

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

NGUYỄN KIM THẢO

**XÂY DỰNG HỆ THỐNG QUẢN LÝ VÀ
THỰC THI DỊCH VỤ ỨNG DỤNG QUẢN LÝ**

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

TP. HCM, 2019

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

NGUYỄN KIM THẢO - 1512517

**XÂY DỰNG HỆ THỐNG QUẢN LÝ VÀ
THỰC THI DỊCH VỤ ỨNG DỤNG QUẢN LÝ**

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

**GIÁO VIÊN HƯỚNG DẪN
ThS. NGÔ CHÁNH ĐỨC**

KHÓA 2015 - 2019

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TP.HCM, ngày tháng năm

Giáo viên hướng dẫn

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Khóa luận đáp ứng yêu cầu của Khóa luận cử nhân CNTT.

TP.HCM, ngày tháng năm

Giáo viên phản biện

LỜI CẢM ƠN

Hoàn thành khóa luận tốt nghiệp đối với em là một cột mốc lớn mà em thật sự hạnh phúc và tự hào. Để có được kết quả này không chỉ là sự nỗ lực cả cá nhân em mà còn nhờ sự giúp đỡ của giảng viên hướng dẫn đề tài, của bộ môn và nhà trường.

Trước hết, em muốn gửi lời cảm ơn sâu sắc đối với thầy Ngô Chánh Đức. Được làm việc với thầy và được thầy chỉ bảo, góp ý là những kinh nghiệm quý giá đối với em trong việc hoàn thành đồ án tốt nghiệp và cả trong công việc sau này.

Em chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên, Đại học Quốc gia Tp. Hồ Chí Minh đã tạo điều kiện thuận lợi cho em trong 4 năm học tập tại trường và thực hiện đề tài tốt nghiệp.

Em xin chân thành cảm ơn quý Thầy Cô trong Khoa Công Nghệ Thông Tin đã tận tình giảng dạy, trang bị cho chúng em những kiến thức quý báu trong suốt quá trình học tập để có thể thực hiện được đề tài. Em cũng xin gửi lòng biết ơn đến gia đình và bạn bè, những người đã giúp đỡ, cổ vũ em rất nhiều trong lúc em gặp khó khăn cũng như trong suốt quá trình thực hiện đề tài tốt nghiệp. Em cũng xin cảm ơn những người bạn đã hỗ trợ giúp em đã hoàn thành luận văn này.

Mặc dù đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong nhận được sự góp ý và tận tình chỉ bảo của quý Thầy Cô.

Một lần nữa, em xin chân thành cảm ơn và mong nhận được sự chỉ bảo của quý Thầy Cô để đề tài tốt nghiệp được hoàn chỉnh hơn.

TP. Hồ Chí Minh, 25 tháng 3 năm 2019

Nguyễn Kim Thảo

ĐỀ CƯƠNG CHI TIẾT

Tên Đề Tài: XÂY DỰNG HỆ THỐNG QUẢN LÝ VÀ THỰC THI DỊCH VỤ ỨNG DỤNG QUẢN LÝ
Giao viên hướng dẫn: Ngô Chánh Đức
Thời gian thực hiện: Từ ngày 12/9/2018 đến 8/3/2019
Sinh viên thực hiện: 1512517-Nguyễn Kim Thảo
Loại đề tài: Nghiên cứu lý thuyết, tìm hiểu và cải tiến công cụ

Nội Dung Đề Tài: Đây là đề tài theo hướng nghiên cứu lý thuyết, tìm hiểu công nghệ và xây dựng ứng dụng. Nội dung đề tài bao gồm:

- Tìm hiểu lý thuyết về thực trạng, các hướng tiếp cận phát triển nhanh một ứng dụng quản lý. Trong đó, sinh viên sẽ tiến hành khảo sát, đánh giá một số công cụ hiện có. Từ đó, sinh viên rút ra những hạn chế của những hệ thống, phương pháp hiện hành.
- Nội dung đề tài tập trung vào hướng tiếp cận sử dụng quy trình nghiệp vụ để từ đó hỗ trợ phát sinh tự động một phần hoặc hoàn toàn các dịch vụ của một ứng dụng quản lý. Do đó, sinh viên sẽ tìm hiểu về lý thuyết ngôn ngữ mô hình hóa quy trình nghiệp vụ BPMN và các hệ thống liên quan hỗ trợ thực thi BPMN (Camunda, Flowable). Từ đó, sinh viên quyết định hướng tiếp cận để xây dựng một hệ thống quản lý và thực thi dịch vụ của ứng dụng quản lý dựa trên các mô hình nghiệp vụ quản lý.
- Cuối cùng, sinh viên tiến hành đánh giá khả năng tự động hóa quy trình nghiệp vụ của hệ thống được xây dựng thông qua một nhóm các quy trình nghiệp vụ cụ thể.

Kế Hoạch Thực Hiện:

- Tháng 9/2018 nhận khóa luận.
- 3/10 - 9/10: Tìm hiểu thực trạng, tổng quan về nội dung đề tài
- 10/10 - 16/10: Tìm hiểu mô hình quy trình nghiệp vụ BPMN và các công cụ liên quan
- 17/10 - 23/10: Tìm hiểu về Camunda.
- 24/10 - 30/10: Tìm hiểu Camunda-Database và SDK của camunda.
- 31/10 - 6/11: Phân tích yêu cầu và thiết kế hệ thống.
- 14/10 - 13/11: Đề xuất cải tiến và kiến trúc hệ thống
- 14/11 - 27/1: Xây dựng hệ thống kết hợp triển khai thử nghiệm hệ thống
- 28/1 - 18/2: Đánh giá hoàn chỉnh hệ thống, sửa lỗi, cải thiện hệ thống
- 2/2 - 25/2: Viết khóa luận.

Xác nhận của GVHD**Ngày.....tháng.....năm.....****SV Thực hiện****Nguyễn Kim Thảo**

MỤC LỤC

LỜI CẢM ƠN.....	1
ĐỀ CƯƠNG CHI TIẾT	2
DANH MỤC BẢNG.....	8
DANH MỤC HÌNH ẢNH.....	9
DANH MỤC MÃ NGUỒN.....	11
TÓM TẮT KHOÁ LUẬN	13
KẾT QUẢ ĐẠT ĐƯỢC.....	14
CHƯƠNG 1: TỔNG QUAN.....	15
1.1. Giới thiệu đề tài	15
1.1.1. Lí do thực hiện đề tài	15
1.1.2. Mục tiêu của đề tài	15
CHƯƠNG 2: PHÁT TRIỂN NHANH HỆ THỐNG QUẢN LÝ DỰA TRÊN QUY TRÌNH NGHIỆP VỤ	16
2.1. Thực trạng	16
2.2. Các phần mềm hỗ trợ phát triển nhanh các ứng dụng quản lý	17
2.2.1. Dựa vào biểu mẫu.....	17
2.2.1.1. Giới thiệu chung	17
2.2.1.2. Một số phần mềm nổi bật	17
2.2.1.2.1. Powerapps.....	17
2.2.1.2.2. Form.io	18
2.2.2. Dựa vào mô hình hóa	20
2.2.2.1. Giới thiệu chung	20
2.2.2.2. Một số phần mềm nổi bật	20
2.2.2.2.1. Flowable	20
2.2.2.2.2. Camunda.....	21
2.3. Kết luận.....	22
CHƯƠNG 3: BPMN VÀ CAMUNDA VÀ CAMUNDA-DATABASE.....	25
3.1. BPMN.....	25
3.1.1. Giới thiệu về BPMN	25
3.1.2. So sánh BPMN, FlowChart, Activity Diagram.....	25
3.1.3. Các thành phần của mô hình BPMN.....	26
3.1.3.1. Events	26

3.1.3.2.	Information Artifact	30
3.1.3.2.1.	Đối tượng dữ liệu (Data Object)	30
3.1.3.2.2.	Đầu vào (Data Input)	30
3.1.3.2.3.	Đầu ra (Data Output).....	31
3.1.3.2.4.	Kho dữ liệu (Data Store).....	31
3.1.3.3.	Swimlanes.....	31
3.1.3.4.	Flow.	31
3.1.3.5.	Activities.....	32
3.1.3.5.1.	Activity Marker	33
3.1.3.5.2.	Task Type.....	34
3.1.3.6.	Gateways.....	35
3.2.	Camunda.....	38
3.2.1.	Các thành phần trong BPMN 2.0 mà Camunda hỗ trợ.....	38
3.2.2.	Kiến trúc Camunda.....	41
3.2.2.1.	Process Engine	41
3.2.2.2.	Ứng dụng web của Camunda	43
3.2.2.3.	Các công cụ hỗ trợ.....	43
3.2.3.	Một số mô hình triển khai của Camunda	44
3.2.3.1.	Embedded Process Engine.....	44
3.2.3.2.	Shared, Container-Managed Process Engine	44
3.2.3.3.	Standalone Process Engine	46
3.2.3.4.	Clustering Model	47
3.2.3.5.	Multi-Tenancy.	47
3.2.4.	Môi trường mà Camunda hỗ trợ.	47
3.2.4.1.	Môi trường được hỗ trợ từ các thành phần Camunda	47
3.2.4.2.	Môi trường hỗ trợ cho Camunda Cycle.	48
3.2.4.3.	Cơ sở dữ liệu	48
3.2.4.4.	Cơ sở dữ liệu phân nhóm và cơ sở dữ liệu nhân rộng.....	48
3.2.4.5.	Các trình duyệt Camunda hỗ trợ.....	49
3.2.4.6.	Java.....	49
3.2.4.7.	Camunda Modeler	49
3.2.4.8.	Một số hình ảnh về Camunda	49
3.3.	Camunda Database	53

3.3.1.	Sơ đồ thực hiện tổng quát của hệ thống Camunda-database.....	53
3.3.2.	Các thành phần đã mở rộng trong Camunda-database.	55
3.3.2.1.	Mở rộng hệ thống mô hình hóa BPMN.....	55
3.3.2.1.1.	Giới thiệu về Camunda Modeler.....	55
3.3.2.1.2.	Cấu trúc các tập tin mã nguồn của công cụ Modeler.....	56
3.3.2.2.	Mở rộng hệ thống vận hành quy trình bằng DEP.....	57
3.3.2.2.1.	Giới thiệu về DEP.....	57
3.3.2.2.2.	Mô hình kiến trúc của DEP.....	57
3.3.2.2.3.	Cách hoạt động của DEP.....	58
3.3.3.	Những khuyết điểm trong hệ thống Camunda-Database.....	60
	CHƯƠNG 4: HỆ THỐNG ĐỀ XUẤT.....	61
4.1.	Giới thiệu về hệ thống.....	61
4.2.	Hướng tiếp cận xây dựng hệ thống.....	62
4.3.	Lựa chọn hệ thống CMS.....	62
4.4.	Kiến trúc tổng quan.....	67
	CHƯƠNG 5: QUÁ TRÌNH CÀI ĐẶT.....	70
5.1.	Mở rộng trên hệ thống Camunda-Database.....	70
5.1.1.	Mở rộng định nghĩa bổ sung cho việc mô hình hóa BPMN.....	70
5.1.2.	Mở rộng hệ thống Camunda-Modeler.....	70
5.1.3.	Mở rộng Process-Engine (tái sử dụng DEP).....	72
5.2	Mở rộng Camunda SDK.....	74
5.2.1	Giới thiệu về Camunda SDK.....	74
5.2.2	Cấu trúc thư mục Camunda SDK.....	76
5.2.3	Mở rộng trên các trường form control.....	77
5.3	Cài đặt Camunda-Extend-System.....	78
5.3.1	Phương pháp tiếp cận.....	78
5.3.2	Mô hình của Camunda-Engine-Extend.....	79
5.3.2.1	Mô hình kiến trúc của CEE.....	79
5.3.2.2	Mô hình database của Camunda-Extend-System.....	80
5.4	Cài đặt Camunda-Wordpress-Plugin.....	82
5.4.1	Phương pháp tiếp cận.....	82
5.4.2	Cấu trúc thư mục.....	83
5.4.3	Các hoạt động của WCP.....	85

CHƯƠNG 6: KẾT QUẢ CÀI ĐẶT VÀ ĐÁNH GIÁ	88
6.1 Kết quả cài đặt.....	88
6.1.1 Camunda-Extend-System.....	88
6.1.2 Wordpress.....	90
6.2 Xây dựng ứng dụng minh họa.....	91
6.3 Đánh giá	97
CHƯƠNG 7: KẾT LUẬN	98
7.1 Kết quả đạt được.....	98
7.1.1 Kiến thức.....	98
7.1.2 Kỹ thuật	98
7.1.3 Sản phẩm	98
7.2 Những khó khăn trong quá trình thực hiện	98
7.3 Hướng phát triển đề tài	99
DANH MỤC THAM KHẢO	101

DANH MỤC BẢNG

Bảng 2. 1 - Điểm chung của Camunda và Flowable.....	23
Bảng 2. 2 - Điểm khác của Camunda và Flowable.....	24
Bảng 3. 1 - Bảng so sánh giữa BPMN, Flow Chart và Activity Diagram	26
Bảng 3. 2 – Bảng mô tả và kí hiệu một số Event đặc biệt.	29
Bảng 5. 1 – Chi tiết các thuộc tính thêm vào UserTask.....	70
Bảng 5. 2 – Các tài nguyên truy cập	75
Bảng 5. 3– Các phương thức truy cập tài nguyên	75
Bảng 5. 4- Chi tiết các form control được thêm vào cấu trúc SDK.....	77
Bảng 5. 5- Chi tiết các attribute mới thêm vào	78
Bảng 5. 6- Các function hook cơ bản trong WCP.....	84

DANH MỤC HÌNH ẢNH

Hình 2.1 - Giao diện của Powerapps.....	18
Hình 2.2 - Giao diện của Form.io	19
Hình 2.3 - Giao diện của Fowable-Modeler	21
Hình 3.1 – Ví dụ cho cổng độc quyền (Exclusive Gateway).....	36
Hình 3.5 – Ví dụ cho cổng bao hàm (Inclusive Gateway)	38
Hình 3.6 - Các loại Activity Camunda hỗ trợ [6]	39
Hình 3.7 – Các loại Gateways [6]	39
Hình 3.8 – Các sự kiện được cài đặt trong Camunda [6].....	40
Hình 3. 9 – Kiến trúc của Process Engine trong Camunda [4]	42
Hình 3.10 – Mô hình Embedded Process Engine [4].....	44
Hình 3.11 - Mô hình Shared, Container-Managed Process Engine [4]	45
Hình 3.12– Mô hình Standalone Process Engine [4].....	46
Hình 3.13 – Mô hình Clustering [4].....	47
Hình 3.14 – Giao diện của Camunda Admin	50
Hình 3.15 –Giao diện của Camunda Cockpit	51
Hình 3.16 – Giao diện của Camunda Tasklist	52
Hình 3.17 – Giao diện của Camunda Modeller	53
Hình 3.18 - Sơ đồ thực hiện của hệ thống Camunda-Database	54
Hình 3.19 – Cấu trúc của các thư mục trong Modeler.....	56
Hình 3.20 - Mô hình Data Extension Plugin (DEP)	58
Hình 3.21 – Luồng xử lý của việc ghi, cập nhật và xóa bỏ dữ liệu	59
Hình 3. 22 – Luồng xử lý của việc đọc dữ liệu.....	59
Hình 4.1 - Mô tả các tính năng của hệ thống đề xuất	62
Hình 4.2 - Giao diện của Wordpress.....	64
Hình 4.3 - Giao diện của Joomla.....	65

Hình 4.4 - Giao diện của Drupal	66
Hình 4.5 - Sơ đồ thực hiện của hệ thống.....	67
Hình 5.1 - Mô tả kiến trúc cho việc render form.	73
Hình 5. 2 - Cấu trúc thư mục Camunda	76
Hình 5. 3- Mô hình kiến trúc của Camunda-Engine-Extend.	79
Hình 5. 4 - Mô hình database của Camunda-Extend-System.	80
Hình 5.5 - Cấu trúc thư mục của wordpress-camunda-plugin	83
Hình 5.6 – Luồng xử lý việc chỉnh sửa Form	86
Hình 5.7 - Luồng xử lý việc thực thi Task.	86
Hình 6.1 - Giao diện login của BWC.....	88
Hình 6.2 - Giao diện bussines-webapp-creator	89
Hình 6.3 - Giao diện thiết kế vụ trong Wordpress	90
Hình 6.4 - Kiến trúc database nghiệp vụ quản lí mượn sách.	91
Hình 6.5 - Các thêm trường author vào form addbook.	93
Hình 6.6 - Thêm catergory có khóa ngoại đến catergories vào form addbook.....	93
Hình 6.7 - Các thêm table vào nghiệp vụ listbook.....	94
Hình 6.8 - Giao diện của Workspace Quản lý đặt book.	95
Hình 6.9 - Giao diện để chỉnh sửa nghiệp vụ addbook.....	95
Hình 6.10 - Giao diện sau khi chỉnh Addbook	96
Hình 6.11 - Giao diện kết quả của listbook.....	96

DANH MỤC MÃ NGUỒN

Mã nguồn 5.1 - Thêm procedure và query vào DataInputAssociation	71
Mã nguồn 5.2 - Thêm thuộc tính form-item vào User Task	72
Mã nguồn 5.3 - Thêm ButtonFormItem vào formItem.....	74
Mã nguồn 5.4 - Cách cài đặt table-field-handler	78

BẢNG CHỮ CÁI VIẾT TẮT

CHỮ VIẾT TẮT	CHỮ VIẾT ĐẦY ĐỦ
BPMN	Bussiness Process Modeling Notation
BPEL	Business Process Execution Language
OMG	Object Management Group
BPMI	Business Process Management Initiative
UML	Unified Modeling Language
SQL	Structured Query Language
DEP	Data Extension Plugin
JDBC	Java DataBase Connectivity
API	Application Programming Interface
XML	eXtensible Markup Language
JVM	Java Virtual Machine
JMS	Java Message Service
CMS	Content Management System
CEE	Camunda Engine Extend
DAO	Data Access Object
WCP	Wordpress Camunda Plugin
BWC	Business Webapps Creator
SDK	Software Development Kit
CDS	Camunda Database System
CES	Camunda Extend System

TÓM TẮT KHOÁ LUẬN

Khóa luận tập trung vào hướng tiếp cận sử dụng quy trình nghiệp vụ để phát sinh tự động một phần hoặc hoàn toàn các dịch vụ của một ứng dụng quản lý.

Khóa luận này nêu ra thực trạng, các hướng tiếp cận phát triển nhanh một ứng dụng quản lý. Một số công cụ hiện có cũng được tiến hành khảo sát, đánh giá. Ngoài ra, khóa luận còn trình bày về lý thuyết ngôn ngữ mô hình hóa quy trình nghiệp vụ BPMN và các hệ thống hỗ trợ thực thi BPMN.

Từ đó, khóa luận đã đề xuất và xây dựng thử nghiệm hệ thống giúp quản lý và thực thi các nghiệp vụ quản lý dưới dạng ứng dụng web với giao diện thân thiện với người dùng cuối. Hệ thống đề xuất dựa trên nền tảng Camunda để thực thi các quy trình nghiệp vụ. Đồng thời, khóa luận đã quyết định chọn Wordpress làm hệ thống quản lý nội dung cho hệ thống này.

Hệ thống đề xuất đã được cài đặt thành công với các yêu cầu đề ra và được thử nghiệm với quy trình nghiệp vụ cụ thể. Tuy nhiên, với các nghiệp vụ phức tạp, hệ thống vẫn chưa đáp ứng do sự hỗ trợ hạn chế mô hình BPMN của Camunda.

KẾT QUẢ ĐẠT ĐƯỢC

Nội dung khóa luận gồm có 7 chương

CHƯƠNG 1: Trình bày lý do và mục tiêu khi thực hiện đề tài “XÂY DỰNG HỆ THỐNG QUẢN LÝ VÀ THỰC THI DỊCH VỤ ỨNG DỤNG QUẢN LÝ”.

CHƯƠNG 2: Giới thiệu về phương pháp phát triển nhanh phần mềm quản lý quy trình nghiệp vụ với mô hình hóa BPMN.

CHƯƠNG 3: Giới thiệu về BPMN và kiến trúc tổng quan của Camunda. Các loại mô hình Process Engine. Tìm hiểu về Camunda Database

CHƯƠNG 4: Trình bày tổng quan về hệ thống đề xuất

CHƯƠNG 5: Trình bày quá trình cài đặt hệ thống đề xuất.

CHƯƠNG 6: Kết quả thu được sau khi cải tiến hệ thống và đánh giá mức độ hoàn thiện của hệ thống.

CHƯƠNG 7: Kết luận.

CHƯƠNG 1: TỔNG QUAN

Trong chương này, em sẽ giới thiệu về lí do thực hiện đề tài và mục tiêu của đề tài.

1.1. Giới thiệu đề tài

1.1.1. Lí do thực hiện đề tài

Trong thời đại công nghệ phát triển nhanh chóng như hiện nay, tin học hóa là nhu cầu tất yếu của mỗi doanh nghiệp. Nhu cầu xây dựng hệ thống quản lý quy trình nghiệp vụ của doanh nghiệp là điều tất yếu trong thị trường hiện nay. Nhưng để doanh nghiệp xây dựng một hệ thống quản lý quy trình nghiệp vụ của thông thường có 2 cách : một là nhờ từ công ty khác tạo ra và phát triển, hai là tự xây dựng đội ngũ IT để phục vụ cho chuyện này. Dù là cách nào, để phát triển hệ thống quản lý quy trình nghiệp vụ cũng thông qua một quy trình phát triển phần mềm nào đó.

Quy trình phát triển phần mềm thông thường phải thông qua nhiều pha. Thông thường là 5 pha : phân tích yêu cầu, thiết kế, cài đặt , kiểm thử, triển khai. Trong đó pha cài đặt là pha tốn thời gian và chi phí nhất, và đòi hỏi nhân lực phải có trình độ lập trình. Việc tiết kiệm chi phí, thời gian và nhân lực để phát triển hệ thống quản lý cho riêng mình là một bài toán nan giải của từng doanh nghiệp. Để giải quyết những vấn đề trên cần có công cụ hỗ trợ cần phải được xây dựng trực quan, thân thiện, dễ sử dụng và tối ưu hóa mức độ tự động.

1.1.2. Mục tiêu của đề tài

Trong đề tài lần này, em hướng tới đến đề xuất xây dựng thử nghiệm hệ thống quản lý và thực thi dịch vụ của ứng dụng quản lý dựa trên các mô hình nghiệp vụ quản lý dựa trên hướng tiếp cận tự động hoá các mô hình quy trình nghiệp vụ BPMN trên cơ sở việc tìm hiểu và ứng dụng các nền tảng thực thi quy trình nghiệp vụ sẵn có.

CHƯƠNG 2: PHÁT TRIỂN NHANH HỆ THỐNG QUẢN LÝ DỰA TRÊN QUY TRÌNH NGHIỆP VỤ

Trong chương này, em sẽ nêu lên thực trạng hiện nay của quá trình phát triển nhanh hệ thống quản lý, đồng thời cũng so sánh ưu nhược những phần mềm hiện có, từ đó chọn ra hệ thống để tiếp tục phát triển.

2.1. Thực trạng

Công nghệ thông tin ngày nay có mặt trong hầu hết các doanh nghiệp và là công cụ hỗ trợ đắc lực trong việc quản lý. Nhưng với sự mở rộng quy mô hệ thống tổ chức doanh nghiệp yêu cầu việc xây dựng hệ thống quản lý trở nên khó khăn, chi phí xây dựng hệ thống khá tốn kém. Theo Forrester [5], năm 2013, các công ty đã chi 542 tỷ đô cho việc xây dựng phần mềm tùy chỉnh, ứng dụng và các phần mềm trung gian. Đặc biệt, mỗi ứng dụng tiêu tốn rất nhiều tiền để cập nhật và bảo trì mỗi năm. Chỉ riêng trong năm 2014, chi phí bảo trì ứng dụng đã tăng 29% [7] và chưa có dấu hiệu dừng lại. Hơn nữa, các ứng dụng không thể thiết kế để đảm bảo tất cả những yêu cầu phát sinh trong tương lai, giả sử một công ty có phần mềm quản lý cho bộ phận nhân sự với nhiều chức năng và cài đặt phức tạp, điều gì sẽ xảy ra nếu cần phải thêm một chức năng mới khác biệt? Quá trình nâng cấp hệ thống sẽ trở nên khó khăn và thời gian lâu dài. Để giải quyết vấn đề đó đòi hỏi phải phát triển một hệ thống hỗ trợ thiết kế được quy trình nghiệp vụ của tổ chức, đồng thời có thể chuyển mô hình đó thành ngôn ngữ thực thi được và tạo thành hệ thống cho doanh nghiệp.

Phát triển nhanh hệ thống quản lý quy trình là một giải pháp tạo ra hệ thống hỗ trợ xây dựng hệ thống quản lý một cách đơn giản, nhanh chóng mà không đòi hỏi nhiều về kiến thức lập trình ở người sử dụng. Nhìn ra được vấn đề này, các công ty hiện nay đã ra đời các công cụ để hỗ trợ phát triển hệ thống quản lý, nổi bật trong đó

có 2 hướng là: dựa vào biểu mẫu ([Form.io](#), [powerapps](#),...) và dựa vào mô hình hóa quy trình nghiệp vụ ([Camunda](#), [Flowable](#),...)

2.2. Các phần mềm hỗ trợ phát triển nhanh các ứng dụng quản lý

2.2.1. Dựa vào biểu mẫu

2.2.1.1. Giới thiệu chung

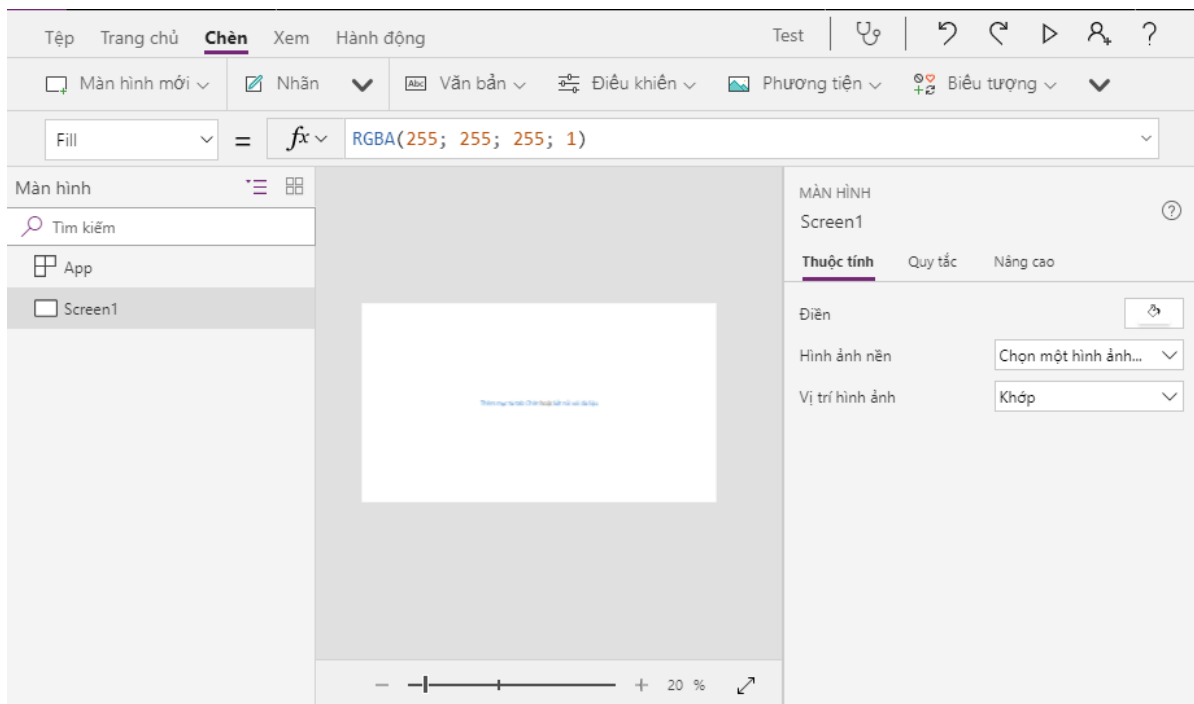
Phần mềm hỗ trợ phát triển nhanh các ứng dụng quản lý dựa vào biểu mẫu ngày càng trở nên phổ biến. Cùng với sự phát triển của các phần mềm thì nhu cầu của doanh nghiệp trở nên đa dạng hơn. Nhiều phần mềm đã trở nên phổ biến như: Powerapps, Form.io... Dựa vào nguồn dữ liệu người dùng cung cấp, các phần mềm hỗ trợ sẽ phát sinh ra ứng dụng phần mềm theo ý muốn người dùng.

2.2.1.2. Một số phần mềm nổi bật

2.2.1.2.1. Powerapps

2.2.1.2.1.1. Giới thiệu

Powerapps là một bộ ứng dụng, dịch vụ do Microsoft phát triển, cung cấp môi trường phát triển ứng dụng nhanh chóng để xây dựng các ứng dụng tùy chỉnh cho nhu cầu kinh doanh. Với Powerapps các doanh nghiệp có thể nhanh chóng tạo ra các ứng dụng tùy chỉnh kết nối với dữ liệu doanh nghiệp của họ được lưu trữ trong các nguồn dữ liệu trực tiếp và tại chỗ khác nhau như SharePoint, Excel, Office 365, SQL Server, MySQL,...



Hình 2.1 - Giao diện của Powerapps

2.2.1.2.1.2. Những tính năng nổi bật

- Xây dựng nhanh ứng dụng bằng cách kéo và thả.
- Cung cấp luồng công việc phong phú
- Biến quy trình thủ công thành quy trình tự động
- Kết nối với nguồn dữ liệu đa dạng
- Chia sẻ tức thì ứng dụng cho người khác
- Tạo ứng dụng trên đa nền tảng (Web, Mobile, PC,...)

2.2.1.2.1.3. Nhược điểm

- Công thức rất nhiều gây khó khăn cho người mới sử dụng.
- Phụ thuộc vào nguồn dữ liệu để tạo layout
- Không thể hiện rõ quy trình đang hiện thực

2.2.1.2.2. Form.io

2.2.1.2.2.1. Giới thiệu

Form.io là một nền tảng quản lý các form và quản lý dữ liệu API dành cho các doanh nghiệp xây dựng các ứng dụng quy trình kinh doanh. Nền tảng đã được sử dụng rộng rãi trên thế giới từ mọi doanh nghiệp, công ty từ nhỏ đến lớn. Một số ngành kinh doanh đã áp dụng form.io như là quản lý, bảo hiểm, tài chính,... [2]

Hình 2.2 - Giao diện của Form.io

2.2.1.2.2.2. Những tính năng nổi bật

- Cung cấp các API cho Progressive Web Application
- Triển khai tại chỗ hoặc đám mây riêng
- Tích hợp bên thứ 3
- Hành động biểu mẫu có điều kiện
- Quản lý và lưu trữ tệp
- Xuất dữ liệu
- Báo cáo API tổng hợp nâng cao
- Tích hợp dữ liệu thời gian thực với Webhooks

2.2.1.2.2.3. Nhược điểm

- Ít có khả năng kiểm soát html

- Không thực sự thân thiện cho việc SEO website
- Chưa có thể tạo ra website, chỉ có thể tích hợp vào website

2.2.2. Dựa vào mô hình hóa

2.2.2.1. Giới thiệu chung

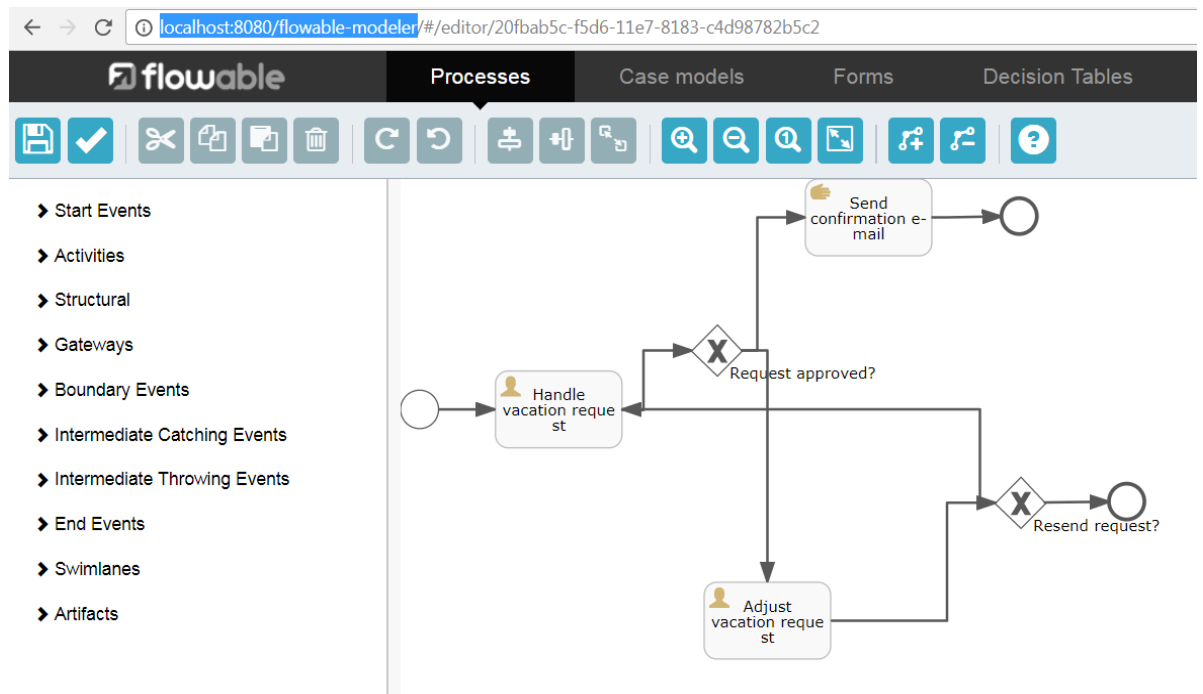
Song hành với sự phát triển các phần mềm hỗ trợ phát triển nhanh ứng dụng quản lý dựa vào biểu mẫu, thì hướng xây dựng các phần mềm dựa vào mô hình hóa cũng phát triển không kém. Phát triển ứng dụng quản lý dựa vào mô hình hóa là từ các sơ đồ mô hình hóa quy trình nghiệp vụ mà người dùng cung cấp tự động xây dựng hệ thống quản lý. Nhiều phần mềm đã trở nên quen thuộc với người dùng như: Flowable, Camunda...

2.2.2.2. Một số phần mềm nổi bật

2.2.2.2.1. Flowable

2.2.2.2.1.1. Giới thiệu

Flowable là một công cụ quản lý nghiệp vụ nhẹ được viết bằng Java. Công cụ xử lý Flowable cho phép triển khai các định nghĩa quy trình BPMN 2.0 (một tiêu chuẩn XML công nghiệp để xác định các quy trình), tạo các phiên bản quy trình của các định nghĩa quy trình đó, chạy truy vấn, truy cập các phiên bản quy trình hoạt động hoặc lịch sử và dữ liệu liên quan, và nhiều hơn nữa. Là một mã nguồn mở được cộng đồng phát triển .



Hình 2.3 - Giao diện của Fowable-Modeler

2.2.2.2.1.2. Những tính năng nổi bật

- Cực kì linh hoạt khi thêm nó vào ứng dụng / dịch vụ /kiến trúc
- Nhúng công cụ vào ứng dụng hoặc dịch vụ của mình
- Cung cấp các REST API
- Cung cấp cái UI mẫu ngoài luồng để làm việc với các quy trình

2.2.2.2.1.3. Nhược điểm

- Không có tài liệu rõ ràng
- Không cho người dùng có thể plugin
- Việc hỗ trợ form cho Task còn hạn chế
- Các nghiệp vụ khi thực thi trên nền tảng web còn hạn chế

2.2.2.2.2. Camunda

2.2.2.2.2.1. Giới thiệu

Camunda là một hệ thống mã nguồn mở được viết bằng Java, giúp quản lý quy trình nghiệp vụ, dùng để định nghĩa và thực thi các quy trình nghiệp vụ trong BPMN 2.0. Cho đến hiện tại, Camunda đã phát triển một số thành phần phổ biến, thiết yếu dựa trên tiêu chuẩn BPMN 2.0. Ví dụ như là Process Engine, Camunda Modeler,...

2.2.2.2.2. Những tính năng nổi bật

- Cực kì linh hoạt khi thêm nó vào ứng dụng / dịch vụ / kiến trúc
- Nhúng công cụ vào ứng dụng hoặc dịch vụ của mình
- Cung cấp các REST API
- Cung cấp cái UI mẫu ngoài luồng để làm việc với các quy trình
- Mã nguồn mở, cho phép thêm plugin.
- Tài liệu rõ ràng

2.2.2.2.3. Nhược điểm

- Việc hỗ trợ form cho Task còn hạn chế
- Các nghiệp vụ khi thực thi trên nền tảng web còn hạn chế
- Việc truy xuất xuống database vẫn chưa có.

2.3. Kết luận

Nhìn chung, đối với những hệ thống quản lý được phát triển dựa theo biểu mẫu có điểm chung là khó mô tả chi tiết trong quá trình thực thi nghiệp vụ, gây khó khăn cho những người trong team cùng phát triển. Ngoài ra nó còn phụ thuộc vào nguồn dữ liệu nguồn trước đó. Từ những lý do này, em sẽ tập trung hướng vào việc phát triển nhanh bằng mô hình hóa quy trình nghiệp vụ.

Cùng với sự phát triển nhanh hệ thống quản lý là sự ra đời của các Engine nhằm phục vụ cho người dùng là các doanh nghiệp có thể thiết kế và triển khai mô hình BPMN. Một số Engine phổ biến hiện nay là:

- Camunda BPM, Camunda Company

- Flowable, Flowable Company

Mỗi Engine có điểm mạnh và điểm yếu riêng, tùy vào mục đích sử dụng của người dùng mà chọn Engine hợp lí. Em đã lập ra các bảng để chỉ ra điểm chung, điểm khác của hai Engine phổ biến trên:

Điểm chung của Camunda và Flowable	
Nhân	BPMN 2.0
Nền tảng	Java
Thư viện	Có hỗ trợ thư viện trong các dự án
Platform	Có Platform thực thi quy trình nghiệp vụ
Khả năng truy vết dữ liệu	Không hỗ trợ
Hỗ trợ lưu dữ liệu	Không hỗ trợ
Nghiệp vụ tùy chỉnh	Chỉ hỗ trợ project có sử dụng thư viện

Bảng 2. 1 - Điểm chung của Camunda và Flowable

	Camunda	Flowable
Database	MS SQL, PostgreSQL, Oracle, MySQL, H2, DB2, MariaDB Có thể cấu hình thay đổi trong quá trình sử dụng	MS SQL, PostgreSQL, Oracle, MySQL, H2, DB2 Có thể cấu hình thay đổi trong quá trình sử dụng.
Môi trường	Có 2 công cụ riêng biệt: một công cụ dùng để thiết kế quy trình nghiệp vụ chạy trên desktop, một dùng để thực thi các quy	Dùng 4 công cụ, mỗi công cụ có một chức năng khác nhau: thiết kế, thực thi, quản lí. Tất cả đều chạy trên môi trường web

	trình nghiệp vụ đó chạy trên môi trường web	
Tài liệu	Có tài liệu mô tả rõ ràng kiến trúc của hệ thống	Không có tài liệu mô tả kiến trúc hệ thống
Khả năng mở rộng	Hỗ trợ Plugin vào hệ thống	Không hỗ trợ Plugin
Khả năng hỗ trợ BPMN 2.0	Hỗ trợ gần như đầy đủ các thành phần của BPMN 2.0	Thành phần được hỗ trợ rất ít
Hỗ trợ Form cho task	Hỗ trợ thiết kế form theo kiểu liệt kê các thành phần của Form	Hỗ trợ thiết kế kéo thả

Bảng 2. 2 - Điểm khác của Camunda và Flowable

Flowable và Camunda đều được dùng để tạo lập, quản lý và xử lý các quy trình nghiệp vụ. Các Engine tự động cài đặt các quy trình đã qua định nghĩa. Hỗ trợ đầy đủ các tính năng của BPMN 2.0 giúp cho các thao tác trên nghiệp vụ trở nên hết sức đơn giản.

Riêng Camunda có tài liệu mô tả kiến trúc rõ ràng giúp cho việc mở rộng hệ thống trở nên dễ dàng, cùng với khả năng plugin vào hệ thống trong khi Flowable không hỗ trợ, đây là điểm mạnh mà Camunda vượt trội hơn Flowable.

Vì vậy em chọn Camunda để làm nền tảng phát triển thêm các tính năng mới mà các công cụ BPMN còn thiếu sót chưa đáp ứng được.

CHƯƠNG 3: BPMN VÀ CAMUNDA VÀ CAMUNDA-DATABASE

Ở chương này, em xin trình bày chi tiết về của mô hình BPMN, và các công cụ của Camunda. Đồng thời em cũng trình bày mô hình logic [4] và một số mô hình triển khai của Camunda và em cũng sẽ giới thiệu về hệ thống Camunda-Database đã được cải tiến trong luận văn tốt nghiệp “NGHIÊN CỨU VÀ CẢI TIẾN HỆ THỐNG CAMUNDA ĐỂ PHÁT TRIỂN NHANH CÁC HỆ THỐNG QUẢN LÝ VỚI BPMN” [1]

3.1. BPMN

3.1.1. Giới thiệu về BPMN

Có rất nhiều dạng mô hình hỗ trợ cho việc phát triển nhanh một phần mềm quản lý, Hiện nay, có nhiều bộ ký hiệu khác nhau giúp đặc tả quy trình nghiệp vụ, cụ thể là: FlowChart, Activity Diagram, BPMN. Trong đó, BPMN thể hiện tinh ưu việt hơn hẳn so với các bộ ký hiệu còn lại. Vậy BPMN là gì?

Business Process Modeling Notation (BPMN): là tập hợp các ký hiệu đồ họa theo quy chuẩn để mô tả một quy trình nghiệp vụ của tổ chức hay còn gọi là mô hình hóa quy trình nghiệp vụ. Ngoài ra, BPMN còn cung cấp một cơ chế thực thi quy trình nghiệp vụ bằng ngôn ngữ BPEL từ một mô hình nghiệp vụ ở mức kí hiệu.

3.1.2. So sánh BPMN, FlowChart, Activity Diagram

Trước tiên, em sẽ thực hiện so sánh để có cái nhìn tổng quát về 3 loại mô hình đã đề cập.

Mô hình	Giống nhau	Khác nhau
BPMN	<p>Đều là mô hình biểu diễn một quy trình nghiệp vụ trong một mô hình nghiệp vụ cụ thể bằng đồ họa kí hiệu</p>	<p>Các kí hiệu có tiêu chuẩn rõ ràng và đa dạng, phù hợp để mô tả cụ thể các quy trình phức tạp, cần nhiều xử lí.</p> <p>Hỗ trợ chuyển đổi được sang ngôn ngữ thực thi BPEL.</p>
Flow Chart		<p>Các kí hiệu tương đối đơn giản, dễ hình dung nhưng không quá đa dạng về thành phần gây khó khăn cho việc đặc tả các quy trình có các xử lí đặc biệt (có triggers, events,...).</p>
Activity Diagram		<p>Các kí hiệu cũng tương đối đầy đủ như BPM. Không hỗ trợ chuyển đổi được sang ngôn ngữ thực thi BPEL.</p>

Bảng 3. 1 - Bảng so sánh giữa BPMN, Flow Chart và Activity Diagram

Có thể thấy, mô hình BPMN có nhiều ký hiệu, ngữ nghĩa nên nó gần với thực tế. Ngoài ra nó còn rõ ràng, dễ đọc, dễ hiểu nên phù hợp hơn cho các đối tượng không chuyên về công nghệ thông tin. Hơn nữa, BPMN có hỗ trợ chuyển đổi sang ngôn ngữ thực thi BPEL, mấu chốt quan trọng trong việc phát triển nhanh hệ thống quản lý. Đồng thời các công cụ hỗ trợ việc xây dựng, cài đặt và triển khai các quy trình dựa trên BPMN phổ biến hơn so với các mô hình còn lại. Do đó việc chọn BPMN sẽ giúp cho quá trình trao đổi giữa bộ phận phát triển và doanh nghiệp trở nên dễ dàng hơn, từ đó rút ngắn thời gian phân tích, thu thập yêu cầu, tăng thời lượng cài đặt và triển khai, hơn nữa còn giảm thiểu sự sai sót do hiểu nhầm ý nghĩa biểu tượng hoặc không hiểu tường tận quy trình BPMN.

3.1.3. Các thành phần của mô hình BPMN

3.1.3.1. Events

Event giúp diễn tả một sự kiện xảy ra trong nghiệp vụ nhưng không được chủ đích bởi user mà do ảnh hưởng của yếu tố bên ngoài. Các sự kiện (event) gồm 3 thành phần quan trọng:

Start Event: Là sự kiện giữ nhiệm vụ bắt đầu một quy trình, không cần điều kiện kích hoạt.

Kí hiệu bằng một vòng tròn viền nhạt.



End Event: Là sự kiện giữ nhiệm vụ kết thúc một quy trình, có thể xem là kết quả của quy trình.

Kí hiệu bằng một vòng tròn viền đậm.



























Intermediate Event: Là sự kiện diễn ra ngay tức thì, ở giữa một quy trình, giúp xác định điều kiện làm gián đoạn quy trình.

Kí hiệu bằng một vòng tròn kép.



Trong quy trình nghiệp vụ sẽ xảy ra những sự kiện phức tạp hơn, BPMN có hỗ trợ một nhóm các sự kiện đặc biệt để thể hiện những ý nghĩa khác nhau.



Event type	Start	Intermediate		End	Mô tả
	Catch	Catch	Throw	Throw	
Timer					Được sử dụng để diễn tả một sự việc <i>liên quan đến thời gian</i> , được kích hoạt sau khoảng thời gian xác định (1 ngày, 1 tháng...).
Link					Nó không có ngữ nghĩa thực thi đặc biệt nhưng đóng vai trò đi đến một sự kiện khác trong cùng một mô hình quy trình.
Error					Được kích hoạt bởi một lỗi cụ thể, sử dụng để ghi lại lỗi và xử lý lỗi.
Cancel					Sự kiện này nghĩa là người dùng đã lựa chọn hủy thực thi một quy trình.
Compensation					Được sử dụng để mô tả việc một quy trình được quay lại trạng thái trước đó trong lúc thực.
Signal					Được sử dụng để gửi nhận tín hiệu giữa các Pool (cùng hoặc khác Participant) hoặc giữa các quy trình với nhau.

Multiple					Dùng để tóm tắt các loại sự kiện có trong một mô hình và được kích hoạt khi một trong số các loại này thỏa mãn điều kiện.
Parallel Multiple					Dùng để tóm tắt các loại sự kiện có trong một mô hình nhưng chỉ được kích hoạt khi tất cả các loại này thỏa mãn điều kiện.
Terminate					Là sự kiện dừng tất cả. Khi nhận được Event này, toàn bộ quy trình sẽ được dừng lại.



Bảng 3. 3 – Bảng mô tả và kí hiệu một số Event đặc biệt.

Các event sẽ có một số ràng buộc nhất định vào quy trình và thuộc một trong 2 dạng:

Throwing Events: gửi những sự kiện xảy ra trong quá trình thực thi một quy trình.

Các Throwing Events sẽ có màu đen, ví dụ:  (End Event Throw Message),  (Intermediate Event Throw Message).

Catching Events: bắt những sự kiện xảy ra trong quá trình thực thi một quy trình.

Các Catching Events sẽ có màu trắng, ví dụ:  (Start Event Throw Message),  (Intermediate Event Throw Message).

3.1.3.2. Information Artifact

Đây là thành phần quan trọng trong bất cứ quy trình nào, nó đại diện cho thông tin, dữ liệu. Các thành phần của Information Artifact gồm có: Đối tượng dữ liệu (Data Object), Đầu vào (Data Input), Đầu ra (Data Output), Kho dữ liệu (Data Store)

3.1.3.2.1. Đối tượng dữ liệu (Data Object)

Data Object là thể hiện của dữ liệu trong quy trình như email, tài liệu, form..

Tham chiếu Data Object là một cách để sử dụng lại các Data Object trong cùng một diagram. Chúng có thể chỉ định các trạng thái khác nhau của cùng một Data Object tại các điểm khác nhau trong một tiến trình.

Kí hiệu của Data Object:



Một loại Data Object đặc biệt của BPMN là bộ các Data Object (Data Object Collection), đại diện cho một mảng các Data Object.

Kí hiệu của Data Object Collection:



3.1.3.2.2. Đầu vào (Data Input)

Data Input là một loại dữ liệu đặc biệt được sử dụng làm đầu vào của một tiến trình. Data Input trong diagram quy trình nghiệp vụ hiển thị đầu vào của dữ liệu cho tiến trình cấp cao nhất hoặc hiển thị đầu vào của tiến trình được gọi.

Kí hiệu của Data Input:



3.1.3.2.3. Đầu ra (Data Output)

Data Output là một loại dữ liệu đặc biệt được tạo ra như đầu ra của một tiến trình. Data Output trong diagram quy trình nghiệp vụ cấp cao nhất hiển thị đầu ra của quy trình.



Kí hiệu của Data Output:

3.1.3.2.4. Kho dữ liệu (Data Store)

Data Store cho phép các hoạt động truy xuất hoặc cập nhật thông tin lưu trữ sẽ tồn tại.



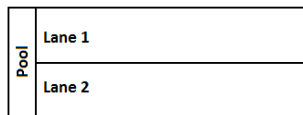
Kí hiệu của Data Store:

Ngoài ra, để thể hiện luồng đi của dữ liệu trong mô hình, BPMN sử dụng liên kết dữ liệu (Data Association).

Kí hiệu của Data Association:>

3.1.3.3. Swimlanes

Có hai cách thức để nhóm các phần tử mô hình hóa chính thông qua Swimlanes là Pool và Lane. Trong đó, Pool là biểu diễn đồ họa của một Thành phần tham gia còn Lane là một phân vùng thuộc một Process (đôi khi thuộc một Pool).




Kiểu hiệu :

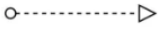
3.1.3.4. Flow.

Có 4 cách dùng để thể hiện luồng đi trong quy trình nghiệp vụ, cụ thể gồm:


Luồng tuần tự (Sequence Flow): Luồng mô tả thứ tự thực hiện của các hoạt động trong quy trình.

Kí hiệu của Sequence Flow: 

Luồng thông điệp (Message Flow): Luồng để trao đổi thông tin giữa các Lane hoặc Pool.

Kí hiệu của Message Flow: 

Luồng mặc định (Default Flow): Nếu không có điều gì xảy ra, quy trình nghiệp vụ sẽ đi theo luồng mặc định.

Kí hiệu của Default Flow: 

Luồng điều kiện (Conditional Flow): Đầu vào của luồng là một điều kiện. Quy trình nghiệp vụ sẽ đi theo luồng này nếu điều kiện được thỏa mãn.

Kí hiệu của Conditional Flow: 

Liên kết (Association): Sử dụng để liên kết giữa các Marker Anotation với các thành phần khác trong mô hình

Kí hiệu của Association: 


3.1.3.5. Activities

Activities dùng để mô tả công việc trong quy trình nghiệp vụ. Activities chia làm 2 loại chính: Task và Sub-process

Task: là hoạt động nguyên tố, không thể chia nhỏ ra được nữa.

Kí hiệu của Task: 

Sub-process là hoạt động lớn có thể phân chia thành có hoạt động con.

Kí hiệu của Sub-process: 

Tuy nhiên, để thể hiện được nhiều activities phức tạp hơn, BPMN cung cấp thêm một số Marker để thể hiện hành vi thực hiện của các task và Task Type để thể hiện rõ nghĩa hơn các task.

3.1.3.5.1. Activity Marker

Activity Marker bao gồm 4 loại chính: Loop, Compensation, Multiple Instance, Ad Hoc.

Loop: Một task trong quy trình nghiệp vụ sẽ được lặp đi lặp lại nhiều lần.

Kí hiệu của Loop: 

Compensation: Là tác vụ mô tả sự backup, trở lại task ban đầu trong quy trình nghiệp vụ.

Kí hiệu của Compensation: 

Ad Hoc: Bao gồm các tác vụ trong một Sub-process mà chưa biết được thứ tự thực hiện chúng và thứ tự thực hiện đó chỉ được hình thành khi cần thiết và dùng cho một mục đích nhất định.

Kí hiệu của Ad Hoc: 

Multiple Instance: Là tác vụ cho phép thực hiện xong nhiều task cùng một thời điểm.

Kí hiệu của Multiple Instance:



3.1.3.5.2. Task Type

BPMN cung cấp một bộ Task Type để thể hiện tính chất của các task.

Send task: Là các tác vụ BPMN đơn giản được thiết kế để gửi thông điệp đến những Swimlanes khác (thường là khác Pool), khi các thông điệp được gửi đi thì các tác vụ này cũng được hoàn thành

Kí hiệu của Send task:



Recieve task: Có tác vụ gửi thì sẽ cần tác vụ nhận, nên Receive task là một tác vụ được thiết kế để chờ một thông điệp nào đó từ Swimlanes khác (thường là khác Pool), Khi thông điệp được nhận, tác vụ này cũng được hoàn thành

Kí hiệu của Recieve task:



User Task: Là tác vụ được sử dụng phổ biến trong một quy trình BPMN vì nó đại diện cho một tác vụ điển hình trong một quy trình làm việc. User Task là một tác vụ được thực thi bởi một người và có sự trợ giúp của phần mềm ứng dụng mà mô hình BPMN đang mô tả.

Kí hiệu của User Task:



Manual Task: Trái với User Task, Manual Task là một tác vụ được thực thi bởi một người nhưng sẽ không có sự trợ giúp của phần mềm ứng dụng mà mô hình BPMN đang mô tả

Kí hiệu của Manual Task:



Service Task: Khác với User Task và Manual Task, Service Task không yêu cầu sự tương tác của người dùng mà nó được thực hiện tự động bằng service của hệ thống. Tác vụ này được gọi là Automated Task. Service Task sẽ được thực hiện tự động thông qua các hàm được thêm vào trong quá trình thực thi.

Kí hiệu của Service Task:



Script Task: nó là một loại Automated Task. Script Task là tác vụ được thực hiện bởi chính công cụ thực thi của hệ thống. Một đoạn code sẽ được tạo sẵn dựa trên các ngôn ngữ mà công cụ của hệ thống có thể chạy được. Khi tác vụ này thực thi, công cụ này sẽ chạy đoạn code đã dựng sẵn để thực hiện công việc.

Kí hiệu của Script Task:



Business Rule Task: là task được thực hiện dựa trên những quy tắc nghiệp vụ nào đó.

Kí hiệu của Business Rule Task:



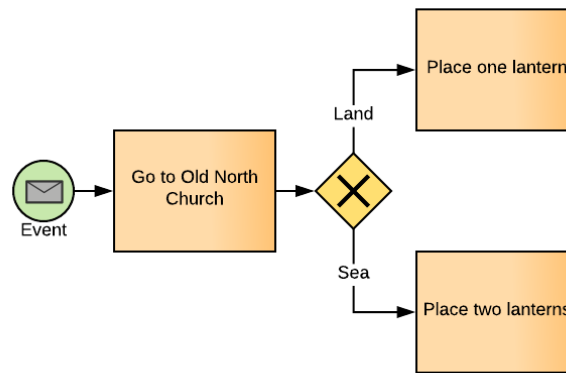
3.1.3.6. Gateways.

Gateway là đối tượng điều khiển dòng để trộn hoặc phân chia các luồng thực thi. Vì vậy nó sẽ quyết định việc rẽ nhánh, trộn... các luồng tiến

trình với nhau tùy thuộc vào loại hành vi được chỉ định. BPMN thì có rất nhiều Gateway, khoảng... 68 loại nhưng có 5 loại Gateway thường được sử dụng

Exclusive Gateway : Cổng độc quyền đánh giá trạng thái của quy trình nghiệp vụ và dựa trên điều kiện, chia flow thành một trong hai hoặc nhiều đường dẫn loại trừ nhau.

Ký hiệu của Exclusive Gateway:

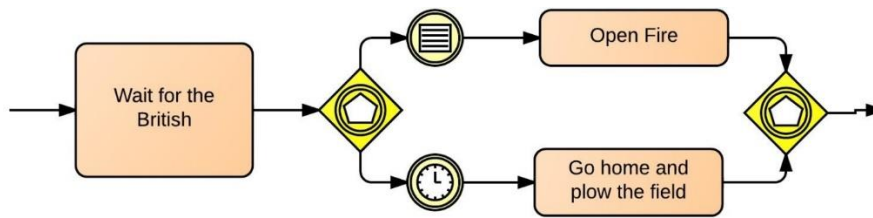


Hình 3.1 – Ví dụ cho cổng độc quyền (Exclusive Gateway)

Event-based Gateways : Cổng dựa trên sự kiện tương tự như cổng độc quyền vì cả hai đều liên quan đến một đường dẫn trong luồng. Tuy nhiên, trong cổng dựa trên sự kiện, bạn đánh giá sự kiện nào đã xảy ra chứ không phải điều kiện nào cũng đã được đáp ứng.

Ký hiệu Event-based Gateways:

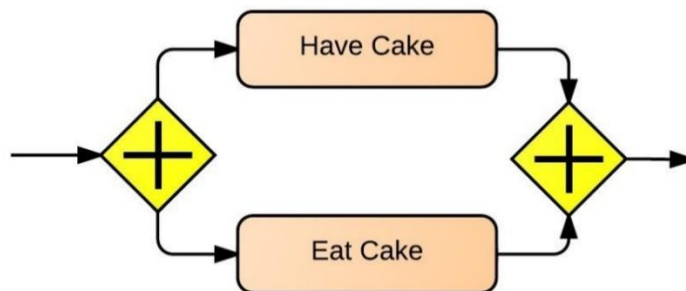




Hình 3.2 – Ví dụ cho cổng dựa trên sự kiện (Event-Based Gateway)

Parallel Gateway: Cổng song song rất khác so với các cổng trước vì không đánh giá bất kỳ điều kiện hoặc sự kiện nào. Thay vào đó, một cổng song song được sử dụng để thể hiện hai nhiệm vụ đồng thời trong một quy trình nghiệp vụ. Nó giống như một ngã ba trong sơ đồ hoạt động UML.

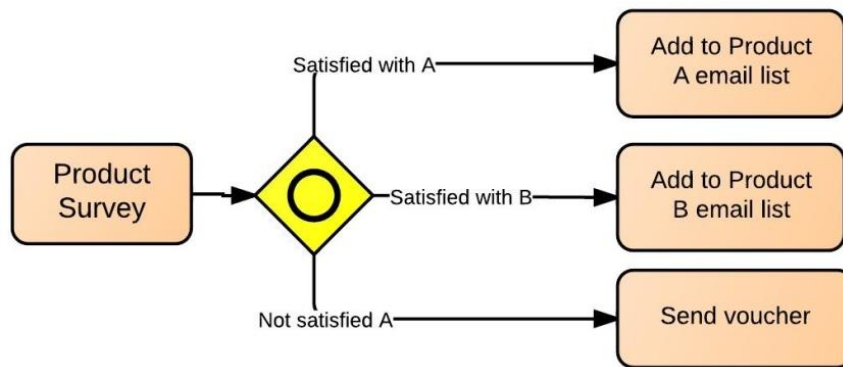
Ký hiệu Parallel Gateway: 



Hình 3.3 – Ví dụ cho cổng song song (Parallel Gateway)

Inclusive Gateways: Một cổng bao hàm phá vỡ luồng tiến trình thành một hoặc nhiều luồng.

Ký hiệu Inclusive Gateways: 



Hình 3.5 – Ví dụ cho cổng bảo hàm (Inclusive Gateway)

Complex Gateway: Các cổng phức tạp chỉ được sử dụng cho các luồng phức tạp nhất trong quy trình nghiệp vụ. Người ta sử dụng các từ thay cho các ký hiệu và do đó, đòi hỏi bản mô tả nhiều hơn. Sử dụng cổng phức tạp nếu cần nhiều cổng để mô tả quy trình nghiệp vụ; nếu không, bạn nên sử dụng một cổng đơn giản hơn.

Ký hiệu Complex Gateway:

3.2. Camunda

3.2.1. Các thành phần trong BPMN 2.0 mà Camunda hỗ trợ.

Những thành phần được Camunda hỗ trợ trong Modeler được đánh dấu bằng màu cam

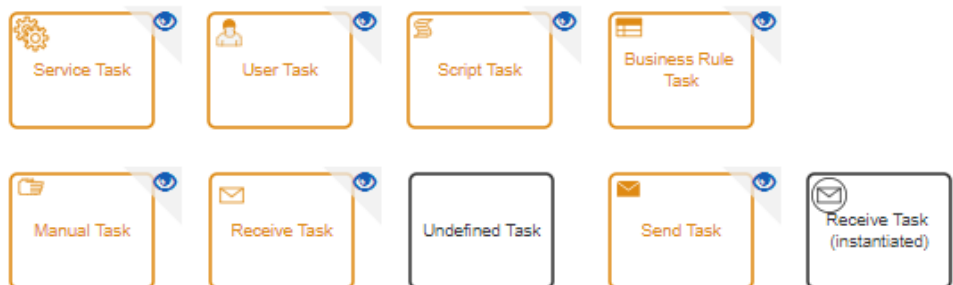
Participants



Subprocesses

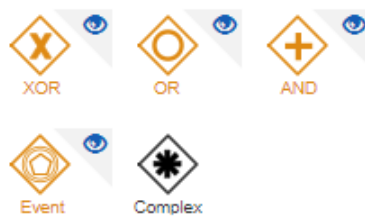


Tasks



Hình 3.6 - Các loại Activity Camunda hỗ trợ [6]

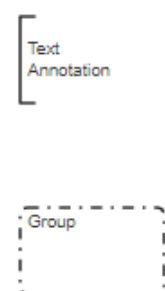
Gateways







































Data






























Artifacts



Hình 3.7 – Các loại Gateways [6]

Type	Start			Intermediate				End
	Normal	Event Subprocess	Event Subprocess non-interrupt	catch	boundary	boundary non-interrupt	throw	
None								
Message								
Timer								
Conditional								
Link								
Signal								
Error								

Escalation								
Termination								
Compensation								
Cancel								
Multiple								
Multiple Parallel								

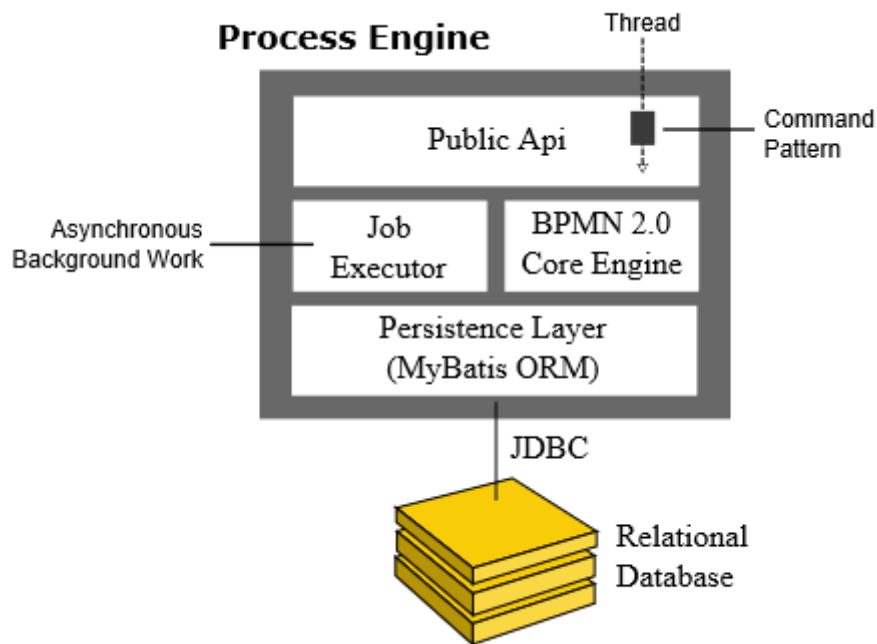
Hình 3.8 – Các sự kiện được cài đặt trong Camunda [6]

3.2.2. Kiến trúc Camunda.

Camunda BPM là một Framework dựa trên nền tảng Java. Các thành phần chính của Camunda được viết bằng Java và tập trung vào việc cung cấp cho các nhà phát triển Java các công cụ họ cần để thiết kế, thực hiện và chạy các quy trình nghiệp vụ. Bao gồm 3 thành phần chính: Process Engine, các ứng dụng web và các công cụ hỗ trợ

3.2.2.1. Process Engine

Process Engine là một thư viện Java giúp thực thi các mô hình BPMN và các workflow của BPMN . Hỗ trợ các hệ quản trị cơ sở dữ liệu như: PostgreSQL, MySQL và H2, đồng thời tích hợp MyBatis API giúp thao tác với cơ sở dữ liệu dễ dàng hơn. Nó tự động ánh xạ giữa các trường của bảng trong cơ sở dữ liệu SQL và các trường trong Java POJO (Plain Old Java Objects) theo tên trường, nhằm phục vụ cho việc mapping ORM (Object Relational Mapping). Hình 3.9 cho thấy kiến trúc của thành phần Process Engine này.



Hình 3. 9 – Kiến trúc của Process Engine trong Camunda [4]

Process Engine Public API: Đây là API hướng dịch vụ cho phép các ứng dụng Java tương tác với Process Engine. Các đảm nhận khác nhau của Process Engine (Process Repository, Runtime Process Interaction, ...) được phân tách thành các dịch vụ riêng lẻ. Public API có kiểu truy cập command: Các luồng vào Process Engine được định tuyến thông qua Command Interceptor được sử dụng để thiết lập Thread Context như là các Transaction.

BPMN 2.0 Core Engine: Đây là nhân của Process Engine , giúp thực thi các tính năng đơn giản cho các cấu trúc đồ thị (PVM - Process Virtual Machine), phân tích cú pháp BPMN 2.0 để chuyển đổi các file XML của BPMN 2.0 thành các Java Object và BPMN Behavior implementations (cung cấp việc triển khai cho các cấu trúc BPMN 2.0 như Gateways hoặc ServiceTask).

Job executor: Đây là thành phần chịu trách nhiệm thực thi, xử lý các công việc bất đồng bộ thuộc về background như Timers và các phần bất đồng bộ khác trong một quy trình.

The Persistence Layer: Đây là thành phần chịu trách nhiệm kết nối cơ sở dữ liệu quan hệ (Relational Database) và đồng nhất các thành phần trong quy trình. Bên cạnh đó, The Persistence Layer còn sử dụng công cụ MyBatis để ánh xạ các quan hệ đối tượng xuống cơ sở dữ liệu.

3.2.2.2. Ứng dụng web của Camunda

Ngoài Process Engine, Camunda còn hỗ trợ thêm người dùng các ứng dụng trên nền tảng Web có thể sử dụng một cách dễ dàng Process Engine. Các ứng dụng bao gồm:

REST API: ứng dụng cho phép sử dụng các Process Engine từ một ứng dụng khác hoặc ứng dụng JavaScript.

Camunda Tasklist: là một ứng dụng web cho phép quản lý nghiệp vụ và tác vụ của người dùng, đồng thời cho phép người dùng tham gia quá trình kiểm tra, giám sát và điều hướng các nghiệp vụ và cung cấp dữ liệu đầu vào.

Camunda Cockpit: Dùng để quản lý các quy trình nghiệp vụ, cho phép tìm kiếm các phiên bản của nghiệp vụ, kiểm tra và sửa chữa những trường hợp có lỗi.

Camunda Admin: Cung cấp quyền quản trị để quản lý nhiều người dùng, nhóm người dùng và quyền.

Camunda Cycle: Ứng dụng web dùng để đồng bộ hóa các mô hình quy trình BPMN 2.0 giữa các modeling tool và modeler.

3.2.2.3. Các công cụ hỗ trợ

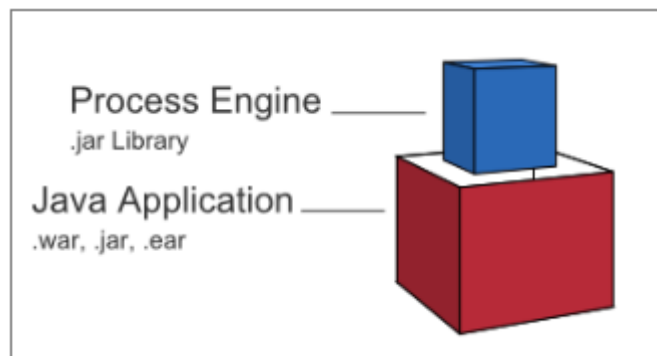
Camunda Modeler: là một ứng dụng máy tính cho mô hình hóa BPMN 2.0 và CMMN 1.1 giúp cho các file lưu trữ trực tiếp trên hệ thống cục bộ.

Camund-bpm-sdk.js: Một JavaScript Framework dùng để parse, render form task và thực thi các mô hình BPMN 2.0 từ nguồn XML.

3.2.3. Một số mô hình triển khai của Camunda

Nền tảng của Camunda BPM là một Framework linh hoạt, có thể triển khai cho nhiều tình huống khác nhau. Phần này cung cấp một cái nhìn tổng quát về các mô hình triển khai phổ biến nhất.

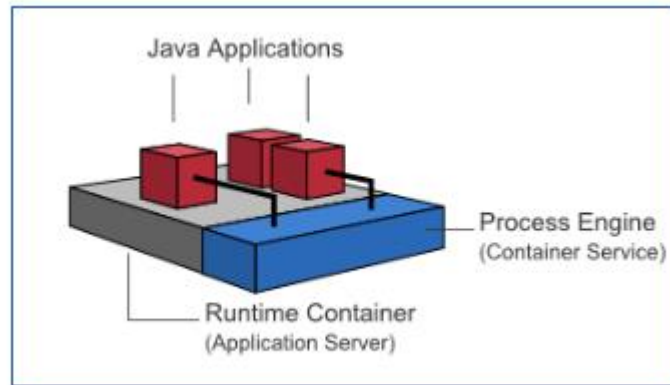
3.2.3.1. Embedded Process Engine



Hình 3.10 – Mô hình Embedded Process Engine [4]

Trong trường hợp này, Process Engine là một thư viện đính kèm vào ứng dụng. Như vậy thì các Process Engine dễ dàng được khởi động và dừng lại cùng với ứng dụng, cùng chạy nhiều Embedded Process Engine trên cùng một cơ sở dữ liệu.

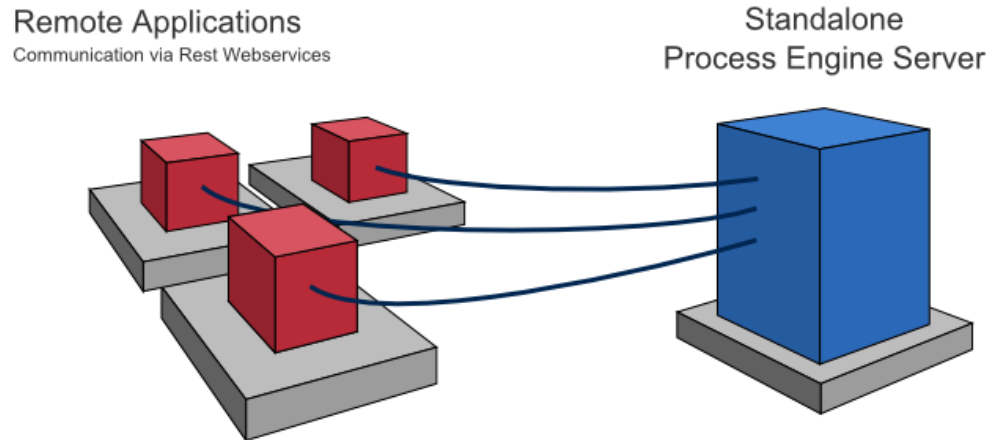
3.2.3.2. Shared, Container-Managed Process Engine



Hình 3.11 - Mô hình Shared, Container-Managed Process Engine [4]

Process Engine được khởi động bên trong Runtime Container (Servlet Container, Application Server, ...), cung cấp như là một dịch vụ của Container và có thể chia sẻ bởi tất cả các dịch vụ và các ứng dụng bên trong Container. Khái niệm này khá giống JMS Message Queue được cung cấp và sử dụng bởi tất cả ứng dụng.

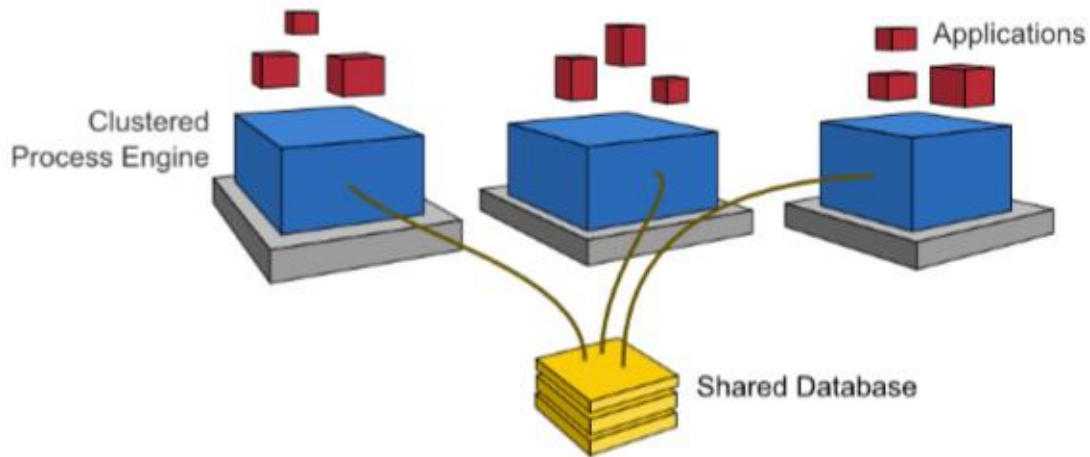
3.2.3.3. Standalone Process Engine



Hình 3.12– Mô hình Standalone Process Engine [4].

Theo trường hợp này, Process Engine được cung cấp như một dịch vụ mạng. Các ứng dụng khác nhau chạy mạng lưới này có thể tương tác với các Process Engine thông qua các kênh liên lạc từ xa. Cách dễ dàng nhất để có thể truy cập các Process Engine từ xa là sử dụng REST API tích hợp

3.2.3.4. Clustering Model



Hình 3.13 – Mô hình Clustering [4]

Để cung cấp khả năng mở rộng và giảm thiểu tỉ lệ xảy ra lỗi trên toàn bộ hệ thống, Process Engine có thể được phân phối đến các nút khác nhau trong một cụm. Mỗi Process Engine hiện tại phải kết nối với cơ sở dữ liệu dùng chung.

3.2.3.5. Multi-Tenancy.

Để đáp ứng nhu cầu cùng phục vụ nhiều bên, Process Engine có hỗ trợ Multi-Tenancy¹. Người dùng nên chọn mô hình phù hợp với nhu cầu phân tách dữ liệu của họ. Các API của Camunda có cung cấp quyền truy cập vào các quy trình và dữ liệu cụ thể cho từng đối tượng thuê.

3.2.4. Môi trường mà Camunda hỗ trợ.

3.2.4.1. Môi trường được hỗ trợ từ các thành phần Camunda

¹ Multitenancy là hệ thống giải quyết vấn đề một physical/compute server có nhiều VM/ container thuộc nhiều tenant/user khác nhau. Nhằm tăng hiệu quả sử dụng tài nguyên khi một server có cấu hình cao chạy nhiều VM/ container có flavor vừa và nhỏ

- Apache Tomcat 6.0 / 7.0 / 8.0 / 9.0
- JBoss Application Server 7.2 and JBoss EAP 6.1 / 6.2 / 6.3 / 6.4 / 7.0 / 7.1
- Wildfly Application Server 8.2 / 10.1 / 11.0
- IBM WebSphere Application Server 8.0 / 8.5 / 9.0 (Enterprise Edition only)
- Oracle WebLogic Server 12c (12R1,12R2) (Enterprise Edition only)
- Ứng dụng Spring Boot cùng với Tomcat (Bao gồm các phiên bản được hỗ trợ).

3.2.4.2. Môi trường hỗ trợ cho Camunda Cycle.

- Apache Tomcat 7

3.2.4.3. Cơ sở dữ liệu

Camunda hỗ trợ các phiên bản hệ quản trị cơ sở dữ liệu.

- MySQL 5.6 / 5.7
- MariaDB 10.0
- Oracle 10g / 11g / 12c
- IBM DB2 9.7 / 10.1 / 10.5 / 11.1 (excluding IBM z/OS for all versions)
- PostgreSQL 9.1 / 9.3 / 9.4 / 9.6
- Microsoft SQL Server 2008 R2/2012/2014/2016 (see Configuration Note).

3.2.4.4. Cơ sở dữ liệu phân nhóm và cơ sở dữ liệu nhân rộng.

Cơ sở dữ liệu phân nhóm hay nhân rộng hỗ trợ cho các điều kiện sau: Giao tiếp giữa Camunda BPM và cơ sở dữ liệu phân cụm phải khớp với cấu hình không phân cụm / không nhân rộng tương ứng. Quan trọng nhất là cấu hình của cơ sở dữ liệu phân cụm phải đảm bảo các hành vi thực hiện ở mức cô

lập READ-COMMITTED. Hệ thống hiện tại chỉ mới support cho MariaDB Galera Cluster.

3.2.4.5. Các trình duyệt Camunda hỗ trợ

- Google Chrome mới nhất
- Mozilla Firefox mới nhất
- Internet Explorer 11
- Microsoft Edge

3.2.4.6. Java

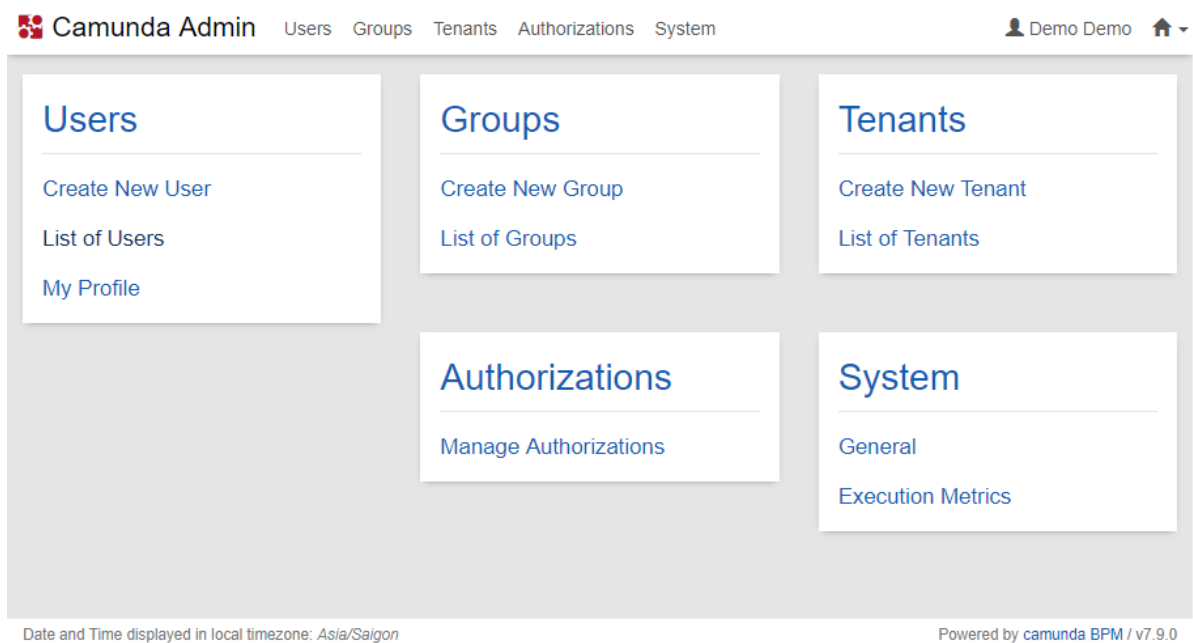
- Java 6/7.
- Java 8 (nếu được hỗ trợ bởi ứng dụng server và container)

3.2.4.7. Camunda Modeler

Camunda Modeler phù hợp với các hệ điều hành sau:

- Windows 7 / 10
- Mac OS X 10.11
- Linux

3.2.4.8. Một số hình ảnh về Camunda



Hình 3.14 – Giao diện của Camunda Admin

Camunda Cockpit Processes Decisions Human Tasks More ▾ Demo Demo

Dashboard » Processes » AddBook : Runtime

Definition Version: 1

Version Tag: null

Definition ID: Process_AddBook:1.06...

Definition Key: Process_AddBook

Definition Name: AddBook

History Time To Live: null

Tenant ID: 1

Deployment ID: 0605953e-3ec1-11e9-a...

Instances Running:

- current version: 6
- all versions: 6

Process Instances Incidents Called Process Definitions Job Definitions

Add criteria 6 🔗 🏠 ▾

State	ID	Start Time ▾	Business Key
✓	77da282f-3efb-11e9-a933-7c7a91318270	2019-03-05T11:02:18	demo 5/21/1191900
✓	21a9d4ad-3ec8-11e9-93d4-7c7a91318270	2019-03-05T04:54:49	
✓	aa713310-3ec7-11e9-93d4-7c7a91318270	2019-03-05T04:51:29	

Date and Time displayed in local timezone: Asia/Saigon Powered by camunda BPM / v7.9.0

Hình 3.15 –Giao diện của Camunda Cockpit

Camunda Tasklist

Keyboard Shortcuts
Create task
Start process
Demo Demo

Create a filter +
Created +

My Tasks (6)
My Group Tasks
Accounting
John's Tasks
Mary's Tasks
Peter's Tasks
All Tasks

Filter Tasks
6

Book Name Entor
BorrowBook
Demo Demo
Created 6 days ago
50

Book Name Entor
BorrowBook
Demo Demo
Created 6 days ago
50

Book Name Entor
BorrowBook
Demo Demo
Created 6 days ago
50

Book Name Entor
BorrowBook
Demo Demo
Created 6 days ago
50

Assign Reviewer
Review Invoice
Demo Demo
Created 6 days ago
50
Invoice ... 10.99
Invoice ... PSARF-5342

Assign Reviewer

Book Name Entor
BorrowBook
Set foll...
Set du...
Add groups
Demo ...

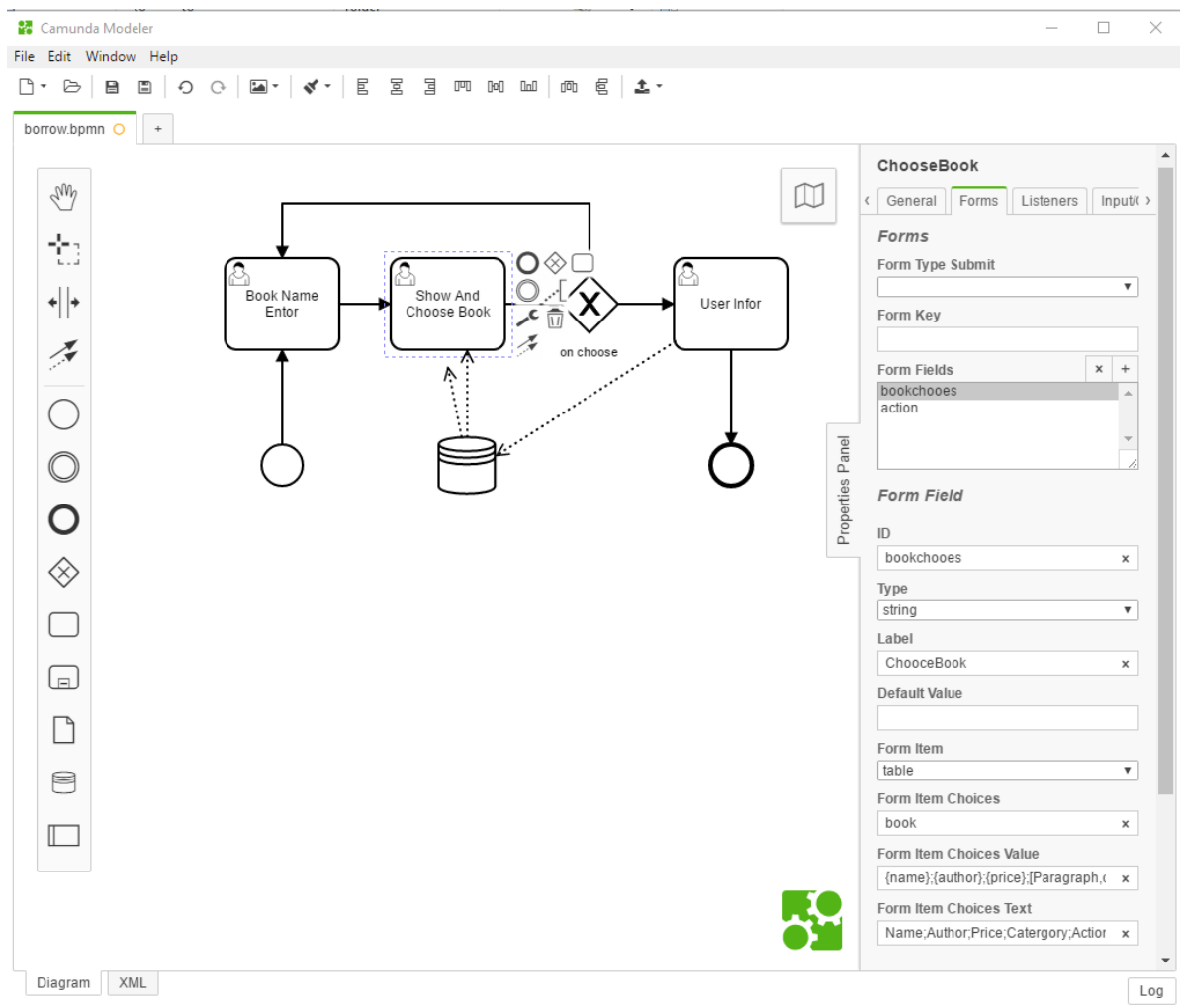
Form
History
Diagram
Description

Name

Save
Complete
Complete

Date and Time displayed in local timezone: Asia/Saigon
Powered by camunda BPM / v7.9.0

Hình 3.16 – Giao diện của Camunda Tasklist



Hình 3.17 – Giao diện của Camunda Modeller

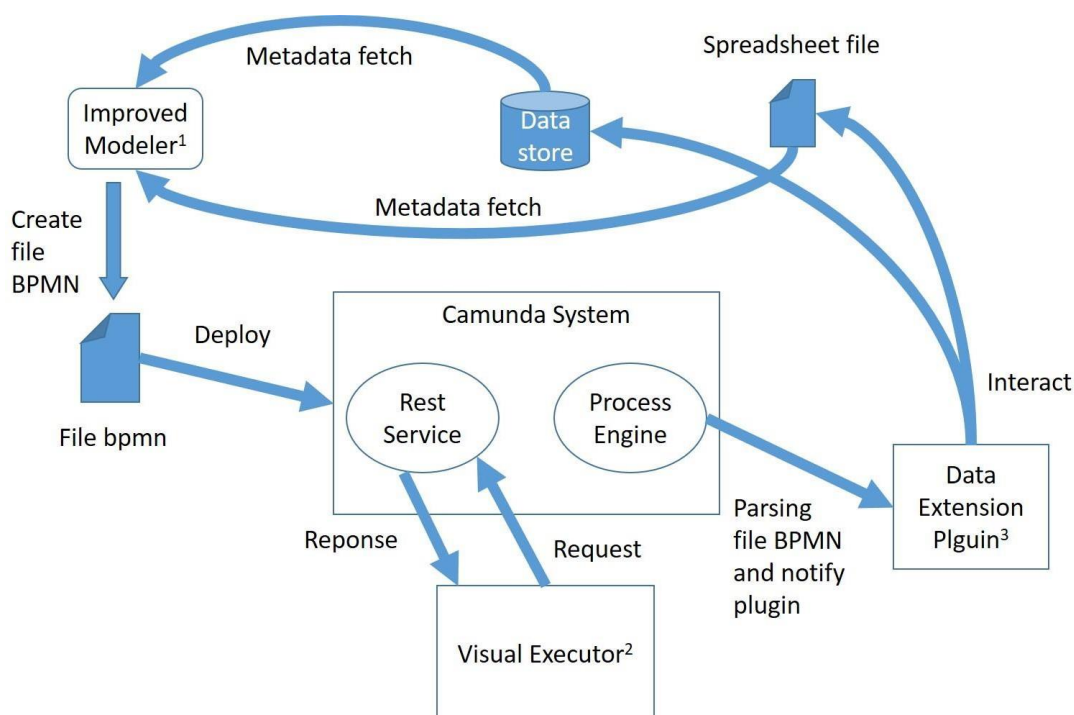
3.3. Camunda Database

3.3.1. Giới thiệu

Với hệ thống Camunda hiện tại vẫn còn tồn tại một số thiếu sót. Đó là trong quá trình thực thi một quy trình nghiệp vụ, đòi hỏi phải có nhu cầu lưu trữ thông tin của nghiệp vụ vào cơ sở dữ liệu hoặc một dịch vụ lưu trữ nào đó. Nhận thấy điều đó, 2 anh Huỳnh Tấn Phát và Nguyễn Chí Thành đã phát triển hệ thống Camunda-Database để tiến hành thực hiện thêm lưu trữ thông tin của nghiệp vụ vào cơ sở dữ liệu hoặc một dịch vụ nào đó

3.3.2. Sơ đồ thực hiện tổng quát của hệ thống Camunda-database

Sau đây em sẽ trình bày sơ đồ của hệ thống Camunda-Database phát triển.



Hình 3.18 - Sơ đồ thực hiện của hệ thống Camunda-Database

Trong đó (1), (2), (3) là những phần đã thực hiện cải tiến hoặc cài đặt trong Camunda-Database.

Camunda Modeler sẽ lấy thông tin từ Data Source để mô tả các quy trình và tạo ra tập tin dưới dạng .bpmn. Người sử dụng sẽ deploy các tập tin này thành các quy trình và sử dụng các quy trình này thông qua REST API. Sau khi các quy trình được tạo ra hoặc triển khai trên Camunda System,

Process Engine sẽ kêu gọi Data Extension Plugin(DEP) để tiến hành chạy. Hiện tại DEP chỉ hoạt động trong khuôn khổ tương tác với User Task.

Sau khi DEP kích hoạt, DEP sẽ lấy thông tin từ BPMN thông qua Process Engine mà Camunda cung cấp, từ đó sẽ thiết lập các câu truy vấn xuống nguồn dữ liệu.

Ngoài ra, do hệ thống Camunda Tasklist gây khó khăn cho việc thực thi quy trình, giao diện rườm rà, và khó tương tác cho người sử dụng. Nên nhóm trước khi tiến hành cải tiến Camunda thành Camunda-Database để thêm vào Visual Executor có vai trò thể hiện lại các thông tin đã được hiển thị trên Camunda Tasklist. Nhằm mục đích bỏ qua một vài bước để loại sự rườm rà khi cần tạo một quy trình cũng như cải thiện giao diện cho người dùng tương tác dễ

3.3.3. Các thành phần đã mở rộng trong Camunda-database.

3.3.3.1. Mở rộng hệ thống mô hình hóa BPMN

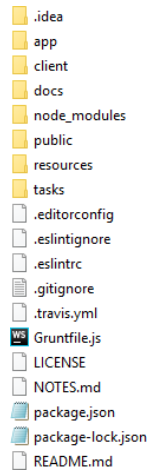
3.3.3.1.1. Giới thiệu về Camunda Modeler

Camunda Modeler được viết trên nền tảng Web, sử dụng Nodejs kết hợp với Electron để tạo thành một ứng dụng chạy độc lập như ứng dụng Desktop mà không cần đến Web Browser.

Camunda Modeler là một công cụ mã nguồn mở viết bằng JavaScript, cho phép người dùng có thể viết Plugin mở rộng cho công cụ này. Tuy nhiên, Plugin này chỉ mở rộng cho phần Client và không hỗ trợ các Plugin xử lý trên Server và với ngữ cảnh hiện tại, Camunda-Database đã mở rộng tính năng mô hình hóa lại các thao tác trên Server và thao tác trên Google Spreadsheet đòi hỏi cần phải phân tách cách thức xử lý tại Server và Client nên đã quyết định

không sử dụng Plugin mà dùng phương pháp thêm một số Module vào mã nguồn của công cụ.

3.3.3.1.2. Cấu trúc các tập tin mã nguồn của công cụ Modeler



Hình 3.19 – Cấu trúc của các thư mục trong Modeler

Như em đã trình bày, công cụ mô hình hóa BPMN này là một ứng dụng dưới dạng Desktop nhưng bản chất của nó hoạt động dựa trên nền tảng web, sử dụng Electron và được chia tách thành hai phần, đó là:

- Client: được tổ chức trong thư mục client, có chức năng hỗ trợ người dùng trong việc thao tác với các thành phần trong BPMN, client sử dụng Grunt để build các file JavaScript thành một tập tin được đặt trong thư mục public.
- Server: được tổ chức trong thư mục app, có nhiệm vụ xử lý các sự kiện liên quan đến hoạt động của ứng dụng như: các thao tác trên cửa sổ, các thao tác trên menu chức năng,...

Khi Electron chạy sẽ tìm file JavaScript chính được chỉ định từ từ package.json để chạy. Mã nguồn trong thư mục client được build vào thư mục public, file javascript này load các tập tin trên Client tại thư mục public và tiến hành khởi chạy ứng dụng. File Gruntfile.js mô tả cấu hình giúp build toàn bộ mã nguồn thành một ứng dụng trên desktop

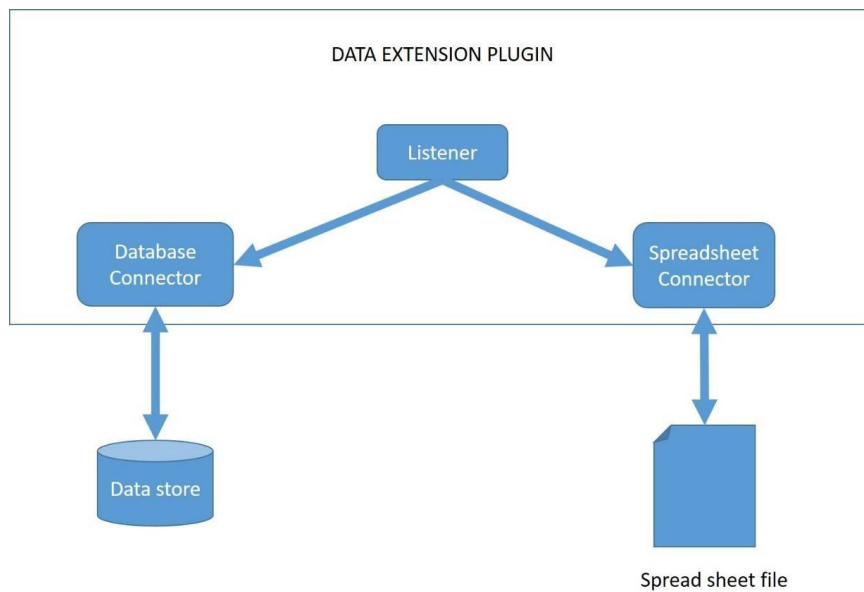
3.3.3.2. Mở rộng hệ thống vận hành quy trình bằng DEP

3.3.3.2.1. Giới thiệu về DEP

Camunda có cung cấp cơ chế cho phép người dùng có thể tự tạo ra Plugin để có thể mở rộng và tùy chỉnh cách mà Camunda hoạt động. Hệ thống Camunda-Database mở rộng cài đặt thêm cho phần cơ sở dữ liệu và thêm dịch vụ dữ liệu cho Google Spreadsheet thông qua cơ chế này. Phương thức thực hiện sẽ chia ra làm hai luồng chính:

- Phục vụ cho tác vụ ghi, cập nhật, xóa bỏ các dòng dữ liệu trong nguồn dữ liệu.
- Phục vụ cho tác vụ đọc thông tin các dòng dữ liệu trong nguồn dữ liệu

3.3.3.2.2. Mô hình kiến trúc của DEP



Hình 3.20 - Mô hình Data Extension Plugin (DEP)

Đây là kiến trúc logic tổng quan được thiết kế cho DEP. Trong đó:

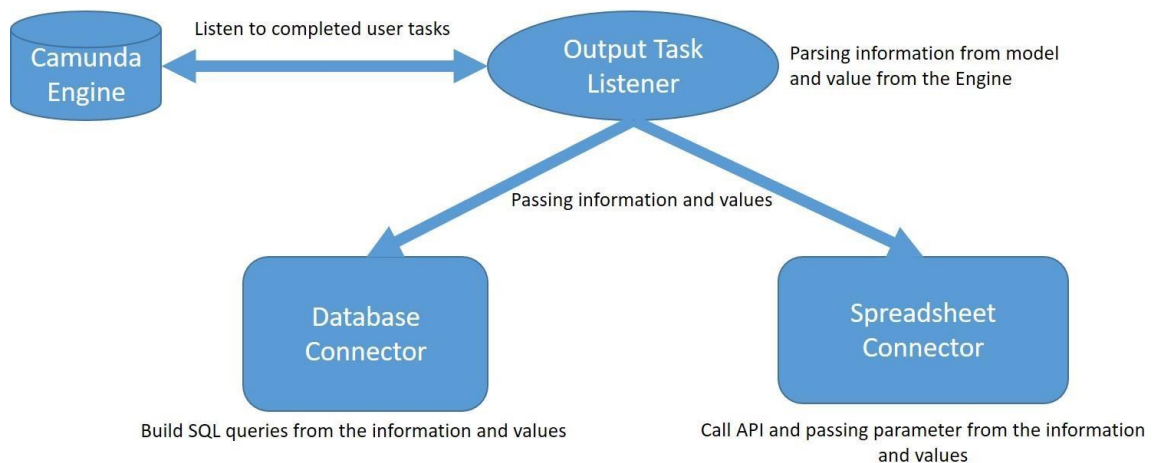
Listener: Khi người dùng thực thi nghiệp vụ, Process Engine sẽ thông báo với các Plugin thông qua Listener. Listener này sẽ lấy các thông tin từ tập tin .bpmn cung cấp cho Connector để có thể truy xuất xuống nguồn dữ liệu

Database Connector: Dùng để thao tác dữ liệu với các cơ sở dữ liệu

Spreadsheet Connector: Dùng để thao tác với Google Spreadsheet thông qua các API của Google.

3.3.3.2.3. Cách hoạt động của DEP.

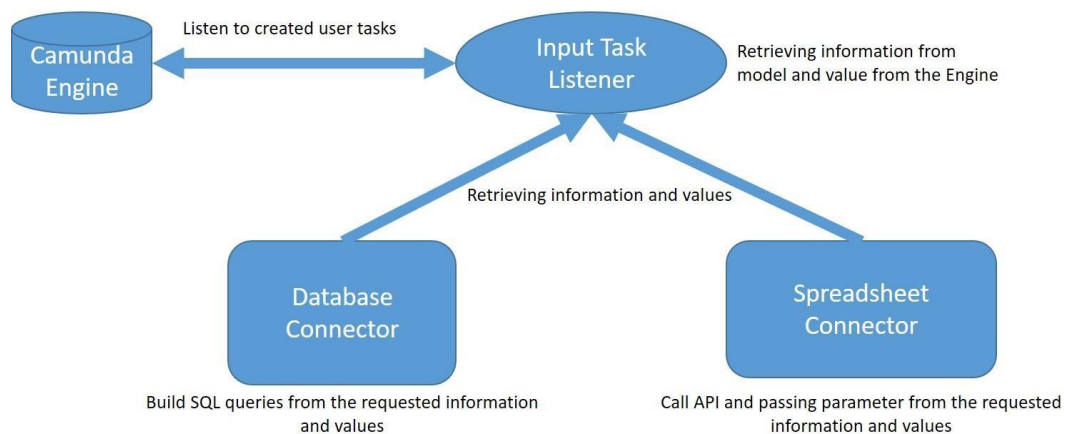
Xử lý ghi, cập nhật, xóa bỏ dữ liệu:



Hình 3.21 – Luồng xử lý của việc ghi, cập nhật và xóa bỏ dữ liệu

Theo Hình 3.21, khi người dùng hoàn tất một tác vụ, Process Engine sẽ thông báo sự kiện cho Output Task Listener. Output Task Listener sẽ chuyển đổi các thông tin và giá trị từ Form Service và tập tin .bpmn rồi sau đó chuyển thành các đối tượng của lớp Data Condition và Data Store Field. Các thông tin này sẽ nhờ Database Connector biến thành các câu truy vấn hoặc gọi các API tùy theo yêu cầu của mô hình người dùng triển khai trên Process Engine.

Xử lý đọc dữ liệu:



Hình 3. 22 – Luồng xử lý của việc đọc dữ liệu

Hình 3.22 biểu diễn cho việc lắng nghe sự kiện các tác vụ “mới khởi tạo” thay vì “mới hoàn thành” giống Hình 3.21. Về những thành phần thì đều giống như Hình 3.21

3.3.4. Những khuyết điểm trong hệ thống Camunda-Database

Mặc dù, Camunda-Database đã cải thiện một phần của Camunda, đó là đã tiến hành thực hiện thêm lưu trữ thông tin của nghiệp vụ vào cơ sở dữ liệu hoặc một dịch vụ nào đó. Tuy nhiên, quá trình hiện thực hóa việc thực thi một quy trình nghiệp vụ bằng cách định nghĩa và thực thi hệ thống, hầu như không có một hệ thống nào hỗ trợ để tạo ra một trang web tương ứng cùng với các field form mà người dùng có thể designer theo hình thức “kéo và chạy”. Camunda-Database hiện tại chỉ có thể render form hoặc cho người dùng truyền vào form cố định bằng cách sử dụng User Task. Và hệ thống chưa tạo ra được hệ thống CMS để quản lý những nghiệp vụ dưới dạng website.

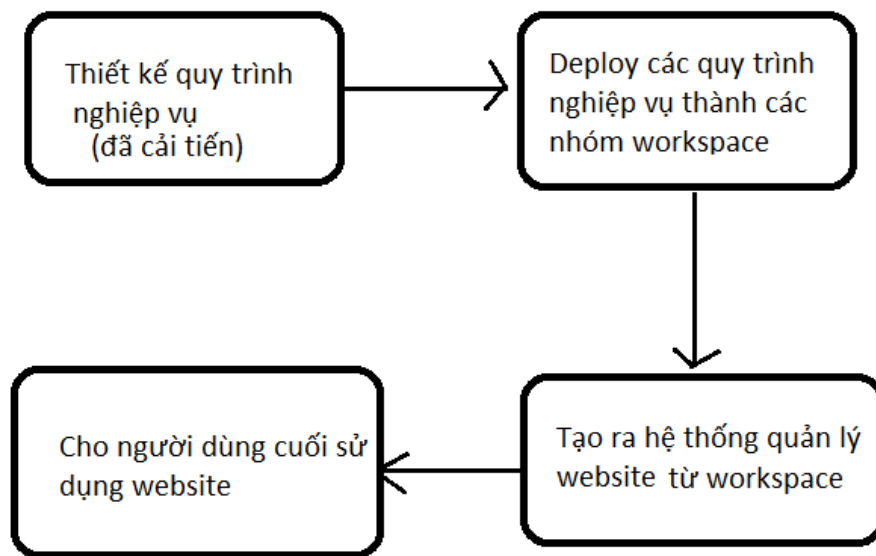
CHƯƠNG 4: HỆ THỐNG ĐỀ XUẤT

Trong chương này, em sẽ phân tích những mong muốn về hệ thống sau khi tìm hiểu về công cụ có sẵn hiện nay và tham khảo hệ thống Camunda-Database. Từ đó, đưa ra những giải pháp và thiết kế mới đáp ứng được nhu cầu về tính năng cũng như tính khả dụng trong việc phát triển các hệ thống quản lý

4.1. Giới thiệu về hệ thống

Như đã trình bày ở phần trên, nhận thấy sự cần thiết phải thay đổi nên em quyết định cải tiến những tính năng mới dựa vào hệ thống Camunda-Database kết hợp với những hệ thống quản lý nội dung (gọi tắt là CMS – Content Management System) hiện có. Mỗi hệ thống CMS sẽ quản trị những quy trình nghiệp có liên quan với nhau gọi là Workspace. Các nghiệp vụ trong hệ thống CMS sẽ có thể tùy chỉnh được giao diện và các biểu mẫu khi thực thi quy trình nghiệp vụ. Ngoài ra cần phải kiểm tra nghiệp vụ đang thực thi có thuộc về hệ thống CMS đó quản lý không, cần phải có thêm hệ thống để quản lý những workspace với CMS này. Tương ứng với những nghiệp vụ trong workspace, là những nghiệp vụ trong hệ thống CMS. Nếu một quy trình nghiệp vụ được thực thi từ hệ thống CMS bất kỳ mà không có trong thông tin trong workspace quản lý hệ thống CMS đó. Thì nghiệp vụ đó sẽ không được thực thi.

Hình 4.1 mô tả rõ các quy trình tính năng của hệ thống mà em đề xuất.



Hình 4.1 - Mô tả các tính năng của hệ thống đề xuất

4.2. Hướng tiếp cận xây dựng hệ thống

Để xây dựng được hệ thống ở trên, em sẽ tiến hành:

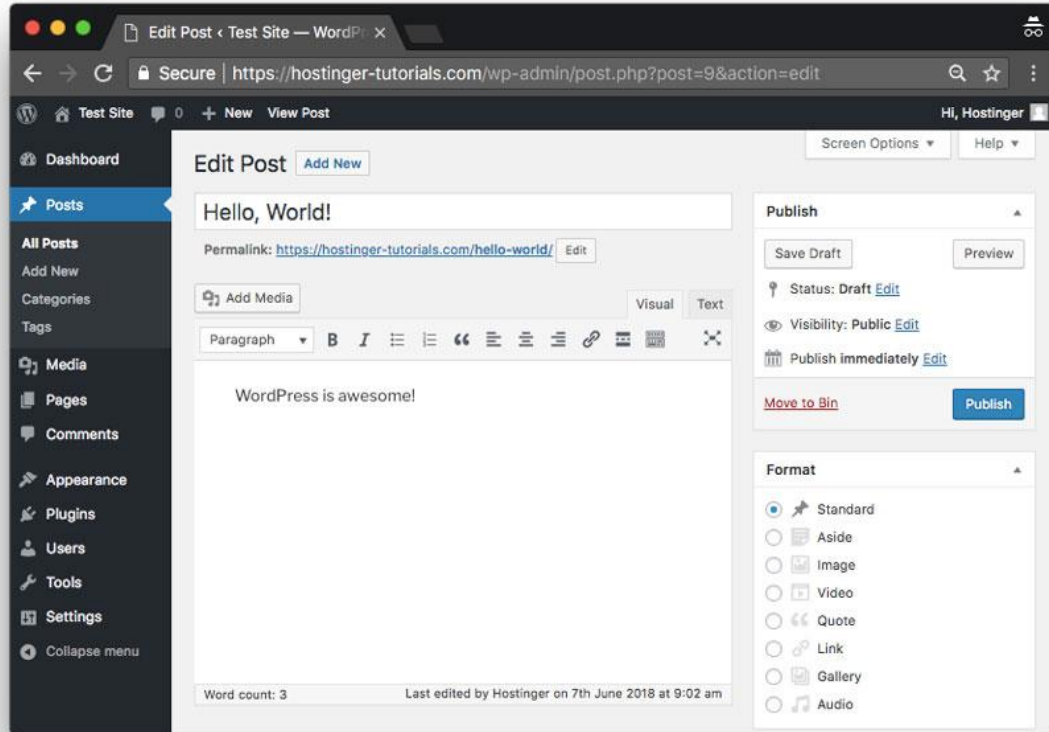
- Cải tiến hệ thống Camunda-Database
- Tiếp tục cải tiến công cụ mô hình hóa bằng BPMN thông qua hệ thống Camunda-Modeler nhằm giúp hỗ trợ người dùng có thêm mô tả chi tiết về thông tin của cơ sở dữ liệu. Khi thay đổi công cụ mô hình hóa đồng nghĩa với việc phải thay đổi cách thức thực thi một quy trình. Em sẽ thêm những tính năng xử lý sau khi cải tiến vào Process Engine của Camunda.
- Xây dựng hệ thống quản lý các workspace và tạo ra hệ thống CMS quản lý được các quy trình nghiệp vụ. Đồng thời hệ thống này sẽ đóng vai trò trung gian giữa CMS và Camunda.
- Thay đổi giao diện người dùng khi thực thi một quy trình nghiệp vụ. Mục đích giúp người dùng có các thao tác sử dụng đơn giản, dễ hiểu.

4.3. Lựa chọn hệ thống CMS

Có rất nhiều phần mềm hỗ trợ người dùng thiết kế trang Web, có thể được kể đến như Wordpress, Joomla và Drupal. Trong phần này em sẽ cung cấp phân tích chi tiết về cả ba hệ thống quản lý nội dung cũng như điểm mạnh của chúng.

❖ WordPress

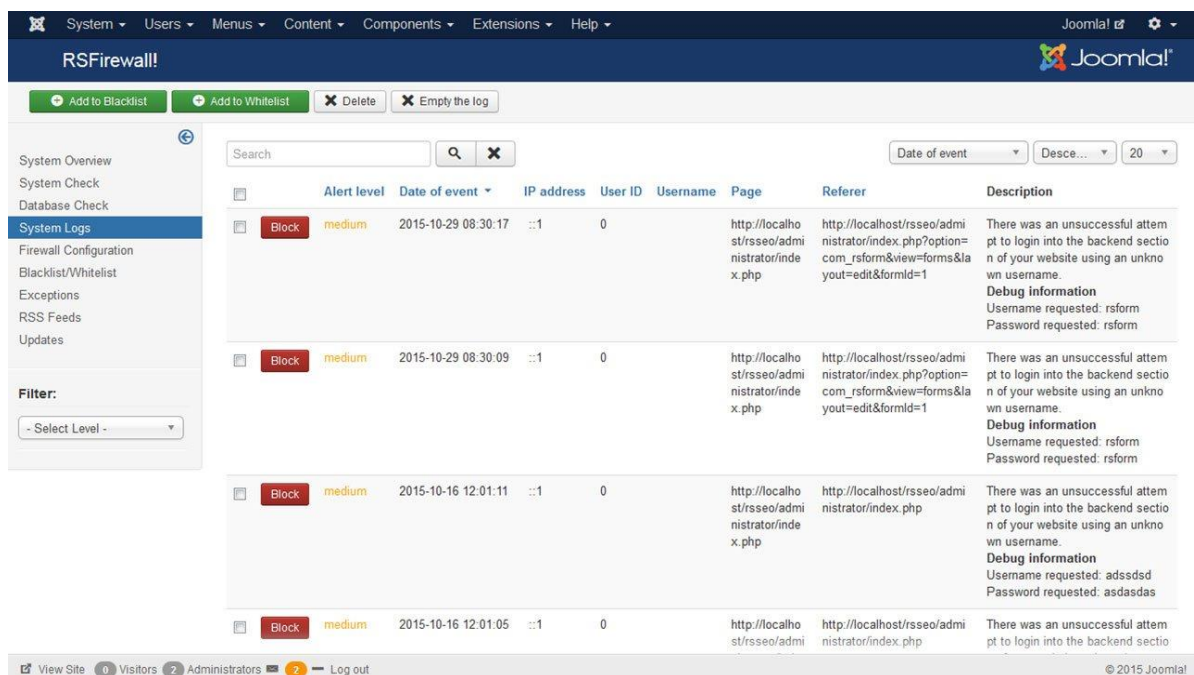
- Hỗ trợ người dùng tạo ra nhiều thể loại trang web khác nhau như: blog, giới thiệu doanh nghiệp, bán hàng – thương mại điện tử
- Dễ sử dụng
- Tổ chức nội dung sử dụng menu trang cũng rất đơn giản. Cho phép người dùng của bạn khi ghé thăm trang web có thể tìm kiếm thông tin một cách dễ dàng.
- Cách tạo một trang template dễ dàng. Có thể sử dụng thêm Custom Field cho trang để nhận giá trị người dùng nhập vào và tự động áp dụng trong code
- Hỗ trợ sẵn một hệ thống xác định chức vụ của thành viên và mỗi chức vụ sẽ được định sẵn các quyền mà WordPress đã làm sẵn



Hình 4.2 - Giao diện của Wordpress

❖ Joomla

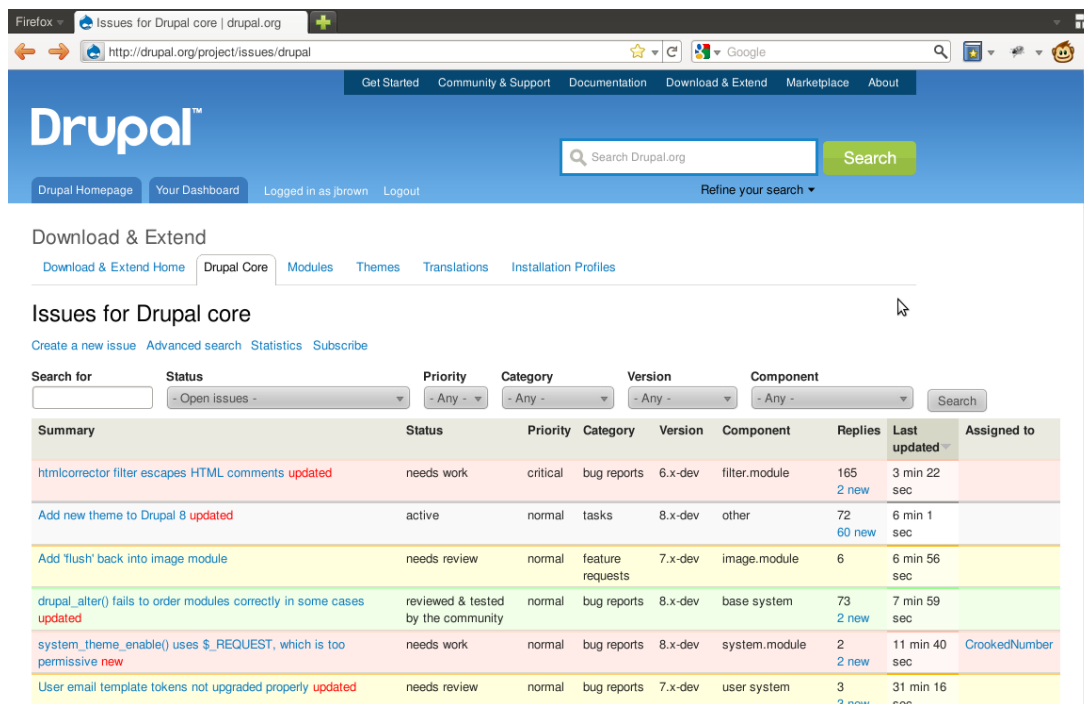
- Quản lý các trang web phức tạp tốt hơn
- Phức tạp hơn so với WordPress và không có nhiều giao diện hay plugin để lựa chọn
- Mặc định đã hỗ trợ rất nhiều View. Việc override các view này cũng tương đối dễ dàng, nó cho phép thay đổi style, structure của layout.
- Quản lý phân quyền bao gồm: Quản lý User, quản lý Group, quản lý quyền



Hình 4.3 - Giao diện của Joomla

❖ Drupal

- Là CMS phức tạp nhất.
- Là hệ quản trị nội dung lý tưởng cho việc tạo ra các trang web rất phức tạp.
- Là một CMS rất linh hoạt với một số lượng lớn plugin có sẵn để mở rộng chức năng.
- Module Views nổi tiếng kết hợp với block visibility sẽ được sử dụng để đạt được multi-layout hay featured pages. Các View hỗ trợ UI mạnh mẽ cho phép người dùng tạo pages, blocks với nhiều kiểu hiển thị nội dung khác nhau mà không cần biết về code.
- Hỗ trợ hệ thống quản lý user bằng Roles. Mỗi role sẽ có những quyền khác nhau tùy theo cài đặt của người quản trị.



Hình 4.4 - Giao diện của Drupal

Như so sánh ở trên, mỗi hệ hống CMS đều có điểm mạnh riêng của mình, do đó em cũng khó khăn trong lựa chọn hệ thống CMS. Nhưng vì mục đích của luận văn có giới hạn là tích hợp vào hệ thống CMS, nên em chỉ lựa chọn trong tiêu chí là hệ thống đơn giản, có khả năng thêm Plugin, dễ sử dụng cho người phát triển và người sử dụng. Dựa vào những tiêu chí này, em thấy Wordpress có phần hơn hẳn.

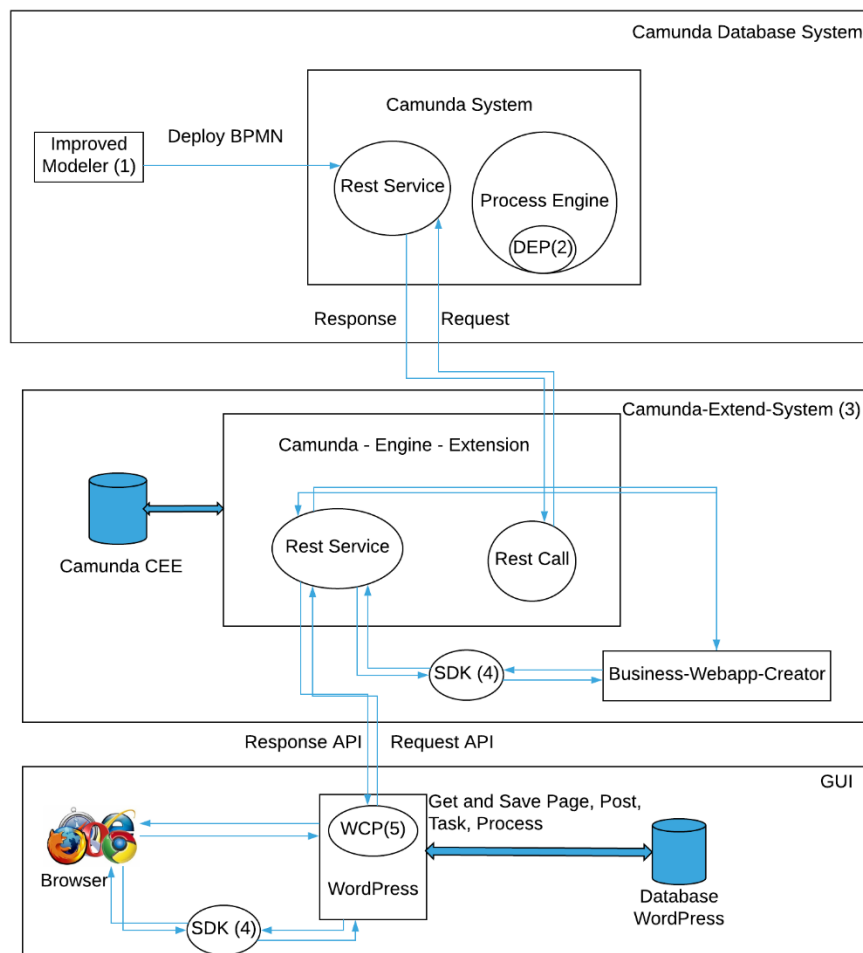
Ngoài ra, Wordpress còn có những ưu điểm như là mã nguồn mở đa năng, nhiều plugin hỗ trợ và một kho giao diện khổng lồ nên có thể xây dựng được tất cả các thể loại trang web từ nhỏ đến lớn. Với những plugin này, cho dù không có kiến thức về code hay lập trình vẫn có thể tạo được nhiều chức năng cho trang web một cách rất đơn giản.

Thêm nữa, để trang web thêm chuyên nghiệp và bắt mắt hơn thì Wordpress còn có một kho giao diện vô cùng lớn cả miễn phí và trả phí tùy theo nhu cầu và sở thích của người phát triển.

Do những ưu điểm của Wordpress nói trên, nên em quyết định chọn Wordpress làm hệ thống CMS cho hệ thống của mình.

4.4. Kiến trúc tổng quan

Sau đây, em xin được trình bày sơ đồ hệ thống em đề xuất :



Hình 4.5 - Sơ đồ thực hiện của hệ thống

Trong đó: (1), (2), (5) là những phần em đã cải tiến của hệ thống cũ và (3), (4) là phần hệ thống em đã cài đặt thêm vào.

Do cần quản lý các workspace và hệ thống CMS(Wordpress) nên em xây dựng hệ thống Camunda-Extend-System. Ngoài ra Camunda-Extend-Sytem(CES) còn đóng vai trò trung gian tiền xử lý các nghiệp vụ được thực thi từ hệ thống Wordpress cung cấp khi giao tiếp với hệ thống Camunda. Để xây dựng hệ thống CES em chia làm 2 thành phần phần chính: Camunda-Engine-Extend và Business-webapp-creator.

Camunda-Engine-Extend (CEE) làm thành phần xử lý chính của CES. CEE có nhiệm vụ tiền xử lý các quy trình nghiệp vụ trước khi đến hệ thống Camunda-Database nhằm mục đích kiểm tra các nghiệp vụ này được phép thực thi hay không. Cụ thể, vì cách thức thực thi nghiệp vụ đều thông qua những REST API, người dùng có thể thực thi nghiệp vụ đó mà không cần thông qua giao diện Wordpress cung cấp nên vì thế rất khó xác định nghiệp vụ đó là do ai đang thực thi. Do đó em xây dựng CEE để xử lý chuyện này. Ngoài ra CEE còn có nhiệm vụ giúp giao tiếp với Database CEE để lấy và lưu trữ những thông tin về wordpress và workspace.

Bussine-webapp-creator (BWC) là một ứng dụng web giúp cho người sử dụng dễ thực thi quá trình deploy các quy trình thành các workspace và từ các workspace tạo ra các wordpress.

Các ứng dụng Wordpress được deploy bởi workspace sẽ giúp cho người thiết kế có thể thiết kế các quy trình thành trang web tương ứng, thích hợp cho end-user sử dụng thông qua **Wordpress-Camunda-Plugin(WCP)**. Wordpress-Camunda-Plugin có hai nhiệm vụ chính là:

- Người thiết kế sẽ thiết kế các form tương ứng cho từng task khi thực thi từng nghiệp vụ.
- Tiếp nhận các yêu cầu thực thi nghiệp vụ của end-user, gửi đến CEE và trả lại kết quả cho end-user.

Với mục tiêu giúp cho người thiết kế có thể tùy chỉnh được giao diện form khi thực thi nghiệp vụ. Cần phải định nghĩa thêm những thuộc tính trong quá trình trình thiết kế nghiệp vụ, điều đó cũng đồng nghĩa với việc thay đổi các công cụ hỗ trợ của Camunda (Camunda Modeler) và thay đổi luôn cách thức thực thi một quy trình nghiệp vụ (cụ thể là thay đổi Process Engine). Và việc này đã được hệ thống Camunda-Database đã tiến hành thay đổi trước đó. Lợi dụng điểm đó, em sẽ tiếp tục tái sử dụng mã nguồn của hệ thống Camunda-Database để tiếp tục cải tiến cho Camunda-Modeler và Process Engine.

Em sẽ trình bày mô tả chi tiết cài đặt cụ thể của từng thành phần ở chương sau.

CHƯƠNG 5: QUÁ TRÌNH CÀI ĐẶT

Trong chương này, em sẽ trình bày chi tiết về những kỹ thuật trong quá trình xây dựng hệ thống.

5.1. Mở rộng trên hệ thống Camunda-Database

5.1.1. Mở rộng định nghĩa bổ sung cho việc mô hình hóa BPMN

Như em đã đề cập ở chương trước, để phục vụ cho việc mở rộng cho việc render form và truy xuất cơ sở dữ liệu, em đã thêm một số thuộc tính vào UserTask của BPMN được mô tả chi tiết như sau:

Thuộc tính	Ý nghĩa
Form Item	Tương ứng cho từng form control trong việc render form
Form Item Choices	Dùng để map với Variable khác trong quá trình thực thi nghiệp vụ.
Form Item Value	Làm key cho Form Item Choices
Form Item Text	Thể hiện trên giao diện người dùng
Name Variables	Gán kết quả trả về từ database vào Variable
Type Ouput	Biểu diễn kiểu trả về của Variables khi truy xuất từ database
Type Submit	Biểu diễn những cách thức submit của form

Bảng 5. 1 – Chi tiết các thuộc tính thêm vào UserTask.

Sở dĩ em phải thêm vào các thuộc tính như trên là do quá trình render form cần phải map primary key với foreign key ở cơ sở dữ liệu

5.1.2. Mở rộng hệ thống Camunda-Modeler

Module chính dùng để thêm các thuộc tính vào BPMN đó là Provider, tại đây Provider sẽ định nghĩa các Tab dùng trong BPMN, với từng Tab, Provider định nghĩa các Property cho từng Tab, các Property này có thể là một

textbox, combobox, table,...được gọi là một Entry, mỗi Entry có chứa dữ liệu HTML phục vụ cho việc hiển thị giao diện các thuộc tính của một thành phần BPMN. Mỗi Entry trong Camunda Modeler có thể cài đặt các phương thức get và set, trong đó, phương thức get dùng để đọc các thuộc tính từ tập tin lưu trữ XML để hiển thị vào các thuộc tính, phương thức set dùng để xử lý các thao tác dùng để thêm một the vào tập tin XML lưu trữ dữ liệu của mô hình BPMN. Dựa vào các Provider cung cấp sẵn của Camunda , em đã thêm vào Property đã đề cập trên bảng 5.1

```
if (is(element, "bpmn:DataInputAssociation")) {
    if (!attr.action) {
        attr.action = 'select';
    }
    var action = entryFactory.comboBoxField({
        id: 'action',
        description: 'Choose Action With Your Database',
        label: 'Action',
        modelProperty: 'action',
        selectOptions: [
            { name: 'select', value: 'Select' },
            { name: 'procedure', value: 'Procedure' },
            { name: 'query', value: 'Query' }
        ],
        attributes: attr.action
    });
    action.get = getValue(getBusinessObject(element));
    action.set = setValue(getBusinessObject(element));
    group.entries.push(action);
}
```

Mã nguồn 5.1 - Thêm procedure và query vào DataInputAssociation

```
// [FormData] form field formItem combo box
group.entries.push(entryFactory.comboBox({
    id: 'form-field-form-item',
    label: translate('Form Item'),
    selectOptions: [
        { name: 'input', value: 'Input' },
```

```

        { name: 'paragraph', value: 'Paragraph' },
        { name: 'select', value: 'Select' },
        { name: 'table', value: 'Table' },
        { name: 'button', value: 'Button' },
        { name: 'uploadfile', value: 'UploadFile' },
        { name: 'loadimage', value: 'LoadImage' },
        { name: 'link', value: 'Link' },
    ],
    modelProperty: 'formItem',
    emptyParameter: true,
    get: function (element, node) {...
    },
    set: function (element, values, node) {...
    },
    hidden: function (element, node) {...
    }
  }));

```

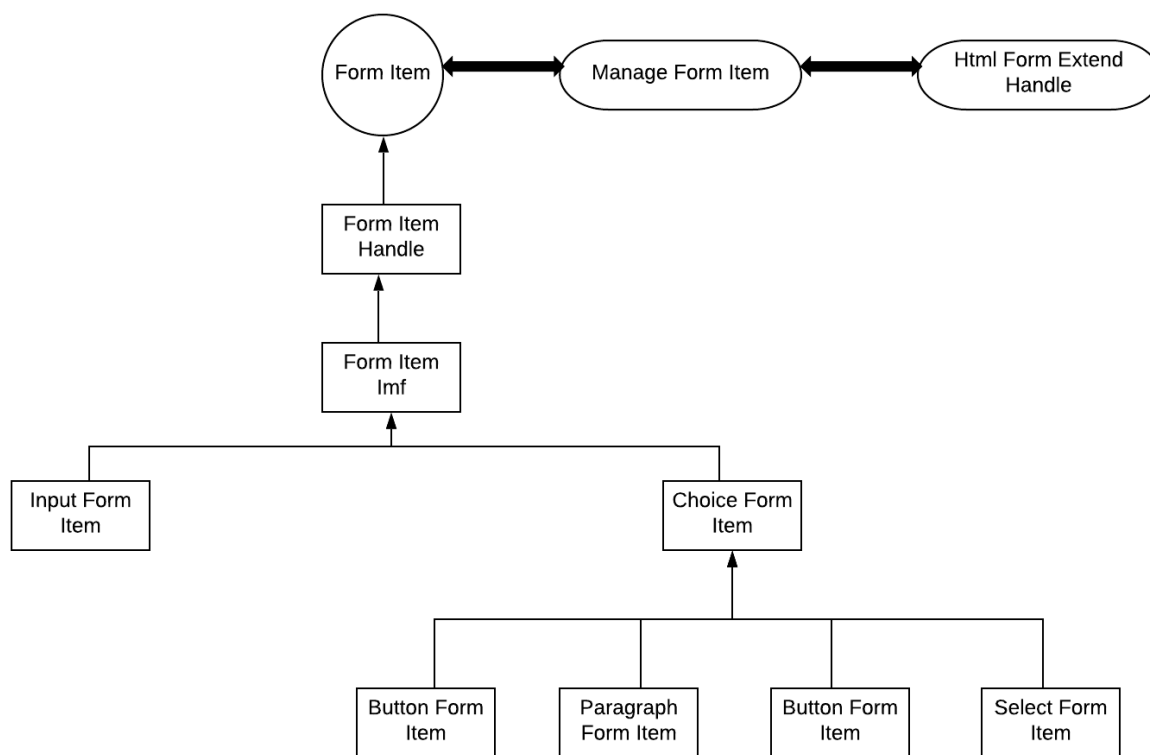
Mã nguồn 5.2 - Thêm thuộc tính form-item vào User Task

Trong mã nguồn 5.1 và 5.2 có entryFactory, đây là thành phần tạo ra các entry giúp cho hiển thị các thuộc tính trên giao diện Modeler.

5.1.3. Mở rộng Process-Engine (tái sử dụng DEP)

Thay đổi công cụ cũng đồng nghĩa với phải thay đổi vận hành quy trình của camunda. Nên em tiếp tục mở rộng thêm phần quy trình được vận hành bằng DEP của Camunda-Database. Ở phần DEP này em chia làm hai phần để phù hợp cho kiến trúc của em.

- **Listener:** Có nhiệm vụ thực thi InputTask và OutputTask. Về phần này do em chỉ thêm thuộc tính vào nên em trực tiếp thay đổi mã nguồn của Camunda-Database.
- **TaskForm:** có nhiệm vụ render form trong quá trình thực thi nghiệp vụ. Để phù hợp cho việc phát triển sau này, em đã đề xuất ra kiến trúc như sau:



Hình 5.1 - Mô tả kiến trúc cho việc render form.

Khi TaskForm được thực hiện, nó sẽ kêu gọi qua HtmlFormHandle để bắt đầu quá trình render Form. Do đó em tạo HtmlExtendFormHandle kế thừa HtmlFormHandle để có thể control được việc Render Form. Để phù hợp phát triển hay vì để HtmlExtendFormHanle render Form thì em sẽ để cho từng FormItem từ render form, và thông qua ManagerFormItem. Sau này khi phát triển chỉ cần kế thừa lại FormItem và định nghĩa lại key trong ManagerFormItem là có thể điều khiển lại render form cho phù hợp.

Mã nguồn 5.3 là ví dụ cho việc thêm ButtonFormItem vào:

```

public class ButtonFormItem extends ChoiceFormItem {
    public static String itemName = "Button";

    public ButtonFormItem() {
        super(itemName);
    }

    @Override
    public void renderExtendFormField(ExtendFormField formField, HtmlDocumentBuilder documentBuilder,
        FormData formData) {
        HtmlElementWriter divElement = new HtmlElementWriter(DIV_ELEMENT).attribute(CLASS_ATTRIBUTE, FORM_GROUP_CLASS);
        documentBuilder.startElement(divElement);

        String formFieldId = formField.getId();
        String formFieldLabel = formField.getLabel();

        // write label
        if (formFieldLabel != null && !formFieldLabel.isEmpty()) {
            HtmlElementWriter labelElement = new HtmlElementWriter(LABEL_ELEMENT).attribute(FOR_ATTRIBUTE, formFieldId)
                .attribute(CLASS_ATTRIBUTE, "hidden type-label").textContent(formFieldLabel);

            // <label for="...">...</label>
            documentBuilder.startElement(labelElement).endElement();
        }
        HtmlElementWriter button = new HtmlElementWriter(BUTTON_ELEMENT, false);
        if (!formField.isBusinessKey()) {
            button.attribute(CAM_VARIABLE_TYPE_ATTRIBUTE, formField.getTypeName())
                .attribute(CAM_VARIABLE_NAME_ATTRIBUTE, formFieldId).textContent(formField.getLabel());
            if (formField.getDefaultValue() != null)
                button.attribute(VALUE_ATTRIBUTE, formField.getDefaultValue().toString());

            button.attribute(CLASS_ATTRIBUTE, "button templatemo-blue-button").attribute(TYPE_ATTRIBUTE, "submit")
                .attribute(NAME_ATTRIBUTE, formFieldId);
        }
        documentBuilder.startElement(button);

        // </select>
        documentBuilder.endElement();

        documentBuilder.endElement();

        // TODO Auto-generated method stub
    }

    @Override

```

Mã nguồn 5.3 - Thêm ButtonFormItem vào formItem.

5.2 Mở rộng Camunda SDK

5.2.1 Giới thiệu về Camunda SDK

Camunda cung cấp một SDK được viết bằng grunt dưới ngôn ngữ javascript dùng để kêu gọi các REST API để thực thi các quy trình nghiệp vụ. Trong đó SDK camunda cung cấp cơ chế truy cập vào Resource của Camunda đồng thời cho chúng ta truy vấn được các nguồn tài nguyên này. Camunda Modeler là một công cụ mã nguồn mở viết bằng JavaScript, có thể cho phép người dùng có thể viết Plugin mở rộng cho công cụ này

Bao gồm các chủ yếu resource mà em sử dụng:

Tên	Ý nghĩa
Process-definition	Truy cập vào các quy trình được định nghĩa và được triển khai trên Camunda
Process-instance	Truy cập vào những Instance của những quy trình đang chạy
Task	Truy cập vào các Task hiện tại trên một quy trình nào đó

Bảng 5. 2– Các tài nguyên truy cập

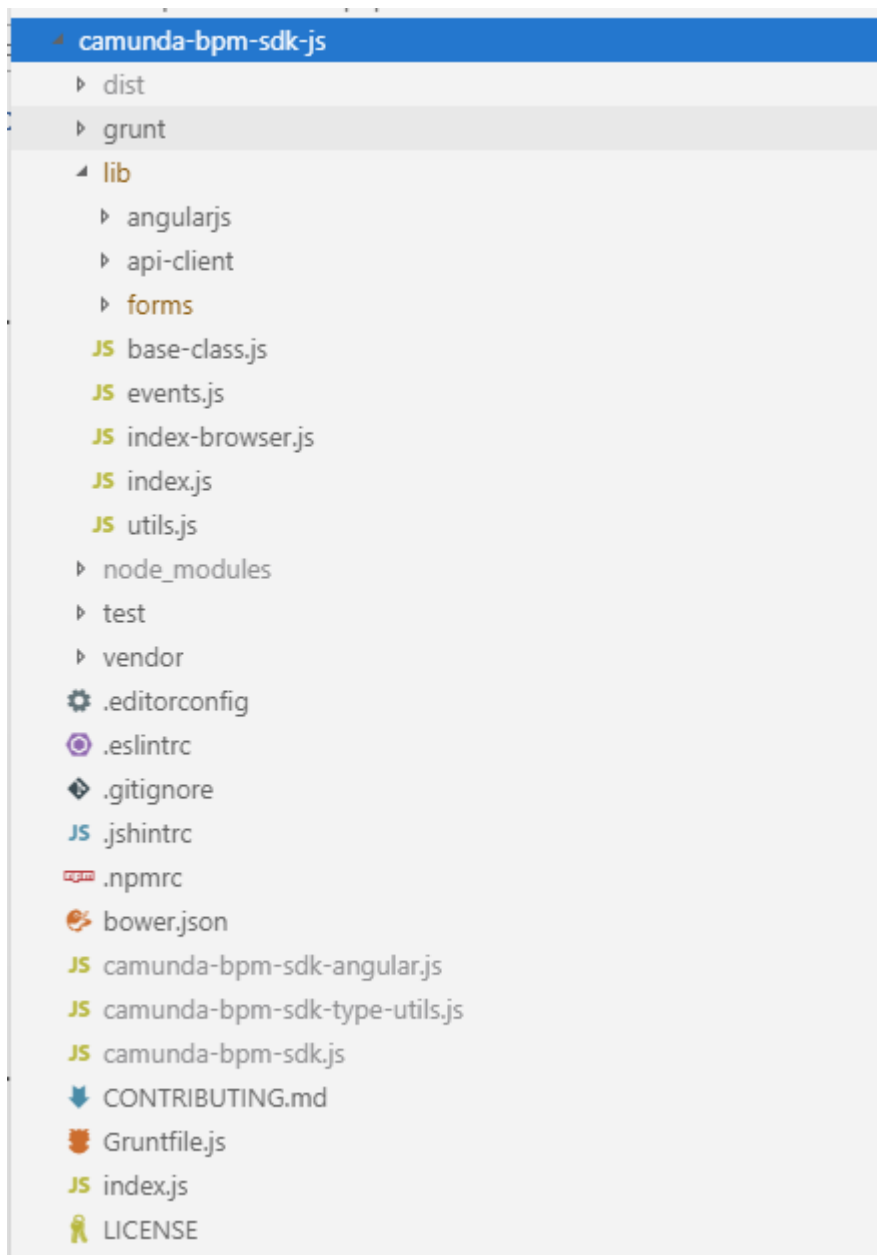
Những Resource này cung cấp những phương thức cho phép chúng ta truy vấn được dữ liệu của các nguồn tài nguyên này, những phương thức đó là:

Tên	Ý nghĩa
List	Lấy danh sách các tài nguyên theo một điều kiện cho trước
Delete	Xóa các nguồn tài nguyên theo một điều kiện cho trước
Count	Lấy số lượng danh sách tài nguyên

Bảng 5. 3– Các phương thức truy cập tài nguyên

Camunda SDK là một công cụ mã nguồn mở viết bằng JavaScript, có thể cho phép người dùng có thể mở rộng cho công cụ này. Hiện tại do em chưa tìm ra được Camunda SDK có cho plugin hay không, nên em quên quyết định theo dõi trực tiếp mã nguồn của SDK.

5.2.2 Cấu trúc thư mục Camunda SDK



Hình 5. 2 - Cấu trúc thư mục Camunda

Camunda SDK được chia làm 3 phần chính được nằm trong thư mục lib.

- Angularjs: nằm trong thư mục lib/angularjs, có nhiệm vụ cung cấp API cho người dùng và thực thi các thành phần còn lại.
- Camunda-client: nằm trong thư mục lib/api-client, cung cấp các API gửi đến ProcessEngine.
- Form: nằm trong thư mục lib/form, bao gồm các nhiệm vụ lưu trữ, get,set các dữ liệu trong các form-control thông qua các API

Khi Grunt chạy, nó sẽ tìm file JavaScript chính được chỉ định từ package.json để chạy. Và mã nguồn của thư mục sẽ build vào thư mục dist.

File Gruntfile.js mô tả cấu hình giúp build toàn bộ mã nguồn thành 3 file javascript

5.2.3 Mở rộng trên các trường form control

Để phục vụ cho việc mở rộng trong quá trình render form thêm các form control để phù hợp với datababse, em xin đề xuất thêm những trường form control trong bảng 5.3.

Tên thẻ	Ý nghĩa thẻ
Paragraph-field-handler	Dùng để mô tả cho label
Select-field-handler	Dùng để mô tả cho select
Table-field-handler	Dùng để mô tả cho trường table

Bảng 5. 4- Chi tiết các form control được thêm vào cấu trúc SDK.

BPMN cho phép chúng ta mở rộng các thêm mới các trường field-handler bằng kế thừa lại AbstractFormField của Camunda đã định ra. Có thể tham khảo dưới mã nguồn dưới đây:

```
var TableFieldHandler = AbstractFormField.extend({
    initialize: function() {...},
```

```

        applyValue: function() {...},

        getValue: function() {...}
    },
    {
        selector: 'table['+ constants.DIRECTIVE_CAM_CHOICES +']'
    }
});

```

Mã nguồn 5. 4 - Cách cài đặt table-field-handler

Trong đó hàm initialize trong mã nguồn 5.4 là hàm cài đặt , dữ liệu khi truy xuất từ ProcessEngine thông qua API sẽ được set vào form-control thông qua hàm applyValue, và lấy dữ liệu của form-control để gửi về ProcessEngine thông qua getValue(). Các trường form-control được xác trong form thông qua selector.

Để phù hợp cho việc foreign key trong cơ sở dữ liệu, em xin đề xuất thêm những “attribute” sau đây.

Tên	Giá trị	Ý nghĩa
DIRECTIVE_CAM_VALUE	Cam-value	Dùng để map với key của Cam-choices
DIRECTIVE_CAM_TEXT	Cam-text	Dùng để hiển thị trên giao diện

Bảng 5. 5- Chi tiết các attribute mới thêm vào

5.3 Cài đặt Camunda-Extend-System.

5.3.1 Phương pháp tiếp cận

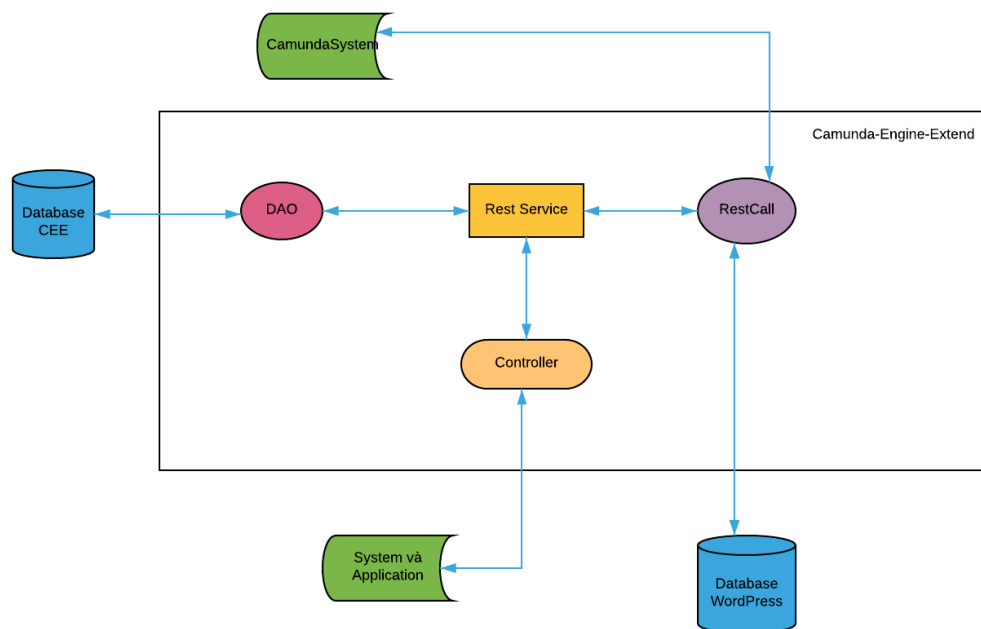
Do cần phải gom nhóm các quy trình nghiệp vụ và tạo ra các trang web tương ứng, cần có một database lưu trữ và một giao diện để xử lý dễ dàng cho designer. Và cần ràng buộc về các RestAPI nên em chia làm hai luồng xử lý chính:

- Bussine-webapp-creator: có hiển thị thông tin của các nghiệp vụ trong từng workspace, và các wordpress trong workspace. Đồng thời còn cung cấp các tương tác cho người dùng tạo các wordpress dễ dàng hơn. Ngoài ra, BWC còn có chức năng tạo ra nhanh hệ thống wordpress (wp-quick-install) mà không cần cài đặt.
- Camunda-Engine-Extend: tiếp nhận các Rest API từ Bussine-webapp-creator và Wordpress để giao tiếp xuống database và Rest Service của Camunda.

Do BWC chỉ có thực hiện dựa vào REST API để lấy thông tin từ database BWC nên phần này em không đề cập đến

5.3.2 Mô hình của Camunda-Engine-Extend

5.3.2.1 Mô hình kiến trúc của CEE

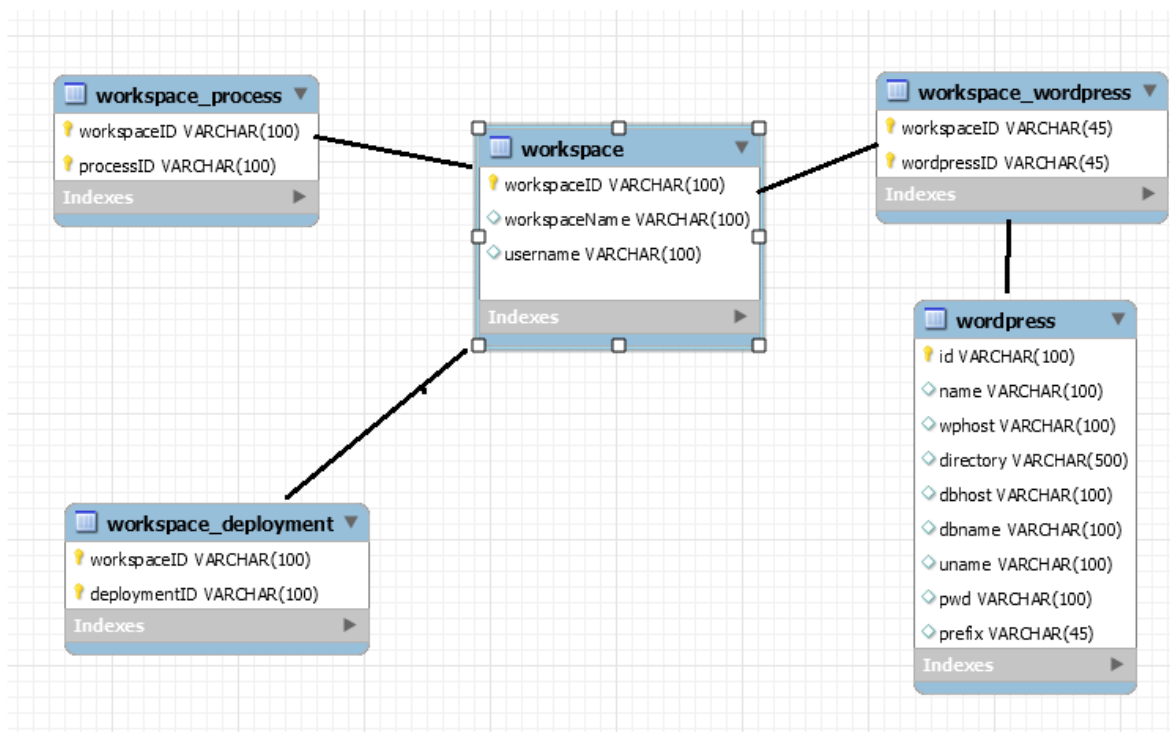


Hình 5. 3- Mô hình kiến trúc của Camunda-Engine-Extend.

Đây là kiến trúc tổng quancủa CEE. Trong đó:

- **Controller:** Dùng để lắng nghe các REST API gửi đến. Và chuyển các thông tin thành các param tương ứng, các thông tin đến rest service để xử lý, rồi nhận kết quả từ rest service để trả kết quả về.
- **Rest Service:** Dùng để lắng nghe các sự kiện từ controller đưa đến, và xử DAO để đưa hoặc lấy dữ liệu từ database hoặc sử dụng Rest Call để gửi các Request API đến RestService Của camunda.
- **DAO:** Dùng để thao tác xuống database.
- **Rest Call:** Dùng để gửi và nhận các Rest Engine của Camunda. Ngoài ra còn giao tiếp đến wordpress-quick-install để deploy một wordpress.

5.3.2.2 Mô hình database của Camunda-Extend-System



Hình 5. 4 - Mô hình database của Camunda-Extend-System.

Trong đó:

- **Workspace:** Lưu trữ thông tin của một workspace.
- **Workspace_deployment:** Mỗi một workspace sẽ có nhiều deployment và mỗi deployment sẽ có nhiều workspace, do đó em tạo quan hệ n-n và deploymentID sẽ tương ứng trong id của Deployment camunda cũ.
- **Workspace_process:** Mỗi một workspace có nhiều processDefinition và mỗi processDefinition cũng sẽ có trong nhiều workspace, do đó em tạo quan hệ n-n và processID sẽ tương ứng trong id của processDefinition camunda cũ.
- **Wordpress:** Lưu trữ thông tin của một wordpress.
- **Workspace_wordpress:** Mỗi một workspace sẽ có nhiều wordpress và mỗi wordpress cũng sẽ có trong nhiều workspace.

5.3.3 Kỹ thuật deploy nhanh Wordpress

Wordpress nổi tiếng vì dễ cài đặt, hầu hết trong các trường hợp cài đặt Wordpress chưa đầy 5 phút với những thủ công đơn giản. Những đồng thời Wordpress cũng cấp một hệ thống tạo ra Wordpress nhanh chóng, đó là wp-install-quick. Wp-install-quick cung cấp 8 bước cơ bản để tạo ra Wordpress thông qua các REST API. Các data của REST API đều giống nhau, chỉ khác nhau phần param “action”.

- **Check_before_upload:** Kiểm tra kết nối đến database có hợp lệ.
- **Download_wp:** Ở bước này hệ thống sẽ kiểm tra trong thư mục cache đã tồn tại wordpress hay chưa, nếu chưa tồn tại thì hệ thống wp-quick-install sẽ đi download mã nguồn của Wordpress về thư mục cache dưới dạng tập tin .zip

- Unzip_wp: Khi đã download mã nguồn của Wordpress về thư mục cache, ở bước này hệ thống sẽ giải nén mã nguồn Wordpress tạo ra hệ thống WordPress.
- Wp_config: Bắt đầu thêm thông tin từ database mà người dùng cung cấp vào hệ thống Wordpress.
- Install_wp: Sau khi thêm thông tin database vào, hệ thống sẽ tạo ra thêm những thông tin cần thiết và lưu trữ xuống database.
- Install_theme: Cài đặt thêm giao diện vào Wordpress.
- Install_plugins: Cài đặt thêm plugin vào Wordpress
- Success: Thông báo thành công sau.

Do đã wp-quick-install đã cung cấp sẵn các REST API cần thiết ,nên hệ thống của em đã dễ dàng có thể tạo ra nhanh Wordpress cho người sử dụng. Cụ thể là ở BWC sẽ cung cấp chức năng “Deploy Wordpress”. Khi người sử dụng, BWC sẽ tạo ra giao diện để người dùng điền thông tin cần thiết vào. Và từ những thông tin này, BWC sẽ gửi thông tin này cho CEE để CEE giao tiếp với wp-quick-install thông qua những REST API

5.4 Cài đặt Camunda-Wordpress-Plugin

5.4.1 Phương pháp tiếp cận

Wordpress có cung cấp cơ chế cho phép người dùng có thể tự tạo ra Plugin để có thể mở rộng và tùy chỉnh cách mà Wordpress hoạt động. Trong quá trình lập trình plugin, wordpress đưa ra thuật ngữ “Hook” để cho phép người lập trình thực thi một hàm nào đó vào một thời gian xác định và vị trí xác định. Wordpress cung cấp các kỹ thuật hook trong plugin bao gồm “Action hook” và “Filter hook”.

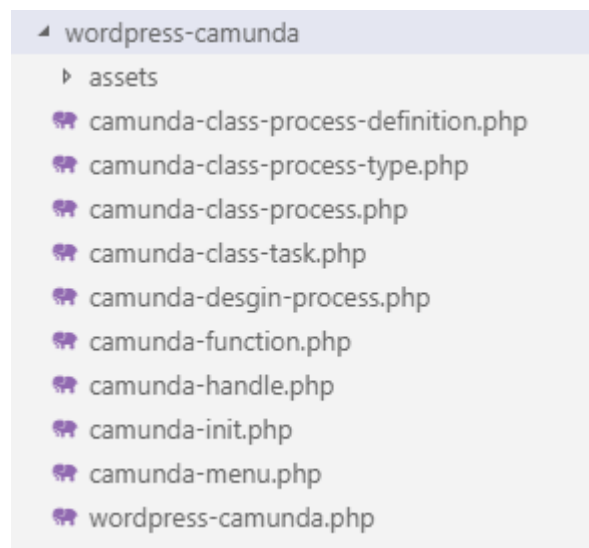
Do các Action hook và filter hook mà wordpress cung cấp cho việc lập trình plugin ,em đã quyết định mở rộng thông qua cơ chế này. Trong đó em tạo thêm một dạng PostType là ‘process’ tương tự với ‘post’.

Phương pháp chia làm hai loại chính:

- Phục vụ cho chỉnh sửa và render form của task trong các ‘process’.
- Phục vụ thực thi các của task trong các ‘process’

Sở dĩ phải phân ra làm hai hướng như vậy là do hai tính chất này độc lập và thực hiện ở hai môi trường khác nhau. Việc chỉnh sửa form của task được lưu ở wordpress, còn việc thực thi task được xử lý chính ở Camunda-Engine-Extend thông qua Rest Api.

5.4.2 Cấu trúc thư mục



Hình 5.5 - Cấu trúc thư mục của wordpress-camunda-plugin

Lý do chọn cách tổ chức thư mục này là do plugin chủ yếu là replace hoặc hook một function php do đó không có theo một kiến trúc nhất định.

Bảng dưới đây liệt kê những hàm hook trong WCP

Tên Hook	Loại	Tên hàm	Ý nghĩa
Init	Action	create_initial_process_types	Khi khởi động Wordpress hàm init sẽ kêu gọi đến hàm này để đăng ký một dạng menu là process
Admin_menu	Action	Camunda_create_menu	Khi chương trình load thành menu của admin, action Admin_menu sẽ kêu gọi hàm này để tạo thêm 1 thanh menu Camunda.
pre_get_posts	Action	add_post_type_is_process	Hàm này để đăng ký thêm loại process trong lúc thực thi câu truy vấn để lấy thông tin của bài viết.
wp_ajax_ajax_plugin_call	Action	ajax_plugin_call_callback	Đăng ký Api cho server Wordpress,
wp_ajax_nopriv_ajax_plugin_call	Action	ajax_plugin_call_callback	Giống Wp_ajax_ajax_plugin_call
wp_head	Action	my_js_variables	Thêm những biến toàn cục vào đầu trang html

Bảng 5. 6- Các function hook cơ bản trong WCP

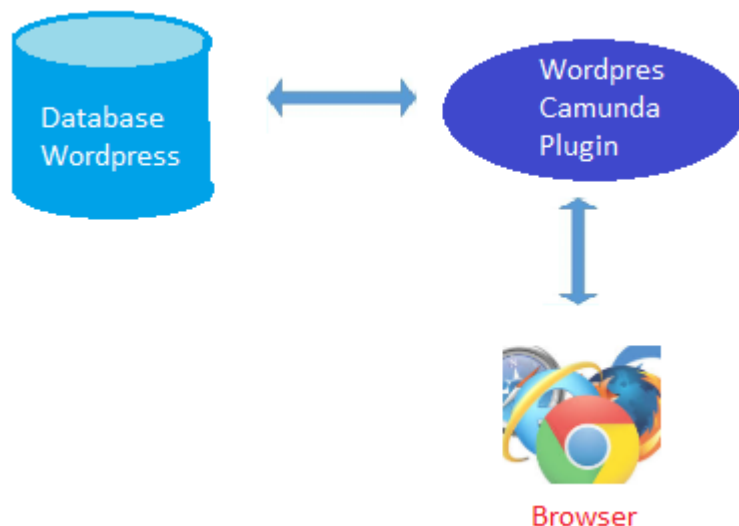
Các hàmHook trong bảng 5.6 được sử dụng trong việc tạo ra giao diện cho người sử dụng tương tác với hệ thống camunda-engine-extend.

5.4.3 Các hoạt động của WCP

Để hiểu được các hoạt động của WCP, em xin trình bày một ví dụ cụ thể là nghiệp vụ **AddBook**. Khi nghiệp vụ AddBook được deploy vào hệ thống Camunda-Extend-Sytem sẽ lưu thông tin về quy trình này vào database của Wordpress. Khi truy cập vào Wordpress, WCP có nhiệm vụ truy cập xuống database để lấy các thông tin về quy trình **AddBook** này lên(bao gồm các form task của **AddBook**). Việc chỉnh sửa các form task này được WCP hỗ trợ vào lưu lại xuống database. Khi việc chỉnh sửa đã xong, người dùng muốn thực thi quy trình **AddBook** này, thì WCP sẽ có nhiệm vụ tiếp nhận các API từ client và gửi các API này đến hệ thống Camunda-Engine-Extend để thực thi nghiệp vụ.

Như đã đề cập trên phần trên, WCP sẽ có hai luồng xử lý chính, cách thức xử lý của từng luồng được mô tả như sau.

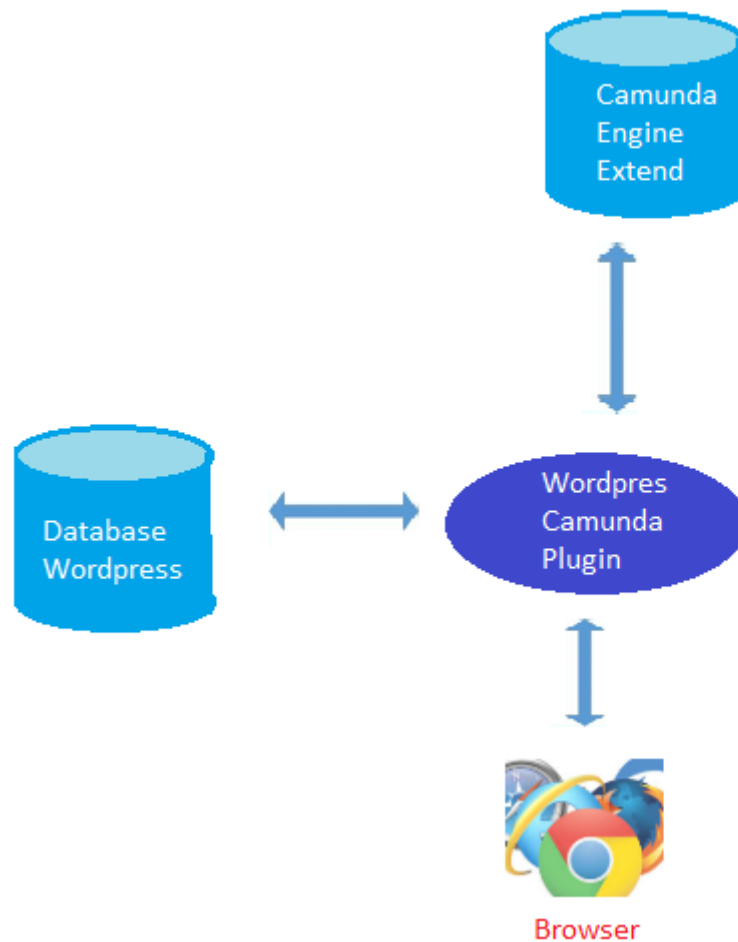
Xử lý chỉnh sửa form của task:



Hình 5.6 – Luồng xử lý việc chỉnh sửa Form

Việc xử lý chỉnh sửa form của WCP cực kì đơn giản là: lưu nguyên một khối html của task xuống database. Khi cần thì load nguyên khối html và chỉnh sửa lại rồi đưa lại cho WCP lưu xuống database lại. Việc chỉnh sửa form chủ yếu là javascript kết hợp với angularjs.

Xử lý thực thi task của ‘Process’:



Hình 5.7 - Luồng xử lý việc thực thi Task.

Do còn hạn chế việc lấy Form Task trong từng nghiệp vụ (mỗi nghiệp vụ chỉ mới lấy được một Form Task đầu tiên). Do đó em thiết kế theo kiểu này để đảm bảo việc lưu trữ Form Task có hiệu quả hơn.

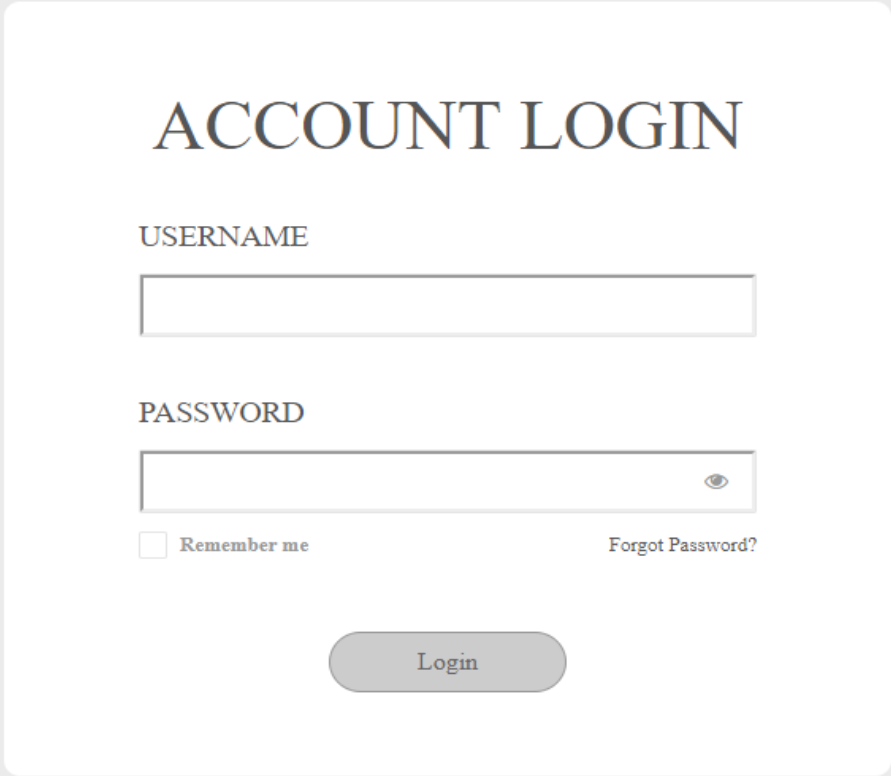
Khi một quy trình được thực hiện, WCP sẽ tiến hành xuống cơ sở dữ liệu để lấy Form Task, nếu trong cơ sở dữ liệu chưa có Form Task, WCP sẽ gửi RestAPI đến CEE để lấy Form Task được định sẵn từ Engine-Rest của Camunda. Sau đó sẽ lưu vào cơ sở dữ liệu và trả về người dùng.

CHƯƠNG 6: KẾT QUẢ CÀI ĐẶT VÀ ĐÁNH GIÁ

Trong chương này, em sẽ trình bày kết quả cài đặt sau khi thực hiện cài đặt hệ thống. Và đồng thời em cũng xây dựng một ứng dụng minh họa để từ đó em rút ra được đánh giá cho hệ thống của mình.

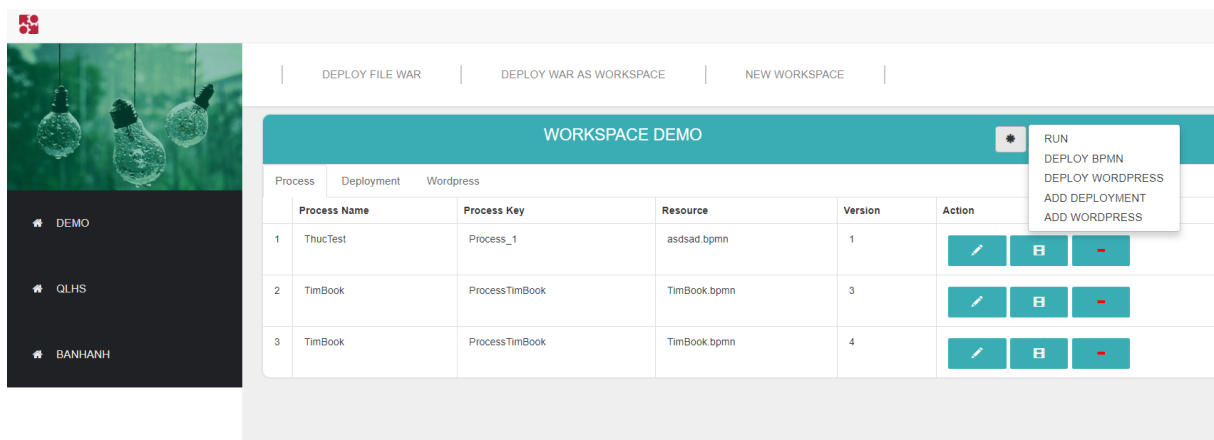
6.1 Kết quả cài đặt

6.1.1 Camunda-Extend-System



The image shows a web-based login form titled "ACCOUNT LOGIN". It features two input fields: "USERNAME" and "PASSWORD". The "PASSWORD" field includes a toggle icon (an eye) to switch between visible and hidden states. Below the "PASSWORD" field, there is a checkbox labeled "Remember me" and a link labeled "Forgot Password?". At the bottom of the form is a "Login" button. The entire form is centered on a light gray background.

Hình 6.1 - Giao diện login của BWC



Hình 6.2 - Giao diện bussines-webapp-creator

Đăng nhập:

- Camunda sử dụng Basic Authentication để xác thực cũng như phân quyền cho các người dùng mà không cung cấp một cơ chế xác thực thông qua API.
- Thiết kế giao diện cho phần đăng nhập trong hệ thống Camunda-Database phát triển dựa vào API, nếu API trả về mã lỗi 401 thì ứng dụng sẽ nhận biết là xác thực người dùng không thành công và yêu cầu đăng nhập lại.

Workspaces:

- Dùng chức năng của New Workspace để tạo các workspace tương ứng. Các workspace được tạo sẽ hiện ra một danh sách, khi bấm vào một workspace, giao diện sẽ load tất cả những process, wordpress, deployment trong workspace đó.

Processes:

- Chứa các thông tin của quy trình vụ trong nhóm Workspace. Hiện tại hệ thống chỉ mới có một chức năng là thực thi process được biểu diễn bởi “cây bút chì” ở cột Action

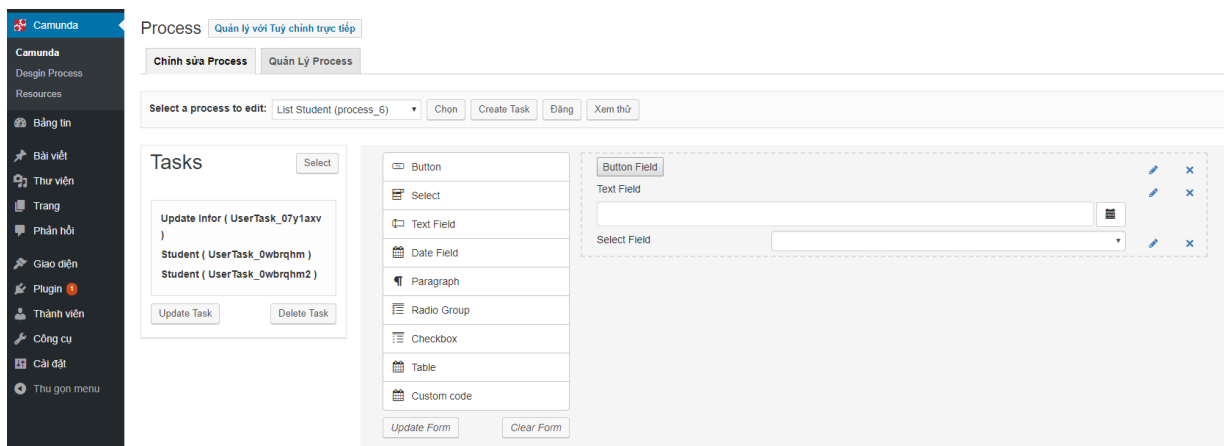
Deployment:

- Người dùng có 2 chế độ để deploy quy trình nghiệp vụ, đó là deploy từng tập tin .bpmn, hoặc deploy một nhóm quy trình nghiệp vụ được đóng gói dưới dạng tập tin .war
- Ở tab deployment sẽ cũng cấp chức năng tạo , thêm, xóa những deployment trong nhóm workspace đó

Wordpress:

- Cung cấp thông tin về những wordpress đã được tạo ra.
- Có thêm chức năng deploy nhanh ra wordpress.

6.1.2 Wordpress



Hình 6.3 - Giao diện thiết kế vụ trong Wordpress

New Task: Do hạn chế trong việc lấy form từ Camunda, nên em cung cấp chức năng tạo task để người dùng có thể designer form trực tiếp trước khi thực hiện nghiệp vụ.

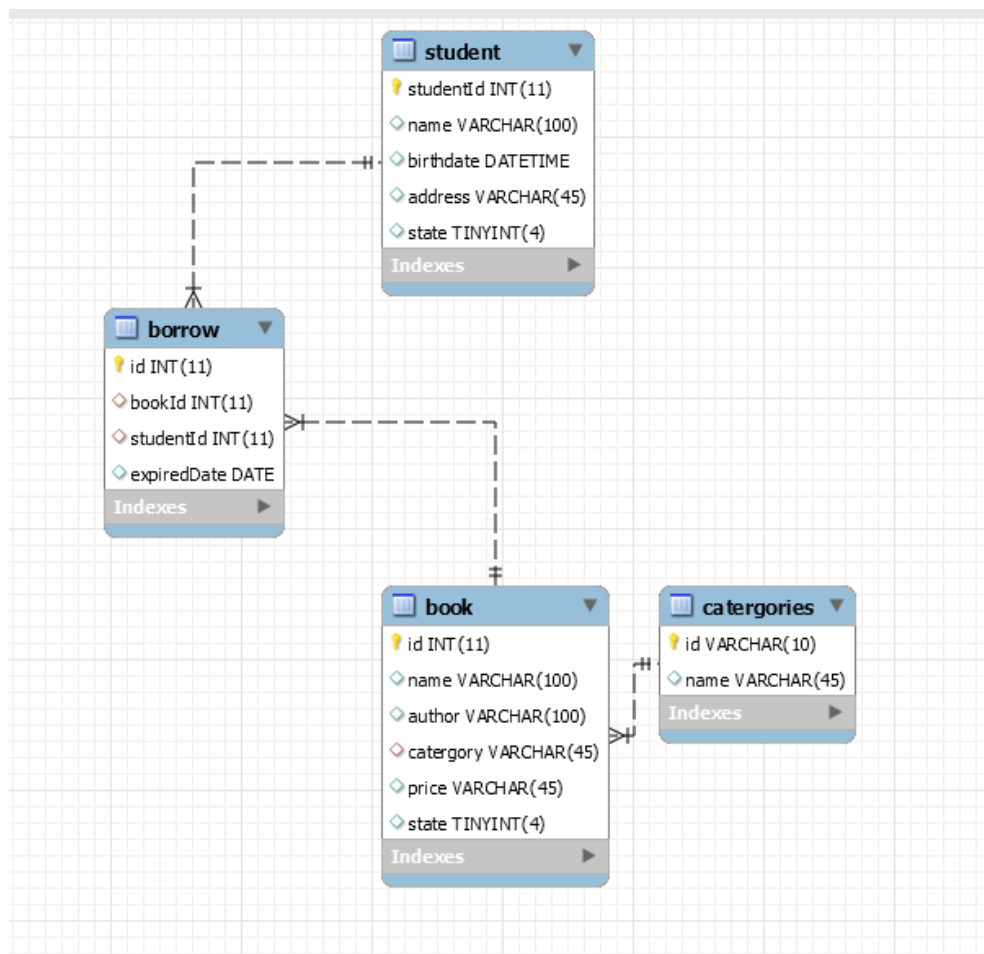
Update và Delete Task : Do để điều chỉnh lại thông tin Task.

Update Form: Đây là phần chính ở Wordpress mà em cung cấp cho người sử dụng. Người sử dụng sẽ dùng các chức năng để tạo các trường form phù hợp với họ mong muốn.

6.2 Xây dựng ứng dụng minh họa

Để đánh giá kết quả đạt được, em xin minh họa bằng việc xây dựng ứng dụng quản lý mượn sách theo từng bước như sau:

- Thiết kế các bảng dữ liệu.



Hình 6.4 - Kiến trúc database nghiệp vụ quản lý mượn sách.

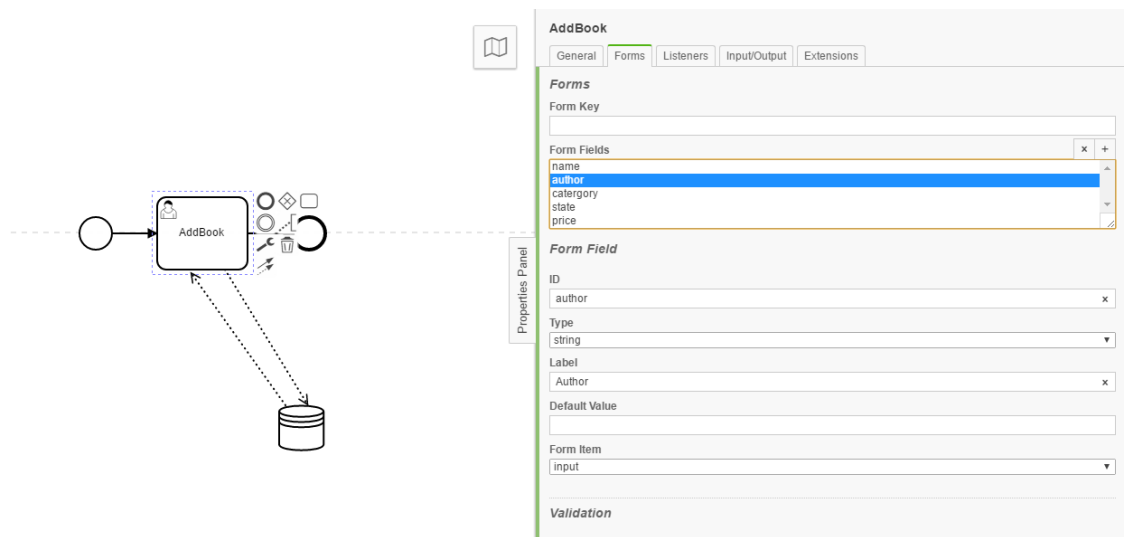
Hình 6.4 Thể hiện thiết kế kiến trúc database trong quản lý mượn sách. Trong đó bảng student chứa dữ liệu thông tin của học sinh. Bảng categories là thông tin danh mục sách. Bảng book chứa thông tin của sách, đồng thời có khóa ngoại là category đến khóa chính của bảng categories. Bảng borrow chứa thông tin đặt sách của học sinh.

Để thể hiện hết các tính năng trong luận văn này, em đề xuất ra 5 nghiệp vụ để thể hiện, bao gồm:

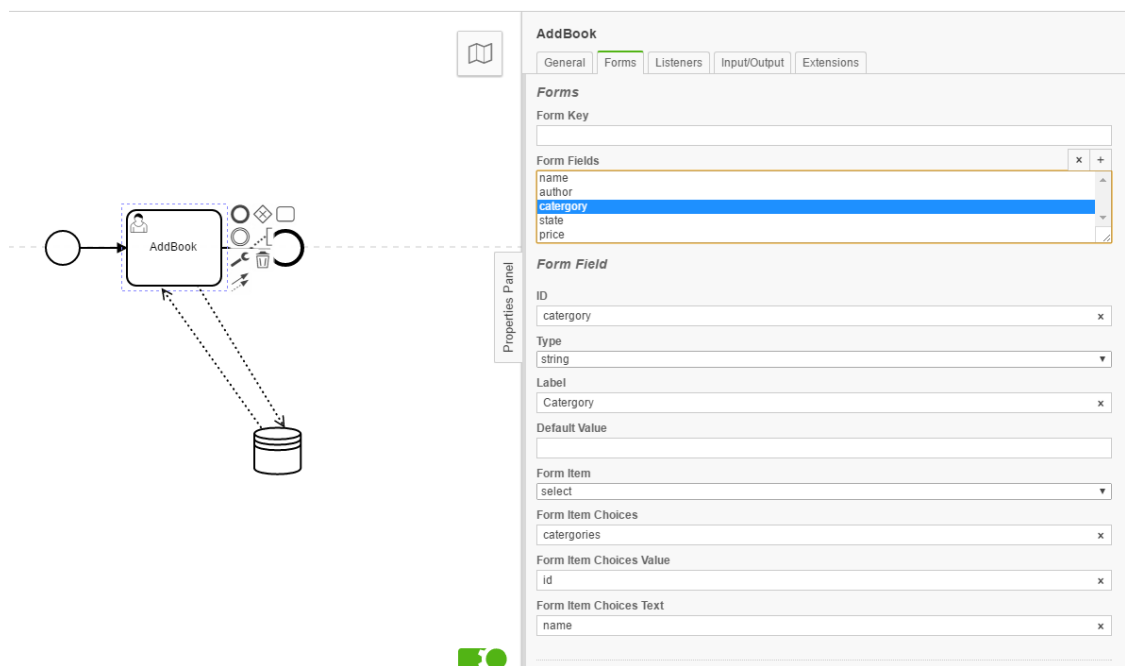
- Nghiệp vụ **AddStudent**: Dùng để thêm 1 “học sinh” vào, nghiệp vụ này không có gì đặc biệt, chỉ cần render ra những trường cơ bản, để giúp cho những người mới bắt đầu làm quen về hệ thống.
- Nghiệp vụ **AddBook**: Dùng để thêm 1 “sách” vào, nghiệp vụ này có thêm tính mapping giữa khóa ngoại category đến khóa chính của bảng categories(id)
- Nghiệp vụ **ListStudent**: Dùng xuất ra danh sách học sinh, bao gồm thêm chức năng xóa và sửa từng học sinh trong danh sách.
- Nghiệp vụ **ListBook**: Dùng để xuất ra danh sách sách, bao gồm thêm chức năng xóa và sửa từng sách trong danh sách sách.
- Nghiệp vụ **Borrow**: Là nghiệp vụ học sinh đặt sách. Đầu tiên người dùng sẽ nhập tên sách cần đặt, sau đó chương trình sẽ hiện danh sách book đã chọn, tiếp theo người chọn sách và điền thông tin vào.

Mô hình hóa nghiệp vụ cần thực hiện bằng Camunda Modeler tạo ra file BPMN.

- Mô hình hóa nghiệp vụ AddBook



Hình 6.5 - Các thêm trường author vào form addbook.



Hình 6.6 - Thêm category có khóa ngoại đến categories vào form addbook.

Hình 6.5 với 6.6 thể hiện các thêm trường author với trường category vào form addbook, và category có thuộc tính form item choices là

categories, form item choices value là id, form item choices text là name. Tức là category là có khóa ngoại đến khóa chính đến bảng categories là id, và thể hiện lên giao diện là name của categories.

UserTask_1wqsc88

General **Forms** Listeners Input/Output Extensions

Forms

Form Key

Form Fields

bookInfo

Form Field

ID

bookInfo

Type

string

Label

Book

Default Value

Form Item

table

Form Item Choices

book

Form Item Choices Value

{name};{author};{price};[Paragraph,category,categories];[Button,Update,action,edit];[Button,Remove,action]

Form Item Choices Text

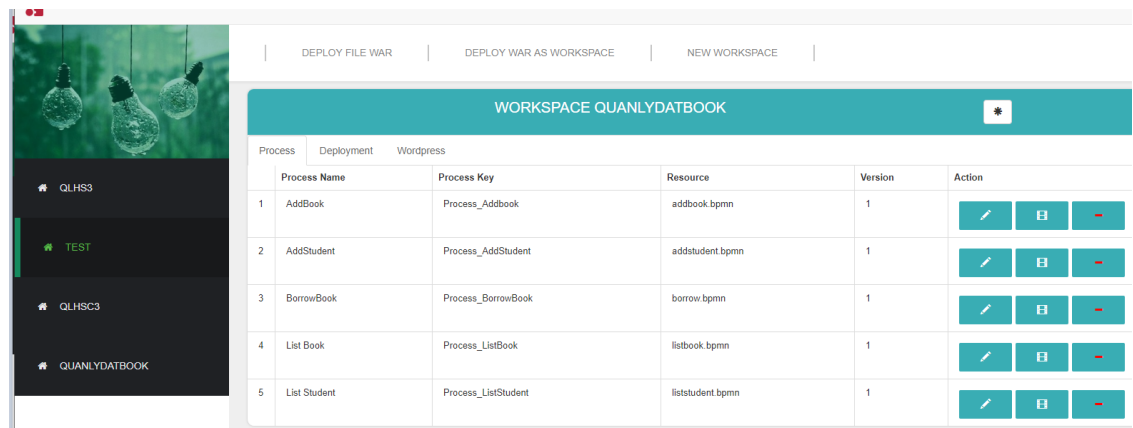
Name;Author;Price;Category;Action

Hình 6.7 - Các thêm table vào nghiệp vụ listbook.

Hình 6.7 có thể hiện cách để render ra table trong form nghiệp vụ listbook. Trong đó FormItemChoices là book, tức là lấy thông tin từ bảng book. FormItemChoicesText là header của table, các thuộc tính được cách nhau bởi dấu chấm phẩy ‘;’. FormItemChoicesValue là từng giá trị thể hiện

trong cột, cũng được cách nhảy bỏ dấu chấm phẩy ‘;’. Còn nếu muốn thêm một số thông tính để mapping bảng thì sửa dụng ‘[‘ và ‘]’ và các thuộc tính cách nhau dấu phẩy ‘,’.

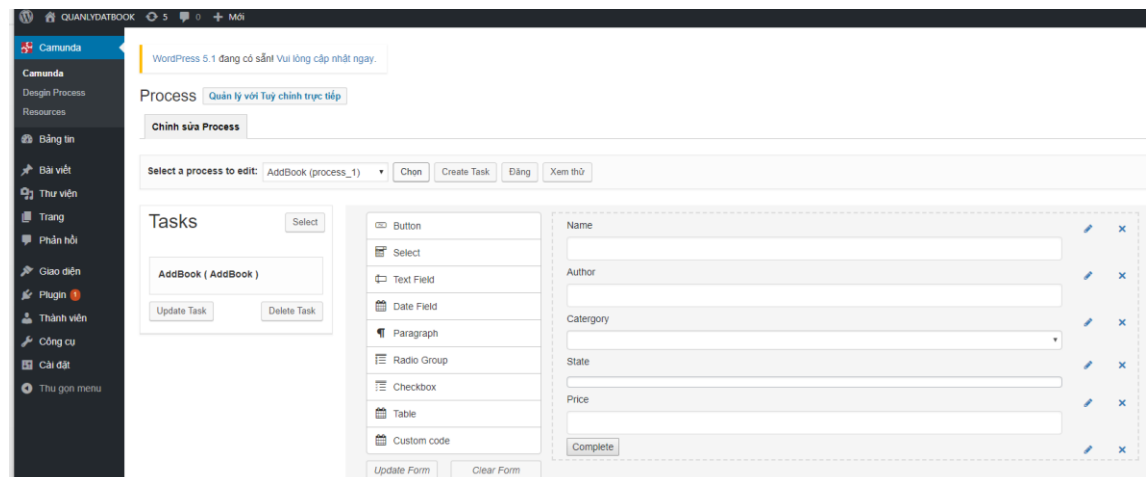
- Gom nhóm các quy trình nghiệp vụ vào chung nhóm (Workspace)



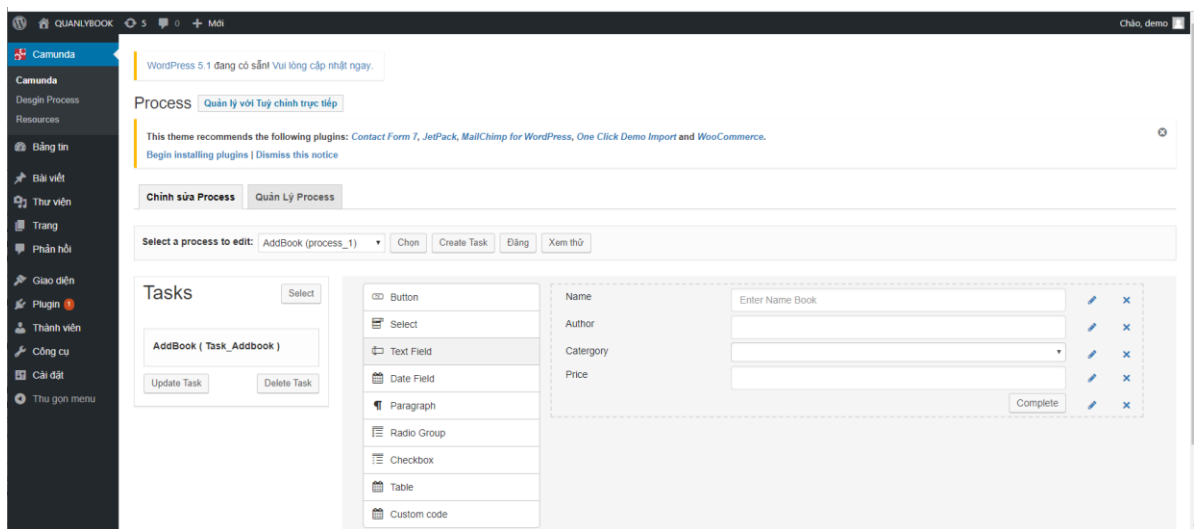
Hình 6.8 - Giao diện của Workspace Quản lý đặt book.

Hình 6.8 là giao diện khi gom các nghiệp trong trong quản lý đặt book lại. Người dùng sẽ dùng các chức năng đã cung cấp để tạo workspace và deployment các quy trình nghiệp vụ.

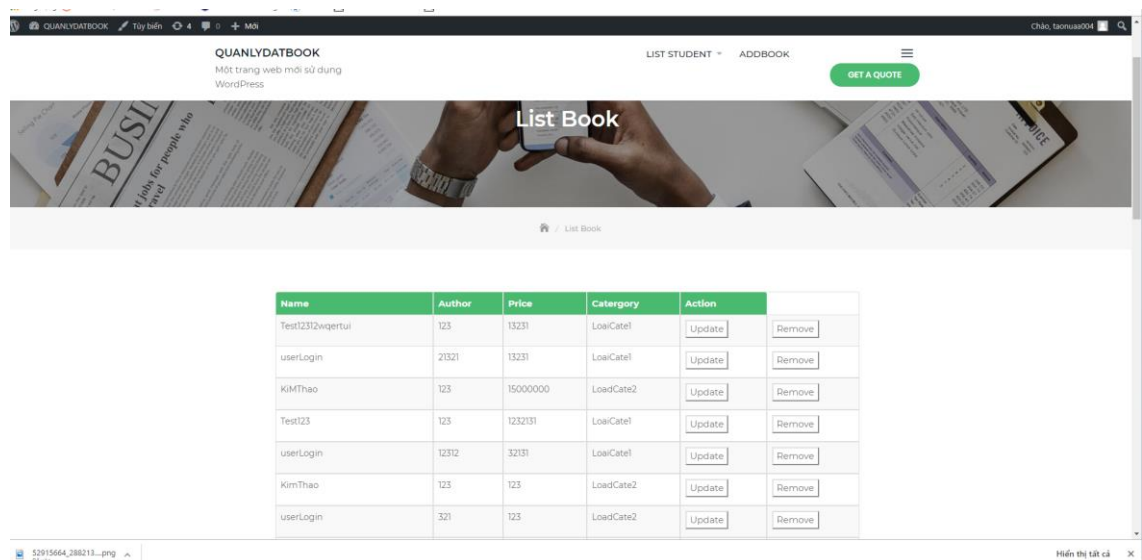
- Tạo ra các trang web thể hiện các nghiệp vụ từ Workspace.



Hình 6.9 - Giao diện để chỉnh sửa nghiệp vụ addbook.



Hình 6.10 - Giao diện sau khi chỉnh Addbook



Hình 6.11 - Giao diện kết quả của listbook.

Hình 6.9,6.10,6.11 Thể hiện các bước để tạo các quy trình nghiệp vụ thành các trang web tương ứng. Wordpress cung cấp rất nhiều giao diện, hoặc cho người sửa cùng custom lại giao diện bằng plugin, nên phần giao diện của quy trình nghiệp sẽ trở nên phong phú hơn.

6.3 Đánh giá

Hệ thống đã rút ngắn được quy trình phát triển phần mềm, giúp cho quá trình phát triển ứng dụng nhanh hơn, gần với nhu cầu khách hàng hơn, khách hàng có thể tham gia vào quy trình, tiết kiệm được thời gian, nhân lực.

Hệ thống hỗ trợ nhiều tính năng, đa dạng người dùng, giúp cho việc tạo ra các trang web quản lý nghiệp vụ gần theo như ý muốn, tùy chỉnh giao diện.

Hệ thống có tính ứng dụng cao, dễ dàng sử dụng, giúp tạo ra 1 trang web quản lý dễ dàng mà không cần biết quá nhiều nghiệp vụ.

CHƯƠNG 7: KẾT LUẬN

Trong chương này, em sẽ trình bày những kết quả đạt được và những khó khăn khi thực hiện đề tài luận văn này. Đồng thời em cũng nêu lên hướng phát triển cho đề tài này trong tương lai

7.1 Kết quả đạt được

Sau khi kết thúc luận văn em đã đạt được như sau:

7.1.1 Kiến thức

- Biết thêm nhiều ngôn ngữ như PHP, AngularJS, Java,...
- Hiểu được mô hình hóa trong quy trình nghiệp vụ
- Biết thêm về những phát triển nhanh quy trình nghiệp vụ
- Biết thêm về kiến trúc của BPMN, Camunda, Wordpress

7.1.2 Kỹ thuật

- Áp dụng được mô hình kiến trúc của Camunda vào hệ thống.
- Biết cách thêm Plugin vào Camunda, Wordpress
- Sử dụng thành thạo các công cụ của Camunda

7.1.3 Sản phẩm

- Camunda SDK phù hợp cho việc Render Form cho Task
- Nâng cấp Camunda Modeler và Process Engine.
- DEP đã có thêm Procedure và có thể viết trực tiếp query vào
- Camunda Modeler có thêm phần Render Form dạng Table, và Select
- Có Wordpress để quản lý thực thi các nghiệp vụ cho người dùng trên môi trường web
- Tạo được hệ thống quản lý các workspace và các wordpress
- Cho phép người dùng có thể tùy chỉnh giao diện cho từng quy trình.

7.2 Những khó khăn trong quá trình thực hiện

Tuy những kết quả thu được có phần khả quan, việc gặp những khó khăn trong quá trình thực hiện luận văn là không thể tránh khỏi. Sau đây là một số khó khăn chủ yếu em đã gặp phải trong quá trình thực hiện luận văn:

- Việc khó khăn đầu tiên trong luận văn này là do kết hợp nhiều ngôn ngữ, bao gồm Wordpress là php, SDK là angular và công cụ Grunt, Modeler được viết bằng Elelctron, Process-Engine, DEP, CEE được viết bằng Java Core, BWC được viết bằng Java Sping Boot.
- Thiếu kinh nghiệm trong việc thiết kế kiến trúc, dẫn đến việc kiến trúc của DEP trở thành kiến trúc một khối như hiện tại.
- Do hệ thống Camunda-Database[1] triển khai nâng cấp Modeler theo kiểu thay đổi mã nguồn, nên em cũng phải thực hiện theo thay đổi mã nguồn chứ không thể triển khai theo kiểu Plugin.
- Việc chuyển đổi thông tin từ tập tin .bpmn sang các đối tượng trong DEP được thực hiện thủ công và không có file XSD riêng.
- Thiếu kiến thức về bảo mật ở web, nên phân xử lí authorization giữa các server còn hạn chế.
- Các tài liệu hướng dẫn có liên quan đến các thành phần dịch vụ của Camunda Engine còn hạn chế dẫn đến việc khó khăn trong cài đặt
- Việc render Form còn hạn chế người dùng phải một số ngôn ngữ được định ra trong ứng dụng.

7.3 Hướng phát triển đề tài

Việc cải tiến một cách hoàn toàn đáp ứng tối đa nhu cầu của người dùng là một công việc mất nhiều thời gian và chi phí. Trong giới hạn về thời gian và nhân lực, em đã cải tiến một số chức năng của Camunda để đáp ứng được nhu cầu của người dùng, tuy nhiên, bên cạnh đó vẫn còn một số hạn chế. Sau đây là một số định hướng về mặt lí thuyết cũng như ứng dụng trong tương lai:

- Tiếp tục nghiên cứu sâu hơn về các vấn đề của BPMN cùng với những công cụ thực thi BPMN hiện tại
- Nghiên cứu về mặt công nghệ giúp cho kiến trúc của những tính năng đã cải tiến trở nên linh hoạt hơn, dễ dàng mở rộng những tính năng mới trong thời gian sắp tới
- Về mặt ứng dụng, tiếp tục hoàn thiện và mở rộng các tính năng đã cải tiến như:
 - Modeler: hiện tại chỉ hỗ trợ các câu truy vấn đơn giản do tập trung vào tính trực quan của công cụ, có thể thêm tính năng nhập câu SQL nâng cao dành riêng cho người dùng có chuyên môn về mặt kỹ thuật. Về phần form của UserTask còn hạn chế cho việc render các form item.
 - Giao diện: trong quá trình thực thi, người dùng có thể gán được công việc cho những người có liên quan bằng những quy trình nghiệp vụ BPMN.
 - Về quá trình thực thi: chưa có thể tạo layout cho những quy trình nghiệp vụ

DANH MỤC THAM KHẢO

- [1] “Huỳnh Tấn Phát – Nguyễn Chí Thành”, 2018, “*Nghiên cứu và cải tiến hệ thống camunda để phát triển nhanh các hệ thống quản lý với bpmn*”, luận văn tốt nghiệp cử nhân CNTT, bộ môn Công nghệ phần mềm, Đại học Khoa học Tự nhiên ĐHQG TpHCM.
- [2] Trang chủ form.io <https://www.form.io/>
- [3] Trang chủ flowable <https://www.flowable.org/docs/userguide/index.html>
- [4] *Camunda architecture*
<https://docs.camunda.org/manual/7.8/introduction/architecture/>
- [5] *Software to be biggest tech spend in 2013*
Forrester https://www.cio.com.au/article/520965/software_biggest_tech_spend_2013_forrester/
- [6] *BPMN introduction* <https://docs.camunda.org/manual/7.8/reference/bpmn20/>
- [7] Miguel Valdes Faura, *When do you really need custom application development?*
<https://techbeacon.com/app-dev-testing/when-do-you-really-need-custom-application-development?fbclid=IwAR2h7EDf9z0KYHcg7r14VLNl9PeqPtIRwPnvtGDen5ihS-BabOtMgM2ARWA>