Degrees Restaurant at Columbus State, API v1.0

Lab 1, Part 1

In this lab, we develop version 1.0 of the Degrees Restaurant at Columbus State website Application Programming Interface (API)

Assignment Workflow and Submission Directions:

1. Create a new GitHub repository.
   a. On the Create a new repository page, enter the required repository name as **java-4-degrees-project**, being sure to mark it as **private**.
2. Add your instructor as a collaborator.
3. Complete the steps in the assignment below, then continue with steps 4-6.
4. Create a **GitHub Pull Request** for your **private** repository and add your instructor as a reviewer.
5. From the command line at the root directory of your project execute the following command: **git bundle create firstname-lastname-degrees-lab1-date.bundle --all** where **firstname** and **lastname** are your first and last names, and date is in the format of **mmddyyyy**.
6. Submit your lab solution in Blackboard with the link to your Pull Request as well as the bundle file you just created.

Degrees Restaurant at Columbus State, API v1.0, Lab Setup and Assignment

For this course, students complete one project by part with ***EACH part building on the previous part.***

Lab Setup

After creating a new repository, complete the lab setup:

Step 1: Spring Initializer

A. Use Spring Initializr to create a new Spring Boot project with the following settings:

- Group: edu.cscc
- Artifact: degrees
- Description: <Provide an appropriate description>
- Spring Boot: 2.3.7
- Project: Maven Project
- Language: Java
- Packaging: jar
- Java: 8
- **Dependencies:**
  - Spring Web

B. To create the project zip file, click: **Generate**

C. Unzip the project into a development directory of your choosing

Step 2: IntelliJ

A. Start IntelliJ (Close any projects that open)

B. From the Welcome screen, choose: **Open**

C. Navigate to the location of your newly unzipped project from Step 1 (page 1)

NOTES:

- If you get a message about Maven projects needing to imported, choose: **enable auto-import**
- It may take a few minutes for IntelliJ to import the project and download all dependencies.

Step 3: Confirm Application Runs

A. From the project pane, expand: **src > test > java > edu.cscc.degrees**

B. Right-click: **DegreesApplicationTests**

C. Choose: **Run DegreesApplicationTests**

IMPORTANT: Make sure the unit test runs successfully before continuing.

Step 4: Commit changes into GitHub

A. Create a local Git repository:

- From the IntelliJ menu, choose: **VCS > Import into Version Control > Create Git Repository**
- Choose the automatically selected default of your project's top directory for your local repo.
- Create a new branch called **"lab-1-solution"**: Git > New Branch > enter a branch name.

B. Check-in your initial commit:

- Add a line with **`.mvn`** at the bottom of your **`.gitignore`** file.
- Choose **VCS > Commit**.
- Check the box to the left of "Unversioned Files."
- In the **Commit Message** box, Enter: Initial commit
- Press "Commit"

C. Push your project to GitHub:

- Open a terminal window and navigate to your top-level project directory
- From the **Code** tab of the GitHub repo, created in step 1, copy and paste the "…or push an existing repository from the command line" commands shown into the terminal window
- After the push completes, refresh the browser page on your repo and verify the **Code** tab now shows the code from your project

Lab Assignment

For this assignment, fully implement API spec from **"lab-1-degrees-api-v1_0.zip"**. Once you have unzipped this file open the "index.html" file in your web browser, which describes the expected endpoints, requests and responses. You can open this file in your web browser to view the documentation.

Lab Implementation Requirements:

1. Create a new branch for your work VCS > Git > Branches then +New Branch
2. Create one, or more, unit tests for each path and method in version 1.0.0 of the API spec which checks:
    a. POST accepts a new item and returns 201 created
    b. the location of the newly created item in the headers
3. Create a REST API controller with the methods necessary to get each test to pass
4. Validate that your endpoint works as expected using Postman by creating a **POST** request to the **"/api/menu/categories"** endpoint.

Lab Assignment Notes:

- Write unit tests using mocks and stubs for your REST API controller. (**Spring Web Essentials, Chapter 4, Our first application programming interface (API)**)
- Although we are performing a POST for our first endpoint, **there is no persistence in this lab**.
- When returning the Location header the value should be at the location "**http://localhost/api/menu/categories/1**".

NOTE: Write **ONE** unit test at a time, adding just enough controller code, to make it pass. If you try to do too much at once, it's easy to be overwhelmed with issues.

Pushing Your Work to GitHub

Once you are satisfied with your work, ***submit the lab for review to your instructor via GitHub BEFORE*** submission to Blackboard.

1. Using git, push your branch to your private repository.
    a. Using the same URL that you would use to clone the repository, use git to add a new "remote" source: **`git remote add origin <the clone URL from the repository>`**
    b. Push your work to the remote repo (you may need to enter your GitHub credentials if prompted): **`git remote push origin --all`**