Câu 1:

- Android, iOS, harmonyOS, kaiOS,…

- Android

+ Đặc điểm: Android là hệ điều hành mã nguồn mở được phát triển bởi Google và phổ biến trên nhiều dòng thiết bị từ các nhà sản xuất khác nhau (Samsung, Xiaomi, Oppo, v.v.).

+ Ưu điểm: - Tính linh hoạt cao và khả năng tùy biến rộng, kho ứng dụng phong phú, với Google Play Store và các kho ứng dụng của bên thứ ba, hỗ trợ nhiều loại thiết bị và nhiều mức giá khác nhau.

+ Khuyết điểm: Phân mảnh do có nhiều nhà sản xuất, dẫn đến sự khác biệt về trải nghiệm người dùng giữa các thiết bị, cập nhật phần mềm không đồng bộ, khiến một số thiết bị không được cập nhật lên phiên bản Android mới nhất.

- iOS

+ Đặc điểm: iOS là hệ điều hành độc quyền của Apple, chỉ được sử dụng trên các thiết bị của hãng như iPhone và iPad.

+ Ưu điểm: Hệ sinh thái đồng bộ và trải nghiệm người dùng mượt mà, cập nhật phần mềm nhanh chóng và đồng bộ trên tất cả các thiết bị được hỗ trợ, bảo mật cao và có nhiều tính năng bảo mật tiên tiến.

+ Khuyết điểm: Thiếu tính linh hoạt và khả năng tùy biến so với Android, giá thành thiết bị cao hơn, không phù hợp với tất cả đối tượng người dùng.

- HarmonyOS

+ Đặc điểm: Được phát triển bởi Huawei, HarmonyOS là hệ điều hành mã nguồn mở và đang dần thay thế Android trên các thiết bị Huawei tại một số thị trường.

+ Ưu điểm: Hỗ trợ đa nền tảng, có thể chạy trên cả điện thoại, TV, thiết bị IoT, v.v. Đồng bộ hóa nhanh chóng giữa các thiết bị trong hệ sinh thái Huawei.

+ Khuyết điểm: Thiếu ứng dụng so với Android và iOS, do không hỗ trợ Google Play Services, phạm vi phổ biến còn hạn chế và chủ yếu trên các thiết bị của Huawei.

- KaiOS

+ Đặc điểm: KaiOS là hệ điều hành dành cho điện thoại phổ thông (feature phone), kết hợp các tính năng của smartphone với giá thành thấp.

+ Ưu điểm: Tối ưu cho các thiết bị có cấu hình thấp, tiêu thụ ít năng lượng, hỗ trợ các ứng dụng phổ biến như Facebook, WhatsApp, Google Assistant trên thiết bị cơ bản.

+ Khuyết điểm: Khả năng tùy biến và chức năng bị hạn chế so với Android và iOS, không hỗ trợ nhiều ứng dụng và thiếu tính năng của một smartphone đầy đủ.

Câu 2:

Native Development (Android và iOS):

- Đặc điểm: Native Development sử dụng các ngôn ngữ và công cụ chính thức của từng nền tảng, cụ thể là Java/Kotlin cho Android và Swift/Objective-C cho iOS.
  - Ưu điểm:
    - Hiệu suất cao và trải nghiệm người dùng tốt, tận dụng tối đa phần cứng của thiết bị.
    - Truy cập đầy đủ vào các API và tính năng hệ điều hành.
  - Nhược điểm:
    - Cần viết mã riêng biệt cho từng nền tảng, tăng chi phí và thời gian phát triển.
    - Yêu cầu kiến thức chuyên sâu về từng hệ điều hành và cần đội ngũ phát triển riêng cho Android và iOS.

Flutter:

- Đặc điểm: Flutter là framework do Google phát triển, sử dụng ngôn ngữ Dart. Nó cho phép phát triển ứng dụng đa nền tảng với một codebase duy nhất.
  - Ưu điểm:

- Hiệu suất gần với ứng dụng native nhờ công cụ đồ họa riêng và không phụ thuộc vào các thành phần giao diện của hệ điều hành.
- Giao diện đẹp và tùy chỉnh dễ dàng với hệ thống widget phong phú.
- Có thể sử dụng một codebase để phát triển ứng dụng trên nhiều nền tảng (Android, iOS, Web, Desktop).
- Nhược điểm:
- Kích thước ứng dụng lớn hơn so với native.
- Dart là ngôn ngữ mới hơn và có cộng đồng người dùng nhỏ hơn Java/Kotlin hay Swift.

React Native:

- Đặc điểm: Được phát triển bởi Meta (Facebook), React Native sử dụng JavaScript và cho phép viết mã một lần để chạy trên cả Android và iOS.
- Ưu điểm:
- Codebase chung cho cả Android và iOS, giúp giảm thời gian và chi phí phát triển.
- Dễ học đối với các lập trình viên đã quen thuộc với JavaScript.
- Cộng đồng lớn và nhiều thư viện hỗ trợ.
- Nhược điểm:
- Hiệu suất thấp hơn so với native do cần sử dụng cầu nối (bridge) giữa JavaScript và mã native.
- Khó tối ưu hóa cho các ứng dụng yêu cầu hiệu suất cao hoặc đồ họa phức tạp.

Xamarin:

- Đặc điểm: Xamarin là nền tảng do Microsoft phát triển, sử dụng ngôn ngữ C# và .NET để phát triển ứng dụng cho cả Android và iOS.
- Ưu điểm:
- Một codebase cho cả hai nền tảng, giúp tiết kiệm thời gian và chi phí.
- Tích hợp tốt với các công cụ và dịch vụ của Microsoft.
- Hiệu suất tốt hơn so với các nền tảng dựa trên WebView.
- Nhược điểm:
- Kích thước ứng dụng lớn hơn và tiêu thụ tài nguyên cao hơn.
- Cộng đồng nhỏ hơn so với Flutter và React Native.

Ionic:

- Đặc điểm: Ionic là framework đa nền tảng dựa trên web, sử dụng HTML, CSS, và JavaScript để xây dựng ứng dụng di động.
- Ưu điểm:
- Codebase chung cho Android, iOS, và web, giúp giảm chi phí phát triển.
- Dễ học và có nhiều thư viện hỗ trợ.
- Nhược điểm:
- Hiệu suất thấp hơn do dựa trên WebView, không phù hợp cho các ứng dụng cần hiệu suất cao.
- Giới hạn trong việc truy cập các API hệ điều hành, cần các plugin bổ sung.

Câu 3:

Hiệu suất: Flutter có hiệu suất gần với ứng dụng native do sử dụng ngôn ngữ Dart và biên dịch trực tiếp sang mã máy, giúp ứng dụng chạy mượt mà hơn.

- Giao diện đẹp: Flutter có bộ công cụ widget phong phú và tùy chỉnh cao, giúp tạo ra các giao diện đẹp mắt và đồng nhất trên cả hai nền tảng Android và iOS.
- Phát triển nhanh: Tính năng Hot Reload cho phép lập trình viên thấy ngay kết quả của các thay đổi mà không cần phải biên dịch lại toàn bộ ứng dụng, giúp tiết kiệm thời gian phát triển.

- Hỗ trợ từ Google: Flutter được Google phát triển và hỗ trợ, có cộng đồng lớn, và các bản cập nhật thường xuyên.
- So sánh với React Native:
- Hiệu suất: Flutter có hiệu suất cao hơn do không phụ thuộc vào cầu nối giữa JavaScript và mã native. Trong khi đó, React Native sử dụng cầu nối JavaScript, điều này có thể làm giảm hiệu suất khi cần xử lý nhiều tác vụ.
- Giao diện: Flutter cung cấp các widget của riêng nó, giúp đồng nhất giao diện và dễ dàng tùy chỉnh hơn. React Native chủ yếu dựa vào các component native, điều này có thể làm giao diện không đồng nhất trên các nền tảng.
- So sánh với Xamarin:
- Cộng đồng: Flutter có cộng đồng phát triển mạnh mẽ hơn, dễ tiếp cận và học hỏi hơn so với Xamarin.
- Ngôn ngữ: Xamarin sử dụng C# và phù hợp với hệ sinh thái Microsoft, nhưng không phổ biến bằng Flutter trong lĩnh vực phát triển ứng dụng di động. Flutter sử dụng Dart, dễ học và tối ưu cho việc phát triển đa nền tảng.

Câu 4:

• Objective-C:

Lý do chọn: Là ngôn ngữ truyền thống của Apple, Objective-C đã được sử dụng để phát triển các ứng dụng iOS từ khi hệ điều hành này ra đời. Ngôn ngữ này có cú pháp phức tạp nhưng vẫn được sử dụng trong nhiều dự án cũ.

• Swift:

Lý do chọn: Swift là ngôn ngữ lập trình mới do Apple phát triển, được ra mắt năm 2014. Với cú pháp ngắn gọn, hiện đại, dễ đọc và dễ học hơn Objective-C, Swift đã trở thành ngôn ngữ chính thức và phổ biến nhất để phát triển ứng dụng iOS. Swift cũng có tính năng an toàn về bộ nhớ và hiệu suất cao, giúp cải thiện tốc độ và độ bảo mật của ứng dụng.

• Java:

Lý do chọn: Java đã tồn tại trong một thời gian dài và có một cộng đồng rộng lớn, nên có rất nhiều tài liệu, thư viện, và công cụ hỗ trợ, Java là ngôn ngữ chính được Google sử dụng cho Android API trong suốt nhiều năm, vì vậy Android SDK và các công cụ phát triển Android (Android Studio) đều hỗ trợ Java rất tốt, Java có thể chạy trên nhiều nền tảng khác nhau, giúp các ứng dụng Android dễ dàng tương tác với các dịch vụ và ứng dụng khác, Java được biết đến với khả năng bảo mật cao và ổn định, điều này rất quan trọng trong việc phát triển ứng dụng di động, Java sử dụng hệ thống Garbage Collection (GC), giúp quản lý bộ nhớ hiệu quả và giảm thiểu các lỗi liên quan đến bộ nhớ.

• Dart:

Lý do chọn: Dart biên dịch thành Native Code và có khả năng chạy rất nhanh, giúp các ứng dụng Flutter hoạt động mượt mà trên Android, Cộng đồng Flutter phát triển mạnh mẽ: Khi Flutter trở thành một framework phổ biến, Dart cũng được phát triển mạnh mẽ để hỗ trợ cộng đồng lập trình viên, hot reload và phát triển nhanh chóng: Dart hỗ trợ Hot reload trong Flutter, giúp lập trình viên thấy ngay lập tức kết quả khi thay đổi mã nguồn, tiết kiệm thời gian phát triển.

Câu 5:

• Swift:

Lý do chọn: Cú pháp hiện đại và dễ học: Swift có cú pháp ngắn gọn và dễ hiểu, giúp lập trình viên phát triển và bảo trì ứng dụng nhanh chóng.

Hiệu suất cao: Swift được biên dịch trực tiếp thành mã máy, mang lại hiệu suất tối ưu cho các ứng dụng iOS.

Hỗ trợ mạnh mẽ từ Apple: Được Apple duy trì và tích hợp chặt chẽ với Xcode và các công cụ phát triển khác.

Tính an toàn và bảo mật: Swift giúp giảm thiểu lỗi nhờ tính năng Optional và kiểm tra lỗi null.

• Python (dùng qua Kivy, BeeWare):

Lý do chọn: Cú pháp dễ học và phát triển nhanh: Python có cú pháp đơn giản, phù hợp cho những ứng dụng nhỏ hoặc thử nghiệm nhanh.

Đa nền tảng: Python hỗ trợ phát triển ứng dụng cho nhiều nền tảng, bao gồm cả iOS, thông qua các công cụ như Kivy hoặc BeeWare.

Cộng đồng lớn: Python có cộng đồng lớn và tài liệu phong phú, hỗ trợ tốt cho phát triển ứng dụng.

Câu 6:

• Thiếu ứng dụng: Windows Phone gặp khó khăn trong việc thu hút các nhà phát triển ứng dụng, dẫn đến việc thiếu các ứng dụng phổ biến trên nền tảng này. Người dùng Windows Phone không có nhiều lựa chọn về ứng dụng, làm giảm sức hấp dẫn của nền tảng này.

• Cạnh tranh từ iOS và Android: iOS và Android đã xây dựng được hệ sinh thái người dùng lớn trước khi Windows Phone ra mắt, khiến Microsoft gặp khó khăn trong việc thu hút người dùng chuyển sang hệ điều hành của mình.

• Thiếu sự nhất quán: Microsoft thay đổi chiến lược phát triển Windows Phone nhiều lần, gây ra sự không nhất quán và làm mất niềm tin của người dùng cũng như các nhà phát triển.

• Hỗ trợ kém từ các nhà sản xuất điện thoại: Nhiều nhà sản xuất lớn ưu tiên Android do tính phổ biến của nó, trong khi số lượng thiết bị chạy Windows Phone rất hạn chế.

Những thách thức này khiến Windows Phone không thể cạnh tranh với iOS và Android, dẫn đến sự sụt giảm thị phần và cuối cùng là bị ngừng phát triển.

Câu 7:

• HTML, CSS, JavaScript: Đây là các ngôn ngữ nền tảng để phát triển giao diện và tính năng cho các ứng dụng web. HTML dùng để xây dựng cấu trúc, CSS dùng để tạo kiểu, và JavaScript để xử lý tương tác.

• React và React Native: React là thư viện JavaScript của Facebook dùng để xây dựng giao diện web. React Native là một framework cho phép phát triển ứng dụng di động đa nền tảng dựa trên mã React, giúp tiết kiệm thời gian và chi phí.

• Flutter (Dart): Google phát triển Flutter để tạo ra các ứng dụng di động đa nền tảng. Ngoài việc phát triển ứng dụng native, Flutter cũng hỗ trợ tạo ứng dụng web, giúp lập trình viên sử dụng một mã nguồn để xây dựng ứng dụng cho nhiều nền tảng.

• Progressive Web Apps (PWA): PWA cho phép ứng dụng web có thể cài đặt và hoạt động như một ứng dụng native trên di động. Các công cụ như Workbox và Lighthouse giúp xây dựng và tối ưu hóa PWA.

Các công cụ này giúp lập trình viên xây dựng ứng dụng web đáp ứng tốt trên thiết bị di động, mang lại trải nghiệm gần giống ứng dụng native.

Câu 8:

• Nhu cầu cao về lập trình viên di động: Với sự phát triển không ngừng của các thiết bị di động và ứng dụng, nhu cầu về lập trình viên di động vẫn luôn ở mức cao. Các công ty tìm kiếm nhân lực có khả năng phát triển ứng dụng iOS, Android và đa nền tảng.

• Kỹ năng phổ biến và quan trọng:

• Kỹ năng lập trình ngôn ngữ native: Biết sử dụng Swift cho iOS và Kotlin cho Android là yêu cầu cơ bản cho lập trình viên di động.

• Phát triển đa nền tảng: Kiến thức về Flutter, React Native, hoặc Xamarin giúp lập trình viên có thể xây dựng ứng dụng cho nhiều nền tảng từ cùng một mã nguồn.

• Kiến thức về UI/UX: Kỹ năng thiết kế giao diện người dùng và hiểu biết về trải nghiệm người dùng (UI/UX) là quan trọng để tạo ra các ứng dụng thu hút và dễ sử dụng.

• API và Backend: Khả năng tích hợp API và làm việc với các hệ thống backend giúp lập trình viên tạo ra ứng dụng có chức năng phức tạp, đồng bộ hóa dữ liệu với máy chủ.

• Kỹ năng bảo mật: Hiểu biết về bảo mật giúp lập trình viên bảo vệ dữ liệu người dùng, một yếu tố quan trọng trong phát triển ứng dụng di động.

Các kỹ năng này giúp lập trình viên đáp ứng tốt nhu cầu phát triển ứng dụng di động hiện nay và nâng cao cơ hội nghề nghiệp trong lĩnh vực này.