Birth Certificate Management System

A MINI-PROJECT BY:

KIRUTHIKA KM(230701153)

LAKSHAYA.S(230701160)

MOUNIKKHA.GA(230701198)

in partial fulfillment of the award of the degree

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project report "**Birth Certificate Management System"** is a Bonafide work of "**Kiruthika KM (230701153) &Lakshaya.s(230701153)&Mounikkha GA(230701153)..**

Submitted for the Practical Examination held on _____

Signature

Ms. Dharshini B S

Assistant Professor,

Computer Science and Engineering,

Rajalakshmi Engineering College (Autonomous),

Thandalam, Chennai-602 105

Internal Examiner **External Examiner**

ABSTRACT

The Birth Certificate Management System is a robust digital solution designed to streamline the process of managing and maintaining birth records. Leveraging the power of Java Database Connectivity (JDBC) and MySQL database, this system offers a comprehensive suite of functionalities for various stakeholders, including government officials, healthcare providers, and the general public.

At its core, the system enables efficient record insertion, retrieval, update, and deletion. Users can input essential birth details such as child's name, date of birth, parents' information, and place of birth. The system rigorously validates input data to ensure accuracy and completeness, safeguarding the integrity of the stored information. Once validated, birth records are securely stored in the MySQL database, providing a centralized repository for easy access and management.

The system's robust search functionality allows users to retrieve specific birth records based on various criteria, including child's name, date of birth, or parent's information. This empowers authorized personnel to quickly locate and access relevant records, streamlining administrative tasks and improving response times.

Furthermore, the system facilitates record updates, enabling authorized users to modify existing birth records to correct errors or update information. The system's built-in validation mechanisms ensure that updates are accurate and consistent with established guidelines, preserving data integrity.

In certain circumstances, authorized personnel may need to delete birth records, such as in cases of duplicate records or legal requirements. The system supports this functionality while maintaining a record of deleted records for auditing and recovery purposes.

The system's technical implementation leverages a combination of powerful technologies: Java provides the core programming language for the application's logic, while JDBC enables seamless communication between the Java application and the MySQL database. MySQL serves as the robust and scalable database to store and manage birth records. The user interface, crafted using either Java Swing or JavaFX, offers a user-friendly experience for interacting with the system.

By automating birth certificate management tasks and providing a centralized repository of birth records, this system offers several significant benefits. It streamlines the process of registering and accessing birth records, enhances efficiency, and ensures data accuracy through rigorous validation and error checking. Moreover, the system prioritizes security by implementing appropriate measures to protect sensitive birth information. It provides easy access to birth records for authorized users, fostering transparency and accountability. Finally, the system's scalable architecture can accommodate increasing numbers of records and users, ensuring its long-term sustainability.

In conclusion, the Birth Certificate Management System is a valuable tool that contributes to improved efficiency, accuracy, and security in vital record keeping. By automating tasks, providing a centralized repository, and offering a user-friendly interface, this system empowers stakeholders to effectively manage and utilize birth records, ultimately benefiting both individuals and society as a whole.

TABLE OF CONTENTS

INTRODUCTION

INTRODUCTION:

Our Birth Certificate Management System is a comprehensive solution that leverages the power of Java, JDBC, MySQL, and JavaFX to streamline the process of managing birth certificate records. This system offers a user-friendly interface that allows for easy addition, deletion, and viewing of birth certificates.

By utilizing MySQL's robust database capabilities and JDBC's efficient data access, we ensure the secure storage and retrieval of critical information. Our system is designed to enhance efficiency, accuracy, and security, making it an invaluable tool for organizations seeking to optimize their birth certificate management processes

IMPLEMENTTATION:

This project uses JAVA FX for building the frontend phase and MYSQL to handle the backend and databases.

WEBSITE FEATURE:

A responsive website is created using features of JAVA FX with an active home page.

The home page offers options like Add CERTIFICATE, Delete certificate ,view certificate also the options to display the same details.

A responsive corresponding pages are linked with each options.

SYSTEM SPECIFICATION

1. Hardware Requirements

- Processor: Intel Core i5 (10th Gen or higher) / AMD Ryzen 5 or equivalent
- RAM: Minimum 8 GB (16 GB recommended for smoother development)
- Storage: 256 GB SSD (Solid State Drive) or higher.

2. Software Requirements

- Frontend: Java FX
- Backend: mysql
- Operating System: Windows 10

SAMPLE CODE

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import java.sql.*;

public class BirthCertificateManagement extends JFrame {
private JTextField nameField, dobField, placeField,
certNumberField;
private JButton addButton, viewButton, deleteButton;
private JTable table;
private DefaultTableModel tableModel;

private static final String url =
"jdbc:mysql://localhost:3306/birth_certificate_db";
private static final String username = "root";
private static final String password = "Vigshan@2116";

private Connection connection;

public BirthCertificateManagement() {
// Database Connection
try {
connection = DriverManager.getConnection(url, username,
password);
} catch (SQLException e) {
e.printStackTrace();
}

setTitle("Birth Certificate Management");
setSize(600, 400);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLocationRelativeTo(null);
setLayout(null);
```

```
JLabel nameLabel = new JLabel("Name:");
nameLabel.setBounds(20, 20, 100, 25);
add(nameLabel);

nameField = new JTextField();
nameField.setBounds(120, 20, 150, 25);
add(nameField);

JLabel dobLabel = new JLabel("Date of Birth:");
dobLabel.setBounds(20, 60, 100, 25);
add(dobLabel);

dobField = new JTextField();
dobField.setBounds(120, 60, 150, 25);
add(dobField);

JLabel placeLabel = new JLabel("Place of Birth:");
placeLabel.setBounds(20, 100, 100, 25);
add(placeLabel);

placeField = new JTextField();
placeField.setBounds(120, 100, 150, 25);
add(placeField);

JLabel certNumberLabel = new JLabel("Certificate Number:");
certNumberLabel.setBounds(20, 140, 150, 25);
add(certNumberLabel);

certNumberField = new JTextField();
certNumberField.setBounds(170, 140, 100, 25);
add(certNumberField);

addButton = new JButton("Add Certificate");
addButton.setBounds(20, 180, 150, 30);
add(addButton);
addButton.addActionListener(new AddCertificateAction());
```

```
viewButton = new JButton("View Certificates");
viewButton.setBounds(200, 180, 150, 30);
add(viewButton);
viewButton.addActionListener(new ViewCertificatesAction());

deleteButton = new JButton("Delete Certificate");
deleteButton.setBounds(380, 180, 150, 30);
add(deleteButton);
deleteButton.addActionListener(new
DeleteCertificateAction());

tableModel = new DefaultTableModel(new String[]{"ID",
"Name", "Date of Birth", "Place of Birth", "Certificate
Number"}, 0);
table = new JTable(tableModel);
JScrollPane scrollPane = new JScrollPane(table);
scrollPane.setBounds(20, 220, 550, 120);
add(scrollPane);
}
```

Add certificate

```
private class AddCertificateAction implements ActionListener {
public void actionPerformed(ActionEvent e) {
try {
PreparedStatement ps =
connection.prepareStatement("INSERT INTO birth_certificates
(name, date_of_birth, place_of_birth, certificate_number)
VALUES (?, ?, ?, ?)");
ps.setString(1, nameField.getText());
ps.setString(2, dobField.getText());
ps.setString(3, placeField.getText());
ps.setString(4, certNumberField.getText());
ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Certificate added
successfully.");
} catch (SQLException ex) {
ex.printStackTrace();
JOptionPane.showMessageDialog(null, "Error adding
certificate.");
```

```
}
}
}
```

View certificate

```
private class ViewCertificatesAction implements ActionListener
{
public void actionPerformed(ActionEvent e) {
tableModel.setRowCount(0);
try {
Statement stmt = connection.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM
birth_certificates");
while (rs.next()) {
tableModel.addRow(new Object[]{
rs.getInt("id"),
rs.getString("name"),
rs.getString("date_of_birth"),
rs.getString("place_of_birth"),
rs.getString("certificate_number")
});
}
} catch (SQLException ex) {
ex.printStackTrace();
JOptionPane.showMessageDialog(null, "Error fetching
certificates.");
}
}
}
```

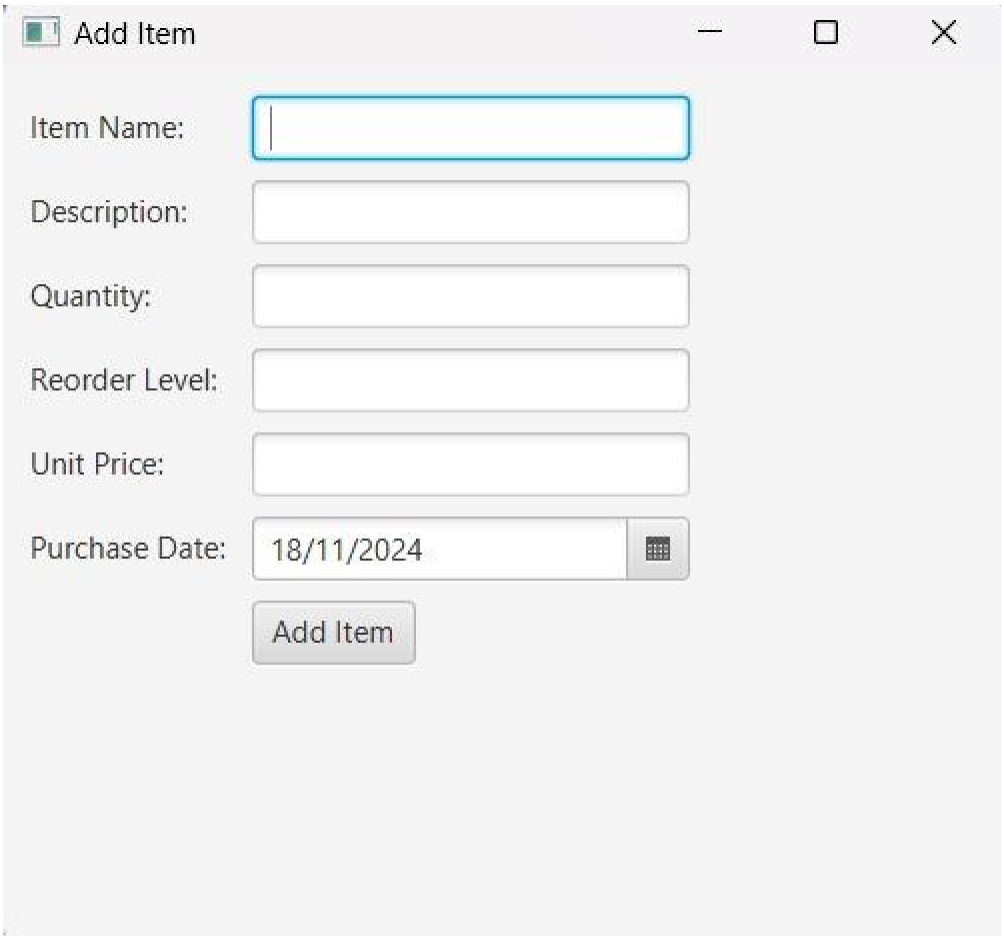
Delete certificate

```
private class DeleteCertificateAction implements
ActionListener {
public void actionPerformed(ActionEvent e) {
int selectedRow = table.getSelectedRow();
if (selectedRow != -1) {
int id = (int) tableModel.getValueAt(selectedRow, 0);
try {
PreparedStatement ps =
```

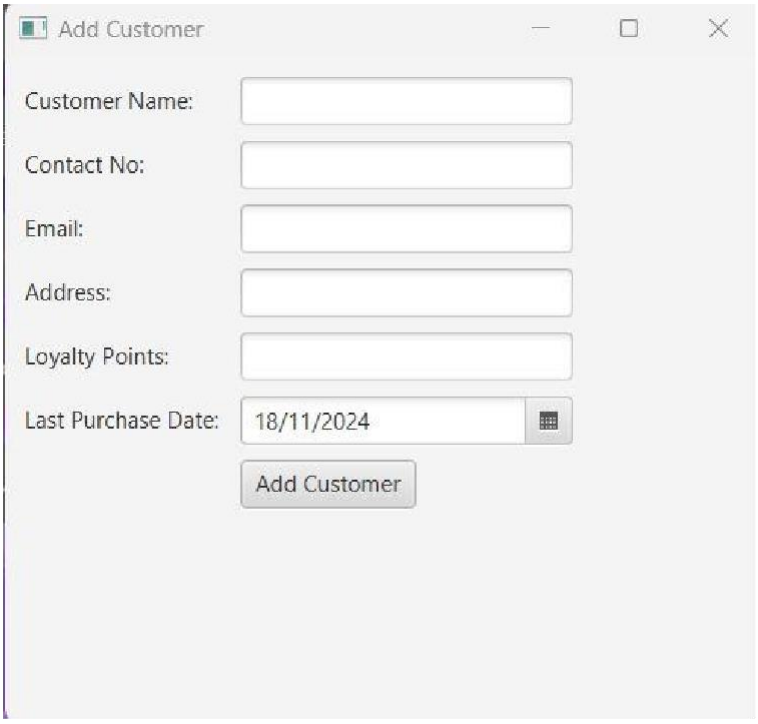
```
connection.prepareStatement("DELETE FROM
birth_certificates WHERE id = ?");
ps.setInt(1, id);
ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Certificate deleted
successfully.");
} catch (SQLException ex) {
ex.printStackTrace();
JOptionPane.showMessageDialog(null, "Error deleting
certificate.");
}
} else {
JOptionPane.showMessageDialog(null, "Select a certificate to
delete.");
}
}
}

public static void main(String[] args) {
SwingUtilities.invokeLater(() -> {
BirthCertificateManagement app = new
BirthCertificateManagement();
app.setVisible(true);
});
}
}
```

Home Page:

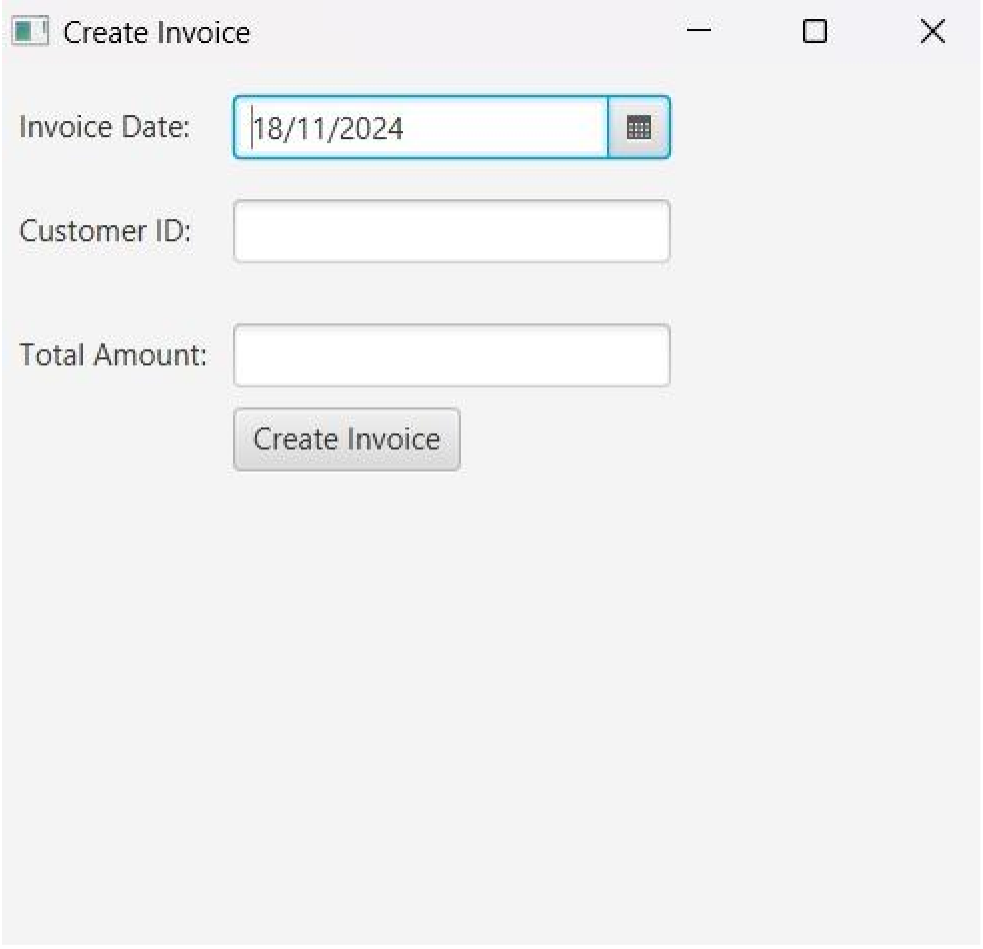
Add Items Page:


Add Item
Item Name:
Description:
Quantity:
Reorder Level:
Unit Price:
Purchase Date:
18/11/2024
Add Item

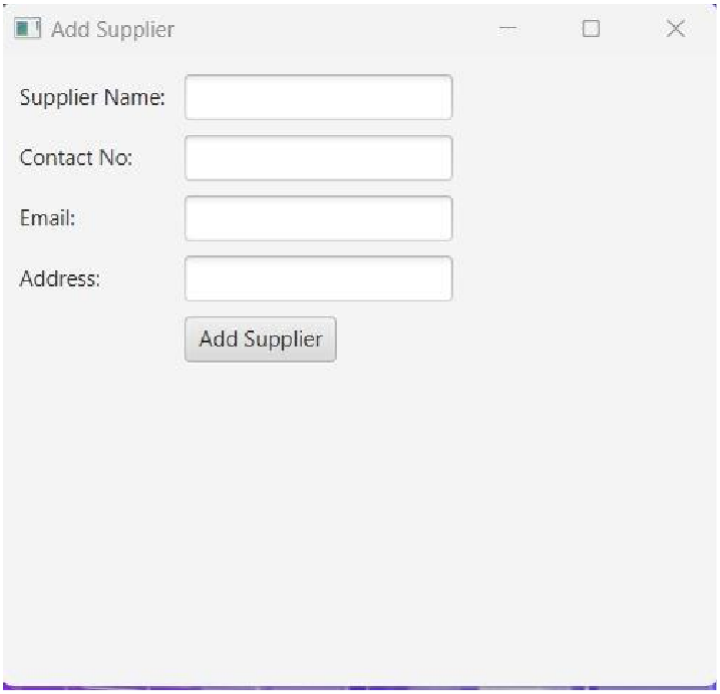
Add Customer Page:



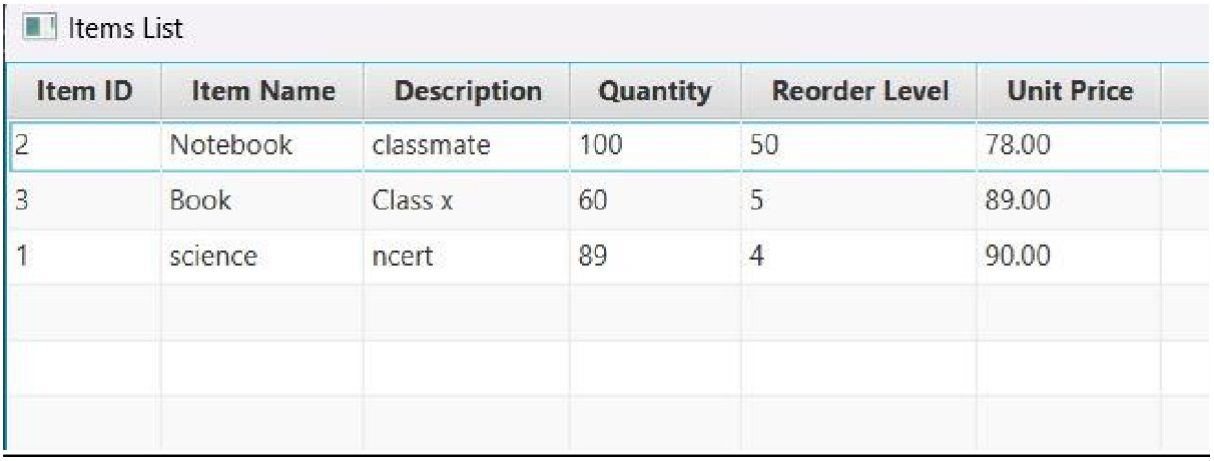
Add Invoice Page:



Add Supplier Page:

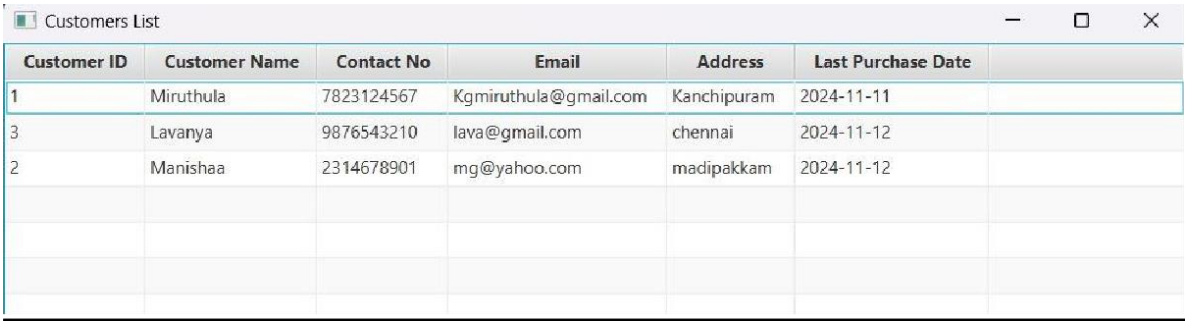


Display Items Page:



Item ID	Item Name	Description	Quantity	Reorder Level	Unit Price
2	Notebook	classmate	100	50	78.00
3	Book	Class x	60	5	89.00
1	science	ncert	89	4	90.00

Display Customer Page:



Customer ID	Customer Name	Contact No	Email	Address	Last Purchase Date
1	Miruthula	7823124567	Kgmiruthula@gmail.com	Kanchipuram	2024-11-11
3	Lavanya	9876543210	lava@gmail.com	chennai	2024-11-12
2	Manishaa	2314678901	mg@yahoo.com	madipakkam	2024-11-12

Display Invoice Page:

Invoices List

Invoice ID	Customer ID	Invoice Date	Bill Amount
1	1	2024-11-12	456.00
2	1	2024-11-12	34.00
3	1	2024-11-11	56.00
4	2	2024-11-12	200.00
5	3	2024-11-12	90.00
6	2	2024-11-12	200.00

Display Supplier Page:

Suppliers List

Supplier ID	Supplier Name	Contact No	Email	Address
1	Classmate	9090345678	Classmatehelp@gmail.com	Guindy,Chennai

CONCLUSION

The Birth Certificate Management System, a Java-based application powered by JDBC and MySQL, offers a comprehensive solution for efficient and secure management of birth certificate records. By leveraging the robust features of JavaFX, the system provides a user-friendly interface that simplifies the processes of adding, viewing, and deleting certificates.

The integration of MySQL ensures the secure storage and retrieval of critical information, while JDBC facilitates efficient communication between the application and the database. The system's robust security measures and user-centric design make it a reliable and efficient tool for organizations to streamline their birth certificate management operations.

REFERENCES

1.https://w3schools.com

2.https://codewithcurious.com

3.https://github.com

4.https://javaatpoint.com