KHOA KỸ THUẬT VÀ CÔNG NGHỆ

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN KẾT THÚC MÔN

CÔNG NGHỆ PHẦN MỀM

HỌC KỲ II, NĂM HỌC 2023-2024

TÊN ĐỀ TÀI

XÂY DỰNG ỨNG DỤNG DI ĐỘNG NGHE NHẠC SOUNDHUB

Sinh viên thực hiện:

110121063	Đinh Tấn Mãi	DA21TTB
110121119	Nguyễn Đình Trí	DA21TTB
110121047	Hứa Phước Lâm	DA21TTB

Giáo viên hướng dẫn: TS. Nguyễn Bảo Ân

Trà Vinh, tháng 06 năm 2024

KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN KẾT THÚC MÔN

CÔNG NGHỆ PHẦN MỀM

HỌC KỲ II, NĂM HỌC 2023-2024

TÊN ĐỀ TÀI

XÂY DỰNG ỨNG DỤNG DI ĐỘNG NGHE NHẠC SOUNDHUB

Sinh viên thực hiện:

110121063	Đinh Tấn Mãi	DA21TTB
110121119	Nguyễn Đình Trí	DA21TTB
110121047	Hứa Phước Lâm	DA21TTB

Giáo viên hướng dẫn: TS. Nguyễn Bảo Ân

Trà Vinh, tháng 06 năm 2024

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước hết, chúng em xin gửi lời cảm ơn chân thành đến thầy, giảng viên dạy môn Công nghệ phần mềm, cùng các bạn bè đã luôn quan tâm, giúp đỡ chúng em trong suốt quá trình học tập và thực hiện báo cáo này.

Nhờ sự hướng dẫn chi tiết, nhiệt tình và những góp ý quý báu của thầy, chúng em đã có thể hoàn thành bài báo cáo một cách tốt nhất. Chúng em đã học hỏi được rất nhiều kiến thức mới và kinh nghiệm thực tiễn từ những chỉ dẫn của thầy, đặc biệt là về các khía cạnh mà chúng em chưa từng tiếp xúc trước đây.

Bên cạnh đó, sự kiên nhẫn và sẵn lòng giải đáp mọi thắc mắc của thầy đã giúp em tự tin hơn trong quá trình nghiên cứu và hoàn thiện bài báo cáo.Chúng em thật sự trân trọng những đóng góp và thời gian mà thầy đã dành cho chúng em.

Một lần nữa, chúng em xin chân thành cảm ơn thầy.Chúng em hy vọng sẽ tiếp tục nhận được sự chỉ dẫn và hỗ trợ từ thầy trong những học kỳ tiếp theo.

Kính chúc thầy luôn dồi dào sức khỏe, hạnh phúc và thành công trong sự nghiệp giảng dạy.

Trân trọng,

MỤC LỤC

CHƯƠNG 1 GIỚI THIỆU 8
1.1 Bối cảnh nghiên cứu 8
1.2 Đặc tả ứng dụng 8
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT 9
2.1 Tổng quan về Agile 9
2.1.1 Giới thiệu Agile 9
2.1.2 Tìm hiểu chung về Agile 10
2.1.3 Các phương pháp Agile 12
2.2 Tổng quan về Scrum 14
2.2.1 Srum là gì ? 14
2.2.2 Mô tả về Scrum 15
2.2.3 Ba giá trị cốt lõi của Scrum 15
2.2.4 Các thành tố tạo nên Scrum? 16
2.2.5 Scrum vận hành như thế nào ? 20
2.3 Tổng quan các công nghệ sử dụng 21
2.3.1 React Native 21
2.3.2 Node.js 23
2.3.3 Express 25
2.3.4 MySQL 26
2.3.5 Docker 27
2.4 Kiến trúc hệ thống 29
2.4.1 Kiến trúc Client-Server 29
2.4.2 Mô hình MVC 30
CHƯƠNG 3 XÁC ĐỊNH NHU CẦU 32
3.1 Yêu cầu chức năng 32
3.2 Yêu cầu phi chức năng 32
CHƯƠNG 4 LẬP KẾ HOẠCH SCRUM 33
4.1 Lập kế hoạch sản phẩm (Product Backlog) 33
4.2 Lập kế hoạch sprint (Sprint Backlog) 35
CHƯƠNG 5 HIỆN THỰC HÓA KẾ HOẠCH 38
5.1 Kết quả đạt được 38
5.1.1 Sprint 1: Xây dựng cơ sở dữ liệu 38
5.1.2 Sprint 2: Tính năng đăng ký và đăng nhập 39

5.1.3 Sprint 3: Quản lý bài hát 41
5.1.4 Sprint 4: Quản lý danh sách phát 42
5.1.5 Sprint 5: Phát nhạc và triển khai 43
CHƯƠNG 6 KẾT LUẬN 46
6.1 Đánh giá tổng quan về dự án 46
6.2 Kết quả đạt được được 46
6.3 Hạn chế 46
6.4 Hướng phát triển trong tương lai 46

DANH SÁCH HÌNH ẢNH, BẢNG

Hình 1. Minh hoạ Agile 9
Hình 2. Phương pháp agile 12
Hình 3. Biểu đồ đồ khảo sát các phương pháp Agile 14
Hình 4. Minh hoạ quy trình Scrum 15
Hình 5. Ba giá trị cốt lõi của Scrum 15
Hình 6. Thành tố tạo nên Scrum 17
Hình 7. Minh hoạ Product Backlog 19
Hình 8. Minh hoạ Sprint Backlog 19
Hình 9. Minh hoạ Burndown chart 20
Hình 10. Minh hoạ vận hành Scrum 20
Hình 11. Minh hoạ Docker 27
Hình 12. Một số thành phần cơ bản của Docker 28
Hình 13. Kiến trúc hệ thống ứng dụng di động nghe nhạc SOUNDHUB 29
Hình 14. Các thành phần mô hình MVC 30
Hình 15. Các task của Epic Thiết kế cơ sở dữ liệu 33
Hình 16. Các task của Epic Triển khai cơ sở dữ liệu 33
Hình 17. Các task của Epic Đăng ký tài khoản 33
Hình 18. Các task của Epic Đăng nhập tài khoản 33
Hình 19. Các task của Epic Chức năng bài hát 34
Hình 20. Các task của Epic Chức năng tìm kiếm bài hát 34
Hình 21. Các task của Epic Chức năng danh sách phát 34
Hình 22. Các task của Epic Chức năng phát nhạc 34
Hình 23. Các task của Epic Chức năng thông tin người dùng 35
Hình 24. Các task của Epic Triển khai 35
Hình 25. Các task của Spring 1 35
Hình 26. Các task của Spring 2 36
Hình 27. Các task của Spring 3 36
Hình 28. Các task của Spring 4 37
Hình 29. Các task của Spring 5 37
Hình 30. Database diagram 38
Hình 31. Burndown chart của Sprint 1 39
Hình 32. Các giao diện figma của Sprint 1 41
Hình 33. Burndown chart của Sprint 2 41
Hình 34. Các giao diện figma của Sprint 2 42
Hình 35. Burndown chart của Sprint 3 42
Hình 36. Các giao diện figma của Sprint 3 43
Hình 37. Burndown chart của Sprint 4 43
Hình 38. Các giao diện figma của Sprint 5 45
Hình 39. Burndown chart của Sprint 5 45

CHƯƠNG 1 GIỚI THIỆU

1.1 Bối cảnh nghiên cứu

Trong kỷ nguyên số hiện nay, âm nhạc không chỉ là một phần của cuộc sống mà còn là một nhu cầu thiết yếu giúp mọi người giải trí, thư giãn, và kết nối với nhau. Sự bùng nổ của các dịch vụ phát nhạc trực tuyến đã thay đổi cách chúng ta tiêu thụ âm nhạc, mang đến những trải nghiệm mới mẻ và tiện lợi hơn. Các nền tảng phát nhạc như Spotify, Apple Music, và YouTube Music đã trở thành những cái tên quen thuộc, cung cấp cho người dùng hàng triệu bài hát chỉ với vài thao tác chạm.

Tuy nhiên, bên cạnh các nền tảng lớn này, vẫn còn nhiều khoảng trống và cơ hội để phát triển những ứng dụng âm nhạc với tính năng độc đáo, phục vụ những nhu cầu đặc thù của người dùng. Chính vì vậy, chúng tôi quyết định phát triển một ứng dụng âm nhạc mới, tập trung vào việc mang lại trải nghiệm cá nhân hóa và chất lượng cao cho người dùng.

1.2 Đặc tả ứng dụng

Ứng dụng sẽ cung cấp một trải nghiệm nghe nhạc đa dạng và giao diện thân thiện cho người dùng, bao gồm các tính năng chính như đăng nhập và đăng ký tài khoản, khám phá các playlist nhạc, tìm kiếm bài hát và nghệ sĩ, tạo và quản lý các danh sách phát cá nhân, theo dõi các nghệ sĩ yêu thích, nghe nhạc trực tuyến và nhiều tính năng khác nhằm tối ưu hóa trải nghiệm người dùng. Đồng thời, ứng dụng sẽ được xây dựng với kiến trúc hiện đại như Client-Server và sử dụng các công nghệ tiên tiến như React Native, Node.js để đảm bảo hiệu suất cao và khả năng mở rộng linh hoạt.

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về Agile

2.1.1 Giới thiệu Agile

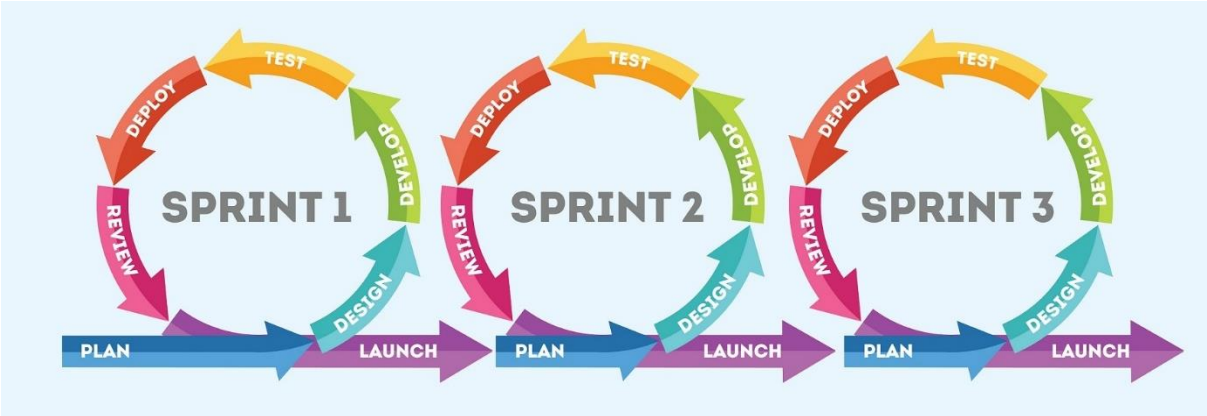
2.1.1.1 Bối cảnh

Có rất nhiều phương pháp khác nhau giúp xác định con đường phát triển phần mềm, bao gồm các quy trình như: mô hình thác nước, mô hình xoắn ốc, mô hình hướng đối tượng, mô hình làm bản mẫu.

Các phương pháp này chủ yếu dựa trên việc dự đoán trước quy trình phát triển phần mềm, tạo ra một bản kế hoạch từ giai đoạn đầu của dự án và xác định thời gian hoàn thành cụ thể. Tuy nhiên, một vấn đề quan trọng mà các phương pháp truyền thống gặp phải là "sự thay đổi yêu cầu người dùng". Các phương pháp này thường hạn chế sự thay đổi yêu cầu từ khách hàng để duy trì kế hoạch ban đầu của dự án, nhưng điều này lại không mang lại sự hài lòng tối đa cho khách hàng. Một phần mềm tốt là phần mềm đáp ứng được sự hài lòng của khách hàng.

Chính từ những hạn chế của các phương pháp truyền thống mà cần thiết phải có một phương pháp hoặc quy trình phát triển phần mềm mới. Quy trình phát triển phần mềm linh hoạt (Agile) đã ra đời và phần nào đáp ứng được yêu cầu này. Phương pháp quản lý dự án linh hoạt (Agile project management) xuất hiện từ đầu những năm 90 với mục đích khắc phục những nhược điểm của các mô hình truyền thống, đặc biệt là mô hình thác nước.

2.1.1.2 Agile là gì ?



Hình 1. Minh hoạ Agile

Agile là tên gọi chung để chỉ các phương pháp phát triển nhanh, linh hoạt. "Agile" nghĩa là nhanh nhẹn, khéo léo, linh hoạt.

Agile không phải là một phương pháp cụ thể mà là một triết lí cùng với nhóm các phương pháp và phương pháp luận phát triển sản phẩm dựa trên nguyên tắc phát triển phân đoạn lặp và tăng trưởng với mục tiêu là phần mềm phải có khả năng biến đổi, phát triển và tiến hóa theo thời gian mà không phải làm lại từ đầu.

Phương pháp Agile cố gắng cực tiểu hoá rủi ro bằng cách phát triển phần mềm trong những khung thời gian ngắn và sự cộng tác chặt chẽ với khách hàng. Điểm nổi bật là khả năng sửa chữa biến đổi phần mềm ngay cả khi dự án đã bắt đầu.

Điểm quan trọng làm lên sự khác biệt của Agile so với các mô hình truyền thống đó là: các mô hình truyền thống là mô hình theo kế hoạch, còn mô hình Agile thì không nhất thiết phải tuân theo kế hoạch, nó có thể có những bước đột phá để tạo ra một phần mềm hiệu quả nhất.

2.1.2 Tìm hiểu chung về Agile

2.1.2.1 Tuyên ngôn Agile

- *Cá nhân và tương tác hơn là quy trình và công cụ (Individuals and Interactions Over Processes and Tools):*

Đúng vậy, quy trình và công cụ đóng vai trò quan trọng trong việc phát triển phần mềm. Tuy nhiên, bản tuyên ngôn Agile nhấn mạnh rằng con người và sự tương tác giữa họ chìa khóa cho sự thành công. Agile đề cao tinh thần làm việc nhóm, khuyến khích chia sẻ thông tin cởi mở và hợp tác chặt chẽ để đạt được mục tiêu chung. Thay vì phụ thuộc hoàn toàn vào quy trình hay công cụ, các cá nhân trong nhóm Agile linh hoạt thích nghi và cùng nhau giải quyết vấn đề.

- *Phần mềm hoạt động hơn là tài liệu đầy đủ (Working Software Over Comprehensive Documentation):*

Agile không phủ nhận tầm quan trọng của tài liệu, nhưng nó đề cao giá trị thực tế của phần mềm hơn là việc tạo ra bộ tài liệu đồ sộ. Agile tập trung vào việc phát triển phần mềm có thể sử dụng được, liên tục thu thập phản hồi của khách hàng và cải tiến sản phẩm dựa trên phản hồi đó. Việc viết tài liệu chỉ được thực hiện khi cần thiết và tập trung vào những thông tin thiết yếu nhất cho người dùng.

- *Hợp tác với khách hàng hơn là đàm phán hợp đồng (Customer Collaboration Over Contract Negotiation):*

Mặc dù hợp đồng là cần thiết để xác định phạm vi và trách nhiệm của dự án, Agile đề cao mối quan hệ hợp tác giữa nhà phát triển và khách hàng. Agile khuyến khích giao tiếp thường xuyên, minh bạch để thấu hiểu nhu cầu và mong muốn của khách hàng. Khách hàng được xem như một phần của nhóm phát triển, tham gia đóng góp ý kiến và đưa ra phản hồi liên tục để sản phẩm được hoàn thiện theo đúng mục đích sử dụng.

- *Ứng phó, phản hồi với các thay đổi hơn là làm theo kế hoạch (Responding to Change Over Following a Plan):*

Kế hoạch đóng vai trò quan trọng trong việc định hướng dự án, nhưng Agile đề cao sự linh hoạt để thích ứng với những thay đổi không thể tránh khỏi. Thay vì cứng nhắc bám sát kế hoạch ban đầu, Agile khuyến khích tiếp nhận phản hồi của khách hàng, điều chỉnh kế hoạch phù hợp và sẵn sàng thay đổi hướng đi khi cần thiết. Nhờ vậy, sản phẩm cuối cùng đáp ứng tốt nhất nhu cầu của khách hàng và thị trường.

2.1.2.2 Nguyên tắc Agile

Dưới đây là 12 nguyên tắc của tuyên ngon Agile:

- Làm hài lòng khách hàng thông qua việc cung cấp phần mềm có giá trị liên tục và sớm: Ưu tiên sự hài lòng của khách hàng thông qua việc trao đổi và bàn giao liên tục, khách hàng sẽ cảm thấy hài lòng khi họ nhận được sản phẩm làm việc đều đặn thay vì chờ đợi thời gian kéo dài giữa các lần releases

- Chào đón việc thay đổi trong suốt quá trình phát triển: Tránh sự chậm trễ khi có yêu cầu thay đổi yêu cầu.

- Chuyển giao phần mềm sử dụng được định kỳ, từ một vài tuần đến một vài tháng, với một thời gian ngắn hơn.

- Khách hàng và đội phát triển phải làm việc cùng nhau hàng ngày trong suốt dự án

- Xây dựng các dự án xoay quanh các cá nhân có động lực. Tạo cho họ một môi trường và hỗ trợ họ những thứ cần thiết và tin tưởng họ để công việc được hoàn thành.

- Thúc đẩy các cuộc trò chuyện trực tiếp: Phương pháp hiệu quả nhất là truyền tải thông tin đến vào bên trong đội phát triển là hội thoại mặt-đối-mặt.

- Phần mềm làm việc được là thước đo của quá trình.

- Các quy trình Agile thúc đẩy sự phát triển bền vững. Các nhà tài trợ cho dự án, các nhà phát triển và người dùng cuối có thể duy trì một tốc độ vô hạn định.

- Liên tục quan tâm đến kỹ thuật xuất sắc và thiết kế tốt giúp nâng cao tính linh hoạt

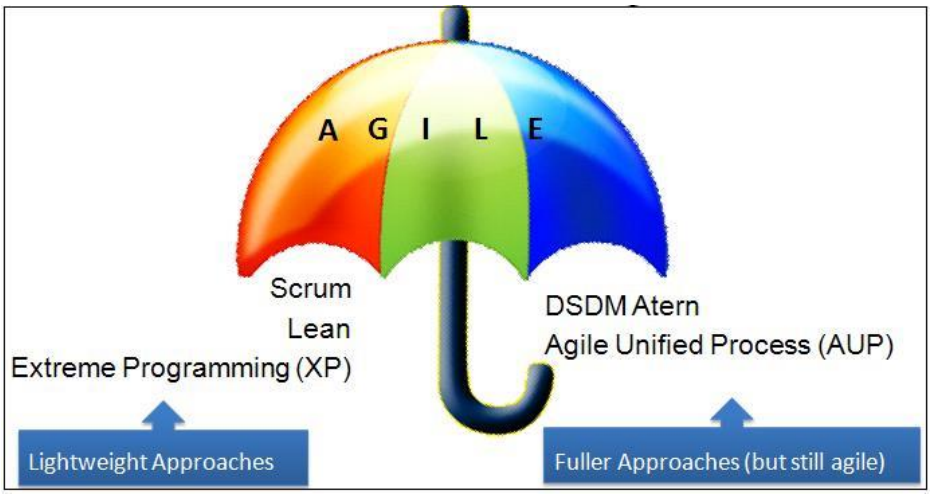
- Tính đơn giản – nghệ thuật tối đa hoá khối lượng công vệc chưa hoàn thành – là điều thiết yếu.

- Các kiến trúc, yêu cầu và thiết kế tốt nhất xuất hiện từ các nhóm tự tổ chức.

- Ở thời điểm kết thúc sprint, các team nên xem lại làm thế nào để hiệu quả hơn, sau đó cùng tự cải thiện ứng xử, cải tiến quy trình, kỹ thuật của mình sao cho phù hợp

2.1.3 Các phương pháp Agile

2.1.3.1 Phân loại



Hình 2. Phương pháp agile

Hiện nay có nhiều phương pháp Agile, nhưng thực sự chúng không phải là một phương pháp mà là một hệ thống các triết lý và tập hợp các giá trị và nguyên tắc.

Agile giống như một cây dù với nhiều phương pháp Agile khác nhau, có thể phân loại chúng thành 2 nhóm cơ bản: lightweight approaches(tiếp cận nhanh) và fuller approaches(tiếp cận đầy đủ hơn).

Trong đó:

- Lightweight approaches là các phương pháp như Scrum, Extreme programming, lean, …
- Fuller approaches là các phương pháp như Dynamic System Development Method DSDM Atern, Agile Unified Process(AUP),…

Sau đây là một số nguyên tắc của các phương pháp trên:

- Scrum: tập trung vào việc quản lý agile và tổ chức tốt hơn cho các đội phát triển.

- XP(Extreme Programming): trong phương pháp này có một số yếu tố quản lý, nhưng nhấn mạnh yếu tố thực hành kỹ thuật hơn các phương pháp khác.
- Lean Software Development: nhằm giảm tối thiểu chi phí đầu tư cho dự án.
- DSDM – Dynamic Systems Development Method: phương pháp này nhằm rút ngắn thời gian các vòng lặp(Sprint) của dự án.

2.1.3.2 Tỉ lệ được sử dụng của các phương pháp Agile

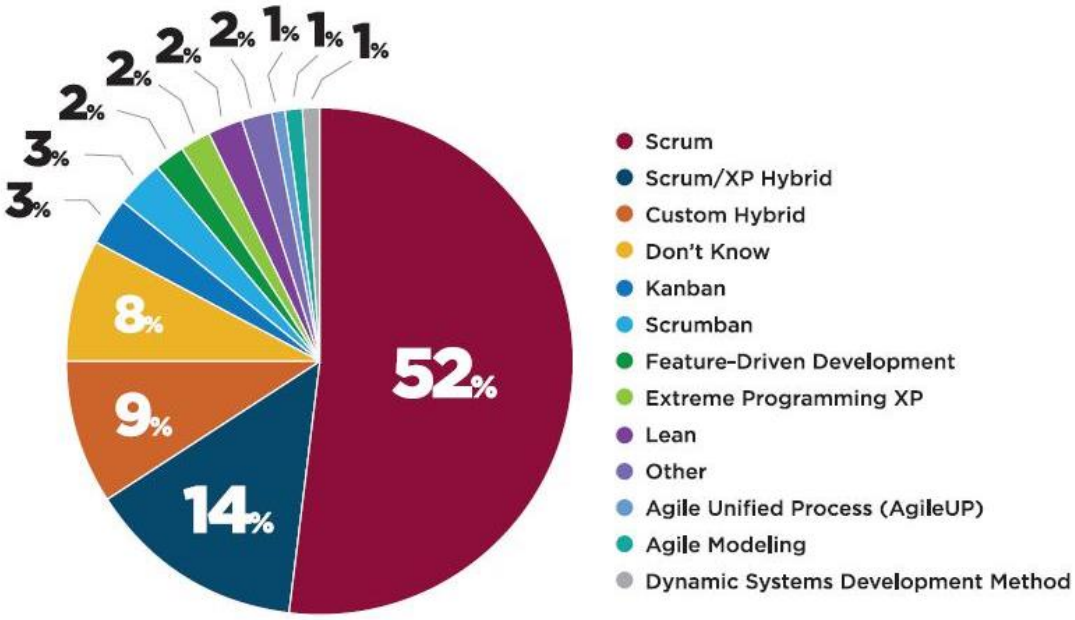
Theo khảo sát mới đây (2011) của Forrester Research, các cách tiếp cận phổ biến trong phát triển phần mềm có thể kể đến gồm Scrum, Iterative (Iterative Development - phát triển lặp), Waterfall, TDD, Kanban, XP.

Phương pháp	Tỉ lệ trả lời "có dùng"
Scrum	81.5%
Iterative	58.5%
Waterfall	44.4%
Test-Driven Development (TDD)	37.1%
Kanban	37.1%
eXtreme Programming (XP)	35.6%
Lean Software Development	32.7%

Bảng 1. Bảng khảo sát tỉ lệ sử dụng của các phương pháp Agile

Bảng thống kê này cho thấy rằng phần lớn các công ty hiện nay đã bắt đầu sử dụng Scrum như phương pháp tiếp cận chính. Ngoài ra, nhiều công ty kết hợp các phương pháp khác nhau trong thực tiễn. Ví dụ, 44.4% các công ty sử dụng Waterfall, điều này có nghĩa là một tỷ lệ đáng kể công ty vừa dùng Waterfall vừa sử dụng Scrum trong hoạt động của mình. Nguyên nhân của hiện tượng này có thể liên quan đến lịch sử phát triển, các ràng buộc về chính sách, luật pháp hoặc văn hóa. Đây cũng là một tình huống khá phổ biến đối với các công ty mới bắt đầu triển khai "ma trận các phương pháp Agile".

Bảng dưới đây của VersionOne khảo sát trên các công ty Agile, cho thấy những phương pháp phổ biến được sử dụng:



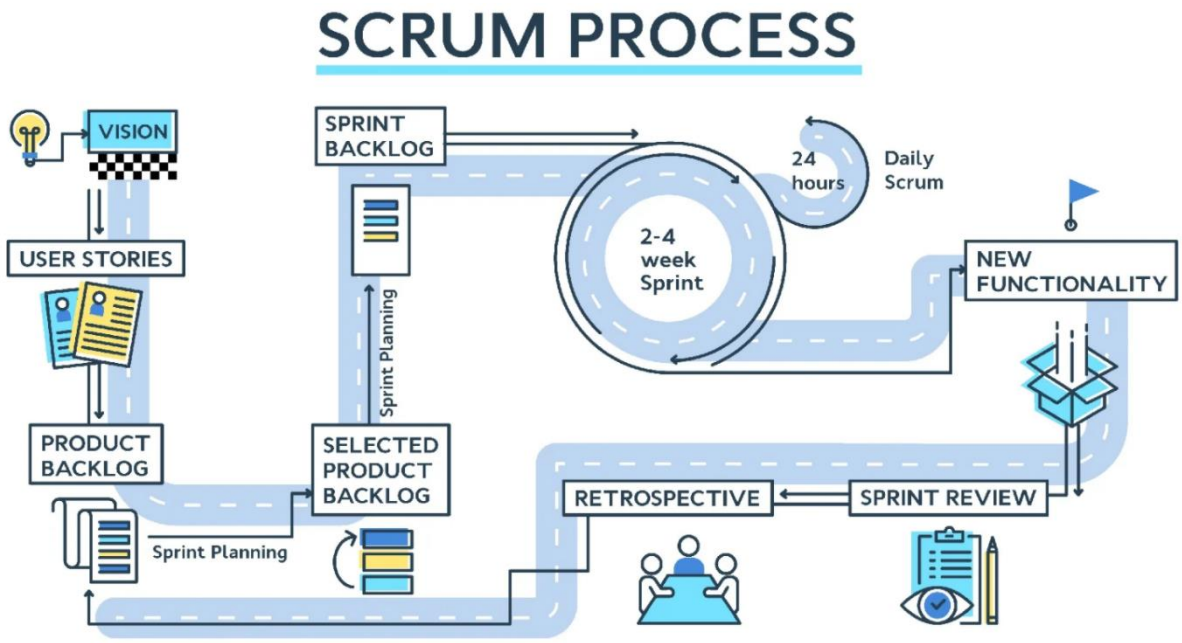
Hình 3. Biểu đồ đồ khảo sát các phương pháp Agile

2.2 Tổng quan về Scrum

2.2.1 Srum là gì ?

Scrum là một phương pháp phát triển phần mềm theo mô hình Agile, nổi bật nhất hiện nay, được áp dụng cho các dự án phần mềm từ đơn giản đến phức tạp. Phương pháp này được Ken Schwaber và Jeff Sutherland phát triển từ đầu những năm 1990. Ngày nay, nguyên tắc và cách thức của Scrum không chỉ giới hạn trong lĩnh vực phát triển phần mềm mà còn mở rộng sang các lĩnh vực khác như dịch vụ, giáo dục, tiếp thị, v.v. Định nghĩa chính thức về Scrum hiện được trình bày trong tài liệu Scrum Guide và được cập nhật định kỳ bởi các tác giả tại http://scrum.org.

2.2.2 Mô tả về Scrum

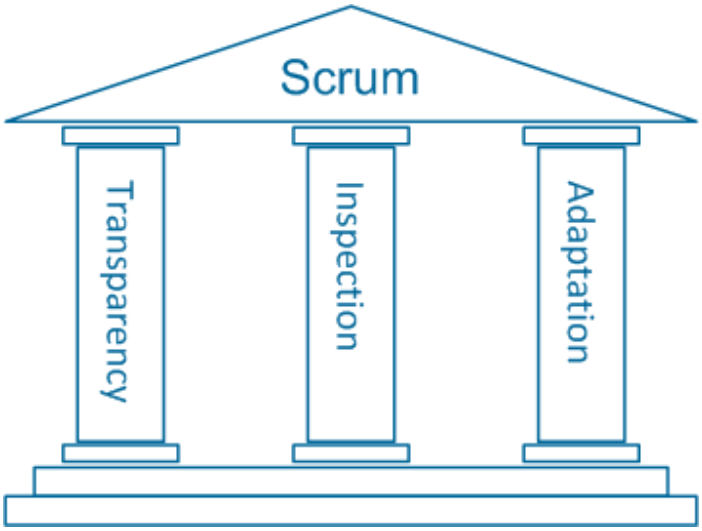


Hình 4. Minh hoạ quy trình Scrum

Scrum chia dự án thành các vòng lặp phát triển được gọi là sprint. Mỗi sprint thường kéo dài từ 2 đến 4 tuần (30 ngày) để hoàn thành. Phương pháp này đặc biệt thích hợp cho những dự án có nhiều thay đổi và yêu cầu tốc độ cao.

Mỗi sprint hoàn thành một số chức năng hoặc mục tiêu cụ thể trong toàn bộ hệ thống. Các nhiệm vụ trong sprint được phân chia thành các danh mục, và đội ngũ sẽ phát triển, đánh giá lại để đảm bảo đạt được mục tiêu ban đầu trong thời gian đã định.

2.2.3 Ba giá trị cốt lõi của Scrum



Hình 5. Ba giá trị cốt lõi của Scrum

Scrum là một phương pháp Agile, tuân thủ các nguyên tắc của Agile Manifesto (Tuyên ngôn Agile). Scrum hoạt động dựa trên ba trụ cột chính: Minh bạch, Thanh tra và Thích nghi.

- *Minh bạch (Transparency)*

Trong Scrum, tính minh bạch là giá trị cốt lõi quan trọng nhất. Để thành công với Scrum, thông tin liên quan đến quá trình phát triển phải rõ ràng và dễ tiếp cận. Các thông tin này bao gồm tầm nhìn về sản phẩm, yêu cầu của khách hàng, tiến độ công việc, các khó khăn và trở ngại, v.v. Nhờ vậy, mọi người trong đội ngũ có đủ thông tin cần thiết để đưa ra các quyết định quan trọng, giúp nâng cao hiệu quả công việc. Các công cụ và cuộc họp trong Scrum luôn đảm bảo thông tin được minh bạch cho tất cả các bên liên quan.

- *Thanh tra (Inspection)*

Việc thanh tra liên tục các hoạt động trong Scrum giúp phát hiện sớm các vấn đề và giải pháp, đảm bảo thông tin đa dạng và hữu ích đến được với tất cả các bên tham gia dự án. Việc kiểm tra kỹ lưỡng và thường xuyên là cơ chế quan trọng để thúc đẩy sự thích nghi và cải tiến liên tục trong Scrum.

- *Thích nghi (Adaptation)*

Scrum, giống như các phương pháp phát triển Agile khác, rất linh hoạt. Dựa trên thông tin minh bạch từ quá trình thanh tra và công việc, Scrum có thể phản ứng nhanh chóng và tích cực với các thay đổi, giúp dự án đạt được thành công.

2.2.4 Các thành tố tạo nên Scrum?

Scrum bao gồm các giá trị cốt lõi (còn gọi là "ba trụ cột của Scrum") là các vai trò (Roles) , các sự kiện (Events) và các công cụ đặc thù của Scrum (Artifacts) .

Hình 6. Thành tố tạo nên Scrum

2.2.4.1 Ba Vai trò (3 Roles)

Trong Scrum, đội ngũ phát triển phần mềm được chia thành ba vai trò với trách nhiệm rõ ràng để đảm bảo tối ưu hóa các công việc đặc thù. Ba vai trò này bao gồm: Chủ sản phẩm (Product Owner), Scrum Master và Nhóm Phát triển (Development Team).

- *Product Owner (Chủ sản phẩm)*

Người chịu trách nhiệm về sự thành công của dự án, được coi là tiếng nói của khách hàng. Chủ sản phẩm đại diện cho khách hàng trong các cuộc họp lập kế hoạch, xác định các yêu cầu và đánh giá kết quả cuối cùng của các nhà phát triển phần mềm.

- *Scrum Master*

Người có hiểu biết sâu sắc về Scrum và đảm bảo rằng nhóm có thể làm việc hiệu quả theo phương pháp Scrum. Scrum Master hỗ trợ nhóm trong việc tuân thủ các quy tắc và quy trình của Scrum.

- *Development Team (Nhóm Phát triển)*

Một nhóm liên chức năng (cross-functional) tự quản lý, chuyển đổi các yêu cầu được tổ chức trong Product Backlog thành các chức năng cụ thể của hệ thống. Nhóm Phát triển chịu trách nhiệm về việc hoàn thành các công việc cần thiết để tạo ra các tính năng phần mềm.

2.2.4.2 Bốn Cuộc họp (4 Events)

Scrum định nghĩa bốn sự kiện chính (các cuộc họp) để tạo môi trường và quy cách hoạt động, cộng tác cho các thành viên trong dự án. Những nghi thức này diễn ra trước khi Sprint bắt đầu (Sprint Planning), trong khi Sprint diễn ra (Daily Scrum), và sau khi Sprint kết thúc (Sprint Review và Sprint Retrospective).

- *Sprint Planning (Họp Kế hoạch Sprint*)

Diễn ra đầu mỗi sprint, nhóm phát triển gặp gỡ với Chủ sản phẩm (Product Owner) để lên kế hoạch làm việc cho Sprint. Công việc lập kế hoạch bao gồm việc chọn các yêu cầu cần phát triển, phân tích và xác định các công việc cần làm kèm theo ước lượng thời gian cần thiết để hoàn thành. Scrum sử dụng cách thức lập kế hoạch từng phần và tăng dần theo thời gian, tức là việc lập kế hoạch không diễn ra duy nhất một lần trong vòng đời dự án mà được lặp đi lặp lại, thích nghi với tình hình thực tiễn trong quá trình phát triển sản phẩm.

- *Daily Scrum (Họp Scrum hàng ngày)*

Scrum Master tổ chức cho Nhóm Phát triển họp hàng ngày trong khoảng 15 phút. Cuộc họp này cho phép các thành viên chia sẻ tiến độ công việc cũng như các khó khăn gặp phải trong quá trình phát triển phần mềm suốt một Sprint.

- *Sprint Review (Họp Sơ kết Sprint)*

Cuối mỗi Sprint, nhóm phát triển cùng với Chủ sản phẩm sẽ xem xét lại các công việc đã hoàn thành trong Sprint vừa qua và đề xuất các chỉnh sửa hoặc thay đổi cần thiết cho sản phẩm.

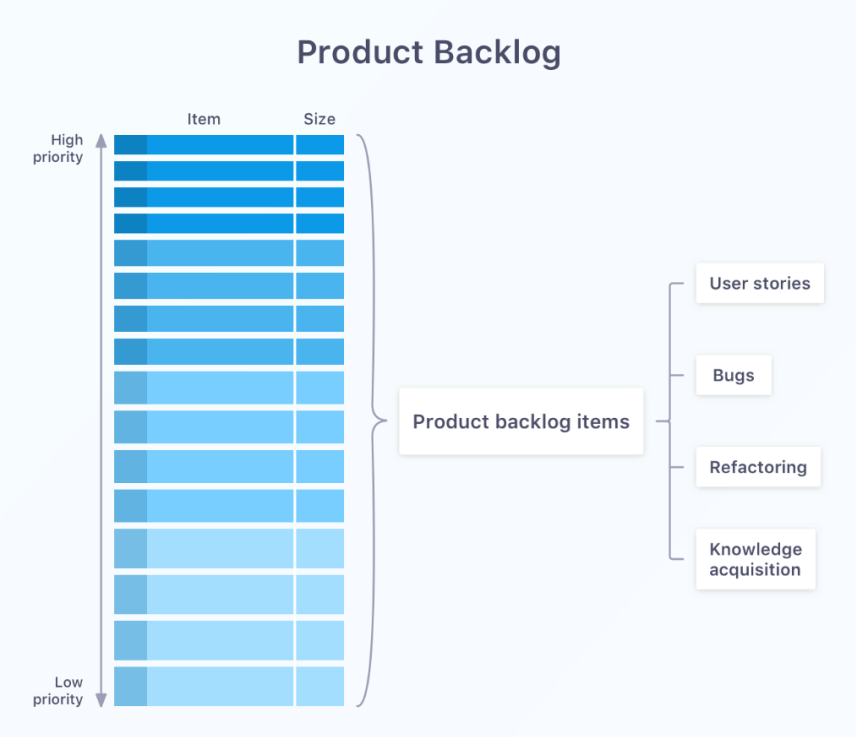
- *Sprint Retrospective (Họp Cải tiến Sprint)*

Dưới sự hướng dẫn của Scrum Master, nhóm phát triển sẽ xem xét lại toàn bộ Sprint vừa kết thúc và tìm cách cải tiến quy trình làm việc cũng như chất lượng sản phẩm.

2.2.4.3 Các công cụ (Artifacts)

Scrum sử dụng các công cụ rất đơn giản nhưng hiệu quả để trợ giúp công việc. Chúng bao gồm bản yêu cầu của chủ sản phẩm được gọi là Product backlog, bản kế hoạch của từng Sprint (Sprint Backlog) và biểu đồ Burndown Chart.

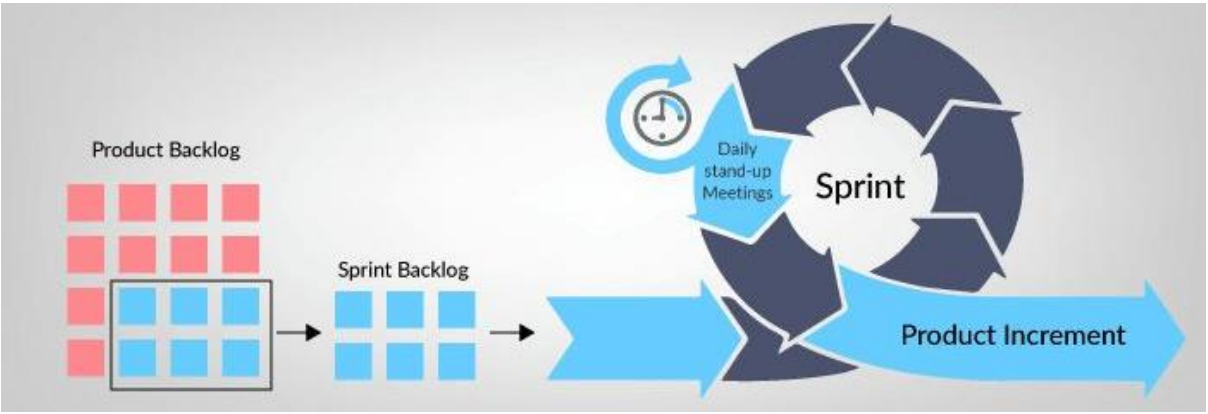
- *Product backlog*



Hình 7. Minh hoạ Product Backlog

Đây là danh sách ưu tiên các tính năng (feature) hoặc đầu ra khác của dự án, có thể hiểu như là danh sách yêu cầu (requirement) của dự án. Các mục trong danh sách này được gọi là các mục tồn đọng của sản phẩm (Product Backlog Items - PBIs). Product Owner chịu trách nhiệm sắp xếp độ ưu tiên cho từng hạng mục (Product Backlog Item) trong Product Backlog dựa trên các giá trị do Product Owner định nghĩa (thường là giá trị thương mại – business value).

- *Sprint backlog*

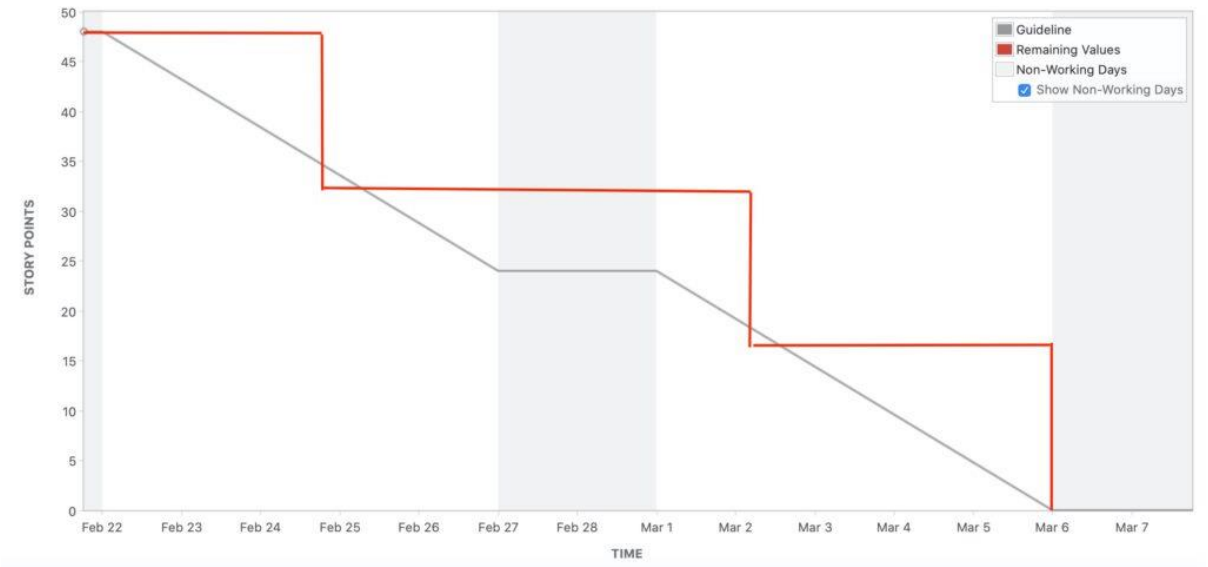


Hình 8. Minh hoạ Sprint Backlog

Đây là bản kế hoạch cho một Sprint; là kết quả của buổi họp lập kế hoạch (Sprint Planning). Với sự kết hợp của Product Owner, nhóm sẽ phân tích các yêu cầu theo độ

ưu tiên từ cao xuống thấp để hiện thực hóa các hạng mục trong Product Backlog dưới dạng danh sách công việc (TODO list).

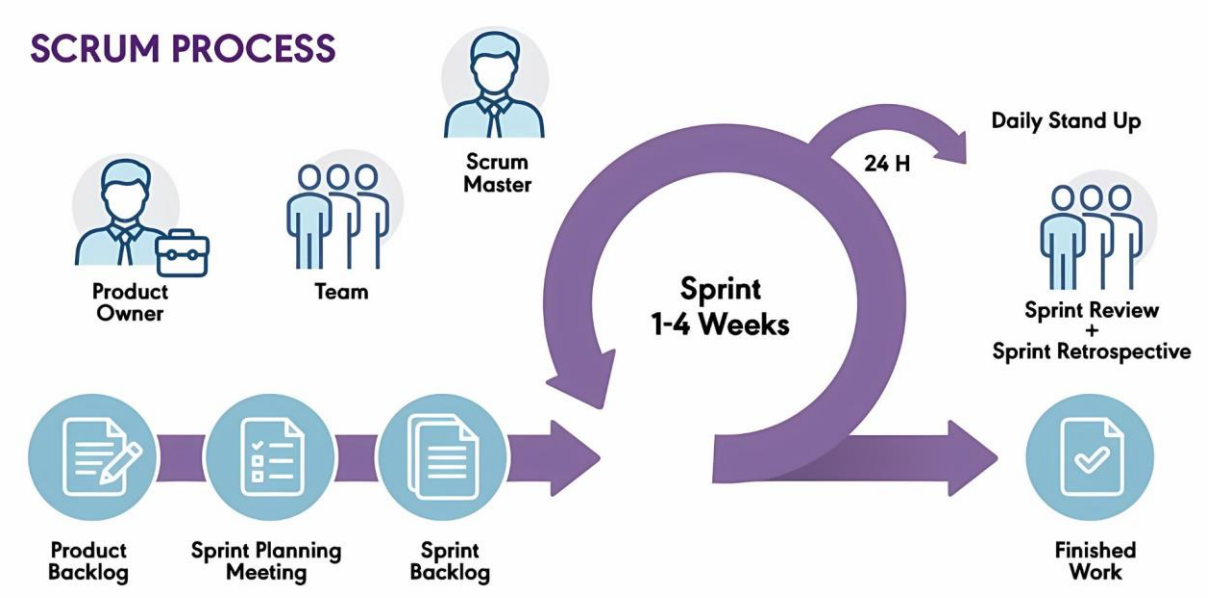
- *Burndown Chart*



Hình 9. Minh hoạ Burndown chart

Đây là biểu đồ hiển thị xu hướng của dự án dựa trên lượng thời gian cần thiết còn lại để hoàn tất công việc. Burndown Chart có thể được dùng để theo dõi tiến độ của Sprint (được gọi là Sprint Burndown Chart) hoặc của cả dự án (Project Burndown Chart). Biểu đồ burndown không phải là một thành tố tiêu chuẩn của Scrum theo định nghĩa mới, nhưng vẫn được sử dụng rộng rãi do tính hữu ích của nó.

2.2.5 Scrum vận hành như thế nào ?



Hình 10. Minh hoạ vận hành Scrum

Product Owner tạo ra Product Backlog chứa các yêu cầu của dự án với các hạng mục được sắp xếp theo thứ tự ưu tiên. Đội phát triển sẽ thực hiện hiện thực hóa dần các yêu cầu của Product Owner qua các giai đoạn Sprint từ 1 đến 4 tuần làm việc, với đầu vào là các hạng mục trong Product Backlog và đầu ra là các gói phần mềm hoàn chỉnh có thể chuyển giao được (Potentially Shippable Product Increment).

Trước khi bắt đầu mỗi Sprint, đội phát triển cùng Product Owner họp để lập kế hoạch cho Sprint. Kết quả của buổi lập kế hoạch này là Sprint Backlog chứa các công việc cần làm trong suốt Sprint. Trong quá trình phát triển, nhóm sẽ cập nhật Sprint Backlog và tham gia họp hàng ngày (Daily Scrum) để chia sẻ tiến độ công việc và giải quyết các vướng mắc.

Nhóm phát triển được trao quyền tự quản lý và tổ chức công việc để hoàn thành nhiệm vụ trong Sprint. Khi kết thúc Sprint, nhóm tạo ra các gói phần mềm có chức năng hoàn chỉnh, sẵn sàng chuyển giao (shippable) cho khách hàng. Buổi họp Sơ kết Sprint (Sprint Review) diễn ra vào cuối Sprint giúp khách hàng thấy được những gì đã hoàn thành, còn những gì phải làm hoặc cần thay đổi.

Sau buổi đánh giá Sprint, Scrum Master và nhóm tổ chức họp Cải tiến Sprint (Sprint Retrospective) để tìm kiếm các cải tiến trước khi Sprint tiếp theo bắt đầu, giúp nhóm liên tục học hỏi và trưởng thành qua từng Sprint.

Các Sprint sẽ được lặp lại cho tới khi các hạng mục trong Product Backlog hoàn tất hoặc khi Product Owner quyết định dừng dự án căn cứ vào tình hình thực tế. Với chiến thuật "ưu tiên giá trị cao", các hạng mục mang lại nhiều giá trị nhất cho chủ dự án luôn được hoàn thành trước. Do đó, Scrum luôn mang lại giá trị cao nhất cho nhà đầu tư dự án. Đồng thời, do quy trình liên tục được cải tiến, nhóm Scrum thường có năng suất lao động rất cao. Đây là hai lợi ích to lớn mà Scrum mang lại cho tổ chức.

2.3 Tổng quan các công nghệ sử dụng

2.3.1 React Native

2.3.1.1 Giới thiệu React Native

React Native là một framework mã nguồn mở được phát triển bởi Facebook, cho phép các nhà phát triển xây dựng các ứng dụng di động sử dụng JavaScript và React. Ra mắt lần đầu tiên vào năm 2015, React Native đã nhanh chóng trở thành một trong những công cụ phổ biến nhất để phát triển ứng dụng di động nhờ vào khả năng cho phép tạo ra

các ứng dụng có hiệu suất gần như ứng dụng gốc mà chỉ cần viết mã một lần cho cả hai nền tảng iOS và Android. Một số ứng dụng nổi tiếng sử dụng React Native như: Facebook, Instagram, Airbnb, Uber, Pinterest,…

2.3.1.2 Thế mạnh React Native

Một số thế mạnh có thể kể đến của React Native như:

- Cross-Platform Development: Viết mã một lần và chạy trên nhiều nền tảng (iOS, Android).

- Native Performance: Sử dụng các thành phần giao diện người dùng gốc, mang lại hiệu suất và trải nghiệm người dùng tốt nhất.

- Live Reloading và Hot Reloading: Cho phép nhà phát triển xem ngay lập tức các thay đổi trong mã nguồn mà không cần phải khởi động lại toàn bộ ứng dụng.

- Modular Architecture: Giúp dễ dàng duy trì và mở rộng ứng dụng.

- Strong Community and Ecosystem: Được hỗ trợ bởi một cộng đồng lớn với nhiều thư viện và công cụ hữu ích.

2.3.1.3 Một số kiến thức cơ bản ReactNative

a*) Cấu trúc cơ bản của một ứng dụng React Native:*

Một ứng dụng React Native thường bắt đầu với một thành phần chính (component) được đăng ký trong AppRegistry. Thành phần này chứa các thành phần con khác để tạo nên giao diện người dùng.

b) Thành phần (Components):

Các thành phần là khối xây dựng cơ bản của giao diện người dùng trong React Native. Có thể sử dụng các thành phần gốc như View, Text, Image hoặc tạo ra các thành phần tùy chỉnh.

c) StyleSheet:

StyleSheet là một API để định dạng giao diện người dùng, tương tự như CSS nhưng với cú pháp JavaScript. Nó giúp định nghĩa và áp dụng các kiểu dáng (styles) cho các thành phần trong ứng dụng.

d) State và Props:

State là một đối tượng được quản lý nội bộ bởi thành phần, dùng để lưu trữ dữ liệu thay đổi theo thời gian và ảnh hưởng đến giao diện người dùng.

Props là các thuộc tính được truyền từ thành phần cha xuống thành phần con, giúp truyền dữ liệu và các hàm giữa các thành phần.

e) Navigation (Điều hướng):

Để điều hướng giữa các màn hình trong ứng dụng, React Native sử dụng các thư viện như react-navigation, cung cấp các công cụ để tạo ra các luồng điều hướng phức tạp.

f) API tương tác với thiết bị:

React Native cung cấp các API để tương tác với các tính năng của thiết bị như camera, GPS, thông báo đẩy, v.v. Các API này có thể được truy cập trực tiếp hoặc thông qua các thư viện của bên thứ ba.

g) Context API và Redux:

Để quản lý trạng thái toàn cục của ứng dụng, React Native có thể sử dụng Context API hoặc Redux. Context API là giải pháp tích hợp sẵn trong React, trong khi Redux là một thư viện quản lý trạng thái phổ biến cung cấp một cách tiếp cận có cấu trúc hơn.

2.3.2 Node.js

2.3.2.1 Giới thiệu Node.js

Node.js, một môi trường mã nguồn mở được Ryan Dahl phát triển vào năm 2009, nhằm thực thi mã JavaScript trên máy chủ, tập trung giải quyết các vấn đề về hiệu suất trong việc xử lý thời gian truyền thông mạng, xử lý yêu cầu, và phản hồi web. Dựa trên Chrome’s V8 JavaScript runtime, Node.js được xây dựng với mục tiêu tạo ra ứng dụng mạng nhanh chóng và có khả năng mở rộng.

Node.js sử dụng mô hình event-driven và I/O non-blocking để tạo ra các ứng dụng nhẹ và hiệu năng cao, đặc biệt phù hợp cho các ứng dụng thời gian thực chạy trên các thiết bị phân tán. Bên cạnh đó, Node.js cung cấp một loạt các module JavaScript đa dạng, từ plugin, add-ons đến extensions, giúp đơn giản hóa quá trình phát triển ứng dụng web. Các module của Node.js được phân chia thành ba loại chính: module lõi, module của bên thứ ba và module do nhà phát triển tự xây dựng.

Với sự linh hoạt của JavaScript, ứng dụng Node.js có thể chạy một cách dễ dàng trên nhiều máy chủ với các hệ điều hành khác nhau như Linux, macOS, Microsoft Windows, và Unix. Điều này tăng tính tương thích và giúp nhà phát triển triển khai ứng dụng trên nhiều môi trường mà không gặp phải vấn đề tương thích lớn.

2.3.2.2 Các đặt trưng của Node.js

- *Đơn luồng (single thread)*

Node.js được thiết kế với đơn luồng (single thread), chủ yếu sử dụng để xây dựng ứng dụng máy chủ web. Mặc dù chỉ sử dụng một single-thread, Node.js có khả năng xử lý nhiều kết nối đồng thời, giúp tiết kiệm RAM và tăng tốc độ so với việc tạo thread mới cho mỗi truy vấn, như trong PHP.

- *Non-blocking*

Đặc điểm quan trọng khác của Node.js là tính non-blocking. Trái ngược với PHP, hầu hết các hàm chức năng trong Node.js được thiết kế để làm việc theo cách non-blocking. Điều này cho phép Node.js thực thi nhiều yêu cầu đồng thời mà không chờ đợi các thao tác trước hoàn thành. Khi cần thực hiện một thao tác, Node.js sẽ gửi nhiệm vụ đó đến vòng lặp sự kiện, tạo một hàm callback, và tiếp tục xử lý các thao tác khác. Điều này giúp tối ưu hóa sử dụng chu kỳ clock và tăng cường hiệu suất của máy chủ.

2.3.2.3 Thế mạnh của Node.js

Hiện nay, Node.js đã trở nên rất phổ biến và thường được các nhà phát triển lựa chọn làm giải pháp phát triển backend. Có nhiều lý do cho sự phổ biến của Node.js và tại sao nên sử dụng Node.js để phát triển server-side:

- *Hiệu suất cao:* Node.js giải quyết nhu cầu hỗ trợ số lượng người dùng lớn và cung cấp trải nghiệm thời gian thực. Với đơn luồng và non-blocking, máy chủ Node.js có thể phục vụ nhiều kết nối cùng lúc với thời gian phản hồi nhanh. Điều này tạo ra môi trường phát triển ứng dụng có hiệu suất cao, ít tốn kém và có khả năng mở rộng.

- *Ngôn ngữ JavaScript thông dụng:* JavaScript là ngôn ngữ phổ biến với cộng đồng lớn và ngày càng phát triển. Node.js sử dụng JavaScript, giúp nhà phát triển tiếp cận hàng ngàn module miễn phí một cách đơn giản và hiệu quả.

- *Môi trường phát triển đơn giản, thời gian phát triển nhanh:* Việc thiết lập môi trường phát triển với Node.js là đơn giản, chỉ cần tải Node qua npm và có thể xây dựng ứng dụng ngay. Node.js chạy trên các nền tảng mở như HTML và JavaScript, không yêu cầu cài đặt phần mềm bên thứ ba để chạy ứng dụng.

Với ưu điểm về tốc độ, hiệu suất, và sự đa dạng về module mở rộng, Node.js là giải pháp tối ưu cho hệ thống thời gian thực, đồng thời mang lại thời gian phát triển nhanh và sử dụng ngôn ngữ JavaScript phổ biến.

2.3.3 Express

2.3.3.1 Giới thiệu Express

Express (hay Express.js) là một framework web Node.js được sáng tạo bởi TJ Holowaychuk và đội ngũ phát triển. Với thiết kế đơn giản và linh hoạt, Express giúp người phát triển xây dựng ứng dụng web và API một cách nhanh chóng. Ưu điểm của nó bao gồm độ nhẹ, dễ học, và hỗ trợ middleware mạnh mẽ, tạo ra một cơ sở cho việc phát triển ứng dụng Node.js hiệu quả.

2.3.3.2 Thế mạnh của Express

Express, một framework web dựa trên Node.js, nổi bật trong lĩnh vực công nghệ với nhiều ưu điểm quan trọng. Điểm mạnh đầu tiên là tính nhẹ nhàng và đơn giản của Express, giúp giảm thiểu độ phức tạp và tăng cường linh hoạt trong quá trình phát triển ứng dụng.

Hệ thống middleware của Express là một điểm độc đáo khác, cho phép tích hợp dễ dàng các chức năng bảo mật và xử lý yêu cầu. Điều này không chỉ giúp đảm bảo an toàn mà còn nâng cao hiệu suất của ứng dụng thông qua các bước xử lý được tối ưu hóa.

Express không chỉ là một lựa chọn đơn giản mà còn là sự kết hợp giữa sự linh hoạt và hiệu quả. Sự đơn giản của nó cũng đồng nghĩa với việc dễ học và triển khai, giúp các nhà phát triển tiết kiệm thời gian và công sức trong quá trình xây dựng ứng dụng web. Với cộng đồng lớn và sự duy trì đều đặn, Express là một công cụ mạnh mẽ, chủ động đóng góp vào sự hiệu quả và tiện lợi của quá trình phát triển ứng dụng web trong thế giới công nghệ ngày nay.

2.3.3.3 Một số kiến thức cơ bản về Express

- *Middleware:*

Middleware là các hàm được thực thi tuần tự trước khi yêu cầu đến đích. Chúng có thể thay đổi đối tượng yêu cầu (req) và đối tượng phản hồi (res). Middleware có thể được sử dụng để thực hiện các nhiệm vụ như xác thực, xử lý lỗi, ghi log, và các chức năng khác. Middleware có thể là toàn cục (áp dụng cho mọi yêu cầu) hoặc chỉ áp dụng cho một nhóm các đường dẫn cụ thể.

- *Routing:*

Express sử dụng hệ thống định tuyến để xác định cách xử lý yêu cầu dựa trên URL và phương thức HTTP. Định tuyến được thiết lập bằng cách ánh xạ URL đến các hàm xử lý cụ thể.

- *Request và Response Objects:*

Đối tượng req chứa thông tin về yêu cầu từ client như tham số, tiêu đề và nội dung. Đối tượng res chứa các phương thức để gửi phản hồi về client như send, json, và render.

- *Template Engines:*

Express hỗ trợ tích hợp với các template engine như EJS, Pug, Handlebars để tạo giao diện người dùng dễ dàng. Template engine giúp hiển thị dữ liệu động trong HTML.

2.3.4 MySQL

2.3.4.1 Giới thiệu MySQL

MySQL, là hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến nhất thế giới, được ưa chuộng rộng rãi trong quá trình phát triển ứng dụng. Được biết đến với tốc độ cao, tính ổn định và sự dễ sử dụng, MySQL cung cấp tính khả chuyển, hoạt động trên nhiều hệ điều hành, và cung cấp một loạt các chức năng mạnh mẽ. Với hiệu suất và tính bảo mật cao, MySQL là lựa chọn lý tưởng cho các ứng dụng yêu cầu truy cập cơ sở dữ liệu trên internet.

MySQL được cung cấp hoàn toàn miễn phí, cho phép người dùng tải về từ trang chủ chính thức. Hệ quản trị cơ sở dữ liệu này có nhiều phiên bản tương ứng với các hệ điều hành khác nhau, bao gồm phiên bản Win32 dành cho hệ điều hành Windows, cũng như phiên bản hỗ trợ Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS, và nhiều hệ điều hành khác.

2.3.4.2 Thế mạnh MySQL

MySQL, là cơ sở dữ liệu mã nguồn mở phổ biến nhất trên toàn cầu, đạt được sự ưa chuộng bởi khả năng xử lý nhanh, tính ổn định, độ tin cậy cao, và khả năng sử dụng một cách dễ dàng. Được ứng dụng rộng rãi từ các nhà phát triển web cá nhân đến các tổ chức lớn trên thế giới, MySQL là sự lựa chọn thông minh để tiết kiệm thời gian và chi phí cho các trang web có lượng dữ liệu lớn và phần mềm đóng gói. Các công ty hàng

đầu như Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube và Zappos.com đều tin dùng MySQL trong các dự án của mình.

MySQL không chỉ là cơ sở dữ liệu mã nguồn mở phổ biến nhất trên thế giới mà còn trở thành lựa chọn hàng đầu cho thế hệ mới của các ứng dụng xây dựng trên nền tảng Linux, Apache, MySQL, và PHP/Perl/Python (LAMP). Với khả năng chạy trên hơn 20 nền tảng khác nhau như Linux, Windows, OS/X, HP-UX, AIX, Netware, MySQL mang lại sự linh hoạt cho người sử dụng.

Các ưu điểm của MySQL bao gồm tính linh hoạt, khả năng thực thi cao, khả năng sử dụng ngay lập tức, hỗ trợ giao dịch mạnh mẽ, độ tin cậy cao cho việc lưu trữ web và dữ liệu, chế độ bảo mật dữ liệu mạnh mẽ, khả năng phát triển ứng dụng hỗn hợp, dễ dàng quản lý, mã nguồn mở tự do và hỗ trợ liên tục 24/7. Tất cả những điều này đồng tục giúp giảm thiểu tổng chi phí sử dụng cơ sở dữ liệu.

2.3.5 Docker

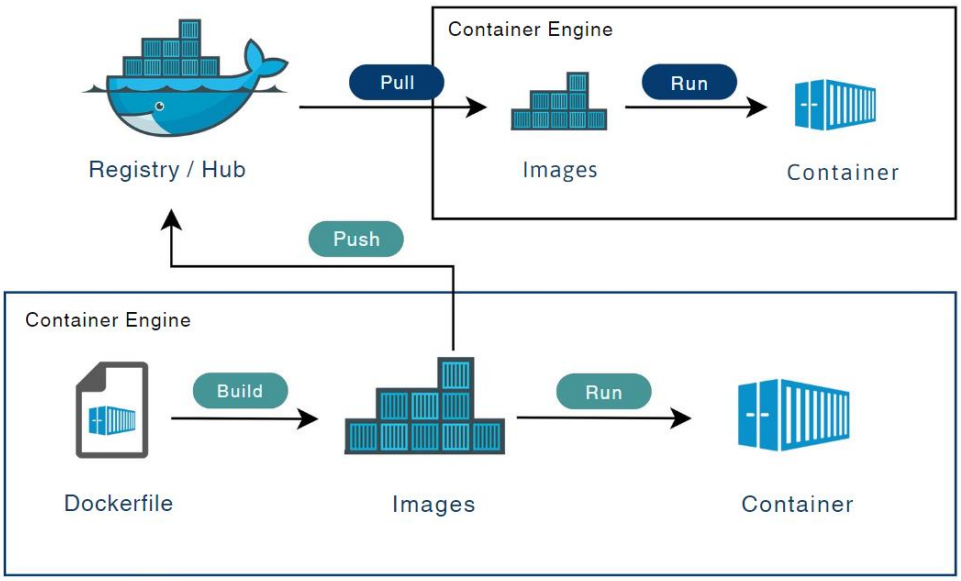
2.3.5.1 Giới thiệu Docker

Hình 11. Minh hoạ Docker

Docker ra đời trong bối cảnh của một thế giới phát triển phần mềm ngày càng phức tạp và đa dạng. Trước đây, việc đóng gói và triển khai ứng dụng có thể gặp phải nhiều vấn đề do sự không đồng nhất giữa môi trường phát triển và môi trường sản phẩm cuối cùng. Điều này gây ra những vấn đề về khả năng di động, đồng nhất hóa và tự động hóa trong quá trình phát triển phần mềm.

Docker giải quyết những vấn đề này bằng cách cung cấp một nền tảng đóng gói ứng dụng trong các container độc lập. Mỗi container chứa tất cả các thành phần cần thiết để chạy một ứng dụng, bao gồm cả mã, thư viện và cài đặt hệ thống. Điều này giúp đơn giản hóa việc triển khai ứng dụng trên bất kỳ môi trường nào mà không cần lo lắng về sự khác biệt giữa các môi trường đó.

2.3.5.2 Một số khái niệm Docker



Hình 12. Một số thành phần cơ bản của Docker

Một container có thể được hiểu đơn giản như một môi trường độc lập, cung cấp mọi thứ cần thiết để chạy ứng dụng của bạn. Giống như việc sử dụng môi trường ảo trong Python (như virtual environment), container cung cấp một cách để đóng gói và chia sẻ ứng dụng cùng với tất cả các phụ thuộc của nó. Containers giúp đơn giản hóa việc di chuyển và triển khai ứng dụng trên nhiều môi trường khác nhau.

Docker Image là một bản sao của một hệ điều hành và môi trường cài đặt, bao gồm cả ứng dụng và các thư viện cần thiết để chạy ứng dụng đó. Nếu container là một "thực thể" đang chạy, Docker Image có thể coi như là một "bản thiết kế" hoặc một "lớp" từ đó có thể tạo ra nhiều containers. Images cung cấp sự khả năng tái sử dụng và chia sẻ, giúp đơn giản hóa quá trình triển khai ứng dụng trên các môi trường khác nhau.

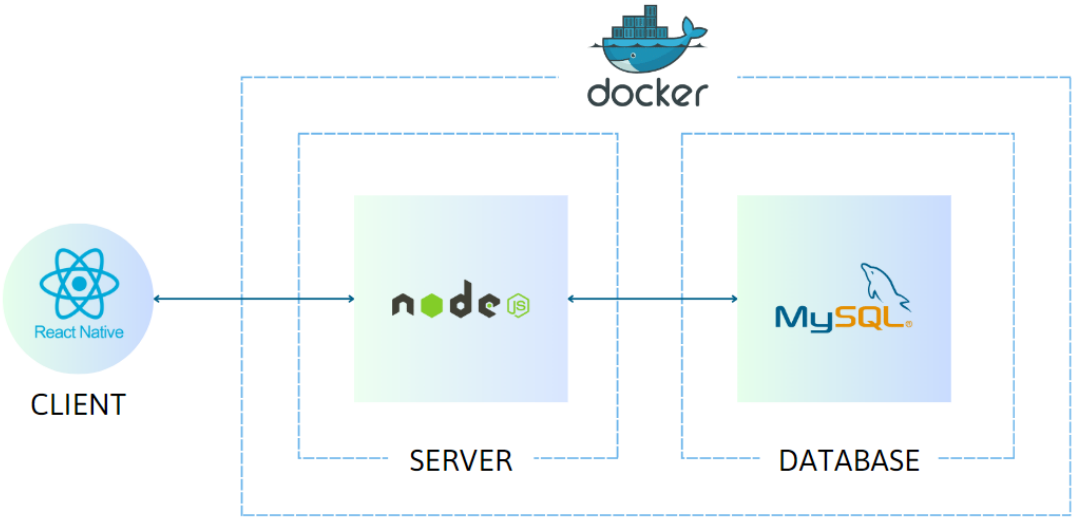
Docker Hub là một nơi trung tâm lưu trữ cho các Docker Image, tương tự như cách GitHub lưu trữ mã nguồn. Tại đây, bạn có thể tìm kiếm, tải về, và chia sẻ các Docker Image. Docker Hub tạo ra một cộng đồng mở để người dùng chia sẻ các image đã tạo sẵn, giúp tăng tốc quá trình phát triển và triển khai ứng dụng.

Docker Engine là một công cụ chứa mọi thứ bạn cần để triển khai và quản lý các container Docker. Nó bao gồm một server daemon, API, và một CLI (Command Line Interface) để tương tác với Docker. Docker Engine giúp quản lý việc tạo, khởi chạy, và quản lý các containers cũng như các images.

2.4 Kiến trúc hệ thống

2.4.1 Kiến trúc Client-Server

Ứng dụng di động nghe nhạc SOUNDHUB được tổ chức thành ba tầng chính, mỗi tầng có vai trò và chức năng cụ thể:



Hình 13. Kiến trúc hệ thống ứng dụng di động nghe nhạc SOUNDHUB

- *Tầng Client:* Đây là phần mềm giao diện người dùng cuối cùng, được triển khai trên thiết bị di động sử dụng ứng dụng React Native. Tầng này tương tác trực tiếp với người dùng, hiển thị giao diện người dùng và thu thập thông tin từ họ. Bằng cách gửi các yêu cầu mạng đến máy chủ, tầng Client thu thập hoặc gửi dữ liệu để cập nhật giao diện người dùng hoặc thực hiện các tác vụ khác.
- *Tầng Server:* Tầng này đóng vai trò là trung tâm của hệ thống, sử dụng một máy chủ Node.js để xử lý các yêu cầu từ ứng dụng React Native và truy vấn cơ sở dữ liệu. Máy chủ này cung cấp các API để cho phép ứng dụng Client truy cập và thao tác với dữ liệu. Nó là nơi chứa các logic kinh doanh và quản lý dữ liệu giữa tầng Client và tầng Database.
- *Tầng Database:* Đây là nơi lưu trữ và quản lý dữ liệu của ứng dụng, thường sử dụng cơ sở dữ liệu MySQL. Tầng này cung cấp các dịch vụ lưu trữ dữ liệu, cũng như cung cấp khả năng truy vấn và cập nhật dữ liệu cho máy chủ Node.js. Thông qua máy chủ, tầng Database cung cấp dữ liệu được yêu cầu từ phía Client và cập nhật dữ liệu mới vào cơ sở dữ liệu khi có yêu cầu từ phía Client. Mô hình MVC

Cấu trúc phân tầng này giúp phân chia rõ ràng các chức năng và trách nhiệm trong hệ thống phần mềm, từ việc tương tác với người dùng, xử lý logic kinh doanh đến quản lý và lưu trữ dữ liệu. Điều này tạo ra một hệ thống linh hoạt, dễ bảo trì và mở rộng.

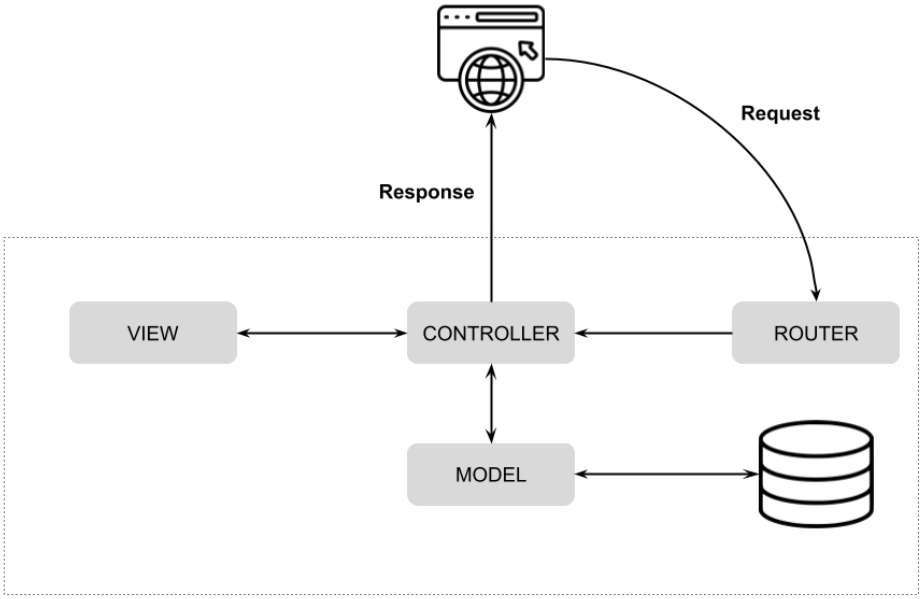
2.4.2 Mô hình MVC

2.4.2.1 Giới thiệu mô hình MVC

Khi phần mềm ứng dụng ngày càng phát triển và đa dạng, việc thiết kế kiến trúc trở nên phức tạp hơn. Các ứng dụng web không chỉ giới hạn ở việc hiển thị thông tin mà trở thành hệ thống thông tin tương tác với người dùng. Để làm cho hệ thống linh hoạt, mềm dẻo, và dễ bảo trì, nhiều giải pháp thiết kế kiến trúc hệ thống đã được đề xuất.

Trong số các giải pháp đó, mô hình kiến trúc MVC (Model - View - Controller) đã được Trygve Reenskaug đề xuất vào những năm 70 của thế kỷ 20 và đã chứng tỏ hiệu quả cao. MVC giúp các nhà phát triển phân chia ứng dụng thành ba phần chính: Model, View và Controller. Mỗi phần có nhiệm vụ riêng biệt và hoạt động độc lập với các phần khác.

2.4.2.2 Các thành phần của MVC



Hình 14. Các thành phần mô hình MVC

Model: Là nơi chứa những nghiệp vụ tương tác với dữ liệu hoặc hệ quản trị cơ sở dữ liệu (mysql, mssql…). Nó sẽ bao gồm các class/function xử lý nhiều nghiệp vụ như kết nối database, truy vấn dữ liệu, thêm – xóa – sửa dữ liệu...

View: Là nơi chứa những giao diện như một nút bấm, khung nhập, menu, hình ảnh… Nó đảm nhiệm nhiệm vụ hiển thị dữ liệu và giúp người dùng tương tác với hệ thống.

Controller: Là nơi tiếp nhận những yêu cầu xử lý được gửi từ người dùng. Nó sẽ gồm những class/ function xử lý nhiều nghiệp vụ logic giúp lấy đúng dữ liệu thông tin cần thiết nhờ các nghiệp vụ lớp Model cung cấp và hiển thị dữ liệu đó ra cho người dùng nhờ lớp View.

CHƯƠNG 3 XÁC ĐỊNH NHU CẦU

3.1 Yêu cầu chức năng

- *Đăng nhập và đăng ký người dùng*

Người dùng có thể đăng nhập vào tài khoản hiện có hoặc đăng ký tài khoản mới thông qua email hoặc mạng xã hội.

- *Tìm kiếm*

Người dùng có thể tìm kiếm bài hát, nghệ sĩ theo từ khóa.

- *Quản lý Playlist*

Người dùng có thể tạo, chỉnh sửa và xóa các playlist cá nhân.

- *Theo dõi Nghệ sĩ*

Người dùng có thể theo dõi nghệ sĩ yêu thích để cập nhật thông tin về các bài hát mới của họ.

- *Nghe nhạc trực tuyến*

Người dùng có thể nghe nhạc trực tuyến từ thư viện âm nhạc đa dạng của ứng dụng.

3.2 Yêu cầu phi chức năng

- *Giao diện thân thiện*

Giao diện người dùng được thiết kế thân thiện và dễ sử dụng để tạo ra trải nghiệm nghe nhạc mượt mà.

- *Hiệu suất*

Ứng dụng phải hoạt động một cách mượt mà và nhanh chóng, đảm bảo thời gian tải trang và phản hồi của người dùng.

CHƯƠNG 4 LẬP KẾ HOẠCH SCRUM

4.1 Lập kế hoạch sản phẩm (Product Backlog)

Gồm có 10 epic với các task tương ứng:

4.1.1.1 Epic: Thiết kế cơ sở dữ liệu

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-15	Thiết kế cơ sở dữ liệu			May 4, 2024	=	Apr 28, 2024	Jun 5, 2024
	CT-6	Phân tích các yêu cầu chức năng và phi chức năng liên quan đến cơ sở dữ liệu	2	Phuoc Lam Hua	Apr 29, 2024	=	Apr 28, 2024	May 6, 2024
	CT-7	Xác định các đối tượng dữ liệu cần lưu trữ	1	Phuoc Lam Hua	Apr 30, 2024	=	Apr 28, 2024	Apr 30, 2024
	CT-8	Xác định các mối quan hệ giữa các đối tượng dữ liệu	1	Phuoc Lam Hua	Apr 30, 2024	=	Apr 28, 2024	Apr 30, 2024
	CT-11	Thiết kế các bảng trong cơ sở dữ liệu	3	tringuyen.21092003	May 1, 2024	=	Apr 28, 2024	May 1, 2024
	CT-13	Thiết lập các khóa chính, khóa ngoại và ràng buộc dữ liệu cho các bảng.	1	Mãi Tấn	May 2, 2024	=	Apr 28, 2024	May 3, 2024
	CT-9	Lập mô hình dữ liệu ERD (Entity-Relationship Diagram) trên dbdiagram	2	Mãi Tấn	May 3, 2024	=	Apr 28, 2024	May 3, 2024
	CT-14	Hoàn thiện mô hình dữ liệu trong dbdiagram	1	tringuyen.21092003	May 4, 2024	=	Apr 28, 2024	May 6, 2024

Hình 15. Các task của Epic Thiết kế cơ sở dữ liệu

4.1.1.2 Epic: Triển khai cơ sở dữ liệu

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-20	Triển khai cơ sở dữ liệu			May 6, 2024	=	Apr 28, 2024	Jun 5, 2024
	CT-16	Tạo cơ sở dữ liệu trong MySQL	2	Mãi Tấn	May 5, 2024	=	Apr 28, 2024	May 6, 2024
	CT-18	Thêm dữ liệu mẫu vào các bảng	1	tringuyen.21092003	May 6, 2024	=	Apr 28, 2024	May 6, 2024

Hình 16. Các task của Epic Triển khai cơ sở dữ liệu

4.1.1.3 Epic: Đăng ký tài khoản

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-22	Đăng Ký Tài Khoản			May 10, 2024	=	May 6, 2024	Jun 5, 2024
	CT-27	Xây dựng API đăng ký	3	Mãi Tấn	May 8, 2024	=	May 6, 2024	May 7, 2024
	CT-24	Thiết kế giao diện đăng ký	1	Phuoc Lam Hua	May 7, 2024	=	May 6, 2024	May 7, 2024
	CT-25	Thiết kế giao diện xác thực email	1	Phuoc Lam Hua	May 8, 2024	=	May 6, 2024	May 7, 2024
	CT-29	Xây dựng API xác thực email	3	Mãi Tấn	May 8, 2024	=	May 6, 2024	May 8, 2024
	CT-30	Xây dựng giao diện đăng ký	3	tringuyen.21092003	May 10, 2024	=	May 6, 2024	May 8, 2024
	CT-31	Xây dựng giao diện xác thực email	3	tringuyen.21092003	May 10, 2024	=	May 6, 2024	May 9, 2024

Hình 17. Các task của Epic Đăng ký tài khoản

4.1.1.4 Epic: Đăng nhập tài khoản

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-23	Đăng Nhập Tài Khoản			May 14, 2024	=	May 6, 2024	Jun 5, 2024
	CT-37	Xây dựng API cho chức năng xác thực tài khoản	3	Mãi Tấn	May 12, 2024	=	May 6, 2024	May 12, 2024
	CT-36	Xây dựng API cho chức năng thay đổi mật khẩu	3	Mãi Tấn	May 12, 2024	=	May 6, 2024	May 11, 2024
	CT-33	Thiết kế giao diện thay đổi mật khẩu	1	Phuoc Lam Hua	May 10, 2024	=	May 6, 2024	May 10, 2024
	CT-32	Thiết kế giao diện đăng nhập	1	Phuoc Lam Hua	May 9, 2024	=	May 6, 2024	May 9, 2024
	CT-35	Xây dựng API cho chức năng đăng nhập	3	Mãi Tấn	May 11, 2024	=	May 6, 2024	May 10, 2024
	CT-38	Xây dựng giao diện đăng nhập	3	tringuyen.21092003	May 13, 2024	=	May 6, 2024	May 11, 2024
	CT-34	Thiết kế giao diện xác thực tài khoản	1	Phuoc Lam Hua	May 11, 2024	=	May 6, 2024	May 12, 2024
	CT-39	Xây dựng giao diện thay đổi mật khẩu	3	tringuyen.21092003	May 14, 2024	=	May 6, 2024	May 13, 2024
	CT-40	Xây dựng giao diện xác thực tài khoản	3	tringuyen.21092003	May 14, 2024	=	May 6, 2024	May 13, 2024

Hình 18. Các task của Epic Đăng nhập tài khoản

4.1.1.5 Epic: Chức năng bài hát

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-41	Chức Năng Bài Hát			May 21, 2024	=	May 14, 2024	Jun 5, 2024
	CT-44	Xây dựng API thêm, xóa, sửa bài hát	3	Mãi Tấn	May 16, 2024	=	May 14, 2024	May 16, 2024
	CT-43	Thiết kế giao diện trang chi tiết bài hát	2	Phuoc Lam Hua	May 16, 2024	=	May 14, 2024	May 15, 2024
	CT-45	Xây dựng API yêu thích bài hát	2	Mãi Tấn	May 17, 2024	=	May 14, 2024	May 17, 2024
	CT-46	Xây dựng giao diện chi tiết bài hát	4	tringuyen.21092003	May 21, 2024	=	May 14, 2024	May 19, 2024

Hình 19. Các task của Epic Chức năng bài hát

4.1.1.6 Epic: Chức năng tìm kiếm bài hát

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-42	Chức Năng Tìm Kiếm Bài Hát			May 22, 2024	=	May 14, 2024	Jun 5, 2024
	CT-50	Xây dựng API lọc bài hát theo thời gian và kí tự đầu	3	Mãi Tấn	May 20, 2024	=	May 14, 2024	May 20, 2024
	CT-48	Thiết kế giao diện tìm kiếm bài hát	2	Phuoc Lam Hua	May 18, 2024	=	May 14, 2024	May 18, 2024
	CT-47	Xây dựng API tìm kiếm bài hát	3	Mãi Tấn	May 19, 2024	=	May 14, 2024	May 17, 2024
	CT-49	Xây dựng giao diện tìm kiếm bài hát	4	tringuyen.21092003	May 22, 2024	=	May 14, 2024	May 21, 2024

Hình 20. Các task của Epic Chức năng tìm kiếm bài hát

4.1.1.7 Epic: Chức năng danh sách phát

Type	Key	Summary	Story point ...	Assignee	Due date	Priority	Created	Updat...
	CT-51	Chức Năng Danh Sách Phát			May 30, 2024	=	May 22, 2024	May 22, 2024
	CT-56	Xây dựng API thêm, xóa, sửa danh sách phát	4	Mãi Tấn	May 24, 2024	=	May 22, 2024	May 24, 2024
	CT-53	Thiết kế giao diện trang chi tiết danh sách phát	2	Phuoc Lam Hua	May 24, 2024	=	May 22, 2024	May 23, 2024
	CT-58	Xây dựng API yêu thích danh sách phát	3	Mãi Tấn	May 24, 2024	=	May 22, 2024	May 23, 2024
	CT-54	Thiết kế giao diện thêm danh sách phát	2	Phuoc Lam Hua	May 25, 2024	=	May 22, 2024	May 27, 2024
	CT-59	Xây dựng giao diện chi tiết danh sách phát	4	tringuyen.21092003	May 26, 2024	=	May 22, 2024	May 25, 2024
	CT-55	Thiết kế giao diện sửa danh sách phát	2	Phuoc Lam Hua	May 26, 2024	=	May 22, 2024	May 26, 2024
	CT-65	Thiết kế giao diện thêm bài hát vào danh sách phát	2	Phuoc Lam Hua	May 27, 2024	=	May 22, 2024	May 26, 2024
	CT-57	Xây dựng API thêm bài hát vào danh sách phát	4	Mãi Tấn	May 28, 2024	=	May 22, 2024	May 27, 2024
	CT-60	Xây dựng giao diện thêm danh sách phát	4	tringuyen.21092003	May 28, 2024	=	May 22, 2024	May 28, 2024
	CT-61	Xây dựng giao diện sửa danh sách phát	4	tringuyen.21092003	May 29, 2024	=	May 22, 2024	May 29, 2024
	CT-66	Xây dựng giao diện thêm bài hát vào danh sách phát	4	tringuyen.21092003	May 30, 2024	=	May 22, 2024	May 29, 2024

Hình 21. Các task của Epic Chức năng danh sách phát

4.1.1.8 Epic: Chức năng phát nhạc

Type	Key	Summary	Story point ...	Assignee	Due date	P...	Created	Updated
	CT-67	Chức Năng Phát Nhạc			Jun 7, 2024	=	Jun 4, 2024	Jun 5, 2024
	CT-70	Xây dựng API phát nhạc	2	Mãi Tấn	Jun 6, 2024	=	Jun 4, 2024	Jun 6, 2024
	CT-69	Thiết kế giao diện phát nhạc	1	Phuoc Lam Hua	Jun 6, 2024	=	Jun 4, 2024	Jun 6, 2024
	CT-71	Xây dựng giao diện phát nhạc	2	tringuyen.21092003	Jun 7, 2024	=	Jun 4, 2024	Jun 7, 2024

Hình 22. Các task của Epic Chức năng phát nhạc

4.1.1.9 Epic: Chức năng thông tin người dùng

Type	Key	Summary	Story point ...	Assignee	Due date	P...	Created	Updated
	CT-75	Chức Năng Cập Nhật Thông Tin Người Dùng			Jun 8, 2024	=	Jun 5, 2024	Jun 5, 2024
	CT-74	Xây dựng API cập nhật thông tin người dùng	2	Mãi Tấn	Jun 7, 2024	=	Jun 5, 2024	Jun 7, 2024
	CT-76	Thiết Kế giao diện cập nhật thông tin người dùng	1	Phuoc Lam Hua	Jun 7, 2024	=	Jun 5, 2024	Jun 7, 2024
	CT-77	Xây dựng giao diện cập nhật thông tin người dùng	2	tringuyen.21092003	Jun 8, 2024	=	Jun 5, 2024	Jun 8, 2024

Hình 23. Các task của Epic Chức năng thông tin người dùng

4.1.1.10 Epic: Triển khai

Type	Key	Summary	Story point ...	Assignee	Due date	P...	Created	Updated
	CT-68	Triển Khai			Jun 10, 2024	=	Jun 4, 2024	Jun 5, 2024
	CT-73	Triển khai Backend lên VPS	3	Mãi Tấn	Jun 10, 2024	=	Jun 4, 2024	Jun 10, 2024
	CT-72	Triển khai Backend bằng Docker	3	Mãi Tấn	Jun 10, 2024	=	Jun 4, 2024	Jun 8, 2024

Hình 24. Các task của Epic Triển khai

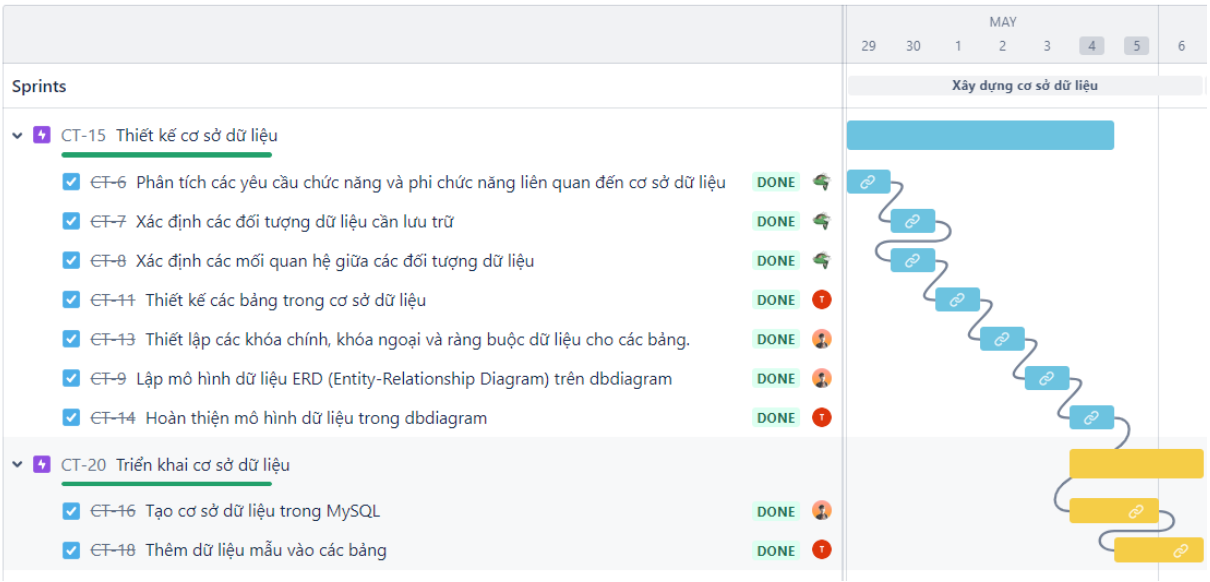
4.2 Lập kế hoạch sprint (Sprint Backlog)

4.2.1.1 Sprint 1: Xây dựng cơ sở dữ liệu

Ngày bắt đầu: 2024/04/29

Ngày kết thúc: 2024/05/06

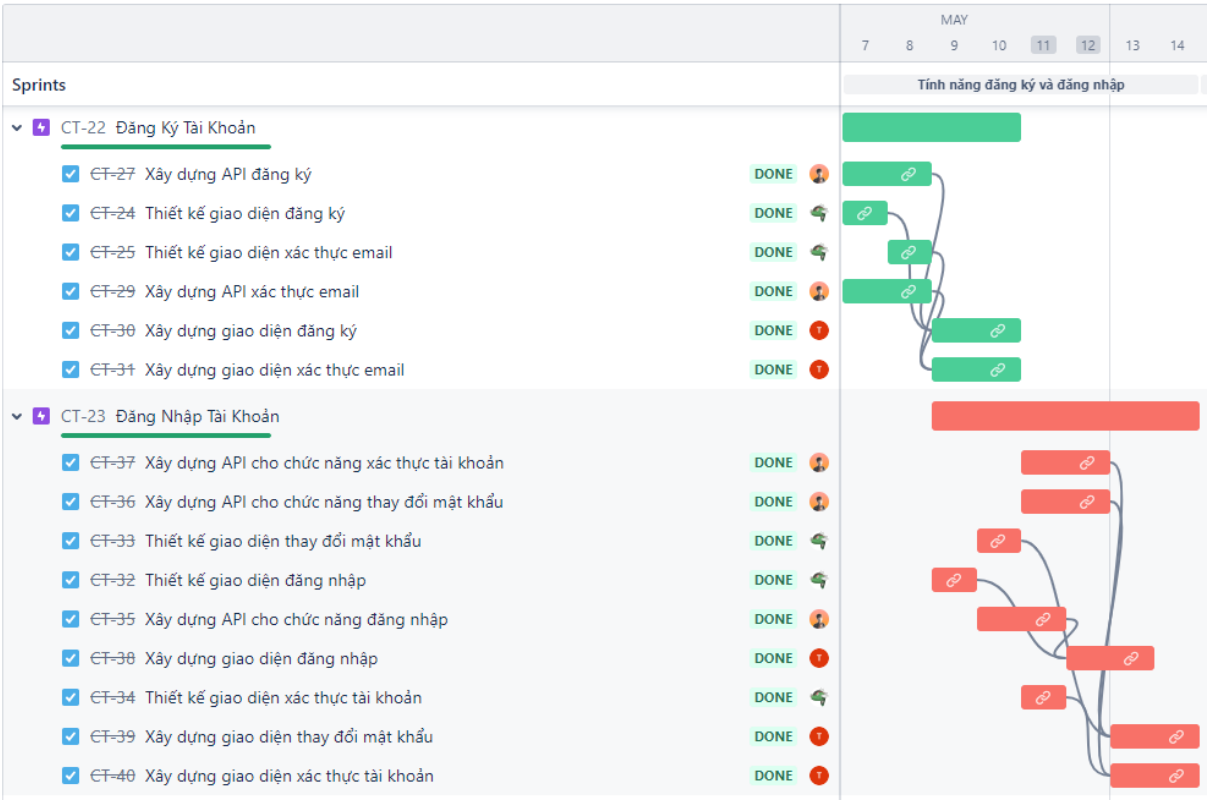
Mục tiêu: Tạo cơ sở dữ liệu cơ bản cho dự án để đảm bảo sự chuẩn bị cho việc phát triển các tính năng chính trong tương lai.



Hình 25. Các task của Spring 1

4.2.1.2 Sprint 2: Tính năng đăng ký và đăng nhập

Ngày bắt đầu: 2024/05/07
Ngày kết thúc: 2024/05/14
Mục tiêu: Hoàn thành tính năng đăng ký và đăng nhập tài khoản cho người dùng.



Hình 26. Các task của Spring 2

4.2.1.3 Sprint 3: Quản lý bài hát

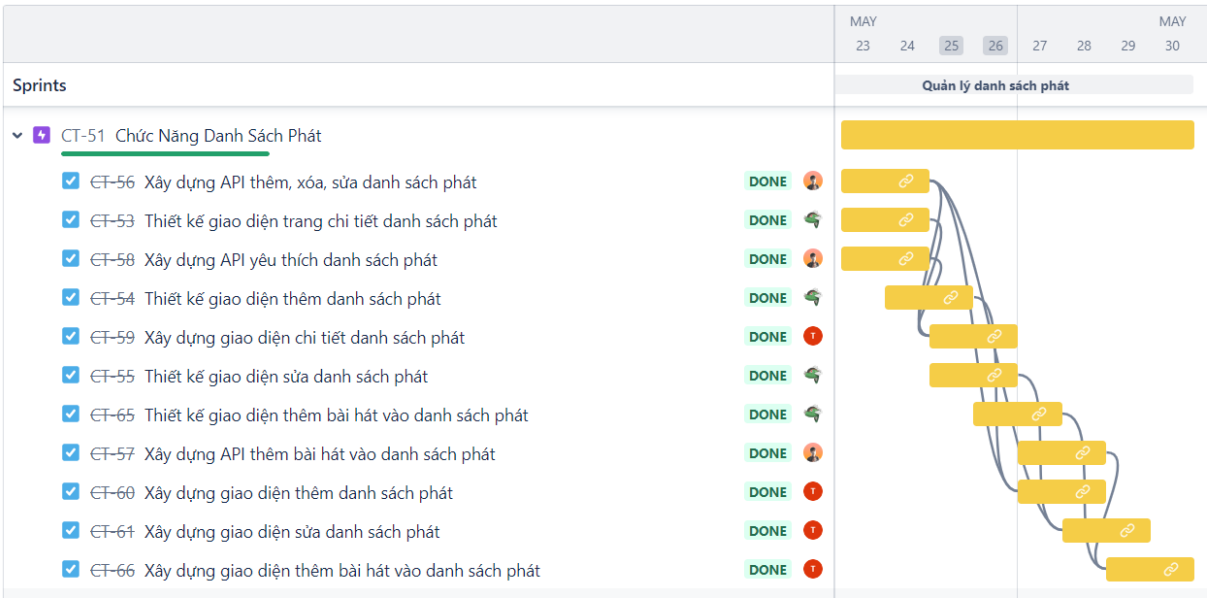
Ngày bắt đầu: 2024/05/15
Ngày kết thúc: 2024/05/22
Mục tiêu: Hoàn thiện các chức năng cơ bản liên quan đến việc quản lý và tìm kiếm bài hát trong ứng dụng.

Hình 27. Các task của Spring 3

4.2.1.4 Sprint 4: Quản lý danh sách phát

Ngày bắt đầu: 2024/05/22
Ngày kết thúc: 2024/05/30

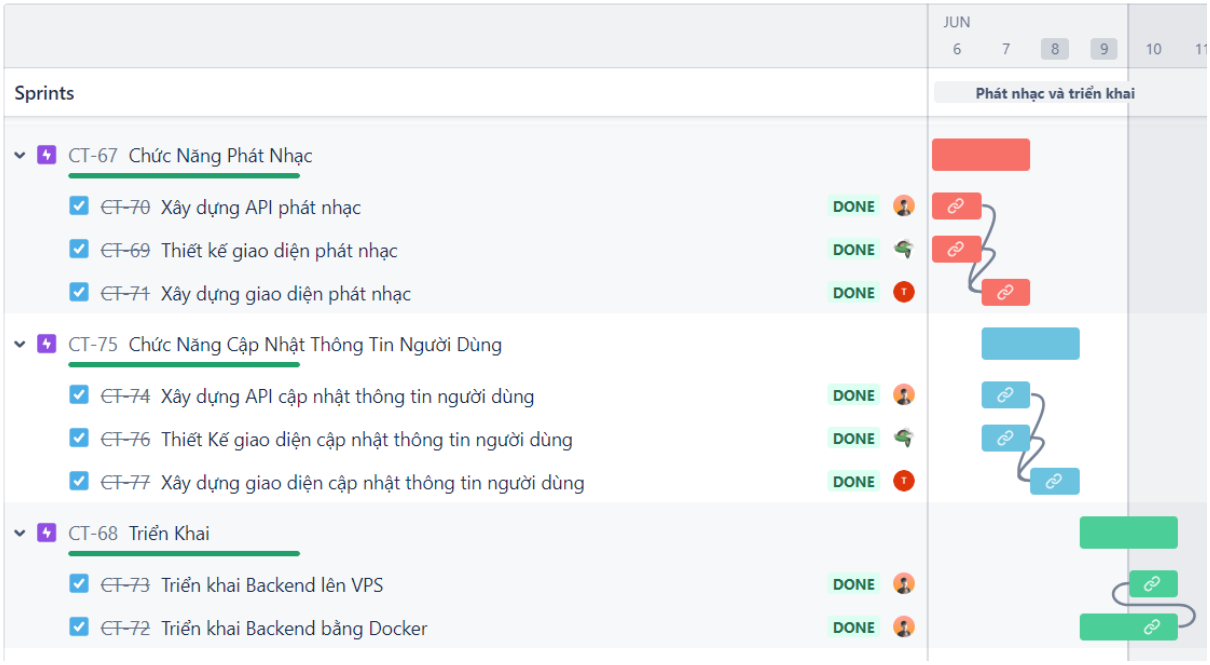
Mục tiêu: Hoàn thiện các chức năng cơ bản liên quan đến việc quản lý danh sách bài hát.



Hình 28. Các task của Spring 4

4.2.1.5 Sprint 5: Phát nhạc và triển khai

Ngày bắt đầu: 2024/06/06
Ngày kết thúc: 2024/06/10
Mục tiêu: Hoàn thiện các chức năng liên quan đến phát nhạc và cập nhật thông tin người dùng. Triển khai Backend bằng Docker lên VPS.



Hình 29. Các task của Spring 5

CHƯƠNG 5 HIỆN THỰC HÓA KẾ HOẠCH

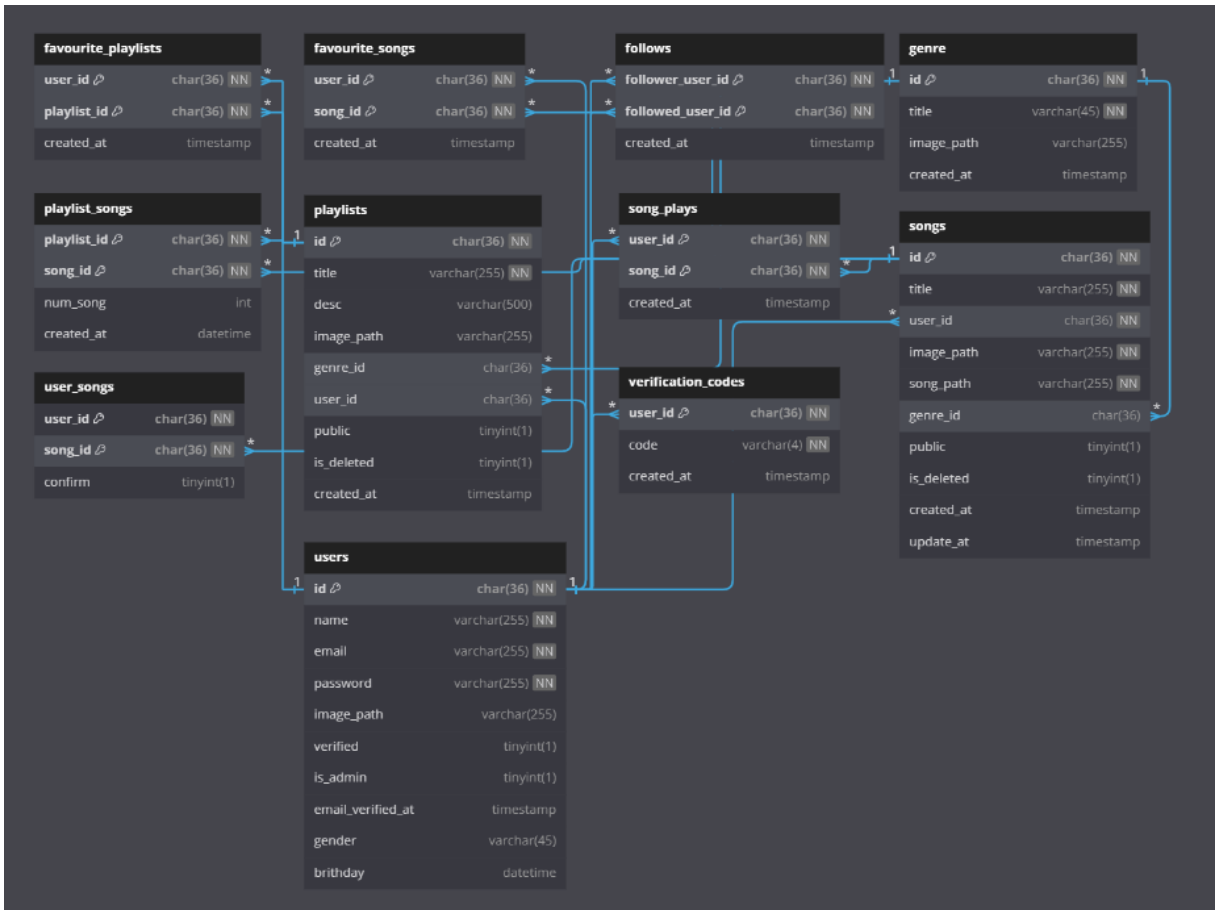
5.1 Kết quả đạt được

5.1.1 Sprint 1: Xây dựng cơ sở dữ liệu

5.1.1.1 Kết quả của Sprint 1

a) Cơ sở dữ liệu

Thiết kế được cơ sở dữ liệu và triển khai cơ sở dữ liệu trên MySQL gồm các bảng như:

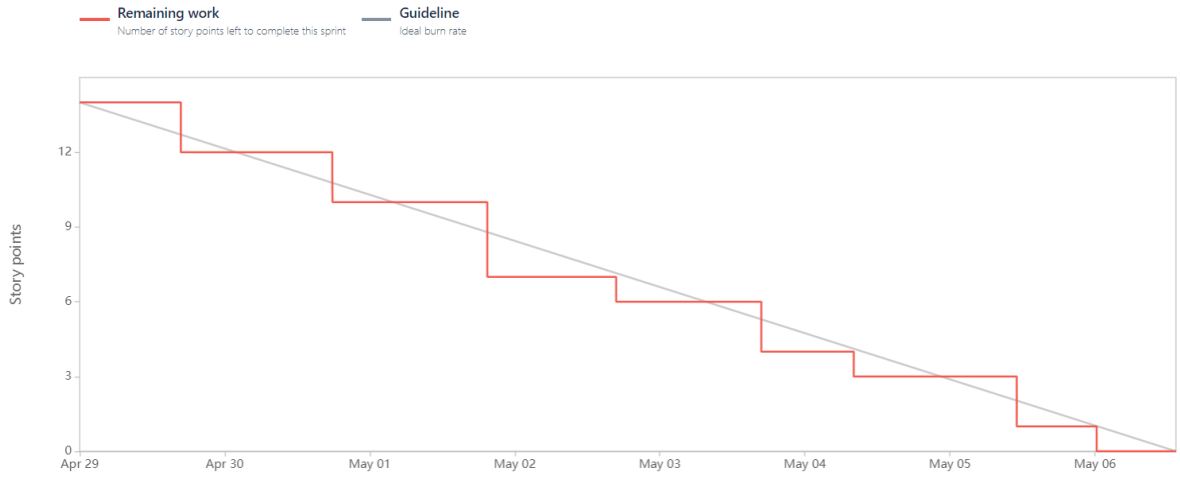


Hình 30. Database diagram

b) Mã nguồn

Mã nguồn được lưu trữ tại https://github.com/tanmaiii/CNPM-TML.git

5.1.1.2 Burndown chart của Sprint 1

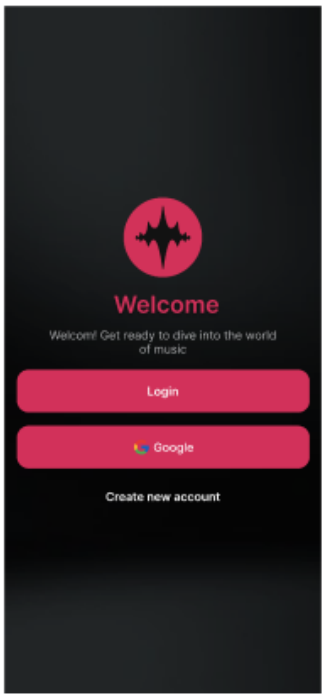


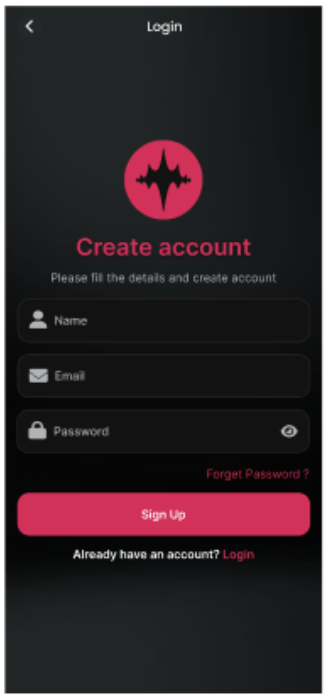
Hình 31. Burndown chart của Sprint 1

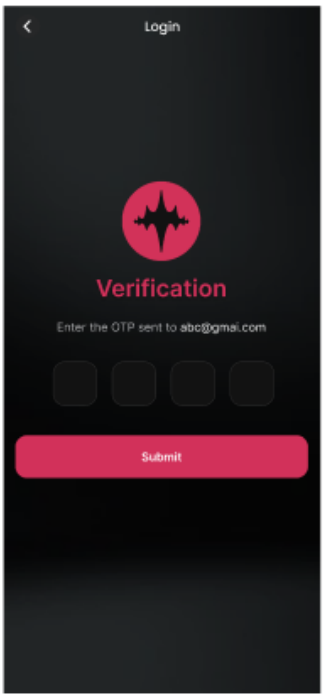
5.1.2 Sprint 2: Tính năng đăng ký và đăng nhập

5.1.2.1 Kết quả của Sprint 2

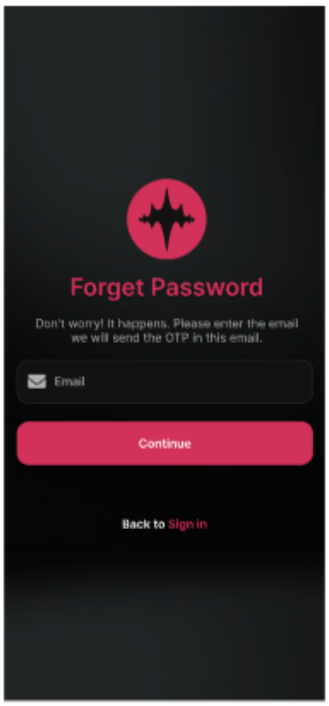
a) Giao diện (figma)

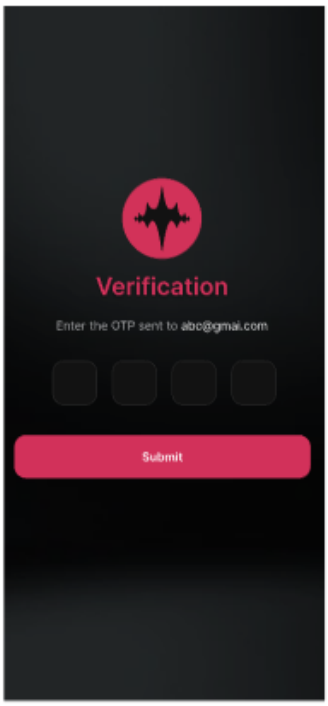


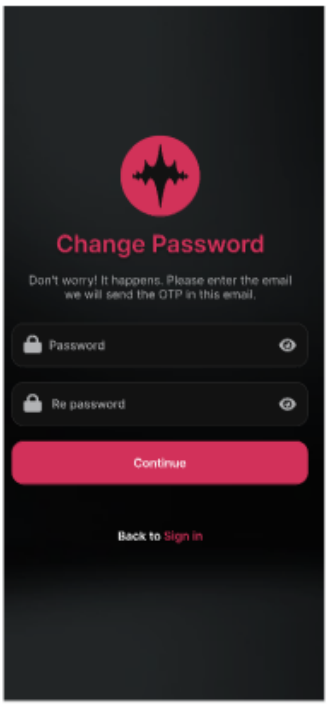


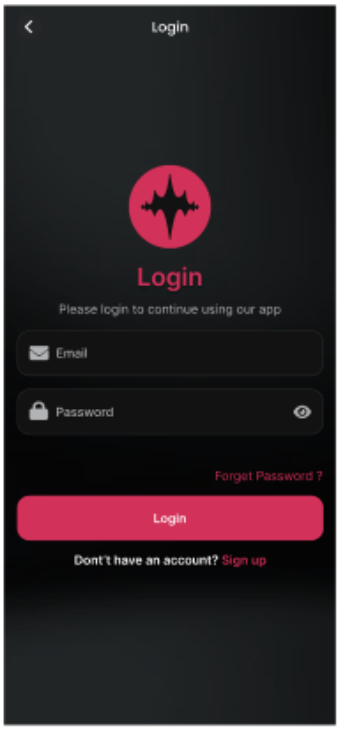


Forgot password Verify code change password







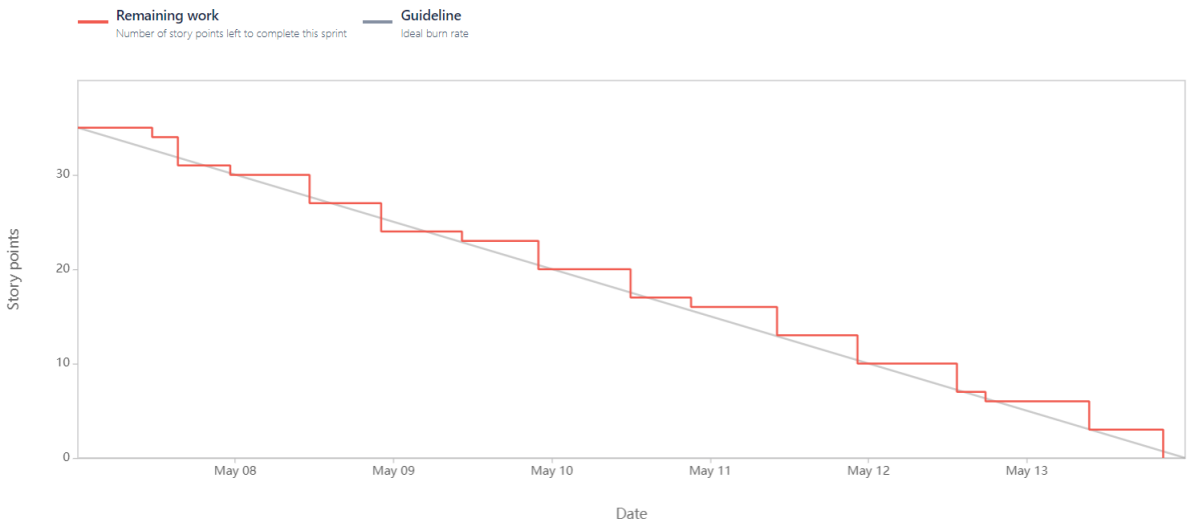


Hình 32. Các giao diện figma của Sprint 1

b) Mã nguồn

Mã nguồn được lưu trữ tại https://github.com/tanmaiii/CNPM-TML.git

5.1.2.2 Burndown chart của Sprint 2

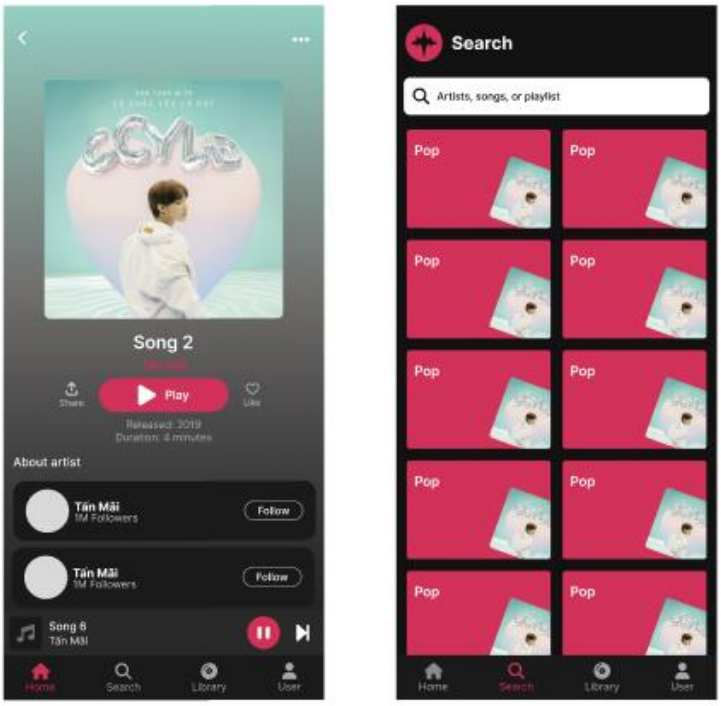


Hình 33. Burndown chart của Sprint 2

5.1.3 Sprint 3: Quản lý bài hát

5.1.3.1 Kết quả của Sprint 3

a) Giao diện (figma)



Hình 34. Các giao diện figma của Sprint 2

b) Mã nguồn

Mã nguồn được lưu trữ tại https://github.com/tanmaiii/CNPM-TML.git

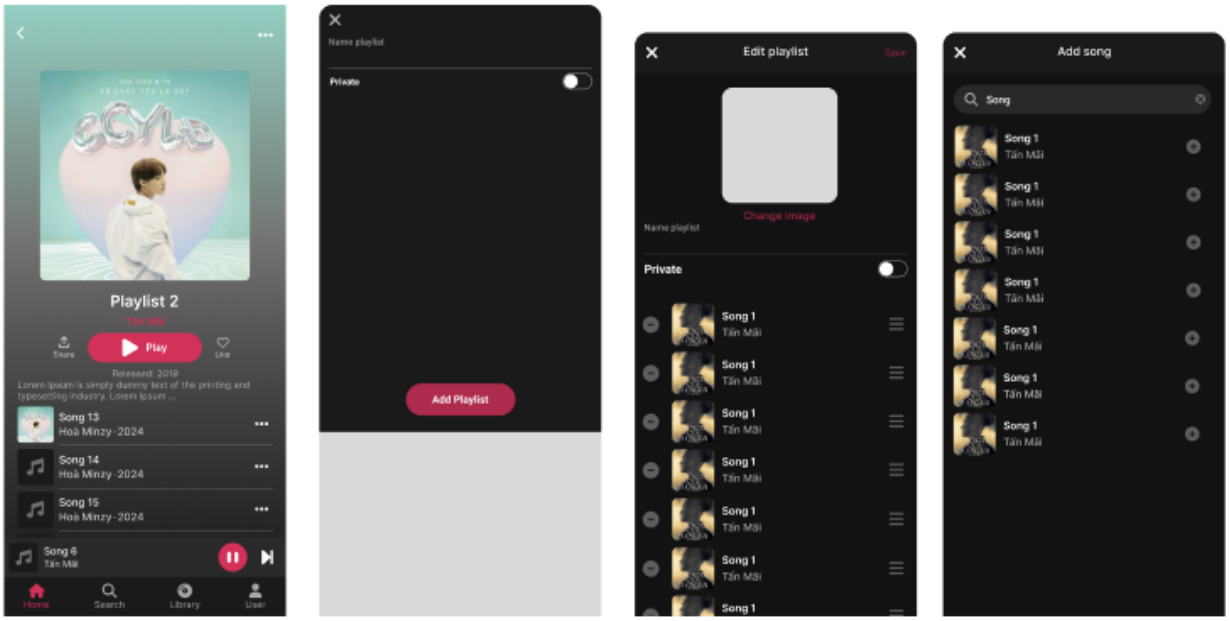
5.1.3.2 Burndown chart của Sprint 3

Hình 35. Burndown chart của Sprint 3

5.1.4 Sprint 4: Quản lý danh sách phát

5.1.4.1 Kết quả của Sprint 4

a) Giao diện (figma)

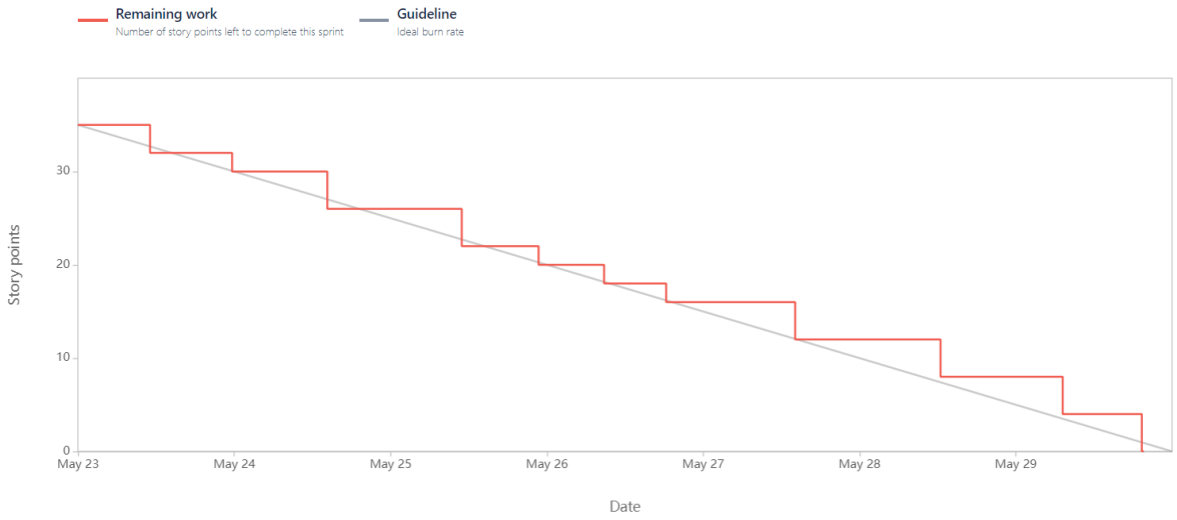


Hình 36. Các giao diện figma của Sprint 3

b) Mã nguồn

Mã nguồn được lưu trữ tại https://github.com/tanmaiii/CNPM-TML.git

5.1.4.2 Burndown chart của Sprint 4

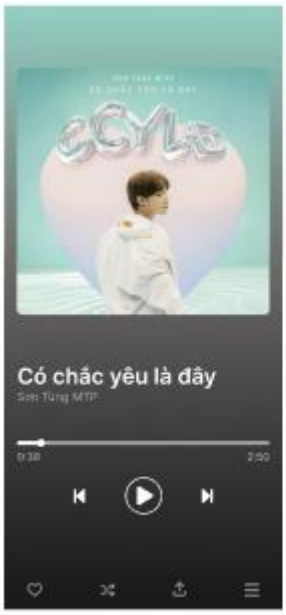


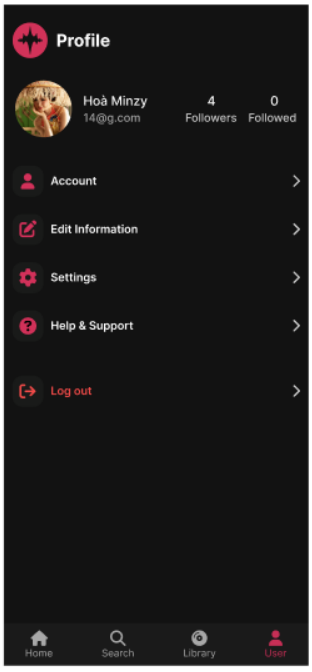
Hình 37. Burndown chart của Sprint 4

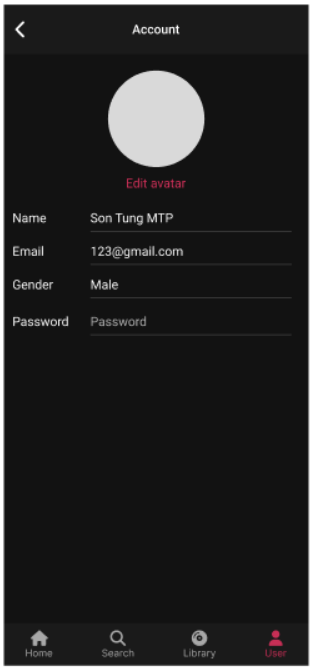
5.1.5 Sprint 5: Phát nhạc và triển khai

5.1.5.1 Kết quả của Sprint 5

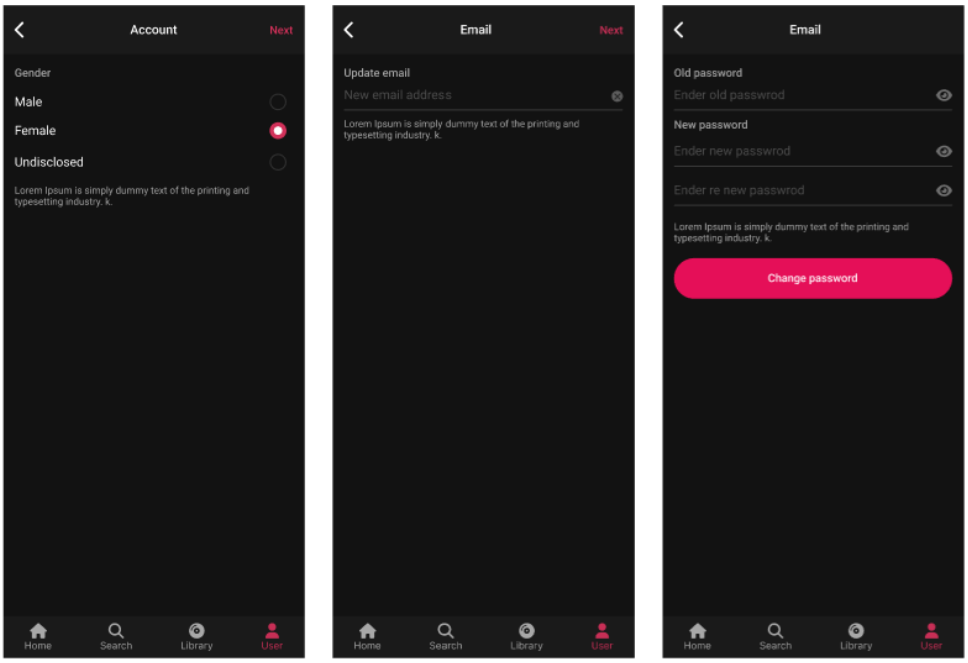
a) Giao diện (figma)


Có chắc yêu là đây


Profile
Hoà Minzy
14@g.com
4
Followers
0
Followed
Account
Edit Information
Settings
Help & Support
Log out
Home
Search
Library
User


Account
Edit avatar
Name
Son Tung MTP
Email
123@gmail.com
Gender
Male
Password
Password
Home
Search
Library
User


Name
Next
Name
Enter your name
Lorem Ipsum is simply dummy text of the printing and typesetting industry k.
Home
Search
Library
User

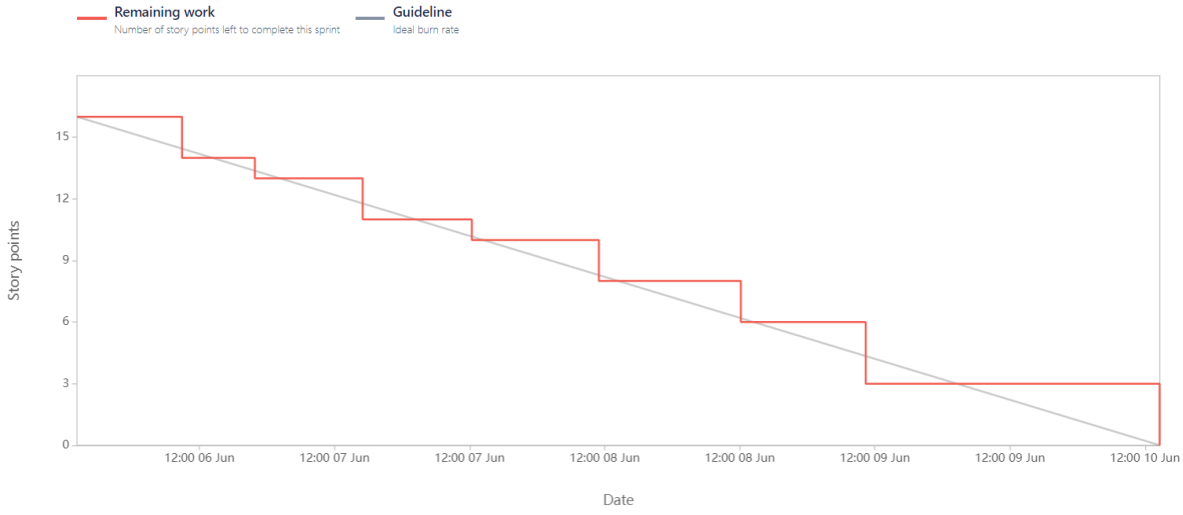


Hình 38. Các giao diện figma của Sprint 5

b) Mã nguồn

Mã nguồn được lưu trữ tại https://github.com/tanmaiii/CNPM-TML.git

5.1.5.2 Burndown chart của Sprint 5



Hình 39. Burndown chart của Sprint 5

CHƯƠNG 6 KẾT LUẬN

6.1 Đánh giá tổng quan về dự án

Dự án phát triển ứng dụng âm nhạc di động sử dụng React Native, Node.js và MySQL đã đạt được nhiều kết quả khả quan. Từ giai đoạn ý tưởng đến triển khai thực tế, chúng tôi đã tạo ra một nền tảng âm nhạc cung cấp trải nghiệm nghe nhạc ổn định, hỗ trợ nhiều tính năng phong phú như khám phá âm nhạc, tạo và quản lý playlist, yêu thích ca sĩ và khả năng nghe nhạc trực tuyến. Quy trình phát triển tuân thủ mô hình Agile đã giúp chúng tôi linh hoạt trong việc điều chỉnh và cải tiến sản phẩm liên tục, đảm bảo đáp ứng tốt nhất nhu cầu của người dùng.

6.2 Kết quả đạt được được

- Giao diện người dùng được thiết kế thân thiện, dễ sử dụng và có tính thẩm mỹ cao, giúp người dùng dễ dàng tiếp cận và sử dụng các tính năng của ứng dụng.

- Sử dụng Node.js và MySQL giúp server hoạt động hiệu quả, khả năng xử lý đồng thời cao và đảm bảo dữ liệu được quản lý tốt.

- Ứng dụng cung cấp nhiều tính năng như tìm kiếm, yêu thích nghệ sĩ, chia sẻ bài hát và playlist, giúp người dùng có trải nghiệm nghe nhạc đa dạng và cá nhân hóa cao.

- Quy trình phát triển linh hoạt: Mô hình Agile giúp đội ngũ phát triển có thể nhanh chóng thích ứng với các thay đổi và cải tiến sản phẩm liên tục.

6.3 Hạn chế

- Tài nguyên hạn chế: Ứng dụng ban đầu có thể gặp khó khăn trong việc xử lý lượng lớn người dùng đồng thời do hạn chế về tài nguyên server.

- Khả năng mở rộng: Mặc dù MySQL là một hệ quản trị cơ sở dữ liệu mạnh mẽ, việc mở rộng hệ thống để xử lý lượng dữ liệu rất lớn và phức tạp có thể đòi hỏi nhiều công sức và tài nguyên.

6.4 Hướng phát triển trong tương lai

- Nâng cấp cơ sở hạ tầng server: Sử dụng các giải pháp cloud để mở rộng tài nguyên một cách linh hoạt và tự động.

- Tối ưu hóa cơ sở dữ liệu: Sử dụng các kỹ thuật tối ưu hóa cơ sở dữ liệu, như sharding và indexing, để cải thiện tốc độ truy xuất và xử lý dữ liệu.

- Cải tiến hệ thống phát nhạc: Sử dụng các công nghệ phát trực tuyến tiên tiến để giảm độ trễ và cải thiện trải nghiệm nghe nhạc trực tuyến.

- Tăng cường bảo mật

- Tích hợp với các nền tảng mạng xã hội: Giúp người dùng dễ dàng chia sẻ nội dung âm nhạc với bạn bè trên các mạng xã hội phổ biến.

TÀI LIỆU THAM KHẢO

[1] W3Schools, "W3Schools", 2024. [Online]. Available: https://www.w3schools.com/. [Accessed: 12- Jun- 2024].
[2] React Native, "React Native", 2024. [Online]. Available: https://reactnative.dev/. [Accessed: 12- Jun- 2024].
[3] Scrum.org, "Scrum.org", 2024. [Online]. Available: https://www.scrum.org/. [Accessed: 12- Jun- 2024].