E-commerce Web Application

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 1/12/2020 | 1.0 |  | Phạm Vũ Duy, Hồ Nguyễn Huy Hoàng |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Architectural Goals and Constraints 4

3. Use-Case Model 4

4. Logical View 4

4.1 Component: abc 4

5. Deployment 4

6. Implementation View 4

# Introduction

“Dang’s Company” is an E-commerce Web Application, its aim is to bring the buyers and the sellers together.

This document elaborates the software architecture document for the “Dang’s Company E-commerce Web Application”. The system architecture is abstracted into many view and components which will be explain in this document.

## Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture the significant architectural decisions which have been made on the system.

## Scope

The software architecture document applies to each static and dynamic aspect of the system.

Under the static behavior of the system, the document discusses the class diagram and other static architecture designs. Dynamic aspects of the system are elaborated using case realizations.

## Definition, Acronyms and Abbreviations

MVC – Model View Control architecture

DWA - Dang’s Company E-commerce Web Application

DB – DataBase

## Reference

<https://www.ecs.csun.edu/~rlingard/COMP684/Example2SoftArch.htm#Definitions,%20Acronyms%20and%20Abbreviations>

<https://www.slideshare.net/PasinduTennage/sample-software-architecture-document>

<https://sceweb.uhcl.edu/helm/RationalUnifiedProcess/process/artifact/ar_sadoc.htm>

## Overview

This document will present a detailed analysis of the architecture of Dang’s Company E-commerce Web Application. The further section will cover the architectural goals including the architectural constraints.

# Architectural Goals and Constraints

## Server side

The DWA will be hosted at “Heroku” JSP server. Mongo DB will be used as central database server. All communication between server and client will using HTTP/HTTPS (free SSL come along with Heroku) – a standard communication protocol.

## Client side

User will be able to access DWA only online. Users/Clients are expected to use a modern web browser which can support Bootstrap 4.0 to get full experience:

Recommend mobile devices browser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Google Chrome** | **Firefox** | **Safari 9.0 +** | **Android Browser 4.4+** |
| **Android** |  |  | N/A |  |
| **iOS** |  |  |  | N/A |

Desktop

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Chrome 45.0+** | **Firefox 38.0+** | **IE 10.0+** | **Microsoft Edge 12.0+** | **Opera 30.0+** | **Safari 9.0+** |
| **Mac** |  |  | N/A | N/A |  |  |
| **Windows** |  |  |  |  |  | N/A |

## Security and privacy

The central security will be handled by the member of the development team. They will be given the full access not only in the application level but also in database level. Creating account for the staff and the owner of the shop are done by administrator. When creating an account, user can choose their password and this password can be changed anytime by them. All the password is encrypted both on database or on communication process between client and server in order to ensure high level of security. The user information will only be seen by the shop owner and the administrator.

## Reliability and Availability

The system will be subjected to several testing step (Unit testing, Integration testing, System testing including Security and Performance testing) before being released to ensure that the system is reliable and worked as intended. The DB Central which is placed on MongoDB, ensure both the security with TLS/SSL encryption and performance such as low latency and response time.

## Performance

The server responds to any request from client within the web script timeouts (30 seconds), also the system performance can depend on available hardware, networks and internet connection capabilities. Therefore, the actual performance can be determined only after the system is deployed and tested. Our aim is to make the loading time on client side become ideal which is lower than 2 seconds.

## Portability and Reuse

The DWA is design to be a complete cosmetic website. But can be extend to sale many kinds of product. In order to maintain reusability, the web using Handlebar Template Engines which can be reuse. Best practice of RUP during development combine with Mongo DB make the structured is well layered.

## Development tools

The project is the combination of many tool:

Programing tool: Visual Code

Database: MongoDB

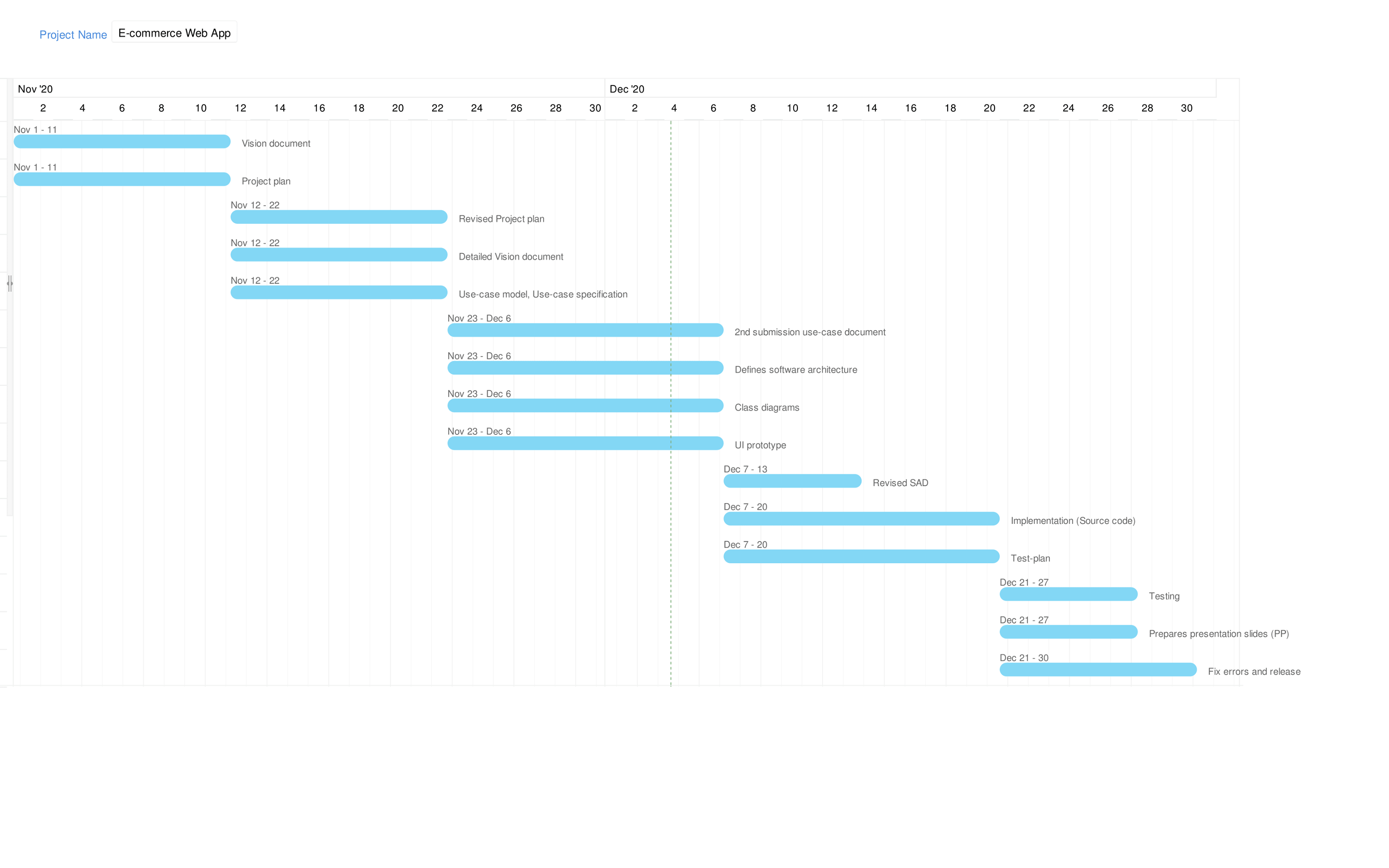
UI Prototype: figma

Meeting platform: Slack, Discord

Schedule: Trello

## Schedule

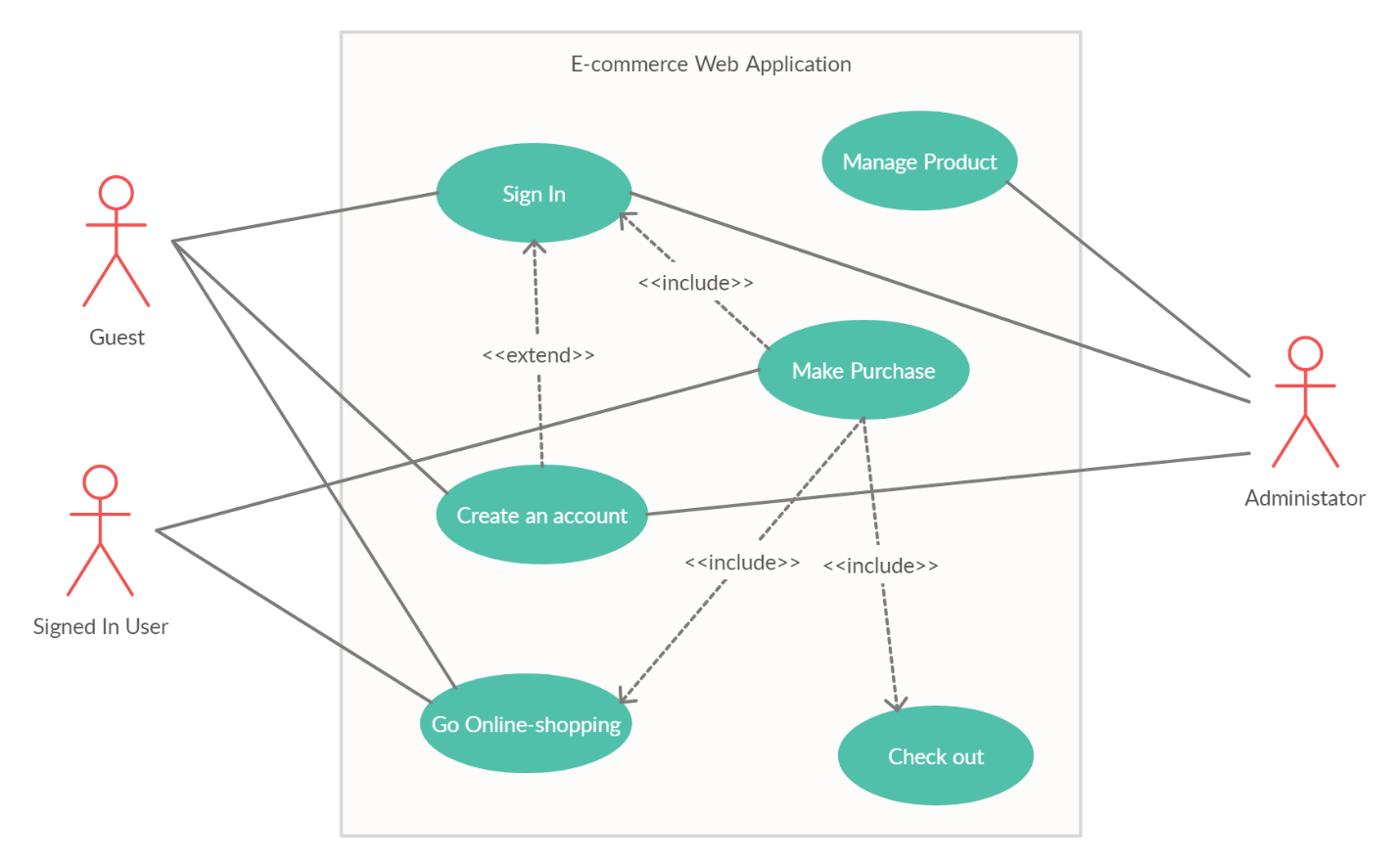
The development process is follows by the combination of Agile and RUP workflow. There are six sprints, each has their own workload and document:



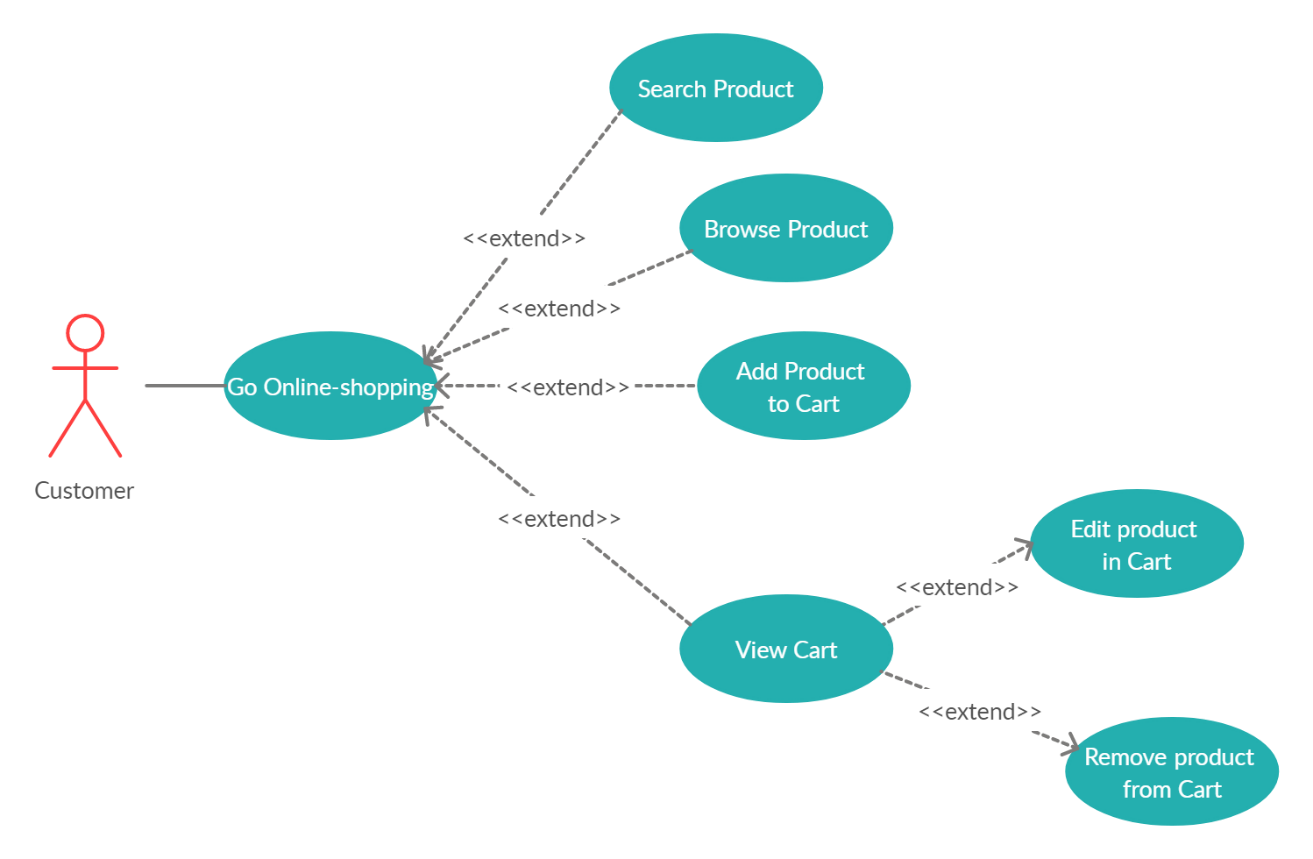
\*Due to the limitation of [Zoho](https://www.zoho.com/vi/projects/gantt-charts.html) Free-trial version that only allow maximum 3 member in each project. So, we are not able to show the specific assignment for each member in the Gantt chart above. This is the schedule with the specific assignment for each team member:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Phases** | **Interation No.** | **Tasks and Artifacts** | **Assignee** | **Start Date** | **End Date** |
| Inception | 1 | - Vision document | Nguyễn Phúc Thịnh | 1/11/2020 | 11/11/2020 |
| - Project plan | Huỳnh Nhật Nam |
| Elaboration | 1 | - Revised project plan | Huỳnh Nhật Nam | 12/11/2020 | 22/11/2020 |
| - Detailed vision document | Nguyễn Phúc Thịnh |
| - Use-case model, use-case specification | Huỳnh Nhật Nam, Nguyễn Phúc Thịnh |
| 2 | - 2nd submission use-case document | Huỳnh Nhật Nam, Nguyễn Phúc Thịnh | 23/11/2020 | 6/12/2020 |
| - Defines software architecture | Phạm Vũ Duy, Hồ Nguyễn Huy Hoàng |
| - Class diagrams | Phạm Vũ Duy, Hồ Nguyễn Huy Hoàng |
| Construction | 1 | - Revised SAD | Phạm Vũ Duy, Hồ Nguyễn Huy Hoàng | 7/12/2020 | 13/12/2020 |
| - UI prototype | Phạm Vũ Duy, Hồ Nguyễn Huy Hoàng |
| - Implementation (Source code) | All team members | 7/12/2020 | 20/12/2020 |
| - Test-plan | Mai Đăng Khánh |
| - Release (See 4.2.3 for better details) | Hồ Nguyễn Huy Hoàng | 14/12/2020 | 20/12/2020 |
| 2 | - Testing | Mai Đăng Khánh | 21/12/2020 | 27/12/2020 |

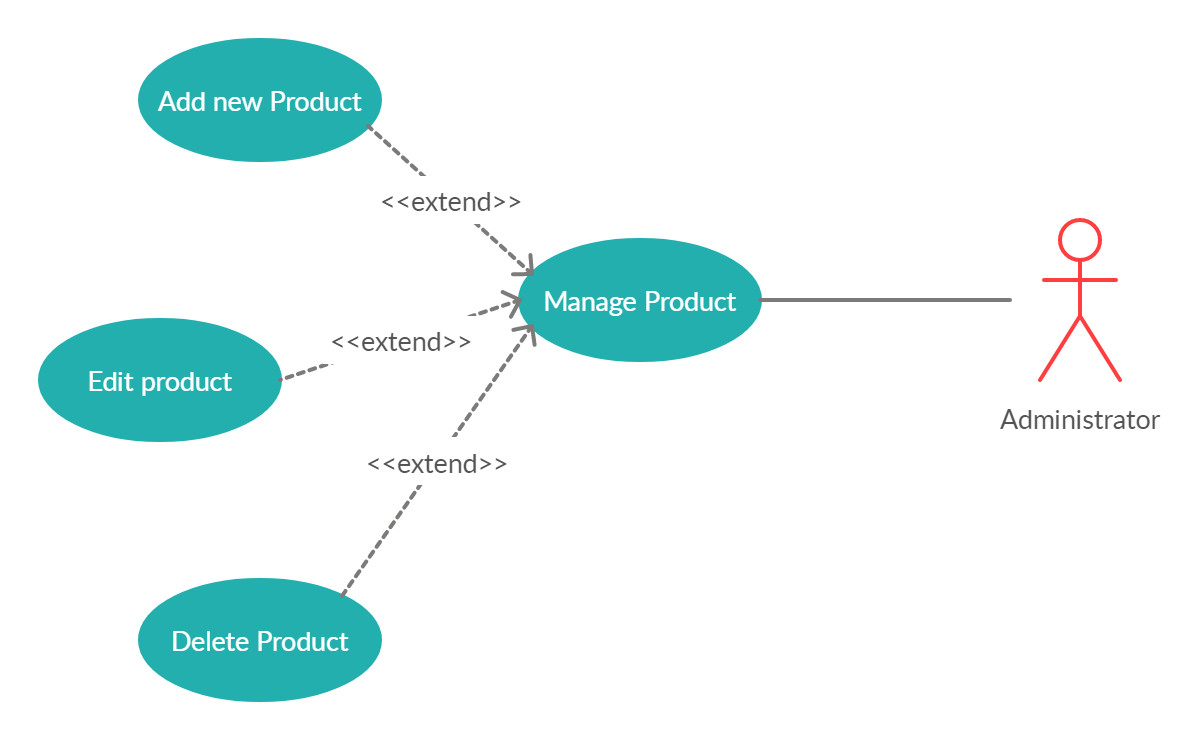
# Use-Case Model



* Go Online-shopping model:



* Check Out model:
* Manage Product:



# Logical View

## Overview

### The subsystem

The DWA can be divided into 3 main sub-system.

1. Web store
2. Order management subsystem
3. Statical data management subsystem

#### Web store

This subsystem provides all the functionalities that is related to user. The main use cases of this subsystem include

1. User login / Shop owner login
2. Create new user
3. Change password
4. Edit profile
5. Searching for product
6. Add product to cart
7. Make comment and rating

Depending on the level of account (access level), some action may be not be allowed.

#### Order management subsystem

The Order Management System (OMS) is play an important role as a median subsystem between the Web Store and the Data manager subsystem. The OMS main functionalities relate to all the customer ordering task.

* Order management
* Checkout handling

#### Statical data manager subsystem

This subsystem is the main system in charge of managing the central database. The subsystem involved in data access and processing operation which require special algorithms and processing capabilities. Only a few required data are fetched from the database

1. Numbers of product in stock
2. User information
3. Report about shop’s monthly income
4. Change product status (add, remove, modify)
5. Comment and rating management

## UML

Diagram

Description automatically generated

## MVC model

The DWA (Dynamic Web Application) is divided into 3 main components: *Controller, Model* and *View.* Those components connected with each other in a strict rule, each of them has a specification job. The reason why this pattern has been chosen is that relate to each specification job of each main components, implementers can work parallel on several component to reduce the developing time. Following that, implementers easy to update and debug each component separately. However, there are also contrast, MVC pattern is hard to deeply understand and it must have strict rules on method to avoid errors and mixed structured.

* The Controller handle the request of the client and make function calls to model to get the corresponding data.
* The Model handle the database, that means it make a request to the database such as get data, insert, update, delete. And its also send the data it has got to the view.
* The View in this case is import data to Html, Javascript and stick css style then send back to controller in order to respond the client request.

The pattern is present bellow to explain the communication method between those main components themself and other components

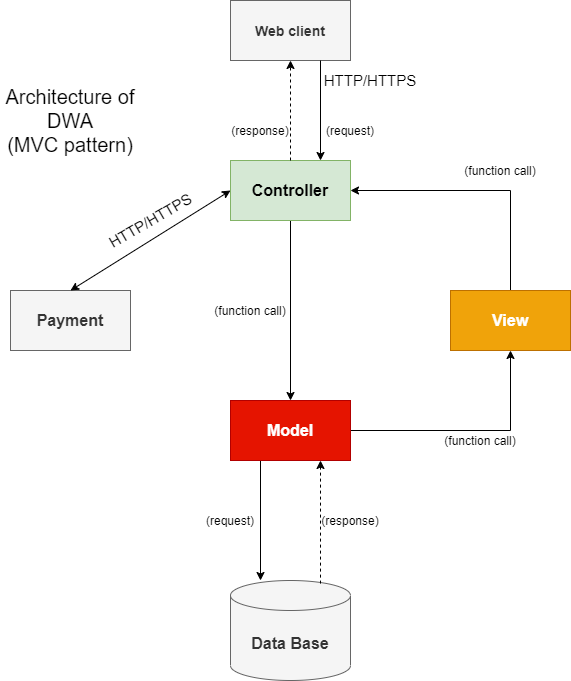


Figure : The MVC pattern for DWA architecture

### Controller

# Deployment

[Leave this section blank for PA3.]

# Implementation View

[Leave this section blank for PA3.]