

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY (PERAK)

BACHELOR OF COMPUTER SCIENCE (HONS)

17-February-2021

# **UCCD3243 SERVER-SIDE WEB APPLICATION DEVELOPMENT**

GROUP ASSIGNMENT

**<Group P8\_3>**

**<Online Order Monitoring System>**

**Tan Xi En 1904098 CS  
Yeoh Siow Poo 1802886 CS  
Yap Jheng Khin 1800224 CS  
Yeong Hui Kei 1806283 CS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Actual Marks** | **Student 1** | **Student 2** | **Student 3** | **Student 4** |
| **Report Section (Group)** | **20** | **Tan Xi En** | **Yeoh Siow Poo** | **Yap Jheng Khin** | **Yeong Hui Kei** |
| Analysis – Application Level – Functional charts | 15 |  |  |  |  |
| Analysis – Application Level – UML diagrams |
| Design – Functional Requirements |
| Strengths and Limitations | 5 |  |  |  |  |
| **Application Section (Individual)** | **60** |  |  |  |  |
| Application – Functional Requirements | 20 |  |  |  |  |
| Application – Design/Architecture | 20 |  |  |  |  |
| Application – Technologies | 20 |  |  |  |  |
| **Application Section (Group)** | **20** |  |  |  |  |
| Professional Outlook (Front End and Back End) | 5 |  |  |  |  |
| Overall Quantity and Quality | 15 |  |  |  |  |
| **Total** | **100** |  |  |  |  |

Checked by,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ts. Dr Anbuselvan Sangodiah)

Contents

[**1.0**](#_heading=h.3znysh7) **Analysis** 1

[**1.1**](#_heading=h.2et92p0) **Functional Hierarchy Chart** 1

[**1.2**](#_heading=h.tyjcwt) **UML Diagram** 2

[1.2.1 Analysis Class Diagram 2](#_heading=h.3dy6vkm)

[1.2.2 Use-case Diagram 3](#_heading=h.1t3h5sf)

[1.2.3 Design Class Diagram 3](#_heading=h.4d34og8)

[1.2.4 Database Diagram 4](#_heading=h.2s8eyo1)

[1.2.5 Package Diagram 4](#_heading=h.17dp8vu)

[**1.3**](#_heading=h.3rdcrjn) **Other Diagram** 5

[1.3.1 Sequence Diagram 5](#_heading=h.26in1rg)

[**1.4**](#_heading=h.lnxbz9) **Functional Requirement** 5

[1.4.1 Product Module 5](#_heading=h.35nkun2)

[1.4.2 Order Module 5](#_heading=h.1ksv4uv)

[1.4.3 Payment Module 6](#_heading=h.44sinio)

[1.4.4 Staff Module 6](#_heading=h.2jxsxqh)

[1.4.5 Admin Module 6](#_heading=h.z337ya)

[1.4.6 System Account Module 7](#_heading=h.3j2qqm3)

[**1.5**](#_heading=h.1y810tw) **Strengths and Limitation** 7

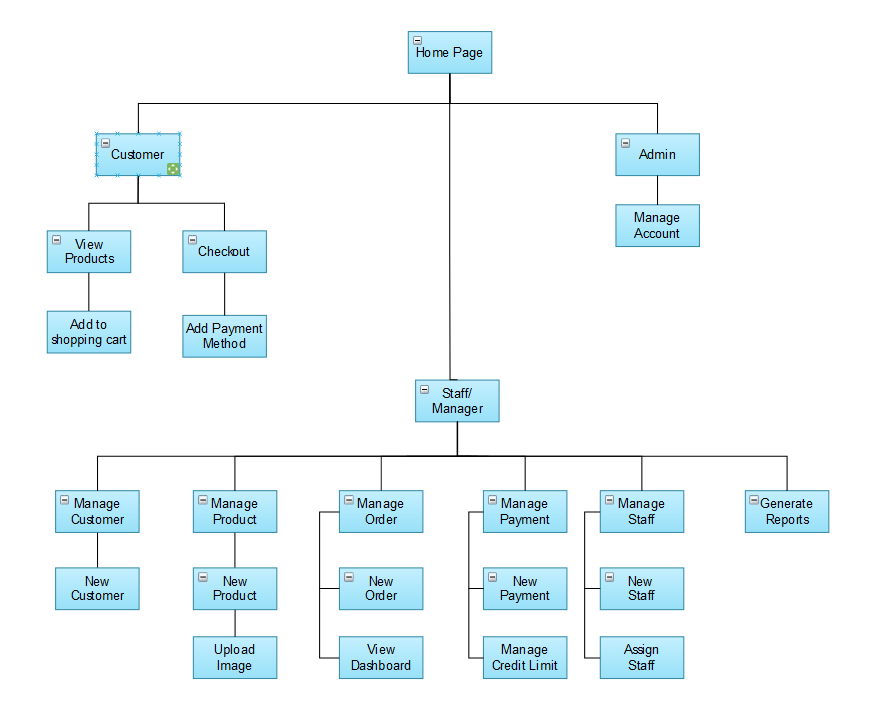
[1.5.1 Strength 7](#_heading=h.4i7ojhp)

[1.5.2 Limitation 8](#_heading=h.2xcytpi)

[**1.6**](#_heading=h.1ci93xb) **Operational Requirements** 8

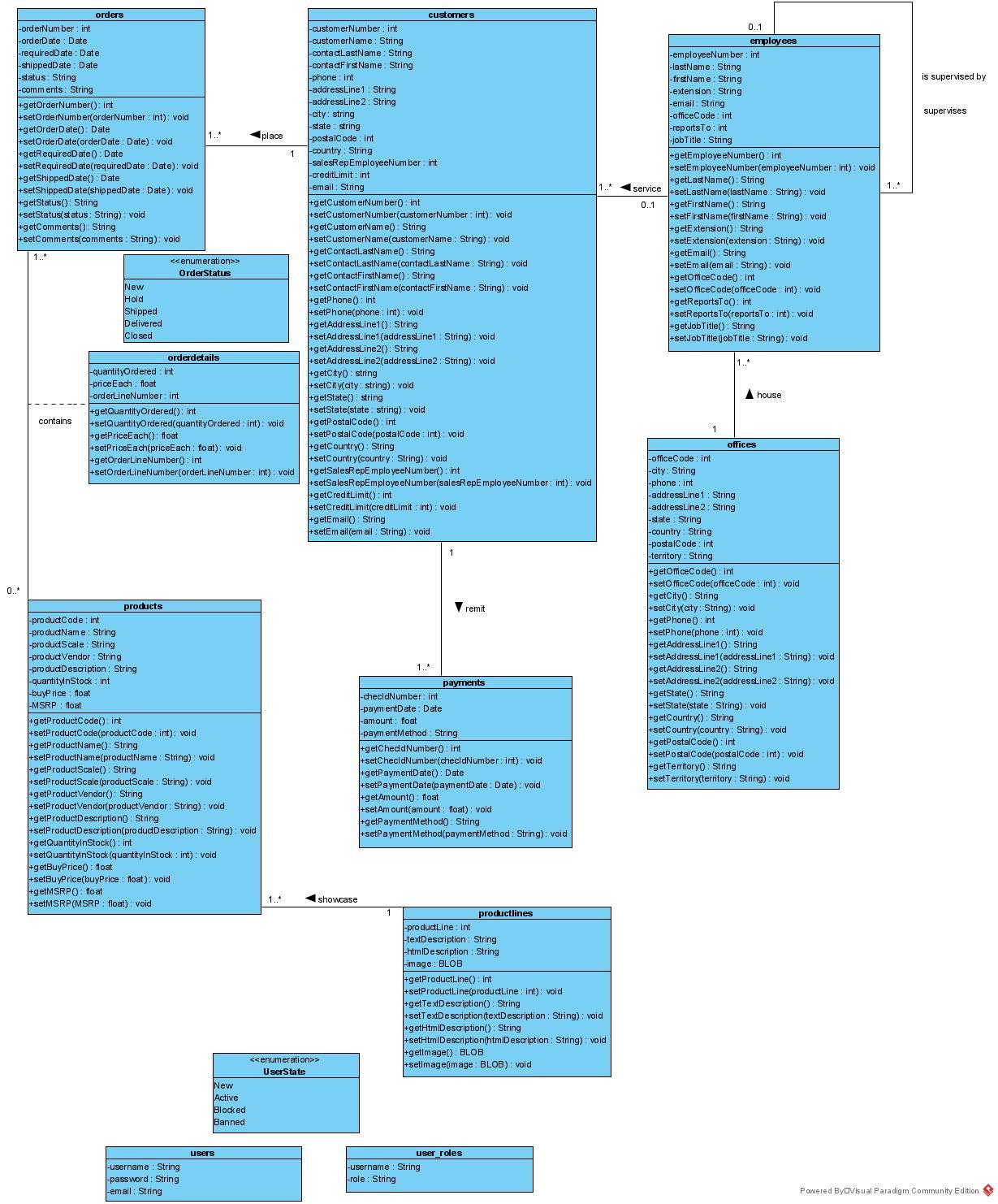
# **1.0 Analysis**

## **1.1 Functional Hierarchy Chart**

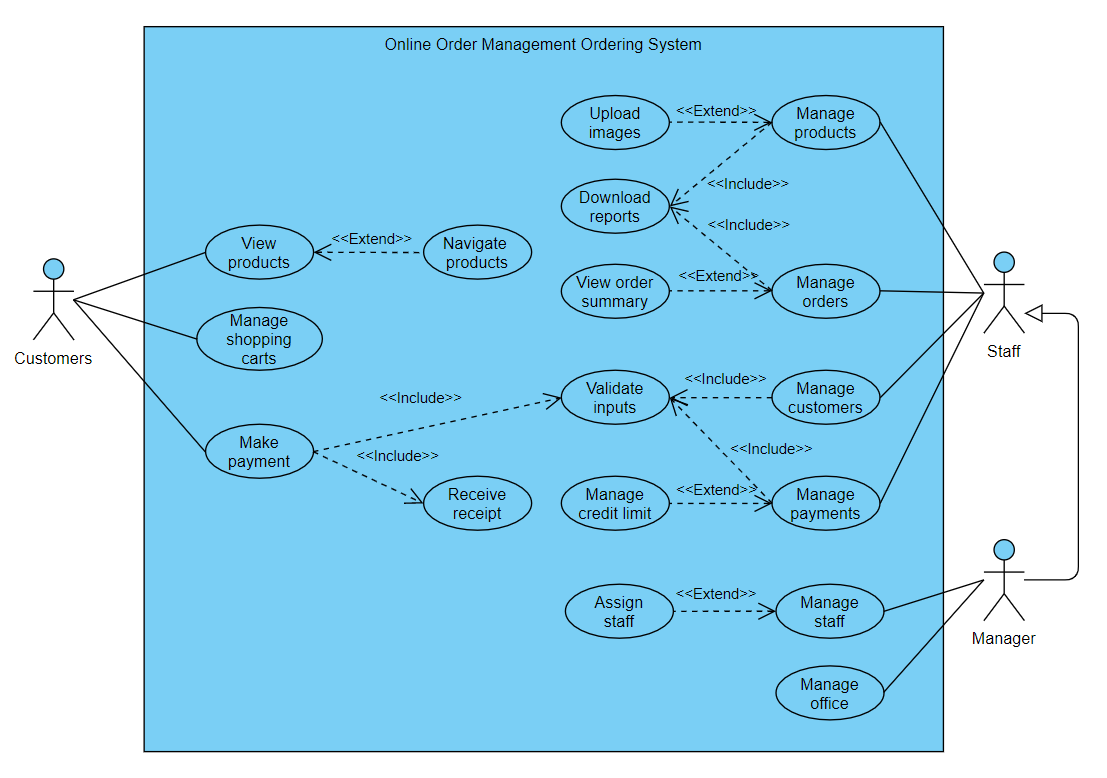


## **1.2 UML Diagram**

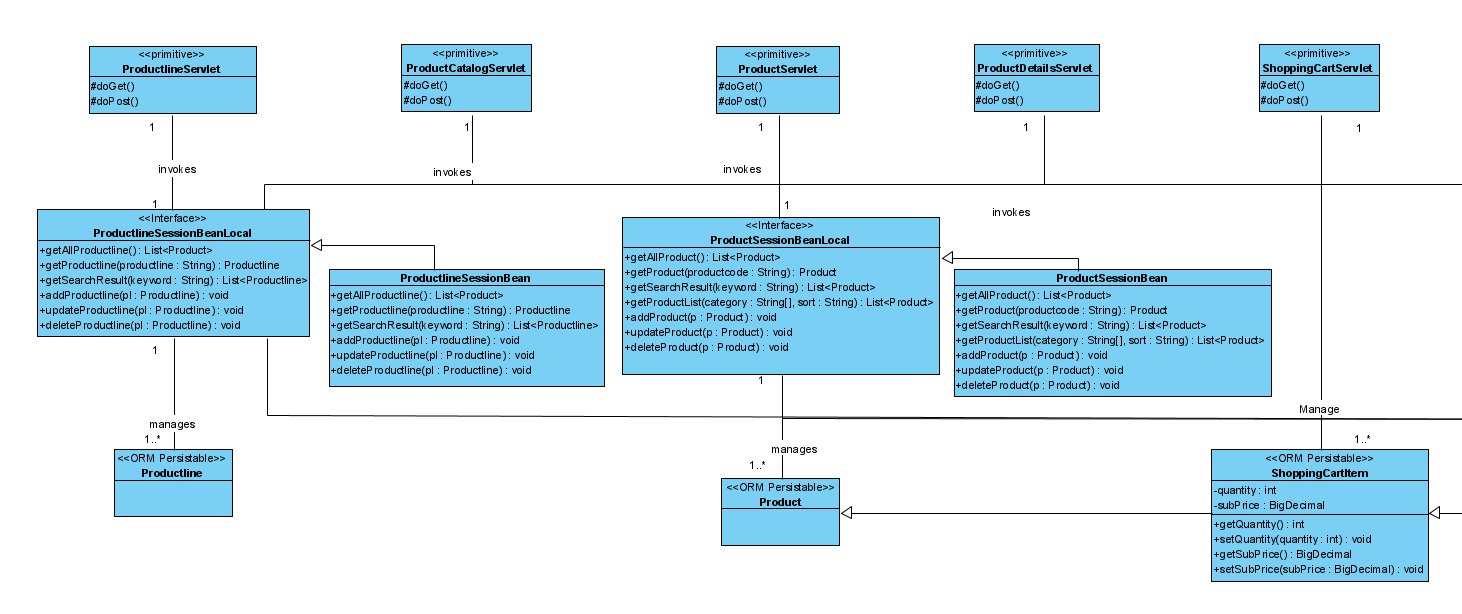
### 1.2.1 Analysis Class Diagram

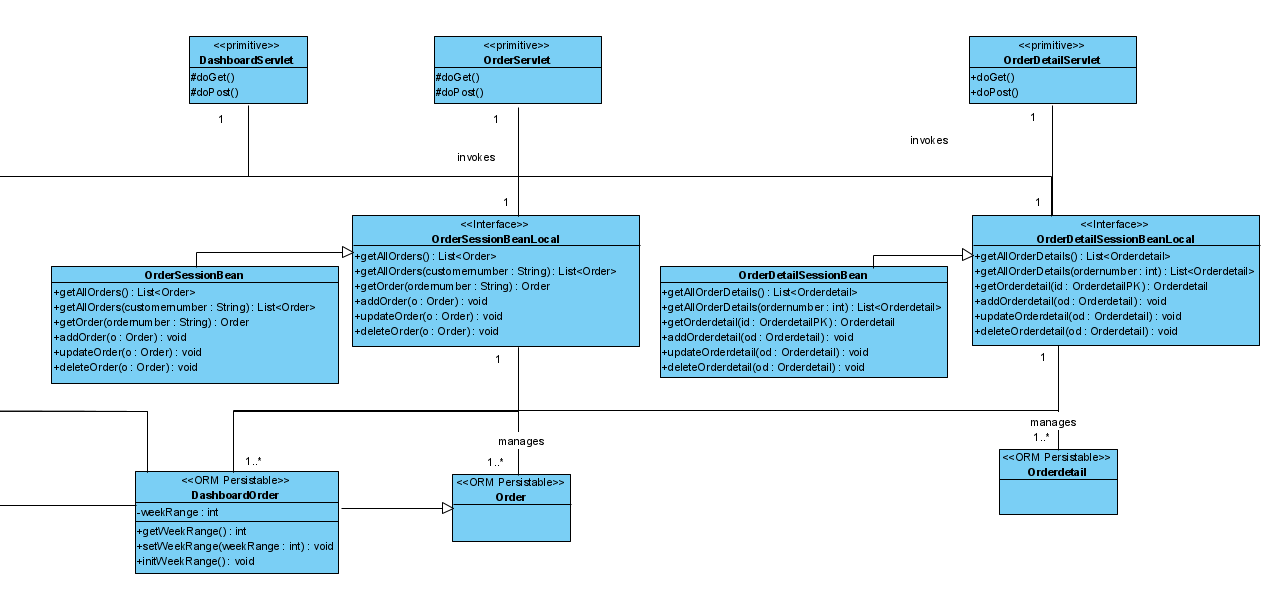


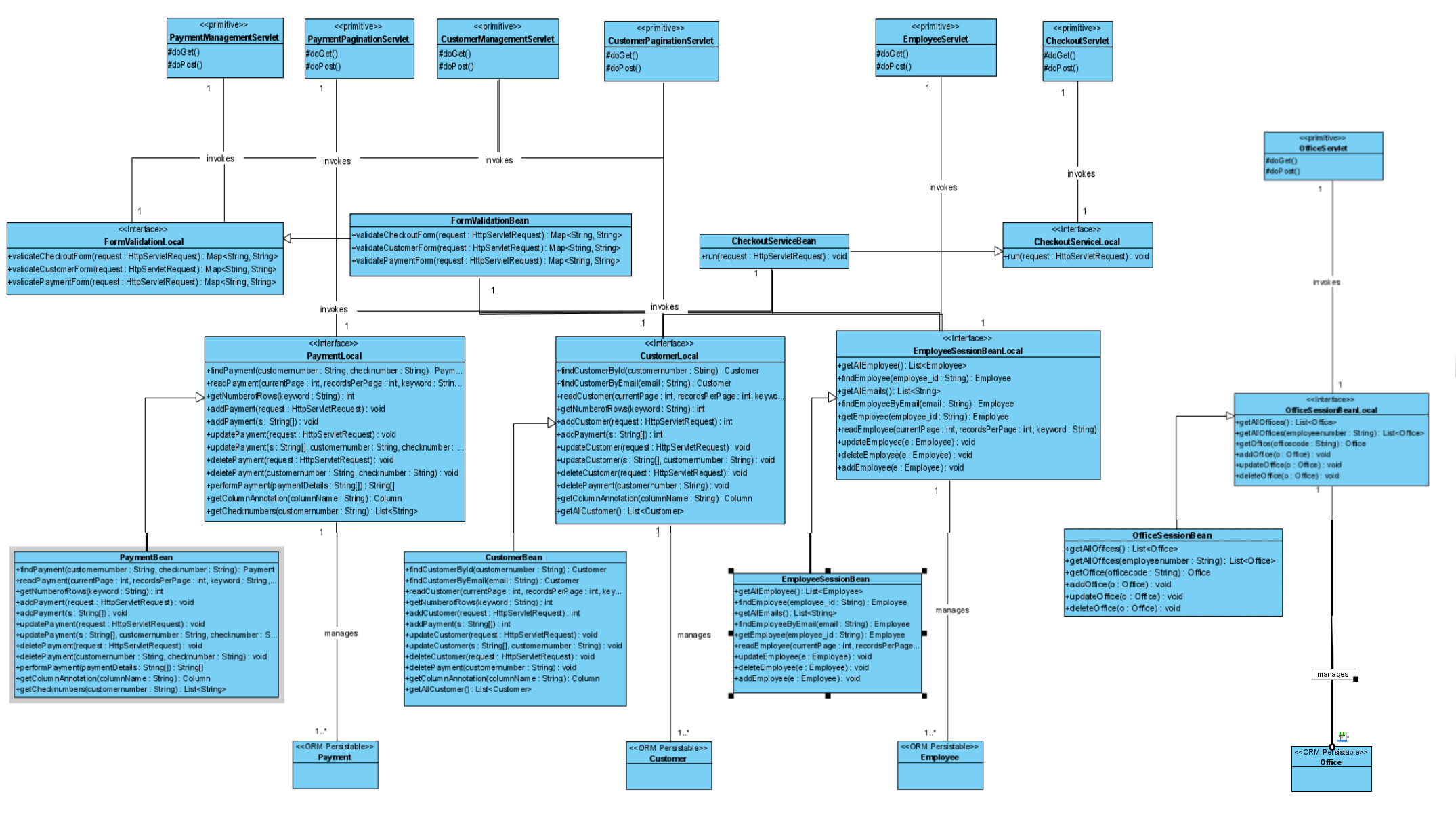
### 1.2.2 Use-case Diagram



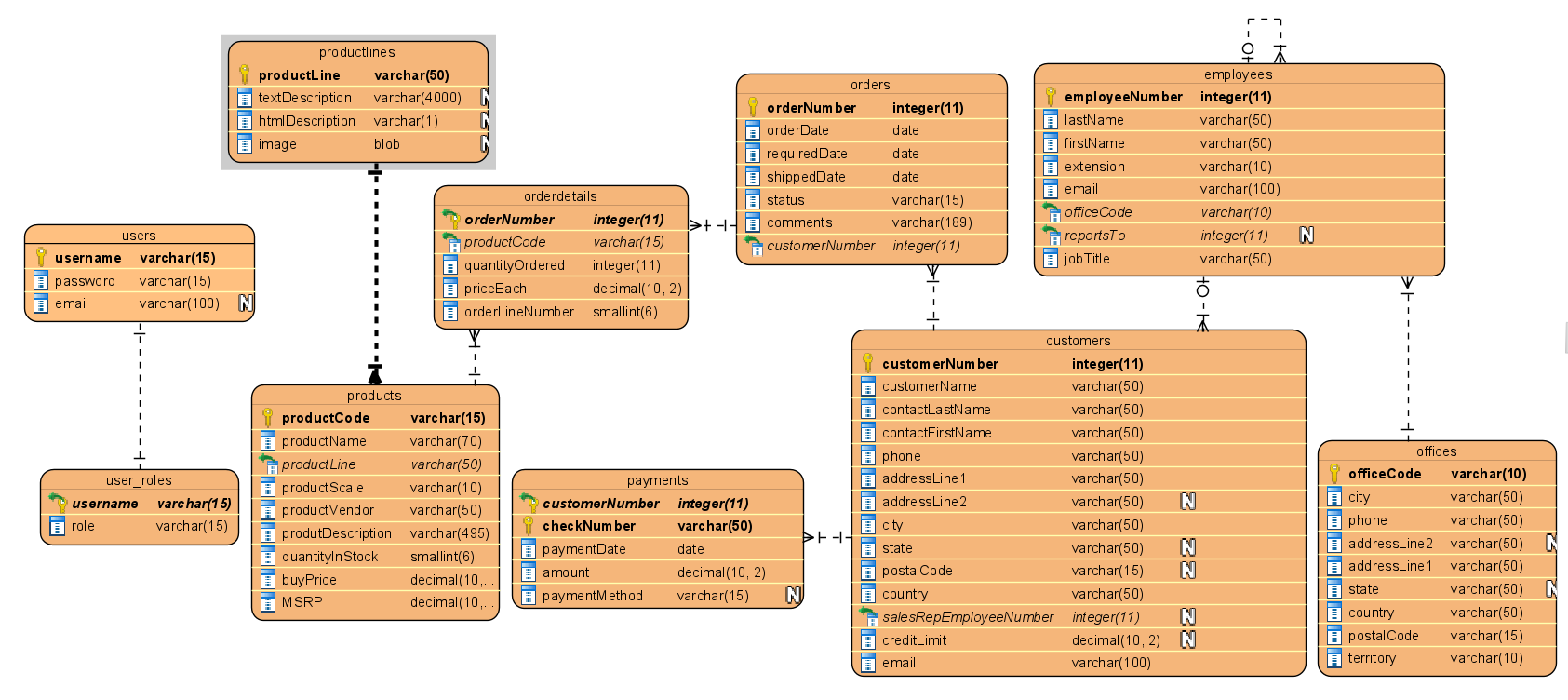
### 1.2.3 Design Class Diagram





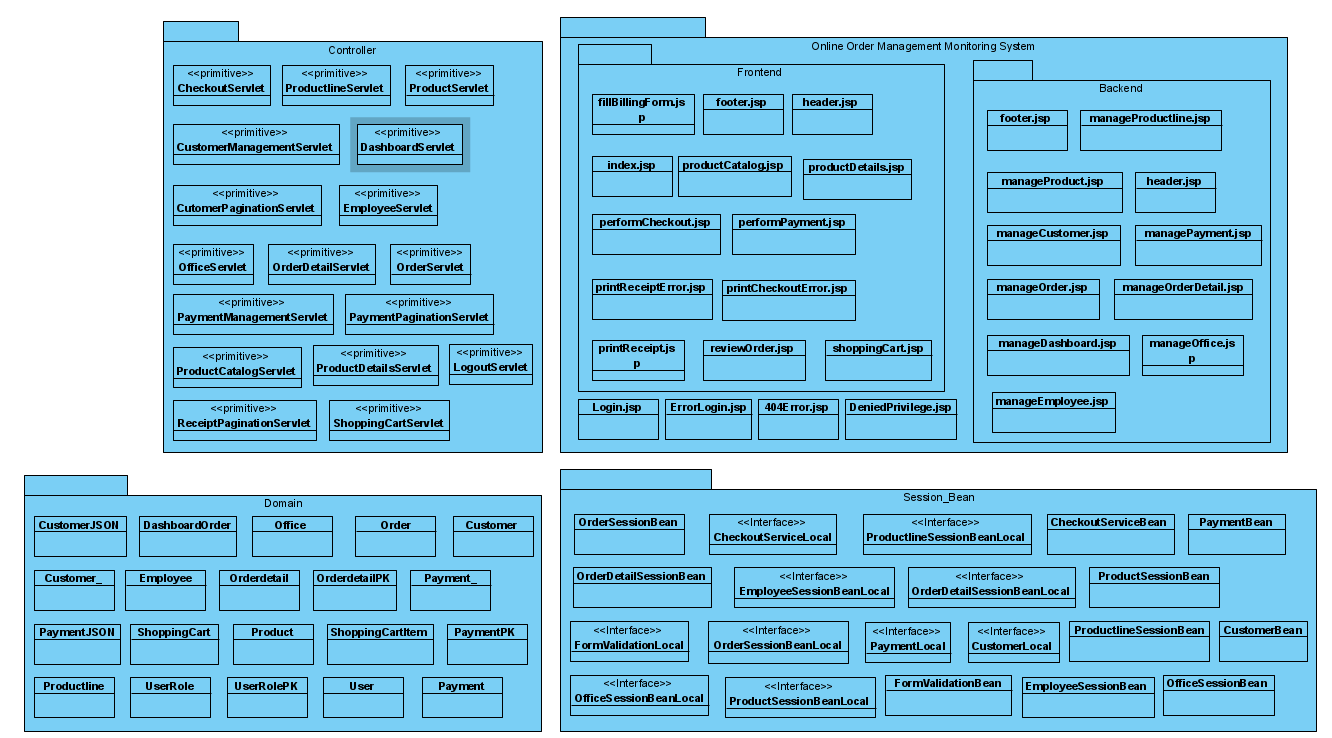


### 1.2.4 Database Diagram



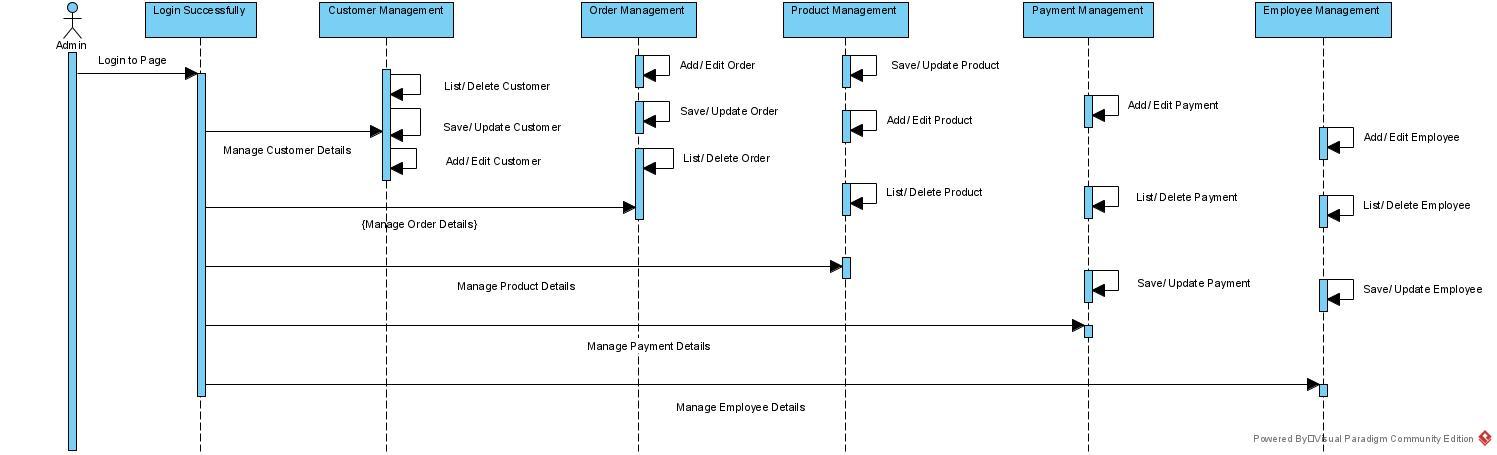
* Payments Table include a new attribute called paymentMethod, datatype of varchar(15), has a Not NULL constraints, which only accept 5 values (“check”,” cash”, “debit card”, “credit card”, “online banking”)
* Customers Table includes a new attribute called email, datatype of varchar(50).
* Users and User\_Roles has a one-to-one relationship.

### 1.2.5 Package Diagram

****

## **1.3 Other Diagram**

### 1.3.1 Sequence Diagram



## 

## **1.4 Functional Requirement**

### 1.4.1 Product Module

* Customers shall be able to navigate to different product categories.
* Customers shall be able to see the dynamic page of each specific product line/product categories.
* Customers shall be able to see the dynamic page of each specific product.
* Staff shall upload product images to the system.
* Staff shall filter and view all the product’s records.
* Staff shall insert the product’s records if given authority to do so.
* Staff shall update all the product’s records if given authority to do so.
* Staff shall delete all the product’s records if given authority to do so.
* Staff are able to download an excel report of the list of products.

### 1.4.2 Order Module

* Staff shall insert the new order’s records if given authority to do so.
* Staff shall filter and view all the order’s records if given authority to do so.
* Staff shall update all the order’s records if given authority to do so.
* Staff shall delete all the order’s records if given authority to do so.
* Staff shall be able to view the order summary for each customer’s order.
* Staff shall update the order status to “Shipped”, “Resolved”, “Cancelled”, “Disputed” or “In Process”.
* Customers shall add products to the shopping cart before proceeding to checkout.
* Staff is able to view monthly orders of all order’s records in the dashboard.
* Staff is able to download an excel report of the list of orders.

### 1.4.3 Payment Module

* Staff shall create the new payment’s records if given authority to do so.
* Staff shall read all the payment’s records if given authority to do so.
* Staff shall update all the payment’s records if given authority to do so.
* Staff shall delete all the payment’s records if given authority to do so.
* Staff shall search and filter all the payment’s records if given authority to do so.
* Staff shall sort all the payment’s records if given authority to do so.
* Staff shall assign sales representatives to the customers if given authority to do so.
* Staff shall receive validation feedback if the inputted payment’s record is wrong.
* Customers shall review the order before performing the checkout.
* Customers shall make the payment for the order when performing checkout.
* Customers shall choose online banking, debit card, or credit card as the payment method during checkout.
* Customers shall receive validation feedback if the inputted checkout’s information is wrong.

### 1.4.4 Staff Module

* Staff shall create the new staff’s records if given authority to do so.
* Staff shall read all the staff’s records if given authority to do so.
* Staff shall update all the staff’s records if given authority to do so.
* Staff shall delete all the staff’s records if given authority to do so.

### 1.4.5 Customer Module

* Staff shall create the new customer’s records if given authority to do so.
* Staff shall read all the customer’s records if given authority to do so.
* Staff shall update all the customer’s records if given authority to do so.
* Staff shall delete all the customer’s records if given authority to do so.
* Staff shall search and filter all the customer’s records if given authority to do so.
* Staff shall sort all the customer’s records if given authority to do so.
* Customer shall update his/her record when filling up billing information during checkout.

### 1.4.6 Office Module

* Staff shall create the new office’s records if given authority to do so.
* Staff shall read all the office’s records if given authority to do so.
* Staff shall update all the office’s records if given authority to do so.
* Staff shall delete all the office’s records if given authority to do so.

### 1.4.7 Admin Module

* Admin shall be able to log in the backend of the system if they input the valid username and password
* Admin shall be able to manage all orders, products, employees and customers in the system backend.

## 

## **1.5 Strengths and Limitation**

### 1.5.1 Strength

Firstly, the Java Criteria API (a part of the Java Persistence API) is used to construct criteria queries for customers and payments. The API allows more flexibility and customization as compared to NamedQuery or NamedNativeQuery. For this assignment, the author creates a custom search function that allows searching between string and non-string attributes, which is not possible if NamedQuery or NamedNative Query is used. According to Oracle (2013), criteria queries use the metamodel class and its attributes to refer to the managed entity classes and their persistent state and relationships. Thus, the prerequisite of constructing a criteria query is defining a metamodel class for each of the managed entity classes. For example, a metamodel class called “Customer\_.java” is created for the managed entity class called “Customer.java”.

Secondly, the facade pattern is applied for checkout service and form validation service, respectively. It simplifies the communication between the servlets and the session bean when performing the web service. For the checkout service, a facade class called “CheckoutServiceBean” is used to mask the complexities of interaction between interdependent subsystems like payment system, order placing system, inventory system, staff information system and customer information system from the servlet. On the other hand, for the form validation service, a facade class called “FormValidationBean” is used to decouple the implementation of the form validation service from the servlet.

Thirdly, the system uses AJAX to increase the responsiveness of the system. As a result, the client can asynchronously send requests and receive responses without re-rendering the whole webpage. For this assignment, the Java EE server sends JSON as the AJAX response back to the client. Thus, a library called Gson is used to serialize Java Objects to JSON. When the client submits a checkout form using AJAX, the “CheckoutServlet” sends input validation messages back to the client. The process repeats until all the input fields in the checkout form are correct and only then the checkout service is started. The same method applies with the “CustomerManagementServlet” and “PaymentManagementServlet”. The purpose of using AJAX to validate input fields is to ensure that the values are valid to be persisted to the database.

Fourth, the system uses apache API to create excel documents. As a result, the client is able to download the list of order reports and product reports whenever the client desires. A library called apache openOffice is used to convert java class to excel files. When the client completes a transaction on products and orders, he is able to download the complete report of the list of orders or reports.

Lastly, this system is designed by using a pattern called Model-View-Controller (MVC). MVC design pattern allows efficient modification on either model part, view part or controller part without interfering with each other. For example, the developers can easily change the view part of the application by adjusting colors or using different UI frameworks without affecting the model part and controller part of the web application. In this assignment, the JSP files in the “frontend” and “backend” folder in the WebContent folder represents the views, the classes in the “domain” package represents the models, while the classes in the “controller” package represents the controllers.

### 1.5.2 Limitation

Firstly, the web application will treat President, VP Sales, and VP Marketing as the staff manager. The dynamic pages of President, VP Sales, and VP Marketing are the same with staff managers.

Secondly, the staff must manually set or raise the credit limit by themselves. There is no built-in recommender system to suggest a credit limit for the customers.

Thirdly, any staff can modify any records for customers, orders and payments. It is not restricted in a way that only the sales representative and manager of that sales representative can perform CRUD operation on the customer’s records. Besides, when a staff deletes a customer’s record, his or her order’s records and payment’s records will be deleted as well.

Fourthly, the customer can make payment without login into the system. Thus, the system cannot authenticate that the user with the username and password stored in the database. Alternatively, the customers use email addresses to “authenticate” themselves without password when making payments through the billing form.

Finally, the sales managers shall manually assign a sales representative to a new customer. There is no optimization or smart algorithm in place to assign the sales representative based on factors such as customers’ geographical location.

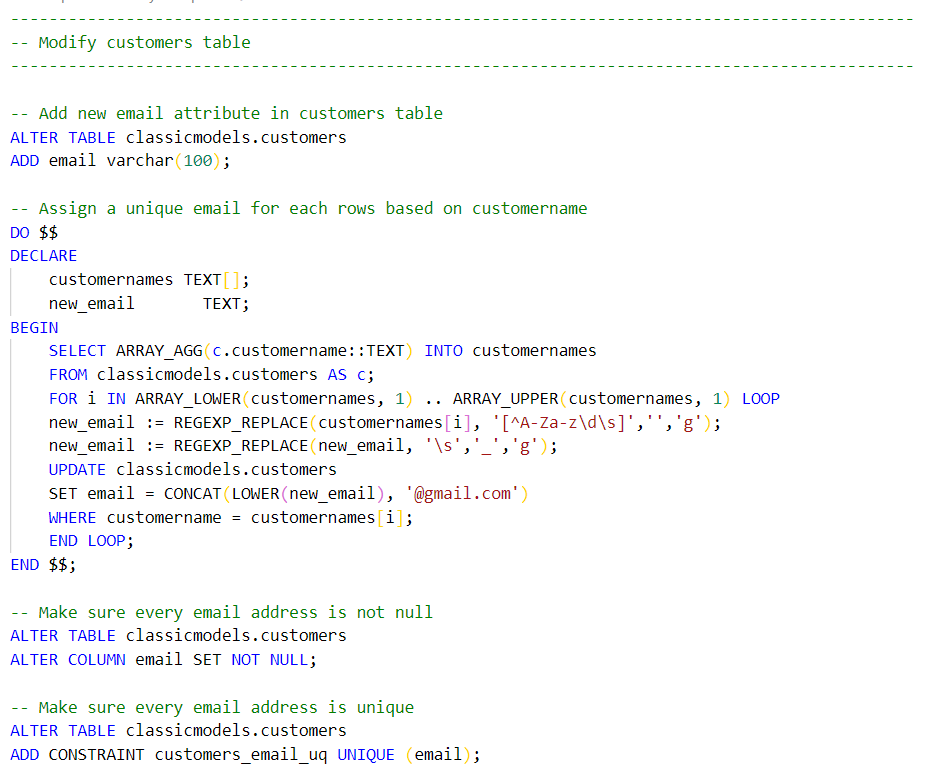
## 

## **1.6 Operational Requirements**

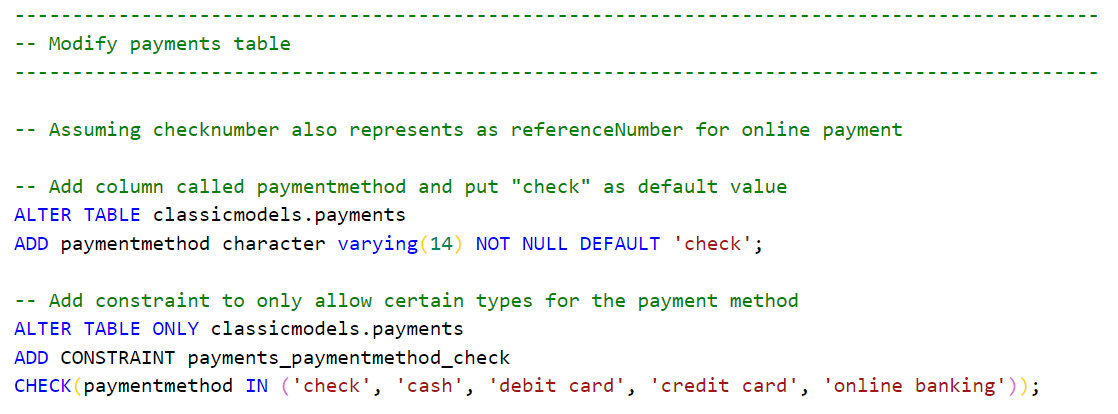
### 1.6.1 Database Requirements

**Note**: All the new modifications of the SQL script can be found in the “classicmodels\_dataset\_modification.sql” in the submitted folder. Run “classicmodels\_dataset \_original.sql” first before running “classicmodels\_dataset \_modification.sql”.

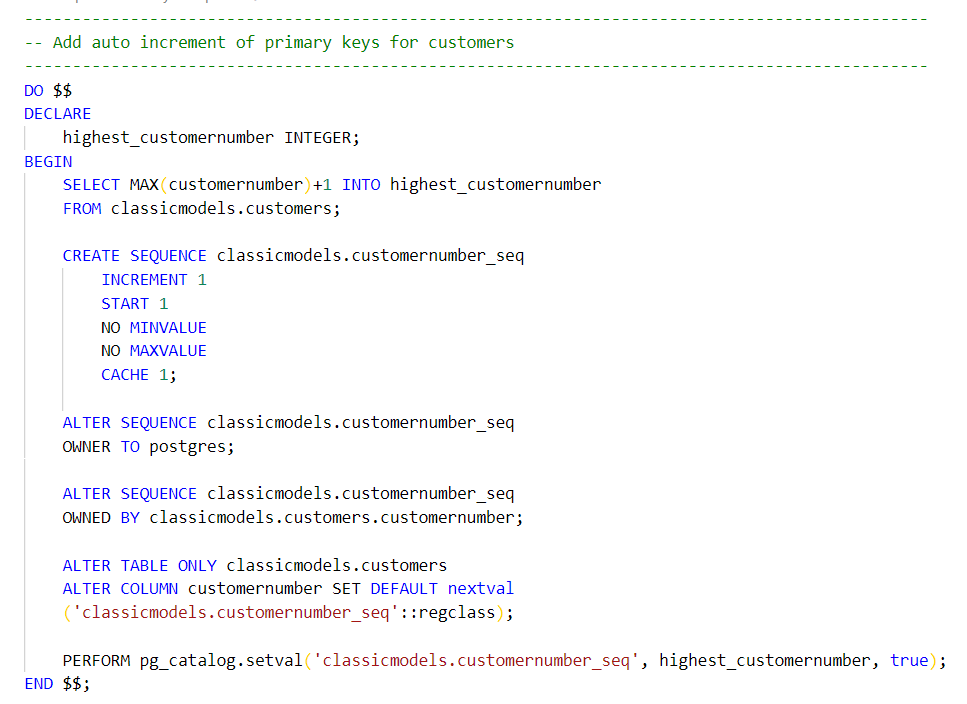
1) As shown in the code snippet below, we have added a new attribute in the CUSTOMER table, which is the email attribute. The email address is used to identify the customer when making payment. Unique constraint is added to ensure that every customer has a unique email address. The email attribute shall be a variable varchar(100). To satisfy the unique constraint, an anonymous block is executed to assign a unique email address to each existing customer’s records.

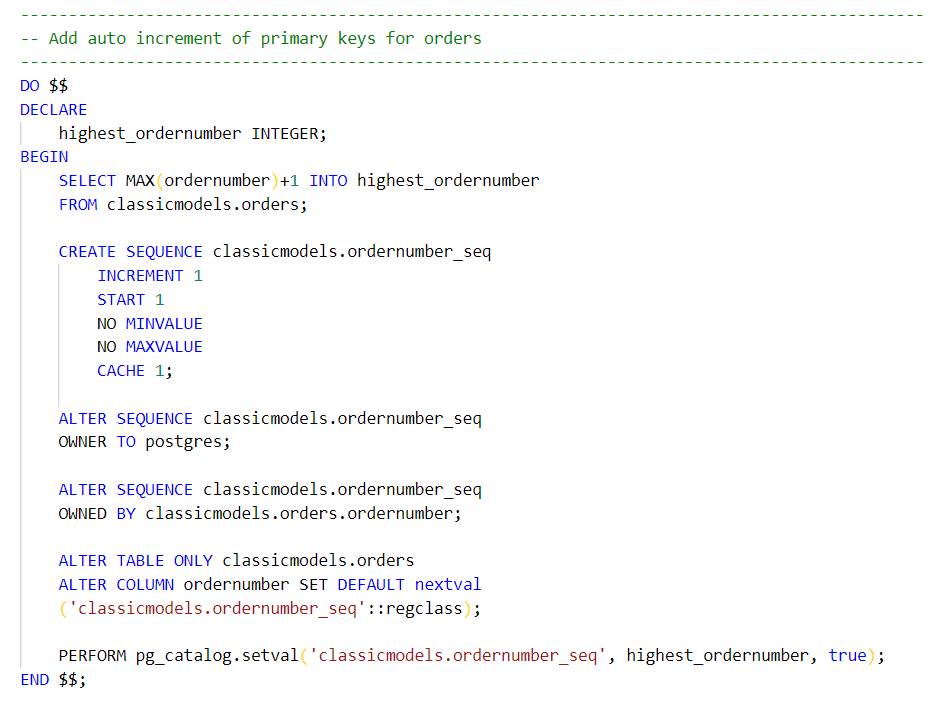


2) As shown in the code snippet below, there is a slight modification on the attributes in the payment tables. A new attribute called “paymentmethod” is added. The “paymentmethod” attribute only accepts values which are “check”,” cash”, “debit card”, “credit card”, “online banking”.



3) As shown in the two code snippets below, a sequence is created to generate unique primary keys for each new record, starting from the last (and therefore largest) primary key on the customers and orders table, respectively.





### 1.6.2 API Requirements

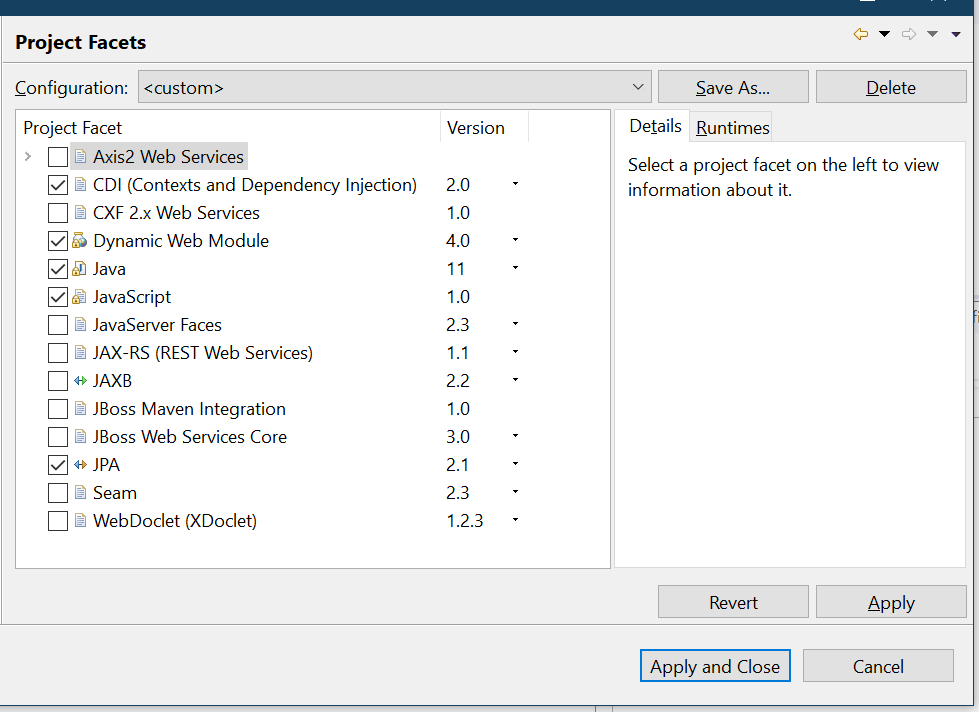
These two APIs can be found in the folder.

1. Gson. gson-2.8.6.jar
2. Apache. Poi-3.1.7.jar

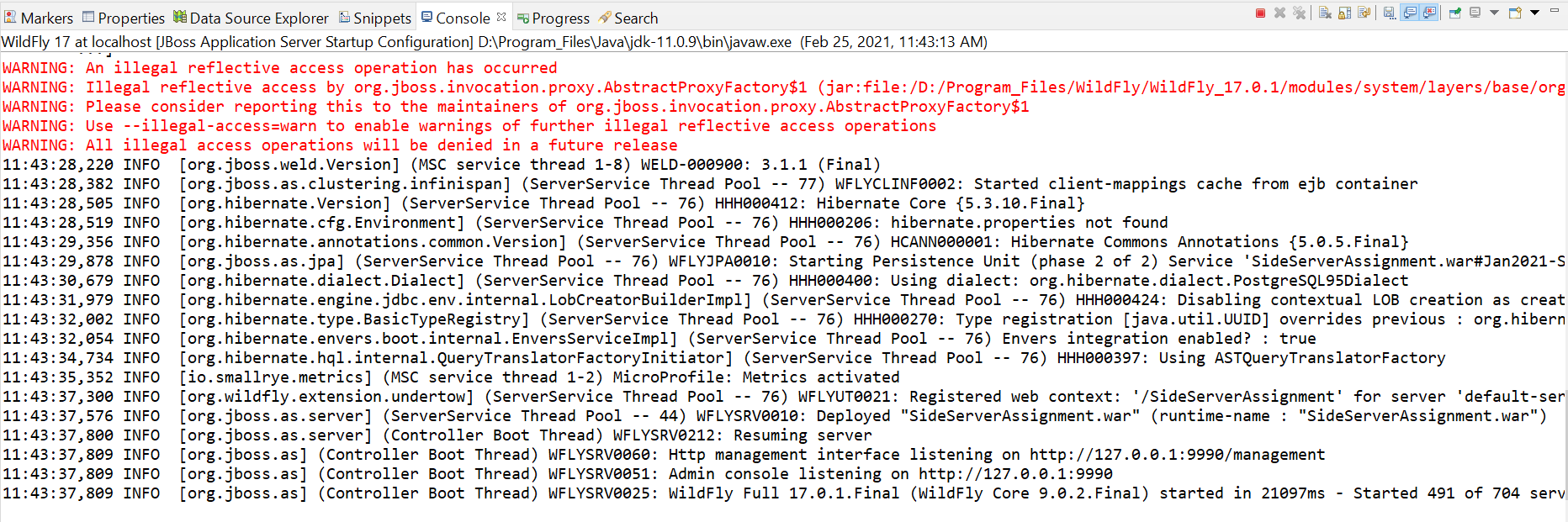
### 

### 1.6.3 Project Facets Requirements

The configuration of Project Facets is shown below. Note that the configuration should match with the diagram below. In particular, JPA should be at least version 2.1 to support the Java Criteria API.



1.6.4 Common Compilation Error



The warning highlighted in red above is due to [internal wildfly server error](https://issues.redhat.com/browse/WFCORE-4514) as shown in the image below. This error is only resolved in WildFly 21.0.0 later. More information can refer to the WFCORE-4514 - Red Hat Issue Tracker.pdf in the submitted folder.