MONASH University

FIT2101 Assignment 1

Analysis of Alternatives

Team 01

Henry Barnett, Jai Clapp, Zoe Parkinson, Nick Wang, Siyuan (Tony) Zhou

Contents

Summary of platforms

Web application:
A web application is a type of application that requires a consistent internet connection and is accessed through a web browser. What makes web applications unique is they follow a model of the user being provided a service through a third party hosted web service. This means a web application is not hosted locally, which makes it an easy and affordable option across multiple different devices.

Native application:
A native application is a type of application that is designed to run solely on a specific device. Native applications are similar to the desktop applications that are stored locally on a device and are launched off the device's operating system. Native applications are unique as they do not require an internet connection to perform functionality. This is because the required data is already installed onto the device when the native application is downloaded.

Criteria for selection

- Compatibility with devices: we need to decide whether it would be beneficial to allow this application to be used across multiple devices, and the cost associated with allowing for compatibility.
- Performance of the application: we need to consider whether the speed of the application is important, weighing up the benefits of increasing speed while increasing cost.
- Internet connection: we need to consider if we want our users to be able to access our application without a solid internet connection, if this is important to the target customer.
- Requirement for a hosting service: we need to consider the cost associated with a hosting service if our application requires one.
- Maintainability of application: we need to consider which application platform will allow for easy maintainability of code, ease of updates, or if this is an unimportant feature.
- Interactive features: we need to consider whether our application requires a large amount of interactive features, e.g. the ability for the user to open the application through a button on their mobile device.
- Safety and security: we need to consider whether our application requires a high level of quality assurance with the application's security.

Comparing on criteria

Compatibility with devices:
A web application provides easy compatibility with all devices. This is due to the fact that web applications are hosted online through a third party. As such, web applications are only limited by the compatibility with browsers across platforms. In comparison, a native application is hosted locally on the device. Therefore, the native application must be designed based on the device it is used on. Native applications are not compatible with a large range of devices. They must be changed for each device. It is clear that web applications are superior for compatibility, and an application of our type should be compatible across devices.

Performance of the application:
A web application's performance is dependent on the internet connection of the user, and reliability of the hosting service. There are more factors that could go wrong and slow down the performance. In comparison, a native application does not rely on an internet connection, the speed of the device is the only factor. Therefore, in terms of pure speed and performance, a native application is superior. However, with the majority of users having access to a reasonable internet connection, it is fairly safe to assume the performance differences will not be huge.

Internet connection:
A web application requires an internet connection and a native application does not. It is clear a native application is superior when we consider ease of access for a user. However, similarly to the performance of the application, we can assume that the majority of users will have a sufficient internet connection. Also, an internet connection allows for some features such as updates being done automatically without the user needing to download and install the update. Therefore, while this factor can still be considered, it is not as important as other factors.

Requirement for a hosting service:
Similarly to the internet connection factor, a web application will require a hosting service and a native application will not. This is due to native applications being hosted locally on the device. A third party host will have a cost associated with it. This cost must be considered when deciding on the platform we decide to use. However, for this factor, a native application is superior.

Maintainability of the application:
Web applications are easy to maintain, updates to the existing application can be done locally then updated on the hosting service with a little down time for maintenance. The user does not need to download anything to update the application. In comparison, a native application requires the user to download the update to apply it. Web applications are superior for maintainability. We need to consider whether our application requires regular updates, and if this is important to users.

Interactive features:
Web applications have limited access to the device features such as hand gestures, device buttons, etc. In comparison, native applications can use the full extent of interactive features. We need to consider whether it is important for our application to utilise these interactive features.

Safety and security:
Web applications are not controlled by a high level of quality assurance for security when the application is published. On the other hand, native applications must be published to an “app store” to be downloaded by users. These “app stores” provide a level of quality assurance, which means native applications are typically “safer”. For our application, we need to consider whether security is important and whether user’s data is at risk.

Recommendations

After evaluating the two platforms against our criteria, native applications are actually superior in more criteria. However, when we consider our application, the two most important criteria are maintainability and compatibility. Our weather application is rather simple in design and does not require high performance to be functional. Compatibility is important to reach a wide range of users, and maintainability is important to provide the latest weather updates. A native application just does not provide these important criteria for selection. Therefore, despite a web application requiring a hosting service and having limited device features at disposal, we will select a web application as our platform.

Summary of languages

JavaScript

JavaScript (JS) is an extremely versatile and prevalent language used for web development. The appeal of JS is that there is a very low barrier to entry (as modern browsers all provide a JS console), propensity towards rapid development due to its weakly-typed variables and forgiving null handling, and available resources to leverage when choosing technologies. Overall, JS tends to be the go-to choice when developing a web application.

Python

Python is a language that is familiar to all programmers and if not, is very easy to pick up. There are robust package resources to support any development need – offering many layers of abstraction to support very complicated behaviour using a small amount of code. Overall, Python tends to be preferred for its versatility and ecosystem of resources to save on a large proportion of development resources.

Criteria for selection

- Ease-of-use: The chosen language must have a low barrier to entry to aid in rapid development.
- Integration with front-end user interface: The chosen language should provide functionalities or frameworks that integrate well with displaying a user interface that is dynamic (e.g. HTML).
- Maturity: This considers resources such as APIs, documentation, support for features, and online community. A more mature language means that there is more support for developers, reducing the time and cost of each sprint.
- Maintainability: The language should have built-in features that help developers write more maintainable code such as abstractions offered by classes, type safety, and exception handling. This reduces the costs of adding new features.
- Testability: Testing needs to be applied in each stage of this project to ensure that the acceptance criteria are actually met. In terms of the language, this means having a well developed and versatile testing framework that allows unit, integration, and system level testing.
- Performance: High-level languages tend to have performance overheads and thus execute slower than low-level languages. However, we anticipate that with modern browsers, optimisations, and availability of CPU resources, the performance of our program is of minor consideration.
- Platform: Certain languages target certain platforms, so the chosen language must match with the chosen platform.

Comparing on criteria

Ease-of-use

JS and Python are both high-level languages that offer a wide range of abstractions and semantic keywords that makes code fairly understandable.

JS uses C-like syntax (code blocks using curly braces, explicit for loops, arbitrary indenting) and operators (e.g. ?: ternary operator) that can act as a barrier to entry for those unfamiliar with the syntax. On the other hand, Python uses English like keywords such that the code often resembles sentences and uses indenting to indicate code blocks, arguably being more intuitive for newcomers.

One point of consideration is also the team's familiarity with the language, which is skewed towards JS due to prior experience.

Integration with front-end user interface

JS has a large advantage in being designed to integrate with HTML/CSS to create a dynamic web page that can be further enhanced using available frameworks.

On the other hand, Python does not naturally integrate with HTML/CSS and must programmatically generate such files to render. This increases the development time and setup needed to add new pages to the application.

Maturity

Both JS and Python are mature languages that offer a wide support network of developers and resources to aid in development.

The main point of difference would be that JS, from the onset, has targeted web development whereas Python is used for a wide range of purposes. Therefore, for the specific purpose of web development, JS would have more support and resources compared to Python.

For example, JS offers many popular front-end frameworks like Angular, React, and Vue as well as back-end frameworks like Express and Node. On the other hand, Python is fairly restricted as the main popular frameworks would Django and Flask

Maintainability

Unfortunately, neither languages offer type safety, but both do have mechanisms for abstraction in classes.

Python has a rich and well-developed system for exception handling and assertions that help with enforcing pre- and post- conditions which makes it easier for developers to make changes. On the other hand, JS does not offer such robust error handling methods and is more reliant on handling of null values.

Testability

Python offers the unittest module that provides rich functionality for many levels of testing including mocking. Combined with assertions and exception handling, python tends to be quite easy to test and can be supported by a comprehensive testing suite.

JS also offers many testing frameworks like MochaJS and Jest that can be used to test parts of the program, however, most of the testing is likely to be done through extensive logging and system level testing.

Performance

As both are largely interpreted languages, the performance of each is about the same. However, JS works in a browser environment, of which each provides its own interpreter that may or may not make optimisations, increasing performance. On the other hand, different implementations of Python have different performance overheads.

Platform

JS is primarily targeted towards web applications but does offer frameworks that can be used to create native applications.

On the other hand, Python does not have a robust system for native applications and primarily offers Django for web applications.

Recommendation

The recommendation on language would be made with strong consideration towards ease-of-use (reducing development time and overheads), maturity (support for developers and robust functionality), integration with front-end, and maintainability (to decrease the cost of fixing bugs).

We see that JS and Python are well matched on many criteria, however, the propensity of JS towards web development complements our recommendation on platform. Although Python offers robust systems for testing, error handling, and readability, JS offers a wide range of resources and available frameworks to be considered, giving the team flexibility in development, as well as naturally integrating to the front-end with frameworks and support for HTML/CSS.

Our recommendation is that JS be used as the language for this project, with further considerations on specific technology stack including front-end and back-end frameworks.