**Project Specifications - Major Practical: Hotel Management System**

**Introduction:**

The aim of our project is to create a hotel booking management system. The program will allow the staff members of the hotel to select a booking from a predefined list of rooms for the customers. Also, the system allows the hotel management staff to view the details of all the customers who have booked a room in the hotel including the customers who have checked out of the hotel. The customers can view and select the type of room they want to stay in. the different room types provide different services to the customers. When the visitor wants to checkout of the room, they can pay for the room and finalise their stay, making the room available for others.

**Design Description:**

**Memory allocation from the stack and the heap**

* **Arrays:** dynamic array used to store customer’s in an array who have booked a room by linking the customer id with the room number.
* **Strings:** for user input output, room types description, customer name.
* **Objects:** rooms, hotel, customer,and the three different room types.

**User Input and Output**

* **I/O of different data types:** customer provides their contact details including name, phone number, and email address.The interface displays list of available room, different types of room that can be chosen from, the bookings made by the customers.

**Object-oriented programming and design**

* **Inheritance:** Inheritance is used in the luxury room class, the standard room class, and the dulex room class (the derived classes) to inherit the features from the already existing class Base class, which is the room class.Some of the features include; facilities access such as swimming pool, spa, gym etc.
* **Polymorphism:** Function overriding is used by having the same functions in the luxury room class, the standard room class, and the dulex room class (the child classes) that are already present in the room class (the parent class). This way we can set the number of beds and bathrooms, set the room number and set the cost for the 3 different types of rooms; the luxury room, the standard room, and the dulex room, which are all derived classes from the room class (the base class).
* **Abstract Classes:** room class is subclassed into the 3 different types of rooms that provides implementations for the abstract methods in the parent room class.

**Class Diagram:**

**Class Description:**

**Customer:**

The behaviours in the customer class allow to:

* Get and set customer’s name.
* Get and set customer’s contact details.
* Set and get customer id.
* Set and get room by assigning customer Id to the room number and vice versa.

**Hotel:**

**The behaviours in the hotel class allow to:**

* Check in a customer
* Check out the customer and make the room available for others.
* Book available rooms.
* Display the booking details.
* Display the room details.

**Room (abstract base class):**

**The behaviours in the room class allow to:**

* Checks the availability of a room. (polymorphism)
* Set and get the number of beds and bathrooms the rooms have. (polymorphism)
* Set and get the cost of the rooms. (polymorphism)
* Provides facility access to the different types of rooms. (inheritance)

**Outline your plan for finishing work, with allocated tasks to the group.**

**Schedule Plan:**

**Week 10 & Week 11:**

* **Poojan:** Test the customer class.
* **Poojan:** Add the option in the interface called “customer details” to view the details of all the customers who booked a room, checked in and checked out of the room.
* **Arsh:** Add the functionality that the different rooms will be displayed based on the budget the customer has. E.g. a customer with $1000 can chose any room from the standard to luxury, whereas a customer with a budget of $250 can also afford the standard room.
* **Poojan:** Think of a way to test the room class because it does not have any inputs.
* **Arsh:** Find a way to test the system using automated testing method.
* **Arsh:** We made our code more efficient by editing the code.

**Be able to describe (and preferably demonstrate) two test situations.**

**Testing Plan:**

**Unit & Automated Testing:**

Our unit tests will cover all public functions in our classes. Each class will have a corresponding test file like main-class.cpp, which will include a main method that will test each function with a set of inputs and test it matches an expected output.

**Class customer:** Customer class has been thoroughly tested with every possible outputs. The process used for testing this class was unit and automated testing. The inputs are the number of customer who want to book a room and the name and email id of each of the customers who want to book in. The output was the details of all the customers that booked a room and also the customer id was assigned to each of the customers.

**Regression Testing:**

Regression testing will be conducted to ensure that previously developed and tested code still performs as per the requirements after a change.

**Class hotel:** After every time we made changes to the room, the customer and the 3 child classes of the room, the hotel class was tested as all the other classes are linked to the hotel class. The testing was conducted while making the hotel booking system interface to ensure that all the menus worked correctly.