

# **ITE 1942 – ICT PROJECT**

## **PROJECT REPORT**

### **Level 01**

#### **Point of sales Sri Lanka**

**Submitted by:**

H.M.K.M.RATHNAYAKA

E2046105

Bachelor of Information Technology (External Degree)

Faculty of Information Technology

University of Moratuwa

**Faculty of information technology**  
**University of Moratuwa**

**Declaration**

I hereby declare that the work I reported in this document work on Point of sales system Sri Lanka submitted to university of Moratuwa is my own work. This work done as the practical requirement for the degree of Batcheler's of information technology under the super vision of Mrs. Madushi. The material contained in the report has not been submitted to any University or Institution for any degree program.

H.M.K.M.RATHNAYAKA

E2046105

Bachelor of Information Technology (External Degree)

Faculty of Information Technology

University of Moratuwa

20<sup>th</sup> of May 2022

## **Abstract**

This project is aimed at developing an information system to solve day to day problems that occur in small retail business environment. This system can be using a terminal interface in any organization. Moreover, process billing information more efficiently, issue an acceptable bill as record of proof to the customer can be done with more effective way. This project is categorized several aspects for the sales and inventory management system. By that in this system we are solving different problem affecting to direct sales management and purchase management.

## List of figures

Figure 1 small enterprise which use traditional information system .....	9
Figure 2 Proposed framework of IT adoption influencing factors in SME context. ....	11
Figure 33-Tier Architecture .....	13
Figure 4 waterfall method .....	15
Figure 5 Process flow diagram .....	19
Figure 6 Flow chart.....	20
Figure 7 Context diagram.....	22
Figure 8 DFD 0 Diagram .....	23
Figure 9 Model-view-controller pattern .....	24
Figure 10 Data inheritance.....	25
Figure 11 Login page .....	32
Figure 12 Dashboard .....	32
Figure 13 Help Desk .....	33
Figure 14 Look up.....	33
Figure 15 Sub total.....	34
Figure 16 Tender view .....	34
Figure 17 Calculator .....	35
Figure 18 Testing process .....	42

## **List of abbreviations**

IMS	Inventory Management System
POS	Point of Sales
TMS	Transaction Management System
SME	Small and Medium Enterprises
DBMS	Database Management System
DFD	Data Flow Diagram
DML	Data Manipulation Language
DDL	Data Definition Language
DCL	Data Control Language
SQL	Structured Query Language
RDMS	Relational Database Management System
MVC	Model-view-controller pattern

## Table of content

1. Introduction.....	9
1.1. Background of the study .....	9
1.2. Significance of the study.....	9
1.3. Aims and objectives.....	10
1.4. the statement of the problem.....	10
1.5. Research methodology.....	11
1.6. Results .....	11
1.6.1. Probable cause of the situation.....	11
1.6.2. Possible solution.....	12
1.7. Conclusion .....	12
2. Background Knowledge .....	13
2.1. Database Design .....	13
2.1.1. Client tier .....	13
2.1.2. Business tier.....	13
2.1.3. Data tier.....	13
2.2. Database theory .....	14
2.2.1. Primary key.....	14
2.2.2. Foreign key .....	14
3. System analysis.....	15
3.1. Background research methodologies.....	15
3.2. Functional requirements .....	16
3.3. Nonfunctional requirements.....	16
3.4. Feasibility analysis .....	17
3.4.1. Economic feasibility.....	17
3.4.2. Technical feasibility .....	17
3.4.3. Operational feasibility .....	17
3.4.4. Time feasibility.....	18
3.5. Conclusion .....	18
4. System design.....	19
4.1. Process flow diagram .....	19
4.2. Flow chart.....	20

4.3.	Pseudo code .....	21
4.4.	Context diagram .....	22
4.5.	DFD 0 Diagram.....	23
4.6.	Summarization.....	23
5.	Tools and technology that used .....	24
5.1.	Development tools .....	24
5.2.	Architectural designing.....	24
6.	System implementation .....	25
6.1.	Application code structure .....	25
6.2.	Controller package .....	26
6.2.1.	Login page.....	26
6.2.2.	Dashboard controller.....	27
6.2.3.	Look up controller .....	29
6.2.4.	Subtotal controller.....	30
6.2.5.	Tender Controller .....	31
6.3.	View package.....	32
6.3.1.	Login page FXML.....	32
6.3.2.	Dashboard FXML.....	32
6.3.3.	Helpdesk FXML .....	33
6.3.4.	Lookup Controller .....	33
6.3.5.	Subtotal view .....	34
6.3.6.	Tender view .....	34
6.3.7.	Calulator view .....	35
7.	Appendix.....	36
7.1.	Add item to the bill.....	36
7.2.	Load table .....	37
7.3.	Search product .....	38
7.4.	Update bill table .....	39
7.5.	Update sold product table.....	40
7.6.	Reset dashboard.....	41
8.	Debugging and testing.....	42
8.1.	Unit testing .....	42
8.2.	Integration testing.....	43

8.3.	System Testing.....	43
8.4.	Special note .....	43
9.	Conclusion and lessons learnt .....	44
9.1.	Project Limitations.....	44
9.2.	Lessons learned .....	44
9.3.	Future enhancements .....	45
9.4.	To summarize .....	45
10.	References .....	46



## 1. Introduction

Small enterprises play a vital role in economic around the globe. Specially in a developing countries like Sri Lanka. The government of Sri Lanka recognizes SMEs as the backbone of the economy since they account for more than 75% of the total number of enterprises, provides 45% of the employment and contributes to 52% to the GDP [1].

But despite the grate contribution to the economic small business suffer from grate failure due to several reasons. Such as limited financial support, market condition, unbeatable competition, unacceptable pricing, lack of entrepreneur skills and so on. Using information technology to make the best decisions at the right time using processed data will reduce small business failure by a considerable amount.

### 1.1. Background of the study

Most of start-up business are innovative though, most of them are going to fail due to specific reasons. However, the fundamental question is why some of the small enterprises fail while others are thriving under similar economic structure.

Since the rate of business failure among SMEs in Sri Lanka is 45% [2] most of them suffer from similar reasons. At present there is a serious economic crisis is occurring in Sri Lanka. So, it makes even harder to stop the inevitable which is business failure. Unless a innovative business sees some opportunity in this crisis and seize it in a split second.

### 1.2. Significance of the study

This research intends to find the solution to the problem of small retail business failure by keeping a smart record of transactions which made the enterprise. also keep an updated information on inventory to purchase right amount of good at the precise time. Even though there is also requirement for produce several types of reports to help take more effective dictions as entrepreneurs, unfortunately in this project scope it will not be covered. This project calls for a complete redesign of technology and tools from traditional method.



*Figure 1 small enterprise which use traditional information system*

### **1.3.Aims and objectives**

Primary goals and secondary goals were recognized during this small background research. As for primary goals to fulfill the requirement for achieving the bachelor's degree of information technology, study and understand the fundamentals of project development and small business requires a tool to analyze information before taking important decisions.

Considering secondary goals,

- To develop an application that deals with the day-to-day requirement of retail business enterprises.
- Develop point of selling system for make transactions more efficient.
- To handle the inventory details like sales details and stock balance details.
- To provide competitive advantage to the organization.
- Evaluate developed system.

### **1.4.the statement of the problem**

SMEs have gained recognition as a major player of employment, income generation, poverty alleviation and regional development [3]. Furthermore, they play vital roles in developing entrepreneurial skills and innovation and promoting economic growth of the country and wealth creation.

Hence the problems of small business failure must be addressed. Moreover, major role of this research is to design a solution which will support small enterprises to continue their operations with effective and efficient manner. After analyzing many existing TMS I have now the obvious vision of the project to be developed. Before I started to build the application, I had many challenges. So, I defined my problem statement as:

- To make desktop-based application software for small business environment.
- Ability to make billing procedure more efficient than manual system.
- Keep track on the inventory in more reliable manner.
- Issue a printed to the customer as record of proof.
- Update sales, inventory information in real time.

Hopefully, in the end we will be able to reduce small business failure by considerable amount.

## 1.5. Research methodology

Both qualitative and quantitative data was needed in order to collect information about small business failure in Sri Lanka. Research papers [4] which can be found on the internet were very helpful to build these theories. Also, electronic papers [2] and online web journals [5] gave a quite help with critical information regarding this report.

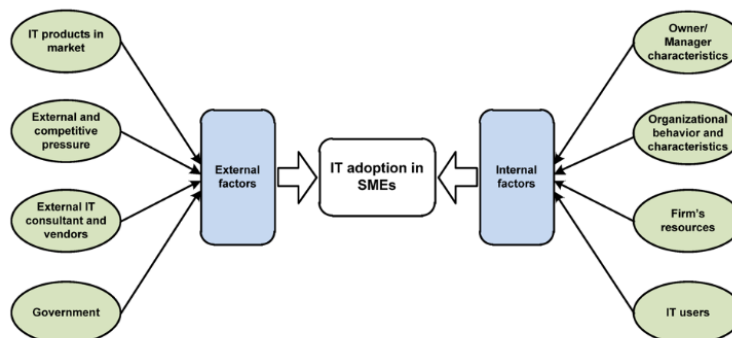
Most of the quantitative data were gathered via secondary data sources and they were essential to prove research objectives. They were specifically gathered by their analysis for the same objective, made them significantly valuable methods. Even though they have several limitations and weakness they were outweighed the strengths. Furthermore, chosen method for this research were the best possible approach for the research objective. Even though some citations were hard to find they had been dealt with.

## 1.6. Results

### 1.6.1. Probable cause of the situation

The failure of small enterprises is not because of only one cause but to several causes. Such as limited financial support, market condition, unbeatable competition, unacceptable pricing, lack of entrepreneur skills, lack of co-operation and networking, revenue does not sufficiently exceed costs, decline performance, and many more.

Which will reduce customer experience significantly. According to Gartner statistics, 80% of a company's revenue comes from 20% of its customers. Loyal customers are the success stories of your business. Involve them in your business strategies, marketing campaign planning, and new product development. Share their case studies, consider their points of view, absorb their feedback (both good and bad), and make them feel important. [5] So there is immediate need of effective information system to solve the given problem at hand.



*Figure 2 Proposed framework of IT adoption influencing factors in SME context.*

### **1.6.2. Possible solution**

Using an efficient information system will allow managers to deal with the external and internal environments more productively. Even though small enterprises invest comparatively little in technology. And when they do, they often acquire equipment, machinery, and software that is inappropriate to their business environment [6]. Also, because of the contingency approach each small business environment needs a different solution I have to be flexible to those changes made around those environments.

### **1.7. Conclusion**

The entrepreneurs need to be conversant with basic business management rules as well as need to have access to basic resources [7]. Using a software-based solution companies should be able to concentrate on cost reduction, quality improvements, service improvements, and product innovation and breakdown organizational barriers between departments. Information technology will allow small business to grow and become a large-scale business in a near future. Decreasing the failure rate of small business will strengthen the backbone of Sri Lankan economy.

## 2. Background Knowledge

This desktop application consists of 3-tier of java FX. Three-layer architecture is dividing the project into three layers that are User interface layer, business layer and data(database) layer where we separate UI, logic, and data in three divisions. [8]

### 2.1.Database Design

Before constructing an information system, a I had to understand basic data design concepts, including data structures and the characteristics of file-oriented and database management systems.

#### 2.1.1. Client tier

This is the top layer of architecture. The topmost level of application is the UI (user interface). It is related to the user interface that is what the user sees including several types of java FX controllers. For example, if a user clicks on the Tender button user wants to print current customer's bill.

#### 2.1.2. Business tier

This layer includes java classes and logical calculations, and operations are performed under this layer. It processes the command, makes logical decisions, and perform calculations. It also acts as a middleware between two surrounded layers that is client and data layer in the database architecture.

#### 2.1.3. Data tier

This layer establishes a connection between database and performs functions on the database. Furthermore, this layer is used to connect the business layer to the database or data source. It contains a lot of useful methods which are used to perform operations on database like insert, delete, update, truncate, etc.

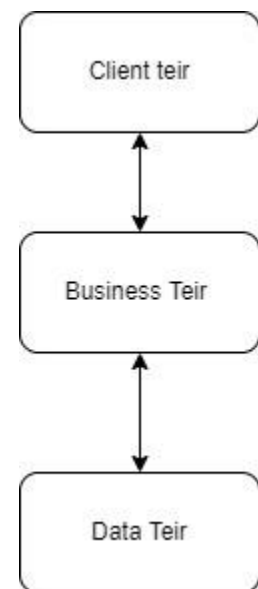


Figure 33-Tier Architecture

## **2.2.Database theory**

A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. In this transaction management software has the relational database model. Moreover, according to Wikipedia relational database can be considered as a collection of data items with pre-defined relationships between them [9].

Using this model, I have to accomplish several objectives,

- Reduce duplication of data.
- Minimize dependency on application programs.
- Maximize data security.
- Easy access to data.

### **2.2.1. Primary key**

A primary key is the column or columns that contain values that uniquely identify each row in a table [10]. It has two main features. They are containing unique value for each row and cannot contain a null value.

### **2.2.2. Foreign key**

A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables [11]. Unlike primary key it can be duplicated and allowed to input null as an acceptable value. Foreign key creates a link between tables when the primary key column or columns reference to another column in another table.

### 3. System analysis

#### 3.1. Background research methodologies

Since planning is the most important thing in a system development procedure, I have implemented considerable time into this step. Main aim in this process of studying is to identify its goal and purposes and create systems and procedures that will efficiently achieve them.

I used different technique to collect information that can give a clear and overall understanding of the proposed information system. Main technique that I used is **interviews**. The store next my home and Veyangoda food city near the station had helped me willingly. They have helped me to realize complex structures, achieving inter compatibility and unity of purpose of sub-systems, main functions of a POS information system. Hence interviews proved to be a most useful fact-finding technique. Also, I used **questionnaires** and **observations** as optional.

Considering the project scope and this is a nonteam assignment I mainly focused use **waterfall method** (referred as linear-sequential model) as the software development process. The Waterfall model is known as the most historical SDLC approach for software development. This model is straightforward to understand and use. The underline concept of waterfall model is that each phase must be completed prior to the beginning of the next phase.

Finally, I focused on several factors while software analyzing,

- Cost and affordability.
- Effective flow of stock transfer and management.
- Difficulty in managing inventory in a reliable manner.
- Establish the feasibility from different angles.
- Establish the system boundaries which would define the scope and the coverage of the system.

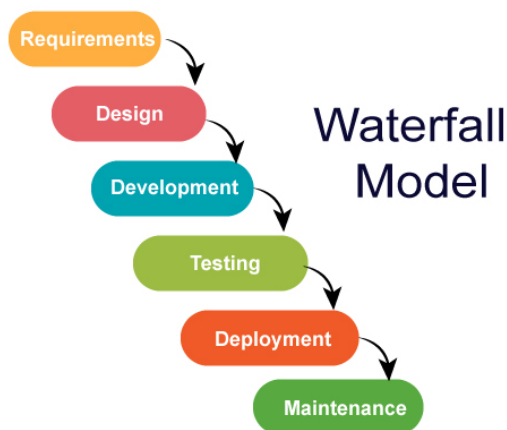


Figure 4 waterfall method

### **3.2.Functional requirements**

The goal of the application is to manage the transaction process in a retail business environment. Once it is automated all the transaction functions of the business it can be effectively managed, and the organization can achieve the competitive advantage. So, product features or functions that analysis must implement to enable users to accomplish their tasks was:

- Helps to search the specific product and remaining stock.
- All transactions have specific entry date along with quantity and rate.
- Be able to input items via barcode and via specific product code.
- Both of admin users and normal are allowed to input business transactions.
- Must allow to remove unnecessary items before calculate subtotal.
- Must be able to return a bill as a hard copy.
- Calculator must come as a built-in feature.

Generally, these functional requirements describe system behavior under specific conditions.

### **3.3.Nonfunctional requirements**

The perfect explanation to the non-functional requirement is “what a system should be” rather than “what a system should do”. So, nonfunctional requirements are documented as below,

- Fast loading time, calculation, and data insertion.
- Easy to learn for nontechnical employees.
- Must be able to go back (under a certain restriction) to resolve mistakes.
- User friendly user interface environment. (But under a limited colors set)
- Ability to maintain or upgrade according to changing user requirements or technology changes.

These nonfunctional requirements talk about the working behavior of a whole system or a component of the system and not a particular function.



### **3.4. Feasibility analysis**

A feasibility study is an assessment of the practicality of a software project or system. While planning this software I have managed to apply feasibility analysis into certain degree considering village retail business shop in my village.

#### **3.4.1. Economic feasibility**

This system is estimated as an economically affordable project. The benefits include increased efficiency, effectiveness, and the better performance of a small business environment. Comparing the cost and benefits of the software solution, software is found to be economically feasible.

#### **3.4.2. Technical feasibility**

Following software and hardware components are needed for development of the current project.

- INTELIJ Idea Ultimate Edition
- Java 8.0 Update 333 (8u333)
- MYSQL community edition
- Scene builder
- Barcode reader
- Thermal printer

Unfortunately barcode reader and a thermal printer were unable to acquire considering prevailing situation in the country. There for, barcode scanning feature and bill printing feature had to be cut down.

#### **3.4.3. Operational feasibility**

This software will have increased efficiency, effectiveness, and the better performance than the current manual system in the retail business shop in my village.

#### **3.4.4. Time feasibility**

A time feasibility study will consider the time frame in which the project is going to take up to its full completion. A project will fail if it takes too long to be completed before it is useful. So, I decided this project should be delivered within a one month considering this mandatory time frame given by the university.

#### **3.5. Conclusion**

These requirements are the building blocks of a software system development. Even though, it is possible to build a software system without any of the analysis, but the abilities of the software will never be determined nor measured. Hence functional requirements give the direction of implementation of this proposed software system and non-functional requirements determine the quality of implementation that end-users will experience.

## 4. System design

The objective of this entire chapter is to build a physical model that will fulfill the requirements of the proposed transaction management system. This phase is responsible for identifying inputs, outputs, processes as well as designing the user interfaces.

### 4.1.Process flow diagram

A Process Flow Diagram (PFD) is a type of flowchart that illustrates the relationships between major components at an industrial plant. [12]

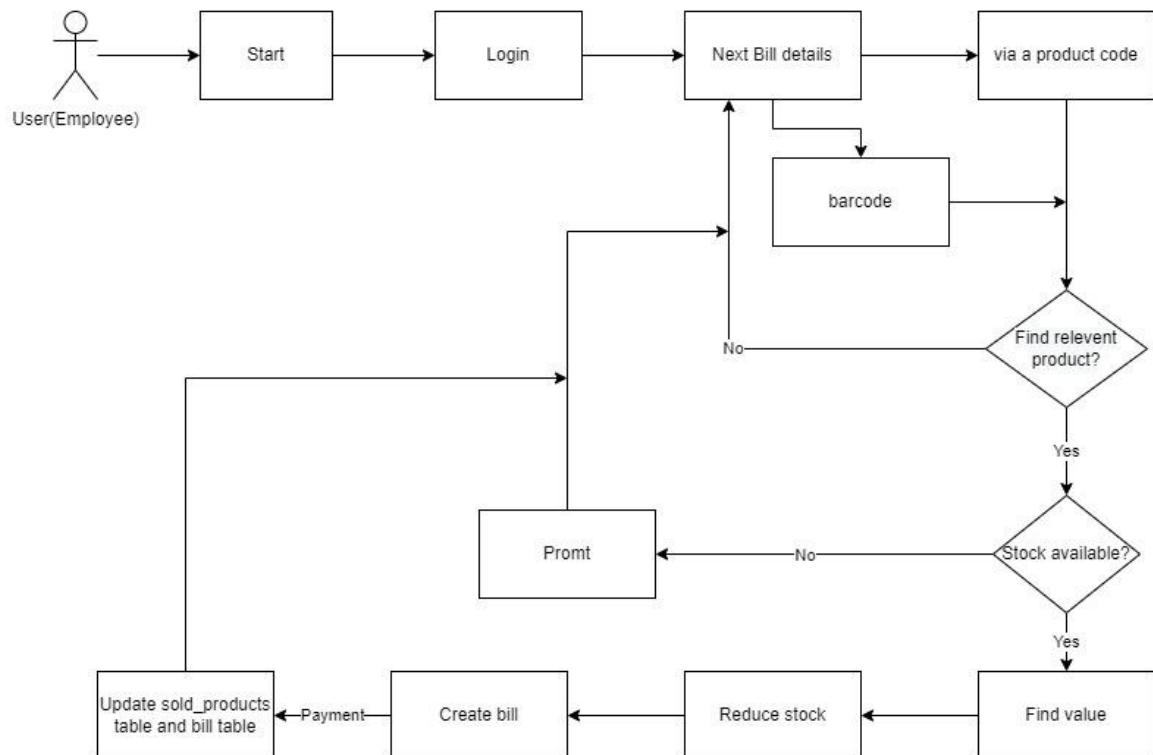


Figure 5 Process flow diagram

## 4.2.Flow chart

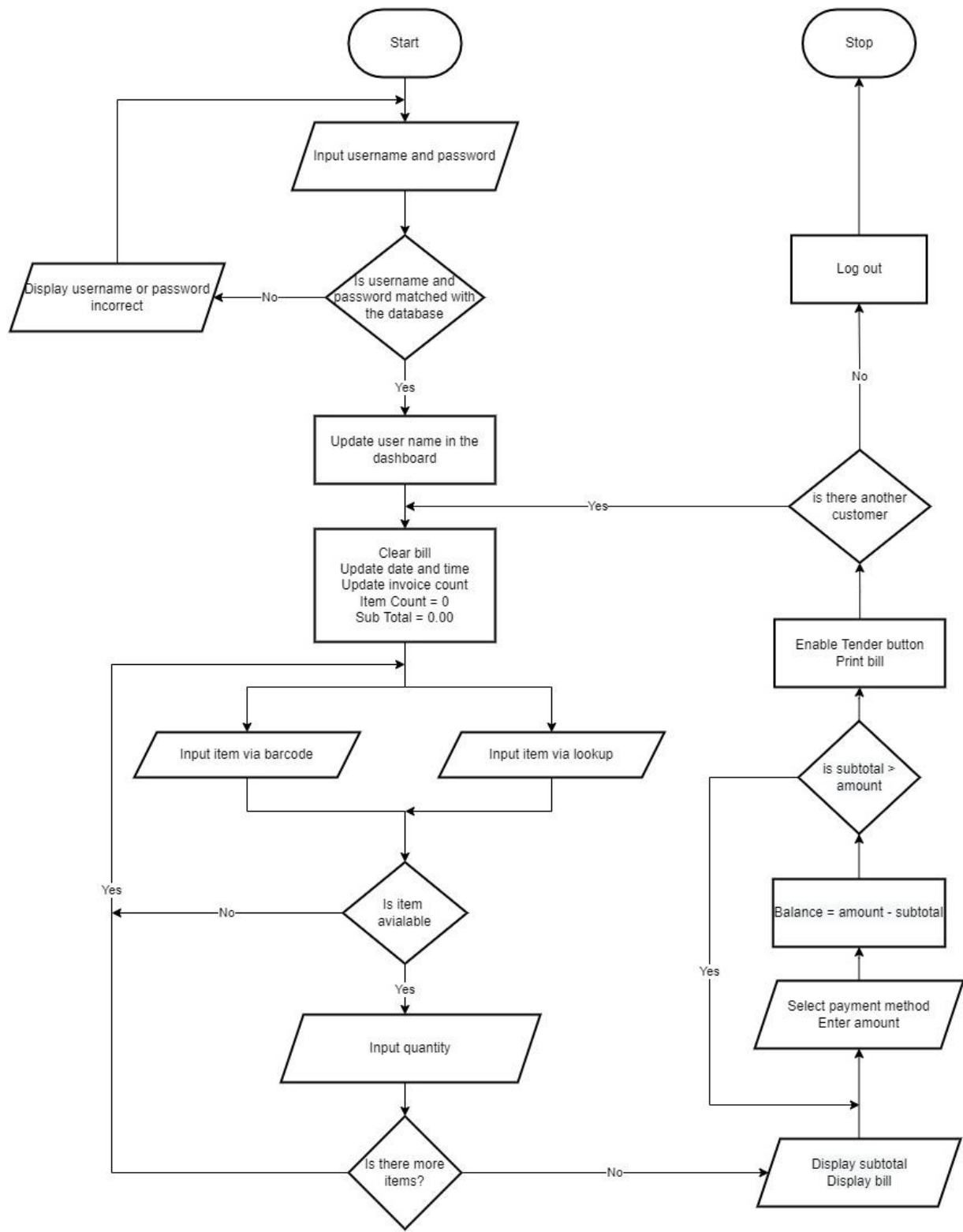


Figure 6 Flow chart

### 4.3.Pseudo code

```
Begin
  Input username and password
  IF username and password is not matched with the database THEN
    Display "username or password is incorrect"
  ELSE
    Update user name in the dashboard
    Clear bill
    Update date and time
    Update invoice count
    Item Count = 0
    Sub Total = 0.00
    Input item via barcode OR Input item via lookup

    WHILE item available in the inventory THEN
      Prompt distinct prices in the database for the entered item
      Input quantity

      Input item via barcode OR Input item via lookup
    END WHILE

    Display subtotal
    Display bill
    amount = 0;

    WHILE is subtotal > amount THEN
      Select payment method and enter amount
      Balance = amount - subtotal
    END WHILE

    Enable Tender button
    Print bill
  ENDIF
END
```

#### 4.4.Context diagram

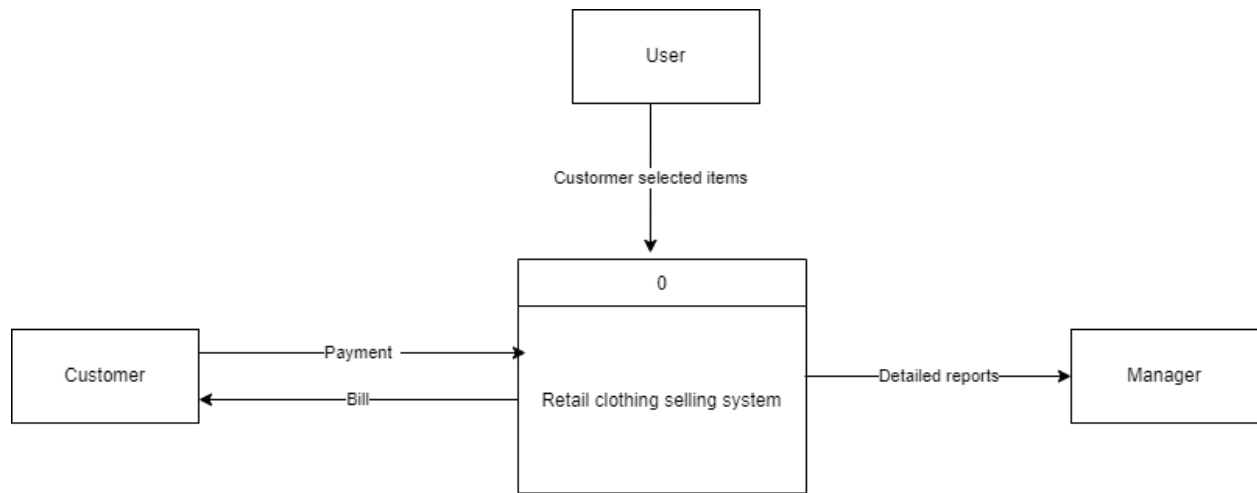


Figure 7 Context diagram

## 4.5.DFD 0 Diagram

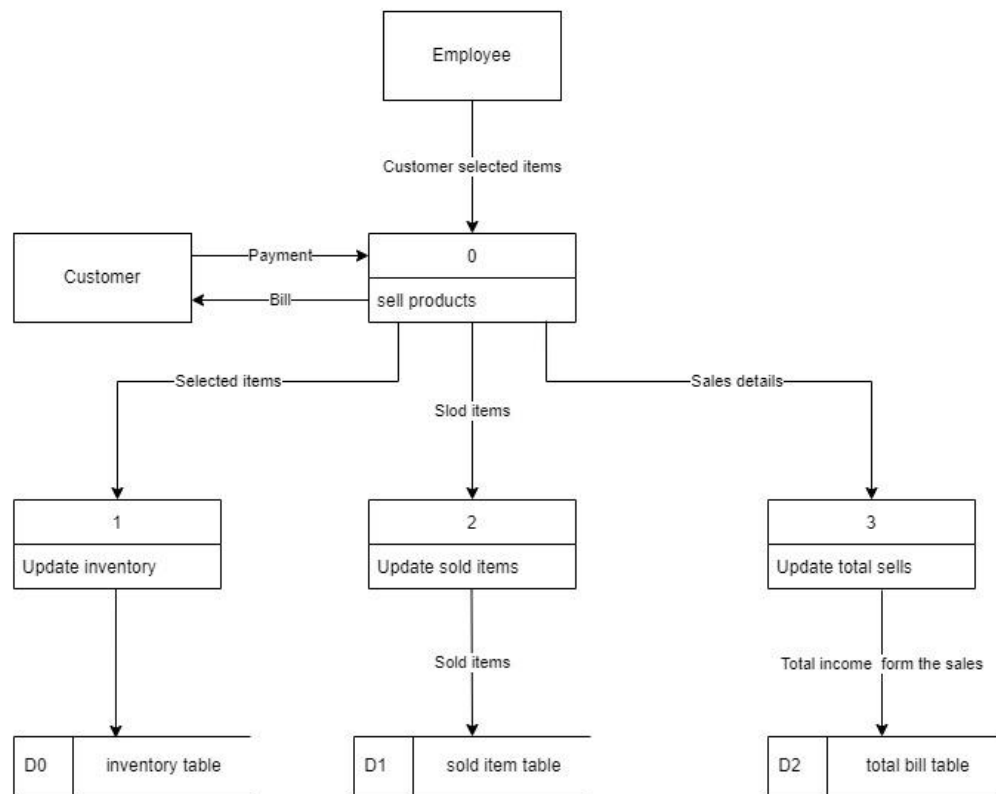


Figure 8 DFD 0 Diagram

## 4.6.Summarization

System design using both graphical and verbal representation make system design easier to understand. Moreover, Visual Clarity, Instant Communication, Effective Coordination, increased efficiency, Effective Analysis and Proper Documentation are other key benefits of effective system design.

## 5. Tools and technology that used

### 5.1. Development tools

INTELIJ Ida Ultimate version were used to develop graphical user interface applications along with Windows Form applications. IntelliJ IDEA is an integrated development environment written in Java for developing computer software. It is developed by JetBrains and is available as an Apache 2 Licensed community edition, and in a proprietary commercial edition [13]. Even though, there are stable versions from java 17 I used java 8 Development kit. Since Java is a high-level, class-based, object-oriented programming language it was a perfect match for my requirements.

According to Wikipedia 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. A relational database organizes data into one or more data tables in which data may be related to each other; these relations help structure the data.

### 5.2. Architectural designing

System architecture translates the logical design of an information system into a physical structure that includes hardware, software, database support, processing methods, and security using many software architectures. Here I used MVC architecture to design my Transaction management software. This pattern divides an interactive application in to 3 parts as,

- Model - contains the core functionality and data.
- View - displays the information to the user (more than one view may be defined)
- Controller - handles the input from the user

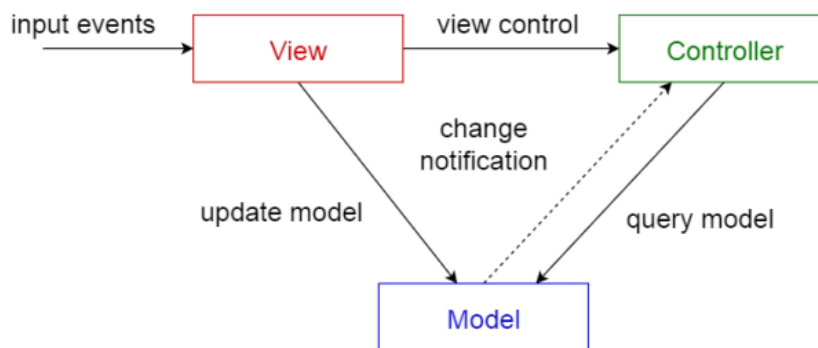


Figure 9 Model-view-controller pattern



## 6. System implementation

### 6.1.Application code structure

This transaction management software was designed using INTELIJ Idea according to 3 tier database architecture and MVC pattern architecture. It provided a blank space for explore my independence while code writing.

Code files were divided into 3 main packages according to MVC architecture. Such as, view, model, controller. Scene builder JavaFX Scene Builder is a visual layout tool that lets users quickly design JavaFX application user interfaces, without coding. It was quite helpful to generate FXML codes according to my user interface design. Also, database connection and data encapsulation packages were used for creating database connection and for create observable lists respectively.

Java OOP concepts like inherence and data encapsulation widely used in this project. (The capability of a class to derive properties and characteristics from another class is called Inheritance and Data encapsulation, also known as data hiding, is the mechanism whereby the implementation details of a class are kept hidden from the user). Furthermore, inherited was implemented as follows,

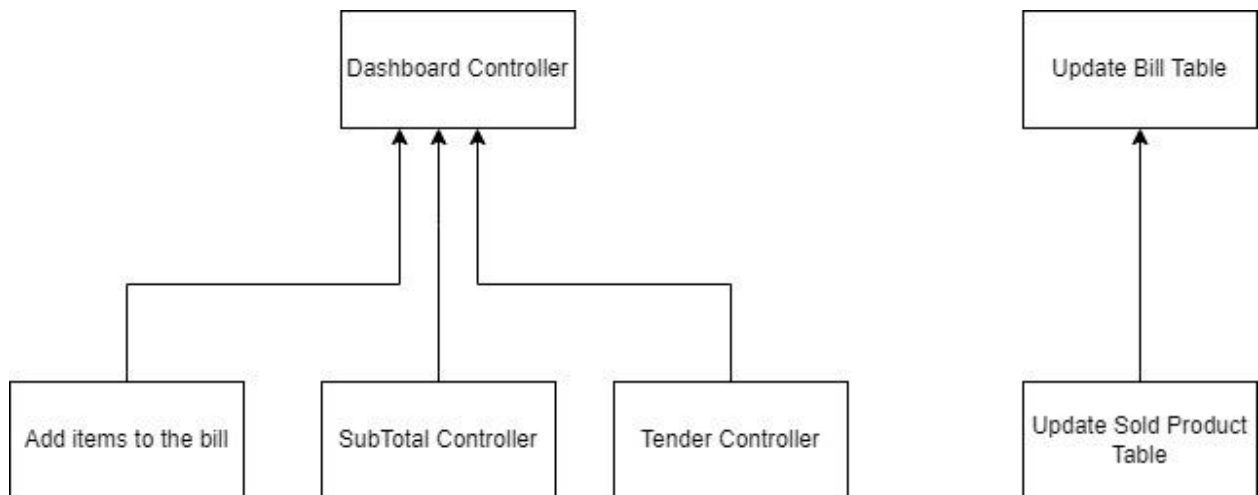


Figure 10 Data inheritance

## 6.2.Controller package

### 6.2.1. Login page

```
String userName = txtUserName.getText();
String passwordText = txtPassword.getText();

try {
    PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM
employee WHERE user_id = ? AND password = ?;");
    preparedStatement.setObject(1,userName);
    preparedStatement.setObject(2,passwordText);
    ResultSet resultSet = preparedStatement.executeQuery();
    boolean next = resultSet.next();

    if (next){
        lblUserName = resultSet.getString("user_id");
        System.out.println(lblUserName);
        Parent parent = FXMLLoader.load(this.getClass().getResource("../View/Dashboard.fxml"));
        Scene scene = new Scene(parent);

        Stage primaryStage = (Stage) root.getScene().getWindow();
        primaryStage.setTitle("Dashboard");

        Image image = new Image("Images/pic001.PNG");
        primaryStage.getIcons().add(image);
        //    primaryStage.centerOnScreen();
        primaryStage.setScene(scene);
    }
    else {
        Alert user_name_or_password_incorrect = new Alert(Alert.AlertType.ERROR, "User name or
password incorrect", ButtonType.OK);
        user_name_or_password_incorrect.showAndWait();
        txtUserName.setStyle("-fx-border-color: #e74c3c");
        txtPassword.setStyle("-fx-border-color: #e74c3c");
        txtUserName.requestFocus();
    }
}
```

### 6.2.2. Dashboard controller

```
public void txtQuantityOnKeyPressed(KeyEvent keyEvent) {
    KeyCode code = keyEvent.getCode();
    if(code == KeyCode.ENTER){
        String userName = lblUserName.getText();           //Change at implementation

        AddItemToTheBill addItemToTheBill = new AddItemToTheBill();

        int setCount = addItemToTheBill.checkTheInventory(txtQuantity.getText(), barCode,userName);
        lblItemCount.setText(String.valueOf(setCount));
        lblSubTotal.setText(" RS. \n "+subTotal);

        tblTempBill.getItems().clear();

        clearInputFields();

        lblItemName.setText("");
        LoadTable loadTable = new LoadTable();
        loadTable.loadTable(tblTempBill);

    }
}

public void newBillOnAction(ActionEvent actionEvent) {
    CheckItemCart checkItemCart = new CheckItemCart();
    boolean result = checkItemCart.checkTempBill();
    ResetDashBoard resetDashBoard = new ResetDashBoard();

    if (result==false){
        tblTempBill.getItems().clear();

        LoadTable loadTable = new LoadTable();
        loadTable.loadTable(tblTempBill);

        lblBillNo.setText(resetDashBoard.resetBillNo());
        lblItemCount.setText(String.valueOf(itemCount));
        lblSubTotal.setText(" RS. \n "+subTotal);
    }

    lblDateTime.setText(resetDashBoard.resetDate());
}
```

```

//txtBarcode press enter
public void txtItemCodeOnKeyPressed(KeyEvent keyEvent) throws SQLException {
    if(keyEvent.getCode() == KeyCode.ENTER){
        switch (selectedMethod){
            case 'A':
                barCode = LookUpController.barCode;
                String itemName = SearchItem.SearchItemWithBarCode(barCode);
                lblItemName.setText(itemName);
                lblItemName.setVisible(true);
                if (!itemName.equals("Not Found")){
                    txtQuantity.requestFocus();}
                selectedMethod = 'B';
                break;
            case 'B':
//                add items via barcode
                barCode = txtBarcode.getText();
                itemName = SearchItem.SearchItemWithBarCode(barCode);
                lblItemName.setText(itemName);
                lblItemName.setVisible(true);
                if (!itemName.equals("Not Found")){
                    txtQuantity.requestFocus();
                }
                break;
            case 'C':
//remove items
                itemName = SearchItem.SearchItemWithItemCode(txtBarcode.getText());
                lblItemName.setText(itemName);
                lblItemName.setVisible(true);
                if (!itemName.equals("Not Found")){
                    Alert confirm = new Alert(Alert.AlertType.WARNING, "Are you sure you want to remove " +
itemName, ButtonType.YES, ButtonType.NO);
                    Optional<ButtonType> buttonType = confirm.showAndWait();

                    if (buttonType.get().equals(ButtonType.YES)){
                        RemoveItemFromTheBill removeItemFromTheBill = new RemoveItemFromTheBill();
                        removeItemFromTheBill.removeItem(txtBarcode.getText());

                        tblTempBill.getItems().clear();
                        LoadTable loadTable = new LoadTable();
                        loadTable.loadTable(tblTempBill);
                    }
                }
                clearInputFields();
                selectedMethod = 'B';break; } } }

```

### 6.2.3. Look up controller

```
public void txtSearchOnAction(ActionEvent actionEvent) {
    String keyWord = txtSearch.getText();
    ObservableList<InventoryTM> items = tblLookUp.getItems();
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();

    //search data in the inventory
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM inventory
WHERE item_code LIKE ? OR description = ?");
        preparedStatement.setObject(1,keyWord+"%");
        preparedStatement.setObject(2,keyWord+"%");
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        System.out.println(next);
        items.clear();

        while (next){
            items.add(new InventoryTM(resultSet.getString("barcode"),resultSet.getString("item_code")
,resultSet.getString("description") ,resultSet.getString("count") ,resultSet.getString("selling_price")));
            resultSet.next();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    tblLookUp.refresh();

    tblLookUp.getColumns().get(0).setCellValueFactory(new PropertyValueFactory<>("barCode"));
    tblLookUp.getColumns().get(1).setCellValueFactory(new PropertyValueFactory<>("itemCode"));
    tblLookUp.getColumns().get(2).setCellValueFactory(new PropertyValueFactory<>("description"));
    tblLookUp.getColumns().get(3).setCellValueFactory(new PropertyValueFactory<>("quantity"));
    tblLookUp.getColumns().get(4).setCellValueFactory(new PropertyValueFactory<>("SellingPrice"));
}
```

## 6.2.4. Subtotal controller

```
public void txtAmountOnKeyPressed(KeyEvent keyEvent) {
    if(keyEvent.getCode()== KeyCode.ENTER){

        //check sufficiency of given ammount
        String amountText = txtAmount.getText();
        double parseDoubleAmount = Double.parseDouble(amountText);
        balance = balance + parseDoubleAmount;
        lblBalance.setText("Rs. "+balance);

        if (balance>0){
            btnTender.setDisable(false);
        }
    }
}

public void rootOnKeyReleased(KeyEvent keyEvent) {
    if(keyEvent.getCode()==KeyCode.ESCAPE){
        Stage primaryStage = (Stage) root1.getScene().getWindow();
        primaryStage.close();
    }
}

public void btnTenderOnActionSubTotal(ActionEvent actionEvent) {
    Object selectPaymentMethodValue = selectPaymentMethod.getValue();

    UpdateBillTable updateBillTable = new UpdateBillTable();
    updateBillTable.updateBillTable(subTotal,itemCount,lblUserName,selectPaymentMethodValue);
    updateBillTable.updateSoldProductTable();
    updateBillTable.clearTempBillTable();

    subTotal = 0;
    itemCount = 0;

    Stage primaryStage = (Stage) root1.getScene().getWindow();
    primaryStage.close();
}
```

### 6.2.5. Tender Controller

```
public void rootOnKeyReleased(KeyEvent keyEvent) {
    if(keyEvent.getCode()==KeyCode.ESCAPE){
        Stage primaryStage = (Stage) root1.getScene().getWindow();
        primaryStage.close();
    }
}

public void selectPaymentMethodOnKeyPressed(KeyEvent keyEvent) {
    btnTender.setDisable(false);
}

public void btnTenderOnActionF4(ActionEvent actionEvent) throws IOException {
    Object selectPaymentMethodValue = selectPaymentMethod.getValue();

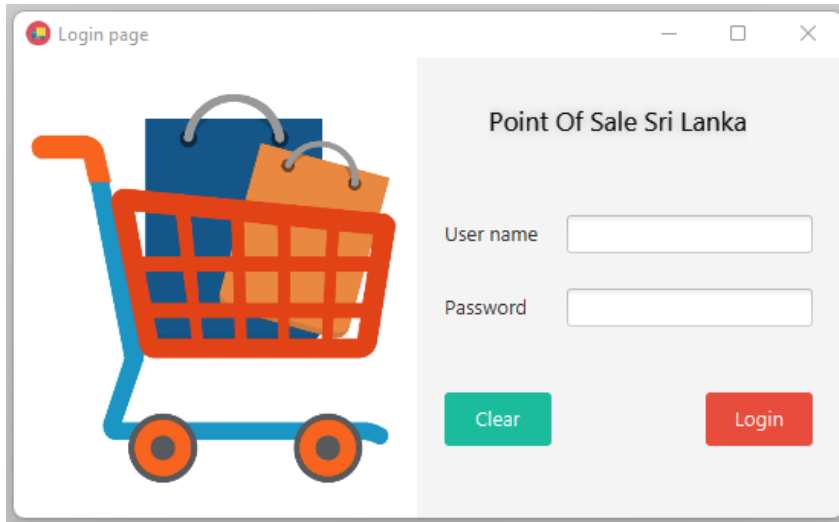
    UpdateBillTable updateBillTable = new UpdateBillTable();
    updateBillTable.updateBillTable(subTotal,itemCount,lblUserName,selectPaymentMethodValue);
    updateBillTable.updateSoldProductTable();
    updateBillTable.clearTempBillTable();

    subTotal = 0;
    itemCount = 0;

    Stage primaryStage = (Stage) root1.getScene().getWindow();
    primaryStage.close();
}
```

## 6.3.View package

### 6.3.1. Login page FXML



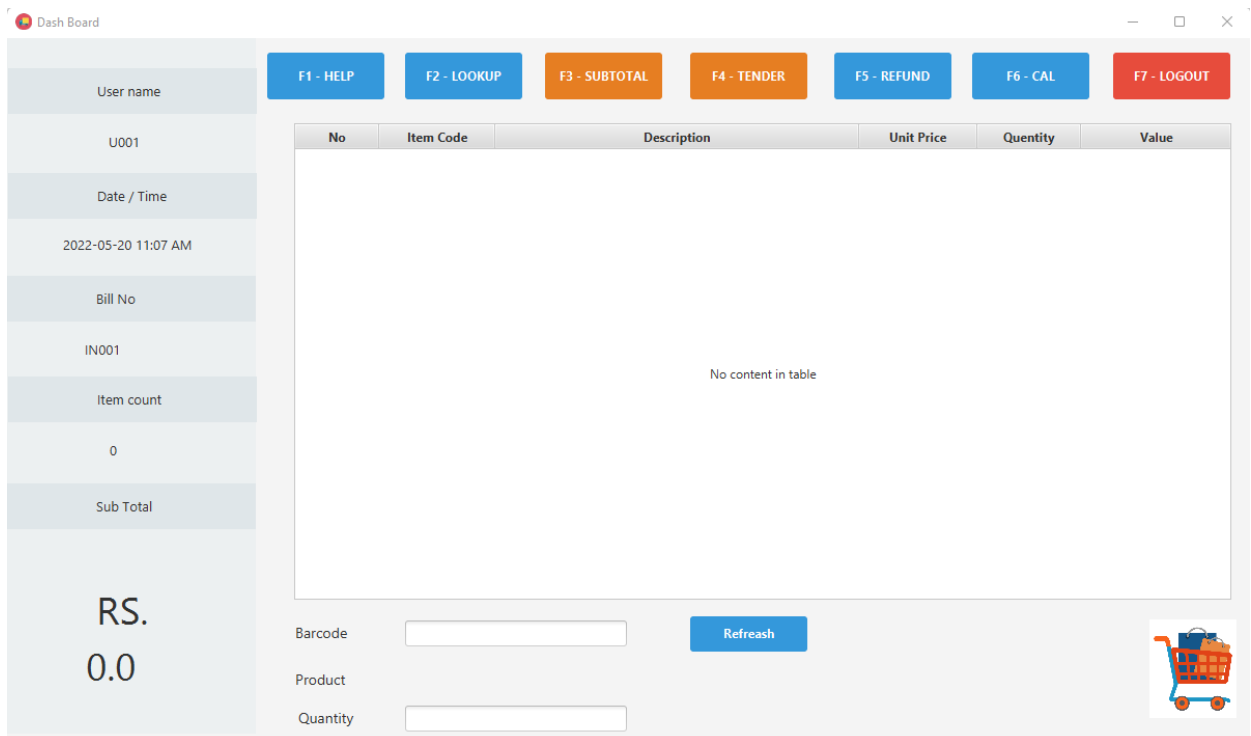
Point Of Sale Sri Lanka

User name

Password

Figure 11 Login page

### 6.3.2. Dashboard FXML



Dashboard

User name: U001

Date / Time: 2022-05-20 11:07 AM

Bill No: IN001

Item count: 0

Sub Total: RS. 0.0

Navigation: F1 - HELP, F2 - LOOKUP, F3 - SUBTOTAL, F4 - TENDER, F5 - REFUND, F6 - CAL, F7 - LOGOUT

No	Item Code	Description	Unit Price	Quantity	Value
No content in table					

Barcode:

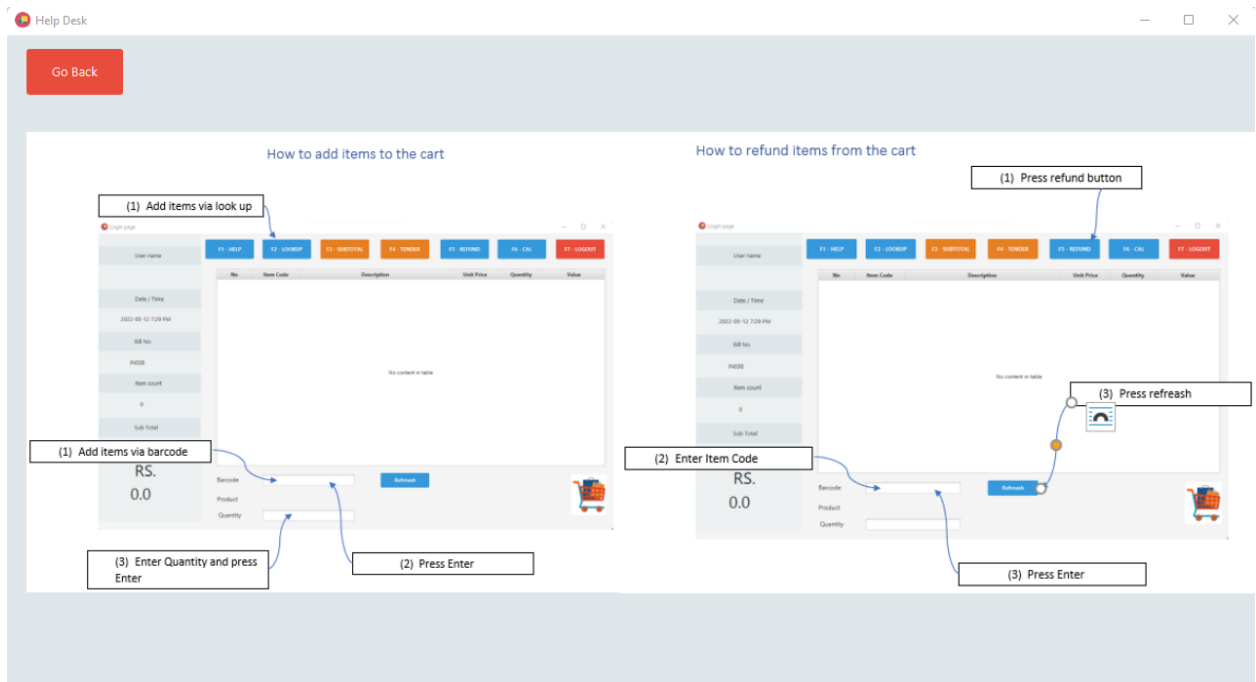
Product:

Quantity:

Figure 12 Dashboard



### 6.3.3. Helpdesk FXML



### 6.3.5. Subtotal view

The screenshot shows a window titled "Sub Total" with a header for "Rathnayaka Stores" at "No27,Kadawatha Rd,Kiribathgoda" with "Tel : 0332233476". Below this, it displays "Sales men: U001", "Date: 2022-05-20", "BillNo:0001", and "Time: 11:10:42". A table lists two items: "1. maliban real chocholate biscuit 400g" (Price 280, Qty 2, Amount 560.0) and "2. dedunu table sault 400g" (Price 55, Qty 3, Amount 165.0). The "NET TOTAL:" is 725.0. On the right, "Sub Total" is Rs. 725.0, "Payment Method" is set to "Cash", "Amount" is 1000, and "Balance" is Rs. 275.0. A red "Tender" button is at the bottom right.

Product	Price	Qty	Amount
1. maliban real chocholate biscuit 400g	280	2	560.0
2. dedunu table sault 400g	55	3	165.0
NET TOTAL:			725.0

Sub Total : Rs. 725.0

Payment Method: Cash

Amount : 1000

Balance : Rs. 275.0

Tender

Figure 15 Sub total

### 6.3.6. Tender view

The screenshot shows a window titled "Tender" with the same header as Figure 15. It displays "Sales men: U001", "Date: 2022-05-20", "BillNo:0001", and "Time: 11:12:38". The table lists the same two items with a "NET TOTAL:" of 725.0. On the right, "Sub Total" is Rs. 725.0, "Payment Method" is set to "Credit card", "Amount" is Rs. 725.0, and "Balance" is Rs. 0.00. A red "Tender" button is at the bottom right.

Product	Price	Qty	Amount
1. maliban real chocholate biscuit 400g	280	2	560.0
2. dedunu table sault 400g	55	3	165.0
NET TOTAL:			725.0

Sub Total : Rs. 725.0

Payment Method: Credit card

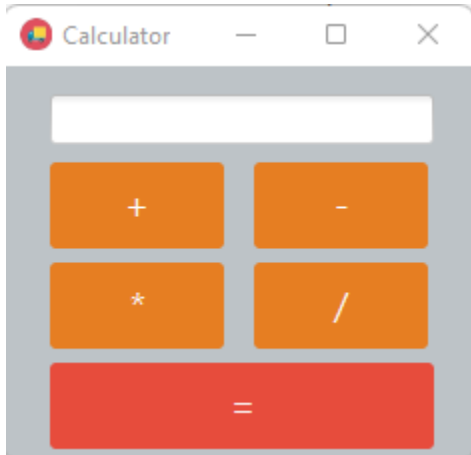
Amount : Rs. 725.0

Balance : Rs. 0.00

Tender

Figure 16 Tender view

### 6.3.7. Calculator view



*Figure 17 Calculator*

## 7. Appendix

I have implemented all of the business logic in the model package. Because in MVC architecture The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. [14]

### 7.1.Add item to the bill

```
public int checkTheInventory(String quantity, String barCode, String userName) {
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT count FROM
inventory WHERE barcode = ?");
        preparedStatement.setObject(1,barCode);
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();

        if (next){
            String availableCount = resultSet.getString(1);
            double doubleAvalableCount = Double.parseDouble(availableCount);
            double doubleQuantity = Double.parseDouble(quantity);
            if (doubleQuantity<doubleAvalableCount){
                itemCount= itemCount+1;
                double newQty = doubleAvalableCount-doubleQuantity;
                String stringNewQty = String.valueOf(newQty);

                updateInventoryTable(barCode,stringNewQty,connection);

                fillTempBillTable(quantity, barCode,userName);
            }
            else {
                Alert exceeded_inventory = new Alert(Alert.AlertType.INFORMATION, "Exceeded inventory",
ButtonType.YES);
                exceeded_inventory.showAndWait();
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return itemCount;
}
```

## 7.2.Load table

```
public void loadTable(Table<TempBillTableTM> tblTempBill){
    ObservableList<TempBillTableTM> items = tblTempBill.getItems();

    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM
temp_bill;");
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        while (next){
            items.add(new TempBillTableTM(resultSet.getString("item_no"),resultSet.getString("item_code")
,resultSet.getString("description") ,resultSet.getString("unit_price") ,resultSet.getString("quantity")
,resultSet.getString("value") ));
            resultSet.next();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

    tblTempBill.refresh();

    tblTempBill.getColumns().get(0).setCellValueFactory(new PropertyValueFactory<>("item_no"));
    tblTempBill.getColumns().get(1).setCellValueFactory(new PropertyValueFactory<>("item_code"));
    tblTempBill.getColumns().get(4).setCellValueFactory(new PropertyValueFactory<>("description"));
    tblTempBill.getColumns().get(2).setCellValueFactory(new PropertyValueFactory<>("unit_price"));
    tblTempBill.getColumns().get(3).setCellValueFactory(new PropertyValueFactory<>("quantity"));
    tblTempBill.getColumns().get(5).setCellValueFactory(new PropertyValueFactory<>("value"));
}
```

### 7.3.Search product

```
public static String SearchItemWithBarCode(String barCode){
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM inventory
WHERE barcode = ?;");
        preparedStatement.setObject(1,barCode);
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        if (next){
            String description = resultSet.getString("description");
            return description;
        }
        return "Not Found";
    } catch (SQLException e) {
        e.printStackTrace();
        return "Not Found"; } }

public static String SearchItemWithItemCode(String itemCode){
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM inventory
WHERE item_code = ?;");
        preparedStatement.setObject(1,itemCode);
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        System.out.println(itemCode);

        if (next){
            String description = resultSet.getString("description");
            return description;
        }
        return "Not Found";
    } catch (SQLException e) {
        e.printStackTrace();
        return "Not Found"; } }
```

## 7.4.Update bill table

//get previous invoice id and make a new one

```
PreparedStatement preparedStatement = connection.prepareStatement("SELECT invoice_no FROM bill  
ORDER BY invoice_no DESC LIMIT 1");
```

```
ResultSet resultSet = preparedStatement.executeQuery();
```

```
boolean next = resultSet.next();
```

```
if (next){
```

```
    invoiceNo = resultSet.getString("invoice_no");
```

```
    String invoiceNumbers = invoiceNo.substring(2, invoiceNo.length());
```

```
    int id = Integer.parseInt(invoiceNumbers);
```

```
    id++;
```

```
    if (id<10){
```

```
        invoiceNo="IN00"+id;
```

```
    }
```

```
    else if (id<100){
```

```
        invoiceNo="IN0"+id;
```

```
    }
```

```
    else {
```

```
        invoiceNo="IN"+id;
```

```
    }
```

```
}
```

```
else {
```

```
    invoiceNo="IN001";
```

```
}
```

//insert in to bill

```
try {
```

```
    PreparedStatement preparedStatement = connection.prepareStatement("INSERT INTO bill VALUES  
(?,?,?, ?, ?, ?)");
```

```
    preparedStatement.setObject(1,invoiceNo);
```

```
    preparedStatement.setObject(2,subTotal);
```

```
    preparedStatement.setObject(3,dateTime());
```

```
    preparedStatement.setObject(4,itemCount);
```

```
    preparedStatement.setObject(5,lblUserName);
```

```
    preparedStatement.setObject(6,selectPaymentMethodValue.toString());
```

```
    preparedStatement.executeUpdate();
```

```
} catch (SQLException e) {
```

```
    e.printStackTrace();
```

```
}
```

## 7.5.Update sold product table

```
public void updateSoldProductTable() {
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();

    try {
        //get product details
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT * FROM
temp_bill");
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        while (next){

            String itemCode = resultSet.getString("item_code");
            String description = resultSet.getString("description");
            String unitPrice = resultSet.getString("unit_price");
            String quantity = resultSet.getString("quantity");
            String value = resultSet.getString("value");

            //      getPurchasedPrice
            String purchasedPrice = getPurchasedPrice(itemCode, connection);
            //update sold product table

            addItemToSoldProductTable(itemCode,description,unitPrice,quantity,value,purchasedPrice,connection
);

            resultSet.next();
        }
    } catch (SQLException sqlEx) {
        sqlEx.printStackTrace();
    }
}
```



## 7.6.Reset dashboard

```
public String resetBillNo() {
    DataBaseConnection dataBaseConnection = new DataBaseConnection();
    Connection connection = dataBaseConnection.getConnection();

    //get previous invoice id and make a new one
    try {
        PreparedStatement preparedStatement = connection.prepareStatement("SELECT invoice_no FROM
bill ORDER BY invoice_no DESC LIMIT 1");
        ResultSet resultSet = preparedStatement.executeQuery();
        boolean next = resultSet.next();
        if (next){
            invoiceNo = resultSet.getString("invoice_no");
            String invoiceNumbers = invoiceNo.substring(2, invoiceNo.length());
            int id = Integer.parseInt(invoiceNumbers);
            id++;
            if (id<10){
                invoiceNo="IN00"+id;
            }
            else if (id<100){
                invoiceNo="IN0"+id;
            }
            else {
                invoiceNo="IN"+id;
            }
        }
        else {
            invoiceNo="IN001";
        }
        System.out.println(invoiceNo);
        return invoiceNo;
    } catch (SQLException e) {
        e.printStackTrace();
        return "Error";
    }
}
```

## 8. Debugging and testing

Debugging and testing plays a vital part in software development. All of the testing was performed in this project by manually. This process allows developers to detect and remove existing and potential errors in a software code that can cause it to behave unexpectedly or crash. Furthermore, testing can be verification and validation or reliability estimation. The objectives of debugging testing are,

- Identify potential errors in the system.
- Check the proper working of the application while data insertion.
- Ensures the product under development is as per the user requirements.

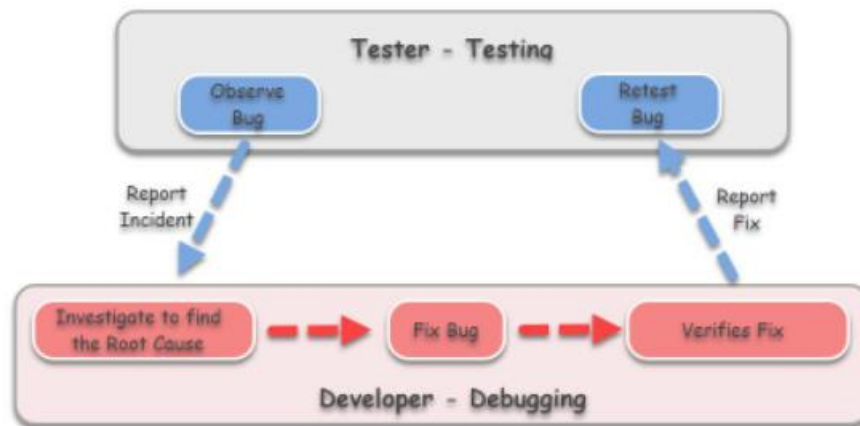


Figure 18 Testing process

### 8.1. Unit testing

This type of testing is the testing of individual components of the software system. It is typically done by me while coding. It requires in-depth information and vast knowledge about the internal program design and software code. Which means white box testing was conducted to test program and its implementation, in order to improve code efficiency or structure.

## **8.2.Integration testing**

Even if the small units of software are working fine individually, we need to find out whether the units can work without errors when they are integrated together or not.

## **8.3.System Testing**

After the software is compiled as product and then it is tested as a whole system. It was accomplished by following three main tests,

- Function testing - Tests all functionalities of the software against the requirement.
- Performance testing - This test proves how efficient, effective the software is and average time taken by the software to do desired task.
- Security and portability - These tests are done when the software is meant to work on various platforms and accessed by number of persons.

## **8.4.Special note**

Unfortunately, acceptance testing was unable to accomplish due to two main reasons. They will be further discussed in chapter 9.

## **9. Conclusion and lessons learnt**

### **9.1. Project Limitations**

Due to less knowledge in certain areas and lack of resources such as a barcode reader and a thermal printer, I was unable to apply several mechanics into the transaction processing software. Such as,

- Input via barcode scanner
- Print bill to the customer

Also, without any human resources by my side it was almost impossible to implement a software of this scale. For instance, it took me an almost a day to learn how to populate data in FXML table. Furthermore, fix data communication between classes made me nearly quit. But thanks to the lectures of university of Moratuwa, I was able to come this far.

### **9.2. Lessons learned**

I was able to learn uncountable lessons during this assignment. Some of them as follows:

- Learn about integrated software development environment.
- Prepare a solution to a simple programming problem by identifying the problem, and its functional, and non-functional requirements.
- Design a solution that matches the requirements.
- Use MYSQL database in software implementation.
- Preparing a complete project report.

### **9.3.Future enhancements**

Some of the scope I can increase for the betterment and effectiveness oar listed below:

- Design more interactive software design.
- Apply Inventory management concepts like “Reorder volume” to business environment.
- Increase user access levels into at least 3 types.
- Include more forms like add inventory.
- Improve security.
- Introduce cloud technology for small business entrepreneurs.

### **9.4.To summarize**

This transaction management software is mostly suitable for small business environments. Even though, it has several limitations they will be addressed in a future update. Other than that software has been developed according to the user requirements which were gathered from village retail shop.

## 10. References

- [1 Anon, "National policy framework for SME development. Sri Lanka," Ministry of Industry and Commerce., 2015.
- [2 C. Bandara, "What causes SMEs to fail in Sri Lanka?," [Online]. Available: <https://www.dailymirror.lk/101755>. [Accessed 23 2 2022].
- [3 Anon, "National strategy for small and medium enterprise sector development in Sri Lanka," in *Ministry of Enterprise Development*, Colombo, 2002.
- [4 P. F. R. B.E.A. Jayasekaraa, "A SYSTEMATIC LITERATURE REVIEW ON BUSINESS FAILURE OF SMALL AND," University of Kelaniya, Sri Lanka, Colombo, 2020.
- [5 A. ONIBALUSI, "thebalancesmb.com," 21 August 2019. [Online]. Available: <https://www.thebalancesmb.com/how-to-overcome-small-business-failure-4142683>. [Accessed 11 04 2022].
- [6 F. W. & J. Skriveris, "The Rise of SME - Technologies Helping SMEs to Compete and Save," 5 1 2021. [Online]. Available: <https://www.plugandplaytechcenter.com/resources/rise-sme-technologies-helping-smes-compete-and-save/>. [Accessed 10 5 2022].
- [7 P. & B. Jennings, "The managerial dimension of small business failure.," 2005.
- [8 G. f. g. org, "Geeks for Geeks," 14 01 2020. [Online]. Available: <https://www.geeksforgeeks.org/what-is-net-3-tier-architecture/>. [Accessed 19 05 2022].
- [9 Wikipida. [Online]. Available: [https://en.wikipedia.org/wiki/Relational\\_database#cite\\_note-2](https://en.wikipedia.org/wiki/Relational_database#cite_note-2). [Accessed 18 05 2022].
- [1 I. Corporation, "IBM Corporation," IBM Corporation, 2014. [Online]. Available: <https://www.ibm.com/docs/en/iodg/11.3?topic=reference-primary-keys>. [Accessed 23 05 2022].
- [1 J. Rabelo, "Foreign Key," 14 08 2020. [Online]. Available: <https://www.techopedia.com/definition/7272/foreign-key>. [Accessed 23 05 2022].
- [1 www.lucidchart.com, "lucidchart.com," lucidchart, [Online]. Available: <https://www.lucidchart.com/pages/process-flow-diagrams>. [Accessed 19 05 2022].
- [1 Community, "GitHub," [Online]. Available: <https://github.com/JetBrains/intellij-community>. [Accessed 19 05 2022].

- [1 tutorialspoint, "tutorialspoint.com," [Online]. Available:
- 4] [https://www.tutorialspoint.com/mvc\\_framework/mvc\\_framework\\_introduction.htm#:~:text=The%20Model%2DView%2DController%20\(%20development%20aspects%20of%20an%20application..](https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm#:~:text=The%20Model%2DView%2DController%20(%20development%20aspects%20of%20an%20application..)  
[Accessed 20 05 2022].