**CEF440: INTERNET PROGRAMMING (J2EE) AND MOBILE PROGRAMMING**

**ANALYSIS OF MOBILE APPLICATION TECHNOLOGIES AND THE COST OF DEVELOPMENT.**

Presented by:

| **Name** | **Matricule** |
| --- | --- |
| Acha Rha’ah Achubang | FE20A001 |
| Amehmbo Ngewung Sonia | FE20A008 |
| Atem Randy Asong | FE20A014 |
| Tambe Salome Ntoh | FE20A109 |
| Tiokeng Samuel Edward | FE20A110 |

**1. What are the different types of Mobile Applications that exist and their differences**

There are three main types of mobile applications: native apps, web apps, and hybrid apps. Here are the differences between each type of application:

1. Native Apps:

Native apps are designed specifically for one platform, such as iOS, Android, or Windows Phone. They are built using platform-specific programming languages and development tools, such as Swift and Xcode for iOS, and Java and Android Studio for Android. Native apps are optimized for the best performance and user experience on their respective platforms.

2. Web Apps:

Web apps are websites that are designed to be accessed on mobile devices. They are built using web technologies such as HTML, CSS, and JavaScript, and are accessed through a mobile browser. Web apps can be designed to look and function like native apps, but they do not require installation on the device.

3. Hybrid Apps:

Hybrid apps are a combination of native and web apps. They are built using web technologies like HTML, CSS, and JavaScript, but are wrapped in a native app shell that allows them to be installed and accessed like a native app. Hybrid apps can access the device's native features, such as the camera and GPS, through plugins.

A comparison of the types listed above is given in the table below:

| **Category** | **Native Apps** | **Web Apps** | **Hybrid Apps** |
| --- | --- | --- | --- |
| Performance | Optimized for the platform, faster load times, smoother animations | Not optimized for specific platforms, slower load times, less smooth animations | Faster and smoother than web apps, but not as fast as native apps |
| User Experience | Takes full advantage of the platform's native features and design, more intuitive and seamless user experience | Can be designed to look and function like native apps, but may not have the same level of integration with the platform's native features | Can be designed to look and function like native apps, but may not have the same level of integration with the platform's native features as native apps |
| Development Cost | Requires specialized skills and tools, which can increase development costs | Requires web development skills and tools, which are often more widely available and less specialized than native app development tools, making it more cost-effective | Requires less specialized development skills than native apps, which can make them more cost-effective to develop |
| Time to Market | Developing native apps for multiple platforms can be time-consuming, which may delay the time to market | Can be developed and deployed quickly compared to native apps | Can be developed and deployed more quickly than native apps for multiple platforms, but may not have the same level of performance or user experience as native apps |

**2. Do a thorough review of programming languages used for mobile programming**

Java

Java is the most popular programming language for Android app development. It is an object-oriented language that is known for its robustness and reliability. Java is used to develop native Android apps, as well as cross-platform apps using frameworks like Xamarin. Java is also used for server-side development, making it a universal language for mobile and web development.

Features of Java:

* Portability: Java apps can run on any platform that supports the Java Virtual Machine (JVM), which makes them highly portable and easy to distribute.
* Object-oriented programming: Java is an object-oriented language that uses objects to represent real-world entities and their interactions. This makes it easier to write complex and modular code.
* Garbage Collection:: Java uses automatic memory management through garbage collection, which helps simplify memory management and reduces the risk of memory leaks and other memory-related issues.
* Rich API: Java has a rich API that makes it easy to access and use a wide range of libraries and tools for development.
* Cross-platform development: Java can be used for cross-platform development, which makes it easier to develop apps that run on multiple platforms.

Swift

Swift is the main programming language for iOS app development. It was developed by Apple and is designed to be fast, safe, and easy to learn. Swift is a modern language that is optimized for performance and is used to develop native iOS apps.

Features of Swift

* Safety: Swift is designed to be safe and prevent common programming errors, such as null pointer exceptions and array bounds violations.
* Speed: Swift is optimized for performance and is designed to be fast and efficient.
* Interoperability: Swift can be used with Objective-C, which is the older programming language used for iOS app development, making it easier to integrate with existing iOS apps.
* Easy to learn: Swift is designed to be easy to learn, with a clean and concise syntax that is similar to many other modern programming languages.
* Open-source: Swift is open-source, which means that it has a large community and many resources available for developers.

Kotlin

Kotlin is a relatively new programming language that has gained popularity for Android app development. It is a modern, open-source language that is designed to be more concise and expressive than Java. Kotlin is fully interoperable with Java, which means developers can easily integrate Kotlin code into existing Java projects.

Features of Kotlin:

* Concise syntax: Kotlin has a concise and expressive syntax that reduces the amount of boilerplate code needed to accomplish tasks.
* Interoperability: Kotlin is fully interoperable with Java, which means that Kotlin code can be easily integrated into existing Java projects.
* Null safety: Kotlin has built-in null safety features that prevent null pointer exceptions, which helps improve code reliability and reduces the risk of crashes.
* Extension functions: Kotlin allows developers to create extension functions, which can simplify code and make it easier to read and maintain.
* Functional programming: Kotlin has support for functional programming constructs, such as lambdas and higher-order functions, which can make it easier to write concise and expressive code.

C#

C# is a programming language developed by Microsoft and is used for developing native Windows Phone apps. It is also used for cross-platform mobile app development using the Xamarin framework. C# is an object-oriented language that is similar to Java in its syntax and structure. C# is also used for server-side development with the .NET Framework.

Features of C#:

* Versatility: C# can be used for both native Windows Phone app development and cross-platform mobile app development using the Xamarin framework.
* Object-oriented programming: C# is an object-oriented language that uses objects to represent real-world entities and their interactions. This makes it easier to write complex and modular code.
* Asynchronous programming: C# supports asynchronous programming, which allows developers to write code that can run concurrently without blocking the main thread. This makes it easier to build responsive and scalable applications.
* Garbage collection: C# uses garbage collection to automatically manage memory allocation and deallocation. This helps simplify memory management and reduces the risk of memory leaks and other memory-related issues.
* Interoperability: C# can be used with other programming languages and frameworks, such as Java and Xamarin, through the use of transpilers and other tools.

Dart

Dart is a programming language developed by Google and is used for building web, desktop, mobile, and server applications. Dart is an object-oriented language that has a syntax similar to Java and C++. It was designed to be easy to learn and use, with a focus on productivity and developer efficiency. Dart is used for developing both client-side and server-side applications, making it a language for web and mobile development.

Features of Dart:

* Asynchronous programming: Dart supports asynchronous programming, which allows developers to write code that can run concurrently without blocking the main thread. This makes it easier to build responsive and scalable applications.
* Garbage collection: Dart uses garbage collection to automatically manage memory allocation and deallocation. This helps simplify memory management and reduces the risk of memory leaks and other memory-related issues.
* Flutter framework: Dart is the primary language used for developing mobile apps using the Flutter framework. Flutter is a popular open-source UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase.
* Interoperability: Dart can be used with other programming languages and frameworks, such as JavaScript and React Native, through the use of transpilers and other tools.

JavaScript

JavaScript is one of the most popular programming languages used in mobile app development. It is a high-level, dynamic, and interpreted language that can be used for both client-side and server-side programming. In mobile programming, JavaScript is primarily used for building hybrid and web-based mobile applications.

Features of JavaScript

* Cross-platform compatibility: JavaScript can be used to build mobile apps that run on multiple platforms, including iOS, Android, and Windows. This is because JavaScript code can be run on any platform that has a compatible runtime environment.
* Lightweight: JavaScript is a lightweight and fast programming language, which makes it ideal for mobile app development.
* Flexibility: JavaScript is a versatile language that can be used for a wide range of applications, including mobile app development. It can be used to build everything from simple mobile apps to complex web applications.
* Rich libraries: JavaScript has a vast array of libraries and frameworks that make mobile app development easier and faster. These libraries provide pre-built solutions for common development problems, such as data storage, UI design, and communication with server-side APIs.
* Easy to learn: JavaScript has a relatively easy learning curve, even for beginners. It uses a syntax that is similar to the C programming language and has plenty of online resources available for learning.

**3. What is a Mobile App development framework?**

A mobile app development framework is a software framework that provides developers with a platform and set of tools to build mobile applications. It is a collection of pre-built components, modules, libraries, and APIs that can be used to create mobile apps quickly and efficiently.

A mobile app development framework typically includes features such as:

User interface design tools: These tools help developers design and customize the look and feel of their mobile app, including the layout, typography, and color scheme.

Cross-platform compatibility: Many frameworks support the development of apps that can run on multiple platforms, such as iOS and Android, using a single codebase.

Native app performance: Some frameworks provide tools and features that help developers optimize their apps for native performance, such as faster load times and smoother animations.

Backend integration: Many frameworks include APIs that allow developers to integrate their mobile app with a backend server or database.

Testing and debugging tools: Good frameworks provide debugging and testing tools to help developers identify bugs and issues during the development process.

Security features: Some frameworks provide built-in security features to help protect sensitive data and user information.

Some popular mobile app development frameworks include

* React Native
* Flutter
* Ionic
* Xamarin
* PhoneGap

**4. Review Mobile App development frameworks**

React Native

React Native is an open-source framework developed by Facebook that uses JavaScript and allows developers to build native mobile apps for iOS and Android platforms. It is known for its high performance and fast development speed, as well as its ability to reuse code across different platforms, which can save developers time and resources. React Native has a large and active community of developers and offers a wide range of plugins and libraries, making it easy to find support and resources. However, it can be complex to set up and requires a good understanding of JavaScript and the React library.

| Pros | Cons |
| --- | --- |
| * Allows apps to look like native apps * Ready-made components allow for faster development time. * Access to native functionalities. * Allow further optimization of native apps. * Hot reload to recompile the apps. | * Navigation is not seamless * Complex UI |

Flutter

Flutter is an open-source framework developed by Google that uses Dart programming language and allows developers to build native mobile apps for iOS and Android platforms. It is known for its fast development speed, hot reloading, and excellent performance, with apps built in Flutter often running at 60fps. Flutter also offers a wide range of plugins and widgets, making it easy to create custom UI designs. However, Flutter is relatively new, and its community of developers is smaller compared to other frameworks. Also, developers need to learn Dart programming language to use Flutter.

| Pros | Cons |
| --- | --- |
| * Complete development ecosystem. * Ability to create your apps or customize pre-existing ones. * Developed and supported by Google. * Fix bugs faster. | * Since the app size can be too large, developers might reduce the number of libraries and packages. * Does not expose many native APIs for use. It increases dependency on the ecosystem. |

Ionic

Ionic is an open-source framework that uses JavaScript, HTML, and CSS to build hybrid mobile apps for iOS and Android platforms. It is known for its fast development speed, ease of use, and low cost. Ionic offers a wide range of pre-built UI components and plugins, making it easy to create custom designs and add functionality to your app. It also has a large and active community of developers, which makes it easy to find resources and support. However, hybrid apps built with Ionic may not perform as well as native apps, and they may require more testing and optimization to ensure good performance.

| Pros | Cons |
| --- | --- |
| * Develop an app once. * Rich and robust applications. * Minimal maintenance. | * Performance is not as quick as it needs to be. * It requires a specific skill set to build complex apps. * Navigation can be notably complex. |

Xamarin

Xamarin is a cross-platform development framework that uses C# programming language and allows developers to build native mobile apps for iOS and Android platforms. It is known for its strong performance, native look and feel, and support for a wide range of APIs and third-party libraries. Xamarin also allows developers to reuse code across different platforms, which can save time and resources. However, Xamarin can be complex to set up and requires a good understanding of C# programming language. Also, Xamarin is not free, and its licensing fee can be expensive for smaller development teams.

| Pros | Cons |
| --- | --- |
| * Xamarin apps act almost like native apps. * Comprehensive cross-platform mobile app development frameworks. * Offers seamless experience. * It is free for small teams | * There is a delay in updating which may cause issues with the app. * Larger app size. * Heavy graphics. * Platform specific code |

**5. How do you collect and analyze requirements(functional and non-functional) for a mobile application that is to be developed**

Collecting and analyzing requirements for a mobile application that is to be developed is a crucial step in the mobile app development process. Here's a general process for collecting and analyzing requirements for a mobile app:

1. Define the purpose and scope of the app: Before collecting requirements, it's essential to define the purpose and scope of the app. This will help ensure that the project stays focused and aligned with business goals.

2. Identify stakeholders: Identify all stakeholders involved in the project, such as clients, end-users, developers, and project managers. This will help ensure that all perspectives are taken into account when gathering requirements.

3. Gather functional requirements: Functional requirements define what the app should do and how it should behave. This can include features, user interactions, and workflows. Gather requirements through interviews, surveys, and workshops with stakeholders.

4. Gather non-functional requirements: Non-functional requirements define how well the app should perform, such as performance, security, and accessibility. Gather requirements through interviews, surveys, and workshops, as well as research industry standards and best practices.

5. Prioritize requirements: Prioritize requirements based on their importance and urgency. This will help ensure that the most critical requirements are addressed first.

6. Create use cases and user stories: Use cases and user stories are essential tools for describing how users will interact with the app. These documents can help clarify requirements and ensure that the app meets user needs.

7. Validate requirements: Validate requirements with stakeholders to ensure that they are accurate, complete, and feasible. This can be done through reviews, demos, and user testing.

8. Document requirements: Document requirements in a clear and concise manner, using diagrams, tables, and other visual aids as needed.

9. Analyze requirements: Analyze requirements to identify any conflicts, gaps, or areas of uncertainty. This can help ensure that the app meets all requirements and is of high quality.

**6. How to estimate the mobile app development cost.**

Estimating the cost of mobile app development can be a complex process as the cost depends on several factors such as the app complexity, features, developer rates, and the platform(s) you're targeting. Here are a few steps to consider when estimating the cost of mobile app development:

1. Define your app requirements: Start by defining your app requirements, features, and functionalities. This will help in determining the complexity of the app and the amount of time and resources required to develop it.

2. Choose the right platform: The cost of mobile app development varies depending on the platform you choose (iOS, Android, or both). If you want to target both platforms, the cost will be higher than targeting a single platform.

3. Estimate the development time: The development time depends on the complexity of the app and the number of features you want to include. Be sure to factor in time for testing and debugging.

4. Include ongoing maintenance costs: Once the app is launched, ongoing maintenance and updates will need to be done so include the time for this as well.

5. Settle a billing method: Agree with the client on a suitable billing method, either per hour or per milestone or a total flat cost for the entire development period.

6. Calculate the total sum: Based on the billing method agreed upon with the client, the right calculations are made for the development/maintenance cost and these are then added to hosting cost and other development requirements costs such as the provision of a database and others.