Frameworks

When working on building new applications, sometimes, work gets complex. Eventually, it gets to the point where just having libraries are not enough. What is needed is structure, and order in the chaos of code. That is where Frameworks come into play.

What is a Framework?

A web framework is like the support beams of a house. It provides a structure or template of resources, services, and APIs to alleviate the overhead associated with web development.

Why are web frameworks important?

Frameworks act as a foundation to build your code up on, and can establish a sense of order when code in an application seems to not have any order at all. Frameworks can be highly customizable to the framework developer's preference, and for the user of the framework being able to add the useful methodology of libraries integrated into the framework's structure, or automatically executing code blocks when needed by calling MVC data. To cut things short, frameworks are needed when code for an application or program seems to have zero structure and rough functionality, because as frameworks, they mandate how your code is structured and how it will run.

Choosing the right framework upfront is critical as a huge amount of effort is required to switch later. There are a lot of factors to consider such as ease of use, security, efficiency, and maintainability. An active and popular framework is important as this gives developers more tools, community, and resources to leverage during development. This also means that patches should come out regularly to resolve bugs and security vulnerabilities. Also keep in mind to look at implementation best practices around the framework to ensure future maintainability and easier upgrades.

What's React?

React is (JavaScript library) a free and open-source front-end JavaScript library for building user interfaces or UI components. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications

JSX:

JSX is a syntax extension to JavaScript. It is used with React to describe what the user interface should look like. By using JSX, we can write HTML structures in the same file that contains JavaScript code. This makes the code easier to understand and debug, as it avoids the usage of complex JavaScript DOM structures.

Components

A Component is one of the core building blocks of React. In other words, we can say that every application you will develop in React will be made up of pieces called components. Components make the task of building UIs much easier. You can see a UI broken down into multiple individual pieces called components and work on them independently and merge them all in a parent component which will be your final UI.

Props:

Props stand for "Properties." They are read-only components. It is an object which stores the value of attributes of a tag and work similar to the HTML attributes. It gives a way to pass data from one component to other components. It is similar to function arguments. Props are passed to the component in the same way as arguments passed in a function.Props are immutable so we cannot modify the props from inside the component. Inside the components, we can add attributes called props. These attributes are available in the component as this.props and can be used to render dynamic data in our render method.

Advantages of React

- Easy to understand and easy to use
- Creating Dynamic Web Applications Becomes Easier
- Reusable Components
- The Support of Handy Tools
- Known to be SEO Friendly

Disadvantages of React

- The high pace of development
- Poor Documentation
- ReactJS only covers the UI Layers of the app and thus there is always a need to choose some other technologies for other development.

SPA

A Single Page Application is a web application or website that interacts with the web browser by dynamically rewriting the current web page with new data from the web server, instead of the default method of the browser loading entire new pages.

This means that the URL of your website will not change completely (page will not reload), instead it will keep getting content and rewriting the DOM with it instead of loading a new page.