Course Code/Course Title:

FSW104-JV Back End Foundations

Course Description: The Back-End Foundations course is an introduction to creating server-side web applications and services. Students will learn how to create server-side APIs and render websites. Language-specific concepts will be covered, outlining the nuances associated with the elective framework.

Course Length: 40 hours	**Prerequisites:** FSW102	**Proficiency Exam** ☐Yes [✓ No]	
Course Start Date: **Course End Date:**	**Meeting Days/Times:**		

Required Resources:
Minimum: PC (Windows) or MacBook laptop. 4GB ram, 256GB HD, Core i5
Recommended: PC (Windows) or MacBook laptop. 8GB ram, 256GB SSD, Core i5

This will become your programming rig. Every student will need their own laptop. We will be downloading programming resources to your laptop, and it will also house your coding projects.

Additional Resources:
Ground students are expected to supply notebooks, pens, pencils, highlighters, folders, ring binders, calculators, USB storage devices and other general supplies as needed to aid in the collection and storage of information in their courses.

A. **For Classes Delivered in an Online Format (for approved courses and campuses).** Online courses are delivered via https://wozu.exeterlms.com in an asynchronous format. Students enrolled in online courses/programs are expected to spend an equivalent amount of time on task, as campus-based students, in meeting course objectives. For Online Courses the total expected hours required for completion of course objectives are identified on the syllabus as **Total Contact Hours** and reflect the sum of theory, laboratory, and outside hours.

Educational Objectives:

Upon successful completion of this Program, students will be able to:

1. Learn the basics of the request/response cycle
2. Learn how to render HTML in response to a request
3. Learn how to render JSON or XML in response to a request
4. Understand how to build a REST endpoint
5. Learn how to interface with a database system from a web application

Course Outline

Lessons:

Week 1

1. **Introduction to Backend Foundations with Java:** Includes Why backend, Spring framework, setup, Postman, Simple backend application, explore example
2. **HTTP and Servlets:** Includes Intro to HTTP, HTTP request lifecycle, HTTP messages, HTTP verbs, Servlet example, test with Postman, changing header information, HTTP Status codes
3. **Spring MVC:** Includes What is MVC, request lifecycle, basic Spring MVC application, creating HomeController, adding a view, run and test, adding functionality, creating a model, update views, update HomeController, adding CSS

Week 2

4. **Data Persistence:** Includes Setup, SQLectron, create a new project, creating MVC components, controller, connecting to SQLite database, adding external dependencies, connect to and query the database, datasource, JNDI or JDBC
5. **ORM:** Includes Object-relational mapping, hibernate, walkthrough, post model, create a data access object (DAO), configure database, add a MessageController
6. **REST and Web API's:** Includes RESTful dogo example, rest controller, adding static files, modify message controller, run and test

Week 3

7. **Security:** Includes Securing a web application
8. **Logging and Debugging:** Includes Debugging, debugging with breakpoints, breakpoints, conditional breakpoints, logging, add log configuration, changing log levels
9. **Front End Integration:** Includes Front-end integration, add CORS support, adding a front-end, connecting with React, Axios, connecting with Angular, GET request, add component, render component, CORS error
10. **Final Project**

Outline:

- **L2 Hands On:** You should leverage what you have learned about the HTTP protocol, annotations, and CRUD operations to create a servlet that responds to the four CRUD operations (i.e., GET, POST, PUT, and DELETE). You will create a class called ProjectServlet that extends the necessary class and implements the necessary CRUD methods. The servlet should be linked to the path /ProjectServlet. Use the given table that contains the CRUD methods, their responses for the client, and the console printing message. Use Postman to send a header ID to the server called Payload-Data with the value of "it works!". Then, implement the code on the client side within any method of your choice which retrieves the header ID and prints the value to the console.
- **L3 Practice Hands On:** You should leverage what you have learned about the MVC architecture, the request lifecycle, and the DispatcherServlet front controller to create a web application which will display a view in the browser. The controller in this application should listen for requests made to the path /beginning and return the beginning.html view as the response. The beginning.html view can contain any text you wish ("Hello, World" is not a bad default). The controller should respond with the view for GET requests.

- **L4 Hands On:** Create a simple MVC web application using Java to query the database. You will be using the MusicBuddy project.
- **L5 Practice Hands On:** Create a route that will allow an entry to be updated and that can accept an object matching the Message model and will update the values of an entity that already exists in the SQLite database. Use Postman to test this route.
- **L6 Hands On:** Create a Get method that maps to the route api/chat and returns JSON data. Display the results returned from the api/chat route to a webpage. Use any library/framework you are comfortable with to consume the API. (React, Angular, jQuery, etc...)
- **L7 Practice Hands On:** Using the application built in the lesson create a new controller that is only accessible when logged in. Then create a new folder called js in the src/main/resources/static folder and allow it to be accessed only after logging in. Add a javaScript file to the folder and add an alert. Then connect the file to the newly created route via a script element.
- **L8 Hands On:** Locate the source of the error in the MVC web application provided for you below. The goal of this project will be to fix the problem with the following program. The IDs are not generated as expected. The goal of generating new IDs for each message was to ensure the IDs could only be odd numbers (i.e., 1, 3, 5, 7, etc.).
- **L9 Practice Hands On:** Take a Spring project and use it as the backend for your application. You will enable CORS and then use the application to serve data to your choice of front-end applications.
- **L10 Final Project:** Final Project

Final Project:

Using Java and Spring, create a new Spring application which will allow users to create a profile and log in using a username and password. Users with a profile can create Posts, containing a string message up to 140 characters. The posts made by a user can be viewed without needing to log in. This application should support the four basic CRUD operations: Create, Read, Update, and Delete.