# **CHƯƠNG I: TỔNG QUAN VỀ CÔNG NGHỆ VÀ CÔNG CỤ PHÁT TRIỂN ỨNG DỤNG ANDROID**

## **1.1. Giới thiệu về Android Studio**

### **1.1.1. Định nghĩa và vai trò**

*1.1.1.1. Android Studio*

Android Studio là môi trường phát triển tích hợp (IDE) chính thức được Google phát triển và duy trì cho việc xây dựng ứng dụng Android. Android Studio được xây dựng dựa trên IntelliJ IDEA, một IDE mạnh mẽ cho Java, với các cải tiến và tích hợp thêm các công cụ cụ thể dành cho phát triển ứng dụng Android.

* **Môi trường phát triển tích hợp (IDE)**: Android Studio cung cấp một nền tảng tích hợp hoàn chỉnh với các công cụ hỗ trợ mọi bước của quy trình phát triển ứng dụng, từ viết mã đến kiểm thử và triển khai.
* **Đặc điểm chính**: Giao diện người dùng trực quan, tích hợp các công cụ mạnh mẽ cho việc thiết kế giao diện người dùng (UI), gỡ lỗi, và tối ưu hóa hiệu suất ứng dụng.

*1.1.1.2. Tính năng nổi bật*

* **Trình biên tập mã nguồn**: Android Studio cung cấp một trình biên tập mã nguồn mạnh mẽ với tính năng hoàn thành mã tự động (code completion), kiểm tra cú pháp, và hỗ trợ các ngôn ngữ lập trình như Java, Kotlin, và XML.
* **Công cụ gỡ lỗi**: Bao gồm một loạt công cụ gỡ lỗi, như trình gỡ lỗi đồ họa, phân tích hiệu suất, và công cụ phân tích bộ nhớ, giúp phát hiện và sửa lỗi trong mã nguồn ứng dụng.
* **Thiết kế giao diện**: Công cụ Layout Editor cho phép tạo và chỉnh sửa giao diện người dùng bằng cách kéo và thả các thành phần, cùng với tính năng Constraint Layout giúp xây dựng giao diện tương thích với nhiều kích thước màn hình khác nhau.
* **Tích hợp hệ thống kiểm thử**: Android Studio hỗ trợ kiểm thử đơn vị (Unit Testing) và kiểm thử giao diện người dùng (UI Testing) thông qua các công cụ như JUnit và Espresso.

### **1.1.2. Cài đặt và cấu hình**

*1.1.2.1. Yêu cầu hệ thống:*

 **Yêu cầu phần cứng**: Android Studio yêu cầu ít nhất 8 GB RAM (khuyến nghị 16 GB) và 4 GB dung lượng ổ đĩa trống. Tùy vào kích thước dự án và số lượng công cụ hỗ trợ cài đặt, dung lượng ổ đĩa cần thiết có thể lớn hơn.

 **Yêu cầu phần mềm**: Android Studio hỗ trợ các hệ điều hành Windows 10 trở lên, macOS 10.14 hoặc mới hơn, và các phiên bản phổ biến của Linux. JDK (Java Development Kit) 11 hoặc mới hơn là cần thiết.

*1.1.2.2. Quá trình cài đặt:*

 **Trên Windows**:

1. Tải xuống tệp cài đặt Android Studio từ trang web chính thức của Google.
2. Chạy tệp .exe để bắt đầu quá trình cài đặt.
3. Làm theo hướng dẫn trên màn hình để hoàn tất cài đặt, bao gồm việc chọn các thành phần bổ sung như Android SDK và Android Virtual Device (AVD).

 **Trên macOS**:

1. Tải xuống tệp .dmg từ trang web của Android Studio.
2. Mở tệp .dmg và kéo Android Studio vào thư mục Applications.
3. Mở Android Studio từ thư mục Applications và làm theo hướng dẫn để hoàn tất cấu hình ban đầu.

 **Trên Linux**:

1. Tải xuống tệp .zip từ trang web của Android Studio.
2. Giải nén tệp .zip vào thư mục mong muốn.
3. Mở terminal, chuyển đến thư mục Android Studio và chạy tệp studio.sh để bắt đầu cài đặt.

*1.1.2.3. Cấu hình ban đầu:*

 **Thiết lập dự án mới**: Khi lần đầu tiên khởi động Android Studio, chọn “Start a new Android Studio project.” Nhập tên dự án, chọn thư mục lưu trữ, và chọn mẫu ứng dụng hoặc template phù hợp.

 **Cấu hình các tùy chọn phát triển**: Cấu hình SDK Android, chọn các phiên bản Android cần thiết, và cài đặt các công cụ hỗ trợ như Android Emulator và Build-Tools để phục vụ cho quá trình phát triển.

## **1.2. Ngôn ngữ lập trình cho Android**

### **1.2.1. Java vs Kotlin**

*1.2.1.1. Java: Ngôn ngữ lập trình truyền thống cho Android*

 **Ưu điểm**: Java là ngôn ngữ lập trình lâu đời và phổ biến với cộng đồng lớn. Được hỗ trợ mạnh mẽ bởi tài liệu và thư viện phong phú. Có nhiều công cụ và nguồn tài nguyên học tập cho Java.

 **Hạn chế**: Mã nguồn thường dài dòng và có thể khó quản lý. Quản lý bộ nhớ phức tạp và thiếu hỗ trợ cho một số tính năng lập trình hiện đại như lambda expressions và extension functions.

*1.2.1.2. Kotlin*

 **Ưu điểm**: Kotlin cung cấp cú pháp ngắn gọn hơn, an toàn hơn với các vấn đề liên quan đến null, và tích hợp tốt với các thư viện Java. Các tính năng như data classes, lambda expressions, và extension functions giúp lập trình hiệu quả hơn.

 **Lý do chọn Kotlin**: Kotlin được Google chính thức hỗ trợ và khuyến khích cho phát triển ứng dụng Android. Kotlin giúp giảm thiểu lỗi và tăng năng suất lập trình nhờ cú pháp đơn giản và tính năng hiện đại.

### **1.2.2. Tính năng và lợi ích của Kotlin**

*1.2.2.1. Cú pháp ngắn gọn*

**Giảm thiểu mã nguồn**: Kotlin cung cấp cú pháp ngắn gọn giúp giảm lượng mã cần viết, làm cho mã nguồn dễ đọc hơn và bảo trì dễ dàng hơn. Ví dụ, các data classes tự động sinh ra các phương thức như toString(), equals(), và hashCode() mà không cần viết thêm mã.

*2.2.2.2. Tính tương thích với Java*

 **Hỗ trợ tương thích**: Kotlin hoàn toàn tương thích với các thư viện và mã nguồn Java hiện có. Điều này cho phép các dự án cũ có thể tích hợp Kotlin mà không gặp vấn đề tương thích.

 **Kết hợp với Java**: Các lớp viết bằng Java có thể được sử dụng trực tiếp trong dự án Kotlin, và ngược lại.

*1.2.2.3. An toàn kiểu*

**Kiểm soát null**: Kotlin cung cấp hệ thống loại an toàn với null (null safety), giúp giảm thiểu lỗi liên quan đến null pointer exceptions bằng cách yêu cầu các biến phải được khởi tạo và kiểm tra.

## **1.3. Cơ sở dữ liệu trong Android**

### **1.3.1. SQLite**

*1.3.1.1. Định nghĩa và vai trò:*

 **Giới thiệu SQLite**: SQLite là một cơ sở dữ liệu nhẹ, tích hợp sẵn trong Android, không yêu cầu một server cơ sở dữ liệu riêng biệt. SQLite cho phép lưu trữ dữ liệu ứng dụng một cách hiệu quả và nhanh chóng.

 **Vai trò**: SQLite được sử dụng để lưu trữ các dữ liệu quan trọng như thông tin người dùng, dữ liệu giao dịch, và cấu hình ứng dụng. Nó cho phép thực hiện các thao tác cơ bản như thêm, sửa, xóa, và truy vấn dữ liệu.

*1.3.1.2. Cấu trúc cơ sở dữ liệu:*

**Thiết kế cơ sở dữ liệu**: Thiết kế các bảng cơ sở dữ liệu bao gồm việc xác định các trường dữ liệu, kiểu dữ liệu, và các chỉ số để tối ưu hóa hiệu suất truy vấn. Tạo các mối quan hệ giữa các bảng nếu cần thiết.

*1.3.1.3. Quản lý cơ sở dữ liệu:*

**Các hoạt động cơ bản**: Thực hiện các thao tác cơ bản như thêm, sửa, xóa và truy vấn dữ liệu sử dụng các lệnh SQL cơ bản.

Sử dụng các lệnh SQL để thực hiện các thao tác như thêm (INSERT), sửa (UPDATE), xóa (DELETE), và truy vấn (SELECT) dữ liệu. Sử dụng lớp SQLiteOpenHelper để quản lý vòng đời của cơ sở dữ liệu.

**Sử dụng SQLiteOpenHelper**: Để quản lý cơ sở dữ liệu và xử lý các phiên bản cơ sở dữ liệu.

### **1.3.2. Room Database**

*1.3.2.1. Giới thiệu:*

**Room Database**: Room là một thư viện của Android giúp làm việc với cơ sở dữ liệu SQLite dễ dàng hơn bằng cách cung cấp một lớp trừu tượng và các tính năng hữu ích như hỗ trợ các truy vấn SQL và ánh xạ dữ liệu tự động.

*1.3.2.2. Tính năng chính:*

 **Hỗ trợ truy vấn SQL**: Room hỗ trợ viết các truy vấn SQL trực tiếp và ánh xạ kết quả vào các đối tượng Java hoặc Kotlin, giúp đơn giản hóa việc tương tác với cơ sở dữ liệu.

 **Các lớp Entity, DAO, Database**:

* **Entity**: Định nghĩa các bảng cơ sở dữ liệu dưới dạng các lớp dữ liệu với các thuộc tính đại diện cho các cột trong bảng.
* **DAO (Data Access Object)**: Cung cấp các phương thức để thực hiện các thao tác trên cơ sở dữ liệu như truy vấn, thêm, sửa, và xóa dữ liệu.
* **Database**: Cung cấp một lớp cơ sở dữ liệu chính, quản lý vòng đời của cơ sở dữ liệu và cung cấp các đối tượng DAO.

*1.3.2.3. Cài đặt và sử dụng:*

**Tích hợp Room**: Thêm các phụ thuộc Room vào tệp build.gradle của dự án, tạo các lớp Entity, DAO, và Database, và sử dụng Room để thực hiện các thao tác cơ sở dữ liệu. Cài đặt và cấu hình Room để dễ dàng quản lý cơ sở dữ liệu và tương tác với dữ liệu.

## **1.4. Các công nghệ bổ sung**

### **1.4.1. Firebase**

*1.4.1.1. Định nghĩa và vai trò:*

**Giới thiệu Firebase**: Firebase là nền tảng phát triển ứng dụng của Google cung cấp một loạt các dịch vụ backend như xác thực người dùng, lưu trữ dữ liệu, và thông báo đẩy, giúp tăng tốc phát triển và triển khai ứng dụng.

*1.4.1.2. Các dịch vụ chính:*

 **Xác thực người dùng**: Cung cấp các dịch vụ xác thực mạnh mẽ, hỗ trợ nhiều phương thức như đăng nhập bằng email và mật khẩu, xác thực qua mạng xã hội, và xác thực qua số điện thoại.

 **Lưu trữ dữ liệu**: Firebase Realtime Database và Firestore cho phép lưu trữ và đồng bộ hóa dữ liệu trong thời gian thực giữa các thiết bị.

 **Thông báo đẩy**: Firebase Cloud Messaging (FCM) cung cấp dịch vụ thông báo đẩy để gửi thông báo đến người dùng trên các thiết bị di động và web.

*1.4.1.3. Tích hợp Firebase:*

**Cài đặt Firebase**: Tạo một dự án Firebase trên Firebase Console, thêm tệp cấu hình google-services.json vào dự án Android, và tích hợp các dịch vụ Firebase cần thiết thông qua các phụ thuộc trong tệp build.gradle.

### **1.4.2. API và Web Services**

*1.4.2.1. Khái niệm API:*

**Giới thiệu về API**: API (Application Programming Interface) là tập hợp các giao diện lập trình cho phép các ứng dụng giao tiếp với nhau. Trong phát triển ứng dụng di động, API thường được sử dụng để kết nối và lấy dữ liệu từ các dịch vụ web và hệ thống bên ngoài.

*1.4.2.2. Kết nối với Web Services:*

 **Sử dụng API**: Kết nối ứng dụng với các dịch vụ web thông qua các yêu cầu HTTP, cho phép gửi và nhận dữ liệu từ các máy chủ từ xa.

 **Xử lý dữ liệu từ xa**: Xử lý dữ liệu trả về từ các dịch vụ web, bao gồm phân tích JSON hoặc XML và cập nhật giao diện người dùng hoặc cơ sở dữ liệu ứng dụng.

*1.4.2.3. Thực hiện yêu cầu mạng:*

 **Gửi yêu cầu HTTP**: Sử dụng các thư viện như Retrofit hoặc Volley để gửi các yêu cầu HTTP và xử lý phản hồi từ server. Các thư viện này giúp đơn giản hóa việc thực hiện các yêu cầu mạng và xử lý dữ liệu trả về.

 **Xử lý phản hồi**: Phân tích dữ liệu JSON hoặc XML trả về từ server và thực hiện các hành động cần thiết, chẳng hạn như cập nhật giao diện người dùng hoặc lưu trữ dữ liệu vào cơ sở dữ liệu.