Resume Builder Web Application

Version: 1.1.0

By

Mahadeva Prasad M
Hridhya
Amrithpal Singh

Introduction

Resume Builder : Is a tool that helps you create a professional resume quickly and easily. They can help you save time by providing templates and design elements to guide users through each resume section, such as personal information, work experience, skills, and education.

Features

- User -Friendly Interface: Easy navigation and intuitive design.
- Customizable Templates: A variety of professionally designed templates to choose from.
- Real-Time Editing: See changes as you make them.
- Export Options: Download your resume in multiple formats (PDF, Word, etc.)
- ATS Optimization: Ensures resume formats and content are optimized for Applicant Tracking Systems (ATS) to increase the likelihood of passing initial screening.
- Integrated Cover Letter Builder: Provides cover letter templates that match the resume design and offer guidance on writing a strong, concise cover letter.
- Portfolio or Project Showcase: Enables users to include a section for showcasing portfolio pieces, links to websites, or personal projects, ideal for creative and tech professionals.
- These features make the software more comprehensive, ensuring users can create polished, professional resumes that enhance their job application success.

System Requirements

- **1.Operating System**: Windows 10, MacOS, Linux.
- **2.Browser**: Latest version of Chrome, Firefox, Safari.
- **3.RAM**: Minimum 4 GB (8 GB recommended).
- **4.Storage**: At least 100 MB of free space.

Functional Requirements

1. User Authentication
- Users must be able to register for an account using an email and password.
- Users must be able to log in using their registered credentials.
- Users must be able to reset their password via a password recovery link sent to their email.

2. Resume Builder
- Users must be able to select from a variety of resume templates.
- Users must be able to add, edit, and delete sections in their resumes (e.g., contact information, work experience, education, skills).
- Users must be able to customize the appearance of the resume (fonts, colors, layout).
- The system should provide content suggestions (e.g., bullet points, action verbs) for different resume sections.

3. Preview and Export
- Users must be able to preview their resume in real-time as they make changes.
- Users must be able to download their resume in PDF format.
- Users must be able to save multiple versions of their resumes for future editing.

4. User Dashboard
- Users must have a dashboard where they can view all saved resumes.
- Users must be able to edit, delete, or duplicate resumes from their dashboard.

5. Template Management
- Administrators must be able to add, update, or delete resume templates from the system.
- Users must be able to filter templates based on categories (e.g., professional, creative).

6. User Profile Management
- Users must be able to view and edit their profile information (e.g., name, email).
- Users must be able to manage their account settings, including password changes.

7. Help and Support
- Users must be able to access a help or FAQ section that provides guidance on using the resume builder.

Non-Functional Requirements

1. Performance
   - The application should load within 2 seconds for optimal user experience.
   - The resume preview should update in real-time without noticeable delay.

2. Scalability
   - The system must support an increasing number of users and resumes without performance degradation.
   - The architecture should be designed to handle peak loads (e.g., high traffic during job application seasons).

3. Usability
   - The user interface must be intuitive and easy to navigate, minimizing the learning curve for new users.
   - The design must be responsive, ensuring functionality on various devices (desktops, tablets, smartphones).

4. Security
   - User data must be stored securely, with sensitive information (e.g., passwords) encrypted.
   - The application should implement measures against common vulnerabilities (e.g., SQL injection, XSS).

5. Compatibility
   - The website must be compatible with all major browsers (Chrome, Firefox, Safari, Edge).
   - The site should function well across different operating systems (Windows, macOS, Linux).

6. Accessibility
   - The website should comply with WCAG 2.1 guidelines to ensure accessibility for users with disabilities.
   - Features like keyboard navigation and screen reader support must be included.

7. Maintainability
   - The codebase must be modular and well-documented to facilitate easy updates and maintenance.
   - The system should allow for easy integration of new features without significant rework.

8. Availability
   - The system should have an uptime of 99.9%, ensuring that users can access the service at any time.
   - Regular backups should be implemented to prevent data loss

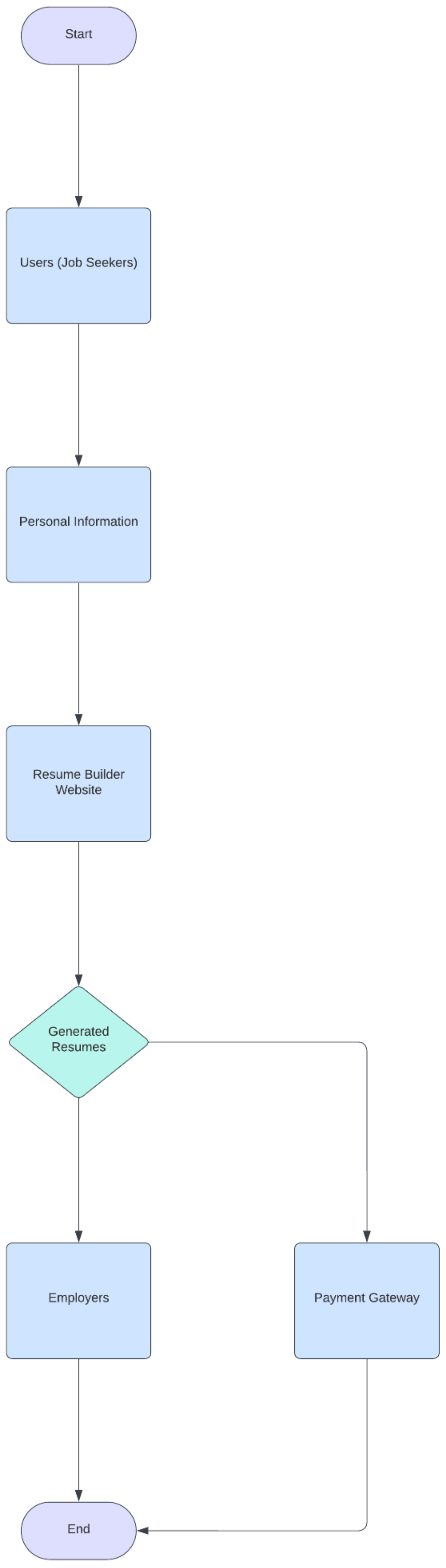
DATAFLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation used to visualize the flow of data within a system. In the context of a Software Requirements Specification (SRS), DFDs play a crucial role in illustrating the interactions between different components and data processes in a system. They help stakeholders understand how data moves through the system, identify potential bottlenecks, and ensure that all functional requirements are accurately captured and represented.

Levels of DFDs:

- **Level 0 :** This diagram provides a high-level overview of the entire system, showing the main processes and external entities that interact with the resume builder website.
- **Level 1 :** This level breaks down the main process (Resume Builder Website) into sub-processes, detailing how data flows through the system.
- **Level 2 :** This level further breaks down one of the sub-processes from Level 1 into detailed steps. Here, we'll focus on the **Resume Creation** process.

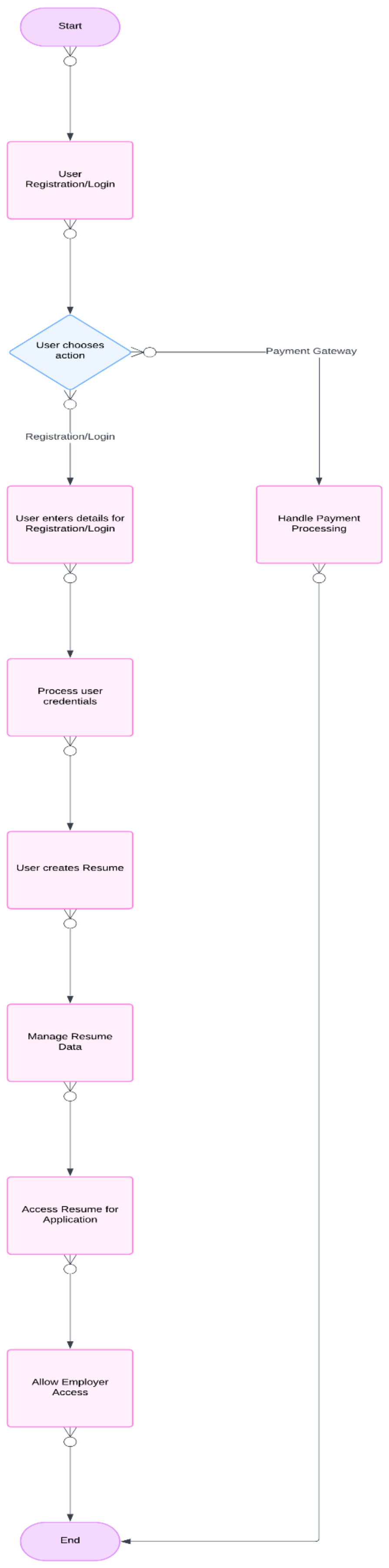
Level 0:DFD



Flow

- **Users:** Job seekers interact with the system by providing their personal information to create resumes.
- **Resume Builder Website:** This is the core system that processes the information from users and generates resumes.
- **Employers:** Employers can access resumes generated by users.
- **Payment Gateway(Optional):** If premium features are offered, users can make payments through this entity.

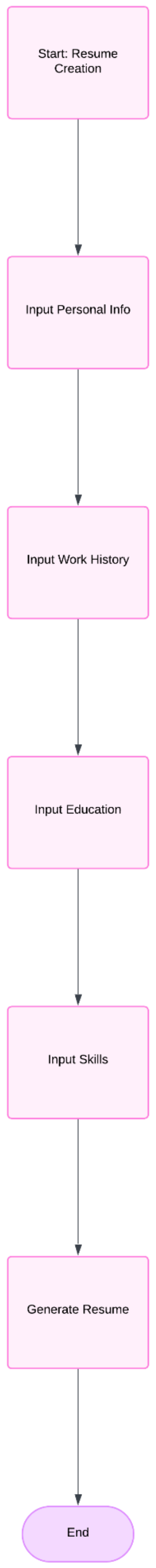
Level 1:DFD



Flow:

- **User Registration and Login:** This process handles user sign-ups and logins, ensuring that users can create an account or access their existing account.
- **Resume Creation:** After logging in, users can create resumes by inputting personal information, work history, education, and skills.
- **Resume Management:** Users can manage their resumes, which includes options to edit, delete, or view existing resumes.
- **Employer Access:** Employers can search for and access the resumes created by users, allowing them to find potential candidates.
- **Payment Gateway:** If users opt for premium features, this process manages payment transactions.

Level 2:DFD



Flow:

- **Input Personal Information:** Users enter their basic information such as name, contact details, and other personal identifiers.
- **Input Work History:** Users provide details of their past job experiences, including job titles, responsibilities, and employment dates.
- **Input Education:** Users input their educational background, including degrees, institutions, and graduation dates.
- **Input Skills:** Users list their skills relevant to their desired job positions.
- **Generate Resume:** Once all the information is collected, the system compiles the data into a professional resume format, which can then be downloaded or printed.

Technology Stack

Front-End Technologies

1.HTML: Structure and layout of web pages.

2.CSS: Styling for the website to improve user experience.

3.Tailwind : Responsive design and components for layout consistency.

Back-End Technologies

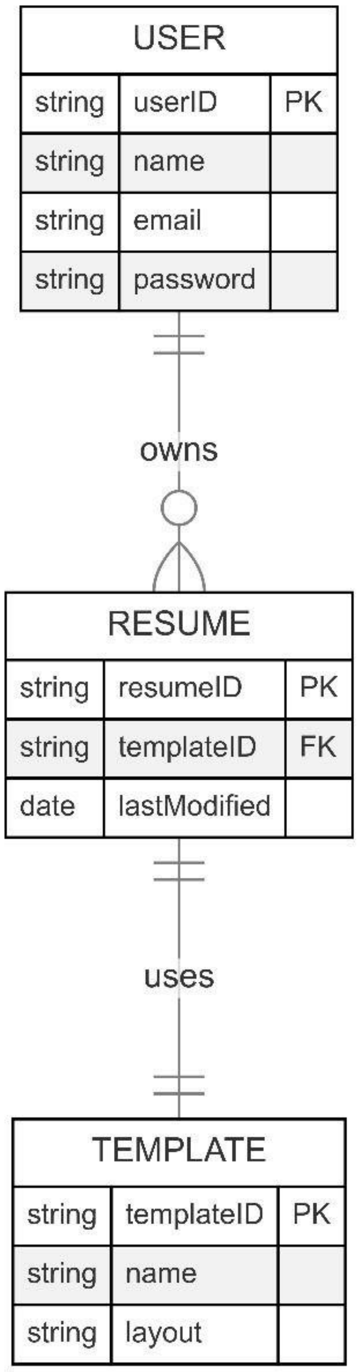
1.Python: Backend logic and functionality.

2.Flask: Web framework for creating API endpoints, handling server requests, and managing data flow.

3.SQLite3: Database for storing user accounts, resume data, and templates.

Database Schema

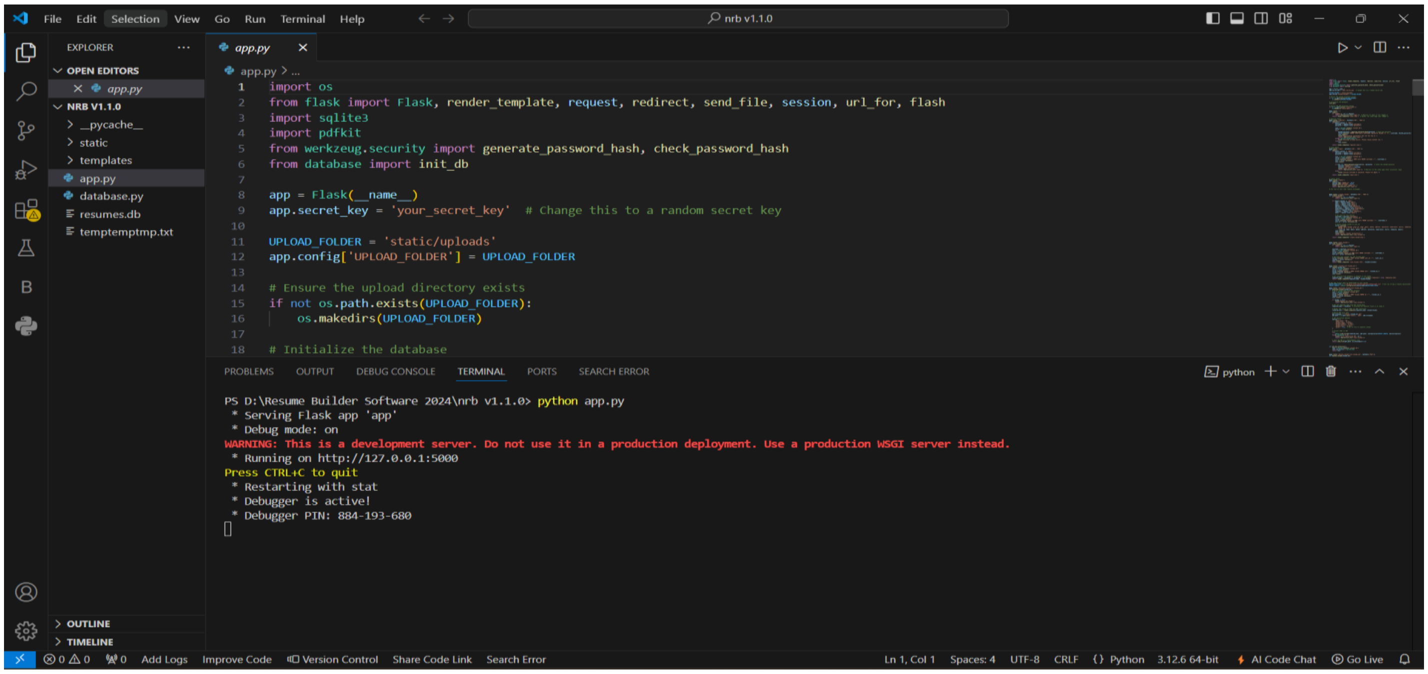
Table	Fields	Description
User	user_id, email, password, name	Stores user credentials and profile information
Resume	resume_id, user_id, template, content	Stores the content and template details of resumes
Template	template_id, name, layout, svg_icon	Stores template data and SVG references



Source code Directory

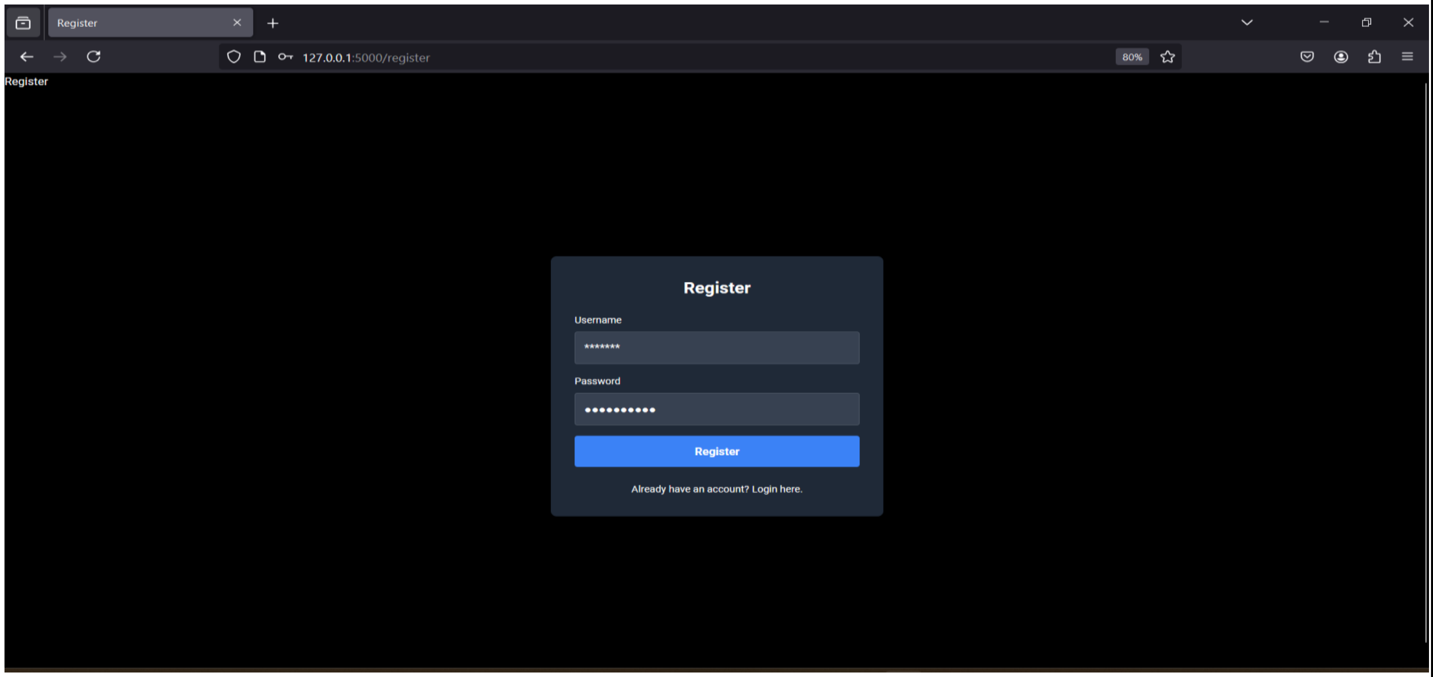
```
Resume Builder flask app/
│
├── app.py                  # Main application file
├── database.py             # Database initialization and connection logic
├── requirements.txt        # Python package dependencies
│
├── static/                 # Static files (CSS, JS, images, uploads)
│   ├── uploads/            # Uploaded files
│   └── pdfs/               # Generated PDF files
│
├── templates/              # HTML templates
│   ├── index.html          # Main index page
│   ├── login.html          # Login page
│   ├── register.html       # Registration page
│   ├── create_resume.html  # Resume creation page
│   ├── view_resumes.html   # Resume viewing page
│   ├── template1.html      # Default template (if needed)
│   ├── creative.html       # Example of a resume template
│   ├── modern.html         # Example of a resume template
│   ├── classic.html        # Example of a resume template
│   ├── elegant.html        # Example of a resume template
│   └── sleek.html          # Example of a resume template
│
├── .gitignore              # Git ignore file
└── README.md               # Project documentation
```

Screenshots


File Edit Selection View Go Run Terminal Help
nrb v1.1.0
EXPLORER
OPEN EDITORS
app.py
NRB V1.1.0
__pycache__
static
templates
app.py
database.py
resumes.db
temptemptmp.txt
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
python
Add Logs Improve Code Version Control Share Code Link Search Error
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.6 64-bit AI Code Chat Go Live

```python
import os
from flask import Flask, render_template, request, redirect, send_file, session, url_for, flash
import sqlite3
import pdfkit
from werkzeug.security import generate_password_hash, check_password_hash
from database import init_db

app = Flask(__name__)
app.secret_key = 'your_secret_key'  # Change this to a random secret key

UPLOAD_FOLDER = 'static/uploads'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

# Ensure the upload directory exists
if not os.path.exists(UPLOAD_FOLDER):
    os.makedirs(UPLOAD_FOLDER)

# Initialize the database
```

```
PS D:\Resume Builder Software 2024\nrb v1.1.0> python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 884-193-680
```



Register

Username

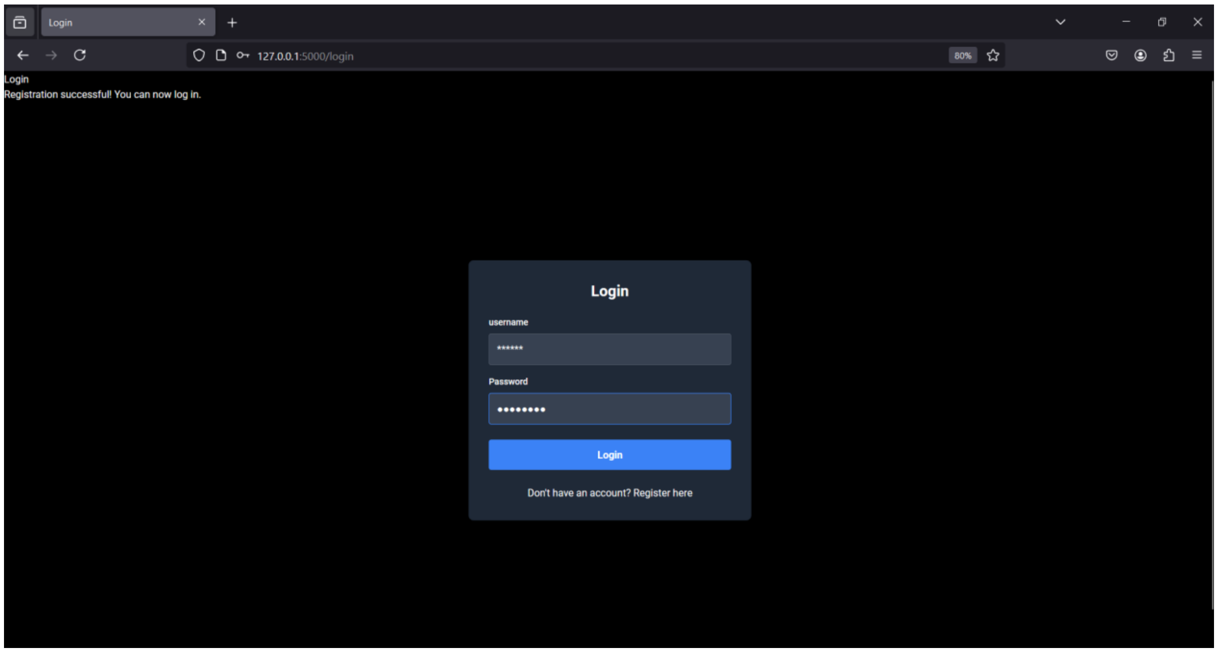
****

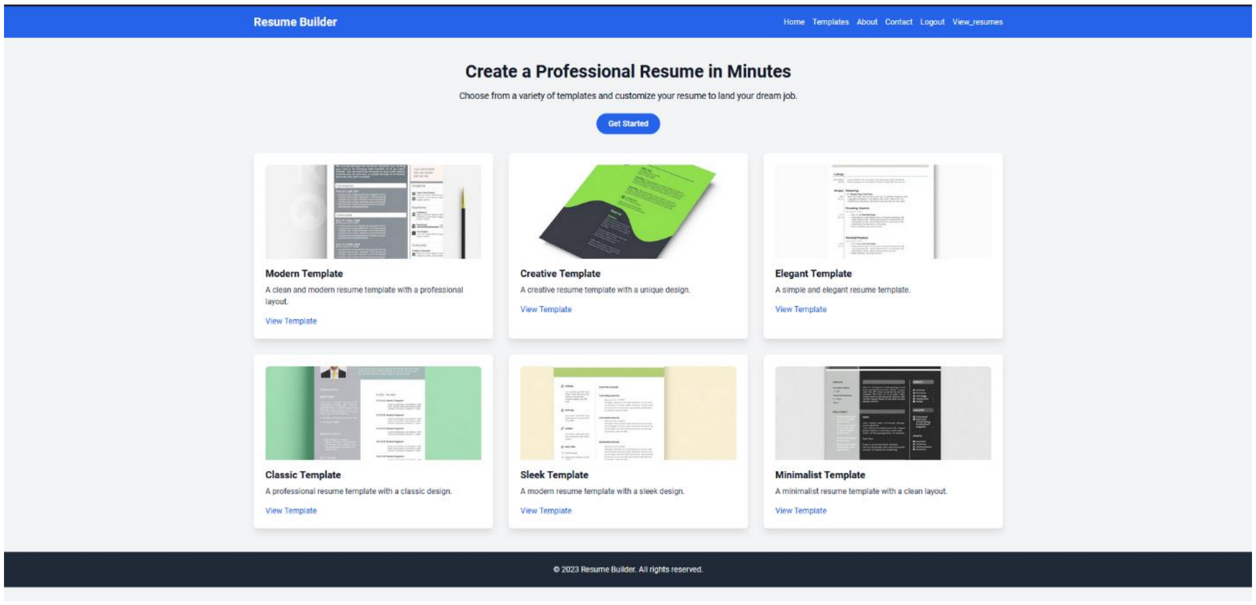
Password

••••••••••

Register

Already have an account? Login here.


Login
Registration successful! You can now log in.



Create a Professional Resume in Minutes

Choose from a variety of templates and customize your resume to land your dream job.

Modern Template
A clean and modern resume template with a professional layout.

Creative Template
A creative resume template with a unique design.

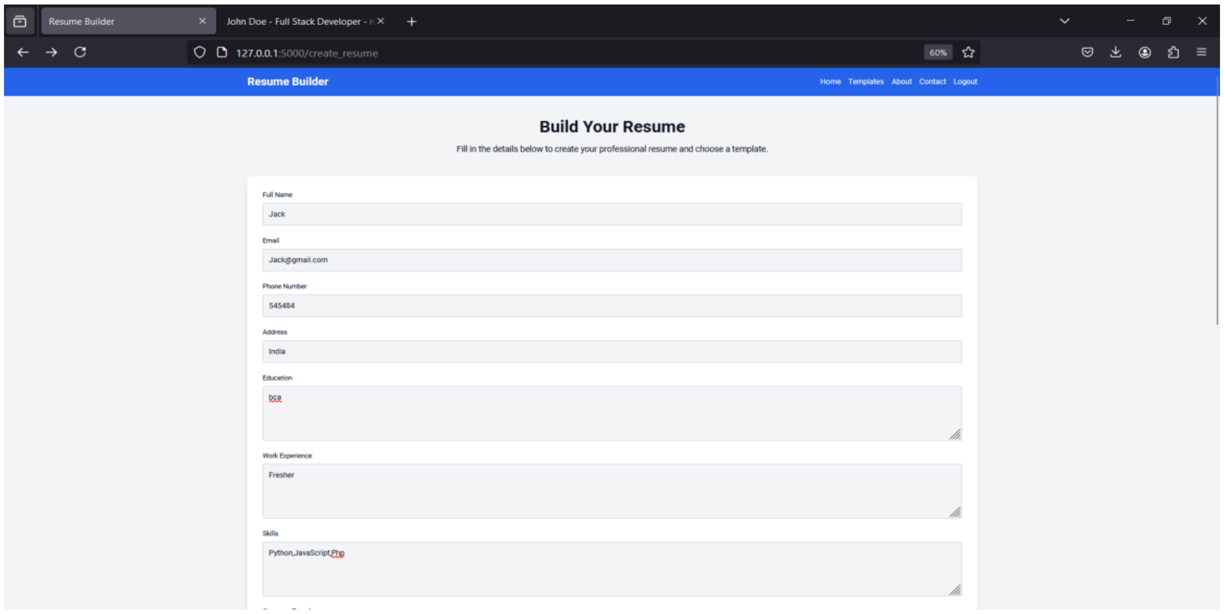
Elegant Template
A simple and elegant resume template.

Classic Template
A professional resume template with a classic design.

Sleek Template
A modern resume template with a sleek design.

Minimalist Template
A minimalist resume template with a clean layout.

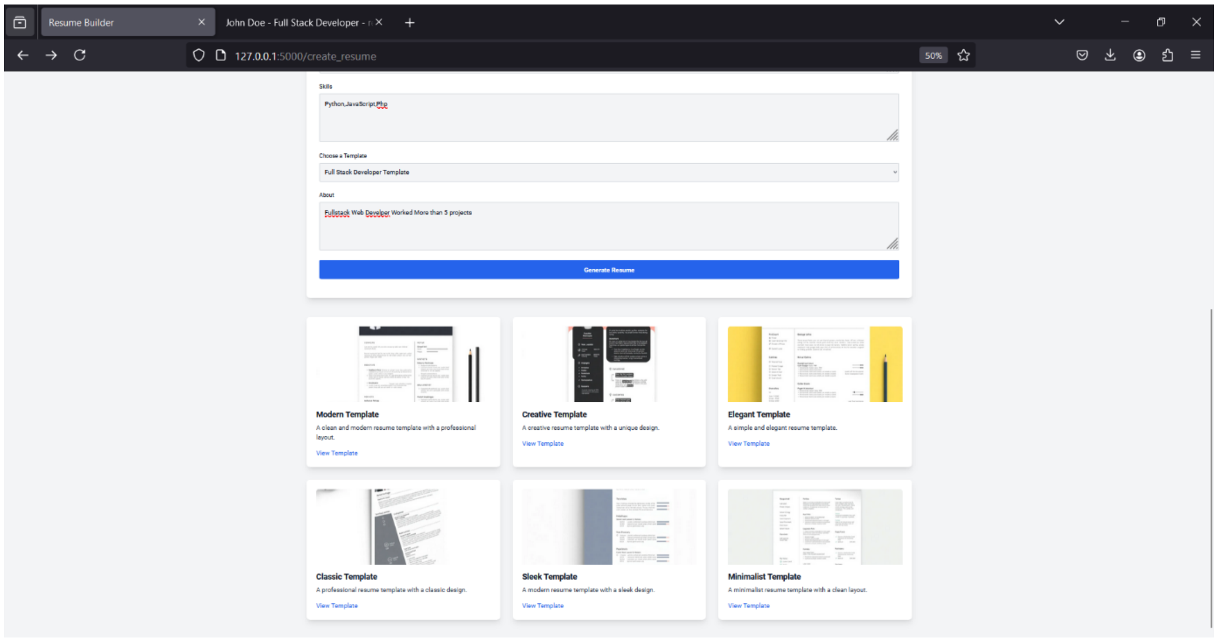
© 2023 Resume Builder. All rights reserved.


Resume Builder
John Doe - Full Stack Developer
127.0.0.1:5000/create_resume
60%
Resume Builder
Home Templates About Contact Logout

Build Your Resume

Fill in the details below to create your professional resume and choose a template.

Full Name	Jack
Email	Jack@gmail.com
Phone Number	545484
Address	India
Education	bca
Work Experience	Fresher
Skills	Python,JavaScript,Php


Resume Builder
John Doe - Full Stack Developer - r
127.0.0.1:5000/create_resume
50%

Skills

Python,JavaScript,Php

Choose a Template

Full Stack Developer Template

About

Fullstack Web Develper Worked More than 5 projects

Modern Template
A clean and modern resume template with a professional layout.
View Template

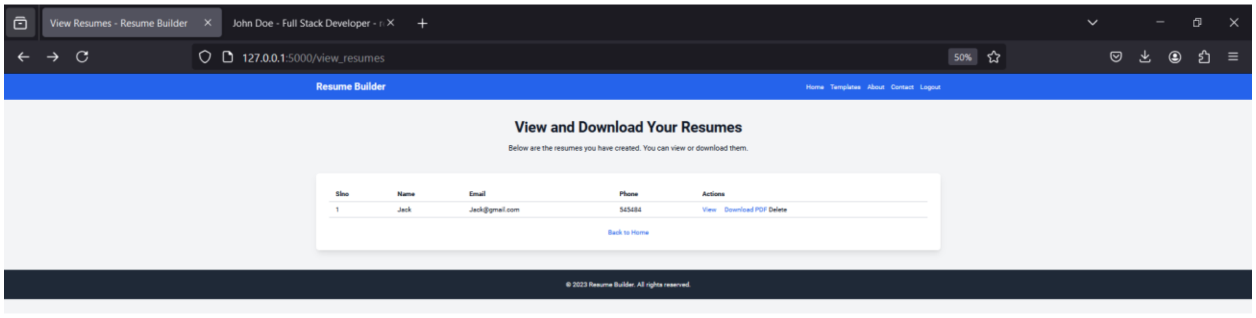
Creative Template
A creative resume template with a unique design.
View Template

Elegant Template
A simple and elegant resume template.
View Template

Classic Template
A professional resume template with a classic design.
View Template

Sleek Template
A modern resume template with a sleek design.
View Template

Minimalist Template
A minimalist resume template with a clean layout.
View Template


View Resumes - Resume Builder
John Doe - Full Stack Developer - r
127.0.0.1:5000/view_resumes
50%
Resume Builder
Home
Templates
About
Contact
Logout

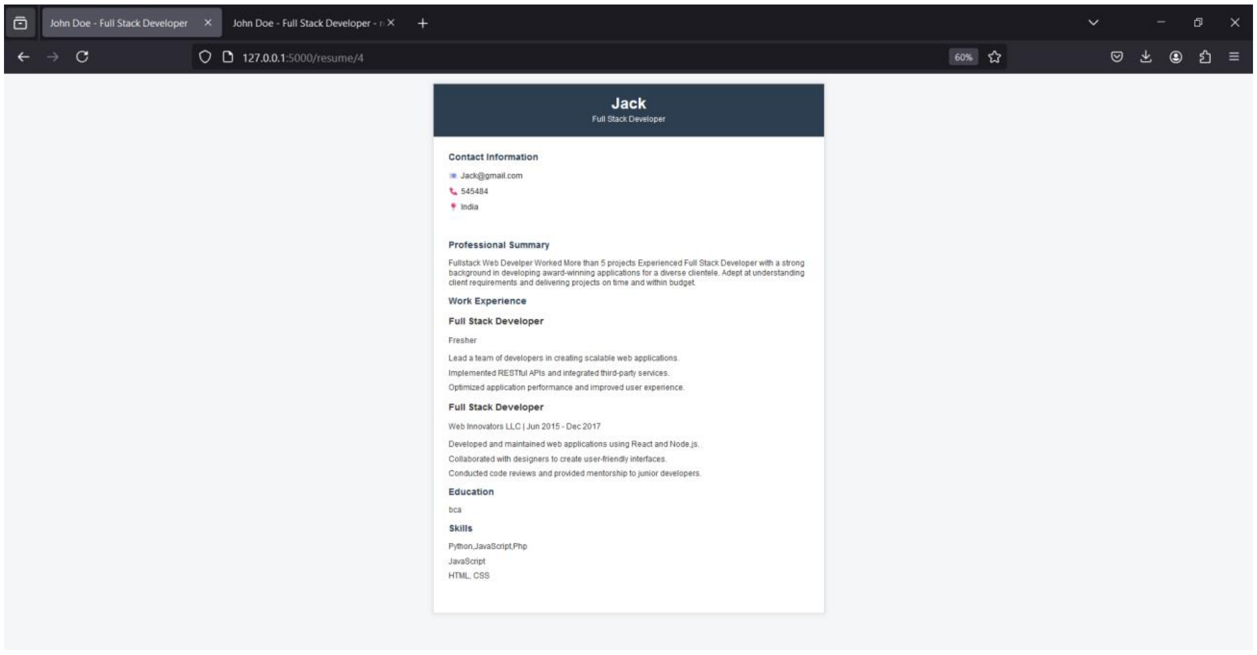
View and Download Your Resumes

Below are the resumes you have created. You can view or download them.

Slno	Name	Email	Phone	Actions
1	Jack	Jack@gmail.com	545484	View Download PDF Delete

Back to Home

© 2023 Resume Builder. All rights reserved.


John Doe - Full Stack Developer
John Doe - Full Stack Developer - r
127.0.0.1:5000/resume/4
60%

Jack
Full Stack Developer

Contact Information

Jack@gmail.com
545484
India

Professional Summary

Fullstack Web Develper Worked More than 5 projects Experienced Full Stack Developer with a strong background in developing award-winning applications for a diverse clientele. Adept at understanding client requirements and delivering projects on time and within budget.

Work Experience

Full Stack Developer

Fresher

Lead a team of developers in creating scalable web applications.

Implemented RESTful APIs and integrated third-party services.

Optimized application performance and improved user experience.

Full Stack Developer

Web Innovators LLC | Jun 2015 - Dec 2017

Developed and maintained web applications using React and Node.js.

Collaborated with designers to create user-friendly interfaces.

Conducted code reviews and provided mentorship to junior developers.

Education

bca

Skills

Python,JavaScript,Php

JavaScript

HTML, CSS

Web Software Deployment Instructions

Project Name : Resume Builder Version: 1.1.0

Last Updated : November 3, 2024

Prerequisites

Before deploying the application, ensure you have the following:

- Server Environment: Access to a server (e.g., AWS, DigitalOcean, etc.)
- Operating System: Linux distribution (Ubuntu, CentOS, etc.) is recommended
- Python Version: Ensure the server has Python installed (3.6 or above recommended)
- Package Manager: `pip` for installing Python packages
- Web Server: Nginx or Apache installed to serve the application
- Database System: MySQL, PostgreSQL, SQLite, or any required database
- Version Control: Git installed for managing your code

Environment Setup

1. Connect to the Server:
Use SSH to connect to your server:
Bash:

```
ssh username@your-server-ip
```

2.Update Package List: Ensure your package list is up to date:
Bash:

```
sudo apt update
sudo apt upgrade
```

3. Install Python and pip: If Python is not already installed, install it along with pip:
Bash:

```
sudo apt install python3 python3-pip python3-venv
```

4. Clone the Repository: Navigate to the directory where you want to deploy the application and clone the repository:
Bash:

```
git clone https://github.com/username/project-name.git
cd project-name
```

5. Create a Virtual Environment: It's best practice to use a virtual environment for Python applications:
Bash:
```
python3 -m venv venv
source venv/bin/activate
```

6. Install Dependencies: Install the required packages specified in your requirements.txt file:
Bash:
```
pip install -r requirements.txt
```

Deployment Steps

1.Set Up the Web Server: Create a configuration file for Nginx (example for serving a Flask application):
Nginx
```
server {
    listen 80;
    server_name your-domain.com;

    location / {
        proxy_pass http://127.0.0.1:8000;  # Assuming your app runs on port 8000
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Save this configuration to /etc/nginx/sites-available/project-name and create a symbolic link to enable it:
Bash:
```
sudo ln -s /etc/nginx/sites-available/project-name /etc/nginx/sites-enabled/
```
Test the Nginx configuration:
Bash:
```
sudo nginx -t
```
Restart Nginx:
Bash:
```
sudo systemctl restart nginx
```

2. Run the Application:
Use a WSGI server like Gunicorn to run your Python application. For example, if you have a Flask app, run:
Bash:
```
gunicorn --bind 127.0.0.1:8000 app:app  # Replace 'app:app' with your application entry point
```
You may want to run Gunicorn in the background or use a process manager like **supervisor** or **systemd** to manage the process.

- **Using Supervisor:**

  Install Supervisor:

  Bash:

  ```
  sudo apt install supervisor
  ```

  Create a configuration file for your application in /etc/supervisor/conf.d/project-name.conf:

  Ini

  ```
  [program:project-name]

  command=/path/to/project-name/venv/bin/gunicorn --bind 127.0.0.1:8000 app:app

  directory=/path/to/project-name

  user=username

  autostart=true

  autorestart=true

  stderr_logfile=/var/log/project-name.err.log

  stdout_logfile=/var/log/project-name.out.log
  ```

  Update Supervisor:

  Bash:

  ```
  sudo supervisorctl reread

  sudo supervisorctl update
  ```

  3. Database Setup:Ensure your database is set up and the necessary migrations have been applied. For example, with Flask-Migrate:

  Bash: :flask db upgrade

Configuration

1.Environment Variables: Set up environment variables required by your application. Create a .env file in your project directory:

Plaintext :
```
DATABASE_URL=mysql://user:password@localhost/dbname
SECRET_KEY=your_secret_key
```

Ensure your application loads these variables (you can use python-dotenv for Flask applications).

2.Static Files:
If your application serves static files, configure Nginx to serve them directly. Add the following to your Nginx configuration:
Nginx:
```
location /static {
    alias /path/to/project-name/static;  # Adjust the path
}
```

Testing the Deployment

After deployment, verify that everything is working correctly:

1. Open your web browser and navigate to http://your-domain.com.
2. Test the application by accessing various routes to ensure they are functioning correctly.

Troubleshooting

If you encounter issues during deployment, consider the following:

- Check server logs for errors:

Bash:
```
# Nginx logs
sudo tail -f /var/log/nginx/error.log

# Application logs (if using Supervisor)
sudo tail -f /var/log/project-name.err.log
sudo tail -f /var/log/project-name.out.log
```

- Ensure all environment variables are set correctly.

- Verify that the database is reachable and that migrations have been applied.

Rollback Instructions

If the deployment fails or issues arise, you may need to rollback to the previous version:

1. **Stop the Current Application:**

   If using Supervisor:

   Bash::

   ```
   sudo supervisorctl stop project-name
   ```

2. Checkout the Previous Version: In your project directory, checkout the previous commit or tag

   Bash::

   ```
   git checkout HEAD~1  # or the specific commit/tag
   ```

3. Reinstall Dependencies and Start the Application Again:
   Bash: :
   ```
   source venv/bin/activate
   pip install -r requirements.txt
   sudo supervisorctl start project-name
   ```

4. Verify the Rollback:
   Test the application again to ensure it is functioning correctly.

Conclusion

In conclusion, this documentation has outlined the features, functionalities, and deployment procedures of the Web Resume Builder software. Designed to simplify the resume creation process, our application offers users a user-friendly interface and customizable templates, ensuring they can create professional and personalized resumes effortlessly.

We have provided detailed instructions on how to register, log in, create and manage resumes, and download them in various formats. The application is built with security and user experience in mind, enabling users to store their information safely while providing easy access whenever needed.

As we move forward, we remain committed to enhancing the software based on user feedback and evolving industry standards. We encourage users to share their experiences and suggestions to help us improve our platform continuously.

Thank you for choosing our Web Resume Builder. We hope this tool empowers you to present your skills and experiences effectively in your job applications. Happy building!