Module 3-16

Consuming Web Services (Part 1)

Asynchronous Communication

- If a web application depends on an external application for data, it’s not reasonable to load all the page elements in a linear fashion.
  - What if the data source is offline? Your page would never finish load.
- These resources are therefore loaded asynchronously.

Axios

- Axios is a library available through NPM that simplifies the process of making asynchronous requests.
- When an asynchronous response is made, the immediate response is not the data itself, but an agreement that the data process will result in a success or failure.
  - This is known as a **Promise** object.

Implementing an Axios service

```
import axios from 'axios';

const http = axios.create({
  baseURL: "http://localhost:3000"
});

export default {

  getBoards() {
    return http.get('/boards');
  },

  getCards(boardID) {
    return http.get(`/boards/${boardID}`);
  }
}
```

We can implement an axios service within VUE by including the code in its own separate JS file.

The highlighted code shows two possible GET requests made to the web service.

Let’s create an Axios Service

Calling the Service from VUE Components (1/3)

```
<script>
import boardsService from '../services/BoardService.js';

export default {
  data() {
    return {
      boards: [],
      isLoading: true
    };
  },
…
}
```

Importing

This is process is similar to that of importing regular VUE components.

Calling the Service from VUE Components (2/3)

```
...
    created() {
    boardsService.getBoards()
      .then(response => {
      this.boards = response.data;
      this.isLoading = false;
    });
  }
};
```

Create a Life Cycle Hook (if needed):

We are creating a component lifecycle hook called created().

The code inside created() will run immediately after the components comes to life (before it's rendered and displayed to the user).

Calling the Service from VUE Components (3/3)

```
...
    created() {
    boardsService.getBoards()
      .then(
            response => {
                  this.boards = response.data;
                  this.isLoading = false;}
      );
  }
};
```

Call the service method

The usage of “then” is known as promise chaining. Because the communication is asynchronous, this means:

Only perform these actions on the **response** object when the request completes.

Let’s have our components use the Axios Service