Software Tools – Web Technologies Tuesday 4 May 2021

Introduction

The focus of this workshop is to give you an introduction to React, which is a JavaScript library that is used to build user interfaces and single page web applications. React was created by Facebook in 2011. It is open source and actively maintained.

The best online resource for React is: https://reactjs.org/docs/getting-started.html

One of the best introductory tutorials for React is: https://www.taniarascia.com/getting-started-with-react/. In this workshop we are first going to follow some of this tutorial to help familiarise you with key React concepts.

We are then going to look at a full React website that gets data from the Petstore API that we were using in the last Tuesday workshop. This is a more complex React website but you are familiar with the web API that is being accessed and the data that is being displayed. Your task will be to modify some of the code.

Activity 1

In the first activity you are going to follow most of the stages of an excellent introductory tutorial for React. You will not do the initial part of the tutorial or the section on Submitting Form Data (although you can do this after the workshop if you would like to increase your React knowledge).

Step One

You need to download and install node.js on your machine. You can access it here: https://nodejs.org/en/

It's okay to go with the default install options.

Node.js is used for both client-side and server-side programming and can be used to do things like: generate dynamic page content, modify data in a database and manipulate files on a server. It is open source.

When you download node.js you will also download npm, which is a package manager for node.js. It can be used to download and install thousands of modules (JavaScript libraries) that are hosted at www.npmjs.com. We will use npm later in the workshop.

Step Two

We are now going to use Create React App which is an environment that you can use to build your first simple React website.

We are now going to follow the steps detailed in: https://www.taniarascia.com/getting-started-with-react/. Open this webpage and navigate to the section "Create React App". To follow the tutorial you will need to open a terminal. On my Windows machine I use PowerShell. When you run "npx create-react-app react-tutorial" in your terminal you will create a React project in a directory called "react-tutorial". npx is used to run packages, in this instance 'create-react-app'.

```
$ npx create-react-app react-tutorial
```

The tutorial suggests that you explore the project structure and a good tool for doing this is Visual Studio Code which will enable you to open the project folder and view all the subfolders and files contained within it. It is also a good tool for editing the files. You should follow the instructions in the tutorial. It gives you a good introduction to:

JSX – the language used in React that looks similar to HTML but is a combination of JavaScript and XML

Components – these are the building blocks of a React website and can be two types: simple and class components. The main App component contains all other components and is rendered in the root element of the webpage.

Props – short for properties and the way that data is passed in to components.

State – how to store and manipulate data.

Given the amount of time we have available in the workshop, I would recommend that you do the tutorial up to the end of the State section. It is worth doing the Submitting Form Data section after the workshop if you want to increase your knowledge of React. However, I recommend skipping it in the workshop and looking at the Pulling in API Data section.

Step Three

Follow the Pulling in API Data section of the tutorial which shows you how to create a new, simple app that pulls in data from Wikipedia and displays it. It uses fetch to get the data which we used in last Tuesday's workshop. This is good preparation for Activity Two where we are going to look at a more complex React app that pulls in data from the Petstore API.

Activity 2

In this activity you are going to clone a React app that pulls data from the Petstore API and displays it. This React app was developed by Marceli Wac for the Software Tools Unit. It is more complex than the React apps you have been building in the initial tutorial.

Step One

Clone the React app from Github: https://github.com/marceliwac/web-softwaretools-react/

Step Two

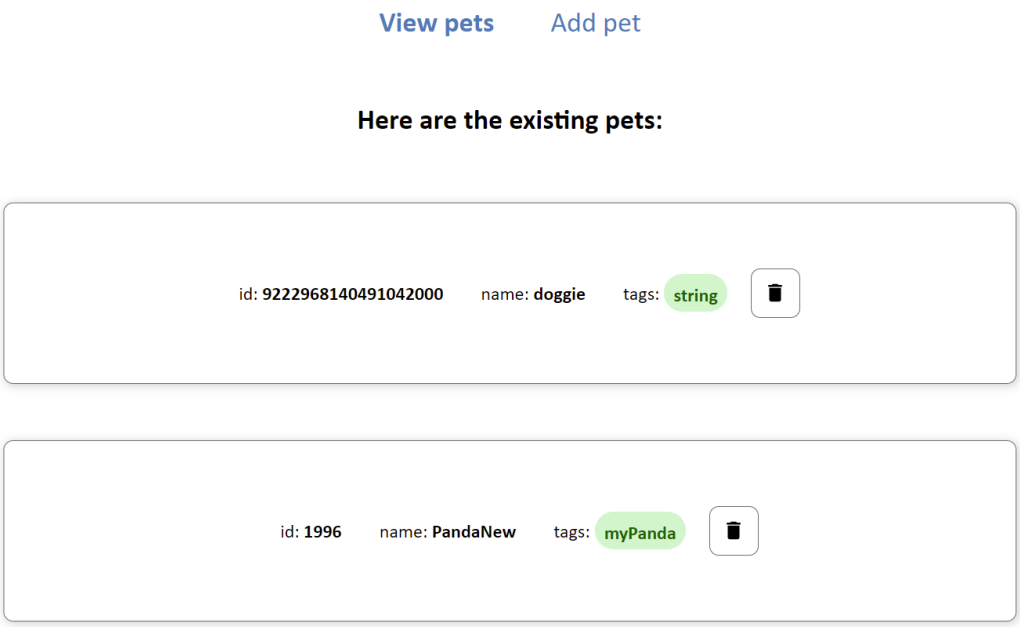
In your terminal, navigate to the project directory and install the dependencies of the project by running:

npm install

Then launch the local version of the website using the command:

npm start

The React app will load in your browser and display the pets that it has pulled from the Petstore API:



Step Two

We are going to explore the React app and look at a number of features.

The React app uses a Router component to manage how users navigate through the different resources in the app and what they view. The two views are 'AddPage' and 'ViewPage' which can be found in separate file in the components folder. For more on the Router component: https://www.educative.io/edpresso/what-is-a-react-router

The app is using axios to pull the data from the Petstore API, rather than Fetch. You can see where this is configured in the api.js file, which is in the tools folder. This api component gets used in AddPage and ViewPage components.

Step Three

One issue with the Petstore API is that it used frequently and so many pets have been added that now when pets are added the default ID they are assigned a value that is beyond the largest int64 value. For this reason, in usePetsApi.js, when get is called only pets with an ID below the overflow value are returned. This is so the ID can be used to delete the pet from the Petstore.

```
React.useEffect(() => {

    api
      .get(path, {params})
      // Ensure that the id used is not the largest possible int64 value (id datatype). This
      // because the number of pets stored in the API has already exceeded it and any DELETE
      // made to pets with this id will fail, since it's not a valid id.
      .then((response) => setData(response.data.filter(p => p.id < 9222968140497253000)))
      .catch((error) => console.error('Whoops, something went wrong!', error));
  }, []);

  return data;
}
```

Your task is to modify the api.post call in AddPage.jsx so that more data is posted to the Petstore API than just the name of the pet. Last Tuesday in the workshop one of the activities involved posting an object containing data in the body of the POST. We can extend the current function, which currently only sends the animal name:

```
    api.post('/pet', {
      name: name,
    }).then(response => {
      setMessage(`Pet has been created with id: ${response.data.id}!`);
      setIsSuccess(true);
    }).catch((e) => {
      setMessage('Something went wrong! Could not add the pet.');
      setIsSuccess(false);
      console.error(e);
    });
```

Instead of just the name, POST an object containing more data to the Petstore API. Initially, create a random ID e.g. using code like the following, which generates a random number between 1 and 10,000,000:

```
const randomID = Math.floor(Math.random() * 10000000) + 1;
```

Send the random ID as part of the object in the POST request. How does this affect what pets you see?

Step Four

Hard code two tags and send those in the POST request so that they are displayed along with the animal name.

Step Five

Update the code in AddPage.jsx so that a user can enter the value of two tags. This will involve modifying the table at the bottom of the page and also ensuring that the state of each tag is stored and correctly initialized and updated. You can see how this is done for the name and follow the coding pattern. Review the section on state in the introductory tutorial if you need more help.