**CLC Milestone Project -1**

**Ryan Coon**

**Justin Gewecke**

**Joshua Ruiz**

**CST-339**

**Dr. Brandon Bass**

**January 23, 2022**

**CST-339 Programming in Java III**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | CLC Project Assignment – Milestone 1 | |
| **Date:** | *January 22, 2022* | |
| **Revision:** | v1.0 | |
| **Team:** | 1. Ryan Coon | |
| 1. Justin Gewecke | |
| 1. Joshua Ruiz | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *As a part of the team, I want to make sure we hit on everything expected while maintaining a clean and responsive e commerce page.* | *Ryan Coon* | *2* | *2* | | *UI Diagram & Sitemap initial creation* | *Justin Gewecke* | *1.5* | *0* | | *As part of the project team, I want to create an application overview that hits upon all of the aspects for this project.* | *Joshua Ruiz* | *1* | *1* | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | https://github.com/rcoon1/CST-339-CLC-Milestone-Project | |
| **Screencast URL:** | n/a | |
| **Peer Review:** | Y | We acknowledge that our team has reviewed this Report and we agree to the approach we are all taking. |

**Planning Documentation**

**Initial Planning:**

The project our group will build is an online vinyl record store, similar to Amoeba Music (https://www.amoeba.com) with the added ability of purchasing the records users have selected.

The website will include pages for each of these core details/features: home/root, about us, registration, login, profile, record browsing, favorites managing, and checkout.

Users will sign up for an account using a basic email/password authentication scheme, after which they will be allowed to use the full site’s functionality.

Users may browse for records on a dedicated page (/records), then either add a record to a Favorites tab (just a collection of records they like and want to buy in the future) and a shopping cart.

Users view their profile information and update their personal data, even deleting their account if they’d like.

Users can have as many records/items in their favorites for their account as they’d like, then add them to their cart whenever they’d like. We’ll have to design our data scheme to account for this kind of scalable nature.

We plan on assigning each part to whomever feels confident in their work to finish the assignment in the time given. If not, we will work as a team and do it via video call to ensure that it gets completed.

**Retrospective Results:**

*The following table should be completed after each Retrospective on things that went well (keep doing).*

|  |
| --- |
| **What Went Well** |
| Using Discord chat as a team worked well to foster speedy communication for quick changes that would otherwise be impossible through the Halo platform forums. |
| Using the Google office suite (Docs and Sheets) helped the group collaborate efficiently, even with limited time to communicate among one another. |
|  |

*The following table should be completed after each Retrospective on things that didn’t go well (stop doing) and what would be done differently next time with an action plan to improve (try doing and continuous improvement).*

|  |  |  |
| --- | --- | --- |
| **What Did Not Go Well** | **Action Plan** | **Due Date** |
| Beginning collaboration partway through the milestone week | We need to begin designing and delegating tasks/user stories from the beginning of the week | **1/24/22** |
|  |  |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

*Step-by-step instructions for setting up your database, and configuring and deploying/installing your application. This section should also include detailed instructions for what configuration files are required by your application, what configuration settings need to be adjusted for various runtime (development or production) environments, and where the files need to be deployed to. This section should also contain detailed instructions for how to clone your application source code from GIT and deploy the application to an externally hosted site.*

**General Technical Approach:**

This program will be written using Spring MVC and utilizing Java as the base language. The Spring MVC framework will provide our visual page elements, styled according to the Bootstrap CSS framework with help from the jQuery JavaScript library. For the database, we will be utilizing MySQL to manage information on the website. As of now, that is as far as our technical approach has been sorted out.

**Key Technical Design Decisions:**

The application must be able to run through an online provider, such as AWS, Heroku, Azure, etc. By requirements for this project, we are committed to designing our application in a layered architecture to support development purely using the Spring MVC framework.

**Known Issues:**

There are no known issues at this point in time.

**Risks:**

The project gets out-of-scope. Being able to finish the project by the end of the class is a must. One of our team members is newer to Spring and may need additional help for each sprint.

**ER Diagram:**

No Database tables have been designed yet, and thus there aren’t any ER diagrams.

**DDL Scripts:**

DDL Scripts do not exist for this project yet.

**Sitemap Diagram:**

**Diagram

Description automatically generated**

**User Interface Diagrams:**

<https://imgur.com/a/IgQkGNw>

**Class Diagrams:**

*You should insert any class diagrams here. Your class diagrams should be drawn correctly with the 3 appropriate class compartments, + and – minus to indicate accessibility, and also the data types for the state/properties as well as method arguments and return types. If you have no supporting documentation, please explain the rationale for why you are able to leave this section as N/A.*

**Service API Design:**

*This section should fully document any service API’s (like REST API’s) that are 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 third party developer to integrate with the service and API. The design can also be captured with tools such as Swagger.*

**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, whiteboard pictures, project schedules, tasks lists, etc. that support your approach, design, and project. If you have no supporting documentation, please explain the rationale for why you are able to leave this section as N/A.*

User Interface diagram:



















