What is Web Development?

I think you all are aware about the word ‘web development’.
Let’s leave what you know or not know. Let’s begin to scratch.

Imagine the Internet is a giant city, and websites are the houses, shops, and buildings in that city. Web development is building these digital houses from the ground up.

Now your question should be what is the Internet ?
The internet is like a global digital highway system that connects all computers, phones, and smart devices worldwide. The internet is a network of connected computers that share information. It allows devices to communicate instantly, no matter where they are.
Just know,

> **The Internet = The Entire City**
> It's the massive network connecting everything (like roads connecting all houses).

Types of Software Applications

Software applications can be broadly classified into different categories depending on their purpose, platform, and usage. Below are the major types:

1. Desktop Applications

Software installed and executed on a standalone computer or laptop. These applications store data locally and often don’t require internet access to function.

Use Cases:

- Office productivity (Word processors, spreadsheets).
- Media editing (photos, videos, audio).
- System utilities (antivirus, file management).

Examples:

- Microsoft Office Suite
- Adobe Photoshop
- VLC Media Player
- Visual Studio

2. Web Applications

Applications accessed via a web browser using the internet. They don't require installation and are hosted on remote servers.

Use Cases:

- Social networking platforms.
- Online banking and e-commerce.
- Online collaboration tools.

Examples:

- Gmail, Yahoo Mail
- Google Docs, Microsoft 365 Online
- Facebook, Twitter, Instagram
- Amazon, Flipkart

3. Distributed Applications

Applications where different parts (components) run on multiple computers connected via a network but appear as a single system to the user.

Use Cases:

- Online multiplayer games.
- E-commerce systems.
- Banking and transaction systems.
- Cloud-based enterprise applications.

Examples:

- Google Drive and Dropbox (cloud storage).
- Online banking systems.
- Multiplayer games like PUBG, Fortnite.

4. Mobile Applications

Applications designed to run on mobile devices like smartphones and tablets. Available via app stores.

Types:

- **Native apps**: Platform-specific (Android/iOS).
- **Hybrid apps**: Cross-platform, built with frameworks like React Native, Flutter.
- **Progressive Web Apps (PWAs)**: Run in browsers but behave like mobile apps.

Use Cases:

- Social networking and communication.
- On-demand services (food, taxi booking).
- Banking and payments.
- Gaming and entertainment.

Examples:

- WhatsApp, Telegram
- Uber, Zomato, Swiggy
- Instagram, TikTok

5. AI Applications

Applications that use Artificial Intelligence and Machine Learning to mimic human intelligence, automate tasks, or provide predictions.

Use Cases:

- Virtual assistants and chatbots.
- Fraud detection in banking.
- Healthcare (diagnosis, drug discovery).
- Autonomous vehicles.

Examples:

- ChatGPT, Google Bard
- Siri, Alexa, Google Assistant
- Netflix and Amazon recommendation systems

6. IoT Applications

Applications that connect and control IoT devices, enabling real-time communication between physical devices and the internet.

Use Cases:

- Smart homes (lighting, thermostats, security).
- Healthcare (wearable health monitors).
- Smart cities (traffic, waste management).
- Industrial IoT (predictive maintenance).

Examples:

- Google Home, Amazon Alexa
- Smartwatch health apps
- Smart appliances (TVs, fridges, ACs)
- Industrial IoT platforms (Siemens, GE Predix)

7. 2D and 3D Applications

Applications used to create, design, or visualize two-dimensional (2D) or three-dimensional (3D) models, graphics, or environments.

2D Applications:

- Focus on flat images and designs.
- Used for illustrations, photo editing, digital art.
- Examples: Adobe Photoshop, Illustrator, Canva, CorelDRAW.

3D Applications:

- Create realistic 3D models, animations, and environments.
- Used in architecture, game development, VR/AR, film-making.
- Examples: Blender, AutoCAD, Unity, Unreal Engine.

Use Cases:

- Graphic design and digital art.
- Game and animation development.
- Architecture and engineering simulations.
- VR/AR experiences.

Web Development Career Paths and Opportunities

A career in web development can open up diverse and exciting opportunities, allowing professionals to branch out into various fields depending on their interests and skills. Here's a detailed breakdown of possible career paths after gaining experience in web development

Web development career paths come with various benefits, including

Job flexibility
Competitive salary packages
Opportunities for innovation
Constant learning
Easy to begin
High demand for skilled professionals
Interesting jobs

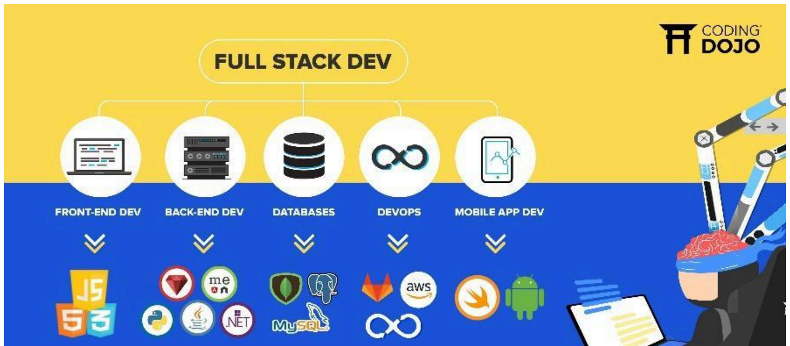
Web development is a promising career, especially as companies embrace digitization. Businesses want to establish a solid online presence and need talented web developers to build and maintain engaging websites. The industry is known for job stability, competitive salaries, and ample growth opportunities.

Do you want to make a successful **career in website development**? Then, there are ample options to explore and choose from. However, you must **learn multiple web development** skills and professional knowledge, which is true for a career in any field. The following are the web development career paths to consider:

1. Full-stack Developer
2. Frontend Developer
3. Backend Developer

1. Full-stack Developer

Full-stack developers manage frontend and backend development, i.e., client-side and server-side. These professionals have extensive knowledge of all aspects of web development, including user interface design, server configuration, and database management.



Whether startups, small businesses, or large enterprises, companies need knowledgeable and skilled professionals to manage their websites and web apps and streamline various tasks related to **web development projects**.

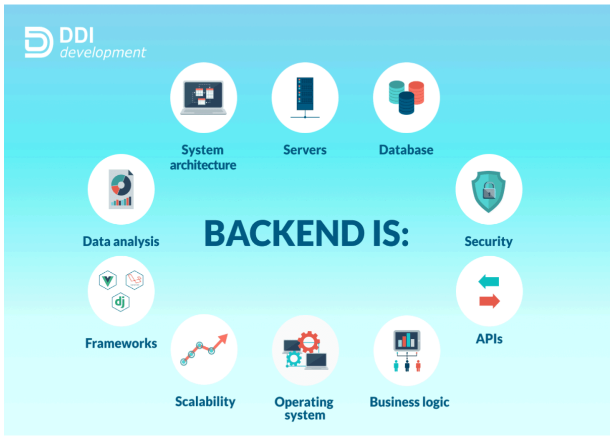
If you want to **become a full-stack developer**, you need to master various technologies, including HTML, CSS< JavaScript, Python, Java, Node.JS, React, MySQL, etc.

2. Frontend Developer

Frontend developers design and create the user interface, providing a seamless experience to users. As the emphasis on user-centric and responsive interfaces increases, the demand for frontend developers will also grow.

A front-end developer uses the knowledge of HTML, CSS, and JavaScript to design different elements of a web page and make it appealing and interactive. To make **frontend web development as a career**, it is important to master different frameworks and libraries, including Vue.JS, Angular, and React. Frontend developers can apply for various job roles, such as web designers, UI/UX designers, and frontend engineers.

3. Backend Developer



Backend developers deal with the server side of a website or web application. They build and maintain the backend, which involves writing code and ensuring smooth functionality. They work with server scripting, databases, and APIs to store, retrieve, and process data without hassle.

They must have expertise in various backend languages, including Ruby, Python, PHP, Java, Django, etc. Moreover, familiarity with cloud computing platforms, including Azure, AWS, and Google Cloud, is also essential if you want to crack the **web development interview** and get your dream job.

Backend developers can apply for multiple job roles, such as system architects, web developers, and software engineers.

Static vs. dynamic websites

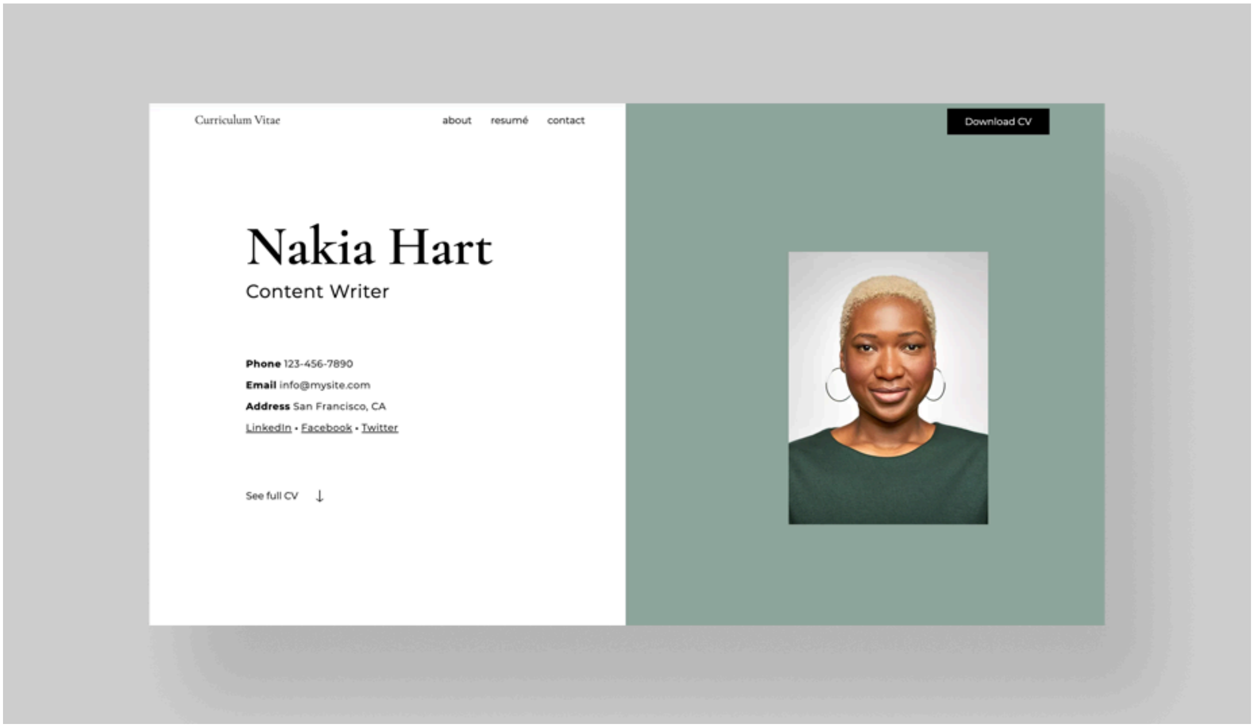
The underlying mechanisms and capabilities of static and dynamic websites differ significantly, impacting factors such as interactivity, customization, scalability and maintenance requirements.

Static websites consist of fixed content stored on the server and delivered to the user without any server-side processing. These websites are simple and easy to host but lack interactivity and require manual updates. Dynamic websites, on the other hand, generate content on the fly using server-side processing and databases. They offer interactivity, personalized content and dynamic features but are more complex to develop and maintain.

What is a static website?

A static website is made up of web pages created using HTML, CSS and Javascript (all examples of web development languages). Each page on a static website is stored as a single HTML file, which is delivered directly from the server to the web page exactly as is. This content essentially becomes a part of the design on your page, and won't change unless the original HTML file is edited at a code level.

Changes to a static website can be done manually, and will only be made page by page, HTML file by HTML file. For example, edits made to the HTML file of a homepage will only be reflected on the homepage. This is true even for elements that are identical across the whole site, such as the footer. If you're using a website builder, changes to static pages will be made automatically every time you use the website editor.



Nakia Hart

Content Writer

Phone 123-456-7890
Email info@mysite.com
Address San Francisco, CA
LinkedIn • Facebook • Twitter

One of the most characteristic aspects of a static site is that every user receives and views the exact same content. Because of this, static websites work best for sites with fewer pages that don't require frequent updates or changes.

A good candidate for a static site is a resume website. This is a type of site with set content for each page and doesn't require many changes to individual pages, or real-time updates based on user behavior. Other examples of common static website types include personal websites, nonprofit websites and purely informative websites (good examples of these include one-page or landing page sites).

Advantages of a static website

Faster page loading speed

The makeup of a static page prioritizes load speed, resulting in a better browsing experience. Because the content on this type of site is pre-written and delivered directly from the server, caching is easier and the content is less likely to load with delays or UX issues, such as broken images.

Quick creation

When you are thinking about how long it takes to build a website and time is an issue, a static website is easier to get live quicker. Static websites are faster to create and publish since they are less complex and don't need to be connected to databases of organized content. This is even more true if built on a WYSIWYG platform.

Lower hosting costs

Static websites can be incredibly wallet-friendly when it comes to hosting fees. Because they don't rely on databases or heavyweight backend processes, they require fewer server resources. This often translates to using lower-cost hosting plans or even free hosting solutions, depending on the scale of your site.

Disadvantages of a static website

Limited scalability

One of the largest disadvantages of a static website only comes into play with larger, content-heavy designs. While it's possible to build hundreds of pages with a static website, it will always be a slow and long process. This is less relevant when creating a personal website though.

Less efficient management

Static websites may be quicker to create, but they can be more time-consuming to manage. Edits to a static website need to be made page by page, and as websites are loaded with more content, or rapidly changing content, this becomes a much more challenging—and in some cases, near impossible—task.

What is a dynamic website?

Built using server-side language and technology, dynamic websites allow for the content of each page to be delivered and displayed dynamically, or on-the-fly, according to user behavior or from user-generated content.

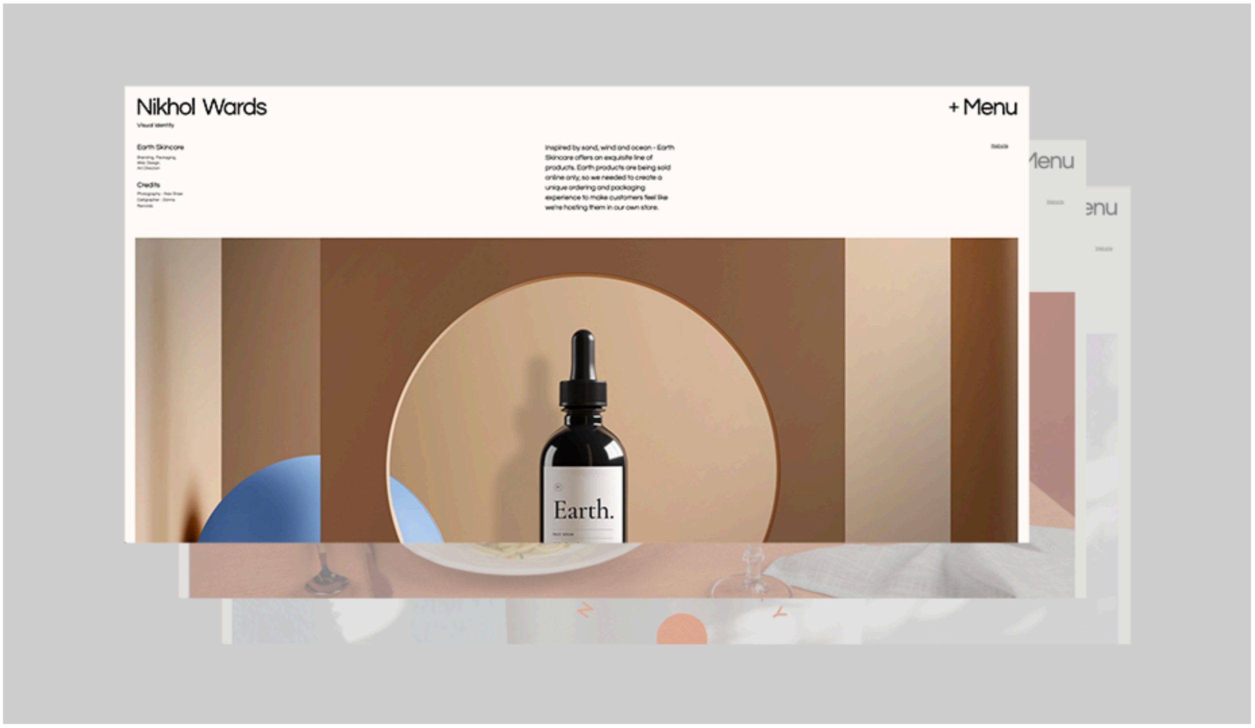
With a dynamic website, all of your data and content are organized in a database or backend content management system (CMS), which connects to your website pages. The way this information is arranged and connected to your site's design controls how and when its content is revealed on a page.

What does all of this mean? Well, dynamic content gives you the ability to customize and personalize the website experience, and what is displayed, for a specific user. It also allows you to make changes to many pages at the same time, since modifications made to one dynamic page can be automatically made across thousands.

For example, dynamic websites enable you to choose which information is displayed to a user based on their location. You can also deliver content to users based on their current or past actions on your site (thanks to cookies), which essentially means each visitor sees a different view of the content on a page. A multilingual website is a great example of when creating a dynamic website might be relevant.

Other examples of well-known dynamic websites include:

1. **Instagram**: as a social media site, dependent on user-generated content, Instagram relies on a dynamic website.
2. **CNN**: media outlets use dynamic websites to update their content, either in response to breaking news or as stories age.
3. **Disney Plus**: as a large streaming site, this dynamic entertainment website's dynamic nature allows its content to be chosen and displayed according to a user's location, subscription and preferences.



Inspired by sand, wind and ocean - Earth Skincare offers an exquisite line of products. Earth products are being sold online only, so we needed to create a unique ordering and packaging experience to make customers feel like we're hosting them in our own store.

Generally, dynamic websites are those which are content heavy and user-driven. Let's say the main purpose of your website is to act as a real estate listing website. You'll need to generate hundreds of pages to list hundreds of available properties. In order to improve the functionality of your site and accommodate a user's intent, the content on these pages will need to reflect the real-time availability of properties. Using dynamic pages will be the most efficient way to display these changes on your site.

Advantages of a dynamic website

For many website creators dynamic pages are the only way to go, and for good reason. Dynamic pages have the following advantages:

Easily updated

Starting a business and building a brand online requires continually updated content. You need to stay current with trends, updates and changes within your business, as well as within your industry. A dynamic website is the most effective way to do this.

With a dynamic website, a content change on one page can be automatically duplicated on other pages without needing to alter the design. This is particularly relevant to sites with a large number of pages since it makes maintaining a website more efficient.

A better user experience

A dynamic website provides content that's tailored to the needs of the user. This might mean displaying information on the page based on their location or changing content to reflect their interests, intent or past actions on the page.

Greater functionality

Static pages can be interactive, but when it comes to functionality, dynamic pages definitely lead the way. Dynamic pages have boundless functionality—limited only by the complexity of the logic and language needed to build them, and the instructions needed to deliver content.

Potential for personalization

Dynamic websites excel in their ability to create a personalized experience for users. By storing and analyzing user data, such as browsing history, preferences or location, dynamic pages can display tailored content that aligns with each visitor's interests.

Disadvantages of a dynamic website

It takes more resources to create

Because of the extra steps needed to organize and connect your database to the right pages, a dynamic website can be more complicated to set up and get running. it will take more time to go live and can be more costly, too.

Performance issues

Dynamic websites have more instructions to process than static websites do. They are also connected to a database or content collection and continually pull information from that in order to display it—which takes time to process and execute. This can impact the performance of a site, although many website creation tools are aware of this issue and make it their mission to prioritize performance across all pages.

Evolution of HTML

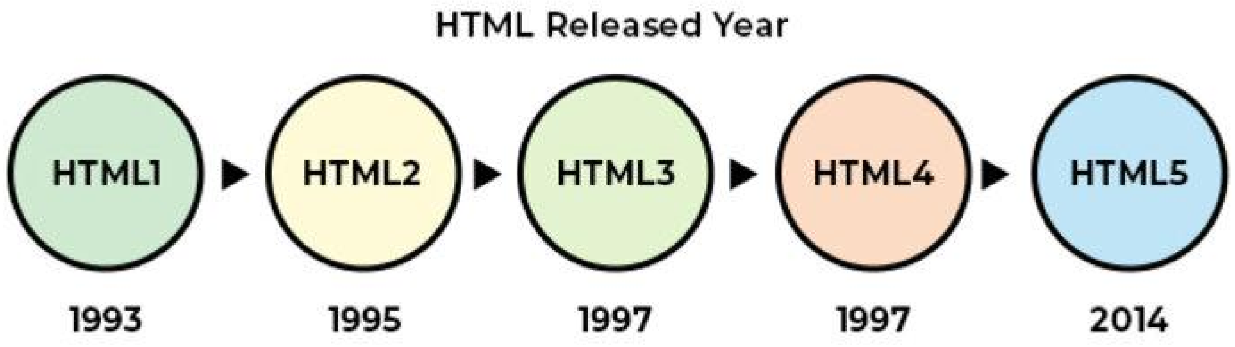
HyperText Markup Language (HTML) was initially developed by Sir Tim Berners-Lee in late 1991. It was designed as a standard language for creating and formatting documents on the World Wide Web. All the web pages on the internet are made from HTML.

From 1990 to 1995, HTML underwent changes and extensions, initially at CERN and then at the IETF. The World Wide Web Consortium (W3C) became the new home for HTML development.

The Idea of Hypertext Before HTML

We must first examine the idea of hypertext in order to comprehend the origins of HTML. Early 20th-century pioneers like **Vannevar Bush** proposed the concept of tying information together through hypertext, envisioning a "memex" machine that could organize enormous volumes of information using linked microfilm.

However, Ted Nelson, an American philosopher and sociologist, first used the word "hypertext" in the 1960s. Nelson's idea of hypertext was to develop a network of connected text and multimedia that permitted non-linear information navigation.



The Timeline of HTML's Evolution

Here you will see the evolution of HTML over the past couple of decades. The major upgrade was done in HTML5 in 2012.

Year	Progress
1991	Tim Berners-Lee created HyperText Markup Language but it was not officially released.
1993	Tim Berners-Lee created the first version of HTML that was published and available to the public.
1995	HTML 2.0 was released with a few additional features along with the existing features.
1997	There was an attempt to extend HTML with HTML 3.0, but it was replaced by the more practical HTML 3.2.
1998	The W3C (World Wide Web Consortium) decided to shift focus to an XML-based HTML equivalent called XHTML.
1999	HTML 4.01, which became an official standard in December 1999, was the most widely used version in the early 2000s.
2000	XHTML 1.0, completed in 2000, was a combination of HTML4 in XML.
2003	The introduction of XForms reignited interest in evolving HTML itself rather than replacing it with new technologies. This new theory recognized that XML was better suited for new technologies like RSS and Atom, while HTML remained the cornerstone of the web.
2004	A W3C workshop took place to explore reopening HTML's evolution. Mozilla and Opera jointly presented the principles that later influenced HTML5.
2006	The W3C expressed interest in HTML5 development and formed a working group to collaborate with the WHATWG. The W3C aimed to publish a "finished" HTML5 version, whereas the WHATWG focused on a Living Standard, continuously evolving HTML.

2012	HTML5 can be seen as an extended version of HTML 4.01, which was officially published in 2012.