# RESUME VERIFICATION SYSTEM

**A MINI-PROJECT REPORT**

***Submitted by***

**LOGA PRABHA GB 240701290**

**SHREMATHI K 240701504**

***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**

**NOVEMBER 2025**

## BONAFIDE CERTIFICATE

Certified that this project **"RESUME VERIFICATION SYSTEM"** is the bonafide work of **"LOGA PRABHA GB, SHREMATHI K"** who carried out the project work under my supervision.

**SIGNATURE**

**Dr. V. JANANEE**

**ASSISTANT PROFESSOR SG**

Dept. of Computer Science and Engg,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

_____

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

# ABSTRACT

In today's recruitment landscape, **verifying the authenticity and completeness of resumes** has become a tedious and error-prone task. The Resume Verification System aims to **automate resume analysis** using a database-driven application that validates the structure, keywords, and contents of resumes.

This desktop application, built using Java Swing and MySQL, allows users to upload, paste, or analyze their resume contents. It performs smart text-based validation such as checking for essential fields (name, email, phone number, education, and skills), **ensures proper word count and keyword presence, and generates a comprehensive score and verification summary**. All results and resumes are stored in a MySQL database for easy retrieval and analytics.

This project demonstrates efficient frontend-backend integration, database connectivity, and content verification logic, making resume evaluation simpler, automated, and reliable for academic or business use.

# ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M.THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head Of The Department **Dr. E.M. MALATHY** and our Deputy Head Of The Department **Dr. J. MANORANJINI** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Dr. V. JANANEE ,** for her valuable guidance and motivation during the completion of this project.

**1. LOGA PRABHA GB**

**2. SHREMATHI K**

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION

The Resume Verification System is a database-driven desktop application designed to validate and verify resumes efficiently. Traditional manual resume screening often leads to inconsistencies and errors. This system automates verification by scanning resume text for crucial details such as contact info, education, skills, and experience, ensuring data completeness and authenticity.

## SCOPE OF THE WORK

This application is designed for HR professionals, educational institutions, and online job platforms to evaluate resumes. It can later be extended to integrate AI-based scoring models, PDF parsing, and email verification tools.

## PROBLEM STATEMENT

With the digital evolution and high job application volumes, fake or incomplete resumes are common. Manual verification consumes excessive time and effort. A semi-automated, scalable tool is needed to verify resume content based on structure and required key points.

## 1.4 AIM AND OBJECTIVES OF THE PROJECT

The main aims of this project are:

- To automate resume verification using a rule-driven algorithm.
- To store and manage user data and resume analysis results in a MySQL database.

- To develop a friendly Java Swing GUI for users to upload and analyze resumes.
- To enhance evaluation accuracy through keyword and field validation.

# CHAPTER 2

## SYSTEM SPECIFICATIONS

### 2.1 HARDWARE SPECIFICATIONS

| Component | Specification |
|---|---|
| Processor | Intel i5 or higher |
| RAM | 8 GB Minimum |
| Storage | 256 GB SSD / 1 TB HDD |
| Display | 1366x768 or higher |

### 2.2 SOFTWARE SPECIFICATIONS

| Component | Specification |
|---|---|
| Operating System | Windows 10 / Linux / macOS |

| | |
|---|---|
| Front End | Java Swing |
| Back End | MySQL Database |
| Programming Language | Java (JDK 17+) |
| Tools Used | IntelliJ IDEA, MySQL Workbench |
| Libraries | JDBC Connector, Swing components |

# CHAPTER 3

## MODULE DESCRIPTION

This application consists of two modules. When the program runs, it will ask for a confirmation to the login window. The person who interacts can login as an Administrator or as a User. The description of the modules are as follows:

1. **LOGIN MODULE**

Provides authentication for admin and user accounts using database-stored credentials. Ensures secure login with email and password.

2. **DASHBOARD MODULE**

After login, users can upload or paste resumes for analysis. It includes buttons to load files, analyze content, and view verification summaries.

3. **VERIFICATION MODULE**

Analyzes resumes for required fields:

Personal Info (name, email, phone)
Education details
Skill keywords
Experience details
Calculates a final score out of 100 based on content quality and completeness.

4 . **DATABASE MODULE**

Handles storing:

- User credentials
- Uploaded resumes and details

- Analysis results

  Uses relational design:

- `users(user_id, email, password, role)`
- `resumes(resume_id, user_id, file_name, resume_text, word_count)`
- `verification_results(result_id, resume_id, overall_score, keyword_hits, education_keywords, skill_keywords, status)`

## 5. HISTORY AND REPORT MODULE

Displays all past verifications, allows searching/viewing previous entries, and provides a detailed summary of each resume.

# CHAPTER 4

## SAMPLE CODING

### 4.1 Database Connection Code

```
// DatabaseConnection.java - Establishing MySQL Connection



import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DatabaseConnection {

    private static final String URL = "jdbc:mysql://localhost:3306/resume_verifier";

    private static final String USER = "root";

    private static final String PASSWORD = "";

      private static Connection connection = null;

     public static Connection getConnection() {

        try {

            if (connection == null || connection.isClosed()) {

                Class.forName("com.mysql.cj.jdbc.Driver");

                connection = DriverManager.getConnection(URL, USER, PASSWORD);
```

```
            System.out.println("✅ Database connected successfully!");
        }
        } catch (ClassNotFoundException e) {
            System.err.println("❌ MySQL JDBC Driver not found!");
            e.printStackTrace();
        } catch (SQLException e) {
            System.err.println("❌ Database connection failed!");
            e.printStackTrace();
        }
        return connection;
    }
}
```

### 4.2 User Authentication Code

```
// UserDAO.java - User Login Authentication
import java.sql.*;
public class UserDAO {
    public static User authenticateUser(String email, String password) {
        Connection conn = DatabaseConnection.getConnection();
```

```
        String query = "SELECT * FROM users WHERE email = ? AND password =
?";



    try {
        PreparedStatement pst = conn.prepareStatement(query);
        pst.setString(1, email);
        pst.setString(2, password);
         ResultSet rs = pst.executeQuery();
         if (rs.next()) {
           User user = new User();
           user.setUserId(rs.getInt("user_id"));
           user.setUsername(rs.getString("username"));
           user.setEmail(rs.getString("email"));
           user.setRole(rs.getString("role"));
           return user;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
```

```
        return null;
    }
}
```

### 4.3 Resume Verification Algorithm

```
// Verification Logic - Email Detection using Regex
private boolean checkEmail(String text) {
 Pattern emailPattern = Pattern.compile(
"[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}"
    );
    Matcher matcher = emailPattern.matcher(text);
    return matcher.find();
}
// Keyword Counting Algorithm
private int countKeywords(String text, String[] keywords) {
    text = text.toLowerCase();
    int count = 0;
    for (String keyword : keywords) {
        if (text.contains(keyword.toLowerCase())) {
            count++;
```

```
        }
    }
    return count;
}
```

### 4.4 Saving Resume to Database

```
// ResumeDAO.java - Saving Resume Data
public static int saveResume(Resume resume) {
    Connection conn = DatabaseConnection.getConnection();
    String query = "INSERT INTO resumes (user_id, file_name, file_path, " +
                   "resume_text, word_count) VALUES (?, ?, ?, ?, ?)";
try {
        PreparedStatement pst = conn.prepareStatement(query,
                                Statement.RETURN_GENERATED_KEYS);
        pst.setInt(1, resume.getUserId());
   pst.setString(2, resume.getFileName());
        pst.setString(3, resume.getFilePath());
        pst.setString(4, resume.getResumeText());
        pst.setInt(5, resume.getWordCount());
          int rowsAffected = pst.executeUpdate();
```

```
        if (rowsAffected > 0) {
          ResultSet rs = pst.getGeneratedKeys();
          if (rs.next()) {
              return rs.getInt(1); // Return generated resume_id
          }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return -1;
}
```

### 4.5 Score Calculation Logic

```
// Scoring Algorithm
private VerificationResult performVerification(String resumeText) {
    VerificationResult result = new VerificationResult();
    int score = 20; // Base score


    // Word count scoring (max 15 points)
    int wordCount = resumeText.split("\\s+").length;
```

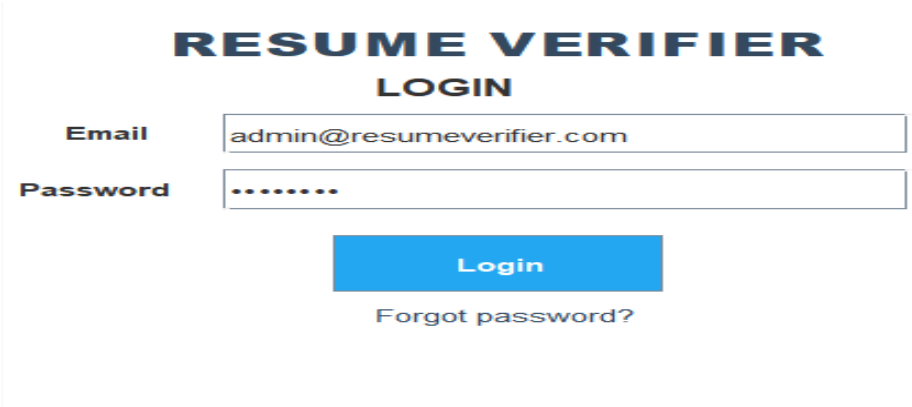
```
if (wordCount > 500) score += 15;

else if (wordCount > 300) score += 10;

else if (wordCount > 150) score += 5;

 // Email and Phone (10 points each)

if (checkEmail(resumeText)) score += 10;

if (checkPhone(resumeText)) score += 10;

// Education keywords (15 points)

String[] educationKeywords = {"B.Tech", "M.Tech", "University",

                              "Degree", "Engineering"};

int eduCount = countKeywords(resumeText, educationKeywords);

if (eduCount > 0) score += 15;

 // Skills (20 points)

String[] skillKeywords = {"Java", "Python", "SQL", "MySQL"};

int skillCount = countKeywords(resumeText, skillKeywords);

if (skillCount >= 5) score += 20;

else if (skillCount >= 3) score += 10;



score = Math.min(score, 100); // Cap at 100
```

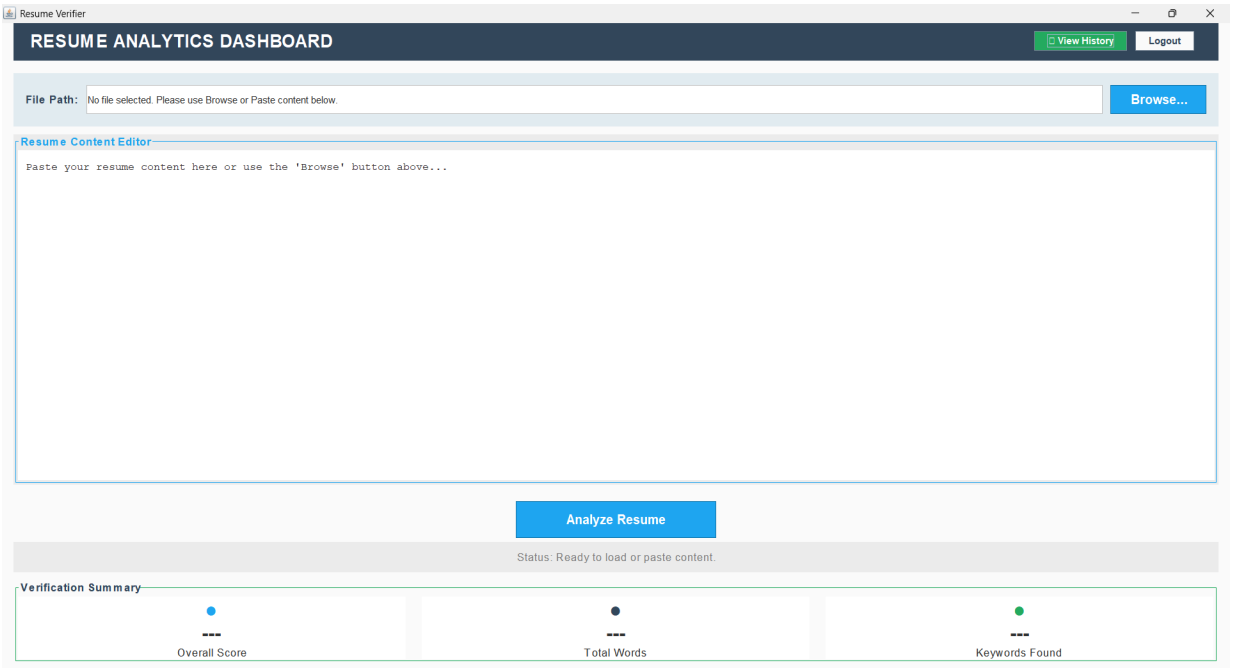
```
    result.setOverallScore(score);

    return result;

}
```

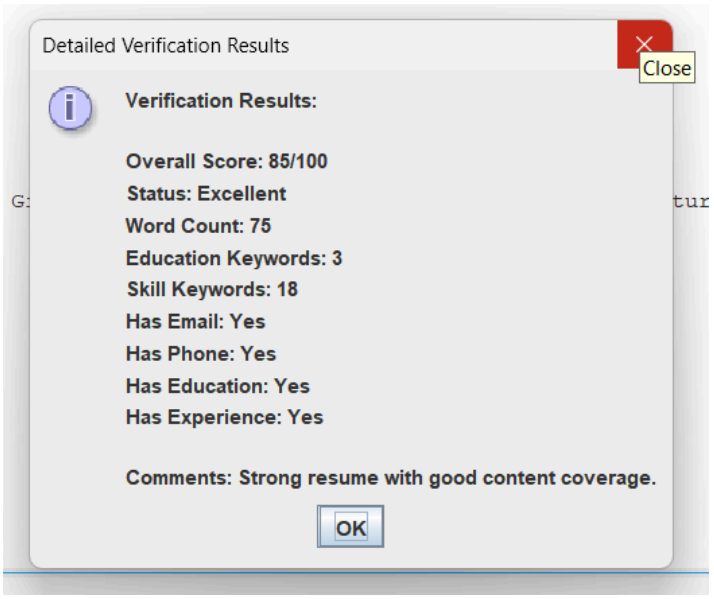
# CHAPTER 5

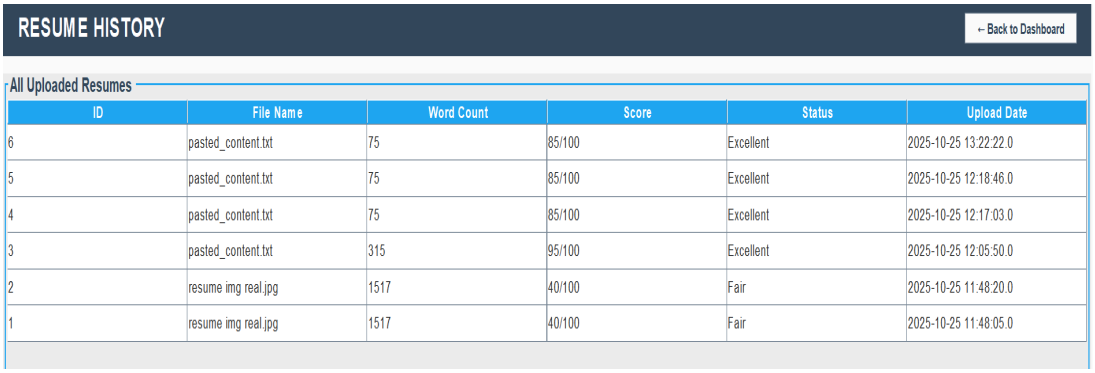
## SCREEN SHOTS

**Fig 5.1 Introduction page**






| ID | File Name | Word Count | Score | Status | Upload Date |
|---|---|---|---|---|---|
| 6 | pasted_content.txt | 75 | 85/100 | Excellent | 2025-10-25 13:22:22.0 |
| 5 | pasted_content.txt | 75 | 85/100 | Excellent | 2025-10-25 12:18:46.0 |
| 4 | pasted_content.txt | 75 | 85/100 | Excellent | 2025-10-25 12:17:03.0 |
| 3 | pasted_content.txt | 315 | 95/100 | Excellent | 2025-10-25 12:05:50.0 |
| 2 | resume img real.jpg | 1517 | 40/100 | Fair | 2025-10-25 11:48:20.0 |
| 1 | resume img real.jpg | 1517 | 40/100 | Fair | 2025-10-25 11:48:05.0 |

# CHAPTER 6

## CONCLUSION AND FUTURE ENHANCEMENT

The Resume Verification System facilitates automated resume validation using keyword and field detection. It simplifies HR and academic screening processes and ensures consistent, data-driven evaluation.

Future Enhancements:

- Integration with PDF/Text extractors.
- AI/NLP-based scoring.
- Direct HR portal integration.
- Export reports to PDF.

# REFERENCES


- **Database System Concepts** — authors: Abraham Silberschatz, Henry F. Korth, S. Sudarshan. mpgcamb.com+2db-book.com+2

- **JDBC API Tutorial and Reference** — authors: Maydene Fisher, Jon Ellis, Jonathan Bruce. Oracle+2informit.com+2

- **Beginning Java 8 APIs, Extensions and Libraries: Swing, JavaFX, JavaScript, JDBC and Network Programming APIs** — author: Kishori Sharan. O'Reilly Media+2Apple+2

- **Swing** — authors: Matthew Robinson and Pavel Vorobiev. Javaranc

- https://www.geeksforgeeks.org/jdbc-connectivity-in-java/

- https://www.w3schools.com/sql/