2180609054

Nguyễn Công Khanh

Câu 1

Một số nền tảng phổ biến cho thiết bị di động thông minh hiện nay, cùng với ưu và khuyết điểm của từng nền tảng:

1. Android

Ưu điểm:

- Mở và linh hoạt: Android là mã nguồn mở, cho phép tùy chỉnh sâu và phát triển ứng dụng dễ dàng.
- Nhiều lựa chọn thiết bị: Có rất nhiều nhà sản xuất sử dụng Android, mang lại nhiều lựa chọn cho người tiêu dùng.
- Cửa hàng ứng dụng phong phú: Google Play Store có hàng triệu ứng dụng và trò chơi.

Khuyết điểm:

- Phân mảnh: Có nhiều phiên bản Android khác nhau, gây khó khăn cho việc phát triển ứng dụng đồng nhất.
- Bảo mật: Có thể bị tấn công nhiều hơn do sự đa dạng của các thiết bị và phiên bản.

2. iOS

Ưu điểm:

- Tính bảo mật cao: Apple có các biện pháp bảo mật nghiêm ngặt, giúp bảo vệ dữ liệu người dùng.
- Trải nghiệm người dùng tốt: Giao diện người dùng mượt mà và dễ sử dụng.
- Hệ sinh thái mạnh mẽ: Tích hợp tốt với các sản phẩm khác của Apple.

Khuyết điểm:

- Khép kín: Hạn chế tùy chỉnh và phát triển ứng dụng hơn so với Android.
- Giá cả cao: Các thiết bị iOS thường có giá cao hơn.

3. Windows Phone (không còn được phát triển)

Ưu điểm:

- Tích hợp với Windows: Dễ dàng đồng bộ hóa với các thiết bị Windows khác.

- Giao diện người dùng độc đáo: Thiết kế giao diện trực quan và hiện đại.

Khuyết điểm:

- Hạn chế ứng dụng: Số lượng ứng dụng và trò chơi ít hơn nhiều so với Android và iOS.
- Ngừng phát triển: Microsoft đã ngừng hỗ trợ và phát triển nền tảng này.

4. HarmonyOS

Ưu điểm:

- Tích hợp đa nền tảng: Hỗ trợ nhiều loại thiết bị, từ điện thoại đến IoT.
- Tính năng bảo mật: Được thiết kế với sự chú trọng đến bảo mật và quyền riêng tư.

Khuyết điểm:

- Thiếu ứng dụng: Chưa có kho ứng dụng phong phú như Android hay iOS.
- Chưa phổ biến: Mới mẻ và chưa được nhiều người dùng chấp nhận.

5. KaiOS

Ưu điểm:

- Tiết kiệm chi phí: Hỗ trợ các thiết bị giá rẻ, mang lại khả năng truy cập Internet cho người dùng không có điều kiện.
- Giao diện đơn giản: Dễ sử dụng cho người mới bắt đầu.

Khuyết điểm:

- Hạn chế ứng dụng: Số lượng ứng dụng hạn chế và không đa dạng như Android hay iOS.
- Thiếu tính năng: Không hỗ trợ nhiều tính năng cao cấp của smartphone hiện đại.

Mỗi nền tảng có những ưu và nhược điểm riêng, tùy thuộc vào nhu cầu và sở thích của người dùng để lựa chọn nền tảng phù hợp nhất.

Câu 2

Các nền tảng phát triển ứng dụng di động phổ biến

1. Native Development

- Nền tảng:

    - Android: Java, Kotlin
    - iOS: Swift, Objective-C
- Ưu điểm:
    - Hiệu suất cao: Ứng dụng chạy mượt mà, tối ưu hóa cho từng nền tảng.
    - Trải nghiệm người dùng tốt nhất: Giao diện và tương tác phù hợp với thiết kế của hệ điều hành.
    - Truy cập đầy đủ vào API: Có thể sử dụng tất cả các tính năng và API của hệ điều hành.
- Khuyết điểm:
    - Chi phí cao: Phát triển và bảo trì ứng dụng cho từng nền tảng riêng biệt.
    - Thời gian phát triển lâu: Cần viết mã cho cả Android và iOS.

2. Cross-Platform Development

- Nền tảng: React Native, Flutter, Xamarin
- Ưu điểm:
    - Viết mã một lần, triển khai trên nhiều nền tảng: Tiết kiệm thời gian và chi phí.
    - Tính khả dụng cao: Dễ dàng bảo trì và cập nhật ứng dụng.
    - Giao diện người dùng gần giống native: React Native và Flutter cho phép tạo giao diện gần giống với native.
- Khuyết điểm:
    - Hiệu suất không tối ưu như native: Có thể gặp vấn đề hiệu suất với các ứng dụng nặng.
    - Hạn chế khi truy cập một số API hệ thống: Đôi khi không thể sử dụng tất cả tính năng của hệ điều hành.

3. Hybrid Development

- Nền tảng: Ionic, Cordova
- Ưu điểm:
    - Dễ dàng phát triển: Sử dụng các công nghệ web (HTML, CSS, JavaScript).

    - Chi phí phát triển thấp: Phát triển nhanh chóng với mã nguồn chung cho nhiều nền tảng.
- Khuyết điểm:
    - Hiệu suất thường kém: Chậm hơn và không mượt mà như ứng dụng native.
    - Giao diện không hoàn hảo: Có thể không đạt mức chất lượng cao nhất về trải nghiệm người dùng.

4. Progressive Web Apps (PWAs)

- Nền tảng: Sử dụng HTML, CSS, JavaScript
- Ưu điểm:
    - Chạy trên bất kỳ trình duyệt nào: Không cần cài đặt từ cửa hàng ứng dụng.
    - Dễ dàng cập nhật: Cập nhật tự động mà không cần người dùng can thiệp.
- Khuyết điểm:
    - Hạn chế tính năng: Không thể truy cập đầy đủ API của hệ điều hành.
    - Trải nghiệm người dùng không hoàn toàn giống ứng dụng di động: Có thể thiếu một số tính năng của ứng dụng gốc.

5. Xamarin

- Nền tảng: Sử dụng C#
- Ưu điểm:
    - Cùng một mã nguồn cho cả Android và iOS: Giúp tiết kiệm thời gian và công sức.
    - Truy cập vào API Native: Cho phép sử dụng các API của hệ điều hành gốc.
    - Tính năng Xamarin.Forms: Hỗ trợ phát triển giao diện người dùng cho cả hai nền tảng từ một mã nguồn chung.
- Khuyết điểm:
    - Kích thước ứng dụng lớn: Ứng dụng được phát triển bằng Xamarin thường có kích thước lớn hơn.

    - Hiệu suất không hoàn hảo: Mặc dù gần với native, nhưng vẫn có thể gặp vấn đề về hiệu suất trong một số trường hợp.

- So sánh sự khác biệt chính

Tiêu chí	Native Development	Cross-Platform Development	Hybrid Development	Progressive Web Apps (PWAs)	Xamarin
Ngôn ngữ	Java/Kotlin (Android), Swift/Objective-C (iOS)	JavaScript, Dart	HTML, CSS, JavaScript	HTML, CSS, JavaScript	C#
Hiệu suất	Tối ưu nhất	Tốt nhưng không bằng native	Thấp hơn native	Thấp hơn native	Gần native nhưng không hoàn hảo
Trải nghiệm người dùng	Xuất sắc	Tốt	Khá	Trung bình	Tốt
Chi phí phát triển	Cao	Thấp hơn native	Thấp	Thấp	Thấp hơn native
Khả năng truy cập API	Hoàn toàn	Hạn chế	Hạn chế	Hạn chế	Gần như hoàn toàn
Kích thước ứng dụng	Nhỏ hơn	Trung bình	Lớn hơn	Nhỏ hơn	Thường lớn

Câu 3

Flutter đã trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng đa nền tảng nhờ vào nhiều yếu tố nổi bật. Dưới đây là một số điểm chính giúp Flutter thu hút sự chú ý, cùng với sự so sánh với React Native và Xamarin.

1. Ưu điểm của Flutter

- Hiệu suất cao: Flutter biên dịch trực tiếp sang mã máy (native code), giúp tăng tốc độ thực thi và mang lại hiệu suất gần như ứng dụng native.

- Giao diện người dùng đẹp và tùy chỉnh: Flutter cho phép xây dựng giao diện người dùng tùy chỉnh với các widget phong phú, dễ dàng tạo ra các giao diện hấp dẫn và linh hoạt.
- Hot Reload: Tính năng này cho phép lập trình viên xem ngay sự thay đổi trong mã mà không cần khởi động lại ứng dụng, giúp tăng tốc quá trình phát triển.
- Mã nguồn chung: Flutter cho phép viết mã cho cả Android và iOS từ một mã nguồn duy nhất, tiết kiệm thời gian và công sức.

Câu 4

một số ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng trên Android, cùng với lý do tại sao chúng lại được chọn:

1. Java

- Giới thiệu: Java là ngôn ngữ lập trình chính được sử dụng trong phát triển Android từ những ngày đầu.
- Lý do chọn:
    - Tính ổn định: Java đã tồn tại lâu dài và được kiểm chứng qua thời gian, mang lại sự ổn định cho các ứng dụng.
    - Hệ sinh thái phong phú: Có nhiều thư viện, framework và công cụ hỗ trợ cho việc phát triển ứng dụng.
    - Khả năng tương thích: Java có thể chạy trên nhiều hệ điều hành và nền tảng khác nhau nhờ vào JVM (Java Virtual Machine).

2. Kotlin

- Giới thiệu: Kotlin được Google công nhận là ngôn ngữ chính thức cho phát triển Android vào năm 2017.
- Lý do chọn:
    - Tính hiện đại: Kotlin cung cấp cú pháp ngắn gọn và dễ đọc hơn so với Java, giúp giảm khối lượng mã.
    - Tính an toàn: Kotlin giúp giảm thiểu lỗi NullPointerException nhờ vào hệ thống kiểu an toàn (null safety).
    - Tích hợp dễ dàng: Kotlin hoàn toàn tương thích với Java, cho phép lập trình viên dễ dàng tích hợp vào các dự án hiện có.

3. C++

- Giới thiệu: C++ được sử dụng chủ yếu cho các ứng dụng yêu cầu hiệu suất cao hoặc khi cần tương tác với mã native.
- Lý do chọn:
    - Hiệu suất cao: C++ cho phép tối ưu hóa mã và quản lý bộ nhớ hiệu quả, thường được sử dụng trong game và ứng dụng đồ họa.
    - Tính linh hoạt: Có thể sử dụng với Android Native Development Kit (NDK) để phát triển các phần của ứng dụng bằng mã native.

4. Dart

- Giới thiệu: Dart là ngôn ngữ được sử dụng với Flutter, một framework phát triển ứng dụng đa nền tảng.
- Lý do chọn:
    - Hiệu suất cao: Dart biên dịch thành mã máy, giúp cải thiện hiệu suất ứng dụng.
    - Hot Reload: Tính năng này giúp lập trình viên xem ngay sự thay đổi mà không cần khởi động lại ứng dụng, tăng tốc quá trình phát triển.
    - Tính năng lập trình hàm: Dart hỗ trợ lập trình hàm, giúp viết mã dễ dàng và linh hoạt hơn.

5. Python

- Giới thiệu: Mặc dù không phải là ngôn ngữ chính thức để phát triển ứng dụng Android, Python có thể được sử dụng thông qua các framework như Kivy.
- Lý do chọn:
    - Dễ học: Python có cú pháp rõ ràng, dễ hiểu, giúp lập trình viên mới dễ dàng tiếp cận.
    - Tính linh hoạt: Có thể được sử dụng cho nhiều mục đích khác nhau, từ phát triển ứng dụng đến khoa học dữ liệu.

Câu 5

Các ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng trên iOS:

1. Swift

- Giới thiệu: Swift là ngôn ngữ lập trình hiện đại được Apple giới thiệu vào năm 2014, và là ngôn ngữ chính thức cho phát triển ứng dụng iOS.
- Lý do chọn:

- Tính hiện đại: Cú pháp của Swift đơn giản và dễ đọc, giúp lập trình viên viết mã nhanh chóng và hiệu quả.
- Tính an toàn: Swift có hệ thống kiểu an toàn, giúp giảm thiểu lỗi và tăng cường bảo mật cho ứng dụng.
- Hiệu suất cao: Swift được tối ưu hóa để chạy nhanh và hiệu quả, gần với hiệu suất của mã C.

2. Objective-C

- Giới thiệu: Objective-C là ngôn ngữ chính được sử dụng trước khi Swift ra đời, và vẫn được hỗ trợ rộng rãi.
- Lý do chọn:
  - Tính tương thích: Nhiều ứng dụng cũ viết bằng Objective-C vẫn đang hoạt động, và Swift có thể tích hợp với mã Objective-C.
  - Cộng đồng và tài liệu phong phú: Có nhiều tài liệu và thư viện hỗ trợ, giúp lập trình viên dễ dàng tìm kiếm giải pháp.

3. C++

- Giới thiệu: C++ có thể được sử dụng trong phát triển iOS, đặc biệt cho các phần yêu cầu hiệu suất cao hoặc xử lý đồ họa.
- Lý do chọn:
  - Hiệu suất cao: C++ cho phép tối ưu hóa mã và quản lý bộ nhớ hiệu quả, thích hợp cho các ứng dụng game và đồ họa phức tạp.
  - Tính linh hoạt: Có thể sử dụng cùng với Objective-C hoặc Swift để phát triển các phần của ứng dụng.

4. Python

- Giới thiệu: Mặc dù không phải là ngôn ngữ chính thức cho phát triển iOS, Python có thể được sử dụng thông qua các framework như Kivy hoặc BeeWare.
- Lý do chọn:
  - Dễ học: Python có cú pháp rõ ràng, dễ hiểu, giúp lập trình viên mới dễ dàng tiếp cận.
  - Tính linh hoạt: Có thể sử dụng cho nhiều mục đích khác nhau, từ phát triển ứng dụng đến khoa học dữ liệu.

5. JavaScript

- Giới thiệu: JavaScript có thể được sử dụng để phát triển ứng dụng iOS thông qua các framework như React Native hoặc Cordova.
- Lý do chọn:
    - Phát triển đa nền tảng: Với JavaScript, lập trình viên có thể phát triển ứng dụng cho cả iOS và Android từ một mã nguồn chung.
    - Cộng đồng lớn: JavaScript có một cộng đồng phát triển mạnh mẽ, với nhiều thư viện và công cụ hỗ trợ.

Câu 6

Windows Phone đã phải đối mặt với nhiều thách thức trong suốt thời gian tồn tại của nó, dẫn đến sự sụt giảm thị phần và cuối cùng là ngừng phát triển. Dưới đây là một số thách thức chính và nguyên nhân dẫn đến sự sụt giảm thị phần của Windows Phone:

1. Thiếu ứng dụng

- Thách thức: Một trong những vấn đề lớn nhất mà Windows Phone gặp phải là thiếu hụt ứng dụng. Nhiều nhà phát triển không mặn mà phát triển ứng dụng cho nền tảng này vì thị phần thấp.
- Nguyên nhân: Việc thiếu ứng dụng phổ biến (như Facebook, Instagram, và nhiều ứng dụng ngân hàng) đã khiến người dùng không muốn chuyển sang Windows Phone.

2. Thị phần nhỏ

- Thách thức: Windows Phone không thể chiếm được thị phần đáng kể trên thị trường di động, thường đứng sau iOS và Android.
- Nguyên nhân: Sự chậm trễ trong việc phát triển và tiếp thị, cùng với sự cạnh tranh mạnh mẽ từ các thiết bị Android và iPhone, đã khiến Windows Phone khó có thể mở rộng thị phần.

3. Thiết kế và trải nghiệm người dùng

- Thách thức: Mặc dù Windows Phone có giao diện người dùng khác biệt và hấp dẫn, nhưng nhiều người dùng không quen thuộc với cách hoạt động của nó.
- Nguyên nhân: Sự khác biệt này có thể làm cho người dùng mới gặp khó khăn trong việc làm quen, dẫn đến việc họ lựa chọn các nền tảng phổ biến hơn.

4. Thiếu sự hỗ trợ từ các nhà sản xuất

- Thách thức: Windows Phone không nhận được sự hỗ trợ mạnh mẽ từ các nhà sản xuất phần cứng như Nokia, HTC, và Samsung.

- Nguyên nhân: Sau khi Microsoft mua lại Nokia, sự hỗ trợ từ các đối tác khác giảm đi, làm cho danh mục sản phẩm trở nên hạn chế hơn.

5. Sự chuyển đổi chiến lược của Microsoft

- Thách thức: Microsoft đã có những thay đổi chiến lược liên tục, từ việc tập trung vào phần cứng đến việc trở lại với phần mềm.
- Nguyên nhân: Sự chuyển đổi này không nhất quán đã tạo ra sự hoài nghi về tương lai của Windows Phone trong tâm trí người tiêu dùng và nhà phát triển.

6. Cạnh tranh từ Android và iOS

- Thách thức: Android và iOS đã phát triển mạnh mẽ, cung cấp nhiều tính năng và ứng dụng, cùng với hệ sinh thái phong phú.
- Nguyên nhân: Sự phát triển này đã thu hút người dùng và nhà phát triển, làm cho Windows Phone trở nên kém hấp dẫn hơn.

7. Ngừng phát triển và hỗ trợ

- Thách thức: Cuối cùng, Microsoft đã quyết định ngừng phát triển Windows Phone vào năm 2017.
- Nguyên nhân: Quyết định này xuất phát từ việc nhận thấy thị phần không thể cạnh tranh với Android và iOS, cùng với việc không có đủ ứng dụng và sự hỗ trợ từ cộng đồng.

Câu 7

1. Ngôn ngữ lập trình

a. HTML (HyperText Markup Language)

- Giới thiệu: Ngôn ngữ đánh dấu cơ bản để tạo cấu trúc cho trang web.
- Vai trò: Xác định các phần tử trên trang như tiêu đề, đoạn văn, hình ảnh, và liên kết.

b. CSS (Cascading Style Sheets)

- Giới thiệu: Ngôn ngữ dùng để định kiểu cho các phần tử HTML.
- Vai trò: Quản lý giao diện và bố cục của trang web, bao gồm màu sắc, phông chữ, và cách bố trí trên các kích thước màn hình khác nhau.

c. JavaScript

- Giới thiệu: Ngôn ngữ lập trình phổ biến dùng để tạo ra các tính năng tương tác cho trang web.
- Vai trò: Thêm chức năng động cho trang web, xử lý sự kiện, và tương tác với người dùng.

d. TypeScript

- Giới thiệu: Biến thể của JavaScript với hỗ trợ kiểu tĩnh.
- Vai trò: Giúp phát triển ứng dụng web lớn và phức tạp hơn dễ dàng hơn, cải thiện khả năng bảo trì và đọc mã.

2. Frameworks và thư viện

a. React

- Giới thiệu: Thư viện JavaScript được phát triển bởi Facebook.
- Vai trò: Dùng để xây dựng giao diện người dùng, đặc biệt cho ứng dụng đơn trang (SPA). Hỗ trợ phát triển nhanh chóng với tính năng tái sử dụng component.

b. Vue.js

- Giới thiệu: Framework JavaScript nhẹ và linh hoạt.
- Vai trò: Dễ học và tích hợp, thích hợp cho việc xây dựng giao diện người dùng tương tác.

c. Angular

- Giới thiệu: Framework JavaScript toàn diện do Google phát triển.
- Vai trò: Cung cấp một giải pháp đầy đủ cho phát triển ứng dụng web, hỗ trợ xây dựng SPA với cấu trúc rõ ràng.

d. Bootstrap

- Giới thiệu: Framework CSS phổ biến để phát triển giao diện web.
- Vai trò: Cung cấp các thành phần giao diện người dùng có thể tái sử dụng, giúp thiết kế responsive dễ dàng hơn.

3. Công cụ phát triển

a. Visual Studio Code

- Giới thiệu: Trình soạn thảo mã nguồn miễn phí và mạnh mẽ.

- Vai trò: Hỗ trợ nhiều ngôn ngữ lập trình và có nhiều tiện ích mở rộng giúp nâng cao năng suất.

b. Chrome DevTools

- Giới thiệu: Bộ công cụ phát triển tích hợp trong trình duyệt Google Chrome.
- Vai trò: Giúp kiểm tra và gỡ lỗi mã HTML, CSS và JavaScript, và kiểm tra hiệu suất trang web.

c. Figma/Adobe XD

- Giới thiệu: Công cụ thiết kế giao diện người dùng.
- Vai trò: Dùng để tạo prototyp và mockup cho ứng dụng web trước khi phát triển.

d. Git

- Giới thiệu: Hệ thống quản lý phiên bản.
- Vai trò: Giúp theo dõi sự thay đổi trong mã nguồn và hợp tác với các lập trình viên khác.

4. Responsive Web Design

- Nguyên tắc: Thiết kế giao diện web đáp ứng được với mọi kích thước màn hình.
- Công cụ hỗ trợ:
    - Media Queries: CSS cho phép thay đổi kiểu dáng dựa trên kích thước màn hình.
    - Flexbox/Grid: Các kỹ thuật CSS cho bố cục linh hoạt.

Câu 8

Nhu cầu về nguồn nhân lực lập trình viên trong lĩnh vực phát triển ứng dụng di động đang gia tăng nhanh .Các kỹ năng được ưa chuộng hiện nay bao gồm ngôn ngữ lập trình như Java, Kotlin cho Android, Swift cho iOS, và Dart cho Flutter, các framework như React Native và Xamarin. Kiến thức về thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX) khả năng tương tác với API.