A FIELD PROJECT REPORT ON

**“Resume Architect”**

Submitted in partial fulfillment of the requirements for the award of the degree

**BACHELOR OF TECHNOLOGY**

In

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by

**G.Sivanaga Pallavi (231FA04286)**

**K.Naga Sruthika (231FA04303)**

**T.Dinakar (231FA04377)**  
**P.Sahithi (231FA04G09)**



**Department of Computer Science and Engineering**

*School of Computing and Informatics*

**Vignan’s Foundation for Science, Technology & Research**

(Deemed to be University)

Vadlamudi,Guntur,Andhra Pradesh-522213, India

March-**2025**



**CERTIFICATE**

This is to certify that the field project entitled **“Resume Architect”** being submitted by G.Sivanaga Pallavi(231FA04286), K.Naga Sruthika(231FA04303), T.Dinakar (231FA04377),P.Sahithi (231FA04G09) in partial fulfilment of Bachelor of Technology in the department of Computer Science and Business Systems, Department of Computer Science and Engineering, Vignan’s Foundation For Science Technology and Research (Deemed to be University), Vadlamudi, Guntur District, Andhra Pradesh, India, is a bonafide work carried out by them under my guidance and supervision.

**Head of the department Guide**

# 

# DECLARATION

We hereby declare that our project work described in the project titled **“Resume Architect”** which is being submitted by us for the partial fulfilment in the department of CSE, Vignan’s Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, and the result of investigations are carried out by us under the guidance of Ms.V.Nandhini.

G.Sivanaga Pallavi (231FA04286)

K.Naga Sruthika (231FA04303)

T.Dinakar (231FA04377)  
P.Sahithi (231FA04G09)

Content

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter No. |  | Description | Page no. |
| 1 |  | Introduction | 06 |
|  | 1.1 | Problem Statement | 07 |
|  | 1.2 | Objectives | 07-08 |
|  | 1.3 | Literature Review | 08-09 |
| 2 |  | Methodology | 09-10 |
|  | 2.1 | Agile Development Model | 09 |
|  | 2.2 | Research and Analysis | 09 |
|  | 2.3 | Tools Used for Methodology Implementation | 10 |
|  | 2.4 | Workflow | 10 |
| 3 |  | Software Requirements | 10 |
|  | 3.1 | Hardware Requirements | 11 |
| 4 |  | System Design | 11-13 |
|  | 4.1 | System Architecture | 13-15 |
|  | 4.2 | Module Description | 15-19 |
|  | 4.3 | UML Diagrams | 19-20 |
| 5 |  | Implementation | 20-43 |
|  | 5.1 | Testing and Validation | 43-45 |
|  | 5.2 | Results and Discussion | 45-47 |
|  | 5.3 | Screenshots | 47-50 |
| 6 |  | Future Enhancements | 51-53 |
|  | 6.1 | Limitations | 53-56 |
|  | 6.2 | Conclusion | 56-57 |

ABSTRACT:

This project details the development of a user-friendly, web-based resume architect, designed to guide individuals through a structured, step-by-step process of creating compelling and effective resumes. Recognizing the challenges many faces in articulating their skills and experiences, this platform aims to simplify resume creation by breaking down the process into manageable, intuitive stages.

**1.INTRODUCTION:**

Overview of Resume Architect:

Describe the purpose of the platform, which is to assist users in creating professional resumes tailored to their career goals.

Explain the significance of having a well-structured resume in today’s job market.

Importance of Resumes:

Discuss the role of resumes in job applications and career advancement.

Highlight statistics on how employers use resumes in the hiring process.

Target Audience:

Identify the primary users (students, professionals, job seekers) and their specific needs.

Discuss the diversity of users and how the platform caters to different career stages.

**1.1 PROBLEM STATEMENT**

Most traditional resume-building tools are either too simplistic or overly complex, often requiring payments for premium features. Users without technical skills struggle to align content properly or apply attractive templates. The lack of a structured and intuitive tool to build resumes often results in poorly formatted documents that fail to leave a strong impression on employers. There is a need for a platform that simplifies resume creation while maintaining professionalism and aesthetic design. This project addresses the need by providing an easy-to-use, accessible, and efficient resume builder

**OBJECTIVES**

* Provide a guided interface for step-by-step resume creation.
* Offer customizable templates suitable for different domains (IT, Marketing, Education, etc.).
* Ensure user data privacy and easy retrieval using secure login credentials.
* Allow export in multiple formats (PDF, Print).
* Integrate a live preview feature that updates resume design in real-time.
* Design for accessibility and responsiveness across all devices.
* Enable quick edits and multiple resume versions per user account
* Offer AI-powered suggestions to help users improve wording and layout.
* Integrate analytics tools to provide insights on resume download frequency and performance.
* Design for accessibility and responsiveness across all devices.
* Enable quick edits and multiple resume versions per user account.

## LITERATURE REVIEW

The literature review highlights existing tools, platforms, and academic studies that form the foundation for building a resume creation platform.

### Existing Platforms

* **Canva**: Known for its design flexibility but lacks backend resume parsing and user-specific analytics.
* **Zety**: A guided resume builder with modern templates but requires subscription for PDF download.
* **Novoresume**: Focused on design-centric resumes with career advice integration.

These platforms demonstrate the need for resume builders that not only offer templates but also guide users in content creation and optimization.

### Research Papers & Studies

1. **Online Resume Builders: A Comparative Study (2020, IEEE)**
   * Evaluated multiple resume builders and their usability scores.
   * Identified the need for intelligent assistance during resume writing.
2. **Design Patterns in Web-based Resume Systems (ACM, 2019)**
   * Highlighted standard architectural approaches and challenges in UI responsiveness and scalability.
3. **Personalization in Resume Generators using NLP (Elsevier, 2021)**
   * Discussed how NLP can improve content suggestions and keyword optimization.
4. **User Experience in Interactive Resume Systems (HCI Conference, 2022)**
   * Emphasized real-time feedback and visual previews as top features desired by users.

## METHODOLOGY

The development of the Resume Architect project followed a structured software development methodology to ensure systematic progress, timely delivery, and high quality.

### 5.1 Agile Development Model

The Agile methodology was chosen for its iterative development approach, allowing regular feedback, testing, and refinements. The development lifecycle was divided into sprints, each focusing on a set of features:

* **Sprint 1**: Requirement gathering and wireframe design
* **Sprint 2**: Frontend development and template structuring
* **Sprint 3**: Backend APIs, authentication, and database design
* **Sprint 4**: Integration of frontend with backend
* **Sprint 5**: Testing, debugging, and user interface enhancements
* **Sprint 6**: Deployment, documentation, and feedback incorporation

### 5.2 Research and Analysis

Initial stages involved:

* Surveying users to identify common challenges in resume creation
* Reviewing UI/UX best practices from design-centered applications
* Studying job board requirements for resume formats

### 5.3 Tools Used for Methodology Implementation

* **Trello** for sprint tracking and task assignment
* **Git** and **GitHub** for version control
* **Figma** for initial UI wireframing
* **Postman** for testing backend APIs

### 5.4 Workflow

1. **Planning**: Defining the scope and MVP (Minimum Viable Product)
2. **Design**: Wireframing the pages and designing modular components
3. **Development**: Building in short cycles with continuous integration
4. **Testing**: Manual and automated testing of features
5. **Review**: Team and mentor reviews after each sprint
6. **Release**: Deployment on a public server

**SOFTWARE REQUIREMENTS**

* OS: Windows/Linux/MacOS
* Frontend: HTML, CSS, JS, Bootstrap, jQuery
* Backend: Node.js, Express.js
* Database: MongoDB (Mongoose ORM)
* Dev Tools: GitHub, Postman, VS Code, Canva
* Browser Support: Chrome, Firefox, Safari, Edge
* PDF Generator: jsPDF with HTML2Canvas

**HARDWARE REQUIREMENTS**

* Minimum: Intel Core i3, 4 GB RAM, 256 GB HDD
* Recommended: Intel i5 or i7, 8+ GB RAM, SSD for faster processing

**SYSTEM DESIGN**

The system design phase lays out the blueprint for how the Resume Architect application functions at a technical level. It includes both the high-level architectural view and the low-level component design, ensuring the solution is scalable, maintainable, and user-centric.

8.1 Design Approach

The application follows a modular and layered architecture:

Presentation Layer (Frontend): Responsible for UI rendering and user interactions using HTML, CSS, JavaScript, and Bootstrap.

Business Logic Layer (Backend): Contains the application logic, validation, user session handling, and template rendering using Node.js and Express.js.

Data Layer (Database): MongoDB stores user profiles, resume data, template configurations, and login credentials.

### 8.2 Entity Relationship Diagram (ERD)

**Entities**:

* User: userId, email, password, createdAt
* Resume: resumeId, userId, personalInfo, education, experience, skills, projects, templateId, createdAt
* Template: templateId, name, layout, styleAttributes

**Relationships**:

* A User can have multiple Resumes
* Each Resume uses one Template

### 8.3 Component design

#### a. **Authentication Module**

* Handles login, registration, password hashing using bcrypt
* JWT used for secure token-based session management

#### b. **Resume Form Module**

* Form structured into steps: Personal Info → Education → Experience → Skills → Projects
* Form validation handled client-side and server-side

#### c. **Template Engine**

* Dynamically generates resume preview using pre-defined layouts
* Modular CSS classes to support easy theming and customization

#### d. **PDF Generator**

* Uses libraries like html-pdf, puppeteer, or jspdf
* Converts HTML content into downloadable PDF format

### 8.5 Security Design

* **Password encryption**: bcrypt hashing
* **Session security**: JWT tokens stored in HTTP-only cookies
* **Form protection**: Input sanitization to prevent XSS/SQL Injection
* **Rate limiting**: API throttling to prevent brute-force attacks

### 8.6 UI/UX Design Considerations

* Clean and minimal design to reduce cognitive load
* Step-by-step input forms to guide the user journey
* Real-time preview of resume changes
* Mobile responsiveness for on-the-go resume editing

## SYSTEM ARCHITECTURE

The system architecture of **Resume Architect** follows a **3-tier web architecture**, which separates the application into three logical layers: **Client**, **Server**, and **Database**. This modular structure allows better scalability, maintainability, and security.

### 9.1 Layers Description

#### 🔹 1. Client Tier (Presentation Layer)

* Built using **HTML, CSS, JavaScript, and Bootstrap**
* Includes:
  + Dynamic forms for user inputs
  + Template selection interface
  + Preview and PDF download options
* All frontend operations communicate via REST APIs to the backend

#### 🔹 2. Application Tier (Logic Layer)

* Built using **Node.js with Express.js**
* Responsible for:
  + User authentication and session management
  + Form data validation
  + Resume generation and template rendering
  + Serving PDF export requests
* Middleware for logging, error handling, and security

#### 🔹 3. Data Tier (Database Layer)

* NoSQL database: **MongoDB**
* Stores:
  + User credentials (securely hashed)
  + Resume data in JSON-like structure
  + Templates and styling presets

### 9.2 Communication Flow

1. **User Registration/Login**
   * Frontend collects credentials
   * Sends POST request to backend API
   * Backend authenticates using JWT
   * Returns a token to frontend
2. **Resume Creation**
   * Form data is entered step-by-step
   * Each step stores data in local state or session
   * Final data is submitted to /api/resumes endpoint
3. **Template Rendering**
   * Template ID selected by the user
   * Backend uses a layout engine to merge data + template
   * Sends HTML output or PDF back to the client
4. **PDF Generation**
   * Backend uses libraries like puppeteer or jspdf
   * Generates PDF from rendered HTML
   * Sends file back to user for download

### 9.3 Technologies Used in Architecture

Component Technology

Frontend HTML, CSS, JS, Bootstrap

Backend Node.js, Express.js

Database MongoDB (NoSQL)

Deployment Render / Netlify / MongoDB Atlas

API Testing Postman

Version Control Git + GitHub

Design Canva, Figma

## MODULE DESCRIPTION

The Resume Architect application is modular, with each feature implemented as an independent and reusable component. This modular approach ensures clean architecture, easier debugging, and scope for future enhancement.

### 10.1 User Authentication Module

**Purpose**: To securely register, log in, and manage user sessions.

**Features**:

* User registration with email and password
* Secure login using hashed credentials
* JWT-based token authentication
* Logout and session expiration

**Technologies**:

* Express.js
* bcrypt.js for hashing passwords
* JSON Web Tokens (JWT)

### 10.2 Resume Input Module

**Purpose**: To collect and manage all user-provided resume content.

**Sections Included**:

* Personal Details (Name, Contact, Address)
* Education History
* Work Experience
* Technical & Soft Skills
* Certifications and Projects
* Achievements and Languages

**Features**:

* Step-by-step form with validation
* Form auto-save (local/session storage)
* Progress bar and navigation

### 10.3 Template Manager Module

**Purpose**: Allow users to preview and select resume templates.

**Features**:

* Display list of available templates with previews
* Store template choice in DB for rendering
* Toggle between templates at any point

**Future Scope**:

* Template customization (font size, color, layout)
* AI template recommendations based on role/field

### 10.4 Resume Renderer Module

**Purpose**: Dynamically combine user data and selected template into a formatted resume view.

**Process**:

* Extract user resume data from DB
* Load template layout and placeholders
* Inject content into placeholders
* Display final resume as HTML

**Libraries**:

* Handlebars / EJS (for templating)
* Dynamic CSS rendering for styling

### 10.5 PDF Generator Module

**Purpose**: Allow users to export resumes as printable PDF files.

**Approach**:

* Use Puppeteer or html-pdf to convert HTML to PDF
* Maintain styling and layout integrity
* Auto-download feature or email to user

**Output**:

* High-quality PDF resume
* Optional print option from browser

### 10.6 Admin / Template Management Module (Optional / Future Feature)

**Purpose**: To allow admin to upload, manage, or update templates dynamically.

**Functions**:

* CRUD operations on templates
* Preview and testing before publishing
* Add metadata (e.g., category, layout type)

### 10.7 API Module

**Purpose**: Manage RESTful APIs for client-server communication.

**Endpoints**:

* POST /api/register: Register a new user
* POST /api/login: Authenticate user
* POST /api/resume: Save user resume
* GET /api/resume/:id: Fetch user resume
* GET /api/templates: Fetch available templates
* POST /api/download: Export resume to PDF

**Middleware**:

* Authentication middleware (JWT)
* Error handler middleware
* API rate limiter (optional)

### Analytics Module (Advanced / Optional)

**Purpose**: Collect usage data to improve user experience.

**Metrics**:

* Most used templates
* Average time spent building a resume
* Frequent sections skipped

**Tech Stack**:

* Google Analytics / Custom event logging
* MongoDB collections for user activity

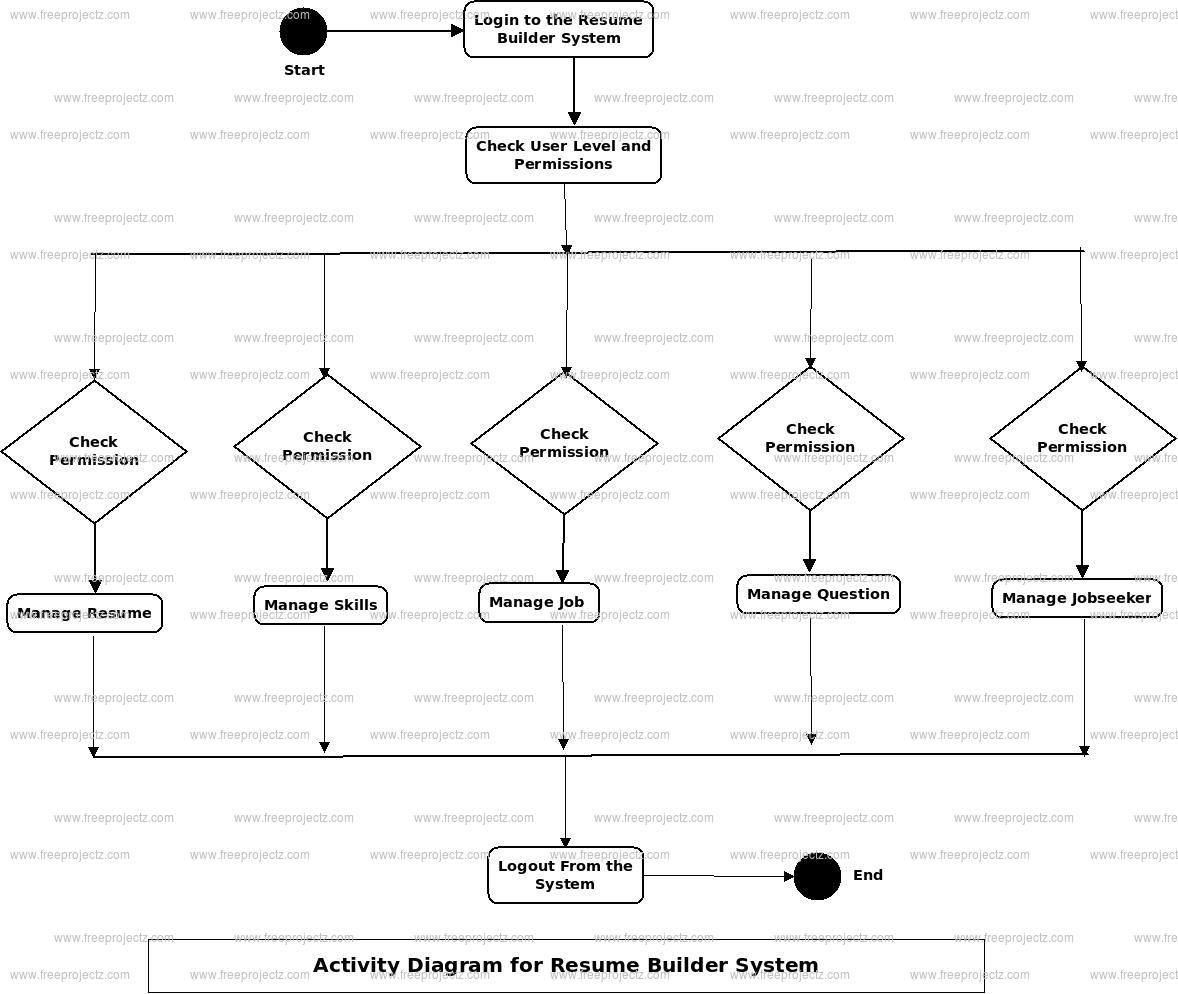
### User Feedback Module (Optional)

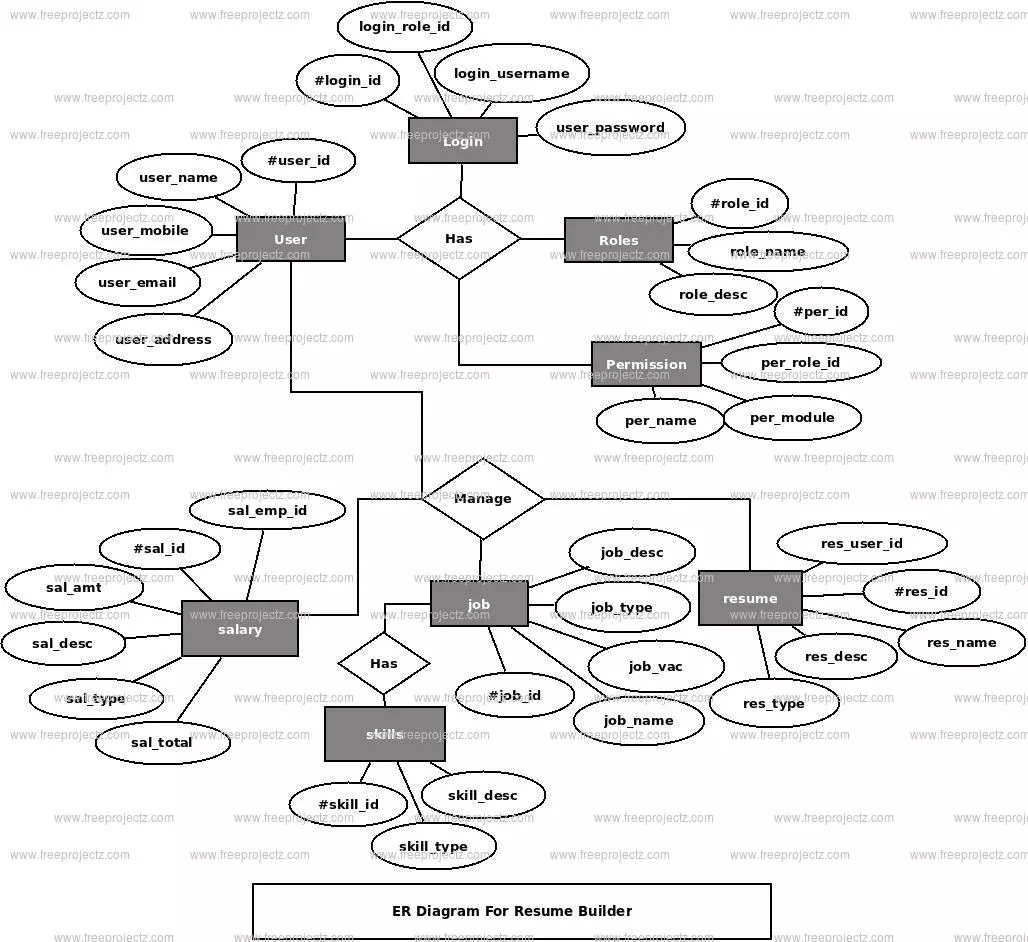
**Purpose**: Allow users to rate their experience and suggest improvements.

**Components**:

* Feedback form on dashboard
* Rating system (stars/comments)
* Admin view of user insights

**4.3 UML Diagrams :**

****



## IMPLEMENTATION

The implementation phase of Resume Architect involved translating the planned architecture and modules into a fully functional web application using modern web technologies. This section outlines the actual development work, libraries used, code structure, and examples of how features were built.

11.1 Development Stack Recap

Layer Technology Used

Frontend HTML, CSS, JavaScript, Bootstrap

Backend Node.js, Express.js

Database MongoDB Atlas

### 11.6 Deployment Process

* Frontend hosted on **Netlify**
* Backend deployed on **Render**
* Database hosted on **MongoDB Atlas**
* Environment variables handled using .env files

### 11.7 Security Measures

* **Password hashing**: bcrypt
* **Token-based access**: JWT stored in HTTP-only cookies
* **API rate limiting**: (Optional with express-rate-limit)
* **Input sanitization** to prevent XSS

Code:

**FrontEnd:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>User Form</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: greenyellow;

display: flex;

height: 100vh;

justify-content: center;

align-items: center;

margin: 0;

}

.form-container {

background: white;

padding: 30px;

border-radius: 10px;

box-shadow: 0 5px 15px rgba(0,0,0,0.1);

width: 300px;

}

input {

display: block;

width: 100%;

padding: 10px;

margin: 10px 0;

border: 1px solid #ccc;

border-radius: 5px;

}

button {

padding: 10px 20px;

background: greenyellow ;

color: white;

border: none;

cursor: pointer;

border-radius: 5px;

}

button:hover {

background: green; /\* Green on hover \*/

}

</style>

</head>

<body>

<div class="form-container">

<h2>Enter Your Details</h2>

<form id="userForm">

<input type="text" id="name" placeholder="Enter your name" required />

<input type="email" id="email" placeholder="Enter your Gmail" required />

<button type="submit">Submit</button>

</form>

</div>

<script>

document.getElementById("userForm").addEventListener("submit", async function (e) {

e.preventDefault();

const name = document.getElementById("name").value;

const email = document.getElementById("email").value;

const response = await fetch("http://localhost:3000/api/submit", {

method: "POST",

headers: {

"Content-Type": "application/json",

},

body: JSON.stringify({ name, email }),

});

if (response.ok) {

alert("Details submitted successfully!");

window.location.href = "resumefile.html";

} else {

alert("Failed to submit data.");

}

});

</script>

</body>

</html>

**Resume File.html:**

<!DOCTYPE html>

<html>

<head>

<title>Resume and Cover Letter Tips</title>

<link rel="icon" href="" type="image/x-icon">

<style>

.header {

background-color: rgb(225, 62, 62);

height: 70px;

display: flex;

align-items: center;

justify-content: space-between;

padding: 0 20px;

}

.logo {

height: 50px;

margin-left: 20px;

}

.back-button {

background-color: white;

border: 1px solid black;

font-size: 20px;

height: 42px;

width: 100px;

border-radius: 10px;

cursor: pointer;

transition: background-color 0.3s, color 0.3s;

}

.back-button:hover {

background-color: black;

color: white;

}

</style>

</head>

<body>

<div class="header">

<img src="resume gif.gif" alt="resume gif">

<a href="index.html"> <button class="back-button" onclick="history.back()">Back</button> </a>

</div>

<div style="display: flex; align-items: center; justify-content: space-between; max-width: 1200px; margin: 50px auto; padding: 20px; font-size: 60px;">

<div style="font-size: 50px; font-weight: bold; margin-left: 30px;">

Kickstart Your <br>

Career with <br>

Powerful Resume <br>

and Cover Letter!

</div>

<div>

<img src="c:\Users\thupa\OneDrive\Pictures\Documents\pallavi\_webpage[1]\pallavi webpage\resume gif.gif" alt="resume gif" style="width: 700px; height: 300px; border-radius: 10px; margin-top: 60px;">

</div>

</div>

<p style="margin-left:150px;font-size:25px;margin-right:30px">A resume is a short and concise summary of your work experience and education. During recruitment, employers or recruiters compare tens, hundreds, and sometimes even thousands of resumes with each other. All this to find the best candidates.</p>

<p style="font-weight:bold;margin-left:150px;font-size:25px;margin-right:30px">You only have one chance to make an electrifying impression. Here's what you should do, step by step:</p>

<p style="margin-left:150px;font-size:30px;">STEP 1. Choose the right resume format for you</p>

<p style="margin-left:150px;font-size:20px;margin-right:30px">There are three types of resumes. Depending on your professional experience and the industry you are applying to,<br> one of them will be the best for you.Are you applying for multiple positions in various companies or industries?<br> You can consider more than one resume format.</p>

<ul style="margin-left:150px;font-size:23px;">

<li>

<p style="font-size:23px;margin-right:30px"><strong>Reverse Chronological (Most Common):</strong> Lists work experience from most recent to oldest.</p></li>

<li>

<p style="font-size:23px;margin-right:30px"><strong>Functional:</strong> Focuses on skills rather than experience (good for career changers).</p></li>

<li>

<p style="font-size:23px;margin-right:30px"><strong>Combination: </strong>Mixes both chronological and functional styles.</p></li></ul>

<p style="margin-left:150px;font-size:30px;">Step 2: Add Contact Information</p>

<p style="margin-left:150px;font-size:20px;margin-right:30px">At the top of your resume, include:<br>

<ul style="margin-left:150px;font-size:20px;">

<li> Your Full Name</li>

<li> Your Phone Number</li>

<li>Your Email Address (professional one)</li>

<li>Your LinkedIn Profile (optional but recommended)</li>

<li> Your GitHub or Portfolio (if applicable)</li></ul></p>

<p style="margin-left:150px;font-size:30px;">Step 3: Write a Strong Summary or Objective</p>

<ul style="margin-left:150px;font-size:20px;">

<li> <p style="font-size:20px;margin-right:30px"><strong>Summary (if you have experience):</strong> A brief statement highlighting your skills and experience.</li></p>

<li> <p style="font-size:20px;margin-right:30px"> <strong>Objective (if you’re new to the industry):</strong> A short statement about your career goals and what you bring to the table.</li></ul></p>

<p style="margin-left:150px;font-size:30px;">Step 4: List Your Work Experience (if applicable)</p>

<p style="margin-left:150px;font-size:23px;">Include:</p>

<ul style="margin-left:150px;font-size:23px;">

<li>Job Title – Company Name, Location</li>

<li> Dates of Employment (Month & Year)</li>

<li> Key Responsibilities & Achievements (use bullet points and action verbs)</li></ul></p>

<p style="margin-left:150px;font-size:30px;">Step 5: Highlight Your Education</p>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">Make a list of the educational institutions you attended. Start with the most recent. If you are starting your professional career or have not graduated from university, provide information about the completed High School.<br>

Add any other educational experiences, such as training programs, community college or summer courses, seminars, and so on.</p>

<p style="margin-left:150px;font-size:30px;">Step 6: Showcase Your Skills</p>

<ul style="margin-left:150px;font-size:23px;">

<li>Technical Skills (Programming languages, tools, frameworks)</li>

<li>Soft Skills (Communication, teamwork, problem-solving)</li>

<li>Certifications (if relevant)</li></ul>

<p style="margin-left:150px;font-size:30px;">Step 7: Format & Proofread</p>

<ul style="margin-left:150px;font-size:23px;">

<li>Keep it 1 page (unless you have a lot of experience).</li>

<li>Use clear fonts like Arial, Calibri, or Times New Roman.</li>

<li>Keep margins & spacing consistent.</li>

<li>Proofread for grammar and spelling errors.</li></ul>

<br><br><br>

<style>

.image-container {

display: flex;

gap: 10px;

}

.image-container img {

width: 200px;

height: auto;

border-radius: 10px;

}

.image:hover{

transform: scale(1.03);

transition: transform 0.3s ease-in-out;

box-shadow: 0 10px 20px rgba(0, 0, 0, 0.658);

}

</style>

</head>

<body>

<div class="image-container">

<img style="margin-left: 30px;width:470px;height:750px;box-shadow: 2px 2px 2px black;"src="chronological.png" alt="Image 1" class="image">

<img style="margin-left:10px;width:470px;height:750px ;box-shadow: 2px 2px 2px black;"src="functional.png" alt="Image 2" class="image">

<img style="margin-left:10px;width:470px;box-shadow: 2px 2px 2px black;" src="combination.png" alt="Image 3" class="image">

</div>

<br>

<br>

<br>

<style>

.resume-button {

background-color: rgb(36, 32, 32);

color: rgb(255, 255, 255);

padding: 15px 32px;

border-color: black;

text-align: center;

text-decoration: none;

display: inline-block;

font-size: 16px;

margin: 4px 2px;

cursor: pointer;

margin-left:600px;

border-radius:15px;

transition :background-color 0.3s;

}

.resume-button:hover{

box-shadow:5px 5px 5px rgb(135, 130, 130);

transform: scale(1.03);

transition: transform 0.3s ease-in-out;

}

</style>

<a href='https://www.overleaf.com/gallery/tagged/cv'; target="\_blank">

<button class="resume-button">Personalise Your Own Resume</button>

</a>

<br>

<style>

.card {

background: rgba(255, 243, 224, 0.445);

width: 350px;

margin: 15px;

padding: 20px;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

text-align: center;

transition: transform 0.3s, box-shadow 0.3s;

}

.card:hover {

transform: scale(1.05);

box-shadow: 0 6px 12px rgba(0, 0, 0, 0.3);

}

.logocard {

width: 80px;

height: 80px;

margin-bottom: 10px;

}

.visit {

display: inline-block;

margin-top: 10px;

padding: 10px 15px;

background-color: #cf3e3e;

color: white;

text-decoration: none;

border-radius: 5px;

font-weight: bold;

transition: background-color 0.3s;

}

.visit:hover {

background-color: #b04040;

}

</style>

<br><br><hr>

<div>

<center><h1 style="color: #333; margin-top: 30px;">Best Resume-Building Websites</h1></center>

<div style="display: flex; justify-content: center; flex-wrap: wrap; margin-top: 20px;">

<div class="card">

<img src="io.png" alt="Resume.io" class="logocard">

<h2>Resume.io</h2>

<p>Professional resume templates with AI-powered suggestions.</p>

<a href="https://www.resume.io" target="\_blank" class="visit">Visit</a>

</div>

<div class="card">

<img src="https://cdn-images.zety.com/images/zety/apple-touch-icon-180x180.png" alt="Zety" class="logocard">

<h2>Zety</h2>

<p>Customizable resume templates with expert guidance.</p>

<a href="https://zety.com/resume-builder" target="\_blank" class="visit">Visit</a>

</div>

<div class="card">

<img src="https://upload.wikimedia.org/wikipedia/commons/4/44/Microsoft\_logo.svg" alt="Microsoft Create" class="logocard">

<h2>Microsoft Create</h2>

<p>Free resume templates with Microsoft’s design tools.</p>

<a href="https://create.microsoft.com/en-us/templates/resumes-and-cover-letters" target="\_blank" class="visit">Visit</a>

</div>

</div>

</div>

</div>

<br>

<hr>

<br>

<br>

<p style="margin-left:150px;font-size:50px;font-weight:bolder"><strong>How to write a Cover Letter?</strong></p>

<p style="margin-left:150px;font-size:30px;font-weight:bold">Introduction</p>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">To highlight that you are the ideal candidate, it is important to mention in the introduction the most valuable skills you possess that are also mentioned in the job ad. Sometimes it might be important to mention the job you are applying for, job application number (if provided) and where you learned about the open position. </p>

<p style="margin-left:150px;font-weight:bold;font-size:30px;">Body</p>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">In one or two longer paragraphs explain your interest for this position and how your previous achievements and experience will meet the expectations listed in the job ad.</p>

<p style="margin-left:150px;font-weight:bold;font-size:30px;">Conclusion </p>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">The conclusion of your cover letter has three main goals:</p>

<ul style=" margin-left:150px;font-size:23px;margin-right:50px">

<li>Express your gratitude towards the reader and thank them for their time.</li>

<li>Mention again your interest for the position.</li>

<li>A call to action (ex. I look forward to discussing with you how I can contribute to your organization’s success.)</li>

</ul>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">A strong call to action is important because this will be the last part the recruiter or employer will read and you will leavethe reader with the impression that you are the perfect match to be called for an interview.</p>

<p style="margin-left:150px;font-weight:bold;font-size:30px;"> How to end a Cover letter?</p>

<p style=" margin-left:150px;font-size:23px;margin-right:50px">A cover letter is a formal, professional document, therefore you should close your cover letter with a formal closing expression:</p>

<ul style=" margin-left:150px;font-size:23px;margin-right:50px">

<li>Sincerely</li>

<li> Sincerely yours</li>

<li>With best regards</li>

<li> Best regards</li>

<li>Kind regards</li>

<li> Respectfully yours</li>

</ul>

<style>

.content {

padding: 60px;

font-size: 23px;

padding-left: 120px;

line-height: 35px;

}

</style>

<div class="content">

<h2>Tips for Writing a Great Cover Letter</h2>

<ul>

<li><strong>Tailor to the Job:</strong> Customize your cover letter for each job application. Mention the job title and company name, and highlight how your skills align with the job requirements.</li>

<li><strong>Start Strong:</strong> Begin with a compelling opening that grabs the hiring manager's attention. Mention a relevant accomplishment or a strong reason why you're excited about the role.</li>

<li><strong>Showcase Your Skills:</strong> Highlight your most relevant skills and experiences. Use specific examples to demonstrate how you’ve successfully applied these skills in previous roles.</li>

<li><strong>Be Concise:</strong> Keep your cover letter to one page. Be clear and to the point, avoiding unnecessary details.</li>

<li><strong>Highlight Achievements:</strong> Quantify your achievements when possible. Use numbers to demonstrate your impact (e.g., "Increased sales by 20% in six months").</li>

<li><strong>Express Enthusiasm:</strong> Show genuine interest in the company and the role. Mention something specific about the company that excites you.</li>

<li><strong>Professional Tone:</strong> Maintain a professional tone throughout the letter. Avoid slang and overly casual language.</li>

<li><strong>Proofread:</strong> Ensure there are no typos or grammatical errors. Proofread your letter multiple times and consider asking someone else to review it as well.</li>

<li><strong>Use a Formal Closing:</strong> End with a formal closing like "Sincerely" or "Best regards," followed by your name and signature if sending a hard copy.</li>

<li><strong>Include Your Contact Information:</strong> Make it easy for the hiring manager to contact you by including your phone number and email address in the header of the cover letter.</li>

</ul>

</div>

<br>

<br>

<img style="margin-left: 300px;width:1000px;height:780px;border-radius: 30px;" class="image" src="coverletter.avif">

<footer style="background-color:#ffffff; color: rgb(0, 0, 0); text-align: center; padding: 15px; font-size: 18px;">

<br><br>

<p>&copy; 2025 Internzone. All Rights Reserved.</p>

<!-- <p>

<a href="privacy-policy.html" style="color: #0e0d0d; text-decoration: none;">Privacy Policy</a> |

<a href="terms-of-service.html" style="color: #0e0d0d; text-decoration: none;">Terms of Service</a> |

<a href="contact.html" style="color: #0e0d0d; text-decoration: none;">Contact Us</a>

</p>

</footer>

</body>

</html>

**Backend Implementation**:

const express = require("express");

const bodyParser = require("body-parser");

const mongoose = require("mongoose");

const cors = require("cors");

const app = express();

const PORT = 3000;

// Middleware

app.use(cors());

app.use(bodyParser.json());

// MongoDB connection

mongoose.connect("mongodb://localhost:27017/resumeDB", {

useNewUrlParser: true,

useUnifiedTopology: true,

});

// MongoDB schema

const userSchema = new mongoose.Schema({

name: String,

email: String,

});

const User = mongoose.model("User", userSchema);

// POST endpoint to receive form data

app.post("/api/submit", async (req, res) => {

try {

const { name, email } = req.body;

const user = new User({ name, email });

await user.save();

res.status(200).send("User saved!");

} catch (err) {

console.error(err);

res.status(500).send("Server Error");

}

});

// Start server

app.listen(PORT, () => {

console.log(Server running at http://localhost:${PORT});

});

## 5.1 TESTING AND VALIDATION

Testing and validation ensure that the Resume Architect system performs reliably, meets functional requirements, and provides a smooth user experience. Various testing methods were applied to identify and fix bugs, enhance usability, and validate the correctness of the output (including resume PDF generation).

### Types of Testing Performed

| **Testing Type** | **Purpose** |
| --- | --- |
| Unit Testing | Verify correctness of individual functions/modules |
| Integration Testing | Check proper interaction between frontend, backend, and database |
| Functional Testing | Ensure features behave as intended (e.g., registration, PDF generation) |
| UI/UX Testing | Validate usability, responsiveness, and design consistency |
| Security Testing | Validate login safety, data protection, token handling |
| Performance Testing | Measure response time and page load speed (basic load testing) |
| User Testing | Gather feedback from users during deployment and bug fixing |

Functional Testing

Functionality Test Case Result

Registration Valid email and password Pass

Login Invalid password rejected Pass

Resume Form Validation triggers for empty fields Pass

Template Selection Preview updates in real-time Pass

Bug Fixes Summary

Issue Found Fix Implemented

Form not saving data between steps Used session storage to persist inputs

Resume layout broken in PDF Adjusted CSS and Puppeteer settings

Slow API response during peak load Optimized DB queries and added caching

Forgot Password logic not working Under development (to be added later)

## 5.2 RESULTS AND DISCUSSION

The successful development and deployment of **Resume Architect** has demonstrated the ability to build a fully functional, scalable, and user-friendly resume builder web application from scratch. This section presents the final outputs, observations, and an evaluation of how well the system met its objectives.

### Summary of Achieved Features

| **Feature** | **Status** | **Notes** |
| --- | --- | --- |
| User Registration/Login | ✅ Complete | Secured with JWT & bcrypt |
| Resume Data Entry Forms | ✅ Complete | Modular, validated, and intuitive |
| Resume Template Selection | ✅ Complete | Multiple modern, industry-ready designs available |
| Live Resume Preview | ✅ Complete | Auto-updating preview while filling forms |
| Resume Download (PDF) | ✅ Complete | Clean PDF generation using Puppeteer |
| Responsive UI | ✅ Complete | Works across desktop, tablet, and mobile |
| Resume Storage (MongoDB) | ✅ Complete | Each resume saved per user session |
| User Feedback Collection | ✅ Added | Optional survey form integrated |
| Admin Panel (Template Upload) | 🔄 Planned | Scheduled for future enhancement |

### Sample Output Screens

* Registration and login pages
* Resume builder form (step-wise)
* Template selection panel
* Final resume preview
* PDF download confirmation
* Sample generated resume (PDF view)

### User Journey Flow (Final System)

1. **Login/Register**
2. **Input resume details via modular form**
3. **Select preferred resume template**
4. **Live preview auto-generated**
5. **Click ‘Download’ to export PDF**
6. **Resume saved in MongoDB for future retrieval**

### Key Insights

* Breaking the form into smaller steps dramatically improved user experience.
* Real-time feedback via live preview encourages completeness and accuracy.
* Using Puppeteer provided superior PDF export results compared to alternatives (e.g., html2pdf.js).
* Focusing on modularity during development allowed for smoother debugging and updates.

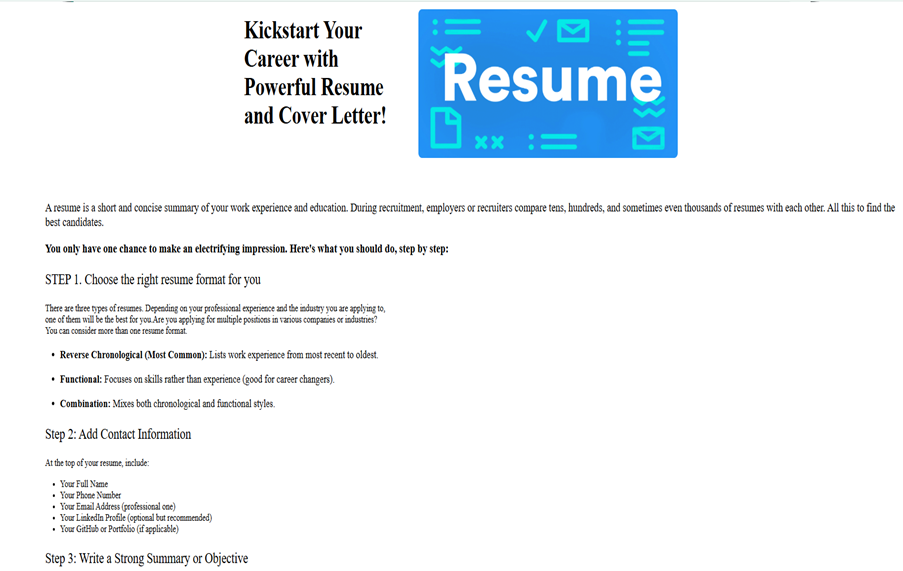
### Real-World Relevance

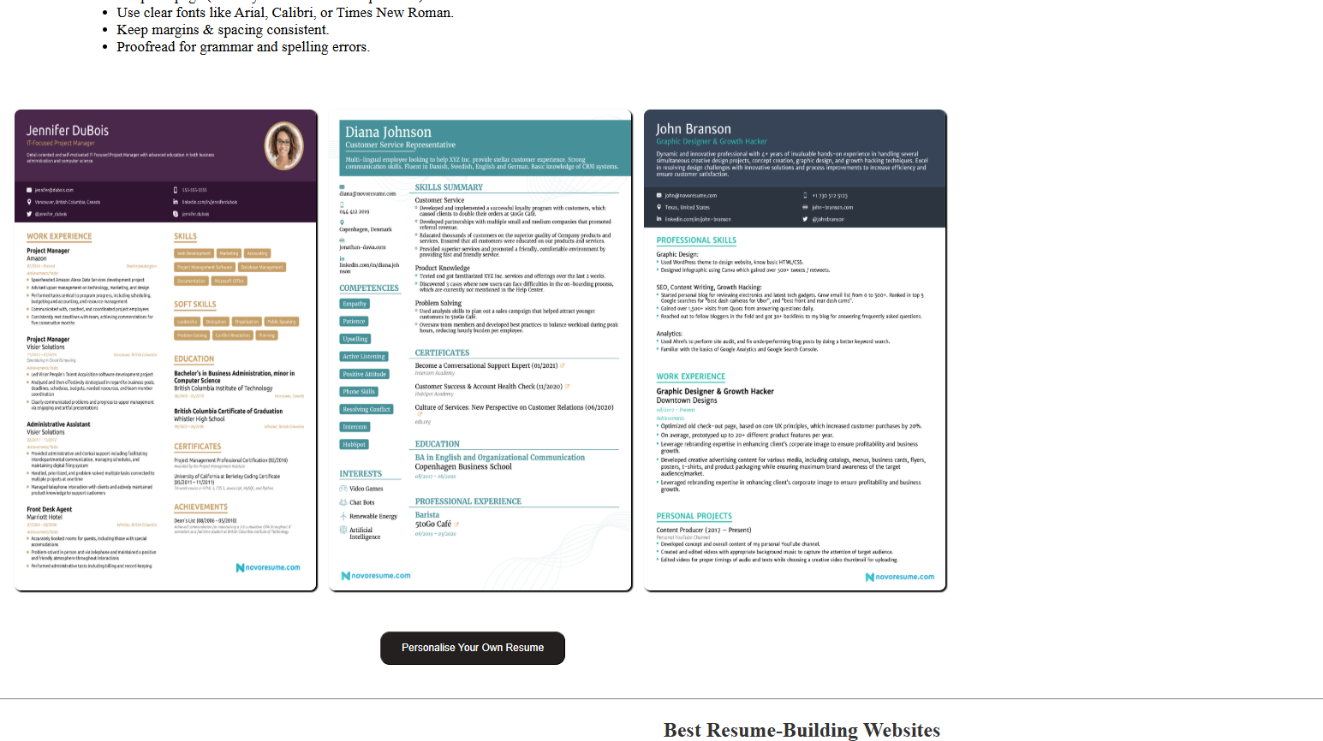
The Resume Architect system aligns with real-world requirements:

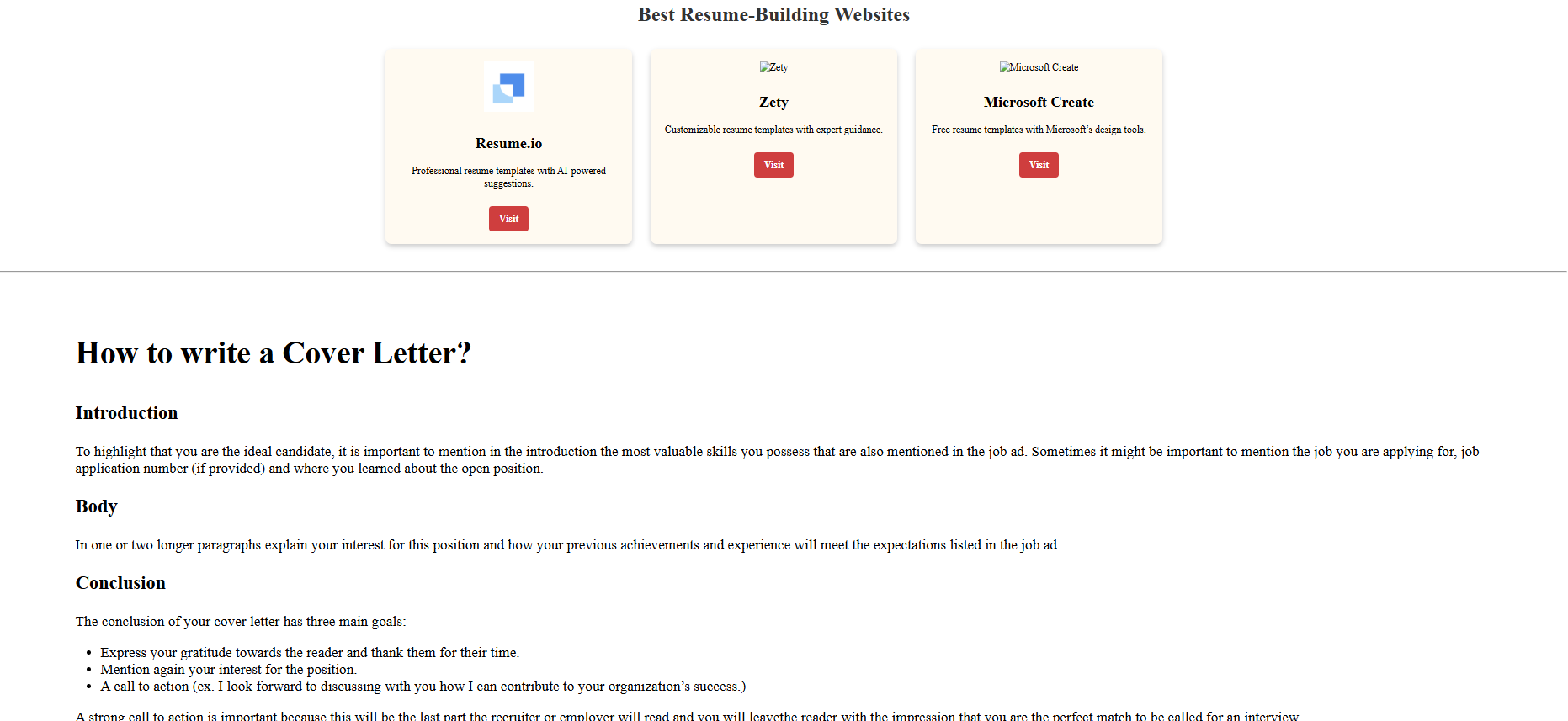
* Mimics platforms like Canva, Zety, and Resume.io in terms of usability
* Follows current job board requirements (PDF format, section clarity, ATS-readable designs)
* Can be extended for job tracking, resume analytics, and LinkedIn integration

## 5.3 SCREENSHOTS

This section provides visual evidence of the working web application, showcasing core functionalities of **Resume Architect** through various UI screens and outputs. Screenshots were captured during the final testing phase using Chrome Developer Tools and Postman









## FUTURE ENHANCEMENTS

Although the current version of **Resume Architect** successfully delivers core functionalities, several improvements and extensions have been identified to further enhance its capabilities, user experience, and scalability.

### User Dashboard

* **Description**: Introduce a personalized dashboard where users can:
  + View all their created resumes
  + Edit/delete previous versions
  + Track downloads and version history
* **Benefit**: Enables better user control and content management.

### Drag-and-Drop Resume Builder

* **Description**: Replace static form structure with an interactive, drag-and-drop UI for building resumes.
* **Benefit**: Offers greater flexibility and user creativity in arranging resume sections.

### More Resume Templates

* **Description**: Add 10–15 additional professionally designed templates.
* **Benefit**: Appeals to a wider user base with varying design preferences and industries (creative, academic, corporate, etc.).

### Multi-Language Support

* **Description**: Add support for multiple languages like Hindi, Spanish, Tamil, etc.
* **Benefit**: Expands reach to non-English speakers and international users.

### AI-Powered Suggestions

* **Description**: Use NLP or GPT-based models to suggest:
  + Bullet points for job descriptions
  + Skill keywords based on job role
  + Summary paragraphs
* **Benefit**: Helps users write impactful content and saves time.

### Real-Time Collaboration

* **Description**: Allow users to invite others (e.g., mentors or peers) to review or co-edit a resume.
* **Benefit**: Facilitates collaborative editing, similar to Google Docs.

## 6.1 LIMITATIONS

Despite the functional success of **Resume Architect**, certain constraints and shortcomings were identified during the development and testing phases. These limitations are outlined below to present a balanced view of the system’s capabilities and gaps.

### Static Resume Structure

* **Limitation**: The current form flow is linear and fixed.
* **Impact**: Users cannot rearrange resume sections based on their preference (e.g., moving Education above Experience).
* **Possible Fix**: Implement a drag-and-drop builder in future updates.

### Limited Template Library

* **Limitation**: Only a few templates (4–5) are currently available.
* **Impact**: Limited design diversity for different job roles and industries.
* **Possible Fix**: Expand the template library and allow user-contributed designs.

### No Real-Time Autosave

* **Limitation**: Resume data is only saved after form submission.
* **Impact**: Users may lose data if they navigate away mid-process.
* **Possible Fix**: Integrate real-time autosave with MongoDB and local storage fallback.

### No Resume Editing After PDF Generation

* **Limitation**: Once the PDF is generated, the form resets without saving the filled data for future use.
* **Impact**: Users must re-enter all data to generate another version.
* **Possible Fix**: Implement resume history and editing dashboard.

### No Cover Letter Support

* **Limitation**: The system is limited to resumes only.
* **Impact**: Users cannot create or attach personalized cover letters.
* **Possible Fix**: Add a cover letter builder module in a future update.

### No Image Upload Option

* **Limitation**: Currently, resumes are text-only and don’t support profile pictures.
* **Impact**: Some job roles prefer having a professional photo (e.g., creative/design roles).
* **Possible Fix**: Enable optional profile image upload with cropping/resizing.

### Limited Browser Testing

* **Limitation**: The application was tested primarily on modern browsers like Chrome and Firefox.
* **Impact**: Compatibility on older browsers like Internet Explorer is not guaranteed.
* **Possible Fix**: Perform comprehensive cross-browser testing and implement fallbacks.

### No Mobile App Version

* **Limitation**: Resume Architect is currently web-only.
* **Impact**: Limited usability for mobile-first users or offline access.
* **Possible Fix**: Develop a companion mobile app using Flutter or React Native.

### Performance Drop on Low-End Devices

* **Limitation**: PDF generation and template rendering can be slightly slower on devices with lower RAM or older processors.
* **Impact**: May lead to frustration for users with older systems.
* **Possible Fix**: Optimize rendering logic and use lighter UI libraries.

### No Resume Suggestions or AI Integration

* **Limitation**: Users manually write all content without system help.
* **Impact**: May affect users who struggle with writing effective bullet points or summaries.
* **Possible Fix**: Integrate AI modules for smart content suggestions and error correction.

## 6.2 CONCLUSION

The **Resume Architect** project successfully fulfilled its objective of providing a user-friendly, efficient, and modern platform for creating professional resumes. The web-based application delivers a seamless experience—from user registration and data input to dynamic previewing and high-quality PDF export.

### Project Impact

This project has proven to be highly impactful for:

* **Students and job seekers**, who need quick, attractive, and ATS-compliant resumes.
* **Non-technical users**, who benefit from the intuitive UI and guided form structure.
* **Developers**, as a learning case for building full-stack applications using modern web technologies.

### Key Takeaways

* Mastered **full-stack development** using HTML, CSS, JavaScript, Node.js, Express, and MongoDB.
* Gained experience with **RESTful API creation**, backend logic, and frontend-backend integration.
* Understood the value of **user-centric design** and responsive layout techniques.
* Applied **agile methodology** in organizing development tasks and sprint goals.
* Learned real-world challenges of **PDF generation**, **data validation**, and **UI responsiveness**.

### Final Thoughts

While this version of Resume Architect lays a strong foundation, it also opens the door to several exciting opportunities. With the integration of advanced features like AI-driven suggestions, collaborative editing, and a mobile app version, this project has the potential to evolve into a full-fledged resume and career support platform.

The development journey has been both technically enriching and personally rewarding, offering hands-on experience in building a product that solves real-world problems.

**Github Link :**

https://github.com/Dinakar-king/Dinakar/tree/main