

Integrating API into React App 2



Integration Considerations

- Review API Design
- Develop front end API Module
 - Endpoints
 - Routes
- React Authentication Context
- React Login/Registration Page

Review API Design

Fully understand the API

- Examine Routes
- Examine Request & Response Requirements
 - Body (e.g. properties)
 - Format (e.g. json)
- Examine Authentication
 - JWT/Sessions
- **TEST WITH POSTMAN**
 - Even better, write unit tests!

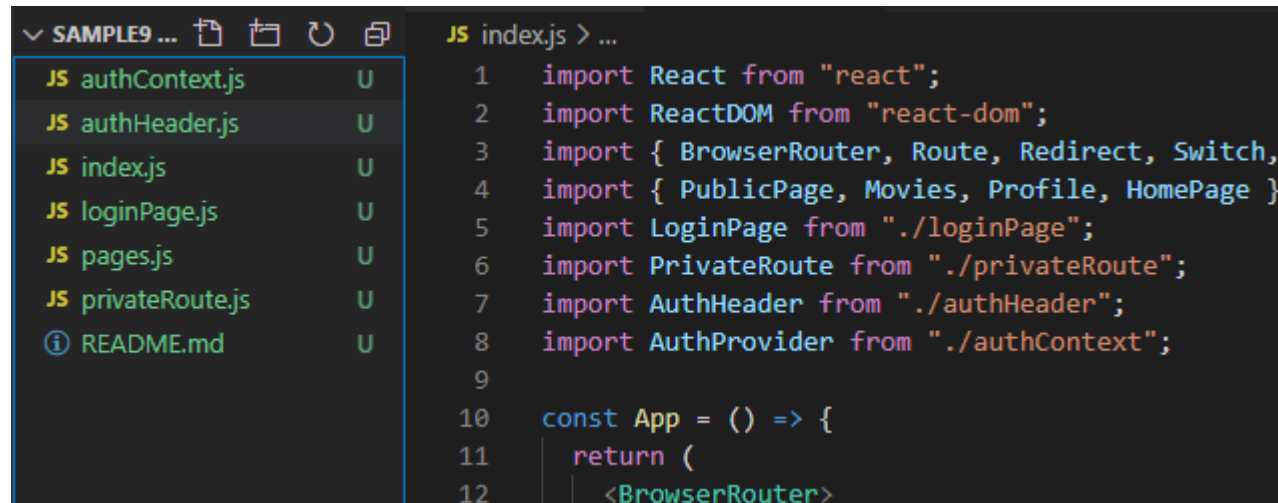
default

GET	/api/movies	List Movies
GET	/api/movies/{id}	Get a movie by id
GET	/api/genres	Get genres
GET	/api/users	Get list of uses (Only for Admins)
POST	/api/users	Authenticate/Register users
GET	/api/users/{userid}/favourites	Get User Favourites
POST	/api/users/{userid}/favourites	Add a favourite

```
[
  {
    "genre_ids": [
      28,
      14,
      878
    ],
    "_id": "5fd73334e4700a102472d5b7",
    "adult": false,
    "backdrop_path": "/jeAQdDX9nguP6YOX6QSWKDPkbBo.jpg",
    "id": 590706,
    "original_language": "en",
    "original_title": "Jiu Jitsu",
    "overview": "Every six years, an ancient order of jiu-jits  
mankind hangs in the balance.",
  }
]
```

Design React App

- Ideally this would be your Movies App from assignment 1
- You have used TMDB up to now
- You can update to include authentication(login/register) pages and routes



The screenshot shows a code editor interface. On the left is a file explorer for a project named 'SAMPLE9 ...'. It lists several files: authContext.js, authHeader.js, index.js, loginPage.js, pages.js, privateRoute.js, and README.md. Each file has a 'U' icon next to it. The 'index.js' file is selected. On the right, the code for 'index.js' is displayed. It shows imports for React, ReactDOM, and various components from 'react-router-dom' and local files. The code starts with a series of import statements, followed by a function definition for 'App'.

```
JS index.js > ...
1  import React from "react";
2  import ReactDOM from "react-dom";
3  import { BrowserRouter, Route, Redirect, Switch,
4  import { PublicPage, Movies, Profile, HomePage }
5  import LoginPage from "../loginPage";
6  import PrivateRoute from "../privateRoute";
7  import AuthHeader from "../authHeader";
8  import AuthProvider from "../authContext";
9
10 const App = () => {
11   return (
12     <BrowserRouter>
```

Write/Update API Module

- Abstract API calls to exported functions.

```
export const getMovies = () => {  
  ⚡ return fetch(  
    '/api/movies',{headers: {  
      'Authorization': window.localStorage.getItem('token')  
    }}  
  ).then(res => res.json());  
};  
  
export const getMovie = id => {  
  return fetch(  
    `/api/movies/${id}`, {headers: {  
      'Authorization': window.localStorage.getItem('token')  
    }}  
  ).then(res => res.json());  
};
```

Update pages to use Contexts

- Login Page:

```
    return (  
    <>  
      <h2>Login page</h2>  
      <p>You must log in to view the protected pages </p>  
      <input id="username" placeholder="user name" onChange={e => {  
        setUsername(e.target.value);  
      }}></input><br />  
      <input id="password" type="password" placeholder="password" onChange={e => {  
        setPassword(e.target.value);  
      }}></input><br />  
      { /* Login web form */}  
      <button onClick={login}>Log in</button>  
      <p>Not Registered?  
      <Link to="/signup">Sign Up!</Link></p>  
    </>  
  );  
);
```

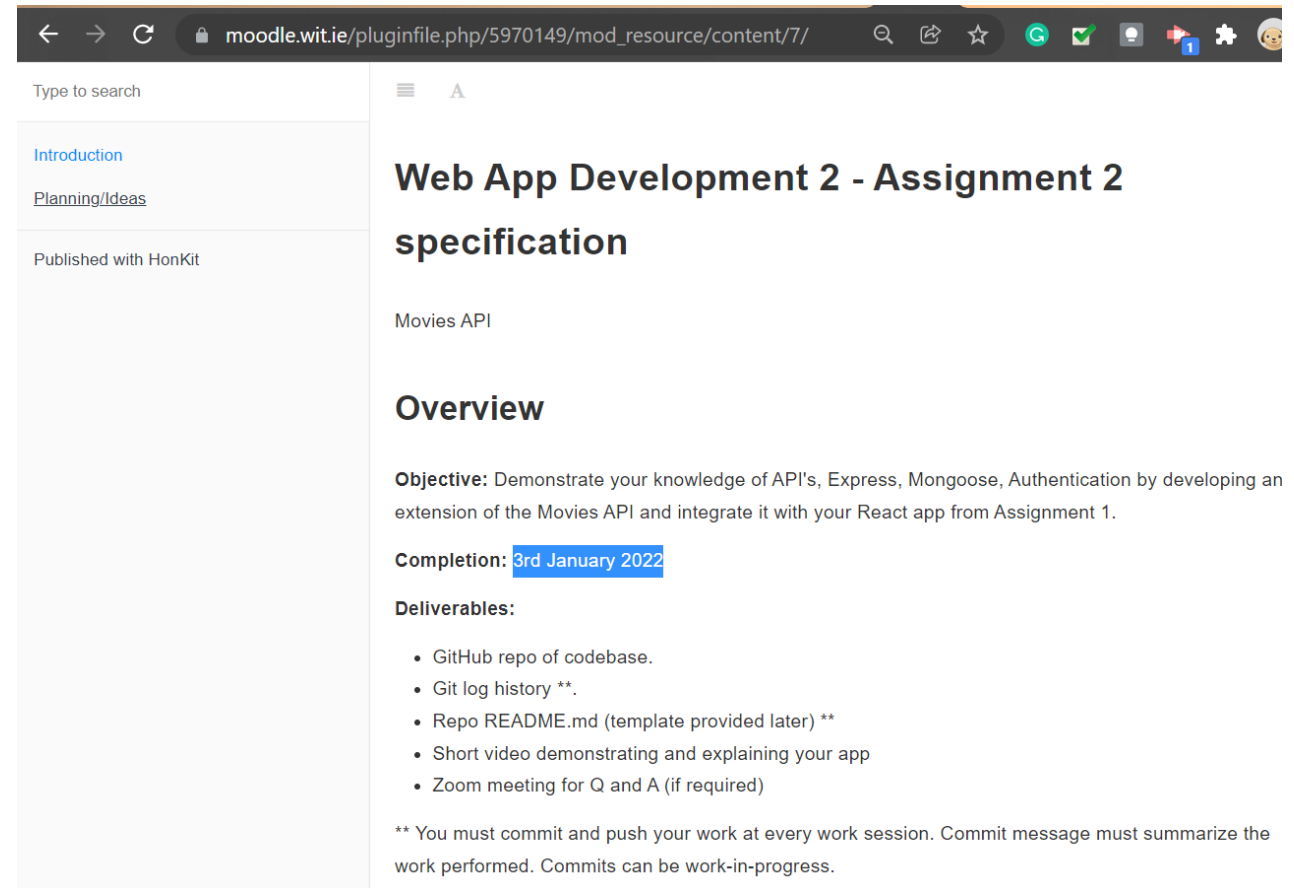
Assignment 2

Web API



Assignment 2

- It's here:
<https://moodle.wit.ie/mod/resource/view.php?id=3870295>
- **40% of your overall mark**
- Submit on Moodle (Repo)
- **Due 3rd January.**



The screenshot shows a web browser displaying a Moodle page. The address bar shows the URL: `moodle.wit.ie/pluginfile.php/5970149/mod_resource/content/7/`. The page title is "Web App Development 2 - Assignment 2 specification". The page content includes a sidebar with links for "Introduction" and "Planning/Ideas", and a main content area with the following sections:

- Web App Development 2 - Assignment 2 specification**
- Movies API
- Overview**
- Objective:** Demonstrate your knowledge of API's, Express, Mongoose, Authentication by developing an extension of the Movies API and integrate it with your React app from Assignment 1.
- Completion:** 3rd January 2022
- Deliverables:**
 - GitHub repo of codebase.
 - Git log history **.
 - Repo README.md (template provided later) **
 - Short video demonstrating and explaining your app
 - Zoom meeting for Q and A (if required)
- ** You must commit and push your work at every work session. Commit message must summarize the work performed. Commits can be work-in-progress.**

Labwork Assessment

- **10% of your overall mark**
- Finish changes to repo by **12th December** (or earlier if you're complete)
 - We will be looking at commit history.
- You've already submitted Moodle(repo)

Assessment

Lab Work Assessment (React)

Please submit a simple text file (.txt) containing your GitHub repository URL.

Hidden from students

 Submit Here

Hidden from students

Assignment 1 (React)

 Specification

 React Assign. (Submit Here)

Labwork Assessment (Web API)

 Web API 2 Labwork (Submit Here)

Assignment 2 (Web API)

 API Assign. (Submit Here)

Extra Stuff Examples

- Helmet
- [Better Error Handling](#)
- [Logging](#)
- <https://blog.bitsrc.io/23-insanely-useful-nodejs-libraries-you-should-know-in-2020-5a9b570d5416>

Helmet

npm package 4.2.0 dependencies none build passing license scan passing

Helmet helps you secure your Express apps by setting various HTTP headers. *It's not a silver bullet*, but it can help!

Quick start

First, run `npm install helmet --save` for your app. Then, in an Express app:

```
const express = require("express");
const helmet = require("helmet");

const app = express();

app.use(helmet());
```