**CST-339 Programming in Java III**

**CLC Project Assignment – Overview**

Introduction:

In this class, you will design and build a complete enterprise class N-Layer application using the Spring Boot framework. The application must adhere to a specified set of high-level functional and technical requirements, as shown in the figure below. You will have the freedom to pick the application you want to design and develop. You will work in teams of no more than 2 unless there is an odd number of students, in which case a team of 3 will be acceptable. Example applications could include, but are not limited to, an eCommerce application, customer management application, order management application, blog site application, or DVD/Book/Music management application.

Each milestone delivery will include a Design Report that captures the technical approach, design decisions, UI wireframe designs, Sitemap designs, ER database designs, and project risks/issues. All code developed during the project will be maintained in a GCU approved private GIT repository.

You will need approval for the project chosen from the instructor prior to moving onto your detailed design and development of your application.

* Spring MVC (views, models, and controllers)
* Spring Core (business and data services)
* Spring Data JDBC (persistence)
* Spring Core (IoC and DI)
* Spring REST Services (API’s)
* Bootstrap (responsive design)
* Spring Boot Embedded Tomcat (as a JAR)
* MySQL (relational database)

Presentation

Bus. API

Data Access API

DB

The design and code must support the following high-level requirements:

* Your application must implement a user registration module and login module.
* Your application must be designed using an N-Layer architecture with distinct and separate presentation components, business services, and persistence services.
* Your application must adhere to industry best practices, exception handling, and error handling as discussed either in topic readings, lectures, or provided as peer code review feedback.
* Your application presentation must be entirely written using Spring MVC compliant pages using Thymeleaf templates.
* Your application must perform data validation on all form data entry fields.
* Your application must support a responsive design using the Bootstrap framework.
* Your application must not have business logic or business rules implemented in Spring MVC views, models, or controllers.
* Your application must implement all CRUD methods on whatever business domain is being addressed (i.e., products, music, blogs, etc.).
  + A page that lists all “products” as a tabular report.
  + A page that allows a user to create a new “product.”
  + A page that allows a user to display the details of a “product.”
  + A page that allows a user to update an existing “product.”
  + A page that allows a user to delete an existing “product.”
* Your application must use a relational database such as MySQL or PostgreSQL. It should be noted that if you develop your application using Spring Data JDBC and use the CrudRepository, you could easily port your application over to use MongoDB, but this will not be possible until you finish Topic 5. You must receive permission from your instructor to use MongoDB.
* Your application must use the Spring JDBC or Spring Data JDBC to access the database.
* Your application must use SpringBeans to implement all business services and persistence services.
* Your application must use proper declarative Spring annotations within all components and use DI for all models, controllers, services, and resources required by the application.
* Your application must be deployed using Spring Boot and use the embedded Tomcat (version 9.x or later).
* Your application must not be able to access secure pages (all but the root, registration, and login pages) without first logging into the application. Your application will automatically redirect the user to the login page if they try to access a secure page without first logging in. Your application must be secured using Spring Security form-based authentication and a database.
* Your API’s must not be anonymous and must be secured using Spring Security, at a minimum using basic HTTP authentication using a database.
* Your application classes must be fully documented using JavaDoc format.
* Your application must be deployed to a GCU approved cloud platform.

Project Milestones:

The team project will be designed and built using an iterative approach and delivered using the following 8 milestones. It should be noted that all milestones will include a Design Report and, except for Milestone 1, application code will be also be delivered to support the appropriate milestone requirements. It is expected that the code will be refactored during each iteration based on peer code review or instructor feedback.

Milestone 1:

* Project Proposal, draft Sitemap, and draft division of work across team.

Milestone 2:

* Main application module (using Spring MVC).
* Registration module (using Spring MVC).
  + Without a database.
* Login module (using Spring MVC).
  + Without a database.
* Responsive design (using the Bootstrap Framework).
* Page layouts using Thymeleaf layouts.
* Design Report.
* Screencast running on local development environment.

Milestone 3:

* Product creation module (using Spring MVC and Spring Core).
  + Registration module and login module will be refactored to use SpringBeans and IoC.
  + Without a database.
* Updated Design Report.
* Screencast running on local development environment.

Milestone 4:

* Refactoring.
  + Registration module and login module will be refactored to use using Spring JDBC or Spring Data JDBC.
  + Product creation module will be refactored to use using Spring JDBC or Spring Data JDBC.
* Built as a JAR file using Maven.
* Deployed to a cloud platform.
* Updated Design Report.
* Screencast running from cloud platform.

Milestone 5:

* Product display module (using Spring MVC and using Spring JDBC or Spring Data JDBC).
* Product update/delete modules (using Spring MVC and using Spring JDBC or Spring Data JDBC).
* Updated Design Report.
* Screencast running from cloud platform.

Milestone 6:

* Refactored login module (using Spring Security).
  + Use form-based authentication and a database.
  + All pages but the login module and registration module will be protected and be secured using Spring Security.
* Updated Design Report.
* Screencast running from cloud platform.

Milestone 7:

* REST API’s:
  + REST API design using Word template or Swagger.
  + REST API 1: Return all products.
  + REST API 2: Return a desired product.
  + All API’s must be secured using Spring Security and, at a minimum, use basic HTTP authentication using a database.
* Design and code cleanup.
* Updated Design Report.

Milestone 8:

* JavaDoc generation.
* Final application code.
* Final project presentation.
* Final Design Report.