

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

PHẠM ÁNH DƯƠNG

KHÓA LUẬN TỐT NGHIỆP
NGHIÊN CỨU MÔ HÌNH AGILE VÀ
PHÁT TRIỂN CÔNG CỤ HỖ TRỢ

KỸ SƯ NGÀNH CÔNG NGHỆ PHẦN MỀM

TP. HỒ CHÍ MINH, 2014

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

PHẠM ÁNH DƯƠNG – 09520047

KHÓA LUẬN TỐT NGHIỆP
NGHIÊN CỨU MÔ HÌNH AGILE VÀ
PHÁT TRIỂN CÔNG CỤ HỖ TRỢ

KỸ SƯ NGÀNH CÔNG NGHỆ PHẦN MỀM

GIẢNG VIÊN HƯỚNG DẪN
ThS. NGUYỄN THỊ THANH TRÚC

TP. HỒ CHÍ MINH, 2014

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số
ngày của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. – Chủ tịch.
2. – Thư ký.
3. – Ủy viên.
4. – Ủy viên.

NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP

(CỦA CÁN BỘ HƯỚNG DẪN)

Tên khóa luận:

NGHIÊN CỨU MÔ HÌNH AGILE VÀ PHÁT TRIỂN CÔNG CỤ HỖ TRỢ

Nhóm SV thực hiện:

Cán bộ hướng dẫn:

Phạm Ánh Dương

09520047

ThS. Nguyễn Thị Thanh Trúc

Đánh giá Khóa luận

1. Về cuốn báo cáo:

Số trang _____ Số chương _____

Số bảng số liệu _____ Số hình vẽ _____

Số tài liệu tham khảo _____ Sản phẩm _____

Một số nhận xét về hình thức cuốn báo cáo:

.....

.....

.....

.....

2. Về nội dung nghiên cứu:

.....

.....

.....

.....

3. Về chương trình ứng dụng:

.....

.....

.....

.....

4. Về thái độ làm việc của sinh viên:

.....

.....

.....

.....

Đánh giá chung:

.....

.....

Điểm từng sinh viên:

Phạm Ánh Dương:...../10

Người nhận xét

(Ký tên và ghi rõ họ tên)

NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP
(CỦA CÁN BỘ PHẢN BIỆN)

Tên khóa luận:

NGHIÊN CỨU MÔ HÌNH AGILE VÀ PHÁT TRIỂN CÔNG CỤ HỖ TRỢ

Nhóm SV thực hiện:

Phạm Ánh Dương

Cán bộ phản biện:

09520047

Đánh giá Khóa luận

1. Về cuốn báo cáo:

Số trang	_____	Số chương	_____
Số bảng số liệu	_____	Số hình vẽ	_____
Số tài liệu tham khảo	_____	Sản phẩm	_____

Một số nhận xét về hình thức cuốn báo cáo:

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

2. Về nội dung nghiên cứu:

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

3. Về chương trình ứng dụng:

.....

.....

.....

.....

4. Về thái độ làm việc của sinh viên:

.....

.....

.....

.....

Đánh giá chung:

.....

.....

Điểm từng sinh viên:

Phạm Ánh Dương:...../10

Người nhận xét

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc tới ThS. Nguyễn Thị Thanh Trúc, người đã trực tiếp tận tình hướng dẫn chỉ bảo chúng em trong quá trình nghiên cứu và hoàn thành luận văn.

Nhóm xin chân thành cảm ơn sự động viên, giúp đỡ, tạo điều kiện của Khoa Công Nghệ Phần Mềm – Đại học Công Nghệ Thông Tin – Đại Học Quốc Gia TP HCM. Chúng em xin cảm ơn các thầy cô giáo trong khoa đã tận tình giảng dạy giúp chúng em có những kiến thức để hoàn thành đề tài này.

Cuối cùng, nhóm chúng em xin cảm ơn gia đình, bạn bè và người thân đã luôn động viên, giúp cho chúng em có được niềm tin, tri thức, kinh nghiệm quý báu trong học tập và cuộc sống.

TP Hồ Chí Minh, tháng 7 năm 2014.

ĐỀ CƯƠNG CHI TIẾT

TÊN ĐỀ TÀI: NGHIÊN CỨU MÔ HÌNH AGILE VÀ PHÁT TRIỂN CÔNG CỤ HỖ TRỢ
Cán bộ hướng dẫn: Nguyễn Thị Thanh Trúc
Thời gian thực hiện: Từ ngày 15/3/2014 đến ngày 25/7/2014
Sinh viên thực hiện: Phạm Ánh Dương – 09520047
Nội dung đề tài: <div><div>1. Mục tiêu</div><div><p>Tìm hiểu và nắm bắt được triết lí của phương pháp phát triển phần mềm Agile. Hiểu rõ giá trị cốt lõi, đặc trưng cùng nguyên nhân tại sao Agile lại đang được ứng dụng rộng rãi.</p><p>Nghiên cứu về một phương pháp phát triển phần mềm tuân theo triết lí Agile là Scrum. Nắm bắt được quy trình của Scrum, cách thực hiện một dự án theo quy trình Scrum.</p><p>Xây dựng công cụ hỗ trợ quản lí dự án theo phương pháp Scrum. Nhóm đã xác định chọn phát triển trên môi trường web nhằm giúp người dùng có thể sử dụng một cách thuận tiện nhất. Trong quá trình thực hiện, nhóm đã sử dụng công nghệ HTML5 WebSocket nhằm xây dựng công cụ như một ứng dụng thời gian thực, giúp người dùng có thể có được những thông tin mới nhất về dự án mà không cần phải tải lại trang web.</p></div></div>

2. Phạm vi

Phạm vi của đề tài là nghiên cứu cơ sở lý thuyết của Agile, phương pháp Scrum và vận dụng các công nghệ phù hợp để xây dựng công cụ quản lý dự án theo phương pháp phát triển phần mềm Scrum.

3. Đối tượng

Công cụ hướng đến đối tượng sử dụng là các đội ngũ phát triển dự án công nghệ thông tin theo phương pháp Scrum.

4. Kết quả mong đợi

- Hiểu được triết lý Agile cùng phương pháp Scrum.
- Xây dựng thành công một công cụ quản lý dự án theo phương pháp Scrum. Công cụ giúp nhóm phát triển dự án công nghệ thông tin xây dựng sản phẩm bám sát theo quy trình của Scrum.
- Xây dựng một ứng dụng web thời gian thực, dữ liệu sẽ được cập nhật thời gian thực thay vì tuân theo mô hình Web yêu cầu và đáp ứng truyền thống.

Kế hoạch thực hiện:

STT	Thời gian	Nội dung công việc	Người thực hiện
1	15/3/2014 - 31/3/2014	Thu thập thông tin về đề tài: khảo sát, tìm hiểu thực trạng, tìm giải pháp giải quyết vấn đề.	Phạm Ánh Dương
2	1/4/2014 - 7/4/2014	Phân tích đề tài, phác thảo các tính năng cần thiết để giải quyết các vấn đề đã thu thập được. Quyết định mô hình giải quyết bài toán đặt ra, lựa chọn giải pháp thích hợp.	Phạm Ánh Dương
3	8/4/2014 - 15/4/2014	Vẽ sơ đồ Use Case và viết mô tả cho các Use Case tương ứng.	Phạm Ánh Dương
4	16/4/2014 -	Thiết kế giao diện của ứng dụng. Hoàn	Phạm Ánh Dương

	1/5/2014	thành phiên bản prototype.	
5	2/5/2014 - 5/5/2014	Xây dựng tính năng quản lí các dự án.	Phạm Ánh Dương
6	6/5/2014 - 12/5/2014	Xây dựng tính năng quản lí User Story.	Phạm Ánh Dương
7	13/5/2014 - 17/05/2014	Xây dựng tính năng quản lí nhân lực cho dự án.	Phạm Ánh Dương
8	18/5/2014 - 25/5/2014	Xây dựng tính năng quản lí Sprint.	Phạm Ánh Dương
9	26/5/2014 - 4/6/2014	Xây dựng tính năng quản lí Task.	Phạm Ánh Dương
10	5/6/2014 - 11/6/2014	Xây dựng tính năng báo cáo thống kê.	Phạm Ánh Dương
11	12/6/2014 - 27/6/2014	Xây dựng tính năng cập nhật thời gian thực.	Phạm Ánh Dương
12	28/6/2014	Kiểm thử.	Phạm Ánh Dương
13	29/6/2014	Sửa các lỗi từ kiểm thử.	Phạm Ánh Dương
14	30/6/2014 - 7/7/2014	Tổng hợp báo cáo khoá luận tốt nghệ.	Phạm Ánh Dương
15	8/7/2014 - 10/7/2014	Chỉnh sửa báo cáo khoá luận tốt nghiệp theo ý kiến của giảng viên hướng dẫn lần 1.	Phạm Ánh Dương
16	11/7/2014 - 13/7/2014	Chỉnh sửa báo cáo khoá luận tốt nghiệp theo ý kiến của giảng viên hướng dẫn lần 2.	Phạm Ánh Dương

17	14/7/2014- 15/7/2014	Hoàn chỉnh báo cáo khoá luận tốt nghiệp.	Phạm Ánh Dương
18	16/7/2014	Nộp báo cáo khoá luận tốt nghiệp.	Phạm Ánh Dương
Xác nhận của CBHD (Ký tên và ghi rõ họ tên)		TP. HCM, ngày....thángnăm..... Sinh viên (Ký tên và ghi rõ họ tên)	

MỤC LỤC

Chương 1.	TỔNG QUAN	4
1.1.	Giới thiệu vấn đề	4
1.2.	Mục tiêu xây dựng	4
Chương 2.	CƠ SỞ LÝ THUYẾT	6
2.1.	Giới thiệu về Agile	6
2.1.1.	Tuyên ngôn Agile	7
2.1.2.	Giá trị cốt lõi của Agile	9
2.1.2.1.	Cá nhân và sự tương tác.....	9
2.1.2.2.	Phần mềm chạy tốt hơn là tài liệu đầy đủ.....	11
2.1.2.3.	Cộng tác với khách hàng hơn là thương thảo hợp đồng	11
2.1.2.4.	Phản hồi với thay đổi hơn là bám sát kế hoạch	12
2.1.3.	Đặc trưng của Agile.....	13
2.1.3.1.	Tính lặp	13
2.1.3.2.	Tính tăng và tiến hóa	14
2.1.3.3.	Tính thích ứng.....	14
2.1.3.4.	Nhóm tự tổ chức và liên chức năng.....	14
2.1.3.5.	Quản lý tiến trình thực nghiệm.....	15
2.1.3.6.	Đối thoại trực diện	15
2.1.3.7.	Phát triển dựa trên giá trị	16
2.2.	Scrum.....	16
2.2.1.	Giới thiệu Scrum	16
2.2.2.	Lý thuyết Scrum	17
2.2.3.	Nhóm Scrum.....	18

2.2.4.	Các sự kiện Scrum.....	20
2.2.5.	Các đồ nghề trong Scrum	25
2.2.6.	Quy trình triển khai Scrum.....	28
2.2.7.	Điểm mạnh của Scrum	29
2.2.8.	So sánh với các phương pháp truyền thống	30
Chương 3.	XÂY DỰNG ỨNG DỤNG	31
3.1.	Phân tích yêu cầu	31
3.2.	Sơ đồ Use Case.....	32
3.2.1.	Sơ đồ Use Case tổng thể.....	32
3.2.2.	Thao tác thông thường.....	33
3.2.2.1.	Use Case đăng nhập.....	33
3.2.2.2.	Use Case chọn dự án làm việc	34
3.2.2.3.	Use Case đăng xuất.....	34
3.2.2.4.	Use Case bình luận	34
3.2.2.5.	Use Case sửa thông tin cá nhân	35
3.2.3.	Quản lý dự án	35
3.2.3.1.	Use Case thêm thành viên.....	36
3.2.3.2.	Use Case xóa thành viên.....	36
3.2.3.3.	Use Case thêm dự án	36
3.2.3.4.	Use Case cập nhật thông tin dự án.....	37
3.2.3.5.	Use Case cập nhật thông tin dự án.....	37
3.2.4.	Quản lý Sprint	38
3.2.4.1.	Use Case khởi tạo Sprint	38
3.2.4.2.	Use Case thêm Story vào Sprint	38

3.2.4.3.	Use Case hoàn thành Sprint.....	39
3.2.5.	Quản lí User Story	40
3.2.5.1.	Thêm Story	40
3.2.5.2.	Sửa Story.....	40
3.2.5.3.	Xóa Story	41
3.2.6.	Quản lí Team	42
3.2.6.1.	Thêm Team.....	42
3.2.6.2.	Sửa Team	42
3.2.6.3.	Xóa Team.....	43
3.2.6.4.	Loại thành viên khỏi Team	43
3.2.6.5.	Thêm thành viên vào Team	44
3.2.7.	Quản lí Task	44
3.2.7.1.	Thêm Task vào User Story	44
3.2.7.2.	Sửa Task	45
3.2.7.3.	Xóa Task.....	45
3.3.	Sơ đồ Activity.....	46
3.3.1.	Chức năng đăng nhập.....	46
3.3.2.	Chức năng tạo dự án.....	46
3.3.3.	Chức năng tạo Sprint.....	47
3.3.4.	Chức năng tạo User Story.....	47
3.3.5.	Chức năng tạo Task	48
3.3.6.	Chức năng tạo Team.....	48
3.4.	Tổ chức cơ sở dữ liệu	49
3.4.1.	Danh sách các bảng trong cơ sở dữ liệu.....	49

3.4.2.	Bảng access_link	50
3.4.3.	Bảng activity.....	50
3.4.4.	Bảng attach	51
3.4.5.	Bảng comment.....	52
3.4.6.	Bảng permission	52
3.4.7.	Bảng dự án.....	52
3.4.8.	Bảng project_user	53
3.4.9.	Bảng role	53
3.4.10.	Bảng role_permission.....	54
3.4.11.	Bảng sprint	54
3.4.12.	Bảng sprint_team.....	55
3.4.13.	Bảng story.....	55
3.4.14.	Bảng story_team.....	56
3.4.15.	Bảng task	56
3.4.16.	Bảng team.....	57
3.4.17.	Bảng user	58
3.5.	Sơ đồ thiết kế giao diện	59
Chương 4.	TRIỂN KHAI ỨNG DỤNG.....	60
4.1.	Cài đặt.....	60
4.2.	Kiến trúc hệ thống	60
4.3.	HTML5 WebSocket	61
4.3.1.	Kiến trúc truyền thống.....	61
4.3.2.	Công nghệ WebSocket	63
4.3.3.	Quy trình bắt tay kết nối của WebSocket.....	64

4.3.4.	Định dạng gói tin	64
4.3.5.	Quy trình bắt tay ngắt kết nối	65
4.4.	Chương trình.....	66
4.4.1.	Màn hình đăng nhập	66
4.4.2.	Màn hình quản lí dự án.....	66
4.4.3.	Màn hình quản lí User Story	67
4.4.4.	Màn hình quản lí nhân lực.....	68
4.4.5.	Màn hình quản lí Sprint.....	68
4.4.6.	Taskboard	69
4.4.7.	Burndown chart	70
4.4.8.	Burnup chart	70
4.4.9.	Màn hình phân quyền	71
KẾT LUẬN		72
HƯỚNG PHÁT TRIỂN		73
TÀI LIỆU THAM KHẢO.....		74

DANH MỤC HÌNH VẼ

Hình 2.1: Mức độ phổ biến của các phương pháp Agile	7
Hình 2.2: Các phân đoạn lặp trong Agile.....	13
Hình 2.3: Các sự kiện trong Sprint.....	21
Hình 2.4: Quy trình triển khai Sprint trong Scrum	28
Hình 3.1: Use Case tổng thể.....	32
Hình 3.2: Use Case các thao tác thông thường	33
Hình 3.3: Use Case quản lí Project	35
Hình 3.4: Use Case quản lí Sprint.....	38
Hình 3.5: Use Case quản lí User Story	40
Hình 3.6: Use Case quản lí Team.....	42
Hình 3.7: Use Case quản lí Task.....	44
Hình 3.8: Activity diagram đăng nhập.....	46
Hình 3.9: Activity diagram tạo dự án.....	47
Hình 3.10: Activity diagram tạo Sprint.....	47
Hình 3.11: Activity diagram tạo User Story	47
Hình 3.12: Activity diagram tạo Task.....	48
Hình 3.13: Activity diagram tạo Team	48
Hình 3.14: Sơ đồ ERD	50
Hình 3.15: Sơ đồ thiết kế giao diện.....	59
Hình 4.1: Mô hình MVC trong Laravel	61
Hình 4.2: HTTP long polling giữ yêu cầu HTTP mở trong một khoảng thời gian để kiểm tra việc cập nhật	62
Hình 4.3: So sánh WebSocket với các công nghệ polling	64
Hình 4.4: Header của gói tin WebSocket.....	65
Hình 4.5: Màn hình đăng nhập.....	66
Hình 4.6: Màn hình quản lí dự án	66
Hình 4.7: Màn hình quản lí User Story.....	67
Hình 4.8: Màn hình quản lí nhân lực	68

Hình 4.9: Màn hình quản lí Sprint	68
Hình 4.10: Màn hình Taskboard	69
Hình 4.11: Màn hình burndown chart.....	70
Hình 4.12: Màn hình burnup chart.....	70
Hình 4.13: Màn hình phân quyền.....	71

DANH MỤC BẢNG

Bảng 2.1: So sánh Sprint Backlog và Product Backlog	27
Bảng 2.2: So sánh Scrum với các phương pháp truyền thống	30
Bảng 3.1: Danh sách các bảng trong cơ sở dữ liệu	49
Bảng 3.2: Chi tiết bảng access_link	50
Bảng 3.3: Chi tiết bảng activity.....	51
Bảng 3.4: Chi tiết bảng sttach	52
Bảng 3.5: Chi tiết bảng comment.....	52
Bảng 3.6: Chi tiết bảng permission	52
Bảng 3.7: Chi tiết bảng project	53
Bảng 3.8: Chi tiết bảng project_user.....	53
Bảng 3.9: Chi tiết bảng role	54
Bảng 3.10: Chi tiết bảng role_permission.....	54
Bảng 3.11: Chi tiết bảng sprint	55
Bảng 3.12: Chi tiết bảng sprint_team.....	55
Bảng 3.13: Chi tiết bảng story.....	56
Bảng 3.14: Chi tiết bảng story_team.....	56
Bảng 3.15: Chi tiết bảng task	57
Bảng 3.16: Chi tiết bảng team.....	58
Bảng 3.17: Chi tiết bảng user.....	58

TÓM TẮT KHÓA LUẬN

Agile nổi lên như một triết lý phát triển phần mềm mới mẻ với cách tiếp cận khác khắc phục những điểm yếu của các phương pháp phát triển phần mềm truyền thống. Nó là một tập các phương thức phát triển phần mềm theo hướng lặp và tăng dần, trong đó Scrum nổi bật lên là phương pháp được sử dụng nhiều nhất.

Khóa luận tập trung tìm hiểu khái niệm, đặc trưng, cách hiện thực của Agile cùng một phương pháp nổi tiếng của Agile là Scrum. Trong quá trình tìm hiểu, nhóm nhận thấy Scrum là phương pháp dễ tiếp cận phù hợp với quy mô đa dạng các dự án phần mềm nên nhóm đã tập trung vào xây dựng công cụ hỗ trợ quản lý dự án theo phương pháp Scrum.

Ứng dụng bám sát vào quy trình phát triển phần mềm theo phương pháp Scrum. Với giao diện trực quan, tiện lợi và cập nhật thời gian thực, ứng dụng nhằm giúp người dùng cảm thấy tiện lợi nhất khi sử dụng, dễ dàng hơn khi xây dựng dự án theo phương pháp Scrum.

MỞ ĐẦU

Trong ngành công nghệ phần mềm ở thời kì ban đầu, mô hình phát triển phần mềm thác nước là xu hướng chủ đạo. Trong mô hình thác nước, việc thiết kế chiếm vai trò quan trọng và cần phải có trước khi phát triển. Với việc sử dụng mô hình thác nước, người ta sẽ bàn giao phần mềm trong cuối quá trình phát triển.

Với việc phân chia từng giai đoạn cụ thể như vậy, việc xảy ra sai sót trong một giai đoạn đặc biệt là các giai đoạn ban đầu có thể khiến việc phát triển bị trì hoãn, tốn nhiều thời gian hơn dự kiến, phần mềm làm ra không thực sự thỏa mãn yêu cầu của khách hàng hoặc tệ hơn là dự án có thể thất bại hoàn toàn.

Trong bối cảnh đó, Agile ra đời quy định một tập các cách tiếp cận mới mẽ nhằm khắc phục những điểm yếu của các phương pháp truyền thống. Phương pháp này đã được nhiều công ty trên thế giới áp dụng từ nhiều năm nay và tại Việt Nam nhiều công ty phần mềm cũng đã bắt đầu chú ý tới vì nó giúp quá trình phát triển phần mềm nhanh chóng, nhanh có sản phẩm và tối ưu hóa đầu tư.

Agile là một tập các quy tắc mà các phương pháp ứng dụng nó phải tuân theo. Các phương pháp phát triển phần mềm như Scrum, XP (Extreme Programming), Feature Driven Development (FDD), Test Driven Development (TDD)... mới thực sự áp dụng các quy tắc này vào thực tế. Trong đó Scrum là quy trình đang được áp dụng rộng rãi nhất và mang lại hiệu quả rất cao (chiếm 52% trong tổng số các phương pháp – thống kê năm 2011).

Là một kĩ sư phần mềm việc tìm hiểu và nắm bắt Agile là một yêu cầu cần thiết vì vậy nhóm sinh viên quyết thực hiện đề tài **“Nghiên cứu mô hình Agile và phát triển công cụ hỗ trợ”** và lựa chọn Scrum làm phương pháp chính để tập trung xây dựng ứng dụng.

Mục đích

Tìm hiểu và nắm bắt Agile và phương pháp Scrum, làm rõ ưu điểm và khuyết điểm của các chương trình đã có, từ đó xác định được mục tiêu phát triển chương trình của mình.

Phạm vi đề tài

Phạm vi của đề tài là nghiên cứu cơ sở lý thuyết và nền tảng công nghệ để vận dụng cài đặt các tính năng cơ bản của một công cụ hỗ trợ quản lý dự án theo phương pháp Scrum. Trong khuôn khổ khóa luận, nhóm sẽ thực hiện một số vấn đề sau:

- Tìm hiểu, nghiên cứu Agile và phương pháp Scrum
- Xây dựng công cụ hỗ trợ quản lý dự án theo phương pháp Scrum.

Chương 1. TỔNG QUAN

1.1. Giới thiệu vấn đề

Hiện nay, có rất nhiều phương pháp/quy trình phát triển phần mềm, mỗi phương pháp có ưu nhược điểm riêng nhưng ta có thể chia làm hai loại chính: các phương pháp truyền thống và các phương pháp linh hoạt (Agile). Điểm khác biệt nổi bật nhất là các dự án theo Agile dựa trên các giá trị thương mại, đáp ứng thích nghi với thay đổi thực tế hơn là theo kế hoạch đề ra như các phương pháp truyền thống.

Bên cạnh những lợi ích mà Agile mang lại, việc áp dụng nó cũng tồn tại một số khó khăn. Cụ thể, khi sử dụng phương pháp Scrum theo cách truyền thống với giấy, bút và bảng vẽ việc triển khai dự án có thể gặp một số vấn đề sau:

- Khó khăn về việc lưu trữ thông tin dự án đã làm trước đó nhằm thống kê, tham khảo hoặc phát triển thêm.
- Khó khăn về không gian khi yêu cầu tất cả các thành viên trong nhóm phải làm việc gần nhau, việc này đôi khi là không thể.
- Trong thời buổi Internet phổ biến, việc sử dụng Agile thủ công không còn phù hợp. Người quản lý hoặc khách hàng không cần phải đến công ty mà vẫn có thể theo dõi tiến độ dự án một cách nhanh chóng và chính xác.
- Các thành viên trong nhóm có thể dễ dàng biết được tiến độ của các thành viên khác. Khi có vấn đề phát sinh thì các thành viên có thể hỗ trợ, giúp đỡ cùng giải quyết vấn đề.
- Một hệ thống quản lý sẽ có thêm những tính năng về báo cáo, thống kê, theo dõi tiến độ thuận tiện hơn nhiều so với việc sử dụng giấy bút truyền thống.

1.2. Mục tiêu xây dựng

Để hỗ trợ cho việc triển khai dự án theo Agile với phương pháp Scrum, đề tài sẽ tập trung vào nghiên cứu các nguyên tắc của Agile cùng với quy trình triển khai

Scrum. Từ những kiến thức đó, nhóm sẽ xây dựng một công cụ hỗ trợ quản lý dự án theo phương pháp Scrum với một số chức năng cơ bản nhất.

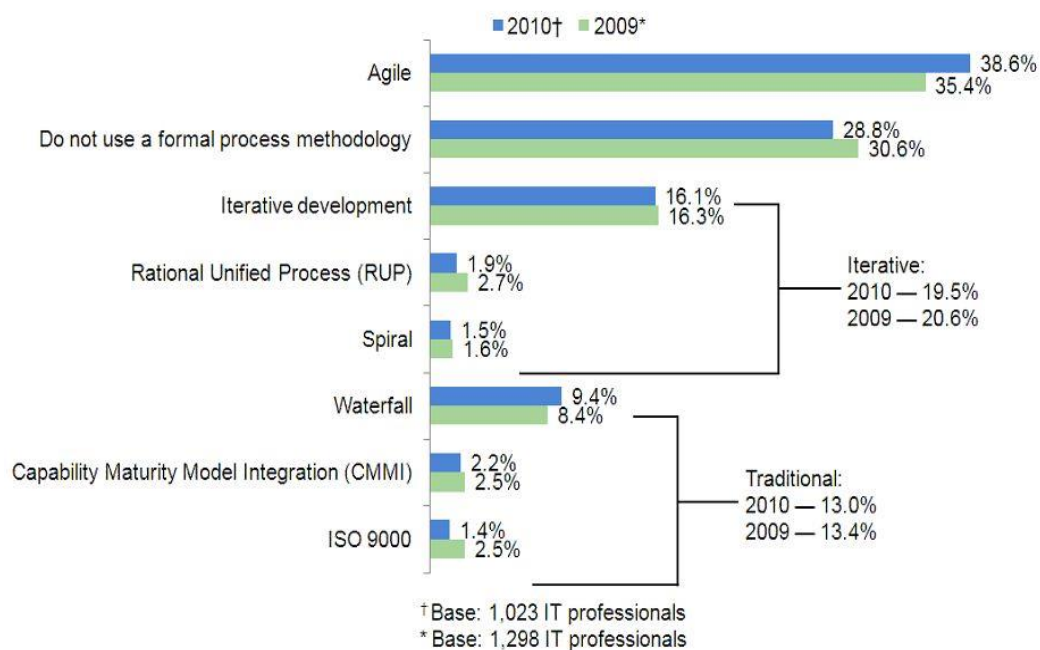
- Quản lý danh sách các dự án của đội ngũ phát triển.
- Hỗ trợ cho Product Owner quản lý danh sách các User Story, theo dõi được tiến độ dự án thông qua tiến độ thực hiện của các Story.
- Quản lý danh sách nhân lực của dự án cùng các Team phát triển.
- Quản lý các thành phần của Scrum như Sprint, User Story, Task.
- Cập nhật thời gian thực, người dùng sẽ không phải liên tục tải lại trang để cập nhật dữ liệu, công cụ sẽ liên tục tự động cập nhật mỗi khi dữ liệu thay đổi.
- Biểu đồ thống kê thuận tiện cho người dùng theo dõi tiến độ của dự án.

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu về Agile

Phát triển phần mềm linh hoạt (Agile software development – gọi tắt là Agile) là một triết lý cùng với nhóm các phương pháp và phương pháp luận phát triển phần mềm dựa trên các nguyên tắc phát triển phân đoạn lặp (iterative) và tăng trưởng (incremental). Agile thường sử dụng cách lập kế hoạch thích ứng (adaptive planning), việc phát triển và chuyển giao theo hướng tiến hóa; sử dụng các khung thời gian ngắn và linh hoạt để dễ dàng phản hồi lại với các thay đổi trong quá trình phát triển. Ngày nay, triết lý Agile đã vượt xa khỏi khu vực truyền thống của mình là phát triển phần mềm mà còn đóng góp sự thay đổi trong cách thức làm việc, quản lý, sản xuất ở các ngành khác như sản xuất, dịch vụ, bán hàng, marketing, giáo dục...

Thuật ngữ "Agile" chính thức được sử dụng rộng rãi theo cách thống nhất kể từ khi "Tuyên ngôn Agile" được giới thiệu ra công chúng năm 2001. Nhờ tính linh hoạt, đa dạng và hiệu quả cao, các phương pháp Agile ngày càng trở thành sự lựa chọn hàng đầu của các khách hàng, nhà phát triển, các công ty phát triển phần mềm. Theo khảo sát của hãng nghiên cứu thị trường Forrester, mức độ phổ biến của Agile hiện đang ở mức cao nhất, và gấp nhiều lần so với các phương pháp truyền thống.



Hình 2.1: Mức độ phổ biến của các phương pháp Agile
(Nguồn: Forrester Research))

2.1.1. Tuyên ngôn Agile

Vào tháng 2 năm 2001, mười bảy nhà phát triển phần mềm đã gặp gỡ nhau ở Snowbird, Utah Resort để thảo luận về các phương pháp phát triển phần mềm gọn nhẹ và linh hoạt. Họ đã cùng nhau công bố "Tuyên ngôn Phát triển phần mềm linh hoạt" ("Manifesto for Agile Software Development", gọi tắt là "Tuyên ngôn Agile") nhằm định nghĩa về Agile. Đây là cột mốc quan trọng của Agile cho dù trước đó, các phương pháp Agile như XP, Scrum... đã được sử dụng thành công ở rất nhiều nơi. Nhưng phải tới khi có sự xuất hiện của "Tuyên ngôn Agile", cùng với sự ra đời của các hiệp hội chuyên ngành Agile như Agile Alliance hay Scrum Alliance, các phương pháp Agile mới có một sự phát triển vượt bậc. Văn bản này rất ngắn, dễ hiểu, và rất quan trọng vì nó đưa ra các giá trị cốt lõi nhất mà toàn bộ những người dùng Agile tuân thủ; mặc dù các phương pháp họ đề xuất hoặc sử dụng trong thực tiễn có thể rất khác nhau. Toàn văn Tuyên ngôn Agile như sau:

Chúng tôi đã phát hiện ra cách phát triển phần mềm tốt hơn bằng cách hiện thực nó và giúp đỡ người khác thực hiện. Qua quá trình này, chúng tôi đã đi đến việc đánh giá cao:

- ***Cá nhân và sự tương tác*** hơn là quy trình và công cụ.
- ***Phần mềm chạy tốt*** hơn là tài liệu đầy đủ.
- ***Cộng tác với khách hàng*** hơn là đàm phán hợp đồng.
- ***Phản hồi với các thay đổi*** hơn là bám sát kế hoạch.

Mặc dù các điều bên phải vẫn còn giá trị, nhưng chúng tôi đánh giá cao hơn các mục ở bên trái.

Bên cạnh đó, các nhà phát triển còn nhấn mạnh mười hai nguyên lý sau Tuyên ngôn Agile nhằm giúp các nhà mọi người vận dụng các phương pháp Agile trong thực tiễn. Các nguyên lý được liệt kê sau đây:

- Ưu tiên cao nhất của chúng tôi là thỏa mãn khách hàng thông qua việc chuyển giao sớm và liên tục các phần mềm có giá trị.
- Chào đón việc thay đổi yêu cầu, thậm chí rất muộn trong quá trình phát triển. Quy trình Agile thực hiện những thay đổi phục vụ cho lợi thế cạnh tranh của khách hàng.
- Thường xuyên chuyển giao phần mềm tới khách hàng, từ vài tuần đến vài tháng, ưu tiên cho các khoảng thời gian ngắn hơn.
- Nhà kinh doanh và nhà phát triển phải làm việc cùng nhau hàng ngày trong suốt dự án.
- Xây dựng các dự án xung quanh những cá nhân tận tụy. Cung cấp cho họ môi trường và sự hỗ trợ cần thiết, và tin tưởng họ để hoàn thành công việc.
- Phương pháp hiệu quả nhất để truyền đạt thông tin tới và trong nội bộ nhóm phát triển là đối thoại trực tiếp.
- Phần mềm chạy tốt là thước đo chính của tiến độ.

- Các quy trình Agile thúc đẩy sự phát triển bền vững. Các nhà tài trợ, nhà phát triển, và người dùng có thể duy trì một nhịp độ liên tục không giới hạn.
- Liên tục quan tâm đến các kỹ thuật nổi trội và thiết kế tốt để gia tăng sự linh hoạt.
- Sự đơn giản - nghệ thuật tối đa hóa lượng công việc chưa xong - là căn bản.
- Các kiến trúc tốt nhất, yêu cầu tốt nhất, và thiết kế tốt nhất sẽ được làm ra bởi các nhóm tự tổ chức.
- Đội ngũ sẽ thường xuyên suy nghĩ về việc làm sao để trở nên hiệu quả hơn, sau đó họ sẽ điều chỉnh và thay đổi các hành vi của mình cho phù hợp.

Các nguyên lý này, cùng với năm điểm cốt lõi trong "Tuyên ngôn Agile" sẽ dẫn đường cho các nhà thực hành Agile (Agile practitioner) vận dụng tốt các phương pháp Agile vào thực tiễn.

2.1.2. Giá trị cốt lõi của Agile

2.1.2.1. Cá nhân và sự tương tác

Cá nhân và sự tương tác giữa họ là nhân tố chủ yếu để nhóm đạt được hiệu suất cao. Các nghiên cứu về "bão hòa truyền thông" trong một dự án cho thấy rằng, khi không có vấn đề trong giao tiếp, nhóm có thể thực hiện công việc tốt hơn 50 lần so với mức trung bình trong lĩnh vực của mình. Để tạo điều kiện cho việc giao tiếp, các phương pháp linh hoạt thường xuyên sử dụng chu trình thanh tra và thích nghi (Inspect and Adapt). Chu trình này có thể diễn ra hằng phút với lập trình cặp (pair-programming), hằng giờ với tích hợp liên tục (continuous integration), hằng ngày với standup meeting, hằng phân đoạn với các buổi họp sơ kết và cải tiến.

Tuy nhiên, chỉ tăng tần suất phản hồi và giao tiếp là không đủ để loại bỏ các vấn đề về truyền thông. Chu kỳ thanh tra và thích nghi làm việc tốt chỉ khi các thành viên trong nhóm thể hiện những hành vi quan trọng sau:

- Tôn trọng giá trị của mỗi cá nhân.
- Trung thực trong truyền thông.

- Minh bạch về dữ liệu, hoạt động, và quyết định.
- Tin tưởng vào sự hỗ trợ của mỗi cá nhân với nhóm.
- Cam kết với nhóm và các mục tiêu của nhóm.

Để thúc đẩy các hành vi này, nhà quản lý linh hoạt phải cung cấp một môi trường hỗ trợ, các nhà huấn luyện phải tạo điều kiện thuận lợi, và các thành viên của nhóm phải thể hiện chúng. Chỉ khi đó nhóm mới có thể phát huy được hết tiềm năng của mình. Đạt tới các hành vi đó khó khăn hơn rất nhiều so với bề ngoài của chúng. Ban lãnh đạo và thành viên nhóm phải tạo điều kiện cho những xung đột tích cực nhằm mục đích:

- Cải tiến quy trình phụ thuộc vào nhóm trong việc tìm ra danh sách các trở ngại hoặc vấn đề trong tổ chức, đối mặt với chúng một cách trung thực, và sau đó loại bỏ chúng một cách có hệ thống tùy theo độ ưu tiên.
- Đổi mới xảy ra với việc tự do trao đổi các ý tưởng mâu thuẫn lẫn nhau, một hiện tượng được nghiên cứu và viết thành tài liệu bởi Takeuchi và Nonaka, những người cha đỡ đầu của Scrum.
- Việc hướng các nhóm tới mục tiêu chung đòi hỏi nhóm phải đối mặt và giải quyết các vấn đề về xung đột.
- Cam kết làm việc cùng nhau sẽ xảy ra chỉ khi mọi người đồng ý với mục tiêu chung và sau đó đấu tranh cho sự tiến bộ của bản thân cũng như nhóm.

Ý cuối cùng ở trên nói về cam kết là đặc biệt quan trọng. Đó là khi mà các cá nhân và các nhóm được cam kết mà họ cảm thấy có trách nhiệm với việc cung cấp các giá trị cao, đó là điểm mấu chốt đối với các nhóm phát triển phần mềm. Các phương pháp linh hoạt tạo điều kiện cho việc cam kết bằng cách khuyến khích các nhóm đưa ra một danh sách các công việc được ưu tiên hóa, để họ tự quản lý công việc của mình, và tập trung vào cải tiến về cách thực hiện các công việc đó.

Để tạo ra các nhóm có hiệu suất cao, các phương pháp linh hoạt coi trọng cá nhân và sự tương tác hơn là quy trình và công cụ. Thực tế cho thấy rằng, tất cả các

phương pháp Agile tìm kiếm sự gia tăng trong truyền thông và cộng tác thông qua việc kiểm tra thường xuyên các chu trình thanh tra và thích nghi. Tuy nhiên, các chu trình đó chỉ thực sự làm việc tốt khi mà các nhà lãnh đạo Agile khuyến khích các cuộc xung đột tích cực, thứ cần thiết để xây dựng một nền tảng vững chắc cho sự trung thực, tính minh bạch, lòng tin, sự tôn trọng, và cam kết từ các nhóm Agile của họ.

2.1.2.2. Phần mềm chạy tốt hơn là tài liệu đầy đủ

Phần mềm chạy tốt là một trong những khác biệt lớn mà phát triển linh hoạt mang lại. Tất cả các phương pháp Agile thể hiện Tuyên ngôn Agile bằng cách nhấn mạnh việc cung cấp một phần nhỏ của phần mềm chạy tốt tới khách hàng sau một khoảng thời gian nhất định.

2.1.2.3. Cộng tác với khách hàng hơn là thương thảo hợp đồng

Trong hai thập kỷ qua, tỉ lệ thành công của các dự án tăng hơn hai lần trên toàn thế giới. Điều này được cho là vì các dự án nhỏ hơn và mức độ chuyển giao thường xuyên đã cho phép khách hàng cung cấp các thông tin phản hồi về phần mềm một cách đều đặn hơn. Các tác giả của bản tuyên ngôn đã làm sáng tỏ điều này khi họ nhấn mạnh rằng việc để khách hàng tham gia vào quá trình phát triển phần mềm là hết sức cần thiết để dự án thành công.

Các phương pháp phát triển linh hoạt đã thúc đẩy giá trị này bằng cách đưa vào một đồng minh tích cực của khách hàng làm việc sát cánh với đội phát triển. Lấy một ví dụ, một nhóm Scrum đầu tiên có hàng ngàn khách hàng. Sẽ là không khả thi nếu cho phép tất cả khách hàng tham gia vào quá trình phát triển sản phẩm, vì vậy họ chọn ra một vị đại sứ của khách hàng, được gọi là Product Owner, để đại diện cho không chỉ tất cả khách hàng trong trường hợp này mà còn bao gồm cả quản lý, dịch vụ khách hàng, và các bên liên quan khác. Product Owner có trách nhiệm cập nhật danh sách yêu cầu về sản phẩm sau mỗi bốn tuần thời điểm mà nhóm Scrum phát hành phiên bản sản phẩm chạy tốt, có tính đến yếu tố thực tế

cùng phản hồi của khách hàng và các bên liên quan. Điều này cho phép khách hàng có thể giúp định hình sản phẩm phần mềm đang được tạo ra.

Cộng tác với khách hàng (hoặc đại diện của khách hàng) trên cơ sở hàng ngày là một trong những lý do lý giải tại sao các dự án Agile có tỉ lệ thành công cao hơn gấp đôi so với các dự án truyền thống tính trung bình trên toàn thế giới. Các phương pháp phát triển theo Agile đều tạo ra một vị trí đặc biệt trong đội ngũ phát triển để dành riêng cho vị khách hàng đại diện này.

2.1.2.4. Phản hồi với thay đổi hơn là bám sát kế hoạch

Phản hồi với thay đổi là điều cần thiết cho việc tạo ra một sản phẩm làm hài lòng khách hàng cũng như mang lại những giá trị kinh doanh. Dữ liệu thống kê cho thấy hơn 60% các yêu cầu về sản phẩm hay dự án bị thay đổi suốt quá trình phát triển phần mềm. Ngay cả khi các dự án truyền thống kết thúc đúng thời gian, đúng kinh phí, với tất cả các tính năng theo kế hoạch, nhưng khách hàng thường không hài lòng vì những gì họ thấy không thật sự đúng như những gì họ muốn. Luật Humphrey nói rằng khách hàng không bao giờ biết những gì họ muốn cho đến khi họ thấy phần mềm hoạt động. Nếu khách hàng không nhìn thấy phần mềm hoạt động cho đến khi kết thúc dự án, sẽ là quá muộn cho việc kết hợp các thông tin phản hồi của họ ở thời điểm này. Các phương pháp phát triển linh hoạt tìm kiếm sự phản hồi của khách hàng trong suốt dự án để có thể kết hợp thông tin phản hồi và thông tin mới ngay cả khi sản phẩm đang được phát triển.

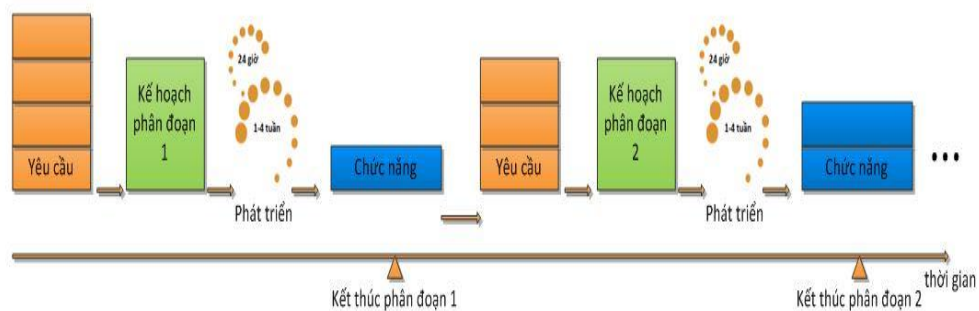
Tất cả các phương pháp phát triển linh hoạt đều được tích hợp sẵn những quy trình với các thay đổi trong một khoảng thời gian đều đặn dựa trên những thông tin phản hồi từ phía khách hàng cũng như bên đại diện của khách hàng. Các kế hoạch được thiết kế để sao cho luôn cung cấp giá trị kinh doanh cao nhất. Bởi vì 80% giá trị nằm trong 20% các tính năng, một dự án phát triển theo Agile chạy tốt có xu hướng kết thúc sớm, trong khi hầu hết các dự án truyền thống thường kết thúc trễ.

2.1.3. Đặc trưng của Agile

Có rất nhiều phương pháp Agile với các hướng tiếp cận rất khác nhau. Bên cạnh các cách thức tổ chức công việc, thiết lập quy trình, các phương pháp Agile còn nghiên cứu và đưa vào sử dụng các công cụ và kỹ thuật đặc thù như công cụ tích hợp liên tục, kiểm thử đơn vị, mẫu thiết kế, tái cấu trúc, phát triển hướng kiểm thử (Test Driven Development), phát triển hướng hành vi (Behaviour Driven Development), hay lập trình theo cặp (Pair Programming)... để đảm bảo và gia tăng tính linh hoạt.

2.1.3.1. Tính lặp

Dự án sẽ được thực hiện trong các phân đoạn lặp đi lặp lại. Các phân đoạn (được gọi là Iteration hoặc Sprint) này thường có khung thời gian ngắn (từ một đến bốn tuần). Trong mỗi phân đoạn này, nhóm phát triển thực hiện đầy đủ các công việc cần thiết như lập kế hoạch, phân tích yêu cầu, thiết kế, triển khai, kiểm thử (với các mức độ khác nhau) để cho ra các phần nhỏ của sản phẩm. Các phương pháp Agile thường phân rã dự án thành các phần nhỏ với quá trình lập kế hoạch đơn giản và gọn nhẹ nhất có thể, và không thực hiện việc lập kế hoạch dài hạn. Thậm chí công việc lập kế hoạch, làm mịn kế hoạch được thực hiện liên tục trong suốt quá trình làm việc.



Hình 2.2: Các phân đoạn lặp trong Agile

(Nguồn: hanoiscrum.net)

2.1.3.2. Tính tăng và tiến hóa

Cuối các phân đoạn, nhóm phát triển thường cho ra các phần nhỏ của sản phẩm cuối cùng. Các phần nhỏ này thường là đầy đủ, có khả năng chạy tốt, được kiểm thử cẩn thận và có thể sử dụng ngay. Theo thời gian, phân đoạn này tiếp nối phân đoạn kia, các phần chạy được này sẽ được tích lũy, lớn dần lên cho tới khi toàn bộ yêu cầu của khách hàng được thỏa mãn. Khác với mô hình phát triển thác nước vốn chỉ cho phép nhìn thấy toàn bộ các chức năng tại thời điểm kết thúc dự án, sản phẩm trong các dự án Agile lớn dần lên theo thời gian, tiến hóa cho tới khi đạt được trạng thái đủ để phát hành.

2.1.3.3. Tính thích ứng

Do các phân đoạn chỉ kéo dài trong một khoảng thời gian ngắn, và việc lập kế hoạch cũng được điều chỉnh liên tục, nên các thay đổi trong quá trình phát triển (yêu cầu thay đổi, thay đổi công nghệ, thay đổi định hướng về mục tiêu...) đều có thể được đáp ứng theo cách thích hợp. Ví dụ, trong Scrum - phương pháp phổ biến nhất hiện nay - trong khi nhóm phát triển sản xuất ra các gói phần mềm, khách hàng có thể đưa thêm các yêu cầu mới, Product Owner có thể đánh giá các yêu cầu này và có thể đưa vào làm việc trong phân đoạn (được gọi là Sprint trong Scrum) tiếp theo. Nhờ đó, các quy trình Agile thường thích ứng rất tốt với các thay đổi.

2.1.3.4. Nhóm tự tổ chức và liên chức năng

Cấu trúc nhóm Agile thường là liên chức năng và tự tổ chức. Theo đó, các nhóm này tự thực hiện lấy việc phân công công việc mà không dựa trên các mô tả cứng nhắc về chức danh hay làm việc dựa trên một sự phân cấp rõ ràng trong tổ chức. Các nhóm này cộng tác với nhau để ra quyết định, theo dõi tiến độ, giải quyết các vấn đề mà không chờ mệnh lệnh của các cấp quản lý. Họ không làm việc theo cơ chế "mệnh lệnh và kiểm soát". Nhóm tự tổ chức có nghĩa là nó đã đủ các kỹ năng cần thiết cho việc phát triển phần mềm, do vậy nó có thể được trao quyền để tự ra quyết định, tự quản lý và tổ chức lấy công việc của chính mình để đạt được hiệu quả cao nhất.

2.1.3.5. Quản lý tiến trình thực nghiệm

Các nhóm Agile ra các quyết định dựa trên các dữ liệu thực tiễn thay vì tính toán lý thuyết hay đưa ra các giả định. Việc phân nhỏ dự án thành các phân đoạn ngắn góp phần gia tăng các điểm mốc để nhóm phát triển thu thập thêm dữ kiện cho phép điều chỉnh các chiến lược phát triển của mình. Nói cách khác, Agile rút ngắn vòng đời phản hồi để dễ dàng thích nghi và gia tăng tính linh hoạt. Theo thời gian, các chiến lược này sẽ tiến gần đến trạng thái tối ưu, nhờ đó nhóm có thể kiểm soát được tiến trình, và nâng cao năng suất lao động.

2.1.3.6. Đối thoại trực diện

Một số mô hình phát triển phần mềm dựa rất nhiều vào công việc giấy tờ, từ việc thu thập yêu cầu người dùng, viết đặc tả hệ thống, các thiết kế hệ thống v.v. Agile không phản đối công dụng của công việc tài liệu hóa, nhưng đánh giá cao việc giao tiếp trực tiếp hơn gián tiếp thông qua giấy tờ. Về yêu cầu của khách hàng, Agile khuyến khích nhóm phát triển trực tiếp nói chuyện với khách hàng để hiểu rõ hơn về cái khách hàng thực sự cần, thay vì phụ thuộc nhiều vào các loại văn bản. Trong giao tiếp giữa nội bộ nhóm phát triển với nhau, thay vì một lập trình viên (thực hiện việc mã hóa) và một kỹ sư (thực hiện việc thiết kế) giao tiếp với nhau thông qua bản thiết kế, Agile khuyến khích hai người này trực tiếp trao đổi và thống nhất với nhau về thiết kế của hệ thống và cùng nhau triển khai thành các chức năng theo yêu cầu.

Bản thân các nhóm Agile thường nhỏ (ít hơn mười hai người, một nhóm lớn thường được phân nhỏ với cơ chế hợp tác đặc thù để luôn luôn có thông lượng giao tiếp tối đa) để đơn giản hóa quá trình giao tiếp, thúc đẩy việc cộng tác hiệu quả. Các dự án lớn muốn dùng Agile thường phải phân thành nhóm nhỏ đồng thời làm việc với nhau hướng đến một mục tiêu chung. Việc này có thể đòi hỏi nhiều nỗ lực trong việc điều phối các mức ưu tiên giữa các nhóm.

Các nhóm phát triển thường tạo ra các thói quen và cơ chế trao đổi trực diện thường xuyên. Một trong các cơ chế thường thấy là các cuộc họp tập trung hằng

ngày (daily meeting, daily Scrum, standup meeting). Tại đây, tất cả các thành viên được yêu cầu nói rõ cho nhóm của mình biết mình đã làm gì, đang làm gì, sắp làm gì và đang gặp phải khó khăn nào trong quá trình làm việc. Khi cơ chế này được thực hiện hiệu quả, nhóm luôn luôn nắm được tình hình công việc của mình, có các hành động thích hợp để vượt qua các cản trở để thực hiện thành công mục tiêu của dự án.

2.1.3.7. Phát triển dựa trên giá trị

Một trong các nguyên tắc cơ bản của Agile là "phần mềm chạy tốt chính là thước đo của tiến độ". Nguyên tắc này giúp nhóm dám loại bỏ đi các công việc dư thừa không trực tiếp mang lại giá trị cho sản phẩm. Ví dụ, một nhóm thấy rằng có thể không cần phải viết ra tất cả các thiết kế của hệ thống, mà họ chỉ cần phác thảo các thiết kế này lên bảng, rồi cùng nhau triển khai các chức năng của hệ thống. Nhưng nếu như Product Owner quyết định rằng, khi chuyển giao sản phẩm, nhóm phát triển phải kèm theo thiết kế chi tiết của hệ thống vì chúng chiếm 20% giá trị của dự án theo yêu cầu của khách hàng, và vì khách hàng cần nó để chứng minh tính đúng đắn của các chức năng, và để phát triển tiếp các phân hệ tiếp theo của sản phẩm; thì nhóm phát triển sẽ phải thực hiện việc tài liệu hóa đầy đủ.

Để vận hành được cơ chế "làm việc dựa trên giá trị", nhóm Agile thường làm việc trực tiếp và thường xuyên với khách hàng (hay đại diện của khách hàng), cộng tác trực tiếp với họ để biết yêu cầu nào có độ ưu tiên cao hơn, mang lại giá trị hơn sớm nhất có thể cho dự án. Nhờ đó các dự án Agile thường giúp khách hàng tối ưu hóa được giá trị của dự án. Vì thế, Agile gia tăng đáng kể độ hài lòng của khách hàng.

2.2. Scrum

2.2.1. Giới thiệu Scrum

Scrum là khung làm việc đã được sử dụng để quản lý quá trình phát triển các sản phẩm phức tạp từ đầu những năm 1990. Scrum không hẳn là một quy trình hay

một kĩ thuật cụ thể để xây dựng sản phẩm; hơn thế, nó là một khung làm việc cho phép bạn sử dụng nhiều quy trình và kĩ thuật khác nhau.

Khung làm việc Scrum bao gồm một nhóm Scrum với các vai trò được phân định rõ ràng, các sự kiện, các đồ nghề và các quy tắc. Mỗi thành phần trong khung làm việc phục vụ một mục đích rõ ràng và có vai trò nòng cốt trong cho thành công của Scrum.

2.2.2. Lý thuyết Scrum

Scrum được xây dựng dựa trên lý thuyết quản lý tiến trình thực nghiệm. Lý thuyết này chỉ ra rằng tri thức đến từ kinh nghiệm và việc ra quyết định được dựa trên những gì đã biết. Scrum sử dụng các tiếp cận lặp, tăng dần để tối ưu hóa việc dự đoán và kiểm soát rủi ro.

Ba yếu tố nòng cốt tạo thành một mô hình quản lý tiến trình thực nghiệm gồm: tính minh bạch, sự thanh tra và sự thích nghi.

Minh bạch

Trong Scrum, tính minh bạch được đề cao như là giá trị cốt lõi cơ bản nhất. Muốn thành công với Scrum, thông tin liên quan tới quá trình phát triển phải minh bạch và thông suốt. Từ đó mọi người ở các vai trò các nhau có đủ thông tin cần thiết để tiến hành các quyết định có giá trị để nâng cao hiệu quả công việc. Các công cụ và cuộc họp trong Scrum luôn đảm bảo thông tin được minh bạch cho các bên.

Thanh tra

Người sử dụng Scrum phải thường xuyên thanh tra các đồ nghề và tiến độ đến đích để phát hiện các bất thường không theo ý muốn. Tần suất thanh tra không nên quá dày để khỏi ảnh hưởng đến công việc. Công tác thanh tra có ích nhất khi được thực hiện bởi người có kĩ năng tại các điểm quan trọng của công việc.

Thích nghi

Nếu một người thanh tra xác định được rằng có vấn đề nào đó vượt quá giới hạn cho phép, và hậu quả của vấn đề đó đối với sản phẩm là không thể chấp nhận được, thì quy trình hoặc phải được điều chỉnh. Sự điều chỉnh phải được tiến hành

càng sớm càng tốt để giảm thiểu các sai sót khác có thể xảy ra.

Scrum cung cấp bốn cơ hội chính thức cho việc thanh tra và thích nghi trong các Sự kiện Scrum, bao gồm:

- Buổi họp kế hoạch Sprint
- Họp Scrum hằng ngày
- Sơ kết Sprint
- Cải tiến Sprint

2.2.3. Nhóm Scrum

Nhóm Scrum bao gồm Product Owner, Team, và Scrum Master. Các nhóm Scrum là các nhóm tự quản và liên chức năng. Các nhóm tự quản tự mình chọn cách thức tốt nhất để hoàn thành công việc của họ, chứ không bị chỉ đạo bởi ai đó bên ngoài nhóm. Các nhóm liên chức năng có đủ kỹ năng cần thiết để hoàn thành công việc mà không phụ thuộc vào bất kỳ người ngoài nào khác. Mô hình nhóm trong Scrum được thiết kế để tối ưu hóa sự linh hoạt, sáng tạo và năng suất.

Các nhóm Scrum chuyển giao sản phẩm theo phân đoạn lặp đi lặp lại và tăng dần, tối đa hóa cơ hội cho các phản hồi. Việc chuyển giao tăng dần các gói sản phẩm hoàn thành đảm bảo một phiên bản khả dụng của sản phẩm luôn luôn sẵn sàng để cung cấp tới Product Owner.

Product Owner

Product Owner chịu trách nhiệm tối đa hóa giá trị của sản phẩm và công việc của nhóm phát triển. Cách thức để đạt được điều đó có thể rất khác nhau giữa các tổ chức, nhóm Scrum và các cá nhân. Product Owner là một người chủ yếu chịu trách nhiệm về việc quản lý Product Backlog:

- Mô tả các hạng mục Product backlog;
- Trình tự của các hạng mục trong Product Backlog để đạt được mục đích và hoàn thành dự án.
- Đảm bảo cho Product Backlog là luôn luôn thông suốt, và rõ ràng tới tất cả mọi người, và chỉ ra những gì mà nhóm Scrum sẽ làm.

- Đảm bảo cho Team hiểu rõ các hạng mục trong Product Backlog với các mức độ cần thiết.

Product Owner có thể thực hiện công việc trên, hoặc để Team làm. Tuy nhiên, Product Owner vẫn phải chịu trách nhiệm chính. Product Owner là một người, không phải là một ủy ban. Product Owner có thể cần tới một ủy ban tham gia vào Product Backlog, nhưng những người trong ủy ban muốn thay đổi trình tự các hạng mục trong Product Backlog phải thuyết phục được Product Owner. Để Product Owner thành công, cả tổ chức phải tôn trọng các quyết định của người này. Các quyết định đó được hiển thị trong nội dung và thứ tự trong Product Backlog. Không ai ngoài Product Owner được phép yêu cầu Team làm gì khác, và Team cũng không được phép làm gì theo lời bất cứ người nào khác.

Team

Team (nhóm phát triển) gồm các chuyên gia làm việc để cho ra các phần tăng trưởng có thể phát hành được vào cuối mỗi Sprint. Chỉ các thành viên của Team mới tạo ra các phần tăng trưởng này. Team được cấu trúc và trao quyền để tổ chức và quản lý công việc của họ. Team có các đặc trưng sau:

- Đó là nhóm tự tổ chức. Không ai (kể cả Scrum Master) có quyền yêu cầu Team làm sao để chuyển Product Backlog thành các phần tăng trưởng có thể chuyển giao được.
- Đó là nhóm liên chức năng, với tất cả các kỹ năng cần thiết để tạo ra phần tăng trưởng của sản phẩm.
- Scrum không ghi nhận một chức danh nào trong Team ngoài Developer, theo tính chất công việc của người này; không có ngoại lệ cho quy tắc này.
- Các thành viên Team có thể có các kỹ năng chuyên biệt và các chuyên môn đặc thù, nhưng họ phải chịu trách nhiệm dưới một thể thống nhất là Team.
- Team không chứa các nhóm con nào khác với các chức năng đặc thù như nhóm kiểm thử hay phân tích nghiệp vụ.

Độ lớn tối ưu của Team là đủ nhỏ để giữ được sự linh hoạt và đủ lớn để hoàn thành công việc. Ít hơn ba người có thể làm giảm sự tương tác và dẫn đến năng suất

thấp. Các Team nhỏ hơn có thể phải đối mặt với các ràng buộc kỹ năng trong suốt Sprint, dẫn đến Team khó có thể chuyển giao gói tăng trưởng phát hành được. Một nhóm nhiều hơn chín người cần nhiều sự điều phối hơn. Các Team lớn phát sinh quá nhiều vấn đề phức tạp để thực hiện việc kiểm soát tiến trình thực nghiệm. Các vai trò Product Owner và Scrum Master không được tính vào kích thước của Team, trừ khi họ cũng kiêm luôn vai trò là thành viên của Team.

Scrum Master

Scrum Master chịu trách nhiệm đảm bảo mọi người hiểu và dùng được Scrum. Scrum Master thực hiện việc này bằng cách đảm bảo nhóm Scrum tuân thủ lý thuyết, thực tiễn và các quy tắc của Scrum. Scrum Master là một lãnh đạo, nhưng cũng là đầy tớ của nhóm Scrum. Scrum Master giúp đỡ những người ngoài nhóm Scrum hiểu cách phải tương tác với nhóm sao cho hiệu quả nhất. Scrum Master giúp đỡ tất cả mọi người thay đổi các mối tương tác này để tối đa hóa giá trị mà nhóm Scrum tạo ra.

2.2.4. Các sự kiện Scrum

Các sự kiện được mô tả trong Scrum nhằm tạo ra thói quen và để giảm thiểu những buổi họp hành vốn không được định nghĩa trong Scrum. Scrum dùng các sự kiện được đóng khung thời gian (time-boxed), nghĩa là mỗi sự kiện có giới hạn thời gian tối đa. Điều này đảm bảo thời lượng vừa đủ để tránh lãng phí thời gian không cần thiết cho sự kiện.

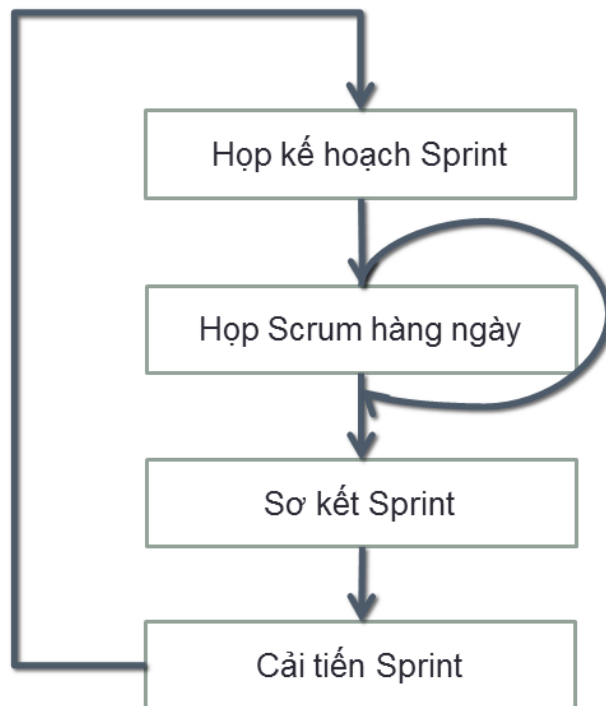
Mỗi sự kiện trong Scrum là một cơ hội chính thức để thực hiện cơ chế thanh tra và thích nghi. Các sự kiện này được thiết kế đặc biệt để đảm bảo sự minh bạch và thanh tra. Nếu không thực hiện được các điều này có thể dẫn đến giảm thiểu tính minh bạch và đánh mất cơ hội để thanh tra và thích nghi.

Sprint

Trái tim của Scrum chính là Sprint, một khung thời gian (time-box) có thời gian một tháng hoặc ngắn hơn để tạo ra các phần tăng trưởng của sản phẩm có thể phát hành được. Sprint có khoảng thời gian nhất quán trong suốt quá trình phát

triển. Một Sprint mới bắt đầu ngay khi Sprint trước khép lại.

Sprint bao gồm một cuộc họp kế hoạch Sprint (Sprint Planning Meeting), các cuộc họp Scrum hằng ngày (Daily Scrum), một buổi sơ kết Sprint (Sprint Review), và một buổi họp cải tiến Sprint (Sprint Retrospective).



Hình 2.3: Các sự kiện trong Sprint

Trong suốt Sprint:

- Không cho phép bất kì sự thay đổi nào ảnh hưởng đến mục tiêu của Sprint.
- Thành phần Team được giữ nguyên.
- Mục tiêu chất lượng không được cắt giảm.
- Phạm vi dự án có thể được làm rõ và tái thương lượng giữa Product Owner và Team.

Mỗi Sprint có thể được coi như một tiểu dự án với độ dài một tháng. Giống như dự án, Sprint được dùng để hoàn tất điều gì đó. Mỗi Sprint có một định nghĩa về việc phải xây dựng cái gì, một bản thiết kế và bản kế hoạch linh hoạt hướng dẫn quá trình xây dựng đó, các công việc cần làm, và sản phẩm của quá trình đó. Sprint được giới hạn trong vòng một tháng. Khi Sprint bị kéo dài quá thì định nghĩa về

việc cần xây dựng cái gì đó có thể bị thay đổi, sự phức tạp sẽ gia tăng và rủi ro sẽ tăng theo. Sprint cũng sẽ giới hạn rủi ro trong phạm vi chi phí của một tháng.

Sprint có thể bị hủy trước khi khung thời gian trôi qua. Chỉ có Product Owner mới đủ thẩm quyền kết thúc Sprint, mặc dù Product Owner có thể chịu ảnh hưởng bởi những bên hữu quan khác, bởi Team, hoặc bởi Scrum Master. Một Sprint có thể bị hủy nếu như mục tiêu Sprint có thể trở nên lỗi thời. Điều này xảy ra khi công ty chuyển hướng kinh doanh hoặc khi công nghệ có sự thay đổi. Nói chung, Sprint có thể bị hủy nếu nó không mang lại điều gì có ích. Thế nhưng, do thời gian trong mỗi Sprint tương đối ngắn, nên việc hủy một Sprint không mấy khi gây hậu quả lớn.

Khi Sprint bị hủy, các phần sản phẩm đã hoàn chỉnh được xem xét lại. Nếu phần nào đó của công việc đó là có thể chuyển giao được, thì Product Owner có thể chấp nhận chúng. Tất cả các hạng mục Product Backlog chưa hoàn tất sẽ được ước lượng lại và đặt ngược trở lại Product Backlog để phát triển tiếp. Việc hủy Sprint sẽ gây lãng phí tài nguyên, do mọi người phải họp lại để lên kế hoạch cho một Sprint mới. Việc hủy Sprint thường gây tổn hại nhất định cho Team, và rất ít khi xảy ra.

Họp kế hoạch Sprint

Công việc trong Sprint được lên kế hoạch trong buổi họp kế hoạch Sprint (Sprint Planning Meeting). Kế hoạch cho Sprint được tạo ra từ nỗ lực cộng tác của toàn bộ nhóm Scrum. Buổi họp kế hoạch Sprint được đóng khung trong tám tiếng cho mỗi Sprint một tháng. Với các Sprint ngắn hơn thì thời gian cho buổi họp được rút ngắn lại. Ví dụ như một Sprint hai tuần có thể chỉ cần họp kế hoạch tới bốn tiếng là đủ. Buổi họp kế hoạch Sprint có hai phần, mỗi phần chiếm một nửa khung thời gian. Hai phần của buổi họp kế hoạch Sprint lần lượt trả lời hai câu hỏi sau đây:

- Sprint này phải chuyển giao cái gì?
- Làm sao để đạt được điều đó?

Họp Scrum hàng ngày

Cuộc họp Scrum hàng ngày (Daily Scrum) ngày được đóng khung trong 15 phút để Team đồng bộ hóa các hoạt động của thành viên và tạo lập kế hoạch cho 24 giờ tiếp theo. Điều này có được nhờ việc thanh tra các công việc kể từ cuộc họp

Scrum hằng ngày trước, và dự báo những công việc sẽ được hoàn thành trước buổi họp lần sau.

Cuộc họp Scrum hằng ngày được tổ chức tại cùng một địa điểm để giảm thiểu sự phức tạp không cần thiết. Trong suốt cuộc họp, mỗi thành viên Team giải thích rõ:

- Việc gì đã được thực hiện kể từ lần họp trước?
- Việc gì sẽ được hoàn thành trước buổi họp lần sau?
- Có vấn đề gì nảy sinh trong quá trình làm việc?

Team sử dụng cuộc họp Scrum hằng ngày để đánh giá tiến độ công việc hướng tới mục tiêu Sprint và đánh giá tiến độ của công việc trong Sprint Backlog. Cuộc họp Scrum hằng ngày tối ưu hóa khả năng để Team có thể đạt được mục tiêu của Sprint. Team thường họp mặt ngay sau khi họp xong Scrum hằng ngày để tái lập kế hoạch cho các công việc còn lại trong Sprint. Hằng ngày, Team có thể giải thích cho Product Owner và Scrum Master biết họ định làm gì với tư cách là một nhóm tự quản để hoàn thành mục tiêu và tạo ra các phần tăng trưởng cần thiết trong Sprint.

Scrum Master đảm bảo việc Team tham gia họp, nhưng chính Team mới có trách nhiệm chính trong cuộc họp Scrum hằng ngày. Scrum Master phải đảm bảo Team giữ cuộc họp ngắn gọn trong phạm vi khung thời gian 15 phút. Scrum Master phải áp đặt quy tắc về việc chỉ có Team mới được tham gia cuộc họp Scrum hằng ngày. Cuộc họp Scrum hằng ngày không phải là cuộc họp báo cáo tình hình, mà là để mọi người biến đổi các hạng mục Product Backlog thành phần tăng trưởng cần thiết.

Họp Scrum hằng ngày sẽ cải tiến quá trình giao tiếp, lược bỏ các buổi họp hành không cần thiết, nhận biết và loại bỏ các trở lực trong quá trình phát triển, nhấn mạnh và phát huy các quyết định nhanh chóng, và nâng cao mức độ hiểu biết của Team về dự án. Cuộc họp này là chìa khóa của cơ chế thanh tra và thích nghi trong Scrum.

Sơ kết Sprint

Buổi sơ kết Sprint (Sprint Review) được tổ chức khi Sprint kết thúc để rà soát lại phần tăng trưởng vừa làm ra trong Sprint đó, và để thực hiện các biện pháp thích nghi nếu cần. Trong cuộc họp này, nhóm Scrum và các bên hữu quan sẽ trao đổi với nhau về những gì vừa hoàn thành trong Sprint vừa rồi và những sự thay đổi trong Product Backlog trong suốt quá trình thực hiện Sprint, người tham dự cuộc họp sẽ hợp tác để thảo luận về những công việc sắp triển khai. Đây là cuộc họp không trang trọng, và việc trình bày về gói tăng trưởng chủ yếu nhằm mục đích cung cấp các phản hồi hữu ích và khuyến khích sự cộng tác giữa các bên.

Cuộc họp này được đóng khung trong bốn giờ cho các Sprint có độ dài một tháng. Sprint ngắn hơn thì thời gian họp rút bớt cho phù hợp. Ví dụ, một Sprint hai tuần chỉ cần đến hai giờ cho buổi sơ kết Sprint.

Buổi sơ kết Sprint có một số đặc điểm sau:

- Product Owner nhận biết phần nào là hoàn thành và phần nào chưa hoàn thành.
- Team thảo luận những điều thuận lợi trong Sprint vừa qua, những khó khăn mà nhóm đã trải qua, và cách thức giải quyết các vấn đề đó.
- Team trình diễn các phần việc đã hoàn thành và trả lời các câu hỏi mọi người về gói tăng trưởng.
- Product Owner trao đổi về Product Backlog. Dựa trên tiến độ hiện thời, Product Owner đưa ra dự đoán ngày hoàn thành dự án.
- Toàn bộ nhóm thảo luận về những gì sẽ làm, nhờ đó buổi sơ kết Sprint cung cấp các giá trị đầu vào cho buổi họp kế hoạch Sprint tiếp theo.

Kết quả của cuộc họp sơ kết Sprint là một bản Product Backlog đã được cập nhật, với các hạng mục dự định sẽ được triển khai trong Sprint tới. Product Backlog có thể được điều chỉnh toàn diện để thích ứng với các cơ hội mới.

Cải tiến Sprint

Buổi họp cải tiến Sprint (Sprint Retrospective) là cơ hội để nhóm Scrum tự thanh tra và đưa ra kế hoạch cho các cải tiến trong Sprint tiếp theo. Buổi họp cải tiến Sprint được tổ chức ngay sau sơ kết Sprint và trước khi cuộc họp kế hoạch

Sprint tiếp theo diễn ra. Cuộc họp này được đóng khung trong phạm vi ba giờ cho các Sprint một tháng. Sprint ngắn hơn thì cuộc họp sẽ được rút ngắn lại cho phù hợp.

Mục đích của cuộc họp cải tiến Sprint là để:

- Thanh tra lại tất cả các yếu tố trong Sprint vừa diễn ra, từ con người, các mối quan hệ, quy trình, và công cụ.
- Nhận biết và xếp đặt lại các hạng mục chủ chốt đã được thực hiện tốt, và các cải tiến dự định.
- Tạo ra một kế hoạch để triển khai các cải tiến cách thức làm việc của nhóm Scrum.

Scrum Master động viên nhóm Scrum cải tiến, trong phạm vi khung làm việc Scrum, quy trình phát triển và các biện pháp thực hành để nâng cao hiệu quả cho Sprint tiếp theo. Trong cuộc họp cải tiến Sprint, nhóm Scrum sẽ lập kế hoạch để gia tăng chất lượng sản phẩm. Kết thúc cuộc họp cải tiến Sprint, nhóm Scrum phải nhận biết ra các cải tiến sẽ được triển khai trong Sprint tới.

2.2.5. Các đồ nghề trong Scrum

Product Backlog

Product Backlog là một danh sách sắp thứ tự tất cả những gì cần thiết của sản phẩm. Product Owner là người chịu trách nhiệm về Product Backlog, nội dung của nó, sự hiện diện, và thứ tự các hạng mục trong đó. Product Backlog có thể không hoàn chỉnh. Phiên bản sớm nhất của Product Backlog chỉ cho thấy các yêu cầu được tìm hiểu rõ ràng từ lúc đầu tiên. Product Backlog sẽ tiến hóa cùng với sản phẩm và môi trường mà nó sẽ được sử dụng. Product Backlog là động, nó thay đổi thường xuyên để nhận biết những gì mà sản phẩm cần phải có để có tính cạnh tranh và hữu ích. Chừng nào sản phẩm còn đó, thì Product Backlog cũng hiện diện.

Product Backlog liệt kê tất cả các chức năng, yêu cầu, cải thiện, và lỗi cần thiết để làm nên sản phẩm trong tương lai. Các hạng mục trong Product Backlog được gắn với các thuộc tính như: mô tả, thứ tự, và ước lượng. Product Backlog

thường được sắp xếp theo các giá trị, độ rủi ro, độ ưu tiên, và sự cần thiết. Các hạng mục đứng đầu danh sách sẽ ưu tiên phát triển trước. Càng ở thứ tự cao hơn, các hạng mục càng được quan tâm nhiều hơn, và được tập trung nỗ lực nhiều hơn.

Khi sản phẩm được đưa vào sử dụng và mang lại giá trị, thị trường sẽ cung cấp các phản hồi, Product Backlog sẽ trở thành một danh sách lớn hơn và toàn diện hơn. Nhu cầu thì không ngừng thay đổi, vì thế một Product Backlog luôn luôn thay đổi. Sự thay đổi trong các yêu cầu nghiệp vụ, điều kiện thị trường, hay công nghệ có thể dẫn đến các thay đổi trong Product Backlog.

Trong trường hợp nhiều nhóm Scrum làm việc với nhau trên cùng một sản phẩm. Một Product Backlog được dùng để mô tả những công việc tới đây của sản phẩm. Khi đó, các hạng mục có thể được nhóm lại theo một tính chất nào đó.

Việc làm mịn Product Backlog là hoạt động thêm vào các chi tiết, ước lượng, và trình tự của các hạng mục trong Product Backlog. Đây là quá trình liên tục, theo đó Product Owner và Team thảo luận về các chi tiết của từng hạng mục. Trong suốt quá trình làm mịn này, các hạng mục liên tục được xem xét và rà soát cẩn thận. Tuy nhiên, chúng có thể được cập nhật tại bất kì thời điểm nào bởi Product Owner.

Sprint Backlog

Sprint Backlog là tập hợp các hạng mục Product Backlog được lựa chọn để phát triển trong Sprint, kèm theo một kế hoạch để chuyển giao phần tăng trưởng của sản phẩm và hiện thực hóa mục tiêu Sprint. Sprint Backlog là một bản dự báo của Team về những chức năng sẽ có trong phần tăng trưởng sẽ được chuyển giao. Sprint Backlog xác định công việc mà Team phải làm để biến các hạng mục Product Backlog thành phần tăng trưởng hoàn thành. Sprint Backlog cho thấy tất cả những việc Team cần phải làm để tiến tới mục tiêu Sprint.

Sprint Backlog là một kế hoạch với chi tiết vừa đủ để những thay đổi về tiến độ công việc có thể nhìn thấy được trong các cuộc họp Scrum hằng ngày. Team chỉnh sửa Sprint Backlog trong suốt Sprint, và Sprint Backlog sẽ được cập nhật trong suốt thời gian đó.

Mỗi khi có thêm việc mới, Team đưa vào Sprint Backlog. Khi công việc bắt

đầu hay kết thúc, giá trị ước lượng về thời gian còn lại để hoàn tất công việc được cập nhật. Khi có phần nào đó của kế hoạch là không cần thiết, chúng sẽ bị bỏ đi. Chỉ có Team mới có thể thay đổi Sprint Backlog trong Sprint. Sprint Backlog là một bức tranh thời gian thực về công việc mà Team lên kế hoạch để hoàn thành trong Sprint, và nó cơ bản thuộc về Team.

So sánh Product Backlog và Sprint Backlog

Product Backlog và Sprint Backlog có nhiều điểm chung nhưng khác mục đích sử dụng. Product Backlog được quản lý bởi Product Owner và cung cấp một cái nhìn ở mức cao về công việc mà Team phát triển phải hoàn thành để tạo ra sản phẩm.

Ngược lại, Team của bạn tạo Sprint Backlog, chứa danh sách chi tiết tất cả các Task mà Team cần phải hoàn thành để kết thúc các User Story trong Sprint. Trong Product Backlog, Team của bạn đánh giá User Story bằng Story Point còn trong Sprint Backlog, Team đánh giá Task bằng số giờ thực hiện.

Product Owner cập nhật Product Backlog hàng tuần, nhưng Team phải cập nhật Sprint Backlog hàng ngày.

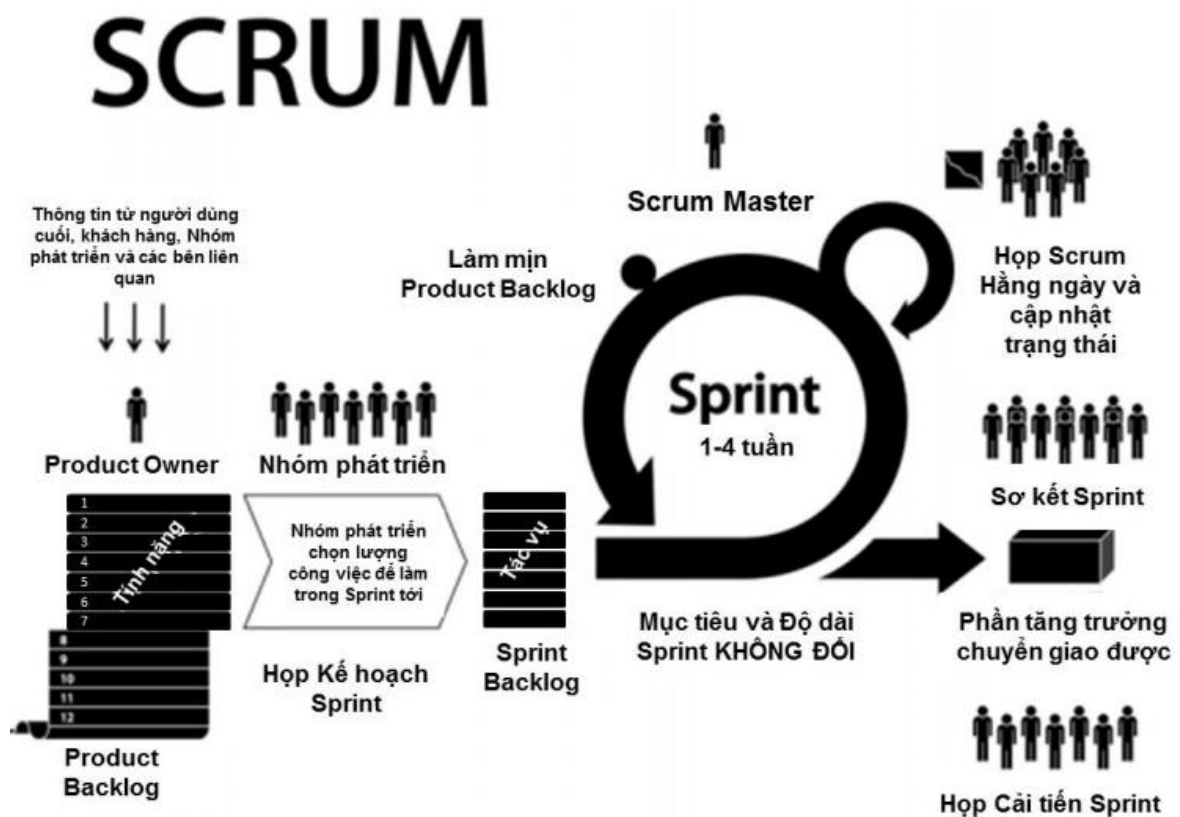
Product Owner duy trì Product Backlog suốt dự án, nhưng Team tạo Sprint Backlog mới vào mỗi Sprint.

Bảng sau tóm tắt những điểm khác nhau giữa Product Backlog và Sprint Backlog.

Mục	Product Backlog	Sprint Backlog
Mức độ chi tiết	Ít chi tiết hơn	Rất chi tiết
Đơn vị ước lượng	Story points	Số giờ
Chủ sở hữu	Product owner	Team phát triển
Duyệt lại	Hàng tuần	Hàng ngày
Kéo dài	Dự án	Sprint

Bảng 2.1: So sánh Sprint Backlog và Product Backlog

2.2.6. Quy trình triển khai Scrum



Hình 2.4: Quy trình triển khai Sprint trong Scrum

(Nguồn: Sách Scrum Căn bản Phiên bản 1.2)

Bước 1: Product Owner đưa ra danh sách các chức năng (User Story) được sắp xếp theo độ quan trọng. Danh sách này tồn tại và cập nhật trong suốt quá trình phát triển sản phẩm.

Bước 2: Bắt đầu mỗi Sprint, nhóm Scrum sẽ tiến hành cuộc họp kế hoạch Sprint.

- Team sẽ chọn ra các User Story để thực hiện trong Sprint.
- Phân chia các User Story thành các Task.
- Kết quả của bước này là một bản Sprint Backlog gồm danh sách các User Story sẽ được thực hiện trong Sprint cùng các Task của nó.

Bước 3: Mỗi ngày Team tổ chức họp hàng ngày để thanh tra quá trình thực hiện, thực hiện các thay đổi nếu cần để hoàn thành công việc. Đồng thời đây cũng là thời điểm để cập nhật các Task và Sprint Backlog.

Bước 4: Trước khi Sprint kết thúc, nhóm Scrum tiến hành họp sơ kết Sprint với các bên liên quan, trình bày về những việc đã làm, giới thiệu kết quả, xem xét lại những việc chưa hoàn thành.

Bước 5: Tiếp theo, nhóm Scrum sẽ tổ chức cuộc họp cải tiến Sprint. Mọi người xem xét lại Sprint vừa diễn ra và nhận phản hồi, ý tưởng để cải tiến trong Sprint tiếp theo.

Sau khi kết thúc Sprint thì Product Owner sẽ cập nhật lại sự thay đổi trong Product Backlog. Lúc này đội ngũ phát triển đã sẵn sàng để bắt đầu Sprint tiếp theo. Quy trình này cứ lặp đi lặp lại cho đến khi hoàn thành tất cả các yêu cầu của khách hàng.

2.2.7. Điểm mạnh của Scrum

Điểm mạnh nhất đó là việc linh hoạt, dự án không được cố định từ đầu về thời gian hoàn thành hay những yêu cầu mà nó sẽ được xác định khi phát triển thực tế.

Phân phối sản phẩm mềm dẻo: sản phẩm chuyển giao được xác định linh hoạt theo môi trường sử dụng thực tế. Thời gian biểu linh hoạt: có thể muộn hoặc sớm hơn so với kế hoạch ban đầu.

Chất lượng sản phẩm tốt và giảm rủi ro sản xuất, chi phí thấp. Khả năng trao đổi giữa khách hàng và nhà phát triển, giữa những thành viên trong đội được đặt lên mức cao.

Tốc độ phát triển nhanh, tiết kiệm thời gian. Việc chuẩn bị hành động cho những thay đổi trong quá trình phát triển tốt hơn vì hầu như hàng ngày luôn có những buổi họp đánh giá lại ở những vòng lặp phát triển.

Các lỗi và các vấn đề được phát hiện sớm hơn rất nhiều so với các phương pháp truyền thống bởi vì khách hàng được tham gia đánh giá rất nhiều và đầu ra của

sản phẩm rất nhanh. Và khi đi sai hướng, có thể hủy ngay Sprint đó để quay lại với bản kế hoạch.

2.2.8. So sánh với các phương pháp truyền thống

Đặc điểm	Waterfall	Spiral	Scrum
Xác định các giai đoạn phát triển	Bắt buộc	Bắt buộc	Chỉ có giai đoạn lập kế hoạch và kết thúc
Sản phẩm cuối cùng	Được xác định trong quá trình lập kế hoạch	Được xác định trong quá trình lập kế hoạch	Xác định trong quá trình xây dựng dự án
Chi phí sản phẩm	Được xác định trong quá trình lập kế hoạch	Thay đổi cục bộ	Xác định trong quá trình xây dựng dự án
Ngày hoàn thành sản phẩm	Được xác định trong quá trình lập kế hoạch	Thay đổi cục bộ	Xác định trong quá trình xây dựng dự án
Đáp ứng với môi trường sử dụng	Trong kế hoạch ban đầu	Trong kế hoạch ban đầu	Xuyên suốt từ kế hoạch đến xây dựng và kết thúc
Kinh nghiệm trao đổi	Đào tạo trước cho đến khi bắt tay làm dự án	Đào tạo trước cho đến khi bắt tay làm dự án	Thực hiện trong quá trình làm dự án
Khả năng thành công	Thấp	Trung bình thấp	Cao

Bảng 2.2: So sánh Scrum với các phương pháp truyền thống
(Nguồn *developer.vumon.vn*)

Chương 3. XÂY DỰNG ỨNG DỤNG

3.1. Phân tích yêu cầu

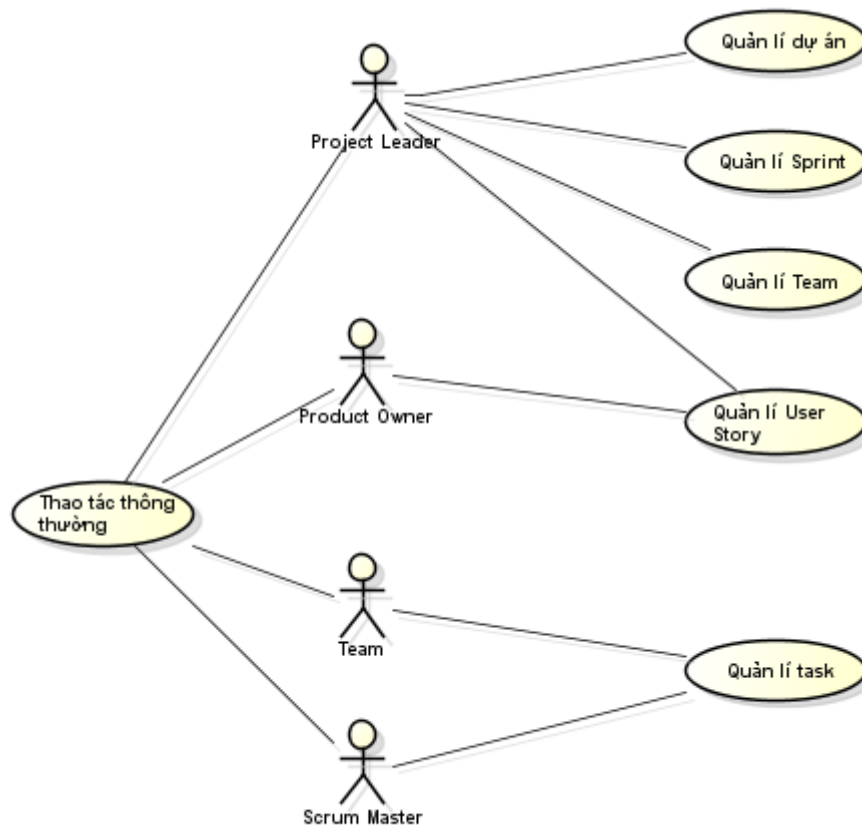
- Công cụ chia đội ngũ tham gia xây dựng dự án thành 4 vai trò sau: Product Owner, Project Leader, Scrum Master, Team.
- Project Leader có quyền quản trị dự án, Sprint, Team, Story và thành viên.
- Công cụ có thể quản lý nhiều dự án khác nhau cùng một lúc. Khi tạo dự án mới Project Leader phải cung cấp các thông tin sau: tên dự án, mô tả, Product Owner, Project Leader, ngày bắt đầu, ngày kết thúc dự kiến và ghi chú bổ sung.
- Các dự án theo mô hình Scrum bao gồm các User Story. Chỉ có 2 vai trò là Product Owner và Project Leader là có quyền thao tác, chỉnh sửa User Story. Một User Story gồm có các mục sau: tên, độ quan trọng, ước tính ban đầu, cách demo, lưu ý.
- Sau khi có User Story, Project Leader sẽ ước lượng quy mô, yêu cầu của dự án, tình hình nhân lực từ đó xây dựng Team phát triển. Mỗi nhóm sẽ gồm các mục sau: tên nhóm, mô tả, các thành viên trong nhóm, Scrum Master của nhóm.
- Project Leader là người khởi tạo Sprint, một Sprint khi được tạo ra sẽ cho tất cả các Team trong dự án cùng thực hiện. Mỗi Sprint sẽ gồm: tên, mô tả, ngày bắt đầu, ngày kết thúc. Project Leader sẽ là người duy nhất có quyền phân công User Story cho nhóm trong Sprint.
- Sau khi các Team cùng Project Leader và Product Owner đã thống nhất trong Sprint hiện tại của nhóm có các Story nào rồi, nhóm sẽ thực hiện chia nhỏ Story thành các Task. Việc này hoàn toàn do nhóm thực hiện và không có bất cứ sự can thiệp nào khác. Mỗi Task sẽ gồm các mục sau: tên, mô tả, thời gian thực hiện, người thực hiện, trạng thái (cần làm, đang làm, cần kiểm thử, và hoàn thành), độ quan trọng và phân loại. Các Task này cùng trạng thái của

chúng sẽ được dùng để thể hiện trong bảng Taskboard. Mỗi Task chỉ phân công cho một người duy nhất thực hiện nhằm dễ dàng hơn cho việc quản lí.

- Công cụ lưu trữ tất cả các thay đổi tới dự án, bất kì một cập nhật nào đều được ghi lại (ví dụ: thêm mới hay sửa User Story, thêm Task hay xóa Task...) vào trong mục Activity nhằm đảm bảo trách nhiệm của các thành viên thực hiện thay đổi trên công cụ.
- Công cụ cung cấp tính năng bình luận, người dùng có thể bình luận khi xem chi tiết dự án, Sprint, Task...

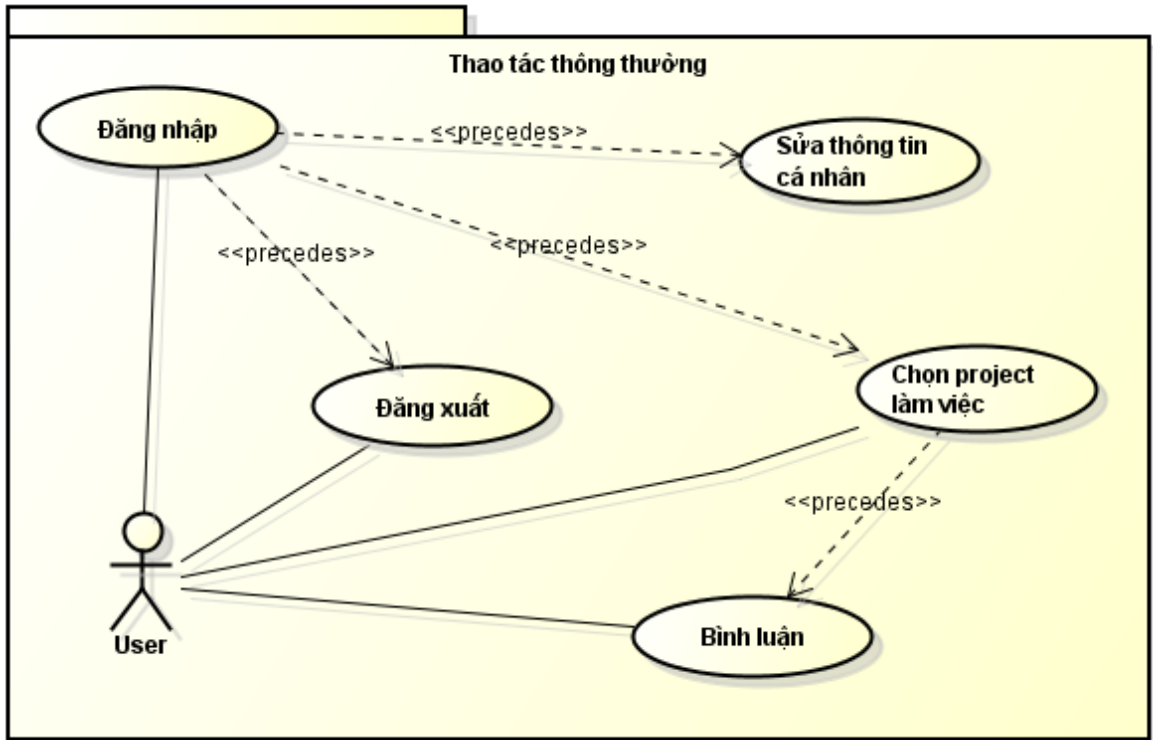
3.2. Sơ đồ Use Case

3.2.1. Sơ đồ Use Case tổng thể



Hình 3.1: Use Case tổng thể

3.2.2. Thao tác thông thường



Hình 3.2: Use Case các thao tác thông thường

3.2.2.1. Use Case đăng nhập

- Actor: Người dùng chưa đăng nhập.
- Người dùng truy cập vào hệ thống.
 - Dòng cơ bản: hệ thống hiển thị màn hình Login, người dùng nhập thông tin đăng nhập (tên đăng nhập, mật khẩu). Sau đó, người dùng nhấp vào nút Login; hệ thống kiểm tra thông tin đăng nhập; nếu thông tin hợp lệ, hệ thống hiển thị trang Project.
 - Dòng thay thế:
 - Người dùng nhập thiếu thông tin: hệ thống hiển thị thông báo cảnh báo.
 - Thông tin đăng nhập không hợp lệ: hệ thống thông báo lỗi cho người dùng.

3.2.2.2. Use Case chọn dự án làm việc

- Actor: người dùng đã đăng nhập.
- Tiền điều kiện: người dùng đã đăng nhập thành công vào hệ thống.
- Người dùng đăng nhập thành công và hệ thống chuyển người dùng qua trang quản lý Project. Người dùng nhấn vào nút Set current vào dự án mà họ muốn làm việc trên đó.
 - Dòng cơ bản: hệ thống thiết lập dự án được chọn làm dự án mặc định cho người dùng đó.
 - Dòng thay thế:
 - Có lỗi trong quá trình thiết lập: hệ thống hiển thị thông báo cảnh báo.

3.2.2.3. Use Case đăng xuất

- Actor: người dùng đã đăng nhập.
- Tiền điều kiện: người dùng đã đăng nhập thành công vào hệ thống.
- Người dùng nhấp vào đường dẫn Log out.
 - Dòng cơ bản: người dùng thoát khỏi hệ thống, hệ thống hiển thị trang Login.

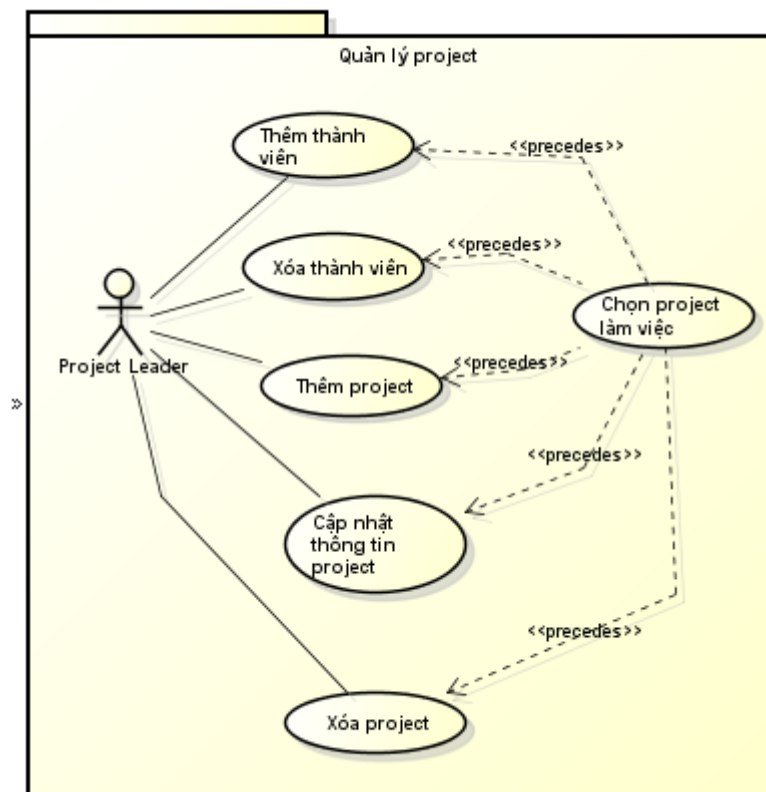
3.2.2.4. Use Case bình luận

- Actor: người dùng đã đăng nhập.
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Người dùng nhấp chọn tab Comment khi xem chi tiết Task, User Story, sprint, dự án.
 - Dòng cơ bản: hệ thống hiển thị khung bình luận cho người dùng, người dùng nhập nội dung bình luận rồi nhấn nút Comment, hệ thống cập nhật và hiển thị nội dung bình luận, đồng thời lưu Activity bình luận.
 - Dòng thay thế:
 - Nếu người dùng không nhập nội dung mà nhấn bình luận: hệ thống thông báo lỗi cho người dùng.

3.2.2.5. Use Case sửa thông tin cá nhân

- Actor: người dùng đã đăng nhập.
- Tiền điều kiện: người dùng đã đăng nhập thành công vào hệ thống.
- Người dùng chọn menu sửa thông tin cá nhân.
 - Dòng cơ bản: hệ thống hiển thị trang Profile. Người dùng tiến hành thay đổi thông tin cá nhân rồi nhấn nút Save. Hệ thống kiểm tra thông tin nhập liệu của người dùng; nếu thông tin hợp lệ hệ thống lưu thông tin vào cơ sở dữ liệu.
 - Dòng thay thế:
 - Nếu thông tin nhập liệu của người dùng không hợp lệ: hệ thống thông báo lỗi.

3.2.3. Quản lý dự án



Hình 3.3: Use Case quản lý Project

3.2.3.1. Use Case thêm thành viên

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader nhấn vào đường dẫn People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực. Project Leader nhấn Add user tại trang quản lý nhân lực.
 - Dòng cơ bản: hệ thống hiển thị khung thêm thành viên. Project Leader nhập các thông tin của thành viên (tên, tuổi, email...) rồi nhấn Add user tại khung thêm thành viên. Hệ thống kiểm tra thông tin nhập liệu, nếu thông tin phù hợp, hệ thống tạo thành viên mới.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.3.2. Use Case xóa thành viên

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader vào trang quản lý nhân lực cho dự án. Project Leader nhấn vào tên User sau đó nhấn vào Delete. Hệ thống hiển thị thông báo xác nhận.
 - Dòng cơ bản: actor đồng ý xác nhận hệ thống thực hiện xóa thành viên.
 - Dòng thay thế:
 - Actor hủy xác nhận: hệ thống không thực hiện cập nhật.

3.2.3.3. Use Case thêm dự án

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader nhấn vào menu Project. Hệ thống hiển thị trang quản lý dự án. Project Leader nhấn Add Project trên trang quản lý dự án.

- Dòng cơ bản: hệ thống hiển thị khung thêm dự án mới. Project Leader nhập thông tin cho dự án rồi nhấn Add project trong khung thêm Project. Hệ thống kiểm tra thông tin nhập liệu; nếu thông tin hợp lệ, hệ thống tạo dự án mới.
- Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

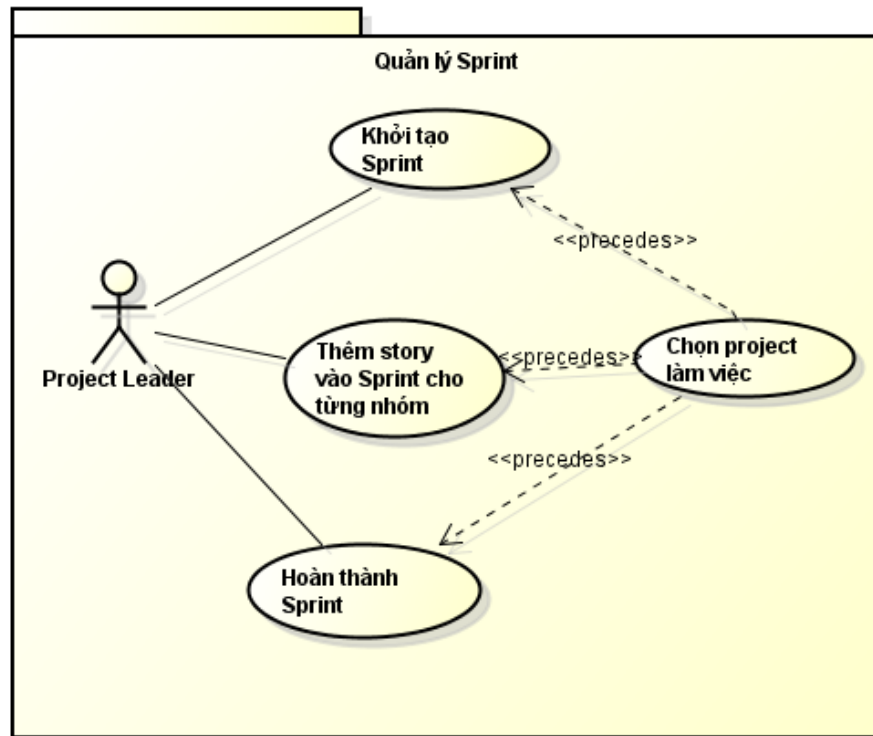
3.2.3.4. Use Case cập nhật thông tin dự án

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader vào trang quản lí dự án, hệ thống hiển thị danh sách dự án. Project Leader chọn sửa một dự án.
 - Dòng cơ bản: hệ thống hiển thị khung sửa dự án. Project Leader tiến hành thay đổi thông tin của dự án rồi nhấn nút Save project information. Hệ thống tiến hành kiểm tra thông tin nhập liệu; nếu thông tin hợp lệ, hệ thống cập nhật dự án.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.3.5. Use Case cập nhật thông tin dự án

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader vào trang quản lí dự án, hệ thống hiển thị danh sách dự án. Project Leader chọn sửa một dự án. Hệ thống hiển thị khung sửa dự án. Project Leader nhấp vào Delete.
 - Dòng cơ bản: hệ thống cập nhật trạng thái và ngày kết thúc của dự án.
 - Dòng thay thế:

3.2.4. Quản lý Sprint



Hình 3.4: Use Case quản lý Sprint

3.2.4.1. Use Case khởi tạo Sprint

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu Sprint trên thanh menu chính. Hệ thống hiển thị trang quản lý Sprint cho dự án. Project Leader nhấn Add sprint.
 - Dòng cơ bản: hệ thống hiển thị khung tạo Sprint. Project Leader tiến hành nhập liệu các thông cần thiết của Sprint sau đó nhấn Add sprint. Hệ thống kiểm tra thông tin nhập liệu; nếu thông tin hợp lệ, hệ thống tạo Sprint mới.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.4.2. Use Case thêm Story vào Sprint

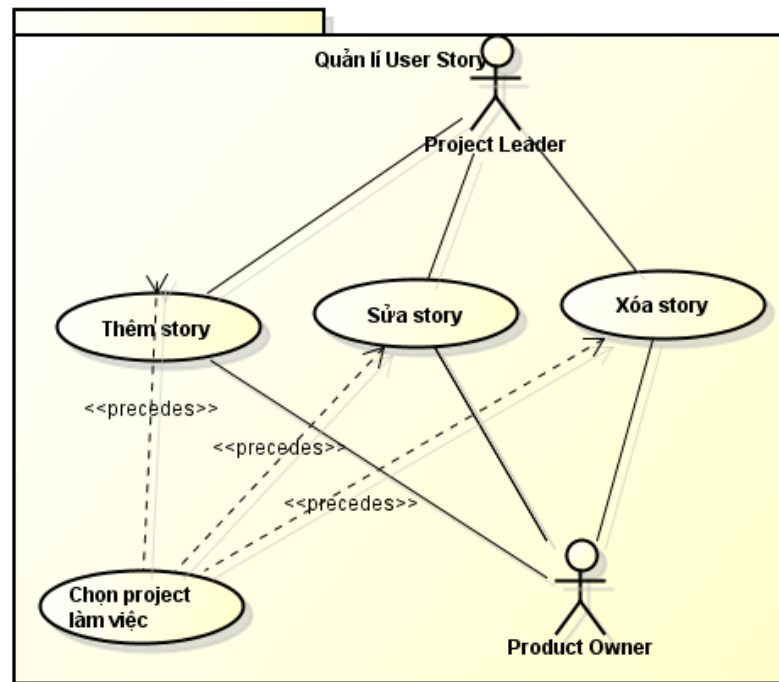
- Actor: Project Leader

- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu Sprint trên thanh menu chính. Hệ thống hiển thị trang quản lí Sprint cho dự án. Project Leader chọn User Story muốn đưa vào Sprint, sau đó kéo thả User Story vào danh sách User Story của nhóm sẽ thực hiện
 - Dòng cơ bản: hệ thống cập nhật danh sách User Story cho nhóm. Nếu tổng số ngày ước tính của các User Story được phân công cho một Team đã vượt quá số ngày làm của Team, hệ thống sẽ xuất hiện cảnh báo.
 - Dòng thay thế:

3.2.4.3. Use Case hoàn thành Sprint

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu Sprint trên thanh menu chính. Hệ thống hiển thị trang quản lí Sprint cho dự án. Project Leader nhấp vào nút Complete Sprint. Hệ thống hiển thị thông báo xác nhận. Nếu trong Sprint còn tồn tại các Story chưa hoàn thành, hệ thống hiển thị thông báo danh sách tên các Story đó.
 - Dòng cơ bản: nếu người dùng đồng ý, hệ thống cập nhật trạng thái của Sprint thành hoàn thành và những Story chưa hoàn thành sẽ được chuyển lại vào danh sách Story để thêm vào các Sprint khác.
 - Dòng thay thế: nếu người dùng không đồng ý thay đổi, hệ thống không thực hiện cập nhật trên dữ liệu và hiển thị trang Sprint.

3.2.5. Quản lý User Story



Hình 3.5: Use Case quản lý User Story

3.2.5.1. Thêm Story

- Actor: Product Owner, Project Leader
- Tiên điều kiện: actor đã thiết lập dự án làm việc.
- Product Owner vào trang User story, sau đó chọn Add story.
 - Dòng cơ bản: hệ thống hiển thị khung thêm User Story. Actor nhập thông tin của User Story; hệ thống kiểm tra thông tin; nếu thông tin hợp lệ hệ thống lưu User Story.
 - Dòng thay thế:
 - Thông tin nhập liệu cho User Story không hợp lệ: hệ thống thông báo lỗi.

3.2.5.2. Sửa Story

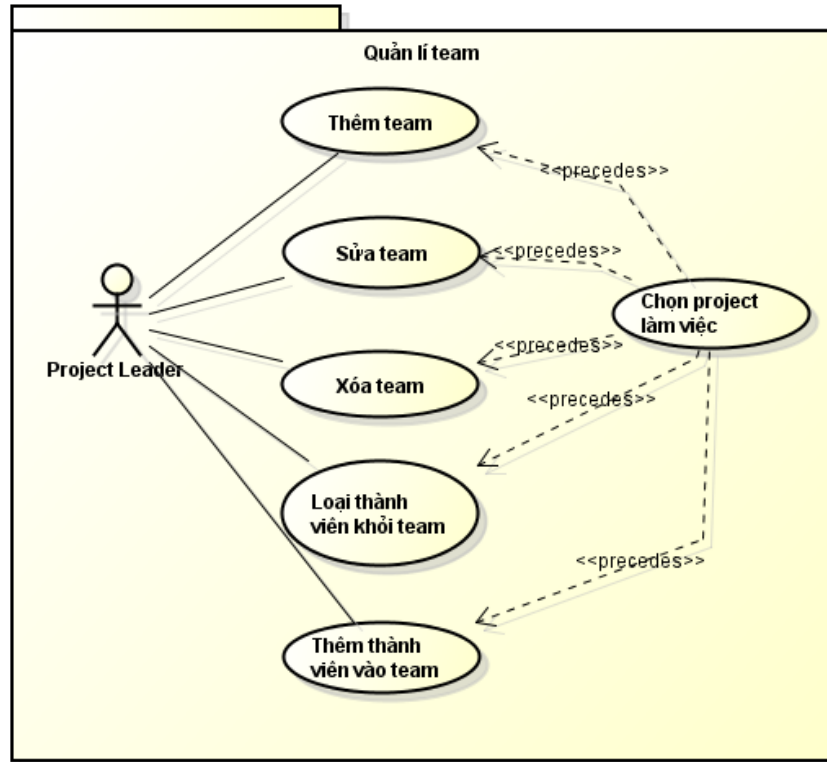
- Actor: Product Owner, Project Leader
- Tiên điều kiện: actor đã thiết lập dự án làm việc.

- Actor vào trang User Story chọn sửa một Story
 - Dòng cơ bản: hệ thống hiển thị khung sửa User Story. Actor tiến hành nhập liệu, sau đó nhấn Save story. Hệ thống kiểm tra dữ liệu nhập vào; nếu dữ liệu hợp lệ hệ thống cập nhật User Story.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.5.3. Xóa Story

- Actor: Product Owner, Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Actor vào trang User Story chọn sửa một Story. Hệ thống hiển thị khung chi tiết Story, actor nhấn Delete, hệ thống xuất hiện thông báo xác nhận.
 - Dòng cơ bản: actor xác nhận, hệ thống tiến hành xóa Story khỏi dự án.
 - Dòng thay thế:
 - Actor hủy xác nhận: hệ thống không thực hiện cập nhật.

3.2.6. Quản lý Team



Hình 3.6: Use Case quản lý Team

3.2.6.1. Thêm Team

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực cho dự án. Project Leader nhấn Add team.
 - Dòng cơ bản: hệ thống hiển thị khung tạo nhóm, Project Leader nhập thông tin nhóm sau đó nhấn Add team. Hệ thống kiểm tra thông tin nhập liệu; nếu dữ liệu hợp lệ, hệ thống tạo Team mới.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.6.2. Sửa Team

- Actor: Project Leader

- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực cho dự án. Project chọn xem chi tiết Team bằng cách nhấp vào tên của Team. Hệ thống hiển thị khung Edit Team; Project Leader sửa đổi thông tin và nhấn Save team.
 - Dòng cơ bản: hệ thống cập nhật thông tin về Team.
 - Dòng thay thế:

3.2.6.3. Xóa Team

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực cho dự án. Project Leader chọn xem chi tiết Team bằng cách nhấp vào tên của Team. Hệ thống hiển thị khung Edit Team, Project Leader nhấn Delete.
 - Dòng cơ bản: hệ thống hiển thị thông báo xác nhận; nếu Project Leader chọn Đồng ý, hệ thống xóa Team khỏi dự án.
 - Dòng thay thế:

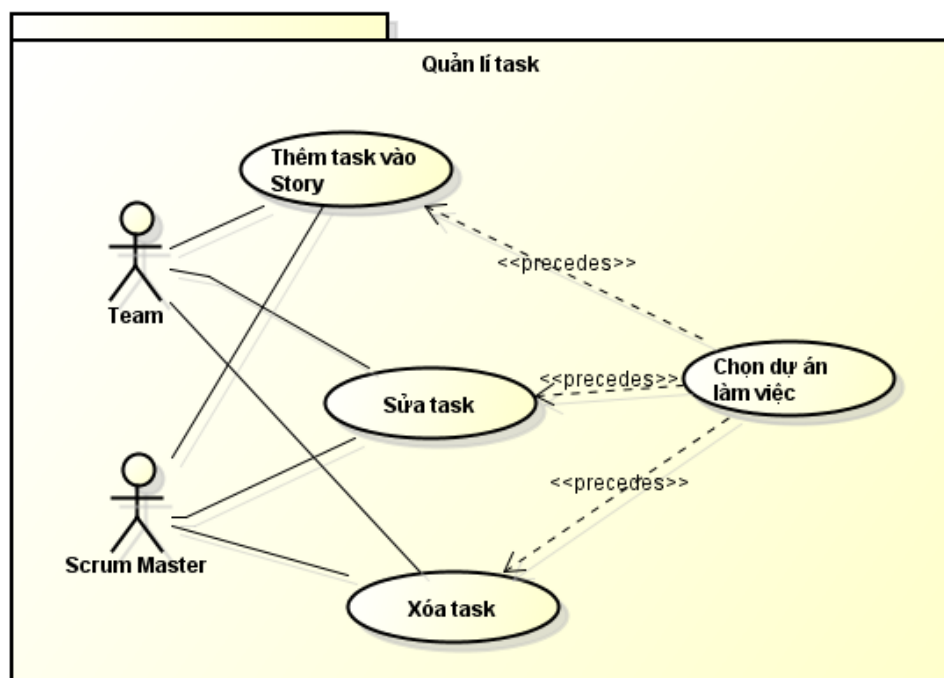
3.2.6.4. Loại thành viên khỏi Team

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực cho dự án. Project Leader chọn thành viên muốn xóa khỏi nhóm và kéo thả khỏi danh sách nhân lực trong Team.
 - Dòng cơ bản: hệ thống xóa thành viên khỏi Team.
 - Dòng thay thế:
 - Hệ thống gặp lỗi khi thực hiện: xuất hiện thông báo cho người dùng.

3.2.6.5. Thêm thành viên vào Team

- Actor: Project Leader
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Project Leader chọn menu People trên thanh menu chính. Hệ thống hiển thị trang quản lý nhân lực cho dự án. Project Leader kéo thả thành viên dự án từ khung danh sách thành viên hoặc từ Team khác vào Team mong muốn.
 - Dòng cơ bản: hệ thống cập nhật danh sách nhóm.
 - Dòng thay thế:
 - Hệ thống gặp lỗi khi thực hiện: xuất hiện thông báo cho người dùng.

3.2.7. Quản lý Task



Hình 3.7: Use Case quản lý Task

3.2.7.1. Thêm Task vào User Story

- Actor: Scrum Master, Team
- Tiền điều kiện: actor đã thiết lập dự án làm việc.
- Actor xem chi tiết một Story, sau đó chọn tab Tasks rồi nhấn Add task.

- Dòng cơ bản: hệ thống hiển thị khung thêm Task cho User Story được chọn. Scrum Master nhập dữ liệu và nhấn Add task; hệ thống kiểm tra thông tin nhập liệu; nếu thông tin hợp lệ, hệ thống cập nhật danh sách Task của Story.
- Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.7.2. Sửa Task

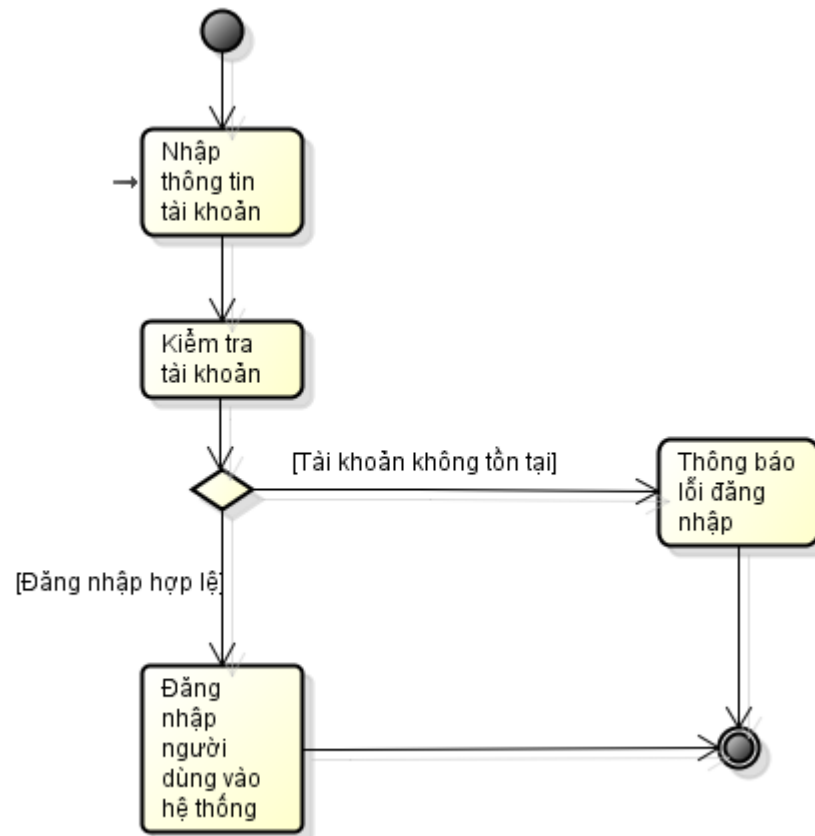
- Actor: Scrum Master, Team
- Actor chọn xem chi tiết Task (có thể từ danh sách Task của User Story hoặc từ trang Taskboard).
 - Dòng cơ bản: hệ thống hiển thị khung chi tiết Task. Scrum Master nhập dữ liệu và nhấn Save task; hệ thống kiểm tra thông tin nhập liệu; nếu thông tin hợp lệ hệ thống cập nhật danh sách Task của User Story.
 - Dòng thay thế:
 - Thông tin nhập liệu không hợp lệ: hệ thống thông báo lỗi.

3.2.7.3. Xóa Task

- Actor: Scrum Master, Team
- Tiên điều kiện: actor đã thiết lập dự án làm việc.
- Scrum Master chọn xem chi tiết một Task rồi nhấn vào nút Delete.
 - Dòng cơ bản: hệ thống hiển thị thông báo xác nhận; nếu Project Leader đồng ý hệ thống thực hiện xóa Task.
 - Dòng thay thế:
 - Nếu Project Leader hủy xác nhận: hệ thống không thực hiện cập nhật.

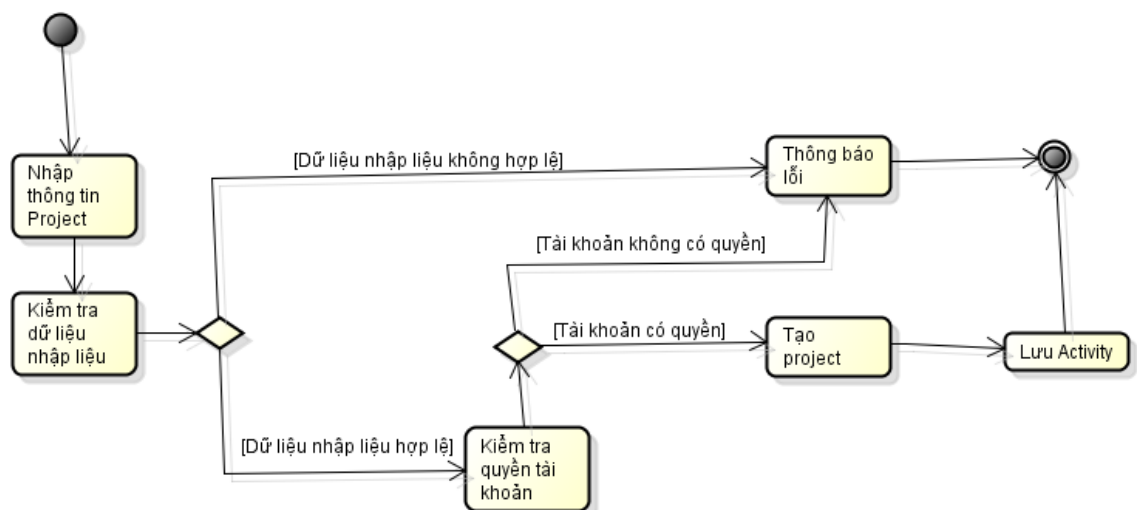
3.3. Sơ đồ Activity

3.3.1. Chức năng đăng nhập



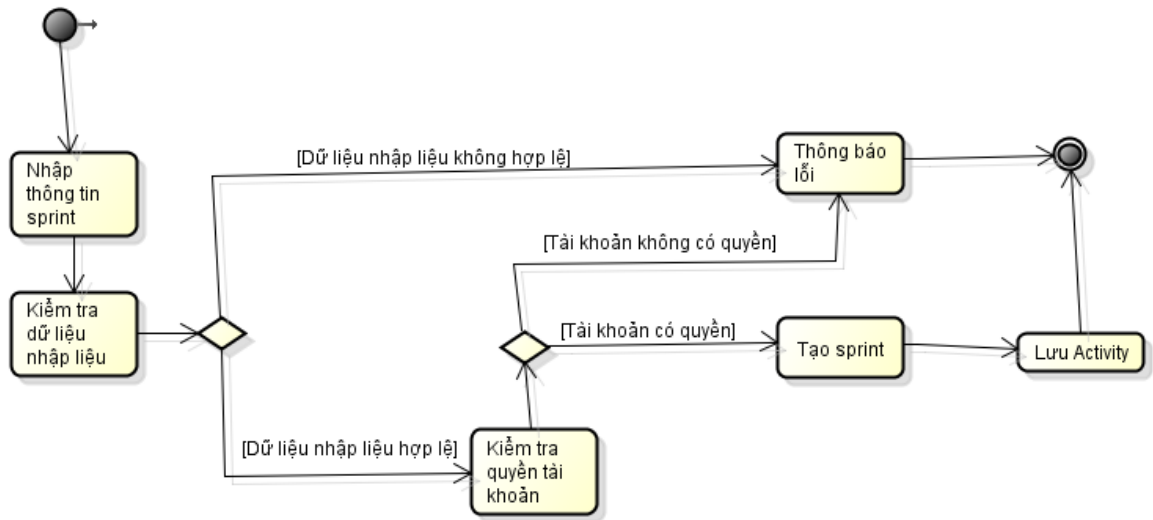
Hình 3.8: Activity diagram đăng nhập

3.3.2. Chức năng tạo dự án



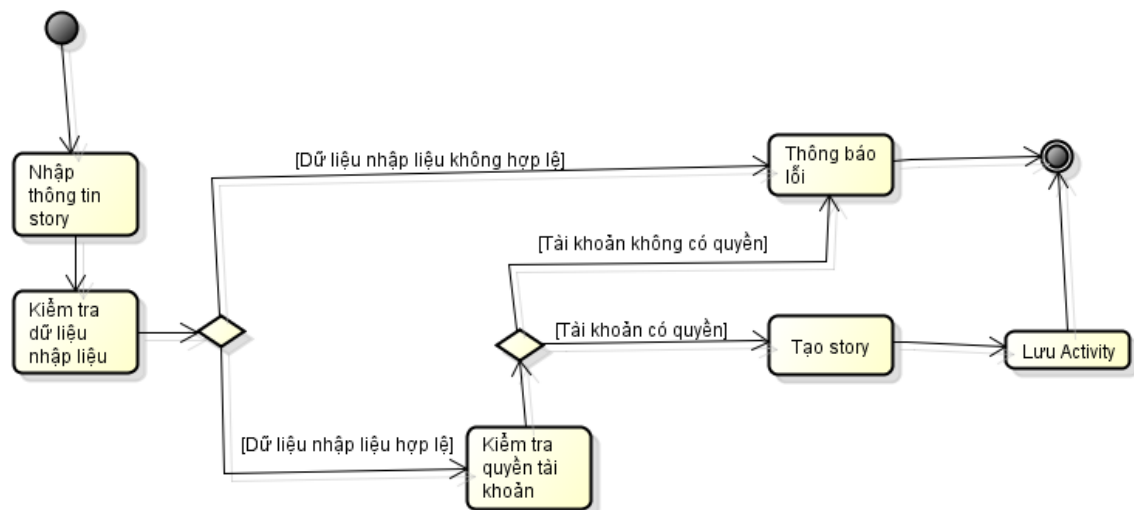
Hình 3.9: Activity diagram tạo dự án

3.3.3. Chức năng tạo Sprint



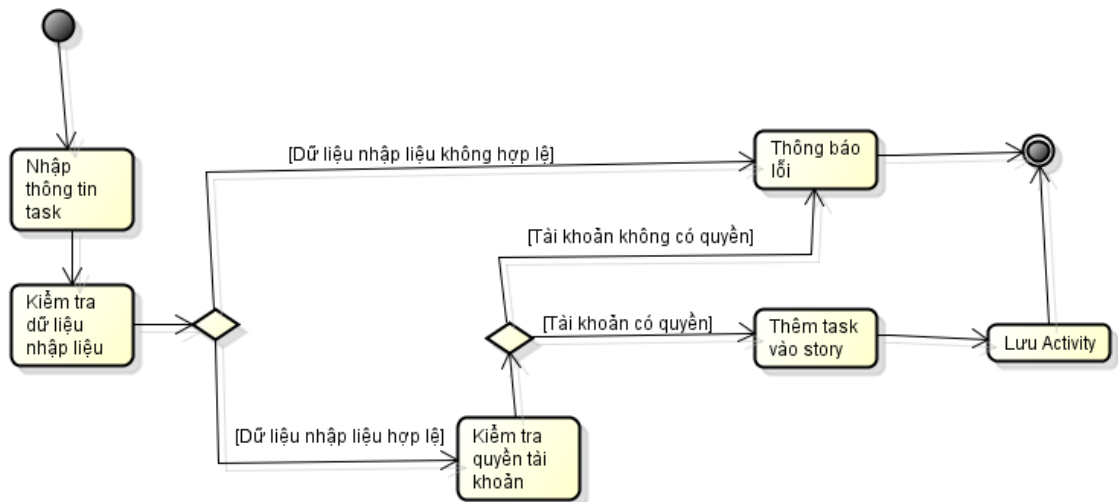
Hình 3.10: Activity diagram tạo Sprint

3.3.4. Chức năng tạo User Story



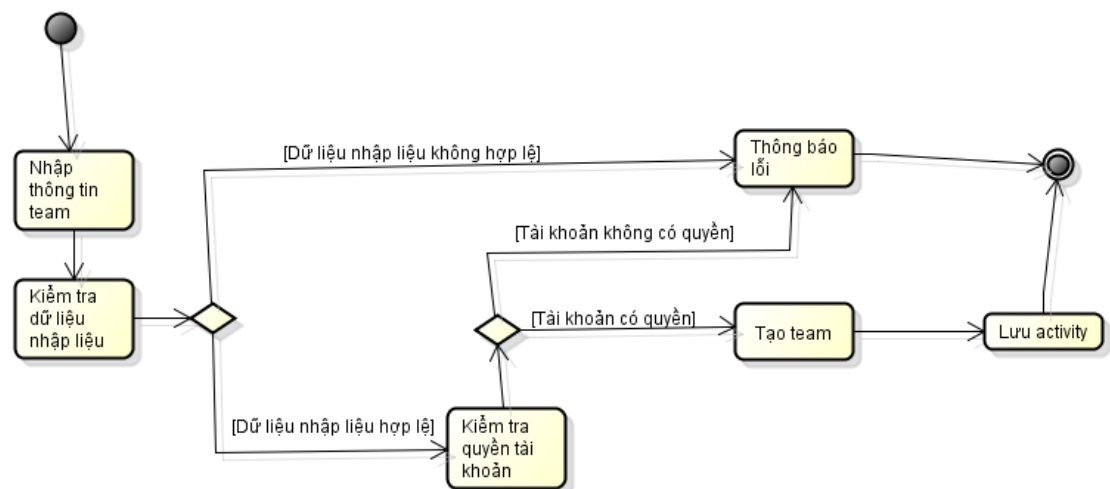
Hình 3.11: Activity diagram tạo User Story

3.3.5. Chức năng tạo Task



Hình 3.12: Activity diagram tạo Task

3.3.6. Chức năng tạo Team



Hình 3.13: Activity diagram tạo Team

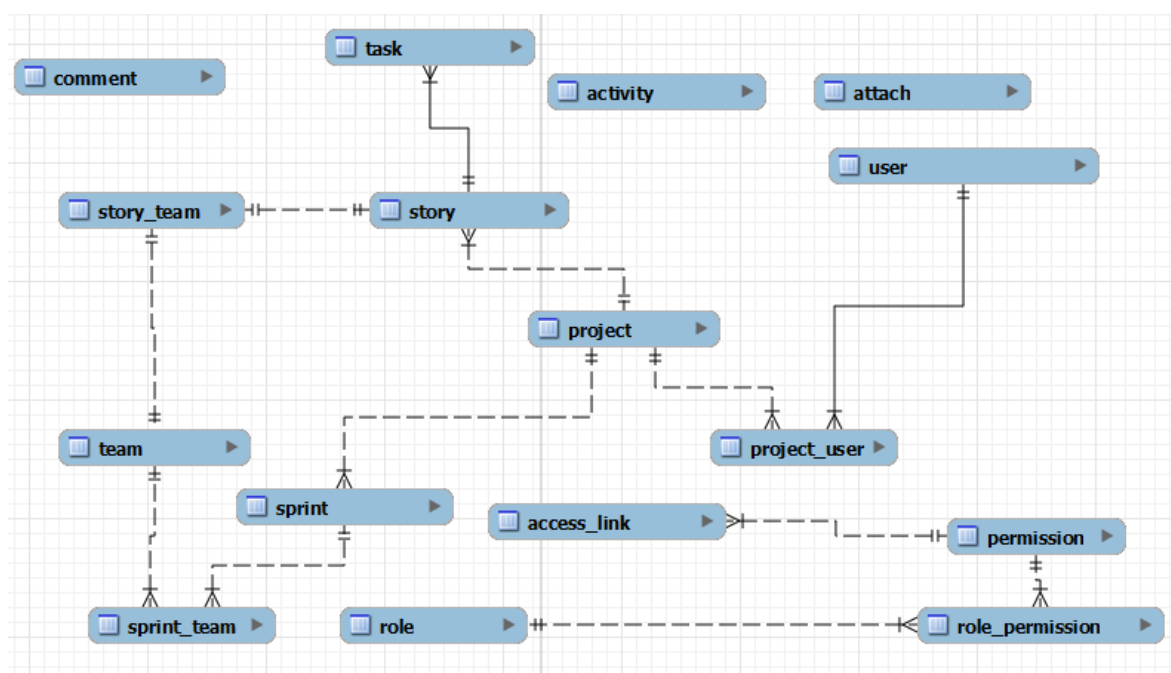
3.4. Tổ chức cơ sở dữ liệu

3.4.1. Danh sách các bảng trong cơ sở dữ liệu

Tên bảng	Ý nghĩa
access_link	Chứa danh sách tất cả các đường dẫn trong dự án, dùng để quản lý phân quyền.
activity	Lưu trữ tất cả các thao tác đến các thực thể (dự án, Sprint, User Story, Task).
attach	Lưu thông tin file đính kèm.
comment	Lưu nội dung các bình luận.
permission	Lưu các quyền của người dùng.
project	Lưu thông tin của dự án.
project_user	Lưu các thành viên tham gia dự án cùng với vai trò của từng thành viên.
role	Lưu các vai trò trong đội ngũ phát triển
role_permission	Lưu các quyền của từng vai trò.
sprint	Lưu trữ thông tin của các Sprint
sprint_team	Lưu số ngày làm việc của từng Team trong mỗi Sprint.
story	Lưu trữ thông tin User Story
story_team	Lưu trữ User Story được phân công cho Team nào trong Sprint nào.
task	Lưu thông tin về các Task.
team	Lưu thông tin về các Team tham gia dự án.
user	Lưu thông tin về các thành viên trong hệ thống.

Bảng 3.1: Danh sách các bảng trong cơ sở dữ liệu

Sơ đồ ERD



Hình 3.14: Sơ đồ ERD

3.4.2. Bảng access_link

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
lid	int	11	Not null	Mã đường dẫn
path	varchar	255	Not null	Đường dẫn
type	tinyint	4	Not null	Loại đường dẫn (0: thông thường, 1: regular expression)
peid	int	11	Not null	Mã quyền

Bảng 3.2: Chi tiết bảng access_link

3.4.3. Bảng activity

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
acid	bigint	20	Not null	Mã hoạt động

type	int	11	Not null	Loại hoạt động (thêm, sửa, xóa).
entity_id	int	11	Not null	Mã thực thể gắn hoạt động
entity_type	varchar	45	Not null	Loại thực thể gắn hoạt động
content	text	0	Not null	Nội dung hoạt động
time	timestamp	0	Not null	Thời gian xảy ra hoạt động
uid	bigint	11	Not null	Mã người dùng thực hiện hoạt động

Bảng 3.3: Chi tiết bảng activity

3.4.4. Bảng attach

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
atid	bigint	20	Not null	Mã đính kèm
file_name	varchar	610	Not null	Tên tập tin đính kèm
entity_id	int	11	Not null	Mã thực thể gắn đính kèm
entity_type	int	11	Not null	Loại thực thể gắn đính kèm
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.4: Chi tiết bảng sttach

3.4.5. Bảng comment

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
cid	bigint	20	Not null	Mã bình luận
uid	int	11	Not null	Mã người dùng bình luận
entity_id	int	11	Not null	Mã thực thể gắn bình luận
entity_type	tinyint	11	Not null	Loại thực thể gắn bình luận
content	text	0	Not null	Nội dung bình luận
parent_id	bigint	20	Not null	Mã bình luận cha
time	timestamp	0	Not null	Thời gian bình luận

Bảng 3.5: Chi tiết bảng comment

3.4.6. Bảng permission

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
peid	bigint	20	Not null	Mã quyền
name	varchar	250	Not null	Tên quyền

Bảng 3.6: Chi tiết bảng permission

3.4.7. Bảng dự án

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
pid	int	11	Not null	Mã dự án
name	varchar	450	Not null	Tên dự án

start_date	date	0	Allow null	Ngày bắt đầu
end_date_es	date	0	Allow null	Ngày kết thúc dự kiến
end_date_real	date	0	Allow null	Ngày kết thúc thực tế
note	text	0	Allow null	Ghi chú
status	int	11	Allow null	Trạng thái dự án
description	text	0	Allow null	Mô tả về dự án
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.7: Chi tiết bảng project

3.4.8. Bảng project_user

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
pid	int	11	Not null	Mã dự án
uid	int	11	Not null	Mã người dùng
tid	int	11	Not null	Mã Team
rid	int	11	Not null	Mã vai trò

Bảng 3.8: Chi tiết bảng project_user

3.4.9. Bảng role

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
rid	int	11	Not null	Mã vai trò
name	varchar	100	Allow null	Tên vai trò

Bảng 3.9: Chi tiết bảng role

3.4.10. Bảng role_permission

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
rid	int	11	Not null	Mã vai trò
peid	bigint	20	Not null	Mã quyền

Bảng 3.10: Chi tiết bảng role_permission

3.4.11. Bảng sprint

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
spid	int	11	Not null	Mã Sprint
pid	int	11	Not null	Mã quyền
name	varchar	200	Not null	Tên Sprint
description	varchar	45	Allow null	Mô tả Sprint
start_date	date	0	Not null	Ngày bắt đầu Sprint thực sự
end_date	date	0	Not null	Ngày kết thúc Sprint thực sự
status	int	11	Not null	Trạng thái của Sprint
start_date_es	date	0	Not null	Ngày bắt đầu dự kiến
end_date_es	date	0	Not null	Ngày kết thúc dự kiến
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.11: Chi tiết bảng sprint

3.4.12. Bảng sprint_team

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
spid	bigint	20	Not null	Mã Sprint
tid	bigint	20	Not null	Mã Team
num_day	tinyint	4	Not null	Số ngày làm việc của Team trong Sprint

Bảng 3.12: Chi tiết bảng sprint_team

3.4.13. Bảng story

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
sid	bigint	11	Not null	Mã User Story
pid	int	11	Not null	Mã dự án chứa User Story
create_user	int	11	Not null	Người tạo User Story
name	varchar	500	Not null	Tên User Story
priority	int	11	Not null	Độ ưu tiên của User Story
time_estimate	int	11	Allow null	Thời gian ước lượng cần thực hiện User Story
point	int	11	Allow null	Điểm của User Story
demo	text	0	Not null	Cách demo User Story
description	text	0	Allow null	Mô tả User Story
status	int	11	Not null	Trạng thái của User

				Story
finish_date	datetime	0	Allow null	Ngày hoàn thành User Story
create_date	datetime	0	Allow null	Ngày tạo User Story
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.13: Chi tiết bảng story

3.4.14. Bảng story_team

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
sid	bigint	20	Not null	Mã User Story
tid	int	11	Not null	Mã Team
spid	int	11	Not null	Mã Sprint
order	tinyint	4	Not null	Thứ tự của User Story

Bảng 3.14: Chi tiết bảng story_team

3.4.15. Bảng task

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
taid	bigint	11	Not null	Mã Task
name	varchar	450	Not null	Tên Task
time_estimate	float	0	Not null	Thời gian ước tính cần thực hiện Task
time_remain	float	0	Allow null	Thời gian còn lại để

				thực hiện Task
status	int	11	Not null	Trạng thái của Task
create_date	timestamp	0	Not null	Ngày tạo Task
create_user	int	11	Not null	Người tạo Task
description	text	0	Allow null	Mô tả Task
uid	int	11	Allow null	Người phân công thực hiện Task
sid	bigint	20	Allow null	Story chứa Task
finish_date	datetime	0	Allow null	Ngày Task hoàn thành
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.15: Chi tiết bảng task

3.4.16. Bảng team

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
tid	int	11	Not null	Mã Team
name	varchar	500	Not null	Tên Team
pid	int	11	Not null	Mã dự án
description	tinytext	0	Allow null	Mô tả về Team
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

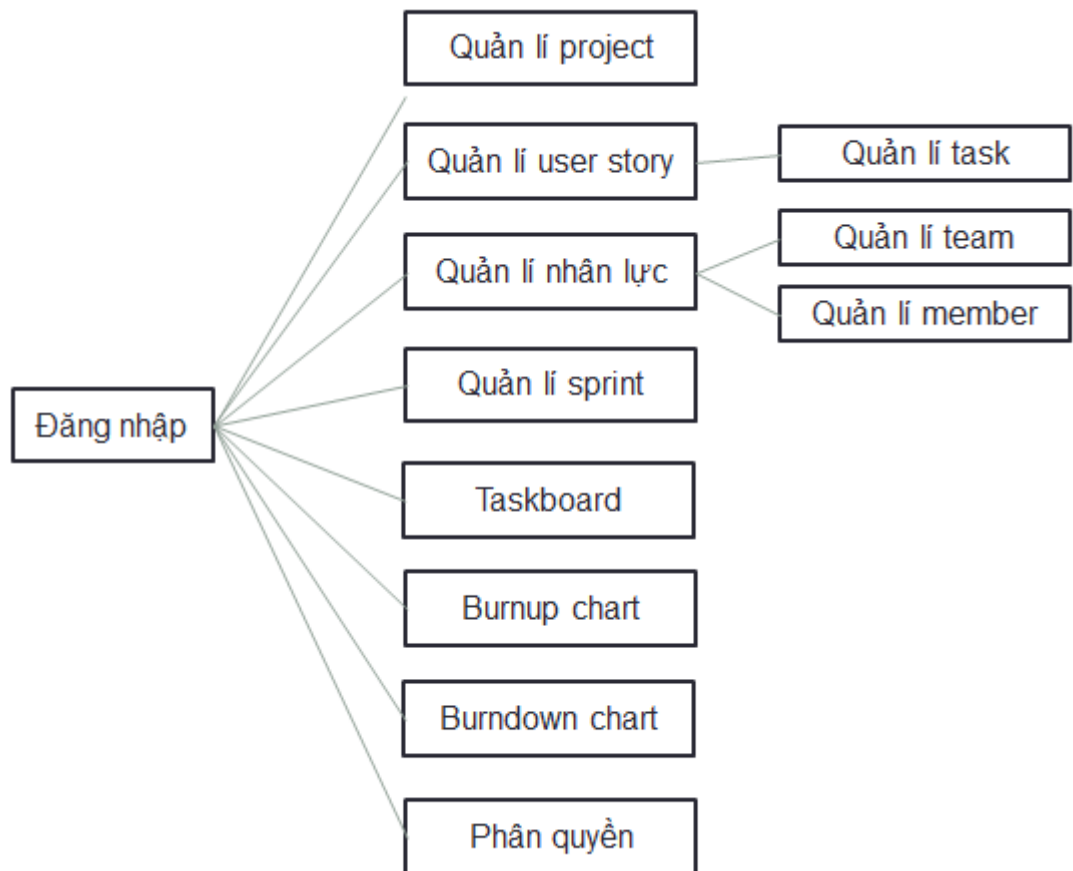
Bảng 3.16: Chi tiết bảng team

3.4.17. Bảng user

Tên thuộc tính	Loại	Độ dài	Ghi chú	Ý nghĩa
uid	int	11	Not null	Mã người dùng
login_nm	varchar	45	Not null	Tên đăng nhập
password	varchar	60	Not null	Mật khẩu
fullname	varchar	500	Allow null	Tên đầy đủ
birthday	date	0	Not null	Ngày sinh
image	varchar	100	Allow null	Hình đại diện
timezone	varchar	250	Allow null	Timezone
remember_token	varchar	100	Allow null	Ghi nhớ đăng nhập
lang	varchar	10	Not null	Ngôn ngữ
country	varchar	255	Allow null	Đất nước
delete_flg	tinyint	4	Not null	Cờ quy định xóa record hay không (0: không xóa, 1: record đã xóa)

Bảng 3.17: Chi tiết bảng user

3.5. Sơ đồ thiết kế giao diện



Hình 3.15: Sơ đồ thiết kế giao diện

Chương 4. TRIỂN KHAI ỨNG DỤNG

4.1. Cài đặt

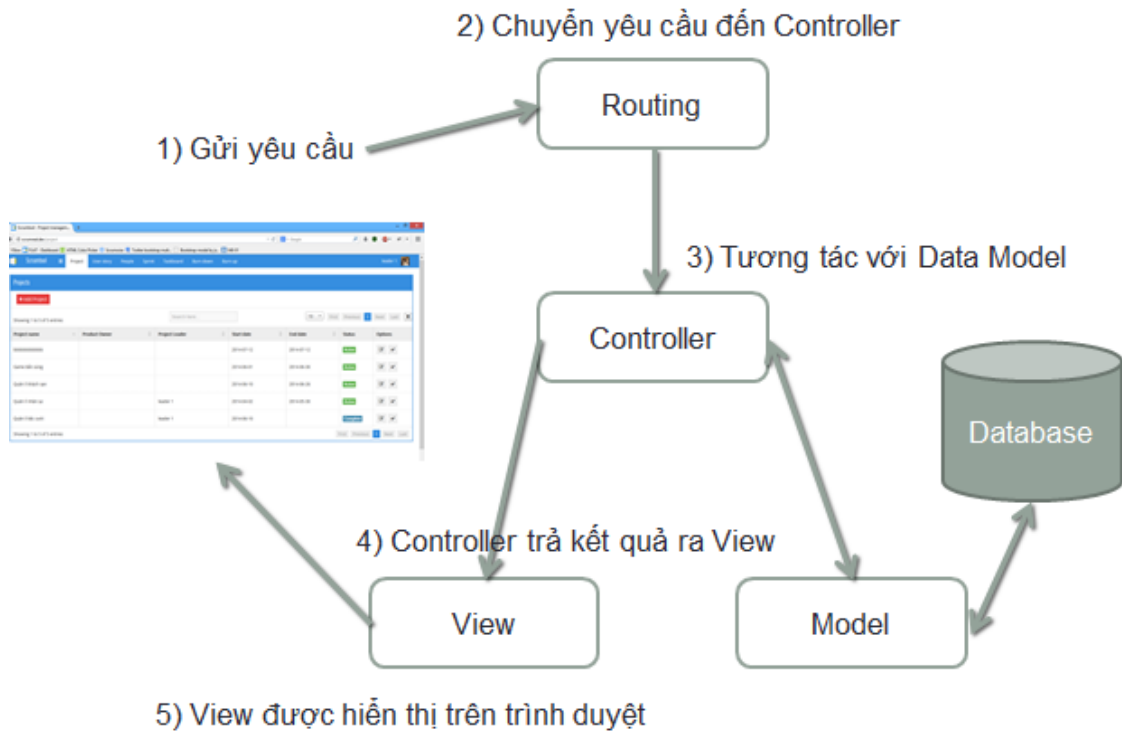
- Công cụ phát triển: NetBean 8.0
- Hệ quản trị cơ sở dữ liệu: MySql 5.6
- Ngôn ngữ phát triển: PHP 5.4
- Web server: Apache 2.2
- Công nghệ sử dụng: HTML5 WebSocket
- Framework: Laravel

4.2. Kiến trúc hệ thống

Công cụ được phát triển trên framework Laravel. Tương tự như nhiều PHP framework khác Laravel tuân theo mô hình MVC. MVC là mô hình thiết kế giúp nhà phát triển phần mềm chia tách giữa việc xử lý logic và giao diện người dùng trong phần mềm. Có ba thành phần trong mô hình MVC, mỗi thành phần có nhiệm vụ riêng biệt và độc lập với các thành phần khác:

- Model: phạm vi mà ứng dụng của bạn được xây lên. Model không chỉ là dữ liệu, nó còn chứa tất cả các xử lý logic trên dữ liệu đó.
- View: đảm nhận việc hiển thị thông tin, tương tác với người dùng.
- Controller: là trung gian, liên kết view và model. Controller chịu trách nhiệm xử lý dữ liệu đầu vào, làm việc với model và quyết định hành động nào cần được thực thi như hiển thị một view hoặc chuyển sang trang khác.

Khi trình duyệt gửi yêu cầu, web server đón nhận yêu cầu và chuyển qua routing engine của Laravel. Laravel router nhận yêu cầu và chuyển qua controller thích hợp dựa vào URL của yêu cầu, controller thực thi yêu cầu. Trong một số trường hợp, controller sẽ lập tức trả lại một view và gửi lại cho trình duyệt. Nhưng thông thường, controller tương tác với model và model làm việc với database lấy dữ liệu trả về.



Hình 4.1: Mô hình MVC trong Laravel

4.3. HTML5 WebSocket

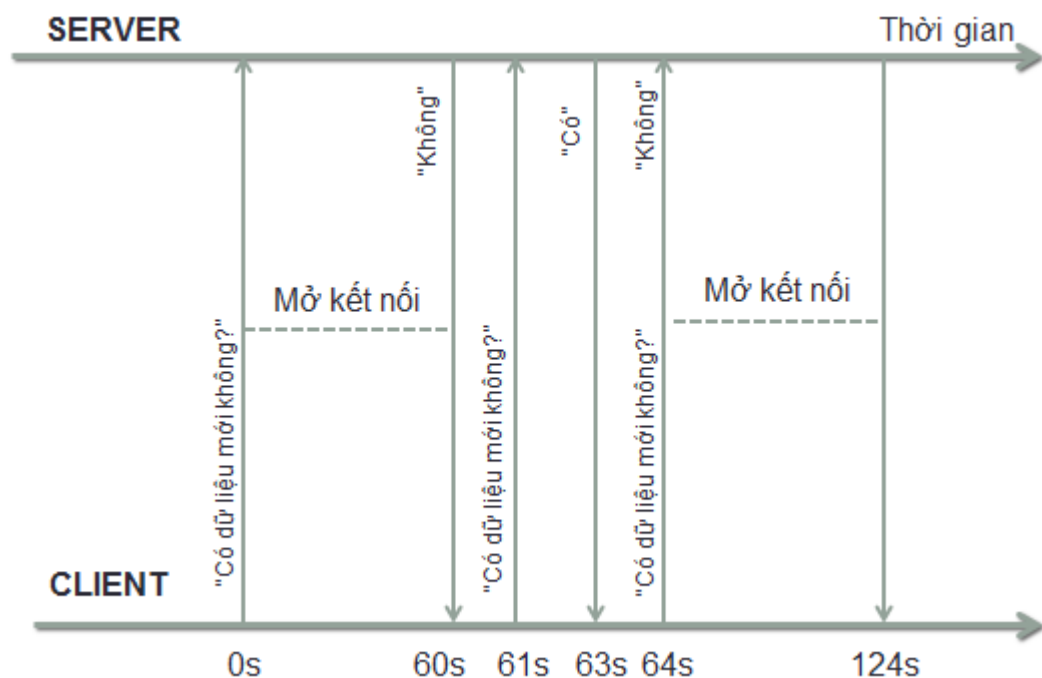
4.3.1. Kiến trúc truyền thông

Giao thức được mọi website sử dụng hiện nay để làm việc với web server là giao thức HTTP. Thiết kế của HTTP theo mô hình yêu cầu từ client gửi lên và server phản hồi yêu cầu đó, nếu client không gửi yêu cầu lên, sẽ không có bất cứ phản hồi nào từ server tới. Mô hình này là phù hợp ở thời kì đầu của công nghệ web khi dữ liệu trao đổi giữa client và server ít và cơ sở hạ tầng mạng còn chưa phát triển. Tuy nhiên, gần đây, các ứng dụng web yêu cầu nhiều thông tin được gửi từ phía server tới client, ví dụ như các ứng dụng chat, ứng dụng web thời gian thực... Lúc này, HTTP bộc lộ các yếu kém sau:

- Thiết kế ban đầu của HTTP là cho chia sẻ tài liệu, không phải là các ứng dụng tương tác như ứng dụng desktop.

- Lượng thông tin mà giao thức HTTP cần để giao tiếp giữa client và server tăng lên nhanh chóng khi mà càng có nhiều tương tác giữa hai bên.
- HTTP được thiết kế bán song công, có nghĩa là dữ liệu đi theo một chiều duy nhất tại một thời điểm: client gửi yêu cầu tới server (1 chiều) và server gửi phản hồi lại client (1 chiều).

Vì những vấn đề kể trên, nhiều công nghệ đã được phát triển như một giải pháp cru cánh như Ajax Polling, HTTP Polling, Long Polling. Mỗi giải pháp có cơ chế hiện thực khác nhau nhưng về bản chất đều gửi gói tin HTTP lên sever và chờ nhận gói tin HTTP trả lời (và để tránh phải gửi qua nhiều gói tin phía server thường giữ yêu cầu lại mà không phản hồi lại ngay cho đến khi có dữ liệu để gửi lại client hoặc hết thời gian gửi yêu cầu). Hình dưới đây minh hoạt hoạt động của Long Polling:



Hình 4.2: HTTP long polling giữ yêu cầu HTTP mở trong một khoảng thời gian để kiểm tra việc cập nhật

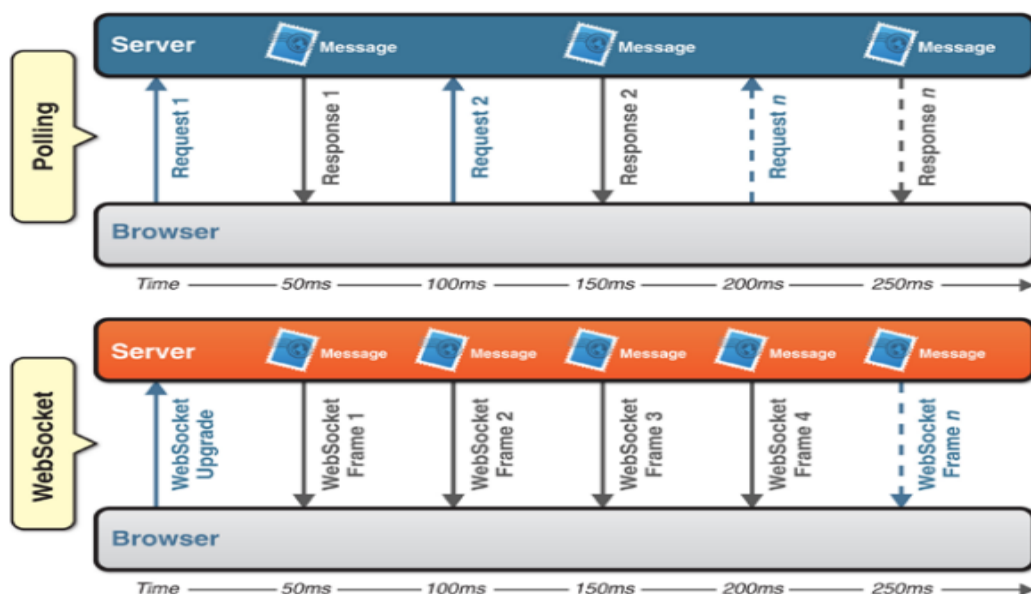
(Nguồn: sách *Real time webapps*)

Tất cả các giải pháp này có thể mang đến một ứng dụng web thời gian thực nhưng về bản chất vẫn sử dụng giao thức HTTP và có hai điểm yếu sau:

- Vì mỗi gói HTTP gửi đi đều kèm thêm HTTP header làm tăng lượng dữ liệu cũng như độ trễ khi giao tiếp giữa client và server.
- Đồng thời, vì vẫn sử dụng HTTP để giao tiếp nên client vẫn phải đợi gói HTTP trả về từ server mới tiếp tục gửi gói HTTP yêu cầu lên server, điều này làm tăng độ trễ cho ứng dụng.

4.3.2. Công nghệ WebSocket

Để khắc phục những vấn đề trên HTML5 đã được phát triển bao gồm thêm cả công nghệ WebSocket. WebSocket song công (full duplex), truyền dữ liệu theo hai hướng (bidirectional), và kết nối thông qua một socket duy nhất. WebSocket giảm độ trễ bởi vì một khi kết nối WebSocket được khởi tạo, server có thể gửi dữ liệu tới client ngay khi có dữ liệu mới. Khác với các phương pháp polling, WebSocket chỉ tạo một yêu cầu, server thì không cần phải đợi yêu cầu từ client để gửi dữ liệu, và client có thể gửi dữ liệu tới server bất cứ thời điểm nào. Việc chỉ dùng một yêu cầu giảm đáng kể độ trễ so với các phương pháp polling do các phương pháp này gửi yêu cầu tới server theo một chu kì cho dù dữ liệu cập nhật hay không.



Hình 4.3: So sánh WebSocket với các công nghệ polling
(Nguồn: sách *The Definite Guide to HTML5 WebSocket*)

4.3.3. Quy trình bắt tay kết nối của WebSocket

Mỗi kết nối WebSocket bắt đầu bằng một yêu cầu HTTP. Yêu cầu này cũng tương tự như nhưng yêu cầu HTTP khác, ngoại trừ việc nó có thêm header: *Upgrade*. Header này chỉ ra rằng nếu client muốn nâng cấp kết nối sang một giao thức khác. Khi kết nối thành công, server gửi lại client: mã phản hồi 101, *Upgrade* header và *Sec-WebSocket-Accept* header. Giá trị của *Sec-WebSocket-Accept* header được kế thừa từ *Sec-WebSocket-Key* header trong gói tin yêu cầu và chứa giá trị trả về phải bằng với giá trị mà client mong muốn.

Ví dụ yêu cầu HTTP từ client:

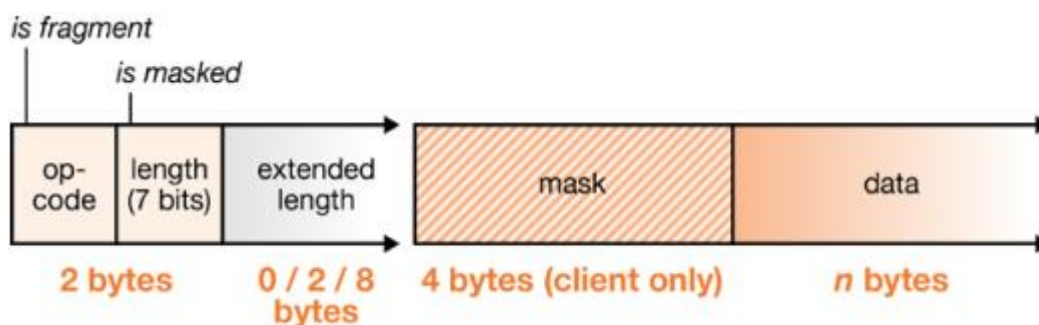
```
GET /echo HTTP/1.1
Host: echo.websocket.org
Origin: http://www.websocket.org
Sec-WebSocket-Key: 7+C600xYybOv2zmJ69RQsw==
Sec-WebSocket-Version: 13
Upgrade: websocket
```

Phản hồi từ server:

```
101 Switching Protocols
Connection: Upgrade
Date: Wed, 20 Jun 2012 03:39:49 GMT
Sec-WebSocket-Accept: fYoqiH14DgI+5ylEMwM2sOLzOi0=
Server: Kaazing Gateway
Upgrade: WebSocket
```

4.3.4. Định dạng gói tin

Khi kết nối WebSocket được mở, client và server có thể trao đổi các gói tin với nhau tại bất cứ thời điểm nào. Các gói tin này có header như sau:



Hình 4.4: Header của gói tin WebSocket

(Nguồn: sách *The Definite Guide to HTML5 WebSocket*)

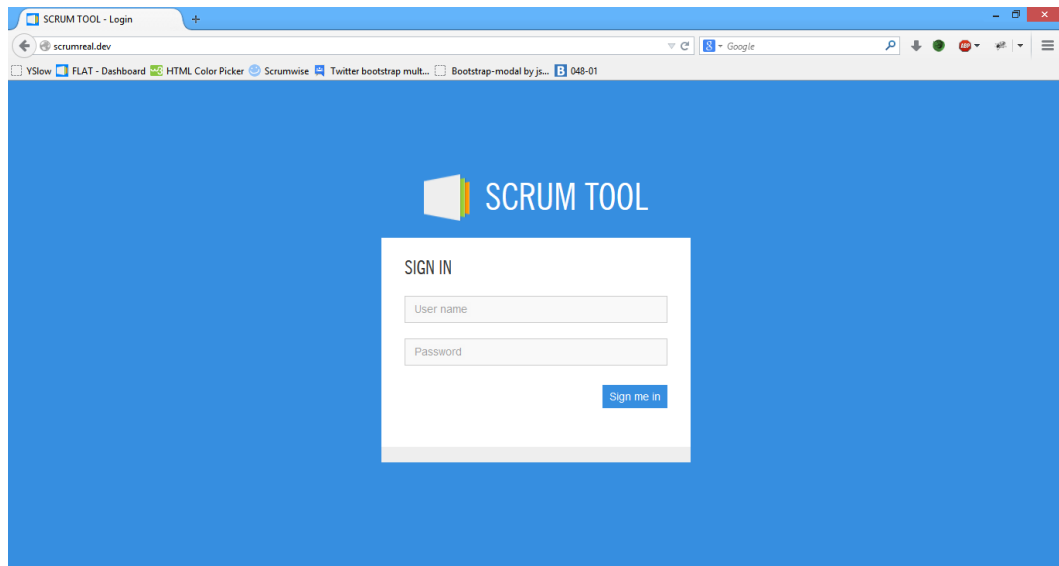
- Opcode: quy định loại gói tin (text, binary, ping, pong).
- Length: quy định độ dài của gói tin.
- Decoding Text: quy định kiểu encode các gói tin thuộc loại text.
- Masking: che dấu dữ liệu, tránh bị nghe trộm, là cơ chế bảo mật của WebSocket.
- Multi-Frame Messages: dữ liệu.

4.3.5. Quy trình bắt tay ngắt kết nối

Khi WebSocket đóng lại, một điểm cuối (có thể là client hay server) mà muốn ngắt kết nối có thể gửi một mã code và lý do chỉ ra tại sao nó đó đóng kết nối socket. Mã code là một số nguyên dương 16 bit, còn lý do đóng kết nối được mã hóa UTF-8.

4.4. Chương trình

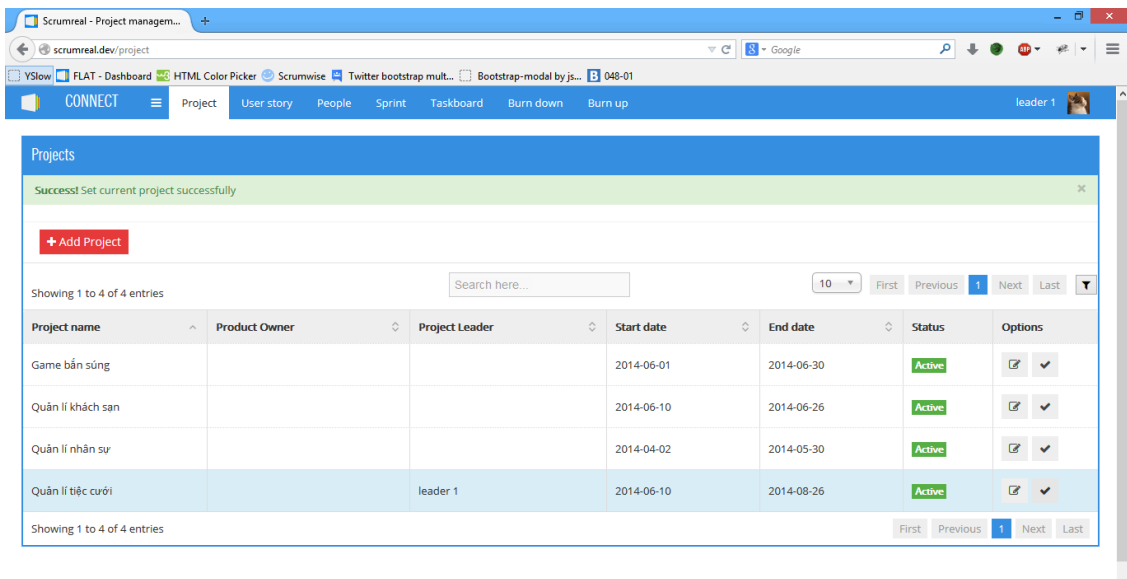
4.4.1. Màn hình đăng nhập



Hình 4.5: Màn hình đăng nhập

Người dùng phải nhập chính xác tên đăng nhập và mật khẩu của mình để sử dụng hệ thống.

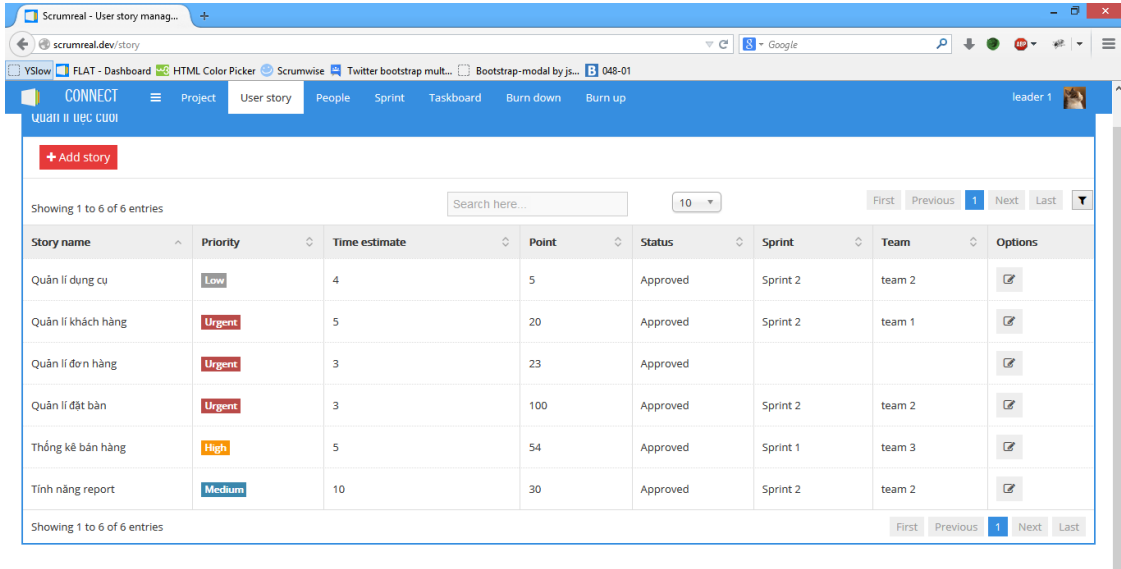
4.4.2. Màn hình quản lý dự án



Hình 4.6: Màn hình quản lý dự án

Ở màn hình này, người dùng có thể chọn dự án làm việc của mình. Thực hiện thêm, xóa, và cập nhật dự án.

4.4.3. Màn hình quản lý User Story

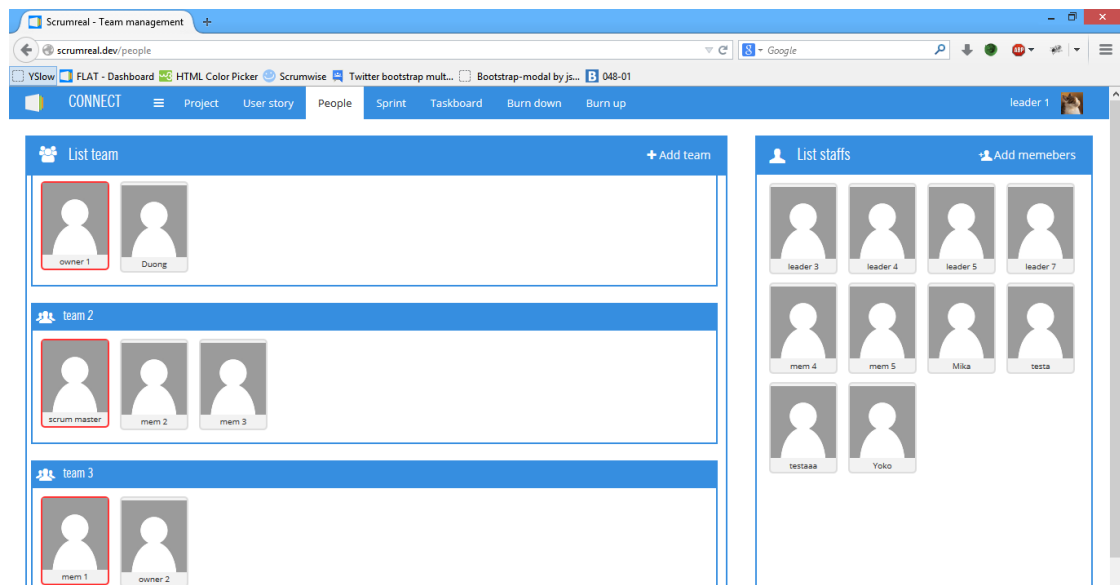


Story name	Priority	Time estimate	Point	Status	Sprint	Team	Options
Quản lý dụng cụ	Low	4	5	Approved	Sprint 2	team 2	[Edit]
Quản lý khách hàng	Urgent	5	20	Approved	Sprint 2	team 1	[Edit]
Quản lý đơn hàng	Urgent	3	23	Approved			[Edit]
Quản lý đặt bàn	Urgent	3	100	Approved	Sprint 2	team 2	[Edit]
Thống kê bán hàng	High	5	54	Approved	Sprint 1	team 3	[Edit]
Tính năng report	Medium	10	30	Approved	Sprint 2	team 2	[Edit]

Hình 4.7: Màn hình quản lý User Story

Ở màn hình này, người dùng có thể xem danh sách tất cả các User Story trong dự án làm việc của mình. Ngoài ra, người dùng còn có thể thực hiện thêm User Story, cập nhật, hoặc xóa User Story. Trong màn hình này, người dùng có thể xem danh sách các Task trong User Story và thực hiện thêm sửa, xóa Task cho từng User Story.

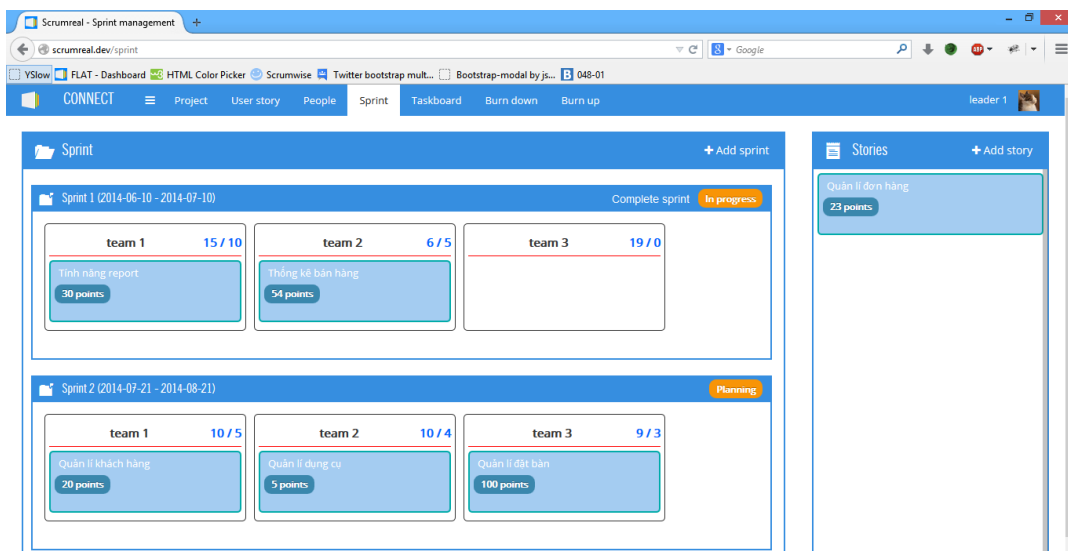
4.4.4. Màn hình quản lý nhân lực



Hình 4.8: Màn hình quản lý nhân lực

Ở màn hình này, người dùng có thể dễ dàng cập nhật nhân sự cho dự án bằng cách kéo thả từ danh sách bên phải vào từng Team. Ngoài ra, việc thêm, sửa, xóa Team cũng có thể được thực hiện tại đây.

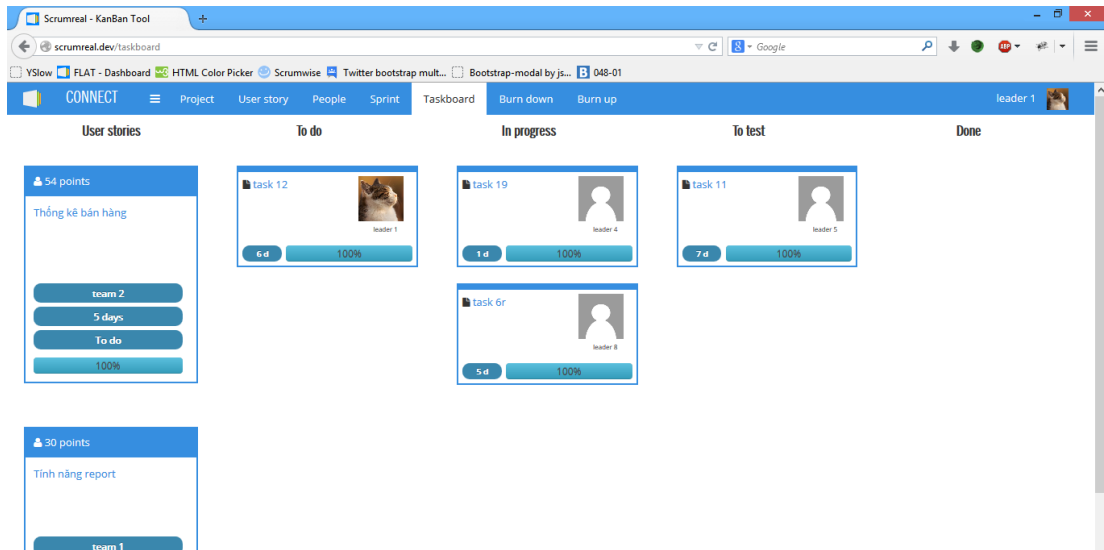
4.4.5. Màn hình quản lý Sprint



Hình 4.9: Màn hình quản lý Sprint

Tại màn hình này, người dùng có thể dễ dàng gắn User Story cho Team thực hiện trong Sprint bằng cách kéo thả từ danh sách bên phải, và thực hiện ngược lại để loại khỏi Sprint. Ngoài ra, người dùng còn có thể thêm, cập nhật và xóa Sprint tại đây.

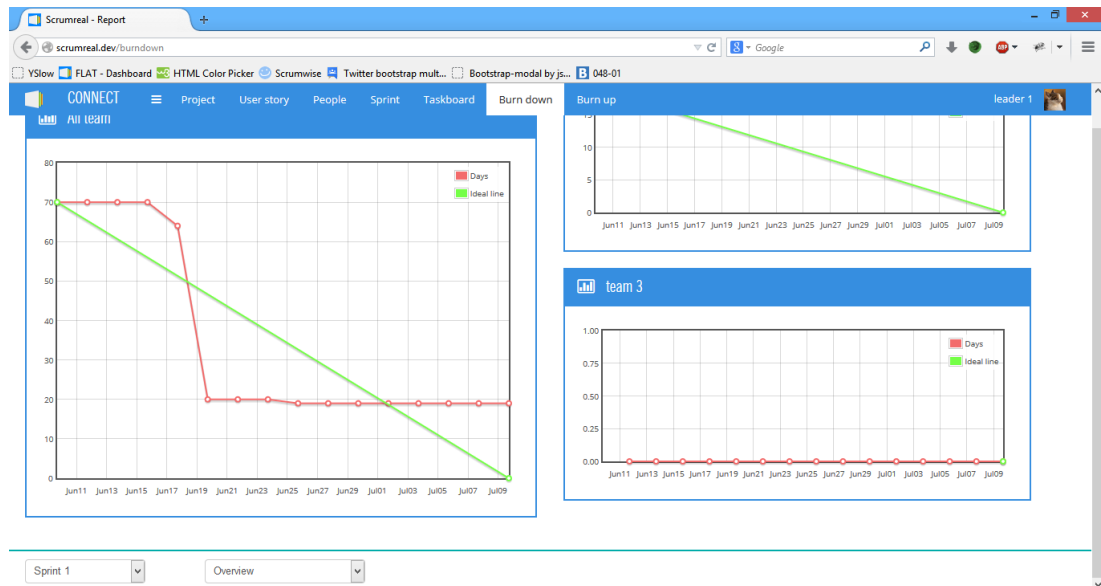
4.4.6. Taskboard



Hình 4.10: Màn hình Taskboard

Màn quản lí Task được xây dựng theo dạng trực quan giúp thuận tiện cho người dùng theo dõi các Task.

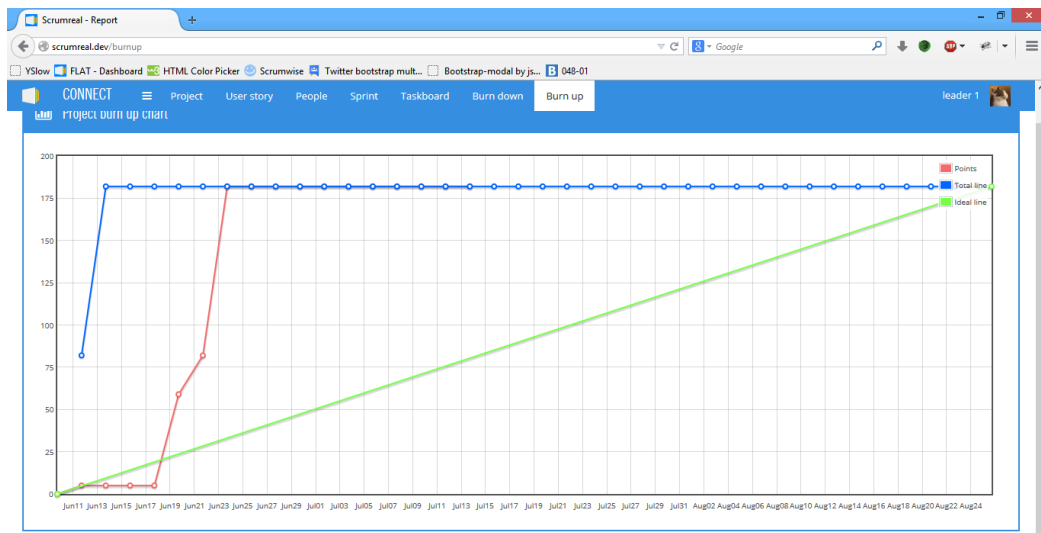
4.4.7. Burndown chart



Hình 4.11: Màn hình burndown chart

Tại màn hình này, người dùng có thể xem các biểu đồ burndown của từng Sprint (cho tất cả các Team, hoặc từng Team riêng lẻ). Biểu đồ burndown, thể hiện tiến độ hoàn thành các Task trong Sprint với đơn vị tính là số ngày của Task.

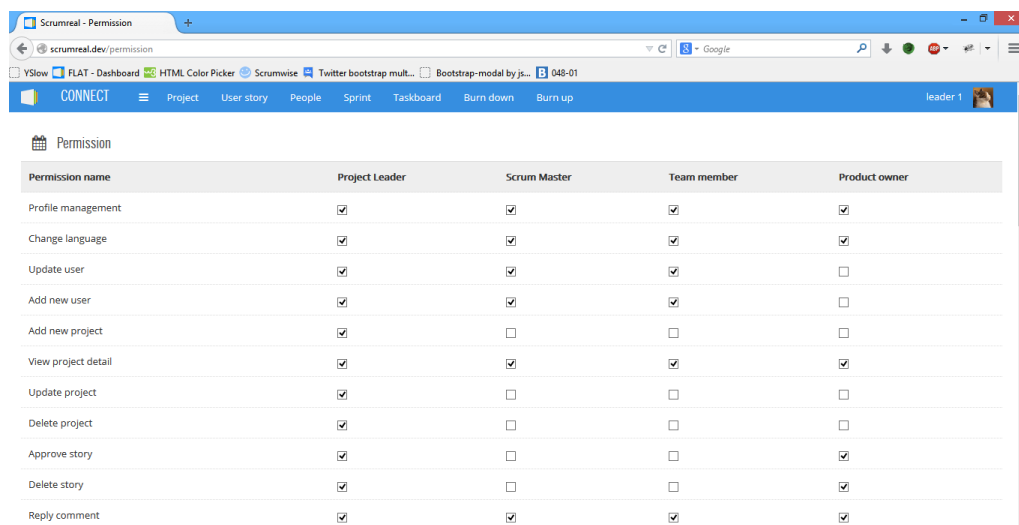
4.4.8. Burnup chart



Hình 4.12: Màn hình burnup chart

Biểu đồ burnup thể hiện tiến độ hoàn thành các User Story trong dự án được tính bằng đơn vị là story point.

4.4.9. Màn hình phân quyền



Permission name	Project Leader	Scrum Master	Team member	Product owner
Profile management	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Change language	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Update user	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Add new user	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Add new project	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
View project detail	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Update project	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Delete project	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Approve story	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Delete story	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Reply comment	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Hình 4.13: Màn hình phân quyền

Màn hình quản lý quyền hạn của từng vai trò trong toàn hệ thống. Mỗi vai trò được cấp các quyền hạn nhất định, và những thành viên trong dự án thuộc vai trò nào sẽ có những quyền như vậy.

KẾT LUẬN

Trong quá trình thực hiện đề tài, nhóm thu được một số kết quả sau:

- Nắm bắt được những kiến thức cơ bản và cần thiết về mô hình Agile và phương pháp Scrum. Từ đó có thể ứng dụng vào công việc và phát triển các dự án sau này.
- Xây dựng công cụ hỗ trợ quản lý dự án theo phương pháp Scrum. Công cụ hỗ trợ hầu hết các bước trong phương pháp Scrum:
 - Quản lý danh sách các dự án, với nhiều tùy chọn, người dùng có thể dễ dàng tìm kiếm, theo dõi, phân loại các dự án.
 - Quản lý danh sách User Story đi kèm với từng dự án. Tương tự như với quản lý dự án, công cụ quản lý Story có rất nhiều tùy chọn, thuận tiện cho người dùng tìm kiếm, sắp xếp.
 - Tính năng quản lý nhóm Scrum trực quan, dễ dàng sử dụng.
 - Tương tự với tính năng quản lý nhóm, tính năng quản lý Sprint rất trực quan, thao tác bằng kéo thả.
 - Giao diện quản lý Task trực quan, người dùng có thể dễ dàng theo dõi được danh sách các Task trong Story, trạng thái và tiến độ của từng Task.
 - Tính năng báo cáo cung cấp hai dạng biểu đồ là burnup chart và burndown chart là hai loại biểu đồ rất hữu ích cho việc theo dõi tiến độ của Sprint cũng như toàn bộ dự án.
 - Tính năng cập nhật thời gian thực, người dùng không phải tải lại trang để cập nhật dữ liệu, công việc này sẽ do hệ thống thực hiện.

HƯỚNG PHÁT TRIỂN

Một số hướng phát triển, mở rộng cho công cụ:

- Xây dựng giao diện cho smartphone và tablet để người dùng có thể thuận tiện sử dụng trên nhiều thiết bị hơn nữa.
- Cải tiến công cụ, bổ sung thêm các tính năng quản lý sự kiện trong Scrum như hợp kế hoạch Scrum, sơ kết Scrum và cải tiến Scrum.
- Tích hợp tính năng chat (cá nhân, nhóm) để tiện giao tiếp giữa các thành viên trong đội ngũ phát triển.
- Bổ sung công cụ bộ bài lập kế hoạch giúp ước tính thời gian cho User Story.

TÀI LIỆU THAM KHẢO

Danh mục tài liệu tiếng Việt

- [1] HaNoi Scrum, *Tổng quan Agile – Phần mở đầu: Đặc trưng* [Online]. Xem chi tiết tại: <http://hanoiscrum.net/hnscrum/learning/106-tongquanagile1>
- [2] HaNoi Scrum, *Scrum là gì?* [Online]. Xem chi tiết tại: <http://hanoiscrum.net/hnscrum/learning/97-agile-manifesto>
- [3] Vumon Developer, *Quy trình Scrum* [Online]. Xem chi tiết tại: <http://developer.vumon.vn/chia-se-kinh-nghiem/quy-trinh-scrum-c823i2488.htm>

Danh mục tài liệu tiếng Anh

- [1] Ken Schwaber và Jeff Sutherland, *The Scrum Guide*. 2013.
- [2] Henrik Kniberg, *Scrum and XP from the Trenches*. C4Media, 2007.
- [3] Vanessa Wang và cộng sự, *The Definite Guide to HTML5 WebSocket*. Appress, 2013, ch. 3, pp. 36-64.
- [4] Jason Lengstorf - Phil Leggetter, *Realtime Web Apps With HTML5 WebSocket, PHP, and jQuery*. Appress, 2013, ch. 1, pp. 6-17.