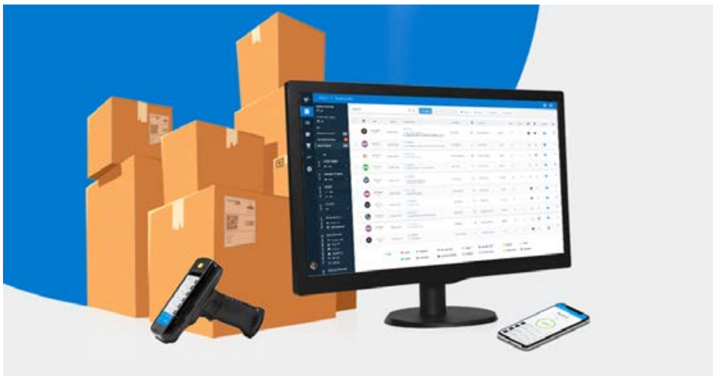


INVENTORY MANAGEMENT SYSTEM

[Document subtitle]

ABSTRACT

You are tasked to implement an inventory management system using this document as a description for specification and instruction guide.

Contents

Inventory Management System

The Application

The objective of this project is to apply the knowledge you have learnt in the Java Lecture Sessions by developing a simple Inventory management application for the local car dealership workshops that provide a collaborative workspace for mechanics and admin clerks. You are required to develop a web application that enables admin clerks and mechanics to browse the parts currently in the inventory (the system contains inventory product details from multiple vendors). It is common for mechanics to specialise items from different dealers (e.g. Toyota, Honda etc)

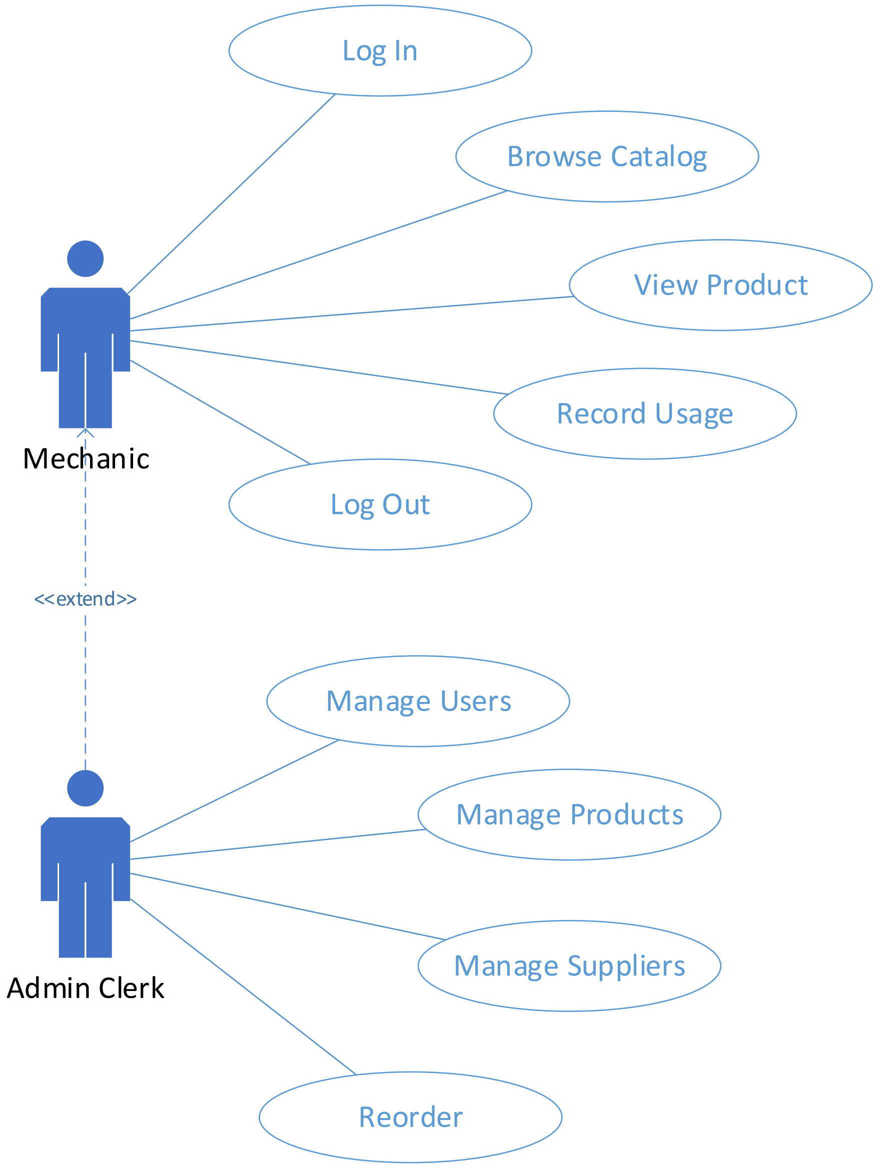
Overview

- There are two typical groups of users for the application system: admin and mechanic.
- Both admin and mechanic can log in and log out of the system. They have different menu sets. Admin can also perform all functions performed by the mechanic.
- Both admin and mechanic can perform product consumption transactions.
- Admin can reorder products with a particular supplier.
- The team can choose to have either separate log in pages of same log in page.

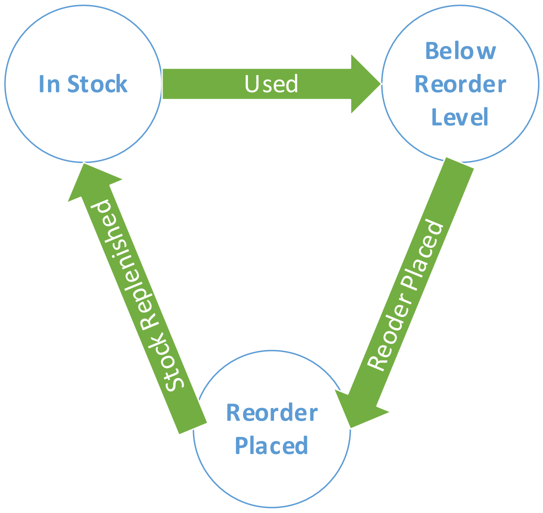
Understanding Features

The features of the system are listed in two groups. Mandatory features must be implemented; a good implementation of all mandatory features alone will earn the team a pass. To maximize the marks attained, you should implement one or more of the Optional features described below. Optional features are not listed in any particular order, and you should prioritize them as you see fit. It is important to note that implementing features (optional as well as mandatory) badly can actually make you lose marks rather than gain them. It is important you give the appropriate amount of thought to the design of each feature you implement. The functionality of your system should be potentially usable in a real-life site. Usability and robustness should be high in your list of priorities.

Use Case Models



State Transition Diagram



Mandatory Features to Implement

Master inventory must have following –item name, description, type, original price, category, price for wholesale, price for retail, price for partner, subcategory, units, product ID (unique), supplier(unique),brand name, brandid (unique)

Sign In, Login and Logout

The application will have two separate entry points (URL) for Mechanic as well as Admin Users. All access to the application will be password-protected, so that user identity can be established. User ids and passwords will be stored in your application database.

For this assignment, you may assume that proper authorization will be added at a later stage. Therefore, you only need to perform authentication from the main access page, and need not secure access to operations, nor verify authorization for every Web Page Access. This means that it is acceptable that a user is able to bypass the system security by manipulating URLs.

Stock Entry Forms

This feature is used by the admin clerk to enter the new stock that has arrived in the dealership premises and update the inventory accordingly. Admin clerk maintains products and can track inventory for products, and for any combination of products, attributes, and options. Make assumption on the required data elements. For example, master product data may include details such as part number, unit price, description, color, dimension, manufacturer, and reorder level, minimum reorder quantity and shelf location. Admin clerk can also adjust inventory to record damaged items that are returned to the manufacturer. All data entry forms perform appropriate server-side validation.

Stock Usage Forms

This feature is used by the mechanics to register the car they are currently fixing and the amount of consumable inventory they have used against the fixing task. A mechanic can look up parts in the dealership's inventory. The mechanic enters the desired part number into a browser and is presented with a list of the parts in the inventory matching the requested part number. The mechanic can then ask for the transaction history details of a part selected in the list. Mechanic is given flexibility to search on various parameters such as part number, part description, color, and manufacturer detail.

Mechanic records the part usage detail against the quantity he has used and customer for whom he used the part. The stocks are to be updated after mechanic commits his usage record. A mechanic can also choose to see transaction history of a particular part based on a date range.

System Order Mail Reminders

Based on the reorder level for every product, an email is triggered as a reminder to admin staff that a particular product needs reorder immediately.

View Product Listing

At any point on time, both admin and mechanic can use the product search page with various filters to look for selected product listing and their respective stock level.

Usage Reports

Based on a given data range, the system should be able to generate relevant inventory usage report for selected product.

CRUD Screens

Proper manage use cases for Users, Mechanics, Suppliers and Products need to be maintained.

Optional Features

Reorder Reports

The Admin clerk prepares Reorder Report on a weekly basis. The report will contain the part number, price, quantity on hand; reorder point, minimum order, order amount and price. The order amount is calculated when the quantity on hand falls below the reorder point (quantity < reorder). It is calculated as the sum of the reorder point and the minimum order less the quantity on hand. Provide a report file report.dat heading "Inventory Report", captions for each column and an "End of Report" message at the end of the report. Print the part number with leading zeros and do other necessary formatting.

Sample content of report

```
      Inventory Reorder Report for products from Supplier S1

      ------------------------------------------------------

===================================================================

Part No. Unit.Price  Qty  Reorder Qty. Min.Ord.Qty  Ord.Qty   Price

===================================================================

0123         1.23     19       20            1          1      1.23

0234         2.34     34       50           25         25     58.50

3456        34.56     56       50            0          0      0.00

4567        45.67      7       10            5          5    228.35

5678         6.78     75       75           25          0      0.00

===================================================================

                                              TOTAL          288.08

===================================================================

                             End of Report
```

Pagination

If many results are returned (say more than 30 records), the system should present them over several pages. Each page will contain navigation facilities so that other pages of search results can be shown (similar to search engine results). The number of results per page should ideally be selectable by the user (e.g. 10, 20, 25).

Mechanic's Fixset

Fixset comprises of selected parts. Example, a car retyre rocedure involves four new tyres, consumable jockey tools, pack of grease, and bolts. Mechanics can add/update/delete/get parts using the provided REST API's. Mechanics can also add/update/delete/get fixsets using the provided REST API's. When composing fixsets, mechanics create set of parts for a fixset category and provide quantity required. It is assumed here that, full parts list is needed for every fixset order placed.

Spring Reactive

Design a WebFLux Mono based publish and subscribe service for mechanic fixset and parts results.

Client Side Technologies

Design a simple client side page to consume and test the messages created.

Tools and Design

Software development tools

Version	Tool Name	Details
Java SE	JDK	You can download a copy online
Spring	STS	You can download a copy online
Data Base	MySQL	Preferably Community Edition

Design Consideration

The team must design, develop, test and release a web-based application, using a standard data store such as MySQL. The team can use a model-view-controller architecture using any web technology discussed in the lectures such as Spring MVC Framework and/or Spring Reactive. A reasonable business layer is expected to validate leave type, claim dates, eligibility and approval processes. The persistent data layer can be implemented using Simple Data framework.

The following are the system design consideration.

- The systems are developed using Spring components.
- You should strictly follow the MVC Architecture.
- You are free to design either thymeleaf based front end or react based front end.
- The system uses MySQL Database to store the data. Do design appropriate data scripts and also populate the database with sufficient test data

Evaluation Criteria

Everybody contributes and no excuses. The team will be evaluated on the technical quality of the application deliverable. Beyond working solution some expectations would be:

- Implementation of best practices; for example, use of right annotations, layering of architecture etc.
- Credits would be allocated for proper exception handling implementations, server validation logic, test cases and utility classes
- Sharing of ideas among team members is always a welcome.

Plagiarism Notice: All students share the responsibility for upholding the academic standards and reputation of the University. Academic honesty is a prerequisite condition in the pursuit and acquisition of knowledge. Academic dishonesty is any misrepresentation with the intent to deceive or failure to acknowledge the source or falsification of information or inaccuracy of statements or cheating at examinations/tests or inappropriate use of resources. Please avoid sharing code, you will be penalized if found guilty.

Deliverables

This assignment is part of the continuous assessment for this course. You will be evaluated for 25 marks on the whole. You will work in your team. No individual work will be accepted. The followings are the deliverables.

- A 20 minute presentation, explaining the design and code of your team work.
- A simple 5 slide presentation with the following
  - Team, Work Distribution
  - ONE Architecture/Layer Description
  - ONE Class Diagram or ER Diagram
  - Technologies Used
  - Lessons Learnt
- Submit your Java project workspace containing all the source code, compiled classes, HTML, servlets and any other files (Like data scripts) required to run the application. If the team is using additional plug-in or library, it is to be bundled with deliverables. Write the names of your group members on the disk.
- Peer evaluation form.

Presentation Date

FINAL SUBMISSION: 23rd December Wednesday 2020. Detail schedule will be announced nearer to the presentations.

Cooperative Team Is Likely a Successful!!! Good Luck..