**CLC Project Guide**

Contents

[Project Overview 2](#_Toc519149619)

[Milestone 1 Project 5](#_Toc519149620)

[Milestone 2 Project 11](#_Toc519149621)

[Milestone 3 Project 14](#_Toc519149622)

[Milestone 4 Project 16](#_Toc519149623)

[Milestone 5 Project 18](#_Toc519149624)

[Milestone 6 Project 20](#_Toc519149625)

# Project Overview

In this course, students will design and build a complete enterprise class N-Layer application using Enterprise Java technologies, that meets a certain set of functional and technical requirements, as shown in the figure below. Groups will have the freedom to pick the application they want to design and develop. Example applications could include, but are not limited to:

* eCommerce application
* Customer Management application
* Order Management application
* Blog Site application
* DVD/Book/Music Management application
* JavaServer Faces (Views, Models, and Controllers)
* JavaServer Faces Controls and Data Table
* JavaServer Faces Facelets (Layouts)
* EJB 3 Business and Persistence Components
* JDBC and JPA
* CDI dependency injection
* JAX-RS REST API
* JBoss EAP (Java EE application server)
* JavaDB (Derby embedded relational database)

Bus. API

REST API

Presentation

Data Access API

DB

The design and code will 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 reading, lectures, or provided as peer code review feedback.
* Your application presentation must be entirely written using JSF xhtml pages using managed beans and controllers.
* Your application must perform data validation on all form data entry fields.
* Your application presentation must use a JSF Data Grid control somewhere in the design to create a tabular report.
* Your application presentation must use Facelets (i.e., layouts) to provide reusable headers, footers, and other user interface elements to support your user interface design.
* Your application must not have business logic or business rules implemented in JSF 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 JavaDB (Derby that is distributed as an embedded database with the JDK). JavaDB is recommended.
* Your application must use EJBs to implement all business services and persistence services.
* Your application must use proper declarative annotations within all components and CDI for all models, controllers, services, and resources required by the application.
* Your application must use the JPA API to access the database.
* Your application must provide at least one REST Service that exposes at least two CRUD REST API methods (using either GET or POST) and returns JSON as a data format.
* Your application must be deployed on JBoss EAP (version 7.x or later).
* Your application must use the JBoss ApplicationRealm for user authentication.
* 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 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. Groups will work in teams of no more than two unless there is an odd number of students in which case a team of three will be acceptable.

All code developed during the project will be maintained in the Bitbucket Version Control System. Application code will 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.

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 “Java Application Programming Project Report Template," located within the Course Materials for detailed instructions.

**Project Outline**

**Milestone 1**

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

**Milestone 2**

* Main Application Module (using JSF design, data validation, and Facelets.
* Registration Module (using JSF design, data validation, and Facelets.
  + Initial persistence will use the JDBC API.
* Login Module v1 (using JSF design, data validation, and Facelets)
  + Initial Login Module will use the database to authenticate the user.
  + Security will be re-implemented as part of Milestone 6.
  + Initial persistence will use the JDBC API.
* Design Report.

**Milestone 3**

* Product Creation Module (using JSF design, EJBs, CDI, and JPA).
  + Registration Module and Login Module will be refactored to use EJBs.
  + Registration Module and Login Module will be refactored to use JPA.
* Updated Design Report

**Milestone 4**

* Product Display Module (using JSF design, EJBs, CDI, and JPA)
* Product Update/Delete Modules (using JSF design, EJBs, CDI, and JPA)
* Product Report Module (using JSF Data Table)
* Updated Design Report

**Milestone 5**

* REST API Module (using JAX-RS)
* Updated Design Report

**Milestone 6**

* Login Module v2 (using FORM based authentication)
  + Security will use JBoss ApplicationRealm for user authentication.
  + All pages, but the Login Module and Registration Module will be protected.
  + Registration Module will be re-factored to remove unused credential fields.
* JavaDoc and Refactoring
* Final Application Code
* Final Design Report

# Milestone 1 Project

In this project, 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 (project proposal, draft sitemap, and draft division of work across the team). The identified application must be formally approved by the instructor.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the project requirements within the Project Overview.
2. Watch the video "Project Preview," located within the Course Materials to get an idea of the scope of the project. The application should have the following pages at a minimum:
   1. Login
   2. Register
   3. Product list page
   4. Create new product
   5. Edit product
3. Features that should be included are:
   1. List all products
   2. Add new product
   3. Edit existing product
   4. Delete an item
   5. Search for an item
4. **Note:** Features such as shopping carts, customer reviews, and order forms are nice options, but are beyond the scope of the application that will be demonstrated during the tutorials.
5. The team must provide a detailed write-up describing what domain and products will be managed by the application. Some examples of domains and products:
   1. Car store: Each product will be a car with properties of year, model name, options package, color, motor size, turbo charged, price, consumer reports grade, 0-60 acceleration, MPG city, MPG highway, fuel type, etc.
   2. Movie rental: Each product will be a movie with title, year, genre, leading actor, studio, director, length, rental price, cost of production, estimated box-office revenue, etc.
   3. Dating site: Each "product" will be a person with properties of sex, age, nationality, favorite hobby, political description, height, weight, "wow" factor value, introvert-extrovert scale rating, etc.
   4. Veterinarian clinic: Each product will be an appointment with properties of date, patient name, primary procedure, owner name, time, cost, etc.
   5. Travel plans: Each product will be a trip with properties of data, trip title, destination, demarcation point, cost, duration, tour guide, strenuous rating, etc.
   6. My bucket list: Each "product" will be an event that you plan to accomplish with properties such as cost, time required, priority ranking, estimated completion date, injury risk rating, etc.
   7. Bible heroes: Each "product" will be a biblical character that you evaluate with properties such as name, time period, principal book of the Bible, primary partner, best-known failure, famous quote, times mentioned in the bible, likability rating, bravery rating, etc.
6. The team must provide an initial Sitemap illustrating all the logical pages of the application and how they will interact with each other. The team must also submit any user interface wireframe concepts or initial designs. This is mandatory.
7. The team must provide 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.
8. The team will document the risks (technical and functional) that need to be managed moving forward with the project. Some risks for developing a project might include:

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Details | Strategy for Risk **Avoidance** | Strategy for **Mitigation** |
| My project design is too ambitious. | Features such as shopping carts, customer comments, appointment scheduling, date matching, accounting and inventory systems will take too long to develop in one semester. | Study the required elements in the project needed to finish the course. | Keep scope limited to the design presented in the tutorials. |
| Computer failure | Lose data, code corrupted, hardware failure, computer too slow for required applications (JBOSS) | Purchase new computer | Backup code daily to USB drive |
| Personal emergency | Sickness. Vacations. Work schedule is difficult. | Don’t take this class until I can devote my full attention. | Work ahead when I have free time. Keep weekends open for homework. |
| Learning curve too steep | I will spend too much time re-learning the basics of Java programming that was taught in previous classes. | Don’t take this class until you are familiar with Java basics.  Review previous courses that are prerequisites. | Take a Java or HTML&CSS intro course on YouTube.com or Udemy.com to catch up quickly. |
| Internet service fails | Cannot submit work at 11:59 p.m. on Sunday due to a technical problem. | Upgrade to a reliable service. | Work at Starbucks. Use phone hotspot. |
| Partner doesn't do his/her share | Project milestones are missed due to late or poor quality of work from "friend." | Ask to see your partner’s previous work or grades before agreeing to work together.  Have a frank conversation about your expectations and work habits. | Do all the work yourself to avoid incomplete results.  Request the teacher to assign a new partner or work alone. |
| Not sure how to accomplish a specific coding task | Questions to professor don’t get answered immediately. No response after 10:00 p.m. | Plan on finishing work one day before the deadline.  Get explanations from online documentation and tutorials. | Submit questions with fully documented problems: copy of source code and run-time results – errors or unexpected output. |

1. It is expected that the team will meet with the instructor if a project cannot be identified. The project must be approved by the instructor.

**Deliverables:**

Submit the following to the learning management system:

A Word document that provides a complete description of what domain and products will be. Describe high-level features and functionality that will be supported by the application.

An Initial Design Report (using the “Java Application Programming Project Report Template,” located in the course materials) with the following sections completed (refer to the screen shot below):

1. Cover page with list of tasks completed (topic, date, revision, team, weekly team status summary, GIT URL, and peer review information)
2. Planning documentation: Outlining how your project will be managed.
3. Design documentation:
   * 1. General Technical Approach
     2. Key Technical Decisions
     3. Risks
     4. Sitemap Diagram
     5. User Interface Diagrams (mandatory)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Java III Application Programming**  **Project Status and Design Report**   |  |  |  | | --- | --- | --- | | **Topic:** | *This should be the Topic Number and Topic Name.* | | | **Date:** | *This should be the date you completed the Report.* | | | **Revision:** | *This should be the revision, starting at 1.0, for your Report.* | | | **Team:** | 1. *This should list the members of your team.* | | |  | | |  | | |  | | | **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | |  |  |  |  | |  |  |  |  | |  |  |  |  | | | | **GIT URL:** | *The GIT GitHub URL that I can use to clone your code and design artifacts.* | | | **Peer Review:** | *Y/N* | We acknowledge that our team has reviewed this Report and we agree to the approach we are all taking. |   **Design Documentation**  **General Technical Approach:**  *You should, in words, describe your approach and design here. You should also summarize any meeting notes, brain storming sessions, etc. that you want to retain thru the design of your project.*  **Key Technical Design Decisions:**  *Any final technical design decisions, such as framework decisions etc., should be documented here. This should list the technology/framework, its purpose in the design, and why it was chosen.*  **Known Issues:**  *Any anomalies or known issues in the code or functionality should be documented here.*  **Risks:**  *Any risks, unknowns, or general project elements that should be tracked for risk management should be documented here.*  **ER Diagram:**  *Image file of your ER database diagram. (example shown)*    **DDL Scripts:**  *This should contain a link to GitHub where the DDL script can be downloaded from. A DDL script is a SQL "export" file. See* [*https://www.youtube.com/watch?v=aAoYaZzWRgw*](https://www.youtube.com/watch?v=aAoYaZzWRgw) *for a tutorial. Or look here* [*https://www.youtube.com/watch?v=TDsH7KZ244o*](https://www.youtube.com/watch?v=TDsH7KZ244o) *for the 35 second version.*  **Sitemap Diagram:**  *Image file of your Sitemap diagram. Here is an example of a site map.*  mage result for â¢ Sitemap Diagram  **User Interface Diagrams:**  *You should insert any wireframe drawings or white board concepts that were developed to support your application. Here is an example.* mage result for wireframe for login page  **Class Diagrams:**  *You should insert any class diagrams here. Your class diagrams should be drawn correctly with:*   1. *Three appropriate class compartments (name, properties and methods)* 2. *+ and – minus to indicate accessibility of each item.* 3. *The data types for the state/properties* 4. *The data types of method arguments and return types.*   *If you have no supporting documentation, please explain the rational why you are able to leave this section as N/A.*  *Here are some examples:*    **Service API Design:**  *This section should fully document any 3rd Party Service Interface API 's being consumed or application specific Service API 's being published, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a 3rd party developer to integrate with the service and API. This requirement is not likely going to be needed until milestone #5, Java Rest Services (JSON data). There are no required 3rd party tools used in the tutorials.*  **Security Design:**  *This section should outline the design for how authentication and authorization was supported. This section should also contain all of the roles and privileges that are supported by the design.*  **Other Documentation:**  *You should insert any additional drawings, storyboards, white board pictures, project schedules, tasks lists, etc. that support your approach, design, and project. If you have no supporting documentation, please explain the rational why you are able to leave this section as N/A.* |

# Milestone 2 Project

In this project, students will design and build web pages to support user registration and login functionality.

**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 overall application theme implemented that will be used for the remaining milestone deliverables.
   4. The application should leverage Facelets to support reusable page components (i.e., headers, footers, menu bars, etc.) and for Page Layouts.
   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, E-mail Address, Phone Number, and Login Credentials (Username and Password).
   3. Initial persistence will use a file or Java hashmap to save the Registration details. The file or Java hashmap will be replaced in Milestone 4 with the JDBC API and a database.
   4. The Registration pages should leverage common Facelets.
   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.
4. Login Module:
   1. Initial Login Module will use the database to authenticate the user.
      1. Initially the credential data will just be saved to a User class to authenticate the user. Later in Milestone 4 the authentication will be done using the JDBC API and a database.
      2. Security will be re-implemented as part of Milestone 6.
      3. It is not required to securely protect any pages in the application.
   2. The Login pages should leverage common Facelets.
   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.

It is expected that the team will perform peer code reviews on all code.

It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.

It is expected that the team will meet with the instructor if there are project management issues.

**Deliverables:**

Submit the following to the learning management system:

An updated “Design Report,” started in Project 1, with the following sections completed:

1. Cover page with list of tasks completed.
2. Planning documentation: task list/schedule or Scrum artifacts.
3. Design documentation:
   1. General Technical Approach
   2. Key Technical Decisions
   3. Install or Configuration Instructions
   4. Known Issues
   5. Risks
   6. Sitemap Diagram
   7. User Interface Diagrams
   8. ER Diagram
   9. DDL Scripts
   10. Class Diagrams (for all Object Models)

All code in GitHub.

All code as a zip file to LoudCloud.

**Milestone 2 Project Scoring Guide**

# Milestone 3 Project

In this assignment, students will continue to design and build their application adding a Product Creation Module.

**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 overall application theme implemented that will be used for the remaining milestone deliverables.
3. Login Module:
   1. Module should be refactored to use EJBs.
4. Registration Module:
   1. Module should be refactored to use EJBs.
5. Product Creation Module:
   1. The menu system should allow a user to create a new "product."
   2. A Product Creation Module implemented using the appropriate JSF controls and data validation.
   3. A Product Object Model should be designed and implemented.
   4. A Database Model should be designed and implemented.

It is expected that the team will perform peer code reviews on all code.

It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.

It is expected that the team will meet with the instructor if there are project management issues.

**Deliverables:**

Submit the following to the learning management system:

An Updated “Design Report” with the following sections completed:

1. Cover page with list of tasks completed.
2. Planning documentation: task list/schedule or Scrum artifacts.
3. Design documentation:
4. General Technical Approach
5. Key Technical Decisions
6. Install or Configuration Instructions
7. Known Issues
8. Risks
9. Sitemap Diagram
10. User Interface Diagrams
11. ER Diagram
12. DDL Scripts
13. Class Diagrams (for all Object Models and Services)

All code in GitHub.

All code as a zip file

# Milestone 4 Project

In this assignment, students will add a Project Report Module and a Product Display/Update/Delete Module.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Report/Display/Update/Delete Product Modules:
   1. The menu system should allow a user to display all "products."
   2. Product Report Module using a styled JSF Data Table. The JSF Data Table style must match the theme of the application.
   3. A Product Display Module implemented by selecting (hyperlink or button) a product from the Product Report and by using the proper JSF controls.
   4. A Product Delete Module implemented by selecting (hyperlink or button) a product from the Product Report. The user should have the option to cancel the delete operation.
   5. A Product Update Module implemented by selecting (hyperlink or button) a product from the Product Report and using the appropriate JSF controls and data validation.
   6. A Data Access Object (find by id, find all, update, and delete methods of CRUD) implemented using an EJB and JDBC.

It is expected that the team will perform peer code reviews on all code.

It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.

It is expected that the team will meet with the instructor if there are project management issues.

**Deliverables:**

Submit the following to the learning management system:

An updated “Design Report” with the following sections completed:

1. Cover page with list of tasks completed.
2. Planning documentation: task list/schedule or Scrum artifacts.
3. Design documentation:
4. General Technical Approach
5. Key Technical Decisions
6. Install or Configuration Instructions
7. Known Issues
8. Risks
9. Sitemap Diagram
10. User Interface Diagrams
11. ER Diagram
12. DDL Scripts
13. Class Diagrams (for all Object Models and Services)

All code in GitHub

All code as a zip file

# Milestone 5 Project

In this assignment, students will add a REST Service.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. REST API Module:
3. An API with proper URI and verbs will be designed as a REST Service.
4. The REST Service can be anonymous and implemented using JAX-RS.
5. The REST Service that exposes at least 2 CRUD REST API methods (using either GET or POST) and returns JSON as a data format.
6. The REST Service will leverage existing Business Services.
7. The REST Service will return proper error messages if invalid data parameters are passed to the API.
8. The design will be documented such that any 3rd party application developer could easily read the design and integrate the API into their application.

It is expected that the team will perform peer code reviews on all code.

It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.

It is expected that the team will meet with the instructor if there are project management issues.

It is expected that the team will create a Test Plan and Test Cases. Execute all Test Cases. Use the “Test Case Template,” found in the course materials for building test plan and test cases on select projects within the course.

**Deliverables:**

Submit the following to the learning management system:

An updated “Design Report” with the following sections completed:

1. Cover page with list of tasks completed.
2. Planning documentation: task list/schedule or Scrum artifacts.
3. Design documentation:
4. General Technical Approach
5. Key Technical Decisions
6. Install or Configuration Instructions
7. Known Issues
8. Risks
9. Sitemap Diagram
10. User Interface Diagrams
11. ER Diagram
12. DDL Scripts
13. Class Diagrams (for all Object Models and Services)
14. Service API

All code in Bitbucket.

All code (as a zip file), design, and test plan.

# Milestone 6 Project

In this assignment, students will secure the application and documentation for the application.

**Execution**

Execute this assignment according to the following guidelines:

1. Review the requirements within the Project Overview.
2. Login Module v2:
3. Login Form will be re-factored to support FORM based authentication.
4. Security will standard enterprise Java configuration.
5. Security will use JBoss ApplicationRealm for user authentication.
6. All pages but the Login Module and Registration Module will be protected.
7. Navigation to protected pages where the user is not authenticated will redirect the user to the Login Module.
8. Registration Module will be re-factored to remove unused credential fields
9. Final Preparation:
10. Create the JavaDoc for the application.
11. Refactor any code as necessary.

It is expected that the team will perform peer code reviews on all code.

It is expected that the team will fully document all code modules, classes, methods, and use inline comments for all code.

It is expected that the team will meet with the instructor if there are project management issues.

It is expected that the team will execute all Test Cases from the Test Plan created in Milestone 6.

**Deliverables:**

Submit the following to the learning management system:

An updated “Design Report” with the following sections completed:

1. Cover page with list of tasks completed.
2. Planning documentation: task list/schedule or Scrum artifacts.
3. Design documentation:
4. General Technical Approach
5. Key Technical Decisions
6. Install or Configuration Instructions
7. Known Issues
8. Risks
9. Sitemap Diagram
10. User Interface Diagrams
11. ER Diagram
12. DDL Scripts
13. Class Diagrams (for all Object Models and Services)
14. Service API

All code in Bitbucket.

Zipped up generated JavaDoc pages

Zipped up code.

All code, design, and test plan