**CST-339 Programming in Java III**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *TOPIC 8: ENTERPRISE WEB APPLICATIONS* | |
| **Date:** | *07/24/2024* | |
| **Revision:** | *This should be the revision, starting at 1.0, for your Report* | |
| **Team:** | 1. *Easten Elefson* | |
| 1. *Chance Jones* | |
| 1. *Nathan Ortiz* | |
| 1. *Kamren Parker* | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *Took the liberty of looking over the code before other teammates ensuring that everything was looking correct while also adding in the needed add-ons to the.* | *Chance J.* |  |  | | *Was the final eye on the code before everything was turned in. Everything that was put into our repository thus far was tested by him to ensure that it was all running correctly and that there were no errors that prevented the code from working.* | *Easten E.* |  |  | | *Worked on the final design document for the final milestone. Did a final check over everything to make sure that we were not missing any key components within the code and within the actual files that we needed including actual needed files such as controllers and functional/explanatory files such as JavaDoc files.* | *Kamren P.* |  |  | | *Handled the final submission of everything including the screencast of all of the final code that was added possible after Kamren had did his final run through.* | *Nathan O.* |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | *https://github.com/HBadg3r/CST339* | |
| **Screencast URL:** | *The URL that can be used to access your screencast demonstration video for the respective assignment.* | |
| **Peer Review:** | *Y/N* | We acknowledge that our team has reviewed this Report and we agree to the approach we are all taking. |

**Planning Documentation**

**Initial Planning:**

*Initial planning consisted of divided the tasks that we needed to complete the final assignments of the class. This not only included milestone 8 in which we all contributed to the wrap up of the milestone with everyone having their part in it to ensure a clean wrap up to the project as a whole.*

* *Easten: In charge of being the final eye on all of the code what was submitted*
* *Chance: Was the first eye when it came to the final preparations adding the final files that Easten then checked.*
* *Kamren: First eye when It came to ensuring that everything needed was there before turn in to prevent any hiccups in the future. Put eyes on the things that we needed to complete before Nathan was to turn everything in*
* *Nathan: In charge of turning in the final milestone including the design document and screencast.*

**Retrospective Results:**

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

|  |
| --- |
| **What Went Well** |
| Testing the code and ensuring that everything works and there are no major errors. |
| Working together to make sure that everything runs well together |
| **\*There were no errors as all were solved in previous milestones** |

*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** |
| **Oveall consistent problems with spring** | **Constant troubleshooting most problems are solved shortly** | **7.28.24** |
|  |  |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

***Setting up the Database:***

●*Install MySQL or your preferred database.*

●*Create a database named bar\_inventory.*

●*Update application.properties with your database connection details.*

***Configuring and Deploying the Application:***

*Clone the project repository from GIT:*

*git clone [Insert GIT URL]*

*cd [Project Directory]*

*Build the project using Maven:*

*mvn clean install*

*Run the application:*

*mvn spring-boot:run*

***Configuration Files:***

●*application.properties for database and server configurations.*

●*Ensure to set the appropriate configurations for development or production environments*

**General Technical Approach:**

*Our approach focuses on leveraging the Spring Boot framework to build an N-layer application that is scalable and user-friendly. We utilized Agile methodologies to ensure continuous improvement and adaptability. Key meetings included Sprint Planning, Daily Stand-ups, and Sprint Reviews to keep track of progress and address any issues promptly. We also utilize Spring Security for the authentication side of things ensuring that there are security features in place to protect secret information.*

**Key Technical Design Decisions:**

***Framework:*** *Spring Boot for the backend to leverage its powerful features and ease of integration.*

***Templating Engine:*** *Thymeleaf for dynamic page generation and seamless integration with Spring.*

***Database:*** *MySQL for reliable and scalable data management.*

***CSS Framework:*** *Bootstrap for responsive and modern UI design.*

**Known Issues:**

*The know issues with the entire application as a whole was working with Spring Boot as it presented a bunch of problems that we then had to troubleshoot as a team which included reworking things, adding in new code and trying to reconstruct things so that it works with the Spring Boot applications including Spring security.*

**Risks:**

*There is always a chance that the code could not work with the updates with the Spring Application as this is where the problems arose and the risks always come back when it comes to changes within the software.*

**ER Diagram:**

●*User (id, firstName, lastName, email, phone, username, password).*

**DDL Scripts:**

*‘Users’ table:*

*CREATE TABLE Users (*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*firstName VARCHAR(50) NOT NULL,*

*lastName VARCHAR(50) NOT NULL,*

*email VARCHAR(100) NOT NULL UNIQUE,*

*phone VARCHAR(20),*

*username VARCHAR(50) NOT NULL UNIQUE,*

*password VARCHAR(255) NOT NULL*

*);*

*‘Products’ table:*

*CREATE TABLE Products (*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*name VARCHAR(100) NOT NULL,*

*description TEXT,*

*price DECIMAL(10, 2) NOT NULL,*

*quantity INT NOT NULL*

*);*

*‘Orders’ table:*

*CREATE TABLE Orders (*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*user\_id INT,*

*order\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,*

*total DECIMAL(10, 2) NOT NULL,*

*FOREIGN KEY (user\_id) REFERENCES Users(id)*

*);*

*‘OrderDetails’ table:*

*CREATE TABLE OrderDetails (*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*order\_id INT,*

*product\_id INT,*

*quantity INT NOT NULL,*

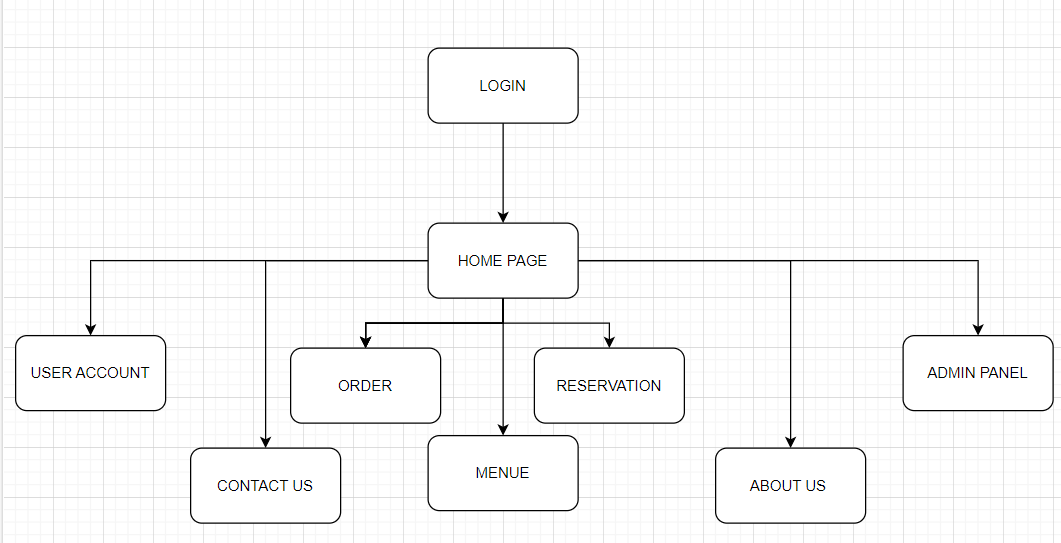
*price DECIMAL(10, 2) NOT NULL,*

*FOREIGN KEY (order\_id) REFERENCES Orders(id),*

*FOREIGN KEY (product\_id) REFERENCES Products(id)*

*);*

**Sitemap Diagram:**

**

●*Main Page -> [Login, Register]*

●*Login -> Product Page*

●*Register -> Product Page*

**User Interface Diagrams:**

●*Main Page, Login Page, Registration Page, Product Page*

**Class Diagrams:**

●***Controllers:*** *HelloWorldController, HomeController, LoginController*

●***Models:*** *User, LoginModel*

**Service API Design:**

*Our REST API is designed to handle user registration, login, and inventory management. Below is an example of the API endpoint documentation:*

***Endpoint: ‘/api/register’***

*Method: POST*

*Parameters:*

*{*

*"firstName": "string",*

*"lastName": "string",*

*"email": "string",*

*"phoneNumber": "string",*

*"username": "string",*

*"password": "string"*

*}*

*Response:*

*{*

*"status": "success",*

*"message": "User registered successfully."*

*}*

***Endpoint: ‘/api/login’***

*Method: POST*

*Parameters:*

*{*

*"username": "string",*

*"password": "string"*

*}*

*Response:*

*{*

*"status": "success",*

*"message": "User logged in successfully."*

*}*

**Security Design:**

The security aspects of the projects were implemented using Spring security which introduced the authorization and authentication of the log in information based upon the stored credentials within the database.

**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.*

***Project Proposal***

●*Bar food and drinks are an iconic staple of American culture and deserve proper inventory management software to keep the food and drinks flowing. Therefore, we are proposing a project that will allow for smoother inventory management and easier ordering capabilities. The inventory will consist of fried foods, as well as mixers and alcoholic beverages. The application will track the level of each item in stock and alert the ordering manager when it is time to restock. The application can also track the items with the highest sell rates, generating reports on what’s most popular amongst consumers. The application will protect the company’s information by requiring login credentials to access the inventory and make purchases. The objectives are to design an N-layer application that is scalable and utilizes a user-friendly interface whilst executing real-time tracking of stock.*