A single-page application is a more modern approach to app development. It was used by Google, Facebook, Twitter, etc. A SPA is an app that works inside a browser and does not require page reloading during use.

On the other hand, a multiple-page application is considered a more classical approach to app development. The multi-page design pattern requires a page reload every time the content changes. It’s a preferred option for large companies with extensive product portfolios, such as e-commerce businesses.

|  |  |  |
| --- | --- | --- |
| SPA | MPA |  |
| SPA’s load faster. As it loads the majority of app resources just once. The page doesn’t reload entirely whenever the user requests a new piece of data. | MPA is slower as the browser must reload the entire page from scratch whenever the user wants to access new data or moves to a different part of the website. | Speed |
| SPA is strongly decoupled, meaning that the front-end and back-end are separate. Single-page applications use APIs developed by server-side developers to read and display data. | In MPA’s, the front-end and back-end are more interdependent. All coding is usually housed under one project. | Coupling |
| SPA aren’t as SEO-friendly because most single-page applications are run on JavaScript, which most search engines do not support. Web pages are indexed through “crawling” or “spidering”. Search engine crawlers download the page’s HTML files which makes static HTML web-pages are easier to rank. | MPA enables better website positioning, as each page can be optimized for a different keyword. Also, meta tags can be included on every page – this positively impacts Google rankings. | Search Engine Optimization |
| In a SPA, all you have to do to keep your page safe is secure data endpoints faster but not necessarily safer. SPA’s are more prone to hacker attacks, as they run on JavaScript, which doesn’t perform code compilation making it more vulnerable to malware. | This probably won’t come as a surprise to you, but the larger the website, the more effort it takes to secure it – think multi-page applications. If you go for an MPA, then you’ll have to secure every webpage. | Security |
| SPA’s are more mobile-friendly, which is worth remembering as a lot of traffic comes from mobile devices. Even Google started to prioritize mobile experience over the desktop. Frameworks applied in SPA development enable you to develop mobile apps. | MPA’s, on the other hand, enable better [information architecture](https://vtldesign.com/web-strategy/website-design-development/brand-architecture-can-lead-better-seo-search-engine-results/). You can create as many pages as required, and you can include as much information on a page as you need without any limits. Navigation is clear, so the user can easily find their way around the website, which positively impacts their experience. | User experience |
| One of the greatest advantages of SPA’s is the reusable backend code. If you think reusable code equals less work, then you’re right. You can apply the same code you used in your web app to your native mobile app. It’s an important piece of information, as applications and websites are frequently used on mobile devices – which is no surprise since most of us are constantly on the run. | Thanks to the clear division between the front-end and back-end, both parts can be developed simultaneously, which speeds up the entire development process. MPA’s take longer to develop as in most cases, the server-side has to be coded from the beginning. | Development process |
| SPA lives and breathes JavaScript, which can be problematic. More search engines started to support JavaScript but with varying results. The level of support highly depends on the JS framework used. If the app is run on a browser with disabled JavaScript, it can cause app functionality problems, which might result in higher bounce rates and lower conversion. JavaScript reliance also contributes to its problems with SEO optimization and security issues. | MPAs can be built without any JavaScript dependency. | JavaScript dependency |