**Module 3 Activity 5: Data Access**

Sai Kishore Reddy Konatham

IFT 544: Middleware Prog & Database Sec

Dinesh Sthapit

24 Sep 2023

Task1: Reusable components to create composite components.

A screenshot of a computer

Description automatically generated

After importing utility.js module in index.js :

A screenshot of a computer program

Description automatically generated

Added type= module in package.json to resolve the issue.

Now getting reference error for the generate random number function , to resolve this we have to add export in utility.js:

A screenshot of a computer

Description automatically generated

Index.js:

A screenshot of a computer

Description automatically generated

Output:

A screenshot of a computer

Description automatically generated

**Task 2: Using Importing External Modules (third-party modules).**

Link: <https://vpic.nhtsa.dot.gov/api/vehicles/GetModelsForMakeId/440?format=xml>

A computer screen shot of a computer

Description automatically generated

Link : <https://vpic.nhtsa.dot.gov/api/vehicles/GetModnelsForMakeId/440?format=json>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Task 3: Data Formats and Data Exchange**

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

Npm install node-fetch

Adding success message:

A screen shot of a computer

Description automatically generated

The `fetch` function is used to asynchronously retrieve data from an API. It operates without blocking the execution of other code.

When a request is made via `fetch`, JavaScript doesn't wait for the response. Instead, it continues executing any synchronous code that follows. In your example, the `console.log('Getting data')` statement is executed immediately after the `fetch` request is triggered. While the HTTP request is in progress, JavaScript proceeds with its other synchronous tasks. This means that 'Getting data' is displayed in the console right away.

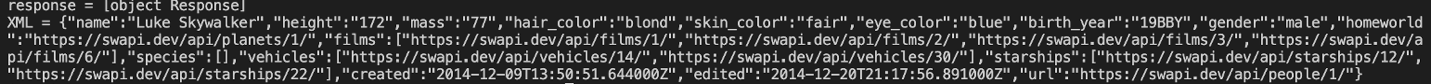
Once the API responds and the data is received, the callback functions specified in the `.then` methods are queued in the JavaScript event loop. They will execute once the current call stack is empty.

Consequently, the sequence of events unfolds as follows: 'Getting data' is printed first, followed by the JSON data and the success message, all of which are displayed once the `fetch` request is fully processed.

**Task 4: Content Negotiation**



After setting accept type to json:



Converting json to xml using json2xml:

A screen shot of a computer

Description automatically generated