**Introduction**

According to this module mainly focus on advanced principles of programming, programming language theory, API, OOP concepts, webservices refactoring to design patterns and other programming related technologies and all. Beside this I must developed online web system for the GoCheeta cab service, and that project basically depend on mentioned areas and all.

As well as I used java and JSP as a main programming language and apart from these few tools and technologies are here. The main actors are administrators and consumers, and they have their own unique operations. Another purpose of this is to enable customers to access from anywhere. Administrators and user can book a cab from registering to the system and allow find out suitable vehicles and their details also. finally, I completed the web system and fulfil all requirements with proper validations also.

**TASK A**

## System Requirement Specification

### **Introduction**

GoCheeta provides the best cab service required by customers. Currently they manage their booking process only on mobile application. So their planning to introduce a web application for attract their customers by improving functionalities. Here this web application allows customer to booking vehicles with online payments and allows admins to manage all the sales, customer, drivers, vehicles with add, update delete functions. The customer would able to booking vehicle by credit card or cash payments and ride details and invoice must be sent to the customer through the system. The admin should have the facility to view rides reports. After implementation of the application, it should be convenient for the cab service center.

### **Purpose**

Website is one of the best marketing tool. It can easily attract people for the business. Therefore, “GoCheeta” cab service centre has decided to implement a web site for enhanced their online booking system from mobile application to the web application. The main purpose of this SRS document is provide a set of details about functional and nonfunctional requirements, tools and technologies that going to use, and gives the basic idea about the system design. It focused on the functionalities related to the vehicle booking system. This document consists varies sections to identify the development process. Basically SRS helps to reduce the development time and bugs and errors that we are facing during the development period.

### **Scope**

“GoCheeta” is a well reputed company that provide a best cab service for the customers. This application has two type of users like customer and the admin. This website allows users to vehicle booking facility, availability checking facility and also it allows administrator to add, update, delete and view functions to manage sales data, booking data, drivers and vehicle data etc. also here customer can make payment online through this website.

**Functional Requirement**

This part describes the functionalities that we are going to include for the cab management system. It gives an idea about tasks that the application needs to perform and how the system will works. Following are main requirements of the system.

* Admin have facility to view sales details
* Admin can manage customer details with add update delete functions
* Admin can manage vehicle and driver details
* Customers can booking vehicle
* Customer can be able to send their service experience trough this application

These are the main functions of the application. These functions would be help to increase efficiency of this vehicle booking system that we are going to develop.

Once system is completed, following features must be available with its activities as appropriate.

**2.1.4 Non-functional Requirements**

This part describe the quality of the application. It will discuss the recoverability, security and maintainability of the system. And also accuracy and the backend function inside the system are considered in this non - functional requirements.

* **Security side of the system**

This system required customer personal details like address, phone number, email, and credit card details so system should be secure from the third party. So it would be helps to increase the security

* **Performance of the system**

System performance means customer be able to access the supplication easily with their username and the password and the system should be user-friendly

* **Quality of the system**

Quality of the software is very important when using a real-time application. All the requirement should be met QA and the BA verifications

**TASK B**

**UML Diagram**

**Use case diagram**

**Class diagram**

**Sequence Diagram**

**TASK C**

**Design Pattern**

Design Patters are well described solution to a common software problem that software developers are facing during the development. They represent the best practices of Object-Oriented Programming. The design pattern concept originated from the book titled **Design Patterns – Elements of reusable Object-Oriented Software** published by the four authors Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. They were later collectively known as the **Gang of Four (GOF).** As per Design Patterns – Elements of reusable Object-Oriented Software, there are 23 design patterns that can be categorized into 3. Creational, Structural and Behavioral patterns (tutorialspoin, 2022).

In this part mainly focus on creational Design patterns related points and we identified 3 main sub design patters under this creational design patterns. Including,

1). Singleton Design pattern

2). Factory Design pattern

3). Abstract Factory pattern.

**Singleton Design pattern**

Given that it offers one of the finest methods for producing an item, this pattern belongs to the category of creational design patterns. In this pattern, a single class oversees creating an object while ensuring that only one object is produced. This class offers a method of directly accessing the class's sole object, eliminating the need to instantiate it first. In essence, a Singleton wraps a special resource and makes it accessible across the entire program.

Table

Description automatically generated

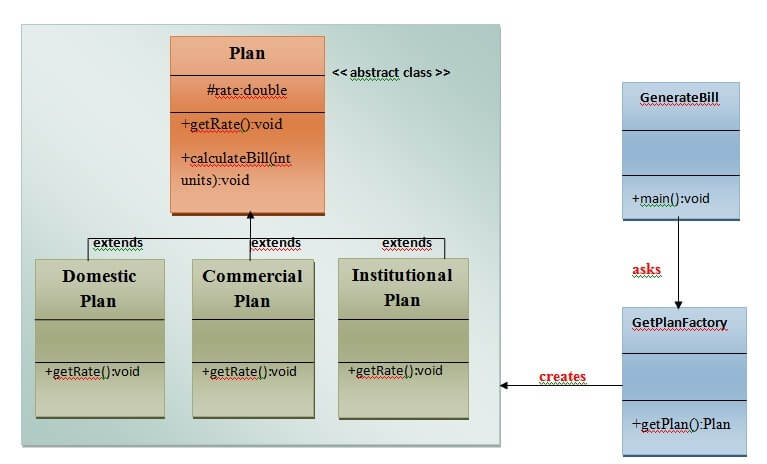
(Tennakoon, 2017)

|  |  |
| --- | --- |
| **Pros** | **Cos** |
| Instance control | Difficult to extend the singleton class |
| Flexibility | Abusing singleton will bring some negative problems |
| Saves Memory | It violated the single responsibility principle |

(Anon, 2020)

**Factory Design pattern**

Factory design pattern is one of the creational patterns. A further layer of abstraction above factory pattern, abstract factory pattern is almost identical to factory pattern. A super-factory that generates more factories is the center of abstract factory patterns. As a result, any desired concrete factory that can produce objects of the desired type is coupled with the abstract factory during runtime.



(Pankaj, 2022)

**Usage of Factory design pattern: -**

* When a class is unsure of the subclasses it will need to develop.
* When a class requests that the objects to be produced be specified by its subclasses.
* When the parent classes decide to create objects for their child classes.

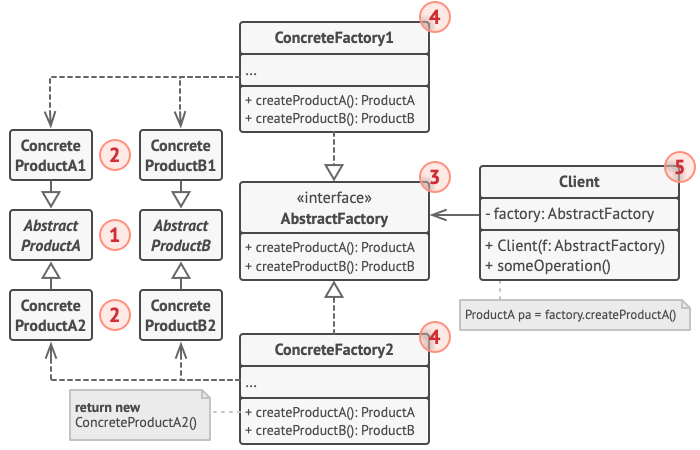
(Pankaj, 2022)

**Abstract Factory pattern.**

If someone is familiar with factory design patterns, they would be aware that each factory class in a factory design pattern produces a distinct subclass depending on the inputs it receives. The usual if-else and switch expressions are used in factory design patterns to accomplish this. The distinction of the abstract factory is that it does away with the if-else statement and uses a separate factory class for each subclass. Then, based on the inputs, an abstract factory class will return the subclasses. Even as it initially seems perplexing via continuous, you will understand the slight variances.

|  |  |
| --- | --- |
| Pros | Cons |
| Simple to maintain | Code may become more complicated |
| Principle of open / closed |  |
| Efficiency |  |

(Goswami, 2012)



(Goswami, 2012)

**TASK D**

**Screen shots of the application**

**Test Plan**

1. **TASK E**
2. **TEST PLAN**

**TASK F**

**User and technical documentation**

**TASK G**

**Git Hub**

**References**

# References

Anon, 2020. *developpaper.* [Online]   
Available at: https://developpaper.com/explain-the-advantages-and-disadvantages-points-for-attention-and-usage-scenarios-of-singleton-mode/  
[Accessed 06 12 2020].

Goswami, M., 2012. *javacodegeeks.* [Online]   
Available at: https://www.javacodegeeks.com/2012/10/abstract-factory-design-pattern-explained.html  
[Accessed 24 October 2012].

Pankaj, 2022. *Community.* [Online]   
Available at: https://www.digitalocean.com/community/tutorials/factory-design-pattern-in-java  
[Accessed 03 August 2022].

Tennakoon, J., 2017. *engineering.99x.io.* [Online]   
Available at: https://engineering.99x.io/singleton-pattern-47bf15ff2f7f  
[Accessed 18 March 2017].