Mobile Development Report

There are currently two main popular mobile operating systems running: Android and iOS where Android dominated with 74.44% market share and iOS followed by 24.98% worldwide. For consumers, the fluency and UI design of an OS can have a great impact on mobile sales.

For developers, the completeness of SDK (Software Development Kit) determines the workload and functionalities of a certain application. So, we will discuss the pros and cons of two main systems development and explain why we choose iOS over Android.

How to run the app:

1. Download XCode on a Mac (any of MacBook, Pro MacBook Air, iMac, iMac Pro, Mac Pro, Mac Mini)
2. Clone the repository on the Mac
3. Open CSC301A1Prototype folder using XCode
4. Run this project with an iPhone simulator(such as iPhone 11)
5. Choose the number of fruits, input promo code, and pay.

How to test:

1. Go to CSC301A1PrototypeUITests file in XCode
2. Run the Simulator first
3. Click  on line 10 and line 21 

(in between line numbers)
4. The simulator automatically launch and the UI tests succeeded

Option1: iOS Development

Development requirement

- Language: Swift / Objective C
    - Why Swift?
        - Swift is easier to learn because of the official documentation.
        - SwiftUI is a Swift-only declarative framework for making UI for multiple platforms with built-in support for new features.
        - Objective C will eventually be out of date for iOS development.
- IDE: Xcode
- Environment: macOS based computer (Limited)

The main reason we choose to develop iOS applications is we both have some experience in iOS development and the community for the project is strong enough to support us to build a light application by purely searching in a platform like Github, Stack Overflows etc. This takes us a little bit further to think that strong community support is essential in development especially for beginners. Lack of resources can waste a lot of time in debugging or finding appropriate methods to develop. However, almost always, a strong community cannot live with a poorly written programming language and/or lack of users of the operating environment (mobile operating system here).

Also, the SDK of iOS is complete and easy to learn. For example, we developed the front end of the application with SwiftUI, which can let us build a simple and fully functional UI in hours. SwiftUI provides multiple views including NavigationView, Text, Stacks, Buttons… This is great especially for a light application like ours.

The distribution method of iOS applications - App Store is greater than the way Android distributes the applications overall. First, from the report, iOS users are more willing to pay for apps compared to Android, and users determine the future of the industry and the willingness of developers for developing applications. Realistically speaking, iOS developers earn more money than Android developers on average.

From these perspectives, iOS development is more demanding than Android development.

Option2: Android Development

Intro to Android App Development:

- Language: Java
    - Why not Android?
        - Java is not as straightforward compared to Swift
        - We are not familiar with Android Studio

        - More and more people start to use iPhones as their sales numbers increase over years.
- IDE: Android Studio
- Environment: Macs or PCs

We thought about Using Android Studio to develop an Android mobile application, however, it lacks simplicity and adds complexity to our project. For one thing, when developing an iOS app, we have a development kit called SwiftUI, which allows us to see how our mobile app looks like with a live preview. However, the Android Studio does not have that functionality which allows us to see the live preview. In other words, every time we want to see how our app looks after changing our code, we have to restart a virtual Android machine, which requires a considerable amount of memory and eats up computer battery quickly.

Nonetheless, both of me and my partner have iPhones, thus allowing us to run our app on our phones natively and easily. On the other hand, both of us are not quite familiar with Android development, meaning we have to learn from scratch. That process might be time-consuming and takes a lot of effet. That, in a time-hungry assignment environment, is not the best option.

Since this is a simple check out page, all we used was front-end development without database or information storage. Thus, we did not use any back-end languages such as MySQL.

Option3: Flutter Development

Intro to Flutter Development:

It allows mobile app developers to make an app with only one single codebase, meaning that we only need to write one codebase, and the app will work on both ios and Android devices.

It consists of two main components:

1. An SDK: tools that help us develop the app. So we can compile our code into native machine code for IOS and Android.
2. A framework: UI elements such as buttons, text fields and etc.

Benefits:

1. Easy to learn but it is quite different from Java or Swift
2. Code size is relatively small for an application
3. Quick compilation allows maximal productivity
4. Documents are well-written

Disadvantages:

1. Flutter uses Dart which is a new programming language with many user-friendly features. But we are have never used Dart and it is as not wide-spread as Swift or Java. So we might not get as many resources for learning Dart.
2. UI kits offered by Flutter are limited and are not as decent as the tools offered by SwiftUI kits or Android Studio kits.

In conclusion, we needed to choose a safe, time-saving approach since we have a close deadline for finishing this project. We decided to choose ios app development by SwiftUI in Xcode, with which we are most familiar, and it is not that time-consuming.

We chose Swift UI tests as our tools to test the app. Since it simulates autonomously how we use and interact with the app and helps us debug with custom test cases.

Web Development Report

When it comes to web development, there are many options including HTML, CSS, JAVA, and PYTHON. We will discuss different aspects of each option and choose what best fits our needs so that we can develop easily and comfortably.

How to run:

1. Download VScode and Chrome
2. Clone our repository
3. Open CheckOutProject with Visual Studio code
4. Run index.html

PYTHON

It is an easy to use language which we are very familiar with. However we can only use it to build a server-side web application, it is hard to build a framework with Python. Thus we will have to use additional front-end programming languages.

Python is not used in a web browser. Browsers like Chrome and Firefox can not run python natively. That means we have to use Python along with other programming languages such as JavaScript.

There are some benefits to using Python. It is probably the language that we are most familiar with and good at. It also has good visualizations for presenting charts and plots using Matplotlib. Additionally, it has a wide range of libraries. Its frameworks also simplify the development process, such as Django and Pyramid.

However, we have to use another programming language such as JavaScript. So we might be better off using only HTML, CSS, and JavaScript.

Angular

Angular is a TypeScript-based front-end application development platform. It is a complete rebuild of Angular.JS. Angular combines declarative templates, dependency injection, end-to-end tooling, and integrated best practices to solve technical problems.

It is also a good tool for web development. First, it was made by Google, which provides enormous community support, since there are many talented front-end developers out there. Additionally, it has a unit testing setup built-in. By using the setup, developers can become good at acquiring real info regarding their web development projects. It is equipped with amazing User-interface tools. Web development has been made lightweight and easy by Angular. That is why many developers prefer Angular.

Nonetheless, Angular has a steep learning curve and takes a lot of time to master, since

it is quite layered and arranged hierarchically, scopes are hard to handle for beginners like us. Thus, the debugging process can probably be difficult and time-consuming for us. Furthermore, there is also a limitation of documentation available, which might cause learning problems especially in our case, where we have a close deadline.

C/C++

C/C++ is the language that most low-level systems such as operating systems, file systems, etc are written in.

It is a good option since this programming language has fast execution time compared to Python. Also, it is very portable and widely-spread. It allows us to gain a more substantial basis of programming language like Python. It is a procedure-heavy programming language, making debugging, testing, and maintaining the program easily.

C++ has a very wide application domain, such as making games, GUI applications, and math simulators.

Nonetheless, C/C++ has very complicated syntax and grammar, and might potentially be less efficient in an object-oriented environment compared to other OOP-based languages.

Also, we have to deal with memory allocation and garbage collection manually. We are not that familiar with this programming language as well. It also has a smaller standard library.

HTML, CSS, JavaScript

Those three languages work in conjunction, we use HTML to define what the web page contains, CSS is used to control the format of each individual component and formatting. JavaScript controls the behavior of the elements on the web page. We really like this option since we are very familiar with web development using those three languages and we have done projects in the past. It feels safe to use those languages.

Database

We first try to implement JSON as a light database to store our coupons along with HTML, CSS, JS. However, JSON is a lightweight text-based open standard data-interchange format which is used as a data model. In our project, coupons can be easily represented as a dictionary with the key as coupon code and the value as the way of discount, so we did not implement the database feature for this project due to the limited amount of data we have.

However, we did not stop here. We searched for the large data manipulation method and found that SQL may be applied in the future of our group project. SQL is a standard language for storing, manipulating and retrieving data in databases. It is powerful and well known, with many functions and concise syntax. SQL optimised for large numbers of table rows and can handle large numbers of transactions in a single query. Most importantly, it is fast for searching data. Based on these properties, SQL is the best choice at least for this course.

Conclusion

In conclusion, we choose HTML, CSS, and JavaScript which we are most familiar with and they are easy to use. We can also find a lot of tutorials on explaining how they work.