**NEWSPAPER AGENCY**

**APPLICATION DEVELOPMENT – III**

**IMPLEMENTATION PHASE**

*Submitted in partial fulfilment for the award of the degree*

*of*

**Master of Science**

***in***

**Information Technology**

*by*

**SANKET SURESH PETHKAR**

**REG NO: 14MIN2879**



**School of Information Technology and Engineering**

March 2018

Table of Contents

[1. INTRODUCTION 7](#_Toc458785225)

[2. SOFTWARE REQUIREMENT SPECIFICATION 8](#_Toc458785226)

[2.1.1. Login to the Application 8](#_Toc458785227)

[2.1.1.1. Introduction - 8](#_Toc458785228)

[2.1.1.2. Processing– 8](#_Toc458785229)

[2.1.1.3. Outputs– 9](#_Toc458785230)

[2.1.1.4. Error Handling- 9](#_Toc458785231)

[2.1.2. User Request for Newspaper 9](#_Toc458785232)

[2.1.2.1. Introduction – 9](#_Toc458785233)

[2.1.2.2. Inputs – 9](#_Toc458785234)

[2.1.2.3. Processing– 9](#_Toc458785235)

[2.1.2.4. Outputs– 9](#_Toc458785236)

[2.1.2.5. Error Handling- 9](#_Toc458785237)

[2.1.3. UserTrack Request 10](#_Toc458785238)

[2.1.3.1. Introduction – 10](#_Toc458785239)

[2.1.3.2. Inputs – 10](#_Toc458785240)

[2.1.3.3. Processing– 10](#_Toc458785241)

[2.1.3.4. Outputs– 10](#_Toc458785242)

[2.1.3.5. Handling- 10](#_Toc458785243)

[2.1.4. Admin view stock 10](#_Toc458785244)

[2.1.4.1. Introduction – 10](#_Toc458785245)

[2.1.4.2. Inputs – 10](#_Toc458785246)

[2.1.4.3. Processing– 10](#_Toc458785247)

[2.1.4.4. Outputs– 11](#_Toc458785248)

[2.1.4.5. Error Handling- 11](#_Toc458785249)

[2.1.5. Admin add/update stock 11](#_Toc458785250)

[2.1.5.1. Introduction – 11](#_Toc458785251)

[2.1.5.2. Inputs – 11](#_Toc458785252)

[2.1.5.3. Processing– 11](#_Toc458785253)

[2.1.5.4. Outputs– 11](#_Toc458785254)

[2.1.5.5. Error Handling- 11](#_Toc458785255)

[2.1.6. Change Password 12](#_Toc458785256)

[2.1.6.1. Introduction – 12](#_Toc458785257)

[2.1.6.2. Inputs – 12](#_Toc458785258)

[2.1.6.3. Processing– 12](#_Toc458785259)

[2.1.6.4. Outputs– 12](#_Toc458785260)

[2.1.6.5. Error Handling- 12](#_Toc458785261)

[2.2. Non-Functional Requirements 13](#_Toc458785262)

[2.2.1. Performance 13](#_Toc458785263)

[2.2.2. Reliability 14](#_Toc458785264)

[2.2.3. Availability 14](#_Toc458785265)

[2.2.4. Security 14](#_Toc458785266)

[2.2.5. Maintainability 14](#_Toc458785267)

[2.2.6. Portability 14](#_Toc458785268)

[3. LITERATURE SURVEY 15](#_Toc458785269)

[3.1 PURPOSE OF LITERATURE SURVEY 15](#_Toc458785270)

[3.2 METHODOLOGY 15](#_Toc458785271)

[4. Detailed Design of The Project 17](#_Toc458785272)

[4.1 PRODUCT PERSPECTIVE 17](#_Toc458785273)

[4.2 SOFTWARE INTERFACES 17](#_Toc458785274)

[4.3 PRODUCT FUNCTIONS 17](#_Toc458785275)

[4.4 ARCHITECTURAL DESIGN 18](#_Toc458785276)

[4.5 USECASE DIAGRAM 18](#_Toc458785277)

[4.6 CLASS DIAGRAM 19](#_Toc458785278)

[4.7 SEQUENCE DIAGRAM 20](#_Toc458785279)

[5. Implementation of System 21](#_Toc458785280)

[5.1 DETAILED MODULE DESCRIPTION 21](#_Toc458785281)

[5.1.1 Introduction 21](#_Toc458785282)

[Process 21](#_Toc458785283)

[5.1.2 Existing System 22](#_Toc458785284)

[5.1.3 Proposed System 22](#_Toc458785285)

[5.1.4 Modules 23](#_Toc458785286)

[5.1.4.1 New User Registration Screen 23](#_Toc458785287)

[5.1.4.2 Login to the Application 23](#_Toc458785288)

[5.1.4.3 Product master. 23](#_Toc458785289)

[5.1.4.4 Customer master .](#_Toc458785290)

[5.1.4.5 Inventory Screen .](#_Toc458785291)

[5.1.4.6 Change Password 23](#_Toc458785292)

[5.1.5 System Architecture 24](#_Toc458785293)

[5.2 CODE 26](#_Toc458785294)

# **INTRODUCTION**

The modern industrial world is very much advanced in technology and competition in the world is intense. If we want healthy growth of business, one must keep business techniques up-to-date. Today computer is the mandatory part of every business in the world. That is why computer is used for major task like storing records, complex calculations, worldwide communication etc. since it is not possible to complete all these tasks by human itself in efficient way.

The Newspaper agency business is closely related to management of things like customer, stock, supplier, newspaper, and so on. So, if we select manpower to perform management of all these things, following problems may arise:

1. Paper based transaction has become old nowadays because it is time consuming and less secured/reliable.
2. The process of storing all these records also raises the issue of security and space.

Computerization solves all these problems and provides reliability and correctness in the process of management.

Performing operation with the help of computer will save the manual work of agent and so the billing work becomes easy no need for manual calculation. Security of data is maintained. We can retrieve or manage records instantly because of user friendly environment of computer.

The process which is carried out currently has certain drawbacks: -

* All work is carried on manually, hence increases the manpower.
* Also, the manual entries, increases the probability of human errors.
* There are lots of ways, through which the system might crack.
* Also there are separate formats required for different types of customers.

So we developed this Newspaper Agency.

* This project is to manage all the newspapers, their suppliers, customers, distributors, etc.
* It makes easy to generate and maintain bills, payments and auditing.

The prime goals of proposed system are:-

* To provide security of the software with the help of password facility.
* To provide User Friendly interface that will make various operations easier to perform.
* To provide automatic bill generation.
* To provide automatic stock inward outward.
* To provide automatic stock update after the selling process.
* To provide data validation.
* To provide facility of storing record of Employees, record of sold items, record of pending order, record of completed order, and record of expense.
* To provide facility of instant report generation.

# **SOFTWARE REQUIREMENT SPECIFICATION**

**2.1 FUNCTIONAL REQUIREMENTS**

# **Login to the Application**

# **Introduction** -

As you hit the exe of the application, it will display a login page.

User will enter the user-id and password in the fields provided.

# **Processing**–

The entry will be checked in the database and user will be welcomed to the home page if user is found registered.

# **Outputs**–

User will land on the application home page once it will be verified from the database.

# **Error Handling**-

1. Empty Filed Validation - The fields should not be empty. Empty field validation will be there.

2. Credentials Validation - User will not be allowed to login if password does not match with the password saved in the database

# **User Request for Newspaper**

# **Introduction** –

User Request for Newspaper

# **Inputs –**

User will request for newspaper by selecting newspaper type, language and no of papers.

# **Processing**–

Validate the given details and record the information in to the database.

# **Outputs**–

For the request, token number is generated.

# **Error Handling**-

Empty Filed Validation - The field should be selected from drop down and textbox should not be empty.

# **User track request**

# **Introduction** –

User track request status.

# **Inputs –**

User will enter token number and click on check status button.

# **Processing**–

Validate the given token number and admin update the status of request.

# **Outputs**–

User get request status message

# **Handling**-

Empty Filed Validation –The entered token no should be valid. Invalid token no is not accepted.

# **Admin view stock**

# **Introduction** –

Admin view stock

# **Inputs –**

In Inventory module, Admin click View Stock link.

# **Processing**–

The stock detail is maintained in database.

# **Outputs**–

Admin view stock details.

# **Error Handling**-

Empty Filed Validation –The entered token no should be valid. Invalid token no is not accepted.

# **Admin add/update stock**

# **Introduction** –

Admin add/update stock

# **Inputs –**

In Inventory module, Admin add/update stock by selecting cylinder type and no of cylinder.

# **Processing**–

The stock detail is maintained in database.

# **Outputs**–

Inventory updated successfully message is displayed.

# **Error Handling**-

Empty Filed Validation –The selected field should be selected and text field should not be empty.

# **Change Password**

# **Introduction** –

For security purpose password can be change over a time period.

# **Inputs –**

Old Password, New Password, Confirm Password.

# **Processing**–

Validate the given details and record the information in to the database.

# **Outputs**–

Database Record, Database successfully updated message.

# **Error Handling**-

Empty Filed Validation - The fields should not be empty. Empty field validation will be there.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Event | Trigger | Source | Use Case | Response | Destination |
| 1 | Login | Enter to System Feature | Admin or User | Entering Login username and password | Valid or Invalid Username or Password | Accessing System |
| 2 | Admin want to Verify User | Verify User | Admin | User Verification | User Verified or Not Verified | User Permitted |
| 3 | Admin wants to Check Stock | View Stock | Admin | Stock in Hand | Stock Details | View Stock |
| 4 | Admin wants to Add Stock | Adding Stock | Admin | Stock Updated | Stock Increased | Stock Updated in database |
| 5 | Admin Add or Update Rates | Update Rates | Admin | Rates Modified | Rates Updated | Database Updation |
| 6 | User Login | Login details | User | Entering Login username and password | Valid or Invalid Username or Password | Accessing System |
| 7 | User Request for Newspaper | Request for newspaper | User | Request for a paper | Request number | Request Added |
| 8 | User tracks request | Enters Request Details | User | Search for request number | Request Pending or complete | Details Displayed |

# **Non-Functional Requirements**

Non-functional requirements define the need in terms of performance, reliability, availability, security, maintainability, portability, security.

# **Performance**

Performance requirements define acceptable response times for system functionality. Although the system is developed suiting for the least system performances, the performance of the system will highly depend on the performance of the hardware and software components of the installing computer.

# **Reliability**

The software should provide automatically generated backup.

* + 1. **Availability**

The availability of the NPMS is up to the internet connection of the client. Since this is client-server related application, it shall be attainable all the time. User should have an account to enter the system.

# **Security**

The software used for this newspaper agency system include the password, so the security is provided. When anyone opens the software it has the provision for entering password. We have to enter the correct password, otherwise we cannot enter into the system. Password is saved in system Registry for more security.

# **Maintainability**

The software product will be easy to maintain and be fully function even when other system upgrades occurs. The system will perform with normalcy under any conditional state.

# **Portability**

The software will be able to support any Window operation system changes. The portability should be effective without disruption from normal performance.

# **LITERATURE SURVEY**

# **PURPOSE OF LITERATURE SURVEY**

# “The Newspaper Agency System” is one can go into almost required solution regarding the Newspaper shop.

# Software we can easily track all the customer detail, supplier detail, newspaper detail, stock detail, bill detail and we can able generate report.

# The main goal of the software is build a good management tool.

# The newspaper agency system will make storing of the customer records, stock records, sales detail, employee information, expense detail and order details in the database it will generate bills and print the bills.

# **METHODOLOGY**

The method being used in developing the system is the system Development Life Cycle (SDLC) The SDLC process includes project identification and selection, project initiation and planning, analysis, design, implementation and maintenance.

**System Development Life Cycle (SDLC)**

The system Development Life Cycle (SDLC) is a traditional methodology used to develop, maintain and replace or enhance information system. The main reason SDLC model is chosen because in SDLC it is possible to complete some activities in one phase in parallel with some activities of another phase. The life cycle can also be interactive that is phase are repeated as required until an acceptable system is found.

**Phase 1: Project identification and selection**

In this phase the project information system needs are identified and analyzed such as identified the title of the project that is Newspaper Agency System, scope and objective of the Newspaper Distributer System.

**Phase 2: Project initiation and planning**

During this phase the Gantt chart has been developed as a time line to determining the task involve in developing the Newspaper Agency System.

**Phase 3: Analysis**

In the phase, the existing system is studies by collecting the information through the Internet and analyzed the information to get alternatives for the use of proposed system. Determine what the Newspaper Agency System should do.

**Phase 4: Design**

Logical design is the fourth phase in SDLC methodology. The functional features chosen for the proposed system in Analysis phase are described. Part of the logical design of the information system is to devise the user interface. The interface plays an important role to connect the user with the system and is thus extremely important.

# **Detailed Design of The Project**

# **PRODUCT PERSPECTIVE**

Newspaper Agency System is done to make the manual process easier by making it a computerized system for billing and maintaining stock. The Newspaper Agency get the order request through phone calls or by personal from their customers and deliver the newspaper to their address based on their demand and previous delivery date. This process is made computerized and the customer’s name, address and stock detail are stored in a database. Based on this, billing for the customer is made simple and easier, since the customer order for newspaper can be accepted only after completing a certain period from the previous delivery. This can be calculated and billed easily through this.

# **SOFTWARE INTERFACES**

* Operating System: Windows XP, Window 7 x64 Edition.
* Front End: Java Swing, AWT.
* Back End: MySQL.

# **PRODUCT FUNCTIONS**

Admin (Server side module)

* Reads posted product details.
* Validates.
* Shows product status.
* Allowance of purchasing process

Purchasing Customer (Server-side module)

* Analyse and Checks existence posted product details
* Purchase the required product
* Make payment through credit card over the network.
* Automatic transaction processed the details and update the admin account.

# **ARCHITECTURAL DESIGN**

The software capabilities and requirements specified in Newspaper Agency System Simulation Software Requirements Specification are transformed into programs that will execute. Software items are partitioned into classes, objects, DFD, ER, Sequence and other diagram.

# **USECASE DIAGRAM**

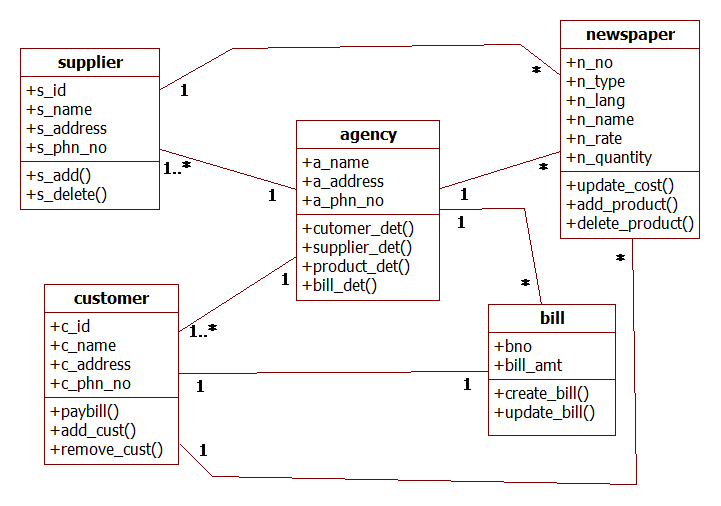
Use Case Diagrams (UCD) can be used to describe the functionality of a system in a horizontal way. That is, rather than merely representing the details of individual features of your system, UCDs can be used to show all of its available functionality.

**USE CASE DIAGRAM FOR ADMIN**



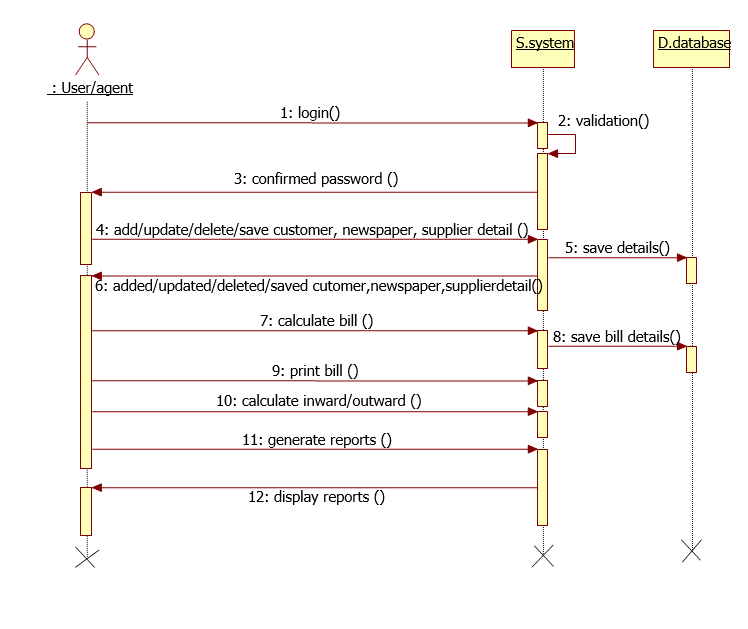
# **CLASS DIAGRAM**

Class diagrams enable us to establish relationship among various classes of the system.



# **SEQUENCE DIAGRAM**

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.



# **Implementation of System**

# **DETAILED MODULE DESCRIPTION**

# **5.1.1 Introduction**

The **Newspaper Agency Management System**’s objective is to make the manual process easier by making it a computerized system for billing and maintaining stock. Without this automation the management of the newspaper agency has become a tedious task. The system will be able to handle many services to take care of all customers in an easy and timely manner. The Main Objective of the Newspaper Agency is to provide a facility to customer by registering with required valid information for customer’s registration ID and for his identity. So customer can register his requirements by posting its valid details. They can request for a newspaper and track the request.

# **Process**

The various users participating in the system are as follows:

The Administrator

The User

1. **Administrator Role**

Admin is super user of this application. He is owner of this application. The admin will authenticate through system.

* + Administrator verifies the user.
  + Administrator maintains stock by adding/ updating / viewing stock detail.
  + Administrator can update the status of the request depending on the availability of stock.
  + Administrator can update the price of the cylinder.
  + Administrator can generate report of the transaction.

1. **User Role**

New users will first register into the system. At the time of registration, the user should provide all the required information.

* User can login with the registered UserId and password.
* User can request the newspaper which includes the type of newspaper and the unit of newspaper.
* When the user post the request token number will be generated.
* User can check the status of the request by entering the token number.

# **Existing System**

The existing system is a manual system. In the present Newspaper Agency system, all the activities are done manually. All data entry is performed by writing data into the book, paper documents. When there is need for retrieving details searching is unavoidable this is a difficult task searching the records manually .This is also too much time consuming when we want to retrieve details according to some specific conditions.

# **Proposed System**

The development of this system contains the following activities, which tries to automate the entire processes keeping in the view of database integration approach

* Users will receive better and quick service.
* Security is ensured by protecting the system with passwords.
* Normalized database tables eliminate data redundancy.
* Provision for minimizing errors in data entry.
* Efficient data storage.
* Real-time response and user-friendliness.
* Time saving.
* User Friendly
* Speed and Accuracy
* Efficiency and flexibility
* Formatted Output

# **Modules**

# **New User Registration Screen**

New Customer in the portal have to registered them for going through the services of the application.

# **Login to the Application**

Here we have 2 different login screens one for Admin and other for User. User can login into the application with registered credential.

The entry will be checked in the database and user will be welcomed to the home page if user is found registered. User will land on the application home page once it will be verified from the database.

# **Product Master.**

Admin can manage the products, see availability and manage the stock with the help of product master. He can set up combination of various products.

# **Customer Master**

Admin can add update and delete customer details.

# **Supplier Master**

Admin can manage the supplier details by adding/ viewing the them.

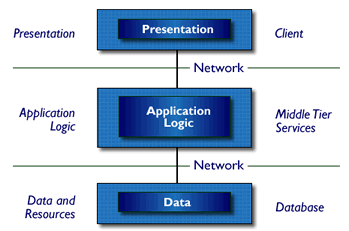
# **Change Password**

This Module provides the facility to change the password for the credential. It will be helpful for the security reasons.

# **System Architecture**

The current application is being developed by taking the 3-tier architecture as a prototype. The 3-tier architecture is the most common approach used for web applications today. In the typical example of this model, the web browser acts as the client, IIS handles the business logic, and a separate tier MS-SQL Server handles database functions.

Although the 3-tier approach increases scalability and introduces a separation of business logic from the display and database layers, it does not truly separate the application into specialized, functional layers. For prototype or simple web applications, the 3-tier architecture may be sufficient. However, with complex demands placed on web applications, a 3-tiered approach falls short in several key areas, including flexibility and scalability. These shortcomings occur mainly because the business logic tier is still too broad- it has too many functions grouped into one tier that could be separated out into a finer grained model.



Three-Tier Architecture

* Tier 1: the client contains the presentation logic, including simple control and user input validation. This application is also known as a thin client. The client interface is developed using Java Swing and AWT controls in some occasions
* Tier 2: the middle tier is also known as the application server, which provides the business processes logic and the data access. The business logic/ business rules can be written in Core Java.
* Tier 3: the data server provides the business data. MySQL database server acts as Tier-3, which is the database layer.

These are some of the advantages of three-tier architecture:

* It is easier to modify or replace any tier without affecting the other tiers.
* Separating the application and database functionality means better load balancing.
* Adequate security policies can be enforced within the server tiers without hindering the clients.

The proposed system can be designed perfectly with the three tier model, as all layers are perfectly getting set as part of the project. In the future, while expanding the system, in order to implement integration touch points and to provide enhanced user interfaces, the n-tier architecture can be used.

# **CODE**

**Login.java**

package com.npa;

import java.awt.BorderLayout;

import java.awt.FlowLayout;

import javax.swing.JButton;

import javax.swing.JDialog;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JRootPane;

import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;

import java.awt.Font;

import javax.swing.JTextField;

import com.npa.data.UserInformation;

import com.npa.model.LoginInfo;

import java.awt.Color;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import javax.swing.border.MatteBorder;

import javax.swing.border.EtchedBorder;

import javax.swing.border.CompoundBorder;

public class Login extends JDialog {

private final JPanel contentPanel = new JPanel();

private JTextField textUserName;

private JPasswordField textPassword;

JLabel lblInValidUser;

UserInformation userInfo;

public boolean IsLogin;

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) {

try {

Login dialog = new Login();

// dialog.setDefaultCloseOperation(JDialog.EXIT\_ON\_CLOSE);

dialog.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

/\*\*

\* Create the dialog.

\*/

public Login() {

setUndecorated(true);

//getRootPane().setWindowDecorationStyle(JRootPane.PLAIN\_DIALOG);

IsLogin = false;

userInfo = new UserInformation();

setBackground(Color.RED);

setTitle("Login");

setBounds(100, 100, 308, 216);

getContentPane().setLayout(new BorderLayout());

contentPanel.setBackground(new Color(51, 204, 255));

contentPanel.setBorder(new EtchedBorder(EtchedBorder.LOWERED, Color.BLACK, Color.GRAY));

getContentPane().add(contentPanel, BorderLayout.CENTER);

contentPanel.setLayout(null);

JLabel lblUserName = new JLabel("User Name:");

lblUserName.setFont(new Font("Tahoma", Font.PLAIN, 14));

lblUserName.setBounds(31, 42, 99, 14);

contentPanel.add(lblUserName);

textUserName = new JTextField();

textUserName.setBounds(107, 41, 124, 20);

contentPanel.add(textUserName);

textUserName.setColumns(10);

JLabel lblPassword = new JLabel("Password:");

lblPassword.setFont(new Font("Tahoma", Font.PLAIN, 14));

lblPassword.setBounds(31, 83, 78, 14);

contentPanel.add(lblPassword);

textPassword = new JPasswordField();

textPassword.setBounds(107, 82, 124, 20);

contentPanel.add(textPassword);

textPassword.setColumns(10);

lblInValidUser = new JLabel("In valid user name or password.");

lblInValidUser.setForeground(Color.RED);

lblInValidUser.setFont(new Font("Tahoma", Font.PLAIN, 14));

lblInValidUser.setBounds(31, 11, 208, 20);

contentPanel.add(lblInValidUser);

lblInValidUser.setVisible(false);

{

JPanel buttonPane = new JPanel();

buttonPane.setBorder(new EtchedBorder(EtchedBorder.LOWERED, Color.BLACK, Color.GRAY));

buttonPane.setBackground(new Color(51, 204, 255));

buttonPane.setLayout(new FlowLayout(FlowLayout.RIGHT));

getContentPane().add(buttonPane, BorderLayout.SOUTH);

{

JButton okButton = new JButton("Login");

okButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

lblInValidUser.setVisible(false);

if(!textUserName.getText().isEmpty() && !textPassword.getText().isEmpty())

{

IsLogin = userInfo.Login(textUserName.getText(), textPassword.getText());

if(IsLogin)

{

setVisible(false);

LoginInfo.UserName = textUserName.getText();

LoginInfo.UserPassword = textPassword.getText();

}

else

lblInValidUser.setVisible(true);

}

}

});

okButton.setActionCommand("OK");

buttonPane.add(okButton);

getRootPane().setDefaultButton(okButton);

}

{

JButton cancelButton = new JButton("Cancel");

cancelButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

cancelButton.setActionCommand("Cancel");

buttonPane.add(cancelButton);

}

}

}

}

**CustomerRequirements.java**

public class CustomerRequirements extends JInternalFrame {

private JTextField txtNOofCopies;

private JTextField textField\_3;

private JTextField textField\_5;

private JLabel lblCustomerid;

protected String Customer\_Name;

private JTextField textField;

private JList customerlist;

private JLabel lblId;

private JLabel lblNewLabel;

private JLabel lblLanguage;

private JComboBox LanguagecomboBox;

private JLabel lblProductid;

private JComboBox ProductidcomboBox;

private JLabel lblNoofcopies;

private JLabel lblStart;

private JButton btnNew;

private JButton btnUpdate;

private JButton btnDelete;

public JButton btnClose;

private JList productlist;

private JLabel lblLanguageName;

private JLabel lblProductidName;

private JDateChooser startDateChooser;

private JDateChooser closeDateChooser;

private CustomerInformation customerInformation;

//private ArrayList<Customer> allCustomer;

private ArrayList<Customer> filterCustomer;

private CustomerInformation customerInfo;

private ArrayList<Customer> allCustomers;

private ArrayList<CustomerRequirement> allRequirements;

private ProductInformation productInfo;

private ArrayList<Product> allProducts;

private Customer selectedCustomer;

CustomerRequirement selectedRequirement;

private ArrayList<Language> allLanguages;

private ArrayList<Product> allProductsByLanguage;

private JLabel lblCustomerRequirements;

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

CustomerRequirements frame = new CustomerRequirements();

frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the frame.

\*/

@SuppressWarnings({ "unchecked", "serial", "rawtypes" })

public CustomerRequirements() {

getContentPane().setFont(new Font("Tahoma", Font.PLAIN, 14));

customerInfo = new CustomerInformation();

allCustomers = customerInfo.GetAllCustomers();

allRequirements = new ArrayList<CustomerRequirement>();

productInfo = new ProductInformation();

allProducts = productInfo.GetAllProducts();

allLanguages = productInfo.GetAllLanguages();

setBounds(100, 100, 885, 638);

getContentPane().setLayout(null);

JPanel panel = new JPanel();

panel.setBorder(new TitledBorder(UIManager.getBorder("TitledBorder.border"), "CustomerRequirements", TitledBorder.LEADING, TitledBorder.TOP, null, Color.RED));

panel.setBounds(274, 283, 445, 281);

getContentPane().add(panel);

panel.setLayout(new FormLayout(new ColumnSpec[] {

FormFactory.RELATED\_GAP\_COLSPEC,

FormFactory.DEFAULT\_COLSPEC,

FormFactory.RELATED\_GAP\_COLSPEC,

FormFactory.DEFAULT\_COLSPEC,

FormFactory.RELATED\_GAP\_COLSPEC,

ColumnSpec.decode("default:grow"),},

new RowSpec[] {

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,}));

lblId = new JLabel("");

panel.add(lblId, "6, 2");

lblCustomerid = new JLabel("Customer\_Name :");

lblCustomerid.setForeground(Color.BLACK);

lblCustomerid.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblCustomerid, "2, 4");

lblCustomerid = new JLabel("");

panel.add(lblCustomerid, "6, 4");

lblLanguage = new JLabel("Language :");

lblLanguage.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblLanguage, "2, 6");

lblLanguageName = new JLabel("");

panel.add(lblLanguageName, "6, 6, fill, default");

LanguagecomboBox = new JComboBox();

LanguagecomboBox.setFont(new Font("Tahoma", Font.PLAIN, 14));

LanguagecomboBox.setVisible(false);

LanguagecomboBox.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

Language selectedLanguage=(Language)LanguagecomboBox.getSelectedItem();

allProductsByLanguage = new ArrayList<Product>();

for(int i=0;i<allProducts.size(); i++)

{

System.out.println(i);

if(allProducts.get(i).LanguageID == selectedLanguage.ID)

{

allProductsByLanguage.add(allProducts.get(i));

}

}

ProductidcomboBox.setModel(new DefaultComboBoxModel(allProductsByLanguage.toArray()));

}

});

LanguagecomboBox.setModel(new DefaultComboBoxModel(allLanguages.toArray()));

LanguagecomboBox.setRenderer(new DefaultListCellRenderer() {

@Override

public Component getListCellRendererComponent(

JList list, Object value, int index, boolean isSelected, boolean cellHasFocus) {

super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);

if (value instanceof Language) {

Language mec = (Language)value;

setText(mec.Name);

}

return this;

}

});

panel.add(LanguagecomboBox, "6, 6, fill, default");

lblProductid = new JLabel("Product Name :");

lblProductid.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblProductid, "2, 8");

lblProductidName = new JLabel("");

panel.add(lblProductidName, "6, 8, fill, default");

ProductidcomboBox = new JComboBox();

ProductidcomboBox.setFont(new Font("Tahoma", Font.PLAIN, 14));

ProductidcomboBox.setVisible(false);

ProductidcomboBox.setRenderer(new DefaultListCellRenderer() {

@Override

public Component getListCellRendererComponent(

JList list, Object value, int index, boolean isSelected, boolean cellHasFocus) {

super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);

if (value instanceof Product) {

Product mec = (Product)value;

setText(mec.Name);

}

return this;

}

});

panel.add(ProductidcomboBox, "6, 8, fill, default");

lblStart = new JLabel("Start Date :");

lblStart.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblStart, "2, 10, fill, default");

startDateChooser = new JDateChooser();

startDateChooser.getCalendarButton().addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

}

});

panel.add(startDateChooser, "6, 10, fill, default");

JLabel lblClosedate = new JLabel("Close\_Date :");

lblClosedate.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblClosedate, "2, 12");

closeDateChooser = new JDateChooser();

panel.add(closeDateChooser, "6, 12, fill, default");

lblNoofcopies = new JLabel("No\_of\_Copies :");

lblNoofcopies.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(lblNoofcopies, "2, 14");

txtNOofCopies = new JTextField();

txtNOofCopies.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel.add(txtNOofCopies, "6, 14, fill, default");

txtNOofCopies.setColumns(10);

((AbstractDocument) txtNOofCopies.getDocument()).setDocumentFilter(new IntDocumentFilter());

btnNew = new JButton("New");

btnNew.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnNew.setToolTipText("To add new customer \r\n\r\n");

btnNew.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

btnUpdate.setText("Save");

lblLanguageName.setVisible(false);

lblProductidName.setVisible(false);

ProductidcomboBox.setVisible(true);

LanguagecomboBox.setVisible(true);

startDateChooser.setDate(null);

closeDateChooser.setDate(null);

txtNOofCopies.setText("");

}

});

btnNew.setBounds(284, 575, 89, 23);

getContentPane().add(btnNew);

btnUpdate = new JButton("Update");

btnUpdate.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnUpdate.setToolTipText("To update customer's details.\r\n\r\n");

btnUpdate.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

if(btnUpdate.getActionCommand() == "Update")

{

CustomerRequirement requirements =new CustomerRequirement();

requirements.ID = selectedRequirement.ID;

requirements.CustomerID = selectedCustomer.ID;

requirements.ProductID = selectedRequirement.ProductID;

requirements.Start\_Date = startDateChooser.getDate();

if(closeDateChooser.getDate() == null)

{

requirements.Close\_Date = null;

}

else

{

requirements.Close\_Date = closeDateChooser.getDate();

}

requirements.No\_of\_Copies = Integer.valueOf(txtNOofCopies.getText());

allRequirements = customerInfo.UpdateCustomerRequirement(requirements, selectedCustomer.ID);

SetRequirements();

}

else

{

CustomerRequirement requirements =new CustomerRequirement();

requirements.ID = customerInfo.generateReqID();

requirements.CustomerID = selectedCustomer.ID;

requirements.ProductID = ((Product)ProductidcomboBox.getSelectedItem()).ID;

requirements.Start\_Date = startDateChooser.getDate();

if(closeDateChooser.getDate() == null)

{

requirements.Close\_Date = null;

}

else

{

requirements.Close\_Date = closeDateChooser.getDate();

}

requirements.No\_of\_Copies = Integer.valueOf(txtNOofCopies.getText());

allRequirements = customerInfo.SaveCustomerRequirement(requirements, selectedCustomer.ID);

SetRequirements();

}

}

});

btnUpdate.setBounds(375, 575, 89, 23);

getContentPane().add(btnUpdate);

btnDelete = new JButton("Delete");

btnDelete.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnDelete.setToolTipText("To delete customer's details\r\n\r\n");

btnDelete.setBounds(464, 575, 89, 23);

getContentPane().add(btnDelete);

btnClose = new JButton("Close");

btnClose.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnClose.setToolTipText("To exit\r\n");

btnClose.setBounds(552, 575, 89, 23);

getContentPane().add(btnClose);

JPanel panel\_2 = new JPanel();

panel\_2.setBorder(new TitledBorder(UIManager.getBorder("TitledBorder.border"), "Products", TitledBorder.LEADING, TitledBorder.TOP, null, Color.RED));

panel\_2.setBounds(106, 283, 158, 277);

getContentPane().add(panel\_2);

panel\_2.setLayout(new FormLayout(new ColumnSpec[] {

FormFactory.RELATED\_GAP\_COLSPEC,

FormFactory.DEFAULT\_COLSPEC,

FormFactory.RELATED\_GAP\_COLSPEC,

ColumnSpec.decode("default:grow"),},

new RowSpec[] {

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

RowSpec.decode("default:grow"),}));

productlist = new JList();

productlist.setFont(new Font("Tahoma", Font.PLAIN, 14));

productlist.addListSelectionListener(new ListSelectionListener() {

public void valueChanged(ListSelectionEvent arg0) {

if(!arg0.getValueIsAdjusting())

{

selectedRequirement =(CustomerRequirement) productlist.getSelectedValue();

if(selectedRequirement != null)

{

lblProductidName.setText(selectedRequirement.SelectedProduct.Name);

startDateChooser.setDate(selectedRequirement.Start\_Date);

closeDateChooser.setDate(selectedRequirement.Close\_Date);

txtNOofCopies.setText(String.valueOf(selectedRequirement.No\_of\_Copies));

lblLanguageName.setText(selectedRequirement.LanguageName);

lblLanguageName.setVisible(true);

LanguagecomboBox.setVisible(false);

ProductidcomboBox.setVisible(false);

lblProductidName.setVisible(true);

}

}

}

});

productlist.setCellRenderer(new DefaultListCellRenderer() {

@Override

public Component getListCellRendererComponent(JList<?> list, Object value, int index, boolean isSelected, boolean cellHasFocus) {

Component renderer = super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);

if (renderer instanceof JLabel && value instanceof CustomerRequirement) {

// Here value will be of the Type 'CustomerRequirement'

((JLabel) renderer).setText(((CustomerRequirement) value).SelectedProduct.Name);

}

return renderer;

}

});

panel\_2.add(productlist, "2, 2, 3, 9, fill, fill");

JPanel panel\_1 = new JPanel();

panel\_1.setBorder(new TitledBorder(UIManager.getBorder("TitledBorder.border"), "Customer", TitledBorder.LEADING, TitledBorder.TOP, null, Color.RED));

panel\_1.setBounds(106, 92, 580, 180);

getContentPane().add(panel\_1);

panel\_1.setLayout(new FormLayout(new ColumnSpec[] {

FormFactory.RELATED\_GAP\_COLSPEC,

FormFactory.DEFAULT\_COLSPEC,

FormFactory.RELATED\_GAP\_COLSPEC,

ColumnSpec.decode("default:grow"),

FormFactory.RELATED\_GAP\_COLSPEC,

FormFactory.DEFAULT\_COLSPEC,

FormFactory.RELATED\_GAP\_COLSPEC,

ColumnSpec.decode("default:grow"),},

new RowSpec[] {

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

FormFactory.DEFAULT\_ROWSPEC,

FormFactory.RELATED\_GAP\_ROWSPEC,

RowSpec.decode("default:grow"),}));

JLabel lblFind = new JLabel("Find:");

lblFind.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel\_1.add(lblFind, "2, 2");

textField = new JTextField();

textField.addKeyListener(new KeyAdapter() {

@Override

public void keyTyped(KeyEvent e) {

char c = e.getKeyChar();

if(!Character.isLetter(c))

if(!Character.isSpaceChar(c))

e.consume();

}

});

textField.addKeyListener(new KeyAdapter() {

@Override

public void keyReleased(KeyEvent arg0) {

if(textField.getText() != "")

{

filterCustomer = new ArrayList<Customer>();

for(int i=0; i<allCustomers.size(); i++)

{

if(allCustomers.get(i).Name.toLowerCase().startsWith(textField.getText()))

{

filterCustomer.add(allCustomers.get(i));

}

}

SetList(filterCustomer);

}

else

{

SetList(allCustomers);

}

}

});

textField.setFont(new Font("Tahoma", Font.PLAIN, 14));

panel\_1.add(textField, "4, 2, 4, 1, fill, default");

textField.setColumns(10);

((AbstractDocument) textField.getDocument()).setDocumentFilter(new AlphabetDocumentFilter());

customerlist = new JList();

customerlist.setFont(new Font("Tahoma", Font.PLAIN, 14));

DefaultListModel model = new DefaultListModel();

for(int i=0;i<allCustomers.size(); i++)

{

model.addElement(allCustomers.get(i));

}

customerlist.setModel(model);

customerlist.setCellRenderer(new DefaultListCellRenderer() {

@Override

public Component getListCellRendererComponent(JList<?> list, Object value, int index, boolean isSelected, boolean cellHasFocus) {

Component renderer = super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);

if (renderer instanceof JLabel && value instanceof Customer) {

// Here value will be of the Type 'Product'

((JLabel) renderer).setText(((Customer) value).Name);

}

return renderer;

}

});

customerlist.addListSelectionListener(new ListSelectionListener() {

public void valueChanged(ListSelectionEvent e) {

if(!e.getValueIsAdjusting())

{

selectedCustomer =(Customer) customerlist.getSelectedValue();

if(selectedCustomer != null)

{

lblCustomerid.setText(selectedCustomer.Name);

allRequirements = customerInfo.GetRequirement(selectedCustomer.ID);

SetRequirements();

}

}

}

});

panel\_1.add(customerlist, "2, 4, 7, 3, fill, fill");

lblCustomerRequirements = new JLabel("Customer Requirements");

lblCustomerRequirements.setHorizontalAlignment(SwingConstants.CENTER);

lblCustomerRequirements.setForeground(Color.RED);

lblCustomerRequirements.setFont(new Font("Tahoma", Font.BOLD | Font.ITALIC, 30));

lblCustomerRequirements.setBounds(274, 27, 383, 39);

getContentPane().add(lblCustomerRequirements);

}

@SuppressWarnings("unchecked")

void SetRequirements()

{

@SuppressWarnings("rawtypes")

DefaultListModel model = new DefaultListModel();

for(int i=0;i<allRequirements.size(); i++)

{

for(int j=0; j<allProducts.size();j++)

{

if(allProducts.get(j).ID==allRequirements.get(i).ProductID)

{

allRequirements.get(i).SelectedProduct=allProducts.get(j);

break;

}

}

model.addElement(allRequirements.get(i));

}

productlist.removeAll();

productlist.setModel(model);

}

@SuppressWarnings({ "rawtypes", "unchecked" })

void SetList(ArrayList<Customer> customers)

{

DefaultListModel model = new DefaultListModel();

for(int i=0;i<customers.size(); i++)

{

model.addElement(customers.get(i));

}

customerlist.removeAll();

customerlist.setModel(model);

}

}