**Technology Comparison Sheet.**

| What type of technology are you comparing? | | |
| --- | --- | --- |
| Server Backends | | |
| Product / Technology | Pros | Cons |
| ASP.NET | Reasonably scalable | Made by Microsoft |
| Type safe | C#, which is made by Microsoft |
| Easy-to-use routers | Doesn’t have support for the more modern JS frameworks |
|  |  |
| Flask (/ other python servers) | Easy routers | Its python (ew)  i.e. not very fast or scalable |
| Fast to develop | Again, very little support for the modern js frameworks (react) |
|  | Very janky SSR/templating |
|  |  |
| Next.js | Suitable integration for modern web frameworks e.g. react |  |
| Easy routers, with support for passing information to react |  |
| Supported by cloudflare for free hosting |  |
| ServerSide Rendering, beneficial for client performance |  |
| NodeJS http.server | Low-level | Slow development |
| Powerful for any needed low-level packet management | Manual routing and handling |
|  | Code duplication |
|  | Hard to scale |

| Which technology will you be selecting? |
| --- |
| Next.js |
| What are the reasons for your choice? |
| Supported by cloudflare, allowing for free hosting during development, including DDOS protection and protection from other common vulnerabilities, for free. NextJS and Cloudflare are both incredibly scalable for any future expansion or large web traffic flow. |

**Technology Comparison Sheet.**

| What type of technology are you comparing? | | |
| --- | --- | --- |
| Frontend frameworks | | |
| Product / Technology | Pros | Cons |
| Svelte | Componentwise for code deduplication | Uses non-vanilla-js-style, unfamiliar, syntax, slowing development |
| Simplicity – simpler than react | Lack of support by community and IDEs slowing development |
|  | Less scalable compared to react |
|  |  |
| React | Componentwise for code deduplication | Less extensible to web/native |
| Easily abstractable |  |
| Can use Server Components to take advantage of SSR for faster client render |  |
|  |  |
| React Native Web | Component-based rendering allowing for code deduplication |  |
| Can compile to native for mobile support using common code |  |
| Reduced code deduplication allowing for faster development and more focus on functionality |  |
| Easy abstractions for code deduplication and extensibility for scalability and faster development |  |
| Vanilla (HTML, CSS, JS) | Small code size for faster load times | Incredibly unscalable when it comes to large projects |
| Static serving – can host on simple hosts like github/cloudflare pages | Much code duplication – no templating |
|  | Browser support varies |
|  |  |

| Which technology will you be selecting? |
| --- |
|  |
| What are the reasons for your choice? |
|  |

**Technology Comparison Sheet.**

| What type of technology are you comparing? | | |
| --- | --- | --- |
| UI Frameworks | | |
| Product / Technology | Pros | Cons |
| None | Reasonably scalable | Made by Microsoft |
| Type safe | C#, which is made by Microsoft |
| Easy-to-use routers | Doesn’t have support for the more modern JS frameworks |
|  |  |
| Tamagui | Easy routers | Its python (ew)  i.e. not very fast or scalable |
| Fast to develop | Again, very little support for the modern js frameworks (react) |
|  | Very janky SSR/templating |
|  |  |
| Tailwind css | Suitable integration for modern web frameworks e.g. react |  |
| Easy routers, with support for passing information to react |  |
| Supported by cloudflare for free hosting |  |
|  |  |

| Which technology will you be selecting? |
| --- |
|  |
| What are the reasons for your choice? |
|  |