ReactJS Training Course (Session 3)

TUAN MAI CHUNG / NashTech

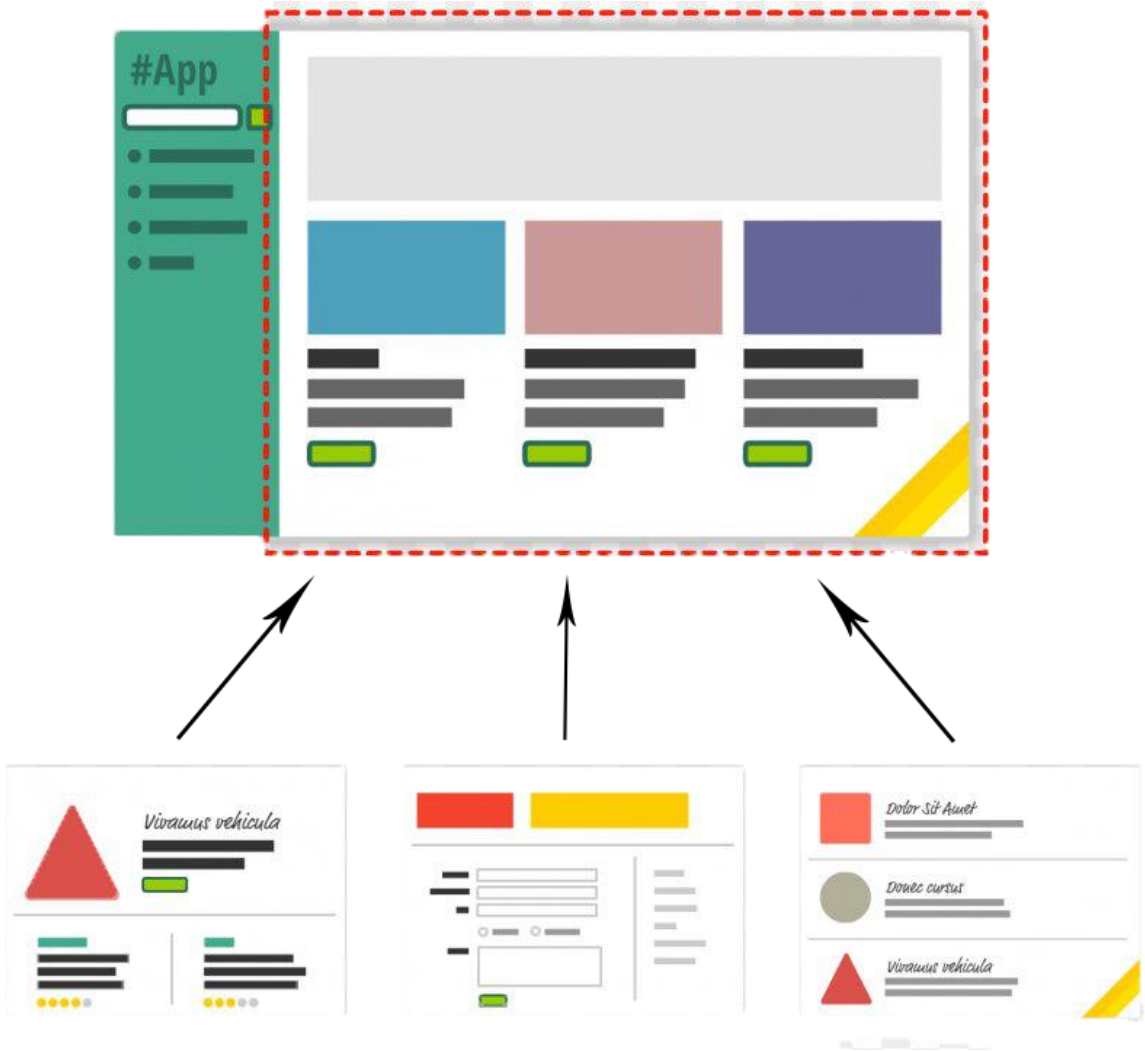
Sep/2020

Nash Tech.

Agenda

1. UNDERSTAND SPA / MPA
2. REACT ROUTER DOM
3. AXIOS
4. EXERCISE

Single Page Application (SPA)



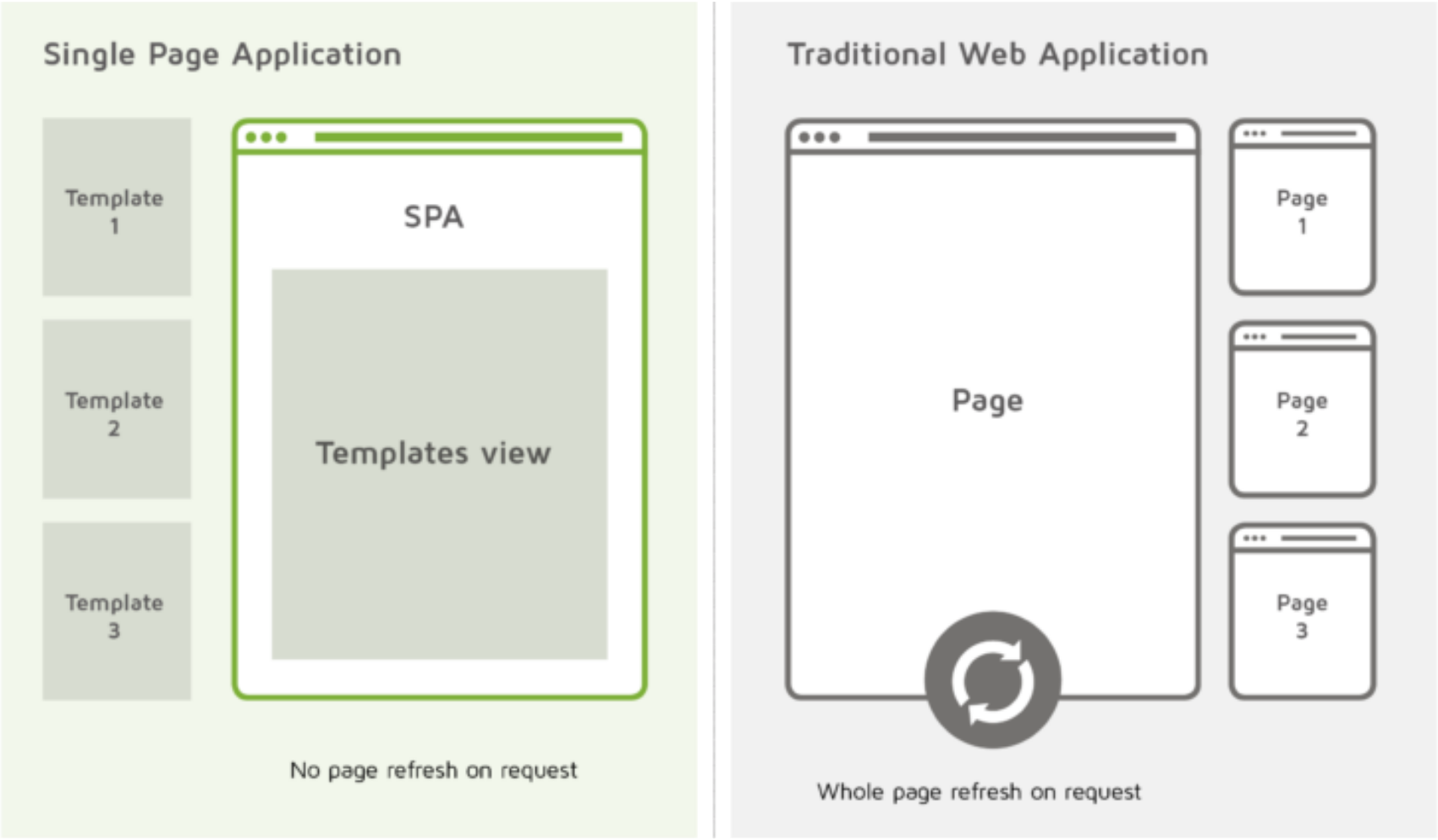
How traditional web work?

- **Multi-page applications (MPA)** are the traditional web **applications** that reload the entire **page** and display the new one when a user interacts with the web **app**
- Each time when a data is exchanged back and forth, a new **page** is requested from the server to display in the web browser.

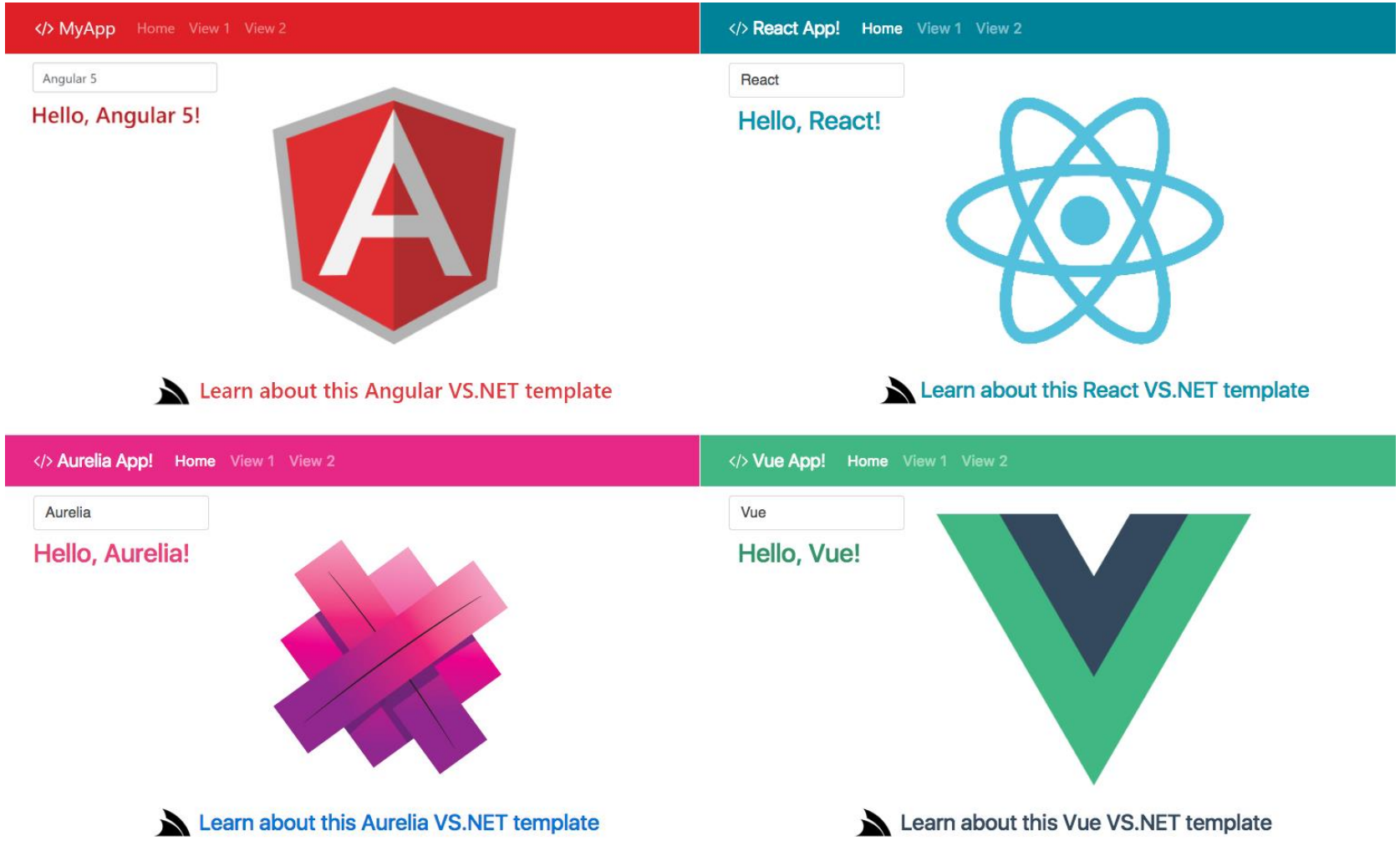
What is SPA?

- **Single Page Application (SPA)** is a web application taking a single HTML page.
- It allows interacting with the page without refreshing it.

SPA vs MPA


Single Page Application
Template 1
Template 2
Template 3
SPA
Templates view
No page refresh on request
Traditional Web Application
Page
Page 1
Page 2
Page 3
Whole page refresh on request

Example of SPA



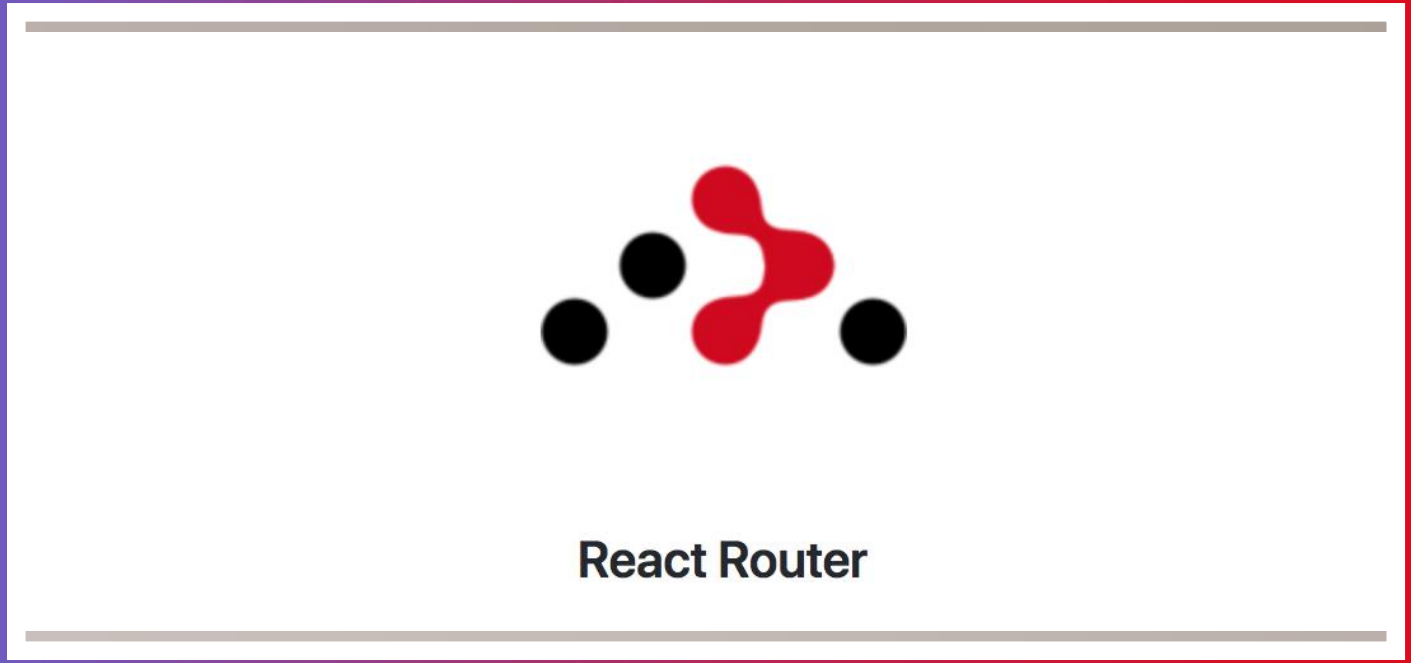
Pros vs Cons of SPA

Pros

- Fast and responsive design
- Adaptable layout
- Increased application performance
- Better UX

Cons

- Poor SEO optimization
- Difficulties in the development process
- Browser history



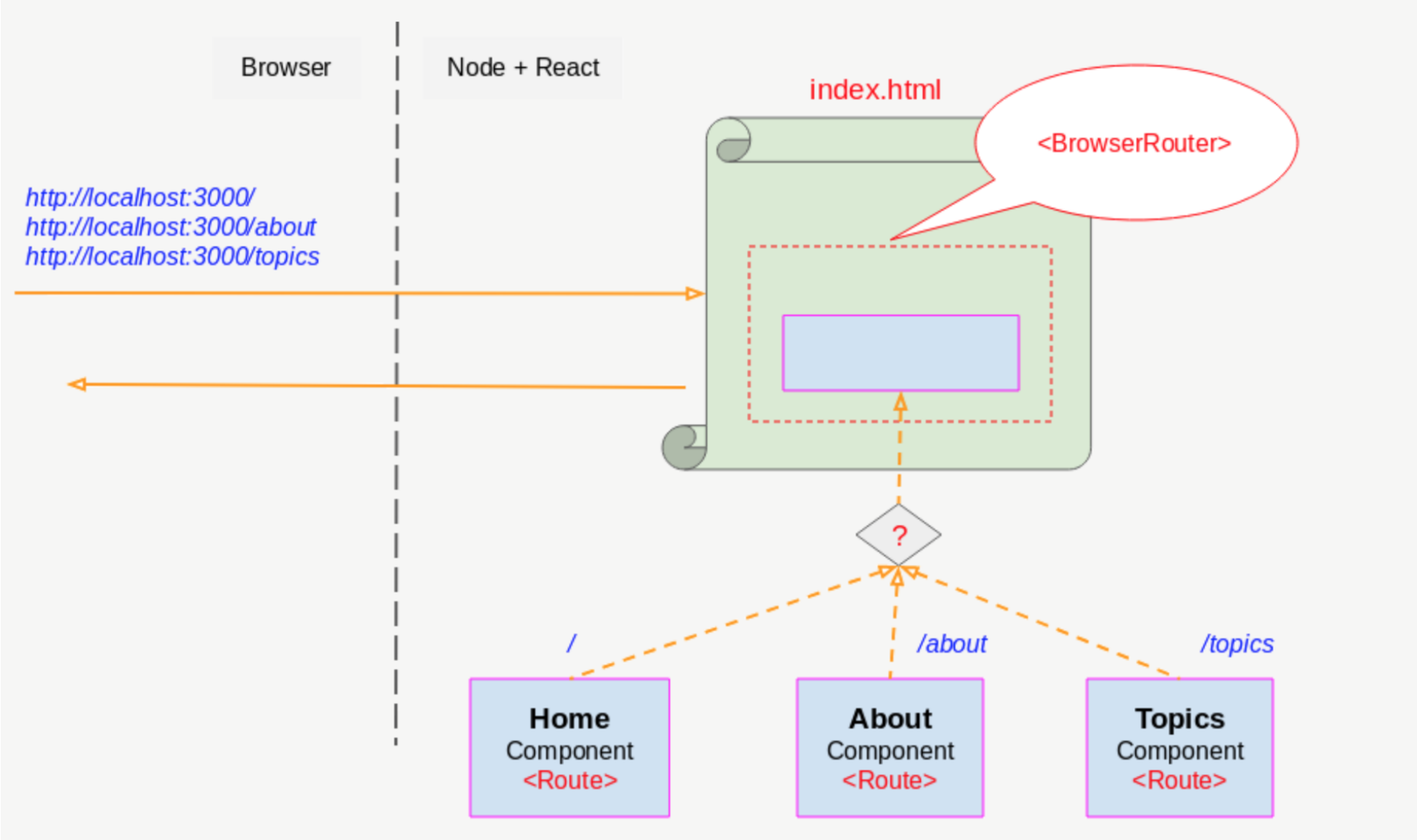
React Router

React Router - Installation

- With NPM:
  - **npm install react-router-dom**
- With Yarn
  - **yarn add react-router-dom**

React Router - Components

- Must have components:
  - BrowserRouter
    - Wrap all your <Route> components
  - Route
    - Render some UI when its path matches the current URL.
- Other components:
  - Link
    - Provides declarative, accessible navigation around your application.
  - Switch
    - Renders the first child <Route> that matches the location.


Browser
Node + React
index.html
<BrowserRouter>
http://localhost:3000/
http://localhost:3000/about
http://localhost:3000/topics
?
/
/about
/topics
Home
Component
<Route>
About
Component
<Route>
Topics
Component
<Route>

BrowserRouter

```
import React from "react";
import "./App.css";

import { BrowserRouter } from "react-router-dom";
function App() {
  return (
    <BrowserRouter>
      {/* ... */}
    </BrowserRouter>
  );
}

export default App;
```

Route

```
function App() {
  return (
    <BrowserRouter>
      <Route exact path="/">
        <Home />
      </Route>
      <Route path="/news">
        <NewsFeed />
      </Route>
    </BrowserRouter>
  );
}
```

Route – props

```
import React from "react";
import { BrowserRouter, Route } from "react-router-dom";
import "./App.css";

function User(props) {
  return <h1>Hello {props.match.params.username}!</h1>;
}

function App() {
  return (
    <BrowserRouter>
      <Route path="/user/:username" component={User} />
    </BrowserRouter>
  );
}

export default App;
```

Route – inline render

```
import React from "react";
import { BrowserRouter, Route } from "react-router-dom";
import "./App.css";

function App() {
  return (
    <BrowserRouter>
      <Route
        path="/user/:username"
        render={() => <h1>Hello {props.match.params.username}!</h1>}
      />
    </BrowserRouter>
  );
}

export default App;
```

Link

```
import React from "react";
import { BrowserRouter, Link } from "react-router-dom";
import "./App.css";

function App() {
  return (
    <BrowserRouter>
      <aside>
        <Link to={`/dashboard`}>Dashboard</Link>
        <Link to={`/about`}>About Nashtech</Link>
      </aside>
      ....
    </BrowserRouter>
  );
}

export default App;
```

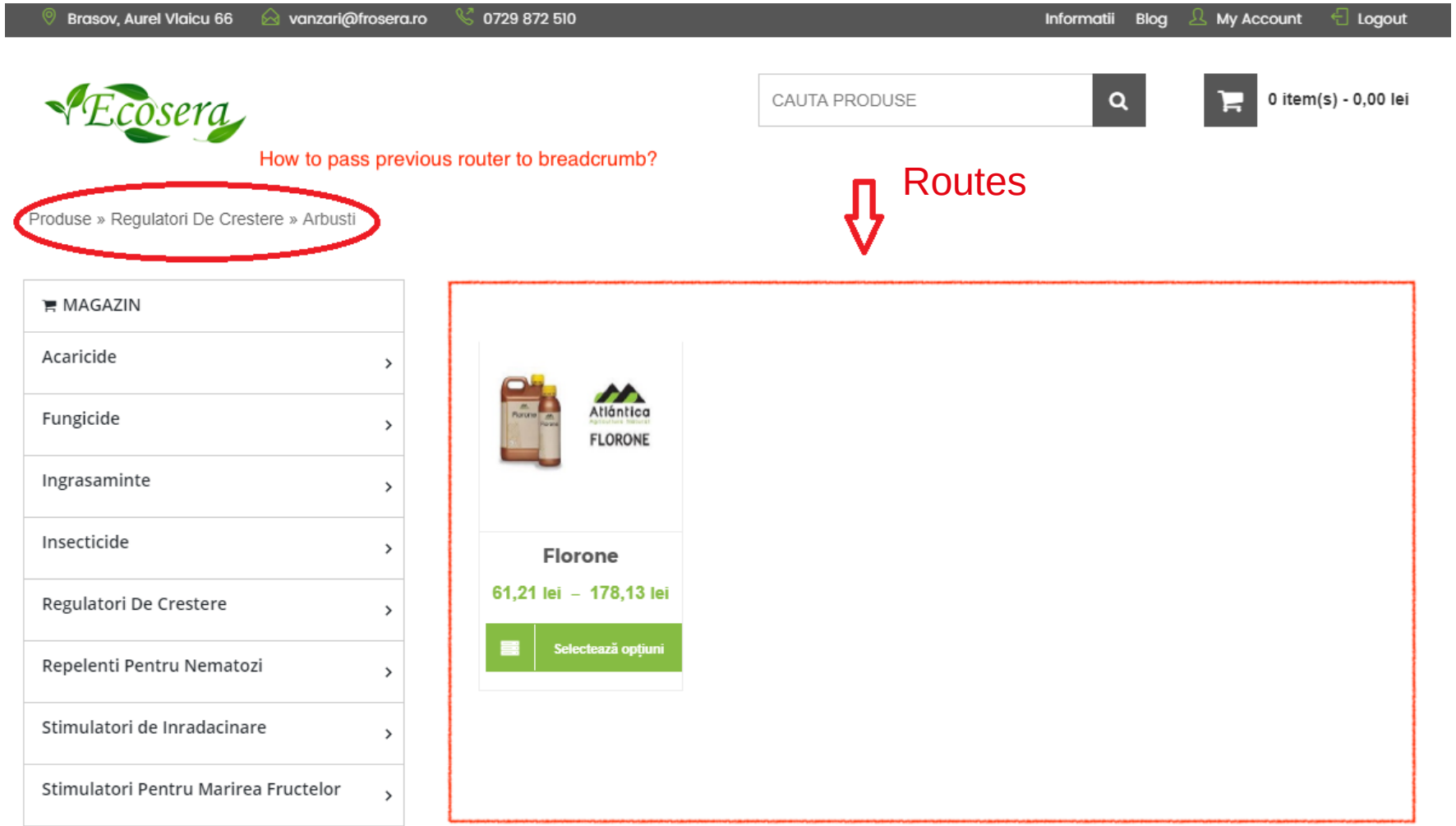
Example

```
function App() {
  return (
    <Router>
    <div>
          <h2>React Router Step By Step Tutorial</h2>
          <nav>
          <ul>
            <li><Link to={'/'} > Home </Link></li>
            <li><Link to={'/contact'} >Contact</Link></li>
            <li><Link to={'/about'} >About</Link></li>
            <li><Link to={'/services'} >Services</Link></li>
          </ul>
          </nav>
          <Switch>
                <Route path = "/" exact component = {Home}></Route>
                <Route path = "/contact" component = {Contact}></Route>
                <Route path = "/about" component = {About}></Route>
                <Route path = "/services" component = {Services}></Route>
          </Switch>
    </div>
    </Router>
  );
}
```

Exercise

- Create **Home** and **Cart** container
- Create new routes for 2 previous containers

Problem?


Brasov, Aurel Vlaicu 66
vanzari@frosera.ro
0729 872 510
Informatii
Blog
My Account
Logout
Ecosera
CAUTA PRODUSE
0 item(s) - 0,00 lei
How to pass previous router to breadcrumb?
Routes
Produse » Regulatori De Crestere » Arbusti
MAGAZIN
Acaricide
Fungicide
Ingrasaminte
Insecticide
Regulatori De Crestere
Repelenti Pentru Nematozi
Stimulatori de Inradacinare
Stimulatori Pentru Marirea Fructelor
Atlántica
FLORONE
Florone
61,21 lei – 178,13 lei
Selectează opțiuni

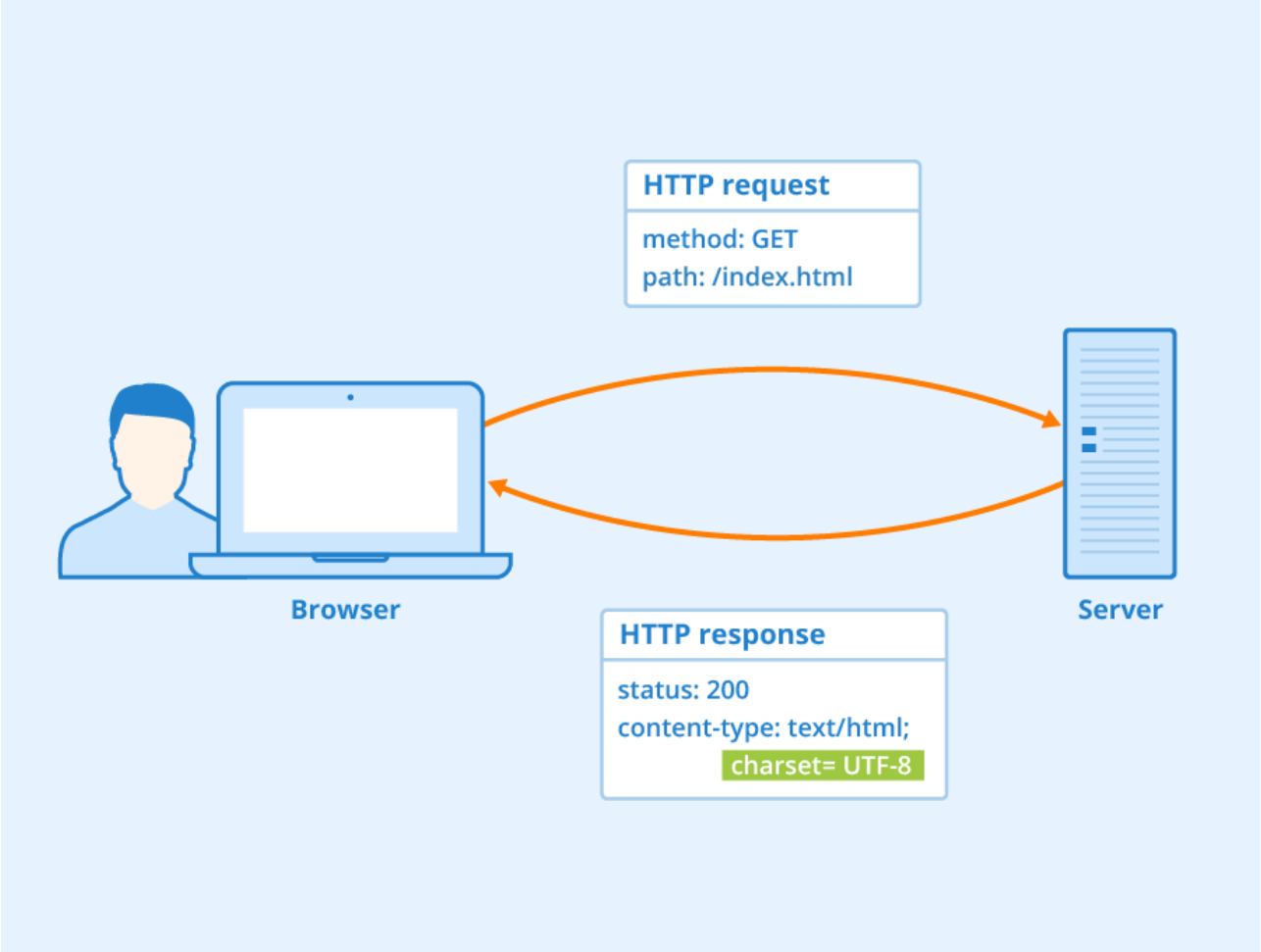
withRouter

- **withRouter** is a higher order component that will pass closest route's **match**, current **location**, and **history** props to the wrapped component whenever it renders.

```
import React from "react";
import PropTypes from "prop-types";
import { withRouter } from "react-router";

// A simple component that shows the pathname of the current location
class ShowTheLocation extends React.Component {
  static propTypes = {
    match: PropTypes.object.isRequired,
    location: PropTypes.object.isRequired,
    history: PropTypes.object.isRequired
  };

  render() {
    const { match, location, history } = this.props;

    return <div>You are now at {location.pathname}</div>;
  }
}

// Create a new component that is "connected" (to borrow redux
// terminology) to the router.
const ShowTheLocationWithRouter = withRouter(ShowTheLocation);
```

AXiOS

HTTP request & response



fetch()

- Allows to make HTTP requests to servers from web browsers.

```
fetch(url)
  .then((response) => {
    // handle the response
  })
  .catch((error) => {
    // handle the error
  });
```

fetch()

```
// user.json

[{
    "username": "john",
    "firstName": "John",
    "lastName": "Doe",
    "gender": "Male",
    "profileURL": "img/male.png",
    "email": "john.doe@example.com"
},
{
    "username": "jane",
    "firstName": "Jane",
    "lastName": "Doe",
    "gender": "Female",
    "profileURL": "img/female.png",
    "email": "jane.doe@example.com"
}
]
```

```
export default class App extends Component {
  state = {
    users: [],
  }

  componentDidMount() {
    fetch('./user.json')
      .then((response) => response.json())
      .then((data) => this.setState({ users: data }))
  }

  render() {
    return (
      <ul>
        {this.state.users.map((user) => (
          <li>{user.firstName}</li>
        ))}
      </ul>
    )
  }
}
```

axios

- With NPM:
  - **npm install axios**
- With Yarn
  - **yarn add axios**

axios

- Performing a request

```
axios
  .get(url)
  .then(function (response) {
    // handle success
    console.log(response)
  })
  .catch(function (error) {
    // handle error
    console.log(error)
  })
  .then(function () {
    // always executed
  })
```

axios

```
import React, { Component } from 'react'
import axios from 'axios'

export default class App extends Component {
  state = {
    users: [],
  }

  componentDidMount() {
    axios.get('/users').then(function (data) {
      // handle success
      this.setState({ users: data })
    })
  }

  render() {
    return (
      <ul>
        {this.state.users.map((user) => (
          <li>{user.firstName}</li>
        ))}
      </ul>
    )
  }
}
```

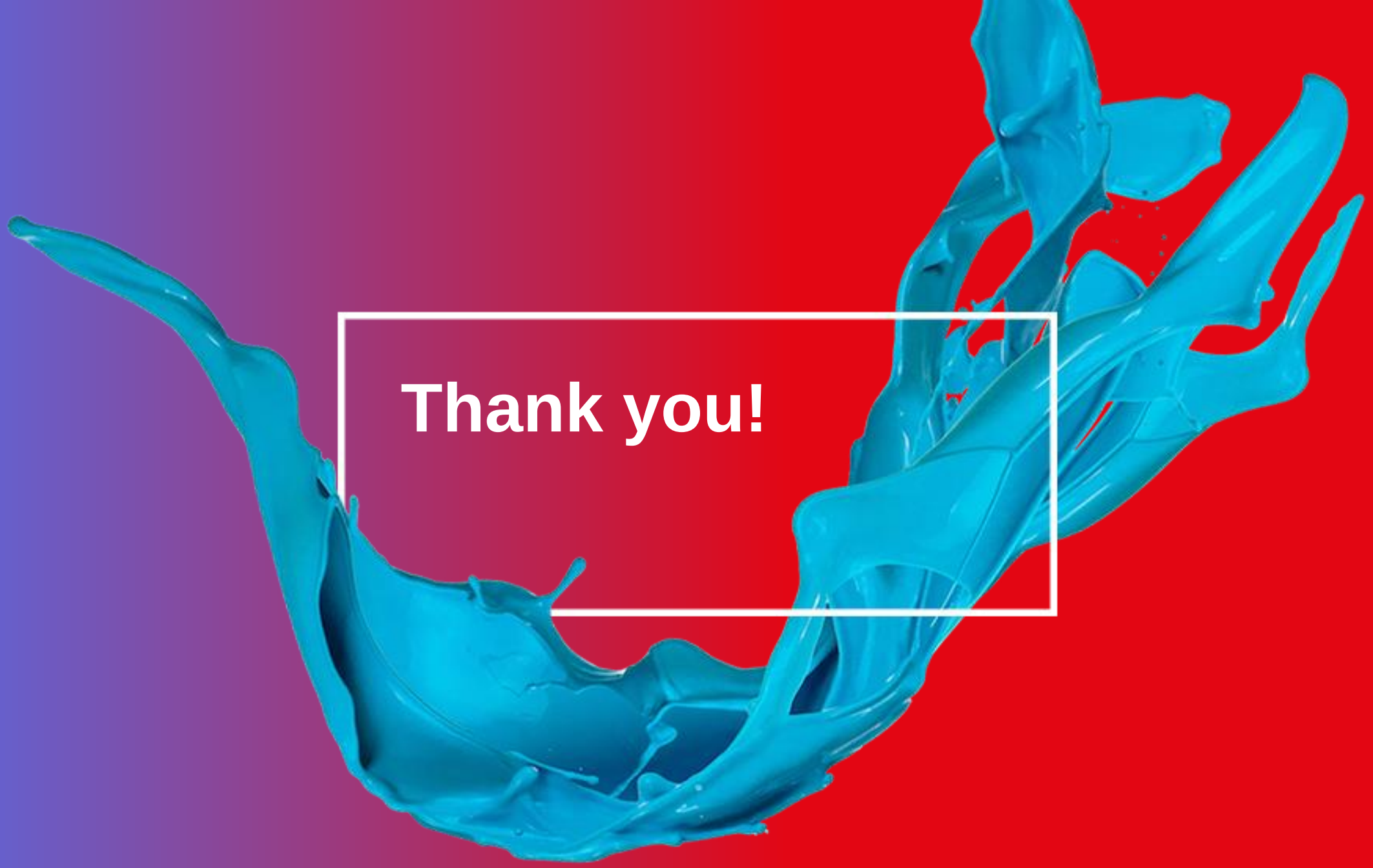
axios

- Advantages of using Axios over the native Fetch API include:
  - Request and response interception
  - Streamlined error handling
  - Protection against XSRF
  - Support for upload progress
  - Response timeout
  - The ability to cancel requests
  - Support for older browsers
  - Automatic JSON data transformation

axios

- For more information:
  - https://www.npmjs.com/package/axios

- Comparison between **axios** / **fetch:**
  - https://www.geeksforgeeks.org/difference-between-fetch-and-axios-js-for-making-http-requests/



Thank you!

Exercise

- Apply **withRouter()**

Router with authentication

```
import React, { useState } from 'react';
import './styles/index.scss';
import Login from './components/Login.jsx';
import Home from './components/Home.jsx';
import About from './components/About.jsx';
import Contact from './components/Contact.jsx';
import { render } from 'react-dom';
import { BrowserRouter, Route } from 'react-router-dom';

function App() {
  const [isAuth, setAuth] = useState(false);
  return (
    <BrowserRouter>
      <Route path="/"
        exact
        render={(props) => <Login {...props}
          isAuth={isAuth}
          callback={() => {
            setAuth(!isAuth)
          }}
        />} />
      <Route path="/home" render={(props) => <Home isAuth={isAuth} {...props} />} />
      <Route path="/about" render={(props) => <Home isAuth={isAuth} {...props} />}  />
      <Route path="/contact" render={(props) => <Home isAuth={isAuth} {...props} />}  />
    </BrowserRouter>
  )
}

const root = document.getElementById('root');

render(<App />, root);
```