**CST-341 CLC Project Guide**

**Directions:** Throughout this course, students will incrementally design and build an enterprise web application using all of the open source technologies studied throughout the course. The culmination of this design and development is the delivery of a responsive web application built using Bootstrap, jQuery, and the Spring Framework. To further promote the open source learning objectives, students will also do some of the development on Ubuntu Linux. **Note:** This project will be developed within Collaborative Learning Communities (CLC) as directed by the instructor.

Contents

[Project Overview 2](#_Toc522791758)

[Milestone 1 4](#_Toc522791759)

[Milestone 2 6](#_Toc522791760)

[Milestone 3 9](#_Toc522791761)

[Milestone 4 12](#_Toc522791762)

[Milestone 5 15](#_Toc522791763)

[Milestone 6 17](#_Toc522791764)

[Milestone 7 19](#_Toc522791765)

[Milestone 8 22](#_Toc522791766)

# Project Overview

In this project, students will work in teams of two to design and build a complete enterprise class N-Layer application using all open source technologies learned from this course. The application must adhere to a specified set of high-level functional and technical requirements, and as shown in Figure 1.

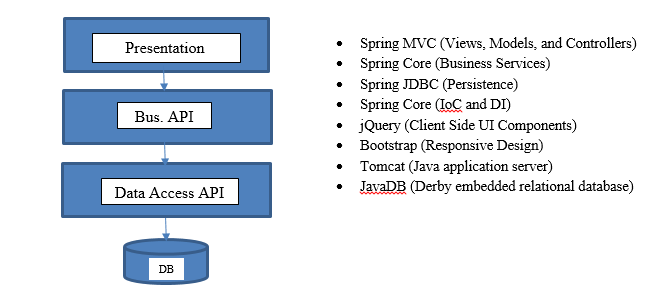


Figure 1

Students will have the freedom to pick the application they want to design and develop. Example applications include, but are not limited to, an eCommerce application, customer management application, order management application, blog site application, DVD/book/music management application, etc.

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

* The application must implement a user registration module and login module.
* The application must be designed using an N-Layer architecture with distinct and separate presentation components, business services, and persistence services.
* The application must adhere to industry best practices, exception handling, and error handling as discussed either in Topic reading, lectures, or provided as peer code review feedback.
* The application presentation must be entirely written using Spring MVC compliant pages. Your application must incorporate at least one user interface component from the jQuery library.
* The application must perform data validation on all form data entry fields.
* The application must support a responsive design using the Bootstrap framework
* The application must not have business logic or business rules implemented in Spring MVC views, models, or controllers.
* The application must implement all CRUD methods on whatever business domain is being addressed (i.e., products, music, blogs, etc.), and include the following:
  + 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"
* The application must use a relational database, such as MySQL or JavaDB (Derby that is distributed as an embedded database with the JDK). JavaDB is recommended.
* The application must use the Spring JDBC to access the database.
* The application must use SpringBeans to implement all business services and persistence services.
* The application must use proper declarative Spring annotations or XML configuration within all components and use DI for all models, controllers, services, and resources required by the application.
* The application must be deployed on Apache Tomcat (version 8.x or later).
* The 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.
* The application classes must be fully documented using JavaDoc format.

**Project Milestones**

The team project is designed and built using an iterative approach and delivered using the milestones outlined below. It should be noted that all milestones will include a design report; except for Milestone 1, application code will be also be delivered to support the appropriate milestone requirements.

All code developed during the project will be maintained in the GitHub Version Control System, which is the VCS that is home to virtually all of the open source projects in the industry. It is expected that the code will be refactored during each iteration based on peer code review or instructor feedback.

Groups will need approval for their project from the instructor prior to moving onto the detailed design and development of the application.

**Project Documentation**

Documentation of all technical decisions and technical designs will be via a formal design report that captures the technical approach, design decisions, UI wireframe designs, Sitemap designs, ER database designs, and project risks/issues. Refer to the "Design Report Template," located within the Course Materials for detailed instructions.

# Milestone 1

**Overview**

In this assignment, students are introduced to the course-long project they will be building. Students will formally identify the application they will be designing and developing, as well as complete some initial upfront planning and high-level design work. The identified application must be formally approved by the instructor. In addition, students will conduct research on open source licenses related to individual involvement or contribution to an existing open source project.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the project requirements within the Project Overview.
2. Then, the team must provide a detailed write up which includes the following:
   1. Description of what domain and products will be managed by the application.
   2. Description of the high level features and functionality that will be supported by the application.
   3. An initial Sitemap illustrating all the logical pages of the application and how they will interact with each other. The team can also submit any user interface wireframe concepts or initial designs. However this is optional.
   4. A high-level breakdown of how the work will be planned, managed, and divided out among the team members. It is recommended to use a formal agile project delivery methodology for this project.
   5. The risks (technical and functional) that need to be managed moving forward with the project.
3. In addition, the team will conduct open source research to include the following:
   1. What are three different open source licenses? How do the licenses differ? How are the licenses the same?
   2. What happens and who owns the code you contribute to an open source project?
   3. Research an open source project from the Apache Foundation and provide a detailed write up for how you can volunteer and contribute to one of their projects.
4. It is expected that the team will arrange a meeting with the instructor if a project cannot be identified.
5. The project must be approved by the instructor.

**Submission**

Submit the following to the learning management system:

1. A word document that provides a complete description of what domain and products will be managed by the application and the high level features and functionality that will be supported by the application.
2. The initial design report (using the "Design Report Template," located within the course materials) with the following sections completed:
3. Cover page with list of tasks completed.
4. Planning documentation: Outlining how your project will be managed.
5. Design documentation:
6. General technical approach
7. Key technical decisions
8. Risks
9. Sitemap diagram
10. User interface diagrams (optional)
11. A Word document that addresses Open Source research questions

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team comprehensively describes the domain and products that will be managed by the application with extensive details and examples. | | 0 pts – 6 pts | 7 pts – 9 pts | 10 pts |  |
| The team comprehensively describes the high-level features and functionality that will be supported by the application with extensive details and examples. | | 0 pts – 6 pts | 7 pts – 9 pts | 10 pts |  |
| The team presents a high-level breakdown of how the work will be planned, managed, and divided out among the team members. The breakdown is well presented and in-depth. | | 0 pts – 6 pts | 7 pts – 9 pts | 10 pts |  |
| The team develops the initial design report. The documentation is well presented and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General technical approach * Key technical decisions * Risks * Sitemap diagram | | 0 pts – 13 pts | 14 pts – 19 pts | 20 pts |  |
| The team addresses in-depth the open source research questions with extensive details and examples. Information and justifications are accurate and appropriate. Subject knowledge is excellent. | | 0 pts – 9 pts | 10 pts – 14 pts | 15 pts |  |
| Prose is largely free of mechanical errors. The writer uses a variety of effective sentence structures, figures of speech, and industry terminology. | | 0 pts – 6 pts | 7 pts – 9 pts | 10 pts |  |
| **TOTAL** | |  |  |  | **/75** |
| **Instructor Feedback** | | | | | |

# Milestone 2

**Overview**

In this assignment, students will design and build web pages to support user registration and login functionality using Spring MVC. Students will apply the MVC design pattern using Spring MVC as well as formally document their design using the "Design Report," provided in the course materials.

**Execution**

Execute this assignment according to the following guidelines:

1. Review requirements within the Project Overview.
2. Main Application Module:
   1. This will be the main application page that is visible when the root URI of the application is accessed.
   2. The main application should provide initial navigation concepts, such as a menu bar, to access the applications core functionality, such as Login and Register.
   3. The main application at this point should have well-defined styles, fonts, colors, and an overall application theme implemented to be used for the remaining milestone deliverables.
   4. The application should leverage Spring MVC for all dynamic page generation.
   5. The application should have a title (and possible a Logo).
3. Registration Module:
   1. A user should be able to register to access the application.
   2. Registration details at a minimum should include First Name, Last Name, Email Address, Phone Number, and Login Credentials (Username and Password).
   3. Initial persistence will use standard JDBC API to save the registration details to a relational database, such as JavaDB (i.e., Derby) or MySQL.
   4. The registration pages should leverage Spring MVC for all dynamic page generation.
   5. A user object model should be designed and implemented.
   6. A database model should be designed and implemented.
   7. All data entered by the user must be validated and provide proper error messages enabling the user to easily correct the data.
   8. It should be noted that this module will be refactored a number of times during the milestone releases. The standard JDBC API will be replaced by Spring JDBC in Milestone 4.
4. Login Module:
   1. Initial Login Module will use the database to authenticate the user.
      1. Initial persistence will use the standard JDBC API to authenticate the user with a relational database.
      2. Security will be re-implemented as part of Milestone 7.
      3. It should be noted that this module will be refactored a number of times during the milestone releases. The standard JDBC API will be replaced by Spring JDBC in Milestone 4.
      4. It is not required to securely protect any pages in the application.
   2. The login pages should leverage Spring MVC for all dynamic page generation.
   3. A principle object model should be designed and implemented.
   4. A database model should be designed and implemented.
   5. Once the user is logged in, the main "product" page should be displayed and the menu system be dynamically updated to reflect the logged in state of the application.
5. Update the initial design report.
6. It is expected that the team will perform peer code reviews on all code.
7. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
8. It is expected that the team will arrange a meeting with the instructor if there are project management issues.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models)

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code and design to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team develops the main application module functionality to correctly provide:   * Initial navigation concepts * Well-defined styles, fonts, colors, and an overall application theme * Leverage of the Spring MVC for all dynamic page generation * Title | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team develops the registration module functionality to correctly provide:   * Registration access to the application to include First Name, Last Name, Email Address, Phone Number, and Login Credentials (Username and Password) * Use standard JDBC API to save the registration details to a relational database, such as JavaDB (i.e. Derby) or MySQL * Leverage of the Spring MVC for all dynamic page generation. * A designed and implemented user object model * A designed and implemented database model * Data validation and provide proper error messages | | 0 pts–13 pts | 14 pts–19 pts | 20 pts |  |
| The team develops the login module functionality to correctly provide:   * Use of the standard JDBC API to authenticate the user with a relational database * Leverage of the Spring MVC for all dynamic page generation * A designed and implemented principle object model * A designed and implemented database model * Proper display to reflect the logged in state of the application | | 0 pts–13 pts | 14 pts–19 pts | 20 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes and methods as well as using inline comments for all code | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models) | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 3

**Overview**

In this assignment, students will continue to design and build their application adding middle layer business services and data services using Spring Core and its IoC container. Students will apply the DI design pattern using Spring Core as well as update their design report.

**Execution**

Execute this assignment according to the following guidelines:

1. Review requirements within the Project Overview.
2. Main Application Module:
   1. The main application at this point should have final styles, fonts, colors, and an overall application theme implemented to be used for the remaining milestone deliverables.
3. Login Module:
   1. Module should be refactored to use SpringBeans, Spring Core, and the DI.
4. Registration Module:
   1. Module should be refactored to use SpringBeans, Spring Core, and the DI.
5. Product Creation Module:
   1. The menu system should allow a user to create a new "product".
   2. A product creation module should be implemented using Spring MVC with all the proper data validation.
   3. A product object model should be designed and implemented.
   4. A database model should be designed and implemented.
   5. A data access object (create method of CRUD) should be implemented using a SpringBean.
   6. The "product" will be saved to the database using standard JDBC API.
6. It is expected that the team will perform peer code reviews on all code.
7. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
8. It is expected that the team will arrange a meeting with the instructor if there are project management issues.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code and design to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team updates the main application module functionality to correctly provide:   * Initial navigation concepts * Well-defined styles, fonts, colors, and an overall application theme * Leverage of the Spring MVC for all dynamic page generation * Title | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team correctly refactors the login module to use SpringBeans, Spring Core, and the DI | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team correctly refactors the registration module to use SpringBeans, Spring Core, and the DI | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team develops the product creation module functionality to correctly provide:   * User to create a new "product" * Proper data validation * A designed and implemented product object model * A designed and implemented database model * A data access object using a SpringBean * Database using standard JDBC API | | 0 pts–13 pts | 14 pts–19 pts | 20 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes, and methods as well as using inline comments for all code. Including peer code reviews. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models) | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 4

**Overview**

In this assignment, students will refactor their data services and replace the standard JDBC APIs with corresponding Spring JDBC APIs. Students will secure their application using Spring Security as well as update their design report.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Main Application Module:
   1. The main application at this point should have final styles, fonts, colors, and an overall application theme implemented to be used for the remaining milestone deliverables.
3. Login Module:
   1. Module should be refactored to use Spring JDBC.
4. Registration Module:
   1. Module should be refactored to use Spring JDBC.
5. Product Creation Module:
   1. Module should be refactored to use Spring JDBC.
6. It is expected that the team will perform peer code reviews on all code.
7. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
8. It is expected that the team will arrange a meeting with the instructor if there are project management issues.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation**:

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code and design to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team updates the main application module functionality to correctly provide:   * Initial navigation concepts * Well defined styles, fonts, colors, and an overall application theme * Leverage of the Spring MVC for all dynamic page generation * Title | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team correctly refactors the login module to use to use Spring JDBC | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team correctly refactors the registration module to use Spring JDBC | | 0 pts–6 pts | 7 pts–9 pts | 10 pts |  |
| The team correctly refactors the product creation module to use Spring JDBC | | 0 pts – 13 pts | 14 pts–19 pts | 20 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes, and methods as well as using inline comments for all code. Including peer code reviews. | | 0 pts – 9 pts | 10 pts – 14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding | | 0 pts – 6 pts | 7 pts – 9 pts | 10 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models) | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 5

**Overview**

In this assignment, students will continue to re-factor their application and leverage the Spring IoC container using DI. Student will build a Maven configuration file and demonstrate through the IDE and command-line interface (CLI) that their application can be built and packaged using Maven.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Product Display, Update, and Delete Modules:
   1. Add new pages using Spring MVC to support the new functionality.
   2. Add new business services using SpringBeans and IoC to support the new functionality.
   3. Add new data services using SpringBeans, IoC, and Spring JDBC to support the new functionality.
3. Create the final version of the Maven POM file and demonstrate how to build and package a Java application using Maven from the IDE and CLI.
4. It is expected that the team will perform peer code reviews on all code.
5. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
6. It is expected that the team will arrange a meeting with the instructor if there are project management issues.
7. Create a test plan and test cases. Execute all test cases. Refer to the "Test Case Template," located within the course materials.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)
* Service API

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code, design, and test plan to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team develops the product display, update, and delete modules functionality to correctly provide:   * New pages using Spring MVC to support the new functionality * New business services using SpringBeans and IoC to support the new functionality * New data services using SpringBeans, IoC, and Spring JDBC to support the new functionality. | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team correctly creates the final version of the Maven POM file and demonstrates fully how to build and package a Java application using Maven from the IDE and CLI. | | 0 pts–13 pts | 14 pts–19 pts | 20 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes, and methods as well as using inline comments for all code. Including peer code reviews. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding. | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team correctly and comprehensively creates a test plan and test cases. All test cases are executed appropriately. | | 0 pts–13 pts | 14 pts–19 pts | 20 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models and Services) * Service API | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 6

**Overview**

In this assignment, students will leverage the jQuery Framework to enhance the user interface of their application.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Application Enhancements:
   1. Identify and leverage one client side user interface component from the jQuery library.
   2. Integrate the new client side interface component into the application that adds a substantive new functionality. The functionality should be approved by the instructor.
3. It is expected that the team will perform peer code reviews on all code.
4. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
5. It is expected that the team will arrange a meeting with the instructor if there are project management issues.
6. Execute all test cases from the test plan created in Milestone 5.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)
* Service API

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code, design, and test plan to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team correctly configures the application enhancements to meet the following requirements:   * Identify and leverage one client side user interface component from the jQuery library * Integrate the new user interface component into the application that provide substantive new functionality. * Your instructor needs to validate and approve the new component and functionality. | | 0 pts-23 pts | 24 pts-34 pts | 35 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes, and methods as well as using inline comments for all code. Including peer code reviews. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding. | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team correctly executes all test cases from the test plan. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models and Services) * Service API | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 7

**Overview**

In this assignment, students will leverage the Bootstrap Framework to make their application responsive such that it could be displayed on other form factors, including mobile phones, tablets, and desktop computers.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Bootstrap Integration:
   1. Integrate the Bootstrap framework into the application framework to support a responsive design such that the application can be run on multiple form factors including but not limited to an iPhone/Android, iPad/Tablet, and desktop browser.
   2. Demonstrate the responsive design either through screenshots from actual devices or by screenshots taken by using Developer extensions available in your browser.
3. It is expected that the team will perform peer code reviews on all code.
4. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
5. It is expected that the team will arrange a meeting with the instructor if there are project management issues.
6. Execute all test cases from the test plan created in Milestone 5.

**Submission**

Submit an updated design report with the following sections completed to the learning management system:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)
* Service API

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the code.
3. Upload all code, design, and test plan to the learning management system.

**Scoring Guide**

*Performance Level Ratings*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Meets Expectations** | Performance consistently met expectations in all essential areas of project construction, at times possibly exceeding expectations, and the quality of work overall was very good. The most critical goals were met. | | | | |
| **Near Expectations** | Performance did not consistently meet expectations. Performance failed to meet expectations in one or more essential areas of project construction and/or recording, one or more of the most critical goals were not met. | | | | |
| **Below Expectations** | Performance was consistently below expectations in most essential areas of project construction and/or recording, reasonable progress toward critical goals was not made. Significant improvement is needed in one or more important areas. | | | | |
| **Criteria** | | **Below Expectations** | **Near Expectations** | **Meets Expectations** | **Earned** |
| The team correctly configures Bootstrap to meet the following requirements:   * Integrate the Bootstrap framework into the application framework to support a responsive design such that the application can be run on multiple form factors including but not limited to an iPhone/Android, iPad/Tablet, and desktop browser. | | 0 pts-23 pts | 24 pts-34 pts | 35 pts |  |
| The team correctly and comprehensively addresses code quality, comments, and readability so that code is:   * Completely functional and responds correctly * Well organized, properly formatted, and understandable * Well documented including all code modules, classes, and methods as well as using inline comments for all code. Including peer code reviews. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team correctly designs the code to be extremely efficient without sacrificing readability and understanding. | | 0 pts–6 pts | 7 pts-9 pts | 10 pts |  |
| The team correctly executes all test cases from the test plan. | | 0 pts–9 pts | 10 pts–14 pts | 15 pts |  |
| The team updates the initial design report. The documentation is well presented, prose is largely free of mechanical errors, and includes all required elements:   * Cover Page (with list of tasks completed) * Planning Documentation (outlining how the project will be managed)   Design Documentation to include:   * General Technical Approach * Key Technical Decisions * Install or Configuration Instructions * Known Issues * Risks * Sitemap Diagram * User Interface Diagrams * ER Diagram * DDL Scripts * Class Diagrams (for all Object Models and Services) * Service API | | 0 pts – 2 pts | 3 pts – 4 pts | 5 pts |  |
| **TOTAL** | |  |  |  | **/80** |
| **Instructor Feedback** | | | | | |

# Milestone 8

**Overview**

In this assignment, students will apply all of the open source technologies covered in the course. Students will re-factor/clean up their application code, as well as finalize their design and formally create JavaDoc for the entire application. In addition, students will demo their application emulating a formal design review that might be held in the industry.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Final Preparation:
   1. Create the JavaDoc for the application.
   2. Refactor any code as necessary.
3. Presentation:
   1. Prepare a 15- to 20-minute presentation on the design of the application.
   2. Give a demo of the application to the class (for ground) or create a screencast of the application (for online).
4. It is expected that the team will perform peer code reviews on all code.
5. It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.
6. It is expected that the team will arrange a meeting with the instructor if there are project management issues.
7. Execute all test cases from the test plan created in Milestone 5.

**Submission**

Submit an updated design report with the following sections:

**Cover Page** (with list of tasks completed)

**Planning Documentation** (task list/schedule or Scrum artifacts)

**Design Documentation:**

* General Technical Approach
* Key Technical Decisions
* Install or Configuration Instructions
* Known Issues
* Risks
* Sitemap Diagram
* User Interface Diagrams
* ER Diagram
* DDL Scripts
* Class Diagrams (for all Object Models and Services)
* Service API

**Make sure to:**

1. Upload and manage all code in GitHub.
2. Zip up the generated JavaDoc pages.
3. Zip up the code.
4. Upload all code, design, and test plan to the learning management system.
5. Present design and demo.

The Final Project uses a scoring guide. Please review the scoring guide prior to beginning the assignment to become familiar with the expectations for successful completion.

The Project Presentation uses a rubric. Please review the rubric prior to beginning the assignment to become familiar with the expectations for successful completion.