UNIVERSITI TEKNIKAL MALAYSIA MELAKA

FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

W O R K S H O P 1 R E P O R T

Name	Nurzakiyah Emalda Binti Abdul Jamal
Matric Numbers	B031720006
Course	BITS
Project Title	Legal Firm File Management System (LFMS)
Supervisor	Dr. Mohd Sanusi Bin Azmi

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Introduction

"Salina Taib & Co." is a legal firm based in Melaka. The firm consist of highly skilled and experienced associates who routinely assist business and individuals in diverse litigation matters.

Nowadays, Information and Communication Technology (ICT) have plays a great role in different fields. In order to exploit the ICT in the legal firm, Legal Firm File Management System (LFMS) is being proposed. LFMS is robust and integrated technology. LFMS will helps to maintain the file management of the legal firm. This system provides a user friendly interface for managing the files and location of the files which will enhance the efficiency of the legal firm file management and ease user's convenience.

In general, the Legal Firm File Management System is based on computer technology that gives services for users, managed by the staffs who give implementation of function relatively in effective times as well as will design for removing time wasting, and saving resources, and easy data access.

1.2 Problem Statement

Working with the current system that is used by the legal firm is quite tedious, complicated and time consuming. Missing files, lost typefaces and even folders that aren't properly sorted can affect the legal firm especially when facing a deadline. In reality, something as simple as a missing linked file can put a hold on a project and disturb the whole work. Furthermore, the file is not placed and organized properly in the provided places. Therefore, it will lead to incident where a misplaced and unorganized file occur in the legal firm. Lastly, data stored on papers for the file is subject to loss due to physical damage. It may lead errors in certain operations such as searching, adding, and removing entries cannot be done efficiently.

1.3 Objectives

This project embarks on the following objectives:

1. To optimize performance through effective and efficient organization by providing an automated inventory for the file.
2. To provide a label for the file details and shelf placement.
3. To keep a track of the file in the legal firm by searching, adding and removing entries.

1.4 Scope

The proposed system project is the Legal Firm File Management System (LFMS). The system will be used in Salina Taib & Co. by the staff to manage the file in the legal firm.

There are a few modules that have been proposed. These modules provide various analysis which would help the management. The modules are:

1. Shelf
   - The modules will record the shelf profile. It will record the shelf number, the location of the shelf. The shelf number will produce an automated unique id for every shelf registration.
2. Compartment
   - The modules will record the compartment details of the shelf for every shelf registered. It will record the compartment number and the file id that has been placed in the compartment of the shelf.
3. Files
   - The module will record all files in the legal firm. It will record the file id, file name, registration date, and compartment number where the file has been assigned. The details of the file can be print.

1.5 Project Significance

Legal Firm File Management System (LFMS) for Salina Taib & Co is a system that will ease the staff who use the system since it will give an advantages to the user. On the other hand, the system will ensure the security in terms of keeping the record of files safely in the database. Thus it will keep the privacy and confidentiality of data on the legal firm. The system also equipped with manageability of data to be retrieved from the database. Furthermore, the system also enables the staff to register new files, search and remove the data easily. Lastly, it will provide an automated inventory for the file so that the files will be placed and organized properly so that there is no a missing or misplaced files.

1.6 Project Requirement

1.6.1 Software Requirement

The software requirement that have been used for this project are:

1. Eclipse IDE for Java Developers
2. JDK 1.8
3. MySQL

1.6.2 Hardware Requirement

The hardware requirement that have been used for this project are:

1. Laptop
2. Mouse
3. Printer

CHAPTER II

ANALYSIS OF PROBLEM

2.1 Problem Description

Working with the current system that is used by the legal firm is quite tedious, complicated and time consuming. Missing files, lost typefaces and even folders that aren't properly sorted can affect the legal firm especially when facing a deadline. In reality, something as simple as a missing linked file can put a hold on a project and disturb the whole work. Furthermore, the file is not placed and organized properly in the provided places. Therefore, it will lead to incident where a misplaced and unorganized file occur in the legal firm. Lastly, data stored on papers for the file is subject to loss due to physical damage. It may lead errors in certain operations such as searching, adding, and removing entries cannot be done efficiently.

2.2 Problem Decomposition

Problem 1: A manual system for the file inventory

Solution 1: Providing an automated inventory for the file.

Problem 2: File is not placed and organized properly in the provided place.

Solution 2: Label the file with the registered shelf location.

Problem 3: Law firm does not keep track of the file inventory.

Solution 3: To keep a track of the file in the legal firm by searching, adding and removing entries.

2.3 Structure Chart

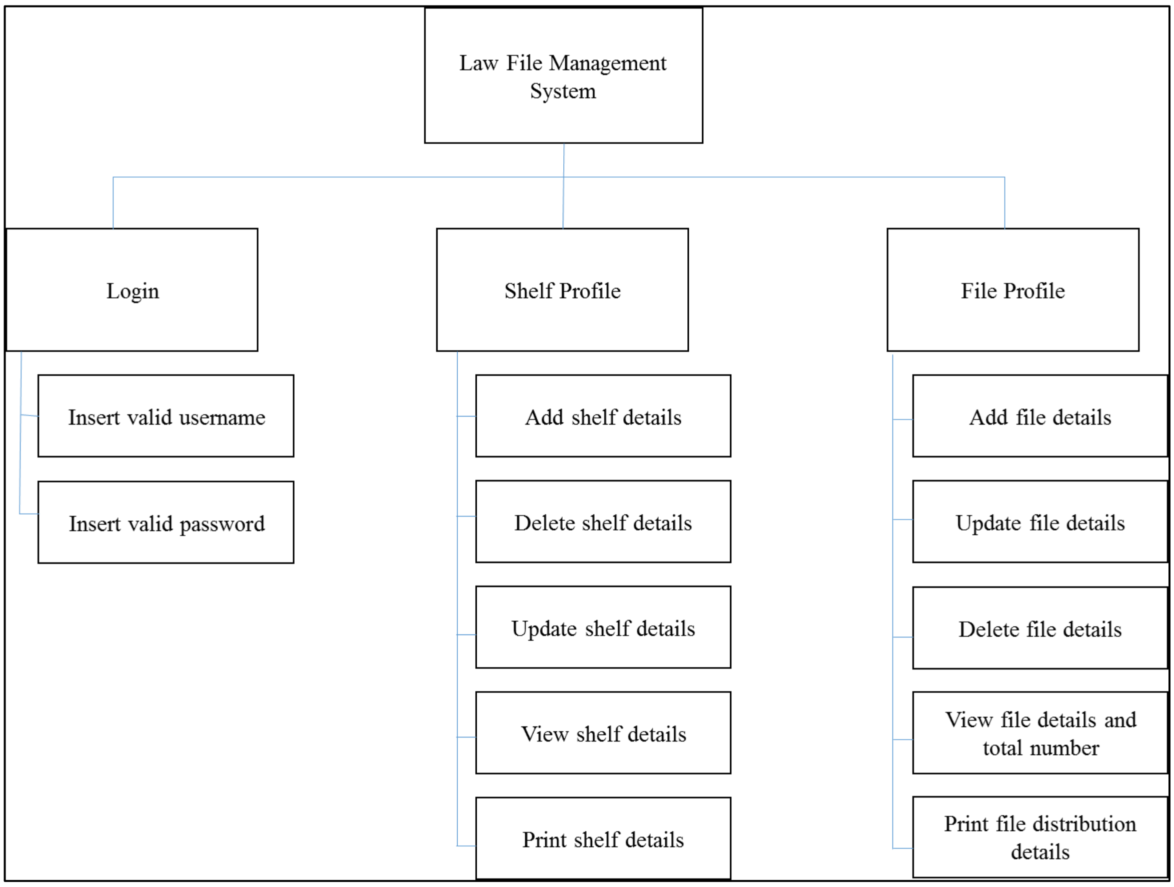


Figure 1: Structure chart of the system

CHAPTER III

DESIGN

3.1 Flow Chart

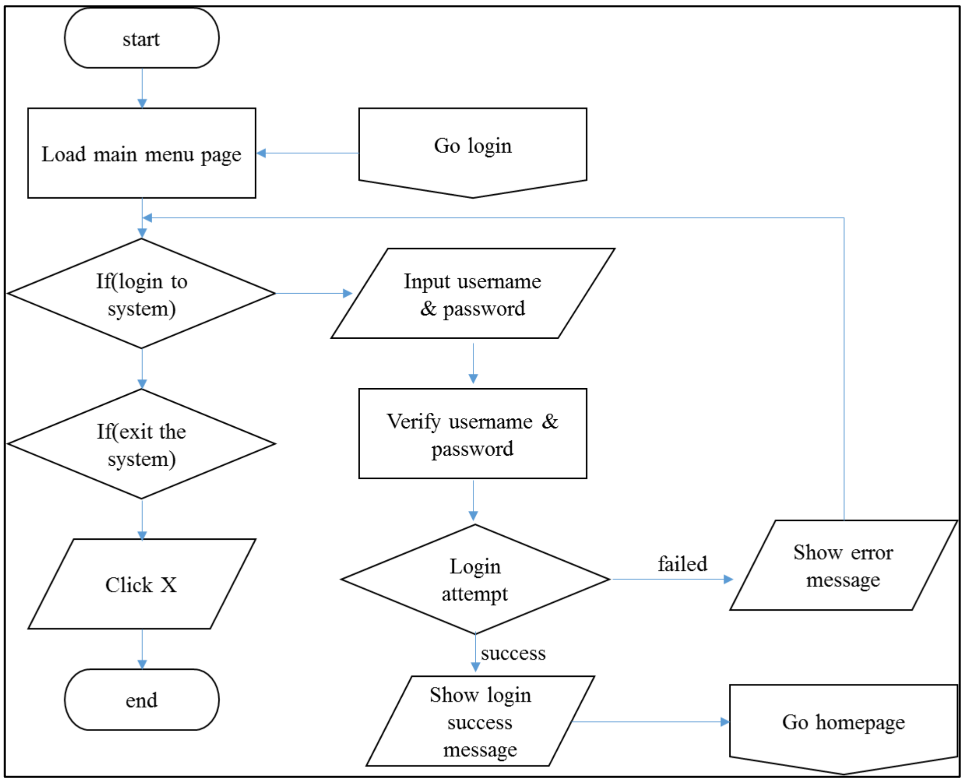


Figure 2: Login flowchart

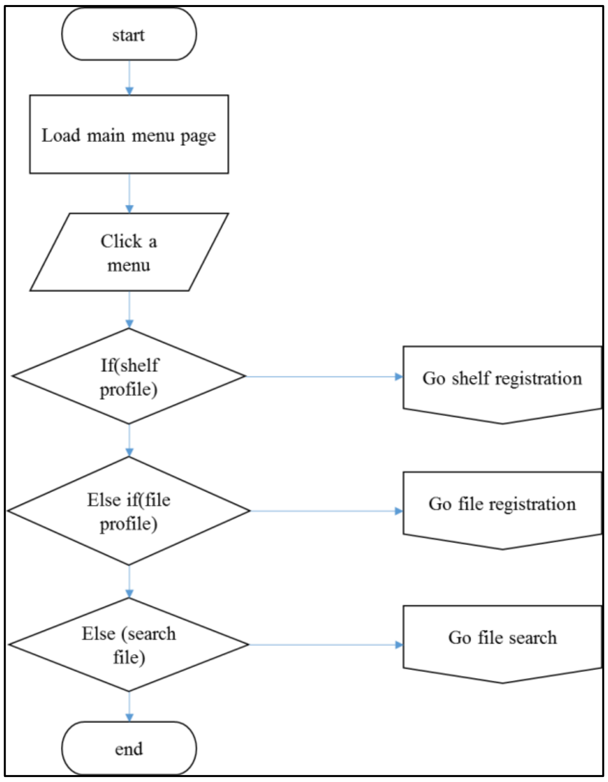


Figure 3: Homepage flowchart

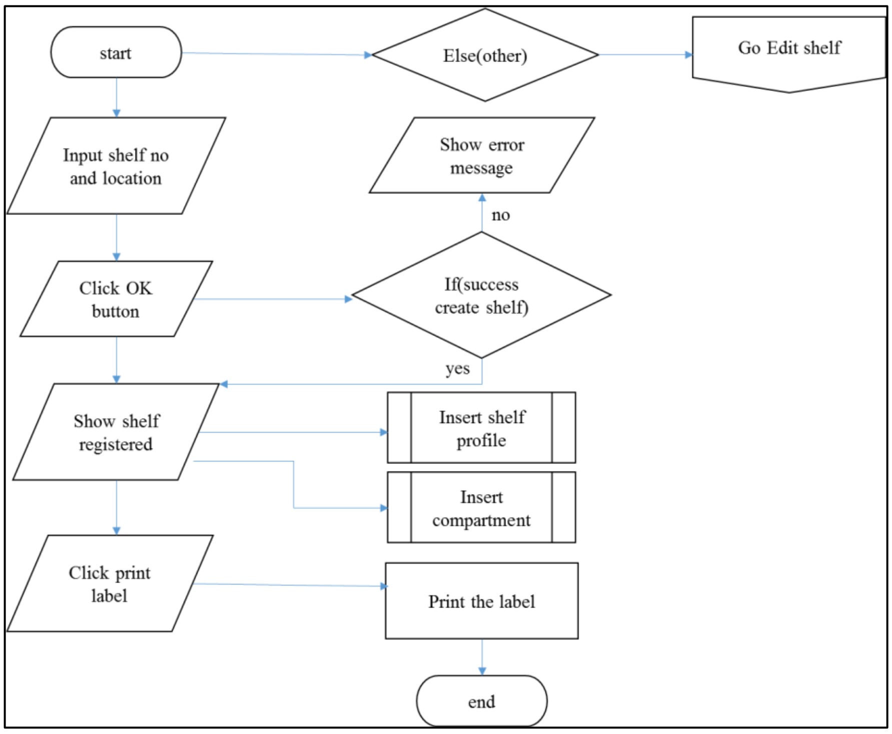


Figure 4: Shelf profile flowchart

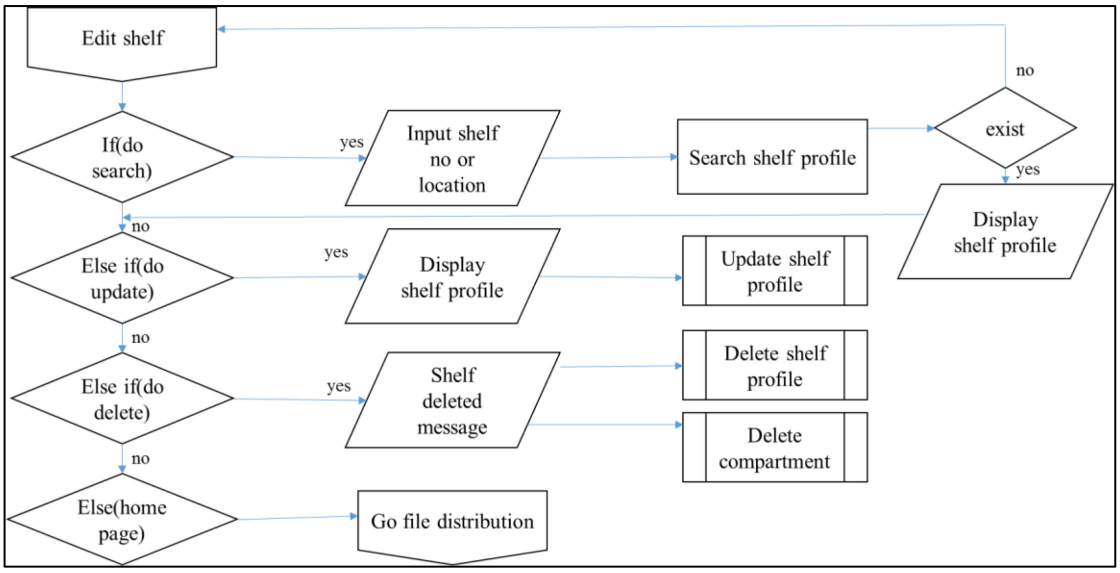


Figure 5: Edit shelf flowchart

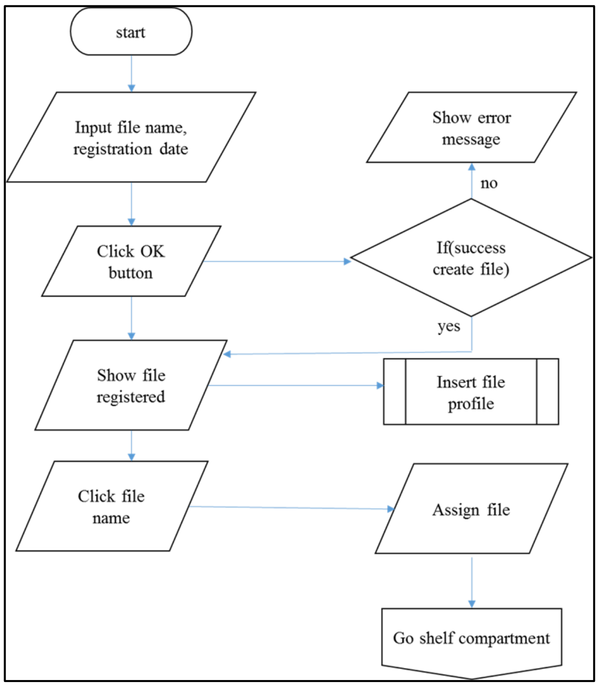


Figure 6: File profile flowchart

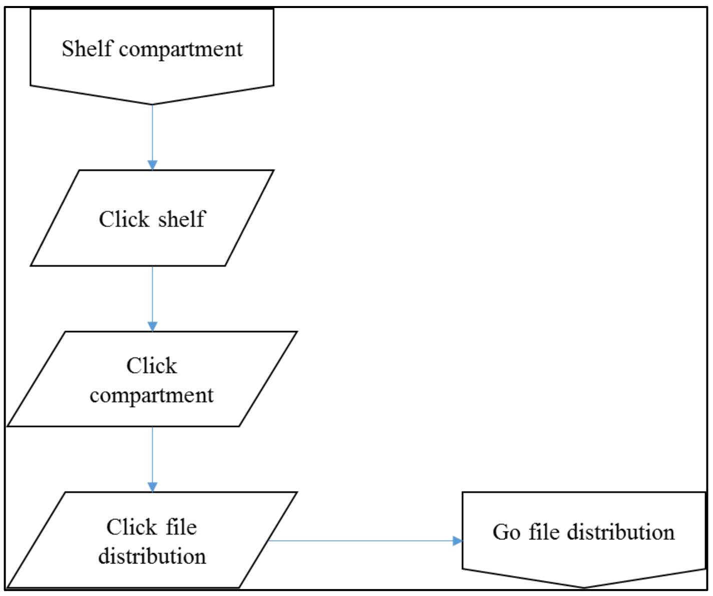


Figure 7: Shelf compartment flowchart

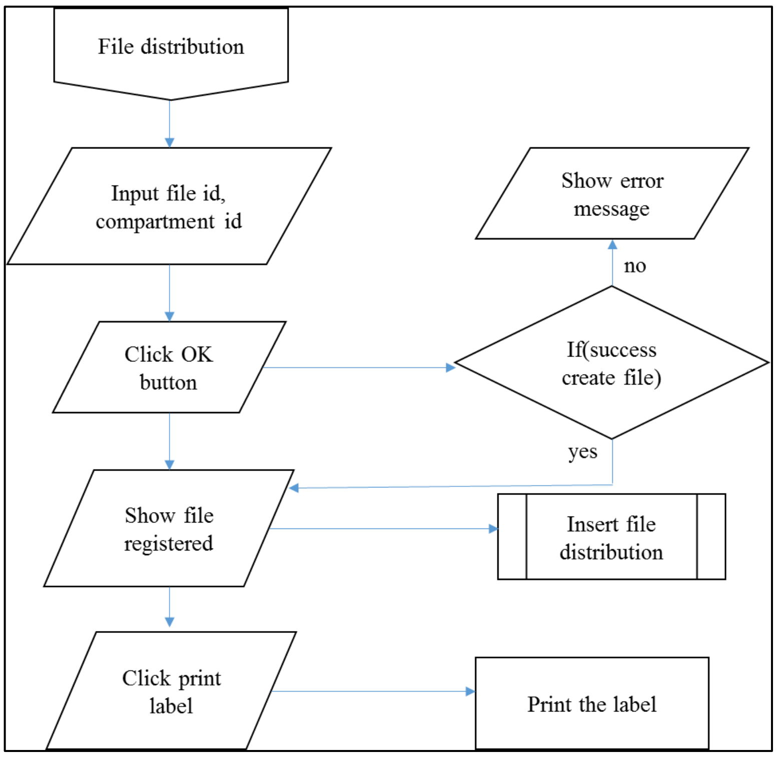


Figure 8: File distribution flowchart

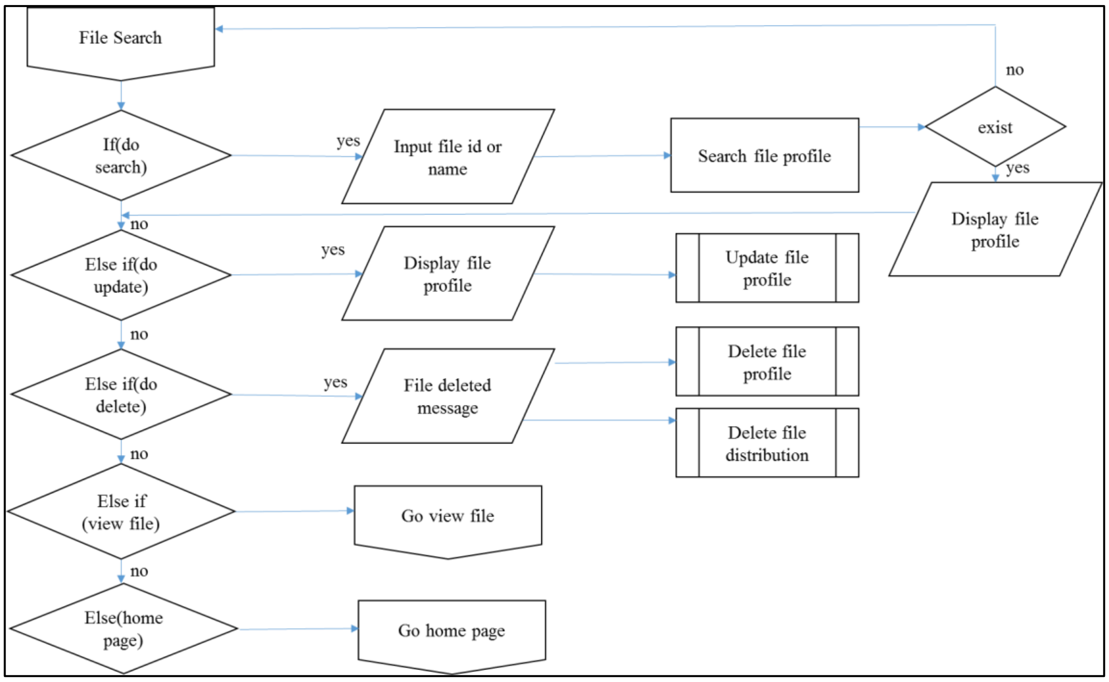


Figure 9: File search flowchart

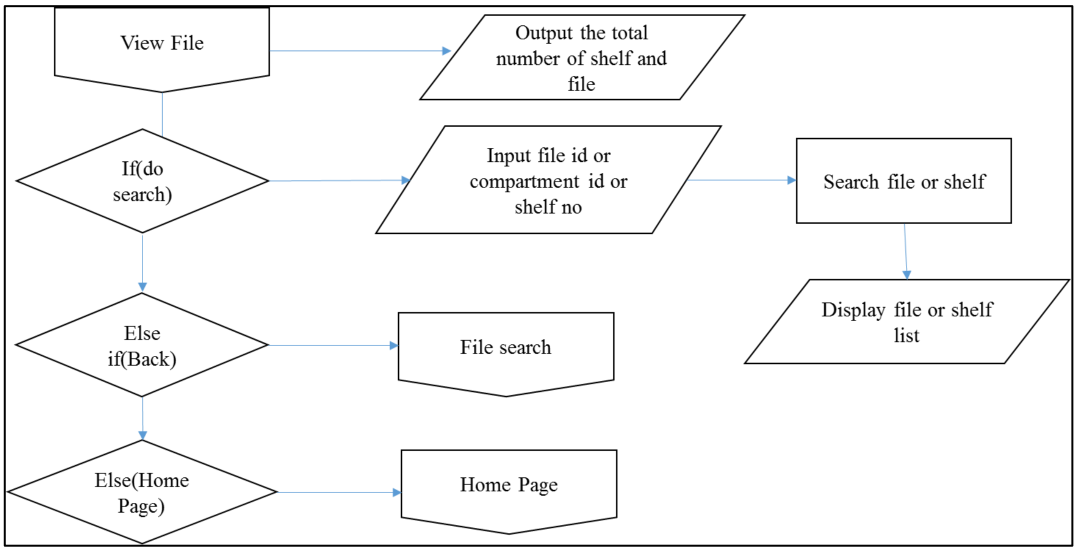

View File
Output the total number of shelf and file
If(do search)
Input file id or compartment id or shelf no
Search file or shelf
Display file or shelf list
Else if(Back)
File search
Else(Home Page)
Home Page

Figure 10: View file flowchart

3.2 Entity Relational Diagram (ERD)

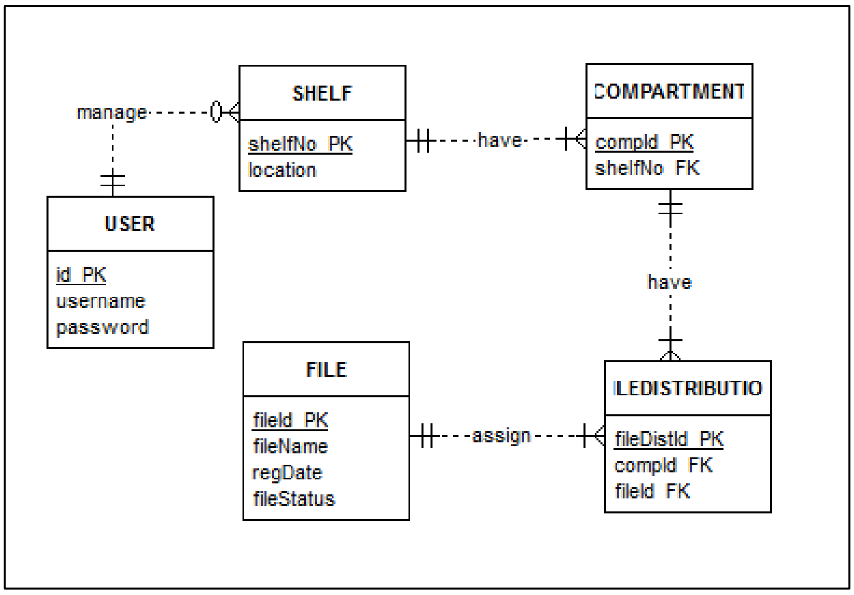


Figure 11: Entity Relational Diagram

3.3 Data Dictionary

Table User

NAME	DATA TYPE	LENGTH	CONSTRAINT	REFERENCE TABLE
id	INTEGER	11	PRIMARY KEY	
username	VARCHAR	20		
password	VARCHAR	10		

Table SHELF

NAME	DATA TYPE	LENGTH	CONSTRAINT	REFERENCE TABLE
shelfNo	VARCHAR	15	PRIMARY KEY	
location	VARCHAR	20		

Table COMPARTMENT

NAME	DATA TYPE	LENGTH	CONSTRAINT	REFERENCE TABLE
compId	VARCHAR	15	PRIMARY KEY	
shelfNo	VARCHAR	20	FOREIGN KEY	SHELF

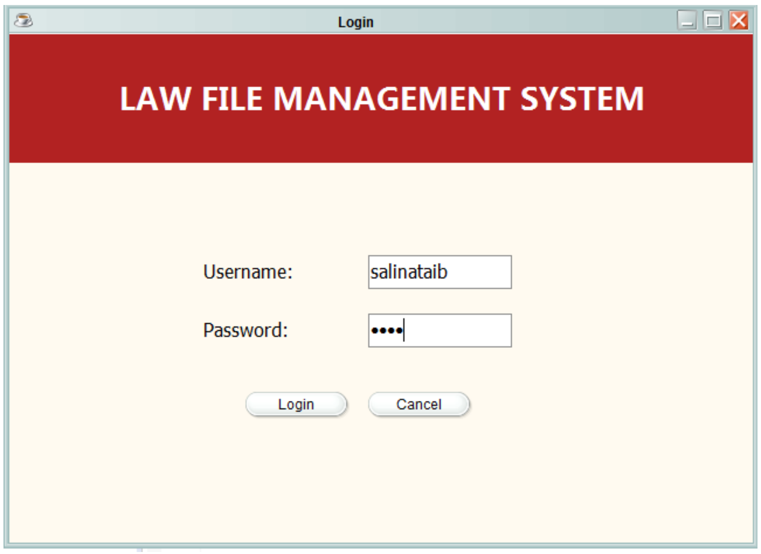
Table FILE

NAME	DATA TYPE	LENGTH	CONSTRAINT	REFERENCE TABLE
fileId	VARCHAR	15	PRIMARY KEY	
fileName	VARCHAR	5		
regDate	DATE			
fileStatus	INTEGER	4		

Table FILE DISTRIBUTION

NAME	DATA TYPE	LENGTH	CONSTRAINT	REFERENCE TABLE
fileDistId	VARCHAR	15	PRIMARY KEY	
compId	VARCHAR	15	FOREIGN KEY	COMPARTMENT
fileId	VARCHAR	15	FOREIGN KEY	FILE

3.4 Interface Design

Interface Name: Login

Login
LAW FILE MANAGEMENT SYSTEM
Username: salinataib
Password: ••••
Figure 12: Login Page

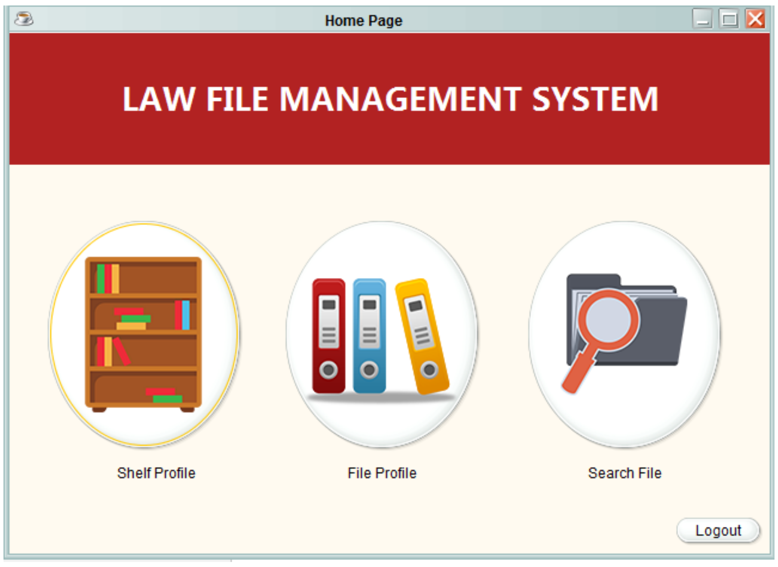
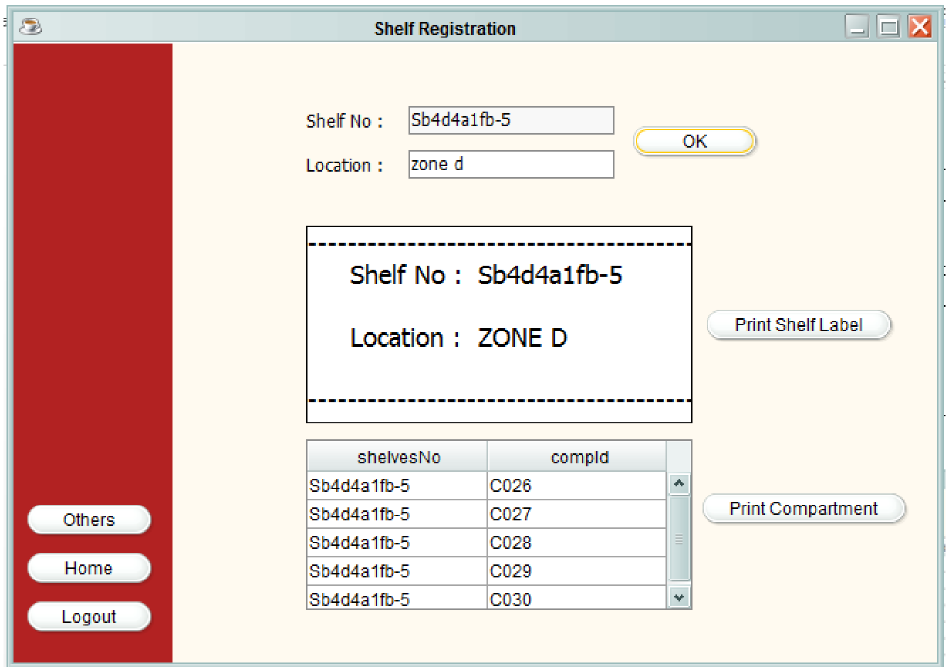



Figure 13: Home Page

Interface Name: Shelf Profile



Shelf No : Sb4d4a1fb-5

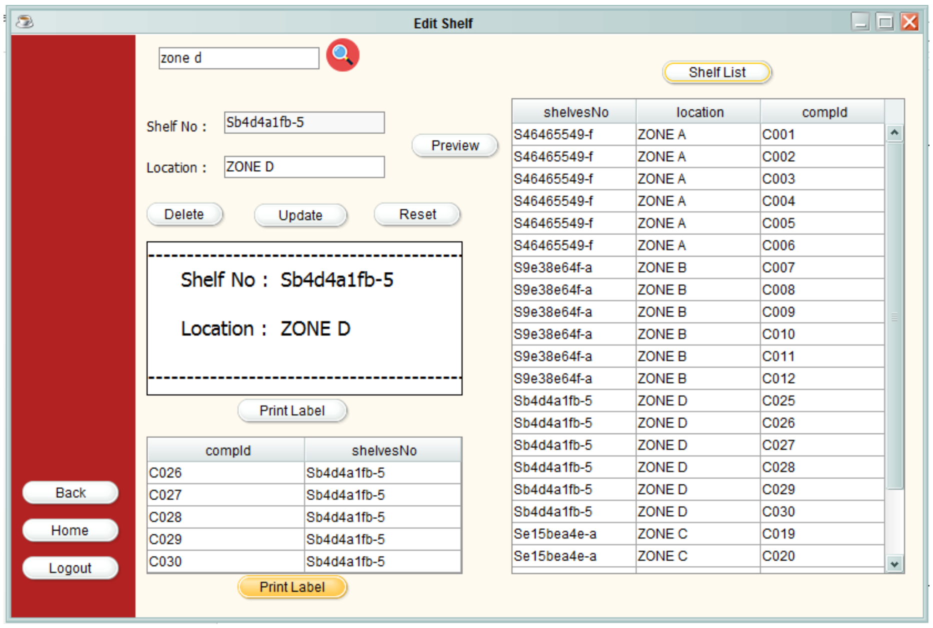
Location : zone d

Shelf No : Sb4d4a1fb-5

Location : ZONE D

shelvesNo	compId
Sb4d4a1fb-5	C026
Sb4d4a1fb-5	C027
Sb4d4a1fb-5	C028
Sb4d4a1fb-5	C029
Sb4d4a1fb-5	C030

Figure 14: Shelf Registration



zone d

Shelf No : Sb4d4a1fb-5

Location : ZONE D

Shelf No : Sb4d4a1fb-5

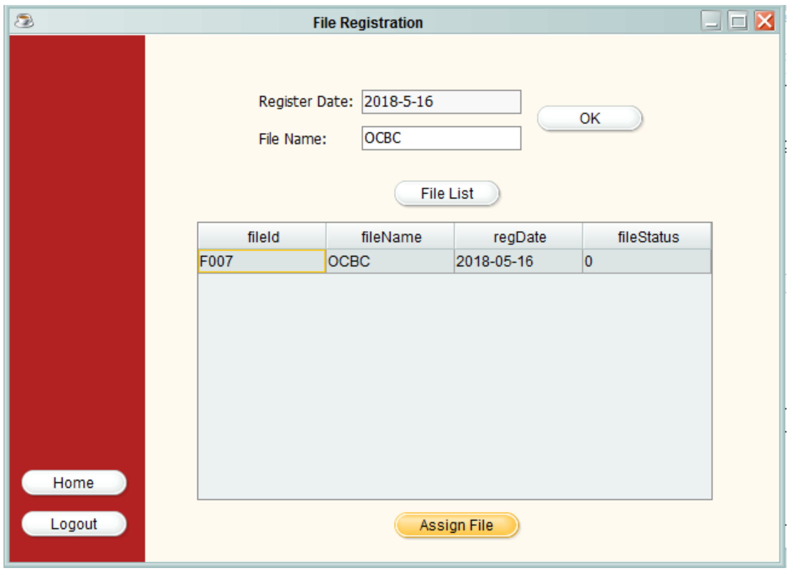
Location : ZONE D

compId	shelvesNo
C026	Sb4d4a1fb-5
C027	Sb4d4a1fb-5
C028	Sb4d4a1fb-5
C029	Sb4d4a1fb-5
C030	Sb4d4a1fb-5

shelvesNo	location	compId
S46465549-f	ZONE A	C001
S46465549-f	ZONE A	C002
S46465549-f	ZONE A	C003
S46465549-f	ZONE A	C004
S46465549-f	ZONE A	C005
S46465549-f	ZONE A	C006
S9e38e64f-a	ZONE B	C007
S9e38e64f-a	ZONE B	C008
S9e38e64f-a	ZONE B	C009
S9e38e64f-a	ZONE B	C010
S9e38e64f-a	ZONE B	C011
S9e38e64f-a	ZONE B	C012
Sb4d4a1fb-5	ZONE D	C025
Sb4d4a1fb-5	ZONE D	C026
Sb4d4a1fb-5	ZONE D	C027
Sb4d4a1fb-5	ZONE D	C028
Sb4d4a1fb-5	ZONE D	C029
Sb4d4a1fb-5	ZONE D	C030
Se15bea4e-a	ZONE C	C019
Se15bea4e-a	ZONE C	C020

Figure 15: Edit Shelf

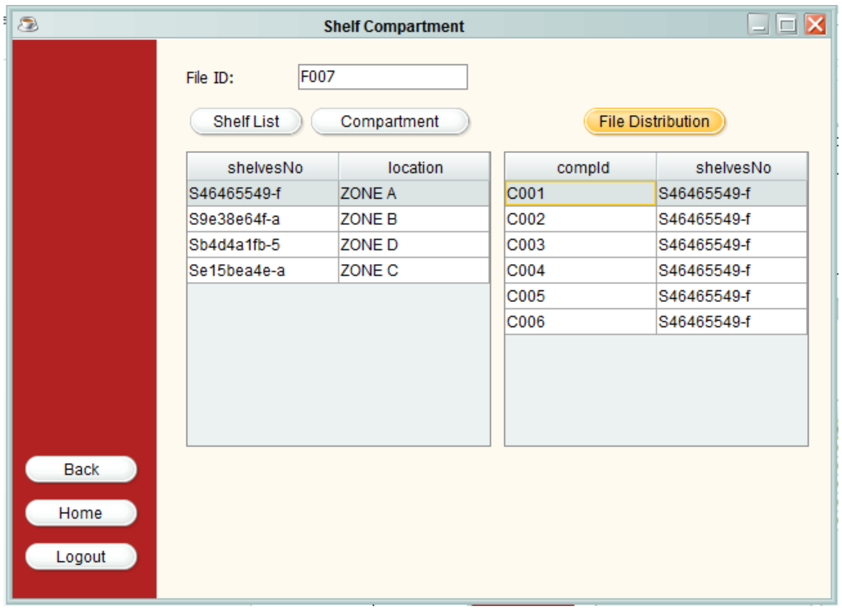
Interface Name: File Profile



Register Date: 2018-5-16
File Name: OCBC

fileId	fileName	regDate	fileStatus
F007	OCBC	2018-05-16	0

Figure 16: File Registration

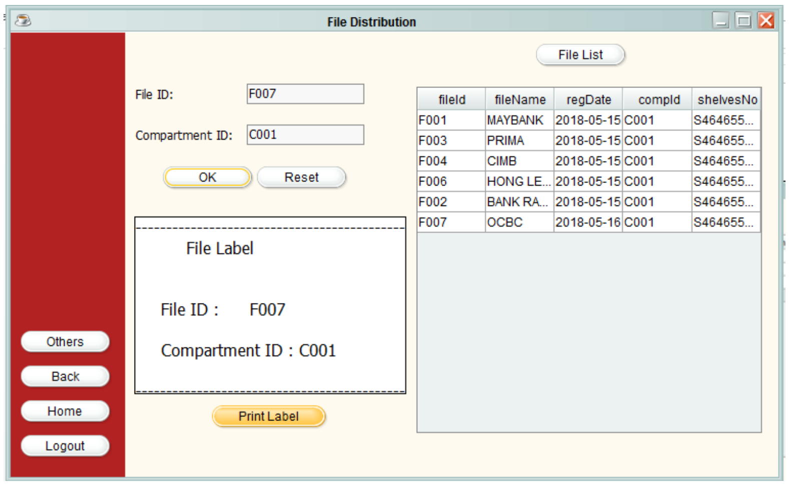


File ID: F007

shelvesNo	location
S46465549-f	ZONE A
S9e38e64f-a	ZONE B
Sb4d4a1fb-5	ZONE D
Se15bea4e-a	ZONE C

compId	shelvesNo
C001	S46465549-f
C002	S46465549-f
C003	S46465549-f
C004	S46465549-f
C005	S46465549-f
C006	S46465549-f

Figure 17: Shelf Compartment



File ID: F007

Compartment ID: C001

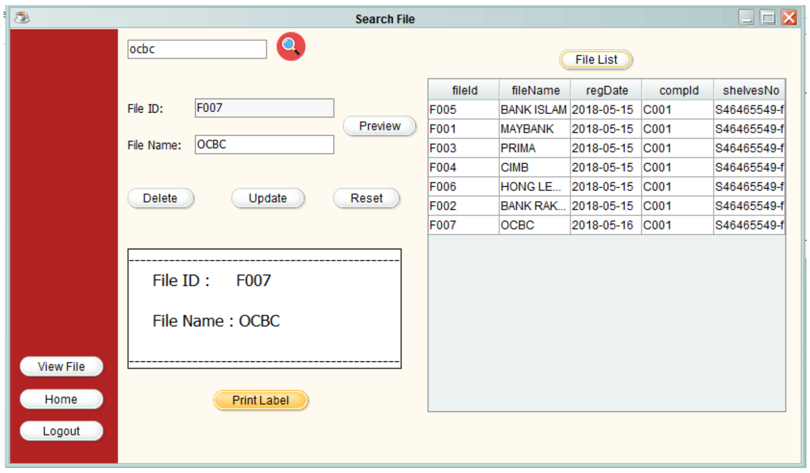
File Label

File ID : F007

Compartment ID : C001

fileId	fileName	regDate	compId	shelvesNo
F001	MAYBANK	2018-05-15	C001	S464655...
F003	PRIMA	2018-05-15	C001	S464655...
F004	CIMB	2018-05-15	C001	S464655...
F006	HONG LE...	2018-05-15	C001	S464655...
F002	BANK RA...	2018-05-15	C001	S464655...
F007	OCBC	2018-05-16	C001	S464655...

Figure 18: File Distribution



ocbc

File ID: F007

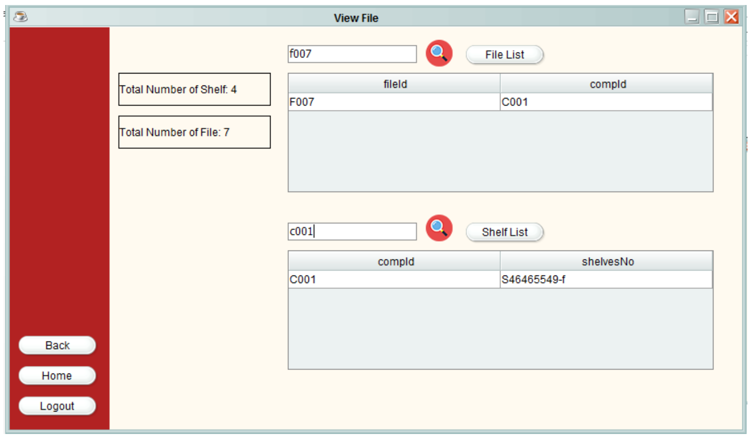
File Name: OCBC

File ID : F007

File Name : OCBC

fileId	fileName	regDate	compId	shelvesNo
F005	BANK ISLAM	2018-05-15	C001	S46465549-f
F001	MAYBANK	2018-05-15	C001	S46465549-f
F003	PRIMA	2018-05-15	C001	S46465549-f
F004	CIMB	2018-05-15	C001	S46465549-f
F006	HONG LE...	2018-05-15	C001	S46465549-f
F002	BANK RAK...	2018-05-15	C001	S46465549-f
F007	OCBC	2018-05-16	C001	S46465549-f

Figure 19: Search File



Total Number of Shelf: 4

Total Number of File: 7

f007

fileId	compId
F007	C001

c001

compId	shelvesNo
C001	S46465549-f

Figure 20: View File

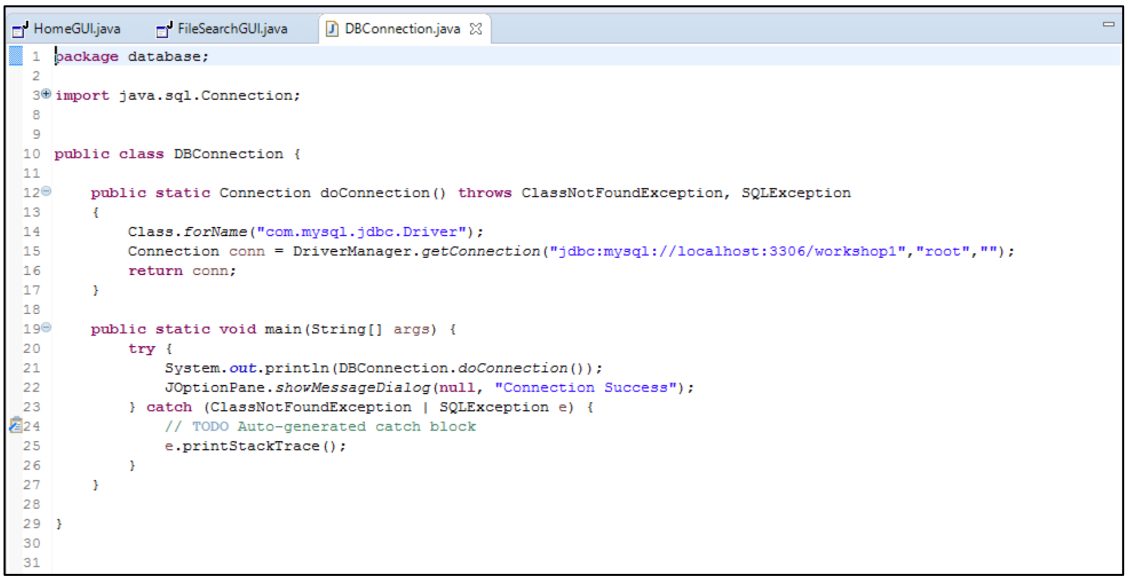
CHAPTER IV

IMPLEMENTATION

4.1 Storing, Retrieve & Manage Data

For data storing and data retrieving into database connection to the MySQL server by importing a library call mysql_connector_java.jar must be done. Then creating a class for the database connection.

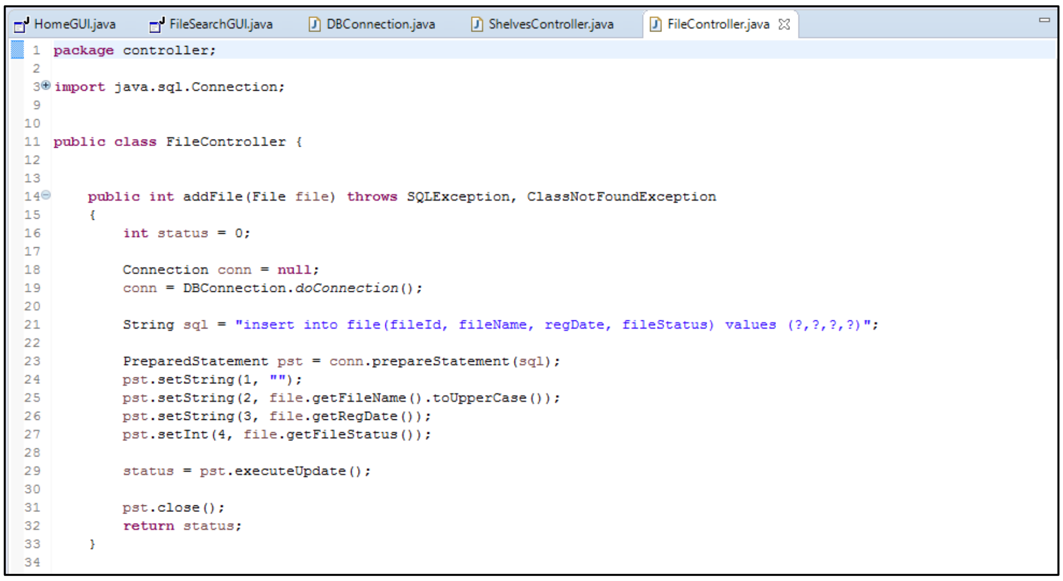
Code snippet:



```java
package database;

import java.sql.Connection;


public class DBConnection {

    public static Connection doConnection() throws ClassNotFoundException, SQLException
    {
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/workshop1","root","");
        return conn;
    }

    public static void main(String[] args) {
        try {
            System.out.println(DBConnection.doConnection());
            JOptionPane.showMessageDialog(null, "Connection Success");
        } catch (ClassNotFoundException | SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

}
```

Storing of data system need to get all the required field and set it into SQL statement and execute the SQL statement to store the data into database.

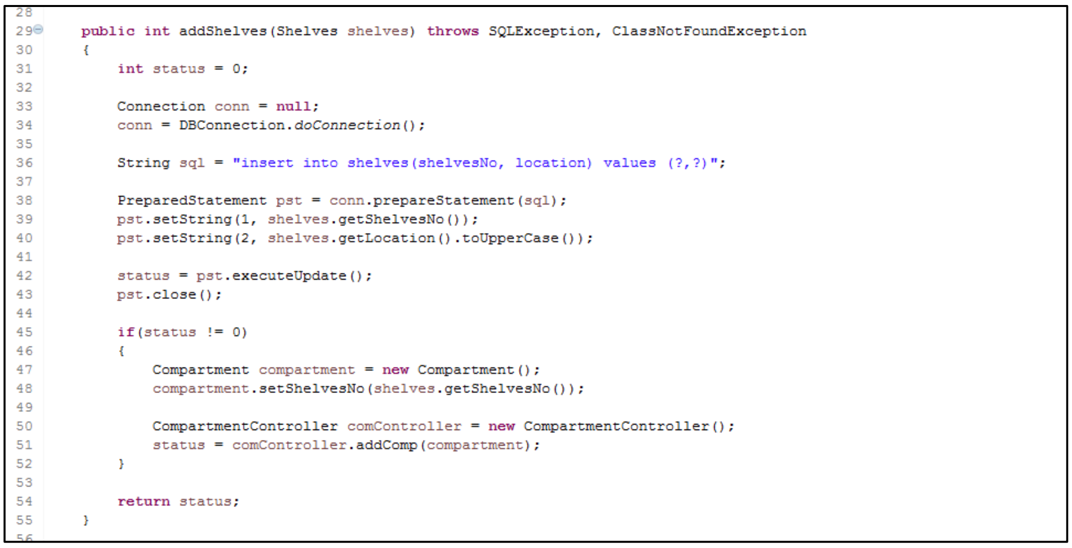
Code snippet:



```java
package controller;

import java.sql.Connection;


public class FileController {


    public int addFile(File file) throws SQLException, ClassNotFoundException
    {
        int status = 0;

        Connection conn = null;
        conn = DBConnection.doConnection();

        String sql = "insert into file(fileId, fileName, regDate, fileStatus) values (?,?,?,?)";

        PreparedStatement pst = conn.prepareStatement(sql);
        pst.setString(1, "");
        pst.setString(2, file.getFileName().toUpperCase());
        pst.setString(3, file.getRegDate());
        pst.setInt(4, file.getFileStatus());

        status = pst.executeUpdate();

        pst.close();
        return status;
    }

```

The code snippet below show the compartment id is automatically inserted when the user register the shelf. The shelf is registered with 6 compartment id.

Code snippet:

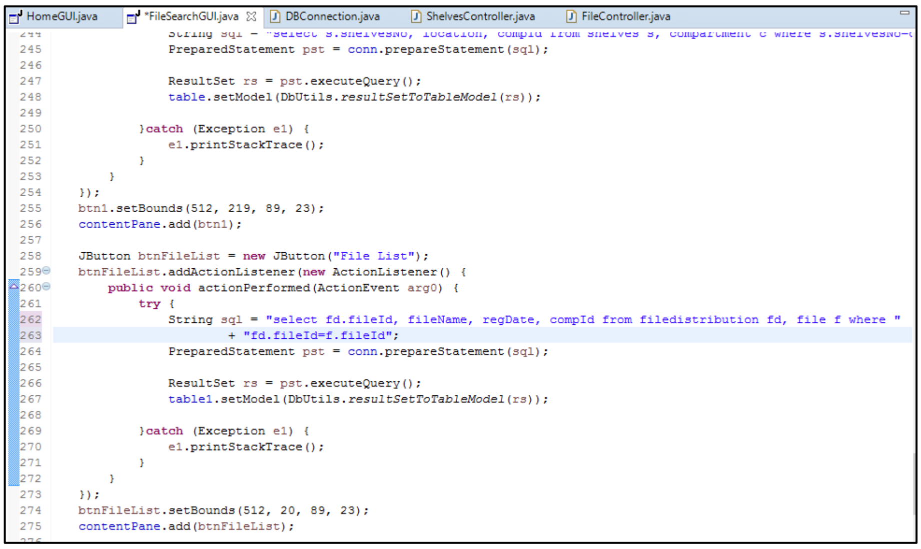


```java

    public int addShelves(Shelves shelves) throws SQLException, ClassNotFoundException
    {
        int status = 0;

        Connection conn = null;
        conn = DBConnection.doConnection();

        String sql = "insert into shelves(shelvesNo, location) values (?,?)";

        PreparedStatement pst = conn.prepareStatement(sql);
        pst.setString(1, shelves.getShelvesNo());
        pst.setString(2, shelves.getLocation().toUpperCase());

        status = pst.executeUpdate();
        pst.close();

        if(status != 0)
        {
            Compartment compartment = new Compartment();
            compartment.setShelvesNo(shelves.getShelvesNo());

            CompartmentController comController = new CompartmentController();
            status = comController.addComp(compartment);
        }

        return status;
    }
```

For data retrieving, the SQL statement is created and executed to get the data from the database and manage the data. There are three ways of retrieving the data. First the data is retrieve from the database. Next, the data retrieved from the user input and tabulate into a table list. Lastly, the data retrieved from the user input and set into the text field.

1. The data is retrieve from the database.

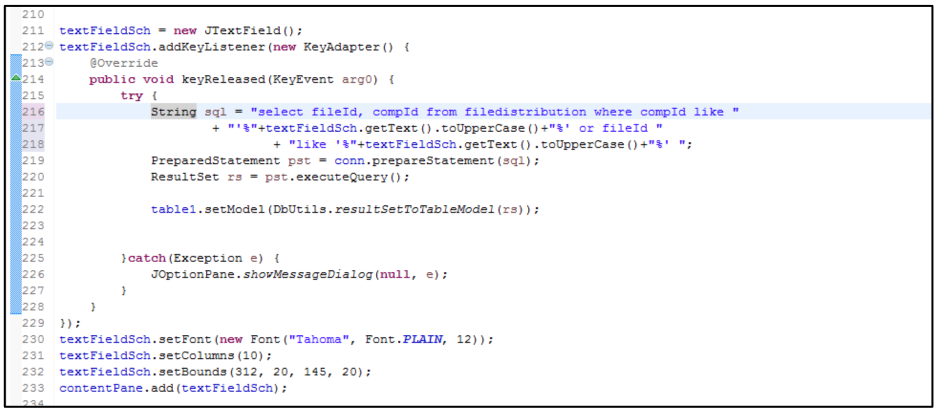
   Code snippet:



```
                String sql = "select s.shelvesNo, location, compId from shelves s, compartment c where s.shelvesNo=
                PreparedStatement pst = conn.prepareStatement(sql);

                ResultSet rs = pst.executeQuery();
                table.setModel(DbUtils.resultSetToTableModel(rs));

            }catch (Exception e1) {
                e1.printStackTrace();
            }
        }
    });
    btn1.setBounds(512, 219, 89, 23);
    contentPane.add(btn1);

    JButton btnFileList = new JButton("File List");
    btnFileList.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            try {
                String sql = "select fd.fileId, fileName, regDate, compId from filedistribution fd, file f where "
                        + "fd.fileId=f.fileId";
                PreparedStatement pst = conn.prepareStatement(sql);

                ResultSet rs = pst.executeQuery();
                table1.setModel(DbUtils.resultSetToTableModel(rs));

            }catch (Exception e1) {
                e1.printStackTrace();
            }
        }
    });
    btnFileList.setBounds(512, 20, 89, 23);
    contentPane.add(btnFileList);
```

2. The data retrieved from the user input and tabulate into a table list.

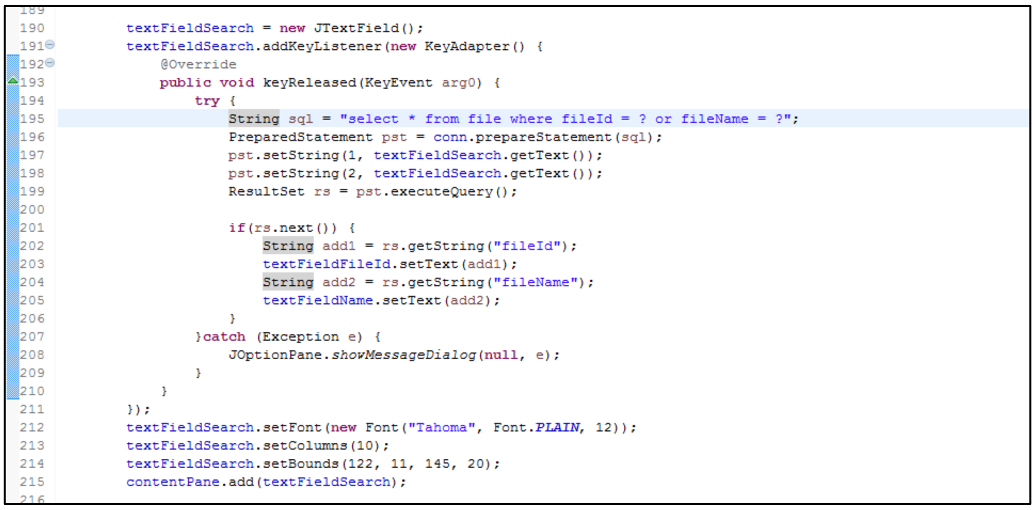
   Code snippet:



```

textFieldSch = new JTextField();
textFieldSch.addKeyListener(new KeyAdapter() {
    @Override
    public void keyReleased(KeyEvent arg0) {
        try {
            String sql = "select fileId, compId from filedistribution where compId like "
                    + "'%"+textFieldSch.getText().toUpperCase()+"%' or fileId "
                            + "like '%"+textFieldSch.getText().toUpperCase()+"%' ";
            PreparedStatement pst = conn.prepareStatement(sql);
            ResultSet rs = pst.executeQuery();

            table1.setModel(DbUtils.resultSetToTableModel(rs));


        }catch(Exception e) {
            JOptionPane.showMessageDialog(null, e);
        }
    }
});
textFieldSch.setFont(new Font("Tahoma", Font.PLAIN, 12));
textFieldSch.setColumns(10);
textFieldSch.setBounds(312, 20, 145, 20);
contentPane.add(textFieldSch);
```

3. The data retrieved from the user input and set into the text field.

   Code snippet:

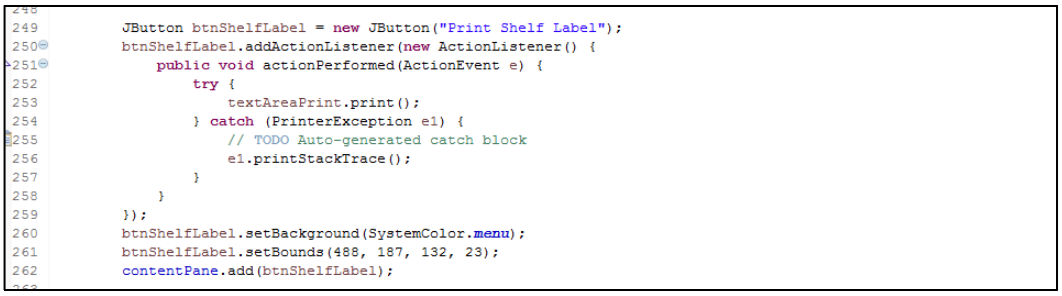


```
        textFieldSearch = new JTextField();
        textFieldSearch.addKeyListener(new KeyAdapter() {
            @Override
            public void keyReleased(KeyEvent arg0) {
                try {
                    String sql = "select * from file where fileId = ? or fileName = ?";
                    PreparedStatement pst = conn.prepareStatement(sql);
                    pst.setString(1, textFieldSearch.getText());
                    pst.setString(2, textFieldSearch.getText());
                    ResultSet rs = pst.executeQuery();

                    if(rs.next()) {
                        String add1 = rs.getString("fileId");
                        textFieldFileId.setText(add1);
                        String add2 = rs.getString("fileName");
                        textFieldName.setText(add2);
                    }
                }catch (Exception e) {
                    JOptionPane.showMessageDialog(null, e);
                }
            }
        });
        textFieldSearch.setFont(new Font("Tahoma", Font.PLAIN, 12));
        textFieldSearch.setColumns(10);
        textFieldSearch.setBounds(122, 11, 145, 20);
        contentPane.add(textFieldSearch);
```

4.2 Printing

The idea of printing in this project is to label all shelf and file in the office to prevent any missing of file and ease the staff to manage the file. The printing label method will print the label from the text area after registration complete.

Code snippet:

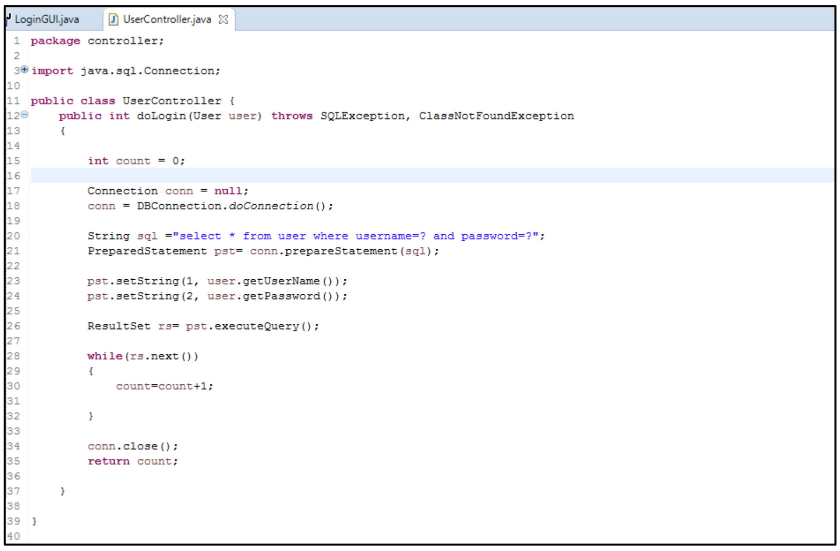


```
        JButton btnShelfLabel = new JButton("Print Shelf Label");
        btnShelfLabel.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    textAreaPrint.print();
                } catch (PrinterException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
            }
        });
        btnShelfLabel.setBackground(SystemColor.menu);
        btnShelfLabel.setBounds(488, 187, 132, 23);
        contentPane.add(btnShelfLabel);
```

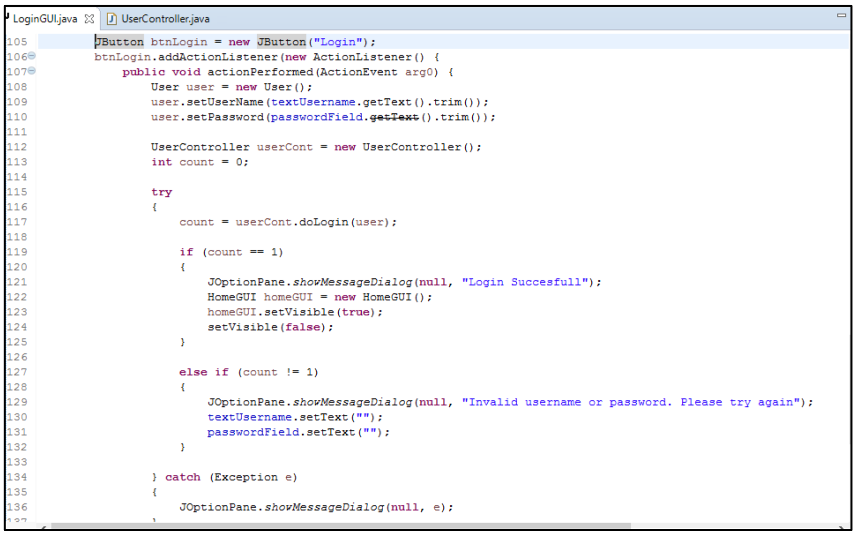
4.3 Security

Security in this system is very important. The username and password will be set in the database and the user will using the default password. The username and password are required to login into the system. If the username or password does not match with the database, the user cannot login to the system.

Code snippet:



```java
package controller;

import java.sql.Connection;

public class UserController {
    public int doLogin(User user) throws SQLException, ClassNotFoundException
    {

        int count = 0;

        Connection conn = null;
        conn = DBConnection.doConnection();

        String sql ="select * from user where username=? and password=?";
        PreparedStatement pst= conn.prepareStatement(sql);

        pst.setString(1, user.getUserName());
        pst.setString(2, user.getPassword());

        ResultSet rs= pst.executeQuery();

        while(rs.next())
        {
            count=count+1;

        }

        conn.close();
        return count;

    }

}
```



```java
        JButton btnLogin = new JButton("Login");
        btnLogin.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
                User user = new User();
                user.setUserName(textUsername.getText().trim());
                user.setPassword(passwordField.~~getText~~().trim());

                UserController userCont = new UserController();
                int count = 0;

                try
                {
                    count = userCont.doLogin(user);

                    if (count == 1)
                    {
                        JOptionPane.showMessageDialog(null, "Login Succesfull");
                        HomeGUI homeGUI = new HomeGUI();
                        homeGUI.setVisible(true);
                        setVisible(false);
                    }

                    else if (count != 1)
                    {
                        JOptionPane.showMessageDialog(null, "Invalid username or password. Please try again");
                        textUsername.setText("");
                        passwordField.setText("");
                    }

                } catch (Exception e)
                {
                    JOptionPane.showMessageDialog(null, e);
```

4.4 SQL Statement

There are many uses of SQL statement in this project to retrieve data and store data. The SQL statement than have been used in this project are selection, aggregation and join statement.

1. **To search file from the system**

   SELECT * FROM file WHERE fileId = ? OR fileName = ?;

2. **To add a new file into the system**

```
INSERT INTO file(fileId, fileName, regDate, fileStatus) VALUES (?,?,?,?);
INSERT INTO filedistribution(fileDistID, compId, fileId) VALUES (?,?,?)";
UPDATE file SET fileStatus = 1 WHERE fileId = '"+filedistribution.getFileId()+"';
```

3. **To update a file from the system**

```
UPDATE file SET fileName = '"+file.getFileName().toUpperCase()+"' WHERE fileId = '"+file.getFileId()+"';
```

4. **To delete a file from the system**

```
DELETE FROM file WHERE fileId = '"+file.getFileId()+"';
DELETE FROM filedistribution WHERE fileId = '"+file.getFileId()+"';
```

5. **To get a total number of file**

```
SELECT COUNT(fileId) AS file FROM filedistribution;
```

6. **To get the list of file and the location**

```
SELECT f.fileId, fileName, regDate, c.compId, shelvesNo FROM file f, filedistribution fd, compartment c WHERE f.fileId=fd.fileId AND fd.compId=c.compId;
```

7. **To search shelf from the system**

```
SELECT * FROM compartment
WHERE compId LIKE '%"+textFieldSearch.getText().toUpperCase()+"%'
OR shelvesNo LIKE '%"+textFieldSearch.getText()+"%';
```

8. **To add a new shelf from the system**

```
INSERT INTO shelves (shelvesNo, location) VALUES (?,?);
```

9. **To delete a shelf from the system**

```
DELETE FROM shelves WHERE shelvesNo = '"+shelves.getShelvesNo()+"';
DELETE FROM compartment
WHERE shelvesNo = '"+shelves.getShelvesNo()+"';
```

10. To get a total number of shelf

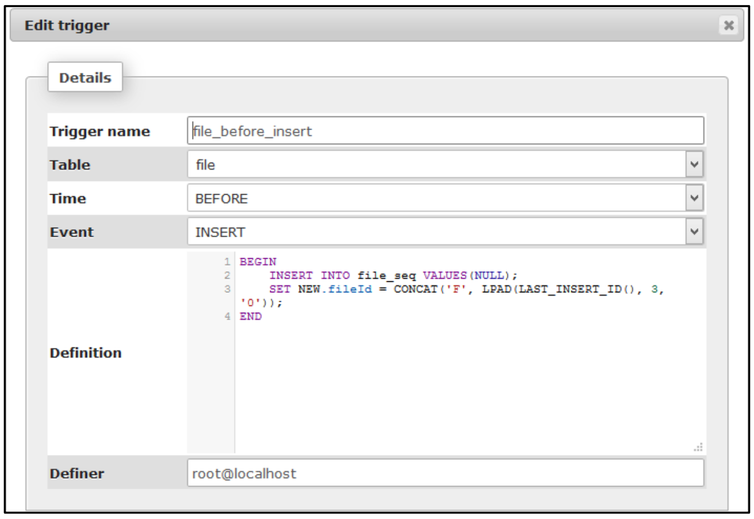
SELECT COUNT (shelvesNo) AS Shelf FROM shelves;

11. To get the list of shelf location

SELECT s.shelvesNo, location, compId FROM shelves s, compartment c WHERE s.shelvesNo=c.shelvesNo;

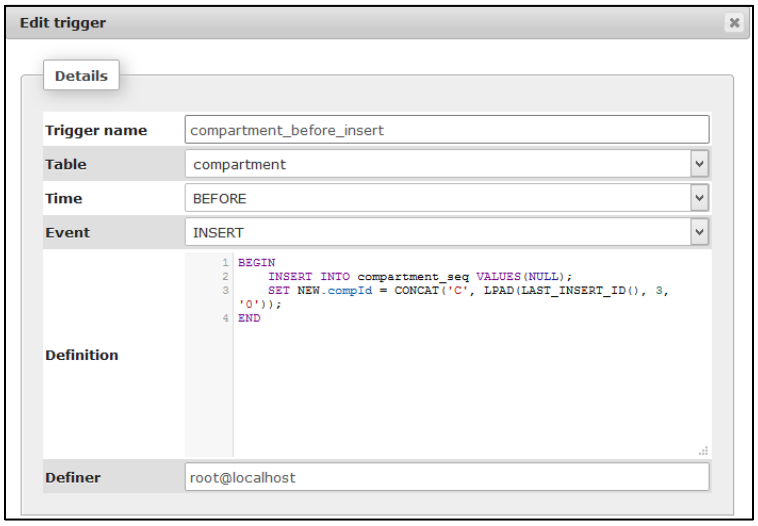
4.5 Trigger

A trigger is a special type of stored procedure that automatically executes when an event occurs in the database server. The trigger statement is used for generating the compartment id, file id and file distribution id.



Trigger name	file_before_insert
Table	file
Time	BEFORE
Event	INSERT
Definition	
Definer	root@localhost

```
BEGIN
    INSERT INTO file_seq VALUES(NULL);
    SET NEW.fileId = CONCAT('F', LPAD(LAST_INSERT_ID(), 3,
'0'));
END
```



Trigger name	compartment_before_insert
Table	compartment
Time	BEFORE
Event	INSERT
Definition	
Definer	root@localhost

```
BEGIN
    INSERT INTO compartment_seq VALUES(NULL);
    SET NEW.compId = CONCAT('C', LPAD(LAST_INSERT_ID(), 3,
'0'));
END
```

CHAPTER V

CONCLUSION

5.1 Conclusion

As a conclusion, this system has been successfully developed and have met the requirements mentioned at the earlier stage of the system. The system has succeeds in achieving its objectives. The system has successfully achieve the first objectives by providing an automated inventory for the file system. The system also succeeds in providing the label for the file details and for the shelf placement. The last objectives of the system also has been achieve where the user can make a searching, adding, updating and deleting file in the boutique for keeping a track of the file management and the placement.

However, there are still a few weakness that need to be improved in the future. The improvement makes the system better and more comprehensive. Nevertheless, as long as this system has achieved the entire objectives, this implies that the purpose for this project has been reached and will be helpful to user.

5.2 Limitation

The limitation of the system are:

1. There is no notification to user if the shelf is fully used.
2. The recorded data will be displayed in one screen and will trouble the user to scroll down and view.
3. The system does not have a security on the database which is there is no backup and recovery if the database crashed.

5.3 Future Improvement

There are some suggestion on how to improve the system. System development is an extremely element process, which requires the developer to reliably check the system to guarantee that it is running smoothly. Some future upgrades that developer would like to consider are:

1. Improve in printing job.
2. Notify user by showing a pop up message state, if the shelf is fully used.
3. Improve the graphic of the user interface.
4. Implement password security by using encryption, to avoid intrusions. Password stored in the database must be in a condition that is safe and only know by the user itself.