A PROJECT REPORT

ON

"RESUME BUILDER WEBSITE"

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AKS UNIVERSITY, SATNA (M.P.)

Submitted in the fulfillment for the award of

Bachelor of Technology

Session (2023-24)

Submitted by:

AYUSH RAJAK
B2155R10106082
&
YASH KUMAR TRIPATHI
B2155R10106024
B. TECH (CSE) 5th Semester

Under the guidance of

Mr. Vinay Kumar Dwivedi

(Assistant professor)

Prof. Dr. Akhilesh A. Waoo

(Head and Associate Dean)

Faculty of Engineering and Technology

DECLARATION

I hereby declare that the work which is being presented in the report, entitled “RESUME BUILDER WEBSITE” in partial fulfillment for the award of degree of Bachelor of Technology in computer science and engineering (B. TECH CSE), submitted to the Department of Computer Science and Engineering, AKS University, Satna (M.P).

I have not submitted the matter presented in this report anywhere for any other degree.

Ayush Rajak (B2155R10106082)
Yash Tripathi (B2155R10106024)

B. TECH (CSE) 5th Semester

Candidate Sign ……………….. **Candidate Sign ………………..**

(B2155R10106082) (B2155R10106024)

CERTIFICATE

This is to certify that work presented in the Project Report Entitled as

RESUME BUILDER WEBSITE

In partial fulfillment of the requirement for the award of **COMPUTER SCIENCE AND ENGINEERING** from **A.K.S UNIVERSITY, SATNA, (M.P)** is an authentic work carried by **Yash Tripathi** and **Ayush Rajak** under my supervision.

To the best of my knowledge the content of this report does not form a basic for the award of any previous degree to anyone else.

……………………………… ...……………………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

…………………………… ...………………………..

Mr. Vinay Kumar Dwivedi **Prof. Dr. Akhilesh A. Waoo**

(Assistant Professor) (Head and Associate Dean)

ACKNOWLEDGEMENT

We (**Yash Tripathi** and **Ayush Rajak**) would like to express my special thanks of gratitude to our project guide (**Mr. Vinay Kumar Dwivedi**) as well as my senior friends and some classmates who helped me to do this wonderful project on the topic (Resume Builder Website), which also helped me in doing a lot of Research and I came to know about so many new things I am really thankful to them.

Secondly, I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Project Associates

YASH TRIPATHI (B2155R10106024)

AYUSH RAJAK (B2155R10106082)

ABSTRACT

This project endeavors to create a robust and user-friendly resume builder web application. The application is designed to streamline the process of generating professional resumes by providing an intuitive interface for users to input their personal, educational, and professional details. Leveraging modern web technologies including HTML, CSS, JavaScript, and jQuery, the application offers a dynamic and responsive experience across various devices.

Develop an intuitive user interface for seamless data input and management. Implement a responsive design to ensure accessibility across devices. Incorporate dynamic form elements and repeaters to facilitate varied resume sections. Validate user inputs to ensure accurate and error-free resume generation

TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

1.1 Introduction:

In today's competitive professional landscape, a well-crafted resume stands as a pivotal tool for individuals seeking to showcase their skills and experiences to potential employers. However, the process of creating a compelling resume often proves to be a daunting and time-consuming task. To address this challenge, the project introduces a dynamic and user-friendly Resume Builder Web Application.

1.2 Background:

The rapid evolution of technology has transformed the recruitment landscape, demanding resumes that not only encapsulate an individual's qualifications and experiences but also resonate with prospective employers. Crafting such resumes necessitates meticulous attention to detail and a structured format, often posing challenges for job seekers across various fields and experience levels

1.3 Purpose:

The primary aim of this project is to develop a comprehensive and accessible platform that simplifies the creation of professional resumes. By amalgamating modern web technologies and an intuitive user interface, this application endeavours to empower users with a tool that enables them to present their skills and experiences effectively.

1.4 Significance:

The significance of a well-constructed resume cannot be overstated in the competitive job market. It serves as the initial point of contact between a job seeker and a potential employer, influencing the impression a candidate leaves. Our application seeks to alleviate the stress and complexity associated with resume creation, catering to individuals from diverse backgrounds and professions.

1.5 Scope:

This project encompasses the development of a web-based resume builder that allows users to input personal, educational, and professional details. The application dynamically structures the entered information into a cohesive and well-formatted resume, offering users a range of templates and customization options.

1.6 Structure of the Report:

This report provides an in-depth analysis of the development process, methodologies, key features, and outcomes of the resume builder web application. It offers insights into the technical aspects, user interface design, functionality, and the impact of the application on facilitating the resume-building process.

1.7Features:

1.Dynamic Resume Building:

Customizable Templates: Offer a variety of professionally designed templates for users to choose from.

Dynamic Sections: Enable users to add, edit, and delete sections such as personal information, education, work experience, skills, projects, achievements, etc.

2.Intuitive User Interface:

User-Friendly Design: A clean and intuitive interface for easy navigation and seamless user experience.

Drag-and-Drop Elements: Integrate drag-and-drop functionalities for adding and arranging sections within the resume.

1.Data Validation and Error Handling:

Real-time Validation: Provide real-time validation for inputs, ensuring data accuracy.

Error Handling: Notify users of any errors or missing information in their resume inputs.

2.Resume Generation and Export

Dynamic Compilation: Automatically compile user-entered data into a well-structured and formatted resume.

Multiple Formats: Allow users to download their resumes in various formats like PDF, DOCX, or HTML.

3.Responsive Design and Compatibility

Device Compatibility: Ensure the application is responsive across devices (desktops, tablets, and mobile phones).

Cross-Browser Support: Optimize for compatibility across various web browser. By integrating these features, the resume builder application aims to provide users with a comprehensive and user-friendly platform for creating and managing professional resumes.

CHAPTER 2

REQUIREMENT ANALYSIS

Requirement analysis for a resume builder web application involves understanding and documenting the needs and specifications from various perspectives, including functional, non-functional, and user requirements. Here's an outline:

2.1 Functional Requirements:

- **User Management**:
  - **Registration and Authentication:** Users should be able to create accounts and log in securely.
  - **Profile Creation:** Ability to create and manage user profiles.

- **Resume Building Features:**
  - **Template Selection:** Users can choose from a range of professionally designed templates.
  - **Dynamic Sections:** Capability to add, edit, and remove sections like personal details, education, experience, skills, etc.
  - **Drag-and-Drop Interface**: Intuitive interface allowing easy rearrangement of sections.
  - **Validation and Error Handling:** Real-time validation of inputs and error notifications.

- **Resume Export and Sharing**:
  - **Multiple Formats:** Option to download resumes in formats like PDF, DOCX, or HTML.
  - **Sharing Options:** Ability to share resumes through email or generate shareable links.

2.2 Non-Functional Requirements:

- **Performance:**
  - **Responsiveness**: Ensure the application is responsive across devices and screen sizes.
  - **Load Time**: Minimal load times for faster user interaction.

- **Security and Privacy:**
  - **Data Encryption**: Ensure secure data transmission and storage using encryption techniques.
  - **User Controls:** Allow users to control settings for their profiles and resumes.

- **Usability and Accessibility**:
  - **User-Friendly Interface:** Intuitive design for easy navigation and use.
  - **Cross-Browser Compatibility:** Compatibility across various web browsers.
  - **Accessibility Standards**: Compliance with accessibility standards for differently-abled users.

- **User Requirements**:
  - **Customization:** Allow users to personalize templates with colors, fonts, etc.
  - **Easy of Use:** Intuitive interface and simple navigation for users of varying technical abilities.
- **Collaboration**:
  - **Collaboration Features**: Capability for multiple users to work on a resume simultaneously.

- **Business Requirements:**
  - **Scalable Architecture:** Ability to handle a growing user base and increased traffic.

- **Maintenance**:
  - **Update and Maintenance**: Provision for regular updates and maintenance for improvements.

- **Feedback Mechanism:**
  - **Feedback Collection:** Mechanism to gather user feedback for future enhancements. By comprehensively analysing these requirements, the development team can outline a roadmap to design, implement, and deploy a resume builder application that aligns with user needs and business objectives.

CHAPTER 3

SYSTEM DESIGN

3.1 Architecture:

1. Architecture Overview:

Front-End Driven: Entirely based on client-side technologies (HTML, CSS, JavaScript).

Single-Page Application (SPA): Provides a seamless user experience without full-page reloads.

Responsive Design: Ensures the resume builder is accessible and functional across devices.

2. Client-Side Structure:

HTML: Structures the content, defining sections and forms for user input.

CSS: Styles the layout, providing a visually appealing interface and responsive design.

JavaScript: Manages interactivity, form validation, and dynamically updating the resume preview.

3. User Interface (UI):

Form-Based Interface: User inputs their resume details through structured forms.

Preview Section: Displays a live preview of the resume as users input their information.

4. Data Handling:

Client-Side Storage: Utilizes local Storage or other client-side storage options to temporarily save resume data.

Data Validation: Implements form validation logic using JavaScript to ensure accurate data entry.

5. User Experience (UX):

Intuitive Design: Provides a simple and user-friendly interface for creating resumes.

Real-Time Updates: Dynamically updates the preview section as users input details.

6. Security Measures:

Client-Side Validation: Ensures inputs are sanitized and validated to prevent malicious entries.

Secure Connection: Ensures the website is served over HTTPS for data integrity.

7. Optimization and Performance:

Asset Optimization: Minifies CSS/JS files to reduce load times.

Lazy Loading: Loads assets only when needed to improve initial load speed.

8. Testing and Debugging:

Manual Testing: Verifies the functionality of forms, data handling, and preview sections.

Browser Developer Tools: Used for debugging JavaScript and inspecting elements.

9. Version Control and Deployment:

Git for Version Control: Manages project versions and facilitates collaboration.

Deployment to Hosting Platforms: Deploys the website to hosting services like GitHub Pages or Netlify.

10. Scalability and Maintenance:

Code Maintainability: Organizes code into modular components for easier maintenance.

Documentation: Provides clear documentation for future maintenance and updates.

3.2 Key Features of this Architecture:

High Responsiveness: Provides an interactive and highly responsive user interface.

Reduced Server Load: The server only serves static files (HTML, CSS, JS) without any data processing, significantly reducing server load.

Improved Performance: Faster loading times and smoother user experiences due to reduced server interaction.

- **Benefits:**

Simplicity: The absence of server logic simplifies the architecture and reduces operational complexities.

Cost-Effectiveness: No server backend means lower hosting costs.

Ease of Development: Focusing on client-side technologies provides ease in development and quicker iterations.

- **Challenges:**

Limited Functionality: As there's no server-side logic or databases, certain complex functionalities may be challenging to implement.

Security Concerns: All sensitive logic or data handling relies solely on client-side validation and may have vulnerabilities.

- **Scalability Considerations:**

Limited Scalability: The absence of server-side logic might limit scalability options as the client side solely manages all functionalities.

Potential Bottlenecks: Heavy client-side processing might affect performance as complexity increases.

This architecture suits projects that require simple functionalities and prioritize a rich user experience without heavy server-side processing or data handling. It's lightweight, easy to deploy, and cost-effective but might face limitations as the project scales in complexity and functionality.

3.2 Model – View –Controller (MVC) Design:

1.Model:

In your project, the Model represents the data and its manipulation logic. However, in a client-side-heavy application without a backend or database, the concept of a traditional model might not be explicitly defined. Instead, the data can be considered the content that users input into the various form fields.

2.View:

HTML & CSS: Your HTML files act as the View in this context, defining the structure and presentation of the content.

User Interface (UI): It's responsible for how the data is displayed and how users interact with the application.

Templates: HTML templates can be seen as views. These templates might represent different sections of a resume like education, experience, skills, etc.

3. Controller:

JavaScript Functions: JavaScript functions act as the Controller by handling user inputs, performing validations, and manipulating the data within the view.

Event Handlers: Functions responding to user actions like button clicks, form submissions, etc., control the flow and interaction within the application.

Logic Handling: They control the application logic and how the data (user inputs) is handled within the UI.

4. Summary:

Model: Data/content that users input.

View: HTML files, templates, and the user interface.

Controller: JavaScript functions, event handlers, and the logic handling user interactions.

In a simpler web application like this, the lines between these components might blur as much of the data manipulation and view updates happen directly through user interaction using JavaScript.

There might not be a clear separation as in a traditional backend-driven MVC architecture, but the basic principles of handling data (Model), presenting it (View), and controlling the application's behavior (Controller) are still followed, albeit in a more integrated manner on the client-side.

User interface design:

Designing a user interface and its components involves several considerations for an effective and visually appealing layout. For your resume builder website, here's a breakdown:

1. Navbar

Brand Logo and Navigation: Ensure clarity and simplicity.
Responsive Toggle Button: To handle smaller screens.

2. About Section

Form for Personal Details:

- First, Middle (optional), and Last Name fields.
- Image upload for profile picture.
- Contact details like email, phone number, address.

Summary or objective input area.

3. Sections for Different Resume Components:

Achievements: Add button to create multiple achievement fields.

Experience: Fields for job title, company, location, start/end date, description.

Education: Fields for school, degree, city, start/end date, description.

Projects: Fields for project name, link, description.

Skills: Input field for skills, with an add button to add more.

4. Preview Section

Render Area for Generated Resume: Displays the user-entered information in a formatted manner.

Sections for Name, Designation, About, Skills, Achievements, Education, Experience, Projects.

5. Print Button

Allows users to print the generated resume.

2 Design Considerations:

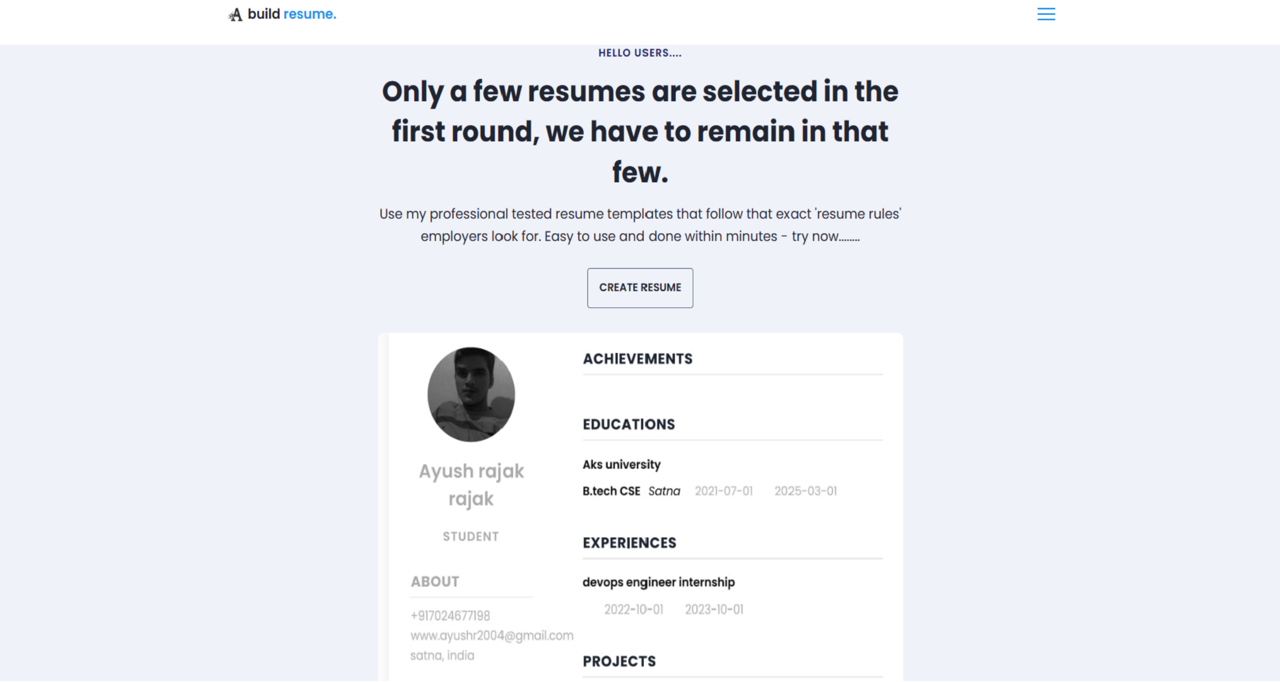
- **Consistency**: Maintain a consistent design theme and color scheme throughout the website.
- **Accessibility:** Ensure the interface is easy to use for all users.
- **Responsive Design**: Ensure the site is mobile-friendly and adapts to different screen sizes.
- **Validation:** Implement validation for form fields to ensure correct data entry.
- **Feedback:** Provide visual feedback for successful form submissions or errors.
- **Readability:** Use legible fonts, proper contrast, and appropriate font sizes for better readability.
- **Whitespace:** Use adequate whitespace to avoid clutter and improve visual hierarchy.
- **User-Friendly Forms:** Clearly labelled fields and placeholders for guidance.

3.4 Theory for Project Report:

- Discuss the rationale behind the chosen UI elements and their placement.

- ➢ Justify design decisions based on user experience principles.
- ➢ Explain the importance of each section in a resume and its relevance to the job-seeking process.
- ➢ Discuss any user testing, iterations, or feedback incorporated into the design.
- ➢ Reflect on the usability aspects and how the design aims to enhance user interaction.
- ➢ You can detail the design process, principles applied, user-centric decisions,

- **Home page:**



HELLO USERS....

Only a few resumes are selected in the first round, we have to remain in that few.

Use my professional tested resume templates that follow that exact 'resume rules' employers look for. Easy to use and done within minutes - try now.......

Ayush rajak rajak

STUDENT

ABOUT

+917024677198
www.ayushr2004@gmail.com
satna, india

ACHIEVEMENTS

EDUCATIONS

Aks university
B.tech CSE *Satna* 2021-07-01 2025-03-01

EXPERIENCES

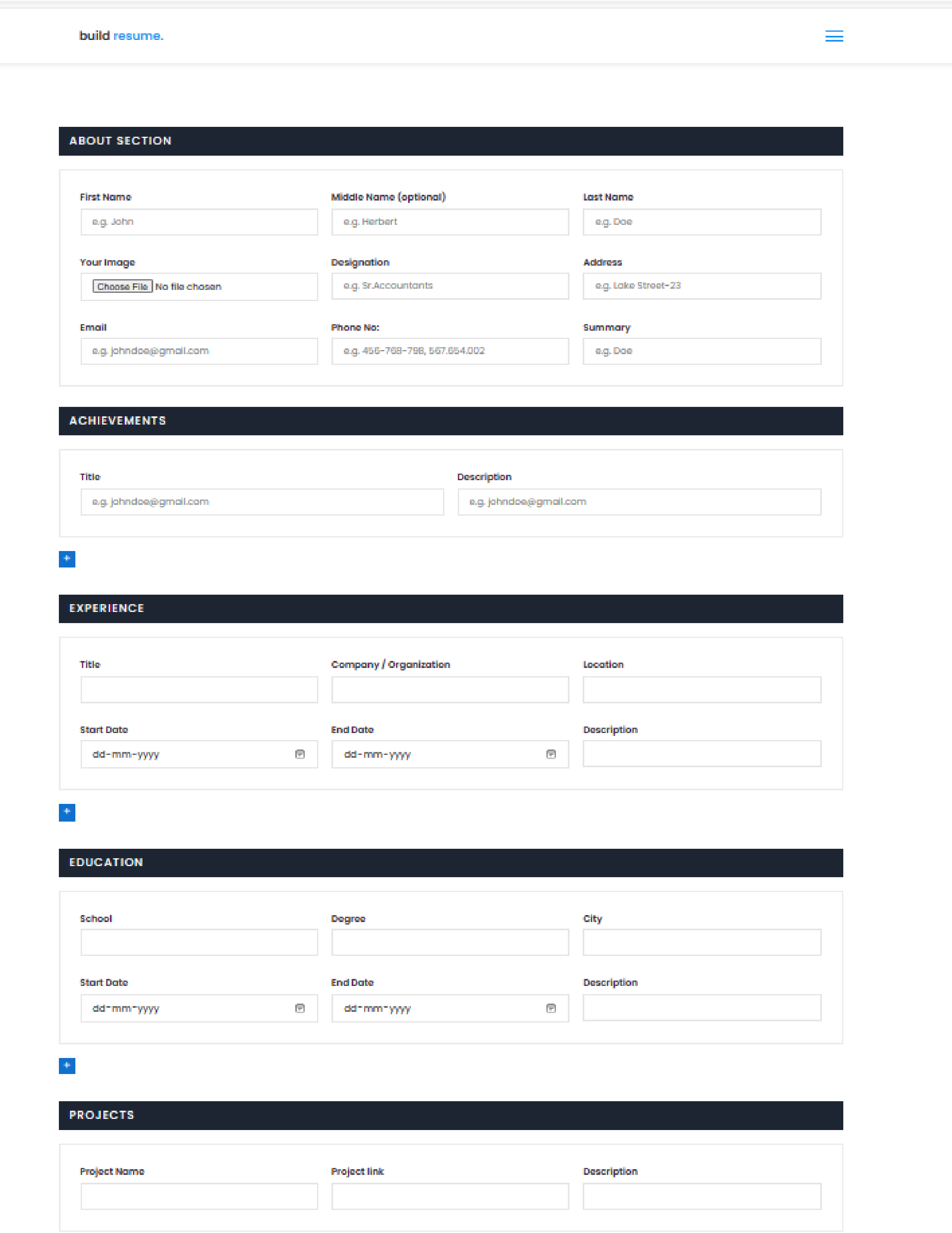
devops engineer internship
2022-10-01 2023-10-01

PROJECTS

- **Functionality of home page:**
  - ➢ **Navigation**: Helps users move around the website easily.
  - ➢ **Header:** Welcomes users and encourages them to create resumes using templates.
  - ➢ **Section One:** Talks about the advantages of using their resume maker.
  - ➢ **Section Two:** Highlights benefits like winning interviews and easy resume writing.
  - ➢ **Footer:** Contains copyright info and site credits.

The site aims to attract users to create resumes by showcasing the benefits of their resume builder in a straightforward and visually appealing manner. It guides users through the process and encourages action through clear call-to-action buttons.

- **Resume form page:**



ABOUT SECTION

First Name	Middle Name (optional)	Last Name
e.g. John	e.g. Herbert	e.g. Doe
Your Image	Designation	Address
Choose File No file chosen	e.g. Sr.Accountants	e.g. Lake Street-23
Email	Phone No:	Summary
e.g. johndoe@gmail.com	e.g. 456-768-798, 567.654.002	e.g. Doe

ACHIEVEMENTS

Title	Description
e.g. johndoe@gmail.com	e.g. johndoe@gmail.com

EXPERIENCE

Title	Company / Organization	Location
Start Date	End Date	Description
dd-mm-yyyy	dd-mm-yyyy	

EDUCATION

School	Degree	City
Start Date	End Date	Description
dd-mm-yyyy	dd-mm-yyyy	

PROJECTS

Project Name	Project link	Description

CHAPTER 4

IMPLEMENTATION

Here’s defined all details of resume builder website project:

4.1 Planning and Design:

Feature Definition: Identify the features your resume builder will offer, such as form input fields (personal details, experiences, skills), template selection, preview generation, and download options.

Wireframes and Mockups: Create wireframes or mockups using tools like Figma, Adobe XD, or Sketch to visualize the layout, user interface, and user experience (UI/UX) of your website.

User Flow: Map out the user journey from landing on the homepage to creating, editing, previewing, and downloading their resume. Define how users will navigate through different sections.

- **HTML**: Structures the content, defining sections and forms for user input.
- **CSS:** Styles the layout, providing a visually appealing interface and responsive design.
- **JavaScript**: Manages interactivity, form validation, and dynamically updating the resume preview.

4.2 code implementation:

Index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <title>ayush resume builder</title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="stylesheet" href="main.css">
    </head>
    <body>

        <nav class = "navbar bg-white">
            <div class="container">
                <div class = "navbar-content">
                    <div class = "brand-and-toggler">
                        <a href = "index.html" class = "navbar-brand">
                            <img src = "logo2.png" alt = "" class = "navbar-brand-icon">
                            <span class = "navbar-brand-text">build <span>resume.</span>
                        </a>
                        <button type = "button" class = "navbar-toggler-btn">
                            <div class = "bars">
                                <div class = "bar"></div>
                                <div class = "bar"></div>
                                <div class = "bar"></div>
                            </div>
                        </button>
                    </div>
                </div>
            </div>
        </nav>

        <header class = "header bg-bright" id = "header">
            <div class = "container">
                <div class = "header-content text-center">
                    <h6 class = "text-uppercase text-blue-dark fs-14 fw-6 ls-1">hello users....</h6>
                    <h1 class = "lg-title">Only a few resumes are selected in the first round, we have to remain in that few.</h1>
                    <p class = "text-dark fs-18">Use my professional tested resume templates that follow that exact 'resume rules' employers look for. Easy to use and done within minutes - try now........</p>
                    <a href = "resume.html" class = "btn btn-secondary text-uppercase">create resume</a>
                    <img src = "templates.jpg">
                </div>
            </div>
        </header>

        <div class="section-one">
            <div class="container">
                <div class = "section-one-content">
                    <div class="section-one-l">
                        <img src = "visual-0c7080adf17f1f207276f613447c924f667dab34b7ac415cd7ef653172defd0b.svg">
                    </div>
                    <div class = "section-one-r text-center">
                        <h2 class = "lg-title">Use the best resume maker as your guide!</h2>
                        <p class = "text">Getting that dream job can seem like an impossible task. We're here to change that. Give yourself a real advantage with the best online
```

```
resume maker: created by experts, imporved by data, trusted by millions of
professionals.</p>
            <div class = "btn">
              <a href = "resume.html" class = "btn btn-secondary text-
uppercase">create resume</a>

            </div>
          </div>
        </div>
      </div>
    </div>

    <div class = "section-two bg-bright">
      <div class="container">
        <div class="section-two-content">
          <div class = "section-items">
            <div class = "section-item">
              <div class = "section-item-icon">
                <img src = "feature-1-
edf4481d69166ac81917d1e40e6597c8d61aa970ad44367ce78049bf830fbda5.svg" alt
= "">
            <div class = "section-item">
              <h5 class = "section-item-title">Resume writing made easy!</h5>
              <p class = "text">Resume writing has never been this effortless.
Pre-generated text, visual designs and more - all already integrated into the resume
maker. Just fill in your details.</p>
            </div>
          </div>
        </div>
      </div>
    </div>
```

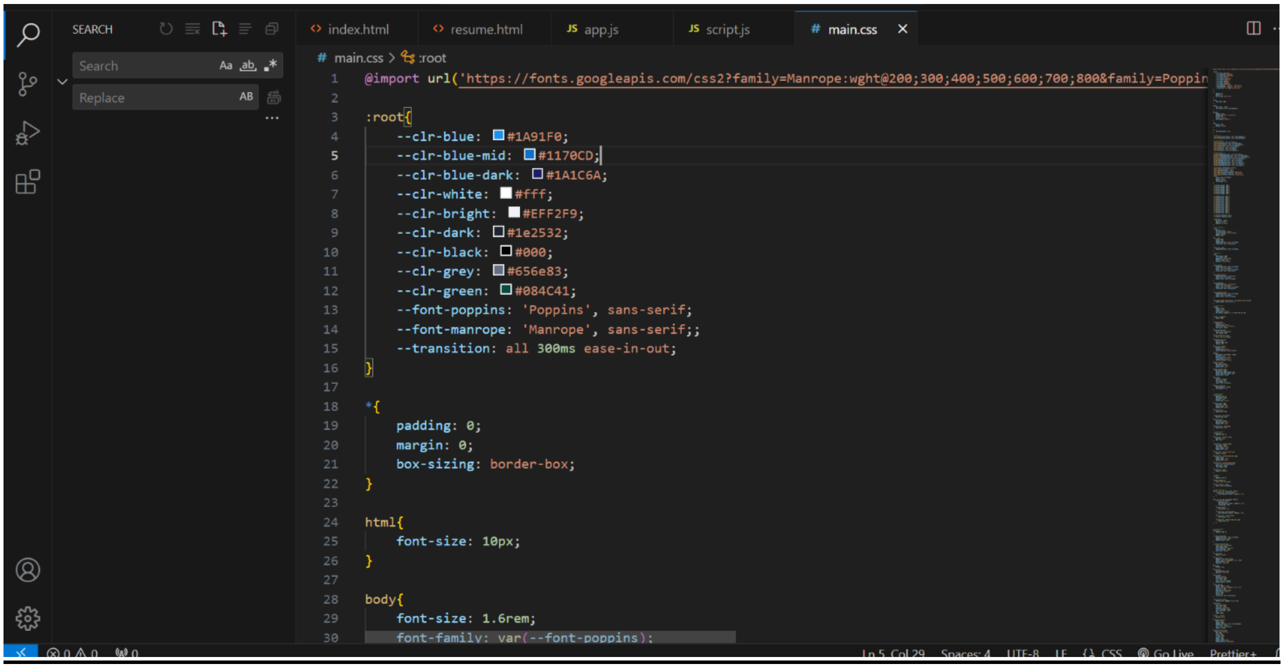
```
        <footer class = "footer bg-green">
          <div class="container">
            <div class = "footer-content text-center">
              <p class="fs-15">&copy;2023 created by ayush. - <span>
    build.resume</span></p>
            </div>
          </div>
        </footer>
      </body>
    </html>
```

Script.js file

```
// form repeater

$(document).ready(function(){
  $('.repeater').repeater({
    initEmpty: false,
    defaultValues: {
      'text-input': ''
    },
    show:function(){
      $(this).slideDown();
    },
    hide: function(deleteElement){
      $(this).slideUp(deleteElement);
      setTimeout(() => {
        generateCV();
      }, 500);
    },
                isFirstItemUndeletable: true
            })
        })
```

Main.css file



```
@import url('https://fonts.googleapis.com/css2?family=Manrope:wght@200;300;400;500;600;700;800&family=Poppin

:root{
    --clr-blue: #1A91F0;
    --clr-blue-mid: #1170CD;
    --clr-blue-dark: #1A1C6A;
    --clr-white: #fff;
    --clr-bright: #EFF2F9;
    --clr-dark: #1e2532;
    --clr-black: #000;
    --clr-grey: #656e83;
    --clr-green: #084C41;
    --font-poppins: 'Poppins', sans-serif;
    --font-manrope: 'Manrope', sans-serif;;
    --transition: all 300ms ease-in-out;
}

*{
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}

html{
    font-size: 10px;
}

body{
    font-size: 1.6rem;
    font-family: var(--font-poppins);
```

Resume.html file:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <title>Resume Page</title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <!-- custom css -->
        <link rel="stylesheet" href="main.css">
    </head>
```

```
<body>
    <nav class = "navbar bg-white">
        <div class="container">
            <div class = "navbar-content">
                <div class = "brand-and-toggler">
                    <a href = "index.html" class = "navbar-brand">
                        <img src = "" alt = "" class = "navbar-brand-icon">
                        <span class = "navbar-brand-text">build <span>resume.</span>
                    </a>
                    <button type = "button" class = "navbar-toggler-btn">
                        <div class = "bars">
                            <div class = "bar"></div>
                            <div class = "bar"></div>
                            <div class = "bar"></div>
                        </div>
                    </button>
                </div>
            </div>
        </div>
    </nav>

    <section id = "about-sc" class = "">
        <div class = "container">
            <div class = "about-cnt">
                <form action="" class="cv-form" id = "cv-form">
                    <div class = "cv-form-blk">
                        <div class = "cv-form-row-title">
                            <h3>about section</h3>
                        </div>
                        <div class = "cv-form-row cv-form-row-about">
                            <div class = "cols-3">
        <div class = "form-elem">
            <label for = "" class = "form-label">First Name</label>
            <input name = "firstname" type = "text" class = "form-control firstname" id = "" onkeyup="generateCV()" placeholder="e.g. John">
            <span class="form-text"></span>
        </div>
        <div class = "form-elem">
            <label for = "" class = "form-label">Middle Name <span class = "opt-text">(optional)</span></label>
            <input name = "middlename" type = "text" class = "form-control middlename" id = "" onkeyup="generateCV()" placeholder="e.g. Herbert">
            <span class="form-text"></span>
        </div>
        <div class = "form-elem">
            <label for = "" class = "form-label">Last Name</label>
            <input name = "lastname" type = "text" class = "form-control lastname" id = "" onkeyup="generateCV()" placeholder="e.g. Doe">
            <span class="form-text"></span>
        </div>
    </div>

    <div class="cols-3">
        <div class = "form-elem">
            <label for = "" class = "form-label">Your Image</label>
            <input name = "image" type = "file" class = "form-control image" id = "" accept = "image/*" onchange="previewImage()">
        </div>
        <div class = "form-elem">
            <label for = "" class = "form-label">Designation</label>
```

```
                    <input name = "designation" type = "text" class = "form-
control designation" id = "" onkeyup="generateCV()" placeholder="e.g.
Sr.Accountants">
                    <span class="form-text"></span>
                  </div>
                  <div class = "form-elem">
                    <label for = "" class = "form-label">Address</label>
                    <input name = "address" type = "text" class = "form-control
address" id = "" onkeyup="generateCV()" placeholder="e.g. Lake Street-23">
                    <span class="form-text"></span>
                  </div>
                </div>

                <div class = "cols-3">
                  <div class = "form-elem">
                    <label for = "" class = "form-label">Email</label>
                    <input name = "email" type = "text" class = "form-control
email" id = "" onkeyup="generateCV()" placeholder="e.g. johndoe@gmail.com">
                    <span class="form-text"></span>
                  </div>
                  <div class = "form-elem">
                    <label for = "" class = "form-label">Phone No:</label>
                    <input name = "phoneno" type = "text" class = "form-control
phoneno" id = "" onkeyup="generateCV()" placeholder="e.g. 456-768-798,
567.654.002">
                    <span class="form-text"></span>
                  </div>
                  <div class = "form-elem">
                    <label for = "" class = "form-label">Summary</label>
                    <input name = "summary" type = "text" class = "form-
control summary" id = "" onkeyup="generateCV()" placeholder="e.g. Doe">
```

```
                    <span class="form-text"></span>
                  </div>
                </div>
              </div>
            </div>

            <div class="cv-form-blk">
              <div class = "cv-form-row-title">
                <h3>achievements</h3>
              </div>

              <div class = "row-separator repeater">
                <div class = "repeater" data-repeater-list = "group-a">
                  <div data-repeater-item>
                    <div class = "cv-form-row cv-form-row-achievement">
                      <div class = "cols-2">
                        <div class = "form-elem">
                          <label for = "" class = "form-label">Title</label>
                          <input name = "achieve_title" type = "text" class =
"form-control achieve_title" id = "" onkeyup="generateCV()" placeholder="e.g.
johndoe@gmail.com">
                          <span class="form-text"></span>
                        </div>
                        <div class = "form-elem">
                          <label for = "" class = "form-
label">Description</label>
                          <input name = "achieve_description" type = "text"
class = "form-control achieve_description" id = "" onkeyup="generateCV()"
placeholder="e.g. johndoe@gmail.com">
                          <span class="form-text"></span>
                        </div>
                      </div>
                      <button data-repeater-delete type = "button" class =
"repeater-remove-btn">-</button>
                    </div>
                  </div>
                </div>
                <button type = "button" data-repeater-create value = "Add" class
= "repeater-add-btn">+</button>
              </div>
            </div>

            <div class="cv-form-blk">
              <div class = "cv-form-row-title">
                <h3>experience</h3>
              </div>

              <div class = "row-separator repeater">
                <div class = "repeater" data-repeater-list = "group-b">
                  <div data-repeater-item>
                    <div class = "cv-form-row cv-form-row-experience">
                      <div class = "cols-3">
                        <div class = "form-elem">
                          <label for = "" class = "form-label">Title</label>
                          <input name = "exp_title" type = "text" class =
"form-control exp_title" id = "" onkeyup="generateCV()">
                          <span class="form-text"></span>
                        </div>
                        <div class = "form-elem">
                          <label for = "" class = "form-label">Company /
Organization</label>
```

```
                <input name = "exp_organization" type = "text" class
= "form-control exp_organization" id = "" onkeyup="generateCV()">
                <span class="form-text"></span>
              </div>
              <div class = "form-elem">
                <label for = "" class = "form-
label">Location</label>
                <input name = "exp_location" type = "text" class =
"form-control exp_location" id = "" onkeyup="generateCV()">
                <span class="form-text"></span>
              </div>
            </div>

            <div class = "cols-3">
              <div class = "form-elem">
                <label for = "" class = "form-label">Start
Date</label>
                <input name = "exp_start_date" type = "date" class =
"form-control exp_start_date" id = "" onkeyup="generateCV()">
                <span class="form-text"></span>
              </div>
              <div class = "form-elem">
                <label for = "" class = "form-label">End
Date</label>
                <input name = "exp_end_date" type = "date" class =
"form-control exp_end_date" id = "" onkeyup="generateCV()">
                <span class="form-text"></span>
              </div>
              <div class = "form-elem">
                <label for = "" class = "form-
label">Description</label>
```

```
                                                    <input name = "exp_description" type = "text" class
= "form-control exp_description" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                            </div>

                                            <button data-repeater-delete type = "button" class =
"repeater-remove-btn">-</button>
                                        </div>
                                    </div>
                                </div>
                                <button type = "button" data-repeater-create value = "Add" class
= "repeater-add-btn">+</button>
                            </div>
                        </div>

                        <div class="cv-form-blk">
                            <div class = "cv-form-row-title">
                                <h3>education</h3>
                            </div>

                            <div class = "row-separator repeater">
                                <div class = "repeater" data-repeater-list = "group-c">
                                    <div data-repeater-item>
                                        <div class = "cv-form-row cv-form-row-experience">
                                            <div class = "cols-3">
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">School</label>
                                                    <input name = "edu_school" type = "text" class =
"form-control edu_school" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">Degree</label>
                                                    <input name = "edu_degree" type = "text" class =
"form-control edu_degree" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">City</label>
                                                    <input name = "edu_city" type = "text" class =
"form-control edu_city" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                            </div>

                                            <div class = "cols-3">
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">Start
Date</label>
                                                    <input name = "edu_start_date" type = "date" class =
"form-control edu_start_date" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">End
Date</label>
                                                    <input name = "edu_graduation_date" type = "date"
class = "form-control edu_graduation_date" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                                <div class = "form-elem">
```

```
                                                    <label for = "" class = "form-label">Description</label>
                                                    <input name = "edu_description" type = "text" class = "form-control edu_description" id = "" onkeyup="generateCV()">
                                                    <span class="form-text"></span>
                                                </div>
                                            </div>

                                            <button data-repeater-delete type = "button" class = "repeater-remove-btn">-</button>
                                        </div>
                                    </div>
                                </div>
                                <button type = "button" data-repeater-create value = "Add" class = "repeater-add-btn">+</button>
                            </div>
                        </div>

                        <div class="cv-form-blk">
                            <div class = "cv-form-row-title">
                                <h3>projects</h3>
                            </div>

                            <div class = "row-separator repeater">
                                <div class = "repeater" data-repeater-list = "group-d">
                                    <div data-repeater-item>
                                        <div class = "cv-form-row cv-form-row-experience">
                                            <div class = "cols-3">
                                                <div class = "form-elem">
                                                    <label for = "" class = "form-label">Project Name</label>
```

```
                          <input name = "proj_title" type = "text" class =
"form-control proj_title" id = "" onkeyup="generateCV()">
                          <span class="form-text"></span>
                        </div>
                        <div class = "form-elem">
                          <label for = "" class = "form-label">Project
link</label>
                          <input name = "proj_link" type = "text" class =
"form-control proj_link" id = "" onkeyup="generateCV()">
                          <span class="form-text"></span>
                        </div>
                        <div class = "form-elem">
                          <label for = "" class = "form-
label">Description</label>
                          <input name = "proj_description" type = "text" class
= "form-control proj_description" id = "" onkeyup="generateCV()">
                          <span class="form-text"></span>
                        </div>
                      </div>
                      <button data-repeater-delete type = "button" class =
"repeater-remove-btn">-</button>
                    </div>
                  </div>
                </div>
                <button type = "button" data-repeater-create value = "Add" class
= "repeater-add-btn">+</button>
              </div>
            </div>

            <div class="cv-form-blk">
              <div class = "cv-form-row-title">
```

```
                <h3>skills</h3>
              </div>

              <div class = "row-separator repeater">
                <div class = "repeater" data-repeater-list = "group-e">
                  <div data-repeater-item>
                    <div class = "cv-form-row cv-form-row-skills">
                      <div class = "form-elem">
                        <label for = "" class = "form-label">Skill</label>
                        <input name = "skill" type = "text" class = "form-
control skill" id = "" onkeyup="generateCV()">
                        <span class="form-text"></span>
                      </div>

                      <button data-repeater-delete type = "button" class =
"repeater-remove-btn">-</button>
                    </div>
                  </div>
                </div>
                <button type = "button" data-repeater-create value = "Add" class
= "repeater-add-btn">+</button>
              </div>
            </div>
          </form>
        </div>
      </div>
    </section>

    <section id = "preview-sc" class = "print_area">
      <div class = "container">
        <div class = "preview-cnt">
```

```
                    <div class = "preview-cnt-l bg-green text-white">
                        <div class = "preview-blk">
                            <div class = "preview-image">
                                <img src = "" alt = "" id = "image_dsp">
                            </div>
                            <div class = "preview-item preview-item-name">
                                <span class = "preview-item-val fw-6" id =
"fullname_dsp"></span>
                            </div>
                            <div class = "preview-item">
                                <span class = "preview-item-val text-uppercase fw-6 ls-1" id =
"designation_dsp"></span>
                            </div>
                        </div>

                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>about</h3>
                            </div>
                            <div class = "preview-blk-list">
                                <div class = "preview-item">
                                    <span class = "preview-item-val" id = "phoneno_dsp"></span>
                                </div>
                                <div class = "preview-item">
                                    <span class = "preview-item-val" id = "email_dsp"></span>
                                </div>
                                <div class = "preview-item">
                                    <span class = "preview-item-val" id = "address_dsp"></span>
                                </div>
                                <div class = "preview-item">
                                    <span class = "preview-item-val" id =
"summary_dsp"></span>
                                </div>
                            </div>
                        </div>

                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>skills</h3>
                            </div>
                            <div class = "skills-items preview-blk-list" id = "skills_dsp">
                                <!-- skills list here -->
                            </div>
                        </div>
                    </div>

                    <div class = "preview-cnt-r bg-white">
                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>Achievements</h3>
                            </div>
                            <div class = "achievements-items preview-blk-list" id =
"achievements_dsp"></div>
                        </div>

                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>educations</h3>
                            </div>
                            <div class = "educations-items preview-blk-list" id =
"educations_dsp"></div>
```

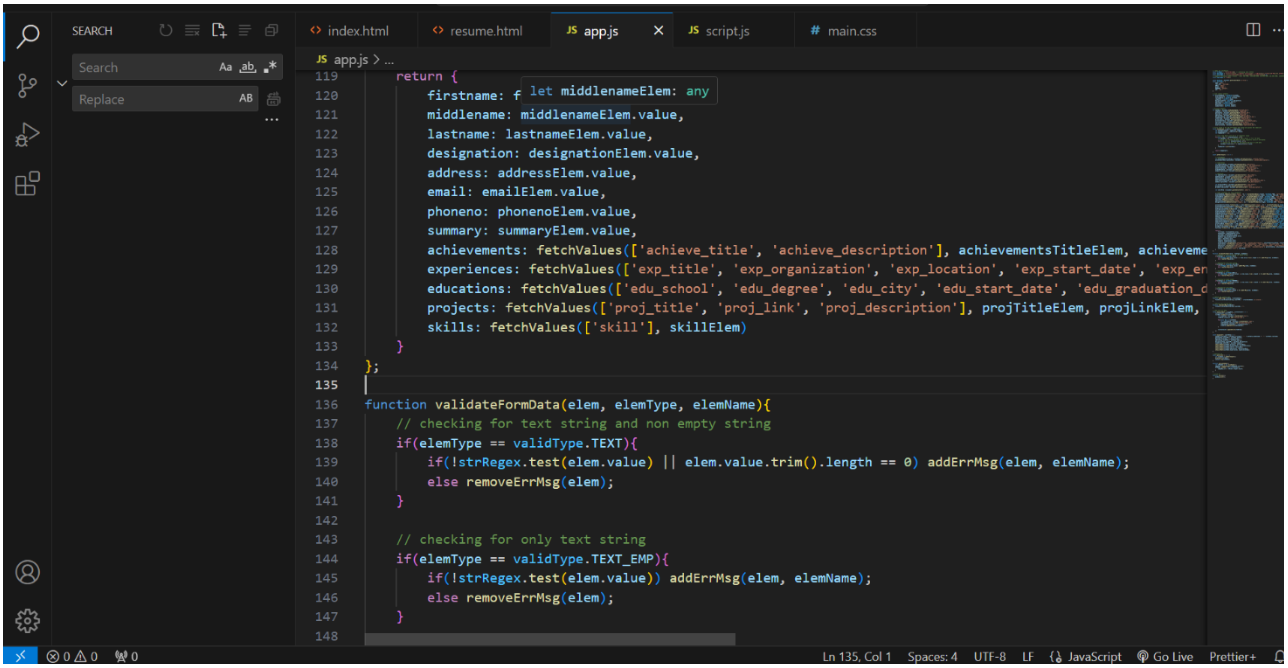
```
                        </div>

                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>experiences</h3>
                            </div>
                            <div class = "experiences-items preview-blk-list" id =
"experiences_dsp"></div>
                        </div>

                        <div class = "preview-blk">
                            <div class = "preview-blk-title">
                                <h3>projects</h3>
                            </div>
                            <div class = "projects-items preview-blk-list" id =
"projects_dsp"></div>
                        </div>
                    </div>
                </div>
            </div>
        </section>

        <section class = "print-btn-sc">
            <div class = "container">
                <button type = "button" class = "print-btn btn btn-primary"
onclick="printCV()">Print resume</button>
            </div>
        </section>
        <!-- jquery cdn -->
        <script src="https://code.jquery.com/jquery-3.6.4.js" integrity="sha256-
a9jBBRygX1Bh5lt8GZjXDzyOB+bWve9EiO7tROUtj/E="
crossorigin="anonymous"></script>
        <!-- jquery repeater cdn -->
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery.repeater/1.2.1/jquery.repeater.js"
integrity="sha512-
bZAXvpVfp1+9AUHQzekEZaXclsgSlAeEnMJ6LfFAvjqYUVZfcuVXeQoN5LhD7
Uw0Jy4NCY9q3kbdEXbwhZUmUQ==" crossorigin="anonymous"
referrerpolicy="no-referrer"></script>
        <!-- custom js -->
        <script src = "script.js"></script>
        <!-- app js -->
        <script src="app.js"></script>
    </body>
</html>
```

App.js file:



```
    return {
        firstname: f
        middlename: middlenameElem.value,
        lastname: lastnameElem.value,
        designation: designationElem.value,
        address: addressElem.value,
        email: emailElem.value,
        phoneno: phonenoElem.value,
        summary: summaryElem.value,
        achievements: fetchValues(['achieve_title', 'achieve_description'], achievementsTitleElem, achieveme
        experiences: fetchValues(['exp_title', 'exp_organization', 'exp_location', 'exp_start_date', 'exp_en
        educations: fetchValues(['edu_school', 'edu_degree', 'edu_city', 'edu_start_date', 'edu_graduation_d
        projects: fetchValues(['proj_title', 'proj_link', 'proj_description'], projTitleElem, projLinkElem,
        skills: fetchValues(['skill'], skillElem)
    }
};

function validateFormData(elem, elemType, elemName){
    // checking for text string and non empty string
    if(elemType == validType.TEXT){
        if(!strRegex.test(elem.value) || elem.value.trim().length == 0) addErrMsg(elem, elemName);
        else removeErrMsg(elem);
    }

    // checking for only text string
    if(elemType == validType.TEXT_EMP){
        if(!strRegex.test(elem.value)) addErrMsg(elem, elemName);
        else removeErrMsg(elem);
    }
```

CHAPTER 5

TESTING

5.1 Objective:

The objective of the testing was to ensure the functionality, usability, and performance of the resume builder website to provide a seamless experience for users in creating professional resumes.

5.2 Key Testing Areas:

- **User Registration and Authentication:** Verified the signup process and user login functionality.
- **Resume Creation**: Evaluated the ease of creating resumes, editing, and saving resume data.
- **Template Selection:** Checked the availability and applicability of various resume templates.

- **Responsive Design:** Tested website responsiveness across multiple devices and screen sizes.
- **Navigation and Links**: Ensured all navigation links and CTAs function correctly.
- **Error Handling:** Validated the effectiveness of error messages and proper handling of incorrect data inputs.
- **Performance Testing:** Analyzed the website's loading speed and resource optimization.
- **Cross-browser Compatibility:** Verified consistent functionality across different browsers.
- **Accessibility Check:** Confirmed compliance with accessibility standards for users with disabilities.

5.3 Results:

- **User Experience:** The website offers an intuitive user experience with easy navigation and a user-friendly interface for creating resumes.
- **Functionality:** All essential functions, including resume creation, template selection, and saving, work as intended.
- **Compatibility:** The website performs consistently across various devices, browsers, and screen sizes.
- **Performance:** Reasonable loading times and optimal resource utilization were observed during performance testing.
- **Accessibility:** The website complies with basic accessibility standards, ensuring inclusivity for users with disabilities.

5.4 Recommendations:

Enhance error messaging for clearer user guidance during data input.

Conduct further accessibility testing to ensure comprehensive compliance.

Implement additional templates or customization options for user diversity.

5.5 Benefits of testing:

Testing offers numerous benefits, contributing to the overall quality, reliability, and success of software or systems. Some key benefits include:

- **Identifying Defects Early:** Testing helps in catching and rectifying defects at an early stage of the development process. Fixing issues early reduces the cost and effort required for later corrections.

- **Enhanced Quality:** By validating functionalities against requirements, testing ensures that the software meets the desired quality standards and performs as expected.

- **Increased Reliability**: Rigorous testing leads to increased reliability and stability of the software, reducing the chances of failures or unexpected behaviour in production environments.

- **Improved User Experience:** Testing helps in ensuring a seamless and user-friendly experience by identifying and fixing usability issues, enhancing customer satisfaction.

- **Cost-Effectiveness:** Early detection and resolution of defects during the testing phase are less expensive than fixing issues found after the product is deployed. It reduces rework and potential losses due to system failures.

- **Risk Mitigation:** Testing helps in identifying potential risks associated with the software, such as security vulnerabilities or performance bottlenecks, allowing for proactive risk mitigation.

- **Compliance and Standards:** Testing ensures that the software adheres to industry standards, regulations, and compliance requirements, avoiding legal issues and penalties.

- **Confidence in Deployment:** Thorough testing provides stakeholders and users with confidence in the software's performance and functionality when deployed in a live environment.

- **Supports Decision-Making:** Test reports and metrics generated during testing provide valuable insights, helping stakeholders make informed decisions about the product's readiness for deployment.

- **Continuous Improvement:** Testing is an iterative process that allows for continuous improvement. Feedback from testing helps in refining the software and development processes for future iterations.

So all the testing plays a pivotal role in ensuring that software or systems meet quality standards, perform reliably, and satisfy user expectations, thereby contributing to the success of the products

CHAPTER 6

DEPLOYMENT

The deployment process typically involves the following steps:

- **Code Review:** Prior to deployment, conduct a thorough code review to ensure code quality, adherence to best practices, and compatibility with the production environment.

- **Testing and Quality Assurance:** Perform comprehensive testing, including unit tests, integration tests, and user acceptance testing (UAT), to validate functionality, identify bugs, and ensure a seamless user experience.
- **Configuration Management:** Prepare and configure the production environment, ensuring that servers, databases, and other necessary resources are set up correctly and securely.
- **Version Control and Tagging:** Use version control systems (e.g., Git) to manage code versions and tag the release candidate to maintain a clear record of changes.
- **Backup and Rollback Strategy:** Establish a robust backup plan and a rollback strategy to mitigate risks and quickly revert to a stable version in case of deployment issues.
- **Deployment to Production:** Deploy the thoroughly tested and approved code to the production environment, following a structured release plan to minimize downtime and user impact.
- **Monitoring and Post-Deployment Checks:** Monitor the application post-deployment to ensure its stability, performance, and security. Conduct additional checks to verify that all features are functioning as expected.
- **Documentation and Communication:** Document the deployment process, including configurations, dependencies, and any specific instructions. Communicate the successful deployment to stakeholders.
- **Feedback Collection**: Gather feedback from users and internal teams to address any immediate issues and plan future enhancements or fixes based on user experience.

Its significance in the journey from development to a functional, user-oriented application.

6.1 Continuous Integration/Continuous Deployment (CI/CD): If applicable, discuss how CI/CD pipelines were set up to automate the deployment process, ensuring efficiency and reducing errors.

Continuous Integration (CI) :

Purpose: CI is a development practice where developers frequently integrate code changes into a central repository. Each integration triggers automated tests to detect and resolve integration errors early.

Key Components:

- **Version Control**: Developers use version control systems like Git to manage changes to the codebase.
- **Automated Builds:** Code changes trigger automated build processes that compile the code, run tests, and generate artifacts.
- **Automated Testing:** CI involves running various tests (unit tests, integration tests, etc.) automatically to ensure new code changes haven't introduced bugs.
- **Feedback Loop:** Developers receive immediate feedback on their code changes, allowing them to fix issues early in the development cycle.

6.2 Continuous Deployment/Delivery (CD):

Purpose: CD involves automating the deployment process, making it possible to release code to production frequently and reliably.

Key Components:

- **Deployment Pipeline:** Automated pipelines define the stages (e.g., build, test, deploy) that code changes pass through before reaching production.
- **Automation:** Deployment to different environments (e.g., staging, production) is automated, reducing manual intervention.
- **Deployment Strategies:** Techniques like blue-green deployments, canary releases, or rolling updates ensure smooth deployments with minimal impact on users.

- ➢ **Monitoring and Rollback:** Continuous monitoring after deployment helps detect issues, and rollback mechanisms are in place in case of failures.

- **Combined Impact:**

CI/CD, when implemented together, streamlines the development process from code changes to deployment. It fosters a culture of collaboration, improves code quality, and accelerates the software delivery lifecycle by automating repetitive tasks. The ultimate goal is to deliver high-quality software efficiently and consistently.

6.3 Scalability and Load Balancing:

If applicable, discuss how the deployed application handles scalability and load balancing to accommodate increased user traffic.

Scalability and load balancing are critical components in ensuring a system's performance, reliability, and ability to handle increasing user demands.

- **Scalability:**

Definition: Scalability refers to a system's capability to handle a growing amount of work by adding resources to accommodate the increased load.

Types of Scalability:

- ➢ **Vertical Scalability (Scaling Up):** Increasing the capacity of a single machine by adding more resources like CPU, RAM, or storage.

Horizontal Scalability (Scaling Out): Adding more machines or nodes to distribute the load across multiple devices. This involves load balancing.

- **Load Balancing:**

Purpose: Load balancing evenly distributes incoming network traffic or application requests across multiple servers or resources.

Types of Load Balancing:

- **Round Robin:** Requests are distributed across servers sequentially, in a circular order.
- **Least Connections:** The load balancer sends incoming requests to the server with the fewest active connections.
- **Weighted Round Robin:** Assigns a weight to each server, distributing traffic based on these assigned weights.
- **Adaptive Load Balancing:** Uses real-time monitoring to adjust server selection based on current server loads.

CHAPTER 7

USER MANUAL

7.1 System Requirements

7.2 Hardware/Software Used

The hardware requirements Quiz Application are

- Processor: Core i3 Processor
- Hard Disk: 80 GB HDD
- Ram: 8 GB and above

The software specifications are:

- Operating System: Window 7 and above
- Html, CSS (front end)
- JavaScript (Back end)
- bootstrap V5 (framework)

CHAPTER 8

MAINTENANCE AND FUTURE ENHANCEMENTS

A resume builder website developed using HTML, CSS, JavaScript, and Bootstrap v5, maintenance and future enhancements are vital for sustaining the platform's functionality, improving user experience, and keeping it competitive. Here's how you might approach maintenance and future enhancements:

8.1 Maintenance:

- **Bug Fixing:** Regularly monitor and fix any bugs or issues reported by users or discovered through testing.
- **Security Updates:** Stay updated with security patches for Bootstrap, libraries, and server-side technologies to prevent vulnerabilities.
- **Browser Compatibility:** Ensure compatibility with different browsers and devices to provide a consistent experience.
- **Performance Optimization**: Continuously optimize the website's performance to ensure faster load times and responsiveness.

8.2 Future Enhancements:

- **Additional Templates:** Expand the range of resume templates offered to provide users with more options.
- **Feature Enhancements:** Implement new features like cover letter builders, skill assessments, or integration with professional networking platforms.
- **User Interface Improvements:** Enhance the user interface based on user feedback to make the resume creation process more intuitive

8.3 Considerations for Maintenance and Enhancements:

- **User Feedback**: Collect and analyse user feedback regularly to understand user needs and prioritize future enhancements.
- **Technology Updates:** Stay updated with the latest trends and technologies in web development to adopt improvements that can benefit the platform.
- **Backup and Version Control:** Maintain backups and use version control systems to manage changes effectively.
- **Documentation:** Keep comprehensive documentation for future developers or team members who might work on the project.

By consistently addressing maintenance needs and planning future enhancements based on user feedback and technological advancements, you can ensure your resume builder website remains relevant, user-friendly, and competitive in the long run.

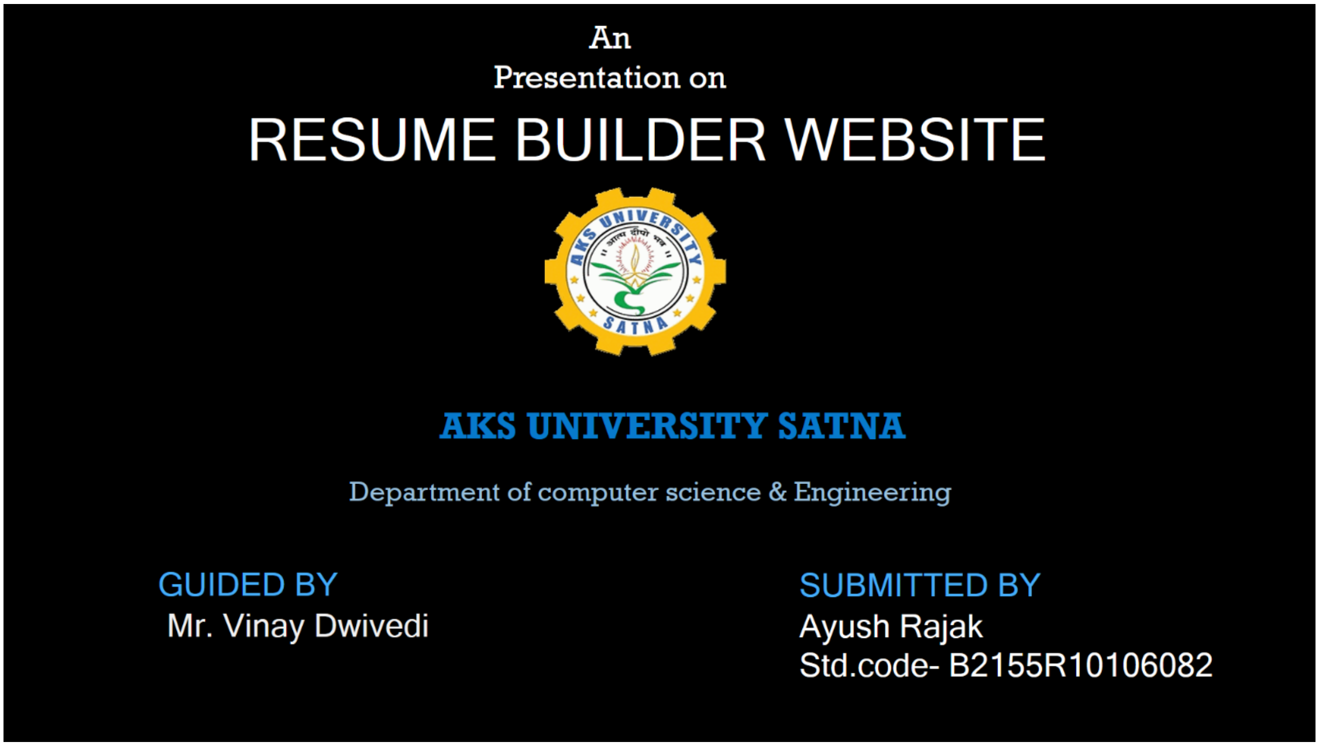
8.4 Conclusion:

Creating a resume builder website using HTML, CSS, JavaScript, and Bootstrap v5 involves a significant effort in design, development, and implementation. Throughout this project, the focus has been on providing users with a seamless experience in crafting professional resumes. In planning future enhancements, considerations for expanding templates, introducing new features, and refining the user interface based on feedback will be pivotal. Integration with external APIs and advancements in editing options will further enhance the platform's capabilities.

8.5 Reference

1. College professor’s
2. Friends.
3. Internet.

Annexure



An
Presentation on

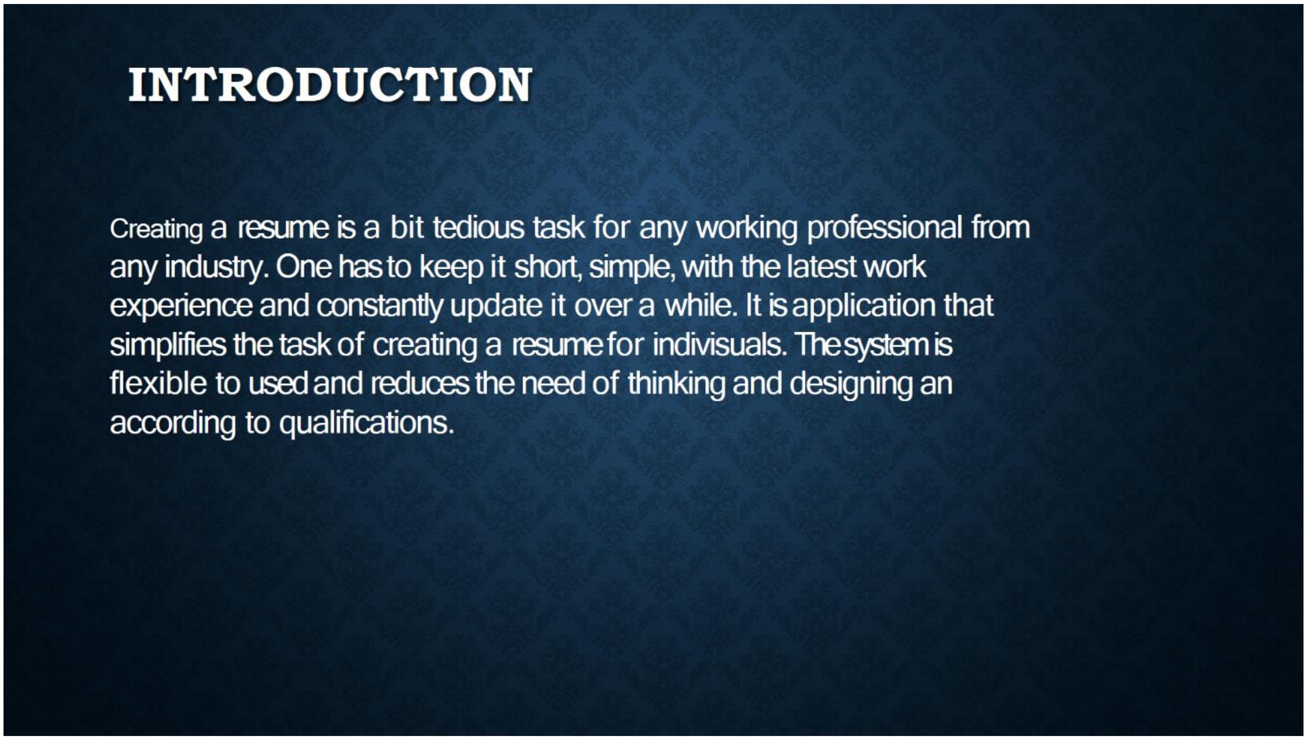
RESUME BUILDER WEBSITE

AKS UNIVERSITY SATNA

Department of computer science & Engineering

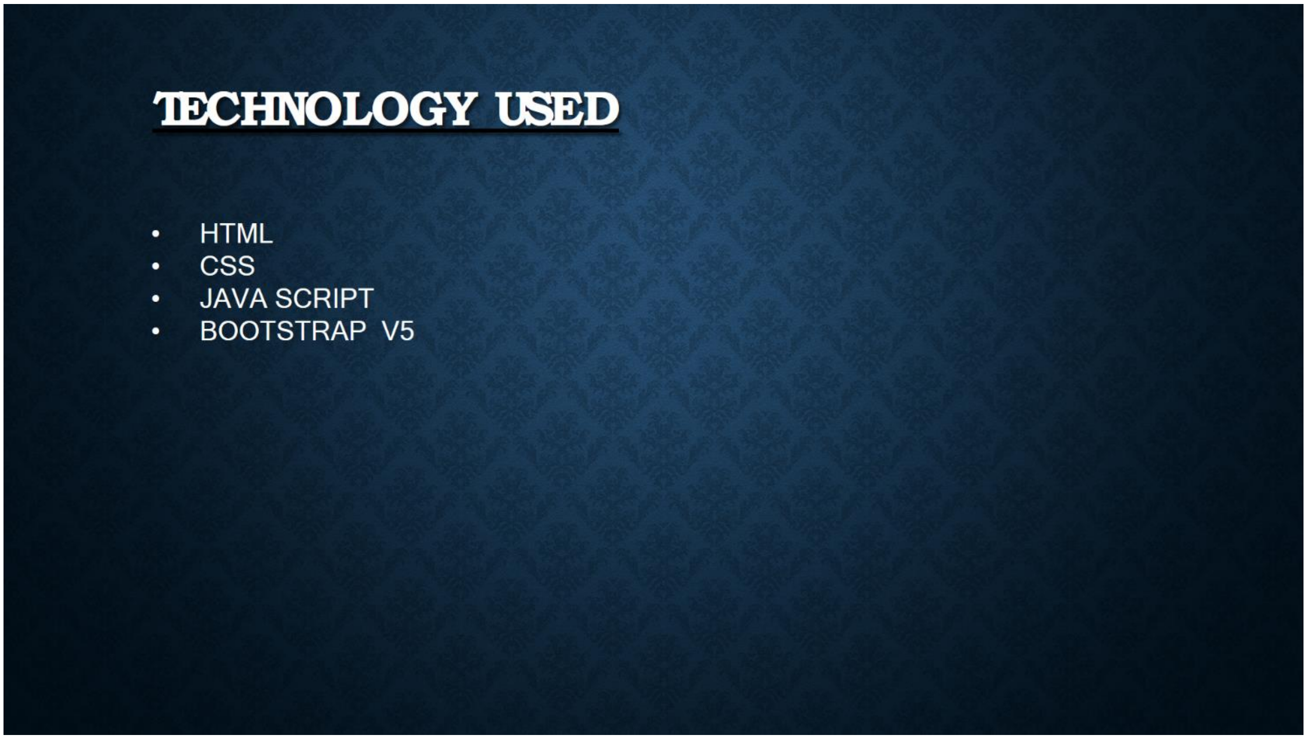
GUIDED BY
Mr. Vinay Dwivedi

SUBMITTED BY
Ayush Rajak
Std.code- B2155R10106082



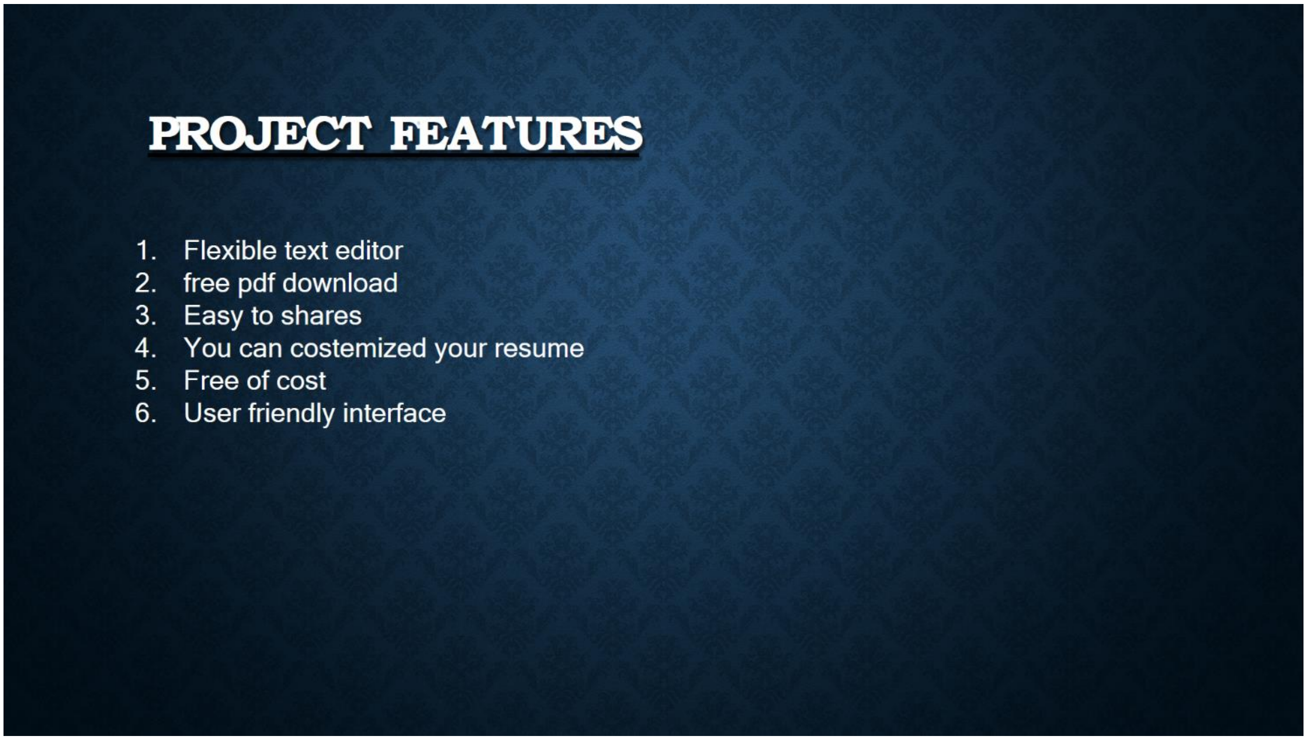
INTRODUCTION

Creating a resume is a bit tedious task for any working professional from any industry. One has to keep it short, simple, with the latest work experience and constantly update it over a while. It is application that simplifies the task of creating a resume for indivisuals. The system is flexible to used and reduces the need of thinking and designing an according to qualifications.



TECHNOLOGY USED

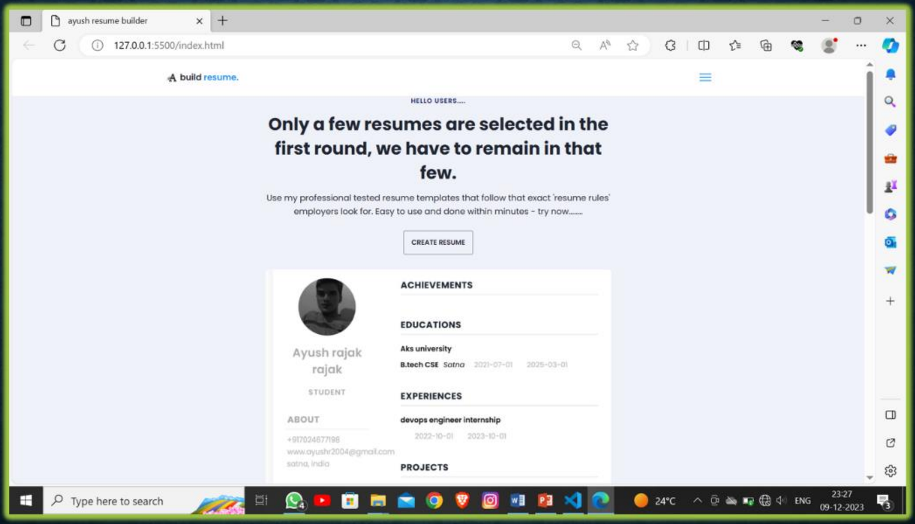
- HTML
- CSS
- JAVA SCRIPT
- BOOTSTRAP V5



PROJECT FEATURES

1. Flexible text editor
2. free pdf download
3. Easy to shares
4. You can costemized your resume
5. Free of cost
6. User friendly interface

WEBSITE SNAPSHOT



build resume.

HELLO USERS....

Only a few resumes are selected in the first round, we have to remain in that few.

Use my professional tested resume templates that follow that exact 'resume rules' employers look for. Easy to use and done within minutes - try now.......

CREATE RESUME

Ayush rajak rajak

STUDENT

ABOUT

+917024677198
www.ayushr2004@gmail.com
satna, india

ACHIEVEMENTS

EDUCATIONS

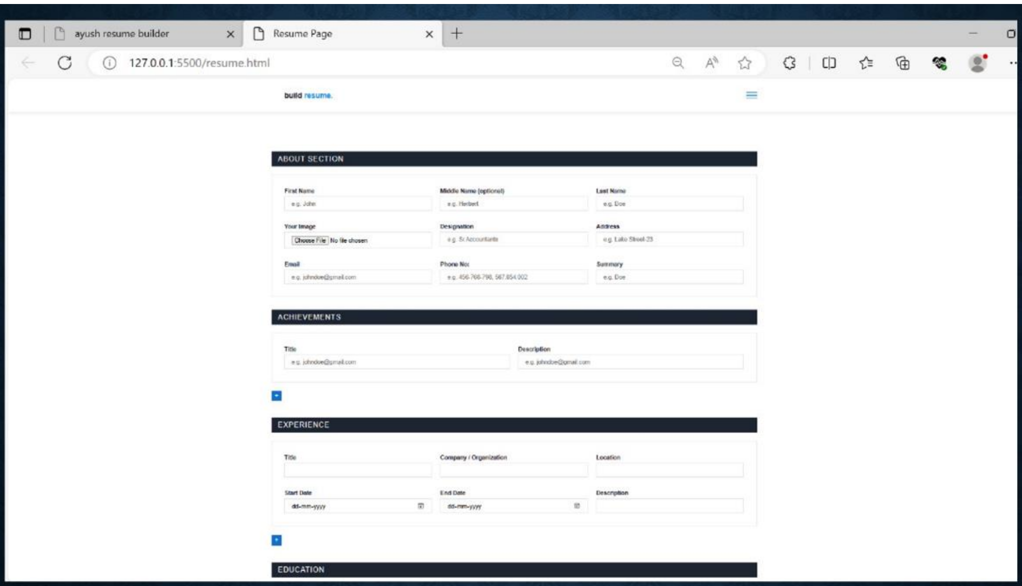
Aks university
B.tech CSE Satna 2021-07-01 2025-03-01

EXPERIENCES

devops engineer internship
2022-10-01 2023-10-01

PROJECTS

Snapshot.1: Home page of web application



build resume.

ABOUT SECTION

First Name: e.g. John
Middle Name (optional): e.g. Herbert
Last Name: e.g. Doe
Your Image: Choose File No file chosen
Designation: e.g. Sr.Accountants
Address: e.g. Lake Street-23
Email: e.g. johndoe@gmail.com
Phone No: e.g. 456-768-798, 567.654.002
Summary: e.g. Doe

ACHIEVEMENTS

Title: e.g. johndoe@gmail.com
Description: e.g. johndoe@gmail.com

+

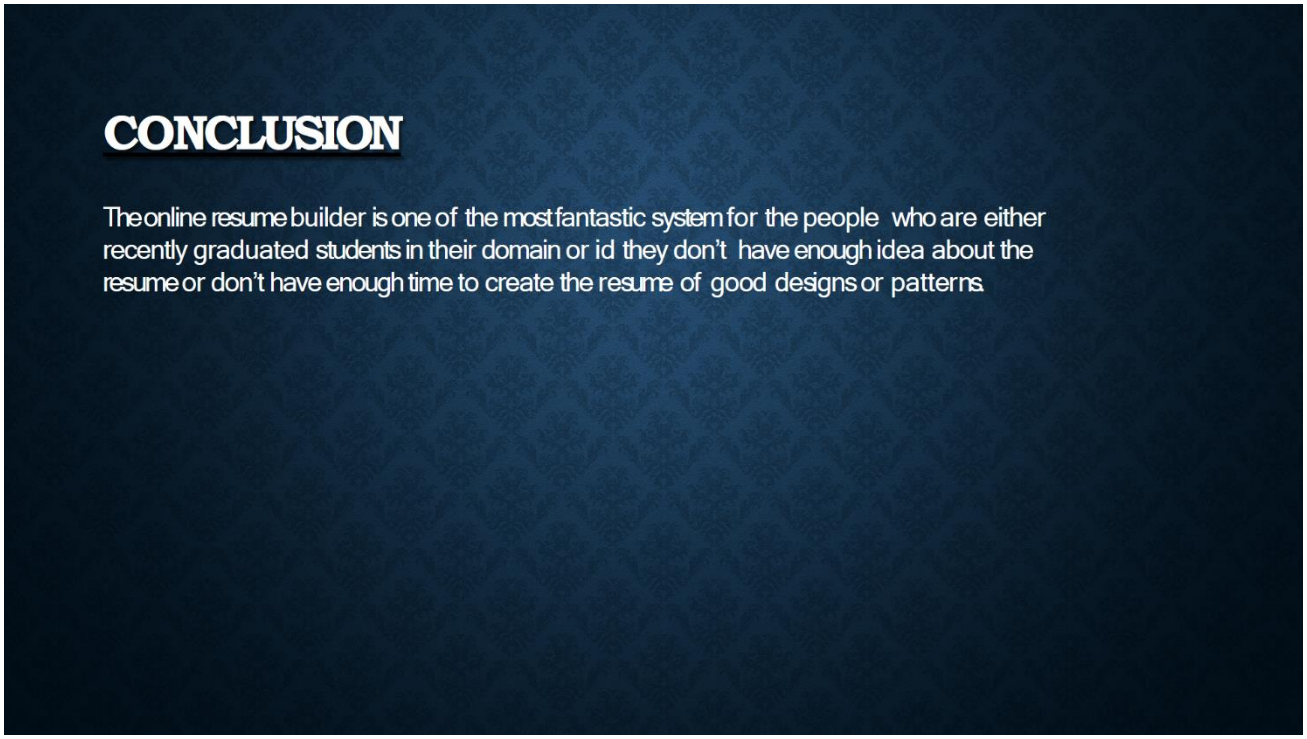
EXPERIENCE

Title
Company / Organization
Location
Start Date: dd-mm-yyyy
End Date: dd-mm-yyyy
Description

+

EDUCATION

Snapshot.2: resume builder input page.



CONCLUSION

The online resume builder is one of the most fantastic system for the people who are either recently graduated students in their domain or id they don't have enough idea about the resume or don't have enough time to create the resume of good designs or patterns.

THANK YOU.....

// THANK YOU FOR READING //
