DCN2 — DCN2 TASK 1: BACK-END APPLICATION PROGRAMMING

BACK-END PROGRAMMING — D288
PRFA — DCN2

COMPETENCIES

4085.1.1 : Develops Object-Oriented Applications
The learner develops object-oriented applications that can be integrated with relational databases.

4085.1.2 : Writes Code
The learner writes code for object-oriented applications using Spring framework.

4085.1.3 : Implements Design Pattern
The learner implements design patterns for object-oriented applications.

INTRODUCTION

Throughout your career in software design and development, you will be asked to create, customize, and maintain applications with various features and functionality based on business requirements. For this assessment, you will create a Spring Framework Java back end for a web application using the solution statements provided in the requirements section of this assessment. The skills you showcase in your completed application will be useful in responding to technical interview questions for future employment. This application may also be added to your portfolio to show to future employers.

You will be building your project using IntelliJ IDEA (Ultimate Edition) in a WGU-provided lab environment in the LabFiles folder. You will be working with an existing MySQL database and Angular front end, which are supplied for you in the lab environment. You will share this project to your GitLab repository and backup regularly. Use the GitLab link in the web links section to create your Gitlab project in the WGU GitLab space, and prior to starting your work, reference the “GitLab How-To” web link to set up your project.

Note: Do not modify the given Angular front-end for this project.

SCENARIO

A travel agency has recently launched a complete overhaul of their front-end vacation bookings application using Angular and JavaScript. Lately, the front-end engineers have encountered various undocumented bugs when sending requests and fetching data from the back-end. Since the back-end was built in the early 1990s and the original developer has since retired and can no longer help troubleshoot, the existing team is concerned about the growing tech debt and lack of ongoing support. Your chief technology officer (CTO)

decided to create a project to port over any mission-critical functionalities to a modern framework and has selected you, a software developer in Java, to start developing the minimally viable product (MVP) to migrate the legacy back-end to the modern Spring framework.

REQUIREMENTS

Your submission must be your own original work. You may not use or reference other students' submissions for this task. For more information please review our Academic Authenticity policies and the Professionalism and Conduct Expectations for College of Information Technology Students.

You must use the rubric to direct the creation of your submission because it provides detailed criteria that will be used to evaluate your work. Each requirement below may be evaluated by more than one rubric aspect. The rubric aspect titles may contain hyperlinks to relevant portions of the course.

*Tasks may **not** be submitted as cloud links, such as links to Google Docs, Google Slides, OneDrive, etc., unless specified in the task requirements. All other submissions must be file types that are uploaded and submitted as attachments (e.g., .docx, .pdf, .ppt).*

Note: External plugins and libraries other than those specified in this task are not allowed.

A. Create a new Java project using Spring Initializr, with *each* of the following dependencies:
   - Spring Data JPA (spring-boot starter-data-jpa)
   - Rest Repositories (spring-boot-starter-data-rest)
   - MySQL Driver (mysql-connector-java)
   - Lombok

   Note: Since the application properties will be empty, you will need to copy over the supplied application properties.

B. Create your subgroup and project by logging into GitLab using the web link provided and do the following:
   - connect your new Java project
   - commit with a message and push when you complete *each* of the tasks listed below (parts B to F, etc.)

     Note: Any submissions that do not have a commit after each task will not be evaluated.

     Note: You may commit and push whenever you want to back up your changes, even if a task is not complete.

   - Submit a copy of the git repository URL and a copy of the repository branch history retrieved from your repository, which must include the commit messages and dates.

     Note: Wait until you have completed all the following prompts before you create your copy of the repository branch history.

C. Construct **four** new packages, **one** for *each* of the following: controllers, entities, dao, and services. The packages will need to be used for a checkout form and vacations packages list.

   Note: The packages should be on the same level of the hierarchy.

Note: Construct a package named config and copy the RestDataConfig.java provided in the laboratory environment to the package. Modify it so that the package and imports have the correct package and import addresses. Copy the application.properties file that is provided in the laboratory environment into your application properties resource file.

D. Write code for the entities package that includes entity classes and the enum designed to match the UML diagram.

E. Write code for the dao package that includes repository interfaces for the entities that extend JpaRepository, and add cross-origin support.

F. Write code for the services package that includes *each* of the following:
- a purchase data class with a customer cart and a set of cart items
- a purchase response data class that contains an order tracking number
- a checkout service interface
- a checkout service implementation class

G. Write code to include validation to enforce the inputs needed by the Angular front-end.

H. Write code for the controllers package that includes a REST controller checkout controller class with a post mapping to place orders.

Note: You do not need to duplicate REST functionality for each repository by creating methods in Java.

I. Add **five** sample customers to the application programmatically.

Note: Make sure the customer information is not overwritten each time you run the application.

J. Run your integrated application by adding a customer order for a vacation with **two** excursions using the unmodified Angular front-end. Provide screenshots for the following:
- that your application does not generate a network error when adding the data
- your database tables using MySQL Workbench to show the data was successfully added

Note: The screenshot should include the front-end view and the inspection console in the browser.

K. Demonstrate professional communication in the content and presentation of your submission.

File Restrictions

File name may contain only letters, numbers, spaces, and these symbols: ! - _ . * ' ( )
File size limit: 200 MB
File types allowed: doc, docx, rtf, xls, xlsx, ppt, pptx, odt, pdf, csv, txt, qt, mov, mpg, avi, mp3, wav, mp4, wma, flv, asf, mpeg, wmv, m4v, svg, tif, tiff, jpeg, jpg, gif, png, zip, rar, tar, 7z

RUBRIC

A:SPRING INITIALIZR

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A new Java project is not created using Spring Initializr.	The new Java project is created using Spring Initializr and includes *all* 4 of the listed dependencies, but the project does not run successfully or has errors.	The new Java project is successfully created using Spring Initializr and includes *all* 4 of the listed dependencies.

B:GIT REPOSITORY

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A GitLab repository is not provided.	The subgroup and project are created in Gitlab, but the submission does not include *all* 3 of the listed requirements, or some of the requirements contain errors.	The subgroup and project are created in GitLab correctly and the submission includes *all* 3 of the listed requirements, and none of the requirements contain errors.

C:PACKAGE CONSTRUCTION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
4 packages are not constructed.	4 new packages are constructed, but the submission is missing 1 or more of the 4 given packages, or the packages are not appropriate for a checkout form and vacations packages list.	4 new packages are accurately constructed, 1 for *each* of the 4 packages: controllers, entities, dao, and services. The packages are appropriate for a checkout form and vacations packages list.

D:ENITIES PACKAGE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The code is not provided for the entities package.	The code is provided for the entities package, but the code does not accurately represent the UML diagram. Or the code is not fully functional.	The code provided for the entities package includes entity classes designed to match the UML diagram. The code is fully functional.

E:REPOSITORIES PACKAGE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT

The code is not provided for the dao package.	The code is provided for the dao package, but the code is missing some repository interfaces for the entities that extend the JpaRepository or does not include *any* cross-origin support. Or the code is not fully functional.	The code provided for the dao package includes repository interfaces for the entities that extend the JpaRepository and adds cross-origin support. The code is fully functional.

F:SERVICES PACKAGE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The code is not provided for the services package.	The code is provided for the services package but does not include 1 or more of the 4 given parameters. Or the code is not fully functional.	The code provided for the services package includes *all* 4 of the given parameters. The code is fully functional.

G:VALIDATION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The code to include validation is not provided.	The code is provided to include validation but does not include *all* of the validation that is needed by the Angular front-end. Or the code is not fully functional.	The code provided includes *all* validation to enforce the inputs needed by the Angular front-end. The code is fully functional.

H:CONTROLLERS PACKAGE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The code is not provided for the controllers package.	The code is provided for the controllers package but does not include a REST controller checkout controller class. Or the class does not include a method with a post mapping to place orders. Or the code is not fully functional.	The code is provided for the controllers package and includes a REST controller checkout controller class with a post mapping to place orders. The code is fully functional.

I:ADD CUSTOMERS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No sample customers are added to the application.	Sample customers are added to the application, but fewer than five.	Five sample customers are added to the application. The sample data is added correctly.

J:RUN APPLICATION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The application does not run when adding a customer order for a vacation with 2 excursions.	The application runs when adding a customer order for a vacation with 2 excursions, but the application generates a network error when adding the data, does not save the data, or the Angular front-end was modified. Or the submission does not provide clear or complete screenshots, or 1 or *both* of the given requirements are incorrect.	The application runs successfully when adding a customer order for a vacation with 2 excursions while using the unmodified Angular front-end. Clear and complete screenshots are provided that show *both* given requirements were completed correctly.

K:PROFESSIONAL COMMUNICATION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
Content is unstructured, is disjointed, or contains pervasive errors in mechanics, usage, or grammar. Vocabulary or tone is unprofessional or distracts from the topic.	Content is poorly organized, is difficult to follow, or contains errors in mechanics, usage, or grammar that cause confusion. Terminology is misused or ineffective.	Content reflects attention to detail, is organized, and focuses on the main ideas as prescribed in the task or chosen by the candidate. Terminology is pertinent, is used correctly, and effectively conveys the intended meaning. Mechanics, usage, and grammar promote accurate interpretation and understanding.

WEB LINKS

Integrated Development Environment – IntelliJ IDEA (Ultimate Edition)
Sign up for free student account using WGU.edu email address.

GitLab

Lab Environment

Spring Initialzr

GitLab How-To
GitLab Instructions Knowledge Base Article

Lab Environment Set Up Instructions
Lab Knowledge Base Article

SUPPORTING DOCUMENTS

ERD Diagram.pdf

IntelliJ Ultimate Edition Directions.pdf

Vacation UML Class Diagram.pdf