Microsoft Power BI

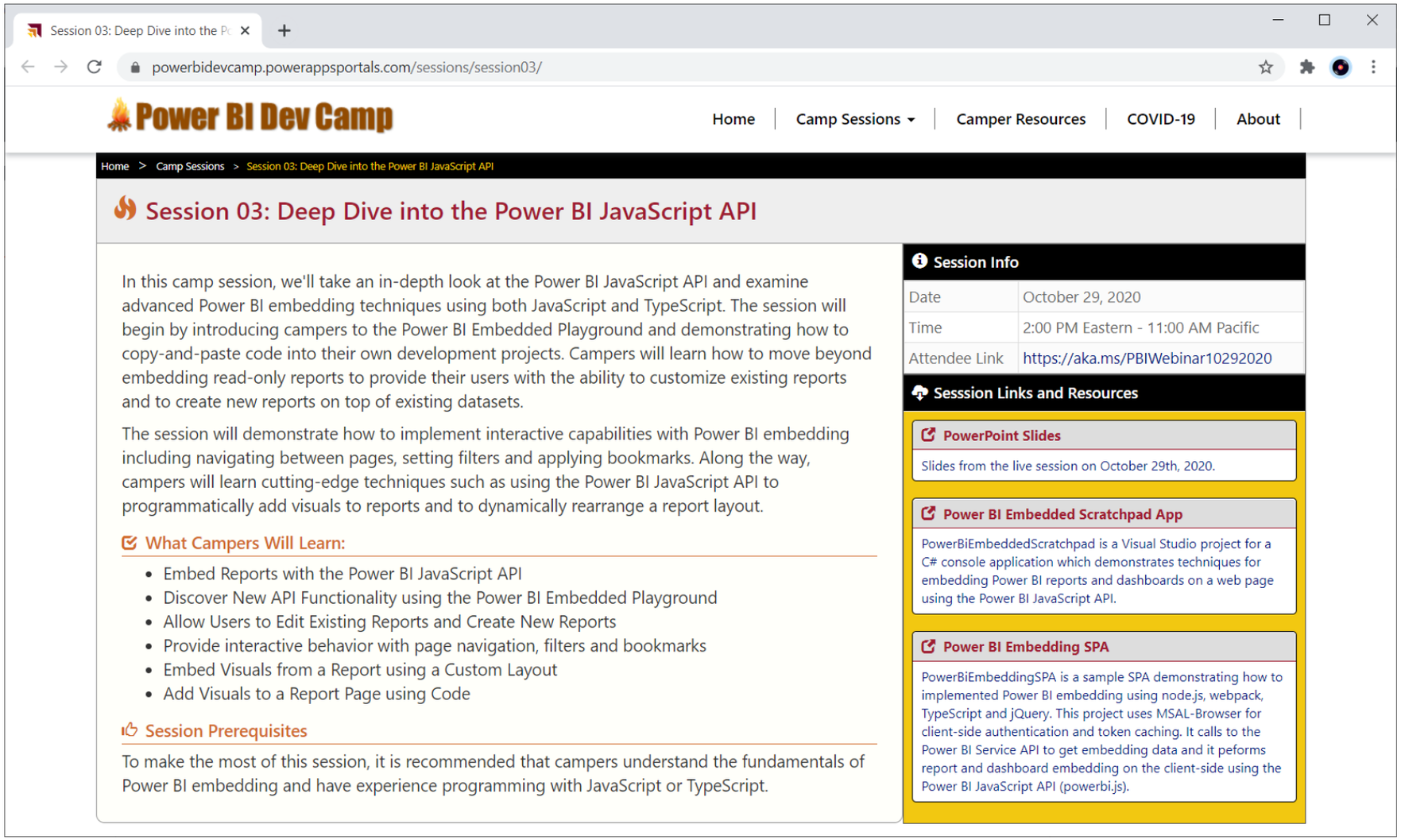
Power BI Dev Camp – Session 5

Embedding Power BI Reports using React.js

Ted Pattison
Principal Program Manager
Customer Advisory Team (CAT) at Microsoft

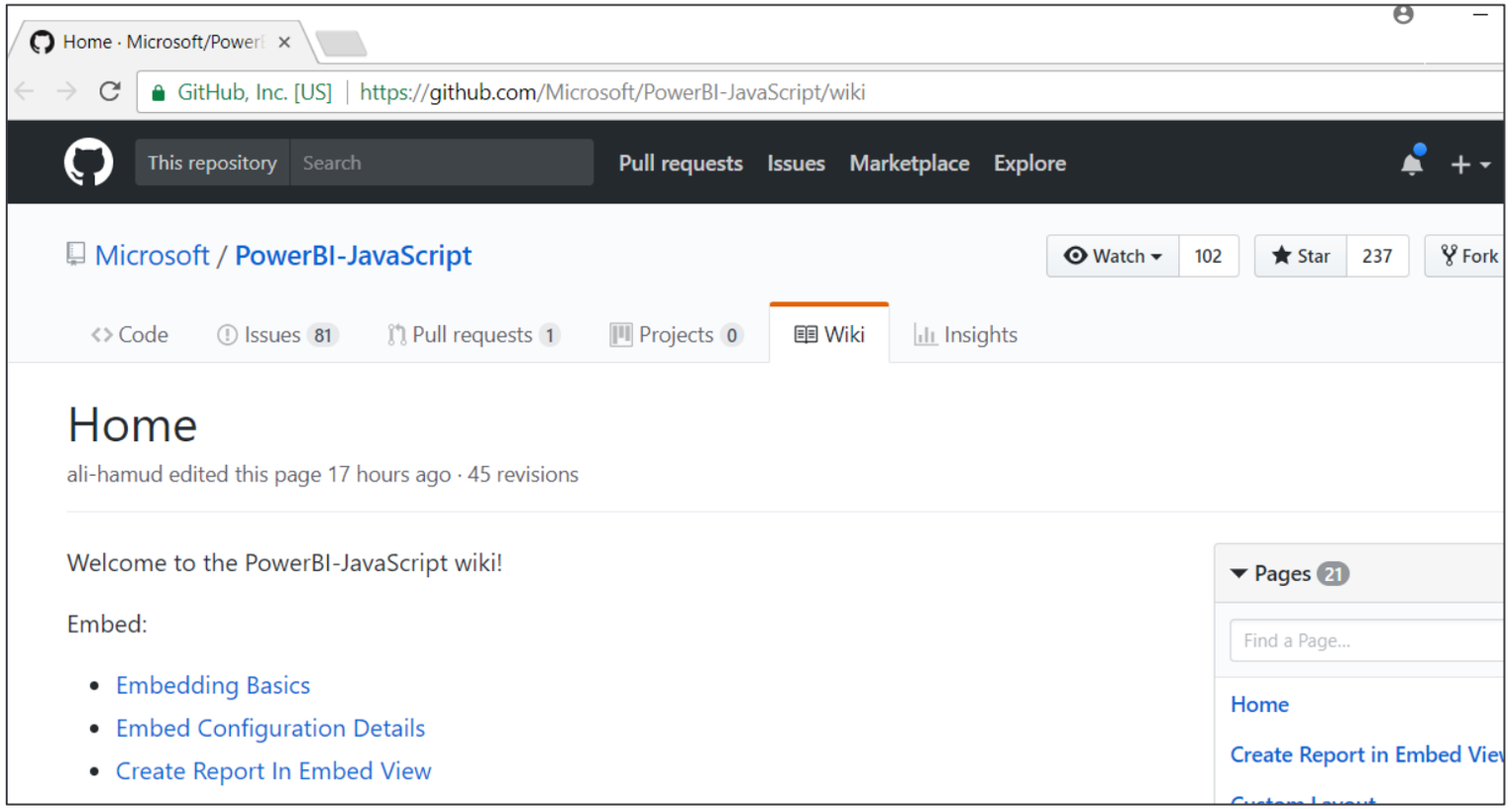
Welcome to Power BI Dev Camp

- Power BI Dev Camp Portal - https://powerbidevcamp.net



Power BI JavaScript API (powerbi.js)

- **Power BI JavaScript API used to embed resources in browser**
  - GitHub repo at https://github.com/Microsoft/PowerBI-JavaScript/wiki
  - GitHub repository contains code, docs, wiki and issues list



Agenda

- Getting Started with React.js
- Creating SPAs using React.js, TypeScript and Webpack
- Authenticating the User using Azure AD
- Extending a React Project with the React Router
- Calling the Power BI Service API using Fetch
- Embedding Power BI Reports
- The Power BI React Component

Introducing React

- **React is a library for building user interfaces**
  - Not as all-encompassing as a framework like Angular
  - Focused on building HTML-based user experiences
  - Based on reusable component-based architecture
  - Components *react* to state changes by updating UI
  - React uses shadow DOM for efficient event handling

React versus ReactDOM

- React and ReactDOM are separate libraries
  - **React (react.js)** is the primary library used to build out user experiences
  - **ReactDOM** (**react-dom.js**) is used to render **React** user experience in the browser
- React library exposes global React object
  - **React** object is the main entry point into React API
  - **React.DOM** wraps standard HTML elements
- ReactDOM library exposes global ReactDOM object
  - **ReactDOM** object used to render React components on web page

```
var reactComponenent = React.DOM.h1(null, "Hello, React!");

var target = document.getElementById("app");

ReactDOM.render(reactComponenent, target);
```

React Component Created Using ES5

- **React component can be created using EcmaScript 5**
  - React component definition created using **React.createClass**
  - React component must be defined with **render** method
  - React component can be instantiated with **React.createElement**

```
var myComponent = React.createClass({
  render: () => {
    return React.DOM.h1(null, "Hello React!")
  }
});

ReactDOM.render(
  React.createElement(myComponent),
  document.getElementById("app")
);
```

Initializing Element Properties

- **Elements created using properties object**
  - Object properties used to initialize element properties
  - Use **className** instead of **class** to assign CSS class
  - Use **htmlFor** instead of **for** to define HTML label

```
render: () => {

  var elementProperties = {
    id: "myElementId",
    className: "myCssClass"
  };

  return React.DOM.h1( elementProperties , "Hello React!");

}
```

Initializing Element Styles

- **Elements styles initialized using style object**
  - style must be defined using an object not a string
  - CSS properties referenced using camel casing

```
render: () => {

  var elementProperties = {
    id: "myElementId",
    style: {
      backgroundColor: "yellow",
      borderStyle: "Solid",
      borderColor: "green",
      padding: 8,
      color: "Blue",
      fontSize: 48
    }
  };

  return React.DOM.h1( elementProperties , "Hello React!");

}
```

React Provides Synthetic Events

- **Replaces standard DOM-based event handling**
  - React creates virtual DOM for elements in component
  - React interacts with real DOM when required
  - Provides faster event registration and processing
  - No need to write browser-specific code

```
private incrementCounter() {
  var previousCount: number = this.state.count;
  this.setState({ count: previousCount + 1 });
}

public render(): React.ReactElement<IBeanCounterProps> {
  return (
    <div className={styles.beanCounter}>
      <h3>Mr Bean Counter</h3>
      <div className={styles.toolbar}>
        <button onClick={(event) => { this.incrementCounter(); }}  >Add another Bean</button>
      </div>
      <div className={styles.beanCounterDisplay} >
        Bean Count: {this.state.count}
      </div>
    </div>
  );
}
```

Understanding JSX (and TSX)

- **JSX provides better syntax for HTML composition**
  - JSX allows extends JavaScript with XML-like syntax
  - JSX syntax must be transpiled into JavaScript code

```
var myHtml = <div id="myAppContainer" style={{ backgroundColor:"yellow", padding:8 }}>
                <h2>Hello JSX</h2>
                <p>I'm composing HTML elements using JSX syntax.</p>
              </div>;


ReactDOM.render( myHtml , document.getElementById("app") );
```

- **JSX/TSX is separate from React library**
  - JSX/TSX commonly used in React development
  - Babel compiler used to transpile JSX to JavaScript
  - TypeScript compiler used to transpile TSX to JavaScript

Agenda

- ✓ Getting Started with React.js
- ➢ Creating SPAs using React.js, TypeScript and Webpack
- Authenticating the User using Azure AD
- Extending a React Project with the React Router
- Calling the Power BI Service API using Fetch
- Embedding Power BI Reports
- The Power BI React Component

Defining React Components using TypeScript

- **Component is class extending `React.Component`**
  - Component usually defined in its own **tsx** file
  - Component class must define **render** method

my-component.tsx

```
import * as React from 'react';

export class MyComponent extends React.Component<any, any> {
    render() {
        return <h2>Hello from my component</h2>;
    }
}
```

  - Component can be instantiated with JSX/TSX syntax

app.tsx

```
import * as ReactDOM from 'react-dom';

import { MyComponent } from "./components/my-component"

window.onload = () => {
  // Create and render component
  ReactDOM.render( <MyComponent/>, document.getElementById("app") );
}
```

Component Properties and State

- **Component can contain properties and state**
  - Properties are initialized by external components
  - Properties are read-only to hosting component
  - State is set internally by hosting component
  - Changing state triggers UI refresh by calling render
  - UI experience created by ***reacting*** to changes in state

React Component Properties

- Defining component with a property

```
component1.tsx
import * as React from 'react';

export interface MyCustomProps {
  Name: string;
}

export class Component1 extends React.Component<MyCustomProps, {}>
  render() {
    return <div>Hello, my name is {this.props.Name}</div>;
  }
}
```

- Instantiating component with a property

```
ReactDOM.render(
    <Component1 Name="Fred" />,
    document.getElementById("app")
);
```

Stateful Component

TS IBeanCounterProps.ts

```ts
export interface IBeanCounterProps {
  StartingValue: number;
}
```

TS IBeanCounterState.ts

```ts
export interface IBeanCounterState {
    count: number;
}
```

BeanCounter.tsx

```ts
import * as React from 'react';
import styles from './BeanCounter.module.scss';
import { IBeanCounterProps } from './IBeanCounterProps';
import { IBeanCounterState } from './IBeanCounterState';

export default class BeanCounter extends React.Component<IBeanCounterProps, IBeanCounterState> {

  constructor(props: any) {
    super(props);
    this.state = { count: this.props.StartingValue };
  }

  private incrementCounter() {
    var previousCount: number = this.state.count;
    this.setState({ count: previousCount + 1 });
  }
```

Stateful Component Rendering

BeanCounter.tsx

```
import * as React from 'react';
import styles from './BeanCounter.module.scss';
import { IBeanCounterProps } from './IBeanCounterProps';
import { IBeanCounterState } from './IBeanCounterState';

export default class BeanCounter extends React.Component<IBeanCounterProps, IBeanCounterState> {

  constructor(props: any) {
    super(props);
    this.state = { count: this.props.StartingValue };
  }

  private incrementCounter() {
    var previousCount: number = this.state.count;
    this.setState({ count: previousCount + 1 });
  }

  public render(): React.ReactElement<IBeanCounterProps> {
    return (
      <div className={styles.beanCounter}>
        <h3>Mr Bean Counter</h3>
        <div className={styles.toolbar}>
          <button onClick={(event) => { this.incrementCounter(); }}  >Add another Bean</button>
        </div>
        <div className={styles.beanCounterDisplay} >
          Bean Count: {this.state.count}
        </div>
      </div>
    );
  }
}
```

Starter Project - package.json

File Edit Selection View Go Run Terminal Help — package.json - react-bean-counter - Visual Studio Code

EXPLORER

- OPEN EDITORS
- REACT-BEAN-COUNTER
  - node_modules
  - src
    - components
      - App.scss
      - App.tsx
      - BeanCounter.scss
      - BeanCounter.tsx
    - images
    - favicon.ico
    - index.html
    - index.tsx
  - npm-shrinkwrap.json
  - package.json
  - tsconfig.json
  - webpack.config.js

package.json > ...

```
1  {
2    "name": "demo1-my-react-app",
3    "version": "1.0.0",
4    "scripts": {
5      "build": "webpack",
6      "start": "webpack-dev-server --open --history-api-fallback"
7    },
8    "devDependencies": {
9      "@types/es6-promise": "^3.3.0",
10     "@types/node": "^10.9.4",
11     "@types/react": "^16.4.14",
12     "@types/react-dom": "^16.0.7",
13     "awesome-typescript-loader": "^5.2.0",
14     "clean-webpack-plugin": "^0.1.19",
15     "copy-webpack-plugin": "^4.5.2",
16     "css-loader": "^0.28.11",
17     "expose-loader": "^0.7.5",
18     "file-loader": "^1.1.11",
19     "html-webpack-plugin": "^3.2.0",
20     "node-sass": "^4.14.1",
21     "office-ui-fabric-react": "^6.69.0",
22     "react": "^16.5.1",
23     "react-dom": "^16.5.1",
24     "sass-loader": "^7.1.0",
25     "style-loader": "^0.21.0",
26     "typescript": "^3.0.1",
27     "url-loader": "^1.0.1",
28     "webpack": "^4.30.0",
29     "webpack-cli": "^3.3.1",
30     "webpack-dev-server": "^3.11.0"
31   }
32 }
```

Starter Project - webpack.config.js

webpack.config.js

```
1   const path = require('path');
2   const HtmlWebpackPlugin = require('html-webpack-plugin');
3   const CopyWebpackPlugin = require('copy-webpack-plugin');
4   const CleanWebpackPlugin = require('clean-webpack-plugin')
5
6   module.exports = {
7       entry: './src/index.tsx',
8       output: {
9           filename: 'scripts/bundle.js',
10          path: path.resolve(__dirname, 'dist'),
11      },
12      resolve: {
13          extensions: ['.js', '.json', '.ts', '.tsx'],
14      },
15      plugins: [
16          new CleanWebpackPlugin(['dist']),
17          new HtmlWebpackPlugin({ template: path.join(__dirname, 'src', 'index.html') }),
18          new CopyWebpackPlugin([{ from: './src/favicon.ico', to: 'favicon.ico' }])
19      ],
20      module: {
21          rules: [
22              { test: /\.(ts|tsx)$/, loader: 'awesome-typescript-loader' },
23              { test: /\.css$/, use: ['style-loader', 'css-loader'] },
24              { test: /\.scss$/, use: ["style-loader", "css-loader", "sass-loader"] },
25              { test: /\.(png|jpg|gif)$/, use: [{ loader: 'url-loader', options: { limit: 8192 } }] }
26          ],
27      },
28      mode: "development",
29      devtool: 'source-map',
30      devtool: 'cheap-eval-source-map'
31  };
```

The Top-level App Component

App.tsx - react-starter - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS

App.tsx src\components

REACT-STARTER

node_modules

src

components

App.css

App.tsx

images

favicon.ico

index.html

index.tsx

npm-shrinkwrap.json

package.json

tsconfig.json

webpack.config.js

App.tsx

```
import * as React from 'react';

import 'bootstrap/dist/css/bootstrap.min.css';
import 'bootstrap';

import './App.css';

export default class App extends React.Component<any, any> {

  render() {

    return (
      <div id="page-container" className="container">
        <div className="row navbar navbar-expand-sm navbar-dark bg-dark" role="navigation" >
          <h1 style={{ 'color': 'white' }} >React Starter App</h1>
        </div>
        <div className="jumbotron">
          <div>This is a sample starter app for with with React and TypeScript.</div>
        </div>
      </div>
    );

  }

}
```

Bootstrapping the App Component

index.tsx - react-starter - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS

index.tsx src

REACT-STARTER

node_modules

src

components

App.css

App.tsx

images

favicon.ico

index.html

index.tsx

npm-shrinkwrap.json

package.json

tsconfig.json

webpack.config.js

index.tsx

```
1  import * as React from 'react';
2  import { render } from 'react-dom';
3  import App from './components/App';
4
5  var topLevelAppComponent = <App />;
6  var target = document.getElementById('react-target');
7
8  render(topLevelAppComponent, target);
```

React Component Hierarchies

• App.tsx - react-lab-exercise2 - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS 1 UNSAVED

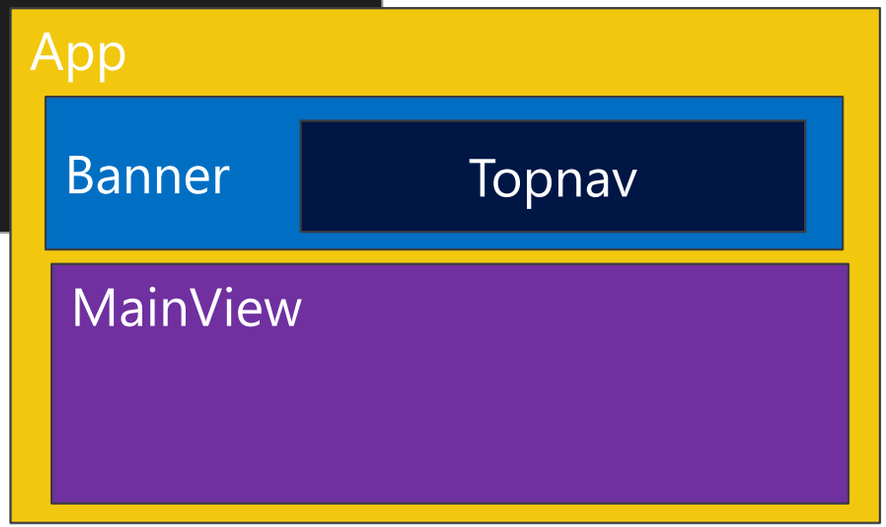
App.tsx src\components

REACT-LAB-EXERCISE2

- node_modules
- src
  - components
    - App.css
    - App.tsx
    - Banner.css
    - Banner.tsx
    - MainView.css
    - MainView.tsx
    - TopNav.css
    - TopNav.tsx
  - images
  - favicon.ico
  - index.html

App.tsx

```
import * as React from 'react';

import Banner from "./Banner";
import TopNav from "./Topnav";
import MainView from "./MainView";

export default class App extends React.Component<any, any> {

  render() {
    return (
      <div id="page-container" className="container">
        <Banner appTitle="React Lab App" >
          <TopNav />
        </Banner>
        <MainView />
      </div>
    );
  }

}
```

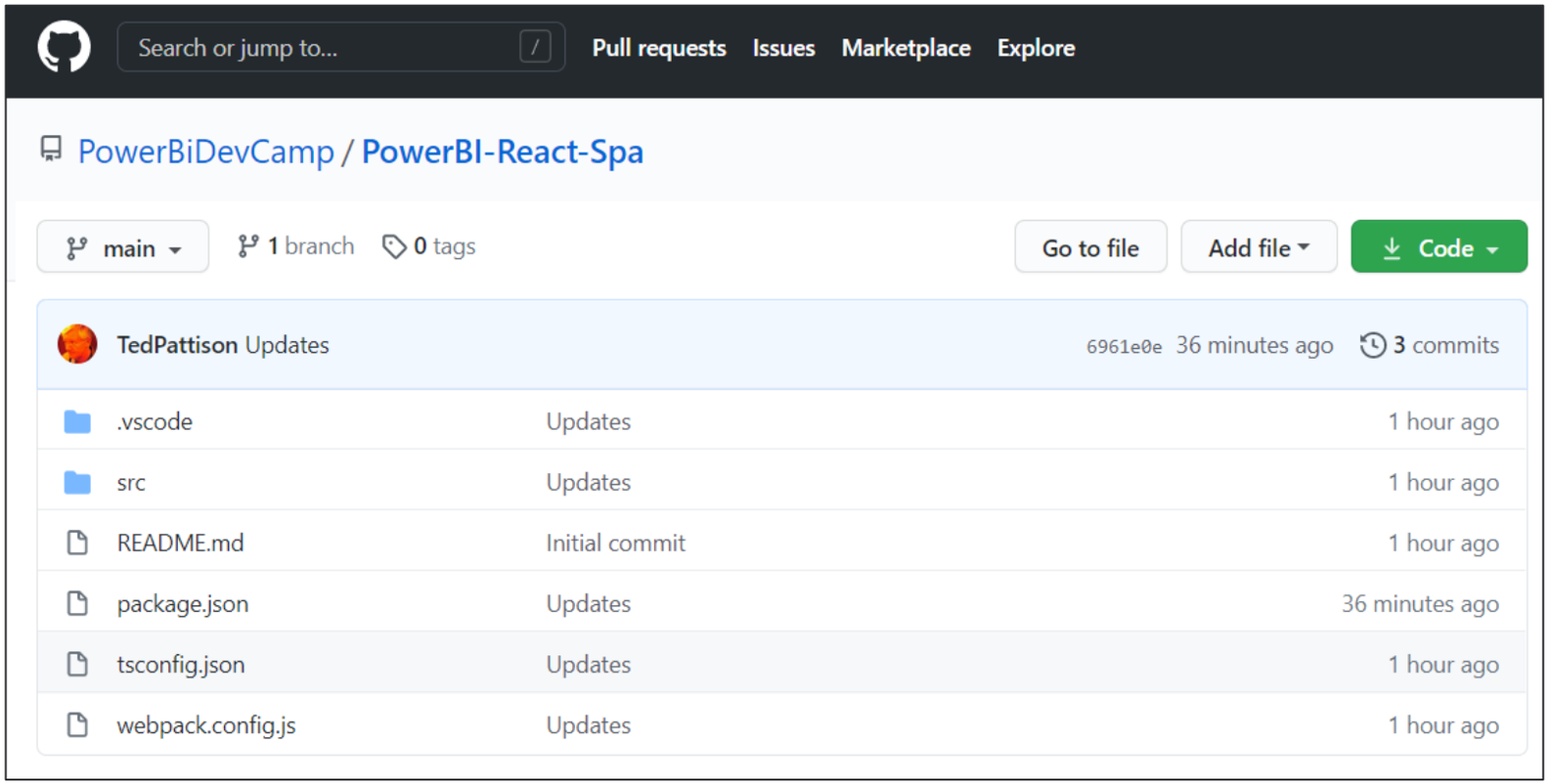


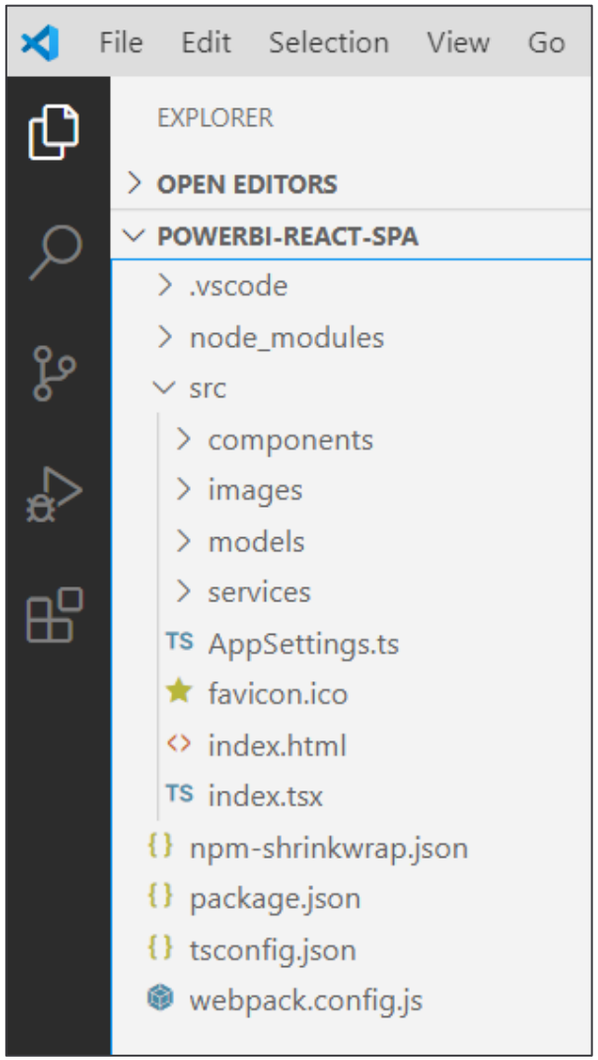
Agenda

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ➢ Authenticating the User using Azure AD
- Extending a React Project with the React Router
- Calling the Power BI Service API using Fetch
- Embedding Power BI Reports
- The Power BI React Component

Demo Application: PowerBi-React-Spa

https://github.com/PowerBiDevCamp/PowerBI-React-Spa





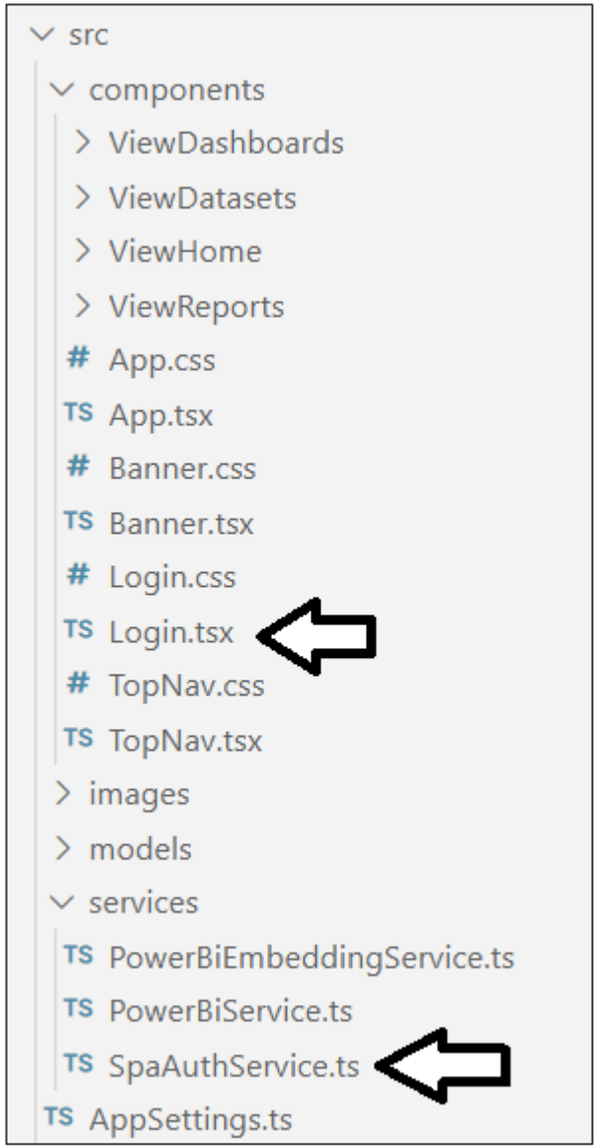
AppSettings.ts

TS AppSettings.ts ×

src > TS AppSettings.ts > ...

```
1  export default class AppSettings {
2
3    static aadTenant: string = "powe[redacted]net";
4
5    static clientId: string = "34de5c7[redacted]ed33";
6
7    static appWorkspaceId: string = "912f2[redacted]d864";
8
9  }
```

User Login and Authentication

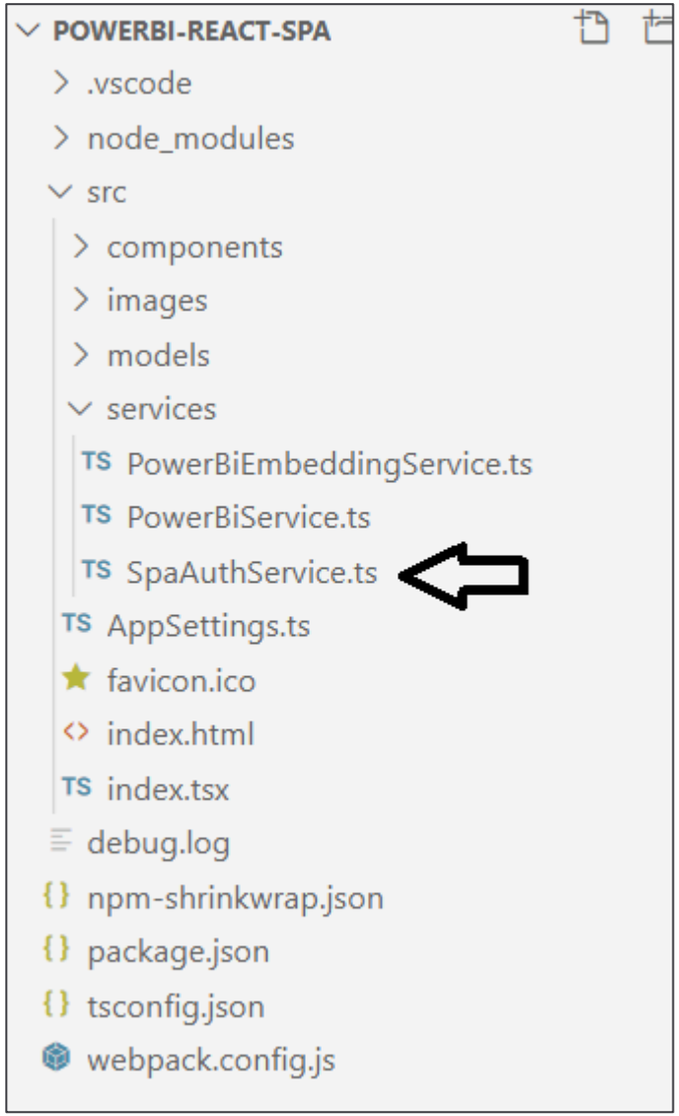


Login.tsx

src > components > Login.tsx > ...

```
1   import * as React from 'react';
2   import IUser from "./../models/IUser"
3   import "./Login.css";
4
5   interface LoginProperties {
6     user: IUser;
7   }
8
9   export default class Login extends React.Component<LoginProperties, any> {
10
11    render() {
12      return (
13        <div id="login" className='collapse navbar-collapse justify-content-end' >
14          {this.props.user.IsAuthenticated ? <div>Hello {this.props.user.DisplayName}</div> : null}
15          <ul>
16            {this.props.user.IsAuthenticated ?
17              <li><a href="#" onClick={this.props.user.logout} >Logout</a></li> :
18              <li><a href="#" onClick={this.props.user.login} >Sign In</a></li>
19            }
20          </ul>
21        </div>
22      );
23    }
24
25  }
```

SpaAuthService.ts



SpaAuthService.ts

src > services > SpaAuthService.ts > SpaAuthService > msalConfig

```
1   import * as Msal from 'msal';
2   import AppSettings from "./../AppSettings";
3
4   export default class SpaAuthService {
5
6       private static clientId: string = AppSettings.clientId;
7       private static authority: string = "https://login.microsoftonline.com/" + AppSettings.aadTenant;
8
9       private static requestScopesPowerBi = {
10          scopes: [
11              "https://analysis.windows.net/powerbi/api/Dashboard.Read.All",
12              "https://analysis.windows.net/powerbi/api/Dataset.Read.All",
13              "https://analysis.windows.net/powerbi/api/Report.ReadWrite.All",
14              "https://analysis.windows.net/powerbi/api/Group.Read.All",
15              "https://analysis.windows.net/powerbi/api/Workspace.ReadWrite.All",
16              "https://analysis.windows.net/powerbi/api/Content.Create"
17          ]
18      };
19
20      public static userIsAuthenticated: boolean = false;
21      public static userDisplayName: string = "";
22      public static userName: string = "";
23      public static accessToken: string;
24      public static uiUpdateCallback: any;
25
26      private static msalConfig: Msal.Configuration = {
27          auth: {
28              clientId: SpaAuthService.clientId,
29              authority: SpaAuthService.authority,
```

Agenda

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ✓ Authenticating the User using Azure AD
- ➢ Extending a React Project with the React Router
- Calling the Power BI Service API using Fetch
- Embedding Power BI Reports
- The Power BI React Component

React Router

- Used to create route map in single page application (SPA)
  - Installed as a pair of npm packages

```
npm install react-router @types/react-router --save-dev
npm install react-router-dom @types/react-router-dom --save-dev
```

- Router must be added in as top-level component above App

index.tsx

```
import * as React from 'react';
import { render } from 'react-dom';
import App from './components/App';
import { HashRouter } from 'react-router-dom';

var topLevelAppComponent =
  <HashRouter>
    <App />
  </HashRouter>;

var target = document.getElementById('react-target');

render(topLevelAppComponent, target);
```

Using React Router

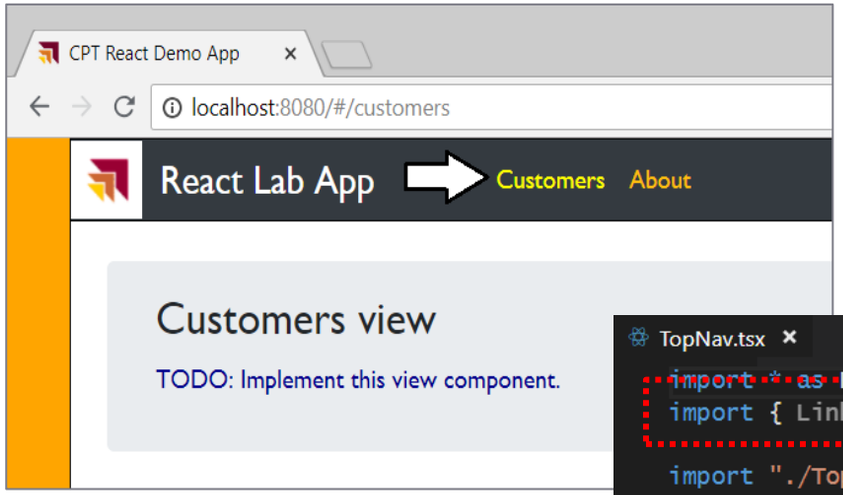
- Import Route and Switch components

```
import * as React from 'react';
import { Route, Switch } from 'react-router-dom';
```

- Create route map in HTML output

```
export default class App extends React.Component<any, any> {

  render() {

    return (
      <div id="page-container" className="container">
        <Banner appTitle="React Lab App" >
          <TopNav />
        </Banner>
        <Switch>
          <Route path="/" exact component={ViewHome} />
          <Route path="/customers" component={ViewCustomers} />
          <Route path="/about" component={ViewAbout} />
        </Switch>
      </div>
    );
```

Creating Route Links



TopNav.tsx

```
import * as React from 'react';
import { Link, NavLink } from 'react-router-dom';

import "./TopNav.css";

export default class TopNav extends React.Component<any, any> {

  render() {
    return (
      <div id="top-nav" className="navbar-collapse collapse" >
        <nav>
          <ul className="nav navbar-nav" >
            <li className="nav-item" >
              <NavLink exact to="/" className="navbar-link" activeClassName="active-nav-link" >
                Home
              </NavLink>
            </li>
            <li className="nav-item" >
              <NavLink to="/customers" className="navbar-link" activeClassName="active-nav-link" >
                Customers
              </NavLink>
            </li>
```

Agenda

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ✓ Authenticating the User using Azure AD
- ✓ Extending a React Project with the React Router
- ➢ Calling the Power BI Service API using Fetch
- Embedding Power BI Reports
- The Power BI React Component

Component Lifecycle Methods

- **componentDidMount**
  - executed after node is added to the DOM
- **componentDidUpdate**
  - executed after component is rendered
  - executed before node is added to the DOM
- **shouldComponentUpdate(newProps, newState)**
  - executed before component is updated

Calling a Web Service using the Fetch API

```
export default class PowerBiService {

  static appWorkspaceId = AppSettings.appWorkspaceId;
  static apiRoot: string = "https://api.powerbi.com/v1.0/myorg/";
  static appWorkspaceApiRoot = PowerBiService.apiRoot + "groups/" + PowerBiService.appWorkspaceId + "/";

  static GetReports = (): Promise<PowerBiReport[]> => {
    var restUrl = PowerBiService.appWorkspaceApiRoot + "Reports/";
    return fetch(restUrl, {
      headers: {
        "Accept": "application/json;odata.metadata=minimal;",
        "Authorization": "Bearer " + SpaAuthService.accessToken
      }
    }).then(response => response.json())
      .then(response => { return response.value; });
  }
```

Agenda

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ✓ Authenticating the User using Azure AD
- ✓ Extending a React Project with the React Router
- ✓ Calling the Power BI Service API using Fetch
- ➢ Embedding Power BI Reports
- The Power BI React Component

Embedding Power BI Reports

EXPLORER

- OPEN EDITORS
- POWERBI-REACT-SPA
  - .vscode
  - node_modules
  - src
    - components
      - ViewDashboards
      - ViewDatasets
      - ViewHome
      - ViewReports
        - EmbeddedReport.tsx
        - EmbeddedReportToolbar.css
        - EmbeddedReportToolbar.tsx
        - ViewReports.tsx

PowerBiService.ts | *EmbeddedReport.tsx*

src > components > ViewReports > EmbeddedReport.tsx > ...

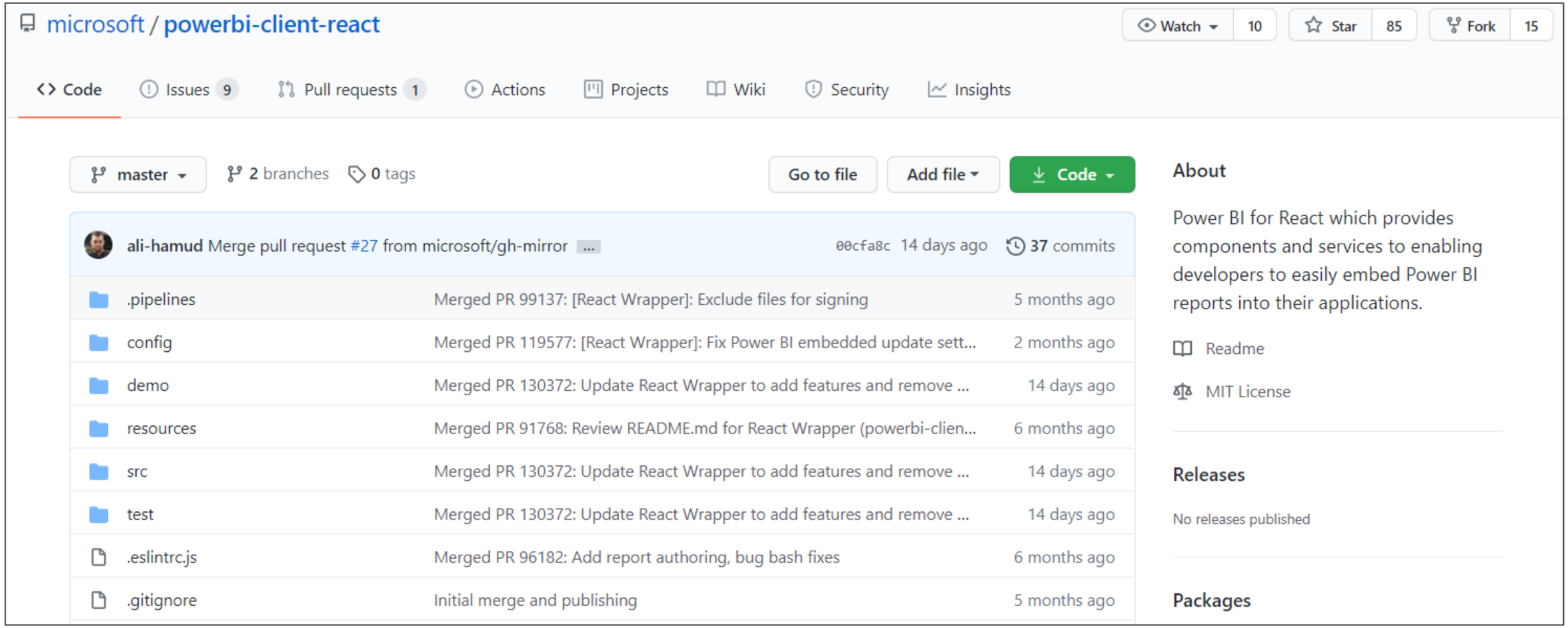
```
import * as React from 'react';
import { withRouter, RouteComponentProps, Link } from 'react-router-dom'

import { PowerBiReport } from "./../../models/PowerBiModels";
import PowerBiEmbeddingService from "./../../services/PowerBiEmbeddingService";

import EmbeddedReportToolbar from './EmbeddedReportToolbar';

import * as powerbi from "powerbi-client";
import * as pbimodels from "powerbi-models";

interface EmbeddedReportProperties {
  reports: PowerBiReport[]
}
```

Agenda

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ✓ Authenticating the User using Azure AD
- ✓ Extending a React Project with the React Router
- ✓ Calling the Power BI Service API using Fetch
- ✓ Embedding Power BI Reports
- ➢ The Power BI React Component

Power BI React Component

https://github.com/microsoft/powerbi-client-react



Summary

- ✓ Getting Started with React.js
- ✓ Creating SPAs using React.js, TypeScript and Webpack
- ✓ Authenticating the User using Azure AD
- ✓ Extending a React Project with the React Router
- ✓ Calling the Power BI Service API using Fetch
- ✓ Embedding Power BI Reports
- ✓ The Power BI React Component

Questions