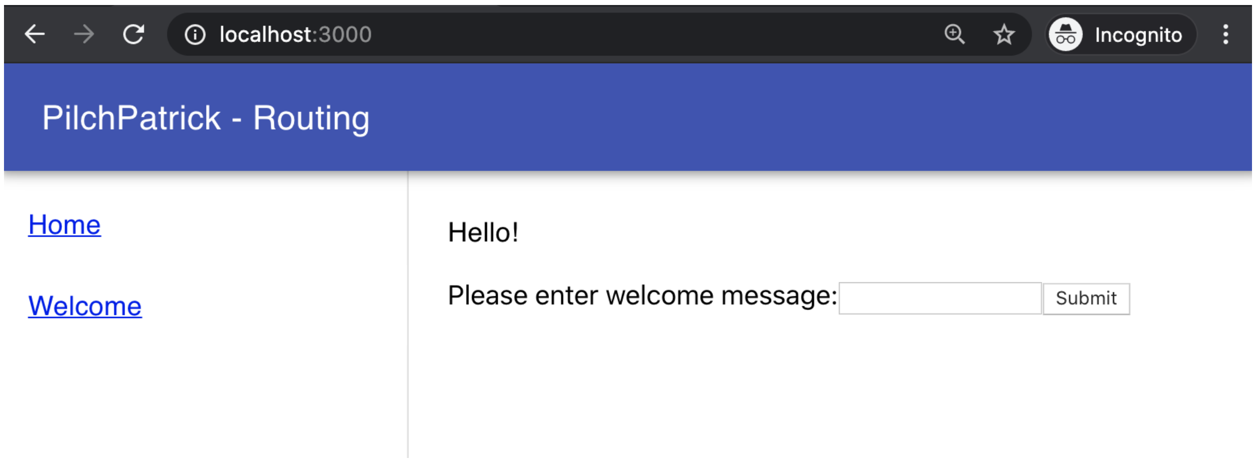
Routing Example

For this assignment, we will be building a basic skeleton of an app that has some navigation within it. The primary features are:

- We can move between the Home and Welcome pages using the navigation drawer, and the browser will recognize this navigation. This means that when the user is on the Home page and then navigates to the Welcome page, clicking "Back" in the browser should return them to the Home page.
- We will pass a message from the Home page to the Welcome page using a query argument. When this URL is opened in another tab or browser, we should be placed onto the Welcome page with the same message intact.

Submission instructions must be followed or points will be deducted.

Home Page



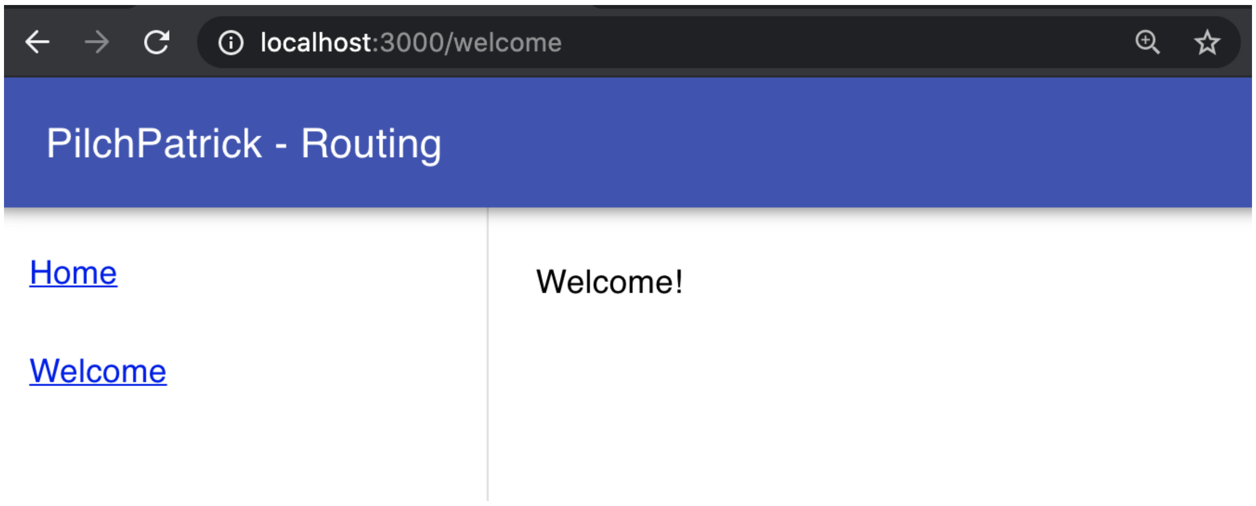
PilchPatrick - Routing

Hello!

Please enter welcome message:

The home page should at a minimum have a text input box where the user can enter a message. The user can enter any text they want. There should be a Submit button that, when clicked, will navigate and pass the message to the Welcome page using a query argument.

Welcome Page


localhost:3000/welcome

PilchPatrick - Routing

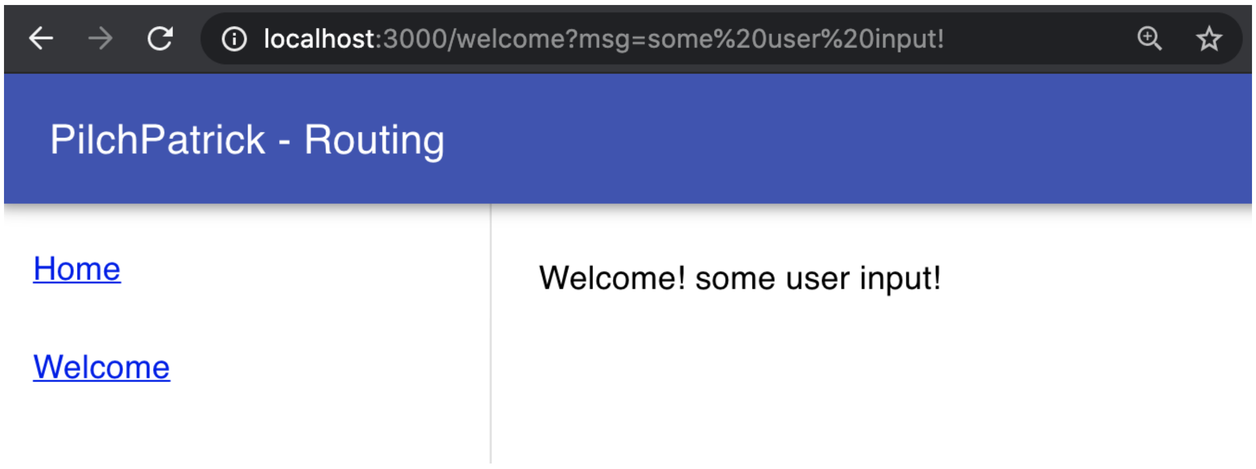
Home

Welcome

Welcome!

When the welcome page is navigated to using a URL path "/welcome" or from the navigation drawer hyperlink, it just needs to display the text "Welcome!"

But when the page is passed a message through a URL query argument, that message should be displayed after the welcome text.


localhost:3000/welcome?msg=some%20user%20input!

PilchPatrick - Routing

Home

Welcome

Welcome! some user input!

Requirements

Styling

The website should visually appear like in the screenshots.

Navigation

The navigation drawer should contain two items: Home and Welcome. Clicking Home should navigate to the path `/home`. Clicking Welcome should navigate to `/welcome`. Proper usage of React Router will mean they should be recognized by the browser. Therefore the following test case should succeed:

1. User opens the website for the first time.
2. User clicks "Welcome" in the navigation drawer: the browser is navigated to path "/welcome" and the welcome page/text appears.
3. User clicks the Back button in the browser: the browser navigates back to path "/" and the home text & input form should be displayed.
4. User clicks the Forward button in the browser: the browser navigates forward to "/welcome" and we see the welcome page again.

Home Page Form

The home page should have a text input box with a submit button. When the submit button is pressed, the user should be navigated to the Welcome page with the message passed through a query parameter. This *should be considered a navigation action* that the user can "Back" from. That means the following test case should pass:

1. The user is on the home page and enters a message into the form field and presses "Submit"
2. The Welcome page is displayed with the text "Welcome! {user input text}"
3. User presses "Back" in the browser.
4. The user is now on the Home page again.

Welcome Page Message

The welcome page should check for a query parameter to display a message. The naming of the query parameter is not important - I used "msg" in my solution. This means that the following test case should pass:

1. The user opens a new browser tab and manually enters the welcome page URL with the message query and presses enter. Example: "http://localhost:3000/welcome?msg=heelloooo" (Again, 'msg' is just the name that I used.)
2. The Welcome page should be the first thing visible and should display the text "Welcome! heelloooo"

How to Implement

First start by getting the layout/structure of the page. The structure of the webpage is basic usage of the AppBar and Drawer material components. This drawer is not collapsible, expandable, etc. - it's just a simple *permanent* drawer. The example documentation page for Drawer has a working example of exactly what is required with almost no modification.

Next write a basic navigation set up using React Router - Web. The "Quick Start" is required reading. See if you can navigate between Home and Welcome and use Back/Forward in the browser to intuitively navigate.

Create a basic React form on the home page to handle user input. When the user clicks Submit, you should have a handler function that updates the state to something that represents the user has indicated submission. In order to programmatically navigate (instead of the user actually clicking a hyper-link element), a <Redirect> element should be rendered. When this element is rendered by React, that will trigger a navigation. Required reading: Primary Components and the API documentation for Redirect.

Your welcome page component should read the query arguments to determine if there is a message to display. The "Query Arguments" example is required reading.