**TRƯỜNG ĐẠI HỌC PHENIKAA**



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**Xây dựng hệ thống đánh giá mặt phẳng của địa hình (Assessing terrain’s surface)**

**20010960 Lê Tiến Hưng 20010960**@st.phenikaa-uni.edu.vn

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Mai Xuân Tráng |
| **Khoa:** | Công nghệ thông tin |
| **HÀ NỘI, 06/2025** | |

# Lời cam kết

Họ và tên nhóm sinh viên:

* Lê Tiến Hưng

Điện thoại liên lạc: 0939624810 . . . . . . Email: **20010960**@st.phenikaa-uni.edu.vn

Lớp: K14 Công nghệ thông tin Việt – Nhật. . . . . . . . . Hệ đào tạo: Chính quy.

Tôi cam kết Bài tập lớn (BTL) là công trình nghiên cứu của bản thân tôi. Các kết quả nêu trong BTL là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong BTL – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi/chúng tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm*  Tác giả/nhóm tác giả BTL  *Họ và tên sinh viên* |

# Tóm tắt

Trong lĩnh vực lập trình hệ thống tự động, đánh giá địa hình đóng vai trò quan trọng trong nhiều ứng dụng, bao gồm giám sát môi trường, nông nghiệp, ứng phó thảm họa và thám hiểm hành tinh. Việc hiểu rõ các điều kiện bề mặt như độ gồ ghề, độ dốc và thành phần vật liệu là điều cần thiết để tối ưu hóa khả năng di chuyển, an toàn và hiệu suất trong các môi trường khác nhau.

Dự án này, có tiêu đề " Thiết Kế, Xây Dựng Hệ Thống Đánh Giá Mặt Phẳng Của Địa Hình", tập trung vào việc phát triển một hệ thống có khả năng đánh giá và phân tích đặc điểm địa hình. Hệ thống được thiết kế để di chuyển qua nhiều loại địa hình khác nhau, thu thập dữ liệu theo thời gian thực, gửi dữ liệu về điện thoại hoặc máy tính bảng và cung cấp thông tin chi tiết về điều kiện bề mặt thông qua sự kết hợp của các cảm biến và thuật toán phân tích.

Mục tiêu chính của dự án là tích hợp các thành phần cơ khí, điện tử và phần mềm vào một nền tảng robot hoạt động hiệu quả. Hệ thống sẽ áp dụng các kỹ thuật phân loại địa hình, hợp nhất dữ liệu cảm biến để thu thập thông tin chính xác, đồng thời triển khai một khung điều hướng tự động hoặc bán tự động. Quá trình thiết kế và chế tạo sẽ bao gồm các yếu tố về cấu trúc, lựa chọn vật liệu và phát triển hệ thống nhúng để đảm bảo độ bền và hiệu suất cao.

Báo cáo này trình bày toàn bộ quá trình phát triển của dự án, bao gồm thiết kế ý tưởng, tạo mẫu thử, tích hợp phần cứng và phần mềm, kiểm thử và đánh giá hiệu suất. Bằng cách triển khai thành công hệ thống robot này, dự án hướng đến việc đóng góp vào sự phát triển của các công nghệ đánh giá địa hình tự động, với tiềm năng ứng dụng trong cả lĩnh vực công nghiệp và nghiên cứu.

# Mục lục

Table of Contents

[Lời cam kết ii](#_Toc198105785)

[Tóm tắt iii](#_Toc198105786)

[Mục lục iv](#_Toc198105787)

[Danh mục hình vẽ vii](#_Toc198105788)

[Danh mục bảng viii](#_Toc198105789)

[Danh mục công thức ix](#_Toc198105790)

[Danh mục các từ viết tắt x](#_Toc198105791)

[Danh mục thuật ngữ xi](#_Toc198105792)

[Chương 1 Giới thiệu đề tài 1](#_Toc198105793)

[1.1 Đặt vấn đề 1](#_Toc198105794)

[1.2 Mục tiêu và phạm vi đề tài 2](#_Toc198105795)

[1.3 Định hướng giải pháp 3](#_Toc198105796)

[1.4 Bố cục đồ án 4](#_Toc198105797)

[Chương 2 Khảo sát và phân tích yêu cầu 6](#_Toc198105798)

[2.1 Khảo sát hiện trạng 6](#_Toc198105799)

[2.2 Tổng quan chức năng 9](#_Toc198105800)

[2.2.1 Biểu đồ use case tổng quan 10](#_Toc198105801)

[2.2.2 Biểu đồ use case phân rã 11](#_Toc198105802)

[2.2.3 Quy trình nghiệp vụ 17](#_Toc198105803)

[2.3 Đặc tả chức năng 21](#_Toc198105804)

[2.3.1 Đặc tả use case: Điều khiển phần cứng 21](#_Toc198105805)

[2.3.2 Đặc tả use case: Xem dữ liệu cảm biến 22](#_Toc198105806)

[2.3.3 Đặc tả use case: Xem biểu đồ gia tốc và con quay hồi chuyển 23](#_Toc198105807)

[2.3.4 Đặc tả use case: Xem kết quả đánh giá địa hình 24](#_Toc198105808)

[2.3.5 Đặc tả use case: Xem, xóa dữ liệu khảo sát 25](#_Toc198105809)

[2.4 Yêu cầu phi chức năng 26](#_Toc198105810)

[2.4.1 Hiệu năng (Performance) 26](#_Toc198105811)

[2.4.2 Độ tin cậy (Reliability) 26](#_Toc198105812)

[2.4.3 Tính dễ sử dụng (Usability) 26](#_Toc198105813)

[2.4.4 Tính dễ bảo trì (Maintainability) 26](#_Toc198105814)

[2.4.5 Các yêu cầu kỹ thuật khác 27](#_Toc198105815)

[Chương 3 Công nghệ sử dụng 28](#_Toc198105816)

[3.1 Lập trình nhúng với ngôn ngữ C trên ESP32 28](#_Toc198105817)

[3.2 Lập trình thiết bị di động với Java 33](#_Toc198105818)

[3.3 Lắp ráp phần cứng 37](#_Toc198105819)

[Chương 4 Phát triển và triển khai ứng dụng 39](#_Toc198105820)

[4.1 Thiết kế kiến trúc 39](#_Toc198105821)

[4.1.1 Lựa chọn kiến trúc phần mềm 39](#_Toc198105822)

[4.1.2 Thiết kế tổng quan 39](#_Toc198105823)

[4.1.3 Thiết kế chi tiết gói 40](#_Toc198105824)

[4.2 Thiết kế chi tiết 40](#_Toc198105825)

[4.2.1 Thiết kế giao diện 40](#_Toc198105826)

[4.2.2 Thiết kế lớp 41](#_Toc198105827)

[4.2.3 Thiết kế cơ sở dữ liệu 41](#_Toc198105828)

[4.3 Xây dựng ứng dụng 41](#_Toc198105829)

[4.3.1 Thư viện và công cụ sử dụng 41](#_Toc198105830)

[4.3.2 Kết quả đạt được 41](#_Toc198105831)

[4.3.3 Minh hoạ các chức năng chính 42](#_Toc198105832)

[4.4 Kiểm thử 42](#_Toc198105833)

[4.5 Triển khai 42](#_Toc198105834)

[Chương 5 Các giải pháp và đóng góp nổi bật 43](#_Toc198105835)

[Chương 6 Kết luận và hướng phát triển 44](#_Toc198105836)

[6.1 Kết luận 44](#_Toc198105837)

[6.2 Hướng phát triển 44](#_Toc198105838)

[Tài liệu tham khảo 45](#_Toc198105839)

[Phụ lục A-1](#_Toc198105840)

[Lời cam kết ii](#_Toc74298230)

[Tóm tắt iii](#_Toc74298231)

[Mục lục iv](#_Toc74298232)

[Danh mục hình vẽ vii](#_Toc74298233)

[Danh mục bảng viii](#_Toc74298234)

[Danh mục công thức ix](#_Toc74298235)

[Danh mục các từ viết tắt x](#_Toc74298236)

[Danh mục thuật ngữ xi](#_Toc74298237)

[Chương 1 Giới thiệu đề tài 1](#_Toc74298238)

[1.1 Đặt vấn đề 1](#_Toc74298239)

[1.2 Mục tiêu và phạm vi đề tài 2](#_Toc74298240)

[1.3 Định hướng giải pháp 2](#_Toc74298241)

[1.4 Bố cục bài tập lớn 2](#_Toc74298242)

[Chương 2 Khảo sát và phân tích yêu cầu 4](#_Toc74298243)

[2.1 Khảo sát hiện trạng 4](#_Toc74298244)

[2.2 Tổng quan chức năng 4](#_Toc74298245)

[2.2.1 Biểu đồ use case tổng quan 4](#_Toc74298246)

[2.2.2 Biểu đồ use case phân rã XYZ 5](#_Toc74298247)

[2.2.3 Quy trình nghiệp vụ 5](#_Toc74298248)

[2.3 Đặc tả chức năng 5](#_Toc74298249)

[2.3.1 Đặc tả use case A 5](#_Toc74298250)

[2.3.2 Đặc tả use case B 5](#_Toc74298251)

[2.4 Yêu cầu phi chức năng 5](#_Toc74298252)

[Chương 3 Công nghệ sử dụng 6](#_Toc74298253)

[Chương 4 Phát triển và triển khai ứng dụng 7](#_Toc74298254)

[4.1 Thiết kế kiến trúc 7](#_Toc74298255)

[4.1.1 Lựa chọn kiến trúc phần mềm 7](#_Toc74298256)

[4.1.2 Thiết kế tổng quan 7](#_Toc74298257)

[4.1.3 Thiết kế chi tiết gói 8](#_Toc74298258)

[4.2 Thiết kế chi tiết 8](#_Toc74298259)

[4.2.1 Thiết kế giao diện 8](#_Toc74298260)

[4.2.2 Thiết kế lớp 9](#_Toc74298261)

[4.2.3 Thiết kế cơ sở dữ liệu 9](#_Toc74298262)

[4.3 Xây dựng ứng dụng 9](#_Toc74298263)

[4.3.1 Thư viện và công cụ sử dụng 9](#_Toc74298264)

[4.3.2 Kết quả đạt được 9](#_Toc74298265)

[4.3.3 Minh hoạ các chức năng chính 10](#_Toc74298266)

[4.4 Kiểm thử 10](#_Toc74298267)

[4.5 Triển khai 10](#_Toc74298268)

[Chương 5 Các giải pháp và đóng góp nổi bật 11](#_Toc74298269)

[Chương 6 Kết luận và hướng phát triển 12](#_Toc74298270)

[6.1 Kết luận 12](#_Toc74298271)

[6.2 Hướng phát triển 12](#_Toc74298272)

[Tài liệu tham khảo 13](#_Toc74298273)

[Phụ lục A-1](#_Toc74298274)

[A Hướng dẫn viết Bài tập lớn A-1](#_Toc74298275)

[A.1 Quy định chung A-1](#_Toc74298276)

[A.2 Tạo đề mục A-2](#_Toc74298277)

[A.3 Bảng biểu A-2](#_Toc74298278)

[A.4 Hình vẽ A-3](#_Toc74298279)

[A.5 Tài liệu tham khảo A-4](#_Toc74298280)

[A.6 Công thức toán học A-4](#_Toc74298281)

[A.7 Tham chiếu chéo A-5](#_Toc74298282)

[A.8 Cập nhật mục lục và tham chiếu chéo A-5](#_Toc74298283)

[A.9 In quyển Bài tập lớn A-6](#_Toc74298284)

[B Đặc tả use case B-6](#_Toc74298285)

[B.1 Đặc tả use case “Thống kê tình hình mượn sách” B-6](#_Toc74298286)

[B.2 Đặc tả use case “Đăng ký làm thẻ mượn” B-6](#_Toc74298287)

[C Công nghệ sử dụng C-6](#_Toc74298288)

[C.1 Công nghệ bảo mật dữ liệu C-6](#_Toc74298289)

[C.2 Công nghệ blockchain C-6](#_Toc74298290)

[D Thiết kế gói D-6](#_Toc74298291)

[D.1 Thiết kế gói cho kiến trúc tổng quan D-6](#_Toc74298292)

[D.2 Thiết kế gói cho chức năng “Trả sách” D-6](#_Toc74298293)

[E Thiết kế lớp E-6](#_Toc74298294)

# Danh mục hình vẽ

[**Hình 1** Ví dụ biểu đồ phụ thuộc gói 7](#_Toc27562454)

[**Hình 2** Ví dụ thiết kế gói 8](#_Toc27562455)

[**Hình 3** Ví dụ hình vẽ A-4](#_Toc27562456)

[**Hình 4** Quy cách đóng quyển A-6](#_Toc27562457)

[**Hình 5** Quy cách ghi chữ phần gáy A-6](#_Toc27562458)

[**Hình 6** Hướng dẫn thiết lập in hai mặt A-7](#_Toc27562459)

# Danh mục bảng

[**Bảng 1** Danh sách thư viện và công cụ sử dụng 9](#_Toc27562460)

[**Bảng 2** Ví dụ sử dụng bảng A-2](#_Toc27562461)

# Danh mục công thức

[**Công thức 1** Khai triển Newton A-5](#_Toc27562462)

Lưu ý: Nếu BTL có ít hơn ba công thức toán học, sinh viên có thể xóa bỏ mục này.

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| UDP | User Datagram Protocol  Giao thức Datagram của người dùng |
| **N/A** | Not Available  Không khả dụng |
| **CSDL** | Cơ sở dữ liệu |
| **HTML** | HyperText Markup Language  Ngôn ngữ đánh dấu siêu văn bản |
| **CNTT** | Công nghệ thông tin |
| **BTL** | Bài tập lớn |
| **SV** | Sinh viên |

# Danh mục thuật ngữ

|  |  |
| --- | --- |
| Module | Thành phần nhỏ gọn tạo thành một phần của một tổng thể |
| **Driver** | Hệ thống điều khiển động cơ |
| **Filter** | Từ khóa lọc |
| **DC** | Direct Connect  Dòng điện chạy liên tục theo một hướng |
| **ESP32** | Module vi điều khiển được sử dụng |
| **MPU6050** | Module cảm biến gia tốc, con quay hồi chuyển được sử dụng |
| **L298N** | Module điều khiển động cơ được sử dụng |
| **I2C** | Inter-Integrated Circuit  Một giao thức truyền thông nối tiếp, dữ liệu được gửi từng bit dọc theo một đường duy nhất |
| **HIGH/LOW** | Trạng thái logic trong kỹ thuật |

# Giới thiệu đề tài

## Đặt vấn đề

Trong quá trình phát triển hạ tầng kỹ thuật, đặc biệt là trong các lĩnh vực xây dựng, nông nghiệp, giao thông và quốc phòng, việc khảo sát và đánh giá địa hình là một bước quan trọng nhằm đảm bảo tính hiệu quả và an toàn cho các công trình. Một trong những yếu tố cơ bản cần được đánh giá là độ phẳng (hay độ nghiêng) của mặt địa hình. Việc này không chỉ giúp xác định mức độ phù hợp của khu vực cho mục đích xây dựng hoặc triển khai thiết bị, mà còn góp phần quan trọng vào việc lập kế hoạch thi công, tối ưu hóa vật tư, và phòng tránh rủi ro.

Tuy nhiên, hiện nay việc đánh giá độ phẳng địa hình vẫn còn phụ thuộc nhiều vào các phương pháp truyền thống như sử dụng thước thủy, máy thủy bình hoặc thiết bị đo laser cầm tay, vốn đòi hỏi nhân lực có kỹ năng chuyên môn, thời gian khảo sát lâu, và khó triển khai trong điều kiện địa hình phức tạp hoặc không gian hạn chế. Điều này đặt ra nhu cầu cần thiết cho một hệ thống có khả năng tự động thu thập dữ liệu, đánh giá nhanh và trực quan tình trạng mặt phẳng địa hình, với độ chính xác và tính linh hoạt cao hơn.

Việc giải quyết bài toán này không chỉ mang lại lợi ích thiết thực trong việc tiết kiệm thời gian và công sức cho người khảo sát, nâng cao tính chính xác và hiệu quả trong các hoạt động đo đạc, mà còn có khả năng hoạt động trong nhiều dạng địa hình, đặc biệt là các nơi có không gian hẹp, không phù hợp cho thiết bị đo đạc cồng kềnh hoặc con người có thể đi vào trực tiếp. Hơn thế nữa, hệ thống đánh giá mặt phẳng địa hình có thể được ứng dụng rộng rãi trong nhiều lĩnh vực khác như nông nghiệp thông minh (đánh giá mặt đất để bố trí hệ thống tưới tiêu), robot di động (xác định khả năng di chuyển ổn định), và trong các thiết bị hỗ trợ người khuyết tật (đảm bảo bề mặt an toàn khi di chuyển).

## Mục tiêu và phạm vi đề tài

Hiện nay, nhu cầu đánh giá độ phẳng bề mặt địa hình một cách nhanh chóng, chính xác và linh hoạt ngày càng trở nên quan trọng trong nhiều lĩnh vực như xây dựng công trình, triển khai thiết bị di động, robot tự hành, hay nông nghiệp chính xác. Đáp ứng nhu cầu đó, trên thị trường đã xuất hiện một số thiết bị và công cụ hỗ trợ đánh giá mặt bằng như máy thủy bình điện tử, cảm biến laser 2D/3D, và các ứng dụng tích hợp trên máy bay không người lái (drone) với khả năng chụp ảnh và xây dựng bản đồ địa hình.

Tuy nhiên, các giải pháp hiện tại thường có những nhược điểm đáng kể như: chi phí cao, thiết bị cồng kềnh, yêu cầu kỹ năng chuyên môn khi vận hành, và không phù hợp để triển khai trong các môi trường chật hẹp hoặc yêu cầu tính linh hoạt cao. Một số thiết bị cầm tay tích hợp cảm biến nghiêng dù có tính cơ động nhưng lại hạn chế về khả năng xử lý dữ liệu, kết nối, và không cung cấp phản hồi thời gian thực theo yêu cầu của các ứng dụng hiện đại.

Trên cơ sở đó, đề tài hướng đến việc xây dựng một hệ thống đánh giá mặt phẳng địa hình ứng dụng công nghệ cảm biến và vi điều khiển, sử dụng các phần tử phần cứng phổ biến hiện nay như **ESP32** (với khả năng kết nối không dây), **cảm biến MPU6050** (tích hợp gia tốc kế và con quay hồi chuyển) để đo đạc và đánh giá mức độ nghiêng của địa hình. Ngoài ra, **driver L298N** cũng được tích hợp để điều khiển module chuyển động, phục vụ cho việc khảo sát thực tế trên địa hình đa dạng, giúp hệ thống có thể được điều khiển từ xa và thu thập dữ liệu tại nhiều vị trí, nhiều dạng địa hình mà con người khó tiếp cận.

Đặc biệt, tôi đã tự thiết kế và phát triển một ứng dụng di động Android nhằm kết nối trực tiếp với hệ thống phần cứng thông qua mạng Wi-Fi (UDP), cho phép hiển thị các thông số cảm biến theo thời gian thực, biểu diễn mức độ nghiêng, hướng chuyển động và lưu trữ dữ liệu khảo sát. Việc phát triển riêng một ứng dụng di động không những tăng tính chủ động và tương thích của hệ thống, mà còn giúp tối ưu trải nghiệm người dùng, đồng thời thể hiện rõ năng lực tích hợp đa nền tảng giữa phần cứng và phần mềm trong khuôn khổ đề tài.

Mục tiêu chính của đề tài là xây dựng một phiên bản mẫu của một hệ thống hoàn chỉnh, bao gồm cả phần cứng lẫn phần mềm, có thể áp dụng để đánh giá mặt phẳng địa hình trong các môi trường quy mô nhỏ đến trung bình, với chi phí thấp, tính linh hoạt cao và khả năng mở rộng trong tương lai.

## Định hướng giải pháp

Để giải quyết bài toán đánh giá mặt phẳng địa hình một cách tự động và trực quan, đề tài lựa chọn định hướng sử dụng công nghệ nhúng với vi điều khiển ESP32, kết hợp cảm biến đo lường MPU6050, kỹ thuật truyền thông không dây (UDP Wi-Fi), và ứng dụng hiển thị thời gian thực trên nền tảng Android. Dữ liệu đo được sẽ được xử lý thông qua các thuật toán tính toán góc nghiêng và độ dốc từ cảm biến gia tốc và con quay hồi chuyển, sau đó gửi về điện thoại để hiển thị và lưu trữ.

Giải pháp của đề tài là xây dựng một hệ thống gồm phần cứng cơ động (một xe robot mini điều khiển bằng ESP32, tích hợp MPU6050 và driver L298N để tự di chuyển), kết nối với ứng dụng Android qua mạng Wi-Fi. Hệ thống có thể thu thập dữ liệu cảm biến theo thời gian thực, tính toán mức độ nghiêng của mặt địa hình tại từng điểm, hiển thị đồ thị và đường đi khảo sát trực tiếp trên giao diện người dùng. Giải pháp này đảm bảo tính linh hoạt, chi phí thấp, và dễ triển khai thực tế.

Đóng góp chính của đề tài là xây dựng thành công một nguyên mẫu hệ thống đánh giá mặt phẳng địa hình tích hợp giữa phần cứng nhúng và phần mềm di động, có khả năng thu thập, xử lý và hiển thị dữ liệu trực quan. Hệ thống hỗ trợ đánh giá mặt bằng theo thời gian thực, đồng thời ghi nhận được dữ liệu để phục vụ phân tích sau khảo sát. Kết quả đạt được là hệ thống hoạt động ổn định, giao tiếp hiệu quả giữa phần cứng và phần mềm, hiển thị được mức độ nghiêng địa hình thông qua đồ thị và biểu diễn đường đi khảo sát một cách trực quan.

## Bố cục đồ án

Đồ án tốt nghiệp “Thiết kế, xây dựng hệ thống đánh giá mặt phẳng của địa hình” được tổ chức thành 6 chương chính (không kể mở đầu và tài liệu tham khảo), mỗi chương đảm nhận một vai trò riêng trong quá trình nghiên cứu, phát triển và đánh giá hệ thống. Cấu trúc được xây dựng theo logic từ khảo sát bài toán, nghiên cứu công nghệ, thiết kế – triển khai hệ thống, đến đánh giá kết quả và định hướng phát triển tương lai.

* **Chương 2: Khảo sát và phân tích yêu cầu**

Nếu như ở chương 1, chúng ta đã đặt ra vấn đề, nêu lên ý tưởng, thì chương 2 đóng vai trò đặt nền móng cho toàn bộ quá trình phát triển của đồ án. Trong chương này, tôi sẽ trình bày kết quả khảo sát hiện trạng thực tế và phân tích các yêu cầu của hệ thống dựa trên nhu cầu ứng dụng trong thực tế. Các yêu cầu chức năng và phi chức năng cần được xác định rõ ràng thông qua biểu đồ use-case và đặc tả nghiệp vụ. Việc xác định chính xác nhu cầu của bên khách hàng, người dùng và phạm vi chức năng giúp định hướng việc phát triển hệ thống đi theo đúng mục tiêu ban đầu đã đặt ra.

* **Chương 3: Công nghệ sử dụng**

Chương 3 giới thiệu và phân tích các công nghệ, thiết bị cả phần cứng và phần mềm được tôi lựa chọn để thiết kế, xây dựng hệ thống. Cụ thể, vi điều khiển ESP32 được chọn làm bộ vi xử lý trung tâm do đặc tính linh hoạt, dễ sử dụng và được hỗ trợ giao tiếp không dây (thông qua Wi-Fi/Bluetooth); Module MPU6050 được sử dụng để đo gia tốc và góc nghiêng; L298N được dùng để điều khiển chuyển động của toàn module; Bên cạnh đó là các công nghệ phát triển ứng dụng Android, cụ thể là ngôn ngữ lập trình Java, Firebase, các thư viện cho phép sử dụng giao thức truyền thông UDP, và cuối cùng là thư viện trực quan hóa dữ liệu. Việc lựa chọn công nghệ phù hợp vừa giúp hệ thống đảm bảo hiệu năng vửa có tính ứng dụng thực tế cao.

* **Chương 4: Phát triển và triển khai ứng dụng**

Chương 4 sẽ trình bày toàn bộ quá trình xây dựng hệ thống từ thiết kế kiến trúc phần mềm, thiết kế chi tiết giao diện và các lớp xử lý. Ngoài ra, chương 4 cũng mô tả chi tiết các thư viện và công cụ đã sử dụng, hình ảnh giao diện người dùng, kết quả thực thi các chức năng chính, cũng như các bước kiểm thử để đảm bảo tính ổn định, chính xác của hệ thống. Cuối cùng, quy trình triển khai hệ thống, chạy thử trên phần cứng và ứng dụng Android cũng được làm rõ nhằm chứng minh tính khả thi và khả năng áp dụng thực tế của giải pháp hệ thống này.

* **Chương 5: Các giải pháp và đóng góp nổi bật**

Chương 5 là nơi làm rõ những điểm mới, sáng tạo, đặc trưng, hoặc tối ưu mà đồ án có thể mang lại so với các hệ thống, giải pháp tương tự. Những cải tiến về thiết kế hệ thống, cách thức truyền dữ liệu thời gian thực, khả năng hiển thị trực quan ngay trên màn hình ứng dụng di động, tất cả hệ thống đều được thiết kế nhỏ gọn, và tính cơ động của module khảo sát và đánh giá. Đây sẽ là phần để thể hiện rõ vai trò chủ động và năng lực nghiên cứu, ứng dụng công nghệ của sinh viên trong suốt quá trình triển khai đề tài sau thời gian học tập tại Trường Đại học.

* **Chương 6: Kết luận và hướng phát triển**

Đây là chương cuối cùng, có vai trò tổng kết toàn bộ quá trình thực hiện đồ án, đánh giá mức độ hoàn thành so với mục tiêu ban đầu đề ra và rút ra bài học kinh nghiệm. Đồng thời, chương này mở ra các hướng phát triển khả dĩ trong tương lai như cải tiến phần cứng, tích hợp AI để phân tích, đánh giá dữ liệu sau khi đo đạc, hoặc đề xuất triển khai hệ thống trên quy mô đầu tư lớn hơn. Qua đó, khẳng định tính thực tiễn và khả năng mở rộng lâu dài của đề tài.

# ­­Khảo sát và phân tích yêu cầu

Sau khi đã đặt vấn đề và xác định mục tiêu của hệ thống trong chương 1, chương 2 sẽ tập trung vào việc khảo sát hiện trạng và phân tích yêu cầu đối với một hệ thống đánh giá mặt phẳng địa hình. Việc hiểu rõ nhu cầu thực tế và hành vi người dùng là cơ sở để xác định chính xác phạm vi chức năng mà hệ thống cần đáp ứng. Chương này sẽ lần lượt trình bày hiện trạng khảo sát, các chức năng chính thông qua biểu đồ use case tổng quan và phân rã, mô tả quy trình nghiệp vụ, đặc tả chi tiết các chức năng chính, và xác định yêu cầu phi chức năng. Những nội dung này là tiền đề để chuyển sang bước thiết kế hệ thống trong chương tiếp theo.

## Khảo sát hiện trạng

Trước khi bắt tay vào thiết kế, xây dựng hệ thống đánh giá mặt phẳng địa hình, tôi nhận thấy rằng việc khảo sát hiện trạng đóng vai trò quan trọng nhằm xác định rõ yêu cầu thực tế, các giải pháp đã và đang được triển khai, cũng như nhận diện các hạn chế còn tồn tại từ các giải pháp đó, từ đó đề xuất, thiết kế một hệ thống tối ưu nhất có thể.

Nguồn thông tin khảo sát được thu thập từ ba nhóm chính: Người dùng, khách hàng tiềm năng; Các sản phẩm, hệ thống, giải pháp đã có mặt trên thị trường; Các ứng dụng tương tự có liên quan đến đo đạc, giám sát và đánh giá mặt phẳng địa hình.

**Khảo sát từ người dùng/khách hàng tiềm năng**

Qua trao đổi với người quen là một kỹ sư xây dựng có nhiều năm kinh nghiệm trong nghề cùng với một số sinh viên chuyên ngành kỹ thuật công trình, xây dựng dân dụng và giao thông đường bộ, tôi đã rút ra được một nhu cầu chung cơ bản về hệ thống được đặt ra là một thiết bị đơn giản, dễ mang vác và sử dụng trong thực địa nhằm đánh giá sơ về độ nghiêng, độ dốc hoặc tính bằng phẳng của một khu vực. Ngoài ra, do giá thành của các thiết bị, giải pháp hiện có thường rất đắt đỏ, hoặc thường khó vận hành đối với người chưa có hoặc có ít kinh nghiệm. Nhu cầu thứ hai đặt ra là một hệ thống có giá thành rẻ, trực quan và có khả năng truyền dữ liệu đến thiết bị di động cá nhân nhằm thuận tiện, dễ dàng cho việc theo dõi, đánh giá mặt phẳng ở mức độ dân dụng.

**Khảo sát các hệ thống, giải pháp đang có trên thị trường**

Hiện nay, trên thị trường tồn tại nhiều hệ thống giám sát và đánh giá địa hình với mức độ hiện đại và độ chính xác cao. Dưới đây là một số hệ thống tiêu biểu:

* **Leica GeoMoS** – Đây là một hệ thống giám sát địa hình thời gian thực do Leica Geosystems (Thụy Sĩ) phát triển. Hệ thống sử dụng toàn đạc điện tử, GNSS và cảm biến nghiêng để đo đạc và cảnh báo các chuyển động địa chất. Tuy nhiên, chi phí triển khai cao và yêu cầu kỹ thuật phức tạp là những hạn chế khi áp dụng cho các mục đích phổ thông hoặc giáo dục.
* **Proxima Slope Monitoring** – Phát triển bởi Proxima Systems (Tây Ban Nha), hệ thống này sử dụng mạng cảm biến không dây để giám sát chuyển động địa hình. Mặc dù có khả năng truyền dữ liệu thời gian thực, song cấu hình hệ thống phức tạp và chi phí triển khai cao khiến nó ít phù hợp cho các dự án nhỏ hoặc học thuật.
* **Moasure ONE** – Là thiết bị cầm tay ứng dụng cảm biến quán tính (IMU), thiết bị này có tính di động cao, dễ sử dụng và phù hợp với người dùng phổ thông. Tuy nhiên, độ chính xác bị giới hạn bởi bản chất của cảm biến và không cho phép mở rộng hay tích hợp với các module điều khiển khác.
* **Pavemetrics LDTM** – Hệ thống này sử dụng laser để lập bản đồ địa hình bề mặt với độ phân giải cao, thích hợp cho các công trình giao thông hoặc khảo sát mặt đường. Đây là một giải pháp cực kỳ chính xác nhưng rất đắt đỏ và đòi hỏi điều kiện triển khai chuyên nghiệp.
* **SAGA GIS** – Là phần mềm mã nguồn mở phục vụ phân tích dữ liệu địa hình từ các bản đồ số hoặc dữ liệu cảm biến. Đây là công cụ mạnh mẽ cho các nhà nghiên cứu, tuy nhiên không phù hợp cho người dùng phổ thông do yêu cầu hiểu biết chuyên môn cao.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Hệ thống/Giải pháp | Tính chính xác | Thời gian thực | Di động | Giá thành | Sử dụng | Mở rộng |
| Leica GeoMoS | Rất cao | Có | Thấp | Rất cao | Khó | Không |
| Proxima Monitoring | Cao | Có | Trung bình | Cao | Trung bình | Hạn chế |
| Moasure ONE | Trung bình | Có | Cao | Trung bình | Dễ | Không |
| Pavemetrics LDTM | Rất cao | Có | Thấp | Rất cao | Khó | Không |
| SAGA GIS | Phụ thuộc dữ liệu | Có | Trung bình | Thấp | Khó | Có |

Bảng 1: Bảng so sánh tổng quan các hệ thống, giải pháp hiện có

**Đánh giá hiện trạng và yêu cầu của đồ án**

Từ những khảo sát và phân tích ở trên, ta có thể thấy rằng các hệ thống, giải pháp hiện tại đều có những ưu điểm nhất định, nhưng vẫn chưa có lựa chọn nào cân bằng được giữa tính chính xác, chi phí thấp, tính di động cao và dẽ dàng tích hợp mở rộng. Đặc biệt, chưa có hệ thống nào có tích hợp cảm biến con quay hồi chuyển đơn giản (như được trang bị trên module MPU6050) với vi điều khiển ESP32 và cho phép kết nối trực tiếp với thiết bị di động riêng của người dùng thông qua giao thức mạng nội bộ (UDP hoặc Bluetooth) phục vụ cho nhu cầu đo đạc địa hình cơ bản, ở mức dân dụng.

**Tính năng quan trọng cần phát triển trong đồ án**

Dựa vào khảo sát, hệ thống cần đáp ứng các tính năng quan trọng sau:

* Thu thập được dữ liệu cảm biến (gia tốc, con quay hồi chuyển) từ 2 module ESP32 và MPU6050.
* Tính toán và hiển thị tình trạng nghiêng, độ dốc bề mặt lên thiết bị di động theo thời gian thực.
* Giao diện ứng dụng di động trực quan, dễ sử dụng, hiển thị số liệu và biểu đồ để đánh giá, theo dõi.
* Hệ thống có thể truyền dữ liệu thời gian thực thông qua mạng nội bộ UDP.
* Khả năng mở rộng để lưu trữ dữ liệu phục vụ mục đích đánh giá trong tương lai.

## Tổng quan chức năng

Hệ thống đánh giá mặt phẳng địa hình được thiết kế nhằm cung cấp một giải pháp tích hợp giữa phần cứng và phần mềm, giúp người dùng dễ dàng khảo sát địa hình, theo dõi độ nghiêng, và điều khiển thiết bị di chuyển từ xa. Các chức năng được chia thành hai nhóm chính: chức năng phần cứng và chức năng phần mềm.

Về phần cứng, hệ thống sử dụng vi điều khiển ESP32 để thu thập dữ liệu cảm biến từ MPU6050 (cảm biến gia tốc và con quay hồi chuyển), điều khiển động cơ qua mạch L298N, và truyền dữ liệu về ứng dụng di động qua giao thức UDP. Về phần mềm, ứng dụng Android cung cấp giao diện người dùng trực quan, cho phép điều khiển thiết bị, nhận và hiển thị dữ liệu cảm biến, và hỗ trợ các chức năng hiển thị biểu đồ theo thời gian thực.

Các chức năng tổng quan của hệ thống bao gồm:

* **Điều khiển robot từ xa**: Cho phép người dùng gửi các lệnh di chuyển như tiến, lùi, rẽ trái/phải tới module robot thông qua kết nối không dây.
* **Thu thập dữ liệu cảm biến**: ESP32 đọc dữ liệu từ MPU6050 về gia tốc, góc nghiêng và nhiệt độ môi trường.
* **Hiển thị dữ liệu cảm biến thời gian thực**: Ứng dụng di động cập nhật liên tục giá trị cảm biến và hiển thị theo định dạng số hoặc biểu đồ.
* **Vẽ biểu đồ độ nghiêng theo thời gian**: Giúp người dùng đánh giá sự thay đổi mặt phẳng địa hình theo lộ trình di chuyển.
* **Lưu trữ và truy xuất dữ liệu khảo sát**: (chức năng mở rộng) – Cho phép lưu lại dữ liệu khảo sát để phân tích hoặc tái sử dụng sau này.

### Biểu đồ use case tổng quan

Dưới đây là biểu đồ Use Case tổng quan mô tả các chức năng chính và tác nhân tham gia trong hệ thống:

A diagram of a person with text

AI-generated content may be incorrect.

Hình 1: Use case tổng quan mô tả các chức năng chính của phần mềm hệ thống

**Tác nhân:**

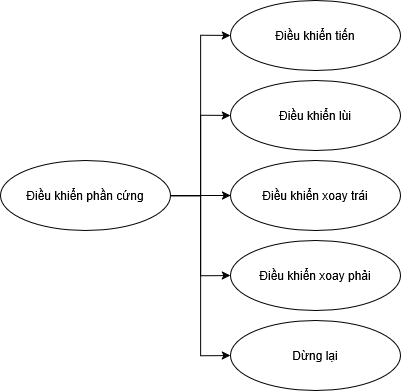
* **Người dùng**: Là người vận hành hệ thống. Có thể là kỹ sư khảo sát địa hình, sinh viên nghiên cứu, hoặc người dùng phổ thông, hoặc bất kỳ ai có nhu cầu khảo sát địa hình với bất kỳ mục đích gì. Vai trò là điều khiển thiết bị và giám sát thông tin.

**Các use case chính:**

* *Điều khiển phần cứng*: Gửi tín hiệu điều khiển tới robot.
* *Xem dữ liệu cảm biến*: Nhận và hiển thị dữ liệu từ cảm biến MPU6050.
* *Xem biểu đồ gia tốc và độ nghiêng*: Hiển thị diễn biến gia tốc và độ nghiêng qua thời gian.
* *Xem, xóa dữ liệu*: (Tùy chọn) Lưu trữ dữ liệu khảo sát lên database để phân tích sau. Hoặc xóa toàn bộ dữ liệu khi có nhu cầu.

### Biểu đồ use case phân rã

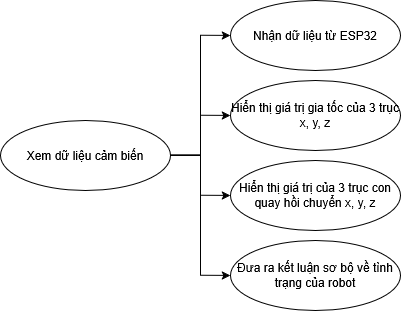
**Biểu đồ use case phân rã chức năng “Điều khiển phần cứng”**



Hình 2: Biểu đồ phân rã chức năng “Điều khiển phần cứng”

Chức năng điều khiển phần cứng bao gồm các thao tác điều hướng. Mỗi lệnh đều được gửi thông qua UDP tới ESP32, đảm bảo độ trễ thấp và đáp ứng thời gian thực. Mỗi hành động được thực hiện thông qua giao diện người dùng trên ứng dụng Android, sử dụng các nút điều hướng có biểu tượng mũi tên để tăng tính trực quan.

**Biểu đồ use case phân rã “Xem dữ liệu cảm biến”**

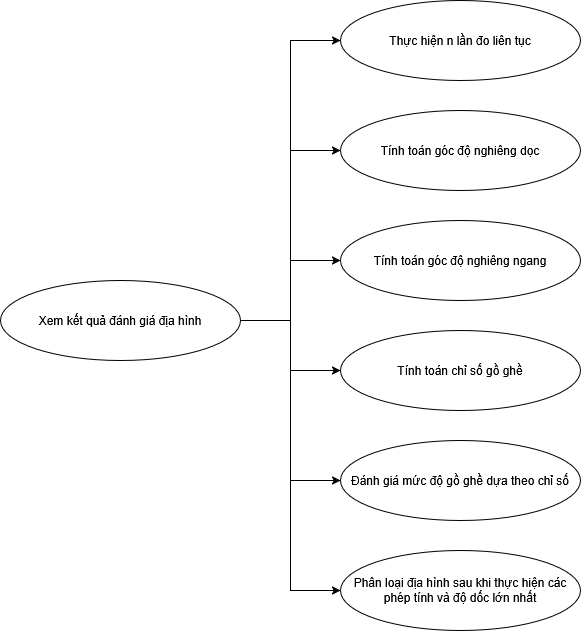
****

Hình 3: Biểu đồ phân rã chức năng “Xem dữ liệu cảm biến”

Biểu đồ này mô tả các thao tác chi tiết khi người dùng thực hiện việc hiển thị dữ liệu cảm biến thời gian thực và các hoạt động của phần mềm để phục vụ người dùng:

* Nhận dữ liệu từ ESP32: Ứng dụng kết nối với module thông qua UDP để nhận luồng dữ liệu cảm biến liên tục.
* Hiển thị giá trị gia tốc của 3 trục x, y, z: Dữ liệu gia tốc được đo qua 3 trục X, Y, Z được hiển thị bằng số (làm tròn đến số thập phân thứ 2).
* Hiển thị giá trị của 3 trục con quay hồi chuyển x, y, z: Dữ liệu được đo qua 3 trục X, Y, Z của con quay hồi chuyển giúp tính toán độ nghiêng.
* Đưa ra kết luận sơ bộ về tình trạng của robot: Kết quả đánh giá bằng chữ về tình trạng của robot (Nghiêng trước, Nghiêng sau, Nghiêng trái, Nghiêng phải, Độ nghiêng nguy hiểm).

**Biểu đồ use case phân rã chức năng “Xem kết quả đánh giá địa hình”**

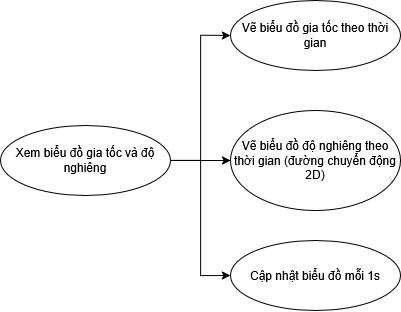
****

Hình 4: Biểu đồ phân rã chức năng “Xem kết quả đánh giá địa hình”

Biểu đồ này mô tả các thao tác chi tiết khi người dùng thực hiện việc đánh giá địa hình, và ứng dụng đưa ra kết luận về địa hình sau quá trình tính toán:

* Thực hiện *n* lần đo liên tục: Người dùng lựa chọn số lượt đo (Phạm vi 10; 25; 50; 100 lần liên tục) để hệ thống thực hiện đo đạc và tính toán.
* Tính toán góc độ nghiêng dọc: Tính góc nghiêng theo trục X (pitch) từ gia tốc kế. Thể hiện thiết bị nghiêng nhiều về trước hay sau.
* Tính toán góc độ nghiêng ngang: Tính góc nghiêng theo trục Y (roll) từ gia tốc kế. Thể hiện thiết bị nghiêng nhiều về trái hay phải.
* Tính toán chỉ số gồ ghề: Tính độ lệch chuẩn (standard deviation) của một danh sách các giá trị — ở đây được hiểu là mức dao động của dữ liệu, hay còn gọi là độ gồ ghề trong ứng dụng đánh giá địa hình.
* Đánh giá mức độ gồ ghề theo chỉ số: Đưa ra đánh giá bằng chữ về mức độ gồ ghề của địa hình thông qua chỉ số gồ ghề tính được ở trên. Giá trị có độ lệch chuẩn cao → Địa hình có nhiều dao động → gồ ghề. Và ngược lại, Giá trị có độ lệch chuẩn thấp → Địa hình có ít dao động → bằng phẳng.
* Phân loại địa hình sau khi thực hiện các phép tính và độ dốc lớn nhất: Phân loại địa hình cuối cùng dựa vào kết quả các phép tính ở trên và độ dốc lớn nhất đo được, cho người dùng cái nhìn tổng quan về loại địa hình (Dốc hay không dốc).

**Biểu đồ Use Case phân rã “Xem biểu đồ gia tốc và độ nghiêng”**

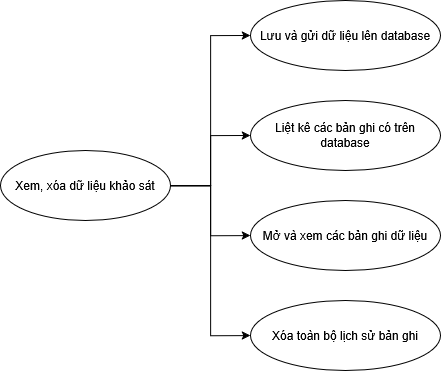


Hình 5: Biểu đồ phân rã chức năng “Xem biểu đồ gia tốc và độ nghiêng”

Biểu đồ này thể hiện các hành động liên quan đến việc hiển thị dữ liệu cảm biến dưới dạng biểu đồ trực quan:

* + Vẽ biểu đồ gia tốc theo thời gian: Sử dụng thư viện biểu đồ để trực quan hóa sự thay đổi gia tốc theo thời gian.
  + Vẽ biểu đồ độ nghiêng theo thời gian (đường chuyển động 2D): Biểu đồ hiển thị độ nghiêng của thiết bị, được trực quan hóa bằng một đường chuyển động 2D.
  + Cập nhật biểu đồ mỗi 1 giây: Hai biểu đồ trên được cập nhật theo chu kỳ là 1 giây/lần khi có dữ liệu mới được gửi về từ robot.

**Biểu đồ Use Case phân rã “Xem, xóa dữ liệu khảo sát”**

****

Hình 6: Biểu đồ phân rã chức năng “Xem, xóa dữ liệu khảo sát”

Biểu đồ này mô tả các thao tác liên quan đến quản lý dữ liệu đã lưu từ quá trình khảo sát trước đó:

* + Lưu và gửi dữ liệu lên database: Phần mềm lưu lại dữ liệu cảm biến và gửi dữ liệu đó dưới dạng một bản ghi lên database (ở đây sử dụng Firebase).
  + Liệt kê các bản ghi có trên database: Ứng dụng truy cập database và hiển thị danh sách các lượt đo riêng lẻ hoặc phiên đánh giá đã được lưu trữ.
  + Mở và xem các bản ghi dữ liệu: Cho phép người dùng mở danh sách liệt kê và xem lại các dữ liệu cảm biến hoặc các kết quả đánh giá.
  + Xóa toàn bộ lịch sử bản ghi: Cho phép người dùng xóa toàn bộ lịch sử bản ghi không còn sử dụng để giải phòng bộ nhớ.

### Quy trình nghiệp vụ

Hệ thống đánh giá mặt phẳng địa hình có hai quy trình nghiệp vụ quan trọng kết hợp nhiều chức năng liên quan giữa phần cứng và phần mềm nhằm đảm bảo người dùng có thể điều khiển, theo dõi dữ liệu và tra cứu thông tin khảo sát một cách dễ dàng. Các quy trình nghiệp vụ này bao gồm: (i) kết nối và vận hành hệ thống, và (ii) tra cứu và xem dữ liệu đo, kết quả đánh giá từ database.

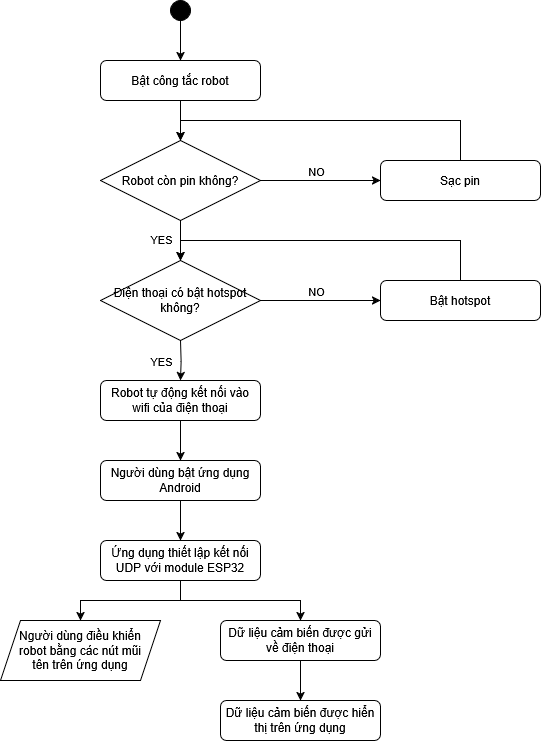
1. **Quy trình nghiệp vụ 1: Kết nối và vận hành hệ thống**

Quy trình này mô tả các bước người dùng cần thực hiện để khởi động hệ thống, kết nối module phần cứng (robot) với điện thoại di động, sau đó điều khiển và theo dõi dữ liệu cảm biến trong thời gian thực thông qua ứng dụng.

**Các bước chính trong quy trình:**

1. Người dùng bật nguồn robot (ESP32, driver L298N, cảm biến MPU6050).
2. Điện thoại di động bật chế độ phát Wi-Fi hotspot.
3. Robot tự động kết nối vào Wi-Fi của điện thoại.
4. Người dùng mở ứng dụng Android.
5. Ứng dụng thiết lập kết nối UDP với module ESP32.
6. Người dùng điều khiển robot di chuyển bằng các nút mũi tên trên ứng dụng.
7. Dữ liệu cảm biến được gửi về điện thoại và hiển thị trực tiếp.

**Biểu đồ hoạt động:**

****

Hình 7: Biểu đồ hoạt động của quy trình kết nối và vận hành robot

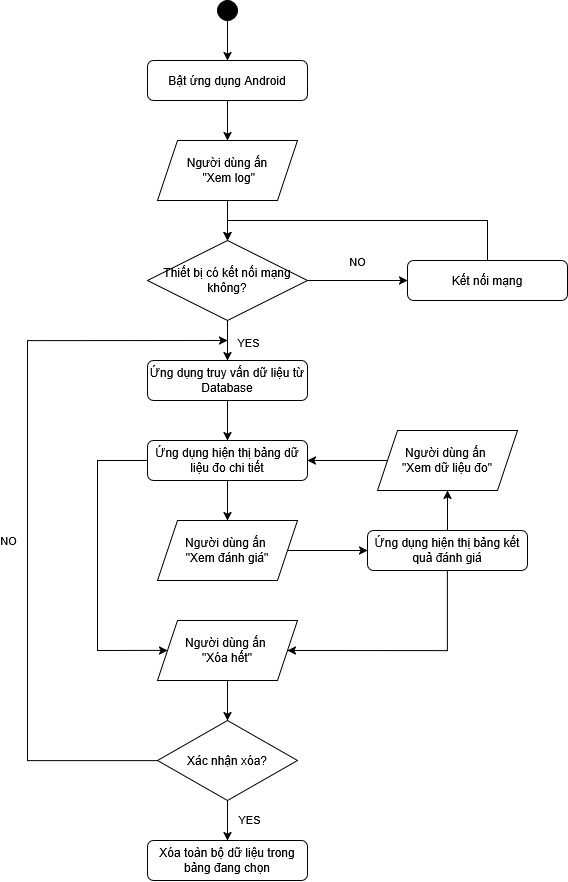
1. **Quy trình nghiệp vụ 2: Tra cứu và xem dữ liệu đo, kết quả đánh giá từ database**

Bất cứ khi nào điện thoại nhận được một chuỗi dữ liệu đo mới, hoặc sau khi hoàn thành một phiên khảo sát, điện thoại sẽ tải lên database một tập dữ liệu. Người dùng có thể xem lại dữ liệu đã lưu trên hệ thống thông qua chức năng “Xem log”. Dữ liệu lưu trên database bao gồm giá trị đo chi tiết từ cảm biến và kết quả đánh giá địa hình.

**Các bước chính trong quy trình:**

1. Người dùng mở ứng dụng và chọn chức năng “Xem log”.
2. Ứng dụng kiểm tra kết nối Internet (Wi-Fi hoặc 4G).
3. Nếu có mạng, ứng dụng truy vấn dữ liệu log từ Firebase.
4. Giao diện hiển thị hai bảng dữ liệu: “Xem đánh giá” và “Xem dữ liệu đo”.
5. Người dùng bấm chọn để chuyển qua lại giữa hai bảng này để xem các bản ghi cụ thể.
6. Người dùng có thể chọn xóa toàn bộ dữ liệu của một trong hai bảng nếu không cần thiết.

**Biểu đồ hoạt động:**



Hình 8: Biểu đồ hoạt động của quy trình tra cứu và xem dữ liệu đo, kết quả đánh giá từ database

## Đặc tả chức năng

### Đặc tả use case: Điều khiển phần cứng

* Mô tả: Người dùng sử dụng giao diện trên ứng dụng Android để gửi lệnh điều khiển từ xa đến module phần cứng (ESP32), nhằm điều khiển hướng di chuyển của robot (tiến, lùi, trái, phải) hoặc dừng lại. Tín hiệu điều khiển được truyền thông qua giao thức UDP.
* Tiền điều kiện:
  + Phần cứng được bật nguồn, không bị hết pin, và đã được kết nối với Wi-Fi nội bộ (hotspot).
  + Ứng dụng đã được bật và đang ở màn hình chính (MainActivity).
* Luồng sự kiện chính:

1. Người dùng bật nguồn module phần cứng và đảm bảo robot đã kết nối với thiết bị di động thông qua mạng nội bộ (hotspot).
2. Người dùng khởi động ứng dụng.
3. Người dùng nhấn một trong các nút điều khiển: Tiến, Lùi, Trái, Phải.
4. Ứng dụng gửi gói tin UDP tương ứng với lệnh điều khiển ("F", "B", "L", "R") đến địa chỉ IP của phần cứng.
5. Module phần cứng nhận tín hiệu và điều khiển động cơ theo hướng được yêu cầu.
6. Khi người dùng nhả nút điều khiển, ứng dụng gửi lệnh dừng ("S") đến phần cứng.
7. Phần cứng dừng chuyển động.

* Luồng sự kiện phát sinh:
  + Nếu module phần cứng chưa kết nối đến hotspot của thiết bị di động: Người dùng không thể điều khiển được robot.
  + Nếu gửi UDP thất bại (lỗi mạng, sai port, địa chỉ IP hoặc kích thước của packet vượt quá định nghĩa): Người dùng không thể điều khiển được robot, hoặc điều khiển được một chút trước khi robot dừng lại.
  + Nếu phần cứng không phản hồi, gặp lỗi (động cơ hỏng) hay hết pin: Robot không di chuyển. Trong trường hợp hết pin đèn led trên các module phần cứng sẽ tắt.
* Hậu điều kiện
  + Robot thực hiện đúng hướng di chuyển theo lệnh điều khiển của người dùng.
  + Robot dừng di chuyển sau khi người dùng nhả nút.

### Đặc tả use case: Xem dữ liệu cảm biến

* Mô tả: Cho phép người dùng theo dõi dữ liệu cảm biến (gia tốc, con quay hồi chuyển) theo thời gian thực trên ứng dụng di động.
* Tiền điều kiện:
  + Robot đã được bật và kết nối với thiết bị di động qua mạng Wi-Fi nội bộ (hotspot).
  + Ứng dụng đã thiết lập kết nối UDP với ESP32.
* Luồng sự kiện chính:
  + 1. Người dùng khởi động ứng dụng.
    2. Ứng dụng tự động thiết lập kết nối với module ESP32.
    3. Dữ liệu cảm biến được gửi từ ESP32 qua UDP đến thiết bị di động.
    4. Ứng dụng thu thập, hiển thị giá trị gia tốc (3 trục x, y, z) và giá trị con quay hồi chuyển (3 trục x, y, z) theo thời gian thục
    5. Ứng dụng xử lý các giá trị của cảm biến và đưa ra đánh giá bằng chữ cho trạng thái hiện tại của robot.
* Luồng sự kiện phát sinh:
  + Nếu ESP32 không kết nối với thiết bị di động: Ứng dụng hiển thị “N/A” ở các dòng ghi chỉ số, đánh giá. Hai biểu đồ gia tốc theo thời gian và đường chuyển động 2D không có đường biểu thị và không tuyến tính theo thời gian. Người dùng vẫn có thể ấn vào “Xem log” và xem dữ liệu từ database nếu thiết bị di động có kết nối mạng. Người dùng có thể ấn “Bắt đầu đánh giá” nhưng ứng dụng sẽ dừng ở: “Đang đánh giá… (0/*n*)” (*n* là số lượt đo đã lựa chọn).
  + Nếu mất kết nối đột ngột: Ứng dụng hiển thị chỉ số, đánh giá, trạng thái biểu đồ cuối cùng trước khi mất kết nối. Người dùng vẫn có thể thực hiện các chức năng khác như khi ESP32 không kết nối với thiết bị di động.
* Hậu điều kiện:
  + Người dùng theo dõi được trạng thái bề mặt địa hình ngay lập tức.
  + Dữ liệu đo được tự động lưu và tải lên database.

### Đặc tả use case: Xem biểu đồ gia tốc và con quay hồi chuyển

* **Mô tả**: Hiển thị biểu đồ trực quan theo thời gian thực dựa trên dữ liệu cảm biến thu thập từ module.
* **Tiền điều kiện**:
  + Robot đã được bật và kết nối với thiết bị di động qua mạng Wi-Fi nội bộ (hotspot).
  + Ứng dụng đã thiết lập kết nối UDP với ESP32.
* Luồng sự kiện chính:
  + 1. Người dùng khởi động ứng dụng.
    2. Ứng dụng tự động thiết lập kết nối với module ESP32.
    3. Dữ liệu cảm biến được gửi từ ESP32 qua UDP đến thiết bị di động.
    4. Ứng dụng thu thập dữ liệu thu thập và tiến hành xử lý.
    5. Ứng dụng hiển thị biểu đồ gia tốc theo thời gian và đường chuyển động 2D.
* Luồng sự kiện phát sinh:
  + Nếu robot di chuyển quá nhanh: Ứng dụng chỉ thực hiện thu thập, biểu diễn dữ liệu mỗi 1 giây và loại bỏ những điểm dữ liệu trong khoảng 1 giây đó.
  + Nếu module cảm biến gửi dữ liệu lỗi: Ứng dụng thông báo lỗi “Skipped invalid motion path point: + dữ liệu skip” trong logcat của Android Studio với filter: “MotionChart”
  + Nếu module cảm biến gửi dữ liệu bất hợp pháp: Trong một vài trường hợp, MPU6050 có thể gửi dữ liệu ‘null’ hoặc một mảng bị hỏng, có thể dẫn đến văng ứng dụng
* Hậu điều kiện:
  + Người dùng quan sát được xu hướng thay đổi của địa hình (dao động, độ nghiêng).
  + Biểu đồ có thể được chụp ảnh màn hình để lưu trữ hoặc đính kèm vào báo cáo khảo sát.

### Đặc tả use case: Xem kết quả đánh giá địa hình

* Mô tả: Người dùng yêu cầu hệ thống thực hiện một chuỗi phép đo liên tục trong thời gian thực để thu thập dữ liệu cảm biến từ thiết bị, tính toán các thông số góc nghiêng và độ gồ ghề, sau đó hiển thị đánh giá tổng quan về loại địa hình.
* **Tiền điều kiện**:
  + Robot đã được bật và kết nối với thiết bị di động qua mạng Wi-Fi nội bộ (hotspot).
  + Ứng dụng đã thiết lập kết nối UDP với ESP32.
* Luồng sự kiện chính:
  + 1. Người dùng khởi động ứng dụng.
    2. Ứng dụng tự động thiết lập kết nối với module ESP32.
    3. Người dùng lựa chọn phạm vi đánh giá (số lần đo để đưa ra đánh giá)
    4. Người dùng ấn “Bắt đầu đánh giá” để bắt đầu phiên đánh giá mới
    5. Dữ liệu cảm biến được gửi từ ESP32 qua UDP đến thiết bị di động. Ứng dụng thu thập dữ liệu này.
    6. Sau khi phiên đánh giá hoàn thành, ứng dụng thực hiện:
       - Tính góc nghiêng dọc (pitch) từ trục X.
       - Tính góc nghiêng ngang (roll) từ trục Y.
       - Tính chỉ số gồ ghề (standard deviation).
       - Đánh giá mức độ gồ ghề thông qua quy tắc phân loại (gồ ghề cao, vừa, thấp).
       - Phân loại địa hình tổng quát dựa vào độ nghiêng lớn nhất và chỉ số gồ ghề.
    7. Ứng dụng hiển thị kết quả đánh giá lên giao diện.
* Luồng sự kiện phát sinh
  + Nếu người dùng không lựa chọn phạm vi đánh giá: Ứng dụng tự động sử dụng giá trị mặc định là 10 lượt đo.
  + Nếu xảy ra lỗi trong hệ thống (Mất kết nối với ESP32): Ứng dụng tạm dừng đánh giá và chờ đến khi tái kết nối để tiếp tục đánh giá.
  + Nếu người dùng ấn “Hủy” khi đánh giá chưa hoàn tất: Ứng dụng hủy và xóa phiên đánh giá, hiển thị lại kết quả đánh giá cuối cùng trên giao diện.
* **Hậu điều kiện**
  + Hệ thống hiển thị bảng kết quả đánh giá địa hình gồm: góc nghiêng lớn nhất, chỉ số gồ ghề, đánh giá mức độ gồ ghề (bằng chữ), và phân loại địa hình cuối cùng (ví dụ: “Dốc nhẹ”, "Bằng phẳng",...).
  + Hệ thống sẽ lưu lại kết quả đánh giá và người dùng có thể lựa chọn thực hiện phiên đánh giá mới.

### Đặc tả use case: Xem, xóa dữ liệu khảo sát

* Mô tả: Người dùng xem lại các bản ghi dữ liệu từ các lượt đo và đợt khảo sát trước và có thể xóa nếu không còn cần thiết.
* Tiền điều kiện:
  + Người dùng khởi động ứng dụng.
  + Thiết bị di động có kết nối mạng (Wi-Fi hoặc mạng di động).
  + Hệ thống đã thực hiện đo đạc, đánh giá trước đó.
* Luồng sự kiện chính:

1. Người dùng khởi động ứng dụng.
2. Người dùng chọn “Xem log”.
3. Danh sách bản ghi hiện ra với đầy đủ đầu mục, thời gian ghi.
4. Người dùng có thể chọn giữa “Xem đánh giá” và “Xem dữ liệu đo” để chuyển đổi giữa hai bảng cơ sở dữ liệu đánh giá và dữ liệu đo.
5. Người dùng có thể chọn “Xóa hết” để xóa toàn bộ dữ liệu trong bảng cơ sở dữ liệu đang được chọn

* Luồng sự kiện phát sinh:
  + Nếu không có bảng ghi nào: Ứng dụng chỉ hiện các đầu mục mà không có bảng dữ liệu ở dưới.
  + Nếu xóa thất bại (mất kết nối, kết nối không ổn định): Ứng dụng không có thông báo gì, nhưng trong logcat của Android Studio sẽ thông báo “Delete Failed” dưới filter “Firebase”.
* Hậu điều kiện:
  + Người dùng xem được dữ liệu lịch sử và quản lý lưu trữ hiệu quả hơn.
  + Bộ nhớ ứng dụng được giải phóng nếu người dùng xóa dữ liệu cũ.

## Yêu cầu phi chức năng

Bên cạnh các yêu cầu chức năng đã được liệt kê, hệ thống còn cần đáp ứng các yêu cầu phi chức năng nhằm đảm bảo hiệu quả vận hành, tính ổn định và khả năng mở rộng. Các yêu cầu phi chức năng quan trọng của hệ thống bao gồm:

### Hiệu năng (Performance)

* Hệ thống cần phản hồi lệnh điều khiển của người dùng trong thời gian ngắn, tối đa **< 100ms** sau khi người dùng thao tác.
* Dữ liệu cảm biến từ thiết bị phần cứng gửi về điện thoại phải được cập nhật **theo thời gian thực**, với chu kỳ là 1000**ms/lần**.
* Quá trình đánh giá địa hình (thực hiện đo *n* lần và tính toán) phải hoàn tất trong vòng ***n* giây** với *n* lần đo.

### Độ tin cậy (Reliability)

* Dữ liệu truyền giữa điện thoại và phần cứng phải đảm bảo độ chính xác, không bị mất gói hoặc trễ không kiểm soát.
* Khi mất kết nối Wi-Fi giữa phần mềm và phần cứng, hệ thống phải phát hiện và cảnh báo người dùng.
* Các kết quả đánh giá phải được tính toán đúng với độ chính xác cao, không gây ra lỗi logic trong phân loại địa hình.

### Tính dễ sử dụng (Usability)

* Giao diện người dùng đơn giản, trực quan, dễ thao tác kể cả với người dùng không chuyên về kỹ thuật.
* Các chức năng điều khiển và xem dữ liệu cần được trình bày rõ ràng, với bố cục hợp lý (nút điều khiển lớn, thông tin cảm biến dễ đọc).
* Ứng dụng Android cần hỗ trợ hiển thị tốt ở chế độ nằm ngang (landscape) để phù hợp với thao tác điều khiển.

### Tính dễ bảo trì (Maintainability)

* Mã nguồn phần mềm được tổ chức theo hướng **module hóa** (chia theo điều khiển, xử lý dữ liệu, hiển thị, ghi log…) để dễ dàng bảo trì hoặc nâng cấp sau này.
* Hệ thống sử dụng công nghệ phổ biến và có tài liệu đầy đủ để thuận tiện cho các nhà phát triển tiếp theo.

### Các yêu cầu kỹ thuật khác

|  |  |
| --- | --- |
| Yếu tố kỹ thuật | Mô tả |
| Kết nối mạng | |  | | --- | |  |  |  | | --- | | Điện thoại Android phát Wi-Fi nội bộ, ESP32 kết nối trực tiếp để điều khiển và nhận dữ liệu. | |
| Cơ sở dữ liệu | Sử dụng **Firebase Realtime Database** để lưu lại log đánh giá và dữ liệu cảm biến. |
| Công nghệ sử dụng | * + - Ứng dụng viết bằng **Java cho Android**.     - Giao tiếp qua giao thức UDP.     - Phần cứng sử dụng ESP32 và MPU6050 để đo lường chuyển động và độ nghiêng |

Bảng 2: Các yêu cầu kỹ thuật khác của hệ thống

# Công nghệ sử dụng

Hệ thống đánh giá mặt phẳng địa hình được phát triển dựa trên sự tích hợp của nhiều công nghệ phần mềm và phần cứng. Trong chương này, các công nghệ được lựa chọn sẽ được trình bày một cách khái quát, đồng thời phân tích vai trò và lý do lựa chọn mỗi công nghệ nhằm giải quyết những yêu cầu đặt ra ở chương 2. Bên cạnh đó, chương này cũng trình bày một số giải pháp thay thế và lý do không sử dụng.

## Lập trình nhúng với ngôn ngữ C trên ESP32

**Giới thiệu về lập trình nhúng và vi điều khiển ESP32**

Lập trình nhúng là quá trình viết phần mềm điều khiển phần cứng, được nhúng trực tiếp vào các thiết bị điện tử để thực hiện các chức năng chuyên biệt. Trong đồ án này, vi điều khiển được sử dụng là ESP32, một dòng chip của hãng Espressif Systems, nổi bật với khả năng tích hợp Wi-Fi, Bluetooth và hiệu năng xử lý mạnh mẽ so với chi phí thấp.

ESP32 có hai nhân xử lý Xtensa LX6, hỗ trợ RTOS, có khả năng thực hiện các tác vụ song song. Nhờ đó, thiết bị có thể vừa xử lý dữ liệu từ cảm biến, vừa truyền nhận dữ liệu qua mạng một cách mượt mà.

**Vai trò của lập trình nhúng trong hệ thống**

Trong hệ thống đánh giá mặt phẳng địa hình, vi điều khiển ESP32 đóng vai trò trung tâm, bộ não, nó đảm nhận các chức năng chính sau:

* Giao tiếp với cảm biến MPU6050 qua giao thức I2C để lấy dữ liệu gia tốc và con quay hồi chuyển (gyroscope).
* Xử lý dữ liệu thô: Xử lý dữ liệu thô từ các thanh ghi của cảm biến, sau đó gửi chuỗi dữ liệu là cho ứng dụng di động để tính toán góc nghiêng dọc (pitch), ngang (roll), và phục vụ việc đánh giá mức độ gồ ghề của địa hình.
* Nhận lệnh điều khiển động cơ từ thiết bị di động thông qua giao thức UDP.
* Điều khiển động cơ DC thông qua **driver L298N** (không sử dụng PWM, chỉ điều khiển logic HIGH/LOW).
* Gửi dữ liệu cảm biến theo thời gian thực về ứng dụng Android để hiển thị biểu đồ và lưu trữ log.

**Cấu trúc chương trình nhúng**

Chương trình được lập trình bằng ngôn ngữ **C** thông qua môi trường **ESP-IDF** (Espressif IoT Development Framework) hoặc Arduino IDE tùy giai đoạn phát triển. Tổng thể chương trình sử dụng các thư viện chính:

|  |  |
| --- | --- |
| Header | Chức năng chính |
| *WiFi.h* | Khởi tạo chức năng kết nối Wi-Fi cho ESP32 |
| *WiFiUdp*.h | Thiết lập UDP server, nhận lệnh từ app, gửi dữ liệu phản hồi. |
| *Wire*.h | Khởi tạo các cổng I2C để cho phép kết nối. |
| *Adafruit\_Sensor*.h | Thư viện bao gồm các thư viện con khác để hỗ trợ khởi tạo MPU6050. |
| *Adafruit\_MPU6050.h* | Thư viện để khởi tạo module MPU6050. |

Bảng 3: Các thư viện được sử dụng để lập trình ESP32

**Cảm biến MPU6050**

MPU6050 là cảm biến tích hợp gia tốc kế 3 trục và con quay hồi chuyển 3 trục, sử dụng chuẩn truyền thông I2C. Trong chương trình nhúng, ESP32 gửi tín hiệu I2C để truy xuất các thanh ghi của cảm biến, lấy ra dữ liệu dạng thô (raw data) từ các trục X, Y, Z. Dữ liệu sau đó được chuyển đổi sang đơn vị vật lý, phục vụ cho các phép tính góc nghiêng và độ lệch chuẩn.

**Chi tiết các giao thức của cảm biến MPU6050**

Cảm biến MPU6050 giao tiếp qua chuẩn I2C (GPIO21 – SDA, GPIO22 – SCL). ESP32 truy xuất các thanh ghi để đọc dữ liệu thô từ 6 trục (3 gia tốc, 3 gyro), từ đó tính toán các giá trị góc nghiêng và mức độ rung/lệch.

Giao tiếp trên bus I2C bắt đầu khi master đặt điều kiện START (S) trên bus, được định nghĩa là chuyển đổi HIGH sang LOW của đường SDA trong khi đường SCL ở mức HIGH (xem hình bên dưới). Bus được coi là bận cho đến khi master đặt điều kiện STOP (P) trên bus, được định nghĩa là chuyển đổi LOW sang HIGH trên đường SDA trong khi SCL ở mức HIGH (xem hình bên dưới).

A black line drawing of a pair of glasses

AI-generated content may be incorrect.

Hình 9: Điều kiện START và STOP của MPU6050

Các byte dữ liệu I2C được định nghĩa là dài 8 bit. Không có giới hạn về số byte được truyền cho mỗi lần truyền dữ liệu. Mỗi byte được truyền phải được theo sau bởi một tín hiệu xác nhận (ACK). Xung cho tín hiệu xác nhận được tạo ra bởi master, trong khi bộ thu tạo ra tín hiệu xác nhận thực tế bằng cách kéo SDA xuống và giữ nó ở mức thấp trong phần CAO của xung đồng hồ xác nhận.

Nếu một slave đang bận và không thể truyền hoặc nhận một byte dữ liệu khác cho đến khi một số tác vụ khác được thực hiện, nó có thể giữ SCL THẤP, do đó buộc master vào trạng thái chờ. Truyền dữ liệu bình thường sẽ tiếp tục khi slave đã sẵn sàng và giải phóng xung (xem hình bên dưới).

A diagram of a bus

AI-generated content may be incorrect.

Hình 10: Giao thức định dạng dữ liệu/ Xác nhận trên I2C của MPU6050

Sau khi bắt đầu giao tiếp với điều kiện BẮT ĐẦU (S), thiết bị chủ gửi một địa chỉ slave 7 bit theo sau là bit thứ 8, bit đọc/ghi. Bit đọc/ghi cho biết thiết bị chủ đang nhận dữ liệu từ hay đang ghi vào thiết bị slave. Sau đó, thiết bị chủ giải phóng đường SDA và chờ tín hiệu xác nhận (ACK) từ thiết bị slave. Mỗi byte được truyền phải theo sau bởi một bit xác nhận. Để xác nhận, thiết bị slave kéo đường SDA THẤP và giữ ở THẤP trong khoảng thời gian cao của đường SCL. Truyền dữ liệu luôn được thiết bị chủ kết thúc bằng điều kiện DỪNG (P), do đó giải phóng đường truyền giao tiếp. Tuy nhiên, thiết bị chủ có thể tạo điều kiện BẮT ĐẦU lặp lại (Sr) và định địa chỉ cho một slave khác mà không cần tạo điều kiện DỪNG (P) trước. Chuyển đổi THẤP sang CAO trên đường SDA trong khi SCL CAO xác định điều kiện dừng. Tất cả các thay đổi SDA phải diễn ra khi SCL ở mức thấp, ngoại trừ các điều kiện bắt đầu và dừng (xem hình bên dưới).

A diagram of data transfer

AI-generated content may be incorrect.

Hình 11: Ví dụ một chu kỳ truyển dữ liệu đầy đủ của MPU6050

* + - * + Dữ liệu sau khi được truyền và đọc sẽ được chuyển đổi sang dạng có thể sử dụng (float) bằng cách chia theo hệ số từ datasheet.

**Điều khiển động cơ qua L298N**

Hệ thống sử dụng driver L298N để điều khiển 2 động cơ DC, tuy nhiên không sử dụng PWM để điều chỉnh tốc độ. Thay vào đó, các jumper ENA và ENB được nối trực tiếp (đóng), khiến động cơ luôn hoạt động ở tốc độ mặc định tối đa khi có tín hiệu điều khiển từ ESP32.

Việc điều khiển chỉ bao gồm thay đổi chiều quay (tiến/lùi, trái/phải) thông qua 4 chân điều khiển IN1–IN4 kết nối với GPIO của ESP32. Trong đó, IN1 và IN2 điều khiển bánh xe bên phải; IN3 và IN4 điều khiển bánh xe bên trái. Khi người dùng nhấn nút điều khiển trên ứng dụng, một lệnh UDP được gửi tới ESP32 để thay đổi các chân điều khiển tương ứng.

Ví dụ về code điều khiển ‘tiến lên’:

void goAhead() {

  digitalWrite(IN\_1, HIGH);

  digitalWrite(IN\_2, LOW);

  digitalWrite(IN\_3, HIGH);

  digitalWrite(IN\_4, LOW);

}

Ở đây ta thấy logic của IN1 và IN3 được thiết lập ở HIGH còn IN2 và IN3 được thiết lập ở LOW. Tổ hợp này đảm bảo cả 2 bên bánh xe đều quay tiến lên phía trước.

**Truyền nhận dữ liệu qua giao thức UDP**

ESP32 thiết lập một **UDP server** trên Wi-Fi nội bộ để nhận lệnh điều khiển và gửi lại dữ liệu cảm biến. Gói tin lệnh là ký tự đơn (F, B, L, R, S) để chỉ các hướng. Gói phản hồi gồm các thông số cảm biến ở định dạng chuỗi.

Ví dụ chuỗi gửi đi:

"Acceleration: X=0.77, Y=-0.51, Z=9.45, Gyro X=-0.01, Y=0.01, Z=-0.02"

Dữ liệu được gửi theo chu kỳ cố định khoảng **1000ms/lần**, đảm bảo cập nhật liên tục và kịp thời trên ứng dụng.

\*QUAN TRỌNG: Chuỗi gửi đi phải trùng khớp 100% với mẫu như trên, không được thừa, thiếu, viết sai in hoa bất cứ chữ cái, số, dấu cách, nào. Ngoại lệ duy nhất là giá trị của các trục có thể là âm hoặc dương. Nếu dữ liệu từ ESP32 gửi đi không trùng khớp với chuỗi này, ứng dụng di động sẽ không nhận được đúng định dạng chuỗi được yêu cầu, và từ đó không để thực hiện các tính năng khác.

**Xử lý và đánh giá dữ liệu địa hình**

ESP32 và ứng dụng di động sẽ phối hợp thực hiện đánh giá địa hình khi nhận được yêu cầu từ ứng dụng, khi nhận được yêu cầu đánh giá, ESP32 sẽ thực hiện đo liên tục *n* lần (*n* là phạm vi đo yêu cầu bởi ứng dụng di động), và sau đó gửi lại *n* kết quả đo để ứng dụng di động thực hiện tính toán.

**Ưu điểm của ESP32**

|  |  |
| --- | --- |
| Tiêu chí | Đặc điểm phù hợp với đồ án |
| Kết nối Wi-Fi | Tạo mạng nội bộ, điều khiển không phụ thuộc Internet |
| Dễ lập trình | Có thể lập trình bằng ESP-IDF hoặc Arduino IDE |
| Hiệu năng ổn định | Đủ mạnh để xử lý đồng thời nhiều nhiệm vụ |
| Tài liệu hỗ trợ | Cộng đồng lập trình ESP32 đông đảo, tài liệu phong phú |

Bảng 4: Ưu điểm của ESP32

**So sánh với các vi điều khiển khác**

|  |  |  |
| --- | --- | --- |
| Nền tảng | Ưu điểm | Nhược điểm |
| Arduino UNO | Dễ dùng, cộng đồng lớn | Không có Wi-Fi, ít RAM |
| STM32 | Tốc độ cao, ổn định | Cần hiểu sâu về cấu trúc phần cứng |
| ESP8266 | Có Wi-Fi, giá rẻ | Ít GPIO, không mạnh bằng ESP32 |

Bảng 5: So sánh với các vi điều khiển phổ biến khác

ESP32 là sự lựa chọn hợp lý nhất cho hệ thống yêu cầu kết nối không dây, xử lý dữ liệu cảm biến và điều khiển phần cứng ở mức đơn giản mà vẫn cần sự ổn định.

## Lập trình thiết bị di động với Java

**Giới thiệu**

Trong hệ thống đánh giá mặt phẳng địa hình, ứng dụng điều khiển di động đóng vai trò là giao diện người dùng chính, đồng thời là trung tâm điều phối giao tiếp giữa người dùng và thiết bị phần cứng (ESP32). Ứng dụng được lập trình bằng **ngôn ngữ Java** trên nền tảng **Android SDK**, hướng tới khả năng thao tác đơn giản, hiển thị dữ liệu trực quan và phản hồi thời gian thực.

Việc lựa chọn nền tảng Android là phù hợp với mục tiêu sử dụng phổ biến trên điện thoại thông minh, không cần thêm thiết bị chuyên dụng, đồng thời dễ dàng triển khai trên diện rộng. Java là ngôn ngữ chính thức được Android hỗ trợ lâu đời, có cộng đồng phát triển lớn và hệ sinh thái thư viện phong phú.

**Vai trò của ứng dụng Android**

Ứng dụng điều khiển trên điện thoại Android có các chức năng chính sau:

* Tạo giao diện điều khiển robot: các nút bấm tiến, lùi, trái, phải, hiển thị dữ liệu cảm biến và đánh giá địa hình.
* Gửi lệnh UDP tới ESP32 theo yêu cầu người dùng.
* Nhận và hiển thị dữ liệu cảm biến từ ESP32 theo thời gian thực.
* Trực quan hóa dữ liệu bằng biểu đồ (MPAndroidChart).
* Ghi log dữ liệu cảm biến và kết quả đánh giá vào cơ sở dữ liệu Firebase.

**Kiến trúc ứng dụng**

Ứng dụng được phát triển theo mô hình **hoạt động đơn MainActivity** kết hợp các lớp phụ trợ. Để đảm bảo khả năng mở rộng và bảo trì, mã nguồn được chia thành các lớp (class) riêng biệt cho từng chức năng:

|  |  |
| --- | --- |
| Thành phần | Vai trò chính |
| Hello.java | Màn hình khởi động của ứng dụng, chứa logo Phenikaa. Sau một lúc tự động chuyển sang màn hình giao diện chính. |
| MainActivity.java | Giao diện chính, xử lý tất cả các lớp con khi được gọi, xử lý các phương thức xử lý vòng đời (lifecycle methods hoặc callback methods) của toàn bộ ứng dụng. |
| SensorLogActivity.java | Giao diện truy vấn và xem dữ liệu đã được lưu trên cơ sở dữ liệu. Có phương thức để xóa toàn bộ một bảng trong cơ sở dữ liệu. |
| UDPService.java | Gửi/nhận dữ liệu UDP giữa app và ESP32. |
| UIUpdater.java | Xử lý, tính toán, hiển thị và cập nhật dữ liệu hiển thị trên giao diện chính mỗi 1 giây, bao gồm dữ liệu cảm biến, trạng thái robot, kết quả đánh giá, biểu đồ gia tốc và biểu đồ đường chuyển động 2D. |
| ButtonController.java | Xử lý các nút bấm của ứng dụng, bao gồm các nút mũi tên điều khiển robot, ‘Xem log’, ‘Bắt đầu đánh giá’ và ‘Hủy’. |
| Logger.java | Lưu dữ liệu lên Firebase Realtime Database. |
| MusicService.java | Đây hoàn toàn là một thành phần sinh viên tự ý thêm vào nhằm mục đích học hỏi, có thể được vô hiệu hóa rất đơn giản. Thành phần này cho phép ứng dụng chơi nhạc ngẫu nhiên trên nền, kèm theo hiệu ứng chuyển tiếp bài hát không độ trễ, trơn tru. |

Bảng 6: Danh sách các lớp của ứng dụng di động

Ứng dụng được thiết kế ở chế độ **nằm ngang (landscape)** để tối ưu không gian thao tác khi cầm ngang điện thoại và điều khiển robot từ xa.

**Giao tiếp UDP**

Giao thức UDP (User Datagram Protocol) được lựa chọn cho việc trao đổi dữ liệu giữa ứng dụng và ESP32 vì:

* UDP có độ trễ thấp, phù hợp với yêu cầu điều khiển thời gian thực.
* Giao tiếp đơn giản, không yêu cầu kết nối phức tạp như TCP.
* Phù hợp với mô hình mạng nội bộ giữa ESP32 và điện thoại.

Ứng dụng tạo socket UDP để gửi các lệnh như "F", "B", "L", "R" (điều khiển chuyển động), "S" (dừng), hoặc "A" (bắt đầu đánh giá). Đồng thời, ứng dụng mở cổng UDP để nhận dữ liệu dạng chuỗi từ ESP32, chứa giá trị gia tốc, con quay hồi chuyển.

**Hiển thị dữ liệu cảm biến**

Dữ liệu cảm biến từ ESP32 bao gồm gia tốc (X, Y, Z), tốc độ quay (gyro), nhiệt độ. Ứng dụng phân tích chuỗi phản hồi và cập nhật các thành phần hiển thị như sau:

* **TextView**: Hiển thị số liệu dạng văn bản.
* **LineChart**: Vẽ biểu đồ theo thời gian cho giá trị gia tốc, giúp người dùng quan sát biến thiên dữ liệu.

Biểu đồ được xây dựng bằng thư viện **MPAndroidChart**, với cấu hình cập nhật theo thời gian thực và làm mới sau mỗi chu kỳ 1 giây.

**Đánh giá địa hình**

Khi người dùng nhấn nút “Đánh giá”, ứng dụng gửi lệnh tương ứng và chờ nhận 10 gói dữ liệu từ ESP32. Các bước xử lý:

1. Thu thập *n* giá trị liên tiếp của gia tốc và con quay hồi chuyển.
2. Tính toán độ lệch chuẩn, trung bình, phương sai.
3. Dựa trên ngưỡng đã định, phân loại địa hình: **Bằng phẳng**, **gồ ghề nhẹ**, **Khá gồ ghề, …**.
4. Hiển thị kết quả và lưu lại vào cơ sở dữ liệu.

**Ghi log dữ liệu lên Firebase**

Firebase Realtime Database được sử dụng làm nơi lưu trữ dữ liệu cảm biến và kết quả đánh giá. Firebase có ưu điểm:

* Dễ tích hợp với ứng dụng Android.
* Hỗ trợ lưu dữ liệu có cấu trúc dạng cây JSON.
* Tự động đồng bộ nếu có kết nối mạng.

**Trực quan hóa và dễ sử dụng**

Giao diện người dùng được thiết kế đơn giản với tiêu chí:

* Các nút điều khiển kích thước lớn, bố trí dễ bấm.
* TextView trình bày rõ ràng dữ liệu.
* Biểu đồ dễ đọc, màu sắc phân biệt rõ ràng.
* Thông báo cảnh báo nếu mất kết nối hoặc lỗi trong đánh giá.

Ứng dụng cũng hỗ trợ **chế độ toàn màn hình**, loại bỏ distraction không cần thiết khi vận hành thiết bị.

**So sánh với các công nghệ khác**

|  |  |  |
| --- | --- | --- |
| Công nghệ | Ưu điểm | Nhược điểm |
| Kotlin | Cú pháp ngắn gọn, hiện đại | Khó tiếp cận với người mới |
| Flutter | Giao diện đẹp, chạy đa nền tảng | Chưa tối ưu với giao tiếp mạng thời gian thực |
| React Native | Dễ học, nhiều thư viện hỗ trợ | Chạy chậm hơn nếu xử lý phức tạp |

Bảng 7: So sánh Java với các công nghệ lập trình mobile khác

**Java được lựa chọn trong đồ án này vì:**

* Tương thích cao với Android SDK.
* Dễ dàng tích hợp các thư viện như Firebase, UDP socket.
* Đáp ứng tốt hiệu năng cho ứng dụng điều khiển nhẹ và hiển thị dữ liệu cảm biến.

## Lắp ráp phần cứng

**Giới thiệu**

Bên cạnh việc lập trình và mô phỏng, việc **tự tay lắp ráp phần cứng thực tế** đóng vai trò trung tâm trong việc triển khai, kiểm thử toàn bộ hệ thống đánh giá mặt phẳng địa hình. Khác với các dự án chỉ dừng lại ở mức độ mô hình hóa hoặc giả lập, trong đồ án này, sinh viên đã thực hiện toàn bộ quá trình xây dựng một thiết bị vật lý có khả năng vận hành độc lập, bao gồm cả cơ khí, điện tử và truyền thông không dây.

Phần cứng do sinh viên tự thiết kế, lắp ráp, hàn mạch và hiệu chỉnh là bước then chốt nhằm kiểm chứng tính khả thi và độ chính xác của hệ thống. Quá trình này cũng giúp sinh viên hiểu rõ hơn mối liên hệ giữa phần mềm và phần cứng trong các hệ thống nhúng.

**Danh sách linh kiện sử dụng**

|  |  |
| --- | --- |
| Tên linh kiện | Mục đích sử dụng |
| ESP32-WROOM | Vi điều khiển chính, xử lý tín hiệu và giao tiếp Wi-Fi |
| MPU6050 | Cảm biến đo gia tốc và tốc độ góc |
| Driver L298N | Mạch điều khiển động cơ DC |
| Động cơ DC (2 cặp) | Di chuyển thiết bị trên mặt phẳng cần đánh giá |
| Khung xe robot acrylic | Khung cố định toàn bộ thiết bị, bánh xe, động cơ |
| Pin lithium 18650 + dây rút | Cung cấp nguồn điện cho hệ thống |
| Dây jumper, breadboard | Kết nối các thành phần |
| Công tắc nguồn | Bật/tắt nguồn thiết bị |

Bảng 8: Danh sách linh kiện sử dụng để lắp ráp phần cứng

**Một số hình ảnh của phần cứng**

**Các vấn đề gặp phải trong quá trình lắp ráp**

* Nhiễu tín hiệu I2C: Khi các dây tín hiệu quá dài hoặc lỏng, tín hiệu từ MPU6050 về ESP32 bị lỗi, cần kiểm tra độ chắc chắn hoặc thêm điện trở pull-up.
* Nguồn yếu: Khi pin yếu, động cơ yếu hoặc vi điều khiển bị reset ngẫu nhiên.
* Lắp lệch tâm: Nếu MPU6050 không đặt ở gần trọng tâm thiết bị, dữ liệu gia tốc sẽ không phản ánh hoàn toàn chính xác các dao động, lực ly tâm.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

### Lựa chọn kiến trúc phần mềm

Để đảm bảo tính dễ bảo trì, mở rộng, cũng như tách biệt rõ ràng giữa giao diện người dùng, xử lý logic và dữ liệu, ứng dụng Android trong hệ thống đánh giá mặt phẳng địa hình được thiết kế theo **mô hình kiến trúc MVC (Model – View – Controller)**.

**Lý do lựa chọn mô hình kiến trúc MVC**

Kiến trúc MVC là một trong những kiến trúc phần mềm phổ biến và phù hợp với các ứng dụng giao diện người dùng như Android. MVC chia ứng dụng thành 3 thành phần chính:

* **Model (M)**: Chịu trách nhiệm quản lý dữ liệu, bao gồm lưu trữ, xử lý logic và truy xuất.
* **View (V)**: Là lớp giao diện, hiển thị thông tin và nhận thao tác từ người dùng.
* **Controller (C)**: Là cầu nối giữa View và Model. Controller nhận sự kiện từ View, xử lý và cập nhật Model hoặc yêu cầu View thay đổi.

Việc sử dụng mô hình kiến trúc MVC mang lại các lợi ích cụ thể như sau:

* Dễ dàng thay đổi giao diện mà không ảnh hưởng đến logic xử lý.
* Đơn giản hóa việc kiểm thử và gỡ lỗi từng phần riêng biệt.
* Dễ mở rộng về sau nếu cần thêm tính năng hoặc module mới.

**Ứng dụng kiến trúc MVC vào hệ thống**

Trong đồ án này, kiến trúc MVC được áp dụng và điều chỉnh theo đặc thù của ứng dụng Android kết hợp phần cứng:

|  |  |
| --- | --- |
| Thành phần MVC | Triển khai cụ thể trong ứng dụng |
| Model | * + - SensorData (lưu dữ liệu cảm biến)     - AssessmentResult (kết quả đánh giá)     - Logger (ghi log vào Firebase)     - UDPService (xử lý giao tiếp UDP)     - Calculate (xử lý tính toán để đưa ra đánh giá)     - Get (các thành phần xử lý việc truy xuất) |
| View | * + - Hello (giao diện khởi tạo ứng dụng)     - MainActivity (giao diện chính)     - SensorLogActivity (giao diện truy xuất, xem dữ liệu từ csdl)     - Biểu đồ (MPAndroidChart) |
| Controller | * + - MainActivity.java (điều phối sự kiện chính)     - ButtonController.java (xử lý lệnh điều khiển)     - AssessmentHandler (điều phối phiên đánh giá)     - UIUpdater (cập nhật giao diện theo dữ liệu mới) |

Bảng 9: Triển khai các thành phần kiến trúc MVC trong ứng dụng Android

**Một số điều chỉnh so với kiến trúc MVC truyền thống**

Do đặc thù là một ứng dụng điều khiển thiết bị vật lý theo thời gian thực, một số điểm được bổ sung/điều chỉnh như sau:

* **Tách riêng lớp UDPService và AssessmentHandler**: nhằm phục vụ việc xử lý dữ liệu mạng và logic đánh giá độc lập với giao diện. Điều này giúp tránh chậm trễ UI khi phải nhận dữ liệu cảm biến liên tục.
* **Sử dụng pattern Observer giữa Controller và View**: View được cập nhật mỗi khi hệ thống nhận được dữ liệu mới, giúp phản hồi nhanh hơn thay vì truy vấn dữ liệu theo thời gian định kỳ.
* **Tách UI thành nhiều fragment**: để dễ quản lý và hỗ trợ tốt chế độ landscape, mỗi fragment quản lý một phần chức năng riêng (giao diện điều khiển, hiển thị biểu đồ, dữ liệu cảm biến, đánh giá).

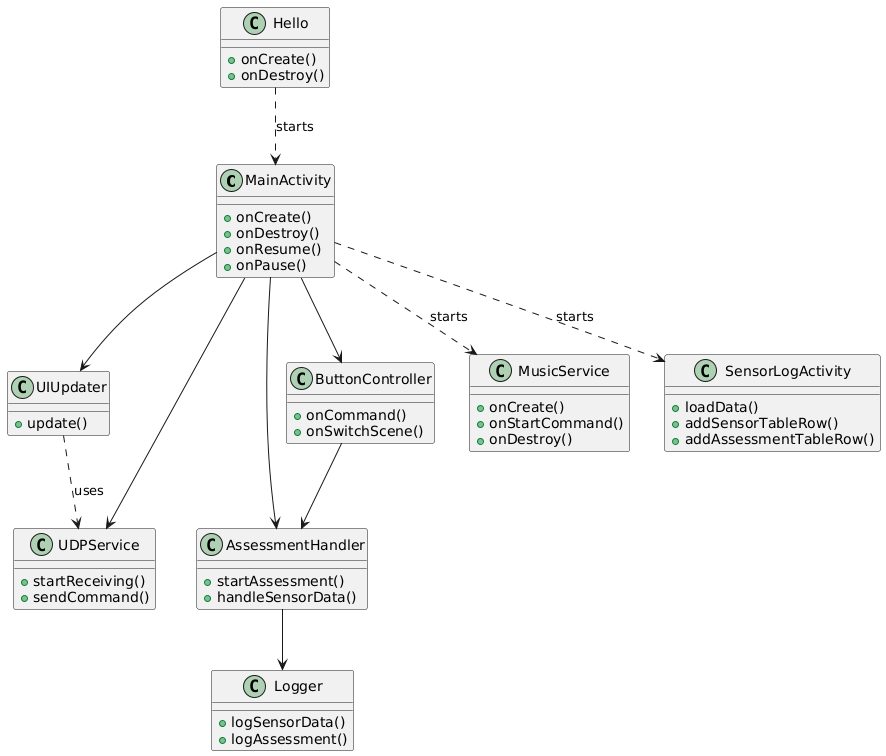
Với mô hình kiến trúc này, ứng dụng đạt được:

* Khả năng phân tách tốt giữa phần giao diện và phần xử lý logic.
* Dễ kiểm thử từng module riêng (ví dụ: chỉ test giao diện mà không cần có phần cứng).
* Hạn chế tối đa việc UI bị treo hoặc ứng dụng bị văng hoàn toàn khi nhận dữ liệu liên tục nhờ sử dụng luồng riêng và cập nhật bất đồng bộ.

### Thiết kế tổng quan

Để đảm bảo tính tổ chức tốt, dễ mở rộng và bảo trì, hệ thống được thiết kế theo hướng phân tầng (layered architecture). Mỗi tầng được đóng gói thành một hoặc nhiều package riêng biệt, mỗi package đảm nhiệm một vai trò cụ thể và không có sự phụ thuộc ngược tầng.

Dưới đây là sơ đồ gói UML thể hiện cấu trúc tổng quan của ứng dụng:



**Hình 12:** Biểu đồ phụ thuộc gói của phần mềm

**Nguyên tắc thiết kế tuân thủ**

Thiết kế phần mềm của hệ thống tuân theo một số nguyên tắc thiết kế quan trọng nhằm đảm bảo tính modular (mô-đun hóa), dễ bảo trì, mở rộng, và phân tách trách nhiệm rõ ràng giữa các thành phần. Dưới đây là các nguyên tắc chính được áp dụng:

#### Phân tách rõ vai trò và trách nhiệm

Mỗi lớp đóng một chức năng cụ thể:

* *MainActivity*: lớp trung tâm khởi tạo và điều phối các thành phần.
* *UIUpdater*: chuyên xử lý cập nhật giao diện.
* *UDPService*: quản lý việc giao tiếp UDP với phần cứng.
* *AssessmentHandler*: thực hiện phiên đánh giá địa hình và xử lý dữ liệu cảm biến.
* *Logger*: ghi dữ liệu vào cơ sở dữ liệu.
* *ButtonController*: xử lý các tương tác từ giao diện điều khiển.
* *SensorLogActivity, Hello, MusicService*: đảm nhiệm các tác vụ riêng biệt, tách biệt hoàn toàn với logic xử lý trung tâm.

#### Tầng điều phối tập trung (MainActivity)

*MainActivity* đóng vai trò như một điểm điều phối chính: khởi tạo và kết nối các thành phần chức năng. Lớp này không có các logic xử lý nặng nề, tất cả các tác vụ chính của ứng dụng được chuyển giao và thực hiện trong các lớp phụ trách riêng biệt như *AssessmentHandler, UDPService, UIUpdater, v.v …*

#### Phân chia xử lý theo luồng logic

* Giao tiếp mạng được đóng gói trong *UDPService*, được *MainActivity* và *UIUpdater* sử dụng nhưng không vượt quyền truy cập sang các thành phần khác.
* Logic đánh giá được đóng gói độc lập trong *AssessmentHandler*, vừa xử lý cảm biến, vừa gọi tới *Logger* để lưu dữ liệu.
* ButtonController là lớp điều phối hành động giao diện, không trực tiếp can thiệp vào giao diện hay dữ liệu mạng, mà chỉ gửi yêu cầu qua *AssessmentHandler, UDPService*.

#### Nguyên tắc giảm phụ thuộc trực tiếp

* Các phụ thuộc được kiểm soát theo hướng “top-down”: *MainActivity, SensorLogActivity* gọi các lớp xử lý logic, chứ không ngược lại.
* Hoàn toàn không có các phụ thuộc vòng (circular dependancy), điều này mang ý nghĩa giúp hệ thống dễ bảo trì, phát hiện lỗi và về cơ bản tránh được các lỗi runtime phức tạp do nhiều lớp logic mâu thuẫn lẫn nhau.

#### Tăng cường khả năng tái sử dụng và kiểm thử

* Các lớp như *Logger, UDPService, AssessmentHandler, ButtonController* hoạt động độc lập, có thể dễ dàng kiểm thử riêng (unit test) mà không phụ thuộc vào giao diện hay hệ thống phần cứng ngoài.
* Lớp *SensorLogActivity* tuy vừa là Model vừa là View (Xử lý truy vấn dữ liệu bảng từ csdl trong cùng một lớp) nhưng chỉ chuyên xử lý hiển thị bảng dữ liệu được ghi trên csdl, từ đó giúp tách riêng nhiệm vụ xem log và đánh giá địa hình.

#### Tính mở rộng tốt

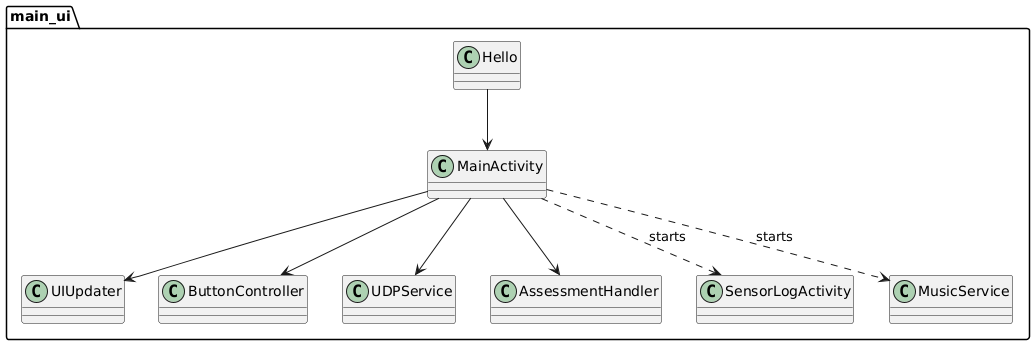
Với thiết kế hệ thống ứng dụng này, nếu cần thêm tính năng (ví dụ như sv đã thực hiện: Thêm *MusicService*), chỉ cần tạo thêm lớp xử lý mới và tích hợp nhẹ nhàng vào *MainActivity* mà không cần lo sẽ ảnh hưởng đến cấu trúc hiện có.

### Thiết kế chi tiết gói

Hệ thống được tổ chức thành nhiều gói chức năng khác nhau, mỗi gói đảm nhiệm một phần công việc riêng biệt trong toàn bộ ứng dụng. Việc phân chia này đảm bảo tính modular hóa và phân tách trách nhiệm, giúp dễ dàng bảo trì, mở rộng và kiểm thử. Dưới đây là phần thiết kế chi tiết theo từng nhóm gói chính trong hệ thống.

#### Gói 1: Giao diện người dùng và điều phối tổng (main\_ui)

**Biểu đồ lớp:**



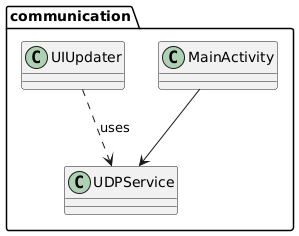
Hình 13: Biểu đồ lớp gói main\_ui

**Giải thích thiết kế:**

* *Hello* là màn hình khởi đầu, chịu trách nhiệm giới thiệu ứng dụng và gọi đến *MainActivity* khi người dùng sẵn sàng thao tác.
* *MainActivity* là thành phần trung tâm của toàn bộ ứng dụng, thực hiện điều phối các thành phần khác như giao tiếp mạng (*UDPService*), cập nhật giao diện (*UIUpdater*), đánh giá địa hình (*AssessmentHandler*) và xử lý sự kiện từ nút bấm (*ButtonController*).
* *MainActivity* không xử lý logic trực tiếp mà ủy quyền cho các lớp phụ trách, giữ cho mã nguồn sạch, dễ bảo trì.

#### Gói 2: Giao tiếp mạng (communication)

**Biểu đồ lớp:**



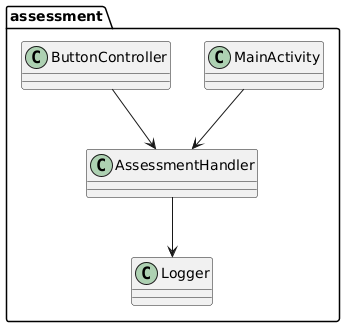
Hình 14: Biểu đồ lớp gói communication

**Giải thích thiết kế:**

* *UDPService* là lớp duy nhất chịu trách nhiệm giao tiếp với phần cứng thông qua giao thức UDP.
* Lớp này cung cấp phương thức gửi lệnh (sendCommand) và bắt đầu nhận dữ liệu (startReceiving) dưới dạng luồng nền.
* *UIUpdater* sử dụng *UDPService* để lấy dữ liệu cảm biến và cập nhật giao diện, còn MainActivity dùng để điều khiển trực tiếp thiết bị.

#### Gói 3: Xử lý đánh giá (assessment)

**Biểu đồ lớp:**



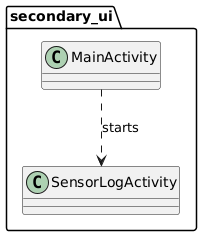
Hình 15: Biểu đồ lớp gói assessment

**Giải thích thiết kế:**

* *AssessmentHandler* chịu trách nhiệm thu thập dữ liệu trong quá trình đánh giá địa hình, xử lý thông tin và phân loại kết quả.
* Lớp này cũng thực hiện logic xử lý trung gian giữa dữ liệu cảm biến và thành phần hiển thị hoặc lưu trữ.
* *Logger* được gọi từ *AssessmentHandler* để lưu dữ liệu cảm biến và kết quả đánh giá vào Firebase.
* *ButtonController* có thể khởi động đánh giá thông qua *AssessmentHandler*.

#### Gói 4: Giao diện người dùng phụ (secondary\_ui)

**Biểu đồ lớp:**



Hình 16: Biểu đồ lớp gói secondary\_ui

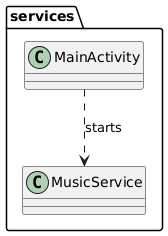
**Giải thích thiết kế:**

* *SensorLogActivity* là một Model và View phụ chuyên phục vụ cho việc xem lại dữ liệu đã được lưu trữ.
* Lớp này cung cấp các phương thức để hiển thị dữ liệu cảm biến (bảng dữ liệu cảm biến, bảng kết quả đánh giá).
* Thiết kế độc lập giúp tránh rối loạn chức năng của *MainActivity*.

#### Gói 5: Xử lý âm thanh (services)

Tuy đây là phần sv tự ý thêm vào, nhưng trong báo cáo vẫn sẽ bao gồm biểu đồ lớp của gói này kèm đề cập về giải thích thiết kế.

**Biểu đồ lớp:**



Hình 17: Biểu đồ lớp gói services

**Giải thích thiết kế:**

* *MusicService* là một dịch vụ Android, phát nhạc nền khi ứng dụng hoạt động.
* Việc xử lý nhạc được tách khỏi giao diện để tránh ảnh hưởng đến luồng chính của ứng dụng.
* *MainActivity* có thể bắt đầu hoặc dừng dịch vụ tùy trạng thái hoạt động.

## Thiết kế chi tiết

### Thiết kế phần cứng

Hệ thống đánh giá mặt phẳng địa hình được xây dựng trên một nền tảng phần cứng bao gồm các thành phần chính: ESP32, MPU6050, mạch cầu H L298N, 4 động cơ DC, pin lithium 18650, mạch sạc pin, switch (công tắc), breadboard, dây jumper và khung robot acrylic. Thiết kế phần cứng được tối ưu nhằm đảm bảo tính nhỏ gọn, ổn định điện áp và độ tin cậy trong quá trình thu thập và xử lý dữ liệu.

#### Mục tiêu thiết kế

* Cung cấp khả năng di chuyển cho robot 4 bánh trên các địa hình thực tế.
* Thu thập dữ liệu gia tốc, góc nghiêng từ cảm biến MPU6050 và gửi về ứng dụng điều khiển qua Wi-Fi.
* Đảm bảo hệ thống hoạt động ổn định, dễ lắp ráp, dễ thay thế linh kiện.
* Tối ưu điện năng sử dụng và đảm bảo an toàn cho các thiết bị điện tử.

#### Sơ đồ mạch

#### Chi tiết kết nối phần cứng

* 1. Nguồn điện
* Pin: 1 viên pin lithium 18650 đấu nối tiếp tạo điện áp đầu ra 3.7V khi đầy, có dung tích 4000mah.

A battery with red wires and a white rectangle

AI-generated content may be incorrect.

Hình 18: Pin sử dụng trong phần cứng

* Mạch sạc pin: Sử dụng mạch sạc pin 18650 tích hợp mạch tăng áp 7W có thể điều chỉnh đến đến 27V. Mạch sử dụng IC TP4056, không có chức năng bảo vệ pin, cần sạc khi có dấu hiệu sụt áp tránh làm hỏng pin. Ở đây, AC/DC converter được điều chỉnh để mạch có đầu ra khoảng 12V, vừa đủ cho robot.

A green circuit board with a white circle around it

AI-generated content may be incorrect.

Hình 19: Mạch sạc pin dùng trong phần cứng

* Switch: Điều khiển đóng tắt nguồn chính cho phần cứng, có thể sử dụng bất cứ switch có dạng ON/OFF nào.
* L298N: Nhận nguồn 12V thông qua switch, đồng thời cung cấp nguồn 5V ổn định cho ESP32 thông qua chân 5V OUT.

Placeholder ảnh l298n

* ESP32: Được cấp nguồn 5V qua chân VIN, lấy từ 5V OUT của L298N.

Placeholder ảnh ESP32

* MPU6050: Được cấp nguồn 3.3V từ chân 3V3 của ESP32.

Placeholder ảnh ESP32

* 1. Điều khiển động cơ
* Hai cặp động cơ DC ghép nối tạo thành robot 4 bánh.
* Động cơ được điều khiển bởi L298N, với cấu hình chân cụ thể như sau:
  + I**N1 – GPIO 27** *(trái tiến)*
  + **IN2 – GPIO 26** *(trái lùi)*
  + **IN3 – GPIO 25** *(phải tiến)*
  + **IN4 – GPIO 33** *(phải lùi)*
* ENA, ENB: jumper đóng, động cơ luôn chạy với tốc độ mặc định (không điều khiển sóng PWN).
  1. Kết nối module cảm biến MPU6050
* Nguồn: 3.3V (chân 3V3 của ESP32)
* GND: GND chung với ESP32
* SDA – GPIO 21
* SCL – GPIO 22
  1. Dây nối (jumper wire), bảng test (breadboard) và khung xe
* Các kết nối hầu hết đã được hàn cứng để hạn chế tình trạng chân tiếp xúc gặp vấn đề (địa hình xóc khiến jumper wire bị đứt, lỏng)
* Breadboard chỉ đóng vai trò cố định cho các module, không đóng vai trò làm tiếp xúc.
* Các module được dán keo hoặc cố định bằng dây rút trên khung xe acrylic.
* Các động cơ DC đi kèm khung xe được bắt vít chắc chắn, hạn chế tối đa rủi ro lỏng, bung rời trong quá trình vận hành.

#### Nguyên lý hoạt động tổng thể

* + 1. Khi người dùng bật công tắc, nguồn điện từ pin sẽ cấp vào L298N.
    2. L298N cấp điện cho các động cơ DC (qua OUT1–OUT4) và cấp 5V cho ESP32 qua chân 5V out.
    3. ESP32 khởi động, bắt đầu kết nối Wi-Fi và thiết lập giao tiếp I2C với MPU6050.
    4. Ứng dụng di động gửi lệnh điều khiển động cơ qua UDP.
    5. ESP32 nhận lệnh và điều khiển động cơ bằng cách thiết lập mức HIGH/LOW cho các chân IN1–IN4.
    6. Dữ liệu từ MPU6050 được thu thập, xử lý và gửi về app qua UDP để hiển thị hoặc đánh giá.

#### Hình ảnh thực tế của phần cứng

Placeholder, 3 ảnh 3 góc

#### Một số lưu ý khi thiết kế

* Tuy không được nhắc đến nhưng tất cả các thành phần điện tử đều cần được nối vào tiếp địa (GND hoặc G) để có một điểm quy chiếu.
* Mạch điều khiển động cơ không sử dụng PWN: Các jumper ENA, ENB phải được đóng mặc định để tránh trường hợp động cơ không hoạt động chính xác.
* Bảo vệ nguồn: Mạch sạc nên là loại có bảo vệ pin hoặc diode chống ngược cực khi cấp nguồn. Bảo đảm tuổi thọ pin và tránh trường hợp chập cháy ngoài ý muốn.
* Sử dụng pin có điện áp phù hợp: Trước khi thực hiện lắp ráp phần cứng, sv đã mắc sai lầm trong tính toán và chỉ sử dụng 01 viên pin có điện áp quá nhỏ so với yêu cầu của các module trong phần cứng (đầu ra 3.7V nhưng L298N yêu cầu 12V). Với phần cứng như trên thì nguồn điện cung cấp nên được sử dụng ít nhất 02 viên pin để tạo điện áp đầu ra 7.4V – 8.4V khi đầy (hoặc sử dụng cụm pin 3 viên để đạt được ~12V)

### Thiết kế giao diện ứng dụng

**Thông tin màn hình đích**

Ứng dụng được thiết kế và thử nghiệm trên thiết bị có độ phân giải 1920x1080 pixels (Full HD), tương đương với tỷ lệ khung hình 16:9 ở chế độ nằm ngang (landscape). Đây là độ phân giải phổ biến cho các máy tính bảng hoặc điện thoại thông minh hiện đại, đảm bảo không gian hiển thị rộng rãi, phù hợp với việc hiển thị đồng thời dữ liệu cảm biến, nút điều khiển, biểu đồ và nút chức năng.

* Mật độ điểm ảnh (dpi): mdpi đến xxhdpi (thiết kế linh hoạt với các kích thước khác nhau).
* Màu sắc hỗ trợ: 32-bit color (ARGB8888), cho phép hiển thị màu sắc rõ ràng và phân biệt tốt giữa các vùng nội dung.
* Định hướng (orientation): thiết kế tối ưu cho chế độ landscape.

**Nguyên tắc và chuẩn hóa thiết kế giao diện**

Trong quá trình thiết kế giao diện, sv đã tuân theo một số chuẩn hóa và quy tắc nhất định để đảm bảo giao diện nhất quán, dễ sử dụng, và thân thiện với người dùng:

### Thiết kế lớp

Phần này có độ dài từ ba đến bốn trang. Sinh viên trình bày thiết kế chi tiết các thuộc tính và phương thức cho một số lớp chủ đạo/quan trọng nhất của ứng dụng (từ 2-4 lớp). Thiết kế chi tiết cho các lớp khác, nếu muốn trình bày, sinh viên đưa vào phần phụ lục.

Để minh họa thiết kế lớp, sinh viên thiết kế luồng truyền thông điệp giữa các đối tượng tham gia cho 2 đến 3 use case quan trọng nào đó bằng biểu đồ trình tự (hoặc biểu đồ giao tiếp).

### Thiết kế cơ sở dữ liệu

Phần này có độ dài từ hai đến bốn trang. Sinh viên thiết kế, vẽ và giải thích biểu đồ thực thể liên kết (E-R diagram). Từ đó, sinh viên thiết kế cơ sở dữ liệu tùy theo hệ quản trị cơ sở dữ liệu mà mình sử dụng (SQL, NoSQL, Firebase, v.v.)

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Sinh viên liệt kê các công cụ, ngôn ngữ lập trình, API, thư viện, IDE, công cụ kiểm thử, v.v. mà mình sử dụng để phát triển ứng dụng. Mỗi công cụ phải được chỉ rõ phiên bản sử dụng. SV nên kẻ bảng mô tả tương tự như Bảng 1. Nếu có nhiều nội dung trình bày, sinh viên cần xoay ngang bảng.

**Bảng 10** Danh sách thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Eclipse Oxygen 64 bit | http://www.eclipse.org/ |
| v.v. | v.v. | v.v. |

### Kết quả đạt được

Sinh viên trước tiên mô tả kết quả đạt được của mình là gì, ví dụ như các sản phẩm được đóng gói là gì, bao gồm những thành phần nào, ý nghĩa, vai trò?

Sinh viên cần thống kê các thông tin về ứng dụng của mình như: số dòng code, số lớp, số gói, dung lượng toàn bộ mã nguồn, dung lượng của từng sản phẩm đóng gói, v.v. Tương tự như phần liệt kê về công cụ sử dụng, sinh viên cũng nên dùng bảng để mô tả phần thông tin thống kê này.

### Minh hoạ các chức năng chính

Sinh viên lựa chọn và đưa ra màn hình cho các chức năng chính, quan trọng, và thú vị nhất. Mỗi giao diện cần phải có lời giải thích ngắn gọn. Khi giải thích, sinh viên có thể kết hợp với các chú thích ở trong hình ảnh giao diện.

## Kiểm thử

Phần này có độ dài từ hai đến ba trang. Sinh viên thiết kế các trường hợp kiểm thử cho hai đến ba chức năng quan trọng nhất. Sinh viên cần chỉ rõ các kỹ thuật kiểm thử đã sử dụng. Chi tiết các trường hợp kiểm thử khác, nếu muốn trình bày, sinh viên đưa vào phần phụ lục.

Sinh viên sau cùng tổng kết về số lượng các trường hợp kiểm thử và kết quả kiểm thử. Sinh viên cần phân tích lý do nếu kết quả kiểm thử không đạt.

## Triển khai

Sinh viên trình bày mô hình và/hoặc cách thức triển khai thử nghiệm/thực tế. Ứng dụng của sinh viên được triển khai trên server/thiết bị gì, cấu hình như thế nào. Kết quả triển khai thử nghiệm nếu có (số lượng người dùng, số lượng truy cập, thời gian phản hồi, phản hồi người dùng, khả năng chịu tải, các thống kê, v.v.)

# Các giải pháp và đóng góp nổi bật

Chương này có độ dài tối thiểu 5 trang, tối đa không giới hạn. Sinh viên cần trình bày tất cả những nội dung đóng góp mà mình thấy tâm đắc nhất trong suốt quá trình làm BTL. Đó có thể là một loạt các vấn đề khó khăn mà sinh viên đã từng bước giải quyết được, là giải thuật cho một bài toán cụ thể, là giải pháp tổng quát cho một lớp bài toán, hoặc là mô hình/kiến trúc hữu hiệu nào đó được sinh viên thiết kế.

Chương này là **cơ sở quan trọng** để các thầy cô đánh giá sinh viên. Vì vậy, sinh viên cần phát huy tính sáng tạo, khả năng phân tích, phản biện, lập luận, tổng quát hóa vấn đề và tập trung viết cho thật tốt.

Mỗi giải pháp hoặc đóng góp của sinh viên cần được trình bày trong một mục độc lập bao gồm ba mục con: (i) dẫn dắt/giới thiệu về bài toán/vấn đề, (ii) giải pháp, và (iii) kết quả đạt được (nếu có).

Sinh viên lưu ý **không trình bày lặp lại nội dung**. Những nội dung đã trình bày chi tiết trong các chương trước không được trình bày lại trong chương này. Vì vậy, với nội dung hay, mang tính đóng góp/giải pháp, sinh viên chỉ nên tóm lược/mô tả sơ bộ trong các chương trước, đồng thời tạo tham chiếu chéo tới đề mục tương ứng trong Chương 5 này. Chi tiết thông tin về đóng góp/giải pháp được trình bày trong mục đó.

Ví dụ, trong Chương 4, sinh viên có thiết kế được kiến trúc đáng lưu ý gì đó, là sự kết hợp của các kiến trúc MVC, MVP, SOA, v.v. Khi đó, sinh viên sẽ chỉ mô tả ngắn gọn kiến trúc đó ở Chương 4, rồi thêm các câu có dạng: “Chi tiết về kiến trúc này sẽ được trình bày trong phần 5.1”.

# Kết luận và hướng phát triển

## Kết luận

Sinh viên so sánh kết quả nghiên cứu hoặc sản phẩm của mình với các nghiên cứu hoặc sản phẩm tương tự.

Sinh viên phân tích trong suốt quá trình thực hiện BTL, mình đã làm được gì, chưa làm được gì, các đóng góp nổi bật là gì, và tổng hợp những bài học kinh nghiệm rút ra nếu có.

## Hướng phát triển

Trong phần này, sinh viên trình bày định hướng công việc trong tương lai để hoàn thiện sản phẩm hoặc nghiên cứu của mình.

Trước tiên, sinh viên trình bày các công việc cần thiết để hoàn thiện các chức năng/nhiệm vụ đã làm. Sau đó sinh viên phân tích các hướng đi mới cho phép cải thiện và nâng cấp các chức năng/nhiệm vụ đã làm.

# Tài liệu tham khảo

Lưu ý: Sinh viên không được đưa bài giảng/slide, các trang Wikipedia, hoặc các trang web thông thường làm tài liệu tham khảo.

Một trang web được phép dùng làm tài liệu tham khảo **chỉ khi** nó là công bố chính thống của cá nhân hoặc tổ chức nào đó. Ví dụ, trang web đặc tả ngôn ngữ XML của tổ chức W3C https://www.w3.org/TR/2008/REC-xml-20081126/ là TLTK hợp lệ.

Có năm loại tài liệu tham khảo mà sinh viên phải tuân thủ đúng quy định về cách thức liệt kê thông tin như sau. Lưu ý: các phần văn bản trong cặp dấu < > dưới đây chỉ là hướng dẫn khai báo cho từng loại tài liệu tham khảo; sinh viên cần xóa các phần văn bản này trong BTL của mình.

<**Bài báo đăng trên tạp chí khoa học**: Tên tác giả, tên bài báo, tên tạp chí, volume, từ trang đến trang (nếu có), nhà xuất bản, năm xuất bản>

1. Hovy E. H., Automated Discourse Generation Using Discourse Structure Relations, Artificial Intelligence, Elsevier Science Publishers, 63: 341-385, 1993.

<**Sách**: Tên tác giả, tên sách, volume (nếu có), lần tái bản (nếu có), nhà xuất bản, năm xuất bản>

1. Peterson L. L. and Davie B. S., Computer Networks: A Systems Approach, 2nd ed., Mogran-Kaufmann, 1999.
2. Nguyễn Thúc Hải, Mạng máy tính và các hệ thống mở, Nhà xuất bản giáo dục, 1999.

<**Tập san Báo cáo Hội nghị Khoa học**: Tên tác giả, tên báo cáo, tên hội nghị, ngày (nếu có), địa điểm hội nghị, năm xuất bản>

1. Poesio M. and Di Eugenio B., Discourse Structure and Anaphoric Accessibility, In Proc. of the ESSLLI Workshop on Information Structure, Discourse Structure and Discourse Semantics, Helsinki, 2001.

<**Bài tập lớn, Luận văn Thạc sĩ, Tiến sĩ**: Tên tác giả, tên bài tập lớn/luận văn, loại bài tập lớn/luận văn, tên trường, địa điểm, năm xuất bản>

1. Knott D., A Data-Driven Methodology for Motivating a Set of Coherence Relations, Ph.D. Thesis, University of Edinburgh, UK, 1996.

<**Tài liệu tham khảo từ Internet**: Tên tác giả (nếu có), tựa đề, cơ quan (nếu có), địa chỉ trang web, thời gian lần cuối truy cập trang web>

1. Berners-Lee T., Hypertext Transfer Protocol (HTTP), CERN, ftp:/info.cern.ch/pub/www/doc/http-spec.txt.Z, last visited May 2010.
2. Princeton University, WordNet, http://www.cogsci.princeton.edu/~wn/index.shtml, last visited May 2010.

# Phụ lục

Phần phụ lục là không bắt buộc. Nếu sinh viên không có nhu cầu trình bày thêm, có thể xóa bỏ phần này. Lưu ý là phần phụ lục chỉ được đánh chỉ mục đến cấp 2, sinh viên không được phép chia nhỏ hơn nữa.

Hướng dẫn viết Bài tập lớn

Quy định chung

Dưới đây là một số quy định và hướng dẫn viết Bài tập lớn mà bắt buộc sinh viên phải **đọc kỹ và tuân thủ nghiêm ngặt**.

Sinh viên cần đảm bảo tính thống nhất toàn báo cáo (font chữ, căn dòng hai bên, hình ảnh, bảng, margin trang, đánh số trang, v.v.). Để làm được như vậy, sinh viên chỉ cần sử dụng các định dạng theo đúng template BTL này. Khi paste nội dung văn bản từ tài liệu khác của mình, sinh viên cần chọn kiểu Copy là “Text Only” để định dạng văn bản của template không bị phá vỡ/vi phạm.

Tuyệt đối cấm sinh viên đạo văn. Sinh viên cần ghi rõ nguồn cho tất cả những gì không tự mình viết/vẽ lên, bao gồm các câu trích dẫn, các hình ảnh, bảng biểu, v.v. Khi bị phát hiện, sinh viên sẽ **không được phép bảo vệ BTL**.

Tất cả các hình vẽ, bảng biểu, công thức, và tài liệu tham khảo trong BTL nhất thiết phải được SV giải thích và tham chiếu tới ít nhất một lần. Không chấp nhận các trường hợp sinh viên đưa ra hình ảnh, bảng biểu tùy hứng và không có lời mô tả/giải thích nào.

Sinh viên tuyệt đối không trình bày BTL theo kiểu viết ý hoặc gạch đầu dòng. BTL không phải là một slide thuyết trình; khi người đọc không hiểu sẽ không có ai giải thích hộ. Sinh viên cần viết thành các đoạn văn và phân tích, diễn giải đầy đủ, rõ ràng. Câu văn cần đúng ngữ pháp, đầy đủ chủ ngữ, vị ngữ và các thành phần câu.

Khi thực sự cần liệt kê, sinh viên nên liệt kê theo phong cách khoa học với các ký tự La Mã. Ví dụ, nhiều sinh viên luôn cảm thấy hối hận vì (i) chưa cố gắng hết mình, (ii) chưa sắp xếp thời gian học/chơi một cách hợp lý, (iii) chưa tìm được người yêu để chia sẻ quãng đời sinh viên vất vả, và (iv) viết BTL một cách cẩu thả.

Trong một số trường hợp nhất thiết phải dùng các bullet để liệt kê, sinh viên cần thống nhất Style cho toàn bộ các bullet các cấp mà mình sử dụng đến trong báo cáo. Nếu dùng bullet cấp 1 là hình tròn đen, toàn bộ báo cáo cần thống nhất cách dùng như vậy; ví dụ như sau:

* Đây là mục 1 – Thực sự không còn cách nào khác tôi mới dùng đến việc bullet trong báo cáo.
* Đây là mục 2 – Nghĩ lại thì tôi có thể không cần dùng bullet cũng được. Nên tôi sẽ xóa bullet và tổ chức lại hai mục này trong báo cáo của mình cho khoa học hơn. Tôi muốn thầy cô và người đọc cảm nhận được tâm huyết của tôi trong từng trang báo cáo BTL.

Tạo đề mục

Đề mục giúp tạo bố cục cho tài liệu. Để các tính năng tự động – ví dụ tính năng cập nhật mục lục, hoặc tính năng tham chiếu chéo – của Word hoạt động được, sinh viên cần tuân thủ theo các style đã tạo trong tài liệu này. Để hiển thị các style này, sinh viên vào tab Home trong thanh Ribbon của Word.

Để tạo đề mục cấp 1, 2, 3, 4, 5, sinh viên gõ tiêu đề cho đề mục của mình rồi chọn các style là Heading 1, 2, 3, 4, 5 tương ứng. Sinh viên hạn chế dùng tới đề mục cấp 4, và phải trong trường hợp thực sự cần thiết mới dùng đến đề mục cấp 5.

Phần phụ lục chỉ cho phép có hai cấp tiêu đề. Hai style tương ứng với hai cấp này là “Heading 7, Phụ lục cấp 1” và “Heading 8, Phụ lục cấp 2”.

Bảng biểu

Sinh viên lưu ý không để bảng tràn ra lề (margin) trên, dưới, trái hoặc phải của trang. Do không gian nhỏ hẹp, bảng nên có font là 12pt hoặc nhỏ hơn. Độ dãn dòng của bảng nên là 1 line. Căn lề bảng là căn giữa, nhưng nội dung văn bản trong bảng nên được căn lề trái.

Sinh viên có thể viết tắt các từ trong bảng để tiết kiệm không gian nhưng phải giải thích các từ viết tắt này ở phần Chú thích bảng. Ví dụ áp dụng được minh họa trong Bảng 2.

**Bảng 11** Ví dụ sử dụng bảng

Chú thích: Y: Year; RS: Risk Set; G: Graduated; AB: Academically Excluded;C: Censored; HRG: Hazard Ratio – Graduated

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Y** | **RS** | **G** | **AE** | **C** | **HRG**  **(%)** |
| **1** | 11.959 | 0 | 725 | 1619 | 0,0 |
| **2** | 10.457 | 0 | 474 | 1513 | 0,0 |
| **3** | 7.365 | 1213 | 335 | 966 | 16,7 |
| **4** | 900 | 599 | 145 | 405 | 55,3 |

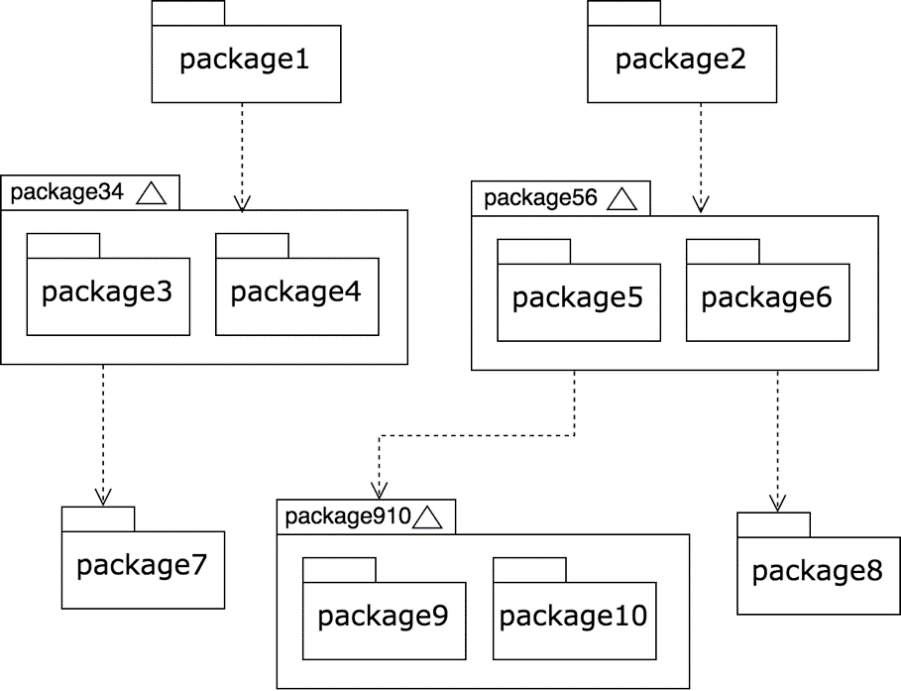
Sinh viên được tùy ý lựa chọn định dạng (template) cho các bảng trong BTL của mình, nhưng phải đảm bảo tính nhất quán trên toàn BTL. Template bảng phải đảm bảo phần heading cột trong bảng có font in đậm và nổi bật (highlighted) hơn so với các nội dung khác trong bảng.

Vì bảng có thể kéo dài nhiều trang, tiêu đề của bảng nên để ở phần đầu của bảng. Sinh viên không thêm tiêu đề bảng bằng tay. Để thêm tiêu đề bảng tự động, sinh viên nhấn chuột phải vào bảng, chọn “Insert Caption”, chọn “Label” là “Bảng”, rồi nhấn nút “OK”. Sau đó, sinh viên nhập vào nội dung tiêu đề và căn chỉnh “Center” cho tiêu đề này. Lưu ý, sinh viên cần bôi đậm bằng tay cụm từ **Bảng n**. Kết quả thu được có dạng như sau “**Bảng 2** Ví dụ sử dụng bảng”. Sinh viên nên xoay ngang trang giấy trong trường hợp bảng có nhiều cột với nhiều nội dung văn bản.

Hình vẽ

Tương tự như bảng, sinh viên không được để hình vẽ tràn lề trang. Căn lề cho hình vẽ là căn giữa (Center). Cách thêm tiêu đề hình vẽ tương tự như cách thêm tiêu đề bảng, nhưng sinh viên chọn “Label” là “Hình” thay vì “Bảng”.

Tiêu đề hình vẽ phải đặt ở dưới hình vẽ. Nếu hình vẽ được copy từ trên mạng, sinh viên bắt buộc phải ghi rõ nguồn. Sinh viên nên thống nhất công cụ sử dụng và style cho hình vẽ trong toàn BTL. Các chi tiết trong hình vẽ phải được bố trí gọn gàng; chữ trong hình phải đảm bảo nhìn được rõ nét khi in báo cáo trên giấy A4. Khi resize ảnh, cần giữ nguyên tỷ lệ dài rộng, tránh làm méo hoặc vỡ hình. Ví dụ hình vẽ được minh họa trong Hình 3.



**Hình 20** Ví dụ hình vẽ

Tài liệu tham khảo

Sinh viên cần hạn chế tối đa dùng trang Web làm tài liệu tham khảo. Chỉ chấp nhận trang Web làm TLTK khi trang đó là nơi công bố chính thức của tổ chức hoặc cá nhân nào đó. Ví dụ, trang web đặc tả ngôn ngữ XML của tổ chức W3C https://www.w3.org/TR/2008/REC-xml-20081126/ là TLTK hợp lệ.

Khi giới thiệu về công cụ, API, thư viện, hoặc nền tảng nào đó, sinh viên có thể đưa ra địa chỉ URL của các tiện ích này. Sinh viên lưu ý địa chỉ URL đó không phải là tài liệu tham khảo. Trong các trường hợp tương tự như vậy, sinh viên nên tạo “Footnote”. Sinh viên tạo “Footnote” bằng cách vào mục “References”, chọn “Insert Footnote”. Ví dụ tạo Footnote như sau: TensorFlow[[1]](#footnote-1) là nền tảng học máy mã nguồn mở đang được sử dụng rộng rãi hiện nay. Lưu ý: số Footnote phải đặt sát với từ được mô tả. Như trong ví dụ trên, số 1 được đặt ngay cạnh chữ TensorFlow (không có dấu cách).

Công thức toán học

Giống như bảng, hình vẽ, và tài liệu tham khảo, công thức toán học cần được đánh số, giải thích, và tham chiếu đầy đủ.

Để thêm tiêu đề (caption) cho công thức, sinh viên đặt con trỏ văn bản vào dòng văn bản dưới công thức. Sau đó vào menu “Insert” (lưu ý là menu Insert, không phải là tab Insert trong thanh Ribbon) và chọn mục “Caption”. Từ Popup Menu hiện ra, sinh viên chọn “Label” là “Công thức”, rồi nhấn nút “OK”. Sinh viên bôi đậm chữ **Công thức n**, và căn tiêu đề ra giữa. Công thức 1 là ví dụ mẫu cho sinh viên tham khảo.

**Công thức 1** Khai triển Newton

Tham chiếu chéo

Tham chiếu chéo (Cross-reference) là tiện ích hữu hiệu cho người viết báo cáo. Nó giúp tạo các liên kết tham chiếu (hyperlink) tới các hình ảnh, bảng biểu, tài liệu tham khảo, và các đề mục một cách tự động. Ví dụ, ngay trong câu này, một tham chiếu đã được tạo ra tới mục 2.1. Người đọc dễ dàng nhấp chuột vào liên kết 2.1 để ngay lập tức chuyển đến mục đó.

Để tạo tham chiếu chéo tới các đề mục (heading), sinh viên vào tab “References”, rồi tìm và nhấn chọn “Cross-refrence”. SV chọn “Refrence type” là “Heading” và chọn “Insert Reference to” là “Heading number (no context)”. Sau đó, SV chọn phần đề mục muốn tham chiếu rồi bấm “Insert”.

Để tạo tham chiếu chéo tới các hình vẽ, bảng biểu và công thức, sinh viên cũng vào tab “References” và chọn “Cross-refrence”. SV chọn “Refrence type” là “Hình”, “Bảng”, hoặc “Công thức’. SV chọn “Insert Reference to” là “Only label and number”, trỏ đến phần muốn tham chiếu rồi bấm “Insert”. Nếu font chữ trong liên kết tham chiếu tạo ra được in đậm (bold), SV chuyển về dạng thường cho chuẩn tắc.

Thực hiện tương tự các bước mô tả ở trên, sinh viên có thể tạo tham chiếu chéo tới các tài liệu tham khảo. Sinh viên chọn “Refrence type” là “Numbered item”, chọn “Insert Reference to” là “Paragraph number”, trỏ đến phần tài liệu tham khảo muốn tham chiếu rồi bấm “Insert”. Ví dụ, tham chiếu chéo tới tài liệu tham khảo [2], [3], [4] đã được tạo.

Cập nhật mục lục và tham chiếu chéo

Trong suốt quá trình viết BTL, sinh viên sẽ tạo ra nhiều xáo trộn như thay đổi vị trí hình và bảng, thay đổi thứ tự tài liệu tham khảo, thay đổi tên đề mục, v.v. Vì vậy, để hoàn thành BTL, sinh viên cần cập nhật lại các thành phần mục lục, danh sách hình ảnh/bảng/công thức và tham chiếu chéo được tạo tự động trong BTL của mình.

Để cập nhật các thành phần này, sinh viên bấm Ctrl+A để chọn toàn báo cáo, nhấn chuột phải và chọn “Update Field”, rồi liên tục chọn mục “Entire table” khi được hỏi trong hộp thoại Popup Menu. Sau đó sinh viên search chữ “Error” trên toàn báo cáo để kiểm tra xem có lỗi đánh chỉ mục hoặc lỗi tham chiếu nào không. Đồng thời, sinh viên nên tự soát lại bằng tay toàn bộ nội dung quyển BTL để tránh mọi sai sót.

In quyển Bài tập lớn

Do hiện nay có nhiều phiên bản Word cho nhiều nền tảng máy tính, sinh viên nhất thiết phải xuất BTL ra định dạng PDF rồi mang tới cửa hàng in ấn để tránh sai sót. Quyển BTL nên được in một mặt trên các trang giấy A4.

Đặc tả use case

Nếu trong nội dung chính không đủ không gian cho các use case khác (ngoài các use case nghiệp vụ chính) thì đặc tả thêm cho các use case đó ở đây.

Đặc tả use case “Thống kê tình hình mượn sách”

Cách trình bày tương tự như phần hướng dẫn ở mục 2.3.

Đặc tả use case “Đăng ký làm thẻ mượn”

Cách trình bày tương tự như phần hướng dẫn ở mục 2.3.

Công nghệ sử dụng

Công nghệ bảo mật dữ liệu

Công nghệ blockchain

Thiết kế gói

Thiết kế gói cho kiến trúc tổng quan

Thiết kế gói cho chức năng “Trả sách”

Thiết kế lớp

1. https://www.tensorflow.org/, lần truy cập cuối: 28/06/2018 [↑](#footnote-ref-1)