**Mobile Application Report A1**

**Saad Khan, Gabriel Bailey**

**Frontend and Backend**

The frontend and backend choices are basically the same since we are developing an app. We could follow different design patterns which would implement the backend in different ways but nonetheless for mobile applications these categories can be combined. Here they are in a table format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Question | Android (Java/Kotlin) | iOS (Swift) | Flutter | React Native (JS) |
| Familiar with the language? | ✓ | ~ | X | ~ |
| Is it known for simplicity? | ~ | ~ | ~ | ~ |
| Has supporting software (i.e. IDEs, etc.)? | ✓ | ✓ | ~ | ~ |
| Popularity1 | 8 | 8 | 5 | 3 |
| Current version is mature? | ✓ | ✓ | ~ | ~ |
| Fast/efficient?2 | ~ | ✓ | ~ | ✓ |
| Smooth UI Experience | ✓ | ✓ | ✓ | ✓ |
| Testing infrastructure (basic ones, here) | junit | swift | inbuilt | inbuilt |

I am most familiar with Android as I did an internship where I built android application. So from my perspective, android would be the easiest to work with. In terms of popularity, android studio has a lot of documentation in term of common to uncommon bugs, tests and so on. iOS is quite similar in this regard. However, I find flutter and react native to be lacking in term of documentation as not many people use them for production development of apps.

Frameworks and support for android and iOS is vastly superior than the other two options. All of them can provide a smooth user experience so in that regards there is no clear winner.

Backend technologies are the same. The backend on the mobile app is written on the same infrastructure. I find android and iOS can separate out the backend and frontend infrastructure for better testing capability, much more efficiently. The recently released MVVM model in Android using LiveData is particularly helpful in this regard. React Native and Flutter and more so focused on building apps on both platforms and can have much tighter coupling between classes.

**What I chose?**

I chose android to develop my application. Most people in the world own phones that run on android. iOS is mostly popular in North America. Android was also the language I was most familiar with. In terms of programming experience, android provides a much better experience than Flutter and React Native at least

**Continuous Integration and Continuous Deployment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Question | GitHub actions | Circle CI | Jenkins | GitLab CI |
| Familiar with the language? | ✓ | X | X | X |
| Support for android | ✓ | ✓ | ✓ | ✓ |
| Support for github | ✓ | ✓ | ✓ | ✓ |
| Smooth Experience | ✓ | ✓ | ✓ | ✓ |

There are multiple choice for implementing continuous integration and deployment within GitHub. The reality is this choice mostly came down to familiarity. I have some experience with GitHub actions in previous internships. GitHub actions is also very easy to set up even for android and provides all the capabilities we need such as autorunning test cases. GitHub actions was the obvious choice in this case

Database Integration

|  |  |  |  |
| --- | --- | --- | --- |
| Question | SQLite | Firebase | MongoDB |
| Familiar with the language? | ✓ | X | ✓ |
| Support for android | ✓ | ✓ | ✓ |
| Based on SQL | ✓ | ✓ | ✓ |
| Documentation and testing | ✓ | ✓ | ✓ |

These were the three options that were considered in term of storing data. SQLite has inbuilt integration into android studio. Firebase is a cloud-based system provided by Google. MongoDB is a popular database provider. I am quite familiar with SQLite and MongoDB. MongoDB requires additional work to set up for our app.

However, once the database requirement was removed by the professor, all these options were discarded. If I had to pick one for this assignment, I would have used SQLite but for a bigger assignment would have used Firebase or MongoDB.