Temasek Polytechnic

School of Informatics and IT

**Diploma in Information Technology (IT)**

Software Design Specifications (DS)

**Project Particulars**

|  |  |
| --- | --- |
| **Tutor** | Mr Mel Goh |
| **Class** | P04 |
| **Project Title** | Delonix Regia Hotel Management System |

**Project Team’s Particulars**

|  |  |
| --- | --- |
| **Matric Number** | **Student Name** |
| **1406932B** | **Daniel Toh** |
| **1400797E** | **Edmund Yeo** |
| **1407193J** | **Dominic Ng** |
| **1405387J** | **Vivian Tan** |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 24/5/2016 | 1 | Finish up 2 | Edmund |
| 25/5/2016 | 2 | Finish up 1, 4 | Daniel |
| 25/5/2016 | 3 | Minor tweark on the report | Edmund |
| 25/5/2016 | 4 | Completed 5 | Vivian |

Contents

[1 DISTRIBUTION OF WORKLOAD 4](#_Toc451999284)

[2 ARCHITECTURE DESIGN 4](#_Toc451999285)

[2.1 What is 3 tier-application? 4](#_Toc451999286)

[2.2 Benefits of 3-tier application 5](#_Toc451999287)

[3 USER INTERFACE (UI) DESIGN 5](#_Toc451999288)

[4 PROGRAM DESIGN 6](#_Toc451999289)

[4.1 What is Program Design 6](#_Toc451999290)

[4.2 Usage 6](#_Toc451999291)

[4.3 What is Use Case Diagram? 6](#_Toc451999292)

[4.4 What is VOPC Diagram? 6](#_Toc451999293)

[4.5 Use Case 7](#_Toc451999294)

[4.6 VOPC Diagram 8](#_Toc451999295)

[*4.6.1* *Receptionist* 8](#_Toc451999296)

[*4.6.2* *Maintenance* 9](#_Toc451999297)

[*4.6.3* *Finance* 9](#_Toc451999298)

[*4.6.4* *F&B* 10](#_Toc451999299)

[5 DATABASE DESIGN 11](#_Toc451999300)

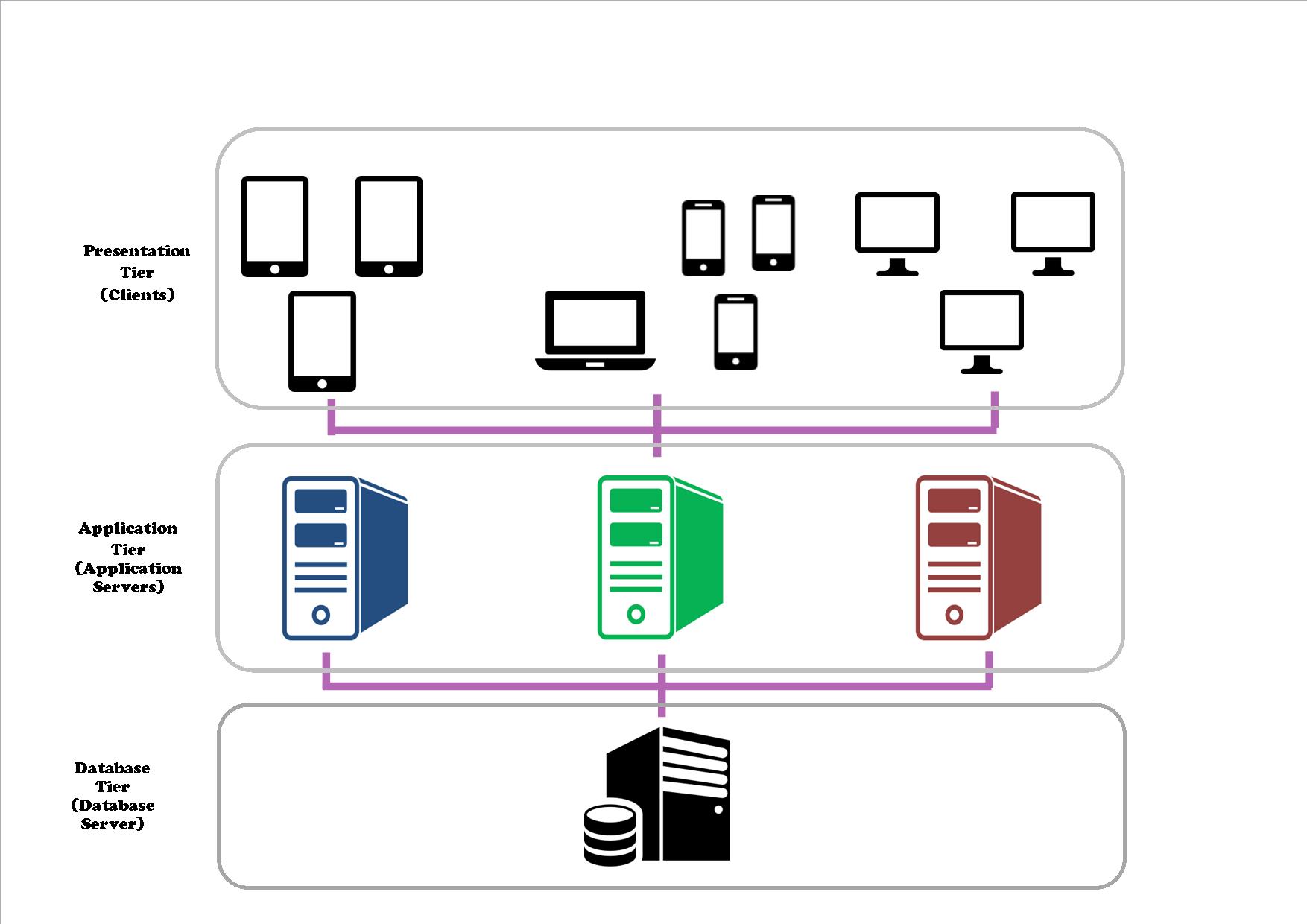
[5.1 Considerations 13](#_Toc451999301)

[5.2 Limitations 13](#_Toc451999302)

# DISTRIBUTION OF WORKLOAD

|  |  |
| --- | --- |
| **Design** | **Members** |
| 1, 4 | Daniel Toh |
| 2 | Edmund Yeo |
| 3 | Dominic Ng |
| 5 | Vivian Tan |

# ARCHITECTURE DESIGN



## What is 3 tier-application?

Three tier-application architecture is a client-server architecture in which the functional process logic data access, computer data storage and user interface are developed and maintained as independent modules on separates. Three-tier architecture is a software design pattern and a well-established software architecture.

Three-Tier architecture allows any one of the three layers/ tiers to be either upgraded or replaced independently. User interface is implemented on a desktop PC and uses a standard graphical user interface with different modules running on the middle layer (application layer). The relational database management system on the database server contains the computer data storage logic.

The three tiers in a three-tier architecture are:

Presentation Tier: Occupies the top level and displays information related to services available on a website. This tier communicates with other tiers by sending results to the browser and other tiers in the network.

Application Tier: Also called the middle tier, logic tier, business logic or logic tier, this tier is pulled from the presentation tier. It controls application functionality by performing detailed processing.

Database Tier: Houses database servers where information is stored and retrieved. Data in this tier is kept independent of application servers or business logic.

## Benefits of 3-tier application

* **Flexibility** - By separating the business logic of an application from its presentation logic, a 3-Tier architecture makes the application much more flexible to changes.
* **Maintainability** - Changes to the components in one layer should have no effect on any others layers. Also, if different layers require different skills (such as HTML/CSS is the presentation layer, PHP/Java in the business layer, SQL in the data access layer) then these can be managed by independent teams with skills in those specific areas.
* **Reusability** - Separating the application into multiple layers makes it easier to implement re-usable components. A single component in the business layer, for example, may be accessed by multiple components in the presentation layer, or even by several different presentation layers (such as desktop and the web) at the same time.
* **Scalability** - A 3-Tier architecture allows distribution of application components across multiple servers thus making the system much more scalable.
* **Reliability** - A 3-Tier architecture, if deployed on multiple servers, makes it easier to increase reliability of a system by implementing multiple levels of redundancy.

# USER INTERFACE (UI) DESIGN

The UI is in Web base as it is easier for the company to keep track of everything and the staff can only use the company system in the hotel so to prevent leak of data out. The design of the UI is user friendly as I have categorized the departments by color and everything that the company need is in the main page so that the CEO can easily access any information he or she wants to know.

Before the users can access to the main page each staff will be given a Username and Password for them to login and after the users have login the main page will show the user profile picture and the name of the user at the top right corner. This is to ensure that whatever the staff have change the system will update and to make sure that if there is any issues that happen the company will know who have change what in the process of using the system.

# PROGRAM DESIGN

## What is Program Design

It is the activity of progressing from a specification of a required program to the program itself. It is one of the first phase in the software life cycle. The purpose of this phase is a specification of what the program is able to do.

The design phase is usually divided into two sub phase, which are the architectural design and another one is the detailed design, the architectural design is to produces a description of the program and its major components of the program while the detailed design then refines the architectural design to a stage where implementation can begin.

The more energy, creativity and hard work that goes towards program design, the greater and higher chance that the program will succeed.

## Usage

Use Case Diagram and VOPC Diagram

## What is Use Case Diagram?

A sequence of interaction (steps) between an actor and a system that results in some benefit to the user.

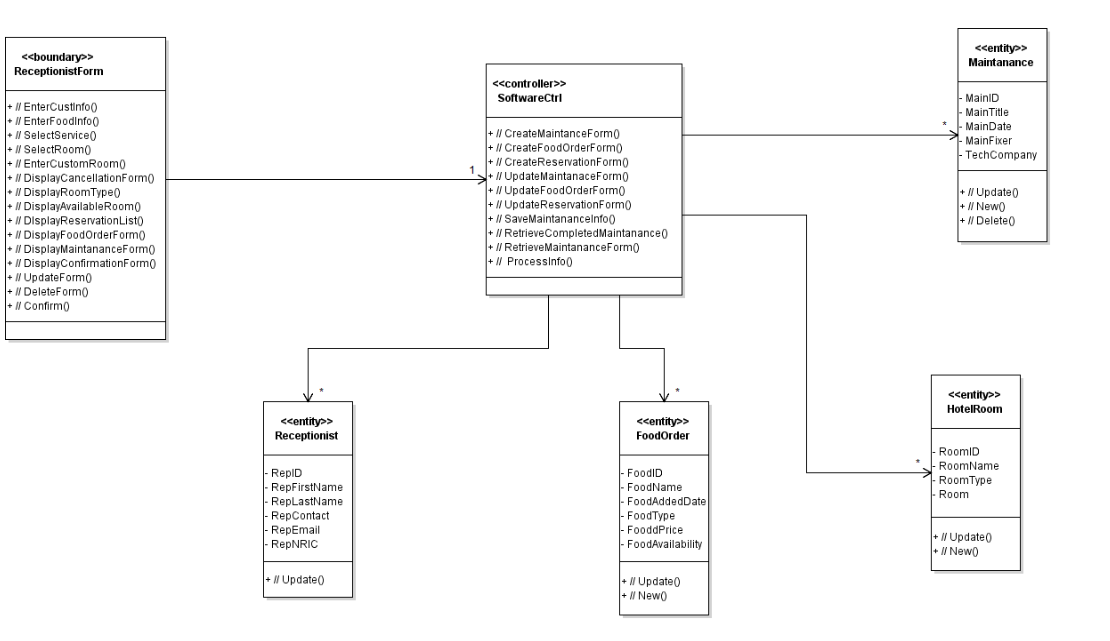
## What is VOPC Diagram?

VOPC Diagram are classes that are involved in the use case diagram. VOPC includes the attributes, operations and relationship of each class.

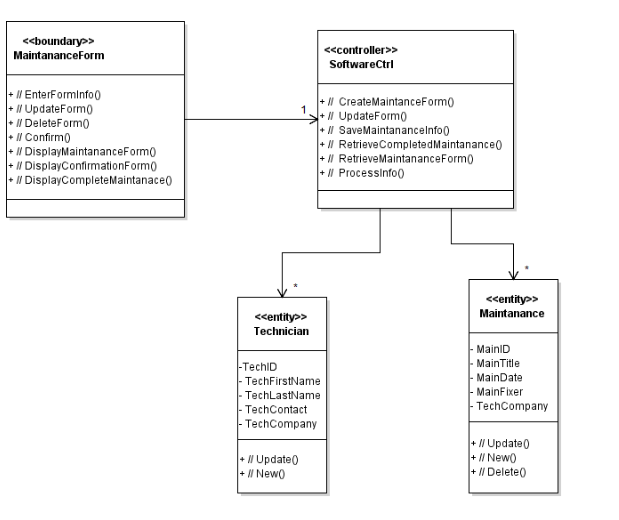
## Use Case

## VOPC Diagram

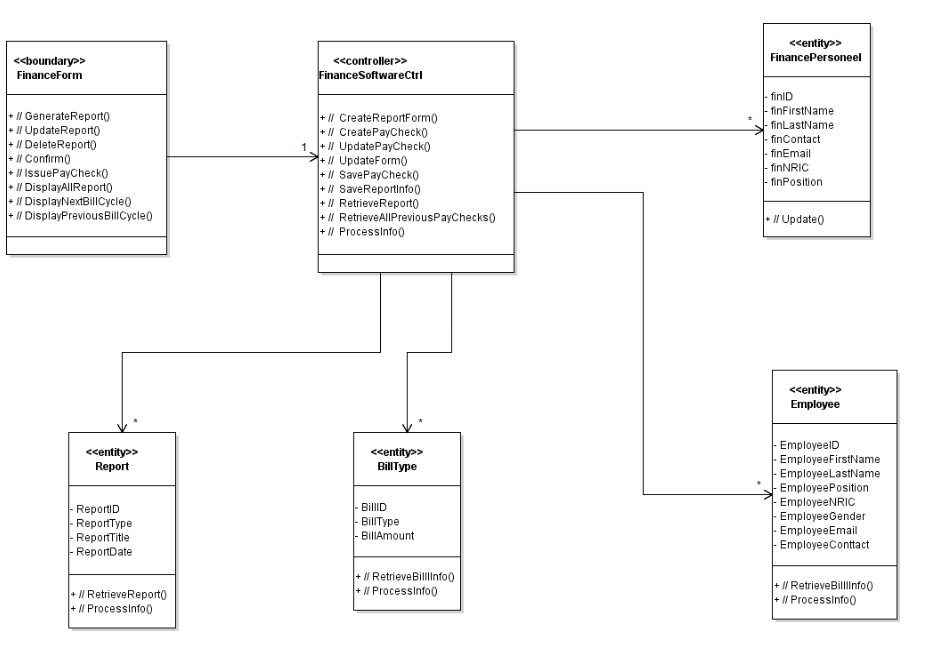
### Receptionist



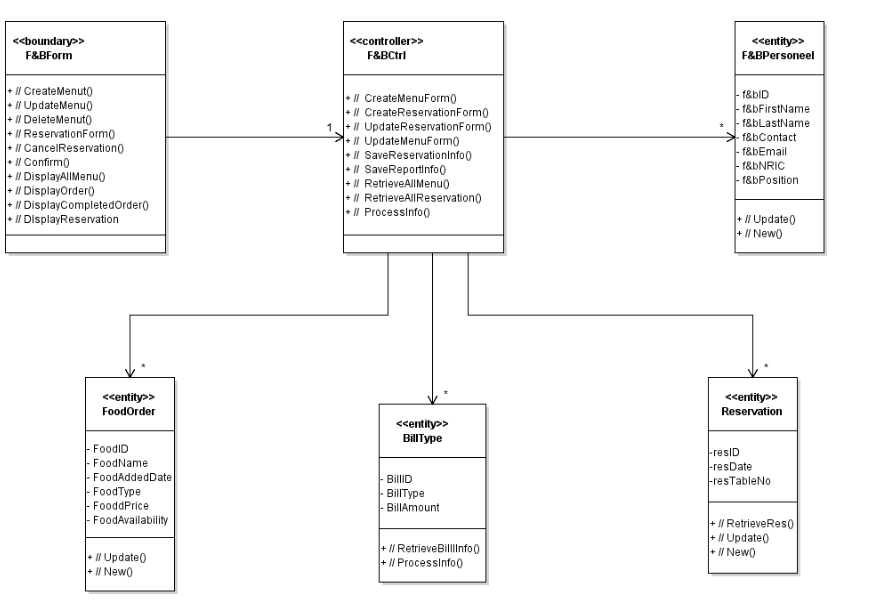
### Maintenance



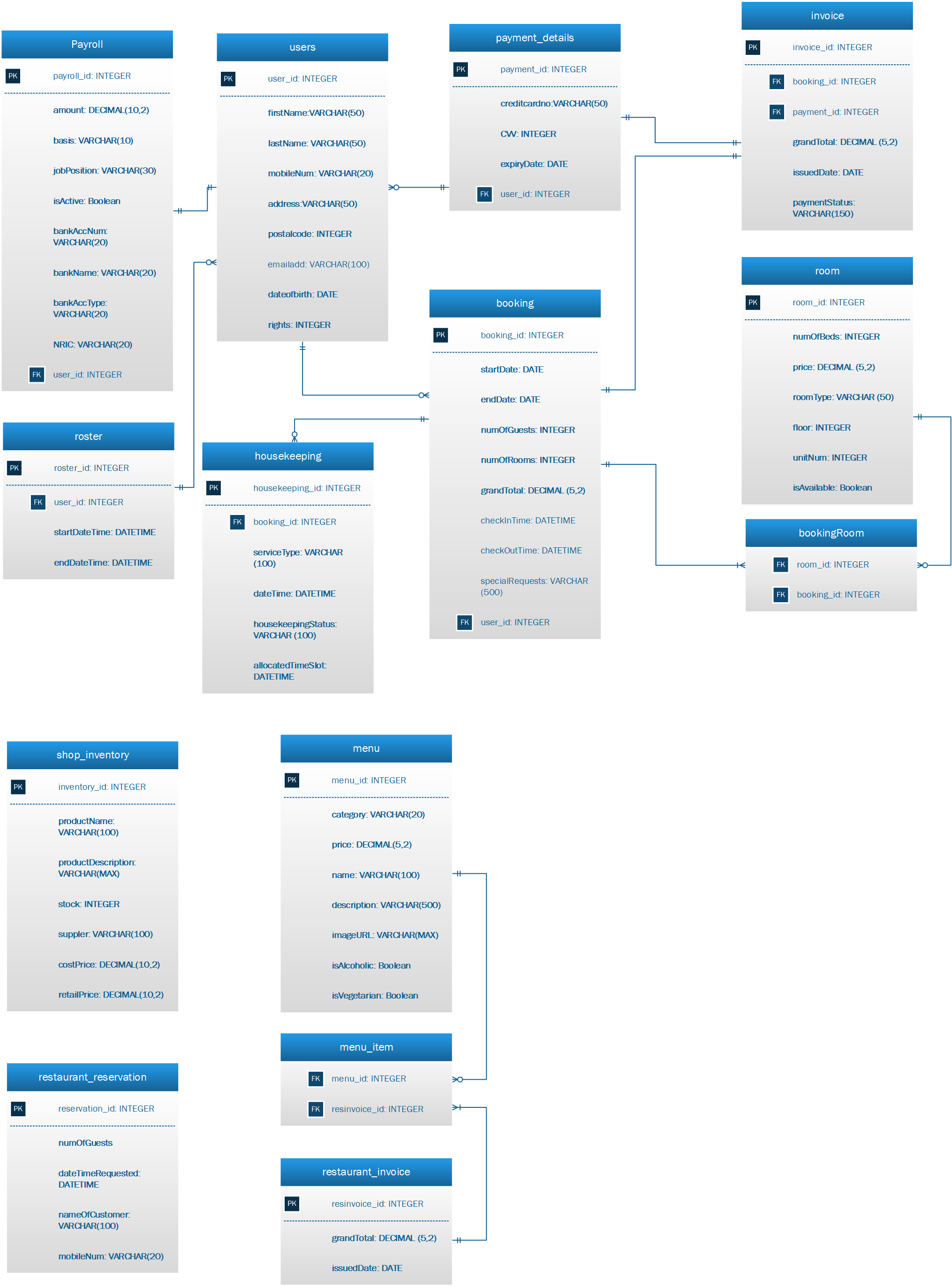
### Finance



### F&B



# C:\Users\AcerV7\Desktop\IT Year 3.1\SWEN\FUCKTHISSHIT.pngDATABASE DESIGN



## Considerations

Only one user table was implemented for both guests and staffs. To differentiate them, a “right” attribute was added. For instance the value “0” would be for guests, value “1” would be staffs. Using user\_id as the foreign key, in the payroll table, each staff will be further specified as what role they are playing in the system. For instance, “Finance Manager” or “Receptionist”.

Since there is only one user table that differentiates the staff from the guests, more details of staff is added through the payroll table. The payroll table specifies the position of the staff, amount payable, and all necessary and significant attributes needed for the finance department to issue a payroll.

When constructing the invoice tables, we realized that if we simply connects the booking table to the invoice table, whenever there is an update in the booking table, the changes will be brought over to the invoice table which is unacceptable. Hence, we added a static attribute to the invoice table which wouldn’t be changed even if there is increase or decrease in price. However, this is only for the past transactions. Every future invoice generated would take the updated attribute value.

Taking into consideration that one booking may include multiple rooms, we created a “Booking room” table. One booking can have multiple booking rooms, with this table instead of directly connecting one booking to one room. As only one foreign key can be added to the booking table, if the “booking room” table does not exist, users cannot book more than 1 room per booking.

The database is constructed to ensure that all requirements of Mr. Wang is met. The database is convenient for retrieving data due to the unified platform. Hence, integrating the different operational systems for Mr. Wang to monitor.

Every attributes in each tables plays its part as they are crucial. For instance, in the “room” table, there is an attribute “isAvailable”. The attribute is a Boolean which helps front-desk updates the status of the room. There are other Boolean attributes that meets the update function requirements by Mr. Wang.

## Limitations

One limitation would be the cost as a process with high speed of data processing and memory of large size in required. Accumulated data over the years may result in the need for a larger memory space which may be costly. Also, budget is required to conduct staff trainings as staffs involved must be familiar with the system.

Another limitation would be database failure. In most organizations, most data is integrated into a single database. If database is corrupted due to power failure or corrupted on the storage media, valuable data may be lost.