



# **Metal Purchasing & Production Management System for Lokupitiya Enterprises**

H.A.R.P. Gunarathne

BIT Registration No: R141837

Index No: 1418378

Name of Supervisor:

Mr. S. M. D. D. Wickramasinghe

**December 2017**




**This dissertation is submitted in partial fulfilment of the requirement of  
the Degree of Bachelor of Information Technology (external) of the University  
of Colombo School of Computing**

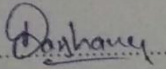
# DECLARATION

## DECLARATION

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a Degree or Diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due references is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organization.

Signature of candidate: ..... Date: 2017/11/02

Name of Candidate: H.A.R.P. Gunarathne.

Signature of Supervisor: ..... Date: 2017/11/02

Name of Supervisor: S.M. D.D. Wickramasinghe

# ABSTRACT

Lokupitiya Enterprises is a well-known metal distributor in Nawagamuwa area. The company is established in 1993. At present they provide various kinds of metals. Due to its fast growth, the company has become a very popular business place by now. But they still use a manual system to manage their day-to-day business activities such as customer order management, issue bills for sales etc. Because of manual system, it needs much effort and time to search historical data and decision making and view details on purchasing. So, the existing method for handling the business activities is problematic due to the manual methods.

This system will help to overcome existing problems like lack of efficiency, data redundancy and the system will be developed as a standalone system. Proposed system supports for tracking order management, user privilege management and proposed system can generate all kinds of useful reports. It will support management to achieve their business goals as well as they can be well informed about their business.

By using this system, a responsible person such as admin or manager can control the accessibility of information relevant to the users who deal with the system and will ensure the security of the information. This system is designed to fulfill all user and system requirements which will also assist all the employees in the business place to work efficiently and maximize their profit.

The proposed system follows MVC architecture and foundation is JAVA language with OO techniques. OOAD was used for the designing phase, as Agile is the process model that is used for the proposed system. This solution will mainly be developed using java, JavaFX, MYSQL, Hibernate and Jasper reports.

# ACKNOWLEDGEMENT

I would like to take this space to acknowledge and extend my heartiest gratitude to those who have helped me in several ways throughout the project work to make this project a success.

First and foremost, I owe my deep gratitude to the University of Colombo School of Computing for offering us this precious degree program and all its staff who guided me from the beginning.

A very special recognition should be given to my project supervisor Mr. S.M.D.D. Wickramasinghe for the extensive assistance and the guidance, if his support is not received the completion of this project would have been extremely complicated.

I also take this opportunity to thank Mr. Ranjith Premalal Lokupitiya, Mr. Isuru Darshana Lokupitiya, Ms. Isharika Lakmali Lokupitiya and all the employees of Lokupitiya Enterprises, who gave me the opportunity to develop this system for their company. All the members gave me an enormous support to complete the project successfully.

It is my duty to thank Mr. R.D.D. Suranga the Managing director Mr. Susith Sanasuma, Administrator, Lecturer panel and all the staff at Earth University College, Gampaha for giving me the academic knowledge for BIT degree program and allowing me to use the college library throughout the period. Also, I honestly thank all my friends at Earth Institute and specially express my gratitude to my well-educated lecturers.

Finally, I thank my family members for their unconditional love and support given in every way possible throughout the process of this degree program for three years

# TABLE OF CONTENTS

DECLARATION .....	i
ABSTRACT .....	ii
ACKNOWLEDGEMENT .....	iii
LIST OF ACRONYMS .....	xi
CHAPTER 01 – INTRODUCTION .....	1
1.1 BACKGROUND .....	1
1.2 MORTIVATION FOR THE PROJECT .....	2
1.3 OBJECTIVES AND SCOPE .....	2
1.3.1 OBJECTIVES .....	2
1.3.2 SCOPE OF THE PROPOSED PROJECT .....	2
1.4 OUTLINE OF THE REMAINING CHAPTERS .....	3
CHAPTER 02 – ANALYSIS .....	4
2.1 REQUIREMENT GATHERING .....	4
2.1.1 INTERVIEWS .....	4
2.1.2 READING COMPANY DOCUMENTATION .....	4
2.1.3 OBSERVATION .....	5
2.2 EXISTING SYSTEM .....	5
2.2.1 WEAKNESSES OF CURRENT MANUAL SYSTEM .....	6
2.3 OUTLINE OF EXISTING SIMILAR SOLUTIONS .....	7
2.3.1 LAK COCONUT PRODUCERS’ COCONUT PURCHASING AND SALES MANAGEMENT SYSTEM .....	7
2.3.2 SKYWARE INVENTORY .....	7
2.4 FUNCTIONAL REQUIREMENTS OF THE PROJECT .....	8
2.5 NON-FUNCTIONAL REQUIREMENTS OF THE PROJECT .....	9

2.6 SOFTWARE PROCESS MODELS .....	10
2.6.1 WATERFALL MODEL.....	10
2.6.2 PROTOTYPING MODEL .....	10
2.6.3 RAPID APPLICATION DEVELOPMENT (RAD) .....	10
2.6.4 RATIONAL UNIFIED PROCESS (RUP).....	10
2.7 SELECTED METHODOLOGY OF THE SYSTEM .....	12
2.8 RELEVANT DIAGRAMS FOR THE SELECTED METHODOLOGY .....	12
2.8.1 USE CASE DIAGRAM .....	12
2.8.2 USE CASE DESCRIPTION .....	12
CHAPTER 03 – DESIGN .....	15
3.1 ALTERNATIVE TECHNICAL SOLUTION EVALUATION.....	15
3.1.1 STANDALONE SYSTEM .....	15
3.1.2 WEB BASED SYSTEM .....	15
3.1.3 REASONS TO CHOOSE STANDALONE SYSTEM.....	15
3.2 SELECTED SOLUTION DESCRIPTION AND JUSTIFICATION .....	16
3.3 RELEVANT DESIGN DIAGRAMS .....	16
3.3.1 STRUCTURE OF THE SYSTEM .....	17
3.3.2 USE CASE DIAGRAM .....	18
3.3.3 ACTIVITY DIAGRAM .....	23
3.3.4 CLASS DIAGRAM.....	24
3.3.6 SEQUENCE DIAGRAM .....	29
3.3.7 ENTITY RELATIONSHIP MODEL.....	30
3.4 USER INTERFACE DESIGN .....	31
CHAPTER 04 – IMPLEMENTATION .....	36
4.1 IMPLEMENTED ENVIRONMENT .....	36
4.1.1 HARDWARE REQUIREMENTS .....	36
4.1.2 SOFTWARE REQUIREMENTS.....	36

4.2 DEVELOPMENT TOOL.....	37
4.2.1 NETBEANS 8.2 .....	37
4.2.2 JAVA LANGUAGE.....	37
4.2.3 MYSQL.....	38
4.2.4 JAVAFX SCENE BUILDER.....	38
4.2.5 VISUAL PARADIGM .....	38
4.2.6 HIBERNATE .....	39
4.2.7 JASPER REPORT.....	40
4.3 CODE AND MODULE STRUCTURE DESCRIPTION .....	40
4.3.1 DATA LAYER IMPLEMENTATION .....	41
4.3.2 HIBERNATE CONFIGURATION.....	41
4.3.3 CREATE JAVA CLASSES .....	41
4.4 INTERFACE LAYER IMPLEMENTATION .....	44
4.5 CONTROL LAYER IMPLEMENTATION .....	49
4.5.1 HIBERNATE SESSION .....	49
4.5.2 DAO (DATA ACCESS OBJECT).....	50
4.5.3 REUSED CODE MODULES .....	51
CHAPTER 05 – EVALUATION .....	53
5.1 TEST STRATEGIES.....	53
5.1.1 UNIT TESTING .....	53
5.1.2 INTEGRATED TESTING .....	54
5.1.3 SYSTEM TESTING.....	54
5.2 TEST PLAN .....	54
5.3 SYSTEM TEST CASES .....	55
5.3.1 TEST CASES AND TEST RESULTS FOR USER MODULE .....	55
5.3.2 TEST CASE FOR LOGIN MODULE.....	57
5.4 USER EVALUATION .....	58

CHAPTER 6- CONCLUSION .....	60
6.1 LESSON LEARNT .....	61
6.2 FUTURE IMPROVEMENTS .....	61
REFERENCES .....	62
APPENDIX-A - SYSTEM DOCUMENTATION .....	64
APPENDIX B - DESIGN DOCUMENTATION.....	70
APPENDIX C – USER DOCUMENTATION .....	76
APPENDIX D – MANAGEMENT REPORTS .....	79
APPENDIX E – TEST RESULTS .....	83
APPENDIX F – CODE LISTING.....	87
APPENDIX G - CLIENT CERTIFICATION.....	92
GLOSSARY .....	93



# LIST OF FIGURES

Figure 2. 1: Existing system use case .....	6
Figure 2.2: Dashboard Skywear inventory .....	7
Figure 2.3: Rapid Unified Process Model .....	11
Figure 2.4: Use case diagram for login .....	13
Figure 2.5: Use case diagram for add new purchase order.....	14
Figure 3.1: Structure of the system.....	17
Figure 3.2: Use case diagram for administrator of the system .....	18
Figure 3.3: Use case diagram for Accountant of the system.....	19
Figure 3.4: Use case diagram for General Manager of the system.....	20
Figure 3.5: Use case diagram for Supervisor of the system .....	21
Figure 3.6: Use case diagram for administrator and general manager .....	22
Figure 3.7: Activity Diagram for login to the system .....	23
Figure 3.8: Class diagram for Purchase Order .....	24
Figure 3.9: Class diagram for employee management .....	25
Figure 3.10: Class Diagram for Vehicle Management .....	26
Figure 3.11: Class Diagram for Delivery Management .....	27
Figure 3.12: Class Diagram for invoice management .....	28
Figure 3.13: Sequence diagram for add employee .....	29
Figure 3.14: Entity Relational Diagram of the System .....	30
Figure 3.15: Main Window User Interface.....	31
Figure 3.16: Main Window UI .....	32
Figure 3.17: Supplier Management UI.....	33
Figure 3.18: Invoice Management UI.....	33
Figure 3.19: Purchase Order UI.....	34
Figure 3.20: User Management UI.....	34
Figure 3. 21: Privilege Management UI.....	35
Figure 3.22: Item Management UI .....	35
Figure 4.1: Shows Architecture of Hibernate Framework .....	39
Figure 4.2: Employee Management FXML Scene Builder .....	44
Figure A. 1: Installing Java Runtime.....	65
Figure A. 2: Installation process of MYSQL server 5.5.....	66

Figure A.3: MySQL server instance configuration wizard .....	67
Figure A.4: Setup wizard for MySQL tools .....	68
Figure A.5: MySQL server connection portal .....	69
Figure B.1: Activity Diagram for add a new user .....	70
Figure B.2: Sequence Diagram for add new User .....	71
Figure B.3: Admin Adds a User - Use Case Diagram.....	72
Figure B. 4: MySQL database tables.....	73
Figure B.5: Invoice Table.....	74
Figure B.6:Vehicles Table.....	74
Figure B.7: PO Table.....	74
Figure B.8: Privilege table.....	75
Figure B.9: Employee Table.....	75
Figure B.10:Customer Table .....	75
Figure C.1: Login window and login success window.....	76
Figure C.2: The Three Buttons located on the right top of the system .....	77
Figure C.3: Forget Password UI.....	77
Figure C. 4: Main Window UI .....	78
Figure C.5: Three Main Categories .....	78
Figure D.1: Due Payment Report .....	79
Figure D.2: vehicle Detail report.....	80
Figure D.3 :Sales report.....	81
Figure D.4: Monthly profit report.....	82
Figure G.1: Client Certification.....	92

# LIST OF TABLES

Table 2.1: Use case diagram for login.....	13
Table 2.2: Use case diagram for add new purchase order.....	14
Table 5.1: Test results for user module.....	56
Table 5.2: Test case for login module. ....	57
Table B.1: Use case narrative of Add a new user to the system.....	72
Table E.1: Test Case and Result for Login Module.....	84
Table E. 2: Test Case and Result for Purchase Order Module.....	85
Table E. 3: Test Case and Result for Invoice Module.....	86

# **LIST OF ACRONYMS**

BIT - Bachelor of Information Technology

IDE - Integrated Development Environment

MVC - Model View Controller

OOD - Object Oriented Design

OOP - Object Oriented Process

RAD - Rapid Application developments

RUP - Rational Unified Model

SQL - Structured Query Language

UML - Unified Modeling Language

# CHAPTER 01 – INTRODUCTION

This chapter introduces the Metal Purchasing and Distribution System for Lokupitiya Enterprises with detail description on what its background, and what its need for the project and motivation objectives and scope.

## 1.1 BACKGROUND

Lokupitiya Enterprises is a leading metal distributor in Nawagamuwa area. That Company product range consists all types of metals. It has been there for more than 34 years now. By now they have got so many customers around Nawagamuwa and other areas around it.

There are more than 200 employees working at Lokupitiya Enterprises and there are many machineries' and vehicles running so it's really difficult to manage such a big process manually.

At present Lokupitiya Enterprises manages their business functions in a non-automated method. And there are a lot of drawbacks like missing documents, lose track of payments, forget to deliver on time, as a result of that the company lost so much profit that they can easily gain due to those reasons. So, it has become a great urge to automate the system to enhance the competency and efficiency of their work, with easy maintenance and management of their day to day activities efficiently and effectively by reducing the paper work and keep track of every delivery and payment.

And with the proposed system they expect to see a great changes and improvements of the business and they hope to improve the system more and more if the project shows beneficial to the company.

## 1.2 MORTIVATION FOR THE PROJECT

- They use manual way to purchase orders, invoices, payments, transport details etc.
- Because of manual system, it needs much effort and time to search historical data and decision making on purchasing.
- Due to purchasing delays, wastage of metal is very high and cause delaying orders.
- Hard to take management decisions because manual system doesn't cover the report generating about process details.

## 1.3 OBJECTIVES AND SCOPE

### 1.3.1 OBJECTIVES

- Provide user friendly simple Graphical User Interface (GUI) system.
- Reduce time and human effort for purchasing, bill generating and payment process.
- Improve the daily, weekly, monthly and yearly statistical report generation about sales and purchase. This system will automate the processes of bill and report generating. And, system will notify about functioning situations.
- Improving reliability.
- Improving accuracy.
- Improving coloration in the management and users.

### 1.3.2 SCOPE OF THE PROPOSED PROJECT

- Mange metal purchasing process.
- Manage employee details.
- Manage supplier details.
- Manage order management details.

- Manage customer details.
- Manage invoices, payments, purchase order details
- Manage vehicle details of the company.
- Management level report generating.

## 1.4 OUTLINE OF THE REMAINING CHAPTERS

Analysis explains the analysis phase of the project. By explaining what kind of methods used to gather requirements and for a better understanding how the appropriate UML diagrams were used to identify requirements.

In systems design, the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage will describe the new system as a collection of modules or subsystems. The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced because of interviews, workshops, and/or prototype efforts. Design elements describe the desired system features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo-code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the system in sufficient detail, such that skilled developers and engineers may develop and deliver the system with minimal additional input design.

In the implementation chapter explains the hardware software requirements, development tools which is used for system, code features and reused existing codes of the system.

Evaluation describes all the testing methodologies, which were used during this project and the entire test cases with their outputs. This chapter includes an evaluation of the system throughout the development stage. It briefly explains additional areas to be considered and the problems encountered.

Conclusion is all about Lessons learnt by implementing this system with a brief description of the company and future improvements of the system are discussed in this chapter.

## **CHAPTER 02 – ANALYSIS**

This chapter will give an overview of the existing system, requirement analysis, requirement gathering techniques, functional and non-functional requirements and outline of existing similar solution.

### **2.1 REQUIREMENT GATHERING**

Requirements Analysis is the process of understanding the customer needs and expectations from a proposed system or application and is a well-defined stage and the most important part of the software development life cycle otherwise it is difficult to develop an excellent product which satisfies the client.

Interviews with the owner and system user and the analyzing old manual system is also helped a lot to gather requirements.

Following are the three techniques used in this project to gather requirements from different stakeholders. Such as suppliers, staff members, general manager...etc.

#### **2.1.1 INTERVIEWS**

This method is used to collect the information from the Managers, staff members and one of supplier those who are related to this process. In this method the analyst sits face to face with the people and records their responses. So, it could be easy to gather details about difficulties that they are facing with current system. Factory officer and management staff gave the information regarding the financial system as well as the supplier registration system.

#### **2.1.2 READING COMPANY DOCUMENTATION**

By reading the existing company document is one of the best techniques to get a clear idea of what the real process is. With the permission of Relevant managers went through the



relevant documents and collected some information regarding the current system. Some of usable documents are mentioned below.

1. Organizational chart.
2. Consumer registration form.
3. Payment invoices and supplier detail forms.
4. Existing transaction bills and sales records.
5. Existing Fertilizer stock records.
6. Vehicle usage documents.

### 2.1.3 OBSERVATION

This is a skill which the analysts must develop. Identified the right information and choose the right person and looked at the right place to achieve those implemented objectives. There should be a clear vision of how each department work and workflow between them. By visiting the site, we can gain knowledge about the factory environment and process flow.

## 2.2 EXISTING SYSTEM

Now, all their day to day transactions are done by manually. When the customer purchases products he will receive a hand-written bill and cashier keeps carbon copy of that bill. Supplier details are recorded in a “supplier information book”. All transaction calculations are done manually by using calculators. All transaction details are recorded in papers and they are stored in their relevant files.

Figure 2.1 shows the existing system use case which used to get a clear idea about the current working process of the system.



Figure 2. 1: Existing system use case

## 2.2.1 WEAKNESSES OF CURRENT MANUAL SYSTEM

- All documents are hand written and therefore huge filing system
- Data redundancy.
- Difficult to find employee, supplier and consumer details.
- Data inaccuracy
- No better vehicle management process.
- Management staff not works in happily.
- Time wasting because of lot of documentations.
- Difficulties of payment process.
- Decision making difficulties due to unavailable of timely reports and information.

## 2.3 OUTLINE OF EXISTING SIMILAR SOLUTIONS

### 2.3.1 LAK COCONUT PRODUCERS' COCONUT PURCHASING AND SALES MANAGEMENT SYSTEM

The system mainly manages three areas that are coconut purchasing and sales. And other areas relevant to the above three functions such as employee management, member registration and order management. In additionally this system will cover report generating, providing relevant graphs and notifications.

The system was developed according to the object-oriented techniques. Unified Modeling Language was used in the analysis and design phases. This system is made by using javaFX and MySQL as a standalone software.

This system is implement in Lak coconut main branch allocated in Gampaha and all the sales and purchasing manage by this system.

### 2.3.2 SKYWARE INVENTORY

Skyware Inventory is an inventory management System that provides a lot of services such as Transaction Management, Item Management, User Account Managements and more [10].

Figure 2.2 Shows the mentioned system.

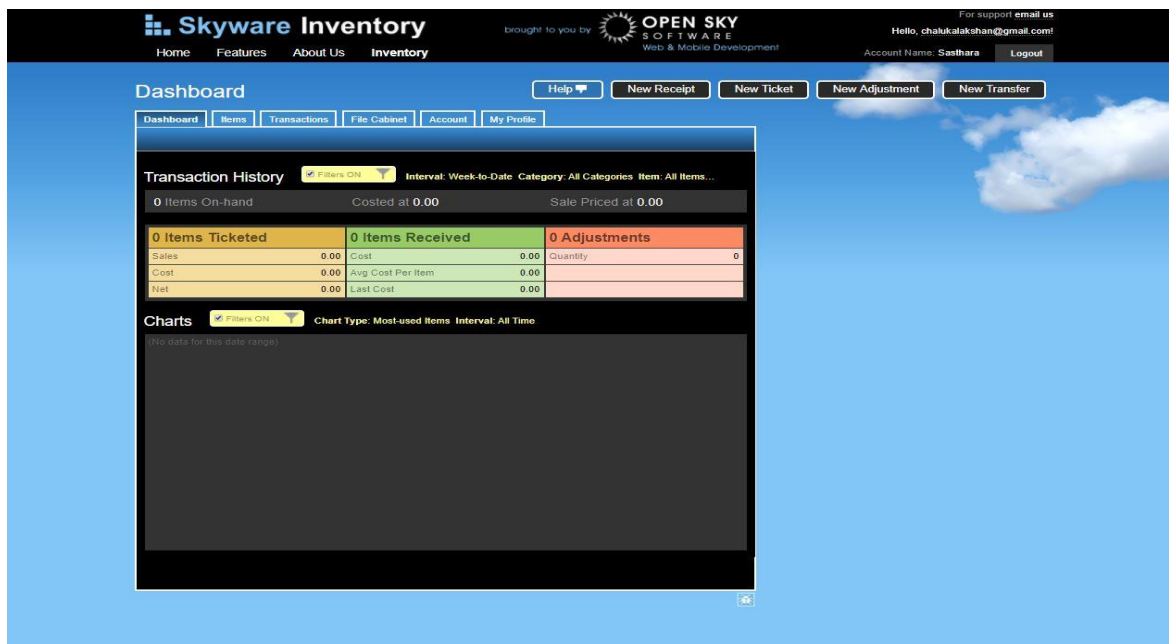


Figure 2.2: Dashboard Skywear inventory

## 2.4 FUNCTIONAL REQUIREMENTS OF THE PROJECT

The system should notify availability of vehicles and other critical situations. Functional requirement may be calculations, technical details, data manipulation and processing that define what system is supposed to accomplish. And, it describes the requirement or services that system should offer for its end users. Most of them are provided by users of the system. Below list shows functional requirement gathered up to now.

- Manage metal purchasing process.

Purchasing metals is one of the main function of the company. So, this is a very critical function that need to priorities and well maintained throughout the process.

- Manage employee details.

Employee management is one the essential part of the system because the company need to keep a record of the employees that working.

- Manage supplier details.

Suppliers are very important part of the company and it must be well managed. So, the supplier details must be including in the project.

- Manage consumer details.

Consumers are the people who buy metals from the company. The purchased details should be stored so the company can keep the track so they can get a idea about the totals losses and profits.

- Manage vehicle details of the factory.

The company delivers the metals to the customers if needed. So, they need to have a clear idea about the vehicles that available so they can manage their deliveries much efficiently.

- Management level report generating.

Reports playing a very important part of any system. By using the data from the system, the management can generate various reports that can be useful to the management. So, the system must have the ability to generate the reports that need to top management to analyses the business.

## 2.5 NON-FUNCTIONAL REQUIREMENTS OF THE PROJECT

Non-functional requirement describes the features that system should have. These are constraint on the services or functions offered by the system. It is very difficult to manage some of requirements are listed below.

- User Friendliness

User Friendliness is very important for the system because the system users must be able to use the system with less effort with accuracy for maximize the productivity.

- Manageability

The system should have the ability to manage by the users so it can give the best out of the system for a long period of time.

- Easy to install

The system should be easy to install with less efforts.

- Efficiency

When using the system, the must be a very good efficiency so the users can give a best result out of them and be able to accomplish something with the least waste of time and effort.

- Security

Security is something that every system must have because the data can be altered by users.

- Accuracy and consistence

These are very important non-functional requirement that should be considered when storing the details of customer, purchase, billing etc.

- Usability

Usability requirement has been achieved by using various techniques such as easy menu navigation, attractive interfaces and use of matching colors.

## 2.6 SOFTWARE PROCESS MODELS

There are various Software development models or methodologies. They are as follows:

- Waterfall model
- Prototyping model
- Rapid Application Development (RAD)
- Rational Unified Process (RUP)

### 2.6.1 WATERFALL MODEL

The waterfall model is a sequential software development model in which development is flowing steadily downward (like a waterfall) through the phases of requirements analysis, design, implementation, testing(validation), integration, and maintenance. So, this model can be use where requirement is clear and well understand [11].

### 2.6.2 PROTOTYPING MODEL

A prototype is a working model that is functionally equivalent to a component of the product. A prototype (a small version of the system) can be used to clear the vague requirements. A prototype should be evaluated with the user participation. There are two types of Prototyping techniques. Throw-away Prototyping and Evolutionary Prototyping [12].

### 2.6.3 RAPID APPLICATION DEVELOPMENT (RAD)

Rapid Application Development (RAD) is an incremental software development process model that emphasizes an extremely short development cycle. If requirements are well understood and project scope is constrained, the RAD process enables a development team to create a 'fully functional system' within very short time periods (e.g. 60 to 90 days) [13].

### 2.6.4 RATIONAL UNIFIED PROCESS (RUP)

RUP Activities emphasize the creation and maintenance of models rather than paper documents. The process focuses on the early development and baseline of software architecture (component based). The RUP places strong emphasis on building systems based

on a thorough understanding of how the delivered system will be used. In this model overall project lifecycle is broken down into following four phases and iteration [1].

- Inception – During this phase, want to establish the business case for the system and define the project’s scope. The business case includes success criteria, risks assessment, estimates of the resources needed and a phase plan.
- Elaboration – During this phase the problem domain analysis is made, and the architecture of the project gets its basic form. At the end of the elaboration phase, can examine the detailed system objectives and scope, the choice of architecture, and the resolution of major risks.
- Construction – During the construction phase, all remaining components and application features are developed and integrated into the product, and all features are tested thoroughly.
- Transition – During the transition phase, deploy the software to the user community or to the working environment.

In figure 2.3 it shows how the overall project lifecycle is broken down into following four phases and iterations against time.

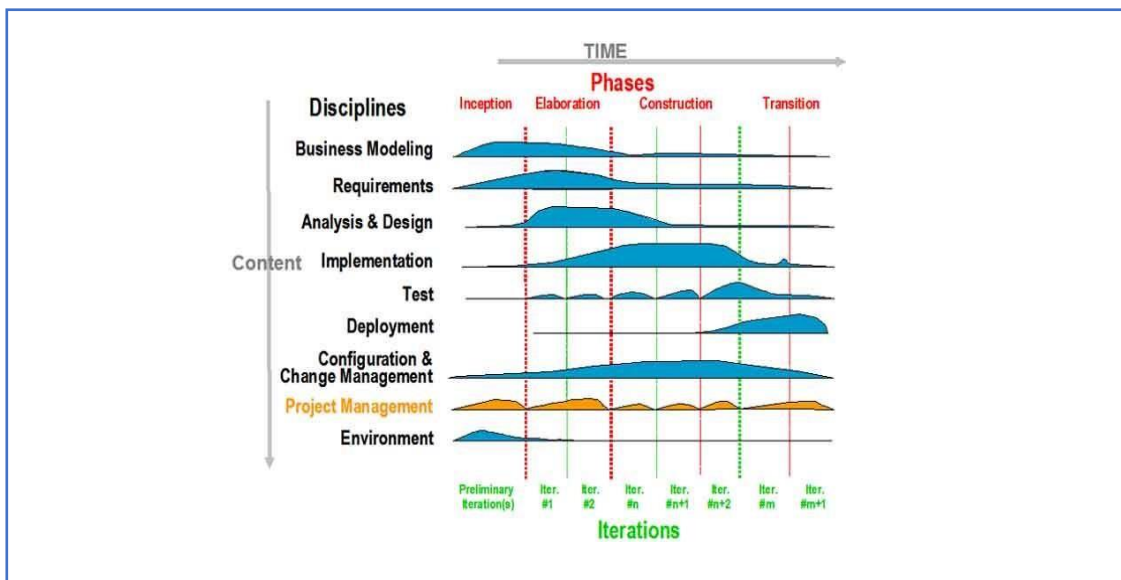


Figure 2.3: Rapid Unified Process Model

## 2.7 SELECTED METHODOLOGY OF THE SYSTEM

RUP (Rational Unified process) has been selected for the proposed system. Reasons of selecting RUP are it is an iterative software development process framework and it supports object-oriented development. Initially client does not have a clear idea about system requirements, so requirements may likely to be changed. Further system has been divided into modules such as Purchase module, Production Module, Reports and transport module etc. Each module has developed incrementally and iteratively. Therefore, RUP is the most appropriate SDLC (Software Development Life Cycle) methodology.

## 2.8 RELEVANT DIAGRAMS FOR THE SELECTED METHODOLOGY

### 2.8.1 USE CASE DIAGRAM

By analyzing the gathered requirements, the behavior of the system was documented in a use case which shows the functions and stakeholders of the system. Using this diagram, one can have a quick and clear idea about the overall system. Stakeholders of the system and their requirements can be identified separately.

Identifying the stakeholders of the system is very critical. Identified stakeholders of this system is following

- Manager
- System Owner
- Accountant
- Supervisor

### 2.8.2 USE CASE DESCRIPTION

Table 2.1 shows the use case for login to the system when user interact with the system



<b>Use Case</b>	Login to the System
<b>Actors</b>	All Users
<b>Overview</b>	
Authorized users who have the privilege to access the system	
<b>Preconditions</b>	
User must have an user account User must enter valid username and password	
<b>Flow Events</b>	
1. The user enters user name and password 2. If entry is invalid, the user is redirected to the login form with an error message 3. If entry is valid the system redirects the user to Main Window	
<b>Post Conditions</b>	
User will be redirected to Main Window or back to the Login form it depends on username and password	

Table 2.1: Use case diagram for login

Figure 2.4 shows the use case diagram for logging to the system as a user.

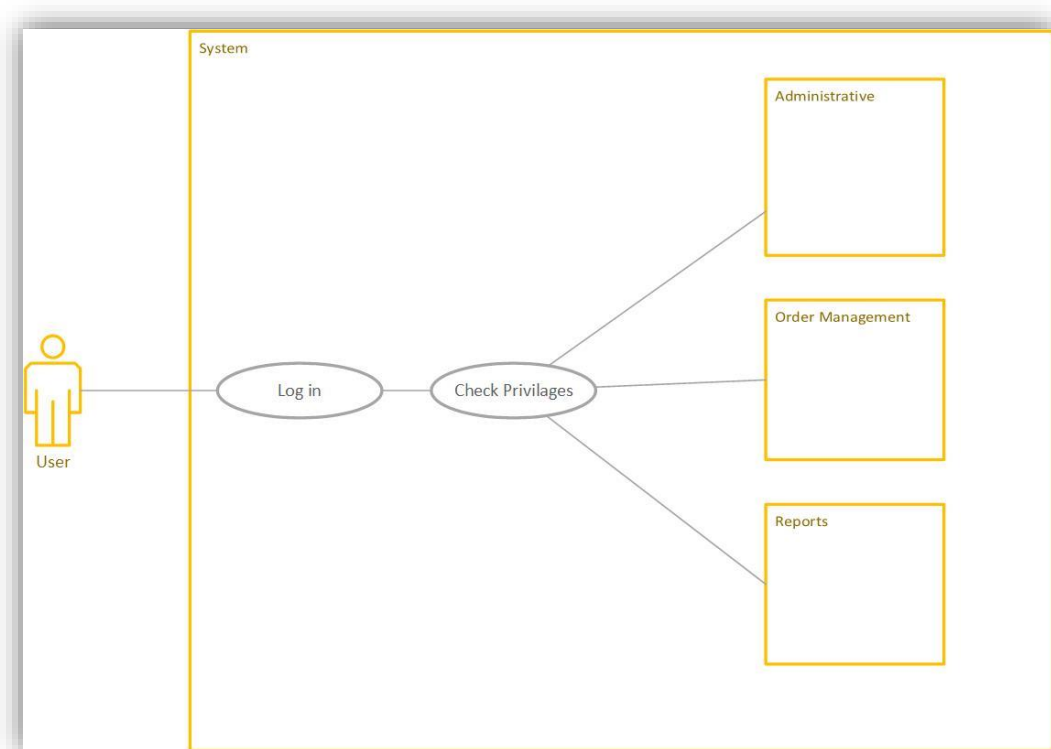


Figure 2.4: Use case diagram for login

Table 2.2 shows the use case Description table for add new purchase order by a user in the system

<b>Use Case</b>	Add new purchase order
<b>Actors</b>	System Administrator
<b>Overview</b>	
Purchase order details are added to the system	
<b>Preconditions</b>	
User must login to the system User must have the privilege to create new purchase order. Relevant supplier, relevant quantities are required by the system.	
<b>Flow Events</b>	
1. Select the supplier 2. Select the needed items. 3. Save Purchase order.	
<b>Post Conditions</b>	
New purchase order details add into the system	

Table 2.2: Use case diagram for add new purchase order

Figure 2.5 is the use case diagram for add new purchase order

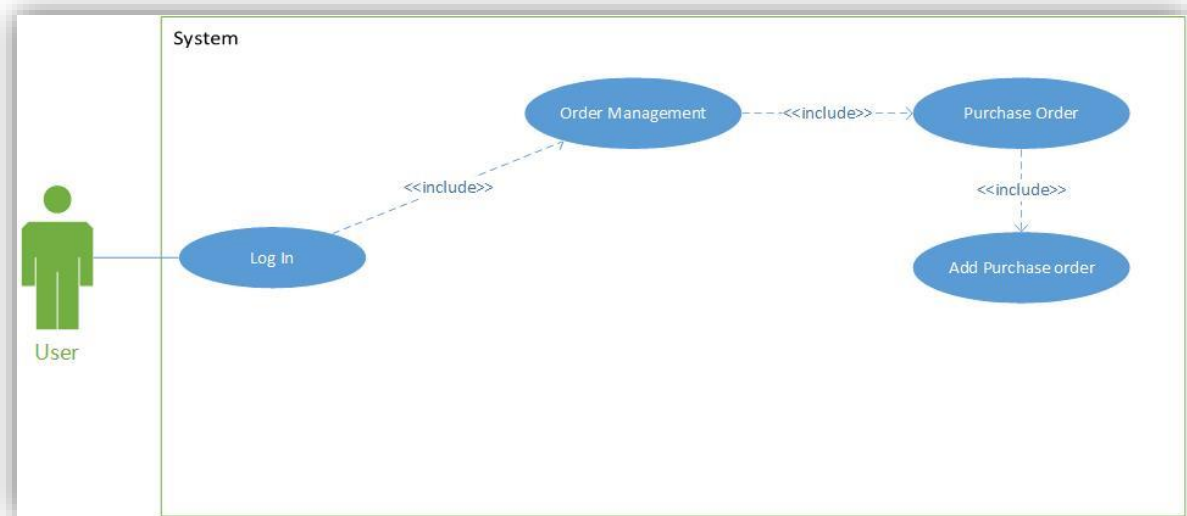


Figure 2.5: Use case diagram for add new purchase order

## **CHAPTER 03 – DESIGN**

This chapter describes the different technical solutions for the gathered requirements in Analysis Phase. UML diagrams such as class diagram, activity diagram and sequence diagram are used in here. And, it includes user interfaces.

### **3.1 ALTERNATIVE TECHNICAL SOLUTION EVALUATION**

There are two different alternate solutions identified for the Inventory Control system they are as follows:

- Standalone system with centralized database
- Web based system

#### **3.1.1 STANDALONE SYSTEM**

Software that is not a part of some bundled software. A program that is run as a separate computer process, not an add-on of an existing process. Standalone program, a program that does not require operating system's services to run.

This program can be compiled and can be run as a standalone program from command line or from within an IDE.

#### **3.1.2 WEB BASED SYSTEM**

Web based system refers to a program that runs with the help of the internet. The deployment, updating, maintenance processes are time consuming. In the case of a network failure the system is unavailable.

#### **3.1.3 REASONS TO CHOOSE STANDALONE SYSTEM**

- The client particularly requested for a standalone system.
- System would be platform independent.
- Maintenance is easy.
- Easy deployment.

## 3.2 SELECTED SOLUTION DESCRIPTION AND JUSTIFICATION

From those systems, standalone system is suitable for this project. After discussion with the client, he also preferred to a stand-alone system. Some of advantages are mentioned below over web-based system.

- It is easy to develop and easy to maintain.
- The factory capacity is not much high.
- Can be implementing Hibernate framework.
- Not necessary to have intranet or networking.
- Easy deployment.
- System would be platform independent so that it's better than a standalone system.
- Can be perform much faster than web-based systems.
- Low cost.
- Can have a total control over standalone applications and protect it from various vulnerabilities.
- Less time consuming.

## 3.3 RELEVANT DESIGN DIAGRAMS

These object models were used for the designing process of the Metal purchasing and production system.

### 3.3.1 STRUCTURE OF THE SYSTEM

Figure 3.1 shows the whole structure of the current system the and whole project mainly separated into three parts, then each part has its own processes that can be used by a valid user.

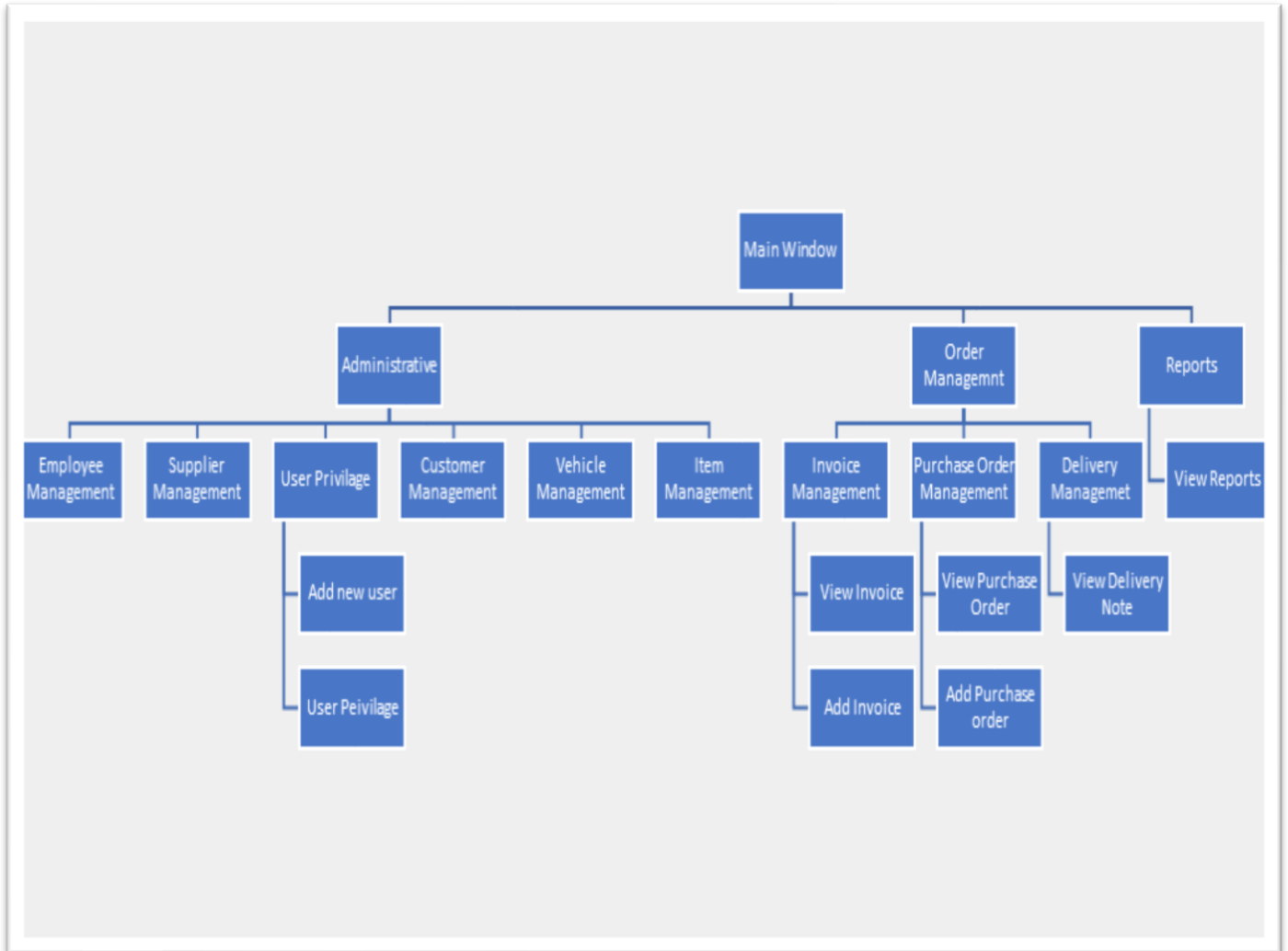


Figure 3.1: Structure of the system

### 3.3.2 USE CASE DIAGRAM

By analyzing the gathered requirements, the behavior of the system was documented in a use case which shows the functions and stakeholders of the system. Using this diagram, one can have a quick and clear idea about the overall system. Stakeholders of the system and their requirements can be identified separately. Identifying the stakeholders of the system is very critical. Identified stakeholders of this system is following

- Administrator
- Manager
- System Owner
- Supervisor

In the figure 3.2 it shows the use case of the administrator accessible modules of the system and the administrator has all privileges to the system.

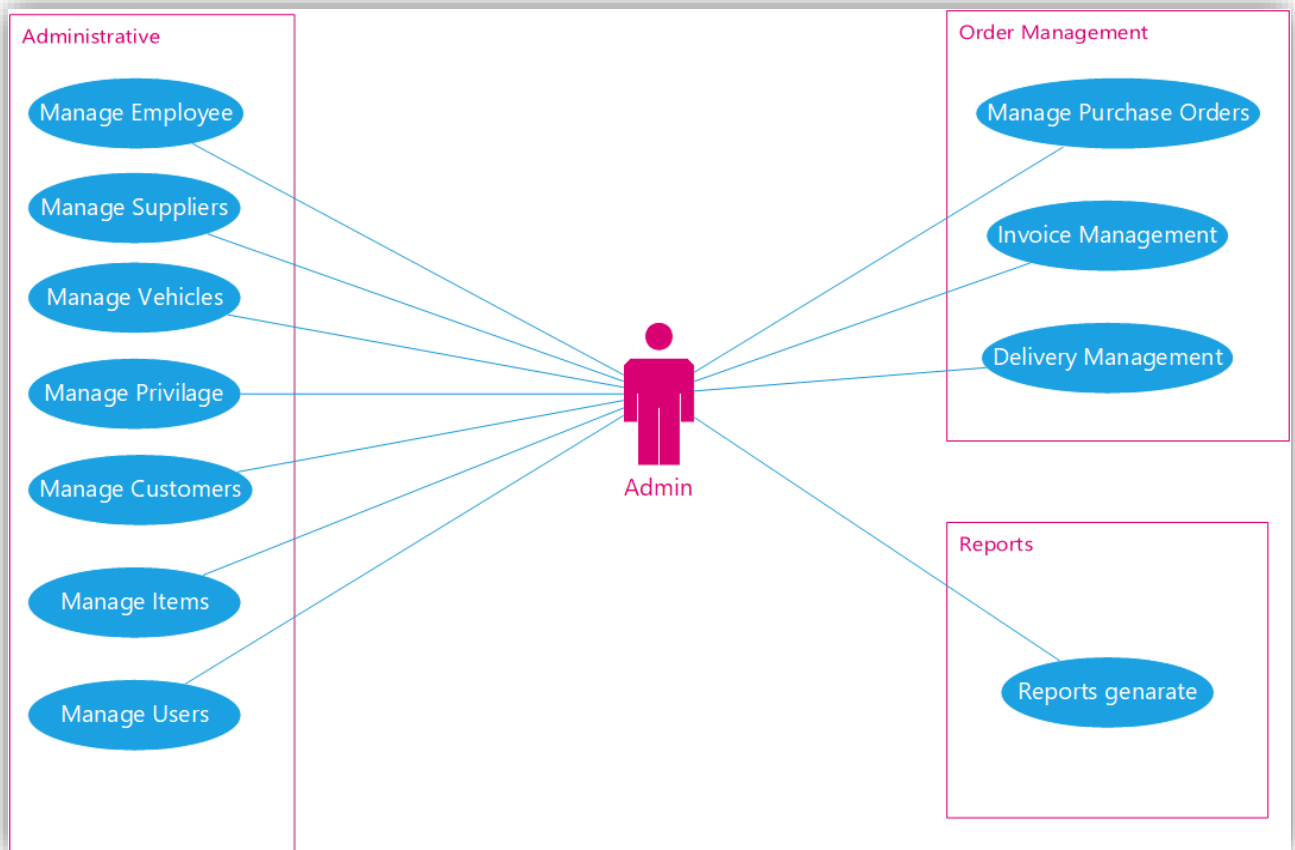


Figure 3.2: Use case diagram for administrator of the system

In the figure 3.3 it shows the use case for the Accountant accessible modules of the current system. The accountant can access the all the modules that are in order management group and the customer management in Administrative group.

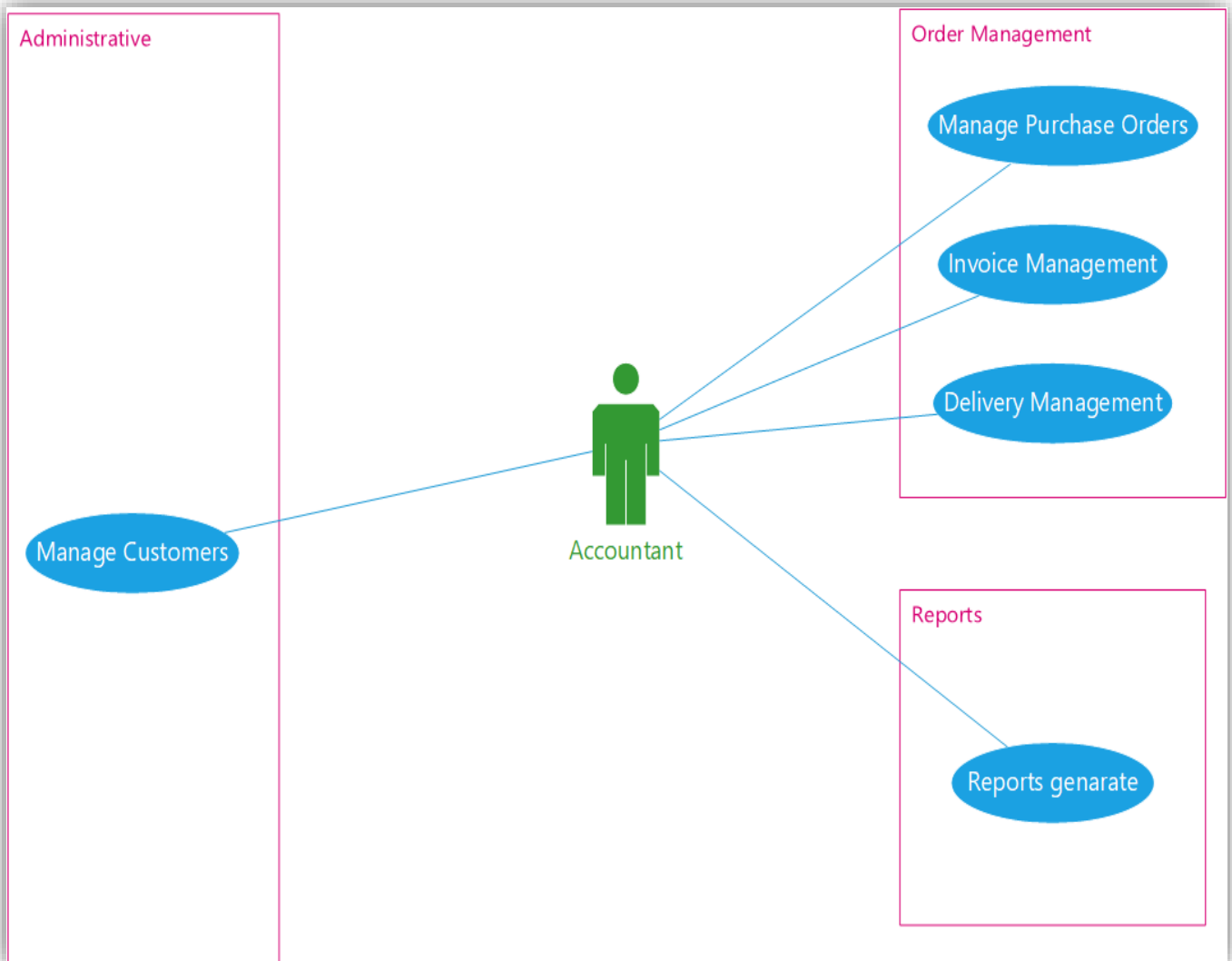


Figure 3.3: Use case diagram for Accountant of the system

In the figure 3.4 shows the use case for the general managers' accessible modules of the system and like administrator has all privileges to the system general manager can't give user privileges.

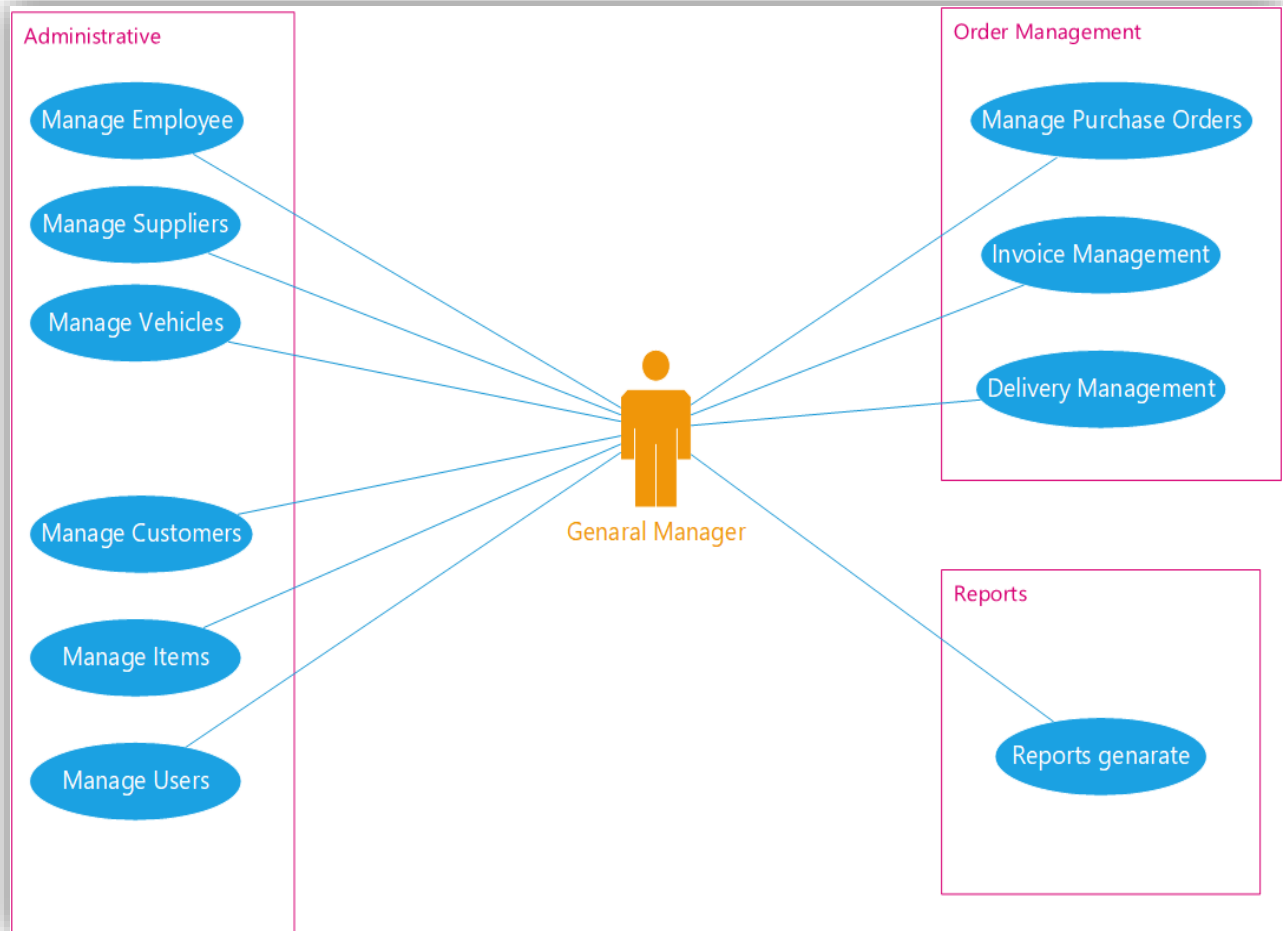


Figure 3.4: Use case diagram for General Manager of the system



In the figure 3.5 it shows the use case for the supervisors' accessible modules of the system.

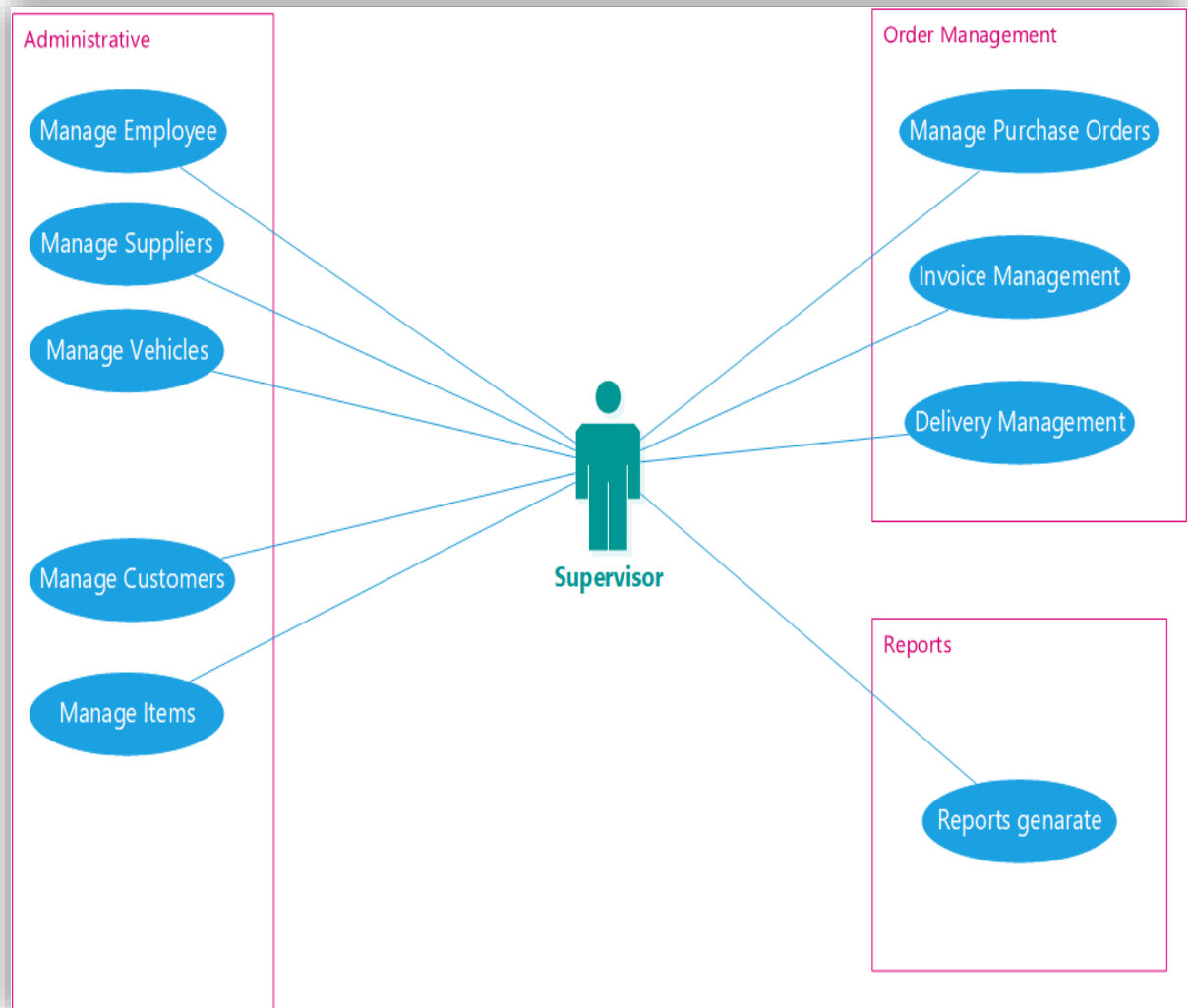


Figure 3.5: Use case diagram for Supervisor of the system

Figure 3.6 is a high-level use case diagram that shows the interaction of the administrator and the general manager.

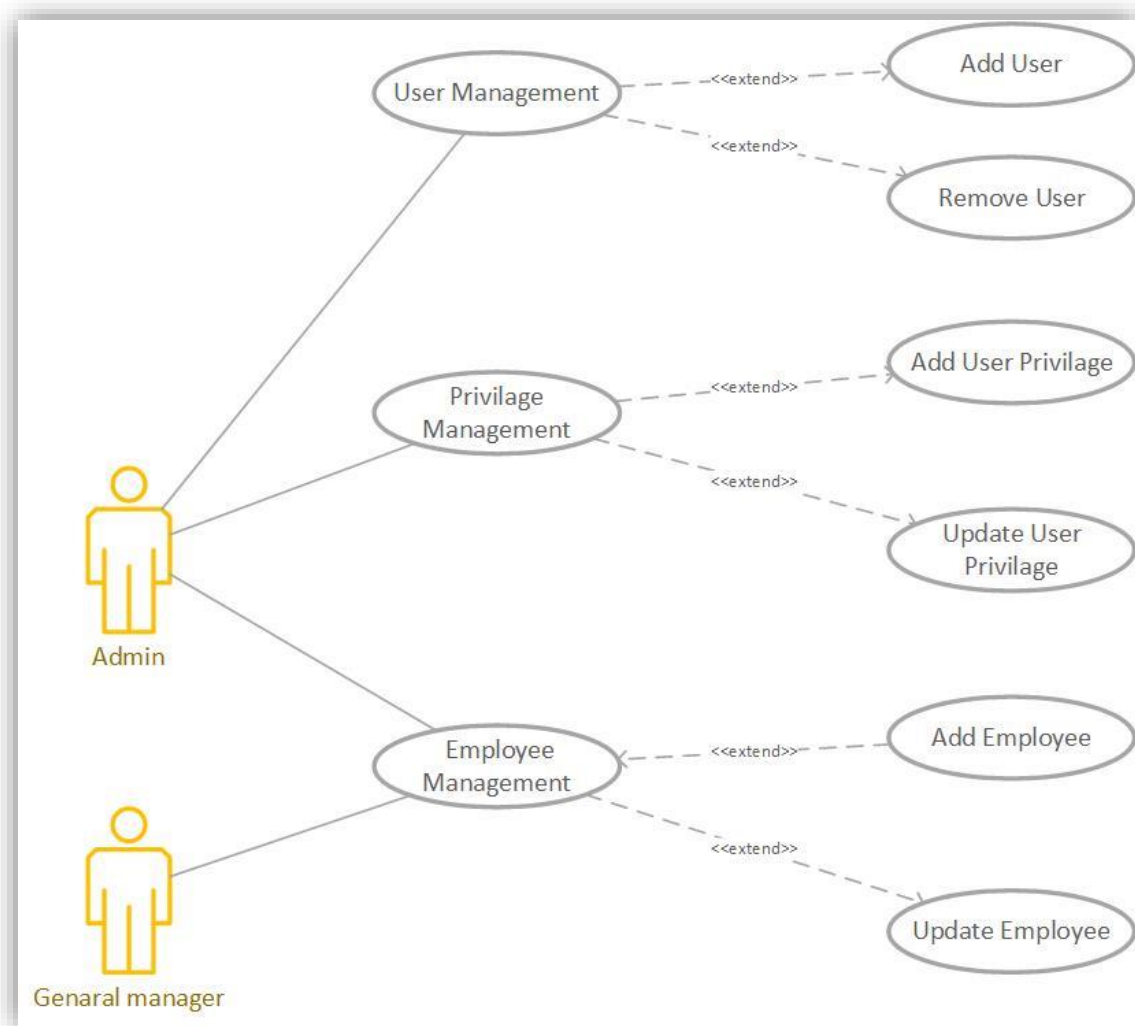


Figure 3.6: Use case diagram for administrator and general manager

### 3.3.3 ACTIVITY DIAGRAM

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deal with all types of flow control by using different elements like fork, join etc. The figure 3.7 is the Activity diagram of login to the system.

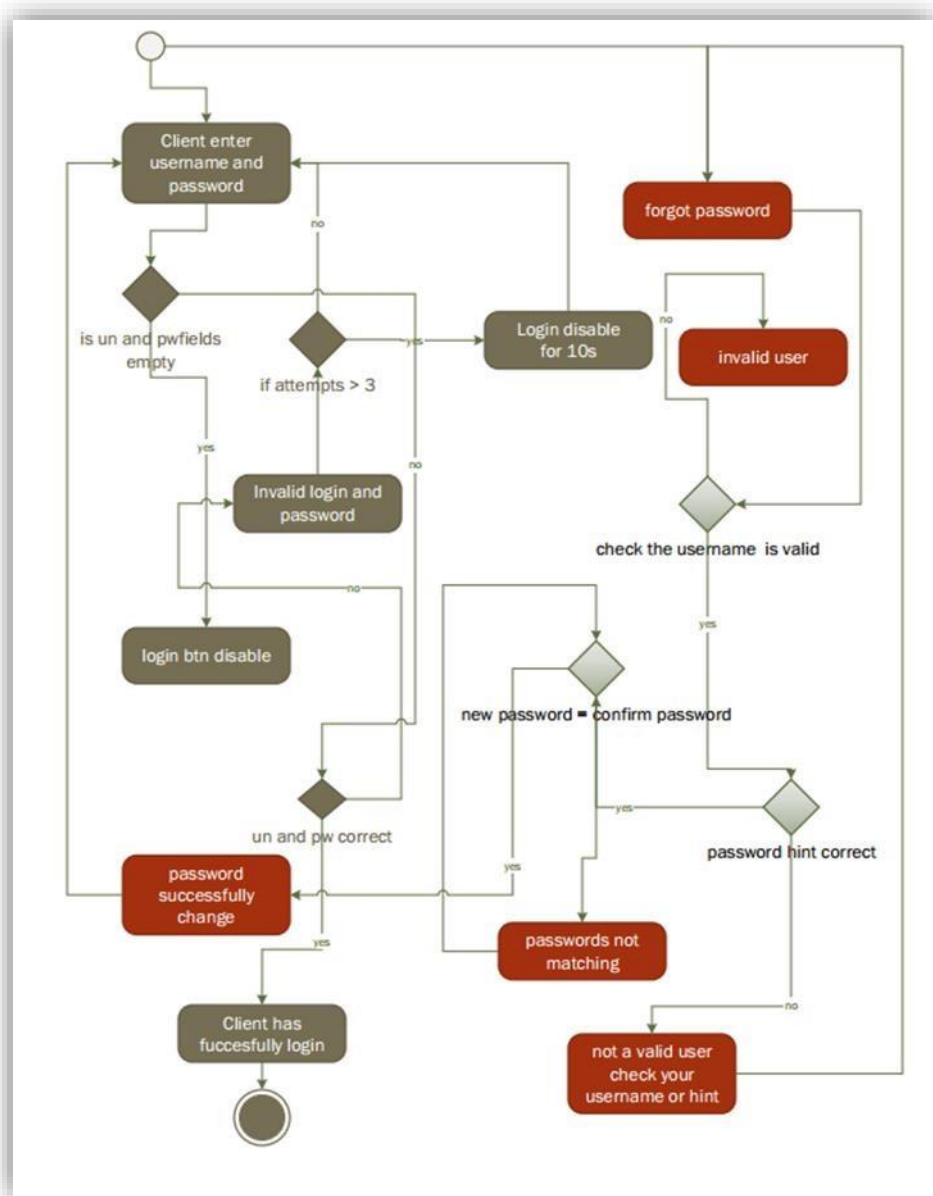


Figure 3.7: Activity Diagram for login to the system

### 3.3.4 CLASS DIAGRAM

The class diagram is an important diagram in Object Oriented Analysis (OOA). The class diagram shows how the different entities (people, things, and data) relate to each other. A class diagram can be used to display logical classes, which are typically the kinds of things the business people in an organization talk about. Following figures (figure 3.8 to 3.12) represents the class diagram of the Inventory Control and Stock Management System.

Figure 3.8 is the class diagram for purchase order (PO). Purchase order has PO status that keep track of the status and the Purchase order items that keep track of the items of the purchase order.

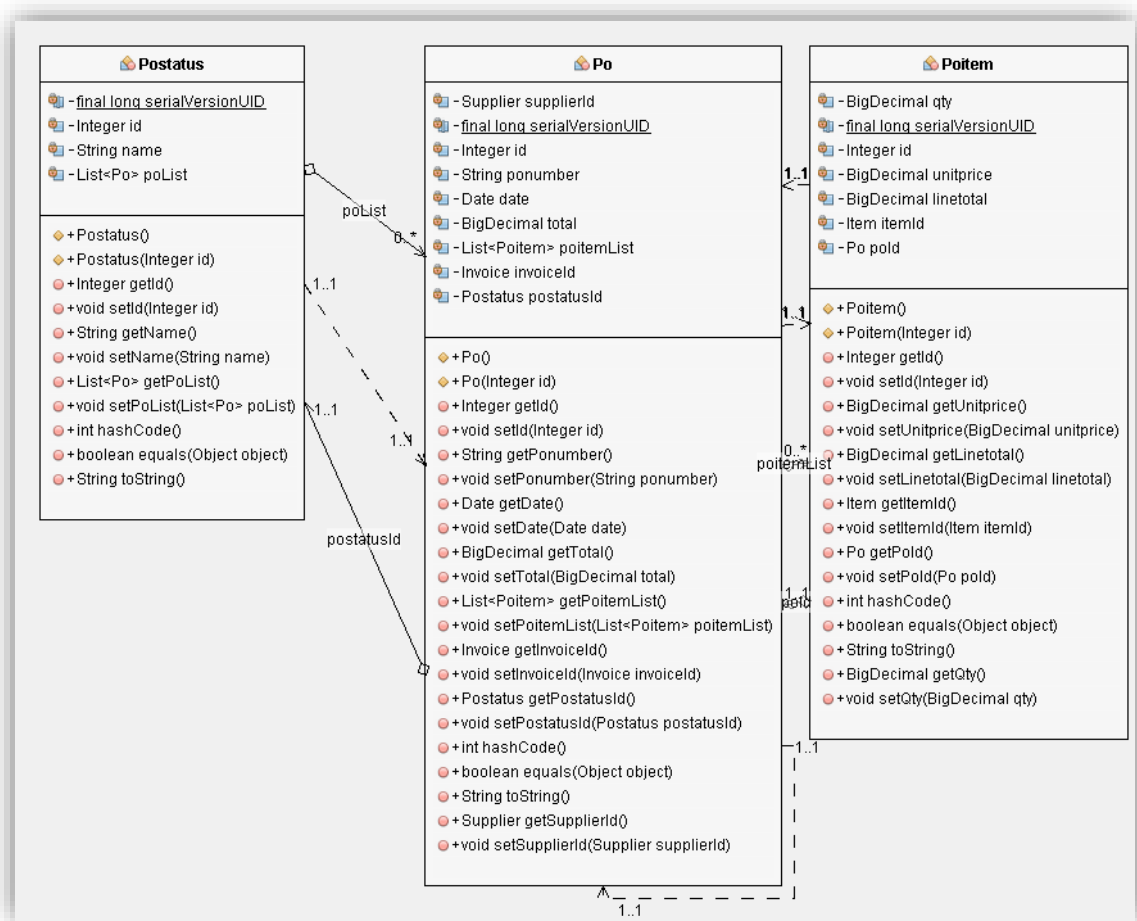


Figure 3.8: Class diagram for Purchase Order

Figure 3.9 is the class diagram of the employee management. Employee class has relationships with classes like employee status, gender, designation, civil status, etc.

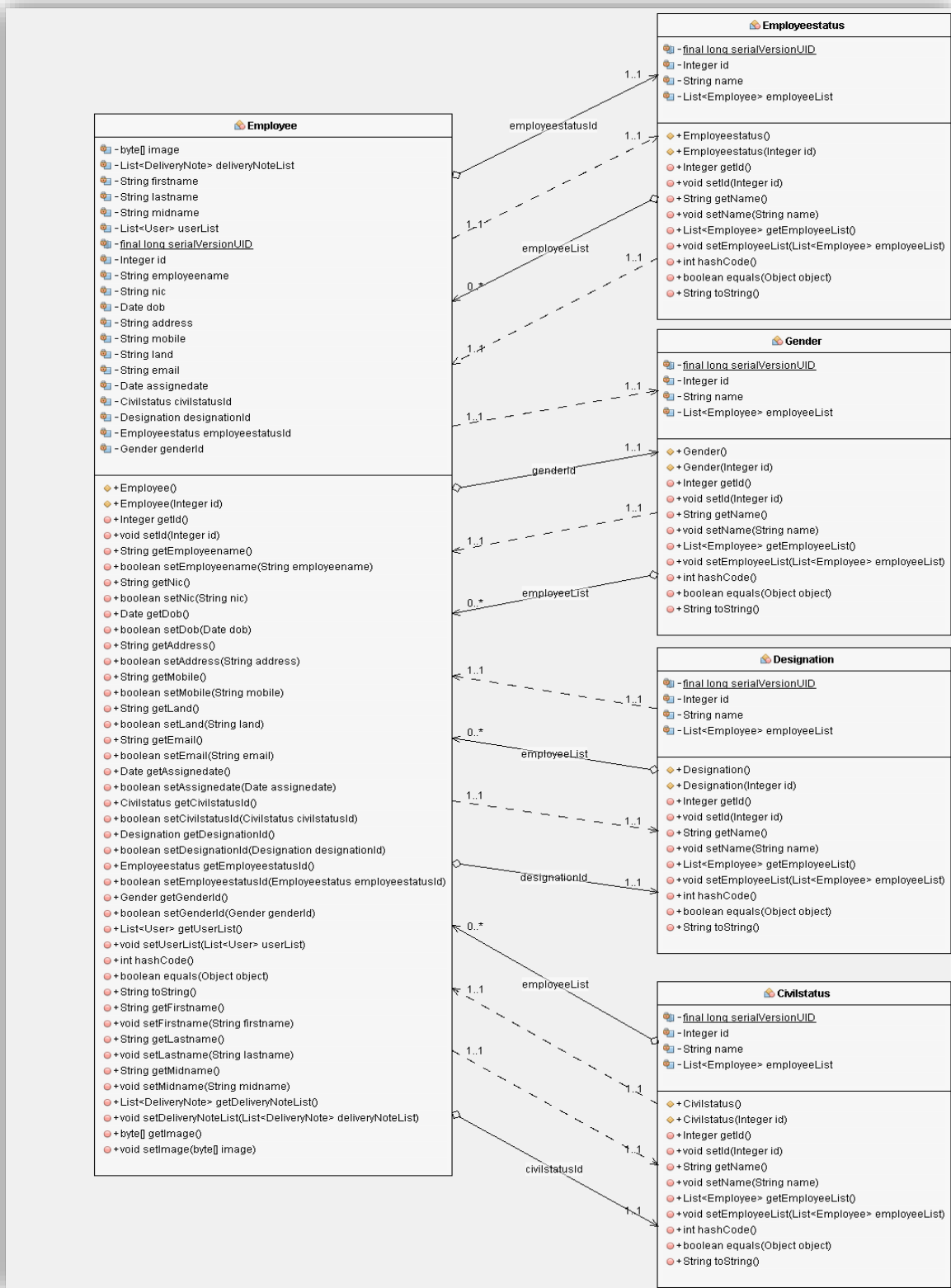


Figure 3.9: Class diagram for employee management

Figure 3.10 is the class diagram for vehicle management and there are the methods that used in this class to get and set data from the database and the relationships between some of the other relative classes that helps to keep the data.

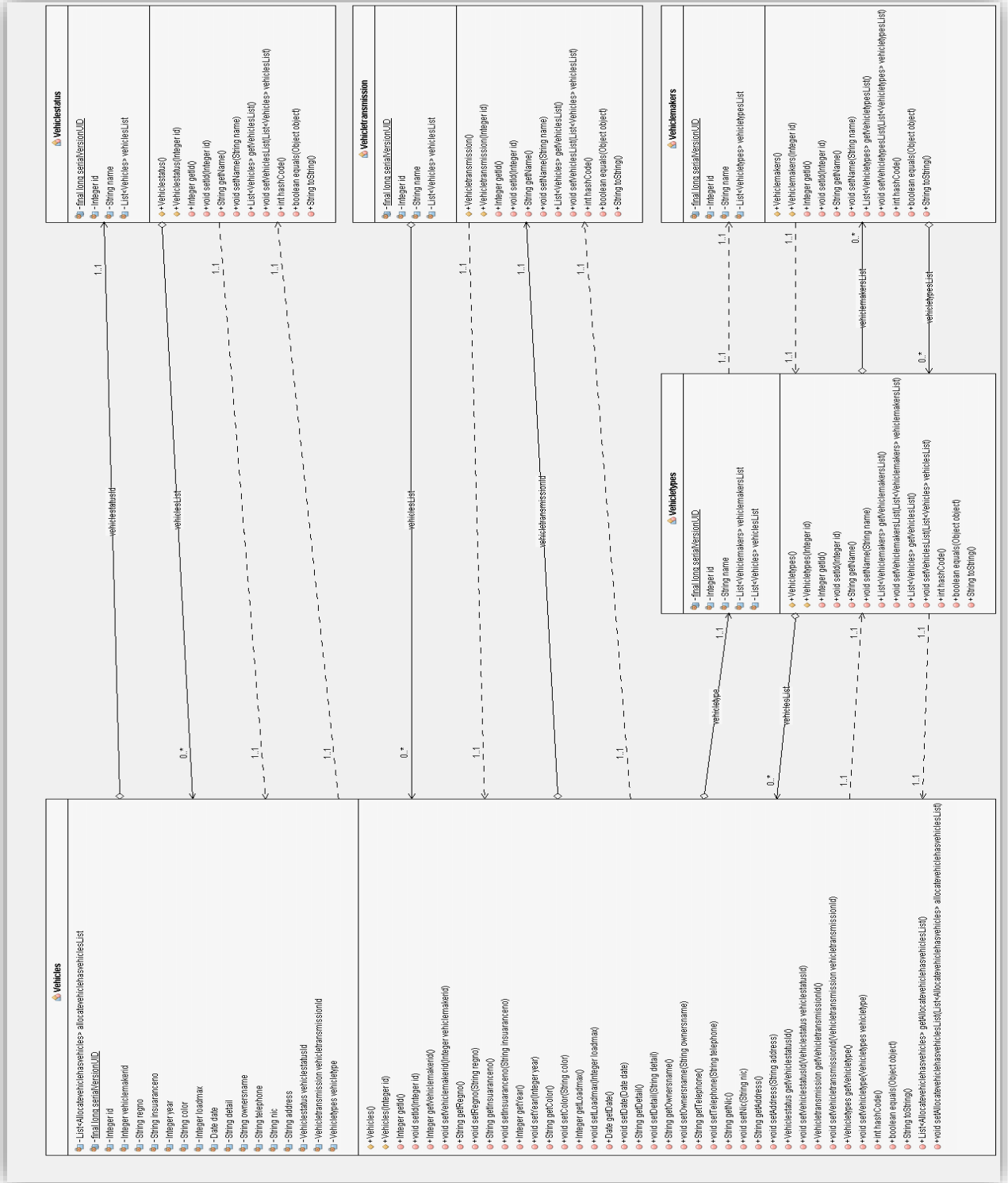


Figure 3.10: Class Diagram for Vehicle Management

Figure 3.11 is the class diagrams use in the delivery process, which has many relations with lots of classes, here are some of them that used in the delivery process.

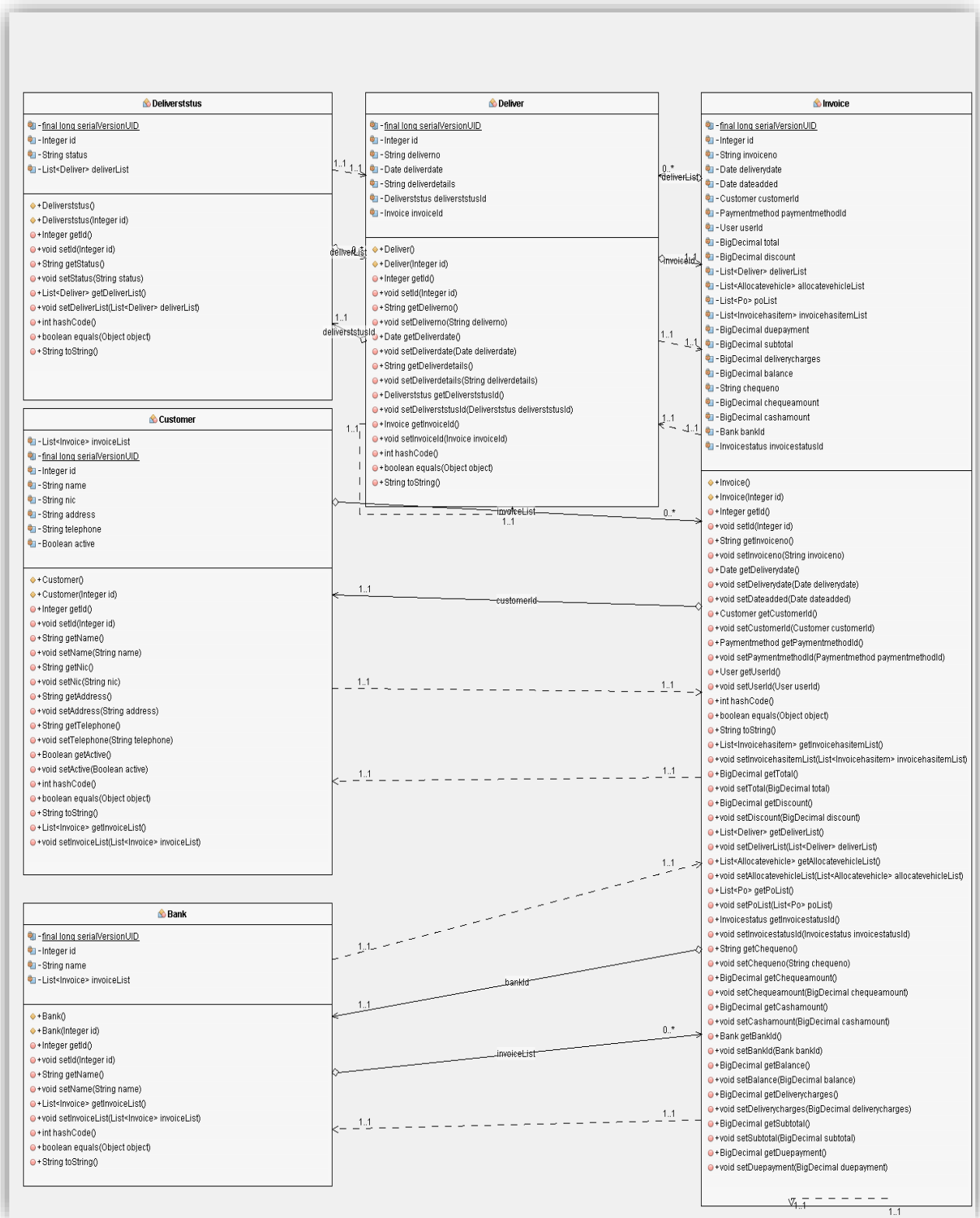


Figure 3.11: Class Diagram for Delivery Management

Figure 3.12 is the inventory class diagram that use in the system.

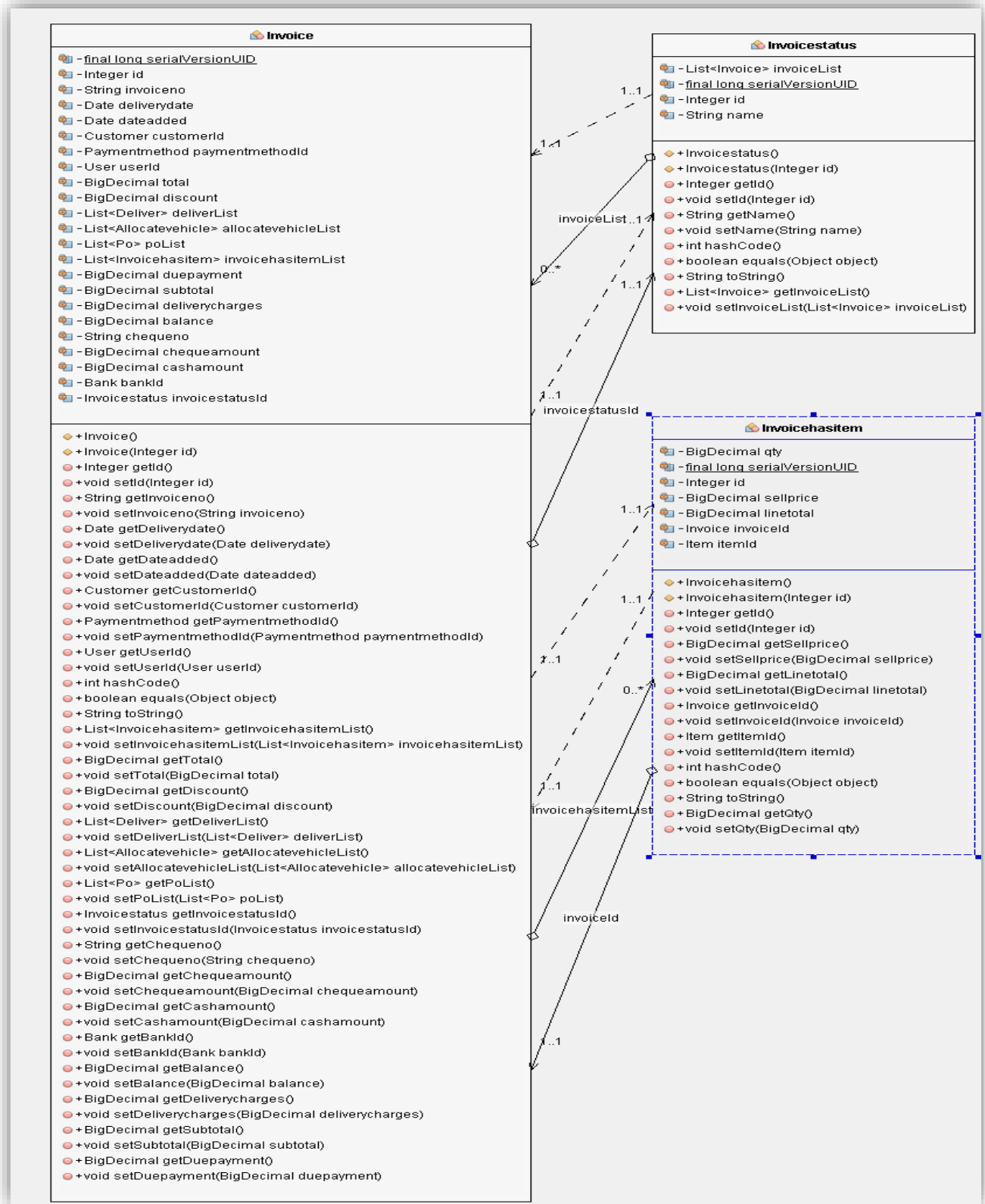


Figure 3.12: Class Diagram for invoice management



### 3.3.6 SEQUENCE DIAGRAM

A sequence diagram is a kind of interaction diagram. It describes the time ordering of the messages between objects in a specific requirement. A sequence diagram shows a set of objects and the messages sent and received by the instance of the objects. We can use a sequence diagram to illustrate the dynamic view of a system.

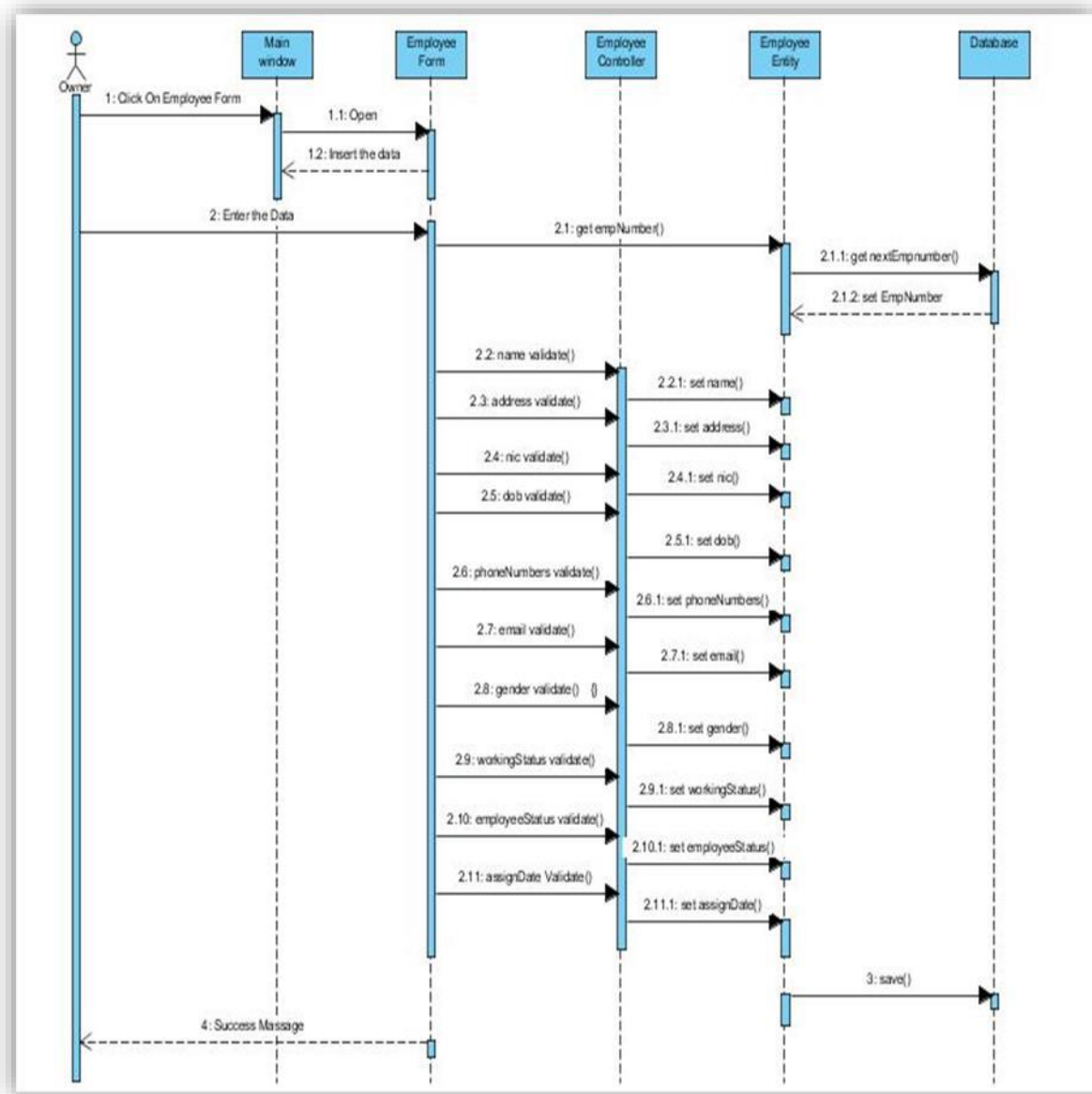


Figure 3.13: Sequence diagram for add employee

### 3.3.7 ENTITY RELATIONSHIP MODEL

“An entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements. It’s implemented in a database such as a relational database. Entities are main component in the model and the relationships that can exist among them.” [3].

Figure 3.14 shows Database design for the system (ER Model)

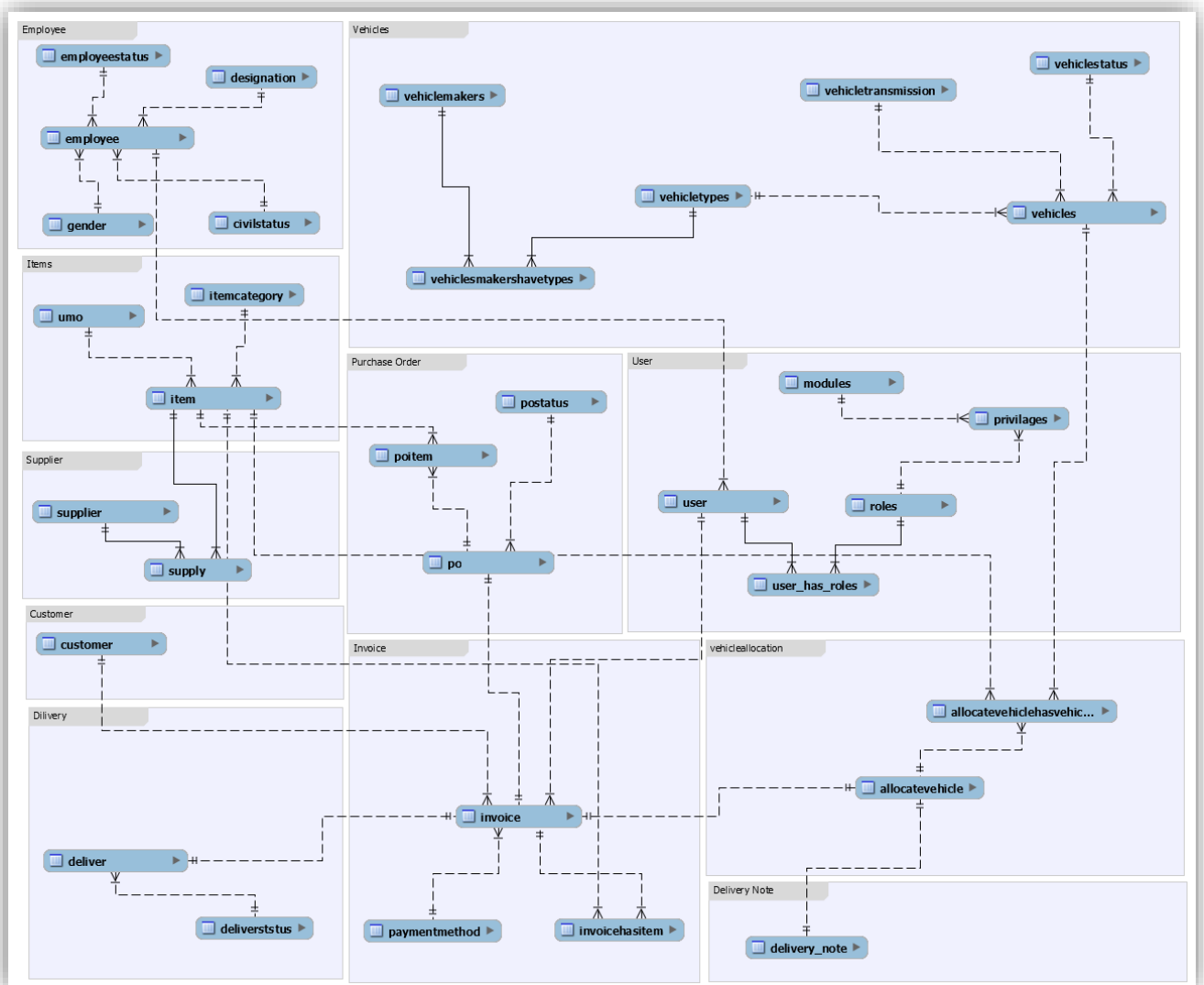


Figure 3.14: Entity Relational Diagram of the System

### 3.4 USER INTERFACE DESIGN

The main source of interaction between a user and the system are the user interfaces. Good user interfaces must be well structured and user friendly. Before developing the user interfaces, we must consider the user interface designing principals. Following are the user interface design principals.

- Structure – design should organize the structure of the locations of graphical user interface components.
- Simplicity – simple design.
- Visibility – required components for a task should be clearly shown to the user.
- Feedback – notify the user with different types of messages.

Ex: - error message, confirmation message, information

The figure 3.15 shows the user interface which will be used for login to the system for enter the system any user has to give valid user name and a password. If user satisfies the authentication test requirements he/she will allow to log into the system. Although, if user couldn't provide the correct user name and password in three times the login form will closed automatically.



Figure 3.15: Main Window User Interface

The figure 3.16 shows the Main Window of the current system when the user login to the system first time user get this Dashboard window, from there the user can access all the accessible parts of the system according to user privileges offer by the admin. It provides a tab pane for link with each category of the system and allows users to navigate through the system easily. According to the user the system main window controls the accessibility of modules by deactivating the unnecessary links. Also, it shows current user, role.

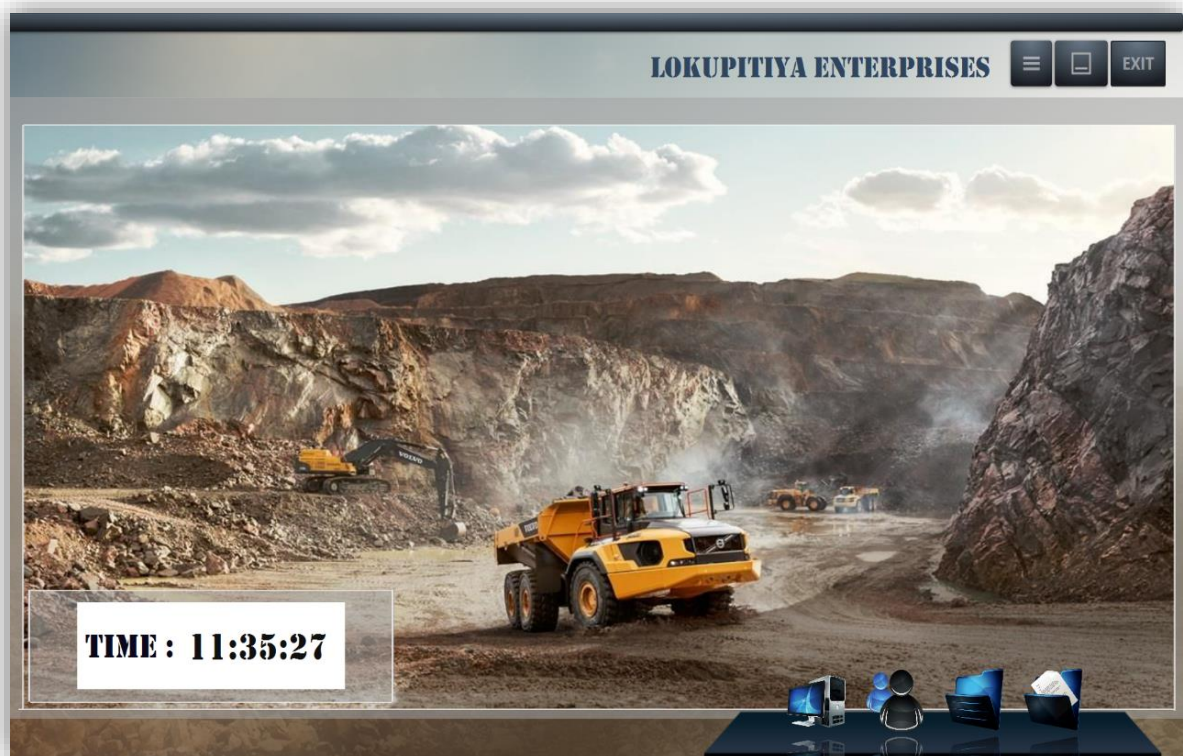


Figure 3.16: Main Window UI

The figure 3.17 shows the user interface which will be used for entering the supplier details. Supplier form provides the facility to insert, update, delete consumer details and it is given to retrieve one or more supplier. Also, validation system is carried out to ensure correctness and consistency of the user inputs.



Figure 3.17: Supplier Management UI

Figure 3.18 is the Invoice Management user interface a user can view all the invoices that added to the system and also add new invoices. The search function allows user to quick search the invoices using invoice no, invoice status and payment method



Figure 3.18: Invoice Management UI

Figure 3.19 is the user interface for the purchase order management in the system from here a user can add a purchase order and also the details of the purchase order.



Figure 3.19: Purchase Order UI

Figure 3.20 is User Management UI of the system and this is the place where all the users given their username and password and the recovery questions. All the employees that has a user account can be search by the table.



Figure 3.20: User Management UI

Figure 3.21 is Privilege Management UI this is the interface that user gets the privileges to access the system and interact with it.



Figure 3. 21: Privilege Management UI

Figure 3.22 is Item Management UI and this is the place where all the item details are stored. A valid user can add update the items and search the items from this UI.



Figure 3.22: Item Management UI

# CHAPTER 04 – IMPLEMENTATION

Implementation also plays a vital role in a system development process. The specifications made in the design phases transform to an executable system which satisfy the client requirements at this implementation stage.

In this chapter, the objective is to describe the implementation process, coding and other activities more understandable as well as giving idea about the resource requirements, tools and languages used.

## 4.1 IMPLEMENTED ENVIRONMENT

It was considered the environment of the business organization and the functional and non-functional requirements of the system while selecting the tools software and other resources. Special attention was paid for Ensure the high performance, technology feasibility, maintainability and the user friendliness were the important aspects of the selection process. Hardware and software configuration used.

### 4.1.1 HARDWARE REQUIREMENTS

- Processor – 3.06GHz or above
- RAM – 4GB or above
- Hard Disk – 250GB or above

### 4.1.2 SOFTWARE REQUIREMENTS

- MySQL Server 5.5
- MySQL Workbench
- MySQL Query Browser
- NetBeans IDE
- JavaFX Scene Builder
- Visual Paradigm
- Windows 8.1



## 4.2 DEVELOPMENT TOOL

### 4.2.1 NETBEANS 8.2

“NetBeans IDE 8.2 provides out-of-the-box code analyzers and editors for working with the latest Java 8 technologies Java SE 8, Java SE Embedded 8, and Java ME Embedded 8. The IDE also has a range of new enhancements that further improve its support for Maven and Java EE with Prime Faces, new tools for HTML5, in particular for AngularJS; and improvements to PHP and C/C++ support” [4].

NetBeans 8.0 is an IDE (Integrated Development Environment) which means, a software application that provides various facilities for developers and computer programmers for software development. NetBeans 8.0 was used as the IDE for the development of the system. It provides all the tools necessary to develop desktop, web and mobile applications. Also, it allows developers to add new plug-ins for their needs. It is primary tool for developing java application, but also support other languages such as, HTML, PHP, JavaScript and C/C++. The NetBeans IDE available for all kind of operating systems such as Windows, OS X, Linux, Solaris and etc....

There are many features in NetBeans 8.0 that facilitate to easy and speedy development in software. Design GUIs for applications quickly and smoothly by using editors and drag-and-drop tools in the IDE. The NetBeans Editor indents lines, matches words and brackets, and highlights source code syntactically and semantically. It also provides code templates, coding tips, and refactoring tools.

### 4.2.2 JAVA LANGUAGE

“Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer. As of 2015, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers.[citation needed] Java was originally developed by James

Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (byte code compiler), GNU Class path (standard libraries), and Iced Tea-Web (browser plugin for applets)” [5].

### 4.2.3 MYSQL

“MySQL is a relational database management system (RDBMS), and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records” [6].

### 4.2.4 JAVAFX SCENE BUILDER

Scene Builder is a UI layout tool for JavaFX and Visual Editor for FXML. It Helps designers and developers to build JavaFX-based UIs and Scene Builder are fully written with JavaFX 2.0 APIs to explore and learn about JavaFX objects.

### 4.2.5 VISUAL PARADIGM

Visual Paradigm is a software design tool designed for agile software projects which supports modelling standards such as UML, ERD, DFD, BPMN, ActiMate, etc. It is facilitated for building software and systems that excel in user experience by supporting

effective use case identification, requirements gathering, flow of events, wire framing, requirement specification generation, etc.

#### 4.2.6 HIBERNATE

“Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. It solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

Hibernate maps Java classes to database tables and from Java data types to SQL data types. It provides simple APIs for storing and retrieving Java objects directly to and from the database. It does not require an application server to operate. It supports almost all the major RDBMS such as MySQL, Oracle, HSQL Database Engine, PostgreSQL etc.

Hibernate uses a powerful query language called HQL (Hibernate Query Language) that is similar to SQL. HQL is fully object-oriented and understands notions like inheritance, polymorphism and association” [7].

The figure 4.1 shows Architecture of Hibernate Framework

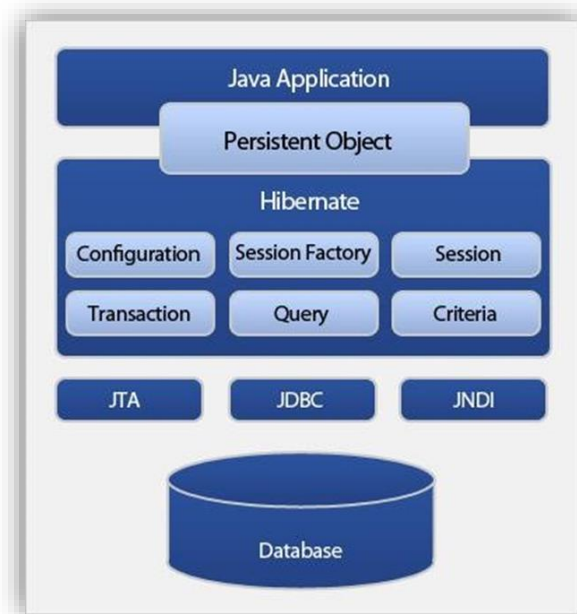


Figure 4.1: Shows Architecture of Hibernate Framework

#### 4.2.7 JASPER REPORT

“Jasper Reports is an open source reporting library that can be embedded into any Java application. It provides the necessary features to generate dynamic reports, including data retrieval using JDBC (Java Database Connectivity), as well as support for parameters, expressions, variables, and groups.

It also includes advanced features, such as custom data sources, script lets, and sub reports. It has flexible report layout and can present data textually or graphically. It is capable of exporting reports to a variety of formats and developers can supply data in multiple ways” [9].

### 4.3 CODE AND MODULE STRUCTURE DESCRIPTION

MVC design pattern and the OOP style were used to develop the proposed system (Model View Controller meant by the MVC). It clearly separates the display logic from the business logic and models hold methods. It also functions to interact with the data base.

Model - this is the layer which controls the data of the system It facilitates to display all needed data and also it governs the procedures to access the data objects and perform any kind of operations. Specialty of this layer is, it is independent from other systems like view and controller.

View - View layer uses model’s data querying methods to collect data for the purpose of presentation View should ensure that its appearance will reflects the state of the model. Whenever the data changes of the model it notifies the views depend on it.

Controller - This is different from the application logic and it is independent. This is the layer which uses Model’s data querying methods to obtain the data for the purpose of presenting.

### 4.3.1 DATA LAYER IMPLEMENTATION

A graphical tool to develop the database used, MYSQL workbench, MYSQL Server 5.5 used as the DBMS in this layer hibernate framework is used to map database tables with our model classes that have relations and relationship between them like, one-to-one, one-to-many, many-to-many, etc. It allows to perform CRUD operations [14].

### 4.3.2 HIBERNATE CONFIGURATION

Hibernate requires knowing in advance where to find the mapping information that defines how your Java classes relate to the database tables. Hibernate also requires a set of configuration settings related to database and other related parameters. All such information is usually supplied as standard Java properties file called hibernate. Properties, or as an XML file named hibernate.cfg.xml. An instance of org.hibernate.cfg. Configuration represents an entire set of mappings of an application's Java types to an SQL database. The org.hibernate.cfg. Configuration is used to build an immutable org. hibernate. Session Factory [15].

### 4.3.3 CREATE JAVA CLASSES

Java Class whose objects or instances will be stored in database tables are called persistence classes in Hibernate. We can auto generate these classes from the database using NetBeans auto entity generating option.

```

package entity;

import javax.persistence.*;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlTransient;
import java.io.Serializable;
import java.math.BigDecimal;
import java.util.Date;
import java.util.List;

/**
 * @author vibodha
 */
@Entity
@Table(name = "po")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Po.findAll", query = "SELECT p FROM Po p")
    , @NamedQuery(name = "Po.findById", query = "SELECT p FROM Po p WHERE p.id = :id")
    , @NamedQuery(name = "Po.findByPonumber", query = "SELECT p FROM Po p WHERE p.ponumber = :ponumber")
    , @NamedQuery(name = "Po.findByDate", query = "SELECT p FROM Po p WHERE p.date = :date")
    , @NamedQuery(name = "Po.findByTotal", query = "SELECT p FROM Po p WHERE p.total = :total")
    , @NamedQuery(name = "Po.findLast", query = "SELECT MAX(p) FROM Po p")
})

```

```

public class Po implements Serializable {

    @JoinColumn(name = "supplier_id", referencedColumnName = "id")
    @ManyToOne(optional = false, fetch = FetchType.EAGER)
    private Supplier supplierId;

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id")
    private Integer id;
    @Column(name = "ponumber")
    private String ponumber;
    @Column(name = "date")
    @Temporal(TemporalType.DATE)
    private Date date;
    // @Max(value=?) @Min(value=?)//if you know range of your decimal fields consider using these annotations
    @Column(name = "total")
    private BigDecimal total;
    @OneToMany(cascade = CascadeType.ALL, mappedBy = "poId", fetch = FetchType.EAGER)
    private List<Poitem> poitemList;
    @JoinColumn(name = "invoice_id", referencedColumnName = "id")
    @ManyToOne(optional = false, fetch = FetchType.EAGER)
    private Invoice invoiceId;
    @JoinColumn(name = "postatus_id", referencedColumnName = "id")
    @ManyToOne(optional = false, fetch = FetchType.EAGER)
    private Postatus postatusId;
}

```

The code below is a java class that auto generated by NetBeans, the database using Po entity of the database to generate this java class.

```
private Postatus postatusId;

public Po() {
}

public Po(Integer id) {
    this.id = id;
}

public Integer getId() {
    return id;
}

public void setId(Integer id) {
    this.id = id;
}

public String getPonumber() {
    return ponumber;
}

public void setPonumber(String ponumber) {
    this.ponumber = ponumber;
}

public Date getDate() {
    return date;
}

public void setDate(Date date) {
    this.date = date;
}

public BigDecimal getTotal() {
    return total;
}

public void setTotal(BigDecimal total) {
    this.total = total;
}

@XmlTransient
public List<Poitem> getPoitemList() {
    return poitemList;
}
```

## 4.4 INTERFACE LAYER IMPLEMENTATION

Users are interacted to the system using interface layer. It facilitates user to do the necessary operations using forms as well as loading views.

GLUON Scene Builder was used for developing user interfaces. By drag and drop facility developers can add UIs controllers to user interfaces easily and rapidly.

When creating user interfaces in the system we need to create two additional files. They are User Interface Controller class and CSS file. User Interface controller class to use and control UIs and CSS class to design the user interface with various colors, images and effects.

Figure 4.2 is the design that create using scene builder

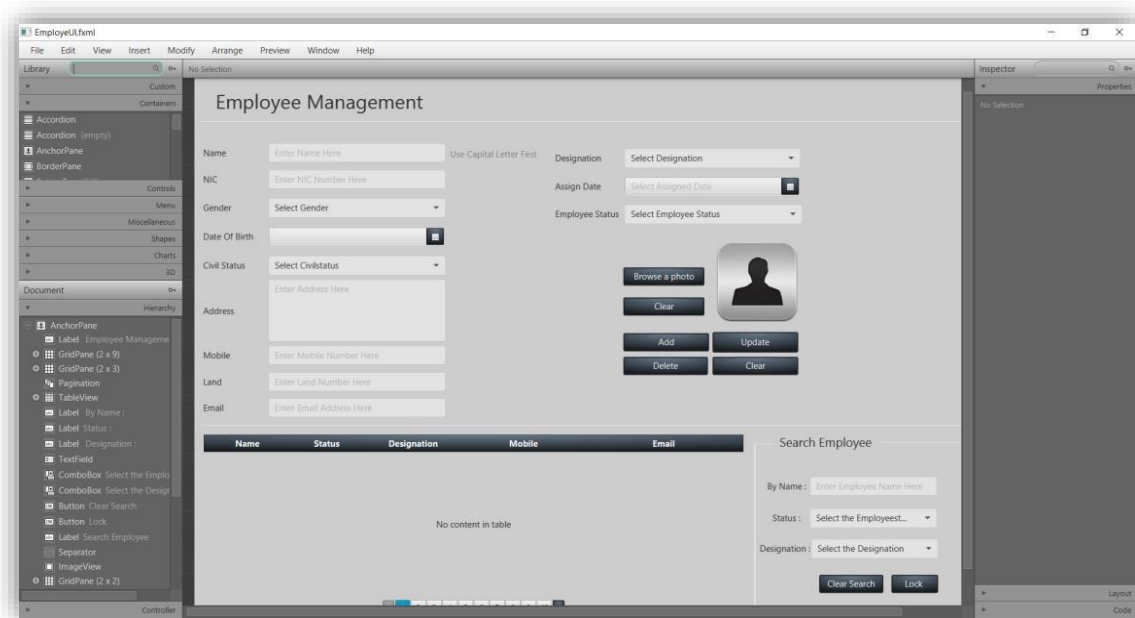


Figure 4.2: Employee Management FXML Scene Builder



Code below shows the Action listener that bind to the button which triggers the Employee Management UI.

```
@FXML
private void tbtnEmployeeAP(ActionEvent event) throws IOException {

    AnchorPane root;
    root = FXMLLoader.load(Main.class.getResource("EmployeeUI.fxml"));
    MainAnchor1.getChildren().clear();
    MainAnchor1.getChildren().add(root);
}
```

Following shows the Java code for user interface controller class.

```
package ui;

import dao.RoleDao;
import entity.Role;
import java.net.URL;
import java.time.LocalDate;
import java.util.Date;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Button;
import javafx.scene.control.CheckBox;
import javafx.scene.control.Label;
import javafx.scene.control.ListView;
import javafx.scene.control.TextField;
import javafx.scene.input.KeyEvent;
import javafx.scene.input.MouseEvent;
import supportive.Notification;
```

```
public class RoleUIController implements Initializable {

    //<editor-fold defaultstate="collapsed" desc="User-define Data">
    Role role;

    String invalid;
    String initial;

    Notification notification = new Notification();
//</editor-fold>

    //<editor-fold defaultstate="collapsed" desc="FXML Data">
    @FXML
    private ListView<Role> lstRole;
    @FXML
    private TextField txtRole;
    @FXML
    private Label lblClose;
    @FXML
    private Button btnAdd;
    @FXML
    private Button btnUpdate;
    @FXML
    private Button btnDelete;
    @FXML
    private Button btnClear;
    @FXML
    private CheckBox checkBoxActive;
//</editor-fold>
```

```
@Override
public void initialize(URL url, ResourceBundle rb) {
    invalid = Style.invalid;
    initial = Style.initial;
    initialForm();
}

private void initialForm(){
    role = new Role();

    lstRole.setItems(RoleDao.getAll());
    txtRole.clear();

    txtRole.setStyle(initial);
    checkBoxActive.setSelected(false);
}

@FXML
private void txtRoleKR(KeyEvent event) {
    if(txtRole.getText() != null)
        role.setName(txtRole.getText().trim());
}
```

Following shows fragment of the CSS code for the user interface style.css file

```
.text-field{
    -fx-background-color: rgba(255,255,255,0.5);
}
.combo-box{
    -fx-background-color: rgba(255,255,255,0.5);
}

.text-area {
    -fx-background-color: rgba(255,255,255,0.5);
}

.text-area .scroll-pane {
    -fx-background-color: transparent;
}

.text-area .scroll-pane .viewport{
    -fx-background-color: transparent;
}

.text-area .scroll-pane .content{
    -fx-background-color: transparent;
}

.jfx-hamburger StackPane {

    -fx-background-color: rgba(255,255,255,0.5);
}
}
```

## 4.5 CONTROL LAYER IMPLEMENTATION

Control layer act as messenger between the models and user interfaces. It gets the request and response from each model and updates the user interface according to the result.

### 4.5.1 HIBERNATE SESSION

Configuration object is used to create a session Factory object which in turn configures Hibernate for the application using the supplied configuration file and allows for a Session object to be instantiated. The session Factory is a thread safe object and used by all the threads of an application.

The Session Factory is a heavyweight object so usually it is created during application start-up and kept for later use. You would need one Session Factory object per database using a separate configuration file. So, if you are using multiple databases then you would have to create multiple Session Factory objects.

Following fragment code is for create session factory.

```
public class HibernateUtil {  
  
    private static final SessionFactory sessionFactory;  
  
    static {  
        try {  
            // Create the SessionFactory from standard (hibernate.cfg.xml)  
            // config file.  
            sessionFactory = new AnnotationConfiguration().configure().buildSessionFactory();  
        } catch (Throwable ex) {  
            // Log the exception.  
            System.err.println("Initial SessionFactory creation failed." + ex);  
            throw new ExceptionInInitializerError(ex);  
        }  
    }  
  
    public static SessionFactory getSessionFactory() {  
        return sessionFactory;  
    }  
}
```

#### 4.5.2 DAO (DATA ACCESS OBJECT)

In computer software, a data access object (DAO) is an object that provides an abstract interface to some type of database or other persistence mechanism. By mapping application calls to the persistence layer, DAO provides some specific data operations without exposing details of the database. This isolation supports the Single responsibility principle. It separates what data accesses the application needs, in terms of domain-specific objects and data types (the public interface of the DAO), from how these needs can be satisfied with a specific DBMS, database schema, etc. (the implementation of the DAO).

Following shows the java code for General Dao class

```
public class GeneralDao {  
    // this method is used insert an object into Database  
    public static void insert(Object object) {...26 lines }  
    // this method is used to get all object from specific entity  
    public static ObservableList select(String nameQuery) {...35 lines }  
    // this method is used to get object from providing relevent parameters  
    public static ObservableList select(String nameQuery, HashMap properties) {...36 lines }  
    // this method is used to update an object from the scehma  
    public static void update(Object object) {...29 lines }  
    // this method is used to delete an object from the scehma  
    public static void delete(Object object) {...26 lines }  
}
```

### 4.5.3 REUSED CODE MODULES

The following re-usable components have been used when implementing the system to add more attractiveness and to maximize the efficiency of the system.

This code is used to add validation to the system. The whole system has validation part and they are common. Validations are validated using Regex.

```

package common;

public class Validations {

    public static boolean booleanName(String name) {
        return name.matches("[A-Z]{1}\\w+ ([A-Z]{1}\\w+)*");
    }

    public static boolean booleanAddress(String address) {
        return address.matches("[_A-Za-z0-9-./.:\\s]{5}[_A-Za-z0-9-./.:\\s]*");
    }

    public static boolean booleanMobileNo(String no) {
        return no.matches("(077|076|078|071|075|072|070)\\d{7}|((\\+94) (77|76|78|71|75|72|070)\\d{7})");
    }

    public static boolean booleanTurns(String turns) {
        return turns.matches("\\d{1,5}");
    }

    public static boolean booleanAllPhones(String no) {
        return no.matches("(077|076|078|071|075|072)\\d{7}|((\\+94) (77|76|78|71|75|72)\\d{7})"
            || no.matches("(063|025|036|055|057|065|032|011|091|033|047|051|021|067|034|081|035|037|023"
            + "|066|041|054|031|052|038|027|045|026|024)\\d{7}");
    }

    public static boolean booleanLandNo(String no) {
        return no.matches("(063|025|036|055|057|065|032|011|091|033|047|051|021|067|034|081|035|037|023"
            + "|066|041|054|031|052|038|027|045|026|024)\\d{7}");
    }

    public static boolean booleanEmail(String email) {
        return email.matches("^[_A-Za-z0-9-]+(\\.[_A-Za-z0-9-]+)*@[A-Za-z0-9-]+(\\.[A-Za-z0-9-]+)*(\\.[A-Za-z]{2,})$");
    }

    public static boolean booleanNic(String nic) {
        Boolean x = nic.matches("\\d{9}[V|v|x|X]") || nic.matches("[1-2][0-9]{11}");
        return x;
    }

    public static boolean booleanRop(String rop) {
        return rop.matches("\\d{1}");
    }
}

```

In this code all the colors used in the system that used to validate are stored.

```
package common;

public class Style {

    public static String valid = "-fx-background-color:lightgreen;";
    public static String invalid = "-fx-background-color:pink;";
    public static String updated = "-fx-background-color:khaki;";
    public static String initial = "-fx-background-color:rgba(255,255,255,0.5);";

}
```



## **CHAPTER 05 – EVALUATION**

To evaluate a project or an activity someone should check or test and verify whether the intended components of the system or the project are full filled satisfactorily, with the specified project activities carried out. The evaluation test should identify the gaps, errors or removing any activity Mentioned in the original proposal.

For a software development project, it should be checked whether the system meets the specifications with validations. Whether it has full filled its intended purpose. Verification is process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. Validation checks that the product design satisfies or fits the intended use or the software meets the user requirements.

### **5.1 TEST STRATEGIES**

Metal Purchasing and Production Management System for Lokupitiya Enterprises was tested following strict test plan in order to ensure the ultimate product is more reliable. Testing was implemented while development progresses going on. Investigation of test cases and test data was carried out following Prototyping.

#### **5.1.1 UNIT TESTING**

One of the software testing method is unit testing. It helps to find out whether individual units of the source, sets of one or more computer program modules together with the relevant control data, usage procedures are suitable for smooth functioning. It also helps to identification of failures and to improve the quality of the system.

Single unit is the smallest test tale portion of the software which has only one or few inputs, most probably only one output. In procedural programming system a unit will be an individual program function or a procedure. Although in object-oriented programming, the smallest unit is a method, which may belong to a base/super class, abstract class or derived/child class. Unit testing is performed by using either White Box Testing or Black

box testing. Unit testing are written and performed to check whether the cods of the source are working properly and to find whether they are accurate.

### 5.1.2 INTEGRATED TESTING

This testing system (Integrated test) is also a software development process to test in multiple ways. Integrated test will be done after the unit level testing and before the system validation phase. Main object of this test is to find out expose faults. Developers themselves or independent Testers perform Integration Testing.

### 5.1.3 SYSTEM TESTING

System testing process will be performed to monitor the entire software or the hardware system. It is based on functional requirement specifications or system requirement specifications. It is also expected test up to the bounds which defined in the software requirements specifications. It tests even beyond the specifications. He system test will be handled by independent testers and it will be after the integration test. This is black box type of testing where external working of the software is evaluated with the help of requirement documents. It is totally based on Users point of view. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware systems.

## 5.2 TEST PLAN

The test plan consists of a series of different tests that will fully exercise the Metal Purchasing and Production Management System for Lokupitiya Enterprises. The primary purpose of these tests is to uncover the systems limitations and measure its full capabilities. The System tests will focus on the behavior of Metal Purchasing and Production Management System for Lokupitiya Enterprises. User scenarios will be executed against the system as well as screen mapping and error message testing. Overall, the system tests will test the integrated system and verify that it meets the requirements defined in the requirements document. Security tests will determine how secure the Metal Purchasing and Production Management System for Company is. Documentation tests will be conducted to

check the accuracy of the user documentation. These tests will ensure that no features are missing, and the contents can be easily understood. Once the Metal Purchasing and Production Management System for Company is ready for implementation, the company system users will perform User Acceptance Testing.

The purpose of these tests is to confirm that the system is developed according to the specified user requirements and is ready for operational use.

### 5.3 SYSTEM TEST CASES

A test case has components that describe an input, action or event and an expected response, to determine if a failure of an application is working correctly.

#### 5.3.1 TEST CASES AND TEST RESULTS FOR USER MODULE

<b>Test No</b>	<b>Test Description</b>	<b>Expected Result</b>	<b>Pass / Fail</b>
1	Select Combo Box 'Employee'	Available Privileges list and Assigned Privileges list will be display in List boxes according to the selected Group Type  If there is no available list Add button will be disabled and Update button will be enabled  If there are no assigned privileges Update button will be disabled and Save button will be enabled	pass
2	Select '>>' button	All the privileges view in available list will be removed and it will be added into assigned list	pass

## Metal Purchasing & Production Management System

3	Select '<<' button	All the privileges view in assigned list will be removed and it will be added into available list	pass
4	Select privileges from available list and click '>' button	Selected privileges will be removed from available list and they will be added into the assigned list	pass
5	Select privileges from assigned list and click '<' button	Selected privileges will be removed from assigned list and they will be added into the available list	pass
6	Click Add button	Confirm Message box will be shown as 'Do you want to update the following data' with assigned list	pass
		<p>If click 'YES'</p> <p>Notification will be shown as 'Privileges Added'</p> <p>If click 'NO'</p> <p>Update will be cancel</p>	pass
7	Click Update button	<p>Confirm Message box will be shown as 'Do you want to update the following data' with assigned list</p> <p>If click 'YES'</p> <p>Notification will be shown as 'Privileges Updated Successfully'</p> <p>If click 'NO' Update will be cancel</p>	pass

Table 5.1: Test results for user module

## 5.3.2 TEST CASE FOR LOGIN MODULE.

Test No	Test Description	Expected Result	Pass/Fail
1	Run the system	Initially login Form's Log in button will be disabled	Pass
2	Type on Username and Password	Log in button will be enabled.	Pass
3	When enter the incorrect Username and Password.	Bellow the Password field message will be shown as "Invalid Username or Password" with Left attempts. Initially User has three attempts to try to login.  When three attempts exceeded all the fields and buttons will be disabled for 10 seconds and bellow the Password field message will be shown as Left Count and "System has blocked"	Pass
4	When enter the correct Username and Password.	Close the log in form and open the Main window	Pass
5	Click the "Forget Password" label	Open the Forget Password window	Pass
6	Click on Close label	When on the mouse cursor will be changed to hand.  When click, alert box will pop up for get the confirmation from user	Pass

Table 5.2: Test case for login module.

## 5.4 USER EVALUATION

As the final evaluation, the system has been tested by the user and it called as the acceptance test. This was carried out in the real working station and at the actual working environment with the available data in working background. Then the user could decide whether it has included the functional and non-functional needs. With the response of the user, decide whether it has been included all required modules and the success of system. The advantages and disadvantages and the user friendliness of the system also monitored. Final evaluation of the system has been decided with the user’s feedback.

The chart below is the user evaluation chart.

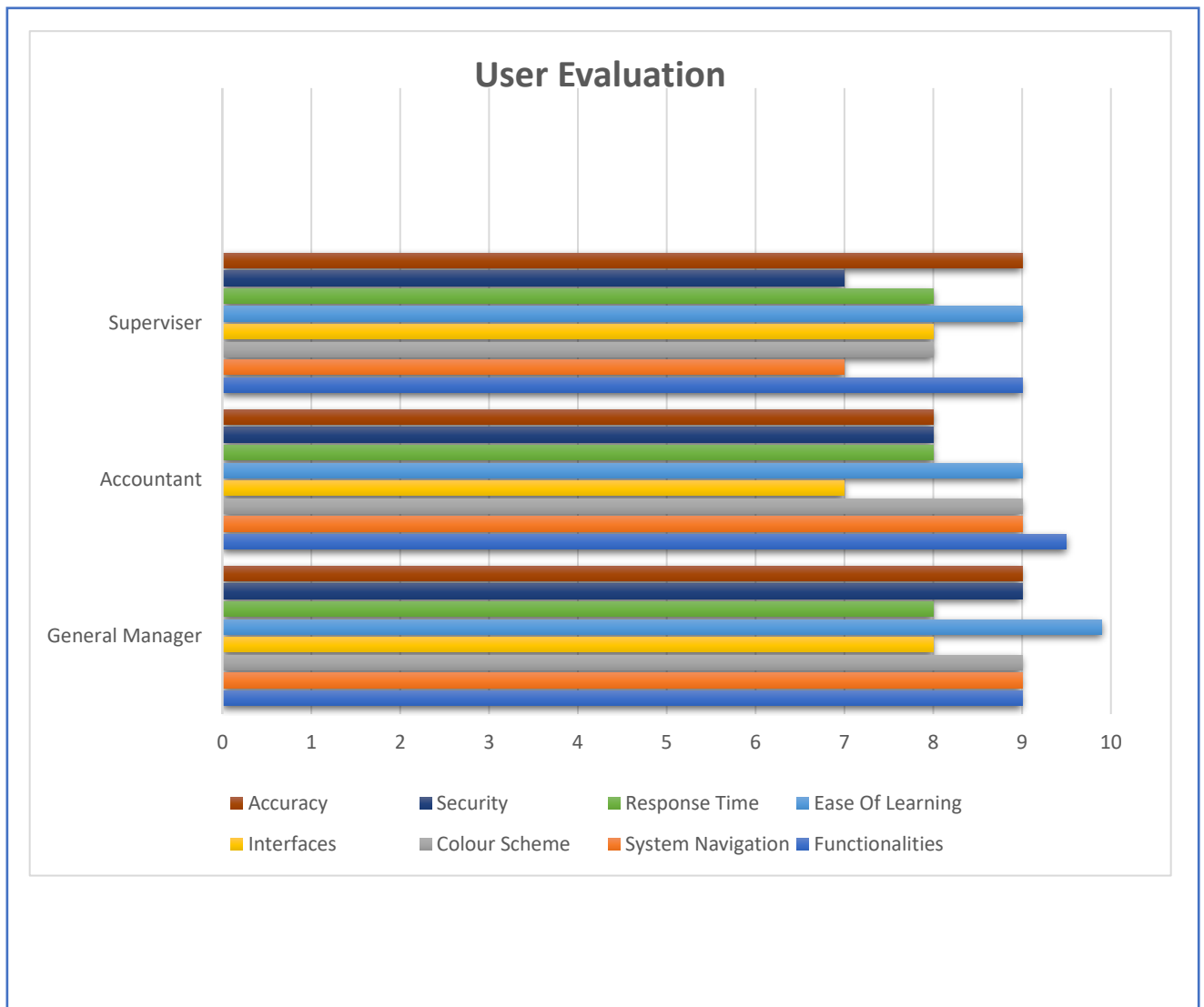


Figure 5. 1: User Evaluation chart

Figure 5.2 is the user evaluation chart which is used to check the feedbacks from the user after testing

## User Evaluation Questionnaire

**Metal Purchasing & Production Management System**

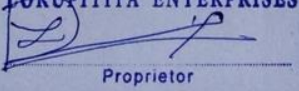
Name of the User : R.P. Lokupitiya

Designation : Owner

System Role of the User: Administrator

Feedback Area	Very Good	Good	Average	Poor	Very Poor
Functionalities		✓			
System navigation	✓				
Color Scheme	✓				
Interfaces	✓				
Ease of Learning	✓				
Response Time	✓				
Security		✓			
Accuracy	✓				
Supportiveness for Management		✓			

**LOKUPITIYA ENTERPRISES**



Proprietor

Signature: .....

Date: 28/11/2017

Figure 5.2: User Evaluation Questionnaire

## CHAPTER 6- CONCLUSION

The Lokupitiya Enterprises 's "Metal Purchasing and Production Management" has come to the successful point with facing different kind of challenges and solving all of them. New skills and knowledge gained throughout the period assisted me to face the challenges and solve all the problems.

From the start of the company, they have maintained the business process by using a traditional manual system. With the manual system they had faced lots of difficulties when dealing with main business processes within the company such as stock handling, purchase management and production management. This Metal Purchasing and Metal Production Management was developed to address those drawbacks as well as to optimize their overall business process and through this to help them to keep their company name at the top of the market.

The main objectives that they expected from the project are given below,

- Reduce the time and human effort for purchase and production management process.
- Improve the efficiency and reliability of daily routine activities such as Sales and Purchases.
- Improve Consumer, Supplier and Employee information management.
- Produce daily, weekly, monthly and yearly statistical reports about sales and supplier payments.
- Provide security and easy access to the system.

Above objectives are successfully achieved by the project. And also, it was understood when using a software system, the security of this data is very important. So new requirements were raised like Keeping a backup and log files and user management.

Such features mention above has made this Purchase and Production Management System a valuable asset for their day-to-day activities and also for longer runs.

By comparing the user feedback, test results, system functionality with the existing system, it was identified as a system which can satisfy the client requirements up to a satisfactory level.



All employees of the company have motivated and work efficiently after introducing the system and also the company increased their business as their activities has been run smoothly and efficiently.

### 6.1 LESSON LEARNT

As an undergraduate the knowledge gained throughout the project was really valuable. In addition, this gave me an exceptional experience of being in a complete software development life cycle, starting from feasibility studies to conclusion of the project. This project gave an opportunity to get extensive knowledge on JavaFX, XML, Hibernate, MVC, MySQL, NetBeans and many more languages, tools and technologies. And, it helped to test and implement most important theories and technologies learnt throughout the BIT degree program.

### 6.2 FUTURE IMPROVEMENTS

In future following features are planning to add to the newly built system as further Improvements.

- Web Based Metal Purchasing and Production Management System will be introduced for ease of connecting branches.
- Allow online customers to purchase needs through the system and deliver orders just they make online payment.
- System will be expanded in the future to payment records using external reading devices making the system fully automated.
- Android System to insert daily updates into the system

# REFERENCES

- [1] "Rational Unified Process Model," [Online]. Available: <https://www.google.lk/search?q=Rational+Unified+Process+Model>. [Accessed 12 August 2017].
- [2] "Wikipedia, the free encyclopedia, "Use Case Diagram"," [Online]. Available: [http://en.wikipedia.org/wiki/Use\\_Case\\_Diagram](http://en.wikipedia.org/wiki/Use_Case_Diagram). [Accessed 05 March 2017].
- [3] "Wikipedia, the free encyclopedia, "Entity Relationship Diagram"," [Online]. Available: [http://en.wikipedia.org/wiki/Entity%E2%80%93relationship\\_model](http://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model). [Accessed 05 March 2017].
- [4] "'NetBeans 8.0'," [Online]. Available: <https://netbeans.org/community/releases>. [Accessed 12 August 2017].
- [5] "Wikipedia, the free encyclopedia, "Java Language"," [Online]. Available: [https://en.wikipedia.org/wiki/Java\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/Java_(programming_language)). [Accessed 12 August 2017].
- [6] "Wikipedia, the free encyclopedia, "MySQL"," [Online]. Available: <https://en.wikipedia.org/wiki/MySQL>. [Accessed 12 August 2017].
- [7] "Wikipedia, the free encyclopedia, "Hibernate (java)"," [Online]. Available: <https://en.wikipedia.org/wiki/Hibernate> [Accessed 12 August 2017].
- [8] "Architecture of Hibernate Framework," [Online]. Available: <http://www.waheedtechblog.in/2013/04/what-is-hibernate.html> [Accessed 12 August 2017].
- [9] "Wikipedia, the free encyclopedia, "Jasper Report"," [Online]. Available: <https://en.wikipedia.org/wiki/JasperReports> [Accessed 12 August 2017].
- [10] "Skywear inventorycontrol," [Online]. Available: <https://www.skywareinventory.com> [Accessed 12 August 2017].
- [11] "Wikipedia, the free encyclopedia," "Waterfall Model " [Online]. Available: [https://en.wikipedia.org/wiki/Waterfall\\_model](https://en.wikipedia.org/wiki/Waterfall_model) [Accessed August 2017].
- [12] "Wikipedia, the free encyclopedia," "Prototype-based programming" [Online]. Available: [https://en.wikipedia.org/wiki/Prototype-based\\_programming](https://en.wikipedia.org/wiki/Prototype-based_programming) August 2017].

- [13] "Wikipedia, the free encyclopedia," "Rapid application development" [Online]. Available:[https://en.wikipedia.org/wiki/Rapid\\_application\\_development](https://en.wikipedia.org/wiki/Rapid_application_development) August 2017].
- [14] "Wikipedia, the free encyclopedia," "Data access layer" [Online]. Available: [https://en.wikipedia.org/wiki/Data\\_access\\_layer](https://en.wikipedia.org/wiki/Data_access_layer) August 2017].
- [15] "Wikipedia, the free encyclopedia," "Hibernate" [Online]. Available: [https://en.wikipedia.org/wiki/Hibernate\\_\(framework\)](https://en.wikipedia.org/wiki/Hibernate_(framework)) August 2017].

# APPENDIX-A - SYSTEM

## DOCUMENTATION

This section contains importance information for administrators, users and anyone who wishes to continue this project. This contains information and steps about installation, configuration

Hardware and software configuration requirements.

### **Hardware Requirements:**

- Personal Computer with processing power similar or higher than 3.06 GHz
- RAM with 2 GB or above capacity.
- Hard Disk with 160 GB or above capacity.

### **Software Requirements:**

- MySQL Server 5.5
- NetBeans IDE
- Windows 10

## A.1. HOW TO SETUP

### A.1.1 Install Java Run Time on Client Machine

Following figure A.1 is some important screenshots of installing JAVA runtime on client machine.

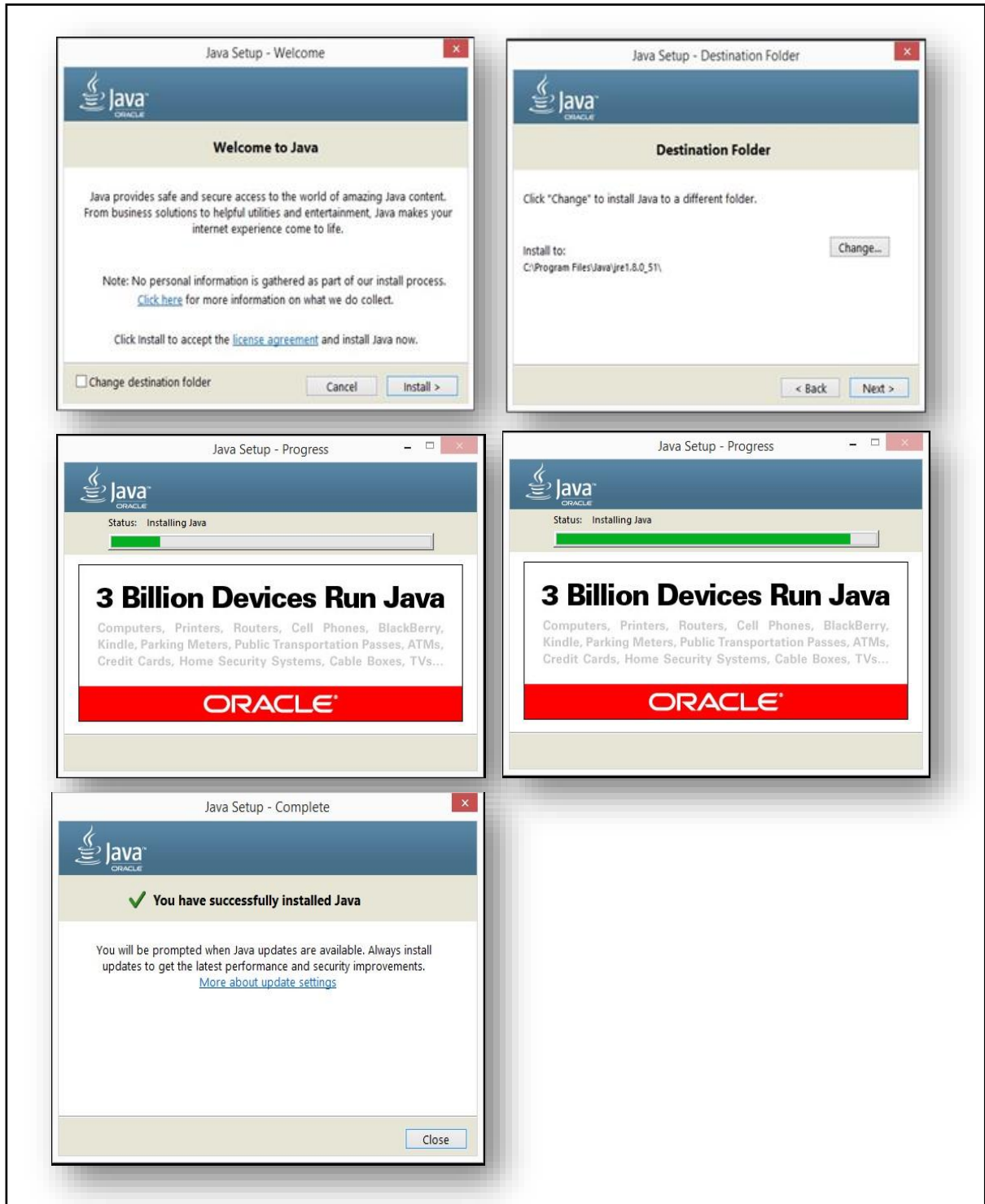


Figure A. 1: Installing Java Runtime

Figure A.2 show the steps 1-4 of installing MySQL server.

Step 1: Download MySQL Community Server 5.5 installation file appropriate for the platform. Double click on MySQL Server setup file and click “Next”.

Step 2: Accept the License Agreement and click “Next”.

Step 3: Choose “Typical” setup type, click “Next” and “Install”.

Step 4: Click “Install” to begin the installation.

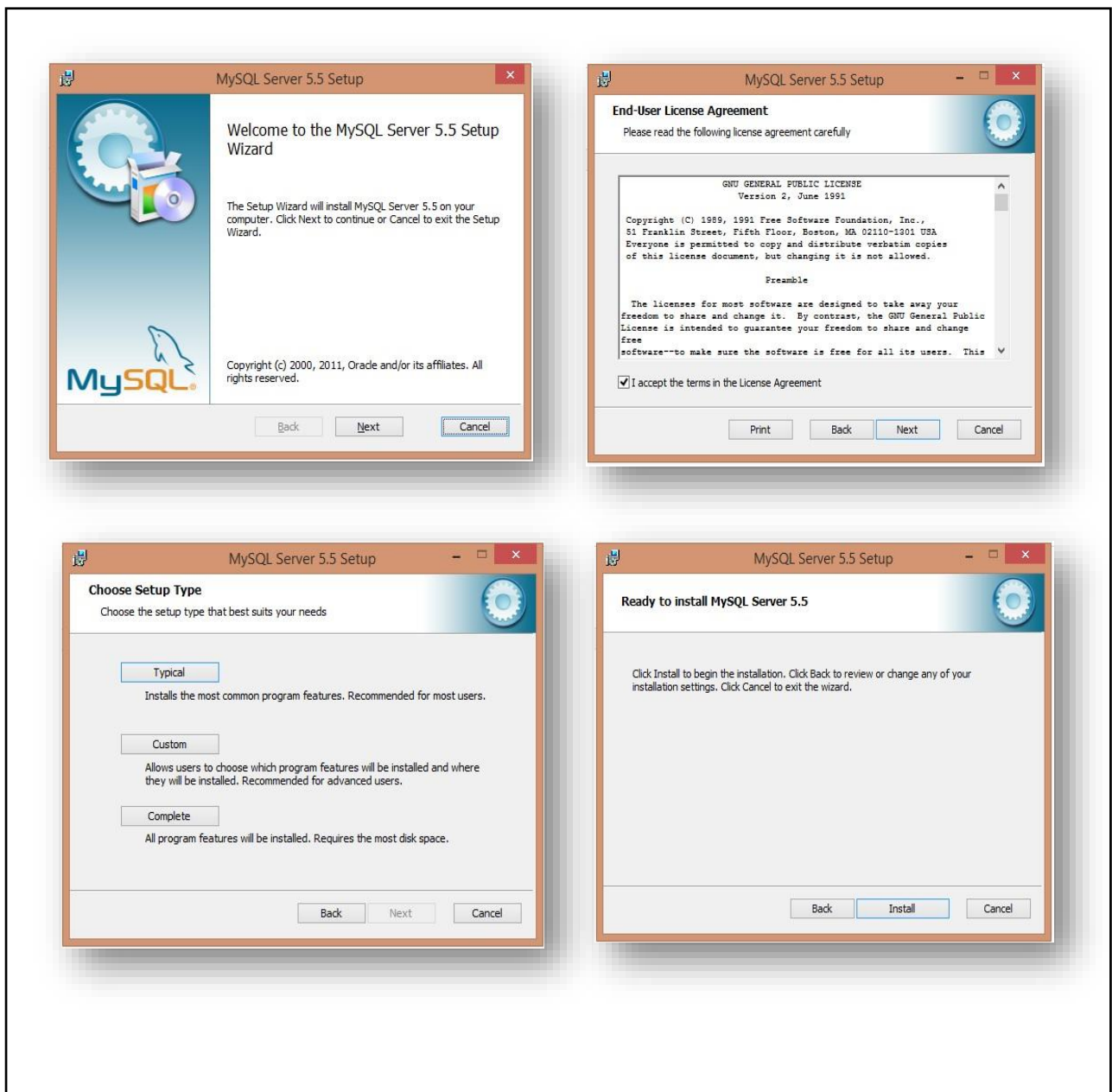


Figure A. 2: Installation process of MYSQl server 5.5

Step 5: After installation process is completed, check “Launch the MySQL Instance and begin installation.

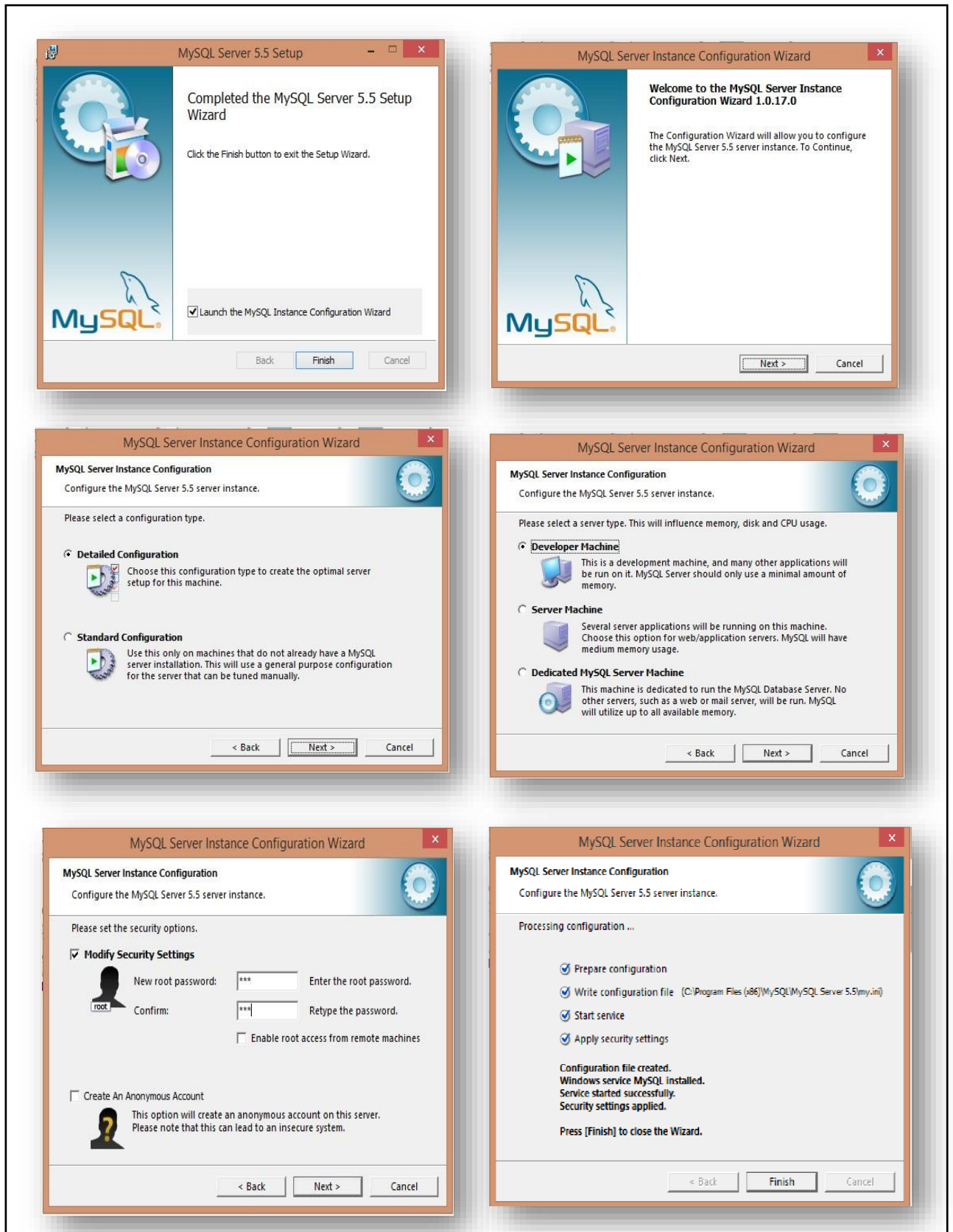


Figure A.3: MySQL server instance configuration wizard

Step 6: Installing Microsoft SQL Query Browser

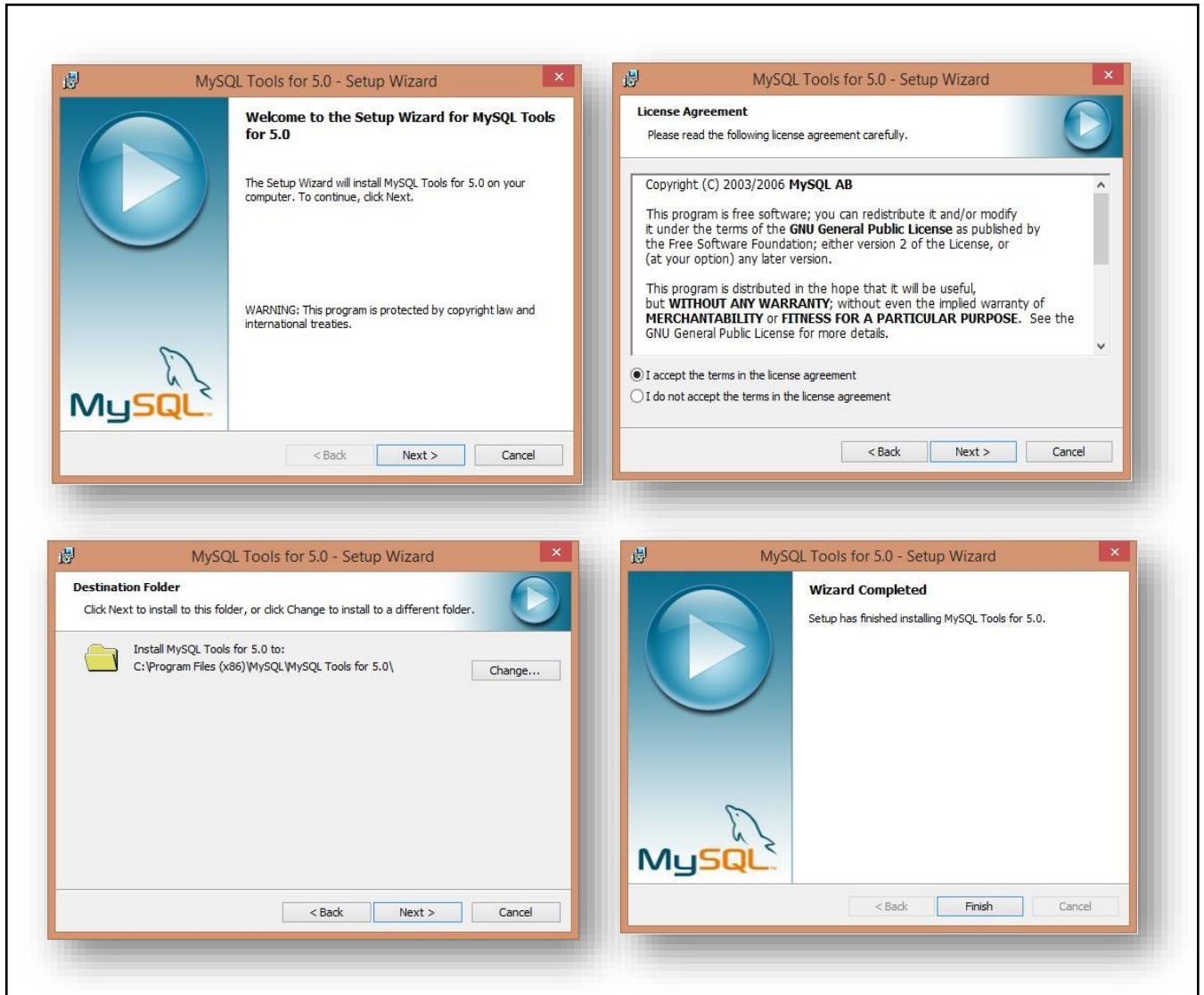


Figure A.4: Setup wizard for MySQL tools



Figure A.5, Now the MySQL server is ready to use. Enter username and password and choose the connection then press ok to connect.

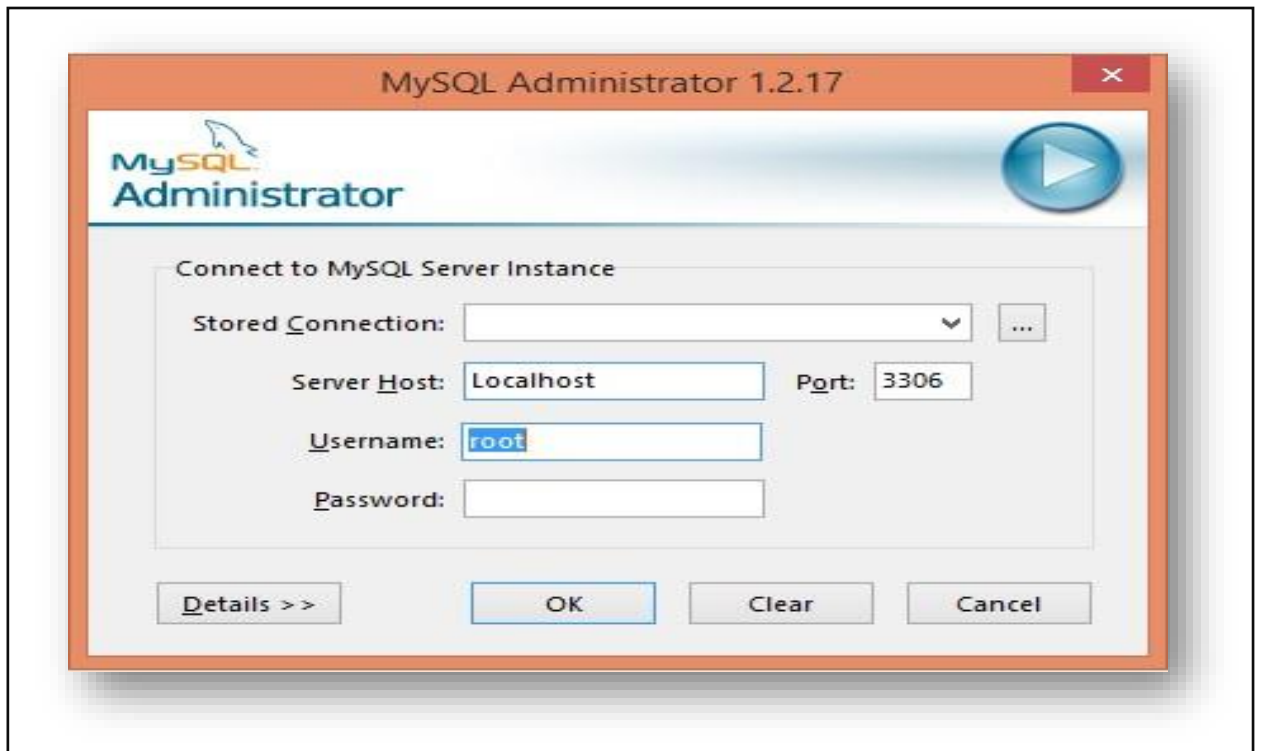


Figure A.5: MySQL server connection portal

# APPENDIX B - DESIGN

## DOCUMENTATION

### B.1 Activity Diagram for add a new user

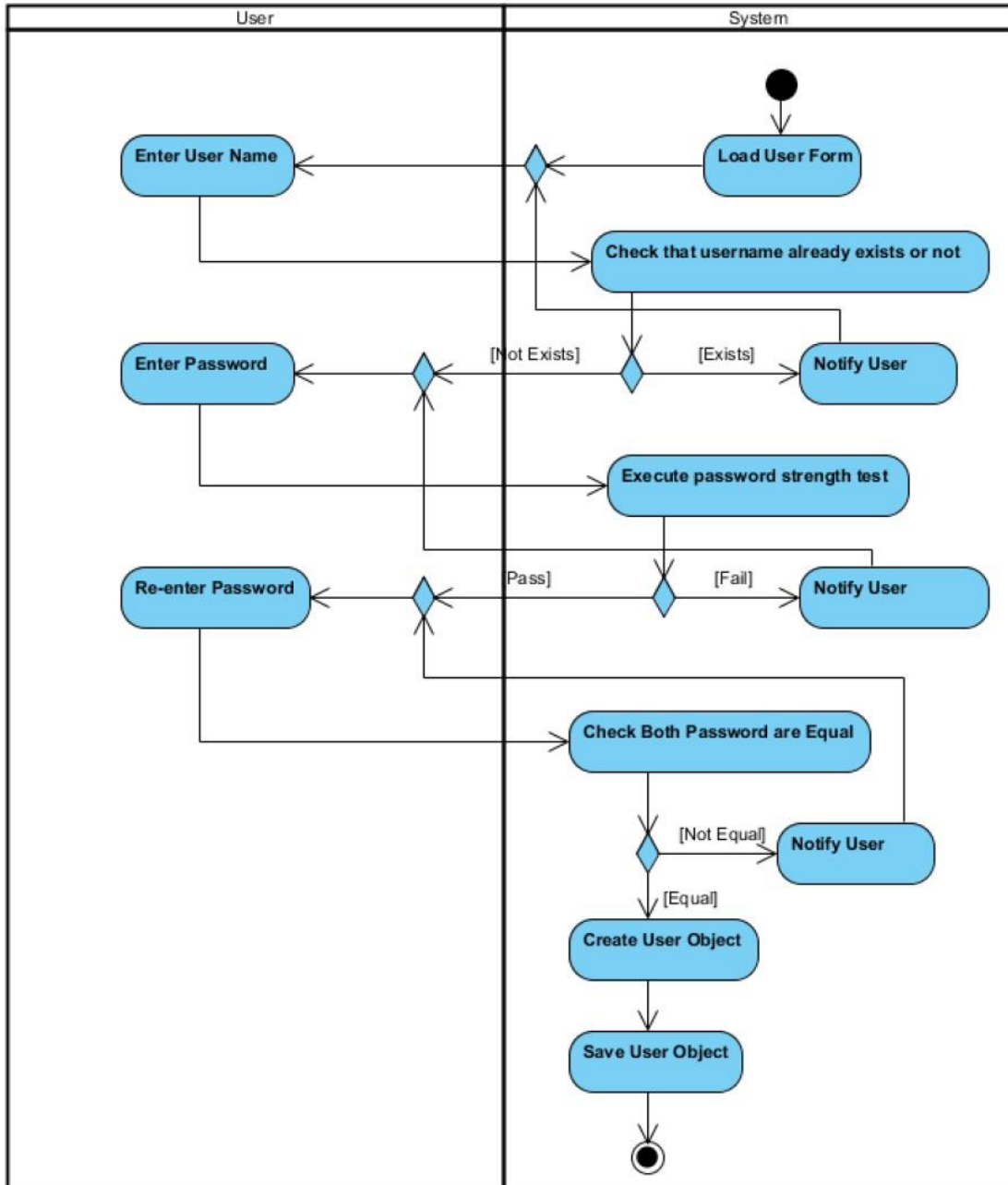


Figure B.1: Activity Diagram for add a new user

## B.2 Sequence Diagram

A sequence diagram is a kind of interaction diagram. It describes the time ordering of the messages between objects in a specific requirement. A sequence diagram shows a set of objects and the messages sent and received by the instance of the objects. We can use a sequence diagram to illustrate the dynamic view of a system.

Figure B.2 is the sequence diagram of adding a new user to the system

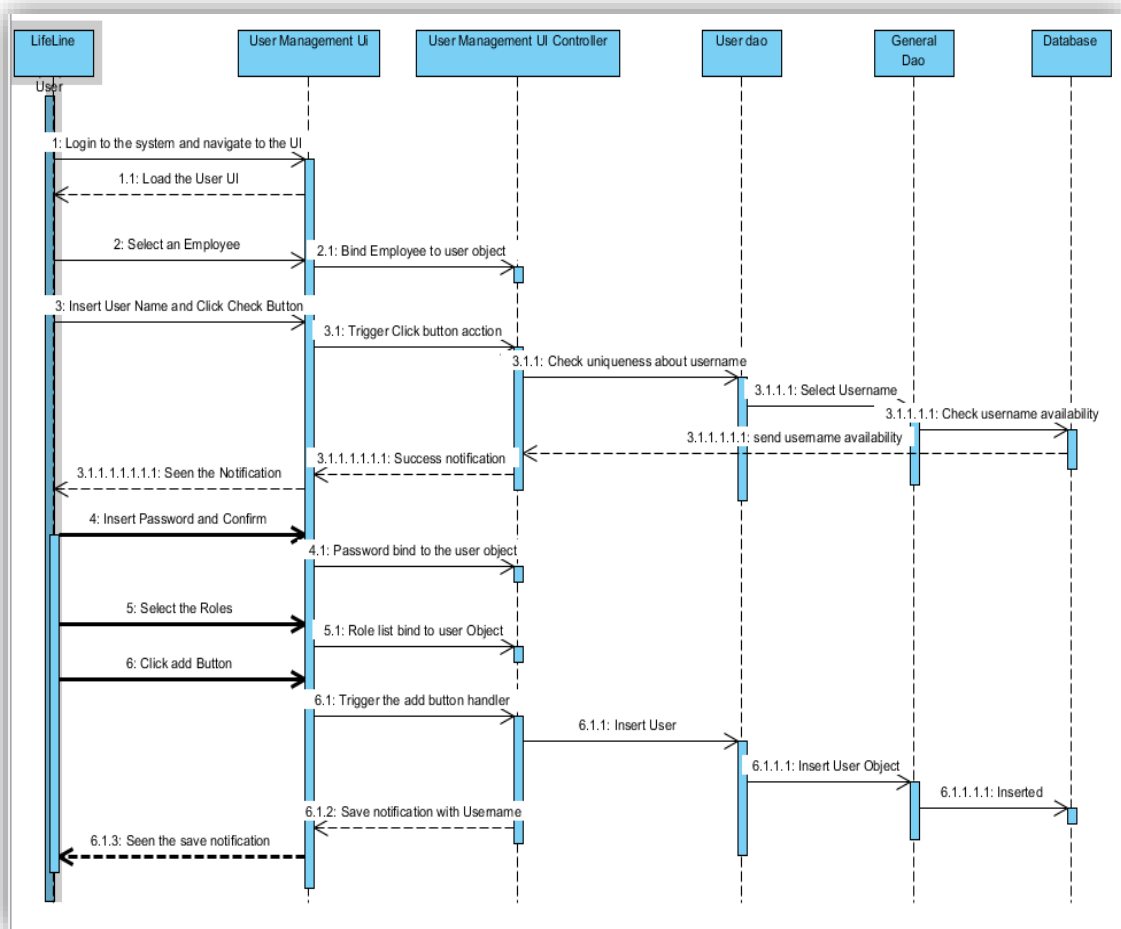


Figure B.2: Sequence Diagram for add new User

### B.3 Use Case Diagram

Figure B.3 shows the use case diagram to add a new user to the system. Only admin can add users and give privileges to them.

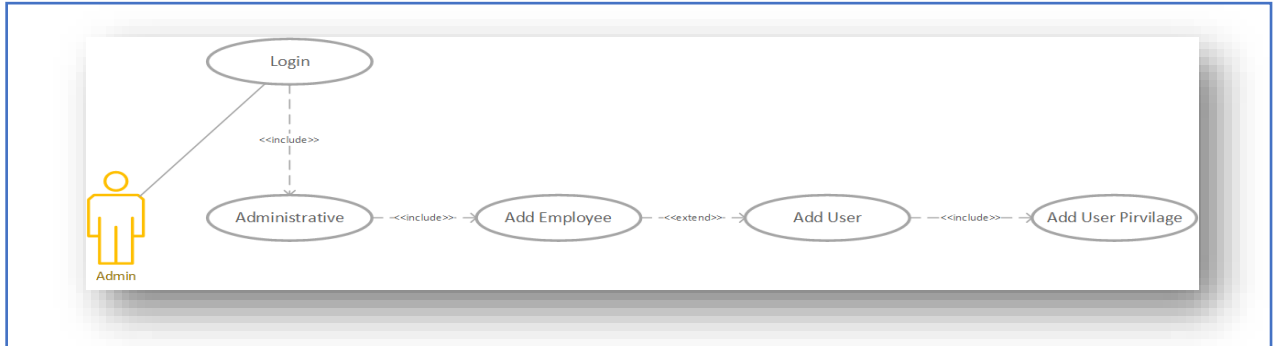


Figure B.3: Admin Adds a User - Use Case Diagram

<b>Use Case Name</b>	Add New User
<b>Actor</b>	Administrator
<b>Overview</b>	
Add New user to the system	
<b>Pre-condition</b>	
Admin must Login to the system under Super user account Go to the relevant User Interface Select Employee management and add new employee Select Add User Give relevant user privileges Click ok and confirm to add a new user	
<b>Flow of Event</b>	
Add Employee first Click User Privilege Select the Employee to add User Privilege Give the relevant privileges Click ok Button	
<b>Post Condition</b>	
Check the user is added to the user table with relevant privileges	

Table B.1: Use case narrative of Add a new user to the system

### B.4 Database Design

Figure B.4 shows the MySQL database tables that generate by using the ER diagram

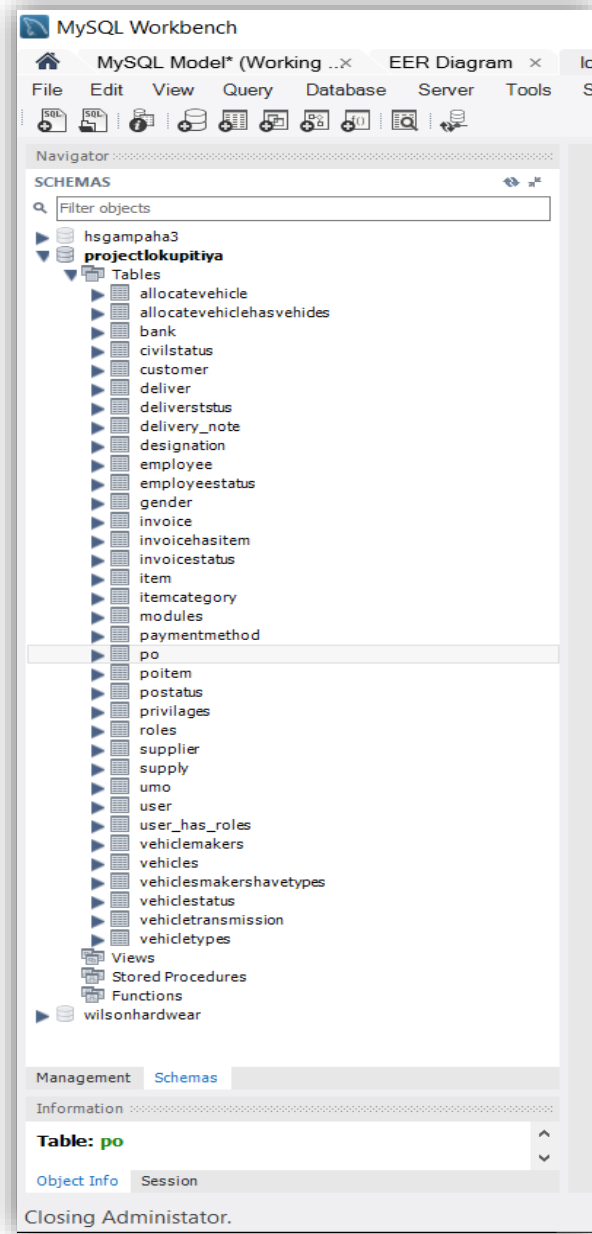


Figure B. 4: MySQL database tables

Figure B.5 to B.10 are some of the table with the structures.

B.5 Invoice table that use to store invoice details.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
invoiceno	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
customer_id	INT(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
deliverydate	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
dateadded	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
paymentmethod_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
user_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
total	DECIMAL(15,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
subtotal	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
discount	DECIMAL(5,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
invoicestatus_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
bank_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
chequeno	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
chequeamount	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
cashamount	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
balance	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
duepayment	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
deliverycharges	DECIMAL(20,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.5: Invoice Table

Figure B.6 Vehicle table that used to store details of vehicles.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
vehicletype	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
vehiclemakerid	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
regno	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
insuranceno	VARCHAR(30)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
year	INT(4)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
vehicltransmission_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
color	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
loadmax	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
vehicestatus_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
detail	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
ownersname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
telephone	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
nic	VARCHAR(30)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
address	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.6: Vehicles Table

Figure B.7 Purchase order table structure

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
ponumber	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
total	DECIMAL(10,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
postatus_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
invoice_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
supplier_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.7: PO Table

Figure B.8 Privilege table structure

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
roles_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
modules_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
sel	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
ins	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
upd	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
del	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.8: Privilege table

Figure B.9 Employee table structure that store employee details

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
employeename	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
nic	CHAR(12)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
gender_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
dob	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
civilstatus_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
address	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
mobile	CHAR(12)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
land	CHAR(12)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
email	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
image	BLOB	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
designation_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
assignedate	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
employeestatus_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
firstname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
lastname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
midname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.9: Employee Table

Figure B.10 Customer table structure that use to store customer details

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
nic	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
address	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
telephone	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
active	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Figure B.10:Customer Table

# APPENDIX C – USER

## DOCUMENTATION

This section describes how the user operates this system.

Figure C.1 -The user should provide the correct username and password to gain authorized access to the system. Once logged in, the user can access system.

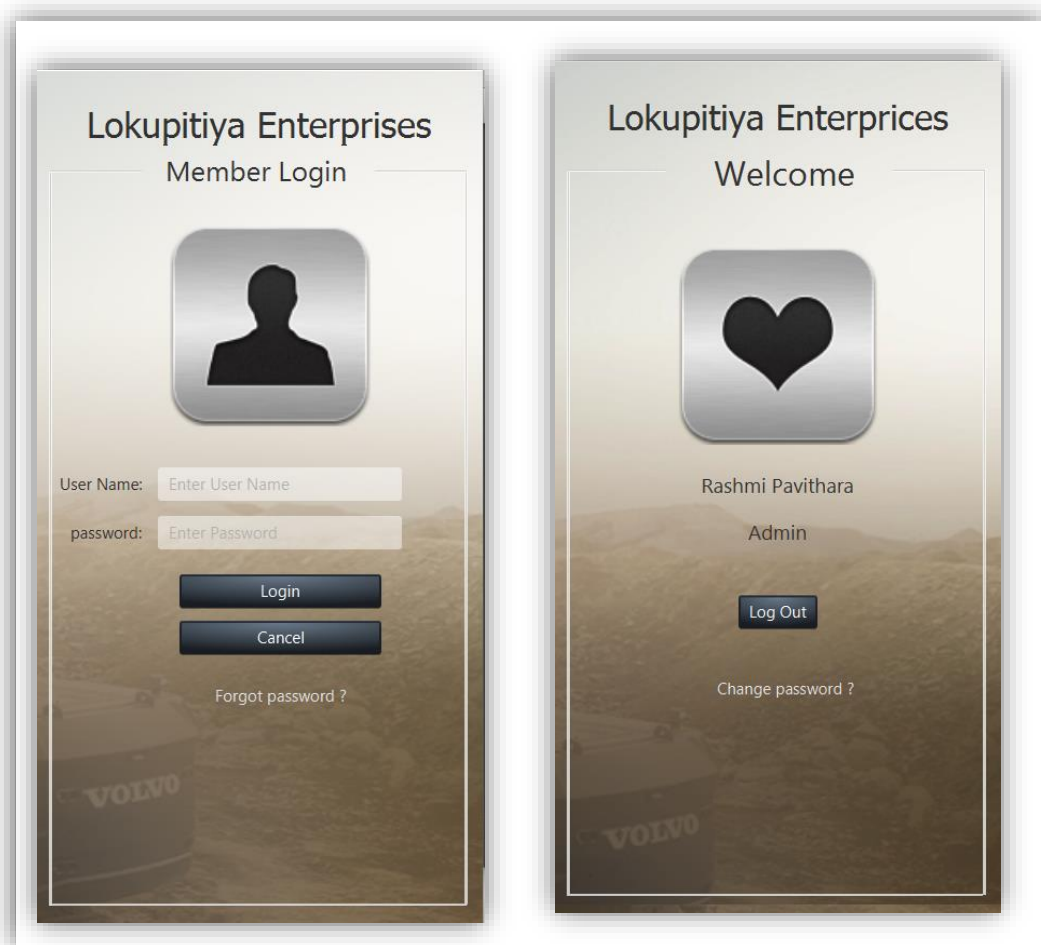


Figure C.1: Login window and login success window



Figure C.2 show three buttons and they are for

- First Button - Drawer button that opens the login UI
- Second Button - Minimize the System
- Third Button - Exit from the System



Figure C.2: The Three Buttons located on the right top of the system

Figure C.3 is the forget password User Interface that used to change password of the users if they forget the password.

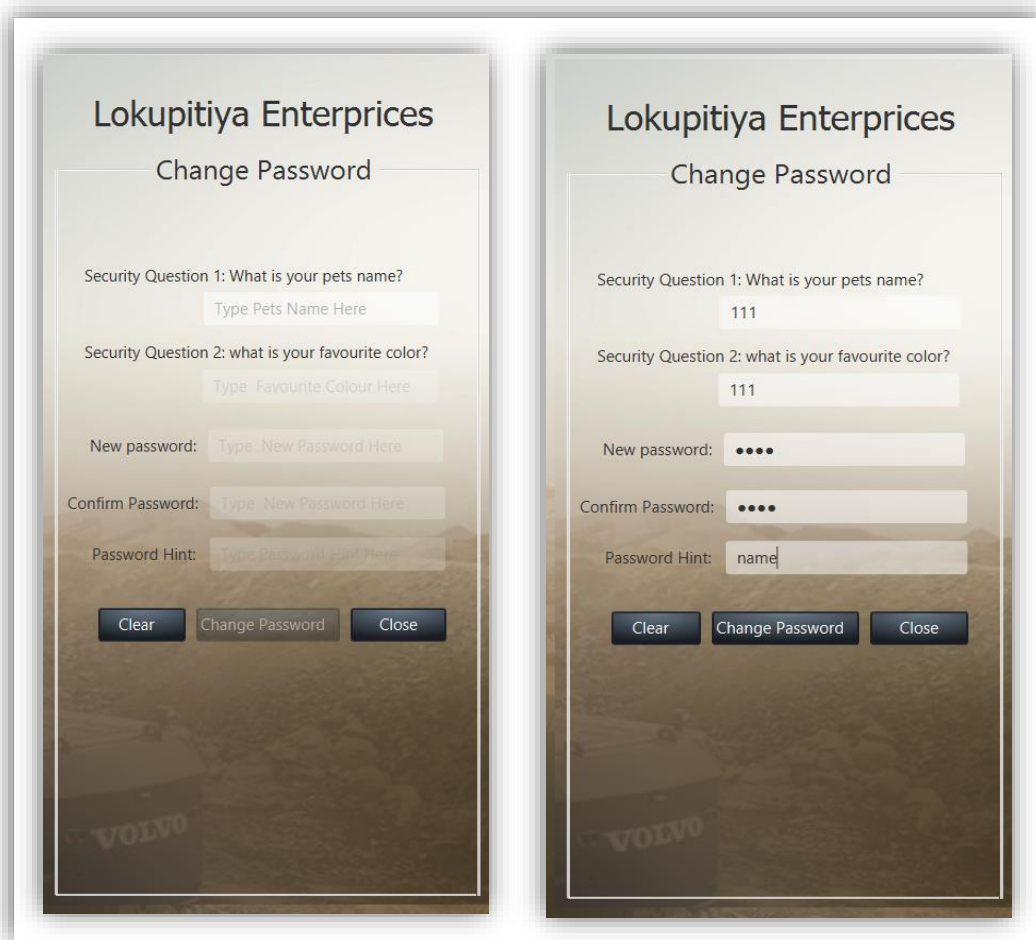


Figure C.3: Forget Password UI

Figure C.4 Shows the Main window interface of the System after log in.

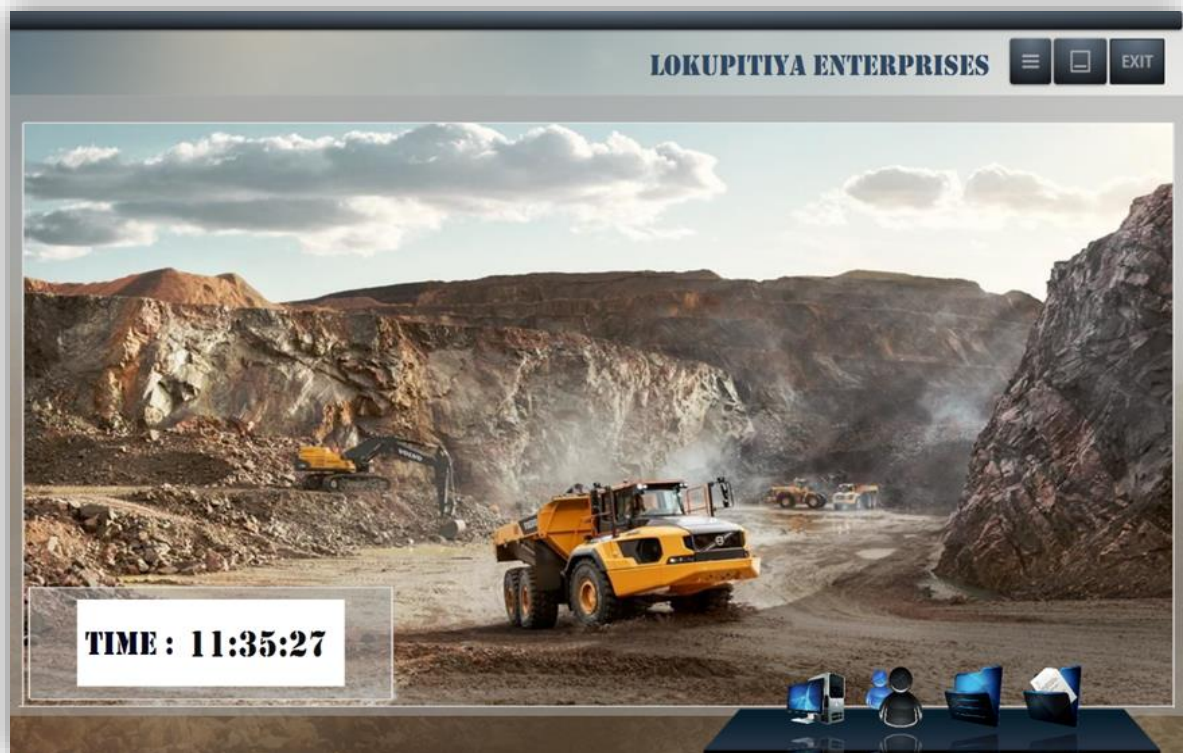


Figure C. 4: Main Window UI

Figure C.5 is the three main categories of the system different users get different privileges to access the system so some of them may not available for particular users.

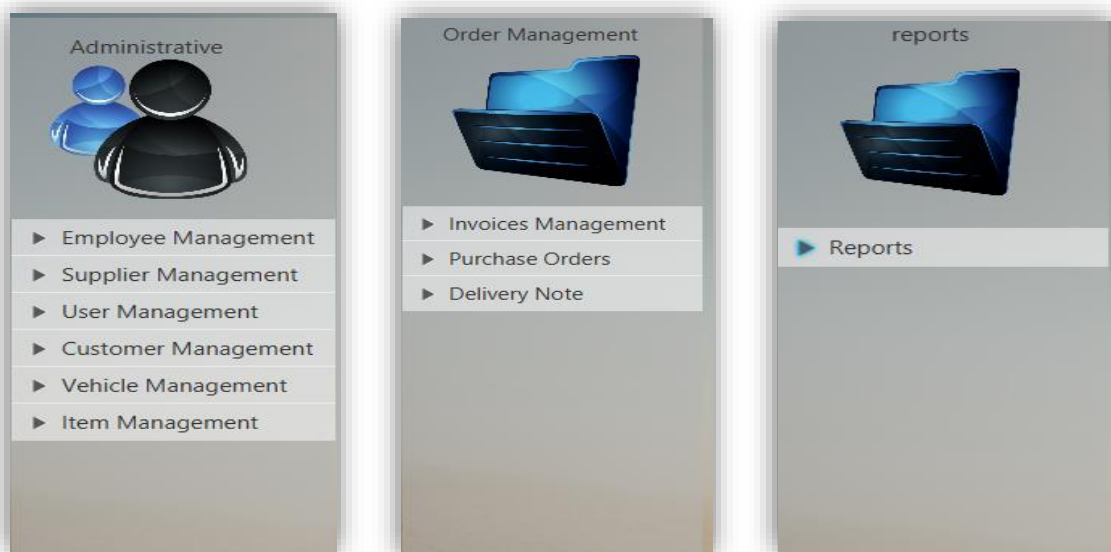


Figure C.5: Three Main Categories

# APPENDIX D – MANAGEMENT

## REPORTS

### Due Payment Report

The system allows users such as managers generate various types of reports in order to use for the decision-making process. As system gives daily, monthly and yearly reports after manipulating them, management can use them to analyze and identify the trends, patterns and seasonal variations of the process and forecast future finance management situations.

Figure D.1 is such report that helps to analyze due payments.

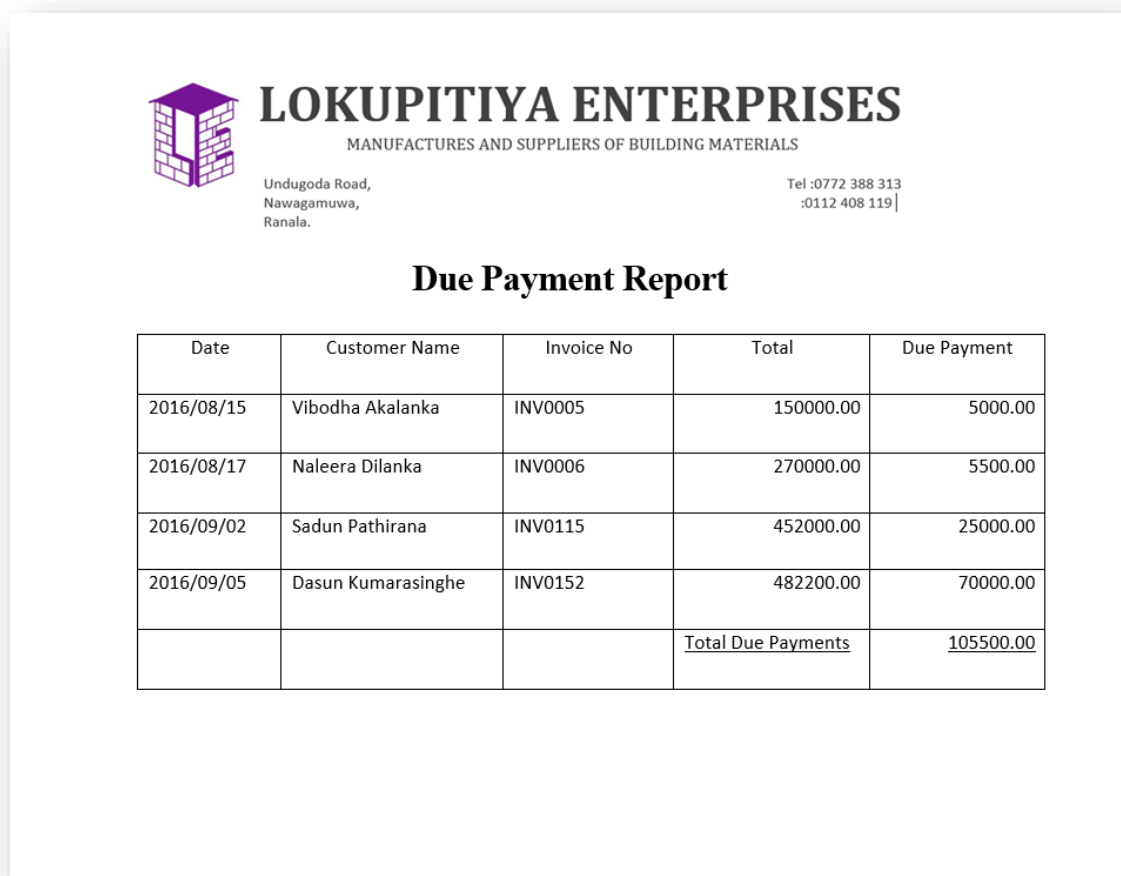


Figure D.1: Due Payment Report

Figure D.2 shows the details of all vehicles of the company.

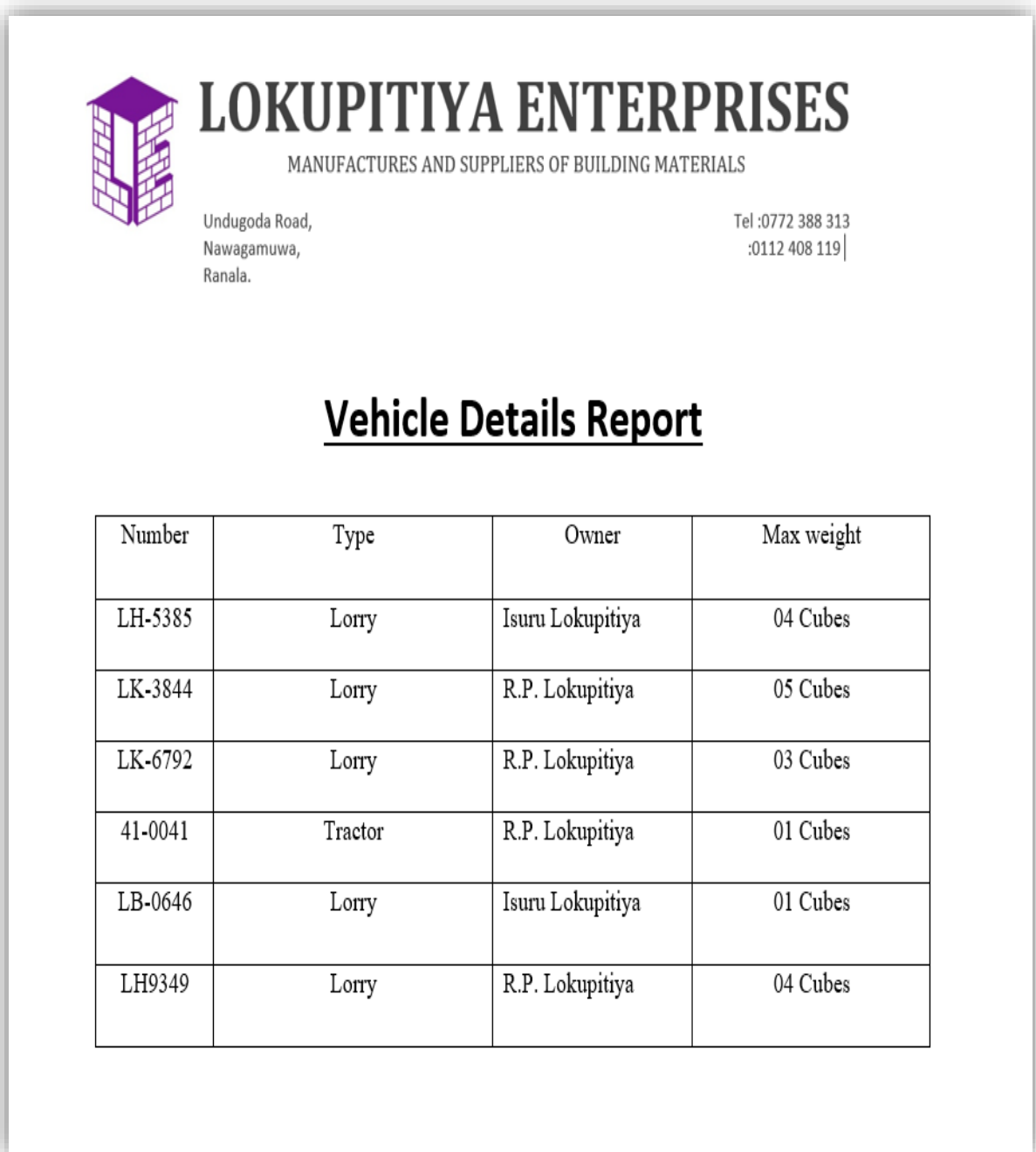


Figure D.2: vehicle Detail report

Figure D.3 shows the sales of current month and the last month. That Support to get decisions of sale.

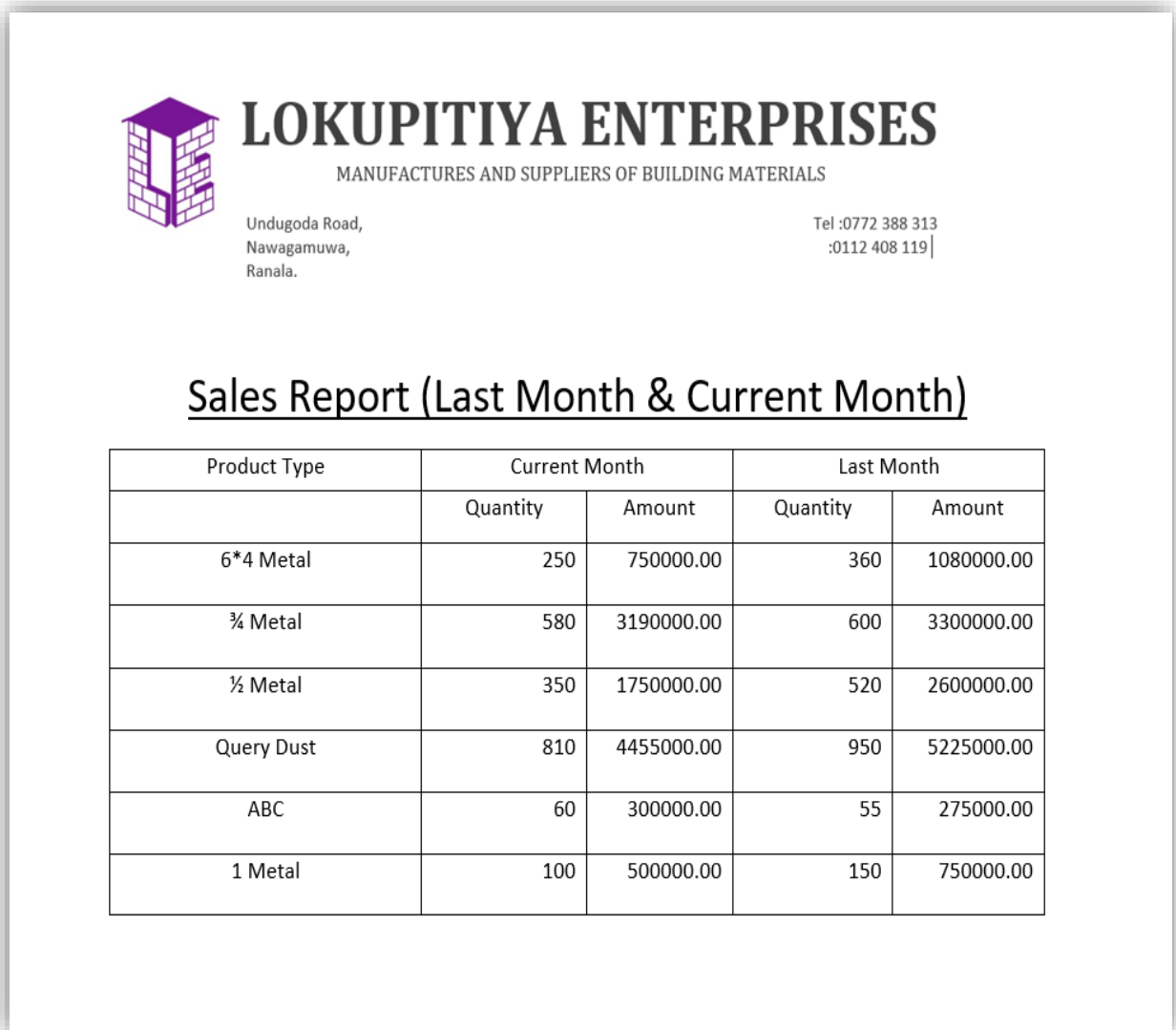


Figure D.3 :Sales report

Figure D.4 shows the monthly profit of the company.

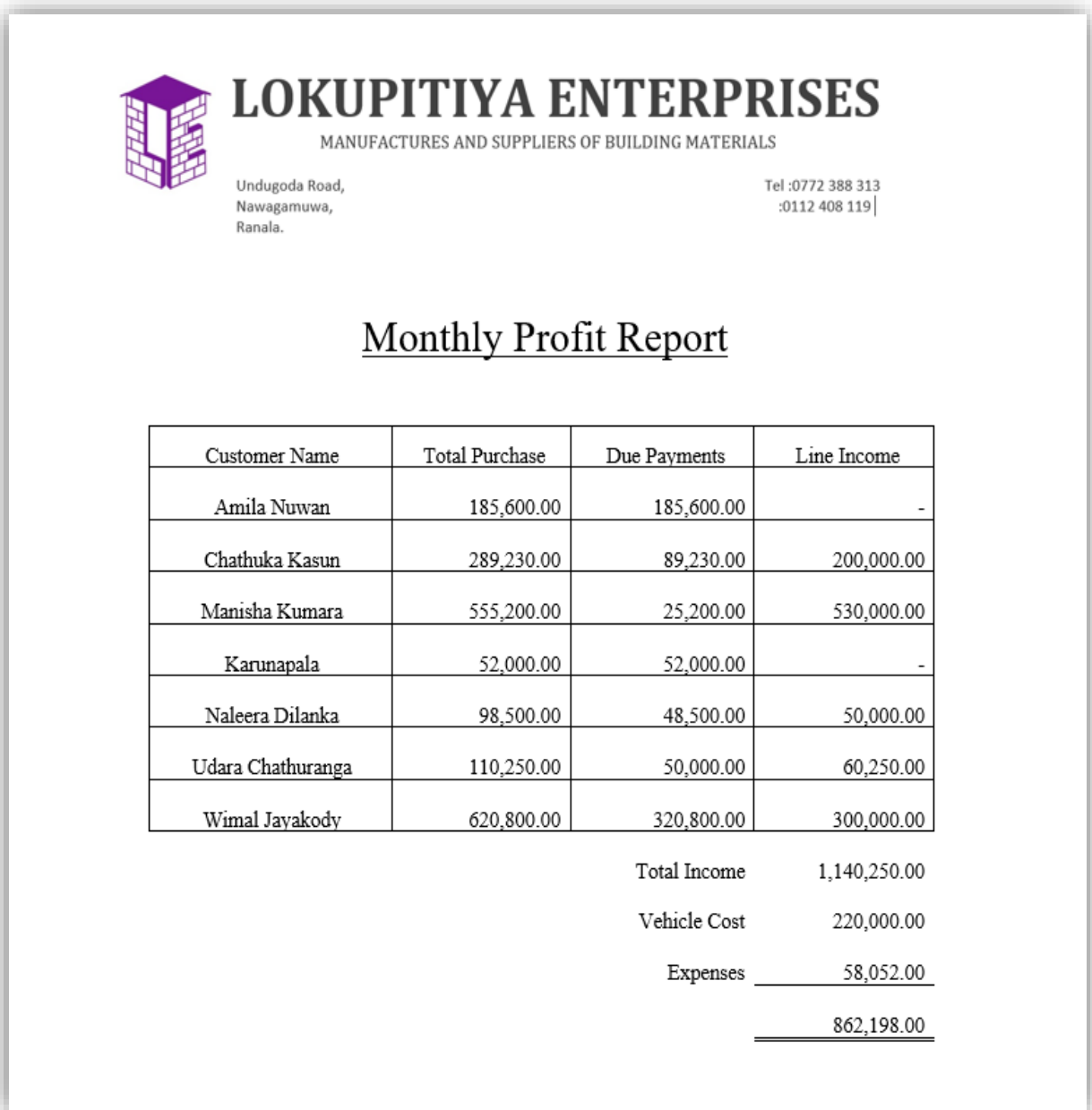





Figure D.4: Monthly profit report

# APPENDIX E – TEST RESULTS

## E.1 Test case and Result for Login Module

Test No	Test Description	Status
1	Initially Login button is disabled. 	Pass
2	When typed something on Username field and Password Field. Login button will be enable. 	Pass
3	Typed only password or username login button disable 	Pass


4	<p>Typed incorrect password or username three times start a 10s counting and disable login</p> 	Pass
5	User Click Close Button to cancel the login	Pass
6	User Click on Forgot Password Label then redirect to forget password UI	Pass

Table E.1: Test Case and Result for Login Module



**E.2 Test case and Result for Purchase Order Module**

<b>Test No</b>	<b>Test Description</b>	<b>Status</b>
1	Only can login as a privilege user	pass
2	The combo box “Select invoice” filled with invoices that doesn’t have purchase orders.	Pass
3	Auto generate invoice number	Pass
4	Auto fill the “Date added”	Pass
5	Auto fill “Required date”	Pass
6	Press “Add PO” then the table fill with the current purchase order	Pass
7	When click the on the purchase order in the table the details of the purchase order filled in the right-side pane	Pass

Table E. 2: Test Case and Result for Purchase Order Module

**E.3 Test case and Result for Invoice Module**

<b>Test No</b>	<b>Test Description</b>	<b>Status</b>
1	Log in as a privilege user	Pass
2	Auto generated invoice number	Pass
3	Current date auto filled	Pass
4	All active customers loaded into the combo box	Pass
5	“Add item” button disable until the necessary details filled	Pass
6	When add items and quantity the price auto generates line total	Pass
7	When add discount the total price subtracts the discount	Pass
8	When click allocate vehicles redirect to the vehicle allocation	Pass
9	When the vehicle allocation adds the total auto update	Pass
10	When click the payment method it disables the text fields that necessary automatically.	Pass
11	According to the payment “due payment” and “Balance” auto update	Pass
12	If all the necessary fields filled then press add invoice button to add the invoice to the system	Pass

Table E. 3: Test Case and Result for Invoice Module

## APPENDIX F – CODE LISTING

Major code fragment for anyone who is interested in referring the functionality of the system are contained in this document. Please refer the CD for the Complete set of Codes for all Code Fragment

### Following code shows the NIC java class

User can add old NIC format or new NIC format into the system, then system will be calculated date of birth and Gender of Employee. For this method need NIC number as a parameter. The following codes below shows the NIC java class.

```
package common;

import java.time.LocalDate;
import java.time.Month;

public class NIC {

    private static int year;
    private static Month month;
    private static int day;
    private static String gender;
    private static LocalDate date;

    public static LocalDate getDate(String NIC) {

        String nic = NIC;

        if(nic.matches("[0-9]{9}[VvXx]")) {

            year = Integer.parseInt("19" + nic.substring(0, 2));
            day = Integer.parseInt(nic.substring(2, 5));

            calculateMonth();
        }
        if(nic.matches(" ")) {

            year = Integer.parseInt(nic.substring(0, 4));
            day = Integer.parseInt(nic.substring(4, 7));

            calculateMonth();
        }

        date = LocalDate.of(year, month, day);
        return date;
    }
}
```

```
public static String getGender(String NIC) {  
    String nic = NIC;  
  
    if(nic.matches("[0-9]{9}[VvXx]")) {  
        day = Integer.parseInt(nic.substring(2, 5));  
  
        if(day > 500) {  
            gender = "Female";  
        }  
        else {  
            gender = "Male";  
        }  
    }  
    if(nic.matches("[0-9]{12}")) {  
        day = Integer.parseInt(nic.substring(4, 7));  
  
        if(day > 500) {  
            gender = "Female";  
        }  
        else {  
            gender = "Male";  
        }  
    }  
  
    return gender;  
}
```

```
private static void calculateMonth() {  
  
    if(day > 500) {  
        day = day - 500;  
    }  
  
    if(day == 0 || day > 366) {  
        System.out.println("Invalid NIC number");  
    }  
    else {  
        if(day > 335) {  
            month = Month.DECEMBER;  
            day -= 335;  
        }  
        else if(day > 305) {  
            month = Month.NOVEMBER;  
            day -= 305;  
        }  
        else if(day > 274) {  
            month = Month.OCTOBER;  
            day -= 274;  
        }  
        else if(day > 244) {  
            month = Month.SEPTEMBER;  
            day -= 244;  
        }  
        else if(day > 213) {  
            month = Month.AUGUST;  
            day -= 213;  
        }  
        else if(day > 182) {  
            month = Month.JULY;  
            day -= 182;  
        }  
        else if(day > 152) {  
            month = Month.JUNE;  
            day -= 152;  
        }  
        else if(day > 121) {  
            month = Month.MAY;  
            day -= 121;  
        }  
        else if(day > 91) {  
            month = Month.APRIL;  
            day -= 91;  
        }  
        else if(day > 60) {  
            month = Month.MARCH;  
            day -= 60;  
        }  
        else if(day < 32) {  
            month = Month.JANUARY;  
        }  
        else if(day > 31) {  
            month = Month.FEBRUARY;  
            day -= 31;  
        }  
    }  
}  
}
```

The following java code is how to calculate the turns of the vehicle when the order is placed by customer.

```
@FXML
private void cmbSelectVehicleAP(ActionEvent event) {
    if (cmbSelectVehicle.getSelectionModel().getSelectedItem() != null) {

        Vehicles selectedVehicle = cmbSelectVehicle.getSelectionModel().getSelectedItem();
        allocatedVehicle.setVehiclesId(selectedVehicle);

        turns = quantity.divide(new BigDecimal(selectedVehicle.getLoadmax()), 2,
            RoundingMode.CEILING).setScale(0, RoundingMode.CEILING).intValue();

        txtTurns.setText(turns.toString());
        allocatedVehicle.setNumberofturns(turns);
    }
}
```

The code below is the method that use to calculate the total distance of the route when a order is placed by customer to deliver.

```
@FXML
private void txtDistanceOfTheRouteKR(KeyEvent event) {
    if (booleanTurns(txtDistanceOfTheRoute.getText())) {
        totalKM = allocatedVehicle.getNumberofturns() * Integer.parseInt(txtDistanceOfTheRoute.getText());
        txtTotalDistance.setText(totalKM.toString());
    } else {
        if (txtDistanceOfTheRoute.getText().isEmpty()) {
            txtTotalDistance.setText("");
        } else {
        }
    }
}
```

the code below is the method that used to calculate the payment of the delivery for allocated vehicles.

```

@FXML
private void txtPerKmKR(KeyEvent event) {
    if (booleanPrice(txtPerKm.getText())) {
        if (event.getCode() == KeyCode.ENTER) {
            Integer othreKm = totalKM - 10;
            BigDecimal other = new BigDecimal(othreKm * Integer.parseInt(txtPerKm.getText()));
            BigDecimal total = allocatedVehicle.getCost().add(other);
            allocatedVehicle.setCost(total);
            lblTotalCost.setText("");
            lblTotalCost.setText(allocatedVehicle.getCost().toString());
        }
        txtPerKm.setStyle(valid);
    } else {
        if (txtPerKm.getText().isEmpty()) {
            lblTotalCost.setText("");
        } else {
            txtPerKm.setStyle(invalid);
        }
    }
}
}

```

This code segment shows the method that used to generate the discount price from the total cost

```

@FXML
private void txtDiscountKR(KeyEvent event) {
    if (booleanDiscount(txtDiscount.getText())) {
        if (event.getCode() == KeyCode.ENTER) {
            BigDecimal disscount = allocatedVehicle.getCost().multiply
(new BigDecimal(txtDiscount.getText()).divide(new BigDecimal(100)));
            BigDecimal disscountedTotal = allocatedVehicle.getCost().subtract(disscount);
            allocatedVehicle.setCost(disscountedTotal);
            lblTotalCost.setText(allocatedVehicle.getCost().toString());
        }
    }
}
}

```

# APPENDIX G - CLIENT

## CERTIFICATION



Figure G.1: Client Certification



# GLOSSARY

**Database** - is an organized collection of data for one or more purposes, usually in digital form.

**Backup** – means keeping a copy of information as an external storage where usually keep in another central computer or in external disks.

**UML** – refers to Unified Modeling Language is a standardized general-purpose modeling language in the field of object-oriented engineering. This includes a set of graphic notation techniques to create visual models of object-oriented software intensive systems.

**Graphical User Interface** - is a type of user interface that allows users to interact with electronic devices with images rather than text commands.

**Relational database management system** - database management system (DBMS) that is based on the relational model (that all data is represented in terms of tuples, grouped into relations.)

**JavaFX**- is a software platform for creating and delivering rich internet applications (RIAs are) that can run across a wide variety of devices

**Object-relational mapping** - is a programming technique for converting data between incompatible type systems in object-oriented programming languages.