**Assignment 11**

**The Asynchronous Loading Employee Management Application**

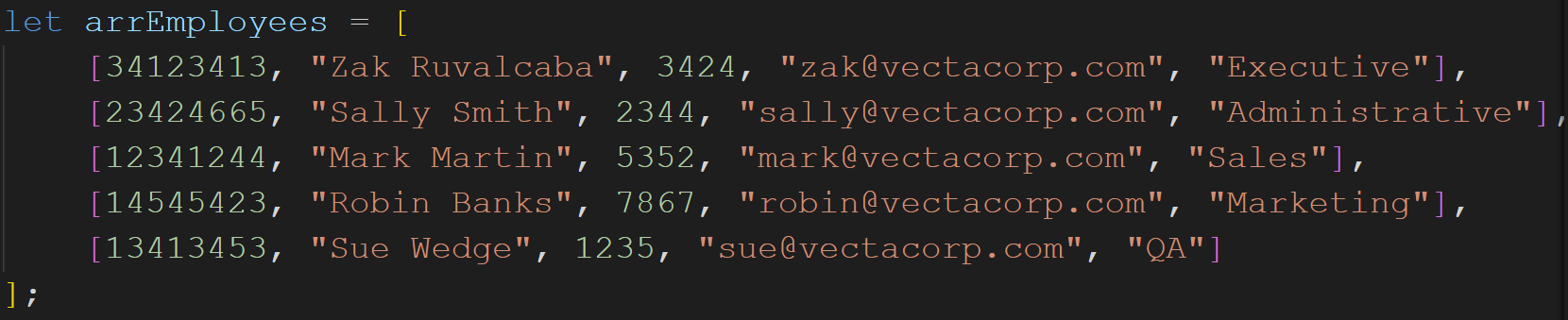
In this assignment you will be responsible for modifying the code of the Employee Management Application so that it loads all employee data in an asynchronous way. This assignment will ultimately prep you for making real API calls to a web server like Node.js. In order to successfully complete this assignment, follow the steps outlined below.

General Considerations

* The add functionality has been removed. In a real asynchronous application, an add operation would be processed by a web API built on top of Node.js.
* The delete functionality has been modified to remove the row from the table. Since we’re no longer using arrays, there’s nothing to actually delete as you can’t delete data contained within a JSON file. Here again, in a real asynchronous application, a delete operation would be processed by a web API built on top of Node.js.
* The web storage code has been removed. When working with data asynchronously, you’ll likely be relying on a database to store data. Since this is the case, any create, retrieve, update, and delete operations moving forward would go through a web API built on top of Node.js.

**Move Employee Data to a JSON File**

You first step will be to move the employee data (pictured below) into a JSON file. Place this within a folder called data and name the file employees.json.



Remember, we’re using nested arrays here. You’ll need to convert each nested array into JSON format in order for it to work correctly. This means that each nested array will now be an object. Since that’s the case, you’ll need to add a key for each value.

**Moving the Data Load into a Module**

Your second step will be to move the loading of the JSON data into a module named init.js. This module folder will be called modules and will reside within the js folder. Within the init.js file, use async / await to load the JSON data and return it back into the buildGrid() function.

General Considerations

* Fetching the data from the module will return a Promise object. This should tell you that you’ll need to either use another async / await within the buildGrid() function or you’ll need to use .then() to process the return JSON object.
* You will no longer be using an array. It will now be an object. So your iterative code within the buildGrid() function will be a bit different. Make sure to remove all references to arrEmployees as this array will no longer be valid.