal.Body>
88                         <Form>
89                             <Form.Group>
90                                 <Form.Label> Counsellor Name </Form.Label>
91                             >
92                                 <Form.Control
93                                     type='text'
94                                     onChange={(e) => {
95                                         setCounsInfo({...counsInfo, name:
96                                         e.target.value,})
97                                     }}
98                                     isValid={validated?!error.name:
99                                     false}
100                                     isValid={validated?!error.name:false}
101                                 />
102                                 </Form.Group>
103                                 <Form.Group>
104                                     <Form.Label> Email </Form.Label>
105                                     <Form.Control
106                                         type='text'
107                                         onChange={(e) => {
108                                         setCounsInfo({...counsInfo,
109                                         counsEmail: e.target.value,})
110                                         }}
111                                         isValid={validated?!error.
112                                         counsEmail:false}

```

```

108                     isValid={validated?!error.counsEmail:
  false}
109                     />
110                 </Form.Group>
111                 <Form.Group>
112                     <Form.Label> Receive Emails? </Form.Label>
113                     <Form.Control
114                         type='checkbox'
115                         defaultChecked={true}
116                         onClick={()=>{setCounsInfo({...
117                             counsInfo, receiveEmail:!counsInfo.receiveEmail})}}
118                         isValid={validated?!error.
  receiveEmail:false}
119                         isValid={validated?!error.
  receiveEmail:false}
120                     />
121                 </Form.Group>
122             </Form>
123             <Modal.Body>
124                 <Button variant="success" onClick={()=>{create
  ()}}>
125                     Add Counsellor
126                 </Button>
127             </Modal.Footer>
128         </Modal>
129         <div className='col-6 pad'>
130             <Card style={{ width: '100%' }}>
131                 <Card.Body>
132                     <Card.Title className='cardUni'>Name</Card.
  Title>
133                     <Card.Text className='cardTitle'>Email</Card.
  Text>
134                     <Card.Text className='cardEmail'>Receive
  Emails?: true/false</Card.Text>
135                     <Button variant="success" className="
  cardButton" onClick={() => {handleOpen()}}>Add</Button>
136                 </Card.Body>
137             </Card>
138         </div>
139     </>
140 )
141 }
142
143 export default CCardAdd
144

```

```
1 // Same algorithm as used in AdminPage/bookings
2
3 import { useEffect, useState } from "react"
4 import axios from 'axios';
5 import CCard from "./CCard";
6 import CCardAdd from "./CCardAdd";
7
8 const EditEmail = () => {
9     const [emails, setEmails] = useState([])
10
11     useEffect(async () => {
12         async function getEmail(){
13             let res = await axios.get('http://localhost:5000/couns/
read')
14             return res
15         }
16
17         getEmail().then((res)=>setEmails(res.data))
18     }, [])
19
20     return (
21         <div className='container'>
22             <div className='row pad'>
23                 {emails.map(instance=>{
24                     return(
25                         <CCard instance={instance} setEmails={setEmails
}/>
26                         )
27                     })}
28                     <CCardAdd setEmails={setEmails}/>
29                 </div>
30             </div>
31     )
32 }
33
34 export default EditEmail
35
```

```
1 // Same algorithm as used in AdminPage/bookings
2
3 export default function validateCouns(counsInfo) {
4     let errors = {};
5     const regex =
6         /^(([^\<^\>()[]\\.,;:\\s@"]+([\\.[^\<^\>()[]\\.,;:\\s@"]+)*|(.+))@(([\\[[\\
7 0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\])|(([a-zA-Z\\-0-9]+\\.\\.[a-zA-Z]{2,}))$/;
8
9     if (counsInfo.name === '' || counsInfo.name === 'Name') {
10         errors.name = 'University name required';
11     }
12     if (counsInfo.counsEmail === '' || counsInfo.counsEmail === 'Email
13 ') {
14         errors.counsEmail = 'Email required';
15     } else if (!regex.test(counsInfo.counsEmail)) {
16         errors.counsEmail = 'Invalid email';
17     }
18
19     return errors
20 }
```

```

1 // Booking card template used to display upcoming bookings
2 // Also features the modal used to edit bookings
3
4 import {useState} from "react";
5 import Card from 'react-bootstrap/Card'
6 import image from './undraw_Page_not_found_re_e9o6.png'
7 import dayjs from "dayjs";
8 import '../admin.css'
9 import Button from 'react-bootstrap/Button';
10 import Modal from 'react-bootstrap/Modal';
11 import Form from 'react-bootstrap/Form'
12 import axios from 'axios';
13 import validateInfo from "./validateCreate";
14
15 const BCard = ({instance, setBooking}) => {
16     const [show, setShow] = useState(false);
17     const [validated, setValidated] = useState(false);
18     const [error, setError] = useState({date:''});
19
20     const [date, setDate] = useState(instance.aptDate);
21     const [tempDate, setTempDate] = useState(instance.aptDate);
22
23     const [sendEmail, setSendEmail] = useState(false)
24
25     const [bookingData, setBookingData] = useState({
26         uniName: instance.booking.uniName,
27         uniRegion: instance.booking.uniRegion,
28         uniRepJobTitle: instance.booking.uniRepJobTitle,
29         uniRepName: instance.booking.uniRepName,
30         uniRepEmail: instance.booking.uniRepEmail,
31         logoUrl:instance.booking.logoUrl,
32     });
33     const [tempBooking, setTempBooking] = useState({
34         uniName: instance.booking.uniName,
35         uniRegion: instance.booking.uniRegion,
36         uniRepJobTitle: instance.booking.uniRepJobTitle,
37         uniRepName: instance.booking.uniRepName,
38         uniRepEmail: instance.booking.uniRepEmail,
39         logoUrl:instance.booking.logoUrl,
40     });
41
42     function cancel(){
43         setBookingData(tempBooking)
44         setDate(tempDate)
45         setShow(false);
46     }
47
48     function handleOpen(){
49         setTempBooking(bookingData);
50         setTempDate(date);
51         setValidated(false);
52         setShow(true);
53     }
54
55     async function getDisplayData() { // Async to prevent stalling
56         let res = await axios.get('http://localhost:5000/booking/
readfordisplay');

```

```

57         return res;
58     }
59
60     async function edit(){
61         if(!(Object.keys(validateInfo(bookingData,date)).length === 0
62 )){           setError(validateInfo(bookingData,date));
63           setValidated(true);
64     } else{
65         const token = localStorage.getItem('adminToken')
66         let verify;
67
68         if (!!token) {
69             verify = await axios.post('http://localhost:5000/
69 login/verify', {adminToken:token});
70         } else {
71             verify = false;
72         }
73
74         if(verify){
75             setValidated(false);
76             let bookingDateInfoInstance = {
77                 id:instance._id,
78                 aptDate: date,
79                 status: 'Booked',
80                 booking: {
81                     uniName: bookingData.uniName,
82                     uniRepName: bookingData.uniRepName,
83                     uniRepJobTitle: bookingData.uniRepJobTitle,
84                     uniRepEmail: bookingData.uniRepEmail,
85                     uniRegion: bookingData.uniRegion,
86                     logoUrl: bookingData.logoUrl,
87                 },
88                 allowMail: sendEmail,
89                 adminToken: token,
90             };
91
92             axios.patch('http://localhost:5000/booking/
92 editbooking', bookingDateInfoInstance);
93             setShow(false);
94         } else{
95             alert('Invalid authentication token')
96         }
97     }
98 }
99
100
101    async function remove(){
102        const token = localStorage.getItem('adminToken')
103        let verify;
104
105        if (!!token) {
106            verify = await axios.post('http://localhost:5000/login/
106 verify', {adminToken:token});
107        } else {
108            verify = false;
109        }

```

```

110
111      if	verify{
112          axios.delete('http://localhost:5000/booking/remove', {
113              data: {id:instance._id,adminToken:token}})
114              .then(()=>{
115                  // eslint-disable-next-line
116                  getDisplayData()
117                  .then((res)=>{
118                      setBooking(res.data);
119                  })
120              }).then(setShow(false));
121      } else{
122          alert('Invalid authentication token')
123      }
124
125      return (
126      <>
127          <Modal show={show} onHide={cancel} backdrop="static">
128              <Modal.Header closeButton>
129                  <Modal.Title>{tempBooking.uniName}</Modal.Title>
130              </Modal.Header>
131              <Modal.Body>
132                  <Form >
133                      <Form.Group>
134                          <Form.Label> ID </Form.Label>
135                          <Form.Control
136                              type='text'
137                              value={instance._id}
138                              readOnly={true}
139                          />
140                      </Form.Group>
141                      <Form.Group>
142                          <Form.Label> University Name </Form.Label>
143                      <Form.Control
144                          type='text'
145                          defaultValue={tempBooking.uniName}
146                          onChange={(e) => {
147                              setBookingData({...bookingData,
148                                  uniName: e.target.value,})
149                          }}
150                          isValid={validated?!error.uniName:
151                          false}
152                          isValid={validated?!error.uniName:
153                          false}
154                      </Form.Group>
155                      <Form.Group>
156                          <Form.Label> University Region </Form.
157                          Label>
158                          <Form.Control
159                              type='text'
160                              defaultValue={tempBooking.uniRegion}
161                              onChange={(e) => {
162                                  setBookingData({...bookingData,
163                                  uniRegion: e.target.value,})
```

```

160                                }()
161                                isValid={validated?!error.
162                                uniRegion:false}
163                                isValid={validated?!error.uniRegion:
164                                false}
165                                />
166                                </Form.Group>
167                                <Form.Group>
168                                    <Form.Label> Date </Form.Label>
169                                    <Form.Control
170                                        type='text'
171                                        defaultValue={dayjs(tempDate).format(
172                                            'M/D/YYYY')}
173                                        onChange={(e)=> {
174                                            setDate(new Date(dayjs(e.target.
175                                            value)))}
176                                            }}
177                                            isValid={validated?!error.date:
178                                            false}
179                                            isValid={validated?!error.date:false}
180                                            />
181                                            </Form.Group>
182                                            <Form.Group>
183                                                <Form.Label> Rep Title </Form.Label>
184                                                <Form.Control
185                                                    type='text'
186                                                    defaultValue={tempBooking.
187                                                    uniRepJobTitle}
188                                                    onChange={(e) => {
189                                                        setBookingData({...bookingData,
190                                                        uniRepJobTitle: e.target.value,})
191                                                        }}
192                                                        isValid={validated?!error.
193                                                        uniRepJobTitle:false}
194                                                        isValid={validated?!error.
195                                                        uniRepJobTitle:false}
196                                                        />
197                                                        </Form.Group>
198                                                        <Form.Group>
199                                                <Form.Label> Rep Name </Form.Label>
200                                                <Form.Control
201                                                    type='text'
202                                                    defaultValue={tempBooking.uniRepName}
203                                                    onChange={(e) => {
204                                                        setBookingData({...bookingData,
205                                                        uniRepName: e.target.value,})
206                                                        }}
207                                                        isValid={validated?!error.
208                                                        uniRepName:false}
209                                                        isValid={validated?!error.uniRepName:
210                                                        false}
211                                                        />
212                                                        </Form.Group>
213                                                        <Form.Group>
214                                                <Form.Label> Rep Email </Form.Label>
215                                                <Form.Control

```

```

205                               type='text'
206                               defaultValue={tempBooking.uniRepEmail
207                               }
208                               onChange={(e) => {
209                               setBookingData({...bookingData,
210                               uniRepEmail: e.target.value,})
211                               }
212                               }
213                               isInvalid={validated?!error.
214                               uniRepEmail:false}
215                               isValid={validated?!error.uniRepEmail
216                               :false}
217                               />
218                               </Form.Group>
219                               <Form.Group>
220                               <Form.Label> Image URL </Form.Label>
221                               <Form.Control
222                               type='text'
223                               defaultValue={tempBooking.logoUrl}
224                               onChange={(e) => {
225                               setBookingData({...bookingData,
226                               logoUrl: e.target.value,})
227                               }
228                               }
229                               isInvalid={validated?true:false}
230                               />
231                               </Form.Group>
232                               <Form.Group>
233                               <Form.Label> Send Email? </Form.Label>
234                               <Form.Control
235                               type='checkbox'
236                               defaultChecked={true}
237                               onChange={(e) => {
238                               setSendEmail(e.target.value)
239                               }
240                               }
241                               isInvalid={validated?true:false}
242                               />
243                               </Form.Group>
244                               </Form>
245                               </Modal.Body>
246                               <Modal.Footer>
247                               <Button variant="danger" onClick={()=>{remove
248                               ()}} style={{'margin-right':'auto'}}>
249                               Delete Booking
250                               </Button>
251                               <Button variant="primary" onClick={()=>{edit
252                               ()}} >
253                               Save Changes
254                               </Button>
255                               </Modal.Footer>
256                               </Modal>
257
258                               <div className='col-4 pad'>
259                               <Card style={{ width: '100%' }}>
260                               <Card.Body>
261                               <Card.Title className='cardUni'>{bookingData.
262                               uniName}</Card.Title>
263                               <Card.Subtitle className='cardRegion'>{
264                               bookingData.uniRegion}</Card.Subtitle>

```

```
253                     </Card.Body>
254                     <Card.Img variant="top" src={!bookingData.
255                         logoUrl?bookingData.logoUrl:image} className='cardImage'>
256                     <Card.Body>
257                         <Card.Text className='cardDate'>{dayjs(date).
258                             format('dddd, MMMM D')}</Card.Text>
259                         <Card.Text className='cardTitle'>{bookingData
260                             .uniRepJobTitle + ': ' + bookingData.uniRepName}</Card.Text>
261                         <Card.Text className='cardEmail'>{bookingData
262                             .uniRepEmail}</Card.Text>
263                         <Button variant="primary" className="cardButton" onClick={() => {handleOpen()}}>Edit</Button>
264                     </Card.Body>
265                 </Card>
266             </div>
267         )
268     )
269 }
```

```

1 // Add booking card template used to display upcoming bookings
2 // Uses a modal that displays a form similiar to the form page
3
4 import {useState} from "react";
5 import Card from 'react-bootstrap/Card'
6 import image from './undraw_No_data_re_kwbl.png'
7 import dayjs from "dayjs";
8 import '../admin.css'
9 import Button from 'react-bootstrap/Button';
10 import Modal from 'react-bootstrap/Modal';
11 import Form from 'react-bootstrap/Form'
12 import axios from 'axios';
13 import validateInfo from "./validateCreate";
14
15 const BCardAdd = ({setBooking}) => {
16     const [show, setShow] = useState(false);
17     const [validated, setValidated] = useState(false);
18     const [error, setError] = useState(false)
19
20     const [date, setDate] = useState("Month/Day/Year"); // Store
21     appointment date
21     const [tempDate, setTempDate] = useState("Month/Day/Year"); // Store for cancel
22
23     const [bookingData, setBookingData] = useState({ // Store booking
24     information
25         uniName: "University Name",
26         uniRegion: "University Region",
27         uniRepJobTitle: "Rep Title",
28         uniRepName: "Rep Name",
29         uniRepEmail: "Rep Email",
30         logoUrl: "",
31     });
31     const [tempBooking, setTempBooking] = useState({ // Store for
32     cancel
33         uniName: "University Name",
34         uniRegion: "University Region",
35         uniRepJobTitle: "Rep Title",
36         uniRepName: "Rep Name",
37         uniRepEmail: "Rep Email",
38         logoUrl: "",
39     });
40
41     function cancel(){
42         setBookingData(tempBooking)
43         setDate(tempDate)
44         setShow(false);
45     }
46
47     function handleOpen(){
48         setTempBooking(bookingData);
49         setTempDate(date);
50         setValidated(false);
51         setShow(true);
52     }
53
53     async function getDisplayData() { // Async to prevent stalling

```

```

54         let response = await axios.get('http://localhost:5000/booking
55         /readfordisplay');
56     }
57
58     async function create(){
59         if(!(Object.keys(validateInfo(bookingData,date)).length === 0
59 ))
60             setError(validateInfo(bookingData,date));
61             setValidated(true);
62         } else{
63             const token = localStorage.getItem('adminToken')
64             let verify;
65
66             if(!token) {
67                 verify = await axios.post('http://localhost:5000/
68             login/verify', {adminToken:token});
69             } else {
70                 verify =  false;
71             }
72
73             if(verify){
74                 setValidated(false);
75
76                 let bookingDateInfoInstance = {
77                     aptDate: date,
78                     status: 'Booked',
79                     booking: {
80                         uniName: bookingData.uniName,
81                         uniRepName: bookingData.uniRepName,
82                         uniRepJobTitle: bookingData.uniRepJobTitle,
83                         uniRepEmail: bookingData.uniRepEmail,
84                         uniRegion: bookingData.uniRegion,
85                         logoUrl: bookingData.logoUrl,
86                     },
87
88                     axios.post('http://localhost:5000/booking/create',
89             bookingDateInfoInstance)
90                     .then(()=>{
91                         // eslint-disable-next-line
92                         getDisplayData()
93                         .then((res)=>{
94                             setBooking(res.data);
95                         })
96                     }).then(setShow(false));
97                 } else{
98                     alert('Invalid authentication token')
99                 }
100             }
101
102         return (
103             <>
104                 <Modal show={show} onHide={cancel} backdrop="static">
105                     <Modal.Header closeButton>
106                         <Modal.Title>University Name</Modal.Title>

```

```

107          </Modal.Header>
108          <Modal.Body>
109          <Form>
110          <Form.Group>
111          <Form.Label> University Name </Form.Label>
112          <Form.Control
113              type='text'
114              onChange={(e) => {
115                  setBookingData({...bookingData,
116                  uniName: e.target.value,})
117              }
118              isValid={validated?!error.uniName:
119              false}
120              isInvalid={validated?!!error.uniName:
121              false}
122          </>
123          </Form.Group>
124          <Form.Group>
125          <Form.Label> University Region </Form.
126          Label>
127          <Form.Control
128              type='text'
129              onChange={(e) => {
130                  setBookingData({...bookingData,
131                  uniRegion: e.target.value,})
132              }
133              isValid={validated?!error.uniRegion:
134              false}
135              isInvalid={validated?!!error.
136              uniRegion:false}
137          </>
138          </Form.Group>
139          <Form.Group>
140          <Form.Label> Date </Form.Label>
141          <Form.Control
142              type='text'
143              onChange={(e)=> {
144                  setDate(new Date(dayjs(e.target.
145                  value)))
146              }
147              isValid={validated?!error.date:
148              false}
149              isInvalid={validated?!!error.date:
150              false}
151          </>
152          </Form.Group>
153          <Form.Group>
154          <Form.Label> Rep Title </Form.Label>
155          <Form.Control
156              type='text'
157              onChange={(e) => {
158                  setBookingData({...bookingData,
159                  uniRepJobTitle: e.target.value,})
160              }
161              isValid={validated?!error.
162              uniRepJobTitle:false}

```

```

152                               isValid={validated?!error.
  uniRepJobTitle:false}
153                               />
154                         </Form.Group>
155                         <Form.Group>
156                           <Form.Label> Rep Name </Form.Label>
157                           <Form.Control
158                             type='text'
159                             onChange={(e) => {
160                               setBookingData({...bookingData,
  uniRepName: e.target.value,})
161                             }})
162                           isValid={validated?!error.
  uniRepName:false}
163                           isValid={validated?!error.uniRepName:
  false}
164                         />
165                         </Form.Group>
166                         <Form.Group>
167                           <Form.Label> Rep Email </Form.Label>
168                           <Form.Control
169                             type='text'
170                             onChange={(e) => {
171                               setBookingData({...bookingData,
  uniRepEmail: e.target.value,})
172                             }})
173                           isValid={validated?!error.
  uniRepEmail:false}
174                           isValid={validated?!error.uniRepEmail
  :false}
175                         />
176                         </Form.Group>
177                         <Form.Group>
178                           <Form.Label> Image URL </Form.Label>
179                           <Form.Control
180                             type='text'
181                             onChange={(e) => {
182                               setBookingData({...bookingData,
  logoUrl: e.target.value,})
183                             }})
184                           isValid={validated?true:false}
185                         />
186                         </Form.Group>
187                       </Form>
188                     </Modal.Body>
189                     <Modal.Footer>
190                       <Button variant="success" onClick={()=>{create
  ()}} >
191                         Create Booking
192                       </Button>
193                     </Modal.Footer>
194                   </Modal>
195                   <div className='col-4 pad'>
196                     <Card style={{ width: '100%' }}>
197                       <Card.Body>
198                         <Card.Title className='cardUni'>University
Name</Card.Title>

```

```
199          <Card.Subtitle className='cardRegion'>
200            University Region</Card.Subtitle>
201          </Card.Body>
202          <Card.Img variant="top" src={image} className='
203            cardImage' />
204          <Card.Body>
205            <Card.Text className='cardDate'>Date</Card.
206            Text>
207            <Card.Text className='cardTitle'>Rep Title:
208            Rep Name</Card.Text>
209            <Card.Text className='cardEmail'>Rep Email</
210            Card.Text>
211            <Button variant="success" className="
212              cardButton" onClick={() => {handleOpen()}}>Add</Button>
213          </Card.Body>
214        </div>
215      </>
216    )
217  }
218
219  export default BCardAdd
220
221
222
```

```
1 // Iterate through bookings and display booking cards and the add
  booking card
2
3 import {useEffect, useState} from "react";
4 import axios from "axios";
5 import '../admin.css'
6 import BCard from "./BCard";
7 import BCardAdd from "./BCardAdd";
8
9 const EditBookings = () => {
10     const [booking, setBooking] = useState([])
11
12     // eslint-disable-next-line
13     useEffect(async () => {
14         async function getDisplayData() { // Async to prevent stalling
15
16             let res = await axios.get('http://localhost:5000/booking/
  readfordisplay');
17             return res;
18         }
19
20         getDisplayData().then((res)=>setBooking(res.data))
21     }, [])
22
23     return (
24         <>
25             <div className='container'>
26                 <div className='row pad'>
27                     {booking.map(instance=>{
28                         return(
29                             <BCard instance={instance} setBooking={
30                                 setBooking}>
31                             )
32                         })
33                     <BCardAdd setBooking={setBooking}/>
34                 </div>
35             </div>
36         </>
37     );
38 }
39
40 export default EditBookings;
```

```
1 // Validation
2
3 export default function validateInfo(bookingData, date) {
4     console.log(bookingData)
5     let errors = {};
6     const regex =
7         /^(([^\<><()[]\.,;:\s@"]+(\.[^\<><()[]\.,;:\s@"]+)*|(.+))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-\-0-9]+\.)+[a-zA-Z]{2,}))/;
8
9     if (isNaN(Date.parse(date))) {
10         errors.date = 'Invalid Date';
11     }
12     if (bookingData.uniName === '' || bookingData.uniName === 'University Name') {
13         errors.uniName = 'University name required';
14     }
15     if (bookingData.uniRepName === '' || bookingData.uniRepName === 'Rep Name') {
16         errors.uniRepName = 'Name required';
17     }
18     if (bookingData.uniRepJobTitle === '' || bookingData.uniRepJobTitle === 'Rep Title') {
19         errors.uniRepJobTitle = 'Job title required';
20     }
21     if (bookingData.uniRepEmail === '' || bookingData.uniRepEmail === 'Rep Email') {
22         errors.uniRepEmail = 'Email required';
23     }
24     } else if (!regex.test(bookingData.uniRepEmail)) {
25         errors.uniRepEmail = 'Invalid email';
26     }
27     if (bookingData.uniRegion === '' || bookingData.uniRegion === 'University Region') {
28         errors.uniRegion = 'Region required';
29     }
30
31     console.log(errors)
32     return errors;
33 }
34
```

```
1 abbr[title] {  
2   border-bottom: none !important;  
3   cursor: inherit !important;  
4   text-decoration: none !important;  
5 }  
6 .react-calendar {  
7   width: 40rem;  
8   max-width: 100%;  
9   background: white;  
10  /* border: 1px solid #a0a096; */  
11  font-family: "system-ui";  
12  line-height: 1.125em;  
13 }  
14 .react-calendar--doubleView {  
15   width: 700px;  
16 }  
17 .react-calendar--doubleView .react-calendar__viewContainer {  
18   display: flex;  
19   margin: -0.5em;  
20 }  
21 .react-calendar--doubleView .react-calendar__viewContainer > * {  
22   width: 50%;  
23   margin: 0.5em;  
24 }  
25 .react-calendar,  
26 .react-calendar *,  
27 .react-calendar *:before,  
28 .react-calendar *:after {  
29   -moz-box-sizing: border-box;  
30   -webkit-box-sizing: border-box;  
31   box-sizing: border-box;  
32 }  
33 .react-calendar button {  
34   margin: 0;  
35   border: 0;  
36   outline: none;  
37 }  
38 .react-calendar button:enabled:hover {  
39   cursor: pointer;  
40 }  
41 .react-calendar__navigation {  
42   height: 44px;  
43   margin-bottom: 1em;  
44 }  
45 .react-calendar__navigation button {  
46   min-width: 44px;  
47   background: none;  
48 }  
49 .react-calendar__navigation button:enabled:hover,  
50 .react-calendar__navigation button:enabled:disabled {  
51   background-color: #e6e6e6;  
52 }  
53 /* .react-calendar__navigation button[disabled] {  
54   background-color: #f0f0f0; */  
55 /*} */  
56 .react-calendar__month-view__weekdays {  
57   text-align: center;
```

```
58  text-transform: uppercase;
59  font-weight: bold;
60  font-size: 0.75em;
61
62 }
63 .react-calendar__month-view__weekdays__weekday {
64  padding: 0.5em;
65 }
66 .react-calendar__month-view__weekNumbers {
67  font-weight: bold;
68 }
69 .react-calendar__month-view__weekNumbers .react-calendar__tile {
70  display: flex;
71  align-items: center;
72  justify-content: center;
73  font-size: 0.75em;
74  padding: calc(0.75em / 0.75) calc(0.5em / 0.75);
75 }
76 .react-calendar__month-view__days__day--weekend {
77  color: #d10000;
78 }
79 /* .react-calendar__month-view__days__day--neighboringMonth {
80  color: #757575;
81 } */
82 .react-calendar__year-view .react-calendar__tile,
83 .react-calendar__decade-view .react-calendar__tile,
84 .react-calendar__century-view .react-calendar__tile {
85  padding: 2em 0.5em;
86 }
87 .react-calendar__tile {
88  max-width: 100%;
89  text-align: center;
90  padding: 0.75em 0.5em;
91  background: none;
92 }
93 .react-calendar__tile:disabled {
94  background-color: #f0f0f0;
95 }
96 .react-calendar__tile:enabled:hover,
97 .react-calendar__tile:enabled:focus {
98  background-color: #fff;
99 }
100 .react-calendar__tile--now {
101  background: #ffff76;
102 }
103 .react-calendar__tile--now:enabled:hover,
104 .react-calendar__tile--now:enabled:focus {
105  background: #ffffa9;
106 }
107 .react-calendar__tile--hasActive {
108  background: #76baff;
109 }
110 .react-calendar__tile--hasActive:enabled:hover,
111 .react-calendar__tile--hasActive:enabled:focus {
112  background: #a9d4ff;
113 }
114 .react-calendar__tile--active {
```

```
115  background: #006dc;
116  color: white;
117 }
118 .react-calendar__tile--active:enabled:hover,
119 .react-calendar__tile--active:enabled:focus {
120  background: #1087ff;
121 }
122 .react-calendar--selectRange .react-calendar__tile--hover {
123  background-color: #e6e6e6;
124 }
125
```

```

1 // Calendar to add or remove available days
2
3 import Calendar from 'react-calendar';
4 import Button from 'react-bootstrap/Button'
5 import './Calendar.css'
6 import { useState } from 'react';
7 import { useEffect } from 'react';
8 import dayjs from 'dayjs';
9 import axios from 'axios'
10
11 const EditCalendar = () => {
12     const [dates, setDates] = useState({available:[],booked:[{}]})
13     const [tempDates, setTempDates] = useState({})
14     const [changeLog, setChangeLog] = useState({})
15
16     async function resetCalendar(){
17         async function getRegionData() { // Async to prevent stalling
18             let response = await axios.get('http://localhost:5000/
booking/readforform');
19             return response;
20         }
21
22         // eslint-disable-next-line
23         let calendarData = (await getRegionData()).data;
24
25         let available = [];
26         let booked = [];
27
28         calendarData.forEach((instance) => { // Simple calendar
formatting
29             if (instance.status === 'Available') {
30                 available.push(dayjs(instance.aptDate).format('DD/MM/
YYYY'));
31
32             } else if (instance.status === 'Booked') {
33                 booked.push(dayjs(instance.aptDate).format('DD/MM/YYYY
')));
34             }
35         });
36
37         return {available:available,booked:booked}
38     }
39
40
41     useEffect(async ()=>{
42         let res = await resetCalendar()
43         setTempDates(res.available);
44         setDates(res);
45     }, [])
46
47     async function handleSubmit(){
48         const token = localStorage.getItem('adminToken')
49         let verify;
50
51         if(!token) {
52             verify = await axios.post('http://localhost:5000/login/
verify', {adminToken:token});

```

```

53         } else {
54             verify = false;
55         }
56
57         if(verify){
58             let submitData = {
59                 changeLog: changeLog,
60                 adminToken: token,
61             }
62
63             axios.post('http://localhost:5000/booking/editcalendar',
64             submitData)
65             setChangeLog({})
66         } else {
67             alert('Invalid authentication token')
68         }
69
70     return (
71         <div className='ccontainer'>
72             <Calendar
73                 onChange={(e)=>{
74                     let changeDate = dayjs(e).format('DD/MM/YYYY')
75
76                     if(dates.available.includes(changeDate)){ // Remove
77
78                         let Arr = dates.available // Get the available array from dates
79                         let newArray = Arr.filter(day=>day!==changeDate) // Remove the selected date from newArray
80
81                         setDates({...dates,available:newArray}) // Set dates to the new filtered array
82
83                         // If the original array already had the selected date as unavailalbe then remove the changes from the changelog
84                         // This prevents accidentally trying to push duplicate changes
85                         if(tempDates.includes(changeDate)){
86                             setChangeLog({...changeLog, [e]:false})
87                         } else {
88                             let newChanges = changeLog
89                             delete newChanges[e]
90                             setChangeLog(newChanges)
91                         }
92
93                     } else { // Add
94                         setDates({...dates,available:[...dates.available,changeDate]}) // Add to dates
95
96                         if(tempDates.includes(changeDate)){
97                             let newChanges = changeLog
98                             delete newChanges[e]
99                             setChangeLog(newChanges)
100                         } else {

```

```
101                     setChangeLog({...changeLog, [e]:true})
102                 }
103             }
104         }
105
106         tileClassName={({ date, view }) => {
107             if (dates.available.includes(dayjs(date).format('
108             DD/MM/YYYY')))) {
109                 return 'cavailable';
110             } else if (dates.booked.includes(dayjs(date).format('
111             DD/MM/YYYY')))) {
112                 return 'cbooked';
113             }
114         }
115         tileDisabled={({ date, view }) => {
116             if (date>new Date()) {
117                 return false;
118             } else {
119                 return true;
120             }
121         }
122         value=''
123         showNeighboringMonth='false'
124     />
125     <Button
126     disabled={Object.keys(changeLog).length === 0}
127     onClick={()=>{handleSubmit()}}>Save changes</Button>
128     <Button
129     disabled={Object.keys(changeLog).length === 0}
130     onClick={async ()=>{
131         let res = await resetCalendar();
132         setTempDates(res.available);
133         setDates(res);
134     }
135
136     }>Cancel</Button>
137   </div>
138 )
139 }
140
141 export default EditCalendar
142
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6     <meta name="viewport" content="width=device-width, initial-scale=1
" />
7     <meta name="theme-color" content="#000000" />
8     <meta
9       name="description"
10      content="Web site created using create-react-app"
11    />
12    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
14
15    <title>React App</title>
16  </head>
17  <body>
18    <noscript>You need to enable JavaScript to run this app.</noscript
>
19    <div id="root">
20
21    </div>
22  </body>
23 </html>
24
```

```
1 # https://www.robotstxt.org/robotstxt.html
2 User-agent: *
3 Disallow:
4
```

```
1 {
2   "short_name": "React App",
3   "name": "Create React App Sample",
4   "icons": [
5     {
6       "src": "",
7       "sizes": "64x64 32x32 24x24 16x16",
8       "type": "image/x-icon"
9     },
10    {
11      "src": "",
12      "type": "image/png",
13      "sizes": "192x192"
14    },
15    {
16      "src": "",
17      "type": "image/png",
18      "sizes": "512x512"
19    }
20  ],
21  "start_url": ".",
22  "display": "standalone",
23  "theme_color": "#000000",
24  "background_color": "#ffffff"
25 }
```

```
1 PORT=5000
2 ATLAS_URI=mongodb+srv://userNirav:userNirav8821@cluster0.k3lyw.mongodb
 .net/counsellorWebsite?retryWrites=true&w=majority
3 MAIL_USERNAME=bishvirtualsignup@gmail.com
4 OAUTH_CLIENTID=549431409213-a77901fhj77v8bjmgt6df6jbikm9tak0.apps.
 googleusercontent.com
5 OAUTH_CLIENT_SECRET=byzmdnWwZnSyyXoONTFgaTIh
6 OAUTH_REFRESH_TOKEN=1//04g53NsW8lntqCgYIARAAGAQSNwF-
 L9IrGGn4g1X15SXPjj0HYp5g-
 nnaFChWVID01mPr02tA344qmb2V6yBNHVeJ2K7YVmjHFpw
7 OAUTH_ACCESS_TOKEN=ya29.A0ARrdaM-
 WyL_ZdBFiZuzwzp9BLRCRGKAE9SImtKVgdQMj10vjAphQJkdn_LvMaZ73XdVdIcQ7zWGsZ
 KBz3fqMptmpXNGEv4oqnx8BIsdsAuCzWRJojcom9_-bnULd-D1XT04j324S-
 1FLnopUXRXPEyFZENPr
8 JWT_SECRET=376*QWe@4i5&@7!dp3!&Jg7As7N9p$Bji2^^
9 ADMIN_PASSWORD=123A
10 IMAGE_API_KEY=
 5ef594258facaf6d70f45f2abb7f91f07ef1e90b03d3b98e1843e345106fc6c4
```

```
1 // Setting up server
2
3 import express from 'express';
4 import mongoose from 'mongoose';
5 import cors from 'cors';
6 import dotenv from 'dotenv';
7
8 dotenv.config();
9
10 const app = express();
11
12 // Some middleware
13 app.use(express.urlencoded({ extended: true }));
14 app.use(express.json({ extended: true }));
15 app.use(cors());
16
17 // Routes
18 import bookingRouter from './API/routes/booking.js';
19 import counsRouter from './API/routes/couns.js';
20 import loginRouter from './API/routes/login.js'
21
22 app.use('/booking', bookingRouter);
23 app.use('/couns', counsRouter);
24 app.use('/login', loginRouter)
25 app.use('*', (req, res) => res.status(404).json({ error: 'Page not
  found' }));
26
27 // Connect to database
28 const url = process.env.ATLAS_URI;
29 mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology:
  true, 'useCreateIndex': true});
30 const connection = mongoose.connection;
31 connection.once('open', () => {console.log('Connected to MongoDB');});
32
33 // Start listening
34 const PORT = process.env.PORT || 5000;
35 app.listen(PORT, () => console.log(`Server running on port: ${PORT
  }`));
36
37 mongoose.set('useFindAndModify', false);
38
```

```
1 # Docker file for server
2
3 FROM node:14.17.4
4
5 WORKDIR /app
6
7 COPY package*.json ./
8
9 RUN npm i
10
11 COPY . .
12
13 EXPOSE 5000
14
15 CMD ["npm", "start"]
16
```

```
1 {
2   "name": "server",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "type": "module",
7   "scripts": {
8     "start": "nodemon index.js"
9   },
10  "keywords": [],
11  "author": "",
12  "license": "ISC",
13  "dependencies": {
14    "axios": "^0.24.0",
15    "cors": "^2.8.5",
16    "dotenv": "^10.0.0",
17    "express": "^4.17.1",
18    "googleapis": "^84.0.0",
19    "jsonwebtoken": "^8.5.1",
20    "mongoose": "^5.13.5",
21    "nodemailer": "^6.6.3",
22    "nodemon": "^2.0.12"
23  }
24 }
25
```

```
1 node_modules
2
```

```
1 // schematic for a booking record
2
3 import mongoose from 'mongoose';
4 mongoose.pluralize(null); // Prevent database from pluralizing
5
6 const schema = mongoose.Schema({
7     aptDate: { type: Date, required: true, unique: true },
8     status: { type: String, required: true },
9     booking: {
10         uniName: String,
11         uniRepName: String,
12         uniRepJobTitle: String,
13         uniRepEmail: String,
14         uniRegion: String,
15         logoUrl: String,
16     },
17 });
18
19 const bookingDateInfo = mongoose.model('bookingDateInfo', schema);
20
21 export default bookingDateInfo;
22
```

```
1 // schematic for a counselor record
2
3 import mongoose from 'mongoose';
4 mongoose.pluralize(null); // Prevent database from pluralizing
5
6 const schema = mongoose.Schema({
7     name: String,
8     counsEmail: {type:String, required:true, unique: true},
9     receiveEmail: {type:Boolean ,required:true}
10 });
11
12 const counsellorEmail = mongoose.model('counsellorEmail', schema);
13
14 export default counsellorEmail;
15
```

```

1 // Counselor route, featuring all endpoints
2
3 import express from 'express';
4 import counsellorEmail from '../models/counsellorEmail.model.js';
5 import jwt from 'jsonwebtoken';
6 import dotenv from 'dotenv';
7
8 dotenv.config();
9
10 const router = express.Router();
11
12 // Read
13 router.get('/read', (req, res) => {
14     counsellorEmail.find()
15         .then((counsellorEmail) => res.json(counsellorEmail))
16         .catch((err) => res.status(400).json('Error: ' + err));
17 });
18
19 router.patch('/edit', (req, res) => {
20     let token = req.body.adminToken
21
22     if (!!token && jwt.verify(token, process.env.JWT_SECRET)){
23         counsellorEmail.findById(req.body.id)
24             .then((counsellorEmail) => {
25                 counsellorEmail.name = req.body.name;
26                 counsellorEmail.counsEmail = req.body.counsEmail;
27                 counsellorEmail.receiveEmail = req.body.receiveEmail;
28
29
30                 counsellorEmail.save()
31                     .then(() => res.json('Couns updated'))
32                     .catch((err) => res.status(400).json('Error1: ' + err
33             )));
34             .catch((err) => res.status(400).json('Error2: ' + err));
35     } else {
36         res.status(403).json('Invalid Token')
37     }
38 });
39
40 // Create
41 router.post('/create', (req, res) => {
42     const name = req.body.name;
43     const counsEmail = req.body.counsEmail;
44     const receiveEmail = req.body.receiveEmail;
45     let token = req.body.adminToken
46
47     if (!!token && jwt.verify(token, process.env.JWT_SECRET)){
48         const newCounsEmail = new counsellorEmail({
49             name,
50             counsEmail,
51             receiveEmail,
52         });
53
54         newCounsEmail.save()
55             .then(() => res.json('Counsellor email added'))
56             .catch((err) => res.status(400).json('Error: ' + err));

```

```
57     } else {
58         res.status(403).json('Invalid Token')
59     }
60 });
61
62 router.delete('/remove', (req, res) => {
63     let token = req.body.adminToken
64
65     if(!token && jwt.verify(token, process.env.JWT_SECRET)){
66         counsellorEmail.findByIdAndDelete(req.body.id)
67             .then(() => {res.json('Couns deleted');})
68             .catch((err) => {res.status(400).json('Error: ' + err)} );
69     } else {
70         res.status(403).json('Invalid Token')
71     }
72 });
73
74 export default router;
75
```

```
1 // Login route, featuring all endpoints
2
3 import express from 'express';
4 import dotenv from 'dotenv';
5 import jwt from 'jsonwebtoken'
6 import counsellorEmail from '../models/counsellorEmail.model.js';
7
8 dotenv.config();
9
10 const router = express.Router();
11
12 router.post('/login', async (req,res) =>{
13     const password = req.body.password;
14     const email = req.body.email;
15
16     const exists = await counsellorEmail.exists({counsEmail:email});
17
18     if (password === process.env.ADMIN_PASSWORD && exists){
19         const token = jwt.sign({email:email},process.env.JWT_SECRET,{expiresIn: 86400 });
20         res.status(200).json({adminToken:token});
21     } else{
22         res.status(200).json('FALSE');
23     }
24 });
25
26 router.post('/verify', (req,res) =>{
27     const token = req.body.adminToken;
28     if(!token){
29         const verified = jwt.verify(token, process.env.JWT_SECRET)
30         if (verified){
31             res.status(200).json(true);
32         } else {
33             res.status(200).json(false)
34         }
35     } else {
36         res.status(200).json(false)
37     }
38 });
39
40 export default router;
```

```

1 // Booking route, featuring all endpoints
2
3 import express, { query } from 'express';
4 import bookingDateInfo from '../models/bookingDateInfo.model.js';
5 import counsellorEmail from '../models/counsellorEmail.model.js';
6 import nodemailer from 'nodemailer';
7 import dotenv from 'dotenv';
8 import axios from 'axios'
9 import jwt from 'jsonwebtoken'
10
11 dotenv.config();
12
13 // Create nodemailer transporter with gmail authorization
14 let transporter = nodemailer.createTransport({
15     service: 'gmail',
16     auth: {
17         type: 'OAuth2',
18         user: 'bishvirtualsignup@gmail.com',
19         clientId: process.env.OAUTH_CLIENTID,
20         clientSecret: process.env.OAUTH_CLIENT_SECRET,
21         refreshToken: process.env.OAUTH_REFRESH_TOKEN,
22         accessToken: process.env.OAUTH_ACCESS_TOKEN
23     },
24     tls: {
25         rejectUnauthorized: false,
26     },
27 });
28
29 async function getLogo(uniName){
30     let query = (uniName.replace(/ /g,"+")) + "+Logo" // Create query
31     let queryString =
32         `https://serpapi.com/search.json?engine=google&q=${query}&
33         google_domain=google.com&gl=us&hl=en&safe=active&tbo=isch&api_key=${
34         process.env.IMAGE_API_KEY}`
35     let res = await axios.get(queryString) // Get result
36     let imageURL = await res.data.images_results[0].thumbnail // Filter for first result
37     return await imageURL
38 }
39
40 const router = express.Router();
41
42 // Get data for calendar formatting
43 router.get('/readforform', (req, res) => {
44     bookingDateInfo.find(
45         { aptDate: { $gte: new Date() } },
46         { aptDate: 1, status: 1, 'booking.uniRegion': 1 })
47         .sort('aptDate')
48         .then((bookingDateInfo) => res.json(bookingDateInfo))
49         .catch((err) => res.status(400).json('Error: ' + err));
50 });
51
52 // Create a booking
53 router.post('/create', (req, res) => {
54     const aptDate = req.body.aptDate;
55     const status = req.body.status;

```

```

54  const uniName = req.body.booking.uniName;
55  const uniRepName = req.body.booking.uniRepName;
56  const uniRepJobTitle = req.body.booking.uniRepJobTitle;
57  const uniRepEmail = req.body.booking.uniRepEmail;
58  const uniRegion = req.body.booking.uniRegion;
59
60  getLogo(uniName)
61  .then( logoUrl => {
62      const newBooking = {
63          aptDate: aptDate,
64          status: status,
65          booking: {
66              uniName,
67              uniRepName,
68              uniRepJobTitle,
69              uniRepEmail,
70              uniRegion,
71              logoUrl,
72          },
73      };
74
75      bookingDateInfo.findOneAndUpdate(
76          { aptDate: aptDate },
77          newBooking)
78      .then(() => {
79          res.json('Booking created');
80          let mailOptions = {
81              from: 'BISH Signup <bishvirtualsignup@gmail.com> ' ,
82              to: uniRepEmail,
83              subject: 'Your presentation information',
84              text: `Thank you for signing up ${uniRepName}!
85              Your presentation is at ${aptDate}` ,
86          };
87
88          transporter.sendMail(mailOptions, (err, info) => {
89              if (err) {
90                  console.log(err);
91              } else {
92                  console.log('Message sent');
93              }
94          });
95
96          counsellorEmail.find({ receiveEmail: true })
97          .then((counsList)=>{
98              counsList.forEach((couns)=>{
99                  let counsMailOptions = {
100                      from: 'BISH Signup <bishvirtualsignup@gmail.
com> ' ,
101                      to: couns.counsEmail,
102                      subject: 'Booking Created',
103                      text: `${uniRepName} has just registered for
the: ${uniName}!
104                      The date they registered for is: ${aptDate}
105                      Representative's job title: ${uniRepJobTitle}
106                      Representative's email: ${uniRepEmail}
107                      University's region: ${uniRegion}` ,
108                  }

```

```

109
110             transporter.sendMail(counsMailOptions, (err, info
111             ) => {
112                 if (err) {
113                     console.log(err);
114                 } else {
115                     console.log('Message sent');
116                 }
117             });
118         });
119     })
120     .catch((err) => res.status(400).json('Error: ' + err));
121 }
122 });
123
124
125 // Get data for display
126 router.get('/readfordisplay', (req, res) => {
127     bookingDateInfo.find(
128         { aptDate: { $gte: new Date() }, status:"Booked" },
129         { aptDate: 1, 'booking': 1 })
130         .sort('aptDate')
131     .then((bookingDateInfo) => res.json(bookingDateInfo))
132     .catch((err) => res.status(400).json('Error: ' + err));
133 });
134
135 // Edit a booking
136 router.patch('/editbooking', (req, res) => {
137     let allowMail = req.body.allowMail;
138     let token = req.body.adminToken
139
140     if(!token && jwt.verify(token, process.env.JWT_SECRET)){
141         bookingDateInfo.findById(req.body.id)
142             .then((bookingDateInfo) => {
143                 bookingDateInfo.booking.uniName = req.body.booking.
144                 uniName;
145                 bookingDateInfo.booking.uniRepName = req.body.booking.
146                 uniRepName;
147                 bookingDateInfo.booking.uniRepJobTitle = req.body.booking.
148                 .uniRepJobTitle;
149                 bookingDateInfo.booking.uniRepEmail = req.body.booking.
150                 uniRepEmail;
151                 bookingDateInfo.booking.uniRegion = req.body.booking.
152                 uniRegion;
153                 bookingDateInfo.booking.logoUrl = req.body.booking.
154                 logoUrl;
155                 bookingDateInfo.aptDate = req.body.aptDate;
156
157                 bookingDateInfo.save()
158                     .then(() => {
159                         res.json('Booking updated')
160                         if(allowMail){
161                             let mailOptions = {
162                                 from: 'BISH Signup <bishvirtualsignup@gmail.
163                                 com> ',
164                                 to: req.body.booking.uniRepEmail,

```

```

158                     subject: 'Updated presentation information',
159                     text: `Your booking information for a
160                         presentation with BISH has changed, new information:
161                         University Name: ${req.body.booking.uniName}
162                         Name ${req.body.booking.uniRepName}
163                         Job title: ${req.body.booking.uniRepJobTitle}
164                         Representative's email: ${req.body.booking.
165                         uniRepEmail}
166                         University's region: ${req.body.booking.
167                         uniRegion}
168                         Presentation date: ${req.body.aptDate}`,
169                         );
170
171                         transporter.sendMail(mailOptions, (err, info
172                         ) => {
173                             if (err) {
174                                 console.log(err);
175                             } else {
176                                 console.log('Message sent');
177                             }
178                         });
179
180                         counsellorEmail.find({ receiveEmail: true })
181                         .then((counsList)=>{
182                             counsList.forEach((couns)=>{
183                                 let counsMailOptions = {
184                                     from: 'BISH Signup <bishvirtualsignup
185                                     @gmail.com> ',
186                                     to: couns.counsEmail,
187                                     subject: 'Updated booking information
188                                     ',
189                                     text: `Updates have been made to a
190                         registered booking, new information:
191                         University Name: ${req.body.booking.
192                         uniName}
193                         Name ${req.body.booking.uniRepName}
194                         Job title: ${req.body.booking.
195                         uniRepJobTitle}
196                         Representative's email: ${req.body.
197                         booking.uniRepEmail}
198                         University's region: ${req.body.
199                         booking.uniRegion}
200                         Presentation date: ${req.body.aptDate
201                         }
202                         Logo URL: ${req.body.booking.logoUrl
203                         },
204                         }
205
206                         transporter.sendMail(counsMailOptions, (
207                             err, info) => {
208                                 if (err) {
209                                     console.log(err);
210                                 } else {
211                                     console.log('Message sent');
212                                 }
213                             });
214                         });
215
216                         }
217
218                         }
219
220                         }
221
222                         }
223
224                         }
225
226                         }
227
228                         }
229
230                         }
231
232                         }
233
234                         }
235
236                         }
237
238                         }
239
240                         }
241
242                         }
243
244                         }
245
246                         }
247
248                         }
249
250                         }
251
252                         }
253
254                         }
255
256                         }
257
258                         }
259
259                         }
260
261                         }
262
263                         }
264
265                         }
266
267                         }
268
269                         }
269                         }
270
271                         }
272
273                         }
274
275                         }
276
277                         }
278
279                         }
279                         }
280
281                         }
282
283                         }
284
285                         }
286
287                         }
288
289                         }
289                         }
290
291                         }
292
293                         }
294
295                         }
296
297                         }
298
298                         }
299
299                         }
300
300                         }
301
301                         }
302
302                         }
303
303                         }
304
304                         }
305
305                         }
306
306                         }
307
307                         }
308
308                         }
309
309                         }
310
310                         }
311
311                         }
312
312                         }
313
313                         }
314
314                         }
315
315                         }
316
316                         }
317
317                         }
318
318                         }
319
319                         }
320
320                         }
321
321                         }
322
322                         }
323
323                         }
324
324                         }
325
325                         }
326
326                         }
327
327                         }
328
328                         }
329
329                         }
330
330                         }
331
331                         }
332
332                         }
333
333                         }
334
334                         }
335
335                         }
336
336                         }
337
337                         }
338
338                         }
339
339                         }
340
340                         }
341
341                         }
342
342                         }
343
343                         }
344
344                         }
345
345                         }
346
346                         }
347
347                         }
348
348                         }
349
349                         }
350
350                         }
351
351                         }
352
352                         }
353
353                         }
354
354                         }
355
355                         }
356
356                         }
357
357                         }
358
358                         }
359
359                         }
360
360                         }
361
361                         }
362
362                         }
363
363                         }
364
364                         }
365
365                         }
366
366                         }
367
367                         }
368
368                         }
369
369                         }
370
370                         }
371
371                         }
372
372                         }
373
373                         }
374
374                         }
375
375                         }
376
376                         }
377
377                         }
378
378                         }
379
379                         }
380
380                         }
381
381                         }
382
382                         }
383
383                         }
384
384                         }
385
385                         }
386
386                         }
387
387                         }
388
388                         }
389
389                         }
390
390                         }
391
391                         }
392
392                         }
393
393                         }
394
394                         }
395
395                         }
396
396                         }
397
397                         }
398
398                         }
399
399                         }
400
400                         }
401
401                         }
402
402                         }
403
403                         }
404
404                         }
405
405                         }
406
406                         }
407
407                         }
408
408                         }
409
409                         }
410
410                         }
411
411                         }
412
412                         }
413
413                         }
414
414                         }
415
415                         }
416
416                         }
417
417                         }
418
418                         }
419
419                         }
420
420                         }
421
421                         }
422
422                         }
423
423                         }
424
424                         }
425
425                         }
426
426                         }
427
427                         }
428
428                         }
429
429                         }
430
430                         }
431
431                         }
432
432                         }
433
433                         }
434
434                         }
435
435                         }
436
436                         }
437
437                         }
438
438                         }
439
439                         }
440
440                         }
441
441                         }
442
442                         }
443
443                         }
444
444                         }
445
445                         }
446
446                         }
447
447                         }
448
448                         }
449
449                         }
450
450                         }
451
451                         }
452
452                         }
453
453                         }
454
454                         }
455
455                         }
456
456                         }
457
457                         }
458
458                         }
459
459                         }
460
460                         }
461
461                         }
462
462                         }
463
463                         }
464
464                         }
465
465                         }
466
466                         }
467
467                         }
468
468                         }
469
469                         }
470
470                         }
471
471                         }
472
472                         }
473
473                         }
474
474                         }
475
475                         }
476
476                         }
477
477                         }
478
478                         }
479
479                         }
480
480                         }
481
481                         }
482
482                         }
483
483                         }
484
484                         }
485
485                         }
486
486                         }
487
487                         }
488
488                         }
489
489                         }
490
490                         }
491
491                         }
492
492                         }
493
493                         }
494
494                         }
495
495                         }
496
496                         }
497
497                         }
498
498                         }
499
499                         }
500
500                         }
501
501                         }
502
502                         }
503
503                         }
504
504                         }
505
505                         }
506
506                         }
507
507                         }
508
508                         }
509
509                         }
510
510                         }
511
511                         }
512
512                         }
513
513                         }
514
514                         }
515
515                         }
516
516                         }
517
517                         }
518
518                         }
519
519                         }
520
520                         }
521
521                         }
522
522                         }
523
523                         }
524
524                         }
525
525                         }
526
526                         }
527
527                         }
528
528                         }
529
529                         }
530
530                         }
531
531                         }
532
532                         }
533
533                         }
534
534                         }
535
535                         }
536
536                         }
537
537                         }
538
538                         }
539
539                         }
540
540                         }
541
541                         }
542
542                         }
543
543                         }
544
544                         }
545
545                         }
546
546                         }
547
547                         }
548
548                         }
549
549                         }
550
550                         }
551
551                         }
552
552                         }
553
553                         }
554
554                         }
555
555                         }
556
556                         }
557
557                         }
558
558                         }
559
559                         }
560
560                         }
561
561                         }
562
562                         }
563
563                         }
564
564                         }
565
565                         }
566
566                         }
567
567                         }
568
568                         }
569
569                         }
570
570                         }
571
571                         }
572
572                         }
573
573                         }
574
574                         }
575
575                         }
576
576                         }
577
577                         }
578
578                         }
579
579                         }
580
580                         }
581
581                         }
582
582                         }
583
583                         }
584
584                         }
585
585                         }
586
586                         }
587
587                         }
588
588                         }
589
589                         }
590
590                         }
591
591                         }
592
592                         }
593
593                         }
594
594                         }
595
595                         }
596
596                         }
597
597                         }
598
598                         }
599
599                         }
600
600                         }
601
601                         }
602
602                         }
603
603                         }
604
604                         }
605
605                         }
606
606                         }
607
607                         }
608
608                         }
609
609                         }
610
610                         }
611
611                         }
612
612                         }
613
613                         }
614
614                         }
615
615                         }
616
616                         }
617
617                         }
618
618                         }
619
619                         }
620
620                         }
621
621                         }
622
622                         }
623
623                         }
624
624                         }
625
625                         }
626
626                         }
627
627                         }
628
628                         }
629
629                         }
630
630                         }
631
631                         }
632
632                         }
633
633                         }
634
634                         }
635
635                         }
636
636                         }
637
637                         }
638
638                         }
639
639                         }
640
640                         }
641
641                         }
642
642                         }
643
643                         }
644
644                         }
645
645                         }
646
646                         }
647
647                         }
648
648                         }
649
649                         }
650
650                         }
651
651                         }
652
652                         }
653
653                         }
654
654                         }
655
655                         }
656
656                         }
657
657                         }
658
658                         }
659
659                         }
660
660                         }
661
661                         }
662
662                         }
663
663                         }
664
664                         }
665
665                         }
666
666                         }
667
667                         }
668
668                         }
669
669                         }
670
670                         }
671
671                         }
672
672                         }
673
673                         }
674
674                         }
675
675                         }
676
676                         }
677
677                         }
678
678                         }
679
679                         }
680
680                         }
681
681                         }
682
682                         }
683
683                         }
684
684                         }
685
685                         }
686
686                         }
687
687                         }
688
688                         }
689
689                         }
690
690                         }
691
691                         }
692
692                         }
693
693                         }
694
694                         }
695
695                         }
696
696                         }
697
697                         }
698
698                         }
699
699                         }
700
700                         }
701
701                         }
702
702                         }
703
703                         }
704
704                         }
705
705                         }
706
706                         }
707
707                         }
708
708                         }
709
709                         }
710
710                         }
711
711                         }
712
712                         }
713
713                         }
714
714                         }
715
715                         }
716
716                         }
717
717                         }
718
718                         }
719
719                         }
720
720                         }
721
721                         }
722
722                         }
723
723                         }
724
724                         }
725
725                         }
726
726                         }
727
727                         }
728
728                         }
729
729                         }
730
730                         }
731
731                         }
732
732                         }
733
733                         }
734
734                         }
735
735                         }
736
736                         }
737
737                         }
738
738                         }
739
739                         }
740
740                         }
741
741                         }
742
742                         }
743
743                         }
744
744                         }
745
745                         }
746
746                         }
747
747                         }
748
748                         }
749
749                         }
750
750                         }
751
751                         }
752
752                         }
753
753                         }
754
754                         }
755
755                         }
756
756                         }
757
757                         }
758
758                         }
759
759                         }
760
760                         }
761
761                         }
762
762                         }
763
763                         }
764
764                         }
765
765                         }
766
766                         }
767
767                         }
768
768                         }
769
769                         }
770
770                         }
771
771                         }
772
772                         }
773
773                         }
774
774                         }
775
775                         }
776
776                         }
777
777                         }
778
778                         }
779
779                         }
780
780                         }
781
781                         }
782
782                         }
783
783                         }
784
784                         }
785
785                         }
786
786                         }
787
787                         }
788
788                         }
789
789                         }
790
790                         }
791
791                         }
792
792                         }
793
793                         }
794
794                         }
795
795                         }
796
796                         }
797
797                         }
798
798                         }
799
799                         }
800
800                         }
801
801                         }
802
802                         }
803
803                         }
804
804                         }
805
805                         }
806
806                         }
807
807                         }
808
808                         }
809
809                         }
810
810                         }
811
811                         }
812
812                         }
813
813                         }
814
814                         }
815
815                         }
816
816                         }
817
817                         }
818
818                         }
819
819                         }
820
820                         }
821
821                         }
822
822                         }
823
823                         }
824
824                         }
825
825                         }
826
826                         }
827
827                         }
828
828                         }
829
829                         }
830
830                         }
831
831                         }
832
832                         }
833
833                         }
834
834                         }
835
835                         }
836
836                         }
837
837                         }
838
838                         }
839
839                         }
840
840                         }
841
841                         }
842
842                         }
843
843                         }
844
844                         }
845
845                         }
846
846                         }
847
847                         }
848
848                         }
849
849                         }
850
850                         }
851
851                         }
852
852                         }
853
853                         }
854
854                         }
855
855                         }
856
856                         }
857
857                         }
858
858                         }
859
859                         }
860
860                         }
861
861                         }
862
862                         }
863
863                         }
864
864                         }
865
865                         }
866
866                         }
867
867                         }
868
868                         }
869
869                         }
870
870                         }
871
871                         }
872
872                         }
873
873                         }
874
874                         }
875
875                         }
876
876                         }
877
877                         }
878
878                         }
879
879                         }
880
880                         }
881
881                         }
882
882                         }
883
883                         }
884
884                         }
885
885                         }
886
886                         }
887
887                         }
888
888                         }
889
889                         }
890
890                         }
891
891                         }
892
892                         }
893
893                         }
894
894                         }
895
895                         }
896
896                         }
897
897                         }
898
898                         }
899
899                         }
900
900                         }
901
901                         }
902
902                         }
903
903                         }
904
904                         }
905
905                         }
906
906                         }
907
907                         }
908
908                         }
909
909                         }
910
910                         }
911
911                         }
912
912                         }
913
913                         }
914
914                         }
915
915                         }
916
916                         }
917
917                         }
918
918                         }
919
919                         }
920
920                         }
921
921                         }
922
922                         }
923
923                         }
924
924                         }
925
925                         }
926
926                         }
927
927                         }
928
928                         }
929
929                         }
930
930                         }
931
931                         }
932
932                         }
933
933                         }
934
934                         }
935
935                         }
936
936                         }
937
937                         }
938
938                         }
939
939                         }
940
940                         }
941
941                         }
942
942                         }
943
943                         }
944
944                         }
945
945                         }
946
946                         }
947
947                         }
948
948                         }
949
949                         }
950
950                         }
951
951                         }
952
952                         }
953
953                         }
954
954                         }
955
955                         }
956
956                         }
957
957                         }
958
958                         }
959
959                         }
960
960                         }
961
961                         }
962
962                         }
963
963                         }
964
964                         }
965
965                         }
966
966                         }
967
967                         }
968
968                         }
969
969                         }
970
970                         }
971
971                         }
972
972                         }
973
973                         }
974
974                         }
975
975                         }
976
976                         }
977
977                         }
978
978                         }
979
979                         }
980
980                         }
981
981                         }
982
982                         }
983
983                         }
984
984                         }
985
985                         }
986
986                         }
987
987                         }
988
988                         }
989
989                         }
990
990                         }
991
991                         }
992
992                         }
993
993                         }
994
994                         }
995
995                         }
996
996                         }
997
997                         }
998
998                         }
999
999                         }
1000
1000                         }
1001
1001                         }
1002
1002                         }
1003
1003                         }
1004
1004                         }
1005
1005                         }
1006
1006                         }
1007
1007                         }
1008
1008                         }
1009
1009                         }
1010
1010                         }
1011
1011                         }
1012
1012                         }
1013
1013                         }
1014
1014                         }
1015
1015                         }
1016
1016                         }
1017
1017                         }
1018
1018                         }
1019
1019                         }
1020
1020                         }
1021
1021                         }
1022
1022                         }
1023
1023                         }
1024
1024                         }
1025
1025                         }
1026
1026                         }
1027
1027                         }
1028
1028                         }
1029
1029                         }
1030
1030                         }
1031
1031                         }
1032
1032                         }
1033
1033                         }
1034
1034                         }
1035
1035                         }
1036
1036                         }
1037
1037                         }
1038
1038                         }
1039
1039                         }
1040
1040                         }
1041
1041                         }
1042
1042                         }
1043
1043                         }
1044
1044                         }
1045
1045                         }
1046
1046                         }
1047
1047                         }
1048
1048                         }
1049
1049                         }
1050
1050                         }
1051
1051                         }
1052
1052                         }
1053
1053                         }
1054
1054                         }
1055
1055                         }
1056
1056                         }
1057
1057                         }
1058
1058                         }
1059
1059                         }
1060
1060                         }
1061
1061                         }
1062
1062                         }
1063
1063                         }
1064
1064                         }
1065
1065                         }
1066
1066                         }
1067
1067                         }
1068
1068                         }
1069
1069                         }
1070
1070                         }
1071
1071                         }
1072
1072                         }
1073
1073                         }
1074
1074                         }
1075
1075                         }
1076
1076                         }
1077
1077                         }
1078
1078                         }
1079
1079                         }
1080
1080                         }
1081
1081                         }
1082
1082                         }
1083
1083                         }
1084
1084                         }
1085
1085                         }
1086
1086                         }
1087
1087                         }
1088
1088                         }
1089
1089                         }
1090
1090                         }
1091
1091                         }
1092
1092                         }
1093
1093                         }
1094
1094                         }
1095
1095                         }
1096
1096                         }
1097
1097                         }
1098
1098                         }
1099
1099                         }
1100
1100                         }
1101
1101                         }
1102
1102                         }
1103
1103                         }
1104
1104                         }
1105
1105                         }
1106
1106                         }
1107
1107                         }
1108
1108                         }
1109
1109                         }
1110
1110                         }
1111
1111                         }
1112
1112                         }
1113
1113                         }
1114
1114                         }
1115
1115                         }
1116
1116                         }
1117
1117                         }
1118
1118                         }
1119
1119                         }
1120
1120                         }
1121
1121                         }
1122
1122                         }
1123
1123                         }
1124
1124                         }
1125
1125                         }
1126
1126                         }
1127
1127                         }
1128
1128                         }
1129
1129                         }
1130
1130                         }
1131
1131                         }
1132
1132                         }
1133
1133                         }
1134
1134                         }
1135
1135                         }
1136
1136                         }
1137
1137                         }
1138
1138                         }
1139
1139                         }
1140
1140                         }
1141
1141                         }
1142
1142                         }
1143
1143                         }
1144
1144                         }
1145
1145                         }
1146
1146                         }
1147
1147                         }
1148
1148                         }
1149
1149                         }
1150
1150                         }
1151
1151                         }
1152
1152                         }
1153
1153                         }
1154
1154                         }
1155
1155                         }
1156
1156                         }
1157
1157                         }
1158
1158                         }
1159
1159                         }
1160
1160                         }
1161
1161                         }
1162
1162                         }
1163
1163                         }
1164
1164                         }
1165
1165                         }
1166
1166                         }
1167
1167                         }
1168
1168                         }
1169
1169                         }
1170
1170                         }
1171
1171                         }
1172
1172                         }
1173
1173                         }
1174
1174                         }
1175
1175                         }
1176
1176                         }
1177
1177                         }
1178
1178                         }
1179
1179                         }
1180
1180                         }
1181
1181                         }
1182
1182                         }
1183
1183                         }
1184
1184                         }
1185
1185                         }
1186
1186                         }
1187
1187                         }
1188
1188                         }
1189
1189                         }
1190
1190                         }
1191
1191                         }
1192
1192                         }
1193
1193                         }
1194
1194                         }
1195
1195                         }
1196
1196                         }
1197
1197                         }
1198
1198                         }
1199
1199                         }
1200
1200                         }
1201
1201                         }
1202
1202                         }
1203
1203                         }
1204
1204                         }
1205
1205                         }
1206
1206                         }
1207
1207                         }
1208
1208                         }
1209
1209                         }
1210
1210                         }
1211
1211                         }
1212
1212                         }
1213
1213                         }
1214
1214                         }
1215
1215                         }
1216
1216                         }
1217
1217                         }
1218
1218                         }
1219
1219                         }
1220
1220                         }
1221
1221                         }
1222
1222                         }
1223
1223                         }
1224
1224                         }
1225
1225                         }
1226
1226                         }
1227
1227                         }
1228
1228                         }
1229
1229                         }
1230
1230                         }
1231
1231                         }
1232
1232                         }
1233
1233                         }
1234
1234                         }
1235
1235                         }
1236
1236                         }
1237
1237                         }
1238
1238                         }
1239
1239                         }
1240
1240                         }
1241
1241                         }
1242
1242                         }
1243
1243                         }
1244
1244                         }
1245
1245                         }
1246
1246                         }
1247
1247                         }
1248
1248                         }
1249
1249                         }
1250
1250                         }
1251
1251                         }
1252
1252                         }
1253
1253                         }
1254
1254                         }
1255
1255                         }
1256
1256                         }
1257
1257                         }
1258
1258                         }
1259
1259                         }
1260
1260                         }
1261
1261                         }
1262
1262                         }
1263
1263                         }
1264
1264                         }
1265
1265                         }
1266
1266                         }
1267
1267                         }
1268
1268                         }
1269
1269                         }
1270
1270                         }
1271
1271                         }
1272
1272                         }
1273
1273                         }
1274
1274                         }
1275
1275                         }
1276
1276                         }
1277
1277                         }
1278
1278                         }
1279
1279                         }
1280
1280                         }
1281
1281                         }
1282
1282                         }
1283
1283                         }
1284
1284                         }
1285
1285                         }
1286
1286                         }
1287
1287                         }
1288
1288                         }
1289
1289                         }
1290
1290                         }
1291
1291                         }
1292
1292                         }
1293
1293                         }
1294
1294                         }
1295
1295                         }
1296
1296                         }
1297
1297                         }
1298
1298                         }
1299
1299                         }
1300
1300                         }
1301
1301                         }
1302
1302                         }
1303
1303                         }
1304
1304                         }
1305
1305                         }
1306
1306                         }
1307
1307                         }
1308
1308                         }
1309
1309                         }
1310
1310                         }
1311
1311                         }
1312
1312                         }
1313
1313                         }
1314
1314                         }
1315
1315                         }
1316
1316                         }
1317
1317                         }
1318
1318                         }
1319
1319                         }
1320
1320                         }
1321
1321                         }
1322
1322                         }
1323
1323                         }
1324
1324                         }
1325
1325                         }
1326
1326                         }
1327
1327                         }
1328
1328                         }
1329
1329                         }
1330
1330                         }
1331
1331                         }
1332
1332                         }
1333
1333                         }
1334
1334                         }
1335
1335                         }
1336
1336                         }
1337
1337                         }
1338
1338                         }
1339
1339                         }
1340
1340                         }
1341
1341                         }
1342
1342                         }
1343
1343                         }
1344
1344                         }
1345
1345                         }
1346
1346                         }
13
```

```

201             })
202         }
203     })
204     .catch((err) => res.status(400).json('Error: ' + err));
205   })
206   .catch((err) => res.status(400).json('Error: ' + err));
207 } else {
208   res.status(403).json('Invalid Token')
209 }
210 });
211
212 // Remove a booking
213 router.delete('/remove', (req, res) => {
214   let token = req.body.adminToken
215
216   if(!token && jwt.verify(token, process.env.JWT_SECRET)){
217     bookingDateInfo.findByIdAndDelete(req.body.id)
218     .then(() => {res.json('Booking deleted')})
219     .catch((err) => {res.status(400).json('Error: ' + err)});
220   } else {
221     res.status(403).json('Invalid Token')
222   }
223 });
224
225 // Edit dates
226 router.post('/editcalendar', (req, res) =>{
227   let changeLog = req.body.changeLog
228   let token = req.body.adminToken
229
230   if(!token && jwt.verify(token, process.env.JWT_SECRET)){
231     let changes = []
232     Object.keys(changeLog).forEach((key) => {
233       if(changeLog[key]==true){
234         changes.push({'insertOne':{"document":{aptDate:key,
235           status:"Available"}}})
236       } else {
237         changes.push({'deleteOne':{"filter":{aptDate:key}}})
238       }
239     });
240
241     bookingDateInfo.bulkWrite(changes)
242     .then(() => {
243       res.json('Dates updated')
244       counsellorEmail.find(
245         { receiveEmail: true },
246         { counsEmail: 1 }
247       )
248       .then((counsList)=>{
249         Object.keys(changeLog).forEach((key) => {
250           let text = '';
251           if(changeLog[key]==true){
252             text = `${text} \n ${key} - Added`
253           } else {
254             text = `${text} \n ${key} - Removed`
255           }
256         })
257       })
258     })
259   }
260 })

```

```

257             counsList.forEach((couns)=>{
258                 let counsMailOptions = {
259                     from: 'BISH Signup <bishvirtualsignup@gmail.
260                         com> ',
261                     to: couns.counsEmail,
262                     subject: 'Changes to available dates',
263                     text: `Updates have been made to the list of
264                         available dates:
265                         ${text}`,
266                     }
267
268                     transporter.sendMail(counsMailOptions, (err, info
269 ) => {
270                         if (err) {
271                             console.log(err);
272                         } else {
273                             console.log('Message sent');
274                         }
275                     });
276                     .catch((err) => {res.status(400).json('Error: ' + err)})
277 } else {
278     res.status(403).json('Invalid Token')
279 }
280 });
281
282
283 // Manually create a booking for insomnia
284 router.post('/createm', (req, res) => {
285     const aptDate = req.body.aptDate;
286     const status = req.body.status;
287     const uniName = req.body.booking.uniName;
288     const uniRepName = req.body.booking.uniRepName;
289     const uniRepJobTitle = req.body.booking.uniRepJobTitle;
290     const uniRepEmail = req.body.booking.uniRepEmail;
291     const uniRegion = req.body.booking.uniRegion;
292
293     getLogo(uniName).then( logoUrl => {
294         const newBooking = new bookingDateInfo({
295             aptDate: aptDate,
296             status: status,
297             booking: {
298                 uniName,
299                 uniRepName,
300                 uniRepJobTitle,
301                 uniRepEmail,
302                 uniRegion,
303                 logoUrl,
304             },
305         });
306
307         newBooking.save()
308             .then(() => res.json('Booking created'))
309             .catch((err) => res.status(400).json('Error: ' + err));
310     })

```

```
311 });
312
313 export default router;
314
```

```
1 // Electron portion of the app (Mostly boilerplate)
2
3 const { app, BrowserWindow } = require('electron')
4
5 const createWindow = () => { // Create an electron window
6     const win = new BrowserWindow({
7         width: 800,
8         height: 600
9     })
10
11     win.loadURL('http://localhost:3000') // Display the electron app
12     onto the window
13 }
14 app.whenReady().then(() => {
15     createWindow()
16 })
```

```
1 # Getting Started with Create React App
2
3 This project was bootstrapped with [Create React App](https://github.
4 com/facebook/create-react-app).
5
6 ## Available Scripts
7
8 In the project directory, you can run:
9
10 #### `npm start`
11 Runs the app in the development mode.\n
12 Open [http://localhost:3000](http://localhost:3000) to view it in your
13 browser.
14 The page will reload when you make changes.\n
15 You may also see any lint errors in the console.
16
17 #### `npm test`
18
19 Launches the test runner in the interactive watch mode.\n
20 See the section about [running tests](https://facebook.github.io/
21 create-react-app/docs/running-tests) for more information.
22 #### `npm run build`
23
24 Builds the app for production to the `build` folder.\n
25 It correctly bundles React in production mode and optimizes the build
26 for the best performance.
27 The build is minified and the filenames include the hashes.\n
28 Your app is ready to be deployed!
29
30 See the section about [deployment](https://facebook.github.io/create-
31 react-app/docs/deployment) for more information.
32 #### `npm run eject`
33
34 **Note: this is a one-way operation. Once you `eject`, you can't go
35 back!**
36 If you aren't satisfied with the build tool and configuration choices
37 , you can `eject` at any time. This command will remove the single
38 build dependency from your project.
39 Instead, it will copy all the configuration files and the transitive
40 dependencies (webpack, Babel, ESLint, etc) right into your project so
41 you have full control over them. All of the commands except `eject`
42 will still work, but they will point to the copied scripts so you can
43 tweak them. At this point you're on your own.
44 You don't have to ever use `eject`. The curated feature set is
45 suitable for small and middle deployments, and you shouldn't feel
46 obligated to use this feature. However we understand that this tool
47 wouldn't be useful if you couldn't customize it when you are ready for
48 it.
49
```

```
42 ## Learn More
43
44 You can learn more in the [Create React App documentation](https://facebook.github.io/create-react-app/docs/getting-started).
45
46 To learn React, check out the [React documentation](https://reactjs.org/).
47
48 ### Code Splitting
49
50 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)
51
52 ### Analyzing the Bundle Size
53
54 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)
55
56 ### Making a Progressive Web App
57
58 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)
59
60 ### Advanced Configuration
61
62 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)
63
64 ### Deployment
65
66 This section has moved here: [https://facebook.github.io/create-react-app/docs/deployment](https://facebook.github.io/create-react-app/docs/deployment)
67
68 ### `npm run build` fails to minify
69
70 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)
71
```

```
1 # See https://help.github.com/articles/ignoring-files/ for more about
  ignoring files.
2
3 # dependencies
4 /node_modules
5 /.pnp
6 .pnp.js
7
8 # testing
9 /coverage
10
11 # production
12 /build
13
14 # misc
15 .DS_Store
16 .env.local
17 .env.development.local
18 .env.test.local
19 .env.production.local
20
21 npm-debug.log*
22 yarn-debug.log*
23 yarn-error.log*
24
```

```

1  {
2    "name": "electron-app",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "@testing-library/jest-dom": "^5.16.2",
7      "@testing-library/react": "^12.1.4",
8      "@testing-library/user-event": "^13.5.0",
9      "axios": "^0.26.1",
10     "bootstrap": "^4.6.0",
11     "concurrently": "^7.0.0",
12     "cross-env": "^7.0.3",
13     "dayjs": "^1.10.8",
14     "electron": "^17.1.2",
15     "react": "^17.0.2",
16     "react-bootstrap": "^1.6.1",
17     "react-dom": "^17.0.2",
18     "react-scripts": "5.0.0",
19     "wait-on": "^6.0.1",
20     "web-vitals": "^2.1.4"
21   },
22   "main": "main.js",
23   "scripts": {
24     "start": "react-scripts start",
25     "build": "react-scripts build",
26     "test": "react-scripts test",
27     "eject": "react-scripts eject",
28     "watch": "webpack --config webpack.common.js --watch",
29     "electron:start": "electron .",
30     "electron-react": "concurrently \\\"cross-env BROWSER=none npm start \\\" \\\"wait-on http://localhost:3000 && electron .\\\""
31   },
32   "eslintConfig": {
33     "extends": [
34       "react-app",
35       "react-app/jest"
36     ]
37   },
38   "browserslist": {
39     "production": [
40       ">0.2%",
41       "not dead",
42       "not op_mini all"
43     ],
44     "development": [
45       "last 1 chrome version",
46       "last 1 firefox version",
47       "last 1 safari version"
48     ]
49   }
50 }
51

```

```
1 import Display from "./components/Display"
2
3 const App = () => {
4     return(
5         <div>
6             <Display/>
7         </div>
8     )
9 }
10
11 export default App
12
```

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './App';
5
6 ReactDOM.render(
7   <React.StrictMode>
8     <App />
9   </React.StrictMode>,
10  document.getElementById('root')
11 );
12
13
14
```

```
1 .cardUni{  
2     font-size: 1.5rem;  
3     font-weight: 300;  
4 }  
5  
6 .cardRegion{  
7     font-size: 1rem;  
8     font-weight: 500;  
9     color: #6c757d;  
10 }  
11  
12 .cardImage{  
13     width: calc(100% + 2px);  
14     border-radius: 0;  
15     margin-left: -1px;  
16     border-left: 1px solid #DFDFDF;  
17     border-right: 1px solid #DFDFDF;  
18 }  
19  
20 .cardDate{  
21 }  
22 }  
23  
24 .cardTitle{  
25 }  
26 }  
27  
28 .cardEmail{  
29 }  
30 }  
31  
32 .pad{  
33     margin-top: 1rem;  
34     margin-bottom: 1rem;  
35 }  
36  
37 .cardButton{  
38     width: 100%;  
39 }  
40  
41 .login-container{  
42     display: flex;  
43     justify-content: center;  
44     align-items: center;  
45 }  
46  
47 .ccontainer {  
48     margin-left: calc(50vw - 20rem);  
49     margin-top: calc(50vh - 15rem);  
50 }  
51  
52 .cbooked {  
53     color: hsla(3, 100%, 20%, 1) !important;  
54     background-color: rgba(255, 65, 54, 0.6) !important;  
55 }  
56  
57 .cbooked:hover {
```

```
58     color: hsla(3, 100%, 20%, 1) !important;
59     background-color: rgba(255, 65, 54, 0.55) !important;
60 }
61
62 .cavailable {
63     color: hsla(127, 63%, 15%, 1) !important;
64     background-color: rgba(1, 255, 112, 0.7) !important;
65 }
66
67 .cavailable:hover {
68     color: hsla(127, 63%, 15%, 1) !important;
69     background-color: rgba(1, 255, 112, 0.5) !important;
70 }
71
72 .topbar{
73     height: 10vh;
74     background: #f8f9fa;
75     position: absolute;
76     width:100%;
77     text-align: center;
78     padding-top: 2vh;
79 }
80
81 .botbar{
82     bottom: 0;
83     height: 10vh;
84     background: #f8f9fa;
85     position: absolute;
86     width:100%;
87 }
88
89 .vaf{
90     display: flex;
91     justify-content: center;
92     align-items: center;
93     padding-top: 15vh;
94 }
```

```
1 body {  
2   margin: 0;  
3   font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto'  
4   , 'Oxygen',  
5   'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue'  
6   ,  
7   sans-serif;  
8   -webkit-font-smoothing: antialiased;  
9   -moz-osx-font-smoothing: grayscale;  
10 }  
11  
12 code {  
13   font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New'  
14   ,  
15   monospace;  
16 }  
17  
18
```

```
1 // Booking card template
2
3 import {useState} from "react";
4 import dayjs from "dayjs";
5 import Card from 'react-bootstrap/Card'
6 import image from './undraw_Page_not_found_re_e9o6.png'
7 import '../admin.css'
8
9
10 const BCard = ({instance}) => {
11     const [date, setDate] = useState(instance.aptDate);
12     const [bookingData, setBookingData] = useState({
13         uniName: instance.booking.uniName,
14         uniRegion: instance.booking.uniRegion,
15         uniRepJobTitle: instance.booking.uniRepJobTitle,
16         uniRepName: instance.booking.uniRepName,
17         uniRepEmail: instance.booking.uniRepEmail,
18         logoUrl:instance.booking.logoUrl,
19     });
20
21     return (
22         <>
23             <div className='col-4 pad'>
24                 <Card style={{ width: '100%' }}>
25                     <Card.Body>
26                         <Card.Title className='cardUni'>{bookingData.uniName}
27                         <Card.Subtitle className='cardRegion'>{bookingData.
28                             uniRegion}</Card.Subtitle>
29                         </Card.Body>
30                         <Card.Img variant="top" src={!bookingData.logoUrl?
31                             bookingData.logoUrl:image} className='cardImage' />
32                         <Card.Body>
33                             <Card.Text className='cardDate'>{dayjs(date).format('
34                             dddd, MMMM D')}</Card.Text>
35                             <Card.Text className='cardTitle'>{bookingData.
36                             uniRepJobTitle + ': ' + bookingData.uniRepName}</Card.Text>
37                             <Card.Text className='cardEmail'>{bookingData.
38                             uniRepEmail}</Card.Text>
39                     </Card.Body>
40                 </div>
41             </>
42         )
43     }
44
45     export default BCard
```

```
1 // Iterate through bookings and use instances to display booking cards
2
3 import {useEffect, useState} from "react";
4 import BCard from "./BCard";
5 import '../admin.css'
6 import axios from 'axios';
7
8 const Display = () => {
9     const [booking, setBooking] = useState([])
10
11     // eslint-disable-next-line
12     useEffect(async () => {
13         async function getDisplayData() { // Async to prevent stalling
14
15             let res = await axios.get('http://localhost:5000/booking/
readfordisplay');
16             return res;
17         }
18
19         getDisplayData().then((res)=>setBooking(res.data.slice(0,3)))
// Only get the first 3 upcoming bookings
20         [], []
21
22         return (
23             <>
24                 <div className='topbar'><h1>Upcoming Bookings</h1></div>
25                 <div className='botbar'></div>
26                 <div className='container va'>
27                     {booking.map(instance=>{
28                         return(
29                             <BCard instance={instance} setBooking={setBooking
}/>
30                         )
31                     })}
32                 </div>
33             </>
34         )
35
36 export default Display
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <link
6       rel="stylesheet"
7       href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.
min.css"
8       integrity="sha384-gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/
iJTQU0hcWr7x9JvoRxT2M Zw1T"
9       crossorigin="anonymous"
10  />
11    <title>React App</title>
12  </head>
13  <body>
14    <noscript>You need to enable JavaScript to run this app.</noscript>
15    <div id="root"></div>
16    <script src=".build/app.js"></script>
17  </body>
18 </html>
19
```

```
1 {
2   "short_name": "React App",
3   "name": "Create React App Sample",
4   "icons": [
5     {
6       "src": "favicon.ico",
7       "sizes": "64x64 32x32 24x24 16x16",
8       "type": "image/x-icon"
9     },
10    {
11      "src": "logo192.png",
12      "type": "image/png",
13      "sizes": "192x192"
14    },
15    {
16      "src": "logo512.png",
17      "type": "image/png",
18      "sizes": "512x512"
19    }
20  ],
21  "start_url": ".",
22  "display": "standalone",
23  "theme_color": "#000000",
24  "background_color": "#ffffff"
25 }
26
```