**Inventory Management System (IMS) Documentation**

## Milestones:

**February 29, 2024:**

**Functionalities as of now:**

* Add product works as intended. The user can add a single product, with just a single size, and a single color and the quantity as well.
* Product list works as intended.

**What needs work on:**

* Work on the Sold, Refund, Edit and Delete button.
* When editing, a user should be able to modify, and add another new variation of the same product.
* When deleting, it will delete the whole product from the list.

**Sold Button:**

* Subtracts the number of sold stock of a product variation of the specific product, size, and color.
* Updates the total # of stocks.
* Creates a variable called totalRevenue which is a multiplication of the Sold Products to the Selling Price and adds it to the profit column in the sales table.

**March 17, 2024:**

**Functionalities as of now:**

* Add product works as intended. The user can add a single product, with just a single size, and a single color and the quantity as well.
* Product list works as intended.
* Sold button works as intended and added to the financial report.
* Delete button works when deleting a product.
* Simple User Login/Logout/Signup works.

**What needs work on:**

* Refund button:
  + Need to create a “refunds” table to keep track of the refunds. Columns would be:
    - **Id**
    - **transaction\_date**
    - **refund\_date**
    - **refunded\_quantity**
    - **refund\_reason**
  + Then the business logic of updating the profit/loss statement for the financial statement should be done.
  + Return the quantity of the refunded product to the original quantity of the original product.
  + Create a “Refunds” view to display all the refunds.
* Edit button:
  + Should be able to add a product variation with this button.
  + Should be able to edit product information correctly and update the database.
* Homepage view:
  + Edit the information provided and the overall UI of the homepage.

**March 29, 2024:**

**Functionalities as of now:**

* Add product works as intended. The user can add a single product, with just a single size, and a single color and the quantity as well.
* Product list works as intended.
* Sold button works as intended and added to the financial report.
* Delete button works when deleting a product.
* Simple User Login/Logout/Signup works.

**What needs work on:**

* Refund button: (DONE April 7, 2024)
  + Refunds the item and puts back the stocks into the inventory, as well as report negative values in the sales report.
* Edit Product Button: (DONE April 9, 2024)
  + Should be able to add a product variation with this button.
  + Should be able to edit product information correctly and update the database.
* Homepage view: (DONE March 29, 2024)
  + Edit the information provided and the overall UI of the homepage.
* Edit Profile functionality: (DONE March 30, 2024)
  + Be able to edit properly the user details.
  + Be able to upload user profile picture.
* PDF and Excel download: (DONE)
  + Be able to download the reports via PDF or Excel file.
  + Export to PDF. (DONE April 9, 2024)
  + Export to Excel. (DONE April 10, 2024)
* Add log4j logging (PENDING)
* Add @ResponseStatus for each HTTP request: (DONE April 12, 2024)
* Do testing for each method (PENDING)
* Do REST controllers for each method. (DONE April 12, 2024)
* Users should have their own inventories and sales report now. (DONE March 31, 2024)
* Work on hashing the password for security or look up how password is handled in enterprise applications. (DONE April 12, 2024)
* Add product should also save the user’s id who saved it. (DONE April 4, 2024)

**April 7, 2024:**

* What I’ve worked on:
  + I’ve added some modifications in the products view and the sales report view.
  + The refund functionality still needs work on. It is currently refunding incorrectly if the quantity sold is 0. (DONE April 8, 2024)

**April 8, 2024:**

* Added some login alerts.
* Worked on adding a product variation on the add product functionality.

**April 9, 2024:**

* Worked on Edit Product functionality.
* Added some pagination on Sales Report page.

**April 10, 2024:**

* Incorrect cost and Revenue to calculate Profit. (DONE April 10, 2024)
* Edit product has no labels for Cost Price and Selling price. (DONE April 10, 2024)
* Work on Refund and Sold functionality, its not correct. (DONE April 10, 2024)

**FOR POSTING IN LINKEDIN:**

🚀 Exciting News! I am thrilled to share the project I've been working on – an Inventory Management System built with Java, Spring Boot, and an array of other cutting-edge technologies!

🏬 Project Overview:

This system is designed to streamline inventory management for any store or business involved in selling. It leverages Java Spring Boot for robust backend functionality, Thymeleaf for dynamic template rendering, and HTML/CSS/JavaScript/Bootstrap for creating a polished frontend interface. The data is managed efficiently using MySQL as the database.

💼 Skills Showcase:

Throughout the development process, I've embraced the role of a versatile full-stack software engineer. From gathering initial requirements and conducting thorough analysis to designing, coding, testing, and deploying the system, I've been involved in every stage of its lifecycle.

📚 Key Learnings:

Spring Security: Implemented to manage authentication and access control effectively.

Logging with Log4j 2: Ensured comprehensive logging for debugging and monitoring purposes.

REST API: Utilized RESTful principles for building robust API endpoints.

JUnit & MockMvc: Employed for unit testing and ensuring the reliability of components.

Git & GitHub: Leveraged for version control and continuous integration/continuous deployment (CI/CD) using GitHub Actions.

GitHub Desktop: Facilitated seamless collaboration and version control management.

🔍 Project Highlights:

Hexagonal Architecture: Structured the project to ensure flexibility and scalability.

Thorough Documentation: Implemented a comprehensive OpenAPI Specification for clear and organized API documentation.

CI/CD Implementation: Leveraged GitHub Actions for automated testing, Docker image generation, and deployment to Docker Hub upon successful tests.

🔗 Interested in exploring the code? Feel free to check it out on GitHub!

I am incredibly proud of the progress made and the skills acquired throughout this project journey. Excited to hear your thoughts and feedback!

**(ALSO ADD THE SWAGGER-UI AND OTHER PHOTOS ON THE POST!!)**

**Also add PDF and excel export.**

**Add security for reading from outside files with the constants.**

**Mention that my wife is using it for her online store for tracking and that it is stored in our local server.**

**Post it on Java community for feedback.**

**Mention following SOLID principles.**

**Tested using MockMvc on controllers and restcontrollers. Tested using Mockito on services.**