**Title:Resume Builder**

1. **Introduction:**

A Resume Builder is a tool designed to help job seekers create professional, visually appealing resumes with ease. By offering pre-designed templates, user-friendly interfaces, and customization options, it simplifies the process of showcasing skills, experience, and achievements, enabling users to make a strong first impression on potential employers.

* 1. **Problem Statement:**

Creating professional resumes is often challenging due to a lack of formatting skills and limited guidance. Existing tools can be complex or restrictive. This project will develop a user-friendly **Resume Builder** with an intuitive **front-end interface** for easy resume creation and formatting, supported by a secure **back-end** for storing and managing user data efficiently.

* 1. **Technology Stack:**

1. Java: The main programming language for application logic and interaction.

2. MySQL: A relational database for storing resumes with fields like name, email, phone, skills, and experience.

3. SQL Query: Used for creating and querying the database and tables.

**1.3 Proposed Methodology:**

Process Flow:

1. Requirement Analysis:

Define the application features (Create Resume, View All Resumes, Exit).

Design the database schema to store user information.

2. Database Design:

Create a MySQL database (ResumeBuilderDB).

Define a table (Resumes) with appropriate fields (id, name, email, phone, skills, experience).

3. Application Development:

Develop the Java-based application:

Menu Options:

Create a Resume: Allows the user to input personal and professional details and save them in the database.

View All Resumes: Fetch and display all stored resumes from the database.Exit: Exits the application.

Input Handling: Use Java to validate and accept user inputs.

Integrate Java with MySQL using the JDBC library for database operations.

4. Implementation of CRUD Operations:

Create: Insert new resumes into the database.

Read: Retrieve all resumes for display.

Update: (Future feature) Modify existing resumes.

Delete: (Future feature) Remove specific resumes.

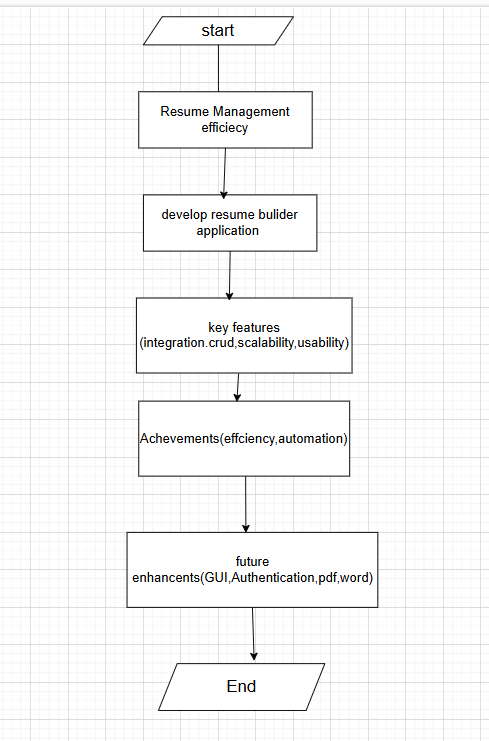
5. Testing:

Perform unit testing on Java functions.

Test database interactions (insertion, retrieval).

6. Deployment:

Package the application and deploy it on the required system

****

**2. Implementation/Case Study:**

import java.sql.\*;

import java.util.Scanner;

public class ResumeBuilder {

// JDBC Connection details

private static final String DB\_URL = "jdbc:mysql://localhost:3306/ResumeBuilderDB";

private static final String DB\_USER = "root"; // Replace with your MySQL username

private static final String DB\_PASSWORD = "password"; // Replace with your MySQL password

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Connect to the database

try (Connection connection = DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASSWORD)) {

while (true) {

System.out.println("=== Resume Builder ===");

System.out.println("1. Create a Resume");

System.out.println("2. View All Resumes");

System.out.println("3. Exit");

System.out.print("Choose an option: ");

int choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

switch (choice) {

case 1:

createResume(scanner, connection);

break;

case 2:

viewAllResumes(connection);

break;

case 3:

System.out.println("Exiting Resume Builder. Goodbye!");

return;

default:

System.out.println("Invalid choice. Please try again.\n");

}

}

} catch (SQLException e) {

e.printStackTrace();

System.out.println("Failed to connect to the database.");

}

}

private static void createResume(Scanner scanner, Connection connection) throws SQLException {

System.out.print("Enter your full name: ");

String name = scanner.nextLine();

System.out.print("Enter your email: ");

String email = scanner.nextLine();

System.out.print("Enter your phone number: ");

String phone = scanner.nextLine();

System.out.print("Enter your skills (comma-separated): ");

String skills = scanner.nextLine();

System.out.print("Enter your experience: ");

String experience = scanner.nextLine();

String sql = "INSERT INTO Resumes (name, email, phone, skills, experience) VALUES (?, ?, ?, ?, ?)";

try (PreparedStatement statement = connection.prepareStatement(sql)) {

statement.setString(1, name);

statement.setString(2, email);

statement.setString(3, phone);

statement.setString(4, skills);

statement.setString(5, experience);

int rowsInserted = statement.executeUpdate();

if (rowsInserted > 0) {

System.out.println("Resume added successfully!\n");

}

}

}

private static void viewAllResumes(Connection connection) throws SQLException {

String sql = "SELECT \* FROM Resumes";

try (Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(sql)) {

if (!resultSet.isBeforeFirst()) { // Check if resultSet is empty

System.out.println("No resumes available.\n");

return;

}

System.out.println("=== All Resumes ===");

while (resultSet.next()) {

System.out.println("Resume ID: " + resultSet.getInt("id"));

System.out.println("Name: " + resultSet.getString("name"));

System.out.println("Email: " + resultSet.getString("email"));

System.out.println("Phone: " + resultSet.getString("phone"));

System.out.println("Skills: " + resultSet.getString("skills"));

System.out.println("Experience: " + resultSet.getString("experience"));

System.out.println("----------------------");

}

}

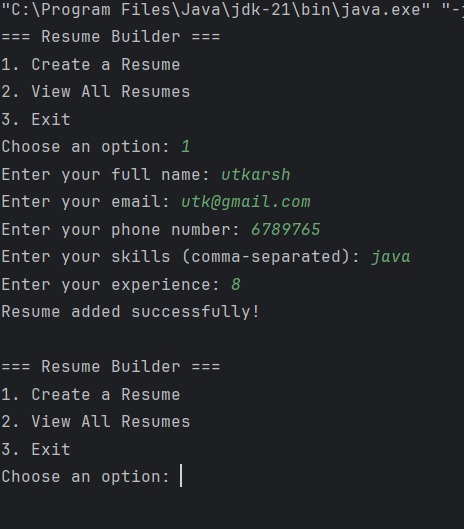
}

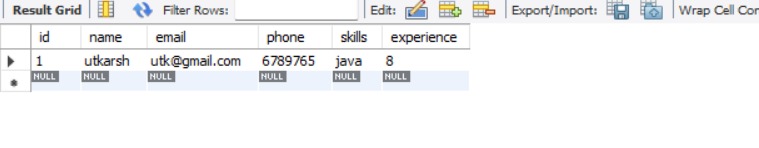
}

**Back-End**

****

1. **Results & Discussions**

****

****

1. **Conclusion:**

The Resume Builder application simplifies resume management by integrating Java with MySQL for efficient storage and retrieval. It features a user-friendly menu, scalable design, and automation of CRUD operations. The system meets its objectives and can be enhanced with a GUI, authentication, and advanced features like exporting resumes. This project provides a solid base for future development in resume management systems.