TECHNICAL MANUAL

4th Year Project 'PetSpace'
Student: Emma O'Connor
Student Number: C00237292
Submission Date: 17/04/2023

Contents

Table of Figures

1. Introduction

The purpose of this manual is to provide instructions to the navigating to the PetSpace application being deployed and hosted by Firebase. The Code structure for the project will also be discussed along with the project code being provided as a visual aid.

2. Installation Instructions

The PetSpace app can be navigated to through the following link https://petspace-project.web.app/ where it is hosted and deployed on the firebase website.

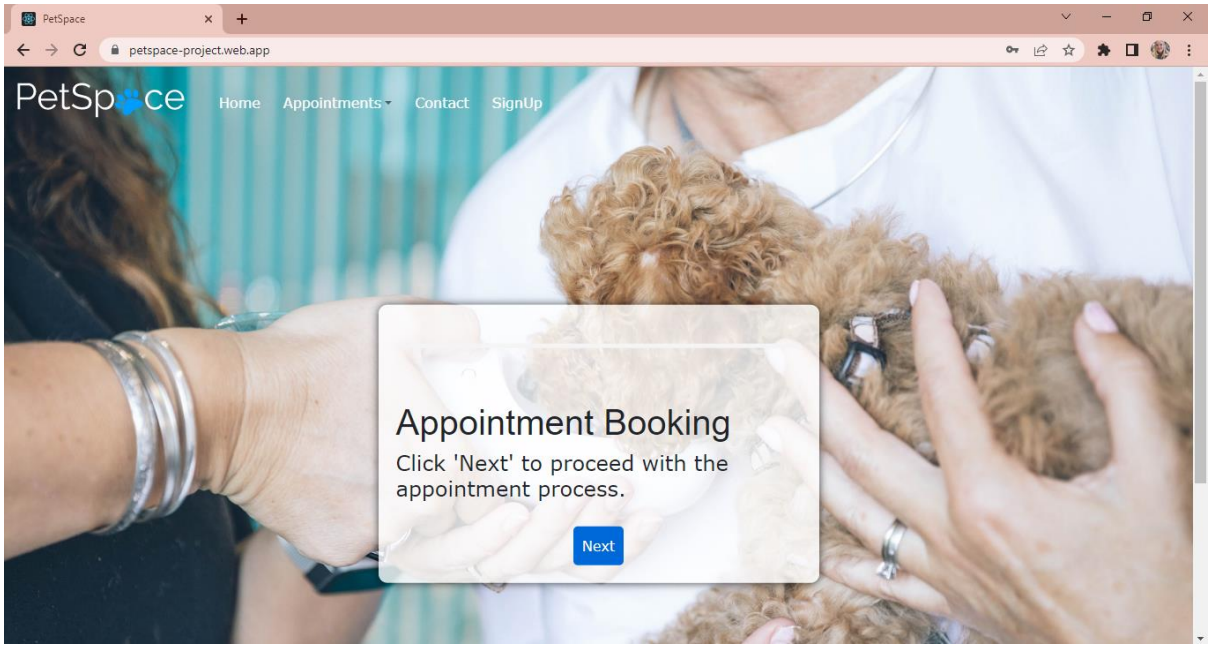

Figure 1 PetSpace

- Android Download APK File (Alternative Option)

For ease of use, there is also an option to download the PetSpace web app view onto android devices from the google drive link below. Google Drive permissions may need to be allowed within Settings > Permissions on the Android app before downloading. Choose 'Install Anyway'. Once downloaded, it can then be installed.

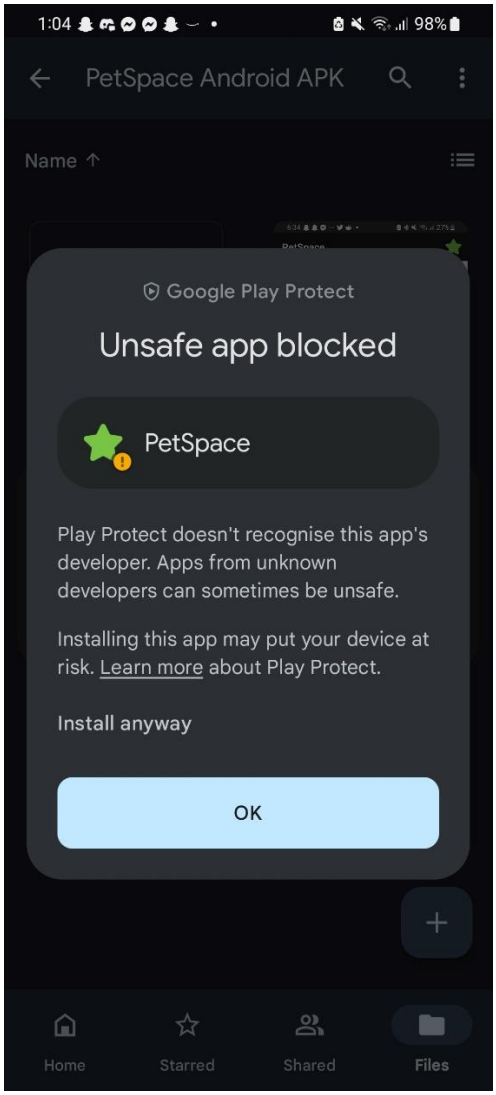

Figure 2 Install Instructions 2

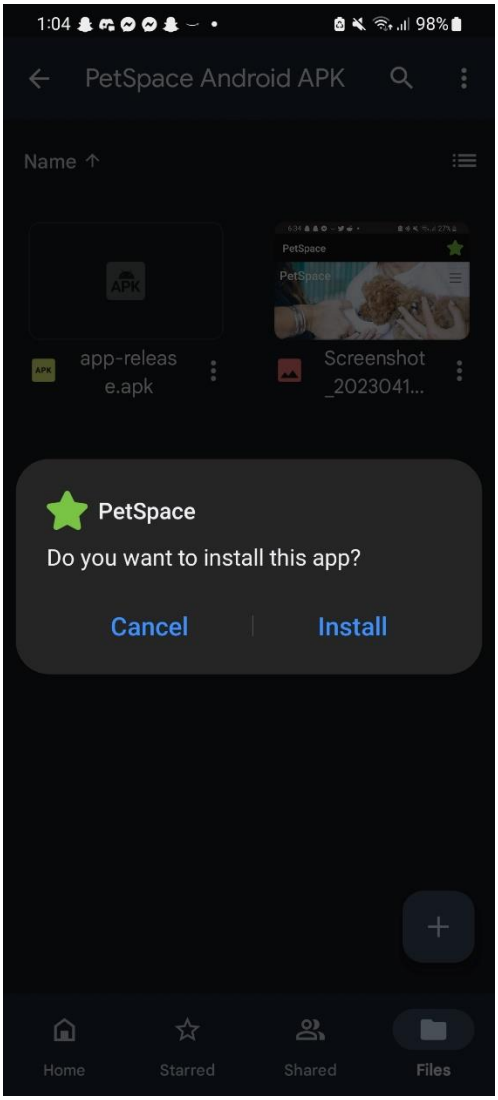

Figure 3 Install Instructions

Google Drive Link:
https://drive.google.com/drive/folders/120SimqxtyPpqvDcmxrOZt0em2m4JUOlV

3. Code Structure

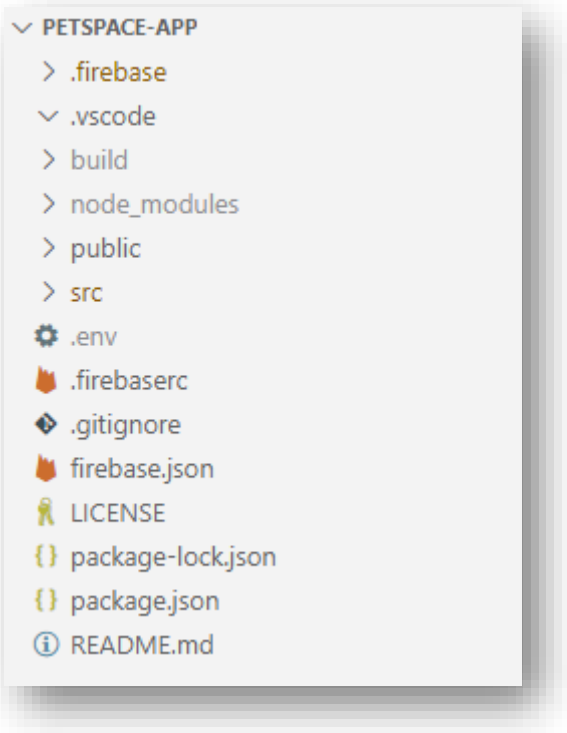

Figure 4 Code Structure 1

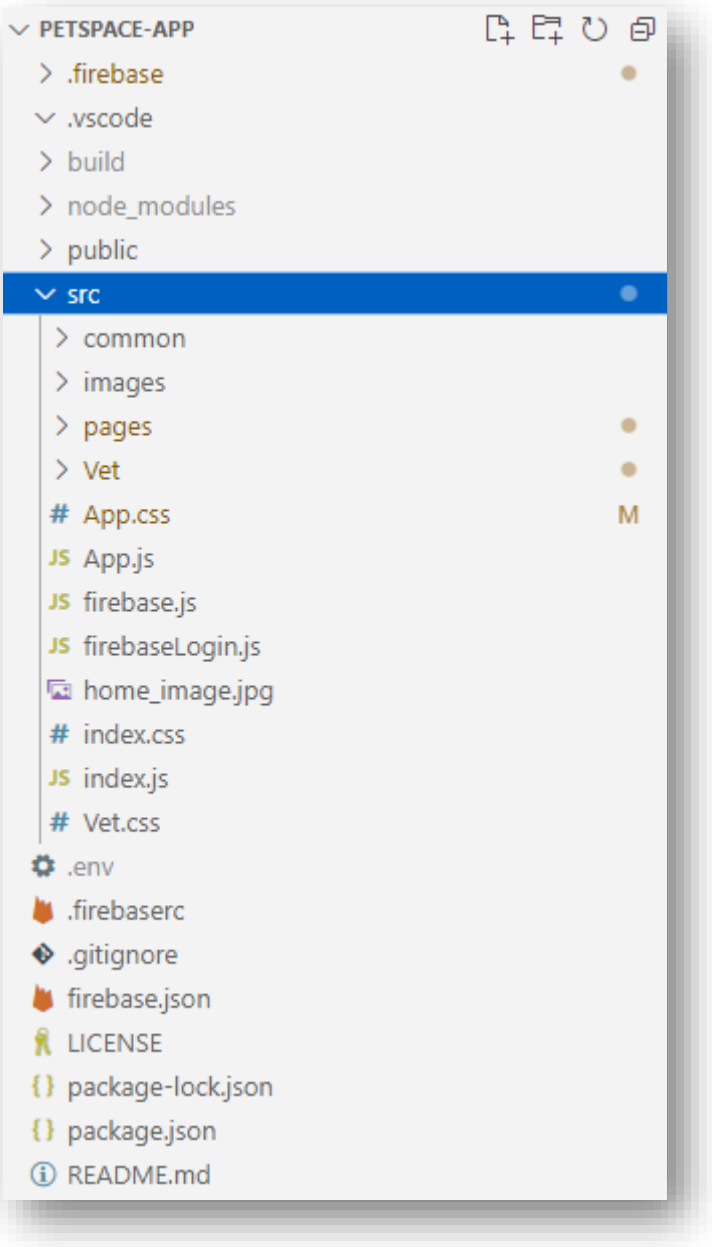

Figure 5 Code Structure 2

From the figures above, the code is located in the src folder. The build folder contains the application build files for firebase. .env contains the configuration keys for firebase. This file is an extra step of protection added to .gitignore and is not available to view online or be compromised by hackers. The common folder holds the shared components between the user side and the vet side.

4. Code

4.1 Navigation.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Navigation.js
* Description: user navigation using react-bootstrap components and react-
router-dom.
* Date: 14/04/2023
*/
//Logo created from: https://logomakr.com
import React from 'react';
import { Navbar, Nav, NavDropdown } from 'react-bootstrap';
import { Link } from 'react-router-dom';
import Logo from '../images/petspace-logo.png';

function Navigation() {

    return (
        <Navbar expand="lg" onScroll="true" fixed="top">
        <Navbar.Brand as={Link} to="/">
          <h4 className="linkText"><a href="/">
        <img src={Logo} alt="Logo" />
      </a></h4>
        </Navbar.Brand>
        <Navbar.Toggle aria-controls="basic-navbar-nav" />
        <Navbar.Collapse id="basic-navbar-nav">
          <Nav className="mr-auto">
            <Nav.Link as={Link} to="/" exact>
              <span className="linkText">Home</span>
            </Nav.Link>

            <NavDropdown title={<span
className="linkText">Appointments</span>}>
              <NavDropdown.Item as={Link} to="/Appointment">
                Book Appointment
              </NavDropdown.Item>
              <NavDropdown.Item as={Link} to="/View">
                View Appointment
              </NavDropdown.Item>
            </NavDropdown>

            <Nav.Link as={Link} to="/Contact">
              <span className="linkText">Contact</span>
            </Nav.Link>
```

```
            <Nav.Link as={Link} to="/SignUp">
              <span className="linkText">SignUp</span>
            </Nav.Link>
          </Nav>
        </Navbar.Collapse>
      </Navbar>
    )
}

export default Navigation;
```

4.2 NavbarVet.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: NavbarVet.js
* Description: vet navigation using react-bootstrap components, firebase
authentication and react-router-dom.
* Date: 14/04/2023
*/

import React from 'react';
import { Navbar, Button } from 'react-bootstrap';
import { Link, useNavigate } from 'react-router-dom';
import { signOut } from 'firebase/auth';
import { auth } from '../firebaseLogin';

function NavbarVet() {
  const navigate = useNavigate();

  const handleLogout = () => {
    signOut(auth)
      .then(() => {
        navigate('/');
      })
      .catch((error) => {
        console.error(error);
      });
  };

  return (
    <Navbar expand="lg" onScroll="true" fixed="top-right">
      <Navbar.Brand as={Link} to="#">
      </Navbar.Brand>
      <Navbar.Toggle aria-controls="basic-navbar-nav" />
      <div className="d-flex justify-content-end">
        <Button className='mr-0' onClick={handleLogout}>Logout</Button>
      </div>
    </Navbar>
  );
}

export default NavbarVet;
```

4.3 App.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: App.js
* Description: root page.
* Date: 14/04/2023
*/

import React, { useState } from 'react';
import { Route, Routes } from "react-router-dom";
import { auth } from './firebaseLogin';

//Import Navigation bars to render depending on view
import Navigation from './common/Navigation';
import NavbarVet from './common/NavbarVet';

//Connected pages
import Contact from "./pages/Contact";
import Appointment from './pages/Appointment';
import View from './pages/View';
import ViewAppointment from './pages/ViewAppointment';
import Login from './pages/Login';
import SignUp from './pages/SignUp';
import VetHome from './pages/VetHome';
import ChangePassword from './pages/ChangePassword';
import './App.css';

function App () {
  const [authenticated, setAuthenticated] = useState(false);

  auth.onAuthStateChanged((user) => {
    if (user) {
      setAuthenticated(true);
    } else {
      setAuthenticated(false);
    }
  });

  return (
    <><div className="App">
      {authenticated ? <NavbarVet /> : <Navigation />}
      <Routes>
        <Route path="/" element={<Appointment />} />
        <Route path="/Appointment" element={<Appointment />} />
        <Route path="/View" element={<View />} />
        <Route path="/ViewAppointment" element={<ViewAppointment />} />
```

```
        <Route path="/Contact" element={<Contact />} />
        <Route path="/Login" element={<Login />} />
        <Route path="/SignUp" element={<SignUp />} />
        <Route path="/VetHome" element={<VetHome />} />
        <Route path="/ChangePassword" element={<ChangePassword />} />
      </Routes>
    </div>
    </>
  );
}

export default App;
```

4.4 .env

```
REACT_APP_FIREBASE_API_KEY = "AIzaSyAc_tSNLefU7D_vRLqRwdCvEiTSlYxZqrc"
REACT_APP_FIREBASE_AUTH_DOMAIN = "petspace-project.firebaseapp.com"
REACT_APP_FIREBASE_PROJECT_ID = "petspace-project"
REACT_APP_FIREBASE_STORAGE_BUCKET = "petspace-project.appspot.com"
REACT_APP_FIREBASE_MESSAGING_SENDER_ID = "704856125780"
REACT_APP_FIREBASE_APP_ID = "1:704856125780:web:26bcccb08a6494cfa098c1"
REACT_APP_FIREBASE_MEASUREMENT_ID = "G-XJGTSP5V05"
```

4.5 Firebase.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: App.js
* Description: root page.
* Date: 14/04/2023
*/

// Import the functions needed
import { initializeApp } from "firebase/app";
import { getFirestore } from "firebase/firestore";

const firebaseConfig = {
  apiKey: process.env.REACT_APP_FIREBASE_API_KEY,
  authDomain: process.env.REACT_APP_FIREBASE_AUTH_DOMAIN,
  projectId: process.env.REACT_APP_FIREBASE_PROJECT_ID,
  storageBucket: process.env.REACT_APP_FIREABSE_STORAGE_BUCKET,
  messagingSenderId: process.env.REACT_APP_FIREBASE_MESSAGING_SENDER_ID,
  appId: process.env.REACT_APP_FIREBASE_APP_ID,
  measurementId: process.env.REACT_APP_FIREBASE_MEASUREMENT_ID
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const firebase = getFirestore(app);
export default firebase;
```

4.6 FirebaseLogin.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: App.js
* Description: root page.
* Date: 14/04/2023
*/

// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { getAuth } from "firebase/auth";

const firebaseConfig = {
  apiKey: process.env.REACT_APP_FIREBASE_API_KEY,
  authDomain: process.env.REACT_APP_FIREBASE_AUTH_DOMAIN,
  projectId: process.env.REACT_APP_FIREBASE_PROJECT_ID,
  storageBucket: process.env.REACT_APP_FIREABSE_STORAGE_BUCKET,
  messagingSenderId: process.env.REACT_APP_FIREBASE_MESSAGING_SENDER_ID,
  appId: process.env.REACT_APP_FIREBASE_APP_ID,
  measurementId: process.env.REACT_APP_FIREBASE_MEASUREMENT_ID
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
export const auth = getAuth(app);
export default app;
```

4.7 Welcome.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Welcome.js
* Description: first page of booking form.
* Date: 14/04/2023
*/

import React from 'react'
import '../../App.css';

const Welcome = () => {
  return (
    <div class="App">
    <div className='form-text-on-image'>
    <h1>Appointment Booking</h1>
      <h4>Click 'Next' to proceed with the appointment process.</h4>
    </div>
    </div>
  )
}

export default Welcome
```

4.8 FormDetails.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: FormDetails.js
* Description: Stepper Form.
* Date: 14/04/2023
* Ref: www.youtube.com. How to Create a Multi-Step Form With React, React
Hooks, Bootstrap and Firebase. [online] Available at:
https://youtu.be/kbvNrd7bBXs [Accessed 17 Apr. 2023].
*/

import React, {useState} from 'react'

// Stepper Pages
import Welcome from './Welcome'
import PersonalDetails from './PersonalDetails';
import PetDetails from './PetDetails';
import AppointmentDetails from './AppointmentDetails';
import BookingDetails from './BookingDetails';
import Summary from './Summary';
import {Button, ProgressBar} from 'react-bootstrap'

const FormDetails = () => {
    //Steps
    const [activeStep, setActiveStep] = useState(0)

    const getSteps = () => {
      return ["Welcome", "Appointment Details", "Pet Details", "Personal
Details", "Booking Details"]
    }

    const steps = getSteps()

    //State variables
    const [appointments, setMultiFormValues] = useState({
    bookingCategory: "",
    date: new Date().toDateString(),
    time: "",
    petName: "",
    petAge: 0,
    petGender: "",
    petType: "",
    petBreed: "",
    petVaccinated: "No",
    petNote: "",
    firstName: "",
```

```
    lastName: "",
    email: "",
    confirmEmail: "",
    phone: "",
  })

  //Calculate the progress
  const progress = ((activeStep + 1) / steps.length) * 100

  //Navigates to the next page
  const handleNext = () => {
    setActiveStep((nextStep) => nextStep + 1)
  }
  //Navigates to the Previous page
  const handleBack = () => {
    setActiveStep((previousStep) => previousStep - 1)
  }

  //Handle form value state on change
  const handleChange = (input) => (e) => {
    setMultiFormValues({...appointments, [input]: e.target.value})
  }

    return (
        <div>

            <div className='form-text-on-image'>

            <ProgressBar now={activeStep * 22} />

            {activeStep === 0 && (
        <Welcome handleChange={handleChange} />
      )}
      {activeStep === 1 && (
        <AppointmentDetails values={appointments} handleChange={handleChange}
/>
      )}
      {activeStep === 2 && (
        <PetDetails values={appointments} handleChange={handleChange} />
      )}
      {activeStep === 3 && (
        <PersonalDetails values={appointments} handleChange={handleChange} />
      )}
      {activeStep === 4 && (
        <BookingDetails values={appointments} handleChange={handleChange} />
      )}
      {activeStep === 5 && (
        <Summary values={appointments} handleChange={handleChange} />
```

```
      )}
        <center>
        <Button hidden={activeStep === 0} className="mr-5"
onClick={handleBack} style={activeStep === 5 ? {display: 'none'} : {}}
>Back</Button>

        <Button className="ml-5" variant="contained" onClick={handleNext}
style={activeStep === 5 ? {display: 'none'} : {}} >{}{activeStep ===
steps.length - 1 ? 'Submit' : 'Next'}</Button>
        </center>

      </div>

    </div>
  )
}

export default FormDetails
```

4.9 AppointmentDetails.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: AppointmentDetails.js
* Description: Appointment form page 2.
* Date: 14/04/2023
*/

import React, { useState, useEffect } from 'react'
import { Form } from 'react-bootstrap'
import firebase from '../../firebase'
import { collection, getDocs, query, where } from "firebase/firestore";
import { Button } from 'react-bootstrap';

const AppointmentDetails = ({ values, handleChange }) => {

  const [appointmentType, setAppointmentType] = useState('');
  const [appointmentDate, setAppointmentDate] = useState('');
  const [appointmentTime, setAppointmentTime] = useState('');
  const [appointments, setAppointments] = useState([]);
 /// const appointmentCollectionRef = collection(firebase, "appointments");

  useEffect(() => {
    const appointmentCollectionRef = collection(firebase, "appointments");

    const getAllAppointments = async () => {
      const q = query(appointmentCollectionRef, where("time", ">", ""));
      const querySnapshot = await getDocs(q);
      const appointmentsData = querySnapshot.docs.map((doc) =>
doc.data().time);
      setAppointments(appointmentsData);
    };
    getAllAppointments();
    console.log(getAllAppointments());
  }, []);

  const appointmentOptions = [
    {
      type: 'Teeth Cleaning',
      dates: 10,
    },
    {
      type: 'Emergency',
      dates: 10,
    },
    {
```

```
      type: 'Follow-Up Checkup',
      dates: 10,
    },
    {
      type: 'Neutering/Spaying',
      dates: 10,
    },
    {
      type: 'Grooming',
      dates: 10,
    },
    {
      type: 'Surgery',
      dates: 10,
    },
    {
      type: 'Checkup',
      dates: 10,
    },
    {
      type: 'Vaccinations and Boosters',
      dates: 10,
    },
  ];

  const handleAppointmentTypeChange = (event) => {
    setAppointmentType(event.target.value);
    values.bookingCategory = event.target.value;
    setAppointmentDate('');
    setAppointmentTime('');
    console.log(values.bookingCategory);
  };

  const handleAppointmentDateChange = (event) => {
    setAppointmentDate(event.target.value);
    values.date = event.target.value;
    setAppointmentTime('');
    console.log(values.date);
  };

  const getAppointmentDates = (type) => {
    const today = new Date();
    const appointmentOption = appointmentOptions.find((option) => option.type
=== type);
    const dates = [];
    const bankHolidays = ['2023-01-01', '2023-03-17', '2023-04-14', '2023-04-
10', '2023-05-01', '2023-06-05', '2023-08-07', '2023-10-30', '2023-12-25',
'2023-12-26']; // array of bank holiday dates in YYYY-MM-DD format for 2023
```

```
    for (let i = 0; i < appointmentOption.dates; i++) {
      const date = new Date(today.getTime() + (i * 24 * 60 * 60 * 1000));
      //Mon - Fri (inclusive) only and exclude bank holidays
      if (date.getDay() !== 0 && date.getDay() !== 6 &&
!bankHolidays.includes(date.toISOString().split('T')[0])) {
        const formattedDate = date.toLocaleString('en-UK', { weekday: 'short',
year: 'numeric', month: 'short', day: 'numeric' });
        dates.push(formattedDate);
      }
    }
    return dates;
  };

  const getAppointmentTimes = () => {
    const times = ['9:00 AM', '10:00 AM', '11:00 AM', '1:00 PM', '2:00 PM',
'3:00 PM', '4:00 PM'];

    return times;
  };

  const times = getAppointmentTimes();


  const handleAppointmentTimeClick = (event) => {
    if (appointmentDate) {
      const time = event.target.value;
      if (!appointments.includes(time)) {
        setAppointmentTime(time);
        values.time = time;
        console.log(values.time);
      }
    }
  };

    return (
        <div>
            <h1>Booking Details</h1>
            <Form className="mt-5">
            <Form.Group className="mt-2">
            <Form.Label controlId="input" htmlFor="appointment-
type">Appointment Type:</Form.Label>
            <Form.Control
            required as="select"
            id="appointment-type"
            defaultValue={values.bookingCategory}
            value={appointmentType}
            onChange={handleAppointmentTypeChange}>
```

```
              <option value="" disabled hidden>-- Select an appointment type --</option>
              {appointmentOptions.map((option) => (
                <option key={option.type} value={option.type}>{option.type}</option>
              ))}
            </Form.Control>

            </Form.Group>
                <Form.Group className="mt-3">
                {appointmentType && (
            <>
              <div>
                <label controlId="input" htmlFor="appointment-date">Appointment Date:</label>
                <Form.Control
                required as="select"
                id="appointment-date"
                defaultValue={values.date}
                value={appointmentDate}
                onChange={handleAppointmentDateChange}>
                  <option value="" disabled hidden>-- Select an appointment date --</option>
                  {getAppointmentDates(appointmentType).map((date) => (
                    <option key={date} value={date}>{date}</option>
                  ))}
                </Form.Control>
              </div>
            </>
          )}
          </Form.Group>

          <Form.Group className="mt-3">
          <div>
          <label controlId="input">Appointment Time:</label>
          <hr></hr>
          {times.map((time) => (
            <Button
              required
              key={time}
              type="button"
              defaultValue={values.time}
              value={time}
              onClick={handleAppointmentTimeClick}
              disabled={!appointmentDate || appointments.includes(time)}
              className={appointmentTime === time ? 'active' : ''}
            >
              {time}
```

```
            </Button>
          ))}

          </div>

          </Form.Group>

          </Form>
      </div>
    )
}

export default AppointmentDetails
```

4.10 PetDetails.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: PetDetails.js
* Description: Appointment form page 3.
* Date: 14/04/2023
*/

import React, { useState } from 'react'
import {Form, FloatingLabel} from 'react-bootstrap'

const PetDetails = ({ values, handleChange }) => {
  console.log(values, handleChange);

  const [validated, setValidated] = useState(false);

  const handleSubmit = (event) => {
    event.preventDefault();
    const form = event.currentTarget;
    if (form.checkValidity() === false) {
      event.stopPropagation();
      setValidated(true);
    } else {
      // Handle form submission
      console.log("Form submitted successfully!");
    }
  };

  return (
    <div>
      <Form className="mt-5" noValidate validated={validated}
onSubmit={handleSubmit}>
        <h4>Pet Details</h4>

          <Form.Group className="mt-2">

          <FloatingLabel
            controlId="floatingInput"
            label="Pet Name"
            className="mb-3"
          >
          <Form.Control
            required
            value={values.petName}
            onChange={handleChange("petName")}
            type="text"
```

```
            placeholder="Pet Name"
            onBlur={() => setValidated(true)}
            pattern="^[a-zA-Z\s]+$"
            isInvalid={validated && !/^[a-zA-Z\s]{2,}$/.test(values.petName) ?
true : null}
            isValid={validated && /^[a-zA-Z\s]{2,}$/.test(values.petName) ?
true : null}
            feedback="Please enter your pet name."
          />
          </FloatingLabel>
        </Form.Group>


          <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Pet Age (months)"
            className="mb-3"
          >
          <Form.Control
            required
            value={values.petAge}
            onChange={handleChange("petAge")}
            type="number"
            placeholder="Age"
            onBlur={() => setValidated(true)}
            isInvalid={!values.petAge && validated}
            feedback="Please enter your pet Age."
          />
          </FloatingLabel>
        </Form.Group>

<Form.Group>
          {['radio'].map((type) => (
        <div key={`inline-${type}`} className="mb-3">
          <Form.Check
            required
            inline
            label="Male"
            name="group1"
            value="Male"
            type={type}
            id={`inline-${type}-1`}
            defaultValue={values.petGender}
            checked={values.petGender === "Male"}
onChange={handleChange("petGender")}

          />
```

```
            <Form.Check
              required
              inline
              name="group1"
              label="Female"
              value="Female"
              type={type}
              id={`inline-${type}-2`}
              defaultValue={values.petGender}
              checked={values.petGender === "Female"}
onChange={handleChange("petGender")}
            />
          </div>
        ))}
          </Form.Group>

          <Form.Group className="mt-2">
            <FloatingLabel
              controlId="floatingInput"
              label="Animal Type"
              className="mb-3"
            >
            <Form.Control
              required
              value={values.petType}
              onChange={handleChange("petType")}
              type="text"
              placeholder="Type of Animal"
              onBlur={() => setValidated(true)}
              pattern="^[a-zA-Z\s]+$"
              isInvalid={validated && !/^[a-zA-Z\s]{3,}$/.test(values.petType) ?
true : null}
              isValid={validated && /^[a-zA-Z\s]{3,}$/.test(values.petType) ?
true : null}
              feedback="Please enter your pet type."
            />
            </FloatingLabel>
          </Form.Group>

          <Form.Group className="mt-2">
            <FloatingLabel
              controlId="floatingInput"
              label="Animal Breed"
              className="mb-3"
            >
            <Form.Control
              required
              defaultValue={values.petBreed}
```

```
            onChange={handleChange("petBreed")}
            type="text"
            placeholder="Breed"
            onBlur={() => setValidated(true)}
            pattern="^[a-zA-Z\s]+$"
            isInvalid={validated && !/^[a-zA-Z\s]{3,}$/.test(values.petBreed)
? true : null}
            isValid={validated && /^[a-zA-Z\s]{3,}$/.test(values.petBreed) ?
true : null}
            feedback="Please enter your pet breed."
          />
          </FloatingLabel>
        </Form.Group>

        <Form.Group className="mt-3">
          <Form.Label>Is the animal vaccinated? if yes, please
check:</Form.Label>
          <Form.Check
           defaultValue={values.petVaccinated}
           onChange={handleChange("petVaccinated")}
           type="checkbox"
           value="yes"
           inline
          />
        </Form.Group>

        <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Note (optional):"
            className="mb-3"
          >
          <Form.Control
          as="textarea"
          rows={4}
            defaultValue={values.petNote}
            onChange={handleChange("petNote")}
            placeholder="Note (optional):"
            maxLength={200} // set maximum character limit to 200
          />
          <div style={{textAlign: "right", marginTop: "-20px", marginRight:
"10px", fontSize: "10px"}}>
            {values.petNote.length}/{200} characters
          </div>
          </FloatingLabel>
        </Form.Group>

    </Form>
```

```
    </div>
  );
};

  export default PetDetails
```

4.11 PersonalDetails.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: PersonalDetails.js
* Description: Appointment form page 4.
* Date: 14/04/2023
*/

import React, { useState } from "react";
import { Form, FloatingLabel } from 'react-bootstrap'

const PersonalDetails = ({ values, handleChange }) => {
  console.log(values, handleChange);
  const [validated, setValidated] = useState(false);

  const handleSubmit = (event) => {
    event.preventDefault();
    const form = event.currentTarget;
    if (form.checkValidity() === false) {
      event.stopPropagation();
      setValidated(true);
    } else {
      // Handle form submission
      console.log("Form submitted successfully!");
    }
  };

  return (
    <div>
      <Form className="mt-5" noValidate validated={validated}
onSubmit={handleSubmit}>
        <h4>Owner Details</h4>

        <Form.Group className="mt-3">
          <FloatingLabel
            controlId="floatingInput"
            label="First Name"
            className="mb-3"
          >
            <Form.Control
              value={values.firstName}
              onChange={handleChange("firstName")}
              type="text"
              placeholder="First Name"
              required
              onBlur={() => setValidated(true)}
```

```
              pattern="^[a-zA-Z\s]+$"
              feedback="Please enter your first name."
              isInvalid={validated && !/^[a-zA-
Z\s]{2,}$/.test(values.firstName) ? true : null}
              isValid={validated && /^[a-zA-Z\s]{2,}$/.test(values.firstName)
? true : null}
            />
          </FloatingLabel>
        </Form.Group>

        <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Last Name"
            className="mb-3"
          >
            <Form.Control
              value={values.lastName}
              onChange={handleChange("lastName")}
              type="text"
              placeholder="Last Name"
              required
              pattern="^[a-zA-Z]{2,}\s*$"
              feedback="Please enter your last name."
              onBlur={() => setValidated(true)}
              isInvalid={validated && !/^[a-zA-
Z]{2,}\s*$/.test(values.lastName) ? true : null}
              isValid={validated && /^[a-zA-Z]{2,}\s*$/.test(values.lastName)
? true : null}
            />
          </FloatingLabel>
        </Form.Group>

        <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Email"
            className="mb-3"
          >
            <Form.Control
              defaultValue={values.email}
              onChange={handleChange("email")}
              type="email"
              placeholder="email"
              required
              onBlur={() => setValidated(true)}
              pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,}$"
```

```
              isInvalid={validated && !/^[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-
z]{2,}$/.test(values.email) ? true : null}
              isValid={validated && /^[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-
z]{2,}$/.test(values.email) ? true : null}
              feedback="Please enter a valid email address."
            />
          </FloatingLabel>
        </Form.Group>

        <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Confirm Email"
            className="mb-3"
          >
            <Form.Control
              defaultValue={values.confirmEmail}
              onChange={handleChange("confirmEmail")}
              type="email"
              placeholder="confirm email"
              required
              onBlur={() => setValidated(true)}
              isInvalid={validated && !values.confirmEmail ||
values.confirmEmail !== values.email ? true : null}
              isValid={validated && values.confirmEmail || values.confirmEmail
=== values.email ? true : null}
              feedback="Please enter a valid email address."
            />
          </FloatingLabel>
        </Form.Group>

        <Form.Group className="mt-2">
          <FloatingLabel
            controlId="floatingInput"
            label="Phone No. (+353)"
            className="mb-3"
            >
            <Form.Control
              required
              value={values.phone}
              onChange={handleChange("phone")}
              type="tel"
              placeholder="phone no."
              onBlur={() => setValidated(true)}
              pattern="^[0-9]{10}$"
              isInvalid={validated && !/^([0-9]{10})$/.test(values.phone) ?
true : null}
```

```
              isValid={validated && /^([0-9]{10})$/.test(values.phone) ? true
: null}
              feedback="Please enter a valid phone number (10 digits with no
special characters)."
            />
          </FloatingLabel>
        </Form.Group>
      </Form>
    </div>
  );
};

export default PersonalDetails;
```

4.12 BookingDetails.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: BookingDetails.js
* Description: Appointment form page 5.
* Date: 14/04/2023
*/

import React, {  } from 'react'
import ListGroup from 'react-bootstrap/ListGroup'

const BookingDetails = ({values}) => {

  const {firstName, lastName, email, phone, petName, bookingCategory, date,
time} = values


  return (
    <div>
    <form className="mt-5">
    <h1>Booking Summary</h1>
      <ListGroup style={{maxWidth:'600px', margin:'auto'}}>
        <ListGroup.Item variant="secondary">NAME: {firstName} {lastName}
</ListGroup.Item>
        <ListGroup.Item variant="secondary">EMAIL: {email}</ListGroup.Item>
        <ListGroup.Item variant="secondary">PHONE: {phone}</ListGroup.Item>
        <ListGroup.Item variant="secondary">PET NAME:
{petName}</ListGroup.Item>
        <ListGroup.Item variant="secondary">BOOKING CATEGORY:
{bookingCategory}</ListGroup.Item>
        <ListGroup.Item variant="secondary">DATE: {date}</ListGroup.Item>
        <ListGroup.Item variant="secondary">TIME: {time}</ListGroup.Item>
      </ListGroup><br></br>

      <input type="checkbox" required name="termsAndConditions" /> 
      <label style={{textDecoration: 'underline'}}>
        I agree to the terms and conditions
      </label>

    <div className="invalid-feedback">You must agree to the terms and
conditions.</div>


      <h5>Click 'Submit' to confirm your booking details.</h5>
      <br></br>
    </form>
```

```
    </div>
  )
}


export default BookingDetails
```

4.13 Summary.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Summary.js
* Description: Appointment form page 6.
* Date: 14/04/2023
* EmailJS. Send email from Javascript – no server code required. [online]
Available at: https://www.emailjs.com/.
* Firebase. Add data to Cloud Firestore | Firebase Documentation. [online].
Available at: https://firebase.google.com/docs/firestore/manage-data/add-data.
*/

import { Button } from 'react-bootstrap';
import firebase from '../../firebase'
import { collection, addDoc } from "firebase/firestore";
import emailjs from '@emailjs/browser';

const SERVICE_ID = 'service_749d259';
const TEMPLATE_ID = 'template_12e1eu5';
const USER_ID = '7M3zeW5o3C-7sb5zq';

const sendEmail = (values, userEmail) => {
  const {
    firstName,
    lastName,
    email,
    confirmEmail,
    phone,
    petName,
    petAge,
    petGender,
    petType,
    petBreed,
    petVaccinated,
    petNote,
    bookingCategory,
    date,
    time,
  } = values;

  const templateParams = {
    from_name: `${firstName} ${lastName}`,
    from_email: email,
    to_email: userEmail,
    cc: email, // add user's email address to cc field
    to_name: 'Emma',
```

```
    message: `Email: ${email}<br/>Confirm Email: ${confirmEmail}<br/>Phone:
${phone}<br/>Pet Name: ${petName}<br/>Pet Age: ${petAge}<br/>Pet Gender:
${petGender}<br/>Pet Type: ${petType}<br/>Pet Breed: ${petBreed}<br/>Pet
Vaccinated: ${petVaccinated}<br/>Pet Note: ${petNote}<br/>Booking Category:
${bookingCategory}<br/>Date: ${date}<br/>Time: ${time}`
  };

  console.log(templateParams);
  console.log(email);
  console.log(userEmail);

  emailjs.send(SERVICE_ID, TEMPLATE_ID, templateParams, USER_ID)
    .then(() => {
      console.log('Email sent successfully!');
    }, (error) => {
      console.error('Error sending email:', error);
    });
};

const Summary = ({ values }) => {
  const now = new Date();
  const {
    firstName,
    lastName,
    email,
    confirmEmail,
    phone,
    petName,
    petAge,
    petGender,
    petType,
    petBreed,
    petVaccinated,
    petNote,
    bookingCategory,
    date,
    time,
  } = values;

  const sendEmailHandler = () => {
    sendEmail(
      {
        firstName,
        lastName,
        email,
        confirmEmail,
        phone,
        petName,
```

```
        petAge,
        petGender,
        petType,
        petBreed,
        petVaccinated,
        petNote,
        bookingCategory,
        date,
        time,
      },
      email // pass user's email address to sendEmail function
    );
  };

  try {
    addDoc(collection(firebase, "appointments"), {
      firstname: firstName,
      lastname: lastName,
      email: email,
      confirmemail: confirmEmail,
      phone: phone,
      petname: petName,
      petage: petAge,
      petgender: petGender,
      pettype: petType,
      petbreed: petBreed,
      petvaccinated: petVaccinated,
      petnote: petNote,
      bookingcategory: bookingCategory,
      date: date,
      time: time,
      createdAt: now.toISOString(),
    });
  } catch (e) {
    console.error("Error:", e);
  }

  const handleButtonClick = () => {
    window.location.href = "/";
  };

  return (
    <>
      <h1>Booking Confirmation Page</h1>
      <center>
      <Button onClick={sendEmailHandler} style={{ display: 'inline-block'
}}>Email Confirmation</Button>
      <br/>
```

```
      <Button onClick={handleButtonClick} style={{ display: 'inline-block'
}}>Return to Homepage</Button>
      </center>
    </>
  );

};

export default Summary;
```

4.14 Appointment.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Appointment.js
* Description: Background Image for homepage and stepper pages.
* Date: 14/04/2023
* https://www.pexels.com/search/vet%20with%20small%20dog/
* https://www.pexels.com/search/blue%20background%20website%20gradient/
*/


import React from 'react';
import FormDetails from './StepperPages/FormDetails';
import '../index.css';
import BackgroundImage from '../home_image.jpg';

function Appointment() {
  return (
    <div>
    <img
      srcSet={`${BackgroundImage} 320w, ${BackgroundImage} 680w,
${BackgroundImage} 960w, ${BackgroundImage} 2430w` }
      src={BackgroundImage}
      alt="Background"  />
       <div className='form-button-text-on-image'>
     
      <FormDetails />
    </div>
    </div>
  );
}

export default Appointment;
```

4.15 View.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: View.js
* Description: View page retrieves appointment for email provided.
* Date: 14/04/2023
*/

import { Form, FloatingLabel, Button } from 'react-bootstrap';
import React, { useState, useEffect } from 'react'
import firebase from '../firebase'
import { collection, getDocs, query, where } from "firebase/firestore";
import ViewAppointment from './ViewAppointment';

function View () {

    const [email, setEmail] = useState('');
    const [record, setRecord] = useState(null);
    const [error, setError] = useState(null);

    const handleInputChange = (e) => {
      setEmail(e.target.value);
    };

    const handleSearch = async () => {
      const q = query(collection(firebase, 'appointments'), where('email',
'==', email));
      const querySnapshot = await getDocs(q);
      setError(null);
      if (!querySnapshot.empty) {
        setError(null)
        setRecord(querySnapshot.docs[0].data());
      } else {
        setRecord(null);
        setError("No record found with that email address.");
      }
    };

    useEffect(() => {
      // reset record when email changes
      setRecord(null);
    }, [email]);

    if (record && email.length > 0) {
      return <ViewAppointment record={record} />;
    }
```

```
    return (
      <main>
        <section>
        <div className='signup-form-container'>
              <div >
                <h1>Please enter the email address used to book your
appointment.</h1>

            <form className='signup-form'>
                    <div>
                        <label htmlFor="email-address">Email address</label>
                            <input
                                type="email"
                                label="Email Address:"
                                required
                                pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-
                               z]{2,4}$"
                                placeholder="Email address"
                                value={email}
                                onChange={handleInputChange}
                              />
                            </div>
                          <Button onClick={handleSearch}>Search</Button>
                          {error && <p>{error}</p>}
                          <div className='animated-clip'>
                        </div>
                    </form>
                </div>
            </div>
        </section>
      </main>
    );
  }

export default View
```

4.16 ViewAppointment.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: ViewAppointment.js
* Description: Page Displays appointment along with functionalities.
* Date: 14/04/2023
*/

import React, { useState, useEffect } from 'react'
import firebase from '../firebase';
import { collection, getDocs, deleteDoc, updateDoc, setDoc, doc, query, where
} from "firebase/firestore";

import '../App.css';
import { Button, Card, Col, Container, Row, Table, Form, Modal} from 'react-
bootstrap';
import { redirect } from 'react-router-dom';

const ViewAppointment = ({ record }) => {
  const [showModal, setShowModal] = useState(false);
  const history = redirect();
  const [error, setError] = useState(false);

  const [petName, setPetName] = useState(record.petname);
  const [petAge, setPetAge] = useState(record.petage);
  const [petGender, setPetGender] = useState(record.petgender);
  const [petType, setPetType] = useState(record.pettype);
  const [petBreed, setPetBreed] = useState(record.petbreed);
  const [petVaccinated, setPetVaccinated] = useState(record.petvaccinated);
  const [petNote, setPetNote] = useState(record.petnote);
  const [showBookingModal, setShowBookingModal] = useState(false);
  const [bookingCategory, setBookingCategory] =
useState(record.bookingcategory);
  const [date, setDate] = useState(record.date);
  const [time, setTime] = useState(record.time);

  const handleCloseModal = () => setShowModal(false);
  const handleShowModal = () => setShowModal(true);
  const handleCloseBookingModal = () => setShowBookingModal(false);
  const handleShowBookingModal = () => setShowBookingModal(true);

  const handleEdit = () => {
    const appointmentsRef = collection(firebase, 'appointments');
    const q = query(appointmentsRef, where('petname', '==', record.petname));

    // Perform form validation
```

```
    if (!petName || !bookingCategory || !date || !time) {
      // If form is invalid, close the modal without updating the document
      handleCloseBookingModal();
      const notificationBanner = document.createElement('div');
      notificationBanner.classList.add('alert', 'alert-danger', 'text-center',
'mb-0');
      notificationBanner.setAttribute('role', 'alert');
      notificationBanner.textContent = 'Please fill in all required fields.';

      document.body.insertBefore(notificationBanner,
document.body.firstChild);

      setTimeout(() => {
        notificationBanner.remove();
      }, 3000);

      return;
    }

    getDocs(q)
      .then((querySnapshot) => {
        if (querySnapshot.size === 1) {
          const docRef = querySnapshot.docs[0].ref;

          updateDoc(docRef, {
            petname: petName,
            bookingcategory: bookingCategory,
            date: date,
            time: time
          })
            .then(() => {
              console.log(`Booking details for pet ${petName} updated
successfully.`);
              handleCloseBookingModal();
              const notificationBanner = document.createElement('div');
              notificationBanner.classList.add('alert', 'alert-success',
'text-center', 'mb-0');
              notificationBanner.setAttribute('role', 'alert');
              notificationBanner.textContent = `Booking details for pet
${petName} updated successfully.`;

              document.body.insertBefore(notificationBanner,
document.body.firstChild);

              setTimeout(() => {
                notificationBanner.remove();
              }, 3000);
            })
```

```
            .catch((error) => {
              console.error(`Error updating booking details for pet
${petName}: ${error.message}`);
            });
        } else {
          console.error(`Expected 1 document, but found ${querySnapshot.size}
documents.`);
        }
      })
      .catch((error) => {
        console.error(`Error querying for pet ${record.petname}:
${error.message}`);
      });
  };

  function getAvailableDates() {
    const today = new Date();
    const endDate = new Date(today.getTime() + 14 * 24 * 60 * 60 * 1000);
    const bankHolidays = [
      new Date("2023-03-17"), // St Patricks Day
      new Date("2023-05-01"), // May Bank Holiday
      new Date("2023-06-05"), // Spring Bank Holiday
      new Date("2023-08-28"), // August Bank Holiday
      new Date("2023-12-25"), // Christmas Day
      new Date("2023-12-26")  // Stephens Day
    ];
    const availableDates = [];

    for (let date = today; date <= endDate; date.setDate(date.getDate() + 1))
{
      // Check if the date is a weekend day
      if (date.getDay() === 0 || date.getDay() === 6) {
        continue;
      }

      // Check if the date is a bank holiday
      if (bankHolidays.some(holiday => holiday.toDateString() ===
date.toDateString())) {
        continue;
      }

      availableDates.push(new Date(date));
    }

    return availableDates;
  }

  const availableDates = getAvailableDates();
```

```
  const times = ['9:00 AM', '10:00 AM', '11:00 AM', '1:00 PM', '2:00 PM',
'3:00 PM', '4:00 PM'];


  const handleSaveChanges = () => {
    const appointmentsRef = collection(firebase, 'appointments');
    const q = query(appointmentsRef, where('petname', '==', record.petname));

    // Perform form validation
    if (!petName || !petAge || !petGender || !petType || !petBreed) {
      // If form is invalid, close the modal without updating the document
      handleCloseModal();
      const notificationBanner = document.createElement('div');
      notificationBanner.classList.add('alert', 'alert-danger', 'text-center',
'mb-0');
      notificationBanner.setAttribute('role', 'alert');
      notificationBanner.textContent = 'Please fill in all required fields.';

      document.body.insertBefore(notificationBanner,
document.body.firstChild);

      setTimeout(() => {
        notificationBanner.remove();
      }, 3000);

      return;
    }

    getDocs(q)
      .then((querySnapshot) => {
        if (querySnapshot.size === 1) {
          const docRef = querySnapshot.docs[0].ref;

          updateDoc(docRef, {
            petname: petName,
            petage: petAge,
            petgender: petGender,
            petvaccinated: petVaccinated,
            pettype: petType,
            petbreed: petBreed,
            petnote: petNote
          })
            .then(() => {
              console.log(`Pet ${petName} updated successfully.`);
              handleCloseModal();
              const notificationBanner = document.createElement('div');
              notificationBanner.classList.add('alert', 'alert-success',
'text-center', 'mb-0');
```

```
              notificationBanner.setAttribute('role', 'alert');
              notificationBanner.textContent = `Pet ${petName} updated
successfully.`;

              document.body.insertBefore(notificationBanner,
document.body.firstChild);

              setTimeout(() => {
                notificationBanner.remove();
              }, 3000);
            })
            .catch((error) => {
              console.error(`Error updating pet ${petName}:
${error.message}`);
            });
        } else {
          console.error(`Expected 1 document, but found ${querySnapshot.size}
documents.`);
        }
      })
      .catch((error) => {
        console.error(`Error querying for pet ${record.petname}:
${error.message}`);
      });
  };

  const handleDelete = () => {
    if (window.confirm("Are you sure you want to delete this appointment?")) {
      const appointmentsRef = collection(firebase, 'appointments');
      const q = query(appointmentsRef, where('petname', '==',
record.petname));

      getDocs(q)
        .then((querySnapshot) => {
          if (querySnapshot.size === 1) {
            const docRef = querySnapshot.docs[0].ref;

            deleteDoc(docRef)
              .then(() => {
                console.log(`Appointment for ${record.petname} deleted
successfully.`);
                window.location.href = "/";
              })
              .catch((error) => {
                console.error(`Error deleting appointment for
${record.petname}: ${error.message}`);
              });
          } else {
```

```
            console.error(`Expected 1 document, but found
${querySnapshot.size} documents.`);
          }
        })
        .catch((error) => {
          console.error(`Error querying for appointment for ${record.petname}:
${error.message}`);
        });
    }
  };

  const handleButtonClick = () => {
    window.location.href = "/";
  };

  return (
    <Container fluid className='view-body'>
      <h1 className='my-5'>Welcome {record.firstname}</h1>
      <Row>
        <Col md={6}>
          <Card className="my-2">
            <Card.Header className="bg-info text-
white">Appointments</Card.Header>
            <Card.Body>
              <Card.Title>Upcoming Appointments</Card.Title>
              <Card.Text>
                <ul>
                  <li><strong>Pet Name:</strong> {record.petname}</li>
                  <li><strong>Appointment:</strong>
{record.bookingcategory}</li>
                  <li><strong>Date:</strong> {record.date}</li>
                  <li><strong>Time:</strong> {record.time}</li>
                </ul>
              </Card.Text>
              <Button href="Appointment" variant="primary">Schedule
Appointment</Button>
            </Card.Body>
          </Card>
        </Col>
        <Col md={6}>
      <Card className="my-2">
        <Card.Header className="bg-info text-white">Pets</Card.Header>
          <Card.Body>
            <Card.Title>Your Pets</Card.Title>
              <Card.Text>
                <ul>
                  <li><strong>Pet Name:</strong> {record.petname}</li>
                  <li><strong>Age:</strong> {record.petage}</li>
```

```
                <li><strong>Gender:</strong> {record.petgender}</li>
                <li><strong>Animal:</strong> {record.pettype}</li>
                <li><strong>Breed:</strong> {record.petbreed}</li>
                <li><strong>Vaccinated:</strong> {record.petvaccinated}</li>
                <li><strong>Note:</strong> {record.petnote}</li>
              </ul>
            </Card.Text>
            <Button variant='warning' onClick={() => setShowModal(true)}><i
className="bi bi-pencil"></i></Button>
          </Card.Body>
        </Card>
      </Col>
    </Row>

      <div className="my-4">
        <Table striped bordered hover>
          <thead>
            <tr>
              <th></th>
              <th>Pet</th>
              <th>Appointment Type</th>
              <th>Date</th>
              <th>Time</th>
              <th>Action</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td></td>
              <td>{record.petname}</td>
              <td>{record.bookingcategory}</td>
              <td>{record.date}</td>
              <td>{record.time}</td>
              <td>
              <Button variant='warning' onClick={() =>
setShowBookingModal(true)}><i className="bi bi-pencil"></i></Button>
              <Button variant="danger" onClick={handleDelete}><i className="bi
bi-trash"></i></Button>
              </td>
            </tr>
          </tbody>
        </Table>
      </div>
      <Button onClick={handleButtonClick}>Back</Button>
      <br />
      <br />

    {/* Update Pet Modal */}
```

```
<Modal show={showModal} onHide={handleCloseModal}>
  <Modal.Header closeButton>
    <Modal.Title>Update Appointment</Modal.Title>
  </Modal.Header>
  <Modal.Body>
    <Form>
      <Form.Group controlId='petName'>
        <Form.Label>Pet Name:</Form.Label>
        <Form.Control
          type='text'
          placeholder='Enter pet name'
          value={petName}
          onChange={(e) => setPetName(e.target.value)}
        />
      </Form.Group>
      <Form.Group controlId='petAge'>
        <Form.Label>Pet Age:</Form.Label>
        <Form.Control
          type='number'
          placeholder='Enter pet age'
          value={petAge}
          onChange={(e) => setPetAge(e.target.value)}
        />
      </Form.Group>
      <Form.Group controlId='petGender'>
        <Form.Label>Pet Gender:</Form.Label>
        <Form.Control
          as='select'
          value={petGender}
          onChange={(e) => setPetGender(e.target.value)}
        >
          <option>Male</option>
          <option>Female</option>
        </Form.Control>
      </Form.Group>
      <Form.Group controlId='petVaccinated'>
        <Form.Label>Pet Vaccinated:</Form.Label>
        <Form.Check
          type='checkbox'
          label='Yes'
          checked={petVaccinated}
          onChange={(e) => setPetVaccinated(e.target.checked)}
        />
      </Form.Group>
      <Form.Group controlId='petType'>
        <Form.Label>Pet Type:</Form.Label>
        <Form.Control
          type='text'
```

```
              placeholder='Enter pet type'
              value={petType}
              onChange={(e) => setPetType(e.target.value)}
            />
          </Form.Group>
          <Form.Group controlId='petBreed'>
            <Form.Label>Pet Breed:</Form.Label>
            <Form.Control
              type='text'
              placeholder='Enter pet breed'
              value={petBreed}
              onChange={(e) => setPetBreed(e.target.value)}
            />
          </Form.Group>
          <Form.Group controlId='petNote'>
            <Form.Label>Pet Note:</Form.Label>
            <Form.Control
              as='textarea'
              placeholder='Enter pet note'
              value={petNote}
              onChange={(e) => setPetNote(e.target.value)}
            />
          </Form.Group>
        </Form>
      </Modal.Body>
      <Modal.Footer>
        <Button variant='secondary' onClick={handleCloseModal}>Cancel</Button>
        <Button variant='primary' onClick={handleSaveChanges}>Save
Changes</Button>
      </Modal.Footer>
    </Modal>

    {/*Update Booking Modal*/}
    <Modal show={showBookingModal} onHide={handleCloseBookingModal}>
      <Modal.Header closeButton>
        <Modal.Title>Edit Booking Details</Modal.Title>
      </Modal.Header>
      <Modal.Body>
        <Form>
          <Form.Group controlId="petName">
            <Form.Label>Pet Name</Form.Label>
            <Form.Control type="text" value={petName} onChange={(e) =>
setPetName(e.target.value)} />
          </Form.Group>
          <Form.Group controlId="bookingCategory">
            <Form.Label>Appointment Type:</Form.Label>
            <Form.Control as="select" value={bookingCategory} onChange={(e) =>
setBookingCategory(e.target.value)} required>
```

```
                <option disabled hidden value="">Select booking
category</option>
                <option value="Checkup">Checkup</option>
                <option value="Emergency">Emergency</option>
                <option value="Teeth Cleaning">Teeth Cleaning</option>
                <option value="Grooming">Grooming</option>
                <option value="Neutering/Spaying">Neutering/Spaying</option>
                <option value="Follow-up Checkup">Follow-up Checkup</option>
                <option value="Surgery">Surgery</option>
                <option value="Vaccinations and Boosters">Vaccinations and
Boosters</option>
              </Form.Control>
            </Form.Group>
            <Form.Group controlId="date">
              <Form.Label>Date</Form.Label>
              <Form.Control as="select" value={date} onChange={(e) =>
setDate(e.target.value)}>
              <option value="">Select a date...</option>
                {availableDates.map(date => (
                  <option key={date.toISOString()}
value={date.toLocaleDateString(undefined, { weekday: 'short', year: 'numeric',
month: 'short', day: 'numeric' })}>
                  {date.toLocaleDateString(undefined, { weekday: 'short', year:
'numeric', month: 'short', day: 'numeric' })}
                  </option>
                ))}
              </Form.Control>
            </Form.Group>
            <Form.Group controlId="time">
              <Form.Label>Time</Form.Label>
              <Form.Control as="select" value={time} onChange={(e) =>
setTime(e.target.value)}>
                <option value="">Select a time...</option>
                {times.map(time => (
                  <option key={time} value={time}>{time}</option>
                ))}
              </Form.Control>
            </Form.Group>
          </Form>
        </Modal.Body>
        <Modal.Footer>
          <Button variant="secondary" onClick={handleCloseBookingModal}>
            Close
          </Button>
          <Button variant="primary" onClick={handleEdit}>
            Save Changes
          </Button>
        </Modal.Footer>
```

```
    </Modal>
  </Container>
  )
}

export default ViewAppointment;
```

4.17 Contact.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Contact.js
* Description: Page for pet owner to contact vet.
* Date: 14/04/2023
* EmailJS. Send email from Javascript – no server code required. [online]
Available at: https://www.emailjs.com/.
* npm. react-moment. [online] Available at:
https://www.npmjs.com/package/react-moment. [Accessed 17/01/2023].
*/

import React, { useState, useRef } from "react";
import { Row, Col, Form, Button } from "react-bootstrap";
import '../App.css';
import emailjs from '@emailjs/browser';

const Contact = () => {
  const form = useRef();

  const sendEmail = (e) => {
    e.preventDefault();

    emailjs.sendForm('service_749d259', 'template_0po9thn', form.current,
'7M3zeW5o3C-7sb5zq')
      .then((result) => {
          console.log(result.text);
          e.target.reset();
          setShowBanner(true);
          setTimeout(() => setShowBanner(false), 3000);
        }, (error) => {
            console.log(error.text);
      });
  };

  const [message, setMessage] = useState("");
  const [showBanner, setShowBanner] = useState(false);

  const handleMessageChange = (event) => {
    setMessage(event.target.value);
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    console.log("Message submitted:", message);
  };
```

```
  const [formData, setFormData] = useState({
    user_name: '',
    user_email: '',
    message: ''
  });

  const validateForm = () => {
    if (!formData.user_name || !formData.user_email || !formData.message) {
      return false;
    }
    return true;
  };

  return (
    <>
    <div className="center-left">
      <p>Have a Query</p>
      <p>but dont want to</p>
      <p>book an appointment?</p>
    </div>
    {showBanner && <div style={{backgroundColor: 'green', color: 'white',
padding: '10px', position: 'fixed', bottom: '0', left: '0', width:
'100%'}}>Message sent successfully!</div>}
      <div Container className="booking-form">
        <Row>
          <Col md={{ span: 8, offset: 2 }}>
            <h2>Contact Us</h2>
            <Form ref={form} onSubmit={sendEmail}>
              <Form.Group controlId="formBasicName">
                <Form.FloatingLabel>Name</Form.FloatingLabel>
                  <Form.Control type="text" name="user_name"
placeholder="Enter your name" onChange={(e) => setFormData({...formData,
user_name: e.target.value})} />
              </Form.Group>

              <Form.Group controlId="formBasicEmail">
                <Form.FloatingLabel>Email address</Form.FloatingLabel>
                  <Form.Control type="email" name="user_email"
placeholder="Enter your email" onChange={(e) => setFormData({...formData,
user_email: e.target.value})} />
              </Form.Group>

              <Form.Group controlId="formBasicMessage">
                <Form.FloatingLabel>Message</Form.FloatingLabel>
                  <Form.Control as="textarea" rows={3} placeholder="Enter your
message" name="message" maxLength={200} value={formData.message} onChange={(e)
=> setFormData({...formData, message: e.target.value})} />
```

```
                  <div
                 style={{
                   textAlign: "right",
                   marginTop: "-30px",
                   marginRight: "20px",
                   fontSize: "10px",
                   }}
                 >
                 {message.length}/{200} characters
                 </div> 
               </Form.Group>
                  
             <Button variant="primary" type="submit" value="Send"
disabled={!validateForm()}>
               Submit
             </Button>
             <div className='animated-clip-contact'></div>
           </Form>
         </Col>
       </Row>
     </div></>
  );
};

export default Contact;
```

4.18 SignUp.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: SignUp.js
* Description: Vet SignUp page.
* Date: 14/04/2023
*/

import React, {useState} from 'react';
import { NavLink, useNavigate } from 'react-router-dom';
import { createUserWithEmailAndPassword } from 'firebase/auth';
import { auth } from '../firebaseLogin';
import { getAuth, sendEmailVerification } from "firebase/auth";
import '../App.css';
import { blue } from '@mui/material/colors';
import styled from 'styled-components';

const SignUp = () => {
    const navigate = useNavigate();
    const [email, setEmail] = useState('')
    const [password, setPassword] = useState('');
    const [userExists, setUserExists] = useState(false);
    const auth = getAuth();


    const onSubmit = async (e) => {
        e.preventDefault();
        await createUserWithEmailAndPassword(auth, email, password)
          .then((userCredential) => {
            // Signed in
            const user = userCredential.user;
            console.log(user);
            // Send email verification
            sendEmailVerification(user);
            navigate('/Login');
          })
          .catch((error) => {
            const errorCode = error.code;
            const errorMessage = error.message;
            console.log(errorCode, errorMessage);
            if (errorCode === 'auth/email-already-in-use') {
              setUserExists(true);
            }
          });
      };
```

```
  return (
    <main >
        <section >
            <div className='signup-form-container'>
                <div>
                <form className='signup-form'>
                    <div>
                    {userExists && (
                        <div className='error-banner'>Email and Password
invalid</div>
                        )}
                        <label htmlFor="email-address">Email address</label>
                            <input
                                type="email"
                                label="Email address"
                                value={email}
                                onChange={(e) => setEmail(e.target.value)}
                                required
                                placeholder="Email address"

                            />
                        </div>

                        <div>
                            <label htmlFor="password">
                                Password
                            </label>
                            <input
                                type="password"
                                label="Create password"
                                value={password}
                                onChange={(e) => setPassword(e.target.value)}
                                required
                                placeholder="Password"
                            />
                        </div>

                        <button className='signup-form button'
                            type="submit"
                            onClick={onSubmit}
                        >
                            Sign up
                        </button>

                <div className='animated-clip'>
                        </div>
                    </form>
```

```
                    <p>
                        Already have an account?{' '}
                        <NavLink to="/login" >
                            Sign in
                        </NavLink>
                    </p>
                </div>
            </div>
        </section>
    </main>
  )
}

export default SignUp
```

4.19 Login.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Login.js
* Description: Vet Login page.
* Date: 14/04/2023
*/

import React, { useState } from 'react';
import { signInWithEmailAndPassword } from 'firebase/auth';
import { auth } from '../firebaseLogin';
import { NavLink, useNavigate } from 'react-router-dom';
import NavbarLoggedIn from '../common/NavbarVet'; // import the new Navbar
import Navigation from '../common/Navigation'; // import the old Navbar
import '../App.css';
import '.././Vet.css';

const Login = () => {
    const navigate = useNavigate();
    const [email, setEmail] = useState('');
    const [password, setPassword] = useState('');
    const [loggedIn, setLoggedIn] = useState(false);
    const [errorMessage, setErrorMessage] = useState(null); // add state for
error message

    const onLogin = (e) => {
      e.preventDefault();
      signInWithEmailAndPassword(auth, email, password)
        .then((userCredential) => {
          const user = userCredential.user;
          setLoggedIn(true);
          navigate("/VetHome");
          console.log(user);
        })
        .catch((error) => {
          const errorCode = error.code;
          const errorMessage = error.message;
          console.log(errorCode, errorMessage);
          setErrorMessage(errorMessage); // set error message state
        });
    };

    return (
      <>
        <main>
          <section>
```

```
            <div className='login-form-container'>
              <div>
                <form className='login-form'>
                  <div>
                    <label htmlFor='email-address'>Email address</label>
                    <input
                      id='email-address'
                      name='email'
                      type='email'
                      required
                      placeholder='Email address'
                      onChange={(e) => setEmail(e.target.value)}
                    />
                  </div>
                  <div>
                    <label htmlFor='password'>Password</label>
                    <div className='forgot-password'>
                      <NavLink to='/ChangePassword'>Forgot Password?</NavLink>
                    </div>
                    <input
                      id='password'
                      name='password'
                      type='password'
                      required
                      placeholder='Password'
                      onChange={(e) => setPassword(e.target.value)}
                    />
                  </div>

                  <button className='login-form button' type='submit'
onClick={onLogin} href='/VetHome'>
                    Login
                  </button>
                  <div className='animated-clip'></div>
                </form>

                {errorMessage && <p className='error-
message'>{errorMessage}</p>} {/* render error message if there is one */}
                <p className='text-sm text-black text-center'>
                  No account yet? {' '}
                  <NavLink to='/SignUp'>Sign up</NavLink>
                </p>
              </div>
            </div>
          </section>
        </main>

        {loggedIn ? <NavbarLoggedIn /> : <Navigation />}
```

```
      </>
    );
  };

  export default Login;
```

4.20 ChangePassword.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: ChangePassword.js
* Description: Reset password page.
* Date: 14/04/2023
*/

import React, { useState } from "react";
import { Form, Button, Alert } from "react-bootstrap";
import { sendPasswordResetEmail } from 'firebase/auth';
import { auth } from "../firebaseLogin";
import '../App.css';
import '.././Vet.css';

const ChangePassword = () => {

const [email, setEmail] = useState("");
const [error, setError] = useState(null);
const [message, setMessage] = useState(null);
const [isEmailSent, setIsEmailSent] = useState(false);

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

    try {
      if (isEmailSent) {
        setMessage('');
        setError('Password reset email has already been sent. Please check
your inbox');
      } else {
        await sendPasswordResetEmail(auth, email);
        setMessage('Password reset email sent. Please check your inbox.');
        setIsEmailSent(true);
        setError('');
      }
    } catch (err) {
      setError(err.message);
      setMessage('');
    }
  };

    return (
        <>
        <main >
            <section>
```

```
                <div className='login-form-container'>
                    <div>
                    {error && <Alert variant="danger">{error}</Alert>}
                    {message && <Alert variant="success">{message}</Alert>}


                    <Form className='login-form'
onSubmit={handleChangePassword}>

                        <Form.Group controlId="formBasicEmail">
                        <Form.Label>Email address</Form.Label>

                        <Form.Control
                            type="email"
                            placeholder="Enter email"
                            value={email}
                            onChange={(event) => setEmail(event.target.value)}
                            required
                          />
                        </Form.Group>
                        <Button variant="primary" type="submit"
disabled={isEmailSent}>
                        {isEmailSent ? 'Reset Email Sent' : 'Send Reset
Email'}</Button>
                        </Form>
                    </div>
                </div>
            </section>
        </main>
     </>
    )
}

export default ChangePassword;
```

4.21 Sidebar.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: Sidebar.js
* Description: Vet sidebar navigation.
* Date: 14/04/2023
*/

import React, { useState, useEffect } from 'react';
import { Nav, Card } from 'react-bootstrap';
import { Calendar, momentLocalizer } from 'react-big-calendar';
import moment from 'moment';
import 'react-big-calendar/lib/css/react-big-calendar.css';
import firebase from '../../firebase';
import { collection, getDocs } from "firebase/firestore";

//Logo created from: https://logomakr.com
import Logo from '../../images/logo.png';

const Sidebar = () => {
    const localizer = momentLocalizer(moment);
    const [calendarDate, setCalendarDate] = useState([]);
    const [appointments, setAppointments] = useState([]);
    const appointmentCollectionRef = collection(firebase, "appointments");

    useEffect(() => {
      const getAllAppointments = async () => {
        // Fetch all documents from the "appointments" collection
        const data = await getDocs(appointmentCollectionRef);
        // Map each document to an object with its data and ID
        setAppointments(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
})));
      };
      // Call the "getAllAppointments" function only once, when the component
mounts
      getAllAppointments();
    }, []);

    useEffect(() => {
      // Update the calendar date whenever the appointments state changes
      setCalendarDate(new Date());
    }, [appointments]);

    const getAppointmentDates = (month) => {
      // Filter appointments by month
      const filteredAppointments = appointments.filter((appointment) => {
```

```
        return moment(appointment.date.toDate()).format('MMMM YYYY') ===
moment(month).format('MMMM YYYY');
      });
      // Map filtered appointments to an array of dates with appointments
      const appointmentDates = filteredAppointments.map((appointment) => {
        return moment(appointment.date.toDate()).date();
      });
      return appointmentDates;
    };

    const renderDay = (date, events) => {
      // Get dates with appointments for the current month
      const appointmentDates = getAppointmentDates(calendarDate);
      // Check if the current day has an appointment
      const hasAppointment = appointmentDates.includes(moment(date).date());
      return (
        <div className={hasAppointment ? 'has-appointment' : ''}>
          {moment(date).date()}
        </div>
      );
    };

    return (
      <div className="sidebar bg-light">
        <h3><a href="/">
        <img src={Logo} alt="Logo" />
      </a></h3>
        <Nav defaultActiveKey="/" className="flex-column">
          <Nav.Link href="#">Dashboard</Nav.Link>
          <Nav.Link href="#past-appointments">Past Appointments</Nav.Link>
          <Nav.Link href="#upcoming-appointments">Upcoming
Appointments</Nav.Link>
          <Nav.Link href="#reports">Diagnostic Reports</Nav.Link>
        </Nav>
        <Card>
          <Card.Body>
            <div style={{ height: 500, width: 350 }}>
              <Calendar
                localizer={localizer}
                defaultDate={calendarDate}
                events={appointments}
                style={{ width: '80%', height: '100%' }}
                views={['month']}
                dayLayoutAlgorithm={'workweek'}
                components={{
                  day: {
                    header: () => null,
                    cell: renderDay,
```

```
                    },
                  }}
                />
              </div>
            </Card.Body>
          </Card>
        </div>
      );
    };

export default Sidebar;
```

4.22 RecentActivitySidebar.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: RecentActivitySidebar.js
* Description: styles recent activity sidebar.
* Date: 14/04/2023
*/

const RecentActivitySidebar = ({ recentActivity }) => {
  return (
    <div className="recent-activity-sidebar">

    </div>
  );
};

export default RecentActivitySidebar;
```

4.23 MainContent.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: MainContent.js
* Description: Vet main content page.
* Date: 14/04/2023
* npm. react-big-calendar. [online] Available at:
https://www.npmjs.com/package/react-big-calendar. [Accessed 17/01/2023].
*/

import React, { useState, useEffect } from 'react';
import moment from 'moment';
import { onAuthStateChanged } from "firebase/auth";
import { signOut } from "firebase/auth";
import { auth } from '../firebaseLogin';
import { useNavigate } from 'react-router-dom';
import { Calendar, momentLocalizer } from 'react-big-calendar';
import firebase from '../firebase'
import { collection, getDocs, deleteDoc, doc } from "firebase/firestore";
import { Container, Row, Col, Card, Table, Button, Nav, Navbar, NavLink } from
'react-bootstrap';

const localizer = momentLocalizer(moment);

function MainContent() {
    const navigate = useNavigate();
    const [appointments, setAppointments] = useState([]);
    const appointmentCollectionRef = collection(firebase, "appointments");

    useEffect(() => {
        const getAllAppointments = async () => {
          const data = await getDocs(appointmentCollectionRef);
          setAppointments(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
})));
        };
        getAllAppointments();
      }, []);

      // Calculate the start and end dates of the next week
      const now = new Date();
      const nextWeek = new Date(now.getTime() + 14 * 24 * 60 * 60 * 1000);

      const upcomingAppointments = appointments.filter(
        (appointment) => new Date(appointment.date) >= now && new
Date(appointment.date) < nextWeek
      );
```

```
useEffect(()=>{
    onAuthStateChanged(auth, (user) => {
        if (user) {
          // User is signed in
          const uid = user.uid;
          // ...
          console.log("uid", uid)
        } else {
          // User is signed out
          // ...
          console.log("user is logged out")
        }
      });
}, [])

const handleLogout = () => {
    signOut(auth).then(() => {
    // Sign-out successful.
        navigate("/");
        console.log("Signed out successfully")
    }).catch((error) => {
    // An error happened.
    });
}

const recentActivity = appointments.filter(
  (appointment) => new Date(appointment.date) < now
);

return (
<div className="main-content">
  <h2>Dashboard</h2>
    <p>You have: {upcomingAppointments.length} future appointments </p>
    <div className="upcoming">
      <table responsive striped bordered hover className="table">
            <thead>
              <tr>
                <th>Date</th>
                <th>Time</th>
                <th>Owner Name</th>
                <th>Pet Name</th>
                <th>Reason for Visit</th>
              </tr>
            </thead>
            <tbody>
              {upcomingAppointments.map((appointment) => (
                <tr key={appointment.id}>
```

```
                        <td>{moment(appointment.date).format('MMMM Do
YYYY')}</td>
                        <td>{moment(appointment.date).format('h:mm a')}</td>
                        <td>{appointment.firstname}</td>
                        <td>{appointment.petname}</td>
                        <td>{appointment.bookingcategory}</td>
                      </tr>
                    ))}
                  </tbody>
              </table>
        </div>
        <div className="calendar">
              <Calendar
                localizer={localizer}
                events={appointments}
                startAccessor="date"
                endAccessor="date"
                messages={{
                  today: 'Today',
                }}
              />
        </div>
      </div>
  );

}

export default MainContent;
```

4.24 RecentActivity.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: RecentActivity.js
* Description: Recent activity page.
* Date: 14/04/2023
*/

import React, { useState, useEffect } from 'react';
import moment from 'moment';
import { useNavigate } from 'react-router-dom';
import firebase from '../firebase'
import { collection, getDocs, deleteDoc, doc } from "firebase/firestore";
import { Container, Row, Col, Card, Table, Button, Nav, Navbar, NavLink } from
'react-bootstrap';

function RecentActivity() {
    const [appointments, setAppointments] = useState([]);
    const appointmentCollectionRef = collection(firebase, "appointments");

    useEffect(() => {
        const getAllAppointments = async () => {
          const data = await getDocs(appointmentCollectionRef);
          setAppointments(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
})));
        };
        getAllAppointments();
      }, []);

      // Calculate the start and end dates of the next week
      const now = new Date();
      const nextWeek = new Date(now.getTime() + 14 * 24 * 60 * 60 * 1000);

      const upcomingAppointments = appointments.filter(
        (appointment) => new Date(appointment.date) >= now && new
Date(appointment.date) < nextWeek
      );

      const recentActivity = appointments.filter(
          (appointment) => new Date(appointment.date) < now
        );

  return (
    <div className="recent-activity">
      <h2>Recent Activity</h2>
      {recentActivity.length > 0 ? (
```

```
                <Table striped bordered hover>
                  <thead>
                    <tr>
                      <th>Date</th>
                      <th>Time</th>
                      <th>Owner Name</th>
                      <th>Pet Name</th>
                      <th>Reason for Visit</th>
                    </tr>
                  </thead>
                  <tbody>
                    {recentActivity.map((appointment) => (
                      <tr key={appointment.id}>
                        <td>{moment(appointment.date).format('MMMM Do
YYYY')}</td>
                        <td>{moment(appointment.date).format('h:mm a')}</td>
                        <td>{appointment.firstname}</td>
                        <td>{appointment.petname}</td>
                        <td>{appointment.bookingcategory}</td>
                      </tr>
                    ))}
                  </tbody>
                </Table>
              ) : (
                <p>No recent activity to display.</p>
              )}
              <p>You have: {upcomingAppointments.length} future appointments
</p>
    </div>
  );
}

export default RecentActivity;
```

4.25 VetHome.js

```
/*
* Student: Emma O'Connor
* Student No: C00237292
* Title: VetHome.js
* Description: Main vet page.
* Date: 14/04/2023
*/

import React, { useState, useEffect } from 'react';
import { onAuthStateChanged } from "firebase/auth";
import { signOut } from "firebase/auth";
import { auth } from '../firebaseLogin';
import { useNavigate } from 'react-router-dom';
import { Row, Col, Card, Table, Button, Modal, Form } from 'react-bootstrap';
import { Calendar, momentLocalizer } from 'react-big-calendar';
import moment from 'moment';
import 'react-big-calendar/lib/css/react-big-calendar.css';
import firebase from '../firebase'
import { collection, getDocs, addDoc, deleteDoc, updateDoc, getDoc, doc,
where, query } from "firebase/firestore";
import '../Vet.css';

import RecentActivitySidebar from "../Vet/components/RecentActivitySidebar";
import Sidebar from '../Vet/components/Sidebar';

const localizer = momentLocalizer(moment);

const VetHome = () => {
    const navigate = useNavigate();
    const [appointments, setAppointments] = useState([]);
    const appointmentCollectionRef = collection(firebase, "appointments");
    const [showOlderAppointments, setShowOlderAppointments] = useState(false);
    const [reports, setReports] = useState([]);
    const [modalContent, setModalContent] = useState('');
    const [showModal, setShowModal] = useState(false);
    const [showUpdateModal, setShowUpdateModal] = useState(false);
    const [selectedReport, setSelectedReport] = useState(null);
    const [searchQuery, setSearchQuery] = useState('');
    const [showForm, setShowForm] = useState(false);
    const [selectedPet, setSelectedPet] = useState("");
    const [showOwnerDetails, setShowOwnerDetails] = useState(false);
    const [note, setNote] = useState("");
    const [noteError, setNoteError] = useState("");
    const [weight, setWeight] = useState("");
    const [weightError, setWeightError] = useState("");
    const [bodyTemp, setBodyTemp] = useState("");
```

```
    const [bodyTempError, setBodyTempError] = useState("");
    const [externalParasites, setExternalParasites] = useState("");
    const [externalParasitesError, setExternalParasitesError] = useState("");
    const [bpm, setBpm] = useState("");
    const [bpmError, setBpmError] = useState("");
    const [medicalHistory, setMedicalHistory] = useState("");
    const [ravsScore, setRavsScore] = useState("");
    const [ravsScoreError, setRavsScoreError] = useState("");
    const [sortOrder, setSortOrder] = useState('asc');

    useEffect(() => {
      const getAllAppointments = async () => {
        // Fetch all documents from the "appointments" collection
        const data = await getDocs(appointmentCollectionRef);
        // Map each document to an object with its data and ID
        setAppointments(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
}))));
      };
      // Call the "getAllAppointments" function only once, when the component
mounts
      getAllAppointments();
    }, []);

      // Calculate the start and end dates of the next week
      const now = new Date();
      const nextWeek = new Date(now.getTime() + 14 * 24 * 60 * 60 * 1000);

      const upcomingAppointments = appointments.filter(
        (appointment) => new Date(appointment.date) >= now && new
Date(appointment.date) < nextWeek
      );

    useEffect(()=>{
        onAuthStateChanged(auth, (user) => {
            if (user) {
              // User is signed in
              const uid = user.uid;
              // ...
              console.log("uid", uid)
            } else {
              // User is signed out
              // ...
              console.log("user is logged out")
            }
          });
    }, [])

    const handleLogout = () => {
```

```
        signOut(auth).then(() => {
        // Sign-out successful.
            navigate("/");
            console.log("Signed out successfully")
        }).catch((error) => {
        // An error happened.
        });
    }

    const recentActivity = appointments.filter((appointment) => {
      if (showOlderAppointments) {
        return new Date(appointment.date) < now;
      } else {
        const oneDayAgo = new Date(now.getTime() - 24 * 60 * 60 * 1000);
        return (
          new Date(appointment.date) >= oneDayAgo &&
          new Date(appointment.date) < now
        );
      }
    });

    function getPastAppointments(appointments) {
      const today = new Date();
      today.setHours(0, 0, 0, 0);
      return appointments.filter((appointment) => {
        const appointmentDate = new Date(appointment.date);
        appointmentDate.setHours(0, 0, 0, 0);
        return appointmentDate < today;
      });
    }

    const handleSearch = (event) => {
    setSearchQuery(event.target.value);
};

const filteredAppointments = upcomingAppointments.filter(
    (appointment) =>
      appointment.petname.toLowerCase().includes(searchQuery.toLowerCase())
);

function handleButtonClick(petname) {
    setSelectedPet(petname);
    setShowForm(true);
}

function AppointmentDetails({ appointments }) {
return appointments.map((appointment) => (
    <div key={appointment.id}>
```

```
      <p>Pet Name: {appointment.petname}</p>
      <p>Age: {appointment.petage}</p>
      <p>Animal Type: {appointment.pettype}</p>
      <p>Sex: {appointment.petgender}</p>
      <p>Breed of Animal: {appointment.petbreed}</p>
      <p>Vaccinated: {appointment.petvaccinated}</p>
      <p>Notes: {appointment.petnote}</p>
    </div>
  ));
}

function handleViewClick(selectedPet) {
  const petAppointments = appointments.filter((appointment) =>
appointment.petname === selectedPet);

  const appointmentDetails = petAppointments.map((appointment) => (
    <div key={appointment.id}>
      <p>Pet Name: {appointment.petname}</p>
      <p>Age: {appointment.petage}</p>
      <p>Animal Type: {appointment.pettype}</p>
      <p>Sex: {appointment.petgender}</p>
      <p>Breed of Animal: {appointment.petbreed}</p>
      <p>Vaccinated: {appointment.petvaccinated}</p>
      <p>Notes: {appointment.petnote}</p>
    </div>
  ));

  const ownerDetails = appointments
    .filter((appointment) => appointment.petname === selectedPet)
    .map((appointment) => (
      <div key={appointment.id}>
        <p>Name: {appointment.firstname}, {appointment.lastname}</p>
        <p>Phone: {appointment.phone}</p>
        <p>Email: {appointment.email}</p>
      </div>
    ));

  let isPetView = true; // initialize view mode as petview

  const toggleView = () => {
    isPetView = !isPetView; // toggle view mode
    setModalContent(
      <div>
        <h3>{selectedPet}'s Appointment {isPetView ? 'Pet' : 'Owner'}
Profile:</h3>
        <br />
        {isPetView ? appointmentDetails : <>{ownerDetails}</>}
```

```
        <button onClick={toggleView}>Switch to {isPetView ? 'Owner' : 'Pet'}
Details</button>
         
        <button onClick={() =>
window.open(`mailto:${ownerDetails[0].props.children[2].props.children}`)}>
          Contact
        </button>
      </div>
    );
  };

  setModalContent(
    <div>
      <h3>{selectedPet}'s Appointment Pet Profile:</h3>
      <br></br>
      {appointmentDetails}
      <button onClick={toggleView}>Owner Details</button>
    </div>
  );
  setShowModal(true);
}

  function handleCloseForm() {
    setShowForm(false);
    setNote("");
    setNoteError("");
    setWeight("");
    setWeightError("");
    setBodyTemp("");
    setBodyTempError("");
    setExternalParasites("");
    setExternalParasitesError("");
    setBpm("");
    setBpmError("");
    setMedicalHistory("");
    setRavsScore("");
    setRavsScoreError("");
  }

  function handleNoteChange(event) {
    const value = event.target.value;
    setNote(value);
    if (!value) {
      setNoteError("Note is required.");
    } else {
      setNoteError("");
    }
  }
```

```
  function handleWeightChange(event) {
    const value = event.target.value;
    setWeight(value);
    if (!value) {
      setWeightError("Weight is required.");
    } else {
      setWeightError("");
    }
  }

  function handleBodyTempChange(event) {
    const value = event.target.value;
    setBodyTemp(value);
    if (!value) {
      setBodyTempError("Body temperature is required.");
    } else {
      setBodyTempError("");
    }
  }

  function handleBpmChange(event) {
    const value = event.target.value;
    setBpm(value);
    if (!value) {
      setBpmError("BPM is required.");
    } else {
      setBpmError("");
    }
  }

  function handleExternalParasitesChange(event) {
    const value = event.target.value;
    setExternalParasites(value);
    if (!value) {
      setExternalParasitesError("External Parasite identification is
required.");
    } else {
      setExternalParasitesError("");
    }
  }

  function handleRavsScoreChange(event) {
    const value = event.target.value;
    setRavsScore(value);
    if (!value) {
      setRavsScoreError("RAVS Score is required.");
    } else {
```

```
      setRavsScoreError("");
    }
  }

  async function handleFormSubmit(event) {
    event.preventDefault();

    try {
      const newReport = {
        petname: selectedPet,
        note,
        weight,
        bodyTemp,
        externalParasites,
        bpm,
        ravsScore,
        createdAt: now.toISOString()
      };

      const reportsCollectionRef = collection(firebase, "reports");
      const querySnapshot = await getDocs(
        query(
          reportsCollectionRef,
          where("petname", "==", selectedPet),
        )
      );
      const existingReport = querySnapshot.docs[0];

      if (existingReport) {
        const existingReportData = existingReport.data();
        const mergedReport = {
          ...existingReportData,
          ...newReport,
          updatedAt: now.toISOString()
        };
        await updateDoc(existingReport.ref, mergedReport);
        alert('Report updated successfully!');
      } else {
        await addDoc(reportsCollectionRef, newReport);
        alert('Report submitted successfully!');
      }

      handleCloseForm();
    } catch (error) {
      console.error(error);
      alert('Failed to submit report!');
    }
  }
```

```
  async function handleReportClick() {
    try {
      const reportsCollectionRef = collection(firebase, 'reports');
      const querySnapshot = await getDocs(reportsCollectionRef);
      const reportsData = querySnapshot.docs.map(doc => ({ id: doc.id,
...doc.data() }));
      setReports(reportsData);
    } catch (error) {
      console.error(error);
      alert('Failed to fetch reports!');
    }
  }

  async function handleReportDeleteClick(reportId) {
    const confirmDelete = window.confirm("Are you sure you want to delete this
report?");
    if (confirmDelete) {
      try {
        const reportDoc = doc(firebase, 'reports', reportId);
        await deleteDoc(reportDoc);
        alert('Report deleted successfully!');
      } catch (error) {
        console.error(error);
        alert('Failed to delete report!');
      }
    }
  }

  const toggleSortOrder = () => {
    if (sortOrder === "desc") {
      setSortOrder("asc");
    } else {
      setSortOrder("desc");
    }
  };

  const sortedAppointments = appointments.sort((a, b) => {
    if (sortOrder === "desc") {
      return moment(b.date) - moment(a.date);
    } else {
      return moment(a.date) - moment(b.date);
    }
  });

  function handlePrintClick() {
    const tableRows = reports.map(report => {
      return `
```

```
      <tr>
        <td>${report.createdAt}</td>
        <td>${report.petname}</td>
        <td>${report.note}</td>
        <td>${report.weight}</td>
        <td>${report.bodyTemp}</td>
        <td>${report.externalParasites}</td>
        <td>${report.bpm}</td>
        <td>${report.ravsScore}</td>
      </tr>
    `;
  }).join('');

  const printContents = `
    <html>
      <head>
        <title>PetSpace Reports</title>
        <style>
          table {
            border-collapse: collapse;
            width: 100%;
          }

          th, td {
            border: 1px solid #ddd;
            text-align: left;
            padding: 8px;
          }

          th {
            background-color: #f2f2f2;
          }
        </style>
      </head>
      <body>
        <h2>Reports</h2>
        <table>
          <thead>
            <tr>
              <th>Date Created</th>
              <th>Pet Name</th>
              <th>Detail</th>
              <th>Weight</th>
              <th>Body Temperature</th>
              <th>External Parasites</th>
              <th>BPM</th>
              <th>RAVS Score</th>
            </tr>
```

```
            </thead>
            <tbody>
              ${tableRows}
            </tbody>
          </table>
        </body>
      </html>
    `;

    const originalContents = document.body.innerHTML;
    document.body.innerHTML = printContents;
    window.print();
    document.body.innerHTML = originalContents;
  }

  return (
    <div style={{ 'margin-left': '30px' }} className="vet-home">
      <Row sm={1}>
        <Col sm={2} md={3} >
          <Card >
            <Sidebar />
          </Card>
        </Col>

        <Col sm={10} md={8} style={{ backgroundColor: "white" }}>
          <Card.Header>
            <h1>Welcome to PetSpace</h1>
            <br></br>
          </Card.Header>
          <h2 id='past-appointments'>Recent Activity</h2>
          <hr></hr>
          {recentActivity.length > 0 ? (
            <ul>
              {recentActivity.map((appointment) => (
                <li key={appointment.id}>
                  {moment(appointment.date).format("MMMM Do YYYY")},{" "}
                  {appointment.time},{" "}
                  <strong>{appointment.bookingcategory}</strong>
                  <> for: </>
                  {appointment.petname},
                  <strong> Owner: </strong>
                  {appointment.lastname}, {appointment.firstname}
                </li>
              ))}
            </ul>
          ) : (
            <p>No recent activity to display.</p>
          )}
```

```
          <Button
            variant="primary"
            onClick={() => setShowOlderAppointments(!showOlderAppointments)}
          >
            {showOlderAppointments ? "Hide older appointments" : "Show older
appointments"}
          </Button>
          <h2 id='upcoming-appointments' style={{ marginTop: "20px"
}}>Upcoming Appointments </h2>
          <hr></hr>
          <div>
            <RecentActivitySidebar recentActivity={recentActivity} />
            <p>You have: {upcomingAppointments.length} future appointments for
the next 2 weeks</p>
          </div>
          <br></br>
          <div className="d-flex justify-content-start mb-1">
            <input
                type="text"
                placeholder="Search for pet:"
                value={searchQuery}
                onChange={handleSearch}
              /> 
              <div className="d-flex justify-content-end mb-1">
            <Button variant="primary" onClick={toggleSortOrder}>
              Sort by Date {sortOrder === "desc" ? "↓" : "↑"}
            </Button>
          </div>
          </div>

          <Table responsive striped bordered hover className="table">
            <thead>
              <tr>
                <th>Date</th>
                <th>Time</th>
                <th>Owner</th>
                <th>Pet</th>
                <th>Reason for Visit</th>
                <th>Evaluate</th>
              </tr>
            </thead>
            <tbody>
              {filteredAppointments.length > 0 ? (
                filteredAppointments.map((appointment) => (
                  <tr key={appointment.id}>
                    <td>{moment(appointment.date).format("MMMM Do YYYY")}</td>
                    <td>{appointment.time}</td>
                    <td>{appointment.firstname} {appointment.lastname}</td>
```

```
                    <td>{appointment.petname}</td>
                    <td>{appointment.bookingcategory}</td>
                    <td>
                    <Button variant="primary" onClick={() =>
handleButtonClick(appointment.petname)}><i class="bi bi-file-earmark-
text"></i> Report</Button> 
                    <Button variant="warning" onClick={() =>
handleViewClick(appointment.petname)}><i className="bi bi-eye-
fill"></i> View</Button>
                    </td>
                  </tr>
                ))
              ) : (
                <tr>
                  <td colSpan="5">No pet found with that name.</td>
                </tr>
              )}

            </tbody>
          </Table>

          <br></br>
          <h2 id='reports' className="mb-3">Reports</h2>
          <hr></hr>
          <button className="btn btn-primary mb-3"
onClick={handleReportClick}>Fetch Reports</button>
           <div className="d-flex justify-content-end">
              <button className="btn btn-secondary"
onClick={handlePrintClick}><i className="bi bi-printer"></i> Print
Reports</button>
            </div>
          {reports.length > 0 && (
            <><table className="table" id="report-table">
              <thead>
                <tr>
                  <th>Date Created</th>
                  <th>Pet Name</th>
                  <th>Detail</th>
                  <th>Weight</th>
                  <th>Body Temperature</th>
                  <th>External Parasites</th>
                  <th>BPM</th>
                  <th>RAVS Score</th>
                </tr>
              </thead>
              <tbody>
                {reports.map(report => (
                  <tr key={report.id}>
```

```
                    <td>{report.createdAt}</td>
                    <td>{report.petname}</td>
                    <td>{report.note}</td>
                    <td>{report.weight}</td>
                    <td>{report.bodyTemp}</td>
                    <td>{report.externalParasites}</td>
                    <td>{report.bpm}</td>
                    <td>{report.ravsScore}</td>
                    <td><Button variant='danger' onClick={() =>
handleReportDeleteClick(report.id)}><i className="bi bi-
trash"></i></Button></td>
                  </tr>
                ))}
              </tbody>
            </table>
            </>
          )}

          <Modal show={showModal} onHide={() => setShowModal(false)}>
            <Modal.Body>{modalContent}</Modal.Body>
            <Modal.Footer>
              <Button variant="secondary" onClick={() => setShowModal(false)}>
                Close
              </Button>
            </Modal.Footer>
          </Modal>

          <Modal show={showForm} onHide={handleCloseForm}>
            <Modal.Header closeButton>
              <Modal.Title>Evaluation Report for {selectedPet}</Modal.Title>
            </Modal.Header>
              <Modal.Body>
              <Form onSubmit={handleFormSubmit}>

                <Form.Group as={Col} controlId="formWeight">
                  <Form.Label>Weight (Kg)</Form.Label>
                  <Form.Control required type="number" value={weight}
onChange={handleWeightChange} isInvalid={weightError} />
                  <Form.Control.Feedback
type="invalid">{weightError}</Form.Control.Feedback>
                </Form.Group>

                <Form.Group as={Col} controlId="formBodyTemp">
                  <Form.Label>Body Temp (F)</Form.Label>
                  <Form.Control required type="number" value={bodyTemp}
onChange={handleBodyTempChange} isInvalid={bodyTempError} />
                  <Form.Control.Feedback
type="invalid">{bodyTempError}</Form.Control.Feedback>
```

```
                </Form.Group>

                <Form.Group as={Col} controlId="formBpm">
                  <Form.Label>Pulse (BPM)</Form.Label>
                  <Form.Control required type="number" value={bpm}
onChange={handleBpmChange} isInvalid={bpmError} />
                  <Form.Control.Feedback
type="invalid">{bpmError}</Form.Control.Feedback>
                </Form.Group>

                <Form.Group as={Col} controlId="formExternalParasites">
                  <Form.Label>External Parasites</Form.Label>
                  <Form.Control required as="select" value={externalParasites}
onChange={handleExternalParasitesChange} isInvalid={externalParasitesError}>
                    <option value="" disabled hidden>Select an option</option>
                    <option value="Fleas">Fleas</option>
                    <option value="Ticks">Ticks</option>
                    <option value="Lice">Lice</option>
                    <option value="Mites">Mites</option>
                    <option value="None">None</option>
                  </Form.Control>
                  <Form.Control.Feedback
type="invalid">{externalParasitesError}</Form.Control.Feedback>
                </Form.Group>

                <Form.Group as={Col} controlId="formRavsScore">
                  <Form.Label>RAVS Score</Form.Label>
                  <Form.Control required as="select" value={ravsScore}
onChange={handleRavsScoreChange} isInvalid={ravsScoreError}>
                  <option value="" disabled hidden>Select an option</option>
                    <option value="Excellent">Excellent</option>
                    <option value="Good">Good</option>
                    <option value="Fair">Fair</option>
                    <option value="Poor">Poor</option>
                    <option value="Critical">Critical</option>
                  </Form.Control>
                  <Form.Control.Feedback
type="invalid">{ravsScoreError}</Form.Control.Feedback>
                </Form.Group>

                <Form.Group as={Col} controlId="formNote">
                  <Form.Label>Note</Form.Label>
                    <Form.Control required type="text" value={note}
maxLength={500} onChange={handleNoteChange} isInvalid={noteError} />
                    <div style={{textAlign: "right", marginTop: "-20px",
marginRight: "10px", fontSize: "10px"}}>
                    {note.length}/{500} characters
                    </div>
```

```
                    <Form.Control.Feedback
type="invalid">{noteError}</Form.Control.Feedback>
                  </Form.Group>
                  <br></br>
                  <div className="d-flex justify-content-end">
                    <Button variant="secondary" onClick={handleCloseForm}
className="ml-auto">Close</Button>
                    <Button variant="primary" type="submit" className="mr-
0">Save Report</Button>
                  </div>
                </Form>
            </Modal.Body>
          </Modal>
        </Col>
      </Row>
    </div>
    );
}

export default VetHome;
```

4.26 App.css

```
/*App.css*/

body {
  background-image: url('E:\PROJECT\petspace-app\petspace-
app\src\home_image.jpg');
  background-size: cover;
  background-position: center;
  background-repeat: no-repeat;
}

/* -------View.js--------- */

.view-body {
  background-color: whitesmoke;
  background-size: cover;
  background-position: center;
  background-repeat: no-repeat;
  width: 100%;
  max-width: 900px;
  margin-top: 70px;
  margin-left: 260px;
}

.view {
  width: 90%;
  margin-left: 35px;
  margin-bottom: 10px;
}

/*----- Contact.js -----*/

.center-left {
  position: relative;
  float: left;
  margin-top: 190px;
  margin-left: 70px;
  text-align: left;
}

.center-left p {
  font-family: Verdana, Geneva, Tahoma, sans-serif;
  font-size: 48px;
  color: white;
}

/* ---------FormDetails-------------- */
```

```
/* adjust the container div to position formDetails over the image */
.form-button-text-on-image {
  position: absolute;
  top: 65%;
  left: 50%;
  transform: translate(-50%, -50%);
  background-color: rgba(255, 255, 255, 0.8); /* add some opacity to the
background */
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.75);
  max-width: 500px;
  max-height: 600px;
  width: 100%;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

/* style the form elements */
.form-button-text-on-image form {
  display: absolute;
  flex-direction: column;
  align-items: center;
  overflow-y: auto;
  max-height: 50%;
  max-width: 600px;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.form-button-text-on-image input[type="text"],
.form-button-text-on-image input[type="number"],
.form-button-text-on-image input[type="email"],
.form-button-text-on-image select {
  width: 100%;
  padding: 10px;
  margin: 5px 0;
  border-radius: 5px;
  border: none;
  outline: none;
  font-size: 16px;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.form-button-text-on-image button {
  margin-top: 20px;
  padding: 10px;
  border-radius: 5px;
  border: none;
```

```
  outline: none;
  background-color: #0069D9;
  color: white;
  font-size: 16px;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
  cursor: pointer;
}

.form-button-text-on-image button:hover {
  background-color: #0050A8;
}

.booking-form {
  margin-left: auto;
  max-width: 500px;
  padding: 20px;
  background-color: transparent;
  border-radius: 5px;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form label {
  font-weight: bold;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form input[type="text"],
.booking-form input[type="email"],
.booking-form input[type="tel"],
.booking-form textarea {
  width: 100%;
  padding: 12px;
  margin-bottom: 15px;
  border: none;
  border-radius: 4px;
  box-sizing: border-box;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form select {
  width: 100%;
  padding: 12px;
  margin-bottom: 15px;
  border: none;
  border-radius: 4px;
  box-sizing: border-box;
  appearance: none;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
```

```
  background: transparent url("data:image/svg+xml,%3Csvg viewBox='0 0 12 8'
xmlns='http://www.w3.org/2000/svg'%3E%3Cpath d='M1.41 0L0 1.41L6 7.41L12
1.41L10.59 0L6 4.59L1.41 0Z'/%3E%3C/svg%3E") no-repeat right 12px center/18px
18px;
}

.booking-form input[type="submit"] {
  background-color: #4CAF50;
  color: white;
  padding: 12px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: all 0.3s ease-in-out;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form input[type="submit"]:hover {
  background-color: #3e8e41;
}

.booking-form input[type="submit"]:focus {
  outline: none;
}

.booking-form .form-group {
  position: relative;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form .form-group i {
  position: absolute;
  right: 10px;
  top: 50%;
  transform: translateY(-50%);
  color: #ccc;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

.booking-form .form-group input[type="text"]:focus + i,
.booking-form .form-group input[type="email"]:focus + i,
.booking-form .form-group input[type="tel"]:focus + i,
.booking-form .form-group textarea:focus + i {
  color: #333;
}

.booking-form .form-group input[type="text"]:focus,
.booking-form .form-group input[type="email"]:focus,
```

```
.booking-form .form-group input[type="tel"]:focus,
.booking-form .form-group textarea:focus {
  border-color: #4CAF50;
  box-shadow: 0 0 5px rgba(76, 175, 80, 0.4);
}

.progress {
  height: 5px;
  margin-bottom: 20px;
  background-color: #f5f5f5;
  border-radius: 10px;
  overflow: hidden;
}

.progress-bar {
  height: 70%;
  color: #fff;
  text-align: center;
  background-color: #33be7d;
  transition: width 0.6s ease;
}

@media (min-width: 576px) {
  .progress {
    margin-bottom: 5px;
  }
}

@media (min-width: 768px) {
  .progress {
    margin-bottom: 10px;
  }
}

@media (min-width: 992px) {
  .progress {
    margin-bottom: 0px;
  }
}

@media (min-width: 1200px) {
  .progress {
    margin-bottom: 60px;
  }
}

.navDropdown.item{
    font-weight: bold;
```

```
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    background-color: transparent !important;
}

.linkText { color: white ; font-family: Verdana, Geneva, Tahoma, sans-serif;
margin-left: 10px;}

  .form-container {
    position: fixed;
    margin: 2% auto;
    min-height: 70%;
    width: 50%;
    box-shadow: 0 2px 16px rgba(0, 0, 0, 0.6);
    background: white;
    padding: 0px 0px 5px 15px;
    box-sizing: border-box;
    border-radius: 8px;
    text-align: left;
  }

  /* ------------SignUp.js---------- */
.signup-form-container {
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

/* Style the form */
.signup-form {
  max-width: 400px;
  padding: 20px;
  border: 1px solid #ccc;
  border-radius: 5px;
  background-color: #fff;
  box-shadow: 0 2px 16px rgba(0, 0, 0, 0.6);

}

/* Style the form inputs */
.signup-form input[type="text"],
.signup-form input[type="email"],
.signup-form input[type="password"] {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
```

```
}

/* Style the form submit button */
.signup-form button[type="submit"] {
  width: 100%;
  padding: 10px;
  background-color: #007bff;
  border: none;
  color: #fff;
  border-radius: 5px;
  cursor: pointer;
}

.signup-form button[type="submit"]:hover {
  background-color: #0069d9;
}

/* Style the form link */
.signup-form .signup-link {
  text-align: center;
  margin-top: 20px;
}

.signup-form .signup-link a {
  color: #007bff;
}

.signup-form .signup-link a:hover {
  text-decoration: none;
  color: #0069d9;
}

.error-banner {
  background-color: #ffe6e6;
  color: #b30000;
  padding: 10px;
  margin-top: 10px;
}

/* -----------Login.js------------ */
.login-form-container {
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

/* Style the form */
```

```
.login-form {
  max-width: 400px;
  padding: 20px;
  border: 1px solid #ccc;
  border-radius: 5px;
  background-color: #fff;
  box-shadow: 0 2px 16px rgba(0, 0, 0, 0.6);
}

.contains {
  width: 80%;
}

/* Style the form inputs */
.login-form input[type="email"],
.login-form input[type="password"] {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

/* Style the form submit button */
.login-form button[type="submit"] {
  width: 100%;
  padding: 10px;
  background-color: #007bff;
  border: none;
  color: #fff;
  border-radius: 5px;
  cursor: pointer;
}

.login-form button[type="submit"]:hover {
  background-color: #0069d9;
}

/* Style the form link */
.login-form .signup-link {
  text-align: center;
  margin-top: 20px;
}

.login-form .signup-link a {
  color: #007bff;
}
```

```
.login-form .signup-link a:hover {
  text-decoration: none;
  color: #0069d9;
}

.error-message {
  background-color: rgb(255, 173, 173);
  padding: 10px;
  border-radius: 5px;
}

/* ---------VetHome.js------------ */

.vet-dashboard {
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  padding: 20px;
}

.upcoming-appointments {
  width: 100%;
  max-width: 800px;
  margin-bottom: 40px;
}

.recent-activity {
  width: 100%;
  max-width: 800px;
}

h2 {
  font-size: 36px;
  font-weight: bold;
  margin-bottom: 20px;
}

p {
  font-size: 18px;
  margin-bottom: 20px;
}

.rbc-calendar {
  height: 100%;
}

.rbc-event {
```

```
  background-color: #f0ad4e;
  color: #fff;
  border: none;
  border-radius: 0;
  box-shadow: none;
  font-size: 14px;
}

.rbc-event:hover {
  background-color: #ec971f;
  cursor: pointer;
}

.rbc-today {
  background-color: #d9534f;
  color: #fff;
}

.rbc-toolbar-label {
  font-size: 24px;
  font-weight: bold;
}

.rbc-header {
  font-size: 18px;
  font-weight: bold;
  text-transform: uppercase;
  color: #666;
}

.rbc-date-cell {
  padding: 10px;
  border: none;
  text-align: right;
  vertical-align: top;
}

.rbc-date-cell > * {
  margin-bottom: 5px;
}

.rbc-event-label {
  font-size: 14px;
  font-weight: bold;
  text-align: left;
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
```

```
  max-width: 100%;
}

/* ---------Blue Image over page--------- */
/* bennettfeely.com. Clippy – CSS clip-path maker. [online] Available at:
https://bennettfeely.com/clippy/. [Accessed 17/04/2023]. */

.animated-clip {
  width: 100%;
  height: 100%;
  position: absolute;
  top: 0;
  right: 0;
  clip-path: polygon(0 0, 37% 0, 50% 69%, 56% 100%, 0 100%, 0 70%);
  background-color: lightblue;
  z-index: -1;
}

.animated-clip-contact {
  width: 100%;
  height: 100%;
  position: absolute;
  top: 0;
  right: 0;
  clip-path: polygon(64% 0, 100% 0%, 100% 100%, 64% 100%);
  background-color: lightblue;
  z-index: -1;
}
```

4.27 Vet.css

```
/* -------Vet Components --------- */

.vet-dashboard {
    display: flex;
    flex-direction: column;
    align-items: right;
    justify-content: right;
    padding: 20px;
  }

  .upcoming-appointments {
    width: 100%;
    max-width: 800px;
    margin-bottom: 40px;
  }

  .recent-activity {
    width: 80%;
    max-width: 800px;
  }

  h2 {
    font-size: 36px;
    font-weight: bold;
    margin-bottom: 20px;
  }

  p {
    font-size: 18px;
    margin-bottom: 20px;
  }

  body {
    background-color: #bcd0e0;
  }
  .rbc-calendar {
    height: 500px;
    font-size: 14px;
  }

  .rbc-event {
    background-color: #f0ad4e;
    color: #fff;
    border: none;
    border-radius: 0;
    box-shadow: none;
```

```
  font-size: 14px;
}

.rbc-event:hover {
  background-color: #ec971f;
  cursor: pointer;
}

.rbc-today {
  background-color: #d9534f;
  color: #fff;
}

.rbc-toolbar-label {
  font-size: 24px;
  font-weight: bold;
}

.rbc-header {
  font-size: 14px;
  font-weight: bold;
  text-transform: uppercase;
  color: #666;
}

.rbc-row {
  height: 100px;
}

.rbc-date-cell {
  padding: 10px;
  border: none;
  text-align: right;
  vertical-align: top;
}

.rbc-date-cell > * {
  margin-bottom: 5px;
}

.rbc-event-label {
  font-size: 14px;
  font-weight: bold;
  text-align: left;
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
  max-width: 100%;
```

```
}

.forgot-password {
  font-size: 12px;
  color: rgba(68, 26, 255, 0.8);
  font-weight: 500;
  text-decoration: none;
  cursor: pointer;
  text-align: right;

}

.vet-home {
  display: flex;
  flex-direction: column;
  height: 100vh;
  margin-top: 40px;
}

.vet-home .row {
  flex-grow: 1;
}

.vet-home .col {
  padding: 0;
}
```

4.28 package.json

```
{
  "name": "petspace-app",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@emailjs/browser": "^3.10.0",
    "@mui/material": "^5.11.3",
    "@testing-library/jest-dom": "^5.16.5",
    "@testing-library/react": "^13.4.0",
    "@testing-library/user-event": "^13.5.0",
    "bootstrap": "^5.2.3",
    "bootstrap-icons": "^1.10.3",
    "dotenv": "^16.0.3",
    "expo-cli": "^6.1.0",
    "firebase": "^9.15.0",
    "moment": "^2.29.4",
    "moment-business-days": "^1.2.0",
    "nodemailer": "^6.9.1",
    "rc-time-picker": "^3.7.3",
    "react": "18.1.0",
    "react-availability-calendar": "^0.3.13",
    "react-big-calendar": "^1.6.8",
    "react-bootstrap": "^2.7.0",
    "react-calendar": "^4.0.0",
    "react-data-table-component": "^7.5.3",
    "react-day-picker": "^8.4.1",
    "react-dom": "18.1.0",
    "react-router-dom": "^6.6.1",
    "react-schedule-meeting": "^4.0.0",
    "react-scripts": "5.0.1",
    "react-time-picker": "^5.1.0",
    "reactstrap": "^9.1.5",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
```

```
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```