Web Frameworks and their role in the future of Web Application development

The web is becoming the new ‘App Store’. Most of the functionalities can be implemented to be used in the web browser, using extensive libraries and APIs. In this project I compare the use of Web Application Frameworks for creating these web apps, and in this essay I write about the challenges I faced, the eureka moments, what I learnt during my comparison of these frameworks for the Charity App, created as part of WEB701 course. In the end I give my limited opinion on how the web application development methods and techniques might change in the future.

The new web frameworks to create web applications emerge as developers work on creating dynamic websites using vanilla javascript and HTML. Solutions to their challenges inspire others to do it the same way. Collective use of these frameworks allows for consistency in everyone's code, easy debugging and new libraries based on top of those frameworks, which make them even more powerful.

In my project for WEB701 I compared Angular, Vue, and Svelte frameworks to create my website. This same website could have been created using vanilla javascript but that requires a lot of extra code which very soon becomes hard to understand and maintain. Most of the frameworks divide this code into different components (group of HTML, CSS and Javascript) These components have state (the values of variable data), the change in state in one component can trigger an event in another, just like DOM events, but the difference being that most of these frameworks create their own virtual DOM and renders data through that Virtual DOM.

This was my second project using web frameworks, the first one was using React. Just the ability to render the HTML using javascript logics, directly inside the HTML template can be enough reason to transition to using web frameworks. However they offer much more than that. The libraries for each framework provides added functionality and features like better testing, linting (error notification while writing the code), better organization of code and several APIs which work only with specific frameworks (Eg- Transition APIs in Angular and Svelte) which in all leads to apps with better UI, less chance of errors, added functionality - all by writing less(most times) and easily maintainable code.

The choice of web frameworks is usually based on the following factors:

1. Your familiarity with the language used by Frameworks (Eg, Angular Uses Typescript while Svelte uses vanilla javascript) or different variations of those languages.
2. Project scale and weather new features would be added in the future.
3. The libraries and APIs provided in the framework. Eg. UI libraries and API to interact easily with users' devices.
4. The team size, i.e does the framework forces consistency in code so that it can be understood by other team members.

5. The online community size and libraries provided on the framework.

The challenges I faced and how I dealt with them when transitioning from using vanilla Javascript to using Web Framework and when creating this project are:

1. Not having my basics of Javascript very clear, the advantages offered by the libraries provided by the framework is hard to implement using just the official documentation. Therefore it is best to see how the basic layer of object prototype in Javascript works and get familiar with anything written the way it is. Because Javascript or any programming language has defined syntax for everything, once the syntax is clear- I understood how everything in javascript is an object/ prototype of another object and then I can easily search for any methods/libraries that are unfamiliar to me, rather than getting confused.
2. I've found that reading the documentation is most of the times the most helpful thing. Following the online guides and including the code from the internet without reading the documentation for libraries/packages being used is most often the cause of errors, because the packages can update and the code you are debugging might not be relevant with the new update.
3. Even though frameworks force the developer to organise the code, not putting every content in the specific components and thinking that you will remember it/figure out can lead to a lot of time wasted debugging if an unusual case occurs. The same is true when the data in one component is bound with multiple components, which can lead to confusion in finding which component is triggering the change in value.

Future of Web Frameworks:

My opinion, based on my limited experience with web frameworks, is that no one framework would rule the rest and all can co-exist. Even though a larger community provides for more libraries to a framework it also hinders the change in basic use of the framework, yet also a framework with new language syntax or an entirely new syntax can be revolutionary for that community or those projects but does not affect the popularity of major frameworks.