full stack development

Summary of full stack development

Recall

- What is front-end?
- What is back-end?

Web browser + Code editor

The three main building blocks of the web

- HTML (structure)
- CSS (looks)
- Javascript (functionality)

Overview

Frontend Development (Client Side)

It is the visible part of website or web application which is responsible for user experience. The user directly interacts with the front end portion of the web application

or website.

HTML is the fundamental technology used to define the structure of a webpage.

- Semantic HTML
- SEO best practices

CSS defines the HTML document's presentation. CSS can also be used for webpage animations.

- CSS Box model
- CSS specificity
- Flexbox, grid
- Responsive designs
- Psuedo elements

Javascript is the brain of a webpage.

- DOM manipulation
- Creating, Selecting, Appending elements
- Event listeners

Alongside learning Javascript basics, you can also learn a CSS framework, such as `Bootstrap`. They allow you to create professional and consistent CSS designs, with less code.

It is also helpful to learn CSS preprocessors, such as `SASS/SCSS`. They add additional features that don't exist in pure CSS. They increase the readability and maintainability of CSS structures.

Once comfortable with javascript basics, it is helpful to learn about using APIs, and fetching data with the `Fetch API`. This will allow you to create limitless applications.

After that, advance by learning:

- Asynchronous Javascript
- Promises
- Async Await

- Scoping, Hoisting, Closures, etc.

After this, learning a Javascript framework or library, such as `React` or `Angular`, comes next. It is important to understand the Javascript language before starting a framework or library.

Learn a version control system, such as `Git`. Git is the best and most popular by far. `Github`.

Learn how to deploy your projects and website. There are many services for deploying static websites, such as `Netlify`, `Vercel`, and `GitHub Pages`.

A package manager is a tool that automatically handles plugins and tools. The most popular ones are `npm` and `yarn`. Npm is also an online database of mostly free packages. Packages contain code written by others to solve a problem, packaged for ease of use.

Javascript Frameworks/Libraries

You must know and understand the core fundamentals and advanced features of Javascript before learning a framework/library. It is also helpful to know:

- ES6+ features
- Array methods
- Shorthands
- Destructuring
- Spread operator
- Promises
- Async Await
- Import/Export

> 📘 Frameworks typically take all of the complex and repetitive tasks in setting up a new web application, and greatly simplify it. They allow you to build powerful, single-page applications with organized and interactive user interfaces.

The frameworks are used to make building and working with programming languages easier. They have many advantages over vanilla Javascript. There are many benefits. The most popular are `React`, `Vue`, and `Angular`, with React taking the lead.

- React is technically a library, not a framework

> 📘 A library, opposed to a framework, simply provides you with more freedom in how you want to get things done. Frameworks have stricter code practices.

React is easy to understand, is the most popular, has cross-platform support, and an amazing community.

Vue is newer and easier to learn than Angular, which is older.

React

When you first start learning React.js, you should know about:

- File structure
  - How everything works with React DOM, and a single HTML div
- React components
  - The difference between function-based components (best practice) and class-based components (old, decrepit)
- JSX syntax
  - Props
  - State
  - Events
  - Conditional Rendering

After learning React basics, learn about React.js Hooks. The essential hooks are `useState`, `useEffect`, `useRef`, `useContext`, `useReducer`, `useMemo`, and `useCallback`. It is also helpful to learn a React.js UI Kit. The most popular are `Material UI`, `Ant Design`, and `Chakra UI`. UI frameworks allow you to import and use different components to create a user interface in your React applications. This allows you to focus on the logic of your projects while still keeping a clean, modern design.

To make the most out of React, learning packages is necessary. The most popular are `React Router`, `Styled Components`, and `Axios`. They are simple npm packages.

Once you are truly familiar with React, you may gravitate to doing everything using props. This is prop-drilling. As you build larger and larger projects they will be harder to keep clean and organized. A state-management tool can help with this. The most popular one is `Redux`. State management is a way to improve communication and data sharing across your components, without prop-drilling.

`Next.js` is a framework built on top of React.js, that allows you to do many more things that vanilla React doesn't naturally allow.

There are great libraries and frameworks that exist for testing as well. Jest, Cypress, Enzyme, Jasmine, and Mocha are a few.

You can also learn about PWA (Progressive Web Applications), and web sockets. Web sockets allow you to send data in real-time. This is useful for messaging applications, social networks, streaming services, etc.

There is also

- CORS
- JSON
- Basic Security (XSS, CSRF)
- Web accessibility

> 📘 Remember: Take your time. It will take months, maybe years to fully learn all of this.

Backend Development

It refers to the server-side development of web application or website with a primary focus on how the website works. It is responsible for managing the database through queries and APIs by client-side commands.

`Node.js` is a runtime environment for Javascript. It allows you to use Javascript on the server-side. Other backend languages are Python, Php, Ruby, Java, etc.

Each language will have a backend framework for that language. For Node.js, it is `Express.js`. It will allow you to create better applications more quickly.

Knowing these concepts will help with backend development:

- HTTP/HTTPS
- APIs
    - REST vs GraphQL
- Web Sockets
- CORS
- MVC
- CI/CD
- Serverless

The backend deals with databases. There are two types, SQL (Structured Query Language) and NoSQL. NoSQL means ‘non-relational’. This means it is not based on the table-like relational structure that SQL uses. There is a different mechanism for the storing and retrieval of data.

The most popular SQL databases are:

- MySQL
- PostgreSQL
- SQL Lite

The most popular NoSQL database:

- MongoDB

`MongoDB` is part of the most popular programming stack: `MERN`

- MongoDB
- Express.Js
- React
- Node.Js

There are other tools that are important when working with databases, such as ODM (Object Document Mappings). These are used to make interacting with the databases easier. For MongoDB, the most popular ODM is `Mongoose`.

Other essential topics:

- Authentication (allows user to log in and register)
- OAuth (simpler way of registration using popular platforms such as Google Login)
- Hashing (bcrypt. They allow you to encrypt user’s passwords)
- Rate Limiting
- Reverse Proxy
- Documentation
- Load Balancer
- Postman