AI-Based Resume and Portfolio Builder

APROJECT REPORT

Submitted by

Aishwary Singh (23BCS11568)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

Chandigarh University

Nov, 2025

BONAFIDE CERTIFICATE

Certified that this project report "**AI Based Resume And Portfolio Builder"** is the bonafide work of **"Aishwary Singh"** who carried out the project work under my supervision.

SIGNATURE

Dr. Sandeep Singh Kang

BATCH HEAD

BE-CSE

SIGNATURE

SUPERVISOR

BE-CSE

TABLE OF CONTENTS

Name:Aishwary Singh UID: 23BCS11568
Branch: BE-CSE Section/Group: KRG-3B
Subject name: Full Stack - 1 Subject code: 23CSH-339

1. PROJECT NAME

AI-Based Resume and Portfolio Builder

2. REFERENCE WEBSITE LINK

https://spring.io – Official Spring Boot Documentation

https://react.dev – React Frontend Documentation

https://postgresql.org – PostgreSQL Database Documentation

https://github.com – For project hosting and version control

3. PROJECT DESCRIPTION

The **AI-Based Resume and Portfolio Builder** is a **full-stack web application** designed to help students and professionals **create, manage, and track versions** of their resumes in one centralized platform.

Unlike traditional static resume builders, this system introduces **intelligent version management**, ensuring that users never lose old data while continuously improving their profiles.

Whenever a user edits their resume, the application automatically generates a **new version** and stores it in a **PostgreSQL database** with details like timestamp, resume ID, and content snapshot. Users can later view, compare, or restore any earlier version seamlessly.

Tech Stack Overview

- **Frontend:** React.js, Axios, Tailwind CSS
- **Backend:** Spring Boot (Java 17), Spring Data JPA, RESTful APIs
- **Database:** PostgreSQL
- **Tools Used:** IntelliJ IDEA, Postman, GitHub, Render.com

Key Modules

1. **User Authentication Module** – Handles registration and login with JWT.
2. **Resume Management Module** – Enables creating and editing resumes.
3. **Version Tracking Module** – Automatically stores versions with timestamps.

4. **Version History Module** – Displays all stored versions to the user.
5. **Restoration Module** – Allows reverting to older versions.

PROBLEM STATEMENT

Most existing resume builders only allow users to create or edit a single copy of their resume. Each new edit **overwrites the previous version**, resulting in the **loss of old data**.

This project aims to address these problems:

- **No version history:** Users can't revisit or restore previous resumes.
- **Data loss:** Every new edit replaces old content.
- **Lack of modularity:** No separation between frontend and backend.

The **proposed solution** is to implement an **automated version tracking mechanism** that saves every modification in real time, offering flexibility and transparency to the user.

5. OBJECTIVES OF THE PROJECT

1. To design a **version-controlled resume management system**.
2. To integrate **Spring Boot REST APIs** with a **React.js frontend**.
3. To implement **CRUD functionality** for resume and version entities.
4. To maintain **data integrity** using JPA relationships.
5. To provide an **intuitive user interface** for viewing and restoring past versions.
6. To demonstrate **real-world full-stack development** practices.

HIGH LEVEL DESIGN

Frontend Design

- Built with **React.js** using reusable components such as:
  - ResumeEditor.jsx
  - VersionList.jsx
  - VersionCard.jsx
- **Axios** handles API requests to the backend.
- **Tailwind CSS** provides a responsive layout for different devices.
- Data is fetched dynamically and updated without page reloads.

Backend Design

- Built using **Spring Boot 3.x**, following layered architecture:
    1. **Controller Layer** – Manages API endpoints.
    2. **Service Layer** – Contains business logic.
    3. **Repository Layer** – Communicates with PostgreSQL using JPA.
    4. **Entity Layer** – Defines database schema models.

React Frontend

↓

Spring Boot Controller

↓

Service Layer

↓

Repository (JPA)

↓

PostgreSQL Database

4. Key Features

User-Oriented Features

- Create, update, and delete resumes.
- Automatically save new versions with every modification.
- View complete version history in chronological order.
- Restore old versions at any time.
- Responsive UI with clear data representation.

Technical Features

- **Spring Boot RESTful APIs** for seamless communication.

- **JPA Repository** with custom methods like:
  `List<ResumeVersion> findByResumeOrderByCreatedAtDesc(Resume resume);`
- Uses **JSON format** for frontend-backend data exchange.
- Secure user access through JWT authentication.
- Scalable and modular microservice-ready structure.

SYSTEM WORKFLOW

1. The user creates or edits a resume on the **React frontend**.
2. A **POST request** is sent to the backend API endpoint (e.g. `/api/resume/save`).
3. The **Controller** processes the request and passes it to the **Service** layer.
4. The **Service** validates and processes business logic, then calls the **Repository**.
5. The **Repository** executes SQL queries via JPA and saves data in **PostgreSQL**.
6. The backend responds with JSON data containing all resume versions.
7. The frontend updates the **Version History** list dynamically.

API Endpoints

- `POST /api/resume` → Save or update a resume.
- `GET /api/resume/{id}/versions` → Fetch all versions of a resume.
- `POST /api/resume/{id}/restore/{versionId}` → Restore specific version.

DATABASE SCHEMA

Entity Relationship

Tables:

- `user`
- `resume`
- `resume_version`

Relationships:

- One **User** → **Many Resumes**
- One **Resume** → **Many ResumeVersions**

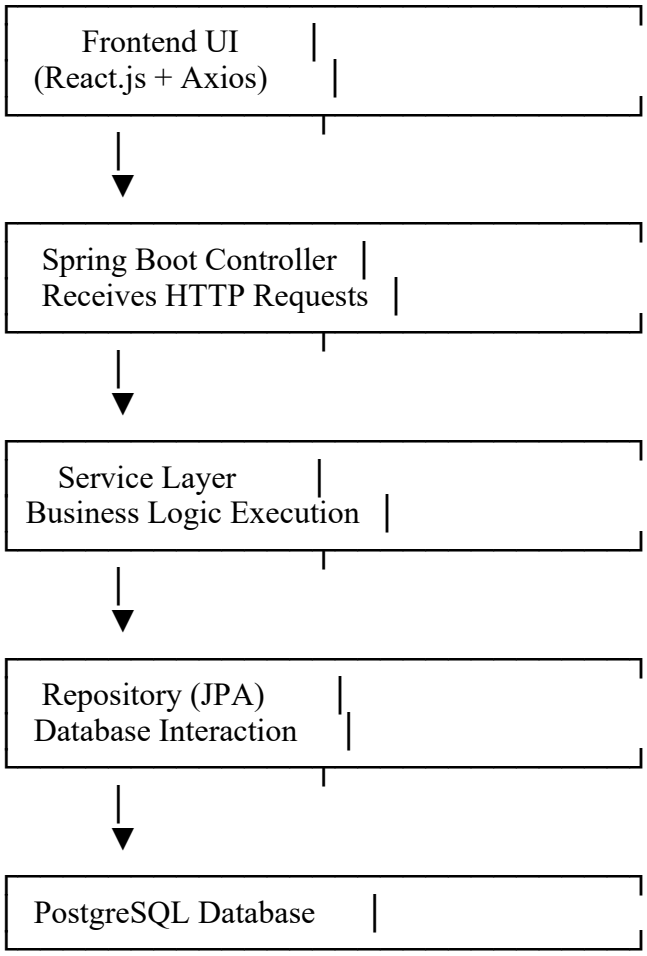
ResumeVersion Entity (Example)

Field	Type	Description
id	Long	Primary Key

Field	Type	Description
resume_id	Long	Foreign Key (Resume)
content	Text	Resume data
created_at	Timestamp	Version creation time

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

FLOWCHART



FUTURE SCOPE

1. Integrate **AI-based resume scoring and suggestions** using NLP.
2. Add **PDF export and LinkedIn synchronization**.
3. Implement **real-time collaborative editing** using WebSockets.
4. Integrate **cloud storage (AWS S3 / Firebase)** for version files.
5. Extend functionality for **portfolio projects and cover letters**.
6. Add **dark mode UI** and analytics dashboard for user insights.
7. Mobile-friendly version using **React Native**.

CONCLUSION

The **Resume Version Management** system represents an innovative step toward smarter and more reliable resume building.
By combining **React.js** and **Spring Boot**, it ensures efficient data flow, scalability, and user-centric design.

This project demonstrates:

- The power of **modular full-stack architecture**.
- The benefits of **automated version control** in web applications.
- Effective **database design and API integration**.

It serves as a practical example of how modern web technologies can simplify real-world challenges, while showcasing the skills required for full-stack development — backend logic, frontend integration, and database management.

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING