Project's Objectives and Learning Goals

This is a group course project with **2** students. You can start on the project from the **first day** of the course and concepts that must be used will be continually discussed during the duration of the semester. This is a multitier application with ORM (Object Relational Mapping) support.

Learning goal:

- This project will focus on:
  - Creating a presentation tier in Java Swing/JSF
  - Creating a middle tier for the service interface in Spring framework
  - Creating a data tier in Hibernate framework

You will need the following tools:

- Java https://java.com/en/download/
- Spring https://spring.io/
- Hibernate https://hibernate.org/
- Java IDE editor:
  - NetBeans https://netbeans.apache.org/download/index.html (great in handling the jdbc:derby database)

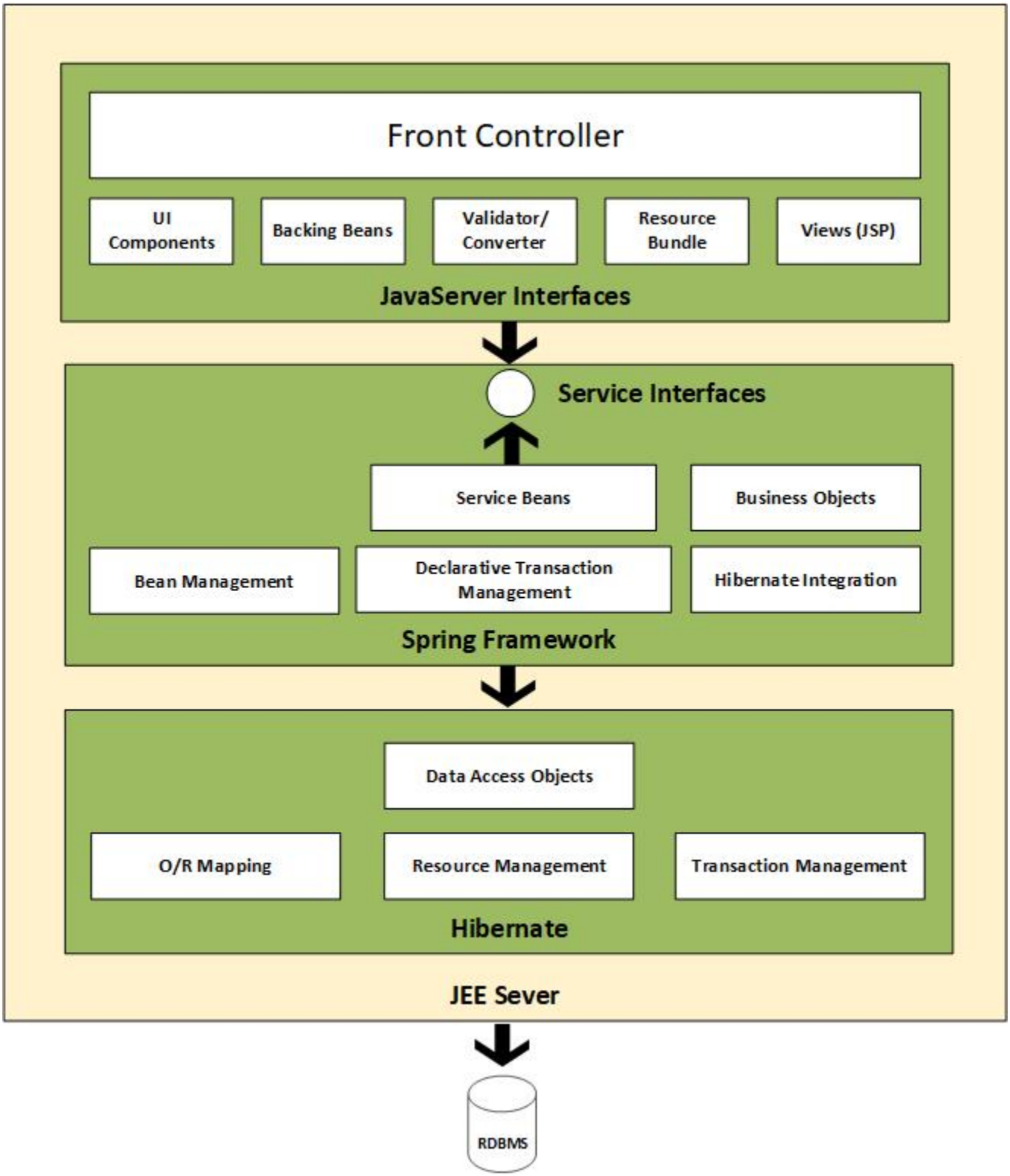
Project Details

The company MA Scene Inc. is a small local boutique store specializing in selling modern style clothing and accessories. The currency is in Canadian dollars. They requested a desktop application to view and sell the items in the store by the sales associates. All the items have a bar code associated with it, stock numbers (quantity of the item available), price, title (name of item), colour, text description, and sizes (if applicable). Before the final purchase, a full order list with prices and a tax amount of 13% is added to the total. The pay now (doesn't have to really work but a pop up window saying "Purchased" is sufficient). Finally, the order history is stored in the system to be viewed at any time with a timestamp of purchase by sales or administrator.

The backend where the administrative personnel can access the shop allows them to add new items to the store, edit them, and remove them completely. An item that is out of stock (0 items left) or not available for a few days has a textual area to add the reason why it is not available to be purchased, and the reason can be viewed by the sales associate too.

The owners of the company have asked you to develop this in-house store system with the credentials mentioned above. If your team wants to create an additional feature of search items by title or bar code number in the front and backend of the store then it will be a 10% bonus on the final delivered product. (**It is a bonus of 10% on this course project grade**.)

To understand how you will approach this project, please look at the following architecture diagram to view the relationship of the components used in designing and programming such a system.


Front Controller
UI Components
Backing Beans
Validator/ Converter
Resource Bundle
Views (JSP)
JavaServer Interfaces
Service Interfaces
Service Beans
Business Objects
Bean Management
Declarative Transaction Management
Hibernate Integration
Spring Framework
Data Access Objects
O/R Mapping
Resource Management
Transaction Management
Hibernate
JEE Sever
RDBMS

There are three layers:

1. Presentation (View): use the Java Swing libraries for the user interface. The front and backend of the store can be accessed by a login system. You will have two login types: sales and administrator.
    - Sales: can view the store's items and submit a purchase order.
    - Administrator: can do all the actions of a sales associate to the store and access/edit/delete items in the backend to be viewed in the front end of the store's application.

    Rules: you can only add one item at a time to an order if it is available. When the item is added, the quantity value is decreased by one. If removed from order and not purchased, then the value goes back to what it was before.

2. Business Logic (Controller): use the Spring framework. Create a Spring boot application without the web dependency. You will have a bootiful application based on Maven with all the features of Spring, starting from dependency injection.

    Hint: using basic dependency injection, load the beans into ApplicationContext and get the objects that way. There are many tutorials online that you can Google that will explain step by step how to approach this.

3. Persistent Database (Model): design SQL queries, create and connect to the database. Perform appropriate CRUD operations as needed. Use Java hibernate for mapping the object-oriented domain model to a relational database.

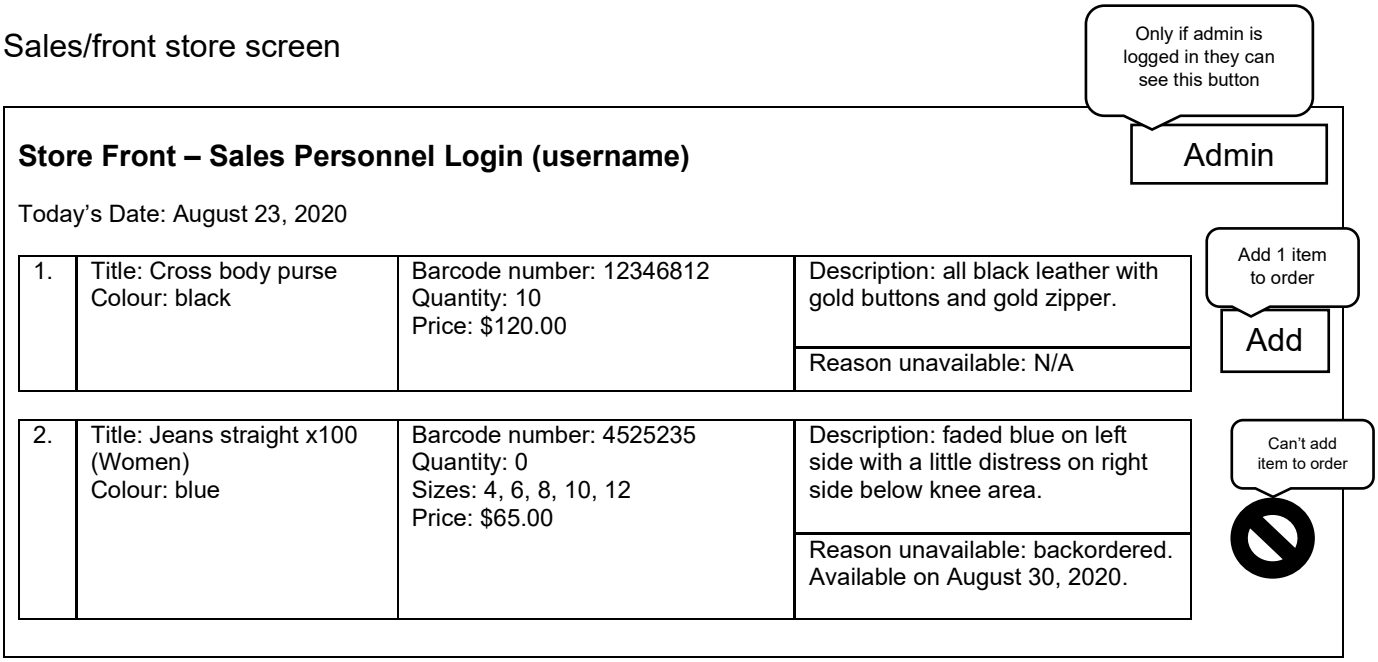
    Note: you must design and create a relational database with appropriate primary and foreign keys.

Possible look and feel:

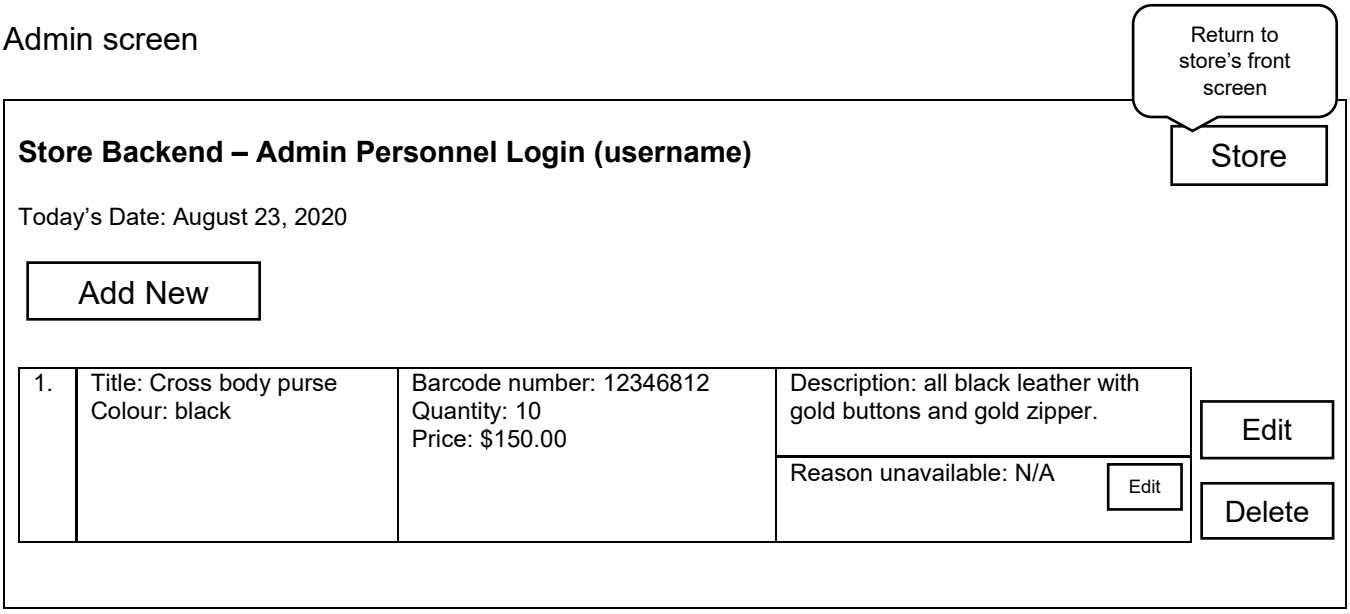
You will have three main interfaces depending on login type:

1. Login screen

2. Sales/front store screen



3. Admin screen



- When adding a new item, it is best to have a pop up window appear to handle the request. If the user choses not to add a new item, a cancel button can be clicked to return to item(s) screen.
- When editing an item, it is best to have a pop up window appear to edit the item's information or cancel.
- When adding a reason why the item is unavailable, the stock value automatically goes down to zero (0). If there is no reason, then the quantity remains the same value.

4. Order Screen

Order history:

1. August 20, 2020 at 11:15 am [view]
2. August 21, 2020 at 1:45 pm [view]
3. August 21, 2020 at 3:22 pm [view]

Current Order(s):

1.	Item: Cross body bag x 1 Barcode number: 123466812	Price: $150.00 Tax: $19.50 Total: $169.50
2.	Item: Silk shirt with collar embroidery Barcode number: 234365756	Price: $120.00 Tax: $16.60 Total: $135.60
		Grand total: $305.10

Remove | Cancel | Complete Purchase

- Remove button: remove order (delete it completely)
- Cancel button: return to store's screen
- Complete Purchase button: complete order transaction
- [view] link: pop up window of the order information

Things to consider:

- When adding an item to the current order, the order items are stored in the current session.
- When the desktop application is closed, the current order disappears and all item quantity values are reset, and the user must login again to the system and start again.

Sit down and design the application first. Start in the beginning of the term and as you learn new concepts in the course, apply these techniques when suitable.

Don't start coding right away!

Group Tasks

Using Java Swing libraries, Spring Framework and Hibernate Framework, a fully operational application is delivered. You must comment your code and provide proper documentation in the root directory of your application. You must explain what each file does and how it is related to other classes; your group can provide diagrams too. **Your group must explain in a separate document what each team member has contributed to the project (coding and design).**

Course Project

Delivery Expectations

- Zip all your files and/or folder containing files in this format:
  Group#_(names)_section#_project.zip
  … and submit in submission box for **Course Project**.
- Your professor will extract the zip file on their end and use an IDE to run the application and view the results.
- The last week of lectures, your professor and your group (and the entire class) will discuss how the course project was done. Your participation marks are part of the group work.

Rubric

Programming and Design

	(Excellent)	**(Good)**	**(Fair)**	**(Poor)**
Program execution	Program executes correctly with no syntax or runtime errors (19-20)		Program executes with a minor (easily fixed error) (12-13)	Program does not execute (0-7)
Correct output	Program displays correct output with no errors (19-20)	Output has minor errors (16-18)	Output has multiple errors (13-15)	Output is incorrect (0-7)
Design of output	Program displays more than expected (12-13)	Program displays minimally expected output (9-10)	Program does not display the required output (5-8)	Output is poorly designed (0-4)
Design of logic	Program is logically well designed (19-20)	Program has slight logic errors that do not significantly affect the results (16-18)	Program has significant logic errors (13-15)	Program is incorrect (0-8)
Standards	Program is stylistically well designed (6-7)	Few inappropriate design choices (i.e. poor variable names, improper indentation) (4-5)	Several inappropriate design choices (i.e. poor variable names, improper indentation) (2-3)	Program is poorly written (0-1)
Documentation (comments)	Program is well documented (10)	Missing one required comment (8)	Missing two or more required comments (4-6)	Most or all documentation missing (0-3)

Programming and design: _____ / 90

Group work: ____ / 10 (based on the document submitted in the zip file and class discussions)

Bonus: ____ / 10*

Total: ___ / 100