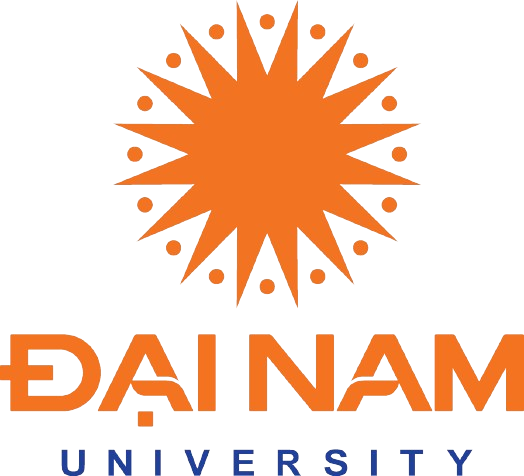
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**



**BÀI TẬP LỚN**

**TÊN HỌC PHẦN:** LẬP TRÌNH MOBILE

**ĐỀ TÀI:** Tìm hiểu API về các loài mèo

**Giáo viên hướng dẫn:** Trần Đức Minh

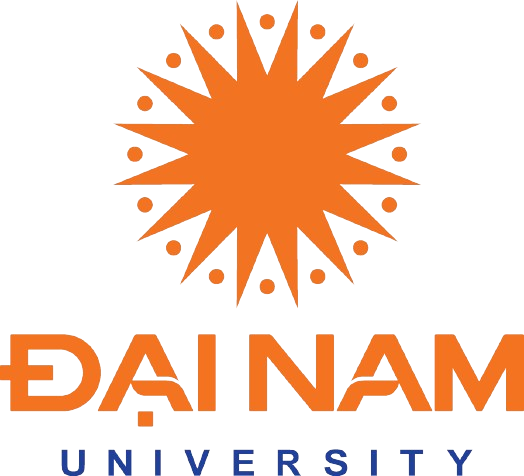
**Sinh viên thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Mã SV | Họ và tên | Lớp |
| 1 | 1571020020 | Nguyễn Ngọc Ánh | CNTT 15-01 |

**Hà Nội, năm 2024**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**



**BÀI TẬP LỚN**

**TÊN HỌC PHẦN:** LẬP TRÌNH MOBILE

**ĐỀ TÀI:** Tìm hiểu API về các loài mèo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng số | Bằng chữ |
| 1 | 1571020020 | Nguyễn Ngọc Ánh | 25/08/2003 |  |  |

**CÁN BỘ CHẤM THI 1 CÁN BỘ CHẤM THI 2**

**Hà Nội, năm 2024**

LỜI NÓI ĐẦU

MỤC LỤC

[CHƯƠNG I. TỔNG QUAN 6](#_Toc169160973)

[1.1 Tổng quan đề tài nghiên cứu 6](#_Toc169160974)

[1.2 Mục tiêu nghiên cứu 7](#_Toc169160975)

[1.3 Kết quả nghiên cứu 7](#_Toc169160976)

[1.4 Cấu trúc đề tài nghiên cứu 7](#_Toc169160977)

[CHƯƠNG II. CƠ SỞ LÝ THUYẾT 9](#_Toc169160978)

[2.1 Tổng quan Java 9](#_Toc169160979)

[2.1.1 Giới thiệu về Java 9](#_Toc169160980)

[2.1.2 Ưu điểm của Java 10](#_Toc169160981)

[2.1.3 Nhược điểm của Java 10](#_Toc169160982)

[2.2 Tổng quan về Android Studio 11](#_Toc169160983)

[2.2.1 Giới thiệu về Android Studio 11](#_Toc169160984)

[2.2.2 Ưu điểm của Android Studio 12](#_Toc169160985)

[2.2.3 Nhược điểm của Android Studio 13](#_Toc169160986)

[2.3 Mô hình 13](#_Toc169160987)

[2.3.1 Giới thiệu về mô hình MVC 13](#_Toc169160988)

[2.3.2 Các thành phần trong mô hình MVC 14](#_Toc169160989)

[2.3.3 Luồng xử lý trong MVC 15](#_Toc169160990)

[2.3.4 Ưu điểm của MVC 15](#_Toc169160991)

[2.3.5 Nhược điểm của MVC 16](#_Toc169160992)

[2.4 Tổng quan cơ sở dữ liệu SQLite 16](#_Toc169160993)

[2.4.1 Giới thiệu về SQLite 16](#_Toc169160994)

[2.4.2 Ưu điểm của SQLite 17](#_Toc169160995)

[2.4.3 Nhược điểm của SQLite 17](#_Toc169160996)

[CHƯƠNG III. TRIỂN KHAI XÂY DỰNG CHƯƠNG TRÌNH 18](#_Toc169160997)

[3.1 Danh sách các chức năng 18](#_Toc169160998)

[3.1.1 Đăng nhập 18](#_Toc169160999)

[3.1.2 Đăng ký 18](#_Toc169161000)

[3.1.3 Tìm kiếm thông tin 18](#_Toc169161001)

[3.1.4 Thêm vào mục yêu thích 18](#_Toc169161002)

[3.1.5 Chi tiết hình ảnh yêu thích 18](#_Toc169161003)

[3.2 Triển khai chương trình 18](#_Toc169161004)

[KẾT LUẬN 19](#_Toc169161005)

[i. Kết quả đạt được 19](#_Toc169161006)

[ii. Hướng phát triển 19](#_Toc169161007)

[TÀI LIỆU THAM KHẢO 20](#_Toc169161008)

**MỤC LỤC HÌNH ẢNH**

**BẢNG CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **TỪ VIẾT TẮT** | **VIẾT ĐẦY ĐỦ** |
| **1** | **API** | Application Programming Interface |
| **2** | **HTTP** | Hypertext Transfer Protocol |
| **3** | **JVM** | Java Virtual Machine |
| **4** | **IDE** | Integrated Development Enviroment |
| **5** | **MVC** | Model-View-Controller |
| **6** | **DBMS** | Database Management System |

# CHƯƠNG I. TỔNG QUAN

## 1.1 Tổng quan đề tài nghiên cứu

Đề tài nghiên cứu “Tìm hiểu API về các loài mèo” nhằm khám phá và tìm hiểu về các API (Application Programming Interface) liên quan đến thông tin về các loài mèo. Trong nghiên cứu này, các API có thể bao gồm các dịch vụ cung cấp thông tin về các loài mèo, như thông tin về giống, thông tin loài, hình ảnh, video hoặc các dữ liệu liên quan.

## 1.2 Mục tiêu nghiên cứu

Mục tiêu của đề tài này là nắm vững cách sử dụng và tương tác với các API về các loài mèo, từ đó thu thập thông tin chi tiết và đa dạng về chúng. Các phương pháp nghiên cứu có thể bao gồm việc tìm hiểu tài liệu API, thực hiện các yêu cầu HTTP để truy cập dữ liệu, phân tích dữ liệu trả về từ các API, và hiểu rõ cách sử dụng các tham số và tùy chọn để tối ưu hóa việc truy xuất thông tin.

Mục tiêu của đề tài “Tìm hiểu API về các loài mèo” bao gồm các phần chính sau:

* Quản lý Firebase: Sử dụng Firebase để xây dựng hệ thống quản lý người dùng cho ứng dụng. Điều này bao gồm tính năng đăng ký, đăng nhập và quản lý thông tin cá nhân của người dùng. Firebase cung cấp một cơ sở hạ tầng mạnh mẽ cho việc xử lý và xác thực người dùng và lưu trữ dữ liệu người dùng một cách an toàn.
* Quản lý danh mục yêu thích mèo thông qua SQLite: Sử dụng cơ sở dữ liệu SQLite để lưu trữ thông tin về các loài mèo mà người dùng yêu thích. Các chức năng bao gồm thêm, xóa và cập nhật thông tin về các loài mèo được người dùng quan tâm. SQLite là một cơ sở dữ liệu nhẹ nhàng và linh hoạt, phù hợp cho việc lưu trữ dữ liệu cục bộ trên thiết bị di động.
* Sử dụng API mèo để lấy hình ảnh: Tìm hiểu và tích hợp API liên quan đến thông tin về các loài mèo, đặc biệt là API cung cấp hình ảnh về các loài mèo. Sử dụng API này để truy xuất và hiển thị hình ảnh của các loài mèo trong ứng dụng. Quá trình này bao gồm việc hiểu cách gửi các yêu cầu đến API, xử lý dữ liệu trả về và hiển thị hình ảnh một cách thẩm mỹ và hiệu quả trong ứng dụng

## 1.3 Kết quả nghiên cứu

Kết quả của nghiên cứu có thể là việc xây dựng ứng dụng hoặc trang web với tính năng liên quan đến các loài mèo, sử dụng dữ liệu được thu thập thông qua các API. Đồng thời, nghiên cứu này cũng có thể mang lại những hiểu biết sâu sắc về các công nghệ liên quan đến API và cách chúng có thể được áp dụng trong việc nghiên cứu và phát triển ứng dụng về động vật.

## 1.4 Cấu trúc đề tài nghiên cứu

Bài tập lớn gồm 3 chương

Chương I. TỔNG QUAN

Tóm tắt sơ lược tổng quan về đề tài, mục tiêu và kết quả cần đạt được

Chương II. CƠ SỞ LÝ THUYẾT

Giới thiệu về Java, Android Studio, Mô hình MVC, Cơ sở dữ liệu SQLite

Chương III. TRIỂN KHAI XÂY DỰNG CHƯƠNG TRÌNH

Nguyên lý hoạt động, viết chương trình API cho loài mèo, kết quả chương trình

KẾT LUẬN

Kết quả đạt được, hướng phát triển

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo từ nhiều nguồn

# CHƯƠNG II. CƠ SỞ LÝ THUYẾT

## 2.1 Tổng quan Java

### 2.1.1 Giới thiệu về Java



Hình 2.1 Hình ảnh Java

Java là một ngôn ngữ lập trình phổ biến và mạnh mẽ, được phát triển bởi Sun Microsystems và hiện đang thuộc sở hữu của Oracle Corporation. Xuất hiện lần đầu vào năm 1995, Java đã nhanh chóng trở thành một trong những ngôn ngữ phổ biến nhất trên thế giới nhờ vào những đặc điểm độc đáo của mình.

Điều làm nổi bật Java là tính chất đa nền tảng, khi mã nguồn của nó được biên dịch thành bytecode có thể chạy trên bất kỳ máy ảo Java (JVM) nào, giúp ứng dụng có thể chạy trên nhiều hệ điều hành khác nhau mà không cần sửa đổi. Java cũng được thiết kế theo hướng đối tượng, hỗ trợ các khái niệm như đóng gói, kế thừa và đa hình, làm cho mã nguồn trở nên dễ bảo trì và tái sử dụng.

Sự bảo mật cao cũng là một điềm mạnh của Java, với mô hình an ninh giới hạn quyền truy cập của chương trình và JVM kiểm soát các tài nguyên như bộ nhớ và file. Java cung cấp thư viện chuẩn rộng lớn và API mạnh mẽ, giúp lập trình viên xây dựng ứng dụng hiệu quả và linh hoạt.

Với sự thuận tiện cho phát triển web thông qua các framework như Spring và Struts, Java tiếp tục là một lựa chọn hàng đầu trong ngành công nghiệp phần mềm. Điều này càng được củng cố bởi cộng đồng lập trình viên lớn và tích cực, cung cấp sự hỗ trợ và tài nguyên đa dạng thông qua các diễn đàn và nguồn mở. Java không chỉ là mộn ngôn ngữ lập trình, mà còn là một hệ sinh thái mạnh mẽ đằng sau sự phát triển của nhiều ứng dụng và hệ thống quan trọng trên khắp thế giới.

### 2.1.2 Ưu điểm của Java

Java đồng nhiên là một trong những ngôn ngữ lập trình được ưa chuộng không chỉ vì tính đa nền tảng và hướng đối tượng mà còn vì một loạt các ưu điểm mà nó mang lại.

Một trong những ưu điểm quan trọng của Java là khả năng đa nhiệm và đồng thời, giúp ứng dụng có thể thực hiện nhiều công việc cùng một lúc mà không làm giảm hiệu suất. JVM quản lý quy trình và bộ nhớ một cách hiệu quả, giúp người lập trình tập trung vào việc phát triển ứng dụng mà không phải quan tâm đến quản lý nguồn tài nguyên.

Tính bảo mật cao cũng là một trong những ưu điểm quan trọng, với mô hình bảo mật cấp độ và cơ chế kiểm soát quyền truy cập, Java ngăn chặn hiệu quả các lỗ hổng bảo mật phổ biến, giúp bảo vệ ứng dụng khỏi các cuộc tấn công và lợi dụng.

Mô hình thu gom rác (garbage collection) là một yếu tố quan trọng giúp giảm bớt gánh nặng của lập trình viên về quản lý bộ nhớ. Hệ thống tự động thu gom rác giúp giải phóng bộ nhớ tự đông, giảm nguy cơ xuất hiện lỗi bộ nhớ và làm cho quá trình phát triển và bảo trì ứng dụng trở nên dễ dàng hơn.

Java còn được biết đến với thư viện chuẩn và API phong phú, giúp lập trình viên tiết kiệm thời gian và công sức khi xây dựng ứng dụng. Cộng đồng phát triển lớn và sự hỗ trợ liên tục từ Oracle cũng đóng góp vào việc làm cho Java trở thành một lựa chọn ổn định và đáng tin cậy trong ngành công nghiệp lập trình phần mềm.

### 2.1.3 Nhược điểm của Java

Mặc dù Java có nhiều ưu điểm nổi bật, nhưng cũng không tránh khỏi một số nhược điểm mà cần được xem xét.

Một trong những nhược điểm quan trọng của Java là sự chậm chạp trong việc thực hiện một số nhiệm vụ so với một số ngôn ngữ lập trình khác. Nguyên nhân chính là do Java sử dụng máy ảo (JVM) để chạy bytecode, điều này có thể tạo ra một lớp trễ và làm giảm hiệu suất ứng dung, đặc biệt là trong các ứng dụng yêu cầu xử lý số liệu lớn và phức tạp.

Một nhược điểm khác là môi trường phát triển chậm và nặng nề. Quá trình biên dịch và chạy các ứng dụng Java có thể mất thời gian đáng kể, đặc biêt là khi so sánh với các ngôn ngữ lập trình biên dịch trực tiếp. Điều này có thể ảnh hưởng đến trải nghiệm phát triển và kiểm thử ứng dung.

Một nhược điểm khác của Java là sự phụ thuộc mạnh mẽ vào máy ảo Java (JVM). Mặc dù điều này mang lại tính đa nền tảng, nhưng cũng tạo ra một sự phụ thuộc lớn vào JVM, điều này có thể tạo ra khả năng không tương thích giữa các phiên bản JVM khác nhau và giữa các hệ điều hành.

Cuối cùng, mặc dù Java có hỗ trợ đa luồng (multithreading), nhưng quản lý luồng trong Java có thể phức tạp và dễ dàng gây ra lỗi nếu không được thực hiện đúng cách, điều này làm tăng khả năng phức tạp của việc phát triển ứng dụng đa nhiệm.

Mặc dù có nhược điểm, Java vẫn là một trong những ngôn ngữ lập trình phổ biến và mạnh mẽ được sử dụng rộng rãi trong nhiều lĩnh vực phát triển phần mềm.

## 2.2 Tổng quan về Android Studio

### 2.2.1 Giới thiệu về Android Studio



Hình 2.2 Hình ảnh Android Studio

Android Studio là một môi trường tích hợp phát triển (Integrated Development Enviroment – IDE) được phát triển bởi Google dành cho việc phát triển ứng dụng trên nền tảng Android. Nó cung cấp một loạt các công cụ và tính năng để giúp nhà phát triển xây dựng và triển khai ứng dụng Android một cách dễ dàng.

Công cụ này cung cấp một giao diện trực quan và các trình biên dịch, trình gỡ lỗi, trình tạo giao diện người dùng, trình quản lý phiên bản và nhiều công cụ khác. Nó hỗ trợ viết code trong ngôn ngữ Java hoặc Kotlin và tích hợp sẵn các thư viện và công cụ phát triển Android.

Một trong những tính năng quan trọng của công cụ này là khả năng sử dụng Gradle, một hệ thống quản lý dự án mạnh mẽ. Gradle cho phép bạn quản lý phụ thuộc, xây dựng, kiểm thử và đóng gói ứng dụng Android một cách linh hoạt và hiệu quả.

Androi Studio đã được công bố lần đầu tiên vào năm 2013 tại hội nghị Google I/O và chính thức phát hành rộng rãi vào năm 2014. Trước đó, các nhà phát triển ứng dụng Android thường sử dụng các công cụ như Eclipse IDE hoặc IDE Java để phát triển ứng dụng.

Sau khi được công bố công cụ này đã trở thành một công cụ phát triển ứng dụng Android chính thức và được Google khuyến khích sử dụng. Nó đã thay thế Eclipse IDE và trở thành môi trường phát triển chính cho các dự án Android.

### 2.2.2 Ưu điểm của Android Studio

Với sự phát triển của công nghệ thông tin, có rất nhiều môi trường giúp nhà phát triển có thể phát triển các ứng dụng nền tảng Android nhưng nó vẫn không thể thay thế Android Studio. Nó có những ưu điểm sau:

* Đây là môi trường phát triển phần mềm chính thức của Google, đây cũng chính là chủ sở hữu của hệ điều hành Android.
* Công cụ này cung cấp một loạt các công cụ phát triển và tính năng hỗ trợ cho quả trình phát triển ứng dụng Android. Điều này bao gồm trình biên dịch, trình gỡ lỗi, trình quản lý dự án, thiết kế giao diện và nhiều tính năng khác để tăng năng suất và hiệu quả của nhà phát triển.
* Giao diện người dùng thân thiện và dễ sử dụng, giúp tăng hiệu suất và giảm thời gian cần thiết để phát triển ứng dụng.
* Hỗ trợ bởi một số tài liệu tham khảo và hướng dẫn phong phú. Có sẵn các tài liệu chính thức từ Google, cùng với các diễn đàn lập trình viên Android và các nguồn tài liệu trực tuyến khác để giúp nhà phát triển tìm hiểu và giải quyết các vấn đề phát triển.
* Công cụ này được hỗ trợ thông qua các khóa học đào tạo về lập trình Android cơ bản và nâng cao. Các khóa học này giúp nhà phát triển nắm vững các khái niệm và kỹ năng cần thiết để phát triển ứng dụng Android chất lượng.

### 2.2.3 Nhược điểm của Android Studio

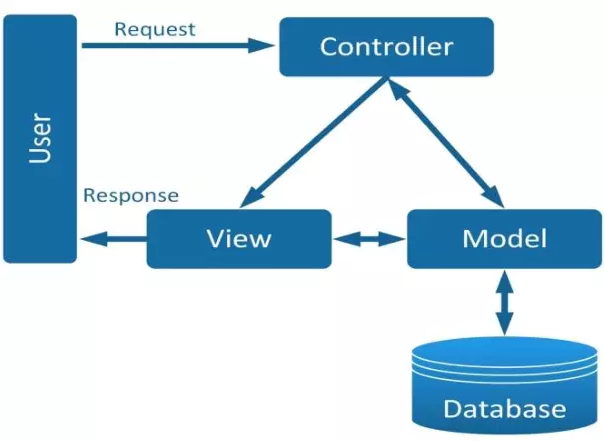
Android mặc dù là một môi trường phát triển ứng ụng di động hiệu quả, nhưng cũng không tránh khỏi một số nhược điểm cần được xem xét. Đầu tiên, việc tài nguyên hệ thống lớn của Android Studio có thể tạo ra áp lực đáng kể đối với các máy tính có cấu hình thấp, làm giảm hiệu suất và làm chậm quá trình phát triển.

Thời gian khởi động Android Studio cũng là một vấn đề, đặc biệt là khi làm việc với các dự án lớn. Quá trình này có thể tăng thời gian chờ đợi của lập trình viên và làm gián đoạn quá trình làm việc. Ngoài ra, sự hỗ trợ không đồng đều cho các ngôn ngữ khác nhau, đặc biệt là C++, có thể tạo ra những thách thức và khó khăn trong quá trình phát triển.

Các vấn đề về plugin và quá trình cập nhật cũng là điểm đáng chú ý. Việc sử dụng các plugin có thể gây ra sự không ổn định hoặc xung đột trong Android Studio, đặc biệt là khi có sự cập nhật. Điều này có thể làm phức tạp quá trình cập nhật và sử dụng các tính năng mở rộng.

## 2.3 Mô hình

### 2.3.1 Giới thiệu về mô hình MVC



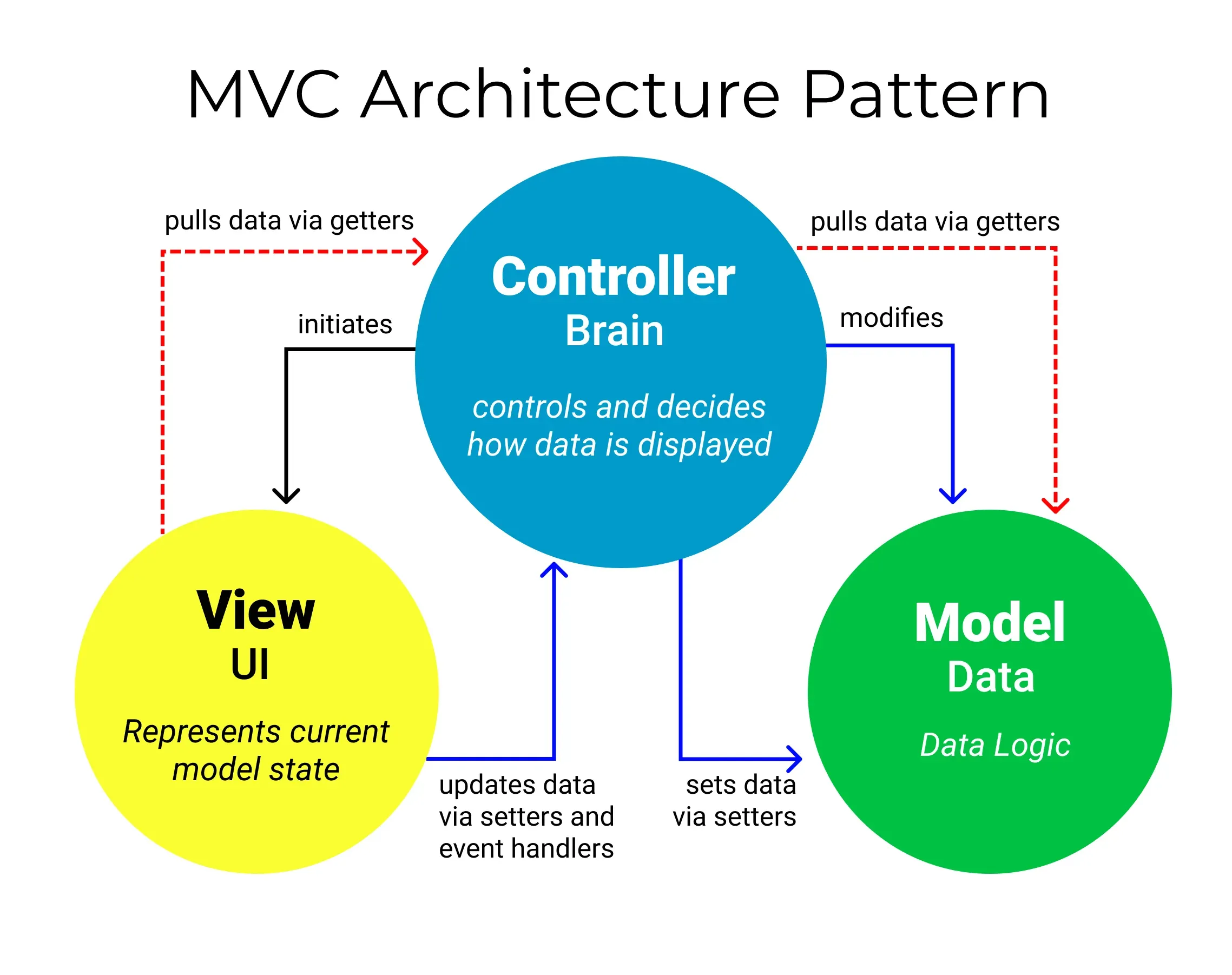
Hình 2.3 Hình ảnh MVC

MVC là viết tắt của cụm từ “Model-View-Controller”. Đây là mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. MVC là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tinh. MVC chia thành 3 phần được kết nối với nhau và mỗi thành phần đều có một nhiệm vụ riêng của nó và độc lập với các thành phần khác. Tên gọi 3 thành phần:

* Model (dữ liệu): Quản lý xử lý các dữ liệu
* View (giao diện): Nơi hiển thị dữ liệu cho người dùng
* Controller (bộ điều khiển): Điều khiển sự tương tác của 2 thành phần Model và View

Mô hình MVC thường được dùng để phát triển giao diện người dùng. Nó cung cấp các thành phần cơ bản để thiết kế một chương trình cho máy tính hoặc điện thoại di động, cũng như là các ứng dụng web.

### 2.3.2 Các thành phần trong mô hình MVC



Hình 2.4 Hình ảnh mô hình MVC

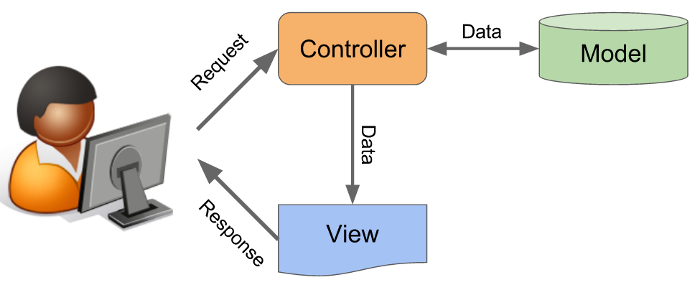
Mô hình MVC gồm 3 loại chính là thành phần bên trong không thể thiếu khi áp dụng mô hình này:

* Model: Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng và là cầu nối giữa 2 thành phần bên dưới View và Controller. Một model là dữ liệu được sử dụng bởi chương trình. Đây có thể là cơ sở dữ liệu, hoặc file XML bình thường hay một đối tượng đơn giản. Chẳng hạn như biểu tượng hay là một nhân vật trong game.
* View: Đây là phần giao diên (theme) dành cho người sử dụng. View là phương tiện hiển thị các đối tượng trong một ứng dụng. Chẳng hạn như hiển thị môt cửa sổ, nút hay văn bản trong một cửa sổ khác. Nó bao gồm bất cứ thứ gì mà người dùng có thể nhìn thấy được.
* Controller: Là bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua View. Một controller bao gồm cả Model lẫn View. Nó nhận input và thực hiện các update tương ứng.

### 2.3.3 Luồng xử lý trong MVC

Luồng xử lý trong MVC cụ thể là:

* Khi một yêu cầu từ máy khách (Client) gửi đến Server. Thì bị Controller trong MVC chặn lại để xem đó là URL request hay sự kiện.
* Sau đó Controller xử lý input của user rồi giao tiếp với Model trong MVC.
* Model chuẩn bị data và gửi lại cho Controller.
* Cuối cùng, khi xử lý xong yêu cầu thì Controller gửi dữ liệu trở lại View và hiển thị cho người dùng trên trình duyệt.



Hình 2.5 Hình ảnh luồng xử lý

### 2.3.4 Ưu điểm của MVC

Đầu tiên nhắc tới ưu điểm của mô hình MVC thì đó là bang thông nhẹ vì không sử dụng viewstate nên khá tiết kiệm băng thông. Việc giảm băng thông giúp website hoạt động ổn định hơn.

Kiểm tra đơn giản và dễ dàng, kiểm tra lỗi phần mềm trước khi bàn giao lại cho người dùng.

Một lợi thế chính của MVC là nó tách biệt các phần Model, Controller và View với nhau. Sử dụng mô hình MVC chức năng Controllercos vai trò quan trọng và tối ưu trên các nền tảng ngôn ngữ khác nhau. Ta có thể dễ dàng duy trì ứng dụng vì chúng được tách biệt với nhau. Có thể chia nhiều developer làm việc cùng một lúc. Công việc của các developer sẽ không ảnh hưởng đến nhau.

### 2.3.5 Nhược điểm của MVC

Bên cạnh những ưu điểm của MVC mang lại thì nó cũng có một số nhược điểm cần khắc phục.

MVC đa phần phù hợp với công ty chuyên về website hoặc các dự án lớn thì mô hình này phù hợp hơn so với các dự án nhỏ lẻ vì khá là cồng kềnh và mất thời gian.

Mô hình MVC (Model-View-Controller) là một kiến trúc phát triển phần mềm mạnh mẽ, nhưng cũng đi kèm với một số nhược điểm cần được xem xét. Ban đầu, việc hiểu và triển khai mô hình này có thể phức tạp, đặc biệt đối với người mới bắt đầu. Sự tách biệt chặt chẽ giữa Model (mô hình), View (giao diện) và Controller (điều khiển) có thể dẫn đến hiệu suất kém trong trường hợp các ứng dụng yêu cầu nhiều tương tác giữa các thành phần này.

Một trong những thách thức khác là việc quản lý mã trở nên phức tạp, đặc biệt là trong các dự án lớn. Cấu trúc MVC có thể tạo ra nhiều lớp và tầng lớp mã, làm cho mã trở nên khó quản lý và khó hiểu. Thời gian phát triển dự án cũng có thể kéo dài hơn so với các mô hình khác, đặc biệt khi yêu cầu của dự án thay đổi thường xuyên.

Đối với các dự án nhỏ và đơn giản, triển khai mô hình MVC có thể cảm thấy quá phức tạp và không cần thiết, làm cho mã trở nên quá nặng nề và khó quản lý. Cuối cùng, việc sử dụng mô hình MVC đòi hỏi kiến thức và kỹ năng cao cho phát triển và bảo trì hiệu quả, và có thể không phù hợp cho những dự án với đội ngũ phát triển không có kinh nghiệm trong việc làm việc với kiến trúc này.

## 2.4 Tổng quan cơ sở dữ liệu SQLite

### 2.4.1 Giới thiệu về SQLite

SQL là hệ quản trị cơ sở dữ liệu (DBMS) quan hệ tương tự như MySQL,… Đặc điểm nổi bật của SQLite so với các DBMS khác là gọn, nhẹ, đơn giản, đặc biệt không cần mô hình client-server, không cần cài đặt, cấu hình hay khởi động nên không có khái niệm user, password hay quyền hạn trong SQLite Database. Dữ liệu cũng được lưu ở một file duy nhất.

SQLite thường không được sử dụng với các hệ thống lớn nhưng với hệ thống ở quy mô vừa và nhỏ thì SQLite không thua các DBMS khác về chức năng hay tốc độ. Vì không cần cài đặt hay cấu hình nên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm,… vì tránh được những rắc rối trong quá trình cài đặt.

### 2.4.2 Ưu điểm của SQLite

SQLite, với những ưu điểm nổi bật, đã chứng minh là một hệ quản trị cơ sở dữ liệu (Hệ quản trị cơ sở dữ liệu – DBMS) lựa chọn xuất sắc cho nhiều ứng dụng di động và môi trường nhúng. Với kích thước nhỏ gọn và khả năng tích hợp trực tiếp vào ứng dụng, SQLite giúp giảm độ trễ và tối ưu hóa hiệu suất, đặc biệt là trong môi trường di động có tài nguyên hạn chế.

Khả năng chịu tải của SQLite là một trong những ưu điểm nổi bật khác, đảm bảo ổn định và đáng tin cậy khi đối mặt với số lượng người dùng đồng thời lớn. Sự tương thích mạnh mẽ với nhiều ngôn ngữ lập trình và môi trường phát triển giúp giảm thiểu vấn đề tương thích, đồng thời làm cho việc tích hợp trở nên dễ dàng.

Việc hỗ trợ giao thức ACID, bao gồm Atomicitty, Consistency, Isolation và Durability làm tăng tính nhất quán và an toàn của dữ liệu. Đồng thời, hiệu suất cao của SQLite trong điều kiện tài nguyên hạn chế và khả năng lưu trữ dữ liệu trong tệp duy nhất giúp quản lý dữ liệu trở nên đơn giản và hiệu quả.

### 2.4.3 Nhược điểm của SQLite

Mặc dù SQLite mang lại nhiều ưu điểm, nhưng cũng có một số nhược điểm cần được xem xét khi áp dụng trong một số tình huống cụ thể.

Trước hết, do SQLite hoạt động trực tiếp trên thiết bị và không có máy chủ cơ sở dữ liệu riêng biệt, điều này có thể gây ra những hạn chế trong khả năng mở rộng của hệ thống. Trong môi trường với số lượng người dùng đồng thời lớn và yêu cầu về xử lý cơ sở dữ liệu phức tạp, SQLite có thể không đáp ứng được như một số giải pháp cơ sở dữ liệu có máy chủ.

Một nhược điểm khác là khả năng chịu tải của SQLite có thể gặp vấn đề khi phải xử lý nhiều thao tác ghi đồng thời. Trong những tình huống yêu cầu nhiều người dùng cùng truy cập và thay đổi dữ liệu, có thể xảy ra tình trạng đụng độ dữ liệu, ảnh hưởng đến tính nhất quán của cơ sở dữ liệu.

SQLite cũng có giới hạn trong việc xử lý các tình huống đòi hỏi nhiều chức năng và tính năng của hệ quản trị cơ sở dữ liệu lớn hơn. Nó không cung cấp các công cụ quản lý và giám sát phức tạp như một số hệ quản trị cơ sở dữ liệu lớn khác, điều này có thể làm giảm khả năng theo dõi và tối ưu hóa hiệu suất.

Cuối cùng, trong các tình huống yêu cầu bảo mật cao, SQLite có thể không đáp ứng được như các hệ quản trị cơ sở dữ liệu có máy chủ, đặc biệt là khi cần quản lý quyền truy cập và kiểm soát người dùng chi tiết.

# CHƯƠNG III. TRIỂN KHAI XÂY DỰNG CHƯƠNG TRÌNH

## 3.1 Danh sách các chức năng

### 3.1.1 Đăng nhập

- Đăng nhập thông qua Firebase giúp quản lý tài khoản người dùng.

### 3.1.2 Đăng ký

- Tạo một tài khoản mới bằng Firebase cung cấp thông tin cá nhân

### 3.1.3 Tìm kiếm thông tin

- Tìm kiếm hình ảnh, thông tin về sản phẩm tại mục tìm kiếm theo từ khóa “tên”

### 3.1.4 Thêm vào mục yêu thích

- Chức năng này cho phép người dùng thêm hình ảnh mèo vào danh sách yêu thích của mình để dễ dàng tìm kiếm loài mèo mà mình yêu thích, loài mèo mà mình muốn tìm hiểu

### 3.1.5 Chi tiết hình ảnh yêu thích

- Chi tiết về một loài mèo cụ thể, bao gồm hình ảnh, tên, giống loài, màu lông, tuổi thọ

## 3.2 Triển khai chương trình

# KẾT LUẬN

## Kết quả đạt được

## Hướng phát triển

# TÀI LIỆU THAM KHẢO

[1] Trần Đức Minh, Giáo trình Lập trình Mobile, Đại Học Đại Nam.

[2] Trần Văn Quang, Nguyễn Văn Thái, Nguyễn Bảo Nam (2022), Báo cáo bài tập lớn môn Mobile, Trường Đại Học Đại Nam.

[3] Trần Đức Minh, Giáo trình Lập trình Mobile, Đại Học Đại Nam.