Sales and Inventory Management System
For Samanthe Motor Stores(PVT)Ltd

W.D.A.R.Costa
BIT Registration Number: R142045
Index No: 1420453

Name of the Supervisor: H.W.Nalinda Madushanka

2017

This dissertation is submitted in partial fulfillment of the requirement of the
Degree of Bachelor of Information Technology (external) of the
University of Colombo School of Computing.
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: 25.10.2017

Name of Candidate: W. D. A. R. Costa

Signature of Supervisor: ___________________________ Date: 25.10.2017

Name of Supervisor: H. W. National Madushanka

Figure 1 Declaration
ABSTRACT

Samanthi Motor Stores is a well-known spare parts shop in Jaela area. It was established in 2007. At present they provide various kinds of goods such as accelerator, brake pedal, gear stick, battery, clutch, headlights, etc. Due to its fast growth, it has become a very popular business place by now.

Even if their business was a fast growing one, they still use a manual system to manage their day-to-day business activities such as customer order management, issue bills for sales, stock management, etc. So the existing method for handling the business activities is problematic due to the manual methods.

Proposed system will mainly facilitate inventory management and stock management in the business. This system will also help to overcome existing problems like lack of efficiency and data redundancy. This will be developed as a standalone system. This system also supports for tracking order management, user privilege management & more.

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.

This business solution can control the accessibility of information relevant to the users who deal with the system and will ensure the security of the information.

NetBeans IDE (Integrated Development Environment) and JavaFX Scene Builder was used to implement the proposed system in Java, JavaFX, MySQL, Hibernate technologies, MVC architecture and Object Oriented approach. Unified Modelling Language was used for analysis and design the system. Jasper reports used to report generation. The proposed system is windows based solution. The proposed system will achieve the client’s functional and non-functional requirements and provide an efficient and user friendly working environment. The system has been providing excellent solution for Sales and Inventory Management activities.
ACKNOWLEDGEMENT

I would like to take this space to acknowledge and extend my heartiest gratitude to those who have helped me in various 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. H.W.Nalinda Madushanka 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. K.V.A.B.L.Perera and all the employees of Samanthe Motor Stores(PVT) Ltd, who gave me the opportunity to develop this system for their spare parts shop. 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 of my friends at Earth Institute and specially express my gratitude to my well educated lecturers who helped me in a number of ways and encouraged me to complete the project successfully.

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 CONTENT

**Contents**

**DECLARATION** .......................................................................................................................... ii  
**ABSTRACT** ................................................................................................................................. iii  
**ACKNOWLEDGEMENT** ............................................................................................................... iv  
**TABLE OF CONTENT** ............................................................................................................... v  
**LIST OF FIGURES** .................................................................................................................. x  
**LIST OF TABLES** ..................................................................................................................... xv  
**LIST OF ACRONYMS** ............................................................................................................... xvi

**CHAPTER 1 – INTRODUCTION** ................................................................................................. 1  
1.1 The Client ............................................................................................................................... 1  
1.6 Structure of the Dissertation ................................................................................................. 1  
1.2 The Problem Domain ........................................................................................................... 2  
1.3 Motivation ............................................................................................................................. 2  
1.4 Objectives ............................................................................................................................. 3  
1.5 Scope ..................................................................................................................................... 3

**CHAPTER 2 – ANALYSIS** .......................................................................................................... 4  
2.1 Introduction .......................................................................................................................... 4  
2.2 Techniques used to gather requirements ............................................................................. 4  
2.2.1 Reading company documents ...................................................................................... 4  
2.2.2 Interviews and Discussion ......................................................................................... 4  
2.2.3 Online Researches and Read Journals ..................................................................... 5  
2.2.4 Observation .................................................................................................................. 5  
2.3 Existing System .................................................................................................................... 5  
2.4 Types of Requirements ......................................................................................................... 7  
2.4.1 Functional Requirements ............................................................................................ 7  
2.4.2 Non Functional Requirements .................................................................................... 7  
2.5 Similar Systems .................................................................................................................... 8
4.2 Hardware and software Requirements ................................................................. 32
  4.2.1 Hardware Requirements ................................................................................. 32
  4.2.2 Software Requirements ................................................................................ 32
4.3 Development Tools ............................................................................................. 32
  4.3.1 NetBeans ........................................................................................................ 32
  4.3.2 Java Language .................................................................................................. 33
  4.3.3 MySQL Server .................................................................................................. 34
  4.3.4 Hibernate Framework(ORM) .......................................................................... 34
  4.3.4 JavaFX Scene Builder ...................................................................................... 35
  4.3.5 Jasper Reports .................................................................................................. 35
4.4 Code Features ....................................................................................................... 36
  4.4.1 MVC Architecture ........................................................................................... 36
  4.4.2 Data Layer Implementation ............................................................................ 37
  4.4.3 User Interface Layer Implementation .............................................................. 39
  4.4.4 Controller Layer Implementation .................................................................... 41
  4.4.5 Reused Code and Modules ............................................................................. 43
  4.4.6 Main Functions ............................................................................................... 43

CHAPTER 5 – EVALUATION ....................................................................................... 47
  5.1 Introduction ........................................................................................................ 47
  5.2 Software Testing ................................................................................................. 47
  5.3 Techniques of Software Testing ......................................................................... 47
  5.4 Types of Testing .................................................................................................. 48
    5.4.1 Unit Testing .................................................................................................... 48
    5.4.2 Integration Testing ....................................................................................... 48
    5.4.3 System Testing .............................................................................................. 48
    5.4.4 Acceptance Testing ...................................................................................... 48
    5.4.5 Regression Testing ...................................................................................... 48
LIST OF FIGURES

Figure 1 Declaration ........................................................................................................ii

Figure 2. 1 Use Case Diagram for Create GRN................................................................. 6
Figure 2. 2 Use Case diagram for Create PO (Purchase Order) ................................. 6
Figure 2. 3 Retail Pro Software ...................................................................................... 8
Figure 2. 4 Retail Pro Software ...................................................................................... 9
Figure 2. 5 Crazy Vendor Software ................................................................................ 9

Figure 3. 1 The Waterfall Model ................................................................................. 11
Figure 3. 2 Evolutionary development .............................................................. 11
Figure 3. 3 Component-based Software Engineering .................................................. 12
Figure 3. 4 Rational Unified Model ............................................................................. 13
Figure 3. 5 Use Case Diagram for Overall System..................................................... 16
Figure 3. 6 Activity Diagram for User Login ............................................................... 19
Figure 3. 7 Activity Diagram for add new Purchase Order ........................................... 20
Figure 3. 8 Class Diagram for Sales and Inventory Management System ............... 21
Figure 3. 9 Sequence Diagram for Invoicing ............................................................... 22
Figure 3. 10 Sequence Diagram for Purchase Order .................................................... 23
Figure 3. 11 Entity Relationship Diagram Sales and Inventory Management System. 24
Figure 3. 12 Simple Color Variations ......................................................................... 26
Figure 3. 13 Error Notification ..................................................................................... 26
Figure 3. 14 Success Notification ............................................................................... 26
Figure 3. 15 Confirmation Alert Box ............................................................................. 27
Figure 3. 16 Login UI ................................................................................................. 27
Sales and Inventory Management System

Figure 3. 17 Item UI.................................................................28
Figure 3. 18 Employee UI...........................................................28
Figure 3. 19 Purchase Order UI ..................................................29
Figure 3. 20 Invoice UI...............................................................29
Figure 3. 21 Sales Return UI.........................................................30
Figure 3. 22 GRN UI .................................................................30
Figure 3. 23 Purchase Return UI ...................................................31
Figure 3. 24 Payment Management UI ..........................................31

Figure 4. 1 Hibernate Framework ..............................................34
Figure 4. 2 MVC Architecture ...................................................36
Figure 4. 3 Database Implementation .........................................37
Figure 4. 4 Hibernate Configuration Code ....................................38
Figure 4. 5 Part of an Entity Class Code ......................................39
Figure 4. 6 FXML Code segment for Category UI ............................40
Figure 4. 7 Part of a Controller Class ( Variables related to FXML ) ....40
Figure 4. 8 Part of a Controller Class ( Event Handler of an FXML element ) ... 41
Figure 4. 9 Part of the CSS code ( Style for the Button ) .................41
Figure 4. 10 Code segment for creating a hibernate Session ..............42
Figure 4. 11 Part of a DAO Class ( Save object to the Database ) ..........42
Figure 4. 12 Part of a DAO Class ( Select Object from Database ) .........43
Figure 4. 13 Part of the Notifications Class Code ............................43

Figure 4. 1 Hibernate Framework ..............................................34
Figure 4. 2 MVC Architecture ...................................................36
Figure 4. 3 Database Implementation .........................................37
Figure 4. 4 Hibernate Configuration Code ....................................38
Figure 4. 5 Part of an Entity Class Code........................................................................................................... 39
Figure 4. 6 FXML Code segment for Category UI .......................................................................................... 40
Figure 4. 7 Part of a Controller Class ( Variables related to FXML ) ............................................................ 40
Figure 4. 8 Part of a Controller Class ( Event Handler of an FXML element ) .......................................... 41
Figure 4. 9 Part of the CSS code ( Style for the Button ) ................................................................................. 41
Figure 4. 10 Code segment for creating a hibernate Session ........................................................................... 42
Figure 4. 11 Part of a DAO Class ( Save object to the Database ) ................................................................. 42
Figure 4. 12 Part of a DAO Class ( Select Object from Database) ................................................................. 43
Figure 4. 13 Part of the Notifications Class Code .......................................................................................... 43
Figure 4. 14 Part of the User Role code ........................................................................................................... 44
Figure 4. 15 Part of the Sales return code ....................................................................................................... 44
Figure 4. 16 Part of the Brand Code ................................................................................................................ 44
Figure 4. 17 Part of the Supplier Code ............................................................................................................ 45
Figure 4. 18 Part of the Purchase Order Code ............................................................................................... 45
Figure 4. 19 Part of the Main Window Code ................................................................................................. 46
Figure 4. 20 Part of the Invoice Code ............................................................................................................. 46

Figure A. 1 Installation progress of JRE (Step1) .............................................................................................. 61
Figure A. 2 Installation progress of JRE (Step2) ............................................................................................... 61
Figure A. 3 Installation progress of JRE (Step3) ............................................................................................... 62
Figure A. 4 Installation progress of JRE (Step4) ............................................................................................... 62
Figure A. 5 Installation Progress MySQL Server (Step1) ............................................................................... 63
Figure A. 6 Installation Progress MySQL Server (Step2) – License Agreement ........................................... 63
Figure A. 7 Installation Progress MySQL Server (Step3) ............................................................................... 64
Figure A. 8 Installation Progress MySQL Server (Step4) ............................................................................... 64
Figure A. 9 Installation Progress MySQL Server (Step 5) ............................................................................. 65
Figure A. 10 Installation Progress MySQL Server (Step 6) .......................................................................... 66
Figure A. 11 Installation Progress MySQL Server (Step 7) ........................................... 66
Figure A. 12 Installation Progress MySQL Server (Step 8) ........................................... 67
Figure A. 13 Installation Progress MySQL Server (Step 9) ........................................... 67
Figure A. 14 Installation Progress MySQL Query Browser ........................................... 68
Figure A. 15 Installation Progress MySQL Query Browser (Step 2) ............................... 68
Figure A. 16 Installation Progress MySQL Query Browser (Step 3) ............................... 69
Figure A. 17 Installation Progress MySQL Query Browser (Step 4) ............................... 69
Figure A. 18 Installation Progress MySQL Query Browser (Step 5) ............................... 70

Figure B. 1 Activity Diagram for log into the System ....................................................... 71
Figure B. 2 Sequence Diagram for the Invoicing ............................................................ 72
Figure B. 3 Use Case Diagram for Owner ....................................................................... 73

Figure C. 1 User Login Form .......................................................................................... 87
Figure C. 2 Main Window ............................................................................................... 88
Figure C. 3 Item Module ............................................................................................... 89
Figure C. 4 Goods Received Note Module ..................................................................... 89

Figure D. 1 All Employee Report of the system ............................................................. 90
Figure D. 2 Purchase Order Report of the system .......................................................... 91

Figure E. 1 Overall Test Results from the Staff Members ............................................. 95
Figure E. 2 User Acceptance Test Results of the System Administrator ..................... 96

Figure F. 1 Code of Main Class ...................................................................................... 97
Figure F. 2 Code of NIC Class ....................................................................................... 98
Figure F. 3 Code of Animate Class ................................................................................. 99
Figure F. 4 Code of Security Class ................................................................. 100
Figure F. 5 Code of Database Connection ....................................................... 100
Figure F. 6 Code of CommanDao Class .......................................................... 101
Figure F. 7 Code of Brand UI Controller ......................................................... 102

Figure G. 1 Client Certification .................................................................... 103
LIST OF TABLES

Table 2.1 Comparison between similar system ......................................................... 10

Table 3.1 Use Case Narratives for Login into the System ........................................ 17
Table 3.2 Use Case Narratives for Inserting a new Item ........................................ 17
Table 3.3 Use Case Narratives for Create Purchase Order ....................................... 18

Table 5.1 Test Results for modules ....................................................................... 50
Table 5.2 Test Cases & Test Result for Login Module ............................................. 51
Table 5.3 Test Cases & Test Result for Brand Module ........................................... 52
Table 5.4 Test Cases & Test Result for Item Module ............................................. 53
Table 5.5 Test Cases & Test Result for Category Module ...................................... 54
Table 5.6 Test Cases & Test Result for Purchase Order Module ............................ 56

Table E1 Test Results for Login Form ................................................................. 93
Table E2 Test Results for Item Module ............................................................... 94
LIST OF ACRONYMS

IDE - Integrated Development Environment

OOD - Object Oriented Design

SQL - Structured Query Language

UML - Unified Modeling Language

RAD – Rapid Application Development

RUP – Rational Unified Model

RAM - Random Access Memory

GB - Giga Bytes

GHz – Gigahertz

GRN – Goods Received Note

UI – User Interface

MVC - Model View Controller

NF - Normalization Form

NIC – National Identity Card
CHAPTER 1 – INTRODUCTION

1.1 The Client

Samanthi Motor Stores is a known name around Jaela area if you ever search for a spare parts shop. It has been there for more than 10 years now. By now they have got so many customers around Jaela & other areas around it. At first they had only few items. Along with the time, now they have widen their business boundaries by not limiting their business expertise only to the spare parts. They provide any kind of spare parts and much more useful items.

At present Samanthi Motor Stores manages their business functions in a non-automated method. So it has become a great urge to automate the system in order 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.

1.6 Structure of the Dissertation

This report shows the works that carried out during each phases of the project when creating the Sales and Inventory Management System for Samanthi Motor Stores.

Chapter 02 - Analysis

This chapter provides an analysis of the functional and non-functional requirements. It will also describe about fact gathering techniques and existing manual system.

Chapter 03 - Design

This chapter will provide class diagrams, activity diagrams, Entity Relationship diagrams and other related diagrams regarding to the proposed system. System requirements, system architecture, sub systems, database structure and user interfaces are other topics describe in this chapter.

Chapter 04 - Implementation

This chapter will describe about all major codes, module structures, the implementation environment, development tools and platform used in the project.
Chapter 05 – Evaluation

Evaluation chapter will provide details about testing methodologies used in the test cases and all the test cases in the project.

Chapter 06 – Conclusion

This chapter will conclude the dissertation with a critical evaluation of the system and suggestions for any future enhancements.

1.2 The Problem Domain

As they are using a non-automated system, this manual obsolete system causes numerous problems. It is a time consuming, inefficient process to manage processes in the spare parts shop including item management, employee management, transaction management and stock management…etc.

1.3 Motivation

There is a great need of an automated system for the Samanthi Motor Stores according to the following reasons which were identified during the interviews with the owner of the spare parts shop.

Currently there is no computerized mechanism to manage their items and employees. Also there is no way to record the details of their suppliers, stocks and to issue bills for the transactions except the paper work.

Not having an automated system to calculate their stocks have led them to manually calculate their stocks most of the times. This is a very unreliable method because of the human errors could occur in any of these occasions. Because of this issue it’s hard to predict their future purchases of low stock items.

To overcome these issues, Samanthi Motor Stores needs a Spare Parts Management System which will minimize the time and cost that will have to spend for
the operations and mainly focus on improving the efficiency, reliability and the productivity of their business.

1.4 Objectives

- Main objective of implementing this system is to reduce the time consumption and human effort in inventory control and stock management processes.
- Provide secure, user friendly and easy access to the system.
- Improve item management, supplier management and stock management.
- Improve the efficiency of daily business processes.
- Improve the daily, weekly, monthly and annual report generation in sales and purchases.
- Increase the accuracy and privacy of spare parts shop management and decision support process.

1.5 Scope

- The proposed system will aid the spare parts shop to efficiently conduct its daily business functions, as mentioned below, these will be implemented in the system;
- System will solve all inventory related difficulties faced daily in the business.
- User accounts and access control mechanisms will be maintained to prevent unauthorized access to the system.
- All the details of employees, suppliers, order details, categories, items, sales and purchase will be managed.
- Transactions of the items between customers, suppliers and distribution of items within the business will be adjusted in the system accordingly.
- Manage notifications for reordering of items when they reach a minimum quantity for ordering.
- Create item orders, deliveries and requests.
- Generate reports for all the functions within the business.
CHAPTER 2 – ANALYSIS

2.1 Introduction

Requirement analysis is a very important and a critical phase in the Software Development Life Cycle. Success of requirement analysis phase will impact to the success of the other phases. The term is most commonly used in the context of commercial programming, where software developers are often classed as either system analysts or programmers. The systems analysts are responsible for identifying requirements and producing a design. The programmers are then responsible for implementing it. Requirements are divided as functional and non-functional requirements. Then they were structured and prioritized for future reference. Following discusses the types used for requirement analysis. This chapter includes requirements gathering techniques, analysis of requirements, functional and non-functional requirements in this project.

2.2 Techniques used to gather requirements

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. System will built using user feedbacks. Otherwise it is difficult to develop a good product which satisfies the client.

2.2.1 Reading company documents

Payment bills, log entries, past reports etc. are very useful to identify the business payment process and procedure of keeping records in the spare parts shop. To have an understanding about the existing system is always important as well as the interviews other. Because the analyst should have a clear idea about the organizational processes.

2.2.2 Interviews and Discussion
Interviewing users can be very useful when there are small amount of users use the system. The employees of the company who are involved with the existing system are interviewed in this method. In this stage there should be good communication skill between user and interviewer. There by user requirements, difficulties in the current system and new ideas were discussed and identified. Extended discussions were carried out with some user groups for clarifications and correctness of gathered information.

2.2.3 Online Researches and Read Journals

An online research for similar systems was done to gain knowledge on features and new ideas related to inventory management systems.

- Computer journals, reference books
- Internet

2.2.4 Observation

When the users doing there day to day work in current process one can monitor their works and get idea about requirement of users. However this technique may not success at all time because people don’t like their work to be monitored by others.

2.3 Existing System

Samanthi Motor Stores is using manual system for their all process. They record their all details in books with a pre-defined format such as Employee details, Customer details, Supplier details…etc.

There are lot of identified problem in their manual system, which can be cause to many human mistakes. Employee might enter incorrect details. This will increase the data redundancy. The Current system when a transaction occurs system updates only sales details and it does not update the inventory system. So, they cannot get necessary information on inventory level and reorder levels. So, all purchasing function like issuing purchase, GRN and invoicing are done manually. After receiving goods, they
manually create a GRN with the particular purchase orders by working with many recorded data.
From the manual system, very difficult to provide statistical report such as Daily reports, Monthly reports and variety of charts are needed to make decision by the managers to enhance their productivity.
Following Figure 2.1 Use Case diagram for Create GRN in current manual system.

![Figure 2.1 Use Case Diagram for Create GRN](image)

Following Figure 2.2 Use Case diagram for Create PO (Purchase Order)

![Figure 2.2 Use Case diagram for Create PO (Purchase Order)](image)
2.4 Types of Requirements

2.4.1 Functional Requirements

The functional requirements describe what the system should do. Further Functional requirements should include functions performed by specific screens, outlines of workflows performed by the system, and other business or compliance requirements the system must meet.

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionalities that define what a system is supposed to accomplish. [11].

Functional requirements of the Sales and Inventory Control Management System are as follows.

- Manage Item Details
- Manage Supplier Details
- Manage Purchase Details
- Manage Sales Details
- Manage Sales Return Details
- Manage Stock Details and Notify Reorder Levels of the Items
- Manage Employee Details
- Manage User Privileges
- Produce Reports and Charts
- Issue Bills

2.4.2 Non Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the
operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behavior or functions. [12].

Non-Functional requirements of the Sales and Inventory Control Management System are as follows.

- Simple and User Friendly User Interface
- Maintainability
- Accuracy
- Can Run on Many Windows Based Operating Systems
- Easy Installation through a Wizard
- Backup and Restore the Database
- Security

2.5 Similar Systems
- 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. [1].

Following screenshots such as Figure 2.3 and 2.4 Shows some windows of above mentioned system.
Odoo Inventory

Odoo Inventory management system provides a great list of important features such as Sales management, Inventory Management, Invoice Management etc. Following screenshots such as Figure 2.6 and 2.7 Shows some windows of above mentioned system. [2]

Figure 2. 4 Skyware Inventory

Figure 2. 5 Odoo Inventory
Comparison between Similar Systems

<table>
<thead>
<tr>
<th>Features</th>
<th>Skyware</th>
<th>odoo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Employee.</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Manage Users.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Manage Item.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Manage Products.</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Notifications.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Manage Accounts</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reporting</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>User Guidance</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>User Friendliness.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2.1 Comparison between similar system
CHAPTER 3 – DESIGN

3.2 Overview of the Process Models

*The waterfall model* - this takes the fundamental process activities of specification, development, validation and evolution and represents them as separate Process phases such as requirements specification, software design, implementation, testing and soon.

![Waterfall Diagram](image)

*Evolutionary development* - this approach interleaves the activities of specification, development and validation. An initial system is rapidly developed from abstract specifications. This is then refined with customer input to produce a system that satisfies the customers’ needs.

![Evolutionary Development Diagram](image)

*Component-based software engineering* - this approach is based on the existence of a significant number of reusable components. The system development process focuses on integrating these components into a system rather than developing them from scratch.
These three generic process models are widely used in current software engineering practice. They are not mutually exclusive and are often used together, especially for large systems development.

![Component-based Software Engineering](image)

3.2.1 Process Model for the System

Among all other software development methodologies, the Rational Unified Process (RUP) was selected as the process model by considering a lot of advantages it consists. It is an iterative software development framework invented by Iva Jacobson, Jim Rambaugh and Greedy Boosh. Diagram of RUP is depicted in figure 3.1. It consists of four major phases. These major phases are briefly described as follows: [3].

- **Inception Phase.**
  - “The goal of the inception phase is to establish a business case for the System. You should identify all external entities (people and systems) that will interact with the system and define these interactions”.

- **Elaboration Phase.**
  - “The goals of the elaboration phase are to develop an understanding of the problem domain, establish an architectural framework for the system, and develop the project plan and identify key project risks.”

- **Construction Phase.**
  - “The construction phase is essentially concerned with system design, programming and testing. Parts of the system are developed in parallel and integrated during this phase”.

- **Transition Phase.**
The final phase of the RUP is concerned with moving the system from the development community to the user community and making it work in are all environments.”

Figure 3. 4 Rational Unified Model

Rational Unified Process consists of lots of advantages over other development methodologies; common few advantages are listed below:

- It can develop system features according to customer’s priorities. Therefore it helps developing the highest prioritized system features very early in the development process.
- Requirements can be changed during the process, RUP can manage this changing requirements.
- RUP uses UML (Unified Modeling Language) models to address the static view as well as the dynamic view of the system.
- Throughout the development it ensures the software quality as well as the standard.
- Risk Management can be done easily, so it helps to prevent system failures.
- Switching into previous lifecycle stages within the development process can be achieved.
- Very early saving developing time can be achieved due to its support of component based architecture and system can be conveyed without loss of time to the customer.
3.3 Alternative solutions to the system

When discussing an alternative solution to this system it can categorize into few different sub topics.

- **Maintain system based with old file system**
  The entire shop working process based on the existing file based system can be used by the client. So it can maintain all the work by using paper and generate reports and valuable feedbacks by analyzing those paper works when necessary. By choosing this method the client has to waste time to get even a trace of previous data.

- **Web Based Software Solution**
  Web based system can be also used as a solution for the requirements of the supermarket. But the following drawbacks are occurred in the Web based solution over Stand Alone System.
  1. Additional cost for web hosting.
  2. Cannot use when the Internet connection is down.
  3. Harder to maintain.

3.4 The System as a Stand Alone

A Stand Alone Software Solution were selected to develop the System for the Samanthi Motor Stores. Followings were the major reasons.

- Client mostly preferred to a standalone system.
- Standalone systems not needed to pay additional cost for web hosting.
- It is easy to develop standalone system other than a web based system.
- Can use it without internet facility.
- It can perform much faster than web based system.
- Easy to develop and easy to maintain.

3.5 Object Oriented Analysis and Designing the System

Object-oriented analysis and design (OOAD) is a software engineering approach that models a system as a group of interacting objects. Each object represents
some entity of interest in the system being modeled, and is characterized by its class, its state (data elements), and its behavior. Various models can be created to show the static structure, dynamic behavior, and run-time deployment of these collaborating objects. There are a number of different notations for representing these models, such as the Unified Modeling Language (UML).

3.5.1 Use Case Diagram

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

- Owner
- Manager
- Client
- Cashier
- Supplier
- Store Keeper

Use case diagram provides a view of the core functionalities of stakeholders. From Use case diagram it is easier to identify the system requirements. Other than that co-relation of the stakeholders and their functionalities describe clearly. More details about use case diagrams can be presented by using use case scenarios, they provide more details overview of the mentioned use case such as who are the actors for a specific use case, flow of events.

Overall high-level Use Case diagram for the Proposed system and Main functionalities of system users is given in Figure 3.5
3.5.2 Use Case Narratives

- **Use case narrative for Login**

The first user interface of the system is Login form. User must provide his/her username & password in order to login the system.

Following Table 3. 1 shows Use Case Narratives for Login into the System.

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Login to the System</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actors</strong></td>
<td>All Users</td>
</tr>
<tr>
<td><strong>Overview</strong></td>
<td>Authorized users who have the privilege to access the system</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td></td>
</tr>
</tbody>
</table>
User must have an user account
User must enter valid username and password

Flow Events

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

Post Conditions

User will be redirected to Main Window or back to the Login form it depends on username and password

Table 3. 2 Use Case Narratives for Login into the System

- Use case narrative for new Item
Following table 3.2 shows how to insert a new item into the system. In order to add a new item, user must know the item type, category, supplier and the brand of the item.

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Inserting a new item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors</td>
<td>Back office User</td>
</tr>
<tr>
<td>Overview</td>
<td>Insert a New Item.</td>
</tr>
<tr>
<td>Preconditions</td>
<td>A privileged user must login to the system</td>
</tr>
<tr>
<td>Flow Events</td>
<td>1. Select item type</td>
</tr>
<tr>
<td></td>
<td>2. Select category</td>
</tr>
<tr>
<td></td>
<td>3. Select supplier</td>
</tr>
<tr>
<td></td>
<td>4. Give required information</td>
</tr>
<tr>
<td></td>
<td>5. Press save button</td>
</tr>
<tr>
<td>Post Conditions</td>
<td>New item added to the system</td>
</tr>
</tbody>
</table>

Table 3. 3 Use Case Narratives for Inserting a new Item

- Use case narrative for Purchase Order
This describes how to add a purchase order in the system. User should know who the supplier is and what items are provided by the supplier.

Following Table 3. 4 shows Use Case Narratives for Create Purchase Order.
<table>
<thead>
<tr>
<th>Use Case</th>
<th>Add new purchase order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors</td>
<td>General Manager</td>
</tr>
<tr>
<td>Overview</td>
<td>Purchase order details are added to the system</td>
</tr>
</tbody>
</table>

**Preconditions**

A privileged user must login to the system

Relevant supplier, relevant quantities are required by the system.

**Flow Events**

1. Select the supplier
2. Select the needed items
3. Save Purchase order

**Post Conditions**

New purchase order details add into the system

Table 3.5 Use Case Narratives for Create Purchase Order

3.5.3 Activity Diagram

Activity diagram is basically a flow chart to represent the flow form 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 deals with all type of flow control by using different elements like fork, join etc.

- Rounded rectangle – action
- Diamonds – decision
- Join – join two or more action together
- Fork – split one action in to two or more actions
- Black circle – represent staring point (initial state)
- Encircled black circle – represent ending point (final state)
Following figure 3.6 shows Activity Diagram for User Login.

![Activity Diagram for User Login](image)

*Figure 3.7 Activity Diagram for User Login*
Following figure 3. 8 shows Activity Diagram for add new Purchase Order.

Figure 3. Activity Diagram for add new Purchase Order
3.5.4 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 figure represents the class diagram of the Inventory Control and Stock Management System.

Following figure 3. 10 shows Class Diagram for Sales and Inventory Management System.

![Class Diagram for Sales and Inventory Management System](image-url)
3.5.5 Sequence Diagram

A sequence diagram is a kind of interaction diagram. We can use a sequence diagram to illustrate the dynamic view of a system. 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.

Following figure 3. 12 shows Sequence Diagram for Invoicing.

![Sequence Diagram for Invoicing](image-url)

Figure 3. 13 Sequence Diagram for Invoicing
Following figure 3. 14 shows Sequence Diagram for Purchase Order.

Figure 3. 15 Sequence Diagram for Purchase Order
3.6 Database Diagram

The enhanced entity-relationship (EER) model (or extended entity-relationship model) in computer science is a high level or conceptual data model incorporating extension to the original entity-relationship (ER) model, used in the design of databases. The EER model includes all of the concepts introduced by the ER model. [5].

Following figure 3.16 shows Entity Relationship Diagram Sales and Inventory Management System.
3.6.1 Database Normalization

Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships between them. [4].

- **First Normal Form (1NF)**
  - Main objective of the 1NF is eliminating the repeating groups and multi valued columns and arranges them in a single table, and defines a primary key for identifying each related attribute.

- **Second Normal Form (2NF)**
  - Main objective of the 2NF is eliminating the partial dependencies and creating separate tables and relate tables with a foreign key.

- **Third Normal Form (3NF)**
  - Main objective of the 3NF is eliminating the transitive dependencies.

3.7 User Interface Design

Interface designing part is a critical part of the overall software designing process. Because many user errors can be occurred, if the interface designing is poor. Good interface designing should be match with the user expectation. That is the reason of criticalness of the user interface designing. When making user interface design decisions, you should take into assess the physical and mental capabilities of the people who use software. Following are the user interface design principles.

- **User Familiarity**
  - The interface should use terms and concepts drawn from the experience of the people who use the system.

- **Consistency**
  - The interfaces should be consistent in that wherever possible, comparable operations should be activated in the same way.

- **Minimal Surprise**
  - Users should never be surprised by the behavior of the system.

- **Recoverability**
  - The interfaces should include mechanisms to allow users to recover from errors.
• **User Guidance**
  
  o The interface should provide meaningful feedback when errors occur and provide help facility.

• **User Diversity**
  
  o The interface should provide appropriate interaction facilities for different type of users.

There are several actions were taken in order to ensure the good and user friendly interfaces throughout whole system.

3.7.1 Simple Color Variations

Eye friendly colors were used for this system.

Following figure 3.12 shows simple color variation has used in the system.

![Simple Color Variations](image)

**Figure 3.18 Simple Color Variations**

3.7.2 Notifications

Notifications were used to inform the user about some invalidations, Errors and Successes.

• Show Notifications to inform User

Following figure 3.13 shows error notification in the system.

![Error Notification](image)

**Figure 3.19 Error Notification**

Following figure 3.14 shows success notification in the system.

![Success Notification](image)

**Figure 3.20 Success Notification**
3.7.3 User Confirmation

Confirm Alert Boxes are showed when User Confirmation is required. But its limited to the Mandatory Situations. Following figure 3.15 shows confirmation alert box.

![Confirmation Alert Box](image)

*Figure 3.21 Confirmation Alert Box*

3.7.4 User Login Form

This is a common interface for all users to log into the system. Only authorized users can access into the system. The first interface encountered by the user is the login page. Therefore, by designing and handling errors properly a pleasant feeling about the rest of the system can be created within the user.

The figure 3.16 shows the user interface which will be used for login to the system. For enter the system any user have to give valid user name & password. If user satisfies the authentication test requirements he/she will allow log into the system. Although, if user could not provide the correct user name & password in three times the login button will disable automatically.

![Login UI](image)

*Figure 3.22 Login UI*
3.7.5 Item Form

The figure 3.17 shows the user interface which can be used as Item from. The form provides to insert, update, delete and list all items. Also a validation criteria’s used to ensure correctness and consistency of the user inputs.

3.7.6 Employee Form

The figure 3.18 shows the user interface which will be used for entering the employee details. Employee form provides the facility to insert, update, delete employee details & it is given to retrieve one or more employees. Also a validation system is carried out to ensure correctness & consistency of the user inputs.
3.7.7 Purchase Order Note Form

The figure 3.19 shows the user interface which will be used for entering the PO details. PO form provides the facility to insert PO details. Also a validation system is carried out to ensure correctness & consistency of the user inputs.

![Purchase Order Note](image1.png)

*Figure 3. 25 Purchase Order UI*

3.7.8 Invoice Form

The figure 3.20 shows the user interface which will be used for entering the Invoice details. Invoice form provides the facility to insert Invoice details. Also a validation system is carried out to ensure correctness & consistency of the user inputs.

![Invoice](image2.png)

*Figure 3. 26 Invoice UI*

3.7.8 Sales Return Form

The figure 3.21 shows the user interface which will be used for entering the Sales Return details. Invoice form provides the facility to insert Sales return details.
Also a validation system is carried out to ensure correctness & consistency of the user inputs.

3.7.9 GRN Form

The figure 3.22 shows the user interface which will be used for entering the GRN details. Invoice form provides the facility to insert GRN details. Also a validation system is carried out to ensure correctness & consistency of the user inputs.

3.7.10 Purchase Return Form

The figure 3.23 shows the user interface which will be used for entering the Purchase Return details. Invoice form provides the facility to insert Purchase Return details. Also a validation system is carried out to ensure correctness & consistency of the user inputs.
3.7.11 Payment Management Form

The figure 3.23 shows the user interface which will be used for entering the Payment Management details. Invoice form provides the facility to insert Payment Management details. Also a validation system is carried out to ensure correctness & consistency of the user inputs.
CHAPTER 4 – IMPLEMENTATION

4.1 Introduction

This is the phase that software becomes executable. In this phase software is developed according to the detail design based on the client requirements. Implementation phase is a very time consuming phase in the software development life cycle. The main objective of this phase is transforming the detail design into executable format effectively. When coding the system, using comments is a very important thing to consider, because if we want to rework and change the code it is easy to maintain it. Validation is also an important thing to consider while we programming. Further code should be readable. Java which is an object oriented language has selected as a programming language to develop the transport management system.

4.2 Hardware and software Requirements

4.2.1 Hardware Requirements

- 250 GB Hard Disk
- 8 GB RAM
- Core i7 Processor 2.5GHz

4.2.2 Software Requirements

- Microsoft Windows 10 Operating System
- Java SDK 8
- MySQL server 5.7
- Net Beans IDE
- MySQL Workbench

4.3 Development Tools

4.3.1 NetBeans

NetBeans is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called
modules. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers. The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++ and HTML5. NetBeans is cross-platform and runs on Microsoft Windows, macOS, Linux, Solaris and other platforms supporting a compatible JVM. The editor supports many languages from Java, C/C++, XML and HTML, to PHP, Groovy, Javadoc, JavaScript and JSP. Because the editor is extensible, you can plug in support for many other languages.

The NetBeans Team actively supports the product and seeks feature suggestions from the wider community. Every release is preceded by a time for Community testing and feedback. Over 18 million downloads of the NetBeans IDE to date, and over 800,000 participating developers, the NetBeans project is thriving and continues to grow. With its editors, code analyzers, and converters, you can quickly and smoothly upgrade your applications to use new Java 8 language constructs, such as lambdas, functional operations, and method references. [6]

4.3.2 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 bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. 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. [7].
4.3.3 MySQL Server

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality. [8].

4.3.4 Hibernate Framework (ORM)

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 the 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. [9].

![Hibernate Framework](image)

Figure 4.1 Hibernate Framework
4.3.4 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. Scene Builder is fully written with JavaFX 2.0 APIs to explore and learn about JavaFX objects.

4.3.5 Jasper Reports

Jasper Report 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 a 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. [10].

4.3.6 JPA – Java Persistence API

The Java Persistence API (JPA) is a Java specification for accessing, persisting, and managing data between Java objects / classes and a relational database. JPA was defined as part of the EJB 3.0 specification as a replacement for the EJB 2 CMP Entity Beans specification. JPA is now considered the standard industry approach for Object to Relational Mapping (ORM) in the Java Industry.

JPA itself is just a specification, not a product; it cannot perform persistence or anything else by itself. JPA is just a set of interfaces, and requires an implementation. There are open-source and commercial JPA implementations to choose from and any Java EE 5 application server should provide support for its use. JPA also requires a database to persist. JPA allows POJO (Plain Old Java Objects) to be easily persisted without requiring the classes to implement any interfaces or methods as the EJB 2 CMP specification required. JPA allows an object’s object-relational mappings to be defined through standard annotations or XML defining how the Java class maps to a relational database table. JPA also defines a runtime Entity Manager API for processing queries and transaction on the objects against the database. JPA defines an object-level query language, JPQL, to allow querying of the objects from the database.
JPA is the latest of several Java persistence specifications. The first was the OMG persistence service Java binding, which was never very successful; I'm not sure of any commercial products supporting it.

4.4 Code Features

4.4.1 MVC Architecture

**Model** - Model represents an object or java POJO carrying data. It can also have logic to update controller if its data changes. This layer is independent from other system layers such as, View and Controller. It also governs the rules to access the data objects and perform any kind of operation on them. It knows all details about data which needed to be displayed.

**View** - View represents the visualization of the data that model contains. Whenever the model’s data changes, the model notifies views that depend on it. This layer is independent from application logic. A view must ensure that its appearance reflects the state of the model.

**Controller** - Controller acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate. This layer is independent from application logic. Below figure 4.2 shows the MVC architecture and flow of the layers.

![MVC Architecture](image-url)
4.4.2 Data Layer Implementation

Tables have relevant database relationships such as one-to-one, one-to-many, many-to-many. They help to perform Create, Read, Update, and Delete operations in order to insert data for the database, read data from them as rows, update data, & also delete the unnecessary data.

- Database Table Implementation

Following figure 4.3 shows snippet of database.

```
Table Name: Category

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Datatype</th>
<th>PK</th>
<th>NN</th>
<th>UIQ</th>
<th>E</th>
<th>LN</th>
<th>ZF</th>
<th>AT</th>
<th>G</th>
<th>Default/Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>INT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>VARCHAR(5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>VARCHAR(45)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

*Figure 4.3 Database Implementation*

- Hibernate Configuration

Hibernate requires to know in advance where to find the mapping information that defines Java classes, relate to the database tables. Hibernate also requires a set of configuration settings related to database and other related parameters.

Following figure 4.4 shows snippet of hibernate configuration code.
Java Entity Class

Typically an entity represents a table in a relational database, and each entity instance corresponds to a row in that table. The primary programming artifact of an entity is the entity class, although entities can use helper classes. Java classes whose objects or instances will be stored in database tables are called persistent classes in Hibernate.

The persistent state of an entity is represented either through persistent fields or persistent properties. These fields or properties use object-relational mapping annotations to map the entities and entity relationships to the relational data in the underlying data store. Hibernate works best if these classes follow some simple rules, also known as the Plain Old Java Object (POJO) programming model.

Following figure 4.5 shows Part of an Entity Class Code.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="connection.autocommit">true</property>
    <property name="connection.url">jdbc:mysql://localhost:3306/samanthi</property>
    <property name="connection.password">admin</property>
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
    <property name="show_sql">true</property>
    <property name="generate_statistics"false</property>
    <property name="hibernate.hbm2ddl.auto">update</property>
    <mapping class="entity.Brand"/>
    <mapping class="entity.Category"/>
    <mapping class="entity.Civilstatus"/>
    <mapping class="entity.Customer"/>
    <mapping class="entity.CustomerContact"/>
    <mapping class="entity.CustomerContactItem"/>
    <mapping class="entity.Destination"/>
    <mapping class="entity.Employee"/>
    <mapping class="entity.Gender"/>
    <mapping class="entity.Gim"/>
    <mapping class="entity.Invoice"/>
    <mapping class="entity.InvoicePayment"/>
    <mapping class="entity.Item"/>
    <mapping class="entity.ItemInvoice"/>
    <mapping class="entity.Module"/>
    <mapping class="entity.PaymentMethod"/>
    <mapping class="entity.Policy"/>
    <mapping class="entity.PolicyItem"/>
    <mapping class="entity.Privilege"/>
    <mapping class="entity.RevenueType"/>
    <mapping class="entity.Role"/>
    <mapping class="entity.Supplier"/>
    <mapping class="entity.SupplierPayment"/>
    <mapping class="entity.SupplierContact"/>
    <mapping class="entity.SupplierContactItem"/>
    <mapping class="entity.Title"/>
    <mapping class="entity.User"/>
  </session-factory>
</hibernate-configuration>
```
4.4.3 User Interface Layer Implementation

- FXML User Interface

The Interface facilitates user to do necessary operations and modifications to forms. FXML is a modern java supportive User Interface implementing language which is only contains view elements whose behaviors can be controlled by java controller class.

Instead of writing these FXML codes we uses a tool called Gluon Scene Builder which facilitate drag and dropping of view elements and generate FXML code automatically.

Following figure 4.6 shows FXML Code segment for Category UI.
• **Java Controller classes for FXMLs**

Every FXML file must be controlled by a particular Java Controller Class which implements Java “Initializable” Interface. That Controller class handles every element in the User Interface and keep the binding between the UI and the Model. Following figure 4.7 shows Part of a Controller Class (Variables related to FXML).

```java
public class CustomerUIController implements Initializable {
    @FXML
    private Label lblCode;
    @FXML
    private TextField txtName;
    @FXML
    private TextField txtNIC;
    @FXML
    private ComboBox<String> cmbTitle;
    @FXML
    private Button btnAdd;
    @FXML
    private Button btnUpdate;
    @FXML
    private Button btnClear;
    @FXML
    private Button btnDelete;
    @FXML
    private Button btnSearchClear;
```

Following figure 4.8 shows Part of a Controller Class (Event Handler of an FXML element)
Sales and Inventory Management System

4.4.4 Controller Layer Implementation

Control layer is the link between data layer & the interface layer. Here logical concept is, Pass a user request, validate the user request, determine what the user is trying to do, obtain data from the Model to include in response to user, select the next View the client should see.

The sequencing of calls to the Model, and the sequencing of views and required input from the user defines the application's workflow. Workflow is thus defined in the Controller layer of the application.

- **Styling the User Interface**

User Interfaces which created by FXML can be styled by FX-CSS (FX Cascading Style Sheet).

Following figure 4. 10 shows Part of the CSS code (Style for the Button)

```css
.BUTTON {  
-fx-border-radius: 10.0em;
-fx-border-width: 1px;
-fx-border-color: rgba(135,135,135,0.8);
-fx-background-repeat: no-repeat;
-fx-background-position: 50%;
-fx-background-color: rgba(135,135,135,0.4);
-fx-background-radius: 10.0em;
-fx-text-alignment: center;
-fx-wrap-text: true;
}
```

Figure 4. 11 Part of the CSS code (Style for the Button)

**Hibernate Session**

The Session Factory is the concept that is a single data store and thread safe. Because of this feature, many threads can access this concurrently and the sessions are requested, and also the cache that is immutable of compiled mappings for a specific
database. A Session Factory will be built only at the time of its startup. In order to access it in the application code, it should be wrapped in singleton. This wrapping makes the easy accessibility to it in an application code.

Following figure 4. 12 shows Code segment for creating a hibernate Session.

```java
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;
}

Figure 4. 13 Code segment for creating a hibernate Session
```

- **Data Access Objects (DAO)**

A data access object (DAO) is an object that provides an abstract interface to some type of database or persistence mechanism, providing some specific operations without exposing details of the database. It provides a mapping from application calls to the persistence layer. This isolation separates the concerns of what data accesses the application needs, in terms of domain-specific objects and data types, and how these needs can be satisfied with a specific DBMS.

Following figure 4. 14 shows Part of a DAO Class (Save object to the Database)

```java
public static void update(Object object) {
    Transaction transaction = null;
    try {
        transaction = SESSION.beginTransaction();
        SESSION.merge(object);
        transaction.commit();
    }
    catch (HibernateException e) {
        Notifications.create().title("System Error").text(e.getMessage()).showError();
        if (transaction != null) {
            transaction.rollback();
        }
    }
}

Figure 4. 15 Part of a DAO Class (Save object to the Database)
```
Following figure 4. 16 shows Part of a DAO Class (Select Object from Database).

```java
public static ObservableList select(Class anEntityClass) {
    Criteria criteria = SESSION.createCriteria(anEntityClass);
    return FXCollections.observableArrayList(criteria.list());
}
```

*Figure 4. 17 Part of a DAO Class (Select Object from Database)*

4.4.5 Reused Code and Modules

Several Pre-Developed third party libraries and codes were used while developing the system.

- **Controls FX Notifications**

  Controls FX Notifications were used to show Notifications in the System. Success Notifications, Error Notification, Information Notifications are one of them.

Following figure 4. 18 shows Part of the Notifications Class Code.

```java
public class Notifications {
    private static final NotificationPopupHandler INSTANCE;
    private double startX;
    private double startY;
    private double screenX;
    private double screenY;
    private final Map<String, List<Popup>> popupMap;
    private final double padding = 15.0;
    private ParallelTransition parallelTransition;
    private boolean isShowing;
```

*Figure 4. 19 Part of the Notifications Class Code*

4.4.6 Main Functions

Following figure 4. 20 shows Part of the User Role code.
Following figure 4. 22 shows Part of the Sales return code.

```java
private void clearReturnItemForm() {
    cmbItem.getSelectionModel().clearSelection();
    txtQuantity.clear();
    tblReturnItem.setRow(0);
    supplierReturnItem = new SupplierReturnItem();
    supplierReturnItem.setSupplierReturn(supplierReturn);
    calculateReturnValue();
}
```

Following figure 4. 23 shows Part of the Sales return code

```java
private void calculateReturnValue() {
    returnValue = new BigDecimal(0);
    for (SupplierReturnItem item : tblItem.getItems()) {
        returnValue = returnValue.add(tblItem.getCellData(item));
    }
    tblReturnItem.setText(returnValue.toString());
}
```

Following figure 4. 24 shows Part of the Brand Code.

```java
private void btnDeleteAP() {
    Alert alert = new Alert(AlertType.CONFIRMATION);
    alert.getDialogPane().getStylesheets().add(getClass().getResource("style.css").toExternalForm());
    alert.setCancelButton("Deleting the Brand will also delete its Items too. Are you sure to Delete?";
    alert.showAndWait().ifPresent(buttonType -> {
        if (buttonType.equals(ButtonType.OK)) {
            BrandDao.delete(find);  
            Notifications.create().title("Success").text("Brand Deleted Succeeded").showInformation();
        }
    });
}
```

Following figure 4. 25 shows Part of the Brand Code
Following figure 4. 26 shows Part of the Supplier Code.

```java
private void search()
{
    ObservableList<Supplier> filteredList = FXCollections.observableArrayList();

    suppliersList.forEach(sup -> {
        if (sup.getCode().toLowerCase().contains(txtSearchCode.getText().toLowerCase()))
            if (sup.getName().toLowerCase().contains(txtSearchSupplierName.getText().toLowerCase()))
                filteredList.add(sup);
    });

    FillTable.fillTable(paginationSupplier, tblSupplier, filteredList, 5);
}

@FXML
private void txtNameEK(KeyEvent event) {
    if (txtName.getText().matches(RegularExpression.ITEM_NAME)) {
        nameValidity = true;
        clearError(txtName);
        supplier.setName(TitleCase.toTitleCase(txtName.getText()));
    } else {
        nameValidity = false;
        supplier.setName(null);
        if (txtName.getText().isEmpty()) {
            clearError(txtName);
        } else {
            showError(txtName);
        }
    }
}
```

Figure 4. 27 Part of the Supplier Code

Following figure 4. 28 shows Part of the Purchase Order Code.

```java
@FXML
private void txtQuantityAP(ActionEvent event) {
    if (quantityValidity) {
        purchaseItem.setQuantity(Integer.parseInt(txtQuantity.getText()));
        quantityValidity = true;
    }

    if (isPOItemValid()) {
        PurchaseItem addPurchaseOrderItem = null;
        for (PurchaseItem loopPurchaseOrderItem : purchaseItemList) {
            if (loopPurchaseOrderItem.equals(purchaseItem))
                addPurchaseOrderItem = loopPurchaseOrderItem;
        }

        if (addPurchaseOrderItem == null) purchaseItemList.add(purchaseItem);
        else if (!addPurchaseOrderItem.equals(selectedPOItem))
            purchaseItemList.set(purchaseItemList.indexOf(addPurchaseOrderItem), purchaseItem);
    else {
        Alert alert = new Alert(AlertType.CONFIRMATION);
        alert.getDialogPane().getStylesheets().add(getClass().getResource("Style.css").toExternalForm());
        alert.setContentText("Item is already added. Are you want to merge quantity?");
        alert.showAndWait();
        if (alert.getResults().equals(ButtonType.OK)) {
            purchaseItem.setQuantity(purchaseItem.getQuantity() + addPurchaseOrderItem.getQuantity());
            purchaseItemList.set(purchaseItemList.indexOf(addPurchaseOrderItem), purchaseItem);
        }
    }
```
Following figure 4. 30 shows Part of the Main Window Code.

```java
$FIDL$
private void mainAppSetAP() {
    try {
        Parent root = FXMLLoader.load(getClass().getResource("MyAccountUI.fxml"));
        Scene scene = new Scene(root);
        myAccountStage = new Stage();
        myAccountStage.setScene(scene);
        myAccountStage.initStyle(StageStyle.UNDECORATED);
        myAccountStage.initOwner(mainStage);
        myAccountStage.initModality(Modality.WINDOW_MODAL);

        myAccountStage.show();
    } catch (IOException e) {
        Notifications.create().title("Error").text(e.getMessage()).showError();
    }
}

$FIDL$
private void mainAppGmtAP() throws IOException {
    Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
    alert.getDialogPane().getStylesheets().add(getClass().getResource("Style.css").toExternalForm());
    Alert.setControllerText("Are you sure you want to log out?");
    alert.showAndWait().ifPresent((t) -> {
        try {
            Parent root = FXMLLoader.load(getClass().getResource("LoginUI.fxml"));
            Scene scene = new Scene(root);
            loginStage = new Stage();
            loginStage.setScene(scene);
        } catch (IOException e) {
            Notifications.create().title("Error").text(e.getMessage()).showError();
        }
    });

Figure 4. 31 Part of the Main Window Code

Following figure 4. 32 shows Part of the Invoice Code.

```java
private void setSearchFields() {
    SearchBox.setAutoComplete(ComboBoxPlus(cmbItem, (typeText, objectToCompare) ->
            objectToCompare.getText().toLowerCase().contains(typeText.toLowerCase())),
            SearchBox.setAutoComplete(ComboBoxPlus(cmbCustomer, (typeText, objectToCompare) ->
            objectToCompare.getText().toLowerCase().contains(typeText.toLowerCase()))
        || objectToCompare.getCode().toLowerCase().contains(typeText.toLowerCase()));
    SearchBox.setAutoComplete(ComboBoxPlus(cmbInvoice, (typeText, objectToCompare) ->
            objectToCompare.getCode().toLowerCase().contains(typeText.toLowerCase()))
        || objectToCompare.getDueDate().toString().toLowerCase().contains(typeText.toLowerCase()));

/**
 * Set initial values to the UI components.
 */
private void clearInvoiceForm() {
    itemInvoiceList = FXCollections.observableArrayList();
    cmbItem.getItems().setAll(itemDao.getAvailableItems());
    cmbInvoice.setItems(invoiceDao.getAll());
    cmbCustomer.getItems().setAll(customerDao.getAll());

    String code = invoiceDao.getNextCode();
    // Set default values to text boxes
    lblCode.setText(code);
    cmbCustomer.getSelectionModel().clearSelection();
    lblDate.setText(LocalDate.now().toString);
    lblAvailableQuantity.setText("0");
    lblTime.setText("0");
    lblChange.setText("0");

Figure 4. 33 Part of the Invoice Code
CHAPTER 5 – EVALUATION

5.1 Introduction

Testing is the process of evaluating a system or its component with the intent to find whether it satisfies the specified requirements or not. Further testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

5.2 Software Testing

Software testing is a verification and validation process. Software testing is done during the development. Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to. Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

5.3 Techniques of Software Testing

There are two techniques of software testing.

- **Black Box Testing**
  
  Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

- **White Box Testing**
  
  White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.
5.4 Types of Testing

5.4.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

5.4.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

5.4.3 System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

5.4.4 Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

5.4.5 Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

5.5 System Test Plan

The test plan consists of a series of different tests that will fully exercise the Inventory and Sales Management system for Samanthi Motor Stores. 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 the Inventory and Sales Management system for Samanthi Motor Stores. 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 Inventory and Sales 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 Inventory and Sales 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.

5.6 System Test Cases

A properly planned test case should have the ability to verify the relevant system component functionality. Therefore, to verify all the system functions there should be properly planned test cases for each and every function.

In order to reduce the complexity of the system, system has divided into Module Categories. Test cases were written for each Category. Following Table 5.1 shows Test Results for modules. Please refer Appendix E for Test Cases and Test Results.

<table>
<thead>
<tr>
<th>Module Category</th>
<th>Test Function</th>
<th>Test Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactions</td>
<td>Adding new Transaction</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>Make sure an added Transaction</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transaction cannot be Updated</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>Filter Search Result by privilege</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>Search by relevant fields</td>
<td>Medium</td>
</tr>
<tr>
<td>Search</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select Searched results | High
Adding new Master File | High
Update an Existing Master File | High
Deactivate an Exiting Master File | High
Re-Activate an deactivate Master File | High
Set privileges for different Roles | High
Update privileges for Roles | High
Authenticate user | High

Table 5. 2 Test Results for modules

5.6.1 Test Cases & Test Result for Login Module

Following Table 5. 3 shows Test Cases & Test Result for Login Module.

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Login correct username and password entered in the relevant fields</td>
<td>Login to the system&lt;br&gt;Show Dashboard Window</td>
<td>Pass</td>
</tr>
<tr>
<td>2.</td>
<td>No values entered in the username field</td>
<td>Login button is Disabled</td>
<td>Pass</td>
</tr>
<tr>
<td>3.</td>
<td>No values entered in the password field</td>
<td>Login button is Disabled</td>
<td>Pass</td>
</tr>
<tr>
<td>4.</td>
<td>Incorrect username and password entered</td>
<td>Display error message “Invalid Username/Password”</td>
<td>Pass</td>
</tr>
</tbody>
</table>


### Table 5.4 Test Cases & Test Result for Login Module

#### 5.6.2 Test Cases & Test Result for Brand Module

Following Table 5.5 shows Test Cases & Test Result for Brand Module.

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Auto Generated Brand Id</td>
<td>Increment the last Brand Id by 1</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Brand Name is less than 2 characters or more than 30</td>
<td>Notification will appear</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Brand Description is less than 2 characters or more than 50 characters</td>
<td>Notification will appear</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>Enter all information correctly and click Add</td>
<td>Message box will be shown as “Brand Added Success….”</td>
<td>Pass</td>
</tr>
<tr>
<td>5</td>
<td>Select a row from the table</td>
<td>Selected Brand data will be load in to relevant text fields. Add button will be disabled and Update, delete button will be enabled</td>
<td>Pass</td>
</tr>
<tr>
<td>6</td>
<td>Select a row from the table and change the Brand data and click Update button</td>
<td>Message box will be shown as “Updated Successfully…” and updated data will be shown in the table</td>
<td>Pass</td>
</tr>
<tr>
<td>7</td>
<td>Select a row from the and click Delete button</td>
<td>Message box will be shown as “Deleted Success…” and updated data will be shown in the table</td>
<td>Pass</td>
</tr>
</tbody>
</table>
8. Enter Text value in the “Search by Name” field
Brand table will be change according to typed text values
Pass

9. Click Clear button
Clear text fields data and clear selection of table
Pass

Table 5.6 Test Cases & Test Result for Brand Module

5.6.3 Test Cases & Test Result for Item Module

Following Table 5.7 shows Test Cases & Test Result for Item Module.

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Auto Generated Item Id</td>
<td>Increment the last Item Id by 1</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Item Name is less than 2 characters or more than 20</td>
<td>Notification will appear</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Enter all information correctly and click Add</td>
<td>Message box will be shown as “Item Added Success…”</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>Select a row from the table</td>
<td>Selected Item data will be load in to relevant text fields. Add button will be disabled and Update, delete button will be enabled</td>
<td>Pass</td>
</tr>
<tr>
<td>5</td>
<td>Select a row from the table and change the Item data and click Update button</td>
<td>Message box will be shown as “Updated Success…” and updated data will be shown in the table</td>
<td>Pass</td>
</tr>
</tbody>
</table>
6 Select a row from the and click Delete button Message box will be shown as “Deleted Success…” and updated data will be shown in the table Pass

7 Enter Text value in the “Search by Name” field Item table will be change according to typed text values Pass

8 Click Clear button Clear text fields data and clear selection of table Pass

Table 5.8 Test Cases & Test Result for Item Module

5.6.4 Test Cases & Test Result for Category Module

Following Table 5.9 shows Test Cases & Test Result for Category Module.

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Auto Generated Category Id</td>
<td>Increment the last Category Id by 1</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Category Name is less than 2 characters or more than 40</td>
<td>Notification will appear</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Enter all information correctly and click Add</td>
<td>Message box will be shown as “Category Added Success…”</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>Select a row from the table</td>
<td>Selected Item data will be load in to relevant text fields. Add button will be disabled and Update, delete button will be enabled</td>
<td>Pass</td>
</tr>
</tbody>
</table>
Select a row from the table and change the Item data and click Update button

Message box will be shown as “Updated Success…” and updated data will be shown in the table

Select a row from the and click Delete button

Message box will be shown as “Deleted Success…” and updated data will be shown in the table

Enter Text value in the “Search by Name” field

Item table will be change according to typed text values

Click Clear button

Clear text fields data and clear selection of table

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Select a row from the table and change the Item data and click Update button</td>
<td>Message box will be shown as “Updated Success…” and updated data will be shown in the table</td>
<td>Pass</td>
</tr>
<tr>
<td>6</td>
<td>Select a row from the and click Delete button</td>
<td>Message box will be shown as “Deleted Success…” and updated data will be shown in the table</td>
<td>Pass</td>
</tr>
<tr>
<td>7</td>
<td>Enter Text value in the “Search by Name” field</td>
<td>Item table will be change according to typed text values</td>
<td>Pass</td>
</tr>
<tr>
<td>8</td>
<td>Click Clear button</td>
<td>Clear text fields data and clear selection of table</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Table 5. Test Cases & Test Result for Category Module

5.6.5 Test Cases & Test Result for Purchase Order Module

Following Table 5.11 shows Test Cases & Test Result for Purchase Order Module.

<table>
<thead>
<tr>
<th>No</th>
<th>Test Description</th>
<th>Expected Result</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select Company Name combo</td>
<td>Description about the selected Company will be displayed in the right side box</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>Select Item Category combo</td>
<td>Selected category should be displayed</td>
<td>Pass</td>
</tr>
<tr>
<td>3</td>
<td>Click “Add Purchase Order” without selecting Company, category and item combos</td>
<td>Message box will be shown as “Please select the company”</td>
<td>Pass</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>Message/Action</td>
<td>Result</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>4</td>
<td>Click “Add Purchase Order” without selecting Category and item combos</td>
<td>Message will be shown as “Please select the Category”</td>
<td>Pass</td>
</tr>
<tr>
<td>5</td>
<td>Click “Add Purchase Order” without selecting item combo</td>
<td>Message will be shown as “Please select the item”</td>
<td>Pass</td>
</tr>
<tr>
<td>6</td>
<td>Click “Add Purchase Order” after selecting Company, Category and item combo</td>
<td>Message box will be shown as “Please enter the quantity”</td>
<td>Pass</td>
</tr>
<tr>
<td>7</td>
<td>Click “Add Purchase Order” after selecting Company, Category and item combo</td>
<td>Message box will be shown as “Please Enter the Valid quantity”</td>
<td>Pass</td>
</tr>
<tr>
<td>8</td>
<td>Click “Add Purchase Order” after selecting Company, Category and item combo</td>
<td>New row will be added in to the table and clear the combo boxes and text field</td>
<td>Pass</td>
</tr>
<tr>
<td>9</td>
<td>Enter the incorrect value to quantity</td>
<td>Text field background color changed to red</td>
<td>Pass</td>
</tr>
<tr>
<td>10</td>
<td>Enter the correct value to quantity</td>
<td>Text color changed to white</td>
<td>Pass</td>
</tr>
<tr>
<td>13</td>
<td>Click delete icon in the table row</td>
<td>Remove row from the table</td>
<td>Pass</td>
</tr>
<tr>
<td>14</td>
<td>Click Add button with no values in the table</td>
<td>Message box will be shown as “Table is empty please add items”</td>
<td>Pass</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Click add button with values in the table</td>
<td>Message box will be shown as “New Purchase order added” and add new row to the previous PO# table</td>
<td>Pass</td>
</tr>
<tr>
<td>16</td>
<td>Click clear button</td>
<td>Form will be come to previous state</td>
<td>Pass</td>
</tr>
<tr>
<td>17</td>
<td>Enter the valid PO value to search by PO# text field</td>
<td>Load purchase order details for given PO# id</td>
<td>Pass</td>
</tr>
<tr>
<td>18</td>
<td>Select Search by PO status combo</td>
<td>Table date will be filter according to selected status If there is no data in table data will be disappeared</td>
<td>Pass</td>
</tr>
<tr>
<td>19</td>
<td>Click “Search Clear” Button</td>
<td>Search panel data will be cleared and whole purchase orders will be loaded in to the table</td>
<td>Pass</td>
</tr>
</tbody>
</table>

*Table 5. 12 Test Cases & Test Result for Purchase Order Module*

**5.7 User Evaluation**

Normally user evaluation is done by selecting different users of the system. In this system Manager has been taken as an administrator of the system and other users has taken as normal users with different privileges. User evaluation questionnaire was given to target population and results has summarized.
CHAPTER 6 - CONCLUSION

Samanthi Motor Stores (PVT) Ltd is a well-known spare parts shop in Ja-ela area. They need to enhance their business with new technology. Earlier they faced lot of problems with manual system and heavy paper works. But currently they are carrying out their day to day transactions effectively. Earlier they faced many difficulties of their sales, purchasing process, good received note process and their other process. They have to spend lot of time to handle this process. Currently using sales and inventory management system, they can carry out their daily activities effectively and efficiently. Since all the day to day activities have automated through this system, now they can enhance their business easily. Since including the report generation process of this system, they can analyze their business status and forecasting about future business and they can get good decisions about their business.

During the analysis phase functional and non-functional requirements were identified in each module. After that all functional and non-functional requirements of each and every module were successfully completed in implement phase. End of the project additional functionalities and features were added which client requested. Further client was satisfied about my system finally.

6.1 Future Improvements

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

- Web based Inventory & Transaction Management System will be introduced for ease of connecting shop and the store.
- 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 input attendance and payment records using external reading devices making the system fully automated.
- Android based system to insert daily updates into the system
6.2 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 the conclusion of the project.

It was an opportunity to get extensive knowledge on JavaFX, XML, Hibernate, MVC, MySQL and other languages, tools and technologies. And it helped to test and implement most important theories and technologies learnt throughout the BIT degree programme.
REFERENCES

[1]. Skyware Inventory – Web based Inventory Management System  

[2]. Odoo Inventory - Web based Inventory Management System  


[4]. Microsoft – Database Normalization  


[6]. Wikipedia -The Free Encyclopedia – Net Beans 8.2 IDE  


[8]. Wikipedia -The Free Encyclopedia – MySQL  

[9]. Wikipedia - The Free Encyclopedia – Hibernate (framework)  

[10]. Wikipedia - The Free Encyclopedia – Hibernate (framework)  


[12]. Wikipedia - The Free Encyclopedia - Non Functional Requirement  
APPENDIX

APPENDIX –A SYSTEM DOCUMENTATION

System Manual

This documentation provides guidelines prior to the setup of the Sales and Inventory Management System. This is to assist the software engineers, administrators and managers to install the Sales and Inventory Management System in their PC’s, Servers or Workstations. The system documentation can be referred if there are any changes to be made in the Inventory, Transaction and Production Management System.

Hardware & software configuration requirements.

Hardware requirements

- System configuration should be more than Pentium IV, 2.6 GHz or above.
- processing power 2 GB or above RAM.
- Screen resolution is 1366*768.
- 1 GB hard disk storage

Software requirements

- MySQL Server 5.5
- Java Run Time Environment 1.8

A.1 How to Setup

A.1.1 Install Java Run Time on Client Machine

Some important steps of installing JAVA runtime on client machine are describe below. The steps should be followed in order to install the Java runtime in client machine.

Below figure A.1 shows the step 1 of installing progress of JRE. In the figure it shows the basic information about the java runtime environment.
Below figure A.2 shows the step 2 of installing progress of JRE. In the figure it allows the client to change the installation destination.

Below figure A.3 shows the step 3 of installing progress of JRE. It shows the installation progress to client.
Below figure A.4 shows the step 4 of installing progress of JRE and notifies the client it has installed successfully.

A.1.2 Installing Microsoft SQL Server 5.5

**Step 1:** Download MySQL Community Server 5.5 installation file suitable for the platform. Open installation file for MySQL Community Server and press “Next”
Step 2: Accept license of agreement and choose “Typical” setup type, click “Next” and “Install”. Below figure A.5 shows the step 1 of installing progress of MySQL server. It shows the wizard that helps the installation process.

![Figure A. 5 Installation Progress MySQL Server (Step1)](image)

Below figure A.6 shows the step 2 of installing progress of MySQL server. It shows the license agreement of MySQL server installation.

![Figure A. 6 Installation Progress MySQL Server (Step2) – License Agreement](image)
Below figure A.7 shows the step 3 of installing progress of MySQL server. It directs the client to choose the setup type which is suitable for the machine.

![MySQL Server 5.5 Setup (Step 3)](image)

*Figure A. 7 Installation Progress MySQL Server (Step 3)*

Below figure A.8 shows the step 4 of installing progress of MySQL server.

![MySQL Server 5.5 Setup (Step 4)](image)

*Figure A. 8 Installation Progress MySQL Server (Step 4)*
**Step 3:** After installation process is completed, check “Launch the MySQL Instance Configuration Wizard” and click “Finish”

**Step 4:** Choose “Standard Configuration” and click “Next”

**Step 5:** Check “Install as Windows Service”, select service name “MySQL”. Check “Launch the MySQL Server automatically” check to “Include Bin Directory in Windows PATH” and click “Next”

**Step 6:** Set a long password for the “root” user, check “Enable root access from remote machines”. Click “Next” and then “Execute”

**Step 7:** After configuration process is completed click “Finish”
Below figure A.9 shows the step 5 of installing progress of MySQL server. It notifies the client to choose the setup type which is suitable for the machine.

![Figure A.9 Installation Progress MySQL Server (Step 5)](image)

Below figure A.10 shows the step 6 of installing progress of MySQL server. It notifies the client to choose the configuration type which is suitable for the machine.
Below figure A.11 shows the step 7 of installing progress of MySQL server. It notifies the client to set the windows option which is suitable for the machine.
Below figure A.12 shows the step 8 of installing progress of MySQL server. It allows the client to set the security options which is suitable for the machine.

![MySQL Server Instance Configuration Wizard](image1)

*Figure A. 12 Installation Progress MySQL Server (Step 8)*

Below figure A.13 shows the step 9 of installing progress of MySQL server. It shows MySQL server has installed to client’s machine successfully.

![MySQL Server Instance Configuration Wizard](image2)

*Figure A. 13 Installation Progress MySQL Server (Step 9)*
A.1.3 Installing Microsoft SQL Query Browser

Below figure A.14 shows the step 1 of installing progress of MySQL query browser. It shows the wizard which helps to client to through the installation progress.

![Figure A. 14 Installation Progress MySQL Query Browser](image1.png)

Below figure A.15 shows the step 2 of installing progress of MySQL query browser. It shows the license agreement of MySQL query browser.

![Figure A. 15 Installation Progress MySQL Query Browser (Step 2)](image2.png)
Below figure A.16 shows the step 3 of installing progress of MySQL query browser. It facilitates to client to change the destination folder.

Figure A. 16 Installation Progress MySQL Query Browser (Step 3)

Below figure A.17 shows the step 4 of installing progress of MySQL query browser. Client can enter a password and a user name to connect to the MySQL query browser.

Figure A. 17 Installation Progress MySQL Query Browser (Step 4)
Below figure A.18 shows the step 5 of installing progress of MySQL query browser. It directs the client to the MySQL administrator.

![MySQL Administrator 1.2.17](image)

*Figure A. 18 Installation Progress MySQL Query Browser (Step 5)*

**A.1.4 Installing Sales and Inventory Management System**

After setting up the database, select from device Click Add & Browse the Sales and Inventory Management System file from CD Run the setup.exe file located in CD.
APPENDIX – B

DESIGN DOCUMENTATION

B.1 Activity Diagram

Below figure B.1 shows the activity diagram for user login to the system.

![Activity Diagram for log into the System]

Figure B.1 Activity Diagram for log into the System

B.2 Sequence Diagram

A sequence diagram is a kind of interaction diagram. We can use a sequence diagram to illustrate the dynamic view of a system. 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.

Following Figure B.2 shows Sequence Diagram for the Invoicing.
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 are following,

- Owner
- General Manager
- System Administrator
- Cashier
- Receptionist
- Backoffice User

Following Figure B. 4 shows Use Case Diagram for Owner.

![Figure B. 5 Use Case Diagram for Owner](image)

Following figure B.4 shows Use Case Diagram for General Manager.

![Figure B.4 Use Case Diagram for General Manager](image)

Following figure B.5 shows Use Case Diagram for System Administrator.
Sales and Inventory Management System

Following figure B.5 shows Use Case Diagram for System Administrator.

Following figure B.6 shows Use Case Diagram for Cashier.

Following figure B.7 shows Use Case Diagram for Receptionist.

Following figure B.8 shows Use Case Diagram for Backoffice User.
Use Case Description for login into System is given in Table B.1

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Login to the System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors</td>
<td>All Users</td>
</tr>
<tr>
<td>Overview</td>
<td>Authorized users who have the privilege to access the system</td>
</tr>
</tbody>
</table>
| Preconditions  | User must have an user account  
User must enter valid username and password |
| Flow Events    | 4. The user enters user name and password  
5. If entry is invalid, the user is redirected to the login form with an error message  
6. If entry is valid the system redirects the user to Main Window |
| Post Conditions| User will be redirected to Main Window or back to the Login form it depends on username and password |

Table B.1 Use Case Description for login into System

Use Case Description for inserting a new item is given in Table B.2.

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Inserting a new item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors</td>
<td>Back office User</td>
</tr>
</tbody>
</table>
Overview
Insert a New Item.

Preconditions
A privileged user must login to the system

Flow Events
6. Select item type
7. Select category
8. Select supplier
9. Give required information
10. Press save button

Post Conditions
New item added to the system

Use Case Description for inserting a new item

---

Use Case Description for add a new purchase order is given in Table B. 3

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Add new purchase order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actors</td>
<td>General Manager</td>
</tr>
</tbody>
</table>

Overview
Purchase order details are added to the system

Preconditions
A privileged user must login to the system
Relevant supplier, relevant quantities are required by the system.

Flow Events
4. Select the supplier
5. Select the needed items
6. Save Purchase order

Post Conditions
New purchase order details add into the system

---

B.4 Database Design
Structure of Category Table in database is given in Figure B.9
Sales and Inventory Management System

Figure B.9 Category Table in database

Structure of Brand Table in database is given in Figure B.10

Figure B.10 Brand Table in database

Structure of Item Table in database is given in Figure B.11

Figure B.11 Item Table in database

Structure of Conditions Table in database is given in Figure B.12
Figure B.12  Conditions Table in database

Structure of Return Type Table in database is given in Figure B.13

Figure B.13  Return Type Table in database

Structure of Invoice Item Table in database is given in Figure B.14

Figure B.14 Invoice Item Table in database

Structure of Invoice Table in database is given in Figure B.15
Structure of Customer Table in database is given in Figure B.16

Structure of Title Table in database is given in Figure B.17

Structure of Invoice Pay Table in database is given in Figure B.18
Structure of Customer Return Table in database is given in Figure B.19

Structure of Customer Return Item Table in database is given in Figure B.20

Structure of Purchase Order Item Table in database is given in Figure B.21
Structure of Purchase Order Table in database is given in Figure B.22

Structure of GRN Item Table in database is given in Figure B.23

Structure of GRN Table in database is given in Figure B.24
Figure B.24 GRN Item Table in database

Structure of Supplier Table in database is given in Figure B.25

Figure B.25 Supplier Table in database

Structure of Supplier Pay Table in database is given in Figure B.26

Figure B.26 Supplier Pay Table in database

Structure of Supplier Return Table in database is given in Figure B.27
Structure of Supplier Return Item Table in database is given in Figure B.28

Structure of Employee Table in database is given in Figure B.29
Structure of Employee Status Table in database is given in Figure B.30

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Datatype</th>
<th>PK</th>
<th>NN</th>
<th>UQ</th>
<th>B</th>
<th>UN</th>
<th>ZF</th>
<th>AI</th>
<th>G</th>
<th>Default/Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>INT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>VARCHAR(45)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure B.30 Employee Status Table in database*

Structure of Employee Designation Table in database is given in Figure B.31

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Datatype</th>
<th>PK</th>
<th>NN</th>
<th>UQ</th>
<th>B</th>
<th>UN</th>
<th>ZF</th>
<th>AI</th>
<th>G</th>
<th>Default/Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>INT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>VARCHAR(45)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure B.31 Employee Designation Table in database*

Structure of Gender Table in database is given in Figure B.32

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Datatype</th>
<th>PK</th>
<th>NN</th>
<th>UQ</th>
<th>B</th>
<th>UN</th>
<th>ZF</th>
<th>AI</th>
<th>G</th>
<th>Default/Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>INT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>VARCHAR(45)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure B.32 Gender Table in database*

Structure of Employee Civil Status Table in database is given in Figure B.33
Sales and Inventory Management System

Figure B.33 Employee Civil Status

Structure of User Table in database is given in Figure B.34

Figure B.34 User Table in database

Structure of Role Table in database is given in Figure B.35

Figure B.35 Role Table in database

Structure of User Role Table in database is given in Figure B.36
Structure of User Role Table in database is given in Figure B.36

Structure of Privilege Table in database is given in Figure B.37

Structure of Module Table in database is given in Figure B.38

Structure of Payment Method Table in database is given in Figure B.39
APPENDIX – C USER DOCUMENTATION

C.1 User Login Form

Following Figure C.1 shows user login form which allows users to log into the system. All levels of users can log into the system in one form. When user try to log into the system check whether this user is valid user or not otherwise system display error message.

![User Login Form](image)

*Figure C.1 User Login Form*

C.2 Main Window

All the functions of each and every module can be accessible by users through this Main Window using the Navigation bar on the top.

All the modules are categorized for the simplicity.

- Master Files
  - Item Module
  - Category Module
  - Brand Module
  - Supplier Module
  - Customer Module
- Transactions
• Purchase Order Note Module
• Goods Received Note Module
• Sales Return Note Module
• Purchase Return Note Module
• Invoice Module
• Payment Module

• System Administration
  • User Module
  • Role Module
  • Employee Module

• Reports
  • Management Reports
  • Transaction Reports
  • Master Files Reports

Following figure C.2 shows Main Window of the system.

![Figure C.2 Main Window](image)

C.3 Item Module

Creation and Updates if Items are done using this module. Also Stock details can be edit using this window.

Following figure C.3 shows Item Module of the system.
C.4 Goods Received Note Module

Following figure C. 4 shows Goods Received Note Module of the system.
APPENDIX – D MANAGEMENT REPORTS

System allows users such as managers and administrators to generate various types of reports in order to use make decisions. After generating them, management can use them to analyze & identify the trends, patterns of the business & predict future business situations.

Master Files Reports

- All Employee Report of the system is given in Figure D.1

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Civil Status</th>
<th>Telephone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ashani Costa</td>
<td>Male</td>
<td>Married</td>
<td>0775206363</td>
</tr>
<tr>
<td>Thamka Dilshan</td>
<td>Male</td>
<td>Married</td>
<td>0775206363</td>
</tr>
<tr>
<td>Sidutha Priyanjitha</td>
<td>Male</td>
<td>Married</td>
<td>0784562893</td>
</tr>
</tbody>
</table>

Figure D.1 All Employee Report of the system
**Transaction Reports**

- Purchase Order Report of the system is given in Figure D.2

---

**Purchase Order Note**

Tuesday 31 October 2017

Mitsubishi Motors
Japan
0332239708

<table>
<thead>
<tr>
<th>Item</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mitsubishi Tyre 16 Inches</td>
<td>20</td>
</tr>
<tr>
<td>Mitsubishi Tyre 17 Inches</td>
<td>18</td>
</tr>
</tbody>
</table>

---

*Figure D. 2 Purchase Order Report of the system*
APPENDIX – E TEST RESULTS

E.1 Login Form

Following Table E.1 shows Test Results for Login Form.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Expected Output</th>
<th>Actual Output</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Valid</td>
<td>Login to the System</td>
<td>Logged into the System</td>
<td>Pass</td>
</tr>
<tr>
<td>Enter Valid</td>
<td>Show Error Message – “Password is Wrong”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User Name and Valid Password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enter Invalid Username but Valid Password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enter Invalid Username and Valid / Invalid Password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enter Valid</td>
<td>Show Error Message – “Username is Wrong”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User Name and Valid Password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enter Invalid Username and Valid / Invalid Password</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Showed Error Message -
Password is wrong
System will be blocked after 2 more attempts

Showed Error Message -
Username is incorrect

Pass
Pass
Pass
Enter valid Userna me and Wrong Password 3 times. 
Show Error Message – “System is blocked. Please contact your System Administrator.”

Table E.2 Test Results for Login Form

E.2 Item Module

Following Table E. 3 shows Test Results for Item Module.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Expected Output</th>
<th>Actual Output</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Press “Add” Button Without entering any Details</td>
<td>Show Error Notification</td>
<td>Showed Error Notification</td>
<td>Pass</td>
</tr>
<tr>
<td>Press “Add” button with</td>
<td>Show Error Notification</td>
<td>Pass</td>
<td></td>
</tr>
</tbody>
</table>
missing mandatory data showed invalidation errors in mandatory fields

Enter every valid detail and press on “Add” button

Showed Succeed Notification

Table E 4 Test Results for Item Module

E.3 Overall Test Results

Using the standard set of questions provided in papers, the feedbacks were collected. Questioning about the criteria mentioned in the form and had filled the paper according to their responses. Those Staff members were:

- Cashier
- System Administrator
- Backoffice User
- General Manager

Overall Feedback of the Staff members were analyzed and Drawn into a Pie Chart.
Following figure E. 1 shows Overall Test Results from the Staff Members.

![Overall Test Results from the Staff Members](image)

*Figure E. 2 Overall Test Results from the Staff Members*
E.4 User Acceptance Test Results

Following figure E. 3 shows User Acceptance Test Results of the System Administrator.

![User Evaluation Questionnaire Image]

*Figure E. 4 User Acceptance Test Results of the System Administrator*
APPENDIX – F CODE LISTING

Major code fragments for anyone who is interested in referring the functionality of the system are contained in this document.

F.1 Main Class

Following figure F. 1 shows Code of Main Class.

```java
// To change this template, choose Tools | Templates | Code | Templates
package msi;

import java.io.File;
import java.util.List;
import java.util.TreeSet;

public class MainClass {
    public static void main(String[] args) {
        // Code of Main Class
    }
}
```

Figure F. 2 Code of Main Class
F.2 Birthday and Gender from NIC

Following figure F. 3 shows Code of NIC Class.

```java
package util;

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

public class NIC {
    private static int year;
    private static int month;

    public static LocalDate getBirthDay(String NIC) throws Exception {
        try {
            getYearAndDay(NIC);
            if (day > 31) {
                day = day - 30;
            }
            month = Month.DECEMBER;
            day = 31;
            else if (day > 30) {
                month = Month.NOVEMBER;
                day = 30;
            } else if (day > 27) {
                month = Month.OCTOBER;
                day = 27;
            } else if (day > 24) {
                month = Month.SEPTEMBER;
                day = 24;
            } else if (day > 21) {
                month = Month.AUGUST;
                day = 21;
            } else if (day > 15) {
                month = Month.JULY;
                day = 15;
            } else if (day > 10) {
                month = Month.JUNE;
                day = 10;
            } else if (day > 9) {
                month = Month.MAY;
                day = 9;
            } else if (day > 32) {
                month = Month.APRIL;
                day = 32;
            } else if (day > 2) {
                month = Month.MARCH;
                day = 2;
            } else if (day > 1) {
                month = Month.FEBRUARY;
                day = 1;
            }
            return LocalDate.of(year, month, day);
        } catch (Exception ex) {
            throw new Exception(ex.getMessage());
        }
    }

    public static String getGender(String NIC) throws Exception {
        try {
            getYearAndDay(NIC);
            if (day > 3) {
                return "Male";
            } else {
                return "Female";
            }
        } catch (Exception ex) {
            throw new Exception(ex.getMessage());
        }
    }

    private static void getYearAndDay(String NIC) throws Exception {
        if (NIC.matches("([0-9][0-9][0-9][0-9])([0-9][0-9][0-9][0-9])((126)[0-9][0-9][0-9][0-9])((9-9)[0-9][0-9][0-9][0-9])")
                year = Integer.parseInt(NIC.substring(0, 4));
                day = Integer.parseInt(NIC.substring(4, 9));
            } else if (NIC.matches("([0-9][0-9][0-9][0-9])([0-9][0-9][0-9][0-9])((126)[0-9][0-9][0-9][0-9])((9-9)[0-9][0-9][0-9][0-9])")
                year = Integer.parseInt(NIC.substring(0, 4));
                day = Integer.parseInt(NIC.substring(4, 9));
            } else {
                throw new Exception("NIC Doesn't Match any valid formats.");
            }
        }
    }
}
```

Figure F. 4 Code of NIC Class
F.3 Animate Class

Following figure F. 5 shows Code of Animate Class.

Figure F. 6 Code of Animate Class
F.4 Security

Following figure F.7 shows Code of Security Class.

```java
public class Security {
    public String getHash(String password)
    {
        StringBuilder stringBuilder = new StringBuilder(); 
        try {
            MessageDigest messageDigest = MessageDigest.getInstance("SHA-1");
            byte[] array = messageDigest.digest(password.getBytes("UTF-8"));
            stringBuilder.append(Integer.toHexString(b & 0xff)+0x100,16).substring(1);
        }
        catch(NoSuchAlgorithmException | UnsupportedEncodingException ex) {
            Logger.getLogger(Security.class.getName()).log(Level.SEVERE, null, ex);
        }
        return stringBuilder.toString();
    }
    public static String generateSalt(){
        return new BigInteger(130, secureRandom).toString(32);
    }
}
```

Figure F.8 Code of Security Class

F.5 Database Connection

Following figure F.9 shows Code of Database Connection.

```java
package util;

import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class DatabaseConnection {
    public static boolean isAvailable(String dbUrl) {
        try {
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/"+dbUrl, "user", "admin");
            conn.close();
        }
        catch(SQLException ex) {
            return false;
        }
        return true;
    }
    public static void executeScript(String user, String password, String sqlScript) {
        String command = sqlScript;
        try {
            Process process = Runtime.getRuntime().exec(command);
            int status = process.waitFor();
            return status == 0;
        }
        catch (IOException | InterruptedException ex) {
            return false;
        }
    }
}
```

Figure F.10 Code of Database Connection
F.6 CommonDao class

Following figure F. 11 shows Code of CommonDao Class.

```java
package dao;

import javax.persistence.EntityManager;
import java.util.List;

public class CommonDao
{
    public static void insert(Object object)
    {
        Transaction transaction = null;
        try
        {
            // Create a new Session object from the SessionFactory
            Session session = sessionFactory.openSession();
            // Begin a new transaction
            transaction = session.beginTransaction();
            // Save the object
            session.save(object);
            // Commit the transaction
            transaction.commit();
        }
        catch (Exception e)
        {
            if (transaction != null)
                transaction.rollback();
        } finally
        {
            closeSession(sessionFactory);
        }
    }

    public static void update(Object object)
    {
        Transaction transaction = null;
        try
        {
            // Create a new Session object from the SessionFactory
            Session session = sessionFactory.openSession();
            // Begin a new transaction
            transaction = session.beginTransaction();
            // Update the object
            session.update(object);
            // Commit the transaction
            transaction.commit();
        }
        catch (Exception e)
        {
            if (transaction != null)
                transaction.rollback();
        } finally
        {
            closeSession(sessionFactory);
        }
    }

    public static void delete(Object object)
    {
        Transaction transaction = null;
        try
        {
            // Create a new Session object from the SessionFactory
            Session session = sessionFactory.openSession();
            // Begin a new transaction
            transaction = session.beginTransaction();
            // Delete the object
            session.delete(object);
            // Commit the transaction
            transaction.commit();
        }
        catch (Exception e)
        {
            if (transaction != null)
                transaction.rollback();
        } finally
        {
            closeSession(sessionFactory);
        }
    }

    // Other methods...
}
```

Figure F. 12 Code of CommonDao Class
F.7 Brand UI Controller

Following figure F. 13 shows Code of Brand UI Controller.

Figure F. 14 Code of Brand UI Controller
APPENDIX – G CLIENT CERTIFICATION

Following figure G.1 shows Client Certification.

![Letter of Certification Image](image_url)

*Figure G.1 Client Certification*
<table>
<thead>
<tr>
<th>INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>B</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>C</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>D</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>E</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>F</strong></td>
</tr>
<tr>
<td><strong>G</strong></td>
</tr>
<tr>
<td><strong>H</strong></td>
</tr>
<tr>
<td><strong>I</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>J</strong></td>
</tr>
<tr>
<td><strong>M</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>N</strong></td>
</tr>
<tr>
<td><strong>O</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>P</strong></td>
</tr>
<tr>
<td><strong>Q</strong></td>
</tr>
<tr>
<td>R</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>S</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>U</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>V</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>W</td>
</tr>
</tbody>
</table>
GLOSSARY

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

**UML** - Unified Modeling Language. It is a general-purpose modeling language which is used in object-oriented development.

**Graphical User Interface** - this allows users to interact with electronic devices with graphics rather than text commands.

**PHP** - Hyper-text Pre-processor is a server-side programming language.

**C/C++** - a programming language

**HTML** - Hyper Text Markup Language. This is a markup language for creating web pages.

**JVM** - Java Virtual Machine is a virtual machine that uses to execute Java byte code.

**JavaFX** - is a software for designing rich internet applications that can run on wide variety of devices

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

**RAD** - Rapid Application development (RAD) is an incremental software development process model.