

Project Progress Report on

CV GENERATING WEB APP

**Submitted in fulfillment of the requirement for the award of the degree
of**

BACHELOR OF COMPUTER APPLICATION

Submitted By:

Ankit Kumar Thakur

2102144

Under the Guidance of

Mr. Aditya Joshi

Assistant Professor

Department of Computer Application



**Department of Computer Application
Graphic Era (Deemed to be University)
Dehradun, Uttarakhand
Jun-2024**

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the Project progress report entitled **“CV Generating web app”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Computer Applications in the Department of Computer Application of the Graphic Era (Deemed to be University), Dehradun shall be carried out by the undersigned under the supervision of **Mr. Aditya Joshi, Assistant Professor**, Department of Computer Application, Graphic Era (Deemed to be University), Dehradun.

Ankit Kumar Thakur

2102144

Sign.

The above mentioned student shall be working under the supervision of the undersigned on the **“CV Generating web app”**

Supervisor

Head of the Department

Examination

Name of the Examiners:

Signature with Date

1.

2.

Work Satisfactory: Yes/ No

CERTIFICATE OF ORIGINALITY

This is to certify that the project report entitled “**CV Generating web app**” submitted to **Graphic Era University, Dehradun** in fulfilment of the requirement for the award of the degree of **BACHELOR OF COMPUTER APPLICATIONS (BCA)**, is an authentic and original work carried out by Mr. Ankit Kumar Thakur with enrolment number GE-21212144 under my supervision and guidance.

The matter embodied in this project is genuine work done by the student and has not been submitted whether to this University or to any other University / Institute for the fulfilment of the requirements of any course of study.

.....

Signature of the Student:

Signature of the Guide

Date :

Date:

Enrolment No.: GE-21212144

Name and Course
of the Student:

Name, Designation and
of the Guide:

Special Note:

ACKNOWLEDGEMENT

I would like to express our deepest gratitude to all those who have contributed to the successful completion of this project on developing “**CV Generating web app**”.

First and foremost, I extend my sincere thanks to our project supervisor, Mr. Aditya Joshi, whose guidance, expertise, and encouragement have been invaluable throughout the development process. Your insightful feedback and unwavering support have greatly enriched the quality of this work.

I also wish to thank the faculty and staff of Department of Computer Application for providing the necessary resources and a conducive environment for research and development. Special thanks to Department of Computer Application for offering technical assistance and access to essential tools and software.

I am grateful to the faculty and staff of Graphic era deemed to be university for providing the necessary resources, facilities, and a conducive environment for the project's execution. Their commitment to fostering a culture of learning and innovation has been instrumental in the project's development.

Lastly, we acknowledge the unwavering support and understanding of our families and friends, who have been a constant source of motivation and encouragement. Your patience and belief in our efforts have been fundamental to our success.

Thank you all for your invaluable contributions and support.

Candidate Name and Signature

Table of Contents

Candidate's Declaration	2
Certificate of Originality	3
Acknowledgements	4
Table of Contents	5
List of Figures	6
Abstract	7
Chapter 1. Introduction and Problem Statement	8-12
Chapter 2. Objectives And Applications	13-17
Chapter 3. System Analysis	18-21
Chapter 4. System Design	22-68
Chapter 5. Coding	69-87
Chapter 6. Result and Output	88-90
Chapter 7. Conclusion	91
Chapter 8. Future work plan	92
Chapter 9. Weekly task	93
Bibliography & References	94-95

List of Figures:

Figure 1.1 Representation of common CV getting rejected

Fig 4.3.1 Client server model

Figure 4.1.2 Describing CSS property

Fig 4.1.4 html,css and js.

Figure 4.9.1 Flow Diagram for CV Generator

Figure4.10.1 Visual Difference between CV and Resume

Figure 5.1 HTML form for input

Figure 5.2 HTML for CV template

Figure 5.3 JavaScript code for functioning

Fig 5.4 Git and Github Flow

Fig 5.5 VS Code logo

Figure 5.6 HTML form for input

Figure 5.7 HTML for CV template

Figure 5.8 JavaScript code for functioning

Figure 6.1 Test case 1

ABSTRACT

In today's competitive job market, a well-crafted CV (Curriculum Vitae) is essential for job seekers to stand out and secure employment opportunities.[1] This web application provides a user-friendly platform for creating professional CVs with ease and efficiency. The web application features a guided step-by-step process, customization options to cater to various industries and personal preferences. Users can input their personal information, work experience, education, skills, and other relevant details, which are then automatically formatted into a polished, professional-looking CV. The platform also includes the ability to export the final document in multiple formats (PDF, Word, etc.). By leveraging modern web technologies, this application aims to simplify the CV creation process, helping users present their qualifications effectively and increase their chances of landing their desired jobs.

Keywords—CV Generator; Curriculum Vitae; WebApplication; Web Development

Chapter 1 Introduction and Problem Statement

1.1 Introduction

In the modern job market, a standout Curriculum Vitae (CV) is a crucial tool for job seekers striving to capture the attention of potential employers. A CV not only highlights a candidate's qualifications, skills, and experience but also serves as the first impression in the recruitment process.[2] However, crafting a compelling and professional CV can be a daunting task for many individuals, especially those who lack design skills or are unfamiliar with industry standards.

This report introduces a web application designed to streamline the CV creation process, making it accessible and straightforward for users of all backgrounds. The CV generating web app offers a comprehensive solution by providing a user-friendly interface, customizable templates, and automated formatting tools. Users can easily input their personal information, work history, education, and other pertinent details, which the application then organizes into a visually appealing and professional document.

The primary goal of this web application is to empower job seekers by simplifying the complexities of CV writing, thereby enhancing their chances of securing employment. By leveraging advanced web technologies and an intuitive design, this application addresses common challenges faced during CV preparation and provides users with a reliable tool to present their qualifications effectively.

This report will explore the features, functionality, and benefits of the CV generating web app, demonstrating its value as an indispensable resource for today's job seekers.

1.2 Problem Statement

The problem Statement for the present work can be stated as follows:

As stated above that, in the highly competitive job market, creating a well-structured and visually appealing Curriculum Vitae (CV) is essential for job seekers to effectively showcase their skills, qualifications, and experience. Despite the importance of a professional CV, many individuals struggle with the task due to a lack of design skills, unfamiliarity with industry standards, and the time-consuming nature of the process. This often results in CVs that fail to capture the attention of employers, diminishing the job seekers' chances of securing interviews and job offers.



Figure 1.1 Representation of common CV getting rejected

Traditional methods of CV creation, such as using word processors or relying on outdated templates, do not adequately address these challenges. They often require significant manual effort to format and customize, leading to inconsistencies and errors. Additionally, these methods do not provide real-time feedback or suggestions for improvement, further complicating the task for users who are unsure of best practices in CV writing.[3]

The problem is exacerbated for individuals who are transitioning into new industries, recent graduates, or those re-entering the workforce, as they may be unfamiliar with the specific requirements and expectations of employers in their target fields. Consequently, there is a clear need for a solution that simplifies the CV creation process, ensures adherence to professional standards, and enhances the overall quality and effectiveness of the final document.

This report addresses these issues by presenting a web-based CV generating application designed to provide an intuitive, efficient, and accessible platform for creating professional CVs. The application aims to eliminate the common obstacles faced by job seekers in CV preparation, offering a comprehensive tool that facilitates the creation of high-quality, industry-appropriate CVs with minimal effort and maximum impact. resource.

1.3 Proposed Solution

To address the challenges faced by job seekers in creating professional and effective CVs, we propose the development of a comprehensive web-based CV generating application. This solution leverages modern web technologies to provide an intuitive and efficient platform that simplifies the CV creation process while ensuring high-quality results. The key features of the proposed solution include[4]:

1. User-Friendly Interface: The application will feature a clean, intuitive interface that guides users through the CV creation process step-by-step. This ensures that even users with minimal technical skills can easily navigate and utilize the tool.

2. Customizable Templates: Users will have access to a variety of professionally designed templates tailored to different industries and job roles. These templates will be fully customizable, allowing users to modify layouts, fonts, colors, and sections to suit their individual preferences and needs.

3. Automated Formatting: The application will automatically format the entered information, ensuring consistency and professionalism. This feature eliminates the manual effort required to adjust margins, align text, and maintain uniformity throughout the document.

4. Real-Time Previews: Users will be able to see real-time previews of their CV as they input their information. This feature provides immediate feedback and allows users to make adjustments on the fly, ensuring the final document meets their expectations.

5. Content Suggestions and Tips: The application will include a library of sample phrases, tips, and best practices for CV writing. It will also offer contextual suggestions based on the information provided, helping users enhance the content and structure of their CVs.

6. Multi-Format Export: Users will have the option to export their CVs in multiple formats, including PDF, Word, and plain text. This flexibility ensures that users can easily submit their CVs in the preferred format of potential employers.

7. Secure Data Management: The application will prioritize user privacy and data security, employing robust encryption and secure storage practices to protect personal information.

8. Mobile Compatibility: The web app will be fully responsive and optimized for mobile devices, allowing users to create and edit their CVs on-the-go.

By incorporating these features, the proposed CV generating web application aims to address the common pain points associated with CV creation. It provides a reliable and accessible tool that enhances the quality and effectiveness of CVs, thereby increasing job seekers' chances of securing employment. This solution not only streamlines the CV writing process but also empowers users to present their qualifications in a professional and impactful manner.

Chapter 2 Objectives And Applications

2.1 Objective:

The proposed work objectives are as follows:

- **Simplify the CV Creation Process:** Provide an intuitive and user-friendly platform that simplifies the task of creating a professional CV, making it accessible to users with varying levels of technical proficiency and design skills.
- **Enhance CV Quality:** Offer high-quality, customizable templates to ensure that users can create visually appealing and professionally formatted CVs that adhere to industry standards.
- **Increase Job Seeker Competitiveness:** Equip users with the tools and guidance needed to craft compelling and well-structured CVs that effectively showcase their qualifications, skills, and experience, thereby increasing their chances of securing job interviews and employment offers.
- **Ensure Flexibility and Accessibility:** Enable users to export their CVs in multiple formats (PDF, Word, plain text) and ensure that the application is mobile-compatible, allowing for seamless use on various devices and operating systems.
- **Provide Industry-Specific Guidance:** Include a library of sample phrases, tips, and best practices tailored to different industries and job roles, helping users to create CVs that meet the specific expectations of their target employers.

By achieving these objectives, the CV generating web application will serve as a valuable tool for job seekers, helping them to present their credentials in the best possible light and navigate the competitive job market with greater confidence and success.

2.2 Applications:

So now the question arises here that, for whom this CV generator is suitable to use? Mainly the use of CV generator can be done by these:

1. Job Seekers

- **Recent Graduates:** New graduates entering the job market can use the application to create their first professional CVs, ensuring that their academic achievements, internships, and skills are presented effectively.
- **Career Changers:** Individuals transitioning to new industries or job roles can leverage industry-specific templates and content suggestions to tailor their CVs to their new career paths.
- **Experienced Professionals:** Professionals seeking new opportunities can quickly update and enhance their existing CVs, highlighting their experience, accomplishments, and skills.

2. Recruitment Agencies:

- **Streamlining CV Preparation:** Recruitment consultants can use the application to assist clients in creating polished CVs that meet employer expectations, thus improving the chances of successful job placements.
- **Consistency and Quality Control:** The application ensures consistency in formatting and presentation, enabling agencies to maintain high standards across all client CVs.

3. Educational Institutions:

- **Career Services:** Universities and colleges can integrate the application into their career services, providing students with a robust tool to prepare professional CVs for internships and job applications.
- **Workshops and Training:** Career counselors can use the application during workshops and training sessions to teach students effective CV writing techniques and best practices.

4. Corporate HR Departments:

- **Internal Mobility:** HR departments can use the application to help employees create CVs for internal job postings and promotions, facilitating career development within the organization.
- **Talent Acquisition:** HR professionals can assist candidates in creating professional CVs, ensuring that potential hires present their qualifications effectively.

5. Freelancers and Contractors and Multiple recruiters:

- **Project Bidding:** Freelancers and contractors can create tailored CVs for specific projects or clients, showcasing relevant skills and experience to increase their chances of winning contracts.
- **Portfolio Integration:** The application can help freelancers integrate their CVs with their portfolios, providing a comprehensive overview of their professional capabilities.
- **Volunteer Recruitment:** Non-profits can use the application to help volunteers create CVs that highlight their skills and experiences, matching them with suitable roles within the organization.

By catering to these diverse user groups, the CV generating web application not only streamlines the CV creation process but also enhances the overall quality and effectiveness of CVs across various contexts.[5] This comprehensive approach ensures that users can present their qualifications in the best possible light, thereby improving their chances of success in their respective endeavors.

2.3 Literature Survey/Related Work

1. Web based resume generator : In this paper abstract says, In reality, most recruiters look over a résumé for no more than 6 seconds on average. To provide valuable inputs into professionally formatted résumé can stand-out as per desired job requirements. Therefore, the purpose of this system is to generate résumé with content recommendations specifically for the IT job field. Researchers opted Waterfall model has been chosen as the methodology because the project is being developed step by step from requirement gathering and analysis to testing and evaluation. Content-based recommendation algorithm is implemented into the system as it can recommend the best skill content suitable for a job application. [6]

2. Web-Application on CV-Building - This academic report presents a research proposal to develop a web-based CV- building platform using ReactJS and the MERN (MongoDB, Express.js, ReactJS, Node.js) stack. Traditional CV creation methods often lack the interactivity and proactive features needed to make an impression on potential employers for a considerable amount of time. The platform's goal is to offer users a versatile and user-friendly interface to create personalized and visually impressive CVs, enhancing their capacity for work. [7]

3. Resume Builder Application with Automated Job Prediction- A resume is a formal document used by individuals to present their backgrounds and skill sets. A resume mainly consists of the individual's educational background, technical skills, work experience, social skills, and awards or publications(if any). This is the first document that describes the individual to an employer or a recruiter. It portrays the individual's image before the interview. Hence, it is essential that one should have an apt resume ready before applying for any company or job. We have proposed a system that will create a formal resume for the user and suggest/predict jobs for the user based on his/her skills. The prediction will help the user in applying for jobs that are aligned with his/her interests and skills. The Resume Builder Application will help users build his/her personal resume. [8]

4. ResumeCraft: A Machine Learning-powered Web Platform for Resume Building -

This paper introduces ResumeCraft, a web-based platform empowering users to build strong resumes and optimize them for ATS compatibility. ResumeCraft leverages Machine Learning (ML) for data analysis and user guidance, while the user interface is built with Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript for a user-friendly experience. The system allows users to input their personal and professional details through a series of form fields, and provides a real-time preview of the resume design as the user inputs their data. The resume generator uses JavaScript to dynamically populate the preview with the user's input, and allows users to select from a range of pre-designed templates and color schemes to customize the look and feel of their resume. [9]

The main reason for designing this web app is the fact that in the competitive job market, the need for a web-based CV generator is paramount to simplify and standardize the CV creation process for job seekers of all backgrounds. Traditional methods of crafting CVs can be time-consuming, error-prone, and daunting for those lacking design skills or familiarity with industry standards. A web-based CV generator provides an intuitive, accessible platform that offers professionally designed templates, automated formatting, real-time previews, and expert content suggestions. This ensures that users can effortlessly create polished, high-quality CVs that effectively showcase their qualifications and experiences, enhancing their chances of securing job interviews and employment opportunities. Additionally, its accessibility from any device allows for flexibility and convenience, meeting the demands of modern job seekers who require efficient and effective tools for career advancement.

Chapter 3 System Analysis

To develop and deploy a Web Application for CV Building, we need to establish a set of system requirements that cover various aspects of the project. The successful development and deployment of the CV generating web application require a comprehensive set of system requirements, encompassing hardware, software, network, and security aspects. These requirements ensure the application is robust, user-friendly, secure, and scalable to handle various user demands.

3.1 Minimum System Requirements

Operating System: Windows 7/8/10, macOS X, Linux

Processor: Intel Core i3 or equivalent

Memory: 4GB RAM

Graphics: Integrated graphics with WebGL support

Storage: 500MB of free space

Network: Broadband Internet connection with 15 Mbps (minimum)

3.2 Recommended System Requirements

Operating System: Windows 10, macOS Mojave, Linux (latest distribution)

Processor: Intel Core i5 or equivalent

Memory: 8GB RAM

Graphics: Dedicated GPU with 2GB VRAM

Storage: 1GB of free space

Network: Broadband Internet connection with 45 Mbps (recommended)

3.3 Software Requirements

The software requirements for a web-based CV generator encompass various components that ensure the easy and smooth accessibility, is accessible to a wide range of users, and provides a secure and engaging experience. Here's an overview of the software requirements.

3.3.1 Operating Systems

CV generator should be compatible with the most commonly used operating systems to ensure a broad user base:

Windows: Windows 7 and later versions.

macOS: macOS X and later versions.

Linux: Most popular distributions.

3.3.2 Web Browsers

TheWebApp should be Operatable on multiple web browsers, requiring them to be up-to-date to support the latest web technologies:

Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, Opera.

3.3.3 Programming Languages

The development of CV generator will utilize various programming languages for different aspects of the game:

HTML5: For structuring and presenting content on the web.

CSS3: For styling and layout of the game interface.

JavaScript: For logic and interactivity on the client side.

Frameworks and Libraries

Frameworks and libraries can expedite development and provide additional functionality.

3.4. Hardware Requirements

The hardware requirements for a web-based CV generator are designed to ensure that the Builder is accessible to a wide range of people while providing a smooth and enjoyable experience. Here's an overview of the recommended hardware specifications.

3.4.1 Processor (CPU)

Minimum: A modern dual-core processor.

Recommended: A quad-core processor or higher, such as Intel i5 or AMD Ryzen 5, for optimal performance¹.

Memory (RAM)

Minimum: At least 4 GB of RAM to handle the game's operations without significant lag².

Recommended: 8 GB of RAM or more to ensure smooth multitasking and quick access to game data.

3.4.2 Graphics (GPU)

Minimum: Integrated graphics with support for WebGL to render the game's graphics.

Recommended: A dedicated GPU with at least 2 GB of VRAM for higher quality visuals and effects. Support for Vulkan 1.2 is also recommended for enhanced graphical capabilities².

3.4.3 Storage

Minimum: 500 MB of free space for the game files, excluding additional project files².

Recommended: 1 GB or more of free space to accommodate future updates and expansions.

Network

Minimum: A stable broadband Internet connection with at least 15 Mbps speed to handle online gameplay without interruptions.

Recommended: A faster Internet connection with 45 Mbps speed or higher to reduce latency, especially for multiplayer modes.

3.4.4 Display

Minimum: A display capable of rendering 720p resolution.

Recommended: A display capable of 1080p resolution or higher for a more immersive gaming experience.

3.4.5 Sound

Minimum: Onboard sound or integrated speaker system.

Recommended: A dedicated sound card or external speakers/headphones for a richer audio experience.

3.4.6 Input Devices

Minimum: A standard keyboard and mouse or touchpad.

Recommended: A mouse and mechanical keyboard for more precise control, or a gamepad for players who prefer console-style controls.

These hardware requirements are based on general standards for web-based CV Builder and should be adjusted according to the specific demands of user and graphical fidelity. Ensuring that peoples' systems meet or exceed these specifications will contribute to a seamless and engaging experience.

3.4.7 Usability Requirements

- User Interface: Intuitive and user-friendly interface with accessibility features
- Cross-Platform Compatibility: Consistent performance across different operating systems and devices
- Mobile Responsiveness: Fully responsive design for seamless mobile use

By adhering to these system requirements, the CV generating web application will be well-equipped to deliver a reliable, efficient, and secure service that meets the needs of job seekers, recruitment agencies, educational institutions, and other stakeholders.

Chapter 4 System Design

System design is a fundamental phase in the development process that sets the foundation for a successful application. It involves creating a blueprint for a system that meets specific requirements and provides a seamless user experience. Let's explore the core principles and components that constitute an effective system design. The core principles of system design are essential for creating a robust, efficient, and scalable system. These principles guide the development process to ensure that the system meets the needs of users and can adapt to future requirements. Here's an in-depth look at each principle:

4.1 Core Principles

4.1.1 Security:

Ensures the confidentiality, integrity, and availability of data.

Incorporates measures like encryption, firewalls, and intrusion detection systems.

Involves regular security audits and updates to safeguard against new threats.

4.1.2 Reliability:

The system is dependable and operates consistently over time.

Includes redundancy and failover mechanisms to handle potential failures.

Relies on thorough testing to guarantee system behavior under various conditions.

4.1.3 Scalability:

Ability to handle increased loads without performance degradation.

Scales horizontally (adding more nodes) or vertically (upgrading existing nodes).

Considers load balancing and resource allocation for optimal performance.

4.1.4 Maintainability:

The system is easy to maintain and update without significant downtime.

Adopts modular design to isolate changes and reduce impact on the entire system.

Documentation is kept up-to-date to assist with maintenance efforts.

4.1.5 Efficiency:

Optimizes resource usage to deliver maximum performance.

Reduces latency and increases throughput for a better user experience.

Employs caching, optimized algorithms, and data compression techniques.

4.1.6 Usability:

The system is user-friendly and intuitive to navigate.

Provides clear feedback and guidance to users.

Accessibility features are included to cater to all user groups.

4.1.7 Flexibility:

The system can accommodate changes in technology and user requirements.

Supports integration with other systems and platforms.

Allows for customization and extension by users or developers.

4.1.8 Portability:

The system can be easily transferred from one environment to another.

Adheres to open standards and avoids vendor lock-in.

Considers cross-platform compatibility during design.

4.1.9 Testability:

The system can be effectively tested to identify and fix issues.

Includes automated tests for regression and performance benchmarks.

Test environments mimic production settings for accurate results.

4.1.10 Documentation:

Comprehensive documentation is provided for users and developers.

Includes system architecture, code comments, and user manuals.

Facilitates knowledge transfer and onboarding of new team members.

4.1.11 Interoperability:

The system works well with other systems and follows industry standards.

Uses APIs and data exchange formats for seamless integration.

Ensures compatibility with different data sources and services.

4.1.12 Modularity:

The system is divided into discrete components that can function independently.

Promotes reuse of components across different parts of the system.

Simplifies updates and maintenance by isolating changes to specific modules.

4.2 Components of System Design.

4.2.1 User Interface (UI):

The UI is the visual layer that users interact with. It includes:

4.2.1.1 Navigation: Clear menus, buttons, and links for seamless movement within the application.

4.2.1.2 Responsive Design: Ensuring the UI adapts to different screen sizes (desktop, tablet, mobile).

4.2.1.3 Accessibility: Making the UI usable for people with disabilities (e.g., screen readers, keyboard navigation).

4.2.1.4 Consistent Styling: A cohesive look and feel across all pages.

4.2.2 Backend Logic:

This component handles the application's core functionality:

4.2.2.1 API Endpoints: Defining routes for data retrieval, updates, and business logic.

4.2.2.2 Controllers: Processing requests, validating input, and orchestrating actions.

4.2.2.3 Services: Abstracting complex logic (e.g., interacting with external APIs, handling payments).

4.2.2.4 Middleware: Implementing cross-cutting concerns (e.g., authentication, logging).

4.2.3 Database:

The database stores and manages data:

4.2.3.1 Schema Design: Creating tables, defining relationships (e.g., one-to-many, many-to-many).

4.2.3.2 Indexes: Optimizing query performance by indexing relevant columns.

4.2.3.3 Normalization: Ensuring data integrity by avoiding redundancy.

4.2.3.4 Data Migration: Handling schema changes over time.

4.2.4 Networking:

Communication between components and external services:

4.2.4.1 RESTful APIs: Defining endpoints for data exchange.

4.2.4.2 WebSocket: Real-time communication (e.g., chat features).

4.2.4.3 CORS (Cross-Origin Resource Sharing): Allowing or restricting access from different domains.

4.2.4.4 Load Balancing: Distributing traffic across multiple servers.

4.2.5 Security:

Protecting the system from threats:

4.2.5.1 Authentication: Verifying user identity (e.g., OAuth, JWT).

4.2.5.2 Authorization: Controlling access to specific features based on roles and permissions.

4.2.5.3 Encryption: Securing data in transit (HTTPS) and at rest (database encryption).

4.2.5.4 Input Validation: Preventing SQL injection, XSS attacks, and other vulnerabilities.

4.3 System Architecture

System architecture plays a pivotal role in determining the overall performance, scalability, and maintainability of an application. In the context of your “CV generator” project, understanding the client-server model and designing the frontend components (HTML, CSS, and JavaScript) is crucial.

4.3.1 Client-Server Model

The client-server model is a fundamental architectural pattern where software systems are divided into two primary components. The Client-Server Model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. This model is central to many networked applications, including email, network printing, and the World Wide Web. In the digital world, a Client is a computer (Host) that is capable of receiving information or using a particular service from the service providers (Servers). A Server, on the other hand, is a remote computer that provides information (data) or access to particular services. So, it is the Client requesting something and the Server serving it as long as it is in the database.

Client:

The client represents the user interface (UI) or the end-user application.

It runs on the user’s device (e.g., web browser, mobile app).

Responsible for presenting information to the user and handling user interactions.

Communicates with the server to request data or perform actions.

Server:

The server hosts the backend logic and data.

It runs on a centralized machine or a cluster of machines.

Handles business logic, data processing, and storage.

Responds to client requests by providing data or performing actions.

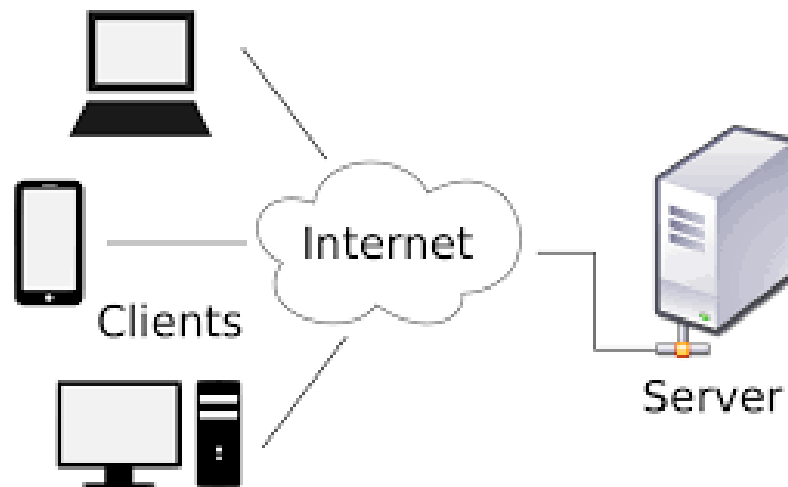


Fig 4.3.1 client server model

Till now we have clear idea of building a Web application for Building CVs. For which we would be using the basic HTML, CSS and JS, along with the basic use of Bootstrap, that provides a collection of HTML, CSS, and JavaScript components and tools that enable developers to build responsive, mobile-first websites with ease. So we rely only on JavaScript for the functioning of the whole Web Application and not any other Designing tool.

The Reason of using JS is to show the power and ability that JS provides and stills competes with other languages and framework.

Building a CV generating web application using JavaScript involves several key steps, from initial design to deployment. First, the front-end is developed using a framework like React.js to create a dynamic and responsive user interface. This UI allows users to sign up, log in, and access a dashboard where they can create, edit, and manage their CVs. Users can choose from various templates and input personal information through intuitive forms. The state management is handled using Redux to ensure data consistency across the application.

For the back-end, Node.js with Express.js is used to build the server-side logic. This involves setting up RESTful API endpoints for user authentication, CV management, and template handling. Authentication is secured using JSON Web Tokens (JWT) to manage user sessions. The application stores user data and CV content in a MongoDB database, which allows for

flexible and scalable data management. The back-end also handles the generation of CVs in different formats like PDF and Word, utilizing libraries such as pdf-lib or docx.

To manage templates and static assets, AWS S3 is used for storage, providing a scalable and reliable solution for file management. The entire application is containerized using Docker, facilitating consistent environments from development to production. Kubernetes is then employed for orchestration, ensuring the application can scale efficiently to handle varying loads.

Security is paramount, so HTTPS is enforced for all communications, and sensitive data is encrypted both in transit and at rest. Regular security audits and the implementation of best practices in coding help protect against common vulnerabilities. Continuous integration and deployment (CI/CD) pipelines are set up using GitHub Actions, automating the testing and deployment process to ensure that updates can be rolled out smoothly and efficiently.

Finally, monitoring and logging are implemented using Prometheus and Grafana for performance monitoring, and the ELK stack (Elasticsearch, Logstash, Kibana) for logging. This setup allows for real-time monitoring and quick troubleshooting of any issues that arise. By following this approach, a robust, scalable, and user-friendly CV generating web application can be developed using JavaScript.

4.4 MODULES AND LIBRARIES

The Modules that we would be using in building the model are:

1. **HTML-** HTML, or HyperText Markup Language, is the standard language used to create and design documents on the web. It structures web content by using elements and tags to define the layout and format of text, images, links, and other multimedia. HTML is a cornerstone technology of the World Wide Web, alongside CSS (Cascading Style Sheets) and JavaScript.



HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. The inclusion of CSS defines the look and layout of content. [10]

Elements and Tags: HTML uses “elements” to mark up the structure and to define content. Elements are denoted by “tags,” which label pieces of content such as “heading,” “paragraph,” “table,” and so on.

Attributes: Elements can have attributes that define additional characteristics or provide metadata. For example, the href attribute of an a tag sets the URL for a hyperlink.

Document Object Model (DOM): The DOM represents the document as a tree structure where each node is an object representing a part of the document.

HTML5: The latest evolution of HTML, providing new elements and APIs for more complex web applications.

Semantic Elements: Elements like <article>, <footer>, <header>, and <section> provide better document structure and are used to define the different parts of a web page.

`<!DOCTYPE html>`: Declares the document type and version of HTML.

`<html>`: The root element that wraps the entire HTML content.

`<head>`: Contains meta-information about the document, like its title and links to scripts and stylesheets.

`<body>`: Encloses all the content that appears on the web page, such as text, images, and videos.

Key Features of HTML:

- **Elements and Tags:**

HTML documents are composed of elements, which are defined using tags. Tags are enclosed in angle brackets, and they often come in pairs: an opening tag `<tagname>` and a closing tag `</tagname>`. Some tags, like `` and `
`, are self-closing.

- **Document Structure:**

An HTML document typically starts with a `<!DOCTYPE html>` declaration, followed by the `<html>` tag that encloses the entire document. The document is divided into two main sections:

- Head (`<head>`): Contains meta-information about the document, such as the title (`<title>`), character set (`<meta>`), styles (`<style>` or links to CSS files), and scripts (`<script>`).
- Body (`<body>`): Contains the actual content of the document, including text, images, links, and other media.

- **Text Formatting:**

HTML provides a variety of tags to format text, such as `<h1>` to `<h6>` for headings, `<p>` for paragraphs, `<a>` for links, `` and `` for bold and italic text, respectively.

- **Lists:**

HTML supports ordered () and unordered () lists, with list items defined using the tag.

- **Tables:**

Tables are created using the <table> tag, with rows defined by <tr>, headers by <th>, and data cells by <td>.

- **Forms:**

HTML forms (<form>) are used to collect user input. They include various input elements like <input>, <textarea>, <select>, and <button>.

- **Media:**

HTML allows the embedding of images (), audio (<audio>), and video (<video>), as well as other media types.

- **Semantic Elements:**

Modern HTML includes semantic elements that provide meaning to the structure of the document, such as `<header>`, `<footer>`, `<article>`, `<section>`, and `<nav>`. These elements enhance accessibility and SEO.

- **Links:**

Hyperlinks are created using the `<a>` tag, allowing navigation between different web pages and resources.

- **Attributes:**

HTML tags can have attributes that provide additional information about the elements, such as `id`, `class`, `src` (for images), `href` (for links), and `style` (for inline CSS). OpenCV is written natively in C++ and has a template interface that works seamlessly with STL containers.

2. **CSS-** CSS, or Cascading Style Sheets, is a stylesheet language used to describe the presentation and design of HTML documents. It controls the layout, colors, fonts, and overall visual appearance of web pages, allowing developers to separate content (HTML) from presentation (CSS)



CSS is designed to enable the separation of content and presentation, including layout, colors, and fonts. This separation can improve content accessibility;[further explanation needed] provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting. [11]

Key Features of CSS:

- **Selectors:**

CSS selectors are used to target HTML elements that you want to style. Common types of selectors include:

- Element Selector: Targets elements by their tag name (e.g., `p { }`).
- Class Selector: Targets elements by their class attribute (e.g., `.classname { }`).
- ID Selector: Targets elements by their id attribute (e.g., `#idname { }`).

- Attribute Selector: Targets elements by their attributes (e.g., [type="text"] { }).

- **Properties and Values:**

CSS uses properties and values to define the styles for elements. For example:

CSS

```
p {  
  color: blue;  
  font-size: 16px;  
  margin: 10px;  
}
```

Figure 4.1.2 Describing CSS property

In this example, color, font-size, and margin are properties, and blue, 16px, and 10px are their respective values.

- **Cascading and Specificity:**

The term "cascading" in CSS means that styles can cascade from one stylesheet to another, allowing multiple stylesheets to be combined. CSS follows a specific order of precedence:

- Inline styles (inside an HTML element) have the highest precedence.
- Internal styles (within a <style> tag in the HTML document) have medium precedence.
- External styles (linked from an external CSS file) have the lowest precedence.
- Specificity determines which styles are applied when there are conflicting rules. Inline styles have the highest specificity, followed by IDs, classes, and finally element selectors.

- **Box Model:**

The CSS box model describes the rectangular boxes that are generated for elements in the document tree. It consists of:

- Content: The actual content of the element (text, image, etc.).
- Padding: The space between the content and the border.
- Border: The border that surrounds the padding (if any).
- Margin: The space outside the border.

- **Layouts:**

CSS provides various layout techniques:

- Floats: Used to float elements to the left or right of a container.
- Flexbox: A layout model for creating flexible and responsive layouts.
- Grid: A powerful 2-dimensional system for creating complex layouts.

- **Responsive Design:**

CSS enables responsive design to ensure web pages look good on all devices (desktops, tablets, and mobile phones). Media queries are used to apply different styles for different screen sizes:

- 3. JavaScript-** JavaScript is a versatile, high-level programming language that is widely used for creating interactive and dynamic web pages. Initially developed by Netscape in 1995, JavaScript has evolved into one of the core technologies of web development, alongside HTML and CSS. It enables developers to enhance the user experience by allowing web pages to respond to user actions, manipulate the DOM (Document Object Model), and communicate asynchronously with servers.



JavaScript (/ˈdʒɑːvəskript/), often abbreviated as JS, is a programming language and core technology of the Web, alongside HTML and CSS. 99% of websites use JavaScript on the client side for webpage behavior. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard.[12]

It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).[13]

A majority of websites use it and all major web browsers have a dedicated JavaScript engine to execute it. JavaScript supports event-driven, functional, and imperative programming styles, and includes APIs for working with text, arrays, dates, regular expressions, and basic manipulation of the Document Object Model (DOM).

Furthermore, JavaScript has also been used in non-web contexts like in PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines and platforms built upon them have also increased the popularity of JavaScript for server-side web applications.

JavaScript was initially created in 10 days in May 1995 by Brendan Eich, while he was an engineer at Netscape. JavaScript was first released with Netscape 2 early in 1996. It was originally going to be called LiveScript, but it was renamed in an ill-fated marketing decision that attempted to capitalize on the popularity of Sun Microsystems's Java language — despite the two having very little in common. This has been a source of confusion ever since.

JavaScript, not to be confused with Java, is standardized in the ECMAScript language specification and is a prototype-based, multi-paradigm, dynamic language with object-oriented, imperative, and declarative (including functional programming) styles.

JavaScript has a C-style syntax, which includes curly-brace syntax, semicolons to separate statements, and many other familiar features. JavaScript allows for the manipulation of HTML document content and structure via the Document Object Model (DOM), enabling changes to be made on-the-fly and creating highly dynamic user experiences.

Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

JavaScript also supports asynchronous programming with callbacks, promises, and `async/await`, and has robust support for error handling with `try/catch/finally`. Modern JavaScript has added support for classes and modules, but these are essentially

syntactic sugar over JavaScript's existing prototype-based inheritance and scoping mechanisms.

JavaScript engines have made tremendous strides in performance, and modern JavaScript engines, such as V8 (Chrome, Node.js), SpiderMonkey (Firefox), and Chakra (Edge), are extremely fast, thanks to just-in-time compilation and other advanced optimization techniques.

JavaScript's flexibility and its expressive, dynamic nature make it a great choice for all sorts of applications, from small scripts to large-scale application development. However, JavaScript's dynamic nature can also make it challenging to write robust, error-free code, and has led to a proliferation of libraries and frameworks aimed at addressing these and other challenges.



Key Features of JavaScript:

- **Dynamic Typing:**

JavaScript is dynamically typed, meaning variables do not require a predefined type and can hold any type of data at any time. This provides flexibility but also requires careful handling to avoid type-related errors.

- **Object-Oriented:**

JavaScript supports object-oriented programming, allowing the creation of objects to model real-world entities. It uses prototypes rather than classical inheritance, making it unique compared to traditional OOP languages like Java or C++.

- **First-Class Functions:**

Functions in JavaScript are first-class citizens, meaning they can be assigned to variables, passed as arguments, and returned from other functions. This feature is fundamental for functional programming and higher-order functions.

- **Event-Driven:**

JavaScript can respond to user interactions such as clicks, form submissions, and keyboard events. Event handling is crucial for creating interactive web applications.

- **Asynchronous Programming:**

JavaScript supports asynchronous operations using callbacks, promises, and the `async/await` syntax. This is essential for tasks like fetching data from APIs without blocking the main execution thread.

- **Cross-Platform:**

JavaScript runs in web browsers, making it inherently cross-platform. Additionally, with the advent of environments like Node.js, JavaScript can also be used for server-side development.

- **Wide Ecosystem:**

JavaScript has a vast ecosystem of libraries and frameworks, such as React, Angular, and Vue.js for front-end development, and Node.js for back-end development. Tools like npm (Node Package Manager) facilitate easy management of dependencies.

Key Concepts:

- **Variables and Data Types:**

JavaScript supports various data types including numbers, strings, booleans, objects, arrays, and special values like null and undefined. Variables can be declared using var, let, or const, with let and const being the preferred options due to block scoping.

- **Functions:**

Functions are defined using the function keyword, and ES6 introduced arrow functions for a more concise syntax. Functions can be anonymous, and higher-order functions can take other functions as arguments or return them.

- **Objects and Arrays:**

Objects are collections of key-value pairs, and arrays are ordered collections of values. Both are essential for managing data structures in JavaScript.

- **Control Structures:**

JavaScript includes traditional control structures such as if statements, for loops, while loops, and switch statements, allowing for complex decision-making and iteration.

- **DOM Manipulation:**

JavaScript interacts with HTML through the DOM, enabling dynamic updates to the content and structure of web pages. Methods like getElementById, querySelector, and createElement are commonly used for DOM manipulation.

- **Event Handling:**

JavaScript handles user interactions through events. Event listeners can be added to elements to execute code in response to user actions like clicks, key presses, and form submissions.

- **Asynchronous Programming:**

Asynchronous operations are crucial for web applications, allowing tasks like data fetching, timers, and I/O operations to run without blocking the main thread. Promises and async/await syntax simplify handling asynchronous code.

- **Modules:**

ES6 introduced modules, allowing code to be split into reusable pieces. The import and export keywords enable the use of modules, promoting better organization and maintainability of code.

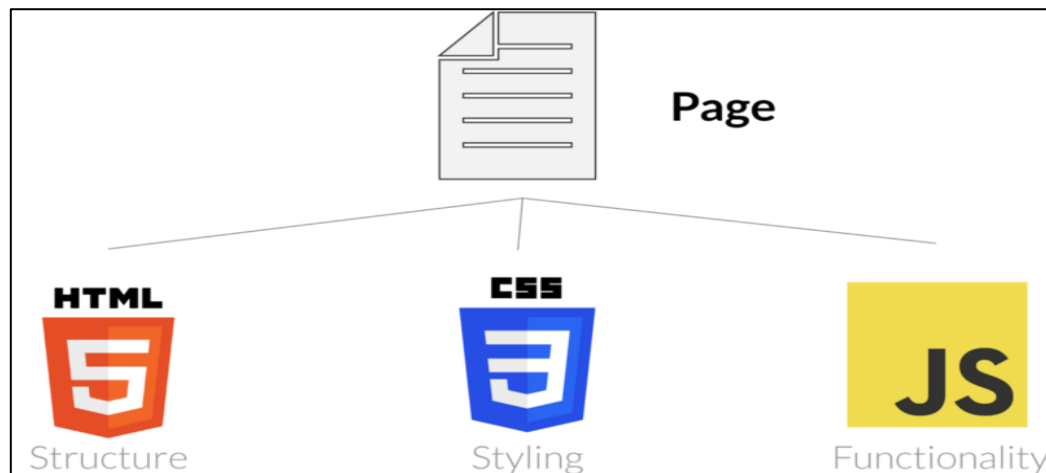


Fig 4.1.4 html,css and js.

- 4. BootsStrap** - Bootstrap is a popular open-source front-end framework that simplifies the development of responsive and visually appealing web applications. Developed by Twitter and released in 2011, Bootstrap provides a collection of CSS and JavaScript tools designed to make the process of creating web pages faster and more efficient. It is widely used for its ease of use, flexibility, and ability to create consistent layouts across different browsers and devices.



Bootstrap (formerly Twitter Bootstrap) is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

As of May 2023, Bootstrap is the 17th most starred project (4th most starred library) on GitHub, with over 164,000 stars.[14] According to W3Techs, Bootstrap is used by 19.2% of all websites.

Bootstrap is an HTML, CSS and JS library that focuses on simplifying the development of informative web pages (as opposed to web applications). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking.

Key Features of Bootstrap:

- **Responsive Design:**

Bootstrap includes a responsive grid system that allows developers to create fluid layouts that adapt to different screen sizes. This grid system is based on a 12-column layout and includes classes for different breakpoints to handle various devices such as desktops, tablets, and mobile phones.

- **Pre-styled Components:**

Bootstrap comes with a variety of pre-styled components such as buttons, forms, tables, navigation bars, alerts, modals, and carousels. These components are designed to be easily customized and integrated into web pages, saving time on basic styling.

- **CSS Classes:**

Bootstrap provides a rich set of CSS classes that simplify styling elements. These classes cover a wide range of functionalities, including typography, spacing, alignment, and visibility.

- **JavaScript Plugins:**

Bootstrap includes a set of JavaScript plugins that enhance the functionality of web pages. These plugins, based on jQuery, enable features such as modals, tooltips, popovers, dropdowns, and carousels. They can be easily activated using data attributes or JavaScript.

- **Customizable:**

Bootstrap is highly customizable, allowing developers to override default styles and create custom themes. The framework uses Sass (Syntactically Awesome Style Sheets) for its styles, enabling the use of variables, nested rules, and mixins to create custom designs.

- **Consistent Design:**

By using Bootstrap, developers can ensure a consistent design across different parts of their web application. This consistency is achieved through a unified set of styles and components.

Key Concepts:

- **Grid System:**

The grid system in Bootstrap is a flexible and responsive layout structure. It uses a series of containers, rows, and columns to layout and align content. Columns can be combined to create wider columns, and they will stack on smaller devices.

- **Responsive Utilities:**

Bootstrap includes utility classes for showing, hiding, or altering the appearance of elements based on the viewport size. These classes include `.d-none`, `.d-block`, `.d-md-none`, and others.

- **Predefined Classes:**

Bootstrap provides a wide range of predefined classes for typography, forms, buttons, images, and more. These classes make it easy to apply consistent styling across different elements.

- **JavaScript Components:**

Bootstrap's JavaScript components enhance interactivity. These components are activated using data attributes or JavaScript code.

- **Customization:**

Developers can customize Bootstrap by modifying the Sass variables and recompiling the CSS. This allows for a tailored look and feel that matches the branding and design requirements of a project.

Bootstrap is a powerful tool for web development, offering a comprehensive set of styles and components that streamline the creation of responsive and modern web pages. Its ease of use, flexibility, and extensive documentation make it a popular choice among developers for both small projects and large-scale web applications.

4.5 Responsive Design Considerations

Responsive design is an approach to web development that ensures a website can adapt and provide an optimal viewing experience across a wide range of devices, from desktop computers to mobile phones. The goal is to create web pages that detect the visitor's screen size and orientation and change the layout accordingly. This adaptability means that, regardless of the device, a site's content will be readable and navigable with a minimum of resizing, panning, and scrolling.

The core of responsive design lies in fluid grids, flexible images, and media queries. Fluid grids are based on percentage values instead of absolute units like pixels, which allows the layout to expand or contract with the browser or device window. Flexible images are also sized in relative units to prevent them from displaying outside their containing element. Media queries, a feature of CSS3, enable the creation of different viewing scenarios. They test for the presence of certain conditions and apply predefined styles when those conditions are met.

When considering responsive design, it's important to start with a mobile-first approach. This means designing for the smallest screen first and then adding more features and content for larger screens. This approach prioritizes performance and user experience on mobile devices, where constraints such as bandwidth and screen size are more pronounced.

Typography is another critical aspect of responsive design. Text must remain legible across all devices, which often requires different font sizes and line heights for different screen widths. System fonts can be a good choice for performance, as they do not require downloading additional resources. However, the choice of typography should also consider the brand identity and readability.

Navigation is a significant challenge in responsive design. On desktops, there's usually enough space to display a full menu, but on mobile devices, screen real estate is limited. A

common solution is to use a hidden navigation menu, which can be toggled with a button (often represented by the ‘hamburger’ icon). This keeps the interface clean while still providing access to all parts of the site.

Buttons and other interactive elements must be designed with touch targets in mind. They should be large enough to be easily tapped with a thumb without the risk of hitting the wrong button. This consideration is part of the larger field of touch ergonomics, which also includes the placement of elements in areas that are easy to reach during one-handed or two-handed use.

Not all content that is shown on a desktop needs to be displayed on a mobile device. Sometimes, less is more, especially when it comes to mobile experiences. Content should be prioritized based on what mobile users need most, and additional content can be added for desktop views. This not only improves loading times but also focuses the user’s attention on the most important information.

Performance is a key consideration in responsive design. Mobile devices often have slower internet connections and less powerful processors than desktop computers. Optimizing images, minifying CSS and JavaScript, and leveraging browser caching can all help to improve load times and the overall speed of the site.

Accessibility should never be an afterthought in responsive design. All users, regardless of their device or any disabilities, should be able to access and use the site effectively. This includes considerations for screen readers, keyboard navigation, and ensuring that all interactive elements are accessible.

Testing is a crucial part of the responsive design process. A site must be tested on multiple devices, browsers, and operating systems to ensure compatibility and performance. This can be done using physical devices, emulators, or online testing services.

4.6 User Experience

User Experience (UX) is a broad and multifaceted field that encompasses all aspects of a user's interaction with a product, system, or service. It's about designing products that are easy to use, enjoyable, and effective at solving user needs. UX is not just about usability; it's about the entire journey a user takes, from their first interaction with a product to their daily use.

The goal of UX design is to create a seamless, simple, and engaging experience for users. This involves understanding the user's needs and goals, the context in which the product will be used, and the tasks that the user will perform. UX designers use a variety of techniques to achieve this, including user research, persona creation, user flow mapping, wireframing, prototyping, and usability testing.

User research is a critical part of UX design. It involves gathering data about the user's needs, behaviors, and preferences. This can be done through methods like interviews, surveys, and observations. The insights gained from user research can help designers make informed decisions about the product's design.

Personas are fictional characters created based on user research. They represent the different user types that might use a product. Personas help designers understand the users' needs, experiences, behaviors, and goals, which can guide the design process.

User flow mapping is a visual representation of the different paths a user can take when interacting with a product. It helps designers understand the user's journey and identify potential issues or bottlenecks.

Wireframing is a low-fidelity representation of a product's layout. It serves as a blueprint for the product and helps designers organize and structure the content.

Prototyping is the creation of a working model of a product. It allows designers to explore ideas, test the product's functionality, and gather feedback before the final product is developed.

Usability testing involves evaluating a product by testing it with users. It helps designers identify any usability issues and make improvements to the product.

In conclusion, UX design is a user-centered process that aims to create products that provide meaningful and relevant experiences to users. It involves a deep understanding of users, a holistic view of user journeys, and a continuous process of testing and refining the product. A good UX design can lead to increased user satisfaction, higher engagement, and ultimately, the success of a product.

While this is a brief overview, the field of UX is vast and constantly evolving, with new methods and approaches being developed all the time. Whether you're a UX designer, a product manager, a developer, or just someone interested in creating better products, understanding the principles of UX design can be incredibly valuable. It's a field that offers endless opportunities for learning, creativity, and making a real impact on people's lives.

4.7 Version control

Version control, also known as source control, is a system that records changes to a file or set of files over time so that you can recall specific versions later. It's a crucial tool in software development, allowing developers to work together on a project without stepping on each other's toes.

Version control systems work by creating a repository, which is a database of changes made to files and directories. Each change or set of changes creates a new "version" of the project, which is stored in the repository. This allows developers to easily track changes, find and fix bugs, and even revert to older versions of the project if necessary.

There are two main types of version control systems: centralized and distributed. Centralized version control systems, like Subversion and Perforce, use a single, central repository. Developers get a working copy of the files, but all changes are saved into the central repository.

On the other hand, distributed version control systems, like Git and Mercurial, give each developer their own local repository, complete with a full history of changes. Developers can make changes in their local repository, then push those changes to other repositories. This allows for more flexible workflows, as developers can work offline or in their own branches without affecting others.

Version control systems also provide mechanisms for branching and merging. Branching allows developers to create a separate line of development, which is useful for working on new features or experimenting without affecting the main project. Once the work is finished and tested, it can be merged back into the main project.

In addition to tracking changes, version control systems also provide tools for managing releases and controlling the state of the project at any given point in time. This makes it easier to manage complex projects, coordinate the work of multiple developers, and maintain stability while new features are being developed.

In conclusion, version control is an essential tool in modern software development. It provides a robust framework for tracking changes, coordinating the work of multiple developers, and managing complex projects. Whether you're a solo developer working on a small project or a large team working on a major software product, version control is a must-have tool in your development toolkit.

4.7.1 Git

Git is a distributed version control system that was created by Linus Torvalds, the creator of Linux, in 2005. It's a tool used by software developers to track changes in source code during software development. With Git, developers can work on the same codebase simultaneously without overwriting each other's changes. This is achieved through branching and merging, which allows developers to create separate branches of the codebase, make changes independently, and then merge those changes back into the main branch when they're ready.

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later. It allows you to revert selected files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more. Using a VCS also generally means that if you screw things up or lose files, you can easily recover.

Git is designed to handle everything from small to very large projects with speed and efficiency. It's easy to learn and has a tiny footprint with lightning-fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

In Git, every developer's working copy of the code is also a repository that can contain the full history of all changes. Unlike centralized version control systems, this makes it less vulnerable to server downtime or network latency. If a developer's machine or data gets lost, all they need to do is clone the repository onto their new machine and they have all their work back, with full history.

Git also provides robust support for non-linear development, with powerful features like branches and merges. This makes it possible for multiple developers to work on different features concurrently. Git also has excellent support for branching and merging, which

allows developers to experiment with new features, isolate their work when necessary, and combine their changes with others in a controlled manner.

Furthermore, Git is free and open-source software. It's released under the terms of the GNU General Public License version 2, which means that it's not only free to use, but developers are also free to study, modify, and distribute the software. In conclusion, Git is a powerful and flexible version control system that has become an essential tool for software development. Its distributed architecture, robust support for branching and merging, and its speed and efficiency make it a superior choice for version control. Whether you're a solo developer working on a small project or part of a large team developing a complex software system, Git offers a range of features and benefits that can enhance your software development process. It's a tool that every developer should have in their toolkit.

4.8 Deployment

Deployment in software development is a critical phase that marks the transition of software from a completed product to one that is actively used in a live environment. It encompasses a series of activities that make a software system available for use, including the installation, configuration, testing, and optimization of the application to ensure it operates effectively in the intended environment.

4.8.1 Understanding Deployment

Deployment is not just about transferring files; it's a meticulous process that ensures the software functions correctly in the target environment. This involves setting up servers, configuring databases, installing dependencies, and executing the code in the new setting. The deployment process can be manual, where a developer or an operations team member carries out these tasks, or automated, where scripts and tools automatically handle the deployment.

4.8.2 The Importance of Deployment

The significance of deployment cannot be overstated. It is the final step that delivers the value of the software to the users. A successful deployment means that the software is available for its intended audience and is ready to fulfill the purposes it was designed for. Deployment is also a stage where the software is made secure and compatible with other systems and platforms.

4.8.3 Deployment Strategies

There are various deployment strategies that teams can adopt, each with its advantages and considerations:

4.8.3.1 Blue-Green Deployment

This strategy involves two identical production environments, only one of which is live at any given time. When deploying a new version, it is released to the inactive environment. If the deployment is successful, traffic is switched over. This approach minimizes downtime and risk because the old version remains operational until the new one is ready.

4.8.3.2 Canary Deployment

In this approach, the new version is rolled out to a small subset of users before a full-scale launch. This allows teams to monitor the performance and stability of the release and catch potential issues early.

4.8.3.3 Rolling Deployment

The update is gradually deployed to instances of the application until all are updated. This method reduces downtime but can be complex to manage if there are significant differences between versions.

4.8.3.4 Automated Deployment

Automation is becoming increasingly prevalent in deployment processes. Tools and platforms enable teams to automate the steps involved in deploying applications, reducing the potential for human error and speeding up the process. Continuous Integration/Continuous Deployment (CI/CD) pipelines are an example of automation in deployment, where code changes are automatically built, tested, and deployed to production or staging environments.

4.8.3.5 Monitoring and Rollback

Post-deployment, it's crucial to monitor the application's performance and health. This includes tracking metrics like response times, error rates, and system resource usage. Monitoring helps detect issues that may not have been apparent during testing. If problems arise, having a rollback plan is essential. This plan allows the team to revert to a previous version quickly to minimize impact on users.

4.8.4 Best Practices

Some best practices for deployment include:

Preparation: Ensure all stakeholders are informed about the deployment and understand the new features or changes.

Testing: Always test the software in an environment that closely resembles production before rolling it out to end-users.

Clear Process: Maintain clear communication with the team to ensure everyone understands the deployment process.

Performance Metrics: Establish performance metrics to measure the effectiveness of the deployment and monitor these post-deployment.

4.8.5 Deployment on Github Pages

GitHub Pages is a static site hosting service that takes HTML, CSS, and JavaScript files straight from a repository on GitHub, optionally runs the files through a build process, and publishes a website. It's designed to host your personal, organization, or project pages directly from a GitHub repository.

4.8.5.1 Getting Started with GitHub Pages

To create a GitHub Pages site, you can start with a new or existing repository. If the repository is under GitHub Free or GitHub Free for organizations, it must be public. For GitHub Pro, Team, Enterprise Cloud, or Server, private repositories are also supported.

4.8.5.2 Creating a Repository for GitHub Pages

If you're creating a user or organization site, the repository must be named `<username>.github.io` or `<organization>.github.io`, where username and organization are your GitHub username and organization name, respectively. This naming convention is crucial as it signals to GitHub that the repository should be treated as a special website repository².

4.8.5.3 Configuring a Publishing Source

You can configure your GitHub Pages site to publish when changes are pushed to a specific branch, or you can write a GitHub Actions workflow to publish your site. The source branch can be any branch in your repository, and the source folder can either be the root of the repository or a /docs folder on the source branch³.

4.8.5.4 Types of GitHub Pages Sites

There are three types of GitHub Pages sites: project, user, and organization. Project sites are connected to a specific project hosted on GitHub. User and organization sites are connected to a specific account on GitHub.com. Project sites are stored in the same repository as their project, while user and organization sites must be named after the user or organization's account¹.

4.8.5.5 Features and Benefits of GitHub Pages

Ease of Use: GitHub Pages simplifies the process of setting up a website. You can create a site by simply pushing HTML, CSS, and JavaScript files to your repository.

Custom Domains: You can use your own custom domain for your GitHub Pages site, enhancing your brand's visibility and professionalism.

Continuous Deployment: GitHub Pages integrates with GitHub Actions, allowing for continuous deployment of your site whenever you push new changes to the repository.

Jekyll Integration: GitHub Pages supports Jekyll, a static site generator, enabling you to create blog posts and other dynamic content using Markdown files.

Community and Support: Being part of GitHub, Pages has a vast community and plenty of documentation, making it easier to find help and resources.

4.8.5.6 Limitations and Considerations

While GitHub Pages is an excellent option for hosting static websites, it does have some limitations.

- static Content Only:** GitHub Pages is not suitable for dynamic sites that require server-side processing, such as those using PHP or a database.
- Public by Default:** GitHub Pages sites are publicly available on the internet, even if the repository is private. Sensitive data should be removed before publishing.

4.8.5.6 Deploying to GitHub Pages

Deploying your site to GitHub Pages can be as simple as pushing your code to the `gh-pages` branch of your repository. You can also use GitHub Actions to automate the deployment process, making it easy to build and deploy your site from any branch.

4.8.5.7 GitHub Pages Tutorial

For those new to GitHub Pages, there are many tutorials available that guide you through the process of creating and deploying your site. These tutorials cover everything from setting up your repository to customizing your site's design and content. GitHub Pages is a powerful and user-friendly platform for hosting static

4.9 Model Architecture:

The Architecture and proposed flowchart for building the Web Application is shown in form of flowchart diagram below.

This Basic circuit Diagram shows the flow data and exact execution of Web Application:

The basic blocks show the modules and data used and arrow reflect the flow of process →

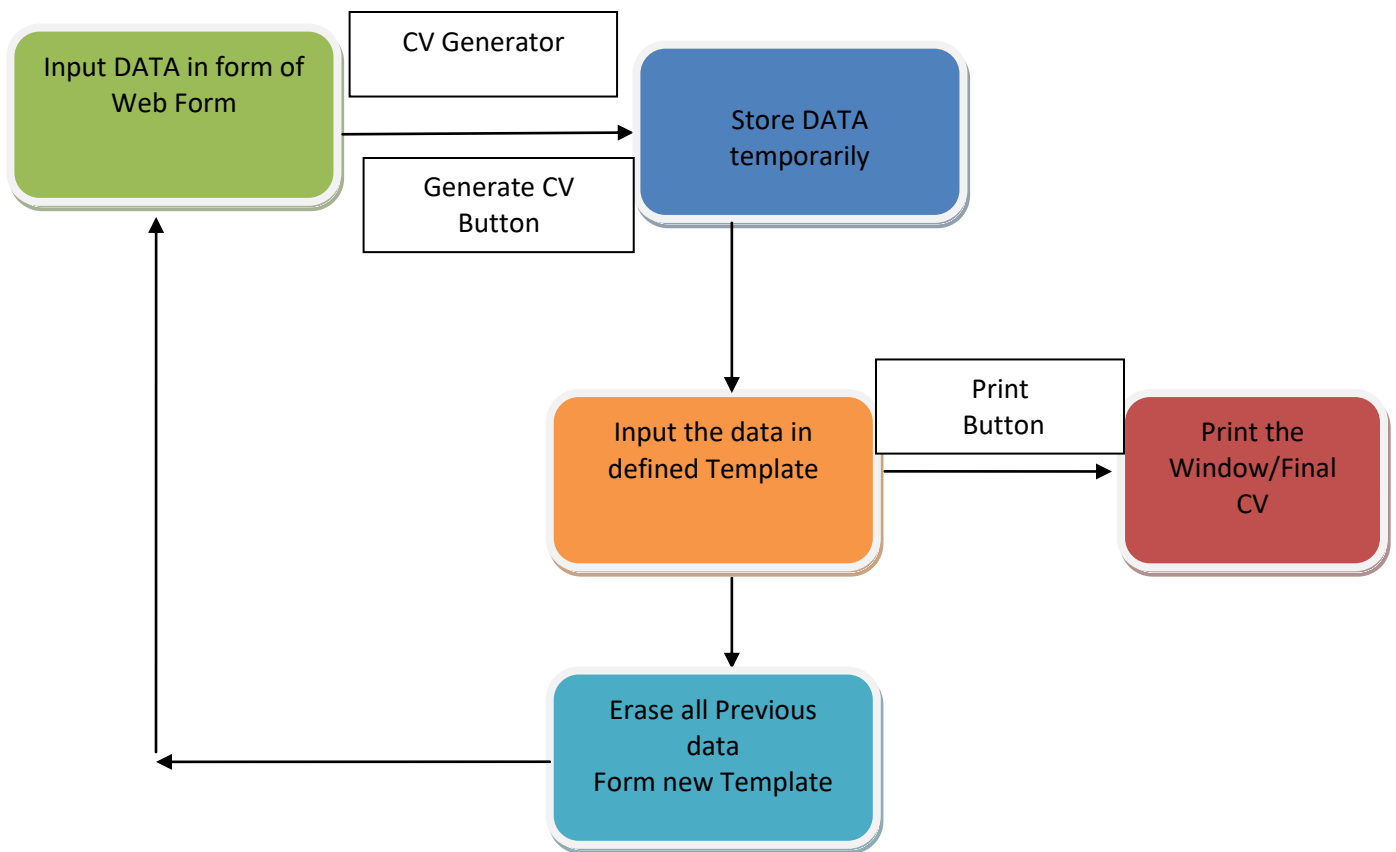


Figure 4.9.1 Flow Diagram for CV Generator

4.10 CV v/s Resume

When applying for jobs, you might come across the terms "resume" and "CV" (Curriculum Vitae). Although they are often used interchangeably, they serve different purposes and have distinct formats. Understanding the differences between a resume and a CV is crucial for job seekers to present their qualifications effectively.

4.10.1 What is CV?

CV is a comprehensive document that provides an in-depth overview of your academic and professional history. It includes detailed information about your education, work experience, research, publications, presentations, awards, and other achievements.

The CV is often used in academic, research, medical, and scientific fields.

CVs can vary in length, ranging from two pages to several pages, depending on your experience and accomplishments.

CVs are commonly used in Europe, the Middle East, Africa, and Asia for job applications, as well as in academic and research settings worldwide.

4.10.2 What is Resume?

A resume is a concise document, typically one to two pages long, that summarizes your education, work experience, skills, and achievements.

Its primary purpose is to provide potential employers with a quick overview of your qualifications tailored to a specific job or industry.

Resumes are commonly used in the United States and Canada for most job applications.

4.10.3 Key Differences

- **Length:**
 - **Resume:** Typically one to two pages, focusing on brevity and relevance to the job applied for.
 - **CV:** Length can vary, often running several pages to comprehensively cover academic and professional achievements.
- **Content:**
 - **Resume:** Includes sections such as contact information, professional summary or objective, work experience, education, skills, and sometimes additional sections like certifications, volunteer work, or interests.
 - **CV:** Includes detailed sections such as contact information, personal statement, detailed education history, work experience, research experience, publications, presentations, awards, honors, professional affiliations, and other relevant activities.
- **Focus:**
 - **Resume:** Tailored to the specific job application, emphasizing skills and experiences relevant to the position.
 - **CV:** Provides a thorough and detailed account of one's entire academic and professional history, often not tailored to a specific position.
- **Usage:**
 - **Resume:** Used for most job applications outside of academia, particularly in North America.
 - **CV:** Used for academic, research, and medical positions, and in regions where CVs are the norm for job applications.

4.10.4 Structural Differences

Resume Structure:

- Contact Information: Name, phone number, email address, LinkedIn profile (optional).
- Professional Summary or Objective: A brief statement highlighting your career goals and key qualifications.
- Work Experience: List of jobs held, with bullet points outlining key responsibilities and achievements. Typically, this section is in reverse chronological order.
- Education: Degrees obtained, institutions attended, and graduation dates.
- Skills: A list of relevant skills, both technical and soft skills.
- Additional Sections (Optional): Certifications, volunteer work, languages, projects, interests, etc.

CV Structure:

- Contact Information: Name, phone number, email address, LinkedIn profile (optional).
- Personal Statement: A detailed overview of your academic interests, career goals, and professional aspirations.
- Education: Detailed information on all degrees obtained, institutions attended, graduation dates, thesis titles, and relevant coursework.
- Work Experience: Comprehensive list of all relevant professional positions held, with detailed descriptions of responsibilities and accomplishments.
- Research Experience: Detailed account of research projects undertaken, methodologies used, and results obtained.
- Publications: A comprehensive list of all published works, including articles, papers, books, and other publications.
- Presentations: A list of conferences, seminars, and other events where you have presented your work.
- Awards and Honors: Detailed account of all academic and professional awards and recognitions.
- Professional Affiliations: Membership in professional organizations and associations.

- Additional Sections: Grants, fellowships, certifications, skills, languages, and any other relevant information.

The Resumes are concise and targeted, suitable for most job applications, focusing on relevant skills and experiences. CVs, on the other hand, are detailed and comprehensive, ideal for academic, research, and scientific positions, providing an in-depth look at your entire career. Understanding these differences ensures that you present your qualifications in the best possible light, depending on the job or position you are applying for.

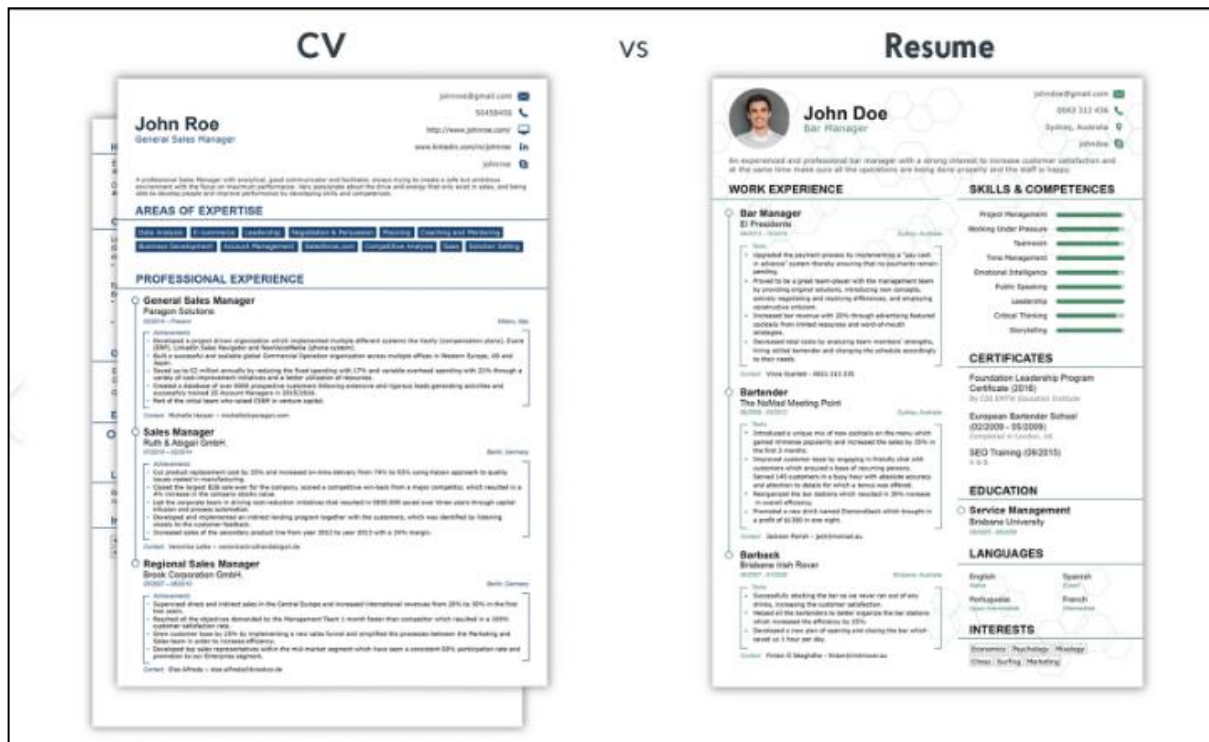


Figure 4.10.1 Visual Difference between CV and Resume

4.10.5 Advantages of CV over Resume:

A CV (Curriculum Vitae) provides a comprehensive and detailed overview of an individual's academic and professional history, making it ideal for positions in academia, research, and fields requiring extensive experience. This detailed document showcases a candidate's full breadth of qualifications and accomplishments, which can be advantageous in fields that value a thorough understanding of one's background.

Compared to a resume, a CV allows for the inclusion of detailed sections such as publications, presentations, research experience, and professional affiliations. This level of detail can highlight a candidate's expertise and contributions to their field, providing a richer picture of their qualifications and achievements.

Additionally, a CV's flexibility in length means that candidates can provide a complete record of their career, without the constraints of brevity that a resume imposes. This can be particularly beneficial for seasoned professionals with extensive experience, ensuring that all relevant information is included and nothing important is omitted.

4.11 IMPLEMENTATION AND WORKING

4.11.1 Implementation:

Building a web-based CV builder involves implementing several key components that work together to provide a seamless user experience for creating, customizing, and generating professional CVs. The development process begins with the front-end interface, which is designed using HTML for structure, CSS for styling, and JavaScript for interactivity. HTML forms the backbone of the application, creating the structure for the input fields where users can enter their personal information, education, work experience, skills, and other relevant details. CSS, along with frameworks like Bootstrap, ensures the interface is visually appealing and responsive across different devices, enhancing usability.

JavaScript plays a crucial role in adding dynamic functionality to the application. It manages user inputs, handles form submissions, and updates the CV preview in real-time as users fill out their details. React.js, a popular JavaScript library, is often used to create reusable components for the form fields and CV preview, facilitating a modular and maintainable codebase, ensuring the application remains responsive to user interactions.

While a database is not required for this implementation, data can be temporarily stored in-memory or using file-based storage to facilitate the CV generation process. For CV generation we would be directly printing the designed CV using the print function of JS to print the final window form of CV image on the Screen window, which are further integrated to convert HTML content into PDF files. This allows users to download their completed CVs in a professional format.

This comprehensive approach to building a web-based CV builder results in a robust, user-friendly application that enables users to effortlessly create and customize their CVs.

4.11.2. Working:

Working Theory of a Web-Based CV Builder

1. User Interface (UI) Design

- **Form Inputs:** Provides fields for user to input personal, educational, and professional information.
- **Template Selection:** Allows users to choose from various CV templates.

2. Client-Side Development

- **HTML/CSS:** Structures the form and styles the UI.
- **JavaScript (JS):** Manages form interactions, data validation, and dynamic content updates.

3. State Management

- **Data Storage:** Uses JavaScript objects to store user input data temporarily.
- **Live Preview:** Updates the CV preview in real-time as users fill out the form.

4. Form Handling

- **Input Validation:** Validates user inputs for correct data formats (e.g., email, phone number).
- **Data Binding:** Binds form inputs to state variables to reflect changes instantly.

6. PDF Generation

- **HTML to PDF Conversion:** Uses libraries like `pdf-lib` or `html2pdf.js` to convert HTML templates to PDF.
- **Dynamic Data Insertion:** Inserts user-provided data into predefined CV templates before PDF conversion.

By following these steps, a web-based CV builder can effectively collect user data, dynamically generate CVs, and provide a user-friendly experience, ensuring that users can create professional CVs efficiently.

4.12 Features

Features of Web Application Based CV Generator include:

1. User Registration and Login:

- Secure user authentication and authorization.
- Social media login options (e.g., Google, LinkedIn).

2. Profile Management:

- User profile customization and management.
- Option to upload a profile picture.

3. Template Selection:

- Multiple professionally designed CV templates.
- Option to preview templates before selection.

4. Easy-to-Use Interface:

- Intuitive and user-friendly design.
- Drag-and-drop functionality for rearranging sections.

5. Form Input Fields:

- Sections for personal information, education, work experience, skills, and more.
- Customizable sections to add additional information (e.g., certifications, awards).

6. Real-Time Preview:

- Live preview of the CV as data is entered.
- Instant updates reflecting changes made by the user.

7. Mobile Compatibility:

- Responsive design for mobile and tablet use.
- Mobile app version for on-the-go CV creation.

8. Formatting Features:

- Customizable number of fields.
- Ability to adjust the number of inputs for few Fields.

9. Multiple CV Versions:

- Option to create and manage multiple CV versions.
- Easy duplication of existing CVs for modification.

10. Export and Download:

- Download CVs in various formats (PDF, DOCX).
- Print-ready CV versions.

These features collectively contribute to the development of effective and efficient object detection systems using Python-based image processing and computer vision techniques.

Chapter 5 METHODOLOGY AND IMPLEMENTATION:

5.1 Technologies Used

5.1.1 HTML5:

HTML5, the fifth major revision of the HTML standard, has revolutionized the way we create and interact with web content. It's a markup language used for structuring and presenting content on the World Wide Web, and it's the cornerstone of modern web development. With its introduction, HTML5 brought a plethora of new features and capabilities that have significantly enhanced the user experience and developer productivity. Among its most notable features are semantic elements that provide more contextual information about the content of a web page, making it easier for search engines and assistive technologies to understand the structure and meaning of web content¹. HTML5 also introduced native support for multimedia content, including audio and video playback, without the need for third-party plugins or software, which was a game-changer for web-based multimedia applications². The <canvas> element, another powerful addition, allows for dynamic, scriptable rendering of 2D shapes and bitmap images, enabling developers to produce graphics and animations directly in the browser³. Improved form re interactive and user-friendly³. Web Storage, one of the key HTML5 features, provides two new objects for storing data on the client's computer - localStorage and sessionStorage - which are mhandling is another area where HTML5 shines, offering new input types and attributes that make forms moore secure and can store larger amounts of data compared to cookies. HTML5 is also a candidate for cross-platform mobile applications because it includes features designed with low-powered devices in mind². Many new syntactic features are included to natively handle multimedia and graphical content, expandable sections are implemented natively rather than depending on CSS or JavaScript, and the language has been designed to be both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc., without XHTML's rigidity, remaining backward-compatible with older software². HTML5 is intended to subsume not only HTML 4 but also XHTML¹ and even the DOM Level 2 HTML itself, including detailed processing models to encourage more interoperable implementations; it extends, improves, and rationalizes the markup

available for documents and introduces markup and application programming interfaces (APIs) for complex web applications.

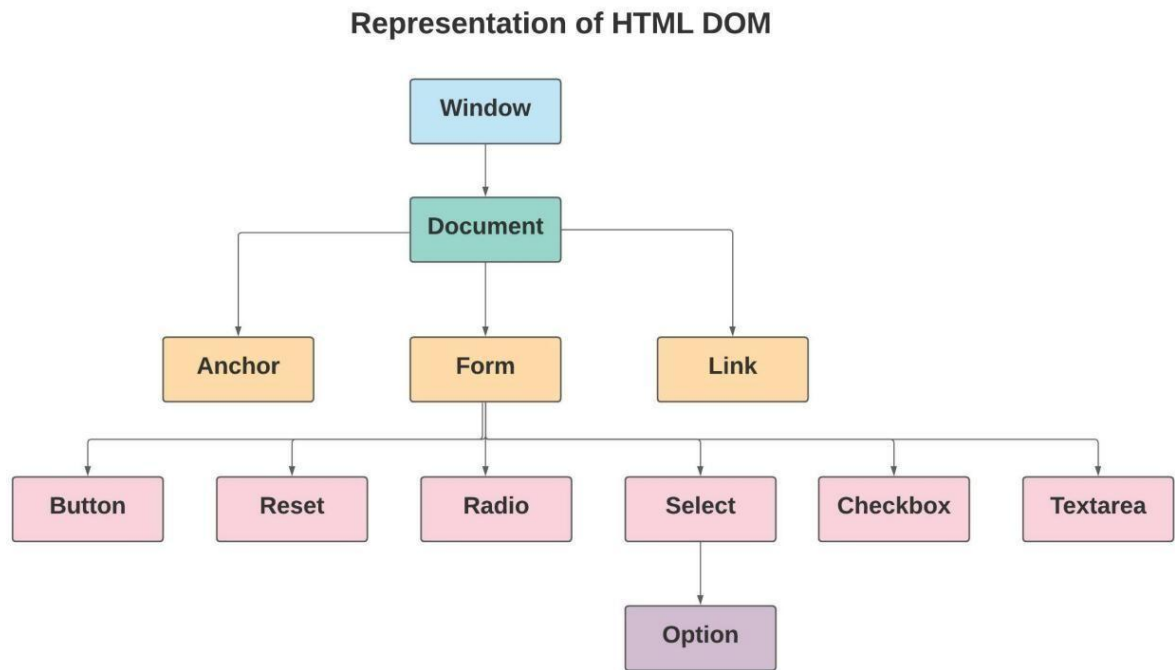


Fig 5.1 HTML Tree

5.1.2 CSS3

To style the captivating world of “CV generator,” we’ve utilized CSS3, the latest standard for cascading style sheets. CSS3 offers advanced styling capabilities with minimal effort, such as rounded corners, shadows, gradients, transitions, and animations, which are essential for creating a visually appealing interface. With media queries, we’ve ensured that “CV generator” is responsive and provides an optimal viewing experience across a range of devices, from desktops to mobile phones.

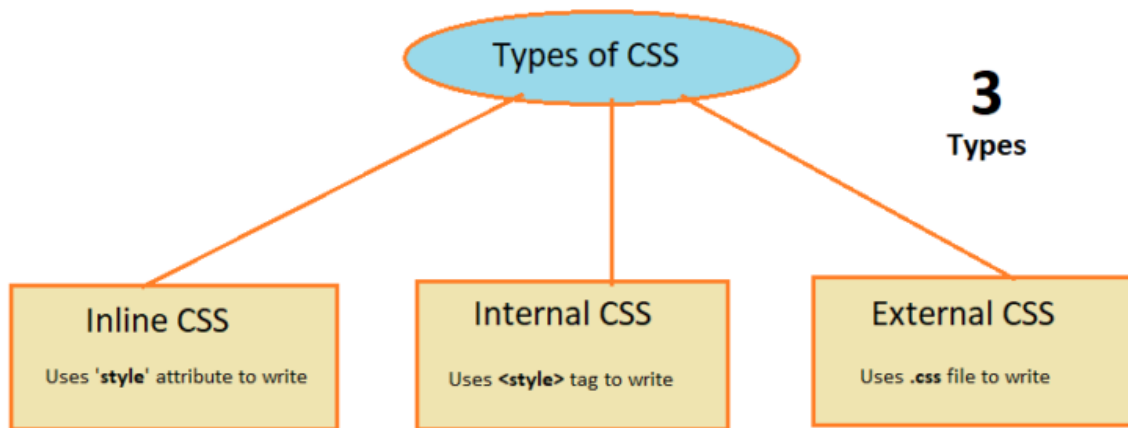


Fig 5.2 CSS types

CSS, or Cascading Style Sheets, is the language for describing the presentation of Web pages, including colors, layout, and fonts, allowing one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation. CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. Each rule or rule-set consists of one or more selectors and a declaration block. In CSS, selectors declare which part of the markup a style applies to by matching tags and attributes in the markup itself. Elements in the document tree are matched against selectors when the user agent (browser) applies the style sheet to the document tree. Selectors may apply to all elements of a specific type, or only those elements that match a

certain attribute; elements can be matched depending on how they are placed relative to each other in the document tree. A CSS rule consists of a selector and a declaration block. The selector points to the HTML element to style, and the declaration block contains one or more declarations separated by semicolons. Each declaration includes a CSS property name and a value, separated by a colon. A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces. CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting. Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device. The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents. In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL. CSS level 1, consisting of the CSS1 specification, was completed and recommended in December 1996. CSS level 2, CSS2, started as a recommendation in May 1998 and was revised in April 2008. CSS level 3, which was started in 1998, is still under development as of 2014. CSS level 4 is also under development. The W3C maintains a list of current CSS specifications. CSS is applied to HTML documents by the browser (the user agent) that renders the document. When an HTML document is loaded into a web browser, it becomes a Document Object Model (DOM), a tree-like structure of nodes. Each node is an object that represents part of the document; most nodes are elements. A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors and a declaration block.

5.1.3 JavaScript

JavaScript breathes life into “CV generator,” turning static pages into a dynamic and interactive experience. It’s a multi-paradigm language that supports event-driven, functional, and imperative programming styles, enabling us to implement complex features like the “Guess the Pokémon” game and real-time updates in the Template. JavaScript’s versatility allows us to handle both client-side and server-side scripting, making “CV generator” an engaging platform for trainers worldwide.

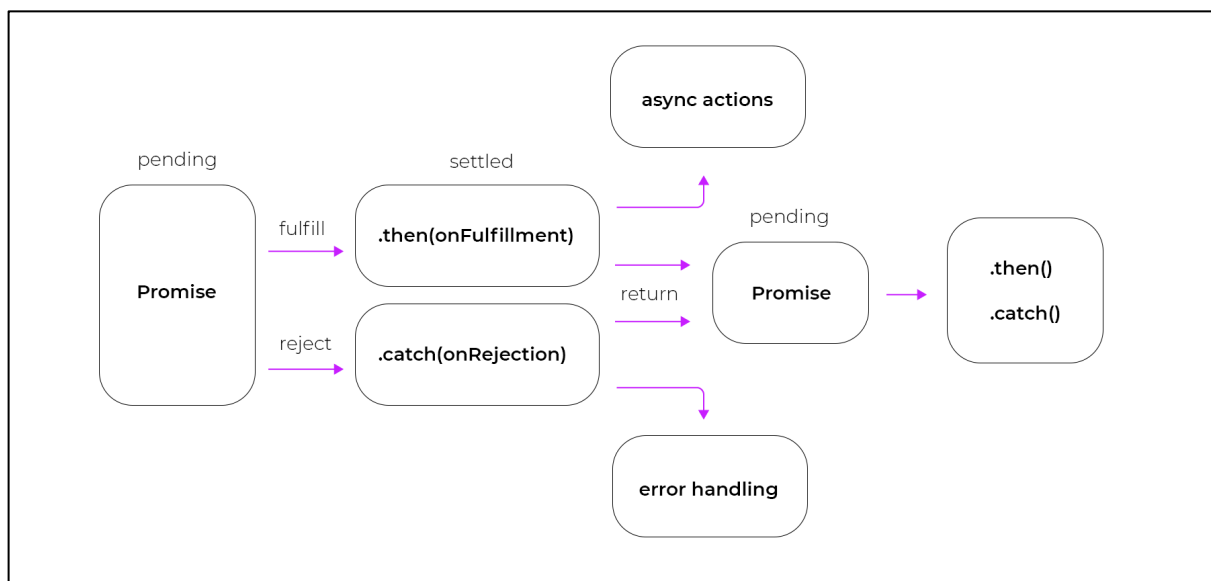


Fig 5.3 Async-await in javascript

JavaScript, often abbreviated as JS, is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has become one of the core technologies of the World Wide Web, alongside HTML and CSS, and is supported by all modern web browsers without the need for plugins. Its capabilities allow developers to build dynamic and interactive web applications, making it an essential part of web development. JavaScript enables client-side script to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles, and it has APIs for working with text, arrays, dates, regular expressions,

and the DOM. The language also provides a platform for game development, desktop and mobile applications, and server-side network programming with environments such as Node.js. JavaScript's versatility extends to the creation of complex features on web pages, including interactive forms, image sliders, and content that updates dynamically. It is also used extensively in creating rich user interfaces and web applications with AJAX. With the advent of HTML5, JavaScript has seen a surge in its usage for multimedia applications, including audio and video manipulation, as well as for graphical and animation purposes through the canvas element. JavaScript's functionality is not limited to web pages—it's also used in server-side network programming with Node.js, game development, and the creation of desktop and mobile applications. The language's use in applications outside of web pages—for example, PDF documents, site-specific browsers, and desktop widgets—is also significant. New APIs provide powerful capabilities to web applications, such as allowing them to store data on the client-side locally and operate offline more efficiently, access file systems, use hardware devices, and communicate with other applications. JavaScript engines have been developed to perform just-in-time compilation, and JavaScript has become a concurrent language with the addition of Web Workers. Despite its name, JavaScript is unrelated to the Java programming language and has a more C-like syntax. JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation, object introspection, and source-code recovery. JavaScript engines, such as Google's V8, Mozilla's SpiderMonkey, and Microsoft's Chakra, are embedded in web browsers, and these engines execute JavaScript code on users' devices

5.1.4 Git

Version control is vital for “CV generator” development, and Git is the tool we’ve chosen for this task. It’s a distributed version control system that tracks changes, allows for collaborative work, and ensures that our codebase remains organized and accessible. With Git, we can manage our project’s development in a decentralized manner, enabling multiple developers to work on different features simultaneously without conflict. Git is a distributed version control system that’s integral to the collaborative process of software development, allowing multiple developers to work on a project simultaneously without overwriting each other’s contributions. Its distributed nature means that every developer has a full-fledged copy of the project repository, including its history, which ensures work can continue even when not connected to a central server. Git’s efficiency is unmatched, handling everything from small to very large projects with speed, thanks to features like cheap local branching, convenient staging areas, and multiple workflows. It supports non-linear development through its powerful branching and merging capabilities, enabling developers to create branches for new features, experiments, or bug fixes, and merge them back into the main codebase when ready. Git’s data integrity and authentication mechanisms ensure that the history of changes remains secure and verifiable, protecting the code from corruption and unauthorized alterations. The platform’s ability to manage and track changes to files, revert to previous versions, and compare changes over time makes it an essential tool for maintaining the history and integrity of a project. Git’s functionality is accessible through a command-line interface, though many graphical user interfaces and integrated development environments also incorporate Git support, making it accessible to a broader range of users. With its open-source nature, Git has a vast community contributing to its continuous improvement, providing a robust ecosystem of tools and extensions that enhance its core capabilities. This community-driven approach has made Git the standard for version control in the software industry, embraced by individual developers and large organizations alike.

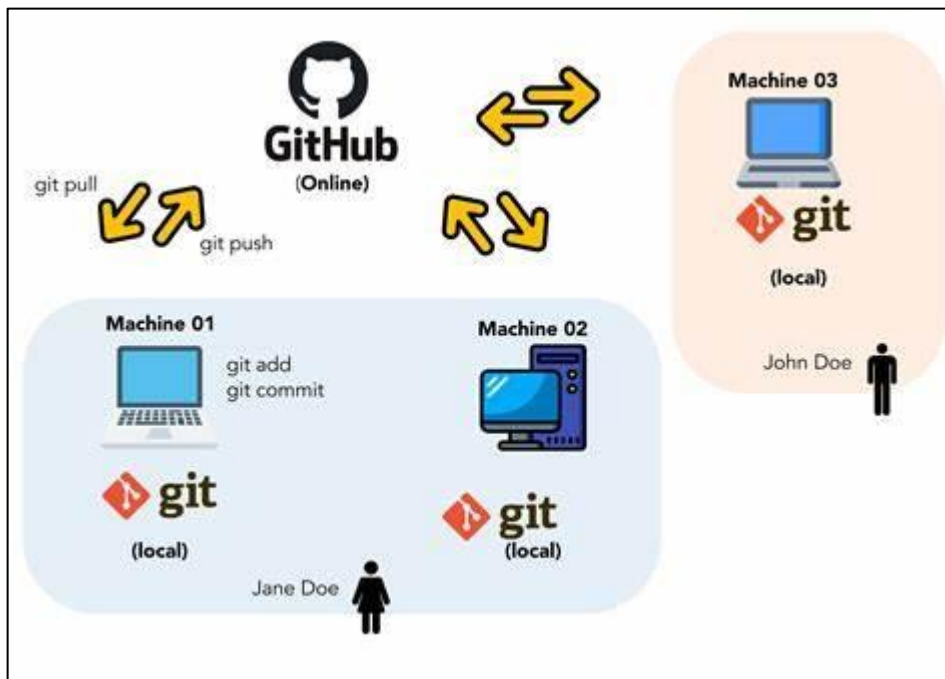


Fig 5.4 Git and Github Flow

5.1.5 GitHub:

GitHub acts as our repository hosting service, leveraging Git’s version control and adding its own features. It provides a web-based graphical interface and access control, along with several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. GitHub has been instrumental in “CV generator” development, allowing us to manage our code, track issues, and collaborate with contributors from around the globe.

5.2.6 Visual Studio Code (VSCode): Our development environment of choice for “CV generator” is VSCode, a powerful and lightweight code editor that’s free and open-source. It offers features like IntelliSense for smart completions, built-in Git commands, debugging tools, and an extensive marketplace for extensions. VSCode’s flexibility and customization options have made it an indispensable tool for writing, testing, and debugging our game’s code. Visual Studio Code (VS Code) is a lightweight yet powerful source code editor that runs on Windows, macOS, and Linux. It combines the simplicity

of a code editor with robust developer tooling, making it an ideal choice for both beginners and seasoned developers. Here's why VS Code stands out.



Fig 5.5 VS Code logo

Fast and Efficient: VS Code's lightning-fast source code editor ensures a frictionless edit-build-debug cycle. It gets out of your way, allowing you to focus on your ideas rather than environment setup.

Language Support: With built-in support for hundreds of languages, VS Code provides syntax highlighting, bracket matching, auto-indentation, and snippets. It adapts to your coding needs seamlessly.

IntelliSense and Code Navigation: IntelliSense offers smart code completion, rich semantic understanding, and navigation. You'll find it easier to explore and understand your codebase.

Debugging Made Easy: VS Code includes an interactive debugger. You can step through source code, inspect variables, view call stacks, and execute commands in the console—all within the editor.

Customizable and Extensible: Make VS Code your own by customizing every feature. Install third-party extensions to enhance functionality. The vibrant community on GitHub contributes to its growth.

Web Development: VS Code excels in web development. It supports JavaScript, TypeScript, Node.js, JSX/React, HTML, CSS, SCSS, Less, and JSON out of the box.

Architecture: VS Code combines web, native, and language-specific technologies. It uses Electron, which blends JavaScript, Node.js, and native app speed. The same editor powers projects like “Monaco” and Internet Explorer’s F12 Tools.

Git Integration: Manage source control without leaving the editor. View pending changes, diffs, and commit directly from VS Code.

Keyboard Shortcuts: Boost productivity with customizable keyboard shortcuts. Modify defaults or match shortcuts from other editors.

Open Source and Community-Driven: Contribute to the growing community on GitHub. VS Code’s extensible architecture invites developers to optimize their unique workflows.

A. Building a simple HTML form that will take input from user:

```
<body>
  <!-- Starting HTML body -->
  <div class="container" id="cv-form">

    <h1 class="text-center my-2">CV GENERATOR</h1>
    <p class="text-center ">Build your own CV with Your own Ideas</p>

    <div class="row-md-6">
      <div class="col-md-6">
        <!-- first col -->
        <h3>Personal Details</h3>

        <div class="form-group">
          <label for="nameFiled">Your Name</label>
          <input type="text" id="nameField" placeholder="Enter Your Name" class="form-control">
        </div>

        <div class="form-group mt-2">
          <label for="contactFiled">Your Contact NO.</label>
          <input type="text" id="contactField" placeholder="Enter Your Contact Nmber" class="form-control">
        </div>

        <div class="form-group mt-2">
          <label for="addressFiled">Your Address</label>
          <textarea id="addressField" rows="4" placeholder="Enter Your Address"
            class="form-control"></textarea>
        </div>

        <div class="form-group mt-3">
          <label for="">Select your photo:</label>
          <input id="imgField" type="file" class="form-control" />
        </div>

        <p class="text-secondary my-3">Important Links</p>

        <div class="form-group mt-2">
          <label for="ldFiled">Your LinkedIn</label>
          <input type="text" id="ldField" placeholder="Enter Your LinkedIn id" class="form-control">
        </div>

        <div class="form-group mt-2">
          <label for="ghFiled">Your GitHub Link</label>
          <input type="text" id="ghField" placeholder="Enter Your Github LIInk" class="form-control">
        </div>
      </div>
    </div>
  </div>
</body>
```

```

</div>
<div class="col-md-6">
  <!-- second col -->
  <h3>Professional details</h3>

  <div class="form-group mt-2">
    <label for="sumFiled">Summary</label>
    <textarea rows="5" id="sumField" placeholder="Give short summary of yours"
      class="form-control"></textarea>
  </div>

  <div class="form-group mt-2" id="we">
    <label for="">Work Experience</label>
    <textarea rows="4" placeholder="Give your Previous Experience"
      class="form-control weField"></textarea>
    <div class="container text-center mt-2" id="weAddButton">
      <button onclick="addNewWEField()" class="btn btn-primary btn-sm">Add</button>
    </div>
  </div>

  <div class="form-group mt-2" id="aq">
    <label for="">Academic Qualification</label>
    <textarea rows="4" placeholder="Give your Previous Experience"
      class="form-control eqField"></textarea>
    <div class="container text-center mt-2" id="aqAddButton">
      <button onclick="addNewAQField()" class="btn btn-primary btn-sm">Add</button>
    </div>
  </div>

</div>

</div>
<div class="container text-center mt-3">
  <button onclick="generateCV()" class="btn btn-primary btn-lg">Generate CV</button>
</div>
</div>

```

Figure 5.6 HTML form for input

OUTPUT:

CV GENERATOR

Build your own CV with Your own Ideas

Personal Details

Your Name

Your Contact NO.

Your Address

Select your photo:

 No file chosen

Important Links

Your LinkedIn

Your GitHub Link

Professional details

Summary

Work Experience

Academic Qualification

B.HTML scripting for CV template:

```
<!-- print Template -->
<div class="container" id="cv-template">
  <div class="row">
    <div class="col-md-4 text-center background">
      <!-- first col -->
      
      <div class="container">
        <p id="nameT1">Default Name</p>
        <p id="contactT">Default Contact detail</p>
        <p id="addressT">default Links</p>
        <hr>
        <hr>
        <p><a id="ldT" href="#1">www.linkedin.com</a></p><br>
        <p><a id="ghT" href="#1">www.github.com</a></p>
      </div>
    </div>
    <div class="col-md-8 py-5">
      <!-- second col -->
      <h1 id="nameT2" class="text-center" style="font-weight: 900;">Default Name</h1>

      <!-- summary -->
      <div class="card nt-4">
        <div class="card-header background">
          <h3>Summary</h3>
        </div>
        <div class="card-body"> </div>
        <p id="summaryT">Lorem ipsum, dolor sit amet consectetur adipisicing elit. Perspiciatis id quaerat
          quas maxime
          magni, laborum inventore natus blanditiis earum deleniti? Modi mollitia et numquam pariatur.
        </p>
      </div>

      <!-- Experience -->
      <div class="card nt-4">
        <div class="card-header background">
          <h3>Experience</h3>
        </div>
        <div class="card-body">
          <ul id="weT">
            <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
            <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
            <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
          </ul>
        </div>
      </div>
    </div>
  </div>
</div>
```

```


<!-- academic Qulaification -->
<div class="card nt-4">
  <div class="card-header background">
    <h3>Academics</h3>
  </div>
  <div class="card-body">
    <ul id="aqT">
      <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
      <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
      <li>Lorem ipsum dolor sit amet consectetur adipisicing.</li>
    </ul>
  </div>
</div>
</div>

<div class="container mt-3 text-center">
  <button onclick="printCV()" class="btn background">
    Print CV
  </button>
</div>
</div>
</div>
<!-- Linking Bootstrap JS -->

```

Figure 5.7 HTML for CV template

OUTPUT:



Default Name

Default Contact detail

default Links

[www.linkedin.com](#)

[www.github.com](#)

Default Name

Summary

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Perspiciatis id quaerat quas maxime magni, laborum inventore natus blanditiis earum deleniti? Modi mollitia et numquam pariatur.

Experience

- Lorem ipsum dolor sit amet consectetur adipisicing.
- Lorem ipsum dolor sit amet consectetur adipisicing.
- Lorem ipsum dolor sit amet consectetur adipisicing.

Academics

- Lorem ipsum dolor sit amet consectetur adipisicing.
- Lorem ipsum dolor sit amet consectetur adipisicing.
- Lorem ipsum dolor sit amet consectetur adipisicing.

Print CV

C. Java Script code:

```
// ALL JS function in here
function addNewWEField(){
    // console.log("Adding new field");

    let newNode=document.createElement("textarea");
    newNode.classList.add("form-control");
    newNode.classList.add("weField");
    newNode.classList.add("mt-2");

    newNode.setAttribute("rows",3);
    newNode.setAttribute("placeholder","Enter Here");

    let weOb = document.getElementById("we");
    let weAddButtonOb = document.getElementById("weAddButton");

    weOb.insertBefore(newNode, weAddButtonOb);
}

function addNewAQField(){

    let newNode=document.createElement("textarea");
    newNode.classList.add("form-control");
    newNode.classList.add("eqField");
    newNode.classList.add("mt-2");

    newNode.setAttribute("rows",3);
    newNode.setAttribute("placeholder","Enter Here");

    let aqOb = document.getElementById("aq");
    let aqAddButtonOb = document.getElementById("aqAddButton");

    aqOb.insertBefore(newNode, aqAddButtonOb);
}
```

```

// generate CV

function generateCV(){
  // console.log("going");
  let nameField=document.getElementById("nameField").value;

  let nameT1 = document.getElementById("nameT1");

  nameT1.innerHTML=nameField;

  document.getElementById("nameT2").innerHTML=nameField;

  document.getElementById("contactT").innerHTML=document.getElementById("contactField").value;

  document.getElementById("addressT").innerHTML=document.getElementById("addressField").value;
  document.getElementById("ldT").innerHTML=document.getElementById("ldField").value;
  document.getElementById("ghT").innerHTML=document.getElementById("ghField").value;

  document.getElementById("summaryT").innerHTML=document.getElementById("sumField").value;

  // Work experienxe
  let wes=document.getElementsByClassName('weField');
  let str="";
  for (let e of wes){
    str=str+`<li> ${e.value} </li>`;
  }

  document.getElementById("weT").innerHTML = str;

  // academic qualifiaction
  let aqs=document.getElementsByClassName('eqField');
  let str1="";
  for(let e of aqs){
    str1+=`<li> ${e.value} </li>`;
  }

  document.getElementById("aqT").innerHTML=str1;

```

```

// code for image

let file = document.getElementById('imgField').files(0);
console.log(file);
let reader = new FileReader();
reader.readAsDataURL(file);
console.log(reader.result);

// set image
reader.onloadend=function(){

    document.getElementById('imgT').src=reader.result;

}

document.getElementById("cv-form").style.display="none";
document.getElementById("cv-template").style.display="block";


}

// print CV
function printCV(){
    window.print();
}

```

Figure 5.8 JavaScript code for functioning

Chapter 6 Result and Output:



Ankit kumar Thakur
+919865352556
Dehradun, Uttarakhand

[www.linkedinin/akthakur](https://www.linkedin.com/in/akthakur)

[www.githubb/akthakur](https://www.github.com/akthakur)

Ankit kumar Thakur

Summary

A BCA graduate, eager to learn and keen for challanges.

Experience

- Worked with Accenture as Application Developer[2024-2026]
- Worked with TCS interviewing board[2023]

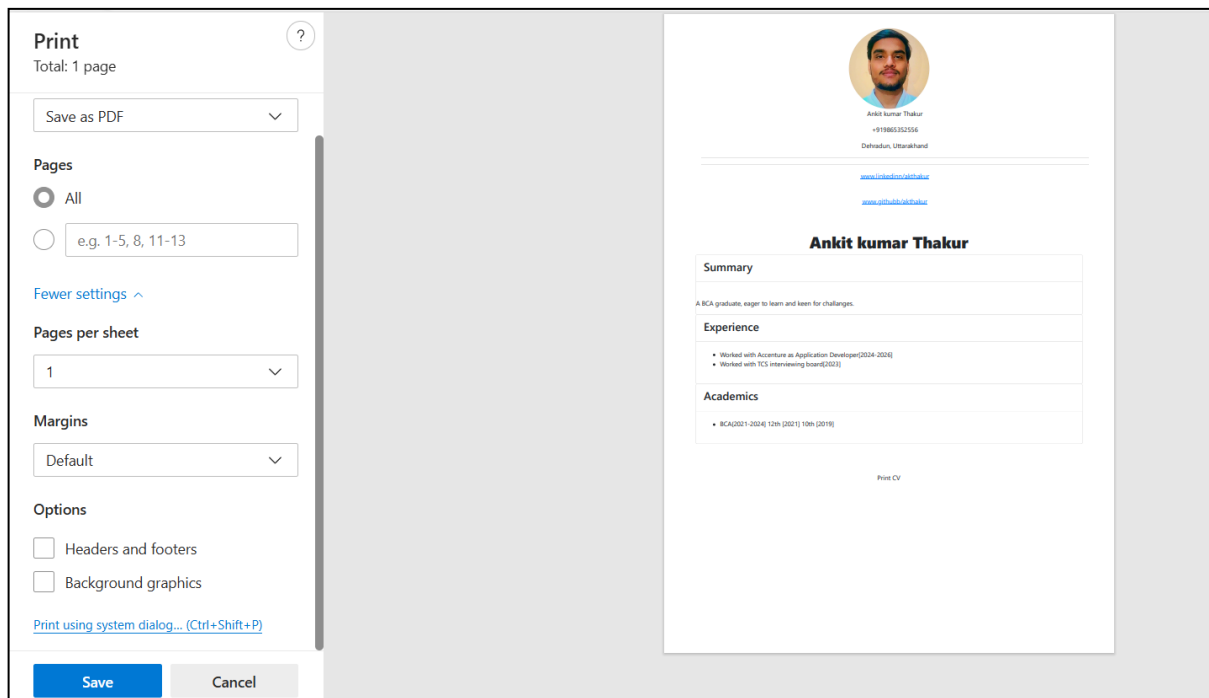
Academics

- BCA{2021-2024} 12th [2021] 10th [2019]

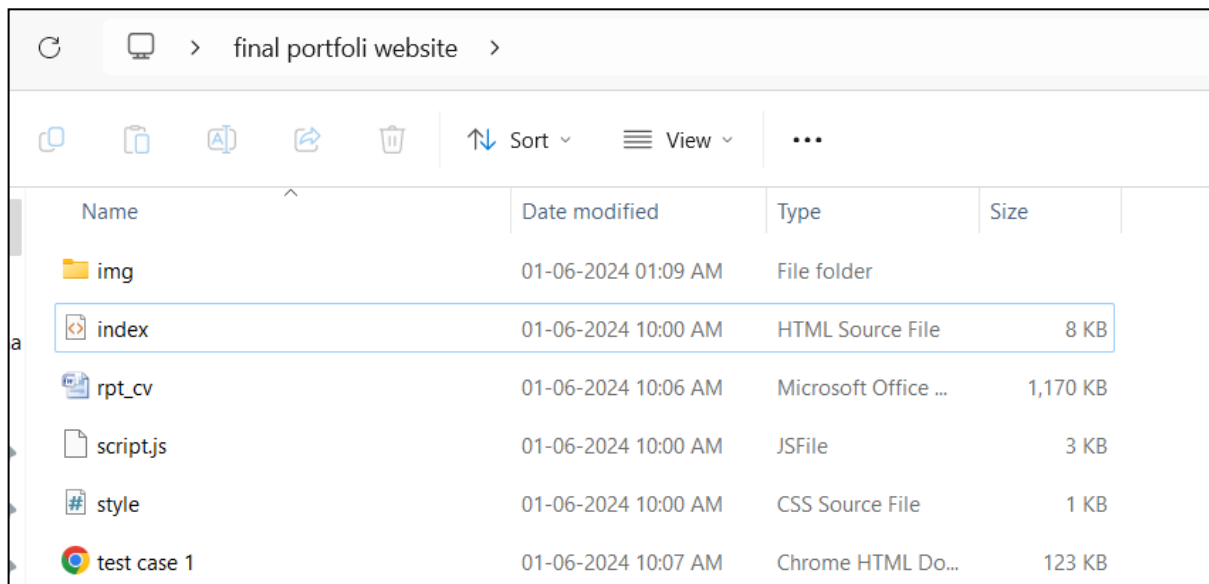
[Print CV](#)

Figure 6 Test case 1

After pressing PrintCV button the following CV is printed/ saved in pdf format



CV is saved in same folder with source code in pdf format.



test case 1.pdf

×


+

620portfoli%20website/test%20case%201.pdf

YouTube Maps Downloads Deloitte Consulting...

Ask Copilot

1 of 1



Ankit kumar Thakur

+919865352556

Dehradun, Uttarakhand

[www.linkedin/akthakur](https://www.linkedin.com/in/akthakur)

[www.githubb/akthakur](https://www.github.com/akthakur)

Ankit kumar Thakur

Summary

A BCA graduate, eager to learn and keen for challenges.

Experience

- Worked with Accenture as Application Developer[2024-2026]
- Worked with TCS interviewing board[2023]

Academics

- BCA[2021-2024] 12th [2021] 10th [2019]

Chapter 7 Conclusion:

In conclusion, a web-based CV generator offers a powerful and user-friendly solution for individuals seeking to create professional and polished CVs with ease. By leveraging modern web technologies, such as React.js for the front-end and Node.js for the back-end, this application streamlines the process of CV creation, allowing users to focus on content rather than formatting. The inclusion of features like multiple template selections, real-time previews, and secure user authentication enhances the user experience, making the tool accessible and efficient for a wide audience.

The benefits of using a web-based CV generator are manifold. Users can quickly input their information, customize their CVs to suit specific job applications, and export them in various formats. This efficiency not only saves time but also ensures that the CVs produced are of high quality and tailored to meet the requirements of potential employers. Furthermore, the application's ability to store and manage multiple CV versions adds a layer of convenience for users applying to various roles.

Overall, the web-based CV generator stands as a testament to the capabilities of web technologies in simplifying complex tasks. By providing an intuitive interface, robust features, and secure handling of personal data, this application addresses the needs of modern job seekers. As technology continues to evolve, further enhancements can be integrated, such as advanced analytics and AI-driven content suggestions, to continually improve the user experience and effectiveness of CV generation.

Chapter 8 Future Work Plan

The report of project work allocated by the supervisor is as follows:

Sl. No.	Work Description	Duration
1.	Advancing the CV generator to work for multiple Templates and Provide advance suggestions for Building both CV and Resume	3 weeks

Chapter 9 Weekly Tasks

The report of project work allocated by the supervisor is as follows:

Week No.	Date: From- To	Work allocated	Work Completed (Yes/No)	Remarks	Guide Signature
1.	09/02/24 16/02/24	Information collection of previously Models	Yes		
2.	19/02/24 06/03/24	Understanding and learning about HTML, CSS, JS along with Bootstrap other modules	Yes		
3.	18/03/24 02/04/24	Building and Defining the builder to work accordingly	Yes		
4.	06/05/24 17/05/24	Final Linking and testing of Generator	Yes		

Bibliography & References

- [1] : Bye Bye, Bullets: The Stack Overflow Developer Story is the New Technical Resume - Stack Overflow
- [2]: How to Make a Resume That Stands Out: Examples & Tips (zety.com)
- [3]:https://www.bing.com/search?q=traditional+cv+vs+online+cv&cvid=9cc0fc1f7d15409e905e8a6320a098b9&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQABhAMgYIAhAAGEAyBggDEAAYQDIGCAQQABhAMgYIBRAAGEDSAQkxMjE0OGowajmoAgSwAgE&FORM=ANAB01&PC=DCTS
- [4]: 10 Resume Writing Tips To Help You Land a Position | Indeed.com
- [5] : Academic CV templates for applications or to send to professors? (MS CS) : r/gradadmissions (reddit.com)
- [6] : Mohd Khairuddin, Nurul Syahmina, et al. "Re: Gen-web-based resume generator with content recommender for it job field." (2022):
- [7]: Khaled, Tahsin Bin. "Web-Application on CV-Building." (2023).
- [8]: Mulla, Aslaan, et al. "Resume Builder Application with Automated Job Prediction." NeuroQuantology 21.1 (2023): 128.
- [9]: Shivhare, Kratika, Sonam Shakya, and Aashi Singh Bhadouria. "ResumeCraft: A Machine Learning-powered Web Platform for Resume Building".
- [10]: <https://en.wikipedia.org/wiki/HTML>

[11] : <https://en.wikipedia.org/wiki/CSS>.

[12]: "ECMAScript 2020 Language Specification". Archived from the original on 8 May 2020. Retrieved 8 May 2020.

[13]: https://en.wikipedia.org/wiki/JavaScript#cite_note-tc39-11

[14] "Search · stars:>100000". GitHub. Retrieved December 4, 2022.