Pet clinic frontend

General description

You will make a React-based frontend for a pet clinic customer registry. The frontend application shall integrate to a Node.js based backend (provided by the course teacher), which you can run locally on your machine. The backend will have some predefined data in it, but the frontend will be able to add and edit some of it.

The frontend shall have two "aspects" – one for the doctor (who can access all pets etc. in the system) and one for a particular pet owner (who can only see own data).

Evaluation

The total points achievable for the project is 50.

There are 10 user stories listed below. Each of the stories is worth 4 points, except for story #10. Story #10 is worth 4 points if Option A is completed, else it does not result in points.

In addition, you will gain additional points bonuses if you perform the following:

- Use of routing (e.g. React Router) is worth 5p
    - This means different URLs in your application render different React pages
    - I would, in fact, not recommend you to try any other way than using routes – it will make your application much more easy to understand and maintain
- Implement test cases for your application, which is worth 5p
    - You must make at least 5 test cases of various functionality in your application
    - I suggest that you use Jest and extend it with e.g. https://testing-library.com/docs/react-testing-library/intro which gives you ability to test behaviour (i.e. what happens when buttons are clicked etc.)
    - It is not enough to make 5 simple tests like the example test in Create-React-App: your tests must be more advanced than that, otherwise you don't get points
        - E.g. ensure that data about a pet (which you get from the backend) is correctly displayed in a component
        - E.g. ensure that links on your app work correctly (check e.g. https://testing-library.com/docs/example-react-router/)
        - If you make forms e.g. using Formik, check out https://testing-library.com/docs/example-react-formik
    - Finally, to get the 5p, the test must **pass** when the **npm test** command is executed

Backend information

The backend is implemented as a Node.js backend and is uploaded as a ZIP to Moodle. You should download it to your computer, unzip it and run **npm install** to install all the dependencies. When you want to start the backend, run **npm start** in the folder to run the backend. Keep the terminal running for as long as you need the backend running. (This will be demoed during classes).

You can have the backend running in the background, and at the same time run the frontend you develop in another terminal window. Your frontend will be able to call the backend because they will be running on different port numbers.

You are not allowed to make any change to the backend functionality. If you detect some issue which must be fixed, please report to the teacher so it can be patched for all students.

Backend endpoints will be:

Method	Path	Purpose	Requires auth?
POST	/login	Logging in and getting a token	No
GET	/pets	Fetching pets (visible for the logged-in user)	Yes
POST	/pets	Create a pet	Yes
GET	/pets/{id}	Get full details of a pet	Yes
PUT	/pets/{id}	Update details of a pet	Yes
GET	/visits	List visits in the system (what is visible to the logged-in user)	Yes
POST	/visits	Create a visit	Yes

The access token used to login shall be put into the “Authorization” header in the HTTP request. The format shall be a Bearer token.

The token used for doctor access is:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxIiwicm9sZSI6ImRvY3RvciIsImlhdCI6MTUxNjIzOTAyMn0.0_MKcjJoHX-Vsjb4vVlWZLZMY-45nMQ22MTXUCAQgng

There will be three pet owners in the database, each with own access token used for pet owner access:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIyIiwicm9sZSI6InBldF9vd25lciIsImlhdCI6MTUxNjIzOTAyMn0.QAtAc6Imr2-NDhRpPcobJfjA20vh_bDk3wMhL_-46Fw

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIzIiwicm9sZSI6InBldF9vd25lciIsImlhdCI6MTUxNjIzOTAyMn0.DA59WWIUeZ-4v4XVyrbXqd9z1I-YlZRCz45oCuyU2T0

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI0Iiwicm9sZSI6InBldF9vd25lciIsImlhdCI6MTUxNjIzOTAyMn0.zqF-TVZor-FIGK_o3_5exzGJy1MuwkscYLjyS9pawVM

User stories

1. As a doctor, I need to list all pets registered in our system so I can find the pet I am looking for.

I expect a list of pets including their name, type, status and when they last visited the clinic. I should be able to click a pet in the list and be taken to see its detailed info.

I need to have a checkbox in the list that allows me to show only pets with status “alive”. If the checkbox is unchecked, the list shall show all pets in my registry.

2. As an owner, I expect to be able to add a new pet to the client registry.

I should be able to enter the name of the pet, the date of birth, and the type of pet info a form. When I submit the form, the pet should be added to the system (unless some error occurred). Ideally, I would be taken to the page showing the detailed info of the pet I just added.

3. As a doctor, I need a page which shows the detailed info about a specific pet.

This shall include the info about the pet, info about the owner and a list of all the visits the pet has had at our clinic. The list of visits shall be ordered in chronological order, with most recent at the top. In this page, I shall be able to see a “doctor’s only” comment about the pet, which is not shown to the pet owner. It is means as internal information for me to write about the pet.

At this page, I should be able to add a new visit for the pet.

4. As a doctor, I need to be able to edit a pet’s info.

I should be able to change the status of the pet to one of the following supported values: “alive”, “deceased”, “missing”, “other”. I should also be able to edit the “doctor’s only” comment.

5. As a doctor, I want to be able to add a visit to a pet.

When a customer calls and wants to make an appointment, I need to be able to add the visit to the database. When I have located the pet and opened its details page, I should be able to add a visit by entering a date (mandatory) and an optional text comment about the reason for the visit.

6. As a doctor, I want to see a list of upcoming visits.

Because I need to know which patients I am seeing on a particular day, I need to be able to list the upcoming visits. They should be sorted in chronological order so that today’s visits are at the top, and future visits lower down in the list.

7. As a pet owner, I want to see a list of my pets

In the pet clinic application, as a pet owner I want to see a list of my pets that are registered at the clinic. By clicking a pet, I should be taken to a page showing the detailed information about the pet. Naturally, I shall not be able to see the pets of any other owners.

8. As a pet owner, I want to see detailed info about my pet

On the detailed info page about one of my pets, I shall see the same information as the doctor, except that as a pet owner I don't see the "doctor's only" text field.

9. As a pet owner, I want to make a visit for my pet.

When I have opened the details page for my pet, I want to be able to create a visit (upcoming reservation) for the pet. I shall enter a date (which must be a future date) and a text comment for the visit (both mandatory).

If the pet already has a visit reserved on the date I have chosen, I will not be able to make a second reservation on that date.

10. Logging in as doctor or pet owner

There needs to be a way for the user to log in as either a doctor or as a pet owner.

Option A: On the front page, make a login form (email + password) which calls the backend POST /login endpoint when submitted. Use the received data to redirect the user to either doctor or pet owner UI.

The credentials to use for logging in are:

Email address	Password
doctor@pets.com	Pet1234
owner1@test.com	qwerty
owner2@woof.net	Bark!
owner3@abc.org	_Dog2023

Option B: On the front page, make two buttons – one for navigating to doctor UI, the other for navigating to pet owner UI. No call to backend's login endpoint, instead use hardcoded access tokens.

Regardless of option A or B being implemented, there must be a logout button which takes the user back to the front page and "erases" the access token stored in the React app's memory.

Submission

As the course mid-assignment, the submission shall be a ZIP of source code containing everything needed to get the frontend built and running. When installed via the ZIP source code, it must work out-of-the-box with the Moodle-provided backend.

Deadline

The assignment has to be submitted by end of February 2024.