College of Engineering, Construction & Living Sciences
Bachelor of Information Technology
ID607001: Introductory Application Development Concepts
Level 6, Credits 15

In-Class Activity: Project 1: Node.js REST API Planning

Instructions

The main purpose of this in-class activity is to plan your **Project 1: Node.js REST API**. In addition, you will explore how to automatically restart your simple **API's** server using **Nodemon** and get an institution by its id.

Code Review

You must submit all program files via **GitHub Classroom**. Here is the URL to the repository you will use for your code review – https://classroom.github.com/a/P656imf2. Checkout from the **main** branch to the **03-in-class-activity** branch by running the command - **git checkout 03-in-class-activity**. This branch will be your development branch for this activity. Once you have completed this activity, create a pull request & assign the **GitHub** user **grayson-orr** to a reviewer. **Do not** merge your pull request.

Getting Started

Open your repository in **Visual Studio Code**. Create a simple **API** as described in the lecture notes.

Nodemon

Nodemon is a tool that helps you develop **Node.js** applications by automatically restarting the application when a file change is detected. It does not require additional changes to the application's code to get started. To use **Nodemon**, install it as a development dependency. In **package.json**, replace the **start** script value **node app.js** with **nodemon app.js**.

Get an institution by its id

In the lecture notes, you looked at how to create, read, update and delete an institution. Extend your **API's** functionality by creating a route and controller function that enables a user to get an institution by its id. **Hint:**

The approach is similar to updating and deleting an institution.

Project 1: Node.js REST API planning

You will be starting your **Project 1: Node.js REST API** assessment next week. Before you start, you need to decide your **API's** theme and the data you are going to display to the user. You need at least **five** collections (**user collection** is included) with at least **three fields** of data. Also, show the relationships between **collections**. **Note:** You need at least **two relationships** between **collections**. You can display this anyway you wish, i.e., UML, text, etc. As long as it is clear to the **course lecturer** when reviewing.