How does the development of a single page application with a client-side framework + server-side API compare to a more traditional web development methodology?

In a published journal article from five years ago, Roser etal (2016), describes the increase in global internet users over the preceding fifteen years. Showing 413 million users at the start of the 21st century, passing the first billion milestone in 2005. Finally at the time of their publication, 2016, 3.4 billion internet users, with a staggering 640,000 new users each day starting their own online experience. These statistics are further updated by Kemp (2021) who reports that as of January 2021 there are approximately 4.66 billion users globally of the internet, representing almost sixty percent of the entire human population. This staggering volume of internet users has by pure necessity of scale also had the effect of increasing the global requirements for web developers. This is in part due to existing businesses, as well as new businesses, moving some if not all their service provision to being online. It is unlikely that any new enterprise would not want the most basic of internet presence in today's marketplace.

Figure 1 Sega Net link, Amos 2011.

As the number of people online has increased so to have the methods for accessing the internet. Where originally, a desktop computer would be needed to access the internet, typically via a browser such as Internet explorer or Netscape Navigator in the earlier days of browsing. More and more devices adopted this browser-based approach to getting online, one such category is that of video game consoles with Sega being the first to release a modem adaptor for its Saturn console in 1996 (Fitzsimmons, 1996). This inclusion of internet connectivity is still found in across the spectrum of videogame console generations that followed. Though there are still many ways that a person can get online, smart television or digital tablets, it could be argued that none is more prolific than the mobile smartphone. O'Dea (2021) reports via the Statista web site that there were 1.5 billion smart phones sold on average, each year since 2016. This could be extrapolated to a conclusion that over 50% of internet users, access the internet from a mobile based device.

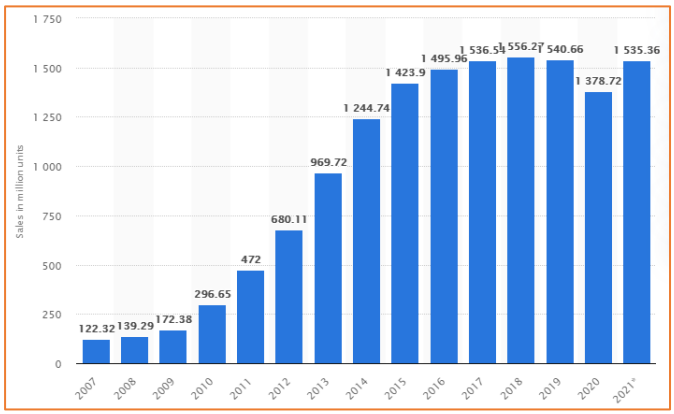

Figure 2 Smart Phone sales 2007 onwards, Statista 2021

Approaching this as if it were a triangle, with the first side being the number of online users, the second side being the various methods of accessing the

internet, then the logical third side is the technologies used to provide the information and services that the internet has to offer to the end user. In the early days this was in the form of HTML based webpages, providing simple information and navigation from one page to another, everything being held on the server and the browser simply showing the data to the user. As time progressed, layers of complexity began to be introduced with such elements as frameworks and additional languages such as PHP and JavaScript. Each of these elements spawning tools for the developers to implement them as well as well as increased functionality for the end user. One thing that is abundantly clear from looking at the complex topic of web development is that it is evolving very rapidly with some elements being carried forward whilst others being left behind to enter obscurity, with git hub repositories that may go un-updated for many years.

Exploring the issues faced in modern web development, one such issue is the move towards "single page applications". Indeed, it can be argued that more of the code that is run is taking place within the client and migrating away from the more traditional server side. As is common with single page applications the user will have the experience of being able to traverse between several pages whilst the reality is that the actual page has not been reloaded and components are being hidden and displayed to the user through interaction. One of problems with this approach to web development is that a single page application will typically result in a client-side codebase that can be cumbersome and unwieldy. This issue can be tackled, in part, with the introduction of reusable components and by enforcing stricter structure to the codebase.

A practice that is becoming more common, is to take the traditional established architectural frameworks that have existed for many years on the server-side and apply them to the more modern client-side single page applications. Many different variations of architectures have been proposed, however, almost all have three common elements, a view layer, a data layer and some form of controller sat between the two. Conventionally a user would see the view, which would be client-side, make some change triggering the controller, which would then talk to the model, server-side, then the model would finally update the view, client-side. One of the more common client-side frameworks makes use of RESTful API calls from the client to the server, where the API will return and manipulate JSON data. The name often given to this approach is JAMstack (JAMstack, 2021), it takes its name from the client-side based Javascript, reusable API's and pre coded Markup.

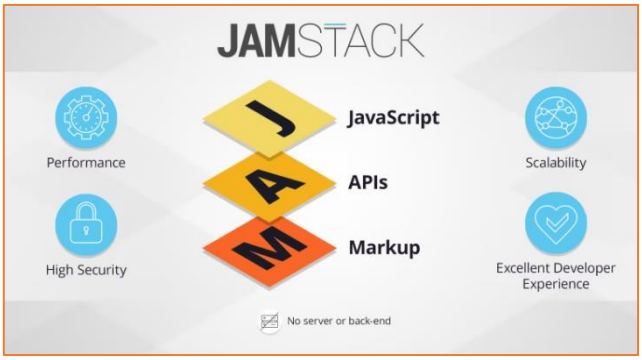

Figure 3 Cloudinary - JAMstack 2019

Web developers have a suite of available JavaScript frameworks, each offering similar base functionality whilst having, in

places, drastically different features. Google's Angular and Facebook's ReactJS are two of the most popular frameworks being used in modern web application development. A key concept utilised by both of these frameworks is UI data binding, this has the goal of being able to simplify the process of developing GUI based applications. It achieves this by making it that when a value is changed in one UI component/element, then the change is propagated throughout all other associated UI components/elements as well as making an API call and altering the model. Another concept that has been introduced to assist developers is observables. Observables are used to tackle data that arrives to the application asynchronously, meaning over time. Observables allow the framework to treat asynchronous data the same way that you would any other data collection such as an array. In his article Gruijs (2017) described observables as lazy collections of multiple values over time.

Helping to realise the creation and development of dynamic single page applications, alongside the above JavaScript frameworks, is an equally diverse and rich offering of web frameworks. Two of the more popular PHP frameworks found most regularly by developers are CodeIgnitor and Laravel. Most commonly web frameworks are created in the PHP language though not exclusively, also most of the more prevalently used frameworks are open source. Concepts adopted by some of these PHP web frameworks mirror those mentioned above, specifically the ethos of code reuse. This would normally occur through the use of components and patterns. Components could arguably trace their lineage back to object orientated programming languages, similarly they have a core use of being designed to make development easier by way of re-use throughout the codebase. Design patterns are more of an abstract concept, more of a way for web developers to tackle similar repeated development problems. When looking at design patterns it is possible to identify that they are typically made up of four distinct individual parts, a name, the problem, the solution and the consequences. This methodology has been tried and tested over many years and been shown to offer benefits such as helping to identify the problem and to generally promote better overall design.

The final element of the JAMstack to be explored is the markup code used to create rich and engaging graphical user interfaces for the end user of the web application. It can be argued that this is perhaps the least changed methodology from more traditional web development, insofar as that the HTML and CSS used is universal when applied to formatting the data for the end user. In her article Kyrnin (2021) makes the statement that whilst there are numerous markup languages, there are essentially only three primary ones that deserve consideration HTML, XML and XHTML. In practical terms components can be made within the JavaScript file for the single page web application and these components are able to be written in markup whilst allowing variables to be passed in with efficiency.

Following the initial stages of development for a web application, one stage that remains universal across both modern and traditional approaches is the testing of the app. Testing

provides the developer with invaluable information on the state of the system, ensuring that it is working as intended or if there are any problems that require further attention, before the application is released. Traditionally testing would be carried out manually and be both time consuming and monotonous. One of the advancements that is now available to modern web application developers is within the field of automated testing, sometimes referred to as continuous testing. There are many suites of tools and testing platforms that specialise in each area of the application, be that the JavaScript front end with Mocha, Jasmine or Jest. All the way to server-side test frameworks such as Postman.

Evaluating the original question, of how the development methodology of a more traditional web app contrasts to the development of a more modern single page application built with a client-side framework and coupled with a server-side API, several conclusions are able to be drawn. It is clear that one of the motivators driving the change in approaches is that of speed, how fast a web application can manipulate the graphical elements and components shown to the user, indeed why reload the entire page when just a small section has been changed or updated. There is also evidence of a clear push towards unifying the languages used to deliver all aspects of a web app within a single framework. Within a single ReactJS file it is possible to define a graphical user component, create it within the file with markup, then render this component multiple times to the user. In fact, the whole ethos of reusability of components is another contrast between the modern and traditional approached to web app development, arguably done in the name of more readable and less messy codebases.

Looking toward the future of web app development, it could be safely argued that whilst some of the largest internet-based companies, Google and Facebook are still consuming the vast levels of internet traffic, then their JavaScript frameworks, Angular and React, will not be disappearing into obscurity any time soon. In her 2021 article Lozhko attempts to identify some of the more popular web development trends that are being either used or being developed right now. Alongside the SPA trend, she draws attention towards serverless applications and architecture, proposing that such a path offers benefits such as avoiding system overloading, costly development and potential data loss. The final conclusion that can safely be made is that web app development is not and has not ever stood still, not even taken a breath, it is being pushed and pulled forwards at great pace by many factors not least the users and the businesses and organizations seeking to monetize the internet.

References

Gruijs, L. (2017) Understanding, creating and subscribing to observables in Angular. Available at: https://luukgruijs.medium.com/understanding-creating-and-subscribing-to-observables-in-angular-426dbf0b04a3 (Accessed: Apr 24, 2021).

Jamstack (2021) For fast and secure sites | Jamstack. Available at: https://jamstack.org/ (Accessed: Apr 24, 2021).

Jennifer Kyrnin (2021) What Are Markup Languages? Available at: https://www.lifewire.com/what-are-markup-languages-3468655 (Accessed: Apr 25, 2021).

Kelly Fitzsimmons (1996) Press release: 1996-09-17: Sega brings fun and games, ease-of-use to TV Internet market. Available at: https://segaretro.org/Press_release:_1996-09-17:_Sega_brings_fun_and_games,_ease-of-use_to_TV_Internet_market (Accessed: 24-04-21).

Mariia Lozhko (2021) 15 Top web development trends in 2021. Available at: https://lanars.com/blog/top-web-development-trends (Accessed: Apr 26, 2021).

Prosper Otemuyiwa (2019) Developer Experience for a Modern Web: JAMstack Delivers. Available at: https://cloudinary.com/blog/developer_experience_for_a_modern_web_jamstack_delivers (Accessed: Apr 25, 2021).

Roser, M., Ritchie, H. and Ortiz-Ospina, E. (2015) 'Internet', Our World in Data, .

s. O'Dea (2021) Cell phone sales worldwide. Available at: https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/ (Accessed: Apr 24, 2021).

Simon Kemp (2021) Digital 2021: Global Overview Report. Available at: https://datareportal.com/reports/digital-2021-global-overview-report (Accessed: Apr 24, 2021).