MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Interior Furniture Augmented Reality System for Online Shopping

Group 3	
Group members	Nguyễn Phước Anh Khoa – SE61742 Phan Hồng Đức - SE61835 Bùi Thanh Thiên- SE61813
Supervisor	Kiều Trọng Khánh
Ext. Supervisor	N/A
Capstone Project code	ifAR

-Ho Chi Minh City, *08/01/2018*-

Table of Contents

Contents

List of Tables

List of Figures

Definitions, Acronyms, and Abbreviations

Name	Definition
ifAR	Interior Furniture Augmented Reality System for Online Shopping
AR	Augmented Reality
RC	Reality Capture
Model	Model is a three-dimensional (3D) object which is used to represent a real-world object in our system. The model is created from seller's series of pictures by system using Semi-auto processing*.
Product	Product is an interior furniture which is sold by retailer on our marketplace.
Semi-auto processing	Semi-auto processing: combines system automatic processes and designer manual processes. Model is generated automatically by system using RC, the model will include two files: a 3D image file (.obj format) and a texture image file (.jpg format). After that, designer is about to upgrade model's quality using 3D editors: Autodesk 3DS MAX or Blender and change format of 3D image file from .obj to.scn to conform mobile application's requirement using Apple Xcode manually. Goal model is included a 3D image file (.scn format) and a texture image file (.jpg format).

A. Report No. 1 Introduction

1. Project Information

- Project name: **Interior Furniture Augmented Reality System for Online Shopping**
- Project Code: **ifAR**
- Product Type: **Web Application, Mobile Application.**
- Start Date: **January 8th, 2018**
- End Date: **April 28th, 2018**

2. Introduction

We would like to introduce the augmented reality project, which is an integral part of the mobile technology in the future. Nowadays, it is difficult to choose the Interior Furniture at shop or expo that conform customers' space in their house. The customers also want to visualizate interior decoration before purchasing furniture. The furniture retailers always need large showroom to demonstrade their products. These are the big problems with both customers and retailers.

We hope this application will affect positively on the quality of shopping life as well as. In future, we intend to extend the system to several other product types, not only interior furniture. The system will become online marketplace provides diversity products, converting pictures to 3D models semi-automatically.

3. Current Situation

In the last few years, Virtual Reality is becoming increasingly popular, as computer graphics have progressed to a point where the images are often indistinguishable from the real world. However, the computer-generated images presented in games, movies, and other media are detached from our physical surroundings. This is both a virtue—everything becomes possible—and a limitation.

The limitation comes from the main interest we have in our daily life, which is not directed toward some virtual worlds, but rather toward the real world surrounding us. Smartphones and other mobile devices provide access to a vast amount of information, anytime and anywhere. However, this information is generally disconnected from the real world. Consumers with an interest in retrieving online information from and about the real world, or linking up online information with the real world, must do so individually and indirectly, which, in turn, requires constant cognitive effort.

In many ways, enhancing mobile computing so that the association with the real world happens automatically seems an attractive proposition. A few examples readily illustrate this idea's appeal. Location-based services can provide personal navigation based on the Global Positioning System (GPS), while barcode

scanners can help identify books in a library or products in a supermarket. These approaches require explicit actions by the user. however, they are rather coarse grained. Barcodes are useful for identifying books, but not for naming mountain peaks during a hiking trip; likewise, they cannot help in identifying tiny parts of a watch being repaired, let alone anatomic structures during surgery.

Augmented reality holds the promise of creating direct, automation and actionable links between the physical world and electronic information. It provides a simple and immediate user interface to an electronically enhanced physical world. The immense potential of augmented reality as a paradigm-shifting user interface metaphor becomes apparent when we review the most recent few milestones in human–computer interaction: the emergence of the World Wide Web, the social web, and the mobile device revolution.

iOS is the world's largest augmented reality platform. After Apple announces their iOS version 11.0 with batch new features and Augmented Reality (AR) is the most interesting feature that both users and developers want to experience. Games and apps now offer fantastically immersive and fluid experiences that go far beyond the screen. By taking advantage of the latest in AR technology, you can digitally redecorate your home, explore a city you've never visited, or even walk with dinosaurs. The possibilities are endless.

ARKit, Apple's augmented reality (AR) technology for building AR apps on iOS, delivers immersive, engaging experiences that seamlessly blend virtual objects with the real world. In AR apps, the device's camera presents a live, onscreen view of the physical world. Three-dimensional virtual objects are superimposed over this view, creating the illusion that they actually existed. The user can reorient their device to explore the objects from different angles and, if appropriate for the experience, interact with objects using gestures and movement.

The basic requirement for any AR experience—and the defining feature of ARKit—is the ability to create and track a correspondence between the real-world space the user inhabits and a virtual space where you can model visual content. When your app displays that content together with a live camera image, the user experiences augmented reality: the illusion that your virtual content is part of the real world.

(Reference: https://developer.apple.com/ios/human-interface-guidelines/technologies/augmented-reality, https://developer.apple.com/documentation/arkit/about_augmented_reality_and_arkit)

ARCore is a platform for building augmented reality apps on Android. ARCore uses three key technologies to integrate virtual content with the real world as seen through your smartphone's camera:

- Motion tracking allows the phone to understand and track its position relative to the world.
- Environmental understanding allows the phone to detect the size and

location of flat horizontal surfaces like the ground or a coffee table.

- Light estimation allows the phone to estimate the environment's current lighting conditions.

Fundamentally, ARCore is doing two things: tracking the position of the mobile device as it moves and building its own understanding of the real world.

ARCore's motion tracking technology uses the phone's camera to identify interesting points, called features, and tracks how those points move over time. With a combination of the movement of these points and readings from the phone's inertial sensors, ARCore determines both the position and orientation of the phone as it moves through space.

In addition to identifying key points, ARCore can detect flat surfaces, like a table or the floor, and can also estimate the average lighting in the area around it. These capabilities combine to enable ARCore to build its own understanding of the world around it.

ARCore's understanding of the real world lets you place objects, annotations, or other information in a way that integrates seamlessly with the real world. You can place a napping kitten on the corner of your coffee table, or annotate a painting with biographical information
about the artist. Motion tracking means that you can move around and view these objects from any angle, and even if you turn around and leave the room, when you come back, the kitten or annotation will be right where you left it.

(Reference: https://developers.google.com/ar/discover)

Until now, Google has not officially announced the technology Arcore so no third-party applications using arcore officially released.

We can see that a complete AR system requires at least three components: a tracking component, a registration component, and a visualization component. A fourth component—a spatial model (i.e., a database)—stores information about the real world and about the virtual world. The real-world model is required to serve as a reference for the tracking component, which must determine the user's location in the real world. The virtual-world model consists of the content used for the augmentation. Both parts of the spatial model must be registered in the same coordinate system.

IKEA Place, the app lets you browse through a wide selection of IKEA products, all of which can be placed right in your home using the new augmented reality functionality built into iOS 11. Chairs, tables, sofas, storage solutions, and more are available. IKEA has added more than 2,000 items from its catalog. From there, you can browse through the furniture collection and see how items fit into a room. Items can be rotated and moved around in a room so you can get an idea of the size of a piece of furniture relative to what's already in the room. Multiple pieces of furniture can be added to the app at the same time, and there's an option to take a photograph that can be saved to the camera roll or shared.

4. Problem Definition

Advantage of current system:

- People can see, interact, feel the real material of interior furniture.
- Customer can directly evaluate the real material of the interior furniture.

Below are disadvantages of current situation:

- The furniture retailer always need large showroom to demonstrate their products.
- Renting space in expo is difficult and expensive.
- Difficult to choose the interior furniture at shop or expo that conforms users space in their house.
- Cost of imparting, preservation is expensive.

5. Proposed Solution

5.1 Feature functions

Our proposed solution is to build an e-commercial system for interior furniture integrating AR technology named "ifAR" which provides users ability to place virtual real-world furniture to the real world via smartphone. Sellers can also create 3D models from scan real-world furniture using their smartphone via ifAR mobile application then selling this furniture with 3D sample on our online marketplace.

ifAR includes a mobile application and web application with following features:

1. **ifAR mobile application:**
   - **Customer:**
     - Place virtual real-world products to real world via smartphone's camera.
     - Contemplate products via 3D models.
     - Go shopping online
     - Upgrade role to Seller.
   - **Seller:**
     - Place virtual real-world objects to real world via smartphone's camera.
     - Contemplate products via 3D models.
     - Go shopping online
     - Sell products.
     - Create 3D model from real-world object overlap images.
     - Manipulate products.
     - Manipulate orders.
2. **ifAR web application:**
   - **Staff:**
     - Manipulate seller's sale requests.
     - Manipulate customer's upgrade role request.
   - **Admin:**
     - Including all features of Staff.

- Manipulate Staff's account.

5.2 Values and Challenges

Values:

- Applying AR & RC technology helps increase convience and save cost, time of trading interior furniture at home for both customers and sellers.
- Experience and trading products are convenient with online marketplace using AR technology.
- User can self create 3D virtual real-world furniture using smartphone.

Challenges:

- Device must be iPhone 6S or later with iOS version 11.3 or later.
- There're some strong competitors such as: Amazon, IKEA…
- AR & RC technologies are existed some shortcomings:
  - Surface detection of AR is low quality with low light level or on too flat surfaces.
  - 3D models which are generated from RC have redundant textures, so that designer have to upgrade 3D models' quality manually.

6. Functional Requirements

Functional requirements of the system are listed as below:

- **Customer's components:**
  - Go shopping online
  - Simulate products through AR view
- **Seller's components:**
  - Go shopping online
  - Simulate products through AR view
  - Request to sell products.
  - Manipulte orders.
  - Manipulate products.
- **Staff's component:**
  - Manipulate seller requests.
  - Manipulate customer's role upgrading requests.
- **Desginer's component:**
  - Include Staff's components.
  - Edit product's 3D model.
- **Administrator's component:**

- Including all components of Designers & Staff's components.
- Manipulate Staff's & Designer's account.

7. Role and Responsibility

No	Full Name	Role	Position	Contact
1	Kiều Trọng Khánh	Project Owner	Supervisor	khanhkt@fpt.edu.vn
2	Nguyễn Phước Anh Khoa	Scrum master	Leader	khoanpase61742@fpt.edu.vn
3	Phan Hồng Đức	Scrum team member	Member	ducphse61835@fpt.edu.vn
4	Bùi Thanh Thiên	Scrum team member	Member	thienbtse61813@fpt.edu.vn

Table 1 Roles and Responsibilities

B. Report No.2 Software Project Management Plan

1. Problem Definition

1.1 Name of this Capstone Project

- **Offical name:** Interior Furniture Augmented Reality System for Online Shopping
- **Abbreviation:** ifAR

1.2 Problem Abstract

Selecting and purchasing interior furniture are complex and inconvinient because imparting, demonstration, trying out furniture are high cost, time consuming. Therefore, we proposed a system for furniture online shopping enhancing experience with AR technology. This technology will bring the most authentic experiencing products for user through a smartphone.

Therefore, we need to develop a system provides online marketplace and uses AR technology to bring virtual real-world objects to real-world environment. There're some strong competitors have implement this concept, but the cost is expensive and services aren't provided in Vietnam.

Using AR technology to bring real experience for user means we are building a mobile application providing semi-automatic 3D furniture models creation from series of pictures and placing those ones to real world via smartphone camera. And we also provide online marketplace that user can trade products with each other, like Lazada, Tiki… This

technology requires us about iOS development, computer vision, 3D design knowledge. We also provide a web application for administrator to Manipulate marketplace and 3D model creating process.

There are some company have provided this concept in the world such as Amazon, IKEA… but in Vietnam, there's no company providing this concept yet (at the time this document was written).

After trying AR, 3D creation technology from variety companies, we decide to use AR technology of ARKit from Apple Swift and 3D creation technology of Reality Capture API from Autodesk. 3D creation process is a semi-auto processing, because creating 3D model from series of pictures is automatic by vision technology. However, the model will be contained redundant background and texture, so that we need to use 3D editor such as Autodesk Maya to filter and upgrade model's quality manually.

1.3 Project Overview

1.3.1 Current Situation

By research other systems, we found some problem current situation below:

- Swift (Programming Language for iOS Application): Our team hasn't experienced in this platform before.
- Augmented Reality: This is not a new technology but it has just become popular recently.
- ARKit: This is a new framework of Swift, it was released recently by Apple in WWDC2017.
- ARCore: This is a platform for building augmented reality apps on Android which similars to ARKit. ARCore uses three key technologies to integrate virtual content with the real world as seen through your phone's camera.
- Vuforia Object Scanner: The Vuforia Object Scanner is an Android application that is used to scan a physical 3D object but its pricing is expensive.
- Autodesk Reality Capture API: The Reality Capture API provides a set of endpoints for the Photo to 3D capability. These endpoints allow you to manipulate the process of generating a 3D mesh from overlapping photos.
- 3D model semi-auto processing: It combines not only computer vision algorithm but also manually 3D editing by people.
- E-Commercial: difficulty to compete with other systems such as Lazada, Amazon, IKEA…

1.3.2 The Proposed System

Because Augmented Reality technology is going to become popular so we decided to research about it. There are many providers for AR solution, such as: Vuforia, Oculus, Apple, Google... with varied pricing. But there are two free, powerful AR platforms from two popular companies: ARKit of Apple and ARCore of Google. ARKit has been realease by apple at WWDC 2017. Besides, ARCore is preview version, it doesn't advoid errors while implementing. ARKit is also used by large number of developer win world. So, we decided to choose ARKit instead of ARCore.

There are some APIs and SDKs that help to create 3D model, such as: Insight3d (open-source image based 3D modeling software), Unity Vuforia, Autodesk Forge Reality Capture API. However, Autodesk Forge Reality Capture API service is free for 1 years and is also cheaper than others. This API can generate 3D model simply and quickly with acceptable quality. Finally, we decided to use 3rd party API services from Autodesk Forge to resolve problem instead of using graphic designer software.

To create 3D model, we build a web application for create 3D model from 2D pictures. First, we use graphic designer software to create 3D models but this solution costs a lot of money and time, so we choose semi-automatic solution. Semi-auto processing is that we use 3rd API to convert 2D pictures to 3D model automatically with acceptable quality. After this step, designers edit 3D model to filter background and upgrade model's quality manually using 3D Editor.

1.3.2.1 Website

Website is main portal for staff and administrator. Website application provide following features:

- **Staff's feature:**
  - Manipulate seller requests.
  - Manipulate customer's role upgrading requests.
- **Desginer's feature:**
  - Include Staff's feature.
  - Edit product's 3D model.
- **Administrator's feature:**
  - Including all components of Designers & Staff's feature.
  - Manipulate Staff's & Designer's account.
- **Seller's feature:**
  - Manipulate products.
  - Manipulate orders.

1.3.2.2 Mobile Application

Mobile application for seller and customer. Mobile application provides following features:

- **For customer:**
  - Login
  - Simulate products through AR view.
  - Contemplate products via 3D view.
  - Go shopping online.
  - Upgrade role to Seller.
- **For seller:**
  - Login
  - Simulate products through AR view.
  - Contemplate products via 3D view.
  - Go shopping online.
  - Request to create 3D model for product.
  - Manipulate products.
  - Manipulate orders.

1.3.2.3 Scheduler

We develop a scheduler in web server application with following features:

- Generate 3D model.
- Create notifications.

1.3.3 Boundaries of the System

This section supposes that the government laws accept our e-commercial system and seller's information. Our system provides e-commercial system for users to trading interior furniture and support AR & RC technologies to improve experience and convience.

The language of this system is English.

The complete product includes:

- Website application for seller, staff and administrator.
- Mobile application for guest, customer and seller.
- Web server.

Our system supports:

- User can simulate products through AR view on smartphone without any addition divice.
- User also contemplates product via 3D view.
- Customer can go shopping online.

- Seller can request to sell products with 3D model which will be generated from seller's pictures.
- Seller is able to manipulate his/her products & orders.
- Staff and Degisner can manipulate seller information and sale requests.
- System supports create 3D model from seller's pictures.
- Support language is English.
- Currency is US Dollar ($).

Our system hasn't supportted features below yet:

- Manipulate staff
- Manipulate inventory.
- Manipulate product's quality.
- Trade models between sellers.
- Use 3D model which provided by seller to represent product in AR view.
- Recommended system for searching product is not available.
- Multi-languages and currencies are not support yet.

(This is simulated system for e-commerce system)

1.3.4 Future Plans

The current system only support for iOS and staff has to approve seller's pictures manually, so we recommend some features for future plans:

- Mobile application will be available on Android.
- Seller can use his/her own 3D model for product or trading model with other sellers.
- Apply color selection on 3D model.
- Support manipulate inventory & product's quality.
- Apply multi-languages and currencies.
- Support rate quality for product.

1.3.5 Development Environment

1.3.5.1 Hardware requirements

For server

Hardware	Minimum Requirements	Recommended
Internet Connection	Cable, Wi-Fi (7 Mbps)	Cable, Wi-Fi (20 Mbps)
Computer Processor	Intel® Core ® i7 2.4GHz	Intel® Core ® i7 2.4GHz
Computer Memory	8GB RAM	12GB or more

Table 2:

Table 2: Hardware Requirement for Server

For smartphone

Hardware	Minimum Requirements	Recommended
Internet Connection	Wi-Fi (7 Mbps)	Wi-Fi (14 Mbps)
Operating System	iOS 11.3	iOS 11.3 or later
Device Version	iPhone 6S	iPhone 7 Plus or later

Table 3: Hardware Requirement for Client

1.3.5.2 Software requirements

Software	Name / Version	Description
Environment	Java EE 7 Swift 4	Specification for developing web application Specification for developing mobile application
Modeling tool	Star UML	Used to design diagram
IDE	Eclipse Neon.3 Release (4.6.3) Xcode 9.3 MySQL Workbench 6.3.9	Programming tools
DBMS	MySQL 5.6.30	Used to create & manage the database for system
Source control	GitKraken Pro (3.5.1)	Used for source control
Web browser	Chrome 42 or later	Testing browser
Mobile OS	iOS 11.3 or later	Testing mobile application

Table 4: Software requirements

2. Project organization

2.1 Software Process Model

This project is developed under Scrum model. Below are the reasons why we choose this model:

- We have researched about Augmented Reality before. The risk of changing algorithm is high because proving accuracy of those algorithms is complicated. We need to use “try and test” method.
- The project contains a complicated system and the AR concept is very new for us, so we we need to try many design before the system run stability.

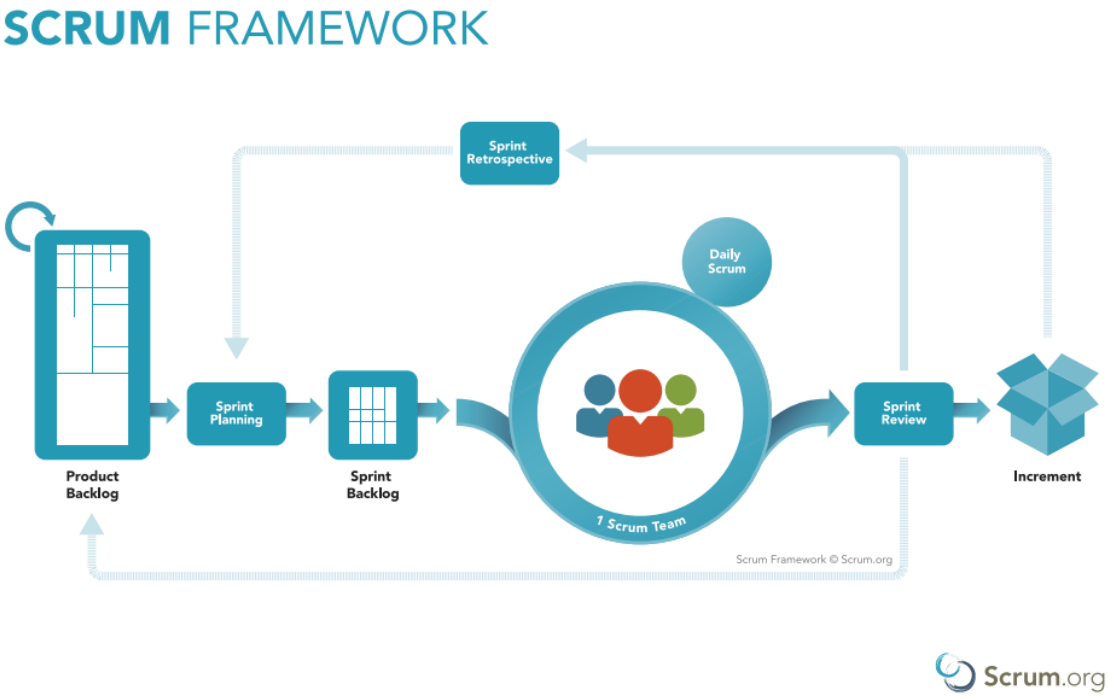


Figure 1 Scrum framework
Reference: https://www.scrum.org/resources/what-is-scrum

2.2 Roles and responsibilities

No	Full name	Role in Group	Responsibilities
1	Kiều Trọng Khánh	Product Owner	• Specify scope and requirements • Control the development process • Give out technique and business analysis support • Product backlog management
2	Nguyễn Phước Anh Khoa	Scrum master	• Managing process • Designing database • Clarifying requirements • Prepare documents • Coding • Testing • Quality management • Risk management • Create test plan • Arrange meeting

3	Phan Hồng Đức	Scrum team member	• Designing database • Create test plan • Clarifying requirements • Prepare documents • Coding • Testing • GUI design
4	Bùi Thanh Thiên	Scrum team member	• Designing database • Create test plan • Clarifying requirements • Prepare documents • Coding • Testing • GUI design

Table 5: Roles and Responsibilities Details

2.3 Tools and Techniques

Tool	Name and version
IDE	- Eclipse Neon.3 Release (4.6.3) - Xcode 9.3
Source control	- GitKraken Pro (3.5.1)
Database Manager	- MySQL Workbench (6.3.9)

Table 6: Tools

Technique	Name and version
Front end	- HTML v5.0 - JQuery v3.2.1 - XML v1.0 - Bootstrap v3.3.7
Back end	- Spring boot starter v4.1.6 RELEASE - Java v1.7 - Swift 4 - ARKit - Autodesk Forge Reality Capture v1
Web server	- Apache Tomcat v8.5.11
Database system	- MySQL 5.6.30

Table 6: Technique

3. Project Management Plan

3.1 Product Backlog

Story ID	Story	Task ID	Task
1	Product Owner (PO) wants to have introduction document	1.1	Project Information
		1.2	Current Situation
		1.3	Problem Definition
		1.4	Proposed Solution
		1.5	Role and Responsibility
		1.6	Functional Requirements
2	Scrum master wants to have Product Backlog	2.1	Create Product Backlog

3	PO wants to have project management plan	3.1	Problem Definition
		3.2	Project Organization
		3.3	Project management plan
		3.4	Coding Convention
4	PO wants to have SRS	4.1	User Requirement Specification
		4.2	Software Requirement Specification
		4.3	Software System Attributes
		4.4	Conceptual Diagram
5	PO wants to have SDD	5.1	Design Overview
		5.2	System Architectural Design
		5.3	Component Diagram
		5.4	Detailed Description
		5.5	Interface
		5.6	Database Design
		5.7	Algorithms
6	Scrum Team (ST) wants to know AR algorithm	6.1	AR problem definition
		6.2	AR Algorithm document
		6.3	Implement Algorithm
7	Scrum Team (ST) wants to know RC algorithm to generate 3D model	7.1	RC problem definition
		7.2	RC Algorithm document
		7.3	Implement Algorithm
8	Guest wants to register account	8.1	Guest_Register
9	PO wants the system have authorization for 2 roles: staff, administrator, customer, seller	9.1	Guest_Login
		9.2	Authenticated User_Log out
10	Customer wants to upgrade role to Seller	10.1	Customer_Edit profile

		10.2	Customer_Request to upgrade role
		10.3	Staff_Approve upgrade role request
11	Seller wants to manipulate product	11.1	Seller_Request to sell product.
		11.2	Staff_Approve product with pricing
		11.3	Seller_Approve the pricing
		11.4	Scheduler_Generate 3D model
		11.5	Staff_Update 3D model's quality
		11.6	Staff_Re-Upload edited model
		11.7	Seller_Get product's information
		11.8	Seller_Get 3D model
		11.9	Seller_Simulate product via smartphone's camera
		11.10	Seller_Update product's information
		11.11	Seller_Remove product
		11.12	Seller_Contemplate products via 3D models.
12	Customer wants to purchase product	12.1	Customer_Search product
		12.2	Customer_Simulate products through smartphone's camera
		12.3	Customer_Add product to cart
		12.4	Customer_Get cart
		12.5	Customer_Execute payment
		12.6	Customer_Contemplate products via 3D models.
13	Customer wants to track orders.	13.1	Customer_Get order
		13.2	Customer_Get order detail
14	Staff wants to manipulate products	14.1	Staff_Search product
		14.2	Staff_Filter products following category

		14.3	Staff_Get product's details
		14.4	Staff_Reject product
15	Staff wants to manipulate user (seller and customer)	15.1	Staff_Search user
		15.2	Staff_Get user's information
		15.3	Staff_Reject customer's upgrade role request.
16	PO wants authenticated user can modify profile	16.1	Authenticated user _Change password
17	PO wants scheduler send notification	17.1	Scheduler_Send notification
		17.2	User_Get notification
		17.3	User_Confirm notification
		17.4	User_Remove notification
18	Scrum master want to test the system	18.1	Create test plan
		18.2	Run test cases
		18.3	Fix bug
		18.4	Test document
19	Scrum master want to verify quality of the system	19.1	Quality document for system
20	PO want to have software user's manual	20.1	Installation guide
		20.2	User manual
21	PO want to have document paper	21.1	Paper document

Table 8: Sprint Backlog

3.2 Sprint Backlog

3.2.1 Sprint 1 (08.01.2018 – 21.01.2018): Project initiation

3.2.1.1: Goal

- Sprint 1 must complete the following tasks:

1.1 Project Information
1.2 Current Situation
1.3 Problem Definition
1.4 Proposed Solution
1.5 Role and Responsibility
1.6 Functional Requirements
2.1 Create Product Backlog
3.1 Problem Definition
3.2 Project Organization
3.3 Project management plan
3.4 Coding Convention
4.1 User Requirement Specification
4.2 Software Requirement Specification
4.3 Software System Attributes
4.4 Conceptual Diagram

3.2.1.2: Development

Task ID	Task	Responsible	Review
1.1	Project Information	DucPH, KhoaNPA	ThienBT
1.2	Current Situation	DucPH, ThienBT	KhoaNPA
1.3	Problem Definition	KhoaNPA	DucPH, ThienBT
1.4	Proposed Solution	DucPH, ThienBT	KhoaNPA
1.5	Role and Responsibility	DucPH, ThienBT	KhoaNPA
1.6	Functional Requirements	KhoaNPA, DucPH	ThienBT
2.1	Create Product Backlog	KhoaNPA	DucPH, ThienBT
3.1	Problem Definition	KhoaNPA	DucPH, ThienBT
3.2	Project Organization	KhoaNPA	DucPH, ThienBT
3.3	Project management plan	KhoaNPA	DucPH, ThienBT
3.4	Coding Convention	KhoaNPA	DucPH, ThienBT
4.1	User Requirement Specification	KhoaNPA, DucPH, ThienBT	KhoaNPA, DucPH, ThienBT

4.2	Software Requirement Specification	KhoaNPA, DucPH	ThienBT
4.3	Software System Attributes	DucPH, ThienBT	KhoaNPA
4.4	Conceptual Diagram	KhoaNPA, DucPH	ThienBT
4.1	User Requirement Specification	KhoaNPA, DucPH, ThienBT	KhoaNPA, DucPH, ThienBT
4.2	Software Requirement Specification	KhoaNPA, DucPH, ThienBT	KhoaNPA, DucPH, ThienBT

3.2.2 Sprint 2 (22.01.2018 – 11.02.2018): Software document description and basic features

3.2.2.1: Goal

- Sprint 2 must complete the following tasks:

5.1 Design Overview
5.2 System Architectural Design
5.3 Component Diagram
5.4 Detailed Description
5.5 Interface
5.6 Database Design
5.7 Algorithms
8.1 Guest_Register
9.1 Guest_Login
9.2 Authenticated User_Log out
16.1 Authenticated user _Change password

3.2.2.2: Development

Task ID	Task	Responsible	Review
5.1	Design Overview	KhoaNPA	ThienBT, DucPH
5.2	System Architectural Design	KhoaNPA	DucPH, ThienBT
5.3	Component Diagram	KhoaNPA	DucPH, ThienBT
5.4	Detailed Description	DucPH, ThienBT	KhoaNPA
5.5	Interface	KhoaNPA	DucPH, ThienBT
5.6	Database Design	KhoaNPA	DucPH, ThienBT
5.7	Algorithms	KhoaNPA, DucPH, ThienBT	KhoaNPA, DucPH, ThienBT
8.1	Guest_Register	KhoaNPA	ThienBT

9.1	Guest_Login	KhoaNPA, ThienBT	DucPH
9.2	Authenticated User_Log out	KhoaNPA, ThienBT	DucPH
16.1	Authenticated user _Change password	KhoaNPA, ThienBT	DucPH

3.2.3 Sprint 3 (26.02.2018 – 11.03.2018): Core flow

3.2.3.1: Goal

- Sprint 3 must complete the following tasks:

6.1 AR problem definition
6.2 AR Algorithm document
6.3 Implement Algorithm
7.1 RC problem definition
7.2 RC Algorithm document
7.3 Implement Algorithm
10.1 Customer_Edit profile
10.2 Customer_Request to upgrade role
10.3 Staff_Approve upgrade role request
11.1 Seller_Request to sell product.
11.2 Staff_Approve product with pricing
11.3 Seller_Approve the pricing
11.4 Scheduler_Generate 3D model

3.2.3.2: Development

Task ID	Task	Responsible	Review
6.1	AR problem definition	KhoaNPA, DucPH	ThienBT
6.2	AR Algorithm document	DucPH	KhoaNPA
6.3	Implement Algorithm	KhoaNPA, DucPH	ThienBT
7.1	RC problem definition	KhoaNPA, DucPH	ThienBT
7.2	RC Algorithm document	KhoaNPA	DucPH, ThienBT
7.3	Implement Algorithm	KhoaNPA, ThienBT	DucPH
10.1	Customer_Edit profile	KhoaNPA	ThienBT, DucPH
10.2	Customer_Request to upgrade role	KhoaNPA, ThienBT	DucPH
10.3	Staff_Approve upgrade role request	KhoaNPA, ThienBT	DucPH

11.1	Seller_Request to sell product.	KhoaNPA	ThienBT, DucPH
11.2	Staff_Approve product with pricing	ThienBT	KhoaNPA, DucPH
11.3	Seller_Approve the pricing	KhoaNPA	ThienBT, DucPH
11.4	Scheduler_Generate 3D model	ThienBT, KhoaNPA	DucPH

3.2.4 Sprint 4 (12.03.2018 – 25.03.2018): Core flow

3.2.4.1: Goal

- Sprint 4 must complete the following tasks:

11.5 Staff_Update 3D model's quality
11.6 Staff_Re-Upload edited model
11.7 Seller_Get product's information
11.8 Seller_Get 3D model
11.9 Seller_Simulate product via smartphone's camera
11.10 Seller_Update product's information
11.11 Seller_Remove product
11.12 Seller_Contemplate products via 3D models.
12.1 Customer_Search product
12.2 Customer_Simulate products through smartphone's camera
12.3 Customer_Add product to cart
12.4 Customer_Get cart
12.5 Customer_Execute payment
12.6 Customer_Contemplate products via 3D models.
13.1 Customer_Get order
13.2 Customer_Get order detail
14.1 Staff_Search product
14.2 Staff_Filter products following category
14.3 Staff_Get product's details
14.4 Staff_Reject product

3.2.4.2: Development

Task ID	Task	Responsible	Review
11.5	Staff_Update 3D model's quality	DucPH, ThienBT	DucPH, ThienBT

11.6	Staff_Re-Upload edited model	ThienBT	KhoaNPA
11.7	Seller_Get product's information	KhoaNPA	ThienBT, DucPH
11.8	Seller_Get 3D model	KhoaNPA	ThienBT, DucPH
11.9	Seller_Simulate product via smartphone's camera	KhoaNPA, DucPH	ThienBT
11.10	Seller_Update product's information	KhoaNPA	ThienBT, DucPH
11.11	Seller_Remove product	KhoaNPA	ThienBT, DucPH
11.12	Seller_Contemplate products via 3D models.	KhoaNPA	ThienBT, DucPH
12.1	Customer_Search product	DucPH	KhoaNPA, ThienBT
12.2	Customer_Simulate products through smartphone's camera	DucPH	KhoaNPA, ThienBT
12.3	Customer_Add product to cart	DucPH	KhoaNPA, ThienBT
12.4	Customer_Get cart	DucPH, ThienBT	KhoaNPA
12.5	Customer_Execute payment	DucPH	KhoaNPA, ThienBT
12.6	Customer_Contemplate products via 3D models.	KhoaNPA	ThienBT, DucPH
13.1	Customer_Get order	KhoaNPA	ThienBT, DucPH
13.2	Customer_Get order detail	KhoaNPA	ThienBT, DucPH
14.1	Staff_Search product	ThienBT	KhoaNPA
14.2	Staff_Filter products following category	ThienBT	KhoaNPA
14.3	Staff_Get product's details	ThienBT	KhoaNPA
14.4	Staff_Reject product	ThienBT	KhoaNPA

3.2.5 Sprint 5 (26.03.2018 – 08.04.2018): Complete coding and software testing

3.2.5.1: Goal

- Sprint 5 must complete the following tasks:

15.1 Staff_Search user
15.2 Staff_Get user's information
15.3 Staff_Reject customer's upgrade role request.
17.1 Scheduler_Send notification
17.2 User_Get notification
17.3 User_Confirm notification
17.4 User_Remove notification
18.1 Create test plan
18.2 Run test cases
18.3 Fix bug
18.4 Test document
19.1 Quality document for system

3.2.5.2: Development

Task ID	Task	Responsible	Review
15.1	Staff_Search user	ThienBT	KhoaNPA
15.2	Staff_Get user's information	ThienBT	KhoaNPA
15.3	Staff_Reject customer's upgrade role request.	ThienBT	KhoaNPA
17.1	Scheduler_Send notification	KhoaNPA, ThienBT	DucPH
17.2	User_Get notification	KhoaNPA, ThienBT	DucPH
17.3	User_Confirm notification	KhoaNPA, ThienBT	DucPH
17.4	User_Remove notification	KhoaNPA, ThienBT	DucPH
18.1	Create test plan	KhoaNPA, ThienBT, DucPH	KhoaNPA, ThienBT, DucPH
18.2	Run test cases	KhoaNPA, ThienBT, DucPH	KhoaNPA, ThienBT, DucPH
18.3	Fix bug	KhoaNPA, ThienBT, DucPH	KhoaNPA, ThienBT, DucPH
18.4	Test document	KhoaNPA, ThienBT, DucPH	KhoaNPA, ThienBT, DucPH

19.1	Quality document for system	KhoaNPA, ThienBT, DucPH	KhoaNPA, ThienBT, DucPH

3.2.6 Sprint 6 (09.04.2018 – 22.04.2018): Complete document and paper

3.2.6.1: Goal

- Sprint 6 must complete the following tasks:

20.1 Installation guide
20.2 User manual
21.1 Paper document

3.2.6.2: Development

Task ID	Task	Responsible	Review
20.1	Installation guide	DucPH, ThienBT	KhoaNPA
20.2	User manual	DucPH, ThienBT	KhoaNPA
21.1	Paper document	KhoaNPA	ThienBT, DucPH

3.3 All Meeting Minutes

All meeting minutes are saved at:
https://drive.google.com/drive/folders/1j4HTd58bhlBSbwdLoHMNTx-TY1EbrGu0?usp=sharing

4. Coding Convention

Using Java and Swift to develop the application.
Summary:

- Code block Convention:
  - No line break before the opening brace.
  - Line break after the opening brace.
  - Line break before the closing brace.
  - Line break after the closing brace, *only if* that brace terminates a statement or terminates the body of a method, constructor, or *named* class.
- Naming Convention:
  - Package names are all lowercase, with consecutive words simply concatenated together (no underscores).
  - Class names are written in UpperCamelCase.
  - Constant names use CONSTANT_CASE: all uppercase letters, with words separated by underscores.
  - Use camel-case style for variables and functions name
- Commenting Convention:
  - Place the comment on a separate line, not at the end of a line of code.

    - Begin comment text with an uppercase letter.
- Log data Convention:
    - Log URL on console before sending request.

Reference: *https://google.github.io/styleguide/javaguide.html*

C. Report No. 3 Software Requirement Specification

1. User Requirement Specification

Our system is designed to solve the problem of helping customers choose interior furniture that they don't know the interior furniture items fit the size of their home or not. Customers can also contemplate products with 360 degree view through their smartphone. We also provide online marketplace that sellers can demonstrate, sell their products with lower costs than the traditional way. This software consists of four user roles that are Customer, Seller, Staff, Admin with Staff and Admin are our employees.

1.1 Customer Requirement

Customer is a person who doesn't have access to the system. Customer can only use shopping function and chooses and places virtual real-world objects to real world on mobile. For payment, customer must login. These are some functions customer can use*:*

- Register
- Login
- Upgrade role to Seller
- Search product
- Add product to cart
- Get cart
- Execute payment
- Place virtual real-world objects to real world on mobile.
- Contemplate products via 3D models.

1.2 Seller Requirement

Seller is a person who doesn't have access to the system. Seller is a role that upgrade from Customer to have more features. There are some functions, Seller can use besiding Customer's function:

- Login
- Search product
- Add product to cart
- Get cart
- Execute payment
- Place virtual real-world objects to real world on mobile.
- Contemplate products via 3D models.

- Request to sell product
- Manipulate products

1.3 Staff Requirement

Staff is an employee in the system who has responsible for technical work such as approving requests from customer to upgrade role to seller, 3D model generating, editing 3D model result. Staff can do the following functions:

- Approve Customer's information to upgrade role from Customer to Seller
- Manipulate customer and seller
- Approve Seller's product
- Update model's quality
- Manipulate product

1.4 Administrator Requirement

Administrator is a person who has all functions of Staff and can also manipulate staffs' account.

1.5 Authorized User Requirement

Authorized user is a person who already logined success into system. Authorized user can do the following task:

- Logout
- Edit profile.

1.5 Scheduler Requirement

Scheduler can run some functions in backend. Scheduler user can do the following task:

- Generate 3D model
- Send notifications

2. System Requirement Specification

2.1 External Interface Requirement

2.1.1 User Interface

The user interface uses language is English for all web application and mobile application.

2.1.2 Hardware Interface

- **Server:**
  - Ram: 2GB

    - CPU: Intel Core i7-7700K CPU 4,20 GHz
        - Disk storage:
        - Database: 9322960 bytes ~ 9 MB
        - Operational system: 20GB
        - Environment: 128MB
        - Total: ~21GB
- **Mobile smartphone:**
    - Chipset: at least dual cores with 1.85 GHz
    - Disk storage: at least 16 GB
    - RAM: at least 2 GB
    - Wifi connection: 802.11ac Wi-Fi with MIMO
    - LTE band: LTE (Bands 1, 2, 3, 4, 5, 7, 8, 12, 13, 17, 18, 19, 20, 25, 26, 27, 28, 29)

2.1.3 Software Interface

- Service 3rd party:
    - AWS S3
    - Autodesk Forge Reality Capture v1
- Driver library:
    - Mssql Jdbc (v4.0)

2.1.4 Communication Protocol

- Use HTTP protocol 1.1 for communication between:
    - Web application and web server
    - Mobile application and web server

2.2 System Overview Usecase

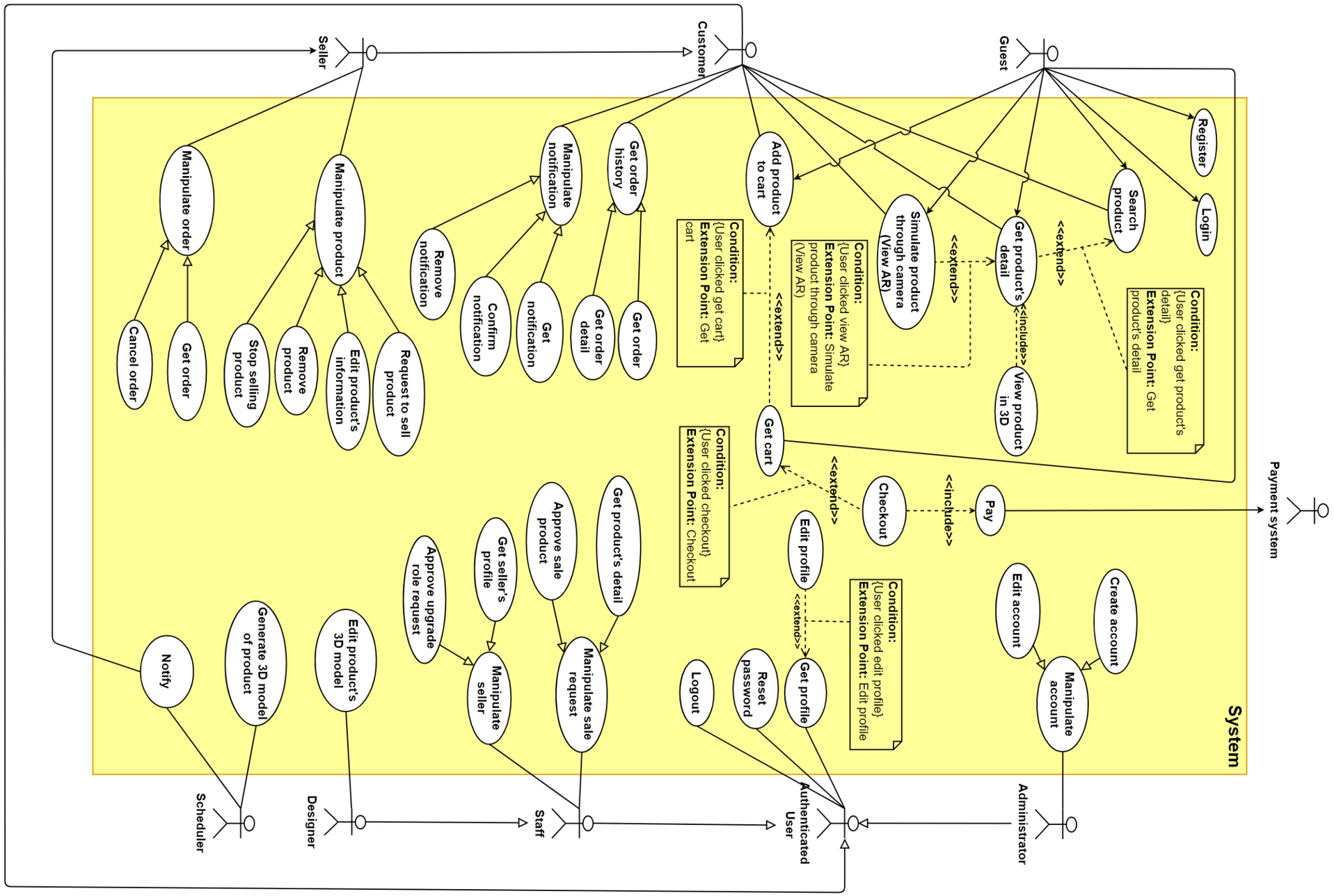


Figure 2: System Overview Usecase

2.3 List of Usecase

2.3.1 < Administrator > Overview Usecase

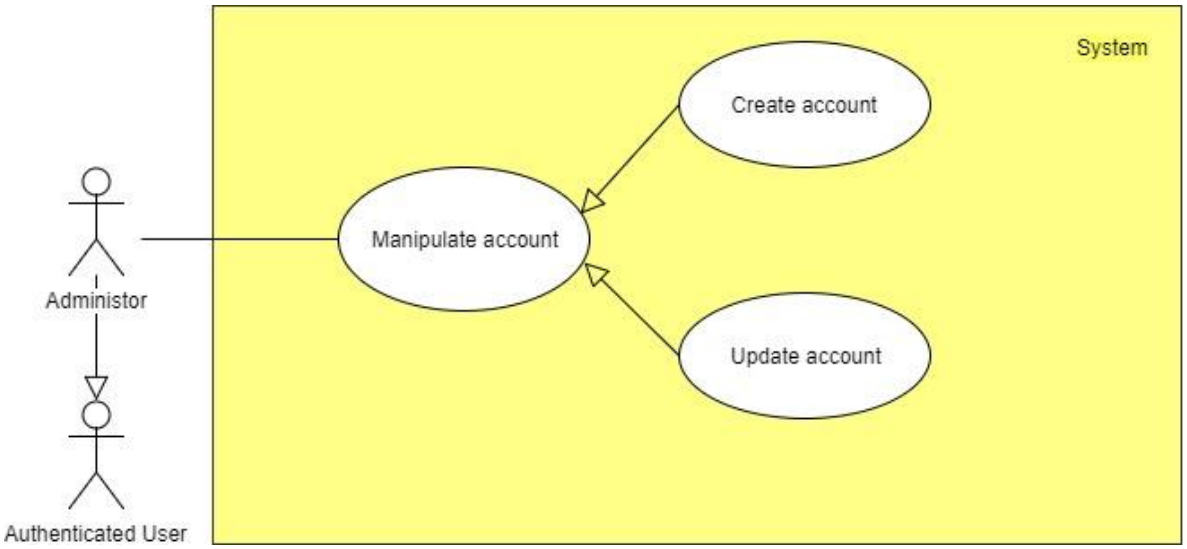


Figure 3: <Administrator> Overview UseCase

2.3.1.1 < Administrator > Create account Usecase

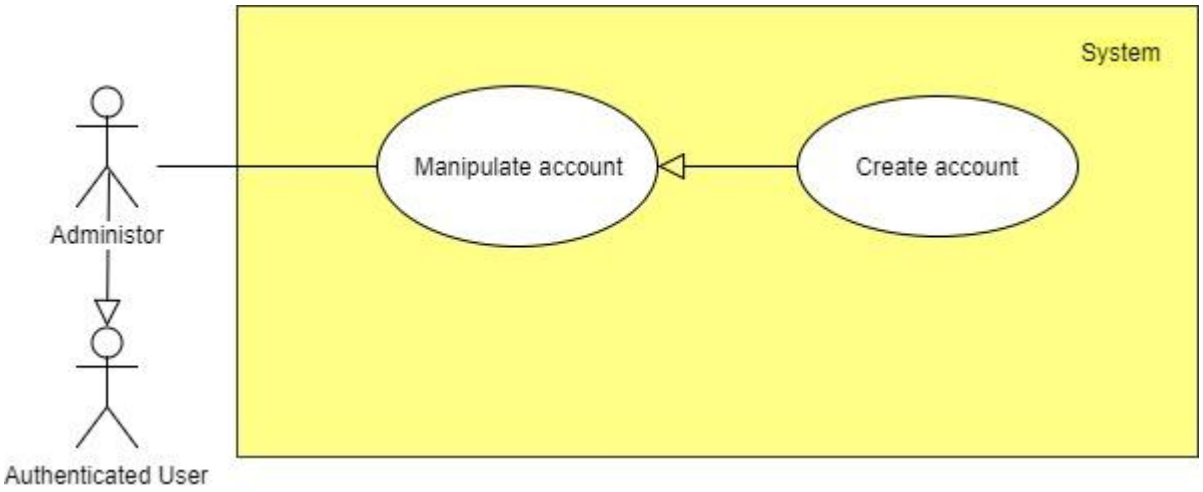


Figure 4: <Administrator> Create account Usecase

USE CASE – UC _A.01			
Usecase No.	A.01	**Usecase Version**	1.0
Usecase Name	Create account		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Administrator.

Summary:

- This use case allows administrator to create a new account for staff.

Goal:

- Staff can login into system with role "Staff" by using this account.

Triggers:

- Administrator sends create account command to the system.

Preconditions:

- User must login into system with role Administrator.

Post Conditions:

- **Success**: New account is added to the system.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Administrator goes to create account view.	Create account view is shown with following labels and fields: • Username: text, required, length 3 – 50 • Fullname: text, required, length 3 – 50 • Address: text, required, length 3 – 250 • City: dropdown list, require • Email: text, required, length 8 – 90 • Phone number: text, required, length 7 – 12 • Identity card: text, required, length 9 – 12
2	Administrator fills out the form.	
3	Administrator sends create account command.	Validate data. [Exception 1] Add new account in the system. System shows message account has been created successfully.

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response

1	Administrator sends command to create account.	System shows error message to ask administrator enters missing required fields.

Relationships: N/A

Business Rules:

- In case of success scenarios, a new account would be added to the system.
- Staff's email must not be duplicate.
- An email address must be validated by this regular expression: /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/
- Phone number and Identity card must be string of numbers.
- Staff's identity code is automatic initialized by the system.
- New account will be active in the system and has role Staff when it is created successfully.
- Password default is "123456", Staff has to change password by himself/herself.

2.3.1.2 < Administrator > Edit account Usecase

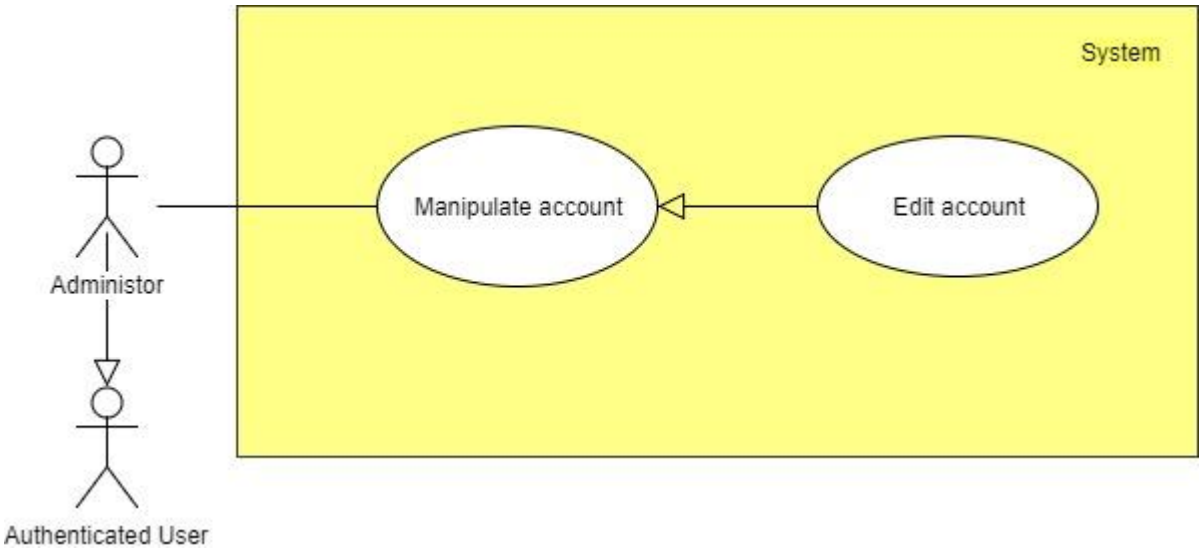


Figure 5: <Administrator> Edit account Usecase

USE CASE – UC _A.02			
Usecase No.	UC _A.02	**Usecase Version**	2.0
Usecase Name	Edit account		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Administrator.

Summary:

- This use case allows administrator to update account information of Staff.

Goal:

- Information will be customized to be reasonable.

Triggers:

- Administrator sends update account information command.

Preconditions:

- User must login into the system with role Administrator.

Post Conditions:

- **Success**: Staff's information is updated succesfully.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Administrator goes to update account information	Update account view is shown with following labels and fields: • Fullname: text, required, length 3 – 50 • Address: text, required, length 3 – 250 • City: dropdown list, require • Email: text, required, length 3 – 80 • Phone number: text, required, length 7 – 11 • Identify card: text, required, length 9 – 12
2	Administrator fills out the form.	
3	Administrator sends command to update account information.	Validate data [Exception 1] Update account information to the system. Reload account detail

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	Administrator sends command to update account.	System shows error message to ask

		administrator to enter missing required fields.

Relationships: N/A

Business Rules:

- Staff information is always loaded from the system.
- In case of success scenarios, staff information would be updated to the system.
- Reloaded customer detail will display customer updated information.
- An email address must be validated by this regular expression: /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/
- Phone number and Identity card must be string of numbers.
- Staff current information must be shown in its respective fields.

2.3.2 < Authenticated User > Overview Usecase

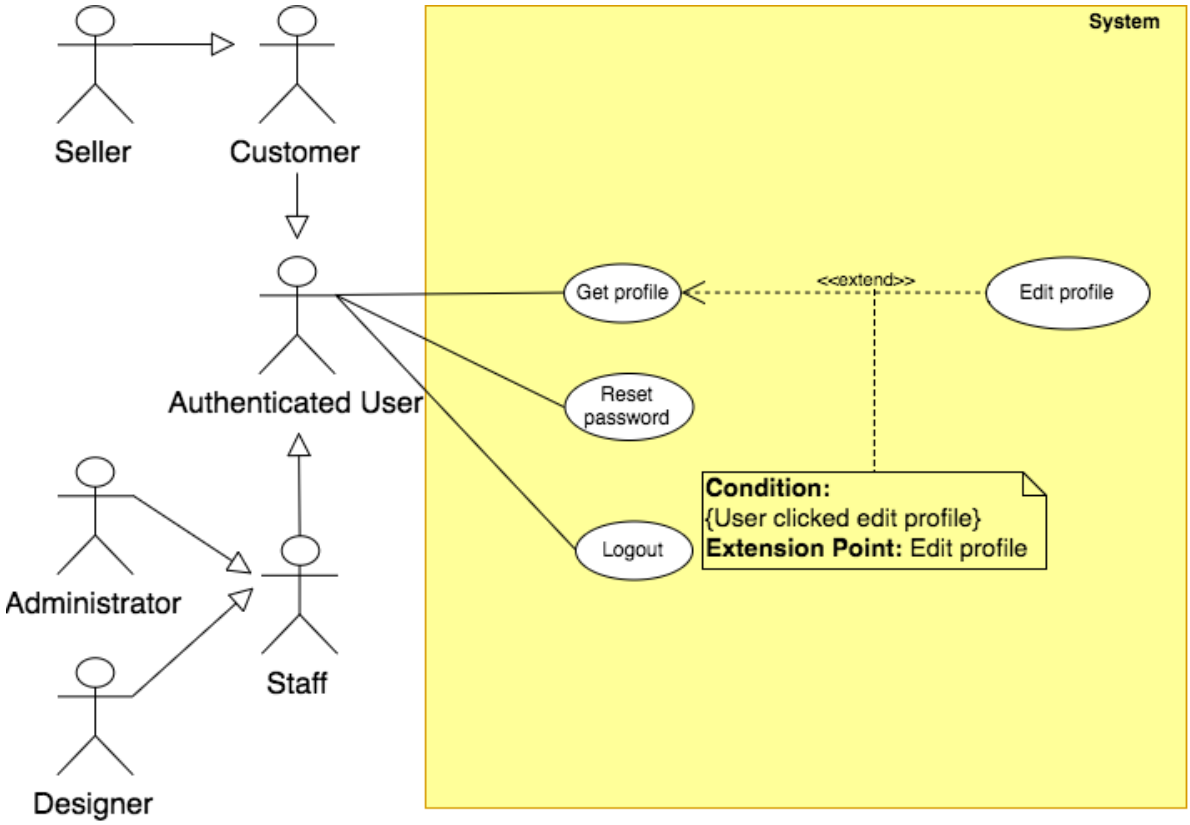


Figure 6: <Authenticated User> Overview Usecase

2.3.2.1 < Authenticated User > Log out Usecase

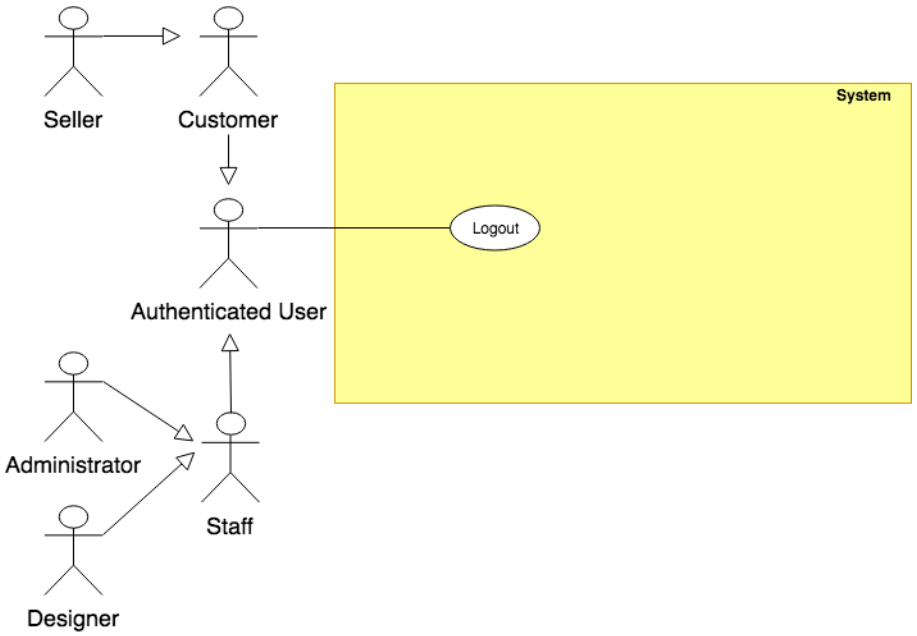


Figure 7: <Authenticated User> Log out Usecase

USE CASE – UC_AU.01			
Usecase No.	**UC_AU.01**	**Usecase Version**	1.0
Usecase Name	Logout		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Authenticated User

Summary:

- This use case allows user to log out the system.

Goal:

- Authenticated user stops accessing the system.
- User's session is removed from the system.

Triggers:

- User sends the logout command.

Preconditions:

- User logged in the system.

Post Conditions:

- Success: User logs out the system.
- Fail: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User sends logout command.	User exits from the system.

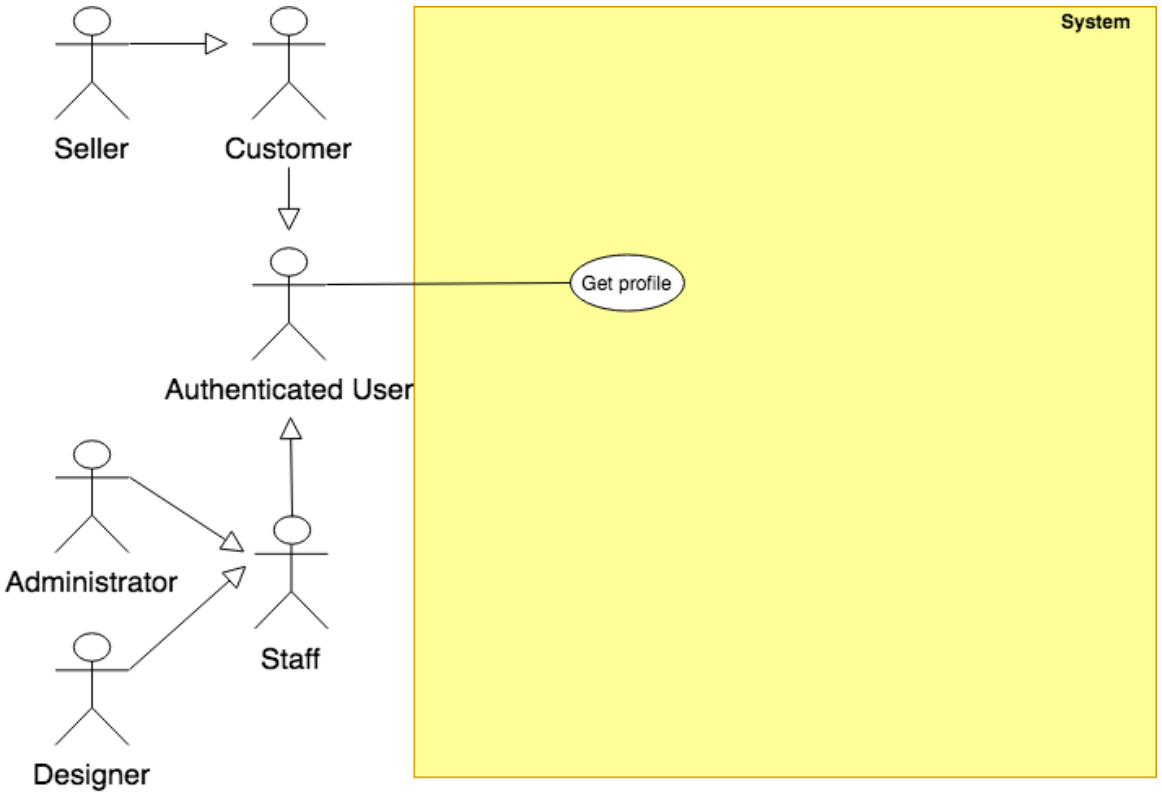
Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- If user is inactive in 30 minutes, system automatically logouts user.



2.3.2.2 < Authenticated User > Get Profile Usecase

Figure 8: <Authenticated User> Get profile Usecase

USE CASE – UC_AU.02			
Usecase No.	**UC_AU.02**	**Usecase Version**	1.0
Usecase Name	Get profile		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Authenticated user.

Summary:

- This use case helps user to get his/her profile's detail.

Goal:

- Authenticated user can get information from his/her profile's detail

Triggers:

- Authenticated user sends get profile command.

Preconditions:

- User must login into the system.
- If user role is staff, this account status must be "active".

Post Conditions:

- **Success**: User's profile is shown.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Authenticated user sgoes to get profile view.	System displays profile information: • Role: text • Username: text • Full name: text • Birthday: text • Sexual: text • Email: text • Address: text • City: text

Exceptions:

No	Actor Action	System Response
1	User sends view personal information command	User has been logged out because of inactivity too long.

Relationships: N/A

Business Rules:

- Profile information is always loaded from the system.
- If user is inactive in 60 minustes, system will automatically log user out.

2.3.2.3 < Authenticated User > Edit Profile Usecase

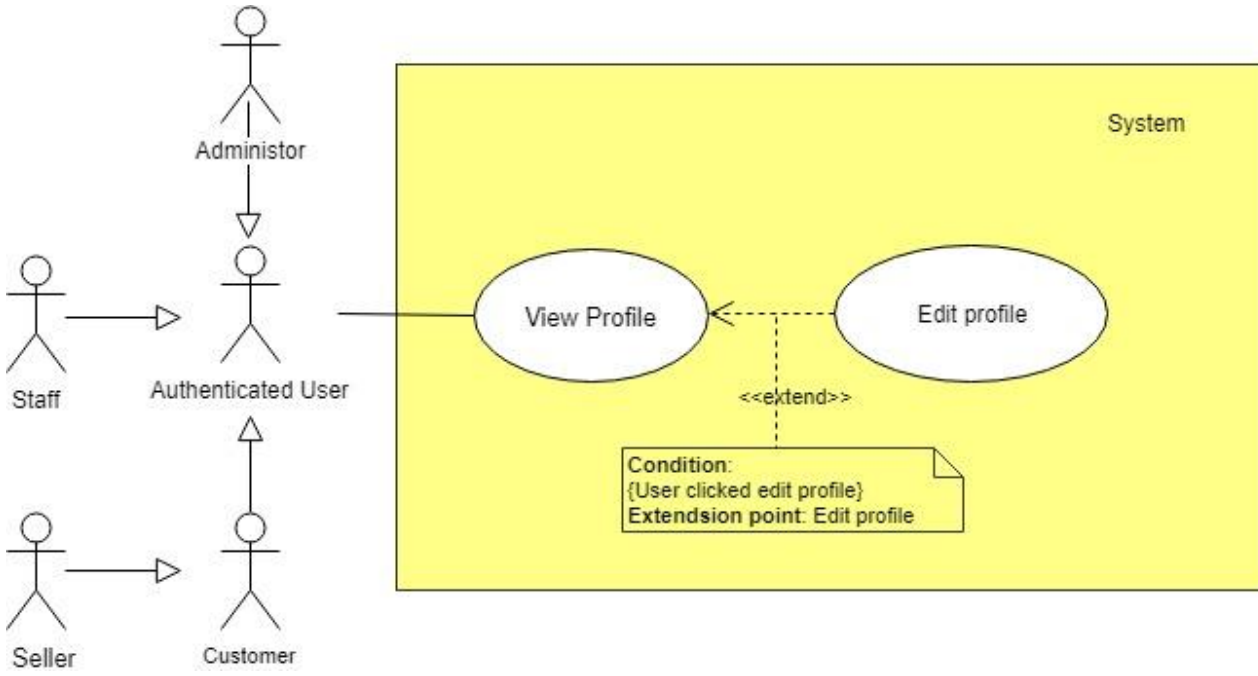


Figure 9: <Authenticated User> Edit profile Usecase

USE CASE – UC_AU.03			
Usecase No.	**UC_AU.03**	**Usecase Version**	1.0
Usecase Name	Edit profile		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Authenticated User.

Summary:

- This use case helps user to change their profile.

Goal:

- Authenticated user can change their information and password for security.

Triggers:

- User sends edit profile command.

Preconditions:

- User must login into the system.
- If user role is staff, this account status mus be “active”.

Post Conditions:

- **Success**: Update user’s profile to the system.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User goes to edit profile view.	Display new view contains labels and textboxes for user to input their information, current and new password: - Fullname: text, required, length 3-50 - Address: text, required, length 3-250 - City: dropdown list, required. - Email: text, required, length 8-90 - Phone number: text, required, length 7-12 - Identity card: text, required, length 9-10 - Current password: text, required, length 6-50 - New password: text, required, length 6-50 - Confirm new password: text, required, length 6-50 If user has “Customer” role, system displays more field and label: - Request upgrade role to Seller: button. [Alternative 1]
2	User fill out required input.	
3	User sends edit profile command. [Alternative 2]	Show message to notify that the profile has been updated successfully.

		[Exception 1,2]

Alternative Scenario:

No	Cause	System Response
1	User goes to edit view	If user has "Customer" role and has sent command to Request upgrade role to Seller, system displays more fields and labels: - Request upgrade role to Seler: text. - Status: text.
2	If user has "Customer" role and sends command to Request upgrade role to Seller	Show message to notify that request succeed.

Exceptions:

No	Cause	System Response
1	If user inputs the wrong current password	Show message to notify that the current password is wrong.
2	If the new password and the confirm password not matched.	Show message to notify that the confirm password is not matched with the new password.

Relationships: N/A

Business Rules:

- In case of successful scenario, user profile would be updated to the system.
- Password must be encrypted before save into the system.
- The new password must different than the old password.
- If user is inactive in 30 minutes, system will automatically log user out.
- The user should be able to know how strong their password is when they input their new password
- Request upgrade role to seller will be sent to the system with inputted information.
- New request status will be set to "Pending".
- A notification will be sent to the staff after the process is completed.

2.3.2.3 < Authenticated User > Reset password Usecase

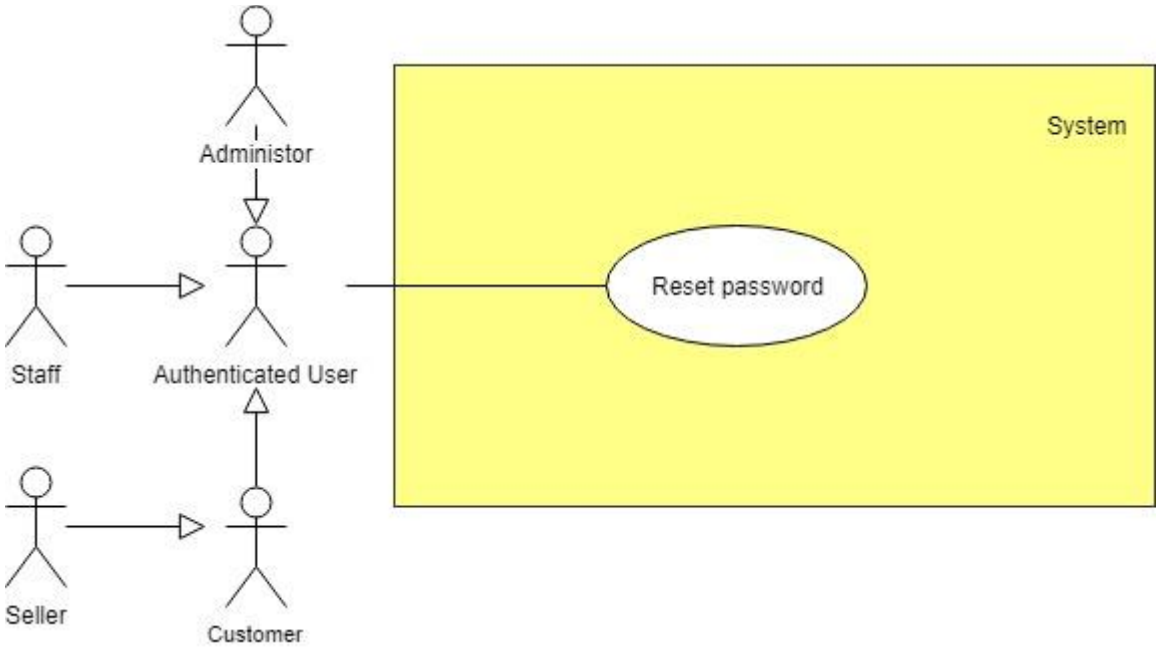


Figure 10: <Administrator> Reset passwrd Usecase

USE CASE – UC_AU.03			
Usecase No.	**UC_AU.03**	**Usecase Version**	1.0
Usecase Name	Reset password		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:
- Authenticated User.

Summary:
- This usecase allows user to reset password.

Goal:
- User can be received new password in their email by the system.

Triggers:
- User sends reset password command.

Preconditions: N/A

Post Conditions:
- **Success**: Systerm will send new password into user's email.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User goes to login view.	

		Login view is shown with following labels and fields: • Email or customer code: text • Password: text • Reset password: button.
2	User goes to reset password view.	System display new view with required information label and textbox: • User email: text, required, length 8-90 .
3	User fills out require information	
4	User sends command to reset password	- Validate email - System display reset password successfully, please check your email message. - System will send new password into user's email. - The new password is updated in the system. [Exception 1,2]

Alternative Scenario: N/A

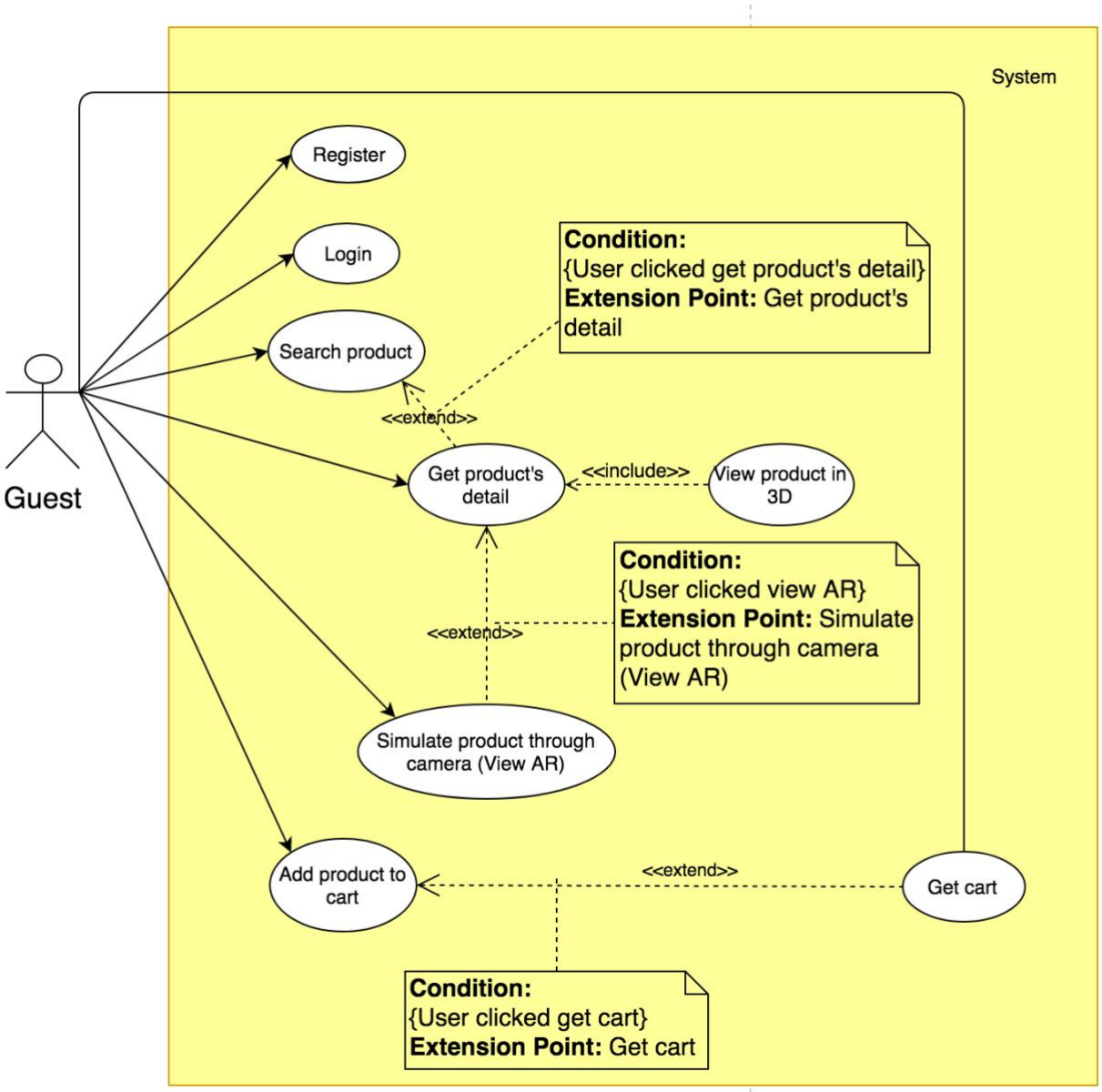
Exceptions:

No	Cause	System Response
1	Missing of required fields	Show message notify user input missed fields
2	Entered email is not existed in the system	Show message notify entered email not existed

Relationships: N/A

Business Rules:

- System will check the entered email is existed in the database or not.
- The new password is updated in the system.
- User can login into the system with new password and change it with edit profile command.



2.3.3 < Guest > Overview Usecase

Figure 11: <Guest> Overview Usecase

2.3.3.1 < Guest > Login Usecase

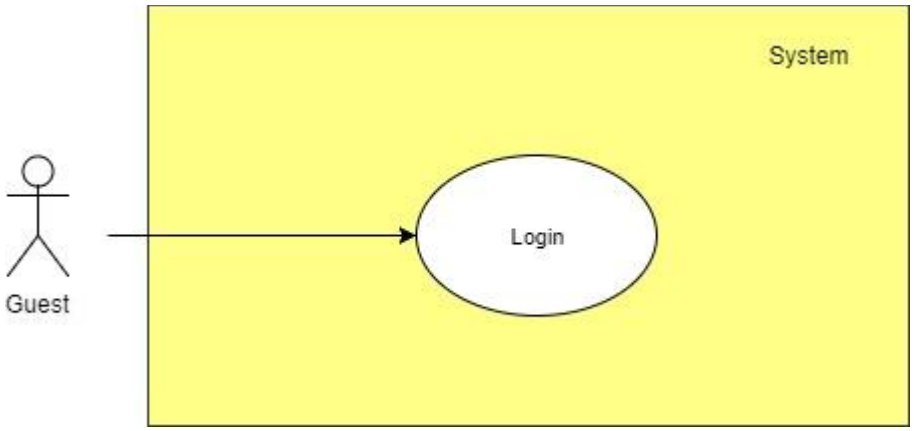


Figure 12:<Guest> Log in Usecase

USE CASE – UC_G.01			
Usecase No.	**UC_G.01**	**Usecase Version**	2.0
Usecase Name	Login		
Author	KhoaNPA		
Date	16/01/2018	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to login into the system

Goal:

- Guest must be authenticated and authorized before access the system.
- New session is created.

Triggers:

- Guest sends the login command.

Preconditions: N/A

Post Conditions:

- **Success:** Guest login the system.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to login view.	System requires identity information from Guest: - Username: text, required, length 3 – 50 - Password: text, required, length 6 – 50
2	Guest fill out information.	
3	Guest sends command to login to system	Guest will login system with their specific role [Exception 1,2]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	Missing of required fields	System notify guest to enter their identity information
2	Guest enter wrong identity information.	Wrong identity information, System shows error message.

Relationships: N/A

Business Rules:

- Password are encrypted before being sent to server.

- After login to system, guest will be redirected to specific view based on their role on the system: • If role is "Staff", the system will display to Staff Dashboard view. • If role is "Administrator", the system will display to Administrator Dashboard view. • If role is "Customer", the system will display to Customer view. • If role is "Seller", the system will display to Seller view.

2.3.3.2 < Guest > Register Usecase

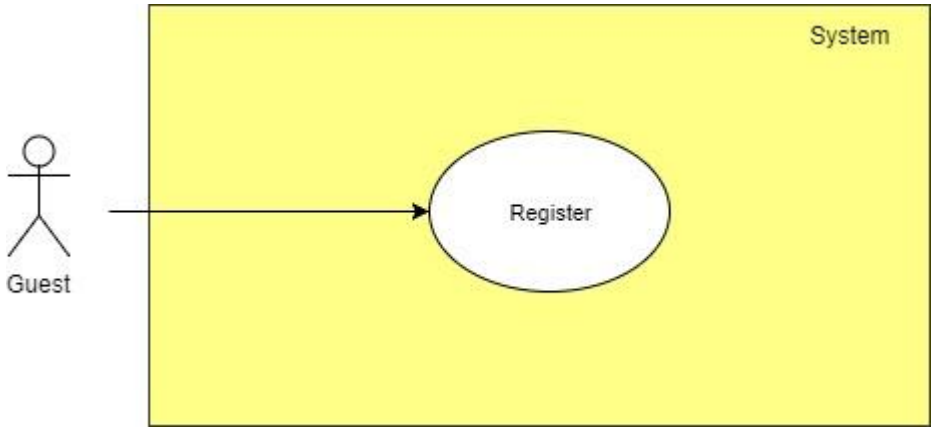


Figure 13: <Guest> Register Usecase

USE CASE – UC_G.02			
Usecase No.	**UC_G.03**	**Usecase Version**	2.0
Usecase Name	Register		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to register to system.

Goal:

- System can recognize the new account.
- User can login into the system with role "Customer".

Triggers:

- Guest sends register command.

Preconditions: N/A

- This username hasn't existed in the system yet.

Post Conditions:

- **Success:** New account is added to the system.
- **Fail: S**hows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to Register view	Register view is shown with following labels and fields: • Username: text, required, length 3 – 50 • Password: text, required, length 3 – 50 • Confirm password: text, required, length 3 – 50 • Fullname: text, required, length 3 – 50 • Address: text, required, length 3 – 250 • City: dropdown list, require • Email: text, required, length 8 – 90 • Phone number: text, required, length 7 – 12
2	Guest fill out the form	
3	Guest sends register command.	Validate data [Exception 1, 2, 3, 4, 5] Add new account to the system. Display register success message.

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	Missing of required fields	Show message notifying guest inputed missed fields.
2	Length of field's value is out of range	Show message notifying guest which field's value is out of range.
3	Password and Confirm password don't match.	Show message notifying Password and Confirm password don't match.
4	Entered email address is not a valid email	Show message notify entered email is not valid
5	Entered email is existed in the system	Show message notify entered email is existed

USE CASE – UC_G.03			
Usecase No.	**UC_G.02**	**Usecase Version**	2.0
Usecase Name	Search product		
Author	KhoaNPA		
Date	17/01/20178	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to search products in the system

Goal:

- Guest can view products and its detail.
- Guest can simulate products through camera.

Triggers:

- Guest sends search command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** System shows a list of products or message "Don't have any product is matched with your requirement".
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to search view.	System suggests actor input information to find some products with actor's criteria: • "Product Name": text, require.
2	Guest sends search command	System shows list of products. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	System can not find any product for actor	System shows message "Don't have any product is matched with your requirement".

Relationships: <Guest> Get product's detail Usecase (G.07)

Business Rules:

- System finds product with approximate name.
- The product just only be showed, if its status is "active".

Relationships: N/A

Business Rules:

- In case of success scenarios, a new account would be added to the system.
- An email address must be validated by this regular expression: /^([a-z0-9_\-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/
- Phone number must be string of numbers.
- New account will be active in the system and has role Customer when it is created successfully.

2.3.3.3 < Guest > Search product Usecase

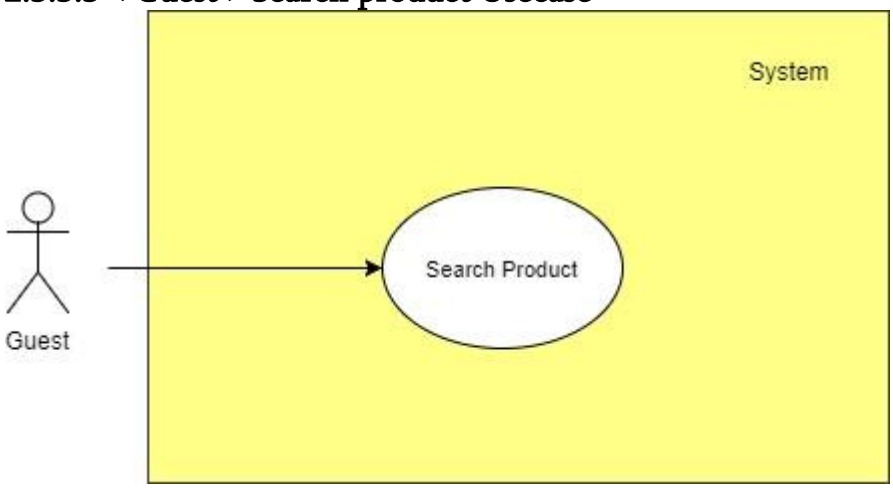


Figure 14: < Guest > Search product Usecase

2.3.3.4 < Guest > Add to cart Usecase

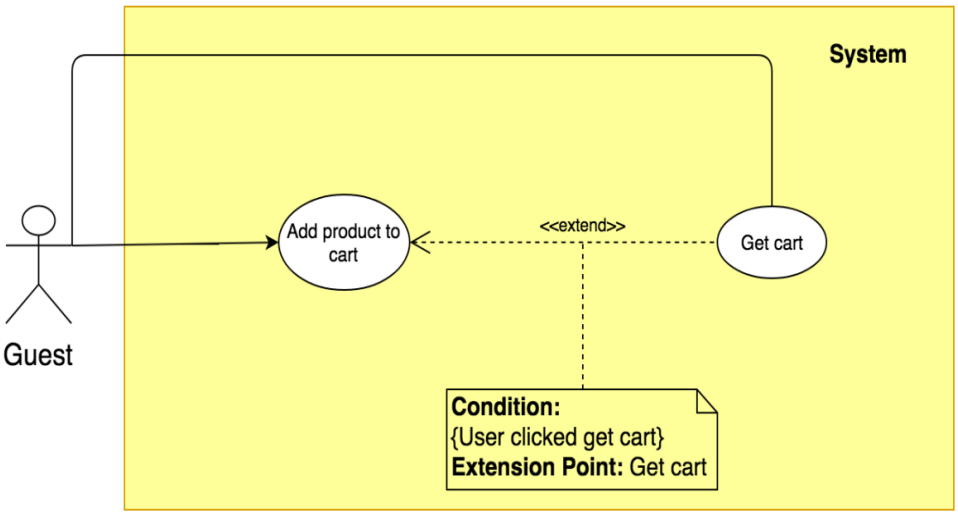


Figure 15: < Guest > Add to cart Usecase

USE CASE – UC_G.04			
Usecase No.	**UC_G.04**	**Usecase Version**	1.0
Usecase Name	Add product to cart		
Author	KhoaNPA		
Date	17/01/20178	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to add product to cart.

Goal:

- New product is added in guest's cart.
- If this product is already existed in guest shopping cart, quantity of product will be increase by one.

Triggers:

- Guest sends add product to cart command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** This product is added to guest shopping cart.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to Product detail view.	

		Product detail view is show information includes: • Product name • Price • Product's image • Product specifications
2	Guest sends add to cart command to the system.	Systerm adds new product to guest shopping cart. Display add to cart success message. [Alternative 1]

Alternative Scenario:

No	Cause	System Response
1	This product is already existed in guest'cart.	System increases this product's quantity in guest shopping cart by one.

Exceptions: N/A

Relationships: <Guest> Get cart Usecase (G.5)

Business Rules:

- System must notify added action for guest.
- If the product is existed in guest's shopping cart, the quantity of the product is increased by one.

2.3.3.5 < Guest > Get cart Usecase

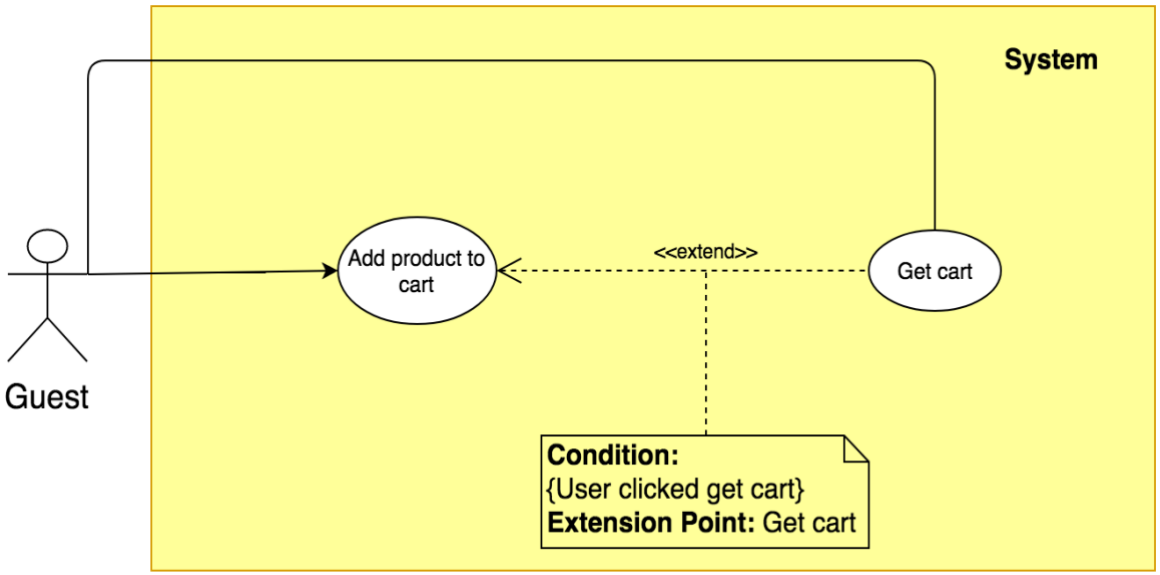


Figure 16: < Guest > Get cart Usecase

USE CASE – UC_G.05			
Usecase No.	**UC_G.05**	**Usecase Version**	2.0
Usecase Name	Get cart		
Author	KhoaNPA		
Date	17/01/20178	**Priority**	Normal

Actor:
- Guest

Summary:
- This use case allows to show guest their cart information.

Goal:
- Guest can view their cart and its details.
- Guest can change the quantity or remove product in their cart.
- System show total price of shopping cart.

Triggers:
- Guest sends get cart command to the system.

Preconditions: N/A

Post Conditions:
- **Success:** System shows the guest shopping cart information.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest sends get cart command	Display cart view with each product's

		details: • Product image • Procduct name • Price • Quantity and total price. [Alternative 1]

Alternative Scenario:

No	Cause	System Response
1	There is no product in shopping cart.	Show message to notify that there is no product in shopping cart.

Exceptions: N/A

Relationships:

- <Guest> Add to cart Usecase (G.04)
- <Guest> Checkout Usecase (G.06)

Business Rules:

- The information is loaded from the system.
- List of products is sorted by order date.
- In each product, guest can change the quantity of products or remove products from cart.
- The price in each product would be price of product.
- The total price would be calculated by price of all products.
- If product is changed about the quantity or is removed, the total price will be reloaded automatic.

2.3.3.6 < Guest > Simulate product through camera Usecase

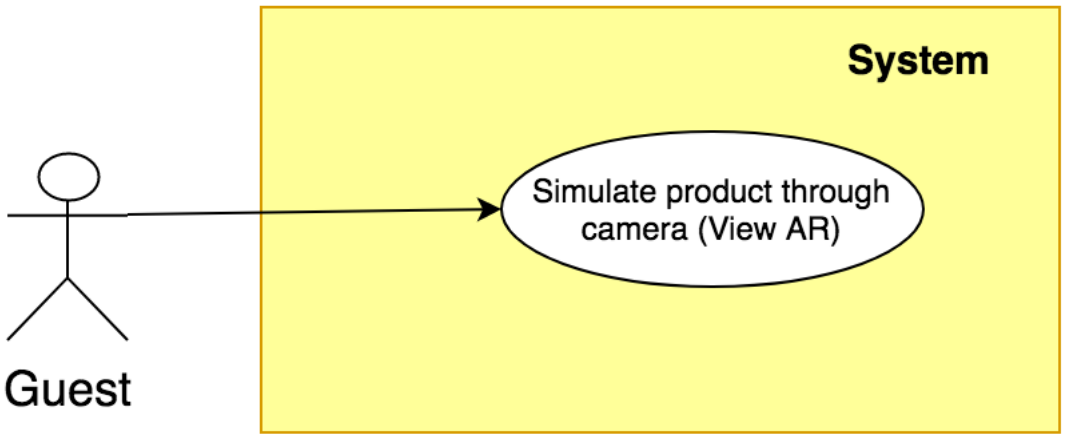


Figure 17: <Guest> Simulate product through camera Usecase

USE CASE – UC_G.07			
Usecase No.	**UC_G.06**	**Usecase Version**	2.0
Usecase Name	Simulate product through camera (View AR)		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to simulate product in real-word through camera.

Goal:

- Product's 3D model is placed in flast surface or plane through camera.
- Guest can use fingers to move or rotating the 3D model.

Triggers:

- Guest sends command to simulate furniture.

Preconditions: N/A

Post Conditions:

- **Success:** 3D model of product is putted to flast surface.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to simulate view	System loaded list of products.
2	Guest chooses a product and sends command to simulate.	System passess through the camera. System displays 3D model of the product into camera monitor. System detects plane, flat surface. [Exception 1]
3	Guest using one finger to move this object to planes, flat surfaces. [Alternative 1, 2]	System tracks position of this object. [Exception 2]

Alternative Scenario:

No	Cause	System Response
1	If guest want to simulate other furniture, sends command to simulater furniture.	System forwards to simulate view.
2	Guest using two fingers to	

	rotate the object.	System holds fixed the object on plane.
3	Guest can move the phone through.	System tracks and hold fixed the object on the position where it is putted.

Exceptions:

No	Cause	System Response
1	System can not load 3D model of this product.	System shows error message.
2	System can not detect the plane, flat surface.	System shows error message.

Relationships: <Guest> Get product's detail

Business Rules:

- System allows the phone to understand and track its position relate to the world.
- System allows the phone to detect the size and location of flat horizontal surfaces.
- System will hold fixed the 3D model in their position, the object won't move while user moves the phone through.
- User can interact (move, rotate) with objects by using their fingers.

2.3.3.7 < Guest > Get product's detail.

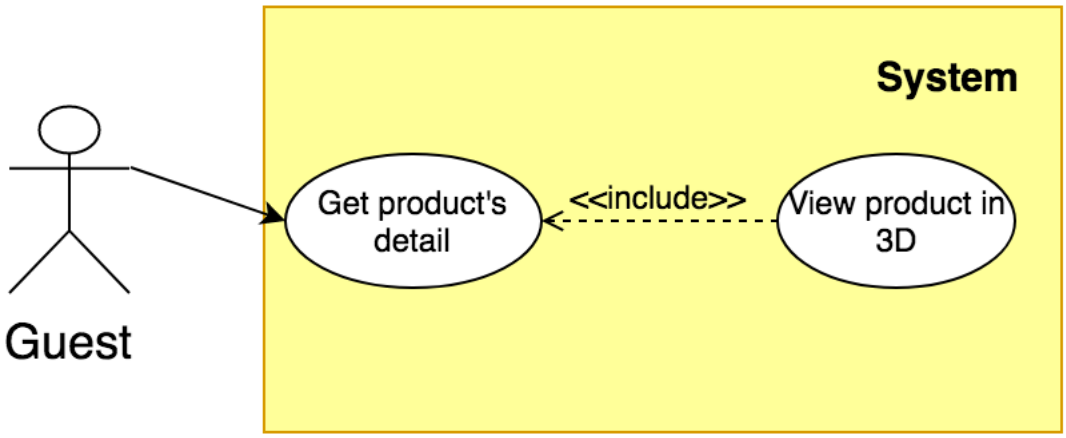


Figure 18: <Guest> Get product's detail.

USE CASE – UC_G.07			
Usecase No.	**UC_G.07**	**Usecase Version**	2.0
Usecase Name	ThienBT.		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:

- Guest

Summary:

- This use case allows guest to get detail of product.

Goal:

- Guest can view information about product.

Triggers:

- Guest sends command to get product's detail.

Preconditions: N/A

Post Conditions:

- **Success:** Product's detail informations is displayed for guest.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to products view	System loaded list of products.
2	Guest chooses a product and sends command to get detail.	System displays product's detail information. [Exception 1]

Alternative Scenario:

No	Cause	System Response
1	If guest want to simulate product through camera (View AR), sends command to simulater product.	System forwards to simulate view.
2	If guest want to view product in 3D, sends command to view product in 3D.	System displays product's in 3D.

Exceptions:

No	Cause	System Response
1	System can not load product informations.	System shows error message.

Relationships:

- <Guest> Search product (G.02)
- <Guest> Simulate product through camera (View AR) (G.06)

Business Rules:

- System allow guest to view product's detail.

- Guest can view information of product.

2.3.3.8 < Guest > View product in 3D.

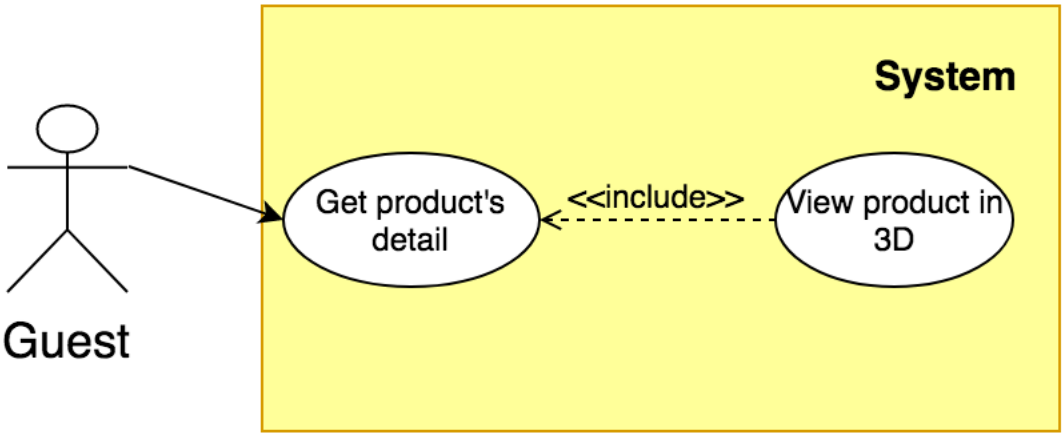


Figure 19: < Guest > View product in 3D

USE CASE – UC_G.07			
Usecase No.	**UC_G.08**	**Usecase Version**	2.0
Usecase Name	View product in 3D.		
Author	ThienBT		
Date	17/01/2018	**Priority**	Normal

Actor:
- Guest

Summary:
- This use case allows guest to view product in 3D.

Goal:
- Guest can view intuitive and overall look in all directions of the product.

Triggers:
- Guest sends command to view product in 3D.

Preconditions:
- Guest get product's detail.

Post Conditions:
- **Success:** Guest view product in 3D.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Guest chooses a product and sends command to get detail	System displays product's detail information.
2	Guest sends command to	

	view product in 3D.	System load product and show product in 3D through mobile screen.

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	System can not load product in 3D.	System shows error message.

Relationships:

- <Guest> Get product's detail. (G.07)

Business Rules:

- System allow guest to view product in 3D.
- Guest can rotate product by using their fingers to view intuitive and overall look in all directions of the product.

2.3.4 < Customer > Overview Usecase

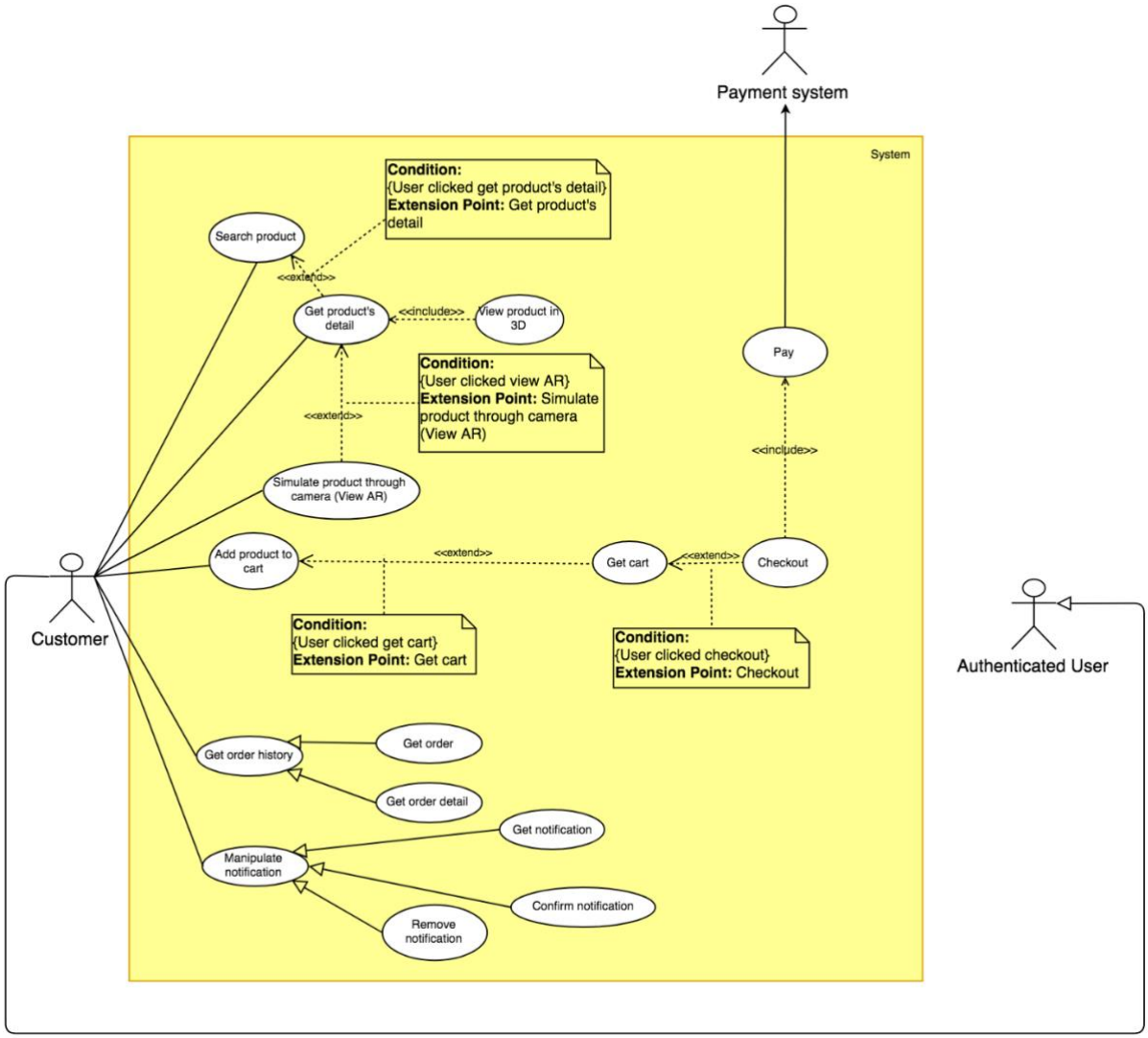


Figure 20: < Customer > Overview Usecase

2.3.4.1 < Customer > Search Product Usecase

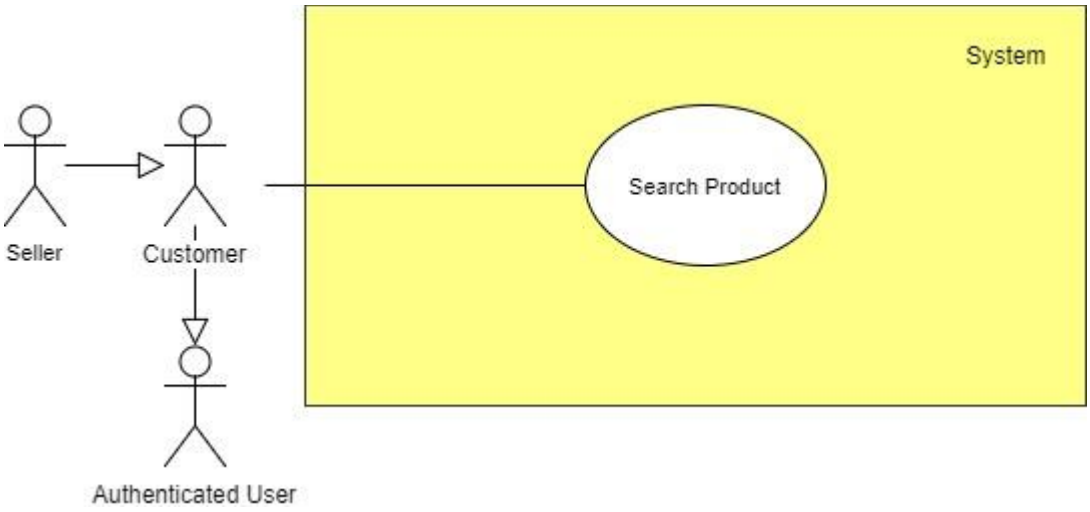


Figure 21:<Customer> Search product Usecase

USE CASE – UC_C.01			
Usecase No.	**UC_C.01**	**Usecase Version**	2.0
Usecase Name	Search product		
Author	DucPH		
Date	17/01/20178	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows Customer to search products in the system

Goal:

- Customer can view products and its detail.

Triggers:

- Customer sends search command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** System shows a list of products.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to search view.	System suggest actor input information to find some products with actor's criteria: • "Product Name": text, required.
2	Customer send search command	System shows list of products.

		[Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	System can not find any product for actor	System shows message "No result found".

Relationships: N/A

Business Rules:

- System finds product with approximate name.
- The product just only be showed, if its status is "Approved".

2.3.4.2 < Customer > Add to cart Usecase

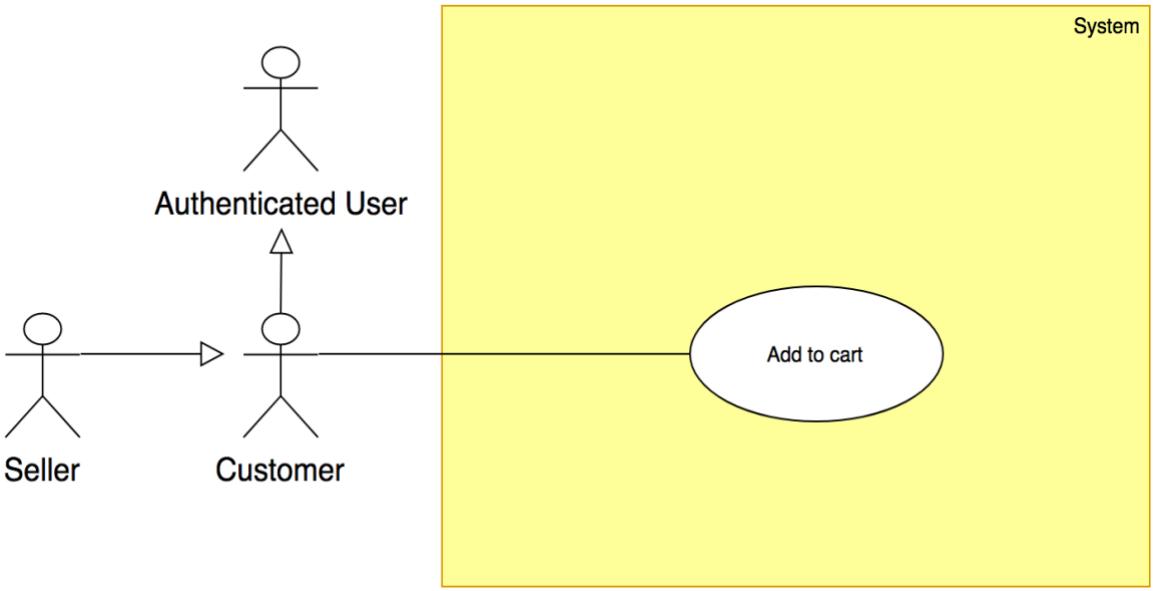


Figure 22: <Customer> Add to cart Usecase

USE CASE – UC_C.02			
Usecase No.	**UC_C.02**	**Usecase Version**	2.0
Usecase Name	Add to cart		
Author	DucPH		
Date	17/01/2018	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customer to add product to their cart.

Goal:

- New product is added in shopping cart.
- If this product is already existed in shopping cart, quantity of product will be

increase by one

Triggers:

- Customer send add product to cart command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** This product is added to customer's cart.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer sends add to cart command to the system.	System adds new product to customer shopping cart.

Alternative Scenario: N/A

Exceptions: N/A

Relationships: <Customer> Get cart Usecase (C.02)

Business Rules:

- If the product is existed in customer's shopping cart, the quantity of the product is increased by one.

2.3.4.3 < Customer > Get cart Usecase

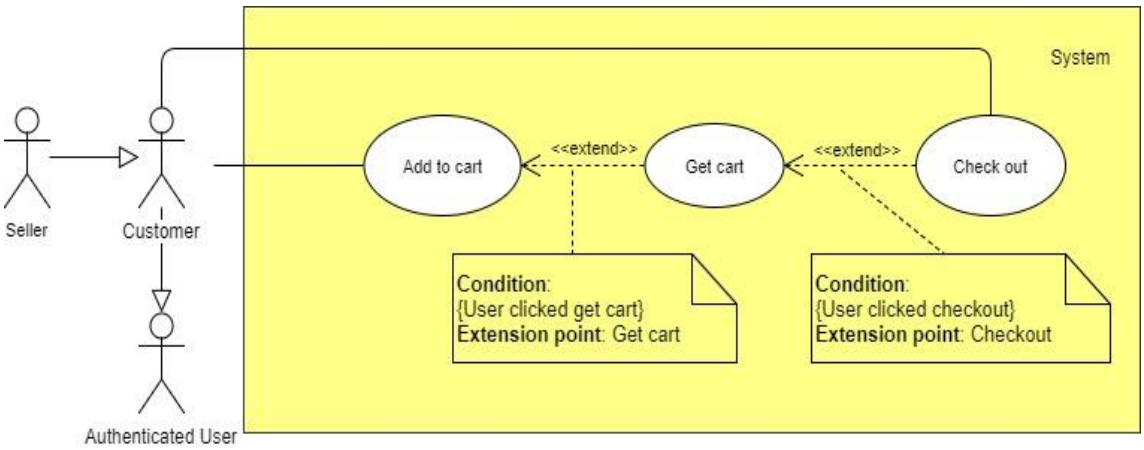


Figure 23: <Customer> Get cart Usecase

USE CASE – UC_C.03			
Usecase No.	**UC_C.03**	**Usecase Version**	2.0
Usecase Name	Get cart		
Author	DucPH		
Date	17/01/20178	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows to show customer's shopping cart.

Goal:

- Customer can view their shopping cart and its details.
- Customer can change the quantity or remove product in their shopping cart.
- System show total price of shopping cart.

Triggers:

- Customer sends get cart command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** System shows the customer'cart information.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer sends get cart command	Display cart view with each product's details: • Product image: JPG • Product name: text • Product provider: text • Product price: numner • Product quantity: number • Total price: number [Alternative 1]

Alternative Scenario:

No	Cause	System Response
1	There is no product in shopping cart.	Display cart view with message: "Your cart is empty".

Exceptions: N/A

Relationships:

- <Customer> Add to cart Usecase. (C.02)
- <Customer> Checkout Usecase. (C.04)

Business Rules:

- The information is loaded from the system.
- In each product, customer can change the quantity of products or remove products from cart.
- The price in each product would be price of product.
- The total price would be calculated by price of all products.

- If product is changed about the quantity or is removed, the total price will be reloaded automatic.

2.3.4.4 < Customer > Checkout Usecase

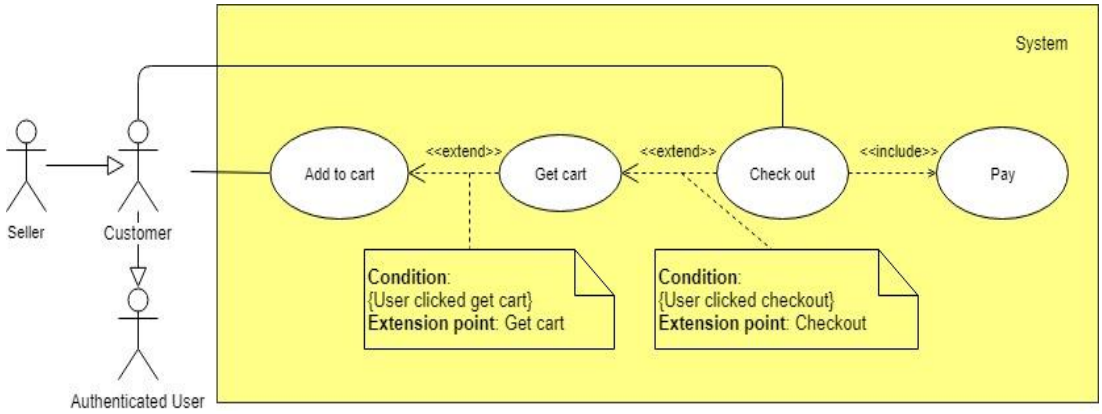


Figure 24: < Customer > Checkout Usecase

USE CASE – UC_C.04			
Usecase No.	**UC_C.04**	**Usecase Version**	1.0
Usecase Name	Checkout		
Author	DucPH		
Date	17/01/2018	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customer to check out their shopping cart.

Goal:

- Customer can fill out the delivery address (default is customer's address).
- Customer can choose the payment gateways.
- System shows shopping cart's information with total price.

Triggers:

- Customer sends check out command.

Preconditions:

- There is at least 1 product in customer shopping cart.

Post Conditions:

- **Success:** System shows customer shopping cart details and payment.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to check out view	Display customer information as delivery address view with following labels and

		fields: • Full name: text input, required, length 3-80. • Address: text input, required, length 3-250. • Phone number: number input, required, length 10-11. [Exception 1]
2	Customer sends processing payment command. [Alternative 1, 2]	Display new view let user select one of following payment methods: • Paypal payment. • COD payment.
3	If customer chooses Paypal payment. [Alternative 1]	Forward to Paypal payment view to process payment. [Exception 2]

Alternative Scenario:

No	Cause	System Response
1	If customer chooses COD payment.	System shows shopping cart's information with total fee.

Exceptions:

No	Cause	System Response
1	Customer sends processing payment command.	System shows error message to ask customer input missing required fields.
2	If customer chooses Paypal payment.	Show message to notify customer that payment failed.

Relationships:

- <Customer> Get cart Usecase (C.03)
- <Payment System> Pay Usecase (PS.01)

Business Rules:

- Transaction code is automatic initialized by the system.
- The fee in each product would be calculated by price multiplied by quantity.

- Total fee would be calculated by products fee.

2.3.4.5 < Customer > Simulate product through camera Usecase

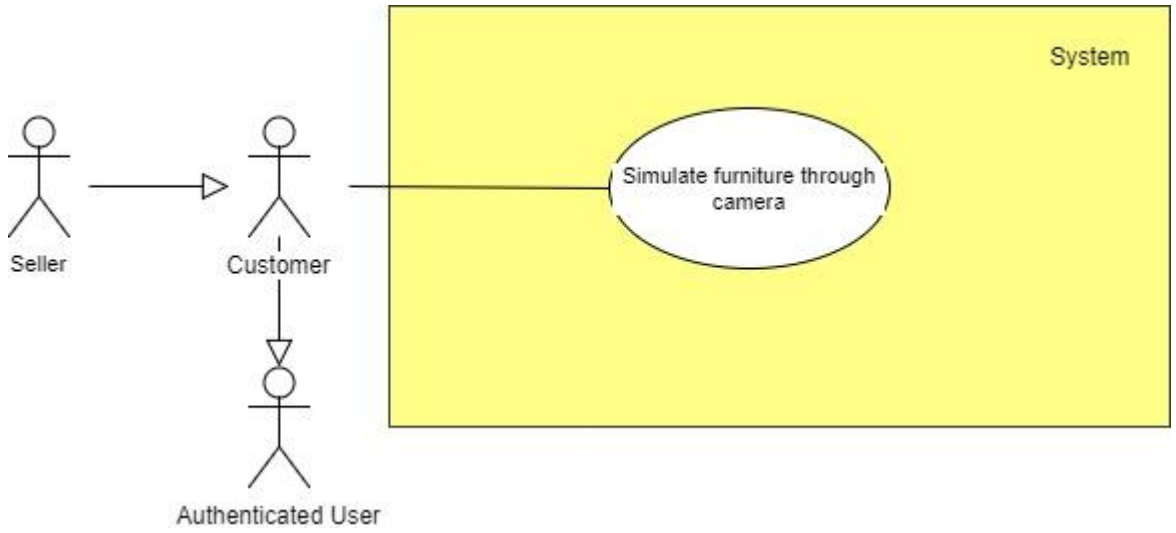


Figure 25: < Customer > Simulate product through camera Usecase

USE CASE – UC_C.05			
Usecase No.	**UC_C.05**	**Usecase Version**	2.0
Usecase Name	Simulate furniture through camera		
Author	DucPH		
Date	17/01/2018	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customer put 3D model of product in real-word through camera.

Goal:

- Product's 3D model is putted in flast surface or plane through camera.
- Customer can use fingers to move or rotate the 3D model in camera.

Triggers:

- Customer sends command to simulate furniture.

Preconditions: N/A

Post Conditions:

- **Success:** 3D model of product is putted to flat surface.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to product	

	view	System loaded list of products.
2	Customer chooses a product and sends command to simulate.	System detect planes, flat surfaces. System display 3D model of the product into camera monitor. [Exception 1]
3	Customer using one finger to move or rotate this object to planes, flat surfaces. [Alternative 1, 2,3]	System tracks position of this object. [Exception 2]

Alternative Scenario:

No	Cause	System Response
1	If guest wants to simulate other furniture, sends command to simulator furniture.	System forward to simulate view.
2	Customer using two fingers to rotate the object.	System holds fixed the object on plane.
3	Customer can move the phone through.	System tracks and hold fixed the object on position where it is putted.

Exceptions:

No	Cause	System Response
1	System can not load 3D model of this product.	System shows error message.
2	System can not detect the plane or flat surface.	System shows error message.

Relationships: N/A

Business Rules:

- System allows the phone to understand and track its position relate to the world.
- System allows the phone to detect the size and location of flat horizontal surfaces.
- System will hold fixed the 3D model in their position, the object won't move while user move the phone through.
- User can interact (move, rotate) with objects by using their fingers.

2.3.4.6 < Customer > Get product's detail Usecase

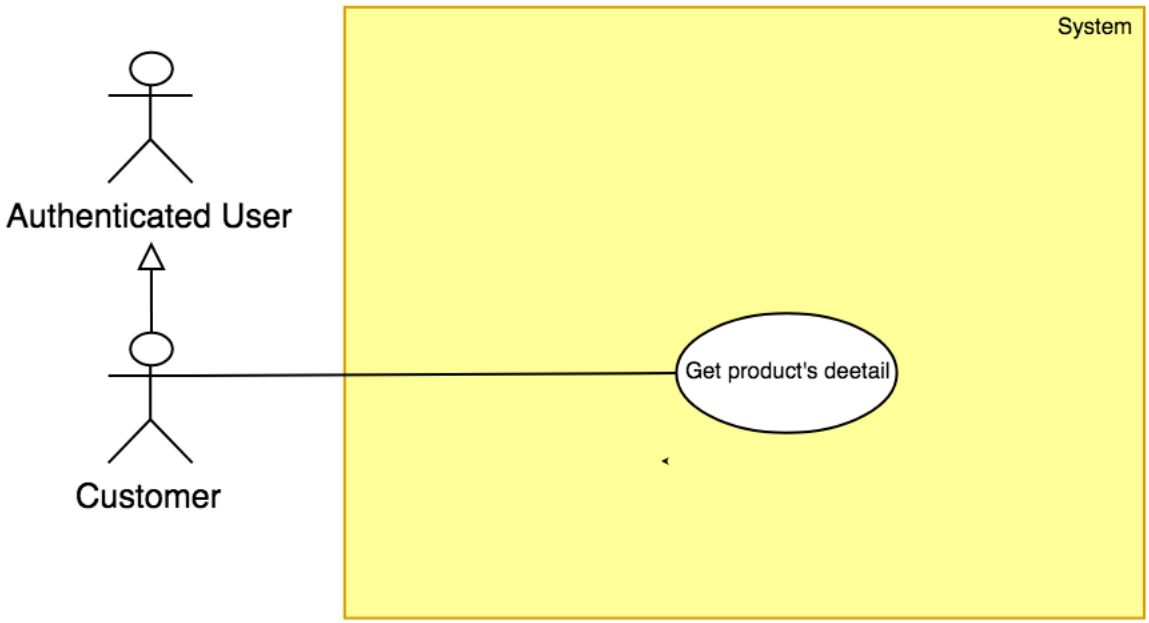


Figure 26: < Customer > Get product's detail Usecase

USE CASE – UC_C.05			
Usecase No.	**UC_C.05**	**Usecase Version**	2.0
Usecase Name	Get product's detail		
Author	DucPH		
Date	17/01/2018	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customer to get product's detail.

Goal:

- Customer can view product's detail.

Triggers:

- Customer sends view product's detail command.

Preconditions: N/A

Post Conditions:

- **Success:** Product's detail information is shown.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to Manipulate order view.	System shows list of products.
2	Seller selects order to view their detail information.	Display a view with product's information: • Name. • Origin. • Category. • Material. • Store. • Length. • Width. • Height. • Weight. • Price. • Quantity. • Describe. • Price.

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- Product's information is always loaded from the system.

2.3.4.7 < Customer > View product in 3D Usecase

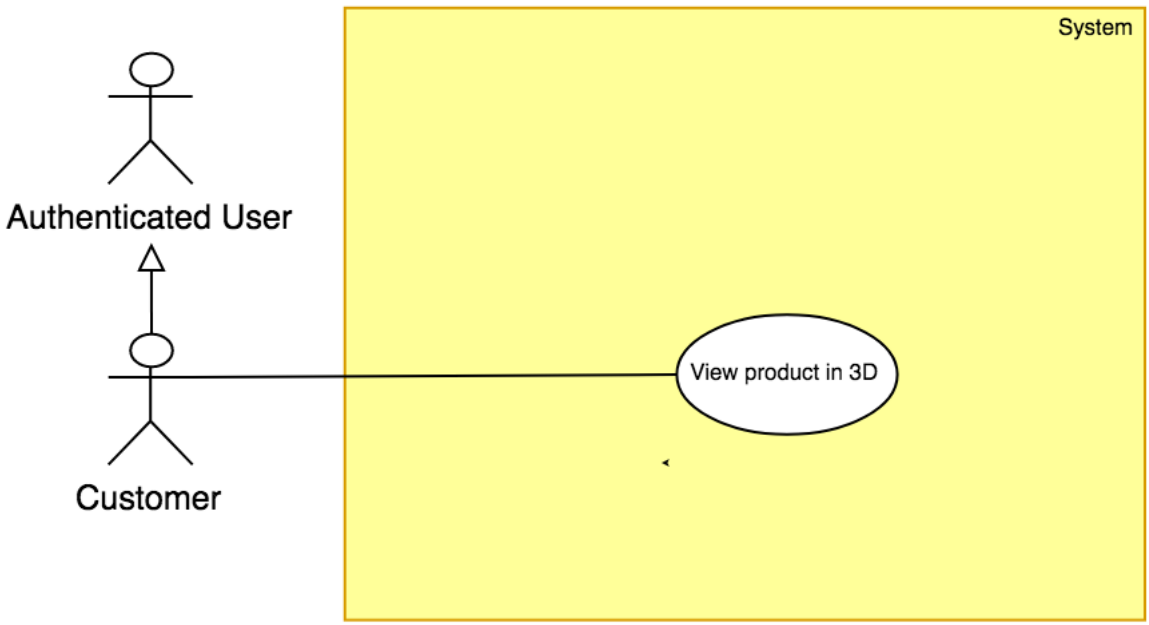


Figure 27: < Customer > View product in 3D Usecase

USE CASE – UC_C.05			
Usecase No.	**UC_C.05**	**Usecase Version**	2.0
Usecase Name	View product in 3D		
Author	DucPH		
Date	17/01/2018	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customer to view product in 3D.

Goal:

- Customer can view product in 3D.

Triggers:

- Customer sends view product in 3D command.

Preconditions: N/A

Post Conditions:

- **Success:** View product in 3D is shown.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to View product view.	System shows product in 3D.

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- Product's 3D model is always loaded from the system.

2.3.4.8 < Customer > Get order Usecase

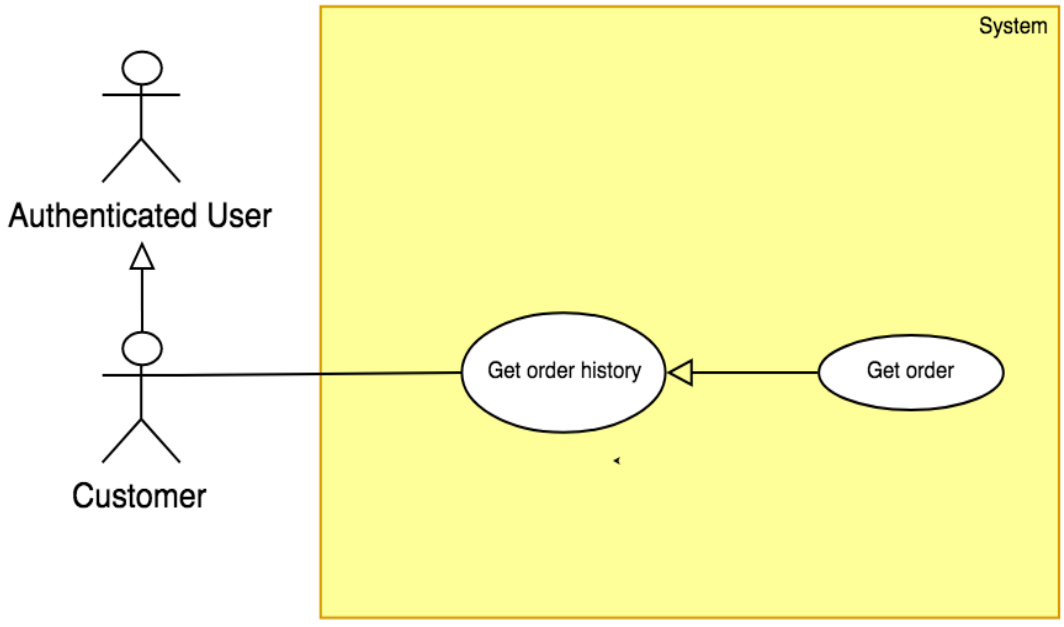


Figure 28: < Seller > Get order Usecase

USE CASE – UC_SL.04			
Usecase No.	**SL.04**	**Usecase Version**	1.0
Usecase Name	Get order		

Author	DucPH		
Date	18/01/2018	Priority	Normal

Actor:

- Customer.

Summary:

- This use case allows customer to get their order details.

Goal:

- Customer can view their order.
- Customer can view order's detail information

Triggers:

- Customer sends view order's detail information command.

Preconditions:

- User must login into system with role Customer.

Post Conditions:

- **Success:** Order's detail information is shown.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to order history view.	Display a table shows list of order: • Order ID: text • Customer name: text • Product ordered: text • Purchased date: date time • Arrival date: date time • Total: number
2	Customer selects order to view their detail information.	Display a table shows list of products in order: • Product ID: text • Quantity: number • Price: number

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- List of order and their detail information is always loaded from the system.

2.3.4.9 < Customer > Get order detail Usecase

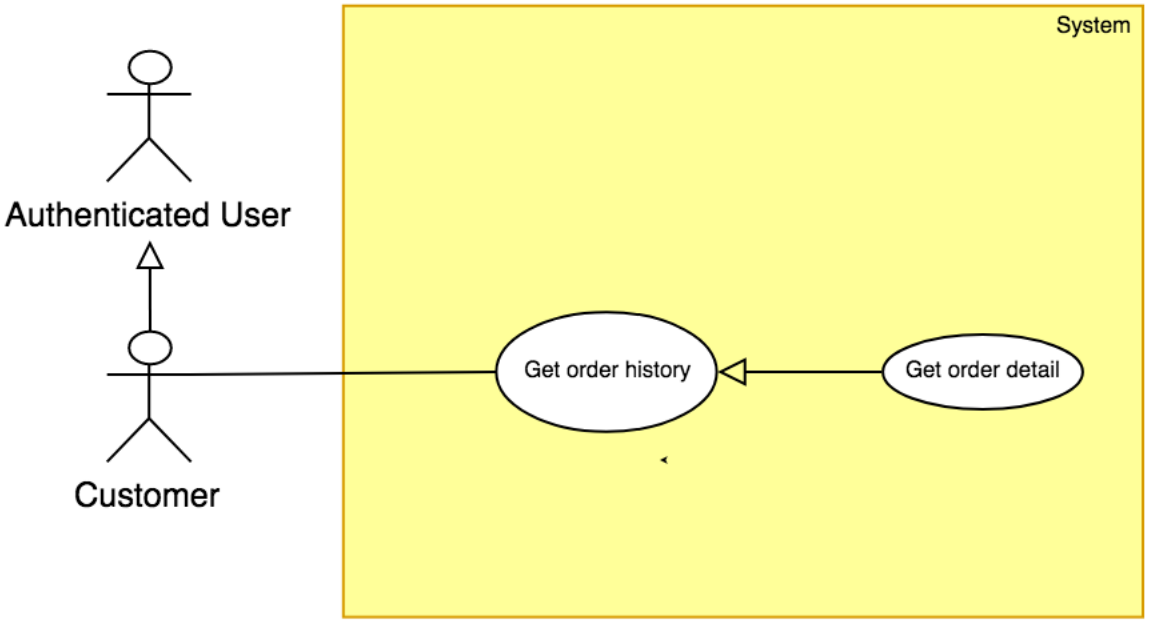


Figure 29: < Seller > Get order detail Usecase

USE CASE – UC_SL.04			
Usecase No.	**SL.04**	**Usecase Version**	1.0
Usecase Name	Get order detail		
Author	DucPH		
Date	18/01/2018	**Priority**	Normal

Actor:

- Customer.

Summary:

- This use case allows customer to get their order detail.

Goal:

- Customer can view their order detail.
- Customer can view order detail.

Triggers:

- Customer sends view order detail command.

Preconditions:

- User must login into system with role Customer.

Post Conditions:

- **Success:** Order detail information is shown.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to order history view.	Display a order detail information: • Product ID: text • Quantity: number • Price: number

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- Order detail information is always loaded from the system.

2.3.5.0 < Customer > Get notification Usecase

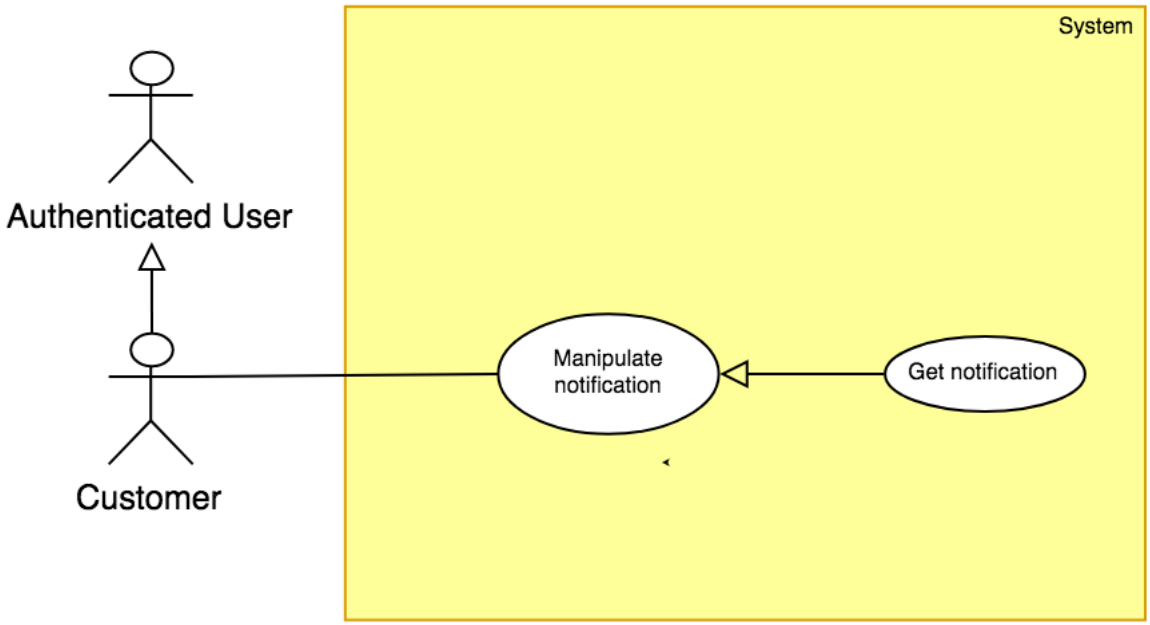


Figure 30: <Customer> Get notification Usecase

USE CASE – UC_C.03			
Usecase No.	**UC_C.03**	**Usecase Version**	2.0
Usecase Name	Get notification		
Author	DucPH		
Date	17/01/20178	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows to show notification's information.

Goal:

- Customer can view their notification's detail.

Triggers:

- Customer sends get notification command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** System shows the notification's information.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer sends get notification command	Display a table shows list of notifications

Alternative Scenario:

No	Cause	System Response

1	There is no notification.	Display cart view with message: "No notification yet".

Exceptions: N/A
Relationships: N/A
Business Rules:
- List of notifications is always loaded from the system.

2.3.5.1 < Customer > Confirm notification Usecase

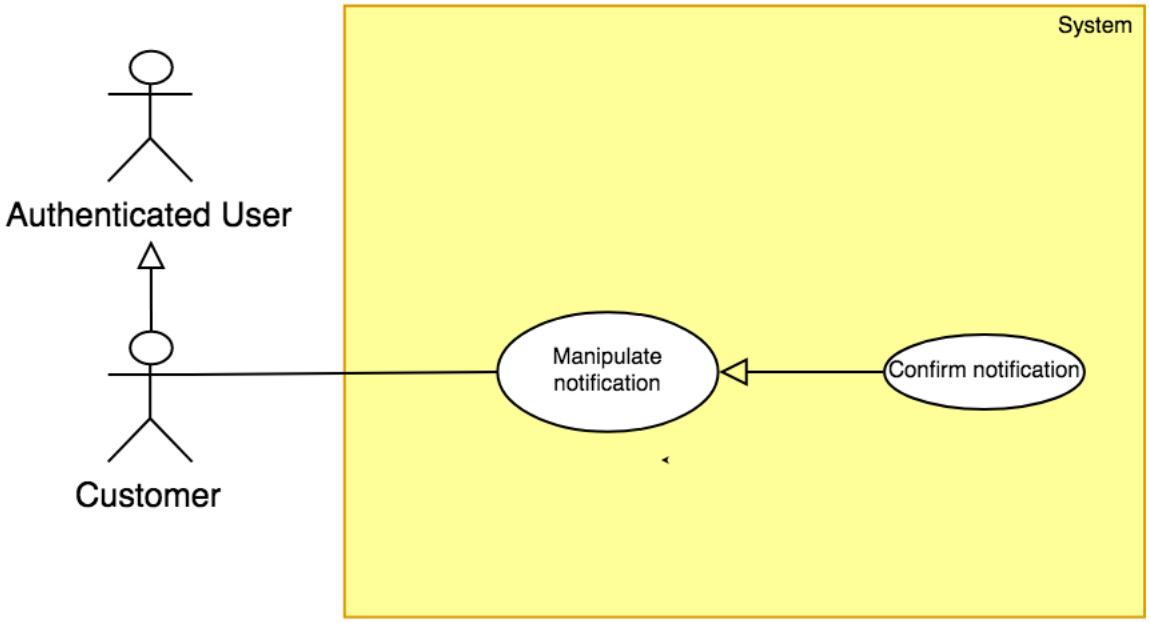


Figure 31: <Customer> Confirm notification Usecase

USE CASE – UC_C.03			
Usecase No.	**UC_C.03**	**Usecase Version**	2.0
Usecase Name	Confirm notification		
Author	DucPH		
Date	17/01/20178	**Priority**	Normal

Actor:
- Customer

Summary:
- This use case allows to confirm notification.

Goal:
- Customer can confirm their notification.

Triggers:
- Customer sends confirm notification command to the system.

Preconditions: N/A

Post Conditions:
- **Success:** System update the notification status.

- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to "Notification view"	Display a table shows list of notifications
2	Customer sends confirm notification command	

Alternative Scenario: N/A
Exceptions: N/A
Relationships: N/A
Business Rules:

- Notification seen value will be set to 1.

2.3.5.2 < Customer > Remove notification Usecase

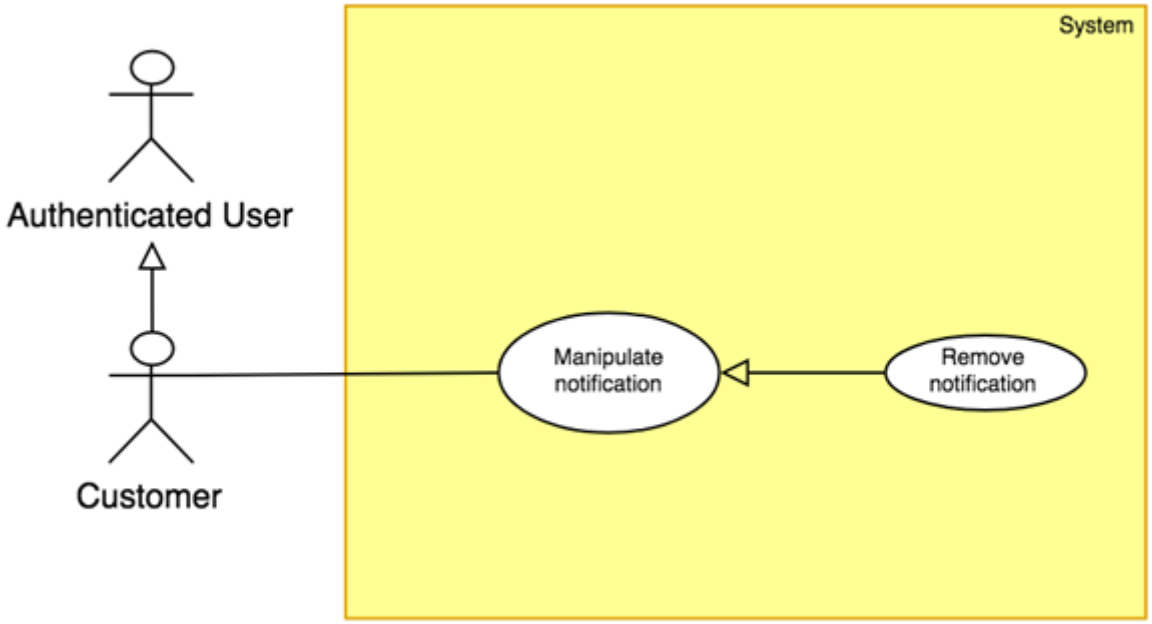


Figure 32: <Customer> Remove notification Usecase

USE CASE – UC_C.03			
Usecase No.	**UC_C.03**	**Usecase Version**	2.0
Usecase Name	Remove notification		
Author	DucPH		
Date	17/01/20178	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows to remove notification.

Goal:

- Customer can remove their notification.

Triggers:

- Customer sends remove notification command to the system.

Preconditions: N/A

Post Conditions:

- **Success:** System removes the notification.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer goes to "Notification view"	Display a table shows list of notifications.
2	Customer choose notification to remove	
3	Customer send remove notification command to the system	System notifies removing notification successfully.

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- Notification will be removed from system.

2.3.5 < Seller > Overview Usecase

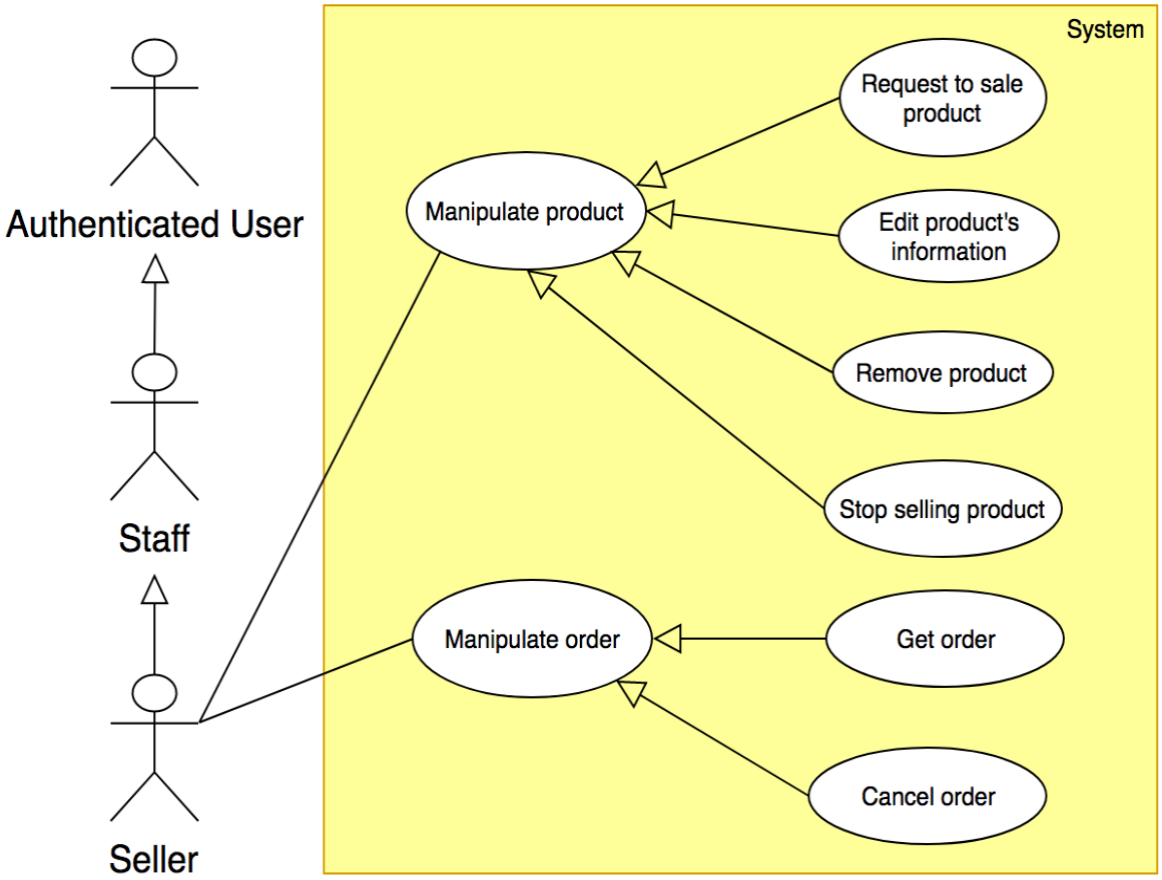


Figure 33: <Seller> Overview UseCase

2.3.5.1 < Seller > Request to sale product Usecase

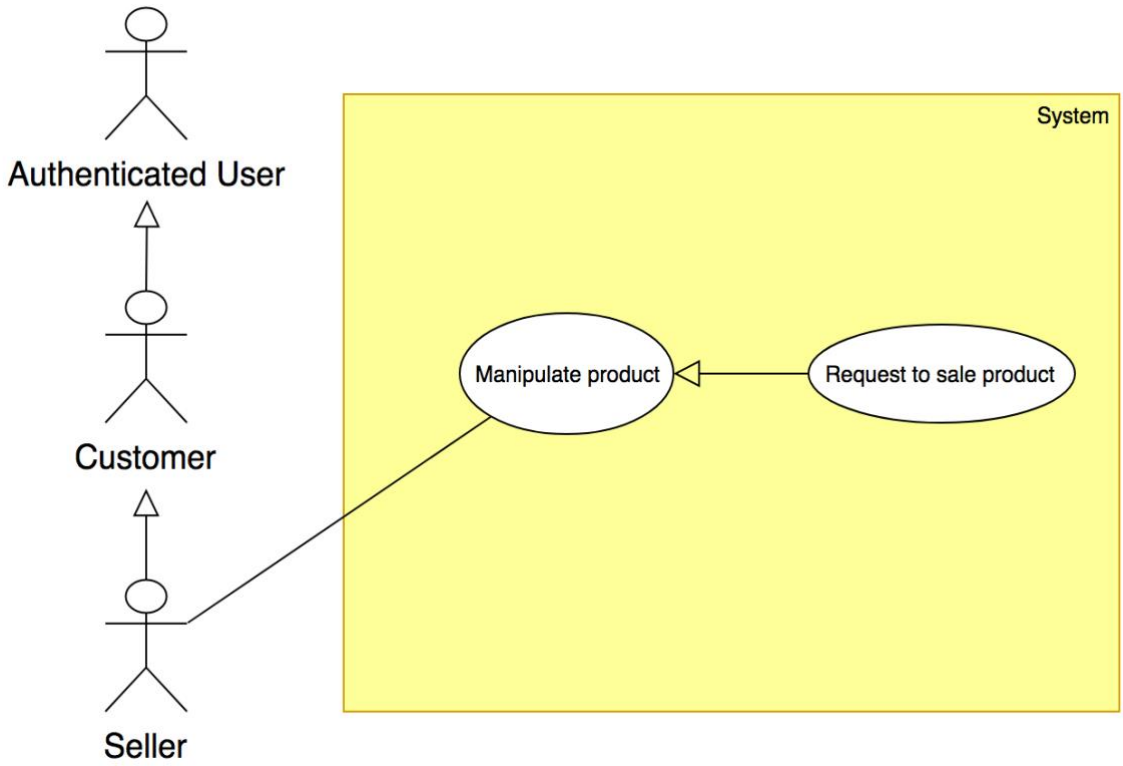


Figure 34: < Seller > Request to sale product Usecase

USE CASE – UC _SL.01			
Usecase No.	SL.01	**Usecase Version**	1.0
Usecase Name	Request to sale product		
Author	DucPH		
Date	17/01/2018	**Priority**	High

Actor:

- Seller.

Summary:

- This usecase allows seller to request to sale new product.

Goal:

- 3D model can be generated from series of picutres.
- 3D model is used to demonstrate seller's product on marketplace.
- New seller's product request will be sent.

Triggers:

- Seller sends sale product command.

Preconditions:

- User must login into system with role as Seller.
- Balance must be higher than 3D generating fee.

Post Conditions:

- **Success:** Request sale new product will be sent.
- **Fail:** System shows error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to sale new product view	System return "Create product" view with following required fields: Display new view require user input some information: • Name: text, required, length 1-100. • Origin: text, required, length 1-50. • Category: text, required, length 1-50. • Material: text, required, length 1-50. • Store: text, required, length 1-50. • Length: number (greater than 0) • Width: number (greater than 0) • Height: number (greater than 0) • Weight: number (greater than 0) • Price: number (greater than 0) • Quantity: number (greater than 0) • Describe: text, required, length 1-1000. • Series of pictures: JPG format, ratio 3:4
2	Seller send "Sale product" command.	If request succeed: Show message request successful. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	If request failed	Show message to notify authenticated seller that request failed.

Relationships: N/A

Business Rules:

- The product request successfully only when staff approves request.
- When staff doesn't approve request, request status will change from "Pending" to "Rejected" with reason message.

- The format of the 2D picture and 3D model are JPEG and SCN respectively.
- At one time, one request is modified by one staff only.
- Designer will edit 3D model manually to upgrade model's quality.

2.3.5.2 < Seller > Remove product Usecase

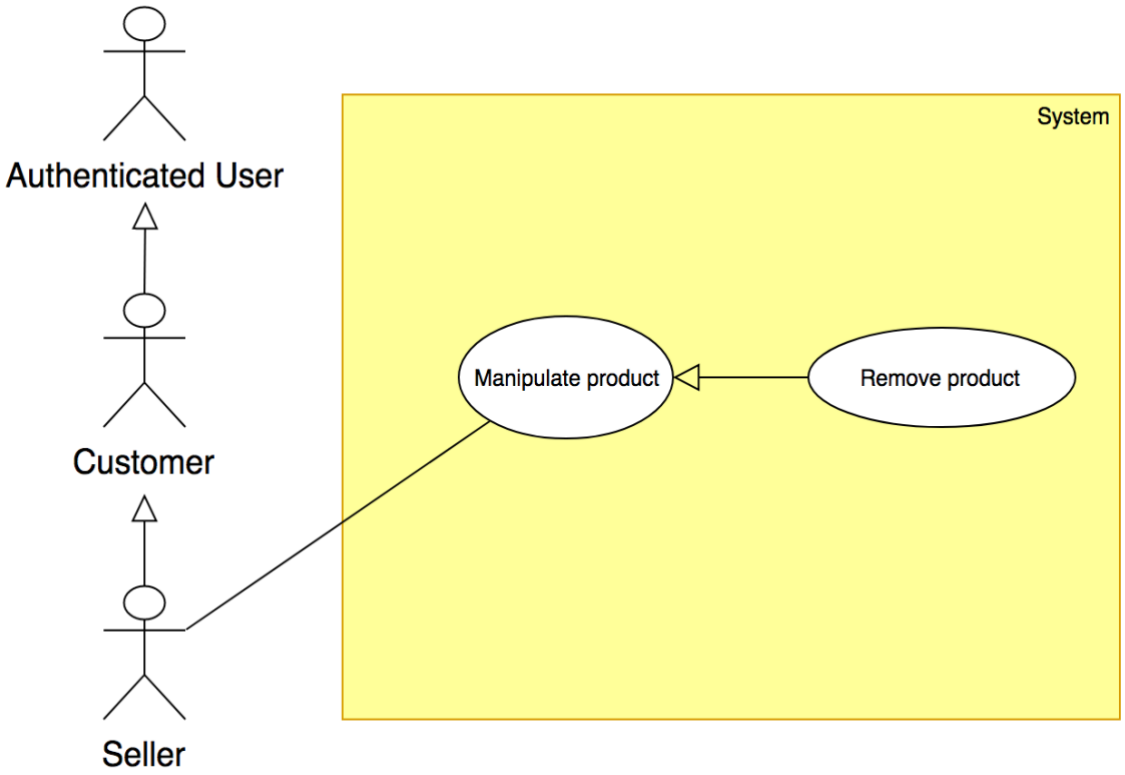


Figure 35: < Seller > Remove product Usecase

USE CASE – UC_SL.02			
Usecase No.	**SL.02**	**Usecase Version**	1.0
Usecase Name	Remove product		
Author	DucPH		

Date	17/01/2018	Priority	Normal

Actor:

- Seller

Summary:

- This use case allows seller to remove their product.

Goal:

- Product is removed from seller's storage and system's storage.

Triggers:

- Seller sends remove product command.

Preconditions:

- User must login into system with role Seller.

Post Conditions:

- **Success:** Product will be removed by seller.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to manipulate product view.	Display a table shows list of seller's products.
2	Seller chooses product to remove.	
3	Seller sends remove product command.	System starts removing product. System show message remove successful. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	If remove failed	Show message to notify user that remove failed.

Relationships: N/A

Business Rules:

- System will check is product created.
- Product must be existed in system, then product will be removed.
- Search bar on the top help user finding product faster.

2.3.5.3 < Seller > Edit product's information Usecase

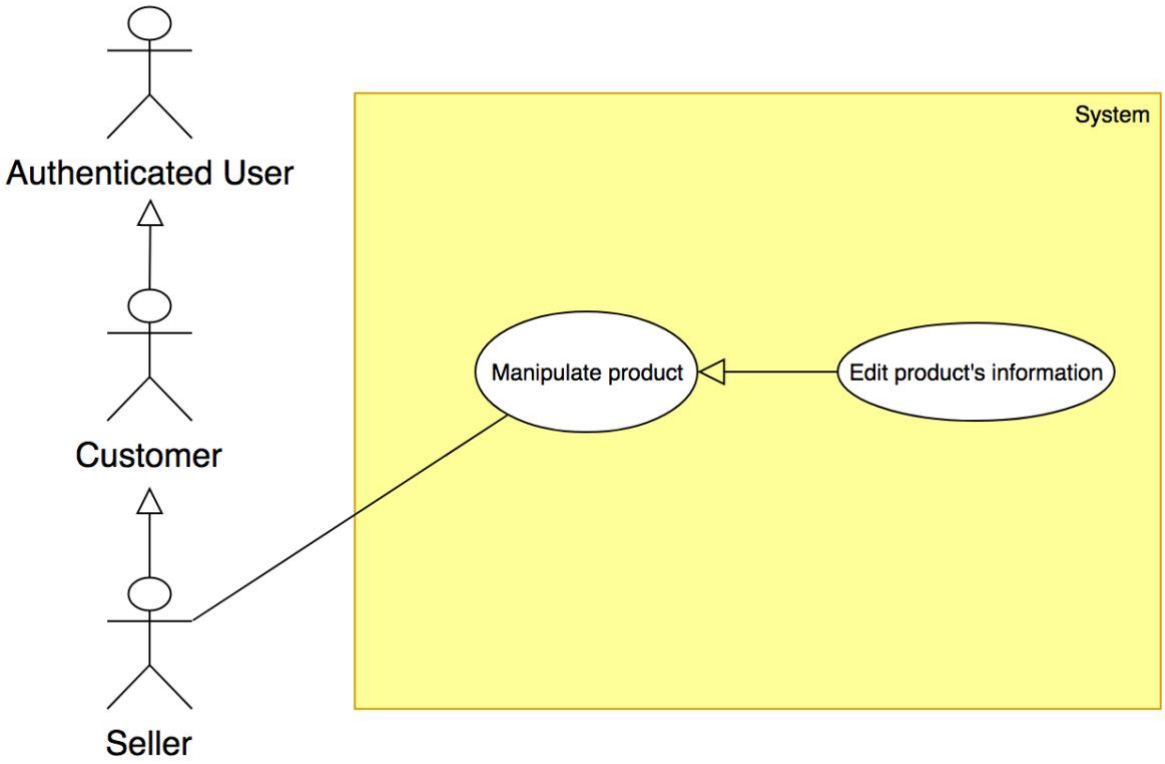


Figure 36: < Seller > Edit product's information Usecase

USE CASE – UC_SL.03			
Usecase No.	SL.03	**Usecase Version**	1.0
Usecase Name	Edit product's information		
Author	DucPH		
Date	17/01/2018	**Priority**	High

Actor:
- Seller

Summary:
- This use case allows seller to update their product's information.

Goal:
- Seller can update their product's information.

Triggers:
- Seller sends sell update product's information command.

Preconditions:
- User must login into system with role Seller.

Post Conditions:
- **Success:** Product's information will be updated by seller.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to manipulate product view.	Display a table shows list of seller's product.
2	Seller chooses product to edit.	Display new view require user input some information: • Name: text, required, length 1-100. • Origin: text, required, length 1-50. • Category: text, required, length 1-50. • Material: text, required, length 1-50. • Store: text, required, length 1-50. • Length: number (greater than 0) • Width: number (greater than 0) • Height: number (greater than 0) • Weight: number (greater than 0) • Price: number (greater than 0) • Quantity: number (greater than 0) • Describe: text, required, length 1-1000. 3D model: file upload input.
3	Seller sends update product's information command.	System starts update product. System show message update successful. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	If update product failed	Show message to notify user that update product's information failed.

Relationships: N/A

Business Rules:

- The product's information is loaded from the system.
- List of products will be sorted by date in descending order.
- Weight unit is kilogram (kg).
- Width, Length, Height unit is millimeter (mm).

2.3.5.4 < Seller > Stop selling product Usecase

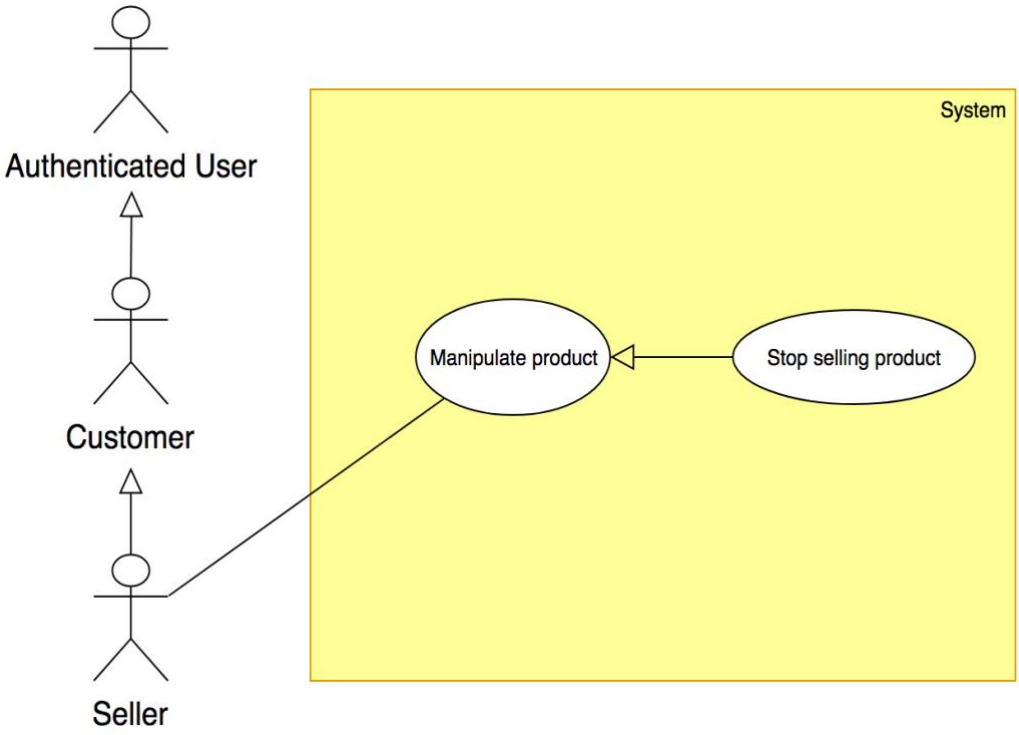


Figure 37: < Seller > Stop selling product Usecase

USE CASE – UC_SL.04			
Usecase No.	**SL.04**	**Usecase Version**	1.0
Usecase Name	Stop selling product		
Author	DucPH		
Date	18/01/2018	**Priority**	Normal

Actor:

- Seller.

Summary:

- This use case allow seller to stop selling product.

Goal:

- Seller can stop selling their product.

Triggers:

- Seller sends stop selling product command.

Preconditions:

- User must login into system with role Seller.

Post Conditions:

- **Success:** Product's status will be updated by seller.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to manipulate product view.	Display a table shows list of seller's products.
2	Seller chooses their product to stop selling.	
3	Seller sends stop selling product command. [Exception 1]	System starts update product's status. System show message stop selling product successful. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	If stop selling product failed	Show message to notify user that stop selling product failed.

Relationships: N/A

Business Rules:

- The information is loaded from the system.
- Seller can stop selling only one product at a time.
- When seller choose 3d model to upload, the 3D model has created by seller before.
- After seller sends stop selling product command, system displays message to notify user that system is processing and a pending list will display with status "Out of stock".

2.3.5.5 < Seller > Get order Usecase

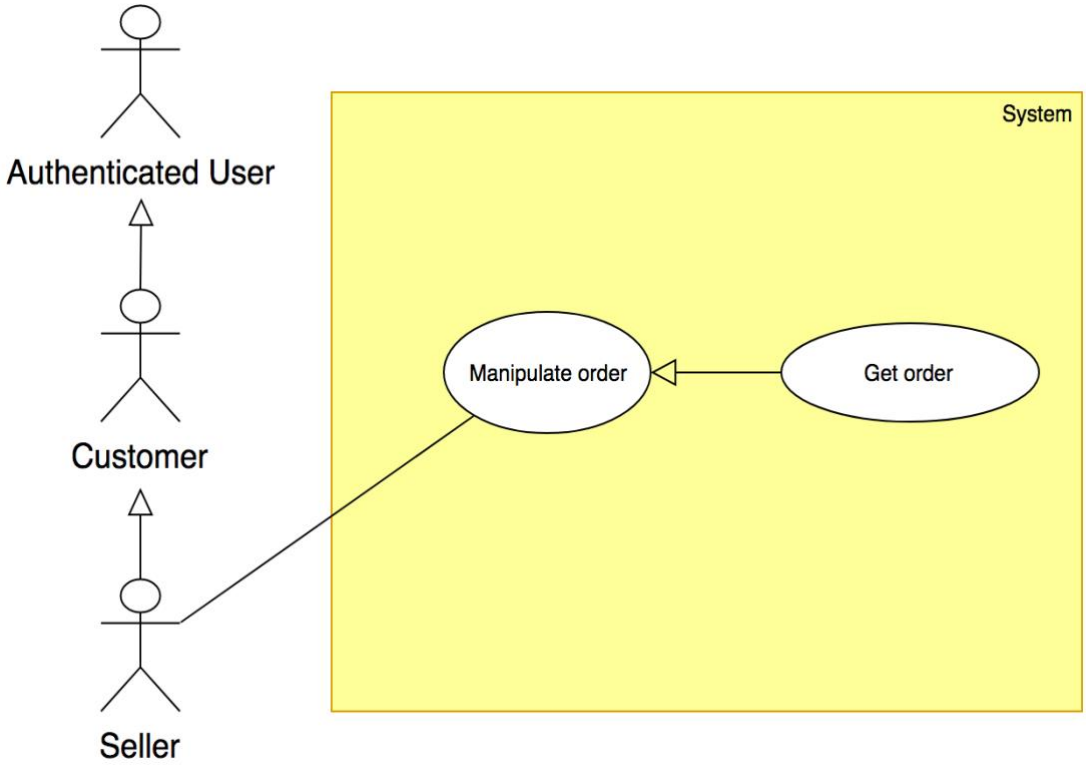


Figure 38: < Seller > Get order Usecase

USE CASE – UC_SL.04			
Usecase No.	**SL.04**	**Usecase Version**	1.0
Usecase Name	Get order		
Author	DucPH		
Date	18/01/2018	**Priority**	Normal

Actor:

- Seller.

Summary:

- This use case allows seller to get their order details.

Goal:

- Seller can view their order.
- Seller can view order's detail information

Triggers:

- Seller sends view order's detail information command.

Preconditions:

- User must login into system with role Seller.

Post Conditions:

- **Success:** Order's detail information is shown.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to Manipulate order view.	Display a table shows list of order: • Order ID: text • Customer name: text • Product ordered: text • Purchased date: date time • Arrival date: date time • Total: number
2	Seller selects order to view their detail information.	Display a table shows list of products in order: • Product ID: text • Quantity: number • Price: number

Alternative Scenario: N/A

Exceptions: N/A

Relationships: N/A

Business Rules:

- List of order and their detail information is always loaded from the system.

2.3.5.6 < Seller > Cancel Order Usecase

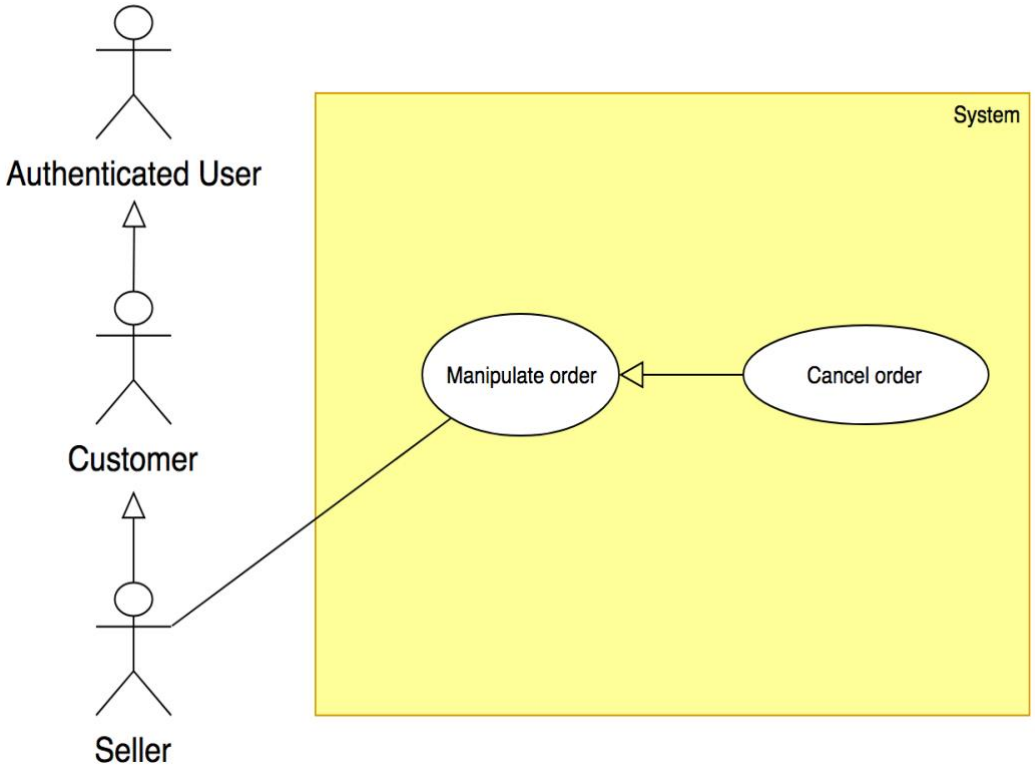


Figure 39: < Seller > Cancel Order Usecase

USE CASE – UC_SL.04			
Usecase No.	**SL.04**	**Usecase Version**	1.0
Usecase Name	Cancel order		
Author	DucPH		
Date	18/01/2018	**Priority**	Normal

Actor:

- Seller.

Summary:

- This use case allows seller to cancel their order.

Goal:

- Seller can cancel their order.

Triggers:

- Seller sends cancel order command.

Preconditions:

- User must login into system with role Seller.

Post Conditions:

- **Success:** The will be canceled successful.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Seller goes to Manipulate order view.	Display a table shows list of order: • Order ID: text • Customer name: text • Product ordered: text • Purchased date: date time • Arrival date: date time • Total: number
2	Seller selects order to view their detail information.	Display a table shows list of products in order: • Product ID: text • Quantity: number • Price: number
3	Seller send cancel order command	System starts update order's status. System show message cancel order successful. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response

1	If cancel order failed	Show message to notify user that cancel order failed.

Relationships: N/A

Business Rules:

- List of bills and their detail information is always loaded from the system.

2.3.6 < Staff > Overview Usecase

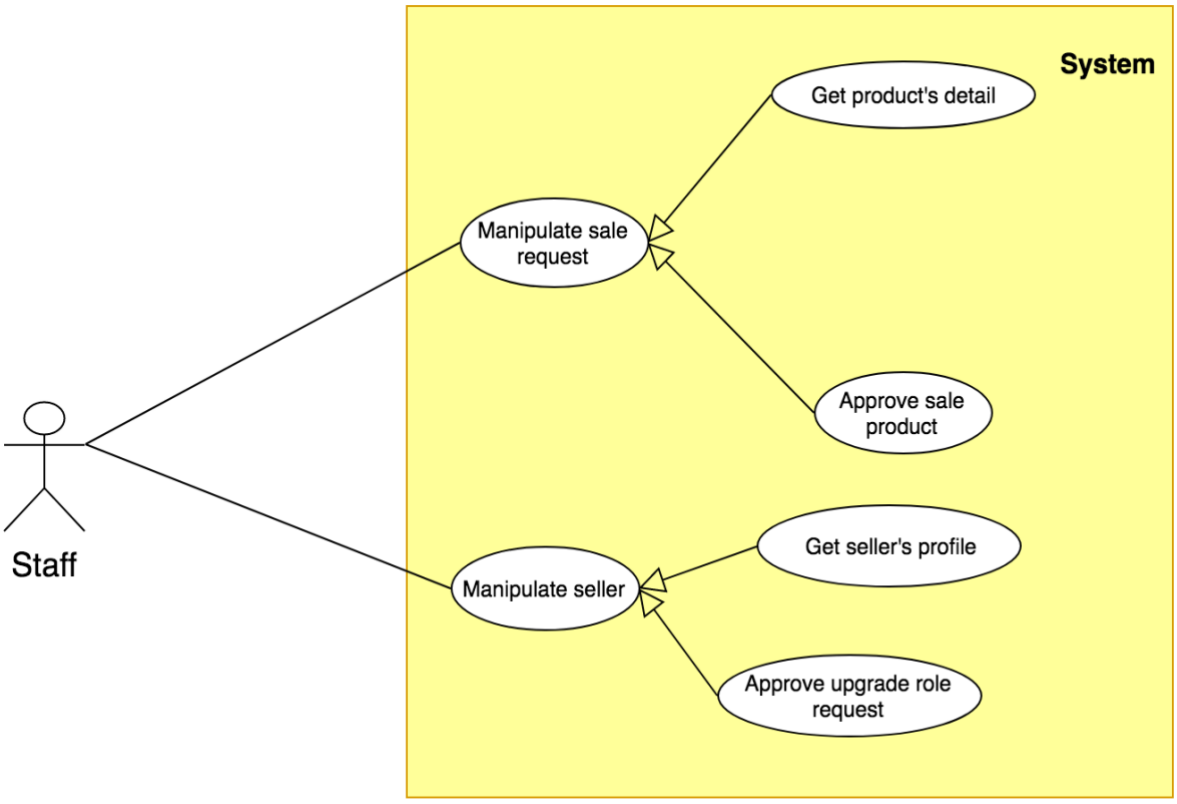


Figure 40: < Staff > Overview Usecase

2.3.6.1 < Staff > Get product's detail

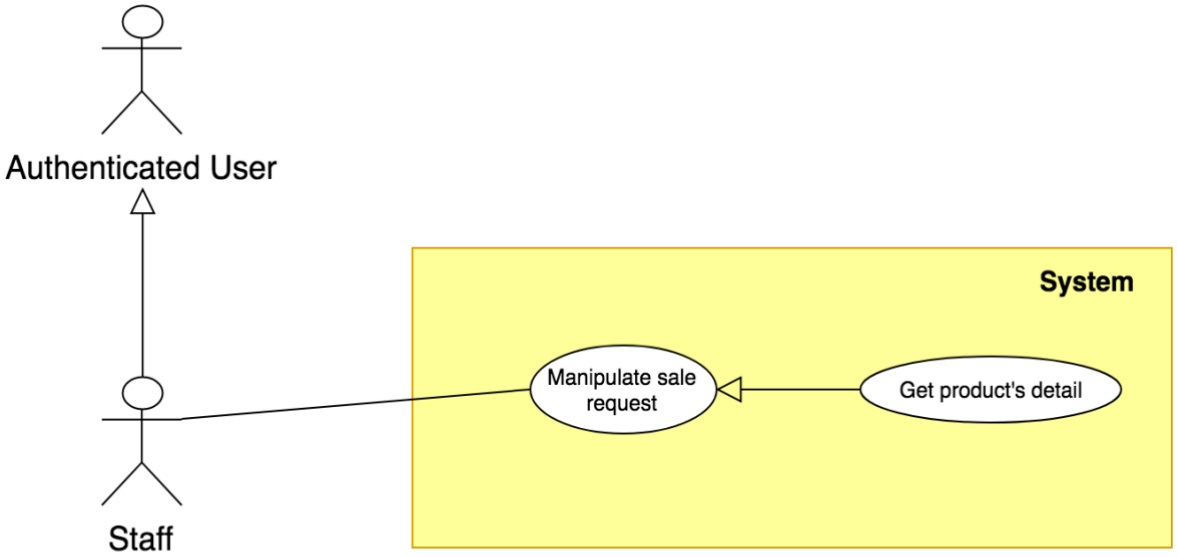


Figure 41: < Staff > Get product's detail Usecase

USE CASE – UC_ST.01			
Usecase No.	**UC_ST.01**	**Usecase Version**	2.0
Usecase Name	Get product's detail		
Author	ThienBT		
Date	23/01/2018	**Priority**	Normal

Actor:
- Staff

Summary:
- This use case allows staff to get detail of product.

Goal:
- Staff can view information about product.

Triggers:
- Staff sends command to get product's detail.

Preconditions: N/A

Post Conditions:
- **Success:** Product's detail informations is displayed for staff.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to seller request manipulate	System loaded list of seller request.
2	Guest chooses a product and sends command to get detail.	

		System displays product's detail information: • Product name: text • Seller's Username: text • Seller's Store: text • Status: text • Price: Double • Quantity: Integer • Category: text • Origin: text • Size: Width x Length x Height: number • Weight: text • Material: text • Color: text • Created Date: yyyy/MM/dd HH:mm:ss • Updated Date: yyyy/MM/dd HH:mm:ss Description: text [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	System can not load product informations.	System shows error message.

Relationships: N/A

Business Rules:

- Staff go to seller request and choose product to view detail.
- System will show product's information.

2.3.6.2 < Staff > Approve sale Product

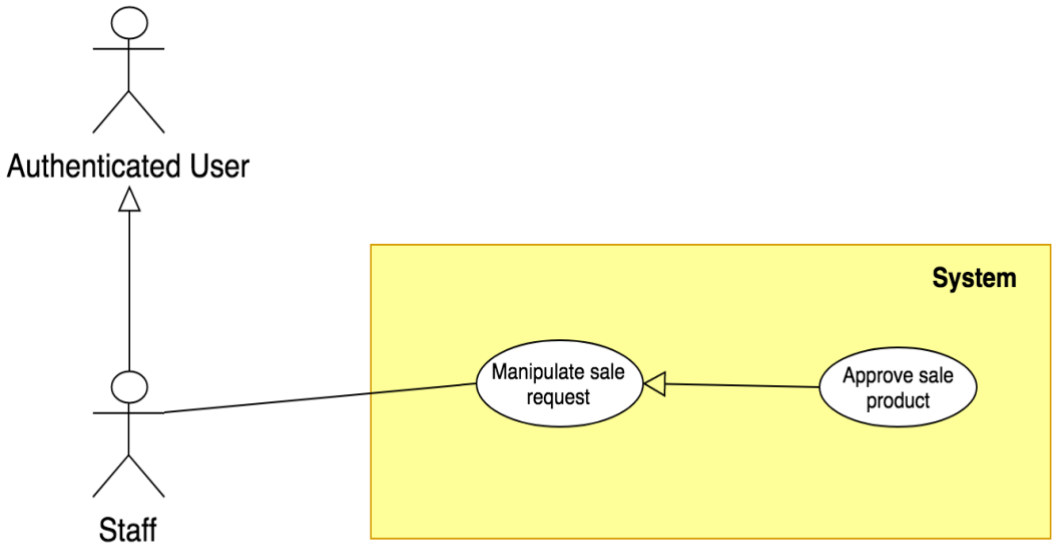


Figure 42: < Staff > Approve sale Product Usecase

USE CASE – UC_ST.02			
Usecase No.	**UC_ ST.02**	**Usecase Version**	2.0
Usecase Name	Approve sale Product		
Author	ThienBT		
Date	23/01/2018	**Priority**	Normal

Actor:

- Staff.

Summary:

- This use case allows staff approve request from seller.

Goal:

- Staff will approve product to generate 3D model of product and product is display in marketplace.
- For any approval to request, user will be notified.

Triggers:

- Staff sends command to approve.

Preconditions:

- User must log into the system as Staff role.
- There is has at least 01 request from other user in the system

Post Conditions:

- **Success:** System shows staff message approve successfully.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to manipulate seller's request view	

		Shows list out of information of requests: • Product name: text • Seller's Username: text • Seller's Store: text • Status: text • Price: Double • Quantity: Integer • Category: text • Origin: text • Size: Width x Length x Height: number • Weight: text • Material: text • Color: text • Created Date: yyyy/MM/dd HH:mm:ss • Updated Date: yyyy/MM/dd HH:mm:ss • Description: text
2	If staff insert service's pricing send command to approve request [Alternative 1]	Change status of request and send notification to seller

Alternative Scenario:

No	Cause	System Response
1	If staff send command to reject	Change status of request and send notification to seller

Exceptions: N/A

Relationships: N/A

Business Rules:

- The information of request is always loaded in the system.
- When staff is checking request details, status of request is changed to "Waiting seller confirm".
- If staff send price and approve request, status of request is change and notify to seller
- If staff reject any request, status of request is change and notify to seller

2.3.6.3 < Staff > Get seller's profile

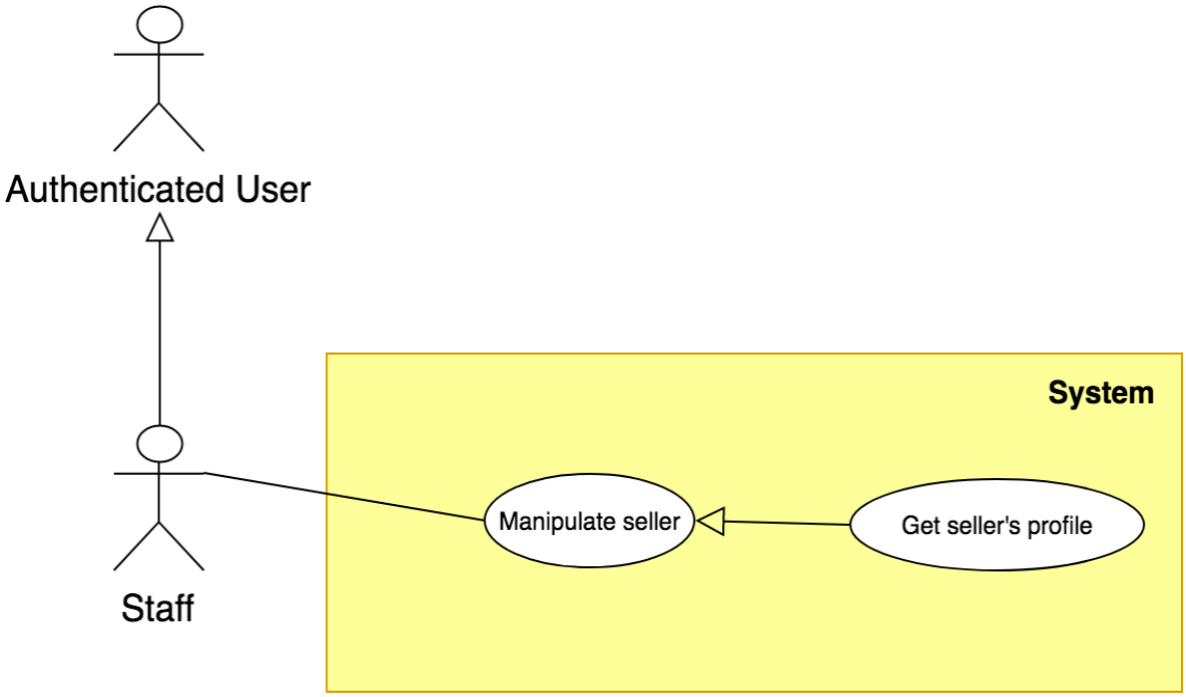


Figure 43: < Staff > Get product's detail Usecase

USE CASE – UC_ST.01			
Usecase No.	**UC_ST.03**	**Usecase Version**	2.0
Usecase Name	Get seller's profile		
Author	ThienBT		
Date	23/01/2018	**Priority**	Normal

Actor:
- Staff

Summary:
- This use case allows staff to get seller's profile.

Goal:
- Staff can view information about seller.

Triggers:
- Staff sends command to get seller's profile.

Preconditions: N/A

Post Conditions:
- **Success:** Seller's detail informations is displayed for staff.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to manipulate user	System loaded list of seller.
2	Guest chooses a seller and	

	sends command to get detail.	System displays seller's detail information: • Product name: text • Seller's Username: text • Seller's Store: text • Status: text • Price: Double • Quantity: Integer • Category: text • Origin: text • Size: Width x Length x Height: number • Weight: text • Material: text • Color: text • Created Date: yyyy/MM/dd HH:mm:ss • Updated Date: yyyy/MM/dd HH:mm:ss Description: text [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Cause	System Response
1	System can not load seller informations.	System shows error message.

Relationships: N/A

Business Rules:

- Staff go to manipulate user and choose seller to view detail.
- System will show seller's information.

2.3.6.4 < Staff > Approve upgrade role request

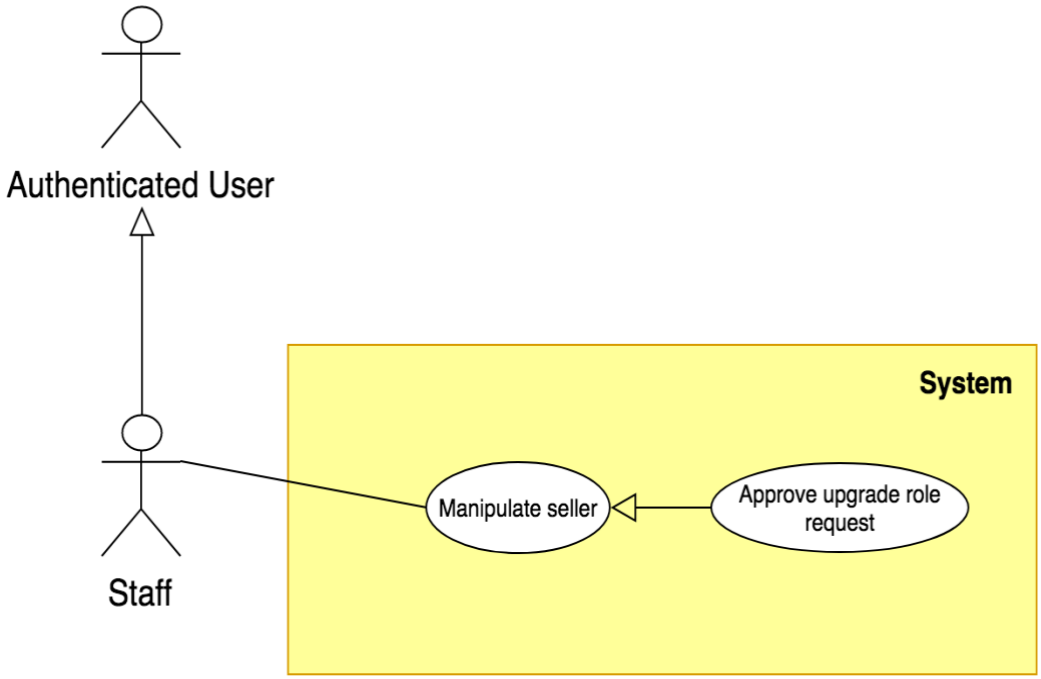


Figure 44: < Staff > Approve upgrade role request Usecase

USE CASE – UC_ST.02			
Usecase No.	**UC_ ST.02**	**Usecase Version**	2.0
Usecase Name	Approve upgrade role request		
Author	ThienBT		
Date	23/01/2018	**Priority**	Normal

Actor:
- Staff.

Summary:
- This use case allows staff approve upgrade role request.

Goal:
- Staff accept request upgrade role from customer to seller, customer information will be updated with role "Seller".
- For any approval to request, user will be notified.

Triggers:
- Staff sends command to approve.

Preconditions:
- User must log into the system as Staff role.
- There is has at least 01 request from customer in the system

Post Conditions:
- **Success:** System shows staff message approve successfully.
- **Fail:** System shows error page

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to manipulate User request view	Shows list out of information of requests: • Username: text • Full name: text • Birth: dd/MM/yyyy • Address: text • Email: text • Phone: number • Create Date: yyyy/MM/dd HH:mm:ss • Status: text • Store name: text • ID Card: number • Store's phone: text • Store Address: text • Store Create Date: yyyy/MM/dd HH:mm:ss • Store Update Date: yyyy/MM/dd HH:mm:ss
2	If staff send command to approve request [Alternative 1]	Change role to become seller.

Alternative Scenario:

No	Cause	System Response
1	If staff send command to reject	Send notification to customer

Exceptions: N/A

Relationships:

Business Rules:

- The information of request is always loaded in the system.
- When staff is checking request details, status of request is changed to "In-process".
- If staff approve request customer role is updated to seller in the system, notify to customer.
- If staff reject, its notify to customer.

2.3.6 < Scheduler > Overview Usecase

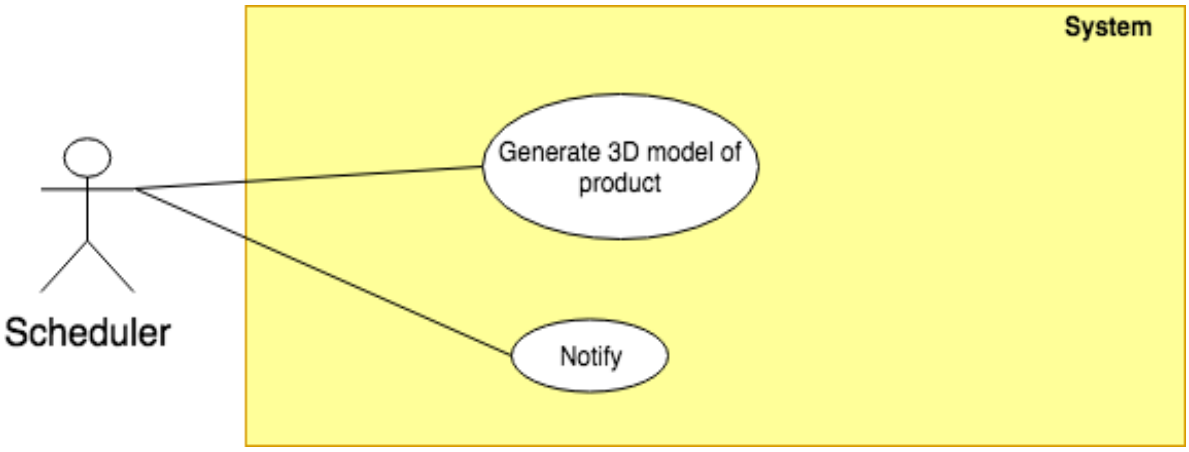


Figure 45: < Scheduler > Overview Usecase

2.3.6.1 < Scheduler > Generate 3D model of product Usecase

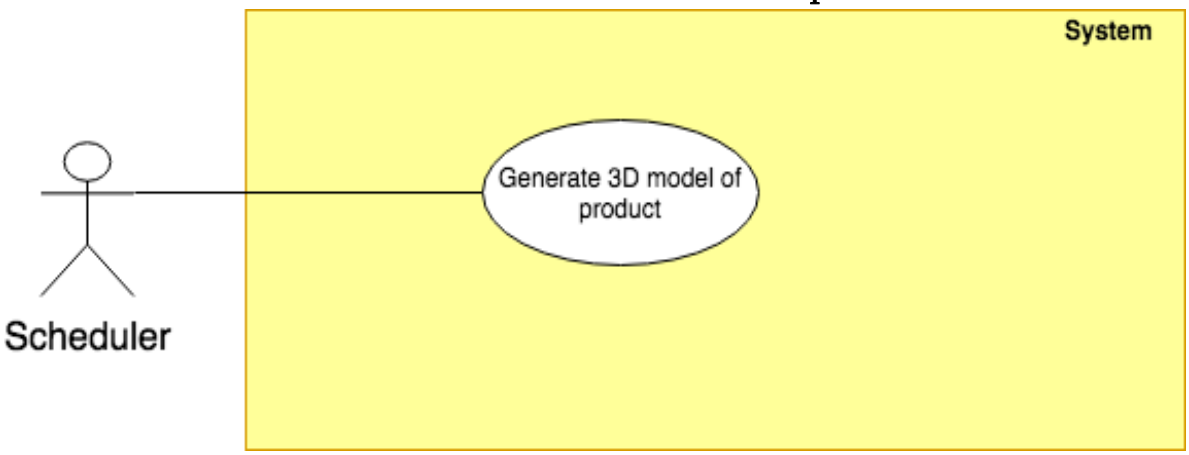


Figure 46: < Scheduler > Generate 3D model of product Usecase

USE CASE – UC _SC.01			
Usecase No.	SC.01	**Usecase Version**	1.0
Usecase Name	Generate 3D model of product		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	High

Actor:

- Scheduler.

Summary:

- Scheduler will generate 3D model of product from series of product's overlap pictures.

Goals:

- 3D model of product is created.

Triggers:

- Seller sends generate 3D model command.

Preconditions:

- The seller provided product's pictures and information.
- The staff approved sale product request.
- The seller approved service pricing.

Post Conditions:

- **Success**: 3D model is created and notify to staff.
- **Fail**: write log file and notify error to staff.

Main Success Scenario:

Step	Actor Action	System Response
1	Scheduler sends command generated 3D model.	System starts generating 3D model.
2		System returns 3D model in zip format. [Exception 1]

Alternative Scenario: N/A

Exceptions:

Step	Cause	System Response
1	Pictures are captured wrong aspects.	System return zip file with empty data.

Relationships:

- < Seller > Request to sell product. (SL.01)
- < Staff > Approve sale product. (ST.02)

Business Rules:

- Seller captures pictures with all aspects of product.
- Pictures with ratio 3:4 and with format JPG.
- Seller must provide at least 40 pictures of product.

2.3.6.2 < Scheduler > Notify Usecase

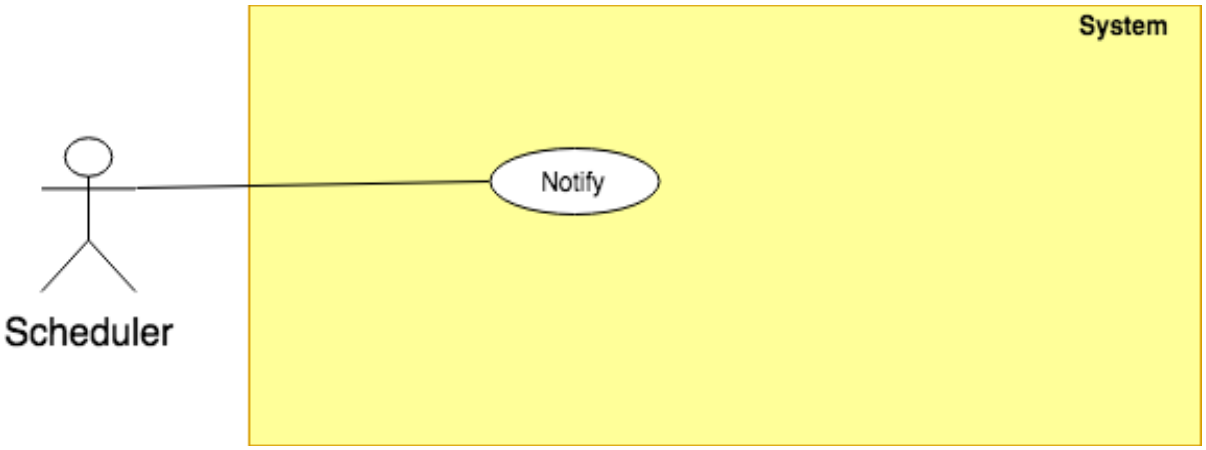


Figure 47: < Scheduler > Notify Usecase

USE CASE – UC _SC.02			
Usecase No.	SC.02	**Usecase Version**	1.0
Usecase Name	Notify		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	Normal

Actor:
- Scheduler.

Summary:
- Scheduler will notify notifications to user.

Goals:
- User's received notifications.

Triggers:
- After staff approves/rejects user's requests (request to sell product, request to upgrade role).
- Sale product is started selling on market.
- Product is bought.

Preconditions:
- User sent requests to system.
- Product is sold on marketplace.

Post Conditions:
- **Success**: Notification is sent to user.
- **Fail**: write log file.

Main Success Scenario:

Step	Actor Action	System Response
1	Scheduler sends send notification command.	System sends notification to user.

Alternative Scenario: N/A

Exceptions: N/A

Relationships:
- < Staff > Approve sale product (ST.02)
- < Staff> Approve upgrade role request (ST.04)
- < Customer > Checkout

Business Rules: N/A

2.3.7 < Designer > Overview Usecase

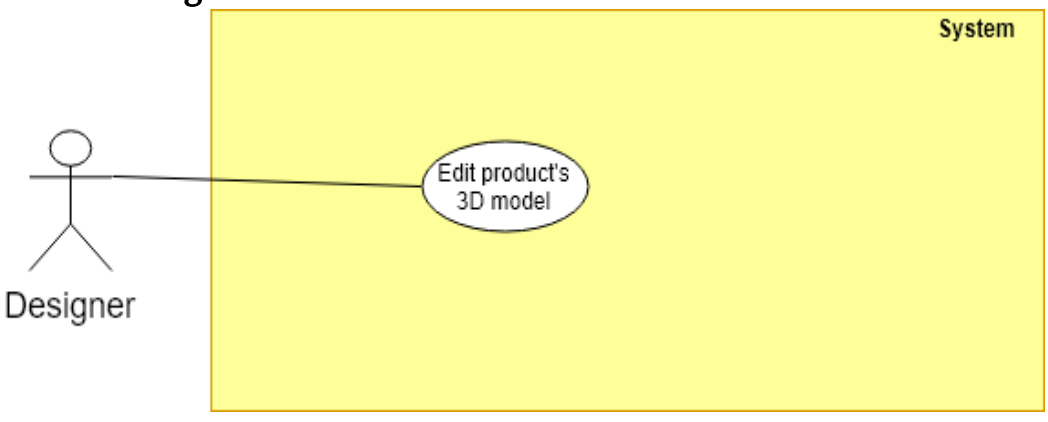


Figure 48: < Designer > Overview Usecase

2.3.7.1 < Designer > Edit product's 3D model Usecase

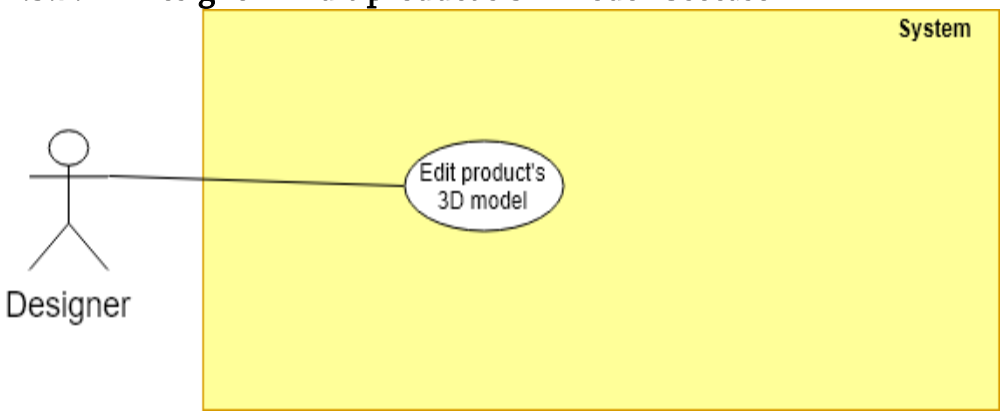


Figure 49: < Designer > Edit product's 3D model Usecase

USE CASE – UC _D.01			
Usecase No.	D.01	**Usecase Version**	1.0
Usecase Name	Edit product's 3D model		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	High

Actor:

- Designer.

Summary:

- Designer will upgrade 3D model's quality, remove redundant textures.

Goals:

- 3D model is upgraded quality.

Triggers:

- Scheduler notifies that 3D model generating is finished.

Preconditions:

- Scheduler finished generating 3D model.

Post Conditions:

- **Success**: 3D model is ready to represent product.
- **Fail**: Designer notifies error.

Main Success Scenario:

Step	Actor Action	System Response
1	Designer sends command to download raw 3D model.	System returns raw 3D model.
2	Designer edits 3D model.	
3	Designer send command tp upload edited 3D model.	System shows message notifying model is uploaded. [Exception 1]

Alternative Scenario: N/A

Exceptions:

Step	Cause	System Response
1	Designer send command tp upload edited 3D model.	System shows connection error message.

Relationships:

- < Staff > Approve sale product (ST.02)
- < Scheduler > Generate 3D model of product Usecase (SC.01)

Business Rules:

- Designer uses 3D editors such as Autodesk Maya, Autodesk 3DS MAX, Blender… to edit 3D model.

2.3.8 < Payment System > Overview Usecase

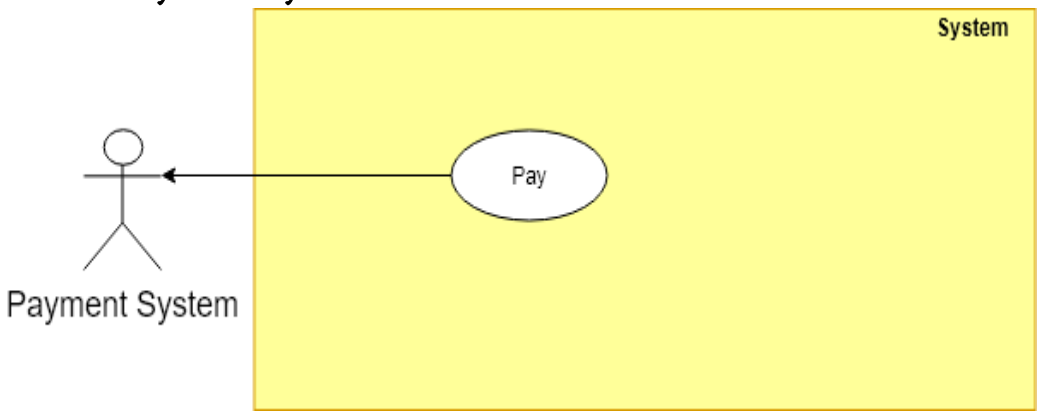


Figure 50: < Payment System > Overview Usecase

2.3.8.1 < Payment System > Pay Usecase

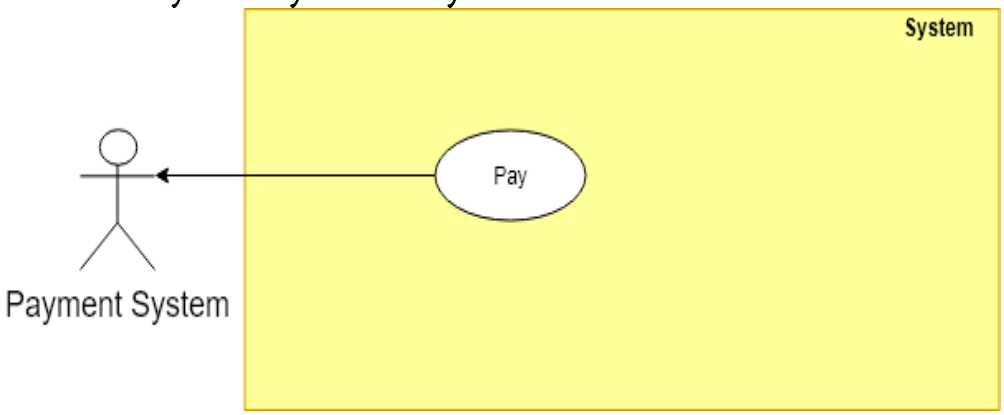


Figure 51: <Payment System > Pay Usecase

USE CASE – UC _PS.01			
Usecase No.	PS.01	**Usecase Version**	1.0
Usecase Name	Pay		
Author	KhoaNPA		
Date	17/01/2018	**Priority**	High

Actor:

- Payment System.

Summary:

- This use case allows payment system processes the payment transaction.

Goals:

- Payment process is completed.

Triggers:

- Customer sends pay for charge by payment system command.

Preconditions:

- Scheduler finished generating 3D model.

Post Conditions:

- **Success**: Payment is completed successfully.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer select payment via PayPal.	Summary of user's payment: • Description: text. • Amount: text. • Total: text. Payment view is shown with following labels and fields: • User's PayPal account: free text input, required, length 3 – 250 • User's PayPal password: free text input, required, length 6 – 32
2	User fills out the form.	
3	User sends pay command.	Validate data [Alternative 1] [Exception 1, 2, 3] Process payment in PayPal

Alternative Scenario:

No	Actor Action	System Response
1	Identity information is not match.	

		Show message notify user input wrong email or password.

Exceptions:

No	Cause	System Response
1	Missing of required fields	Show message notify user input missed fields
2	Length of field's value is out of range	Show message notify user which field's value is out of range
3	Entered email address is not a valid email	Show message notify entered email is not valid

Relationships:

- < Customer > Checkout Usecase (C.04)

Business Rules:

- Use PayPal API to validate customer email and password.
- PayPal will send notification to alert transaction is failed or succeeded.
- An email address must be validated by this regular expression: /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/

3. Software System Attribute

3.1 Usability

- UI website is fit for each browser in each device
  - Font style: Helvetica, Arial, Helvetica Neue, Roboto, Arial, Droid Sans, sans-serif
  - Font size: 12px -26px
  - Color: green, black, red, white, blue, Light Slate Grey, Fuego, Honeysuckle…
  - Background: White, Catalina Blue, Whisper…
  - Theme: Gentelela
- UI mobile application mobile is scalable with each monitor of smart phone:
  - Font size: 13-30pt
  - Font style: San Francisco.
  - Color: Black, Light Gray, Orange, Blue…

- Background: White, Light Grey…

3.2 Reliability

- View AR on mobile successfully with at light level from 35 lux to brighter.
- 3D model generated by system has acceptable quality with accuracy is 80%.

3.3 Availability

- System replies in maximum 2 seconds.

3.4 Security

- Each role of user has a specific permission to interact with the system.
- System always checks for authorization and authentication before doing anything.
- Input data is validated before saving to database.

3.5 Maintainability

- The system is divided into separated modules for easy maintain.

3.6 Portability

- User can use the mobile application on devices running on iPhone 6S or later with iOS 11.3 or later.
- Web application can be run on Chrome browser version 42 or later.

3.7 Performance

- Camera can detect surface under 5 seconds on almost surfaces with light level upper 35 lux.
- User can view maxium with 10 products in AR view.

4. Conceptual Diagram

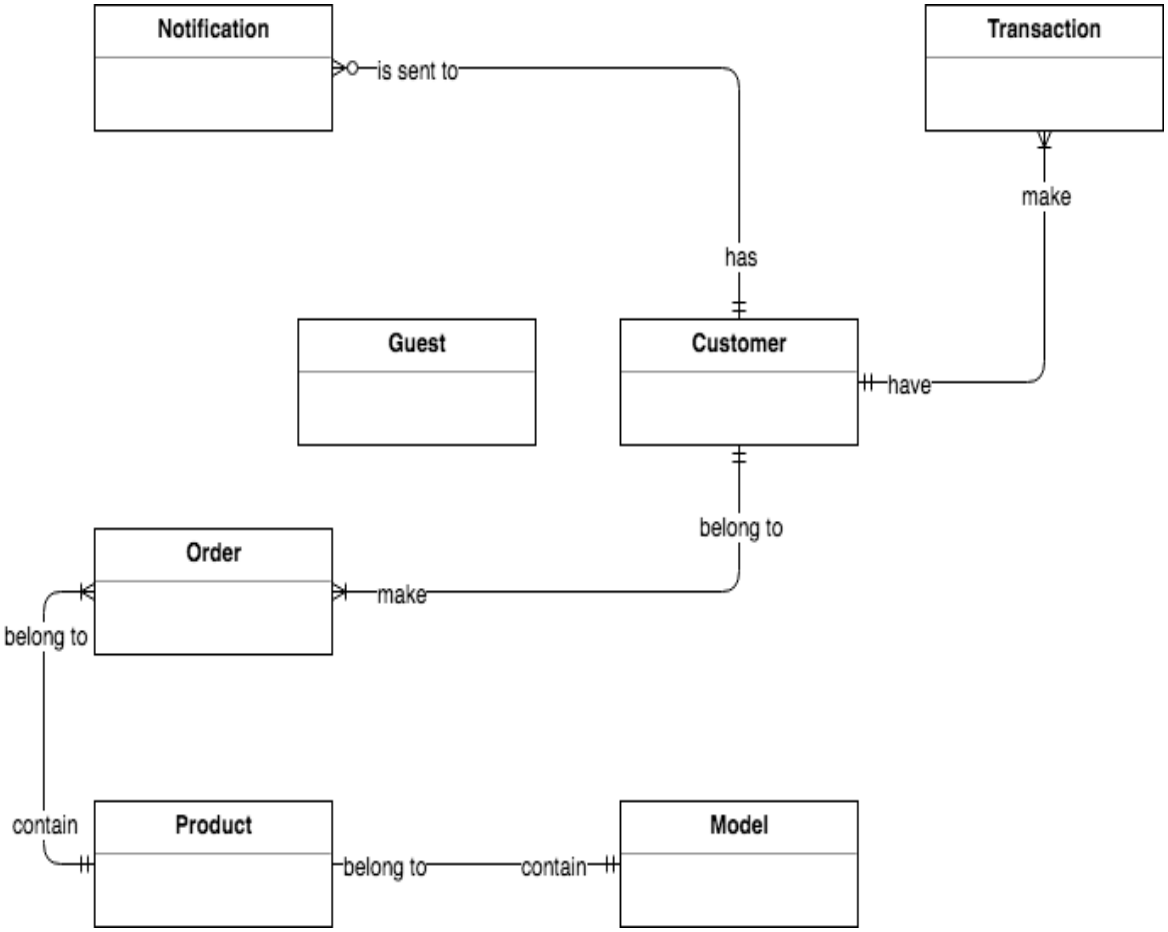


Figure 52: Conceptual diagram

Data Dictionary

Entity Data dictionary: describe all content of all entities	
Entity Name	**Description**
Notification	Contain the notification information.
Transaction	Contain the transaction information.
User	Abstract entity describes a user in system
Order	Contain customer's order information
Product	Contain the product information provided by seller
Model	Contain the model information associated with its product

D. Report No. 4 Software Design Description

1. Design Overview

This document describes the technical and user interface design of **ifAR System**. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.

The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

The database design describes the relationships between entities and details of each entity.

Document overview:

- Section 2: gives an overall description of the system architecture design.
- Section 3: gives component diagrams that describe the connection and integration of the system.
- Section 4: gives the detail design description which includes class diagram, class explanation, and sequence diagram to details the application functions.
- Section 5: describe screens design.
- Section 6: describe a fully attributed ERD.
- Section 7: describe algorithms.

2. System Architectural Design

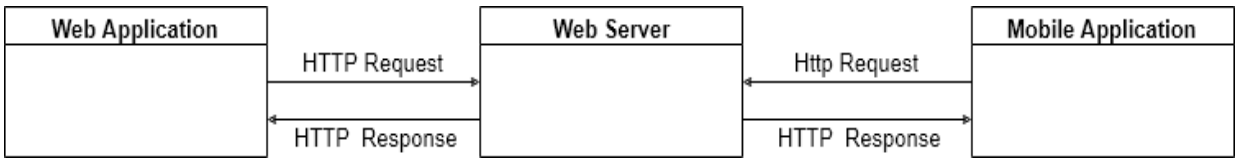


Figure 53 System overview architecture

2.1 Web Server Architecture Description

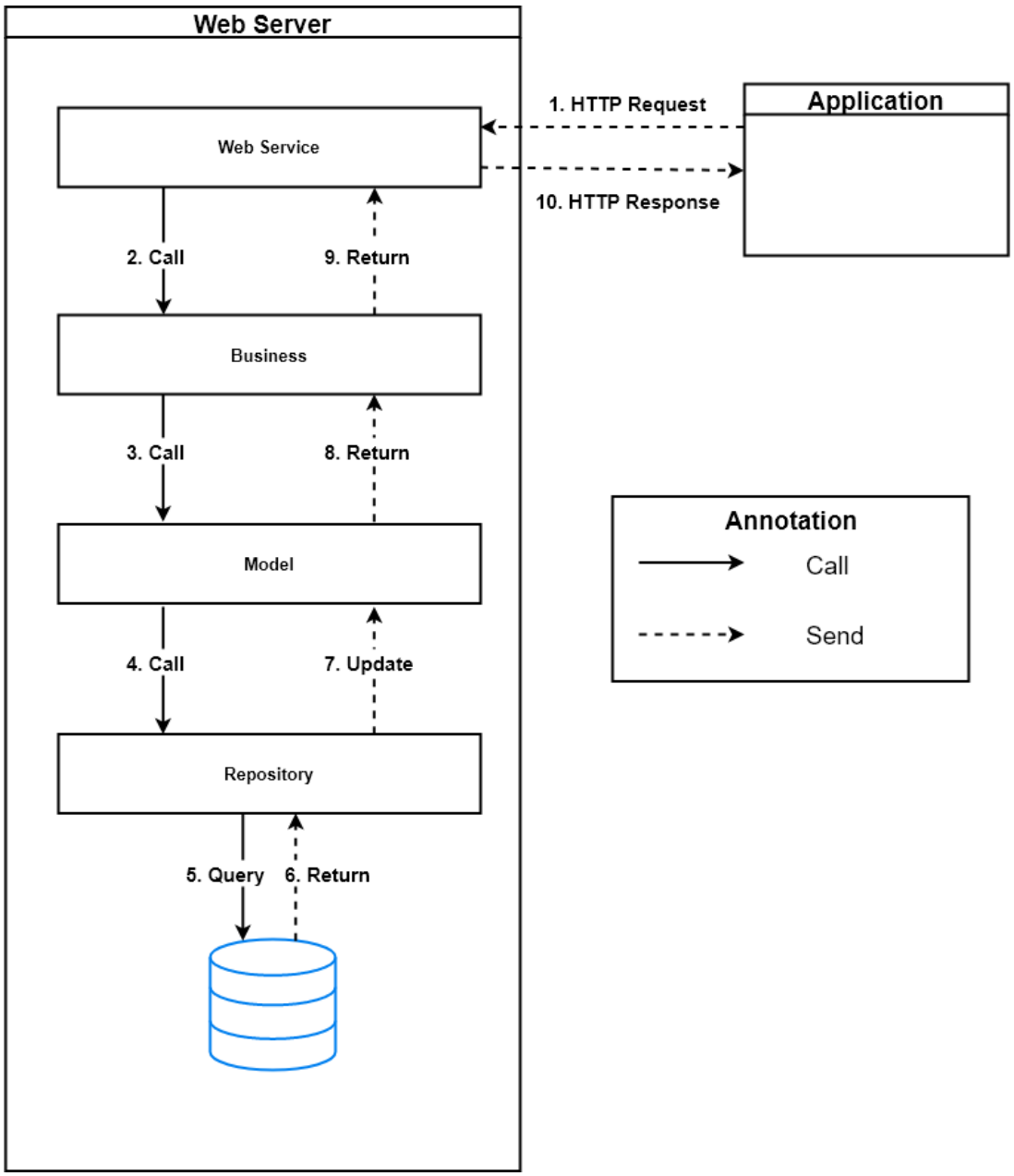


Figure 54 Web server architecture

In Web Application, the system is developed under Spring MVC architecture style. We choose this architecture for Web application because of following advantages:

- Web application contains a Web service (public API for mobile app), with MVC architecture, we can separate business code with Controller and View, so we can use the business code in web service without repeat the code.
- Spring comes with some of the existing technologies like Hibernate framework, security framework and J2EE etc. Hence, we don't need to integrate explicitly those technologies.

- Spring can eliminate the creation of the singleton and factory classes and well defined interface to business layer
- By separating concerns into 3 distinct pieces, we can perform unit testing easily. Our Presentation layer can be tested free of the Model or Controller, and vice-a-versa
- Spring supports all aspects of application development, Business aspects, persistence aspects, etc., so we can develop a complete application.

This project follows MVC architecture with following components:

- **Spring Controller**: is the parts of the application that acts like event handler to handles user interaction. Typically, controller reads data from a request and calls appropriate Business's method then selects view to return to user.
- **View**: The view renders the contents of a model. It gets data from the model and specifies how that data should be presented. It updates data presentation when the model changes. A view also forwards user input to a controller. Depending on the task being performed by the user the model can be looked at from different perspectives.
- **Model**: Represents the business data and any business logic that govern access to and modification of the data. The model notifies views when it changes and lets the view query the model about its state. It also lets the controller access application functionality encapsulated by the model. Typically, when a change in the model is to be reflected from user, it should be reflected in all the model's views.

2.2 iOS Application Architecture Description

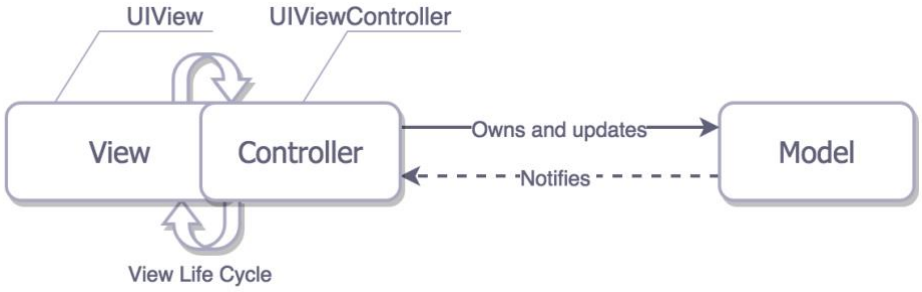


Figure 55 iOS MVC architecture

(Reference: https://medium.com/ios-os-x-development/ios-architecture-patterns-ecba4c38de52)

In iOS application, the system is developed under MVC architecture style. We choose this architecture for iOS because of following advantages:

- **Distribution**—the View and the Model in fact separated, but the

View and the Controller are tightly coupled.

- **Testability**—due to the bad distribution you'll probably only test your Model.
- **Ease of use**—the least amount of code among other patterns. In addition, everyone is familiar with it, thus, it's easily maintained even by the unexperienced developers.

2.3 Web Application Architecture Description

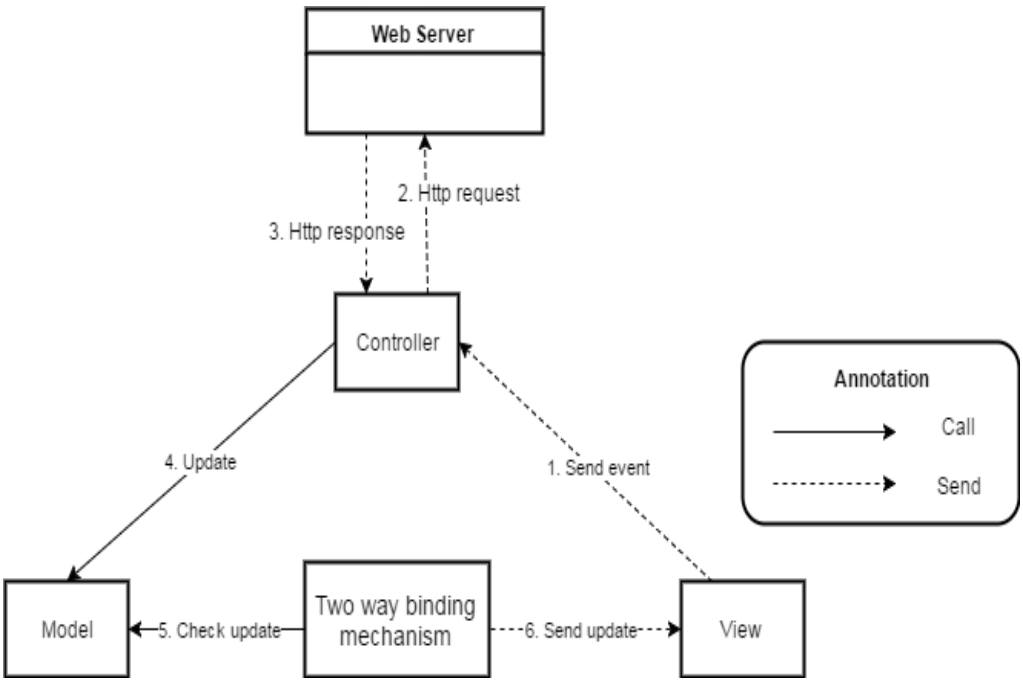


Figure 56 Web applicaiton architecture

In web application, the system is developed under MVC architecture for single-page application style. We choose this architecture for web application because of following advantages:

- With the two-way data binding, the user interface changes are immediately reflected in the underlying data model. Restrict web page refresh each time request to server, web application becomes friendly.

Web application follows MVC architecture with following components:

- **Controller**: is the parts of the application that acts like event handler to handles user interaction.
- **View**: The view renders the contents of a model. It gets data from the model and specifies how that data should be presented. In web application, view often descripts html pages.
- **Model**: Represents the business data and any business logic that govern access to and modification of the data. The model notifies views when it changes and lets the view query the model about its state.

- **Two ways binding mechanism**: Two ways data-binding is an automatic way of updating the view whenever the model changes, as well as updating the model whenever the view changes.

3. Component Diagram

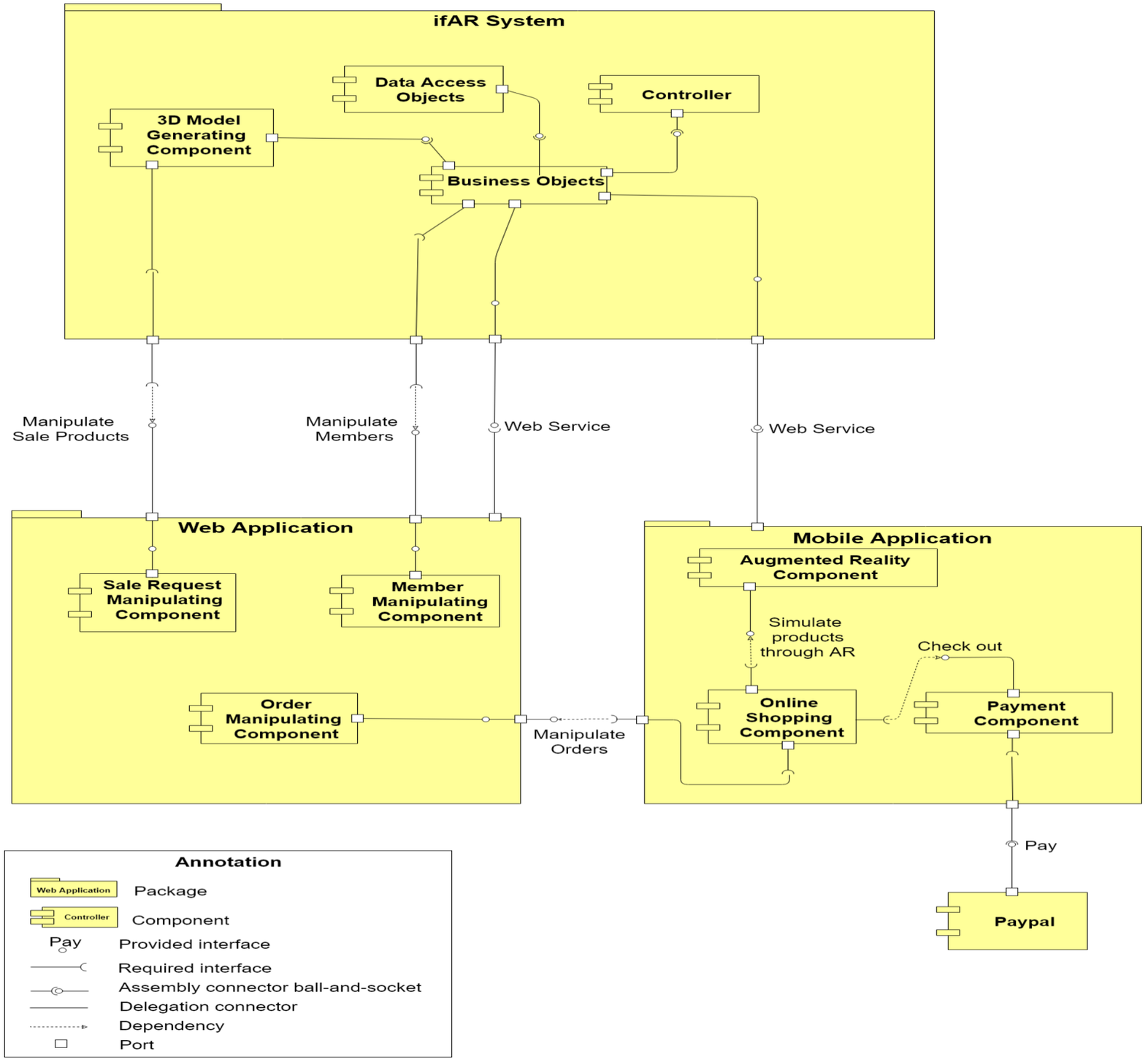

ifAR System
Data Access Objects
Controller
3D Model Generating Component
Business Objects
Manipulate Sale Products
Manipulate Members
Web Service
Web Service
Web Application
Sale Request Manipulating Component
Member Manipulating Component
Order Manipulating Component
Manipulate Orders
Mobile Application
Augmented Reality Component
Simulate products through AR
Check out
Online Shopping Component
Payment Component
Pay
Paypal
Annotation
Web Application
Package
Controller
Component
Pay
Provided interface
Required interface
Assembly connector ball-and-socket
Delegation connector
Dependency
Port

Figure 57 Component diagram

Data dictionary

Name	Description
ifAR System	Interior Furniture Augmented Reality System package
Web Application	Web application package
Mobile Application	Mobile application package
Data Access Objects	Component to handle interaction between the system and database
Business Objects	Common objects to handle domain business operations for each component
Sale Request Manipulating Component	Component to handle seller's sale requests manipulating processes
Controller	Component to handle business management
3D Model Generating Component	Component to handle 3D model generating process
Member Manipulating Component	Component to handle members manipulating process
Order Manipulating Component	Component to handle orders manipulating process
Augmented Reality Component	Component to handle products simulating in augmented reality view
Online Shopping Component	Component to handle online shopping
Payment Component	Component to handle payment process
Paypal	Handle payment process with PayPal API
Web Service	Provide API for mobile applications and web application to interact with the system.
Manipulate Sale Products	Handle sale products
Manipulate Members	Handle members in system
Manipulate Orders	Handle orders manipulating
Pay	Handle payment processes
Check out	Handle check out processes of online shopping
Simulate products through AR	Handle products simulating in augmented reality view

4. Detailed Description

4.1 Class Diagram

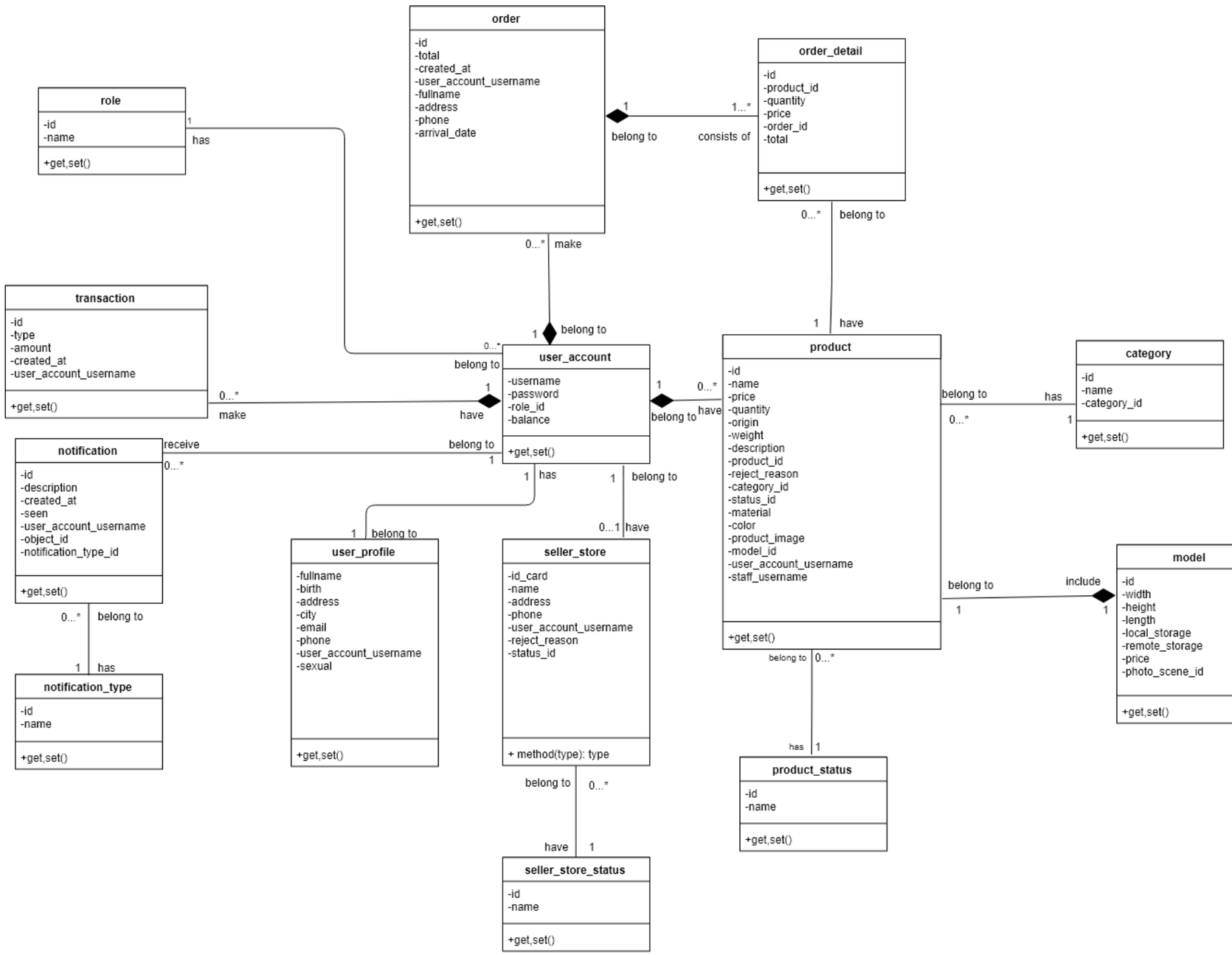


Figure 58 Class diagram

Class Name	Mapping column with Conceptual diagram	Description
role	N/A	Not exist in conceptual diagram. It's used to contain role information.
order	order	Contain customer's order information.
order detail	N/A	Not exist in conceptual diagram. It's used to contain order detail information.
transaction	transaction	Contain customer's transaction information.
notification	notification	Contain notification information.
notification_type	N/A	Not exist in conceptual diagram. It's used to contain notification type of notification.
user_account	user	Contain customer's information.
user_profile	N/A	Not exist in conceptual diagram. It's used to contain user profile information of user account.
seller_store	store	Contain seller's store information.
seller_store_status	N/A	Not exist in conceptual diagram. It's used to contain store status of seller store.
product	product	Contain product's information.
product_status	product_status	Not exist in conceptual diagram. It's used to contain product status of product.
category	N/A	Not exist in conceptual diagram. It's used to contain category information of product.
model	model	Contain model's information of product.

Table 7 Class dictionary

4.2 Class Diagram Explanation

4.2.1 role

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Unique identifier of role.
name	String	Private	Role name.

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.2 order

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Unique identifier of order
total	Double	Private	Order total price
created_at	Date time	Private	Order created date
user_account_username	String	Private	Username of customer
fullname	String	Private	Full name of customer
address	String	Private	Delivery address
phone	String	Private	Delivery phone number
arrival_date	Date time	Private	Arrival date of order

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.3 order_detail

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Unique identifier of order detail
product_id	UUID	Private	Identifier of product in order detail
quantity	Int	Private	Quantity of order detail
price	Double	Private	Selling price of product

order_id	UUID	Private	Identifier of order
total	Double	Private	Order detail total price

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.4 transaction

Attribute

Attribute	Type	Visibility	Description
id	Int	Private	Unique identifier of transaction
type	Boolean	Private	Type of transaction
amout	Double	Private	Amount of transaction
created_at	Date time	Private	Created date of transaction
user_account_username	String	Private	Username

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.5 notification

Attribute

Attribute	Type	Visibility	Description
id	Int	Private	Unique identifier of assistant type
description	String	Private	Notification description information
created_at	Date time	Private	Created date of notification
seen	Boolean	Private	Seen status of notification
user_account_username	String	Private	Username of user account

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.6 notification_type

Attribute

Attribute	Type	Visibility	Description
id	Int	Private	Unique identifier of notification type
name	String	Private	Name of notification type

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.7 user_account

Attribute

Attribute	Type	Visibility	Description
username	String	Private	Unique identifier of user account
password	String	Private	Password
role_id	Int	Private	Identify of role
balance	Double	Private	Balance of account

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.8 user_profile

Attribute

Attribute	Type	Visibility	Description
fullname	String	Private	Full name of user profile
birth	Date time	Private	Birthdate of user profile
address	String	Private	Address of user profile
city	String	Private	City of user profile
email	String	Private	Email of user profile
phone	String	Private	Phone of user profile
user_account_username	String	Private	Username of user account
sexual	Int	Private	Sex of user

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.9 seller_store

Attribute

Attribute	Type	Visibility	Description
id_card	String	Private	ID card of seller store
name	String	Private	Unique identifier of seller store
address	String	Private	Address of seller store
phone	String	Private	Phone number of seler store
user_account_username	String	Private	Username of seller
reject_reason	String	Private	Reject reason of seller store
status_id	Int	Pricate	Status identifier of seller store

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.10 seller_store_status

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Identifier of seller store status
name	String	Private	Name of seller store status

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.11 product

Attribute

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of product
name	String	Private	Name of product
price	Double	Private	Price of product
quantity	Int	Private	Quantity of product
origin	String	Private	Origin of product
weight	Double	Private	Weight of product
description	String	Private	Description of product
product_id	Int	Private	Parent ID of product
reject_reason	String	Private	Reject reason of product
category_id	Int	Private	Category identify of product
status_id	Int	Private	Status identify of product
material	String	Private	Material of product
color	String	Private	Color of product
product_image	String	Private	Image reference link of product
model_id	Int	Private	3D model identifier of product
user_account_username	String	Private	Username of seller
staff_username	String	Private	Username of staff

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.12 product_status

Attribute

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of product status
name	String	Private	Name of product status

Method

Method	Return type	Visibility	Description

Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.13 category

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Unique identifier of category
name	String	Private	Name of category
category_id	Int	Private	Parent ID of category

Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

4.2.14 model

Attribute

Attribute	Type	Visibility	Description
id	UUID	Private	Unique identifier of 3D model
width	Int	Private	Width of model
length	Int	Private	Length of model
height	Int	Private	Height of model
local_storage	String	Private	Reference link at local
remote_storage	String	Private	Reference link at remote storage
price	Double	Private	Servcice pricing of approving product
photo_scene_id	String	Private	The photo scene id generated from third party API when create 3D model

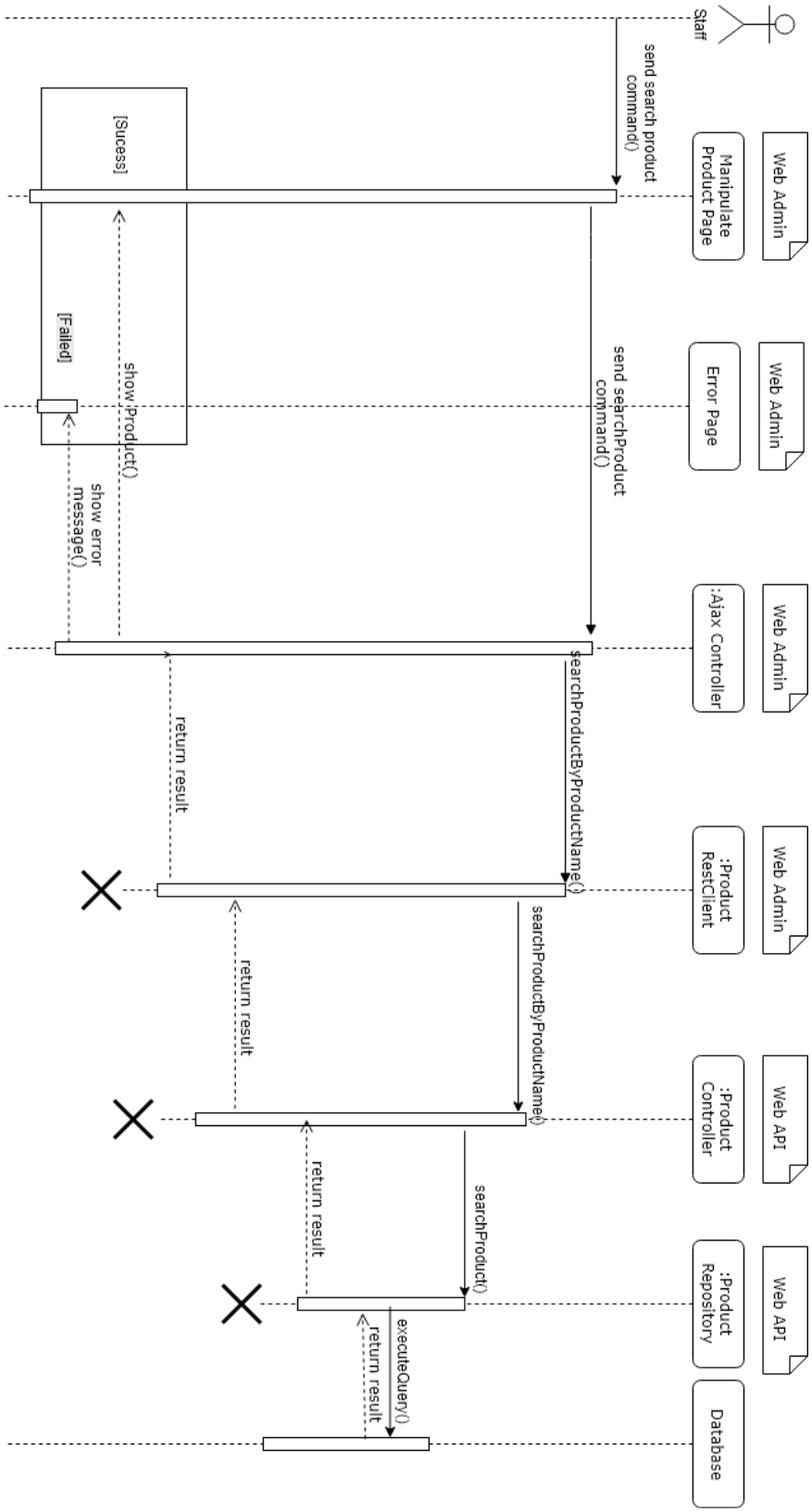
Method

Method	Return type	Visibility	Description
Getter	Attribute type	Public	Get attribute value
Setter	Void	Public	Set value of attribute

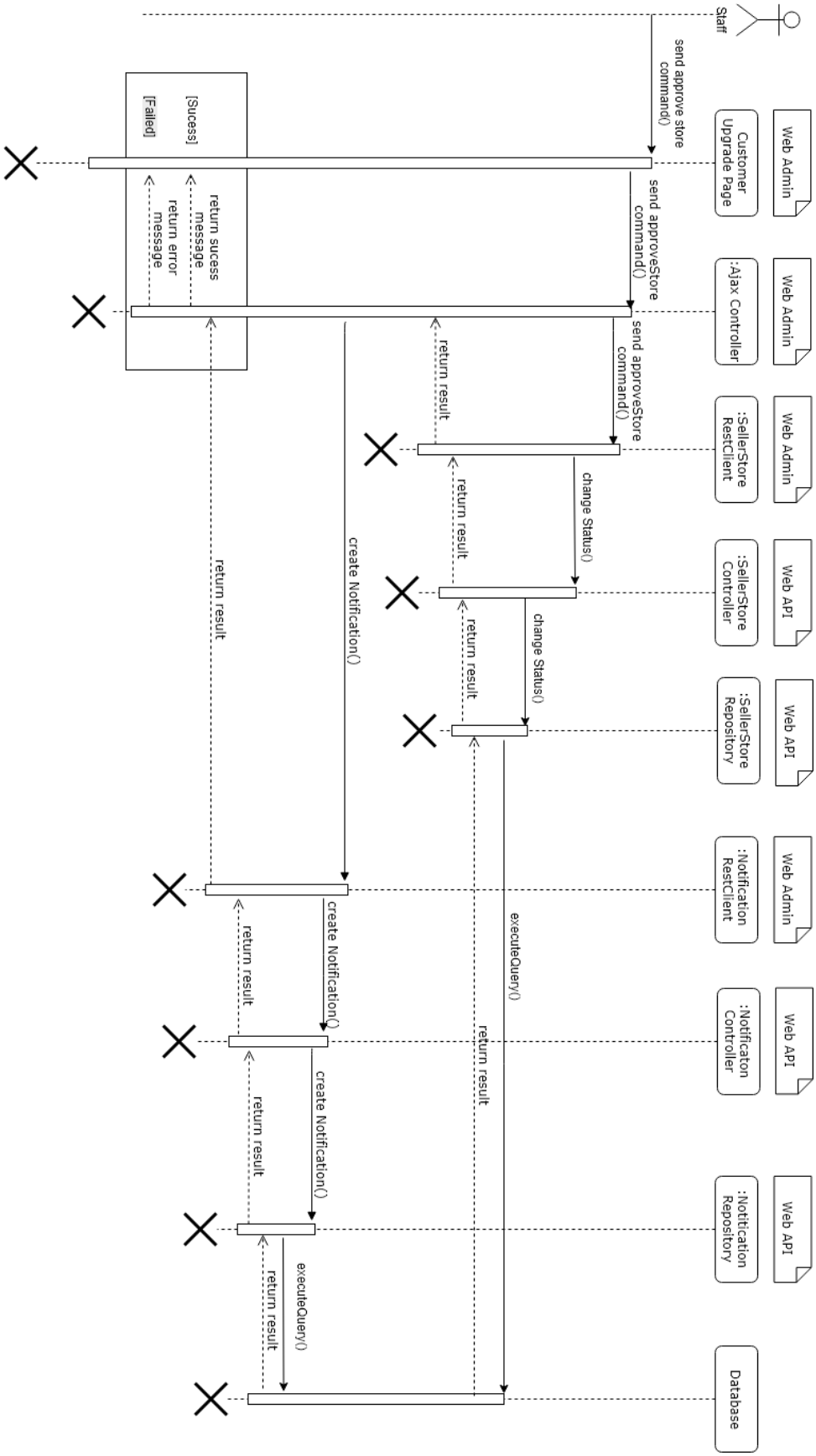
4.3 Interaction Diagram

4.3.1 Sequence Diagram

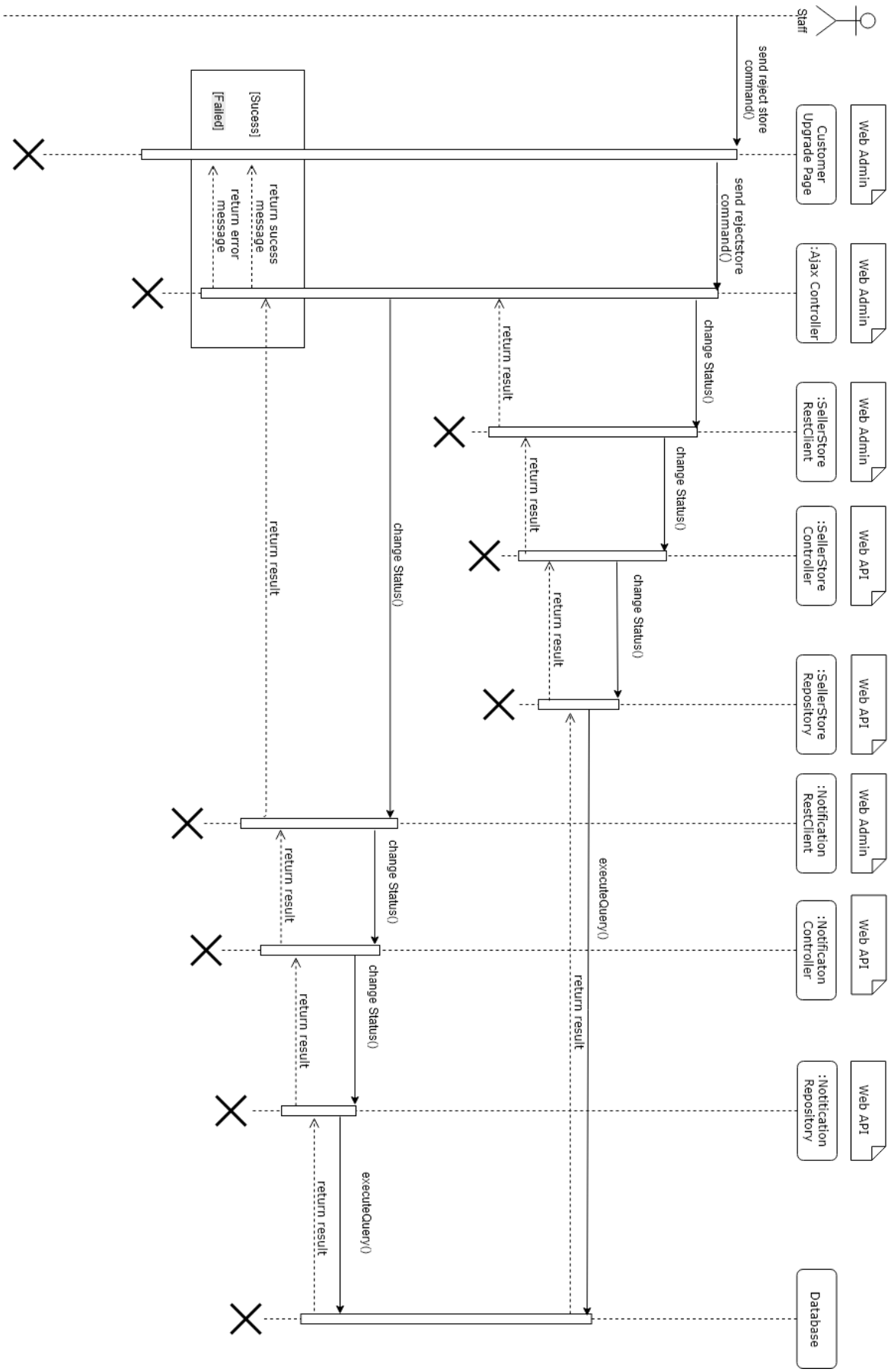
4.3.1.1 Search product



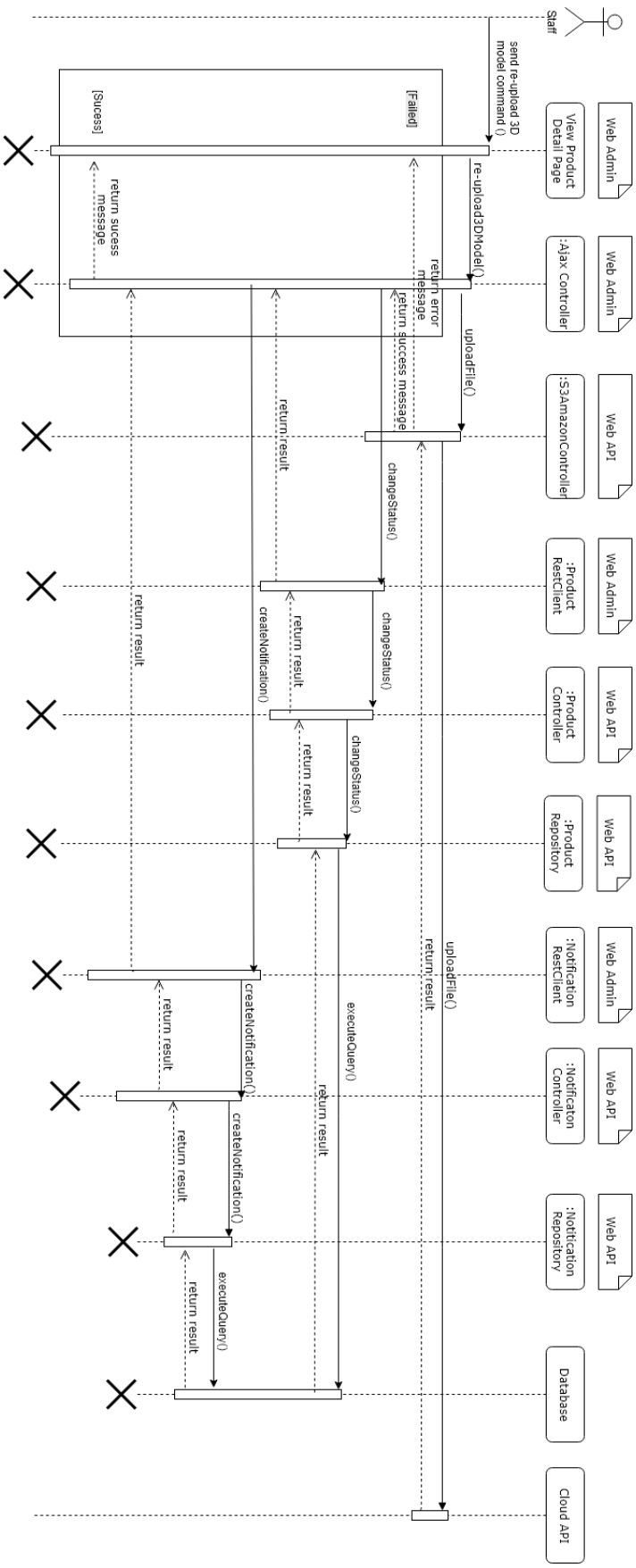
4.3.1.2 Approve upgrade role request



4.3.1.3 Reject upgrade role request

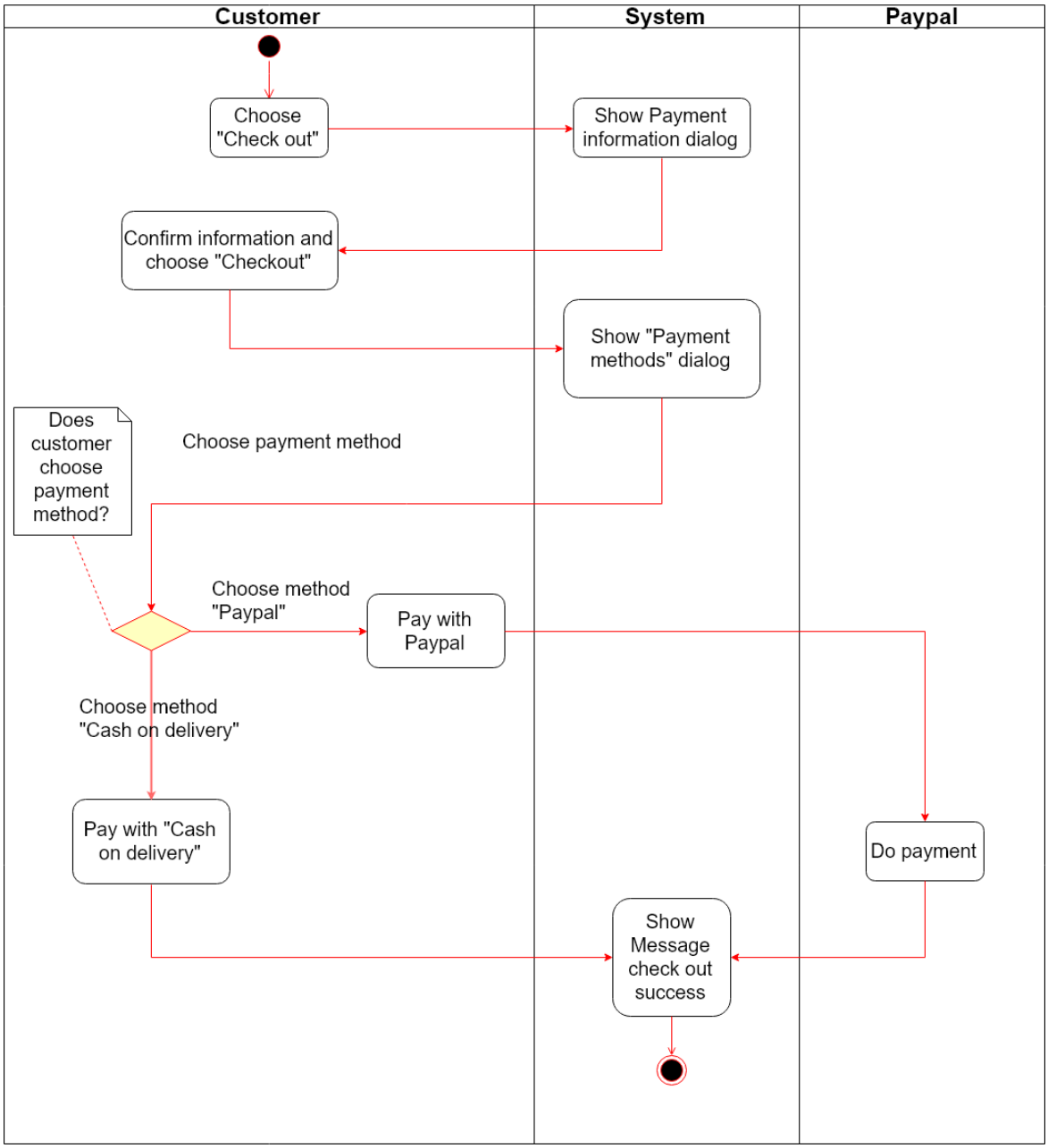


4.3.1.4 Re-upload 3D model

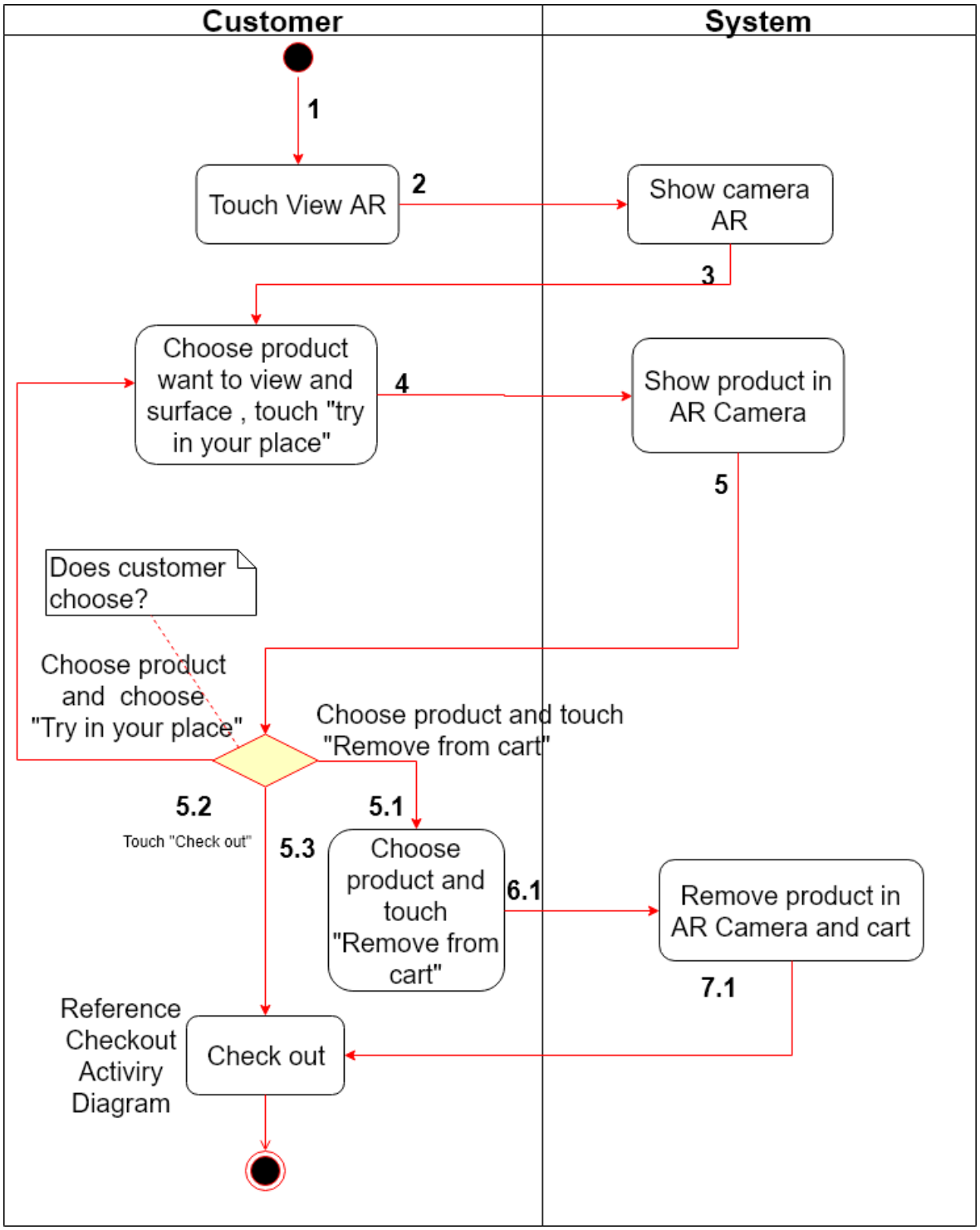


4.3.2 Activity Diagram

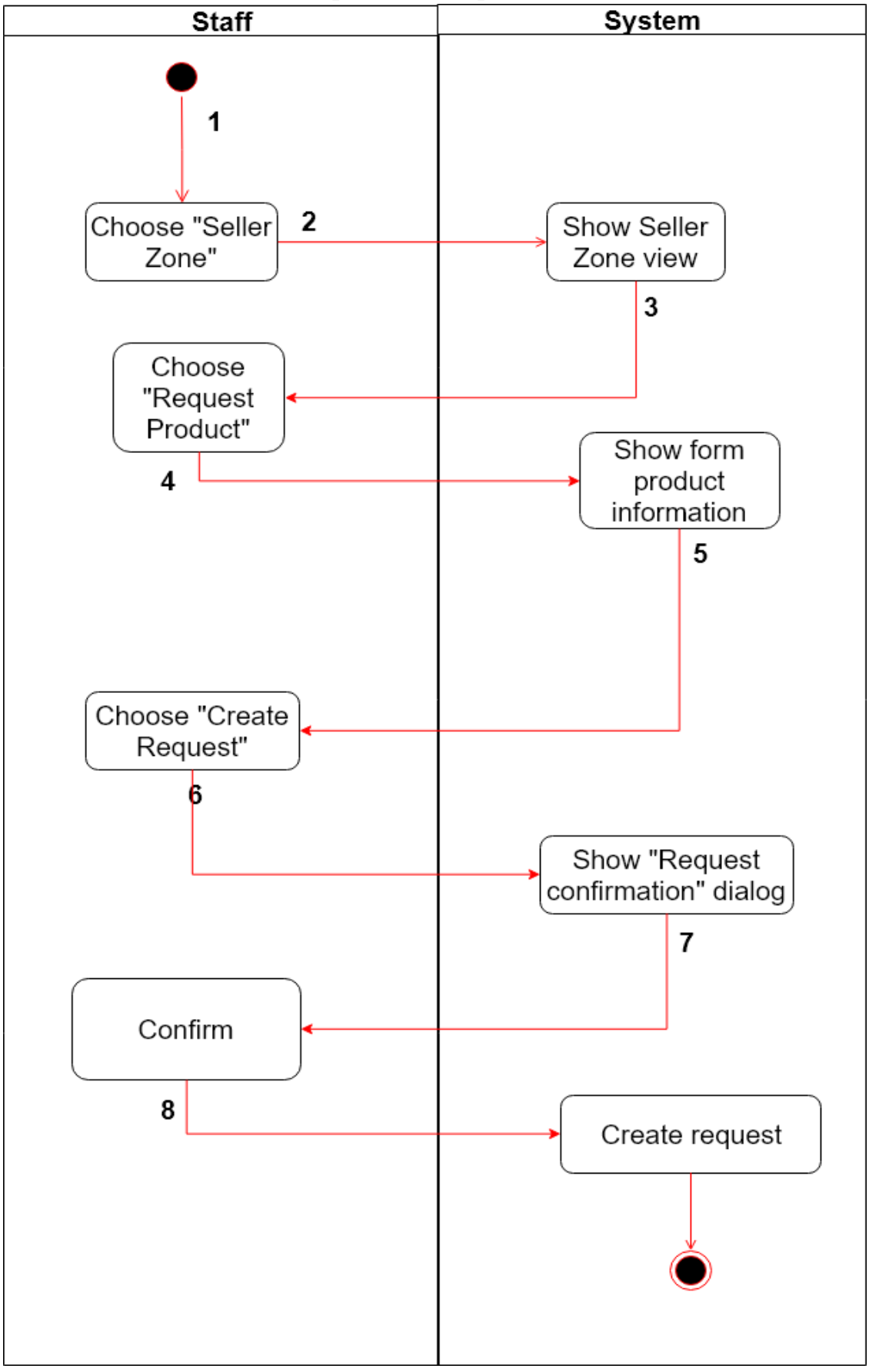
4.3.2.1 Checkout



4.3.2.2 View product in AR view



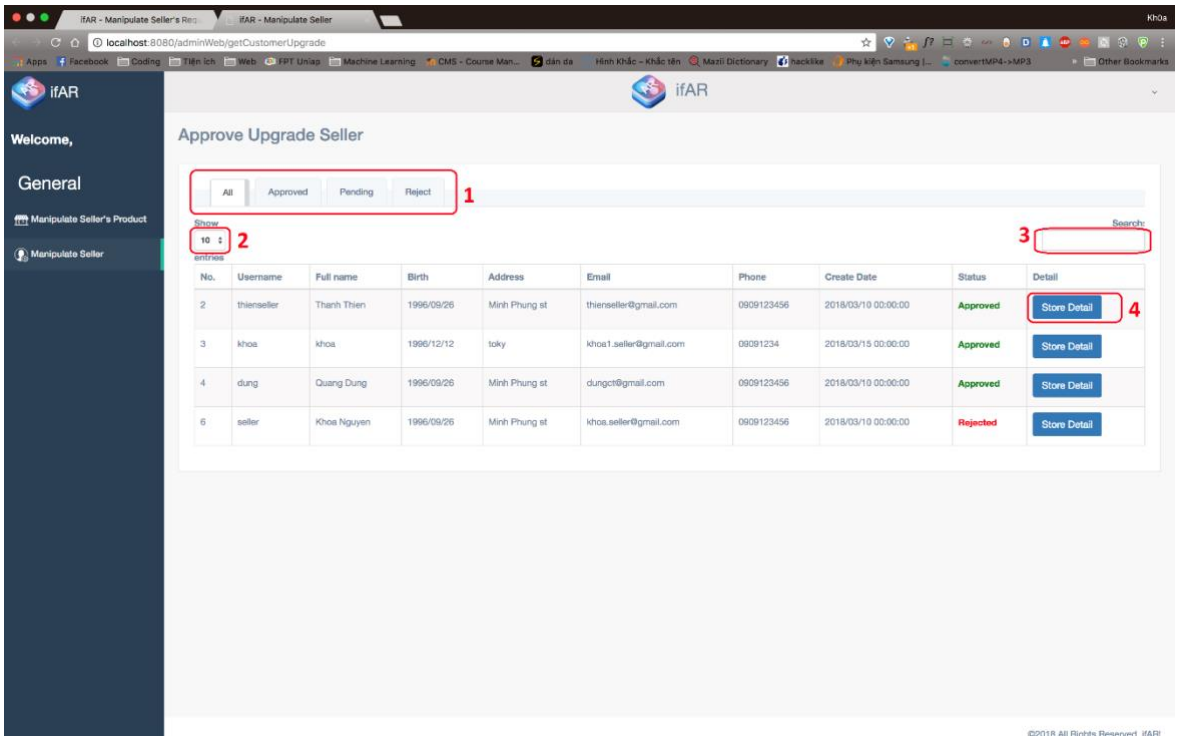
4.3.2.3 Seller request to sell product



5. Interface

5.1 Staff & Designer

5.1.1 Display role request



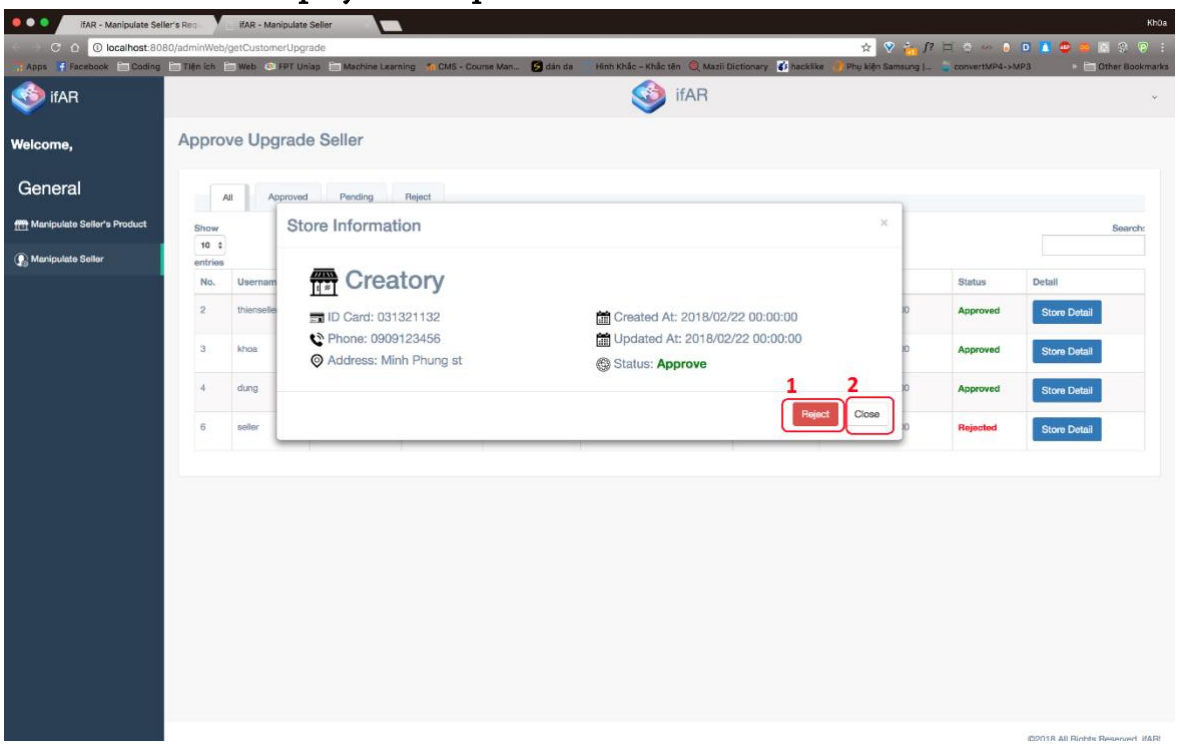
Fields

No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
2	*Choose number of entries*	*Choose number of entries will be display in table*	*No*	*Yes*	*Textbox*	*String*	*3*
3	*Search seller*	*Search seller by username or store name*	*No*	*Yes*	*Textbox*	*String*	*N/A*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Change Tab*	*Switch to other tab following seller's status*	*N/A*	*Change tab*
4	*Show detail of seller*	*Open pop-up which shows seller's detail*	*N/A*	*Show pop-up*

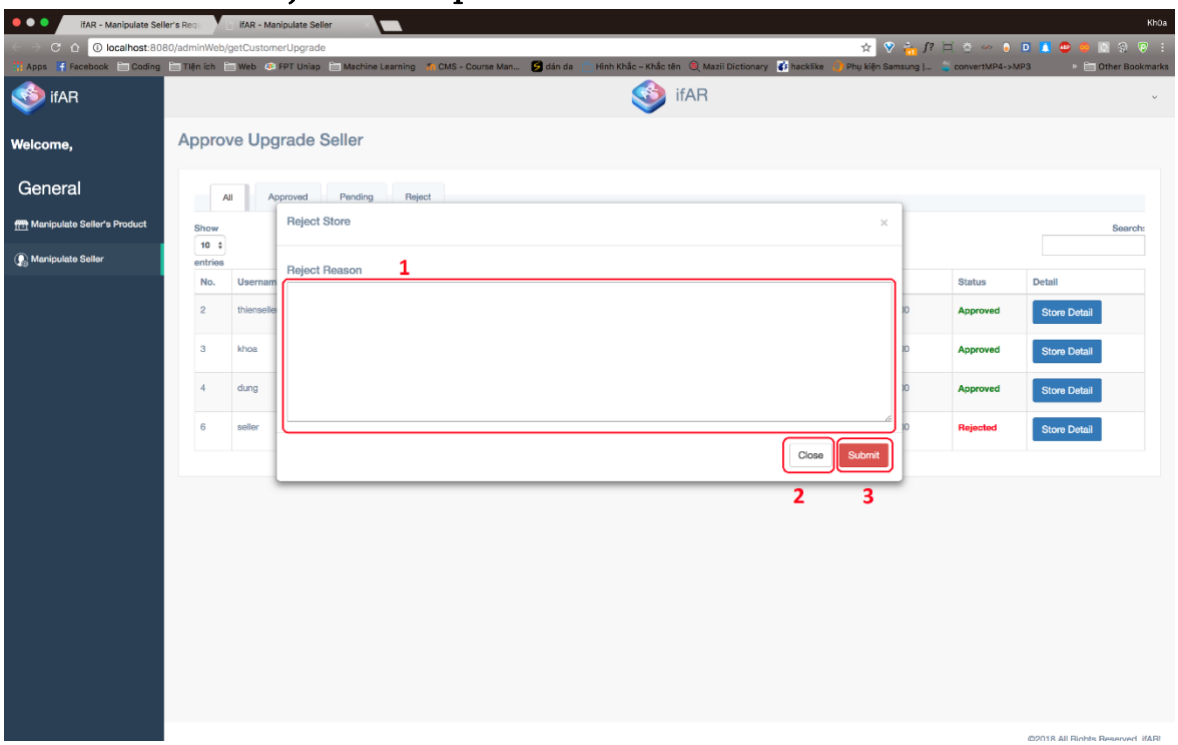
5.1.2 Display role request's detail



Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Reject*	*Reject upgrade role*	*N/A*	*Open pop-up to fill reject reason*
2	*Close*	*Close pop-up*	*N/A*	*Close pop-up*

5.1.3 Reject role request



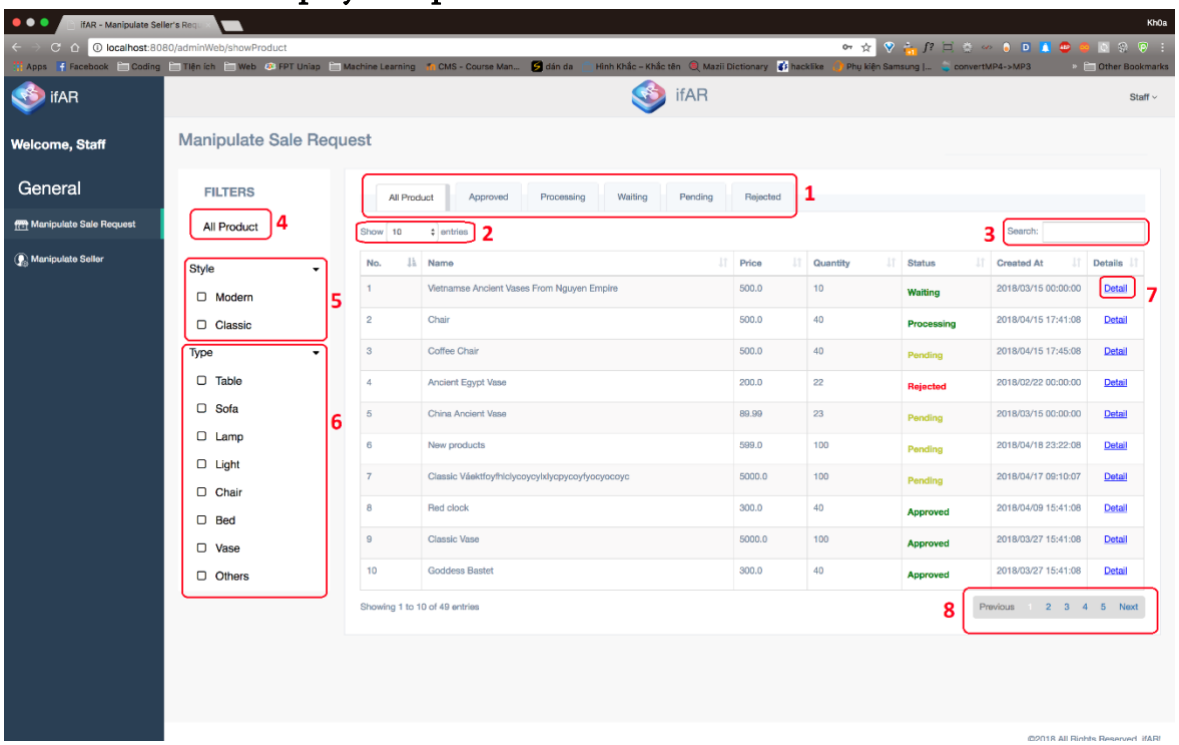
Fields

No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
1	*Reject Reason*	*Fill reject reason why staff rejects this request*	*No*	*Yes*	*TextArea*	*String*	*250*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
2	*Close*	*Close pop-up*	*N/A*	*Close pop-up*
3	*Submit*	*Submit reason to system*	*N/A*	*Close pop-up*

5.1.4 Display sale product



Fields

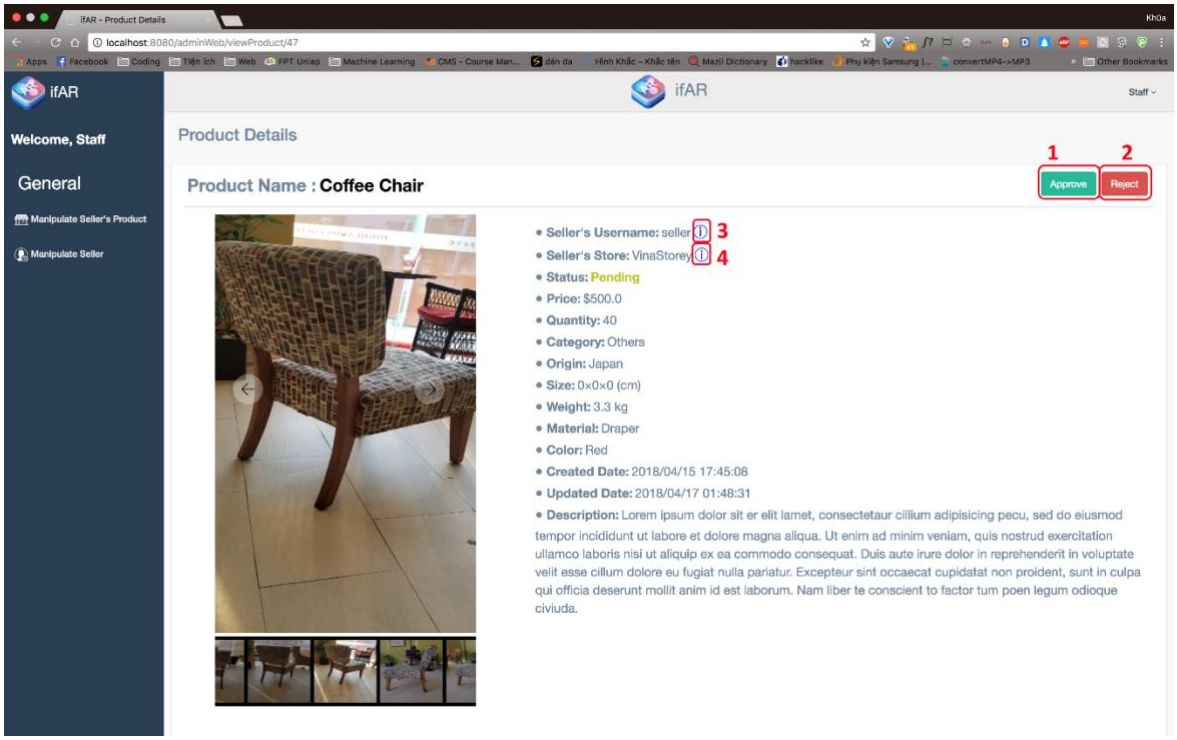
No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
2	*Choose number of entries*	*Choose number of entries will be display in table*	*No*	*Yes*	*Textbox*	*String*	*3*
3	*Search seller*	*Search seller by username or store name*	*No*	*Yes*	*Textbox*	*String*	*N/A*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Change Tab*	*Switch to other tabs*	*N/A*	*Change tab*

		following seller's status		
4	*Display all products*	*Display all products in table*	*N/A*	*Display new data to table*
5	*Filter product by Style*	*Display products following style in table*	*N/A*	*Display new data to table*
6	*Filter product by Category*	*Display products following category in table*	*N/A*	*Display new data to table*
7	*Show detail of seller*	*Open pop-up which shows seller's detail*	*N/A*	*Show pop-up*
8	*Paging*	*Separate data into pages*	*N/A*	*Change page*

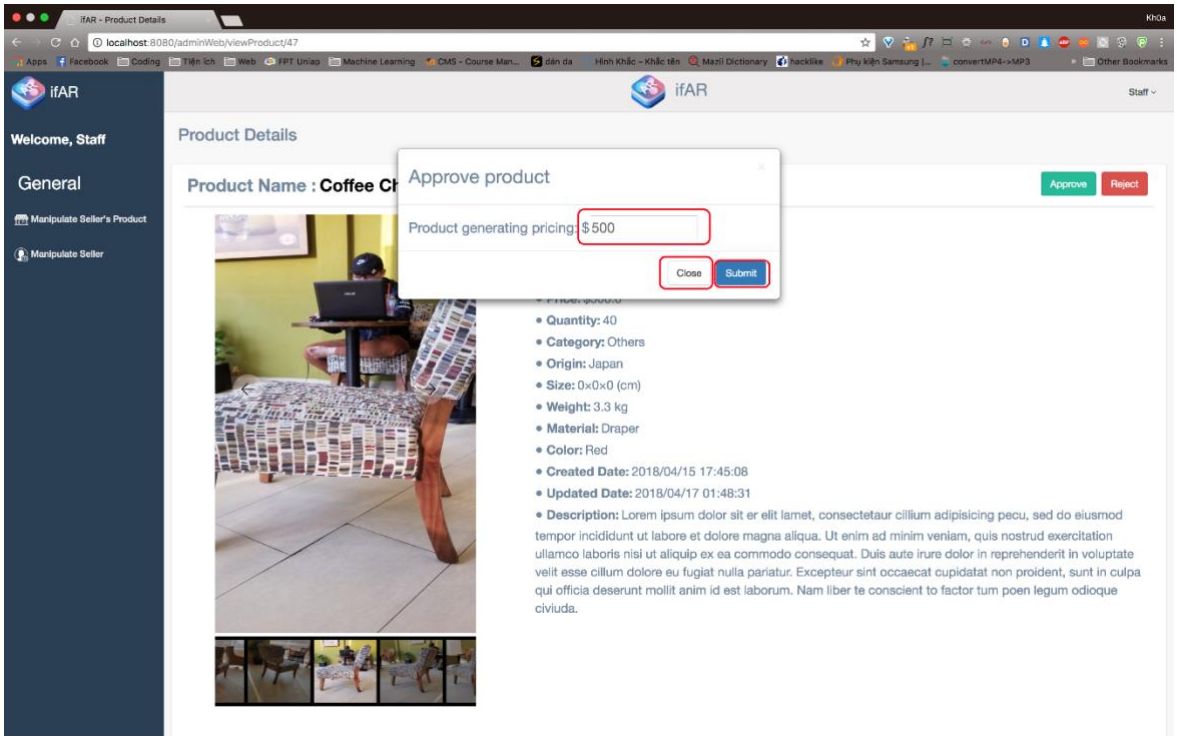
5.1.5 Display sale product detail



Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Approve*	*Approve sale request*	*N/A*	*Show pop-up to fill service pricing*
2	*Reject*	*Reject sale request*	*N/A*	*Show pop-up to fill reject reason*
3	*Seller's information*	*Show seller's information*	*N/A*	*Show pop-up seller's information*
4	*Store's information*	*Show store's information*	*N/A*	*Show pop-up store's information*

5.1.6 Approve sale product request



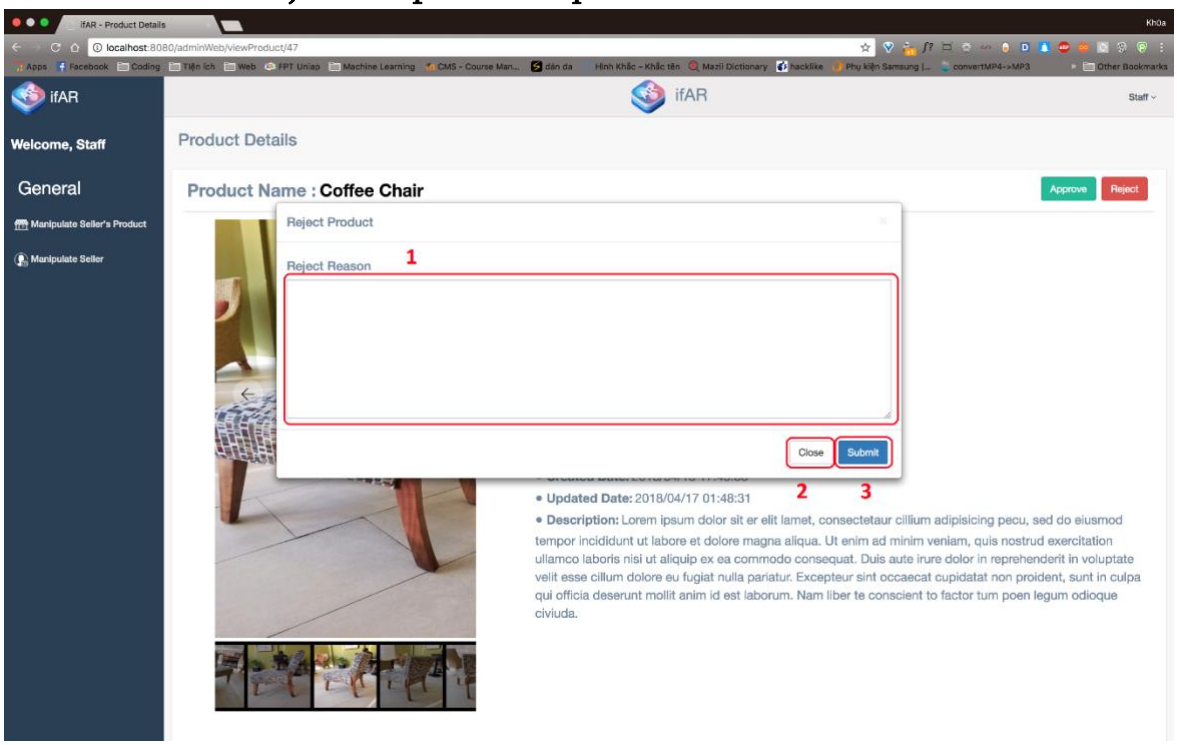
Fields

No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
1	*Service pricing*	*Pricing for approve and generate 3D model of product*	*No*	*Yes*	*Textfield*	*String*	*8*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Submit*	*Submit pricing to system*	*N/A*	*Close pop-up*
2	*Close*	*Close pop-up*	*N/A*	*Close pop-up*

5.1.7 Reject sale product request



Fields

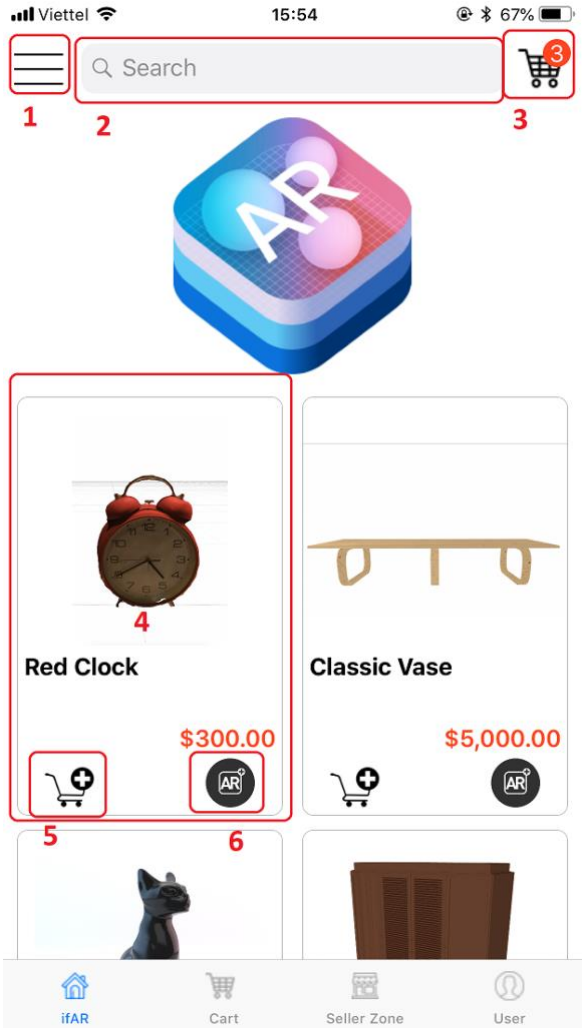
No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
1	*Reject Reason*	*Fill reject reason why staff rejects this request*	*No*	*Yes*	*TextArea*	*String*	*250*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
2	*Close*	*Close pop-up*	*N/A*	*Close pop-up*
3	*Submit*	*Submit reason to system*	*N/A*	*Close pop-up*

5.2 Customer

5.2.1 Select product



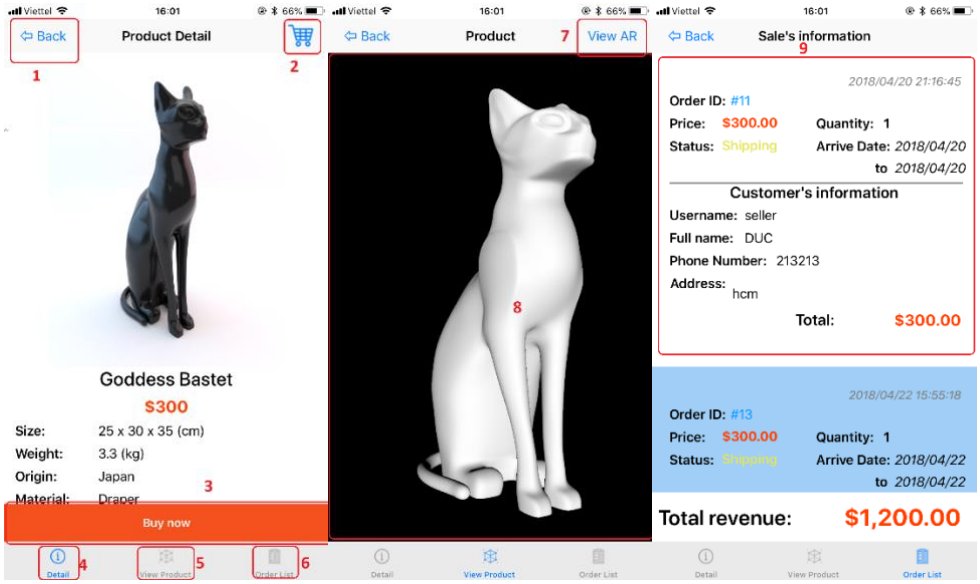
Fields

No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
2	*Search product*	*Search product by name*	*No*	*Yes*	*TextArea*	*String*	*50*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Option*	*Open option*	*No*	*Show pop-up option*
3	*Get cart*	*Get cart*	*No*	*Move to cart view*
4	*Get product's detail*	*View product detail*	*No*	*Move to product's detail view*
5	*Add to cart*	*Add product to cart*	*No*	*Add this product to cart*
6	*View AR*	*View product in AR view*	*No*	*Move to AR view*

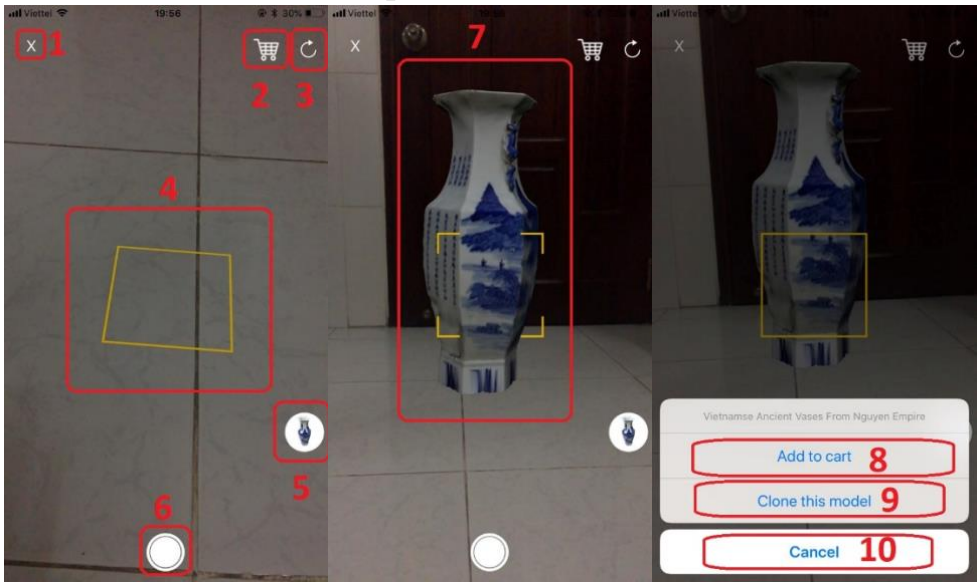
5.2.2 Product detail



Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Back*	*Back to previous view*	*No*	*Display previous view*
2	*Get cart*	*Get cart*	*No*	*Move to cart view*
3	*Add to cart*	*Add product to cart*	*No*	*Add this product to cart*
4	*Product detail*	*Get product detail*	*No*	*Move to product's detail view*
5	*View product in 3D view*	*Simulate product in 3D view*	*No*	*Move to 3D view*
6	*Get order list*	*Get orders of this product*	*No*	*Move to order list view*
7	*View AR*	*View product in AR view*	*No*	*Move to AR view*
8	*Rotate product*	*Rotate product following user's action*	*No*	*Rotate product in 3D view*
9	*Manipulate order*	*Manipulate order of this product*	*No*	*Manipulate order of this product*

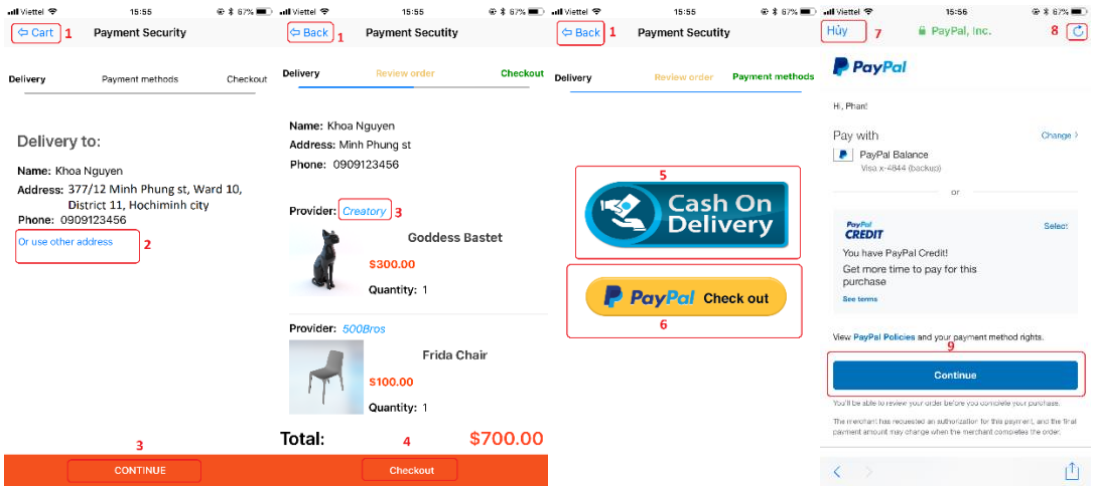
5.2.3 Simulate products in AR view



Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Exit*	*Back to previous view*	*No*	*Display previous view*
2	*Get cart*	*Get cart*	*No*	*Move to cart view*
3	*Reset*	*Resest everything in AR view*	*No*	*Reset surface detection and model in AR view*
4	*Detect surface*	*Detect surface*	*No*	*Surface is detected*
5	*More option*	*More option of product (8, 9, 10)*	*No*	*List buttons (8, 9, 10)*
6	*Screenshot*	*Screenshot*	*No*	*Screenshot in Photos*
7	*Interact product*	*Interact with product in AR view*	*No*	*Change product's aspect*
8	*Add to cart*	*Add to cart*	*No*	*Product is added to card*
9	*Clone model*	*Clone one more product*	*No*	*One more product is added*
10	*Cancel*	*Cancle*	*No*	*Cancle selection option*

5.2.4 Check out

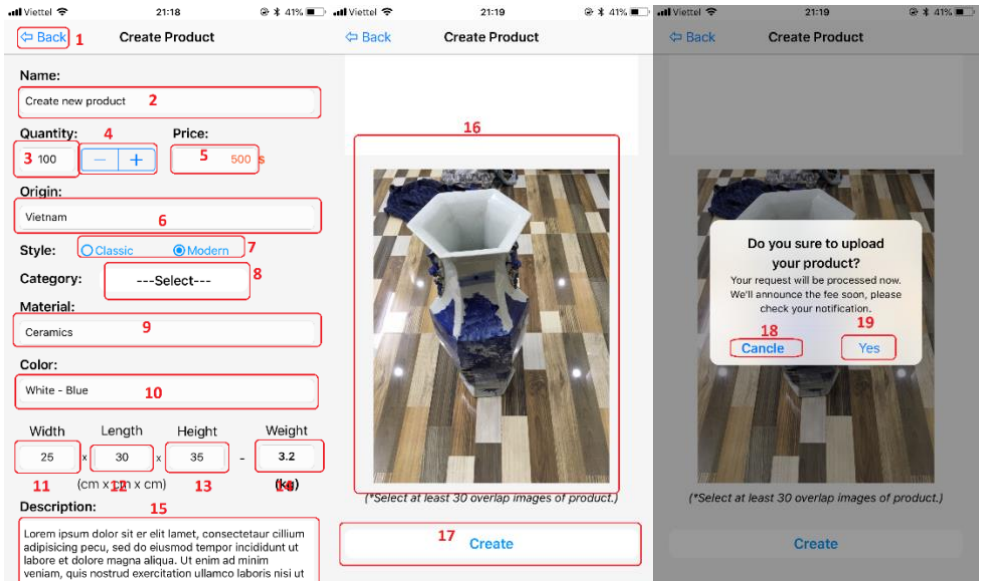


Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Back*	*Back to previous view*	*No*	*Display previous view*
2	*Change address*	*Change delivery address*	*No*	*Show address form*
3	*Get store 's information*	*Get store's information of product*	*No*	*Show store's information*
4	*Checkout*	*Execute payment*	*No*	*Move to payment view*
5	*Cash on delivery*	*Execute payment with Cash on delivery*	*No*	*Finish payment*
6	*Paypad*	*Execute payment with paypal*	*No*	*Open Paypal webview*
7	*Cancel*	*Cancel payment*	*No*	*Display previous view*
8	*Refresh*	*Refresh data*	*No*	*Apply new data*
9	*Checkout*	*Finish payment*	*No*	*Finish payment*

5.3 Seller

5.3.1 Request to sell product



Fields

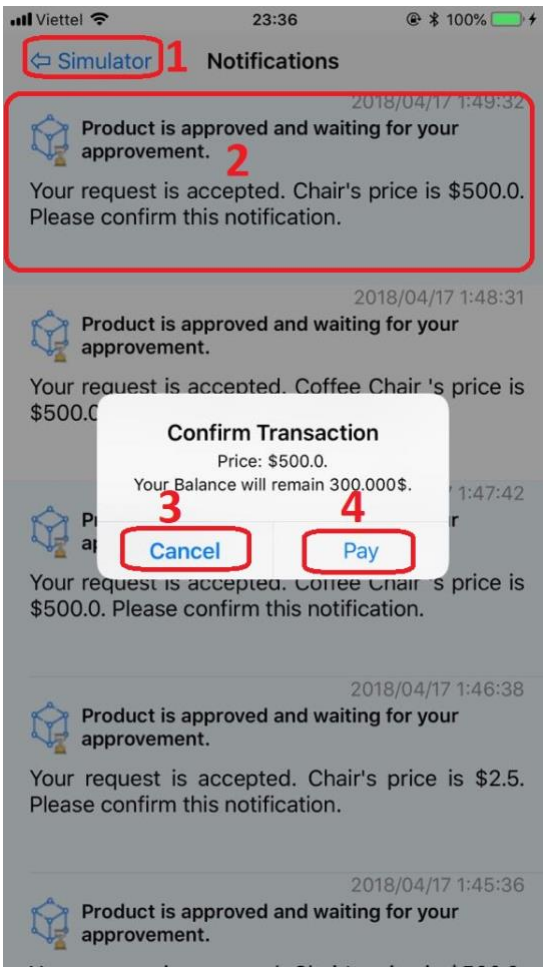
No	*Field Name*	*Description*	*Read only*	*Mandatory*	*Control Type*	*Data Type*	*Length*
2	*Product name*	*Product name*	*No*	*Yes*	*Textfield*	*String*	*50*
3	*Quantiy*	*Quantiy of product*	*No*	*Yes*	*Textfield*	*Int*	*N/A*
5	*Price*	*Price of product*	*No*	*Yes*	*Textfield*	*Double*	*N/A*
6	*Origin*	*Origin of product*	*No*	*Yes*	*Textfield*	*String*	*50*
9	*Material*	*Material of product*	*No*	*Yes*	*Textfield*	*String*	*50*
10	*Color*	*Color of product*	*No*	*Yes*	*Textfield*	*String*	*50*
11	*Width*	*Width of product*	*No*	*Yes*	*Textfield*	*Int*	*N/A*
12	*Length*	*Length of product*	*No*	*Yes*	*Textfield*	*Int*	*N/A*
13	*Height*	*Height of product*	*No*	*Yes*	*Textfield*	*Int*	*N/A*
14	*Weight*	*Weight of product*	*No*	*Yes*	*Textfield*	*Double*	*N/A*
15	*Description*	*Description of product*	*No*	*Yes*	*Textarea*	*String*	*2000*

Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*

1	*Back*	*Back*	*No*	*Back to previous view*
4	*Change quantiy*	*Change quantiy of product*	*No*	*Inscrease/descrease quantity*
7	*Select Style*	*Select Style of product*	*No*	*Radio button is checked*
8	*Select Category*	*Select Category of product*	*No*	*Show selected category*
16	*Select images*	*Select images of product*	*No*	*Display first image of list images*
17	*Create request*	*Create request*	*Yes*	*Show confirmation pop-up*
18	*Cancel request*	*Cancel request*	*No*	*Close pop-up*
19	*Confirm request*	*Confirm request*	*No*	*Request is sent to system*

5.3.2 Notification



Buttons/Hyperlinks

No	*Function*	*Description*	*Validation*	*Outcome*
1	*Back*	*Back*	*No*	*Back to previous view*

2	*Confirm notification*	*Confirm notification*	*No*	*Show pop-up with service pricing*
3	*Cancel*	*Close pop-up*	*No*	*Close pop-up*
4	*Pay*	*Execute payment for notification*	*No*	*Finish payment*

6. Database Design

6.1 Entity relationship diagram (ERD)

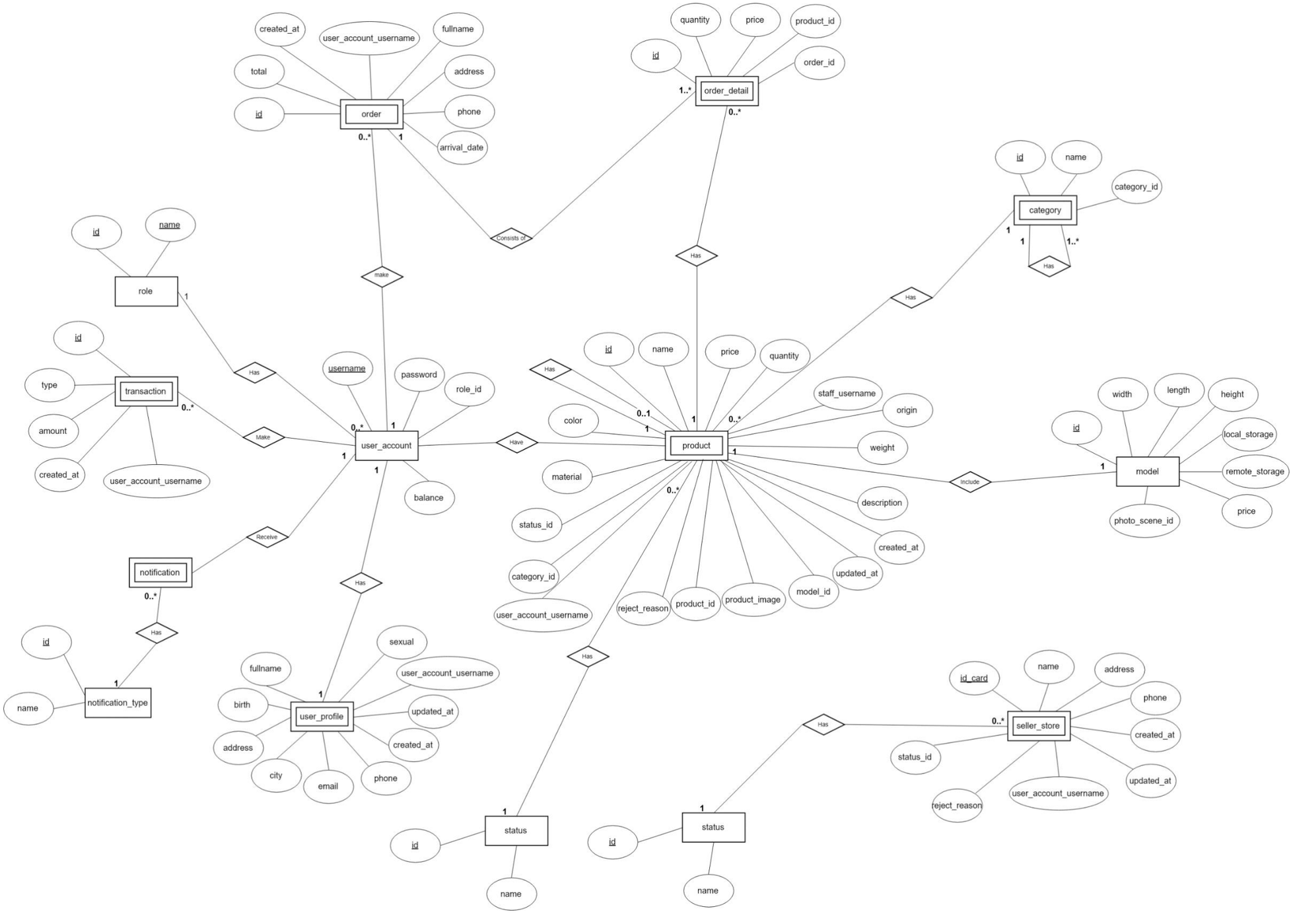


Figure 59 Entity relationship diagram

6.2 Data Dictionary

	Entity Data Dictionary
Entity name	**Description**
role	Contain the role information
order	Contain customer's order information.
order_detail	Contain order detail information.
transaction	Contain the transaction information.
user_account	Contain the user account information of user profile.
product	Contain the product information.
category	Contain the category information of product.
model	Contain the model information of product.
notification	Contain the notification information.
notification_type	Contain the notification type of notification.
user_profile	Contain the user profile information.
seller_store	Contain the seller's store information
status	Contain the status information.

7. Algorithms

7.1 Augmented Reality – Plane Detection

7.1.1 Definition

The plane detection algorithm of Augmented Reality automatically detects multiple planes based on the proposed constrained sampling strategy.

7.1.2 Define Problem

The whole concept behind Augmented Reality is blending together the reality around us with virtual objects that exist only within our App. In order to be able to do that successfully, we need to be aware of the geometry of our surroundings.

In other words, we need to be able to identify the ceilings, walls, tables and other physical objects.

Each plane has many specific point on it which will help us to build an algorithm to recognize plane. But we get some problems:

- Need a lot of time for building a good identification algorithm.
- Solution need to be stable to ensure reliability of system.

7.1.3 Solution

Then we decide to use ARKit to detect plane with following concept:

- **Plane detection:** Plane detection is the ability to determine surfaces or planes in the physical environment. This is thing like the ground floor or may be a table.

- **Hit-testing:** Getting an intersection with the real-world topology so that you can place your virtual object in the physical world.
- **Light estimation:** Use to render or correctly light your virtual geometry to match that of the physical world.

7.1.4 Work flow

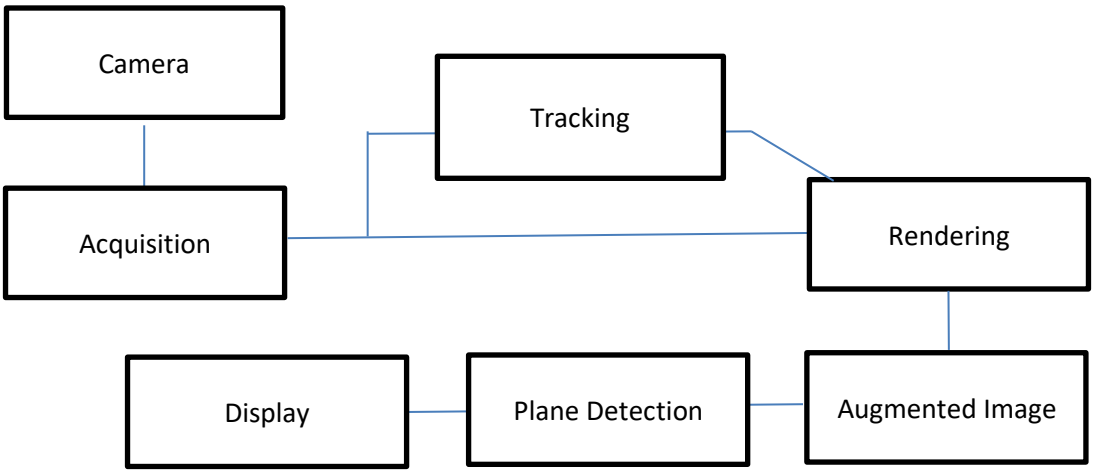


Figure 60 Augmented Reality - Plance detection work flow

7.1.5 Reference

Augmented Reality with ARKit: Detecting Planes: https://digitalleaves.com/blog/2017/10/augmented-reality-with-arkit-detecting-planes/

7.2 Augmented Reality – Placing Object

7.2.1 Definition

The placing object algorithm of Augmented Reality used for place object on existing surface in Augmented Reality. It determines the surface existing in Augmented Reality and place the object on that surface.

7.2.2 Define Problem

After the system detected surface, we need an algorithm to help the system place object on existing surface in Augmented Reality.

To implement the algorithm, we use Hit-testing concept of ARKit to solve the algorithm of placing object on existing surface in Augmented Reality.

7.2.3 Solution

To solve this problem, we should use Hit-testing concept of ARKit. Hit testing searches for real-world objects or surfaces detected through the AR session's processing of the camera image. A 2D point in the image coordinates

can refer to any point along a 3D line that starts at the device camera and extends in a direction determined by the device orientation and camera projection. This method searches along that line, returning all objects that intersect it in order of distance from the camera. After surface detected, hit-testing will support to place the object on that surface.

7.2.4 Work flow

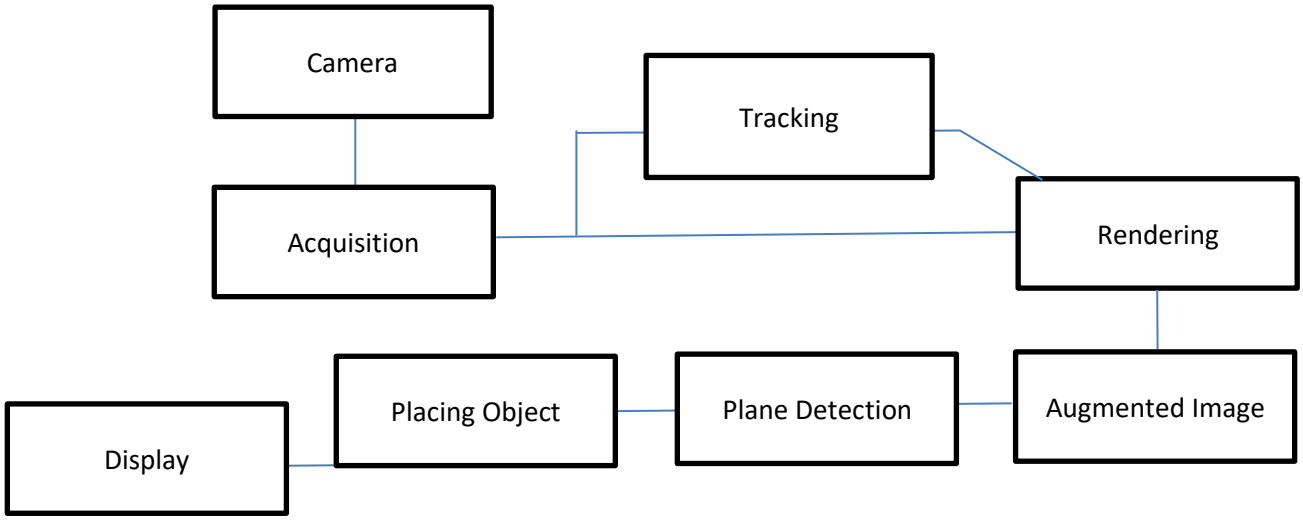


Figure 61: Augmented Reality – Placing Object work flow

7.2.6 References

Apple Developer Documentation: Hit-test method: https://developer.apple.com/documentation/arkit/arframe/2875718-hittest

7.2.7 Example

Place 3D model of Ancient vase on table:

Figure 62: Augmented Reality – Placing Object example

7.2 Reality Capture

7.2.1 Definition

Reality capture is the process of scanning an object, building, or site and producing a digital model representation—allows today's builders to capture site data quickly and more accurately than ever before and connect it directly to the digital design process. The result is a comprehensive 3D model based on millions of data points mapping the entire site, whether it's a building renovation or an infrastructure project, often including a look at the systems that function below the surface. Reality Capture uses object recognition - a process for identifying a specific object in a digital image or video. Object recognition algorithms rely on matching, learning, or pattern recognition algorithms using appearance-based or feature-based techniques, to detect the object inside series of input images.

Thanks to Reality Capture, even user who doesn't have experience in design is able to create 3D model, it also saves huge time and effort comparing to traditional way. The objects have small or medium size and average complexity are created in short time with almost exactly exteriority. Even smartphone's camera can be used for Reality Capture.

7.2.2 Define Problem

In normally, interior furniture is usually represented by list of images or in advance with 360° videos which can't satisfy customer. Customers feel hard to determine the actual size and shape of furniture when they put them to their house.

To solve this problem, some online store applied 3D model to their website to give customer a better view. However, creating 3D model costs much time and money which makes product's price more expensive.

7.2.3 Solution

To solve this problem, we use Reality Capture to generate 3D model with Semi-auto processing that saves a lot of time and money. User only need to provide series of overlap pictures capturing the product with all aspects.

7.2.4 Work flow

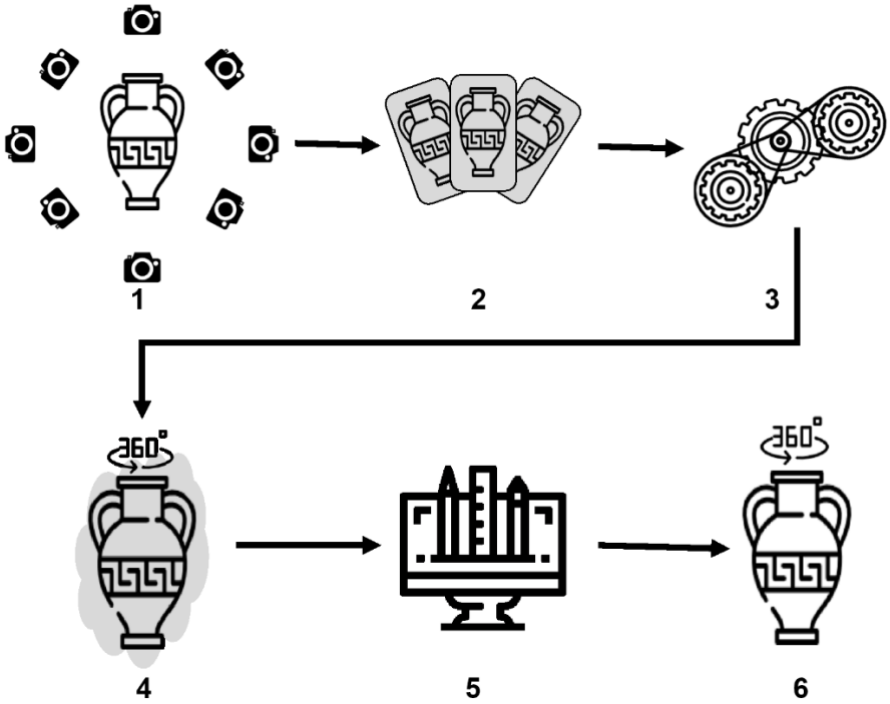


Figure 63: Reality Capture work flow

1. Capture series of overlap pictures of object with same distance at least 30 images.
2. Send these pictures to system with information about object: width, length, heigh and color.
3. Scheduler is going to generate 3D model from pictures provided by user.
4. Result model contains redundant textures (raw model) and need to upgrade quality.
5. Designer has responsibility to edit raw model, the raw model is removed unnecessary texture and fixed size, detail.
6. After being edited, the model is ready to represent the object. This model can be used in both AR view or 3D view.

7.2.5 References

Kylee Swenson (2017). What is Reality Capture?: Redshift by Autodesk. https://www.autodesk.com/redshift/what-is-reality-capture/

7.2.7 Example

The example below is our Ancient Ceramic Vase:

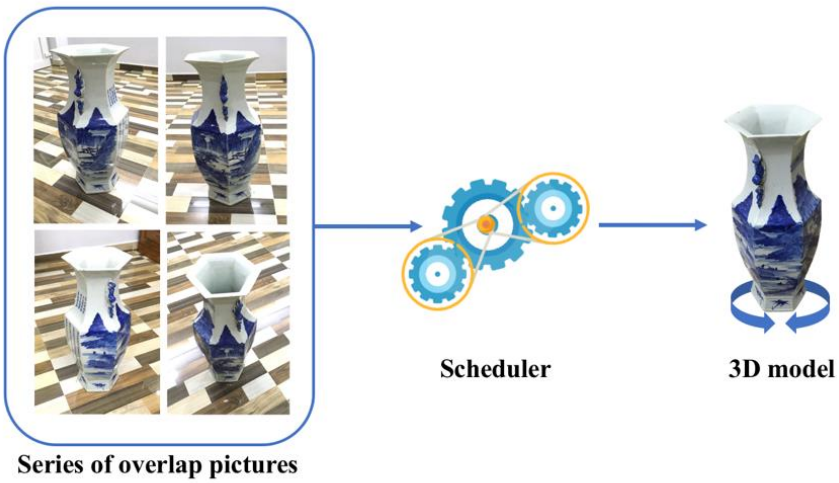


Figure 64: Reality Capture example

E. System Implementation & Test

1. Introduction

1.1 Overview

This section provides in detail all necessary information about implementation information of VASH which includes physical database design, system state machine diagram and testing procedure which includes test plans, test cases and test result.

1.2 Test Approach

We use black box test approach because to estimate our virtual assistant quality, tester need to aware of what VA should do but does not have the knowledge of how it does it. If tester had knowledge about the technical inside it, testing process won't be objectivity.

2. Database Relationship Diagram

2.1 Physical Diagram

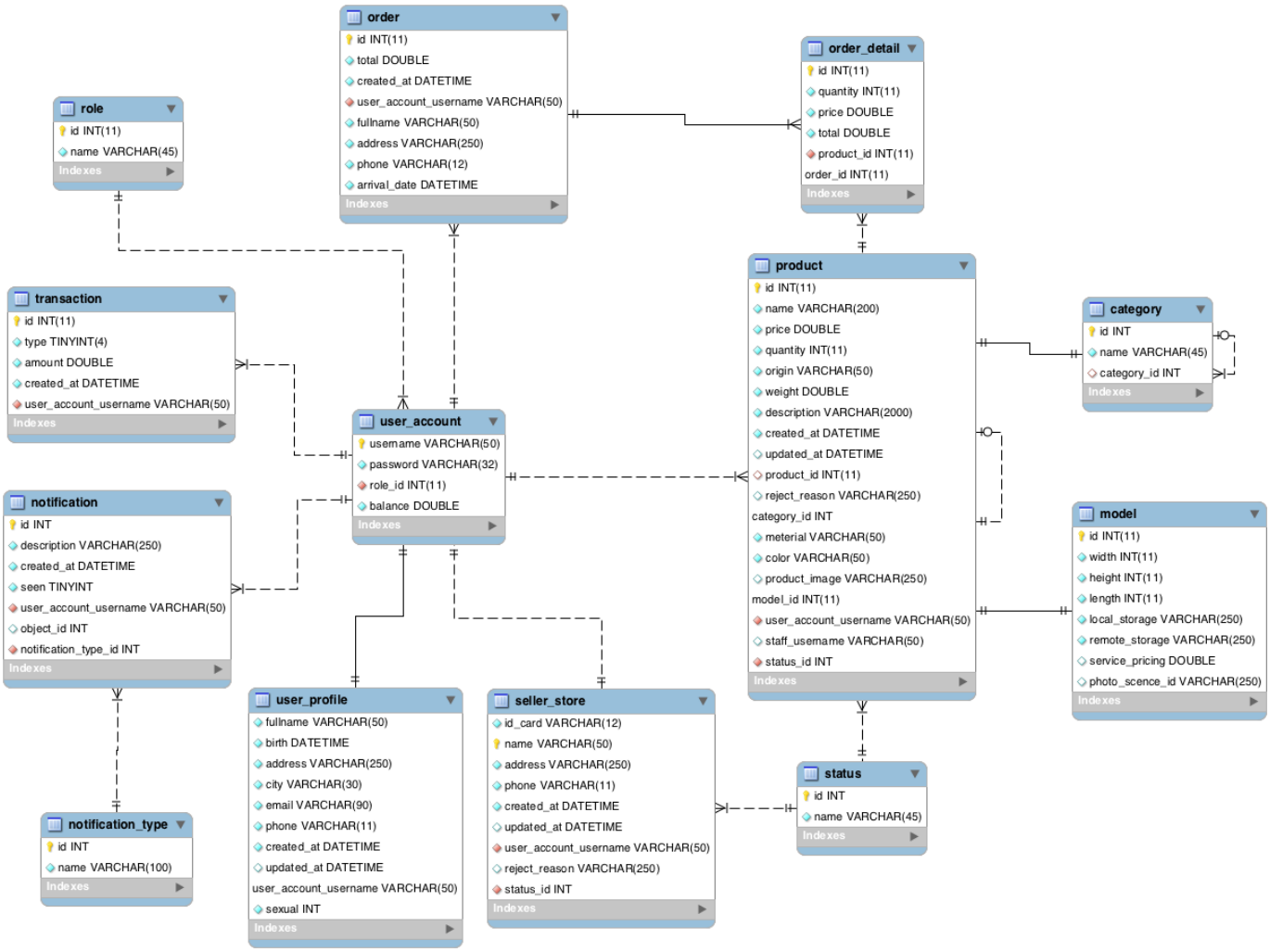


Figure 65: Extend Entity Relation Diagram

2.2 Data Dictionary

Data dictionary: describe content of all tables	
Table Name	**Description**
role	Table which contains the role information
order	Table which contains customer's order information.
order_detail	Table which contains order detail information.
transaction	Table which contains the transaction information.
user_account	Table which contains the user account information of user profile.
product	Table which contains the product information.
category	Table which contains the category information of product.
model	Table which contains the model information of product.
notification	Table which contains the notification information.

notification_type	Table which contains the notification type of notification.
user_profile	Table which contains the user profile information.
seller_store	Table which contains the seller's store information
status	Table which contains the status information.

Entity name	Attributes	Description	Null
role	id {pk}	Unique identifier	No
	name	Role name	No
order	id {pk}	Unique identifier	No
	total	Total of order	No
	created_at	Create date	No
	user_account_username{fk}	Username of user	No
	fullname	Full name of user	No
	address	Delivery address	No
	phone	Phone number of user	No
	arrival_date	Expected arrival date	No
order_detail	id {pk}	Unique identifier	No
	quantity	Quantity of product	No
	price	Price of product	No
	total	Total of order tail	No
	product_id {fk}	Identifier of product	No
	order_id {fk}	Identifier of order	No
transaction	id {pk}	Unique identifier	No
	type	Transaction type	No
	amount	Amount of money in transaction	No
	created_at	Create date	No
	user_account_username{fk}	Username of user	No
user_account	username{pk}	Username of user	No
	password	Password of account	No
	role_int	Unique identifier of role	No
	balance	Balance in account	No
product	id {pk}	Unique identifier	No

	name	Name of product	No
	price	Price of product	No
	quantity	Quantity of product	No
	origin	Origin of product	No
	weight	Weight of product	No
	description	Description of product	No
	created_at	Create date	No
	updated_at	Update date	Yes
	product_id {fk}	Identifier of parent of product	Yes
	reject_reason	Reason why product is rejected	Yes
	category_id {fk}	Identifier of category	No
	material	Material of product	No
	color	Color of product	No
	product_image	Image of product on marketplace	Yes
	model_id {fk}	Identifier of model	No
	user_account_username {fk}	Username of user	No
	staff_username	Username of staff who reviews product	Yes
	status_id	Unique identifier of status	No
category	id {pk}	Unique identifier	No
	name	Name of category	No
	category_id {fk}	Identifier of parent caterory	Yes
model	id {pk}	Unique identifier	No
	width	Width of model	No
	height	Height of model	No
	length	Length of model	No
	local_storage	Reference link of local storage which stores model data	No
	remote_storage	Reference link of remote storage which stores model data	No
	service_pricing	Service pricing of approving product	Yes
	photo_scence_id	Identifier of model in Reality Capture	Yes
notification	id {pk}	Unique identifier	No
	description	Description of notification	No
	created_at	Create date	No

	seen	Check if notification is seen or not	No
	user_account_username {fk}	Username of user	No
	object_id	Identifier of object that relates the notification	Yes
	notification_type_id {fk}	Notification type	No
notification_type	id {pk}	Unique identifier	No
	name	Name of notification type	No
user_profile	fullname	Full name of user	No
	birth	Date of birth of user	No
	address	Address of user	No
	city	City of user	No
	email	Email of user	No
	phone	Phone of user	No
	created_at	Create date	No
	updated_at	Update date	Yes
	user_account_username {fk}	Username of user	No
	sexual	Sexual of user	No
seller_store	id_card {unique}	Identity card of seller	No
	name	Store name	No
	address	Store address	No
	phone	Phone number of store	No
	created_at	Create date	No
	updated_at	Update date	Yes
	user_account_username {pk}	Username of user	No
	reject_reason	Reason why store is rejected	Yes
	status_id {fk}	Identifier of status	No
status	id {pk}	Unique identifier	No
	name	Name of status	No

Table 13: EERD Data Dictionary

3. Performance Measures

3.1 Augmented Reality Plane Detection Performance

The speed of plane detection will depend on light intensity,

camera quality and camera steady. In worst case, the camera cannot detect plane if not enough light or too much light with shiny specular highlights, lack of texture, and fast movement. In happy case, time to detect plane is under 1 second.

3.2 Reality Capture Performance

The quality of 3D model generated from Reality Capture will depend on number of images and quality of images. Time to generate a 3D model is about 60 – 70 minutes and accuracy is about 80%. In worst case, the 3D model will generate fail if amount of number of images not enough (recommended at least 30 images).

4.Test Plan

The purpose of this section is to verify and ensure that VASH meets its design specification and other requirements from user. The following part will describe which features to be tested and which will not.

4.1 Features to be tested

4.1.1 Web application

- Approve sale product request
- Reject sale product request

4.1.2 Mobile application

- Seller requests to sell product
- Simulate products via AR view.
- Execute payment via Paypal.

4.2 Features not to be tested

- Functions are not listed aboves and user management functions won't be test.

5. System Testing Test Case

5.1 Communication Diagram

5.1.1 Manipulate sale product request

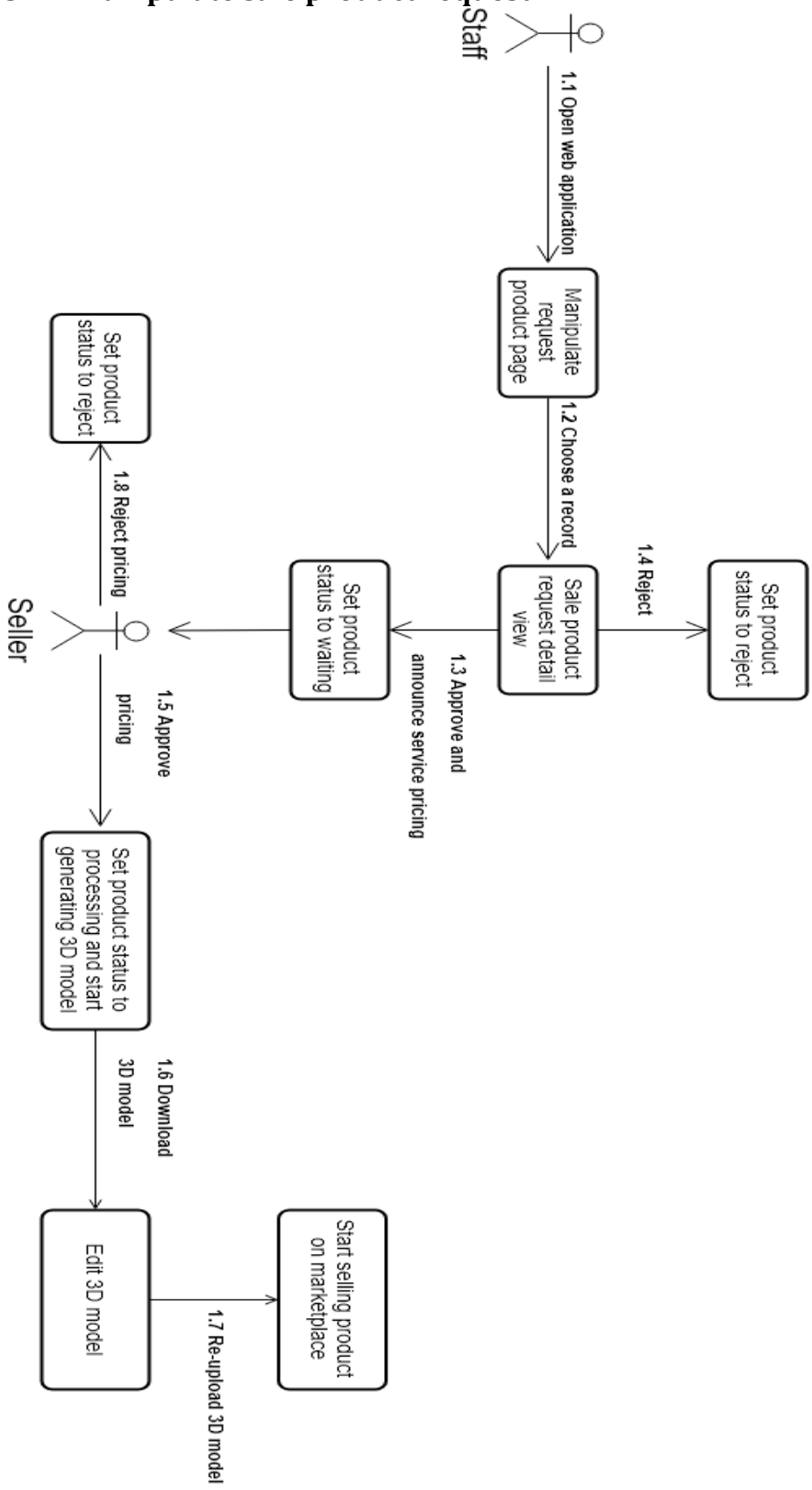


Figure 66: Manipulate sale product request test flow

5.1.2 Simulate product via AR view

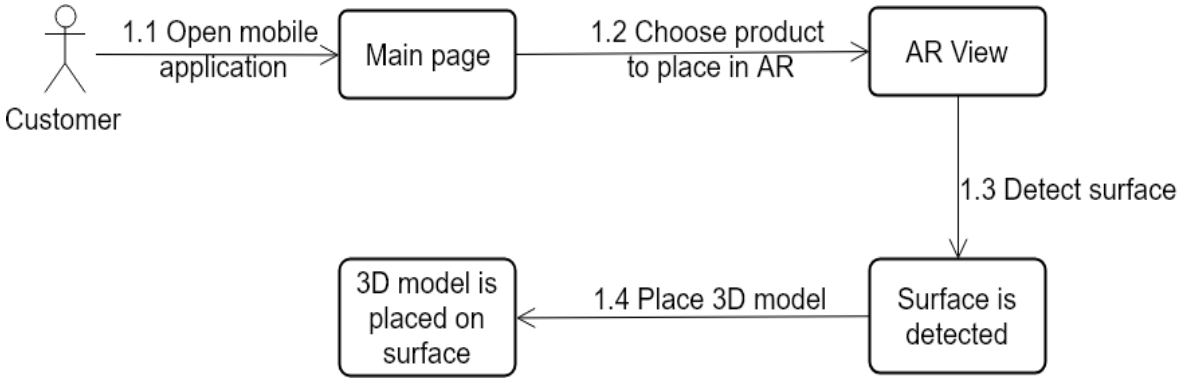


Figure 67: Simulate product via AR view test flow

5.1.3 Checkout via Paypal

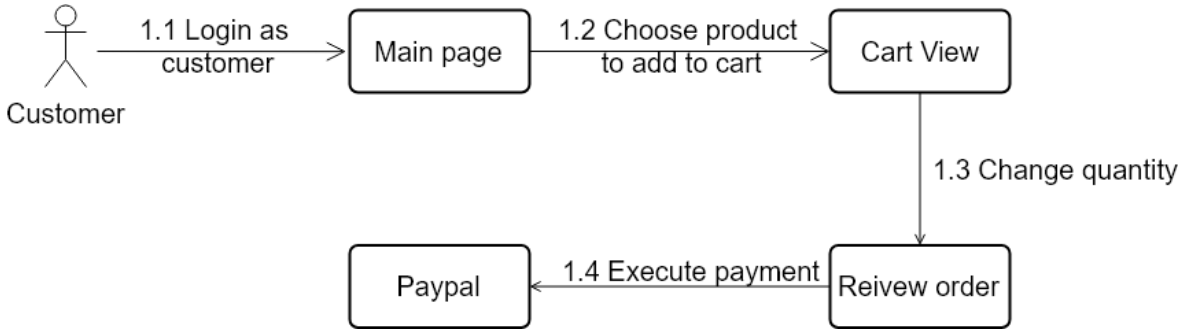


Figure 68: Checkout via Paypal test flow

5.1.4 Seller requests to sell product

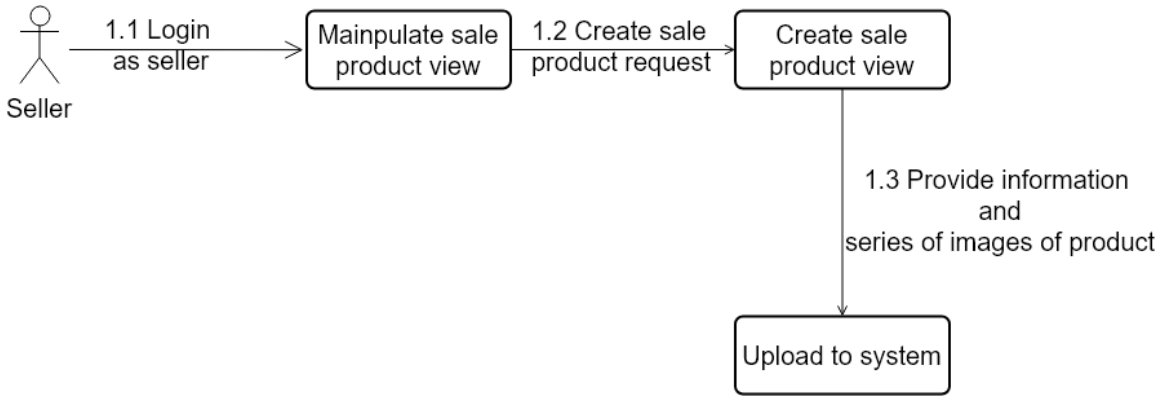


Figure 69: Seller requests to sell product test flow

5.1 Web Application Test case

5.1.1 Approve sale product request

ID	Test Case Description	Test Case Procedure	Expected output	Result	Test Date
ASP1	Approve sale product	0. Login with role "Staff" 1. Go to "Manipulate request product" view. 2. Click "detail" button in record row from table contain list of request product. 3. Click "Approve" button in "Sale product request detail" view. 4. Input field "Service pricing". 5. Click "Submit" button.	Product status change to processing.	Passed	16/04 /2018

5.1.2 Reject sale product request

ID	Test Case Description	Test Case Procedure	Expected output	Result	Test Date
RSP1	Reject sale product	6. Login with role "Staff" 7. Go to "Manipulate request product" view. 8. Click "detail" button in record row from table contain list of request product. 9. Click "Reject" button in "Sale product request detail" view. 10. Input field "Reject reason". 11. Click "Submit" button.	Product status change to rejected.	Passed	16/04 /2018

5.2 Mobile Application Test case

5.2.1 Simulate product via AR view

ID	Test Case Description	Test Case Procedure	Expected output	Result	Test Date
SPC1	Place product's 3D model in flast surface or plane through camera.	12. Go to "Main Page" View. 13. Select product from "Main Page" view to place in AR. 14. Move the phone to detect surface. 15. Tap to place 3D model on surface.	Product's 3D model is placed in flast surface or plane through camera.	Passed	16/04/2018

5.2.2 Request sale product

ID	Test Case Description	Test Case Procedure	Expected output	Result	Test Date
RSP1	Create a new product request.	16. Login with role "Seller" 17. Go to "Manipulate sale product" view. 18. Click "+" button in "Manipulate sale product" view 19. Fill information of product. 20. Choose images of the product. 21. Click "Create" button. 22. Click "Yes" at confirm dialog.	New product request will be created	Passed	16/04/2018

5.2.3 Checkout via Paypal

ID	Test Case Description	Test Case Procedure	Expected output	Result	Test Date

CHK1	Add products to cart and checkout	23. Login with role "Customer" 24. Go to "Main Page" View. 25. Click "Add to cart" button from "Main Page" view to add product to cart. 26. Go to "Cart" view. 27. Click "Checkout" button in "Cart" view. 28. Review order in "Reivew order" view. 29. Click "Pay with Paypal" in "Select payment method" view. 30. Login with Paypal account. 31. Click "Continue" button to proceed checkout.	New customer order will be created	Passed	16/04/ 2018

F. Software User's Manual

1. Installation Guide

1.1 Setting up environment

There are some recommend about specifications after testing performance according to recommend development environment from previous section of this document.

1.1.1 Hardware requirements

For server

Hardware	Description
Internet Connection	Wi-Fi (15 Mbps)
Computer Processor	Intel® Core ® i7 2.4GHz
Computer Memory	8GB RAM

For smartphone

Hardware	Description

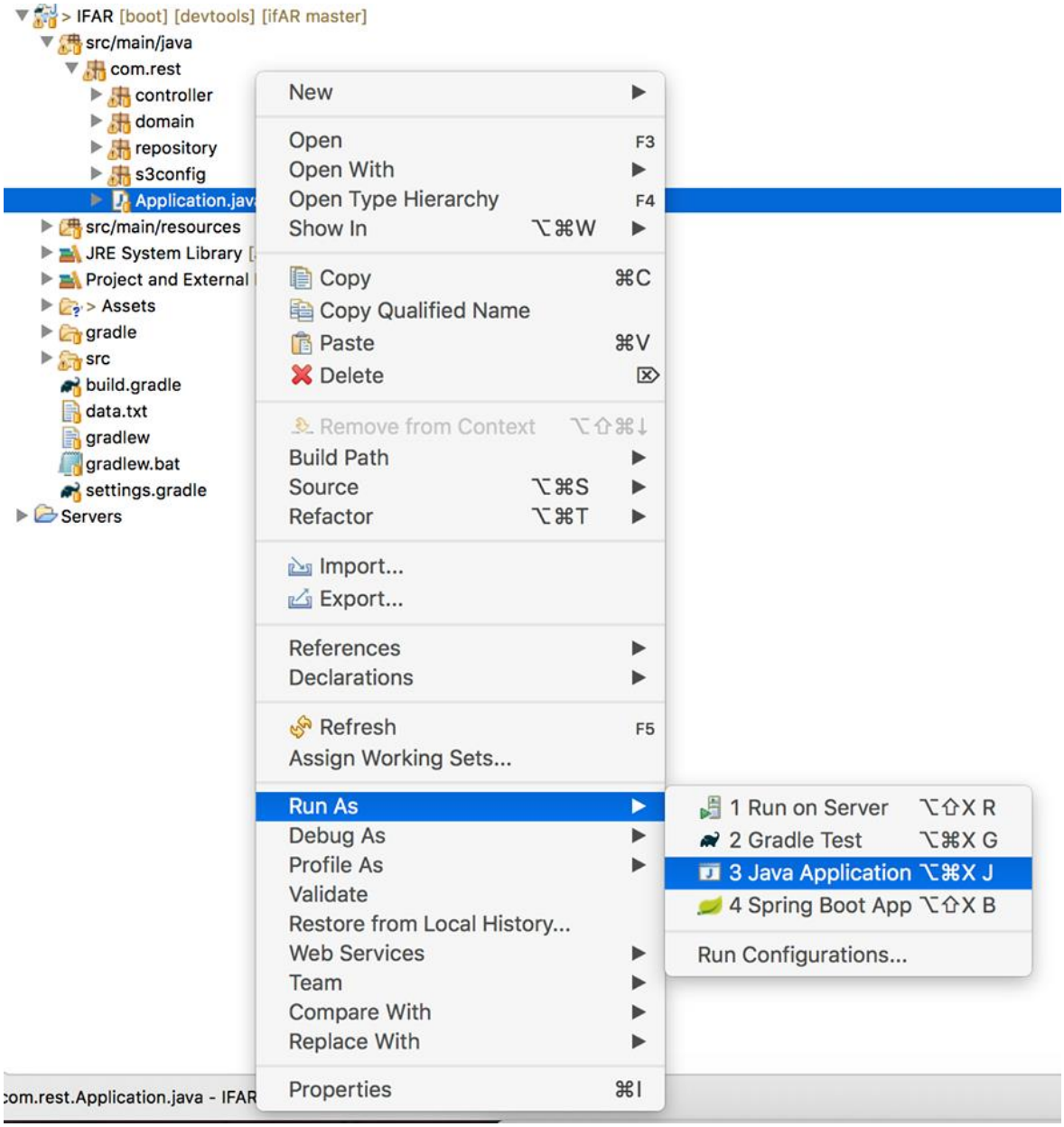
Internet Connection	Wi-Fi (10 Mbps) or LTE (10 Mbps)
Operating System	iOS 11.3
Device Version	iPhone 6S, iPhone 7 Plus

1.1.2 Software requirements

Software	Name / Version
Environment	Java EE 7 Swift 4
Modeling tool	Star UML
IDE	Eclipse Neon.3 Release (4.6.3) Xcode 9.3 MySQL Workbench 6.3.9
DBMS	MySQL 5.6.30
Source control	GitKraken Pro (3.5.1)
Web browser	Chrome 42 or later
Mobile OS	iOS 11.3 or later

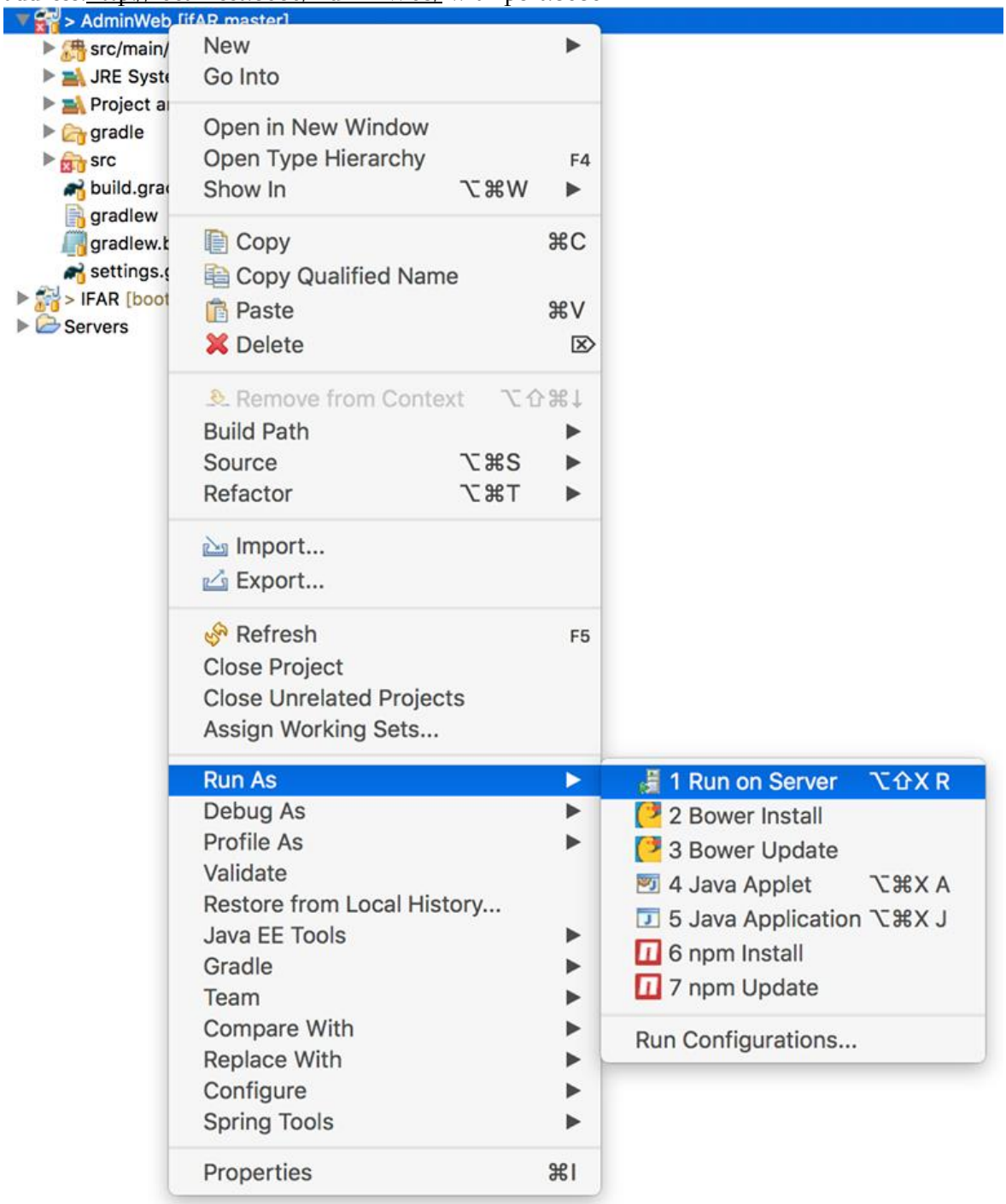
1.3 Web Server Deployment Process

Open Eclipse application. Right click on project web server “IFAR”, choose “Run As” -> “Java Application”. Connect to Web server via host address: http://localhost:62231/IFAR with port 62231



1.5 Web Application Deployment Process

Open Eclipse application. Right click on web application project “AdminWeb”, choose “Run as” -> “Run on Server”. Connect to Web Application via host address:http://localhost:8080/AdminWeb/ with port:8080



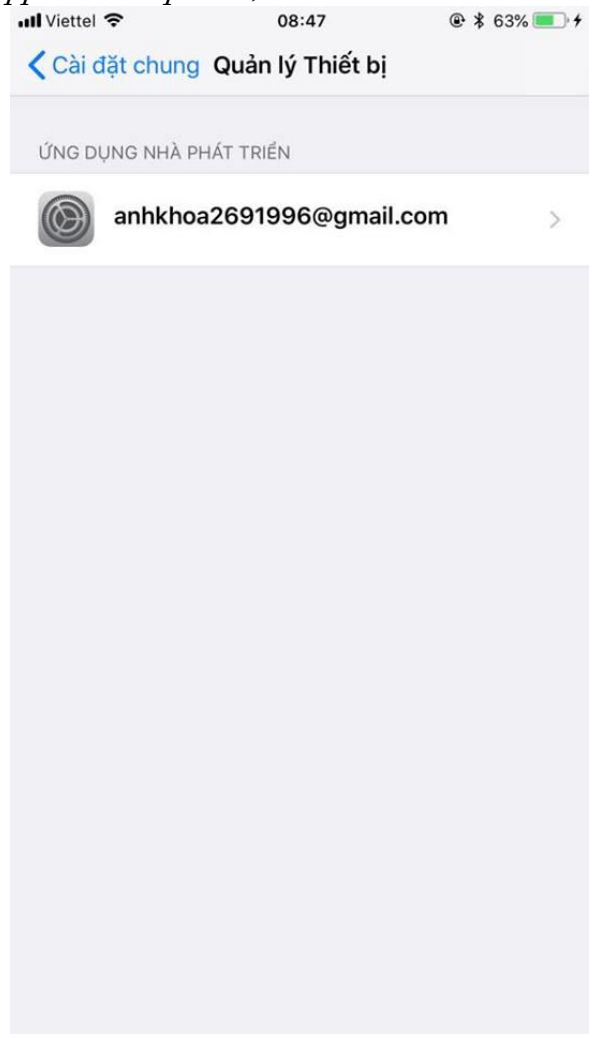
1.7 Mobile Application Deployment Process

Open "Cài đặt" to go to setting

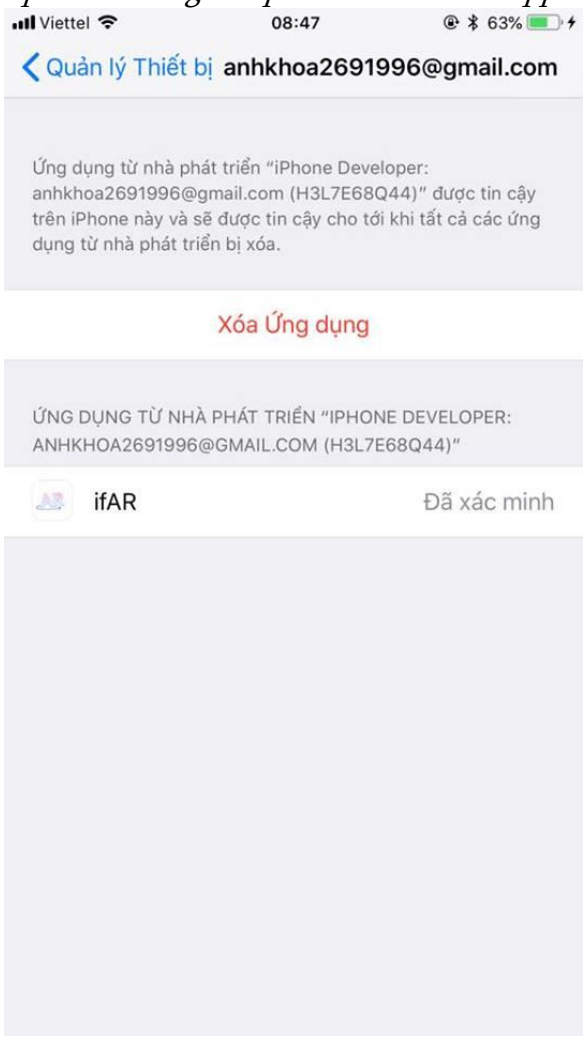
Choose "Cài đặt chung"

Choose "Quản lý thiết bị"

Choose your Apple Developer ID, mine is "anhkhoa2691996@gmail.com"



Then tap "Trust" to grant permission to this application.



2. User Guide

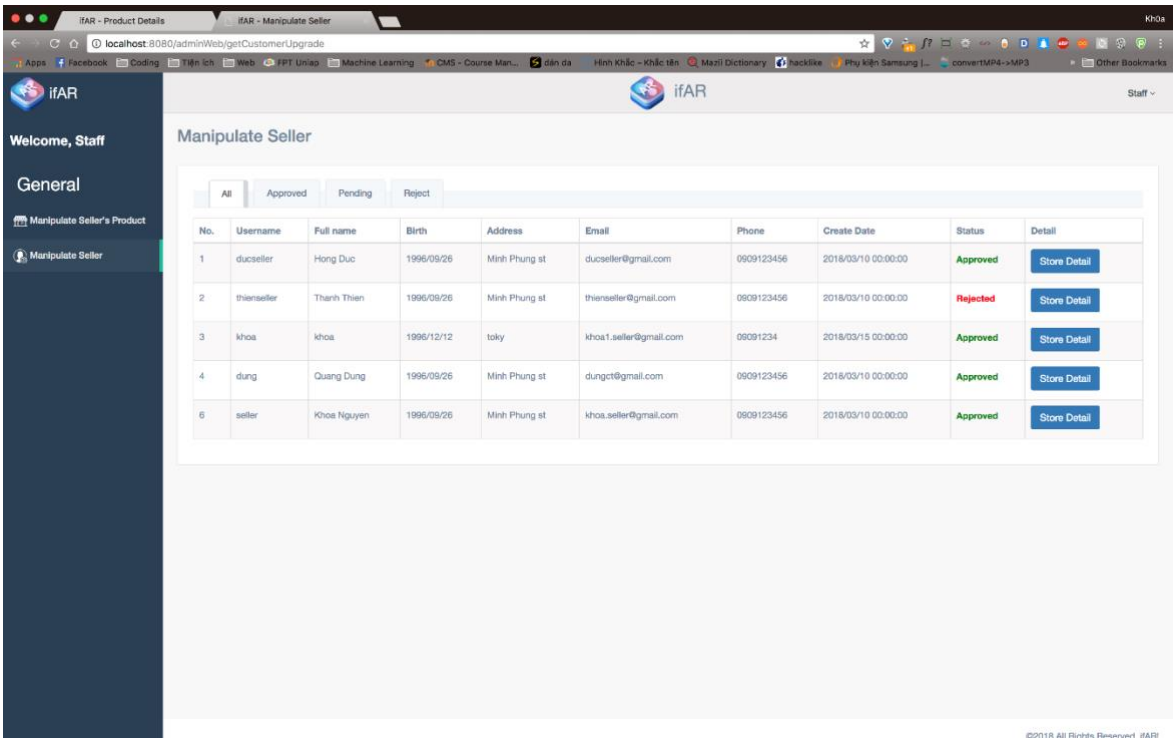
2.1 Web Application

2.1.1 Staff & Designer

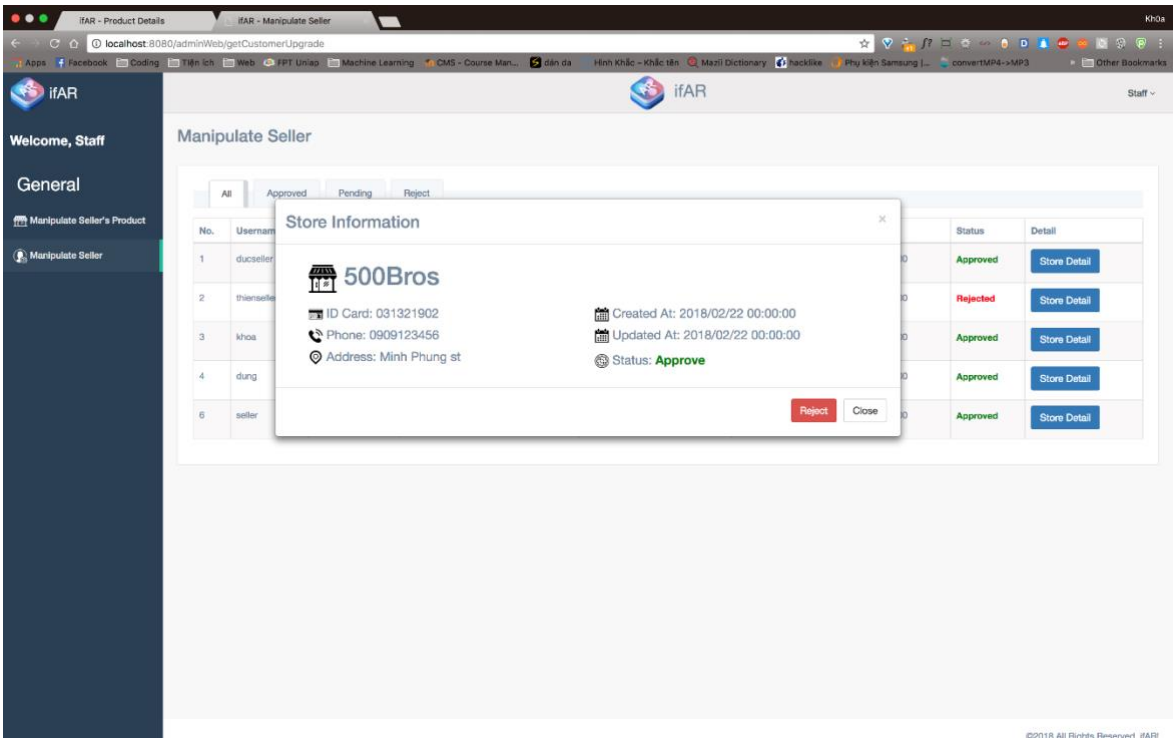
2.1.1.1 Manipulate Seller

Login to Web Application via http://localhost:8080/IFAR, then choose menu "Manipulate Seller" on left edge. Seller's information is listed in table below. This page has 4 tabs: All, Approved, Pending and Rejected which display corresponding status of the seller.

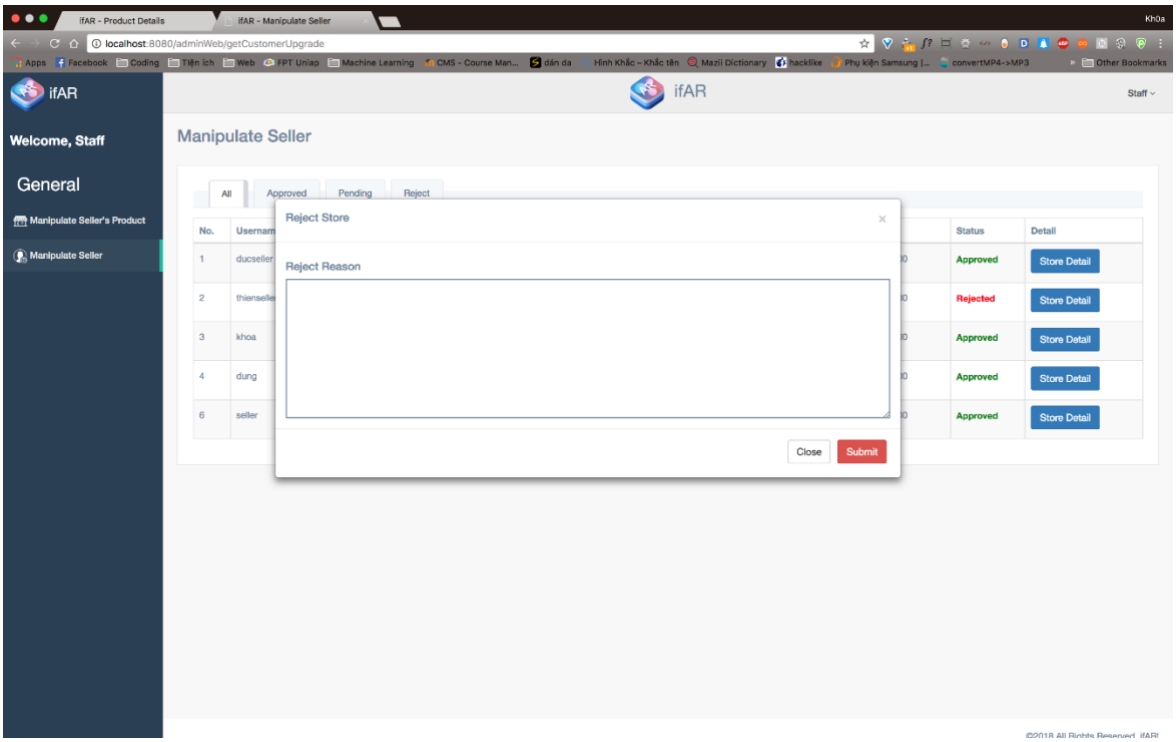
Choose "Store Detail" at the end of each record to see full information of seller and his/her store.



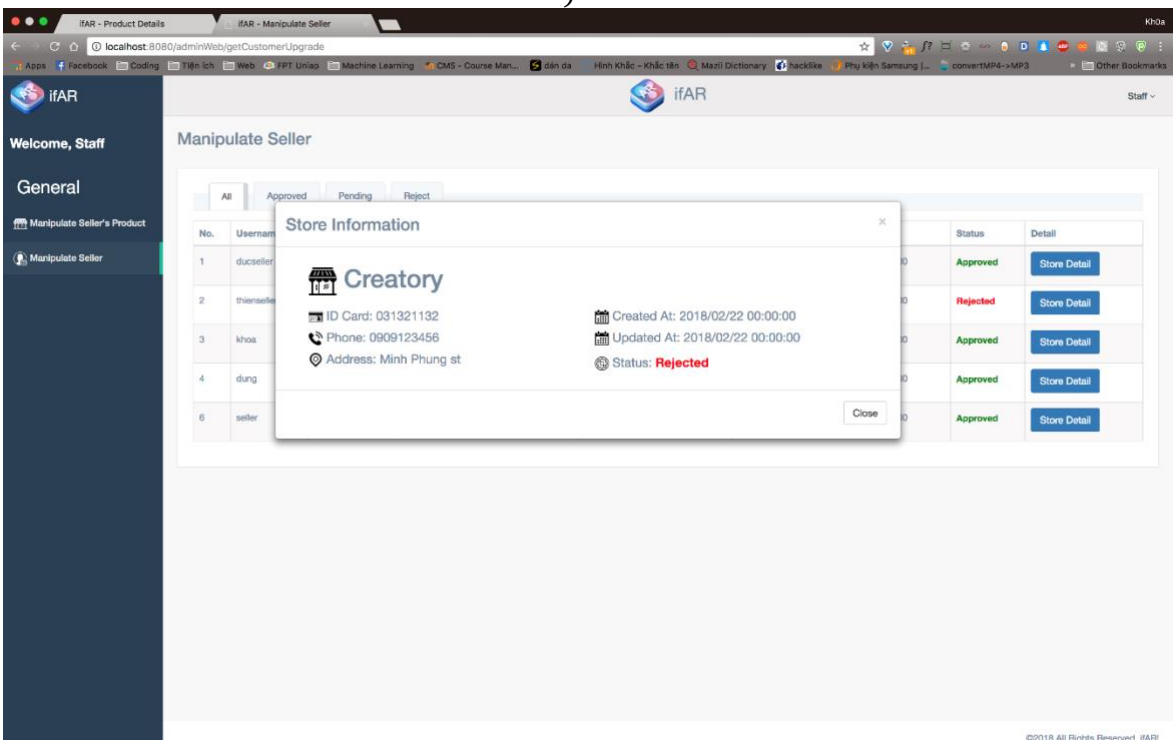
This pop-up will show all information of seller's store, staff can "Reject" or "Approve" this store.



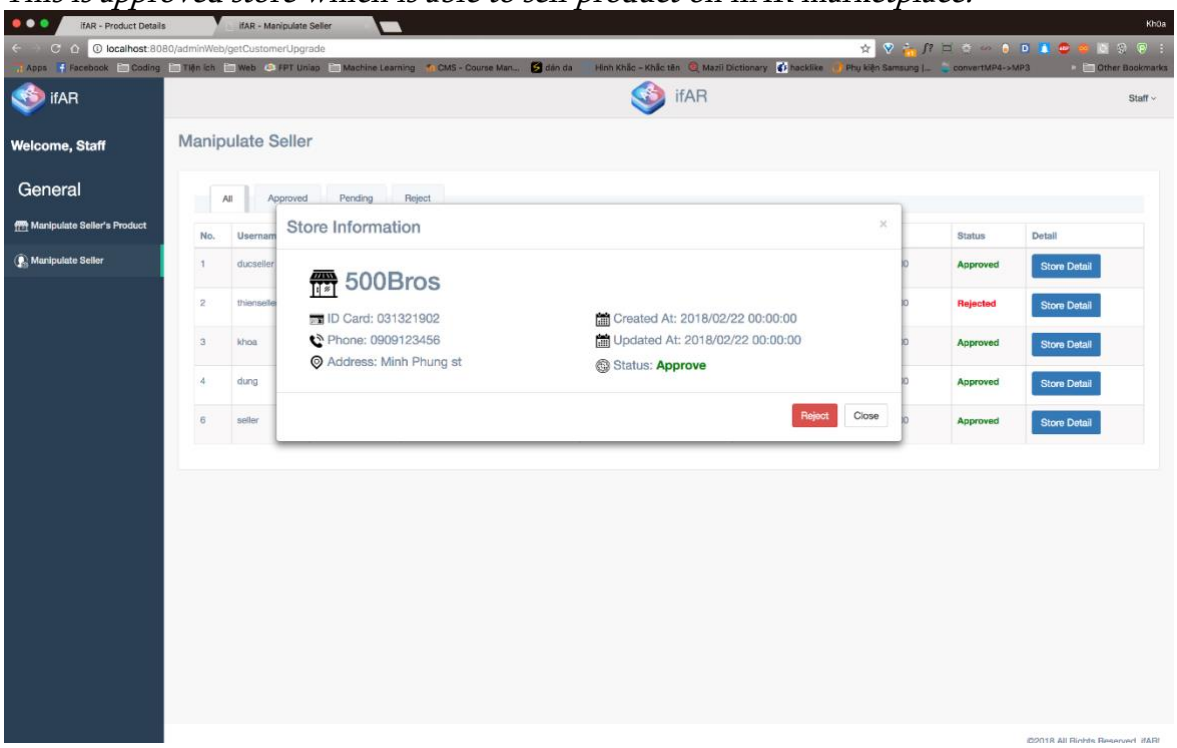
If "Reject" button is chosen, there's a new pop-up which reject reason is filled in and send to seller.



This is rejected store.



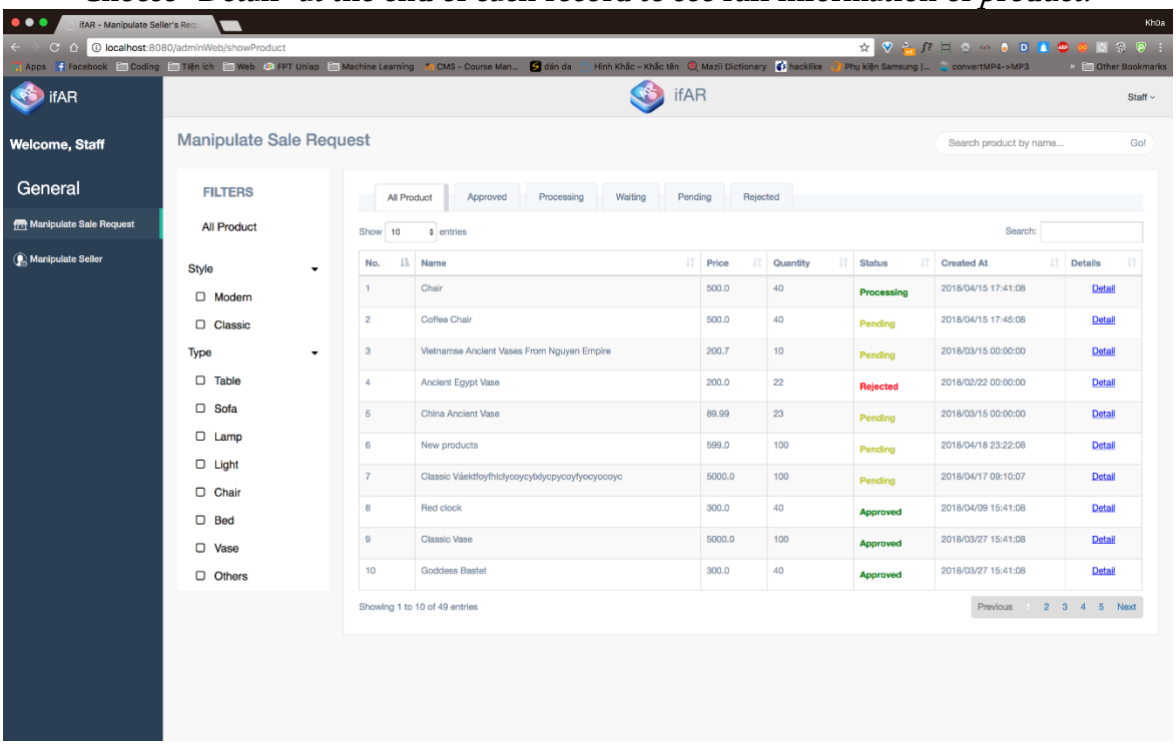
This is approved store which is able to sell product on ifAR marketplace.



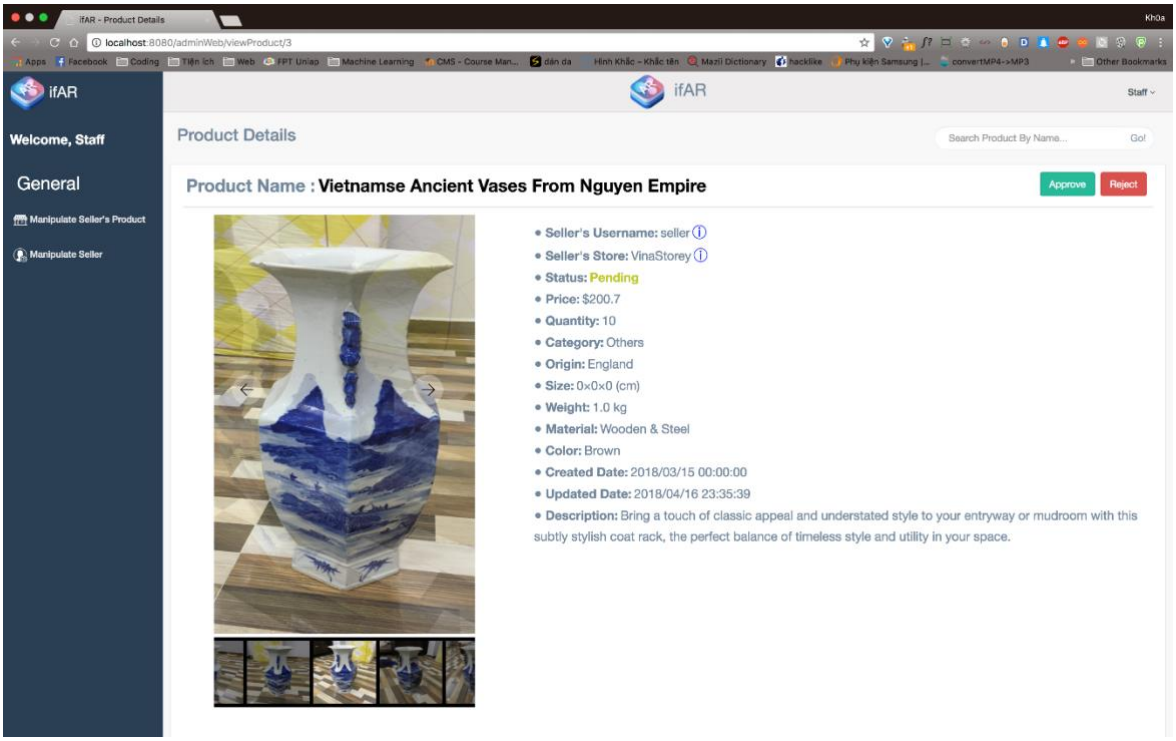
2.1.1.2 Manipulate Sale Request

Login to Web Application via http://localhost:8080/IFAR, then choose menu "Manipulate Sale Product" on left edge. Product's information is listed in table below. This page has 4 tabs: All, Approved, Processing, Waiting, Pending and Rejected which display corresponding status of the product.

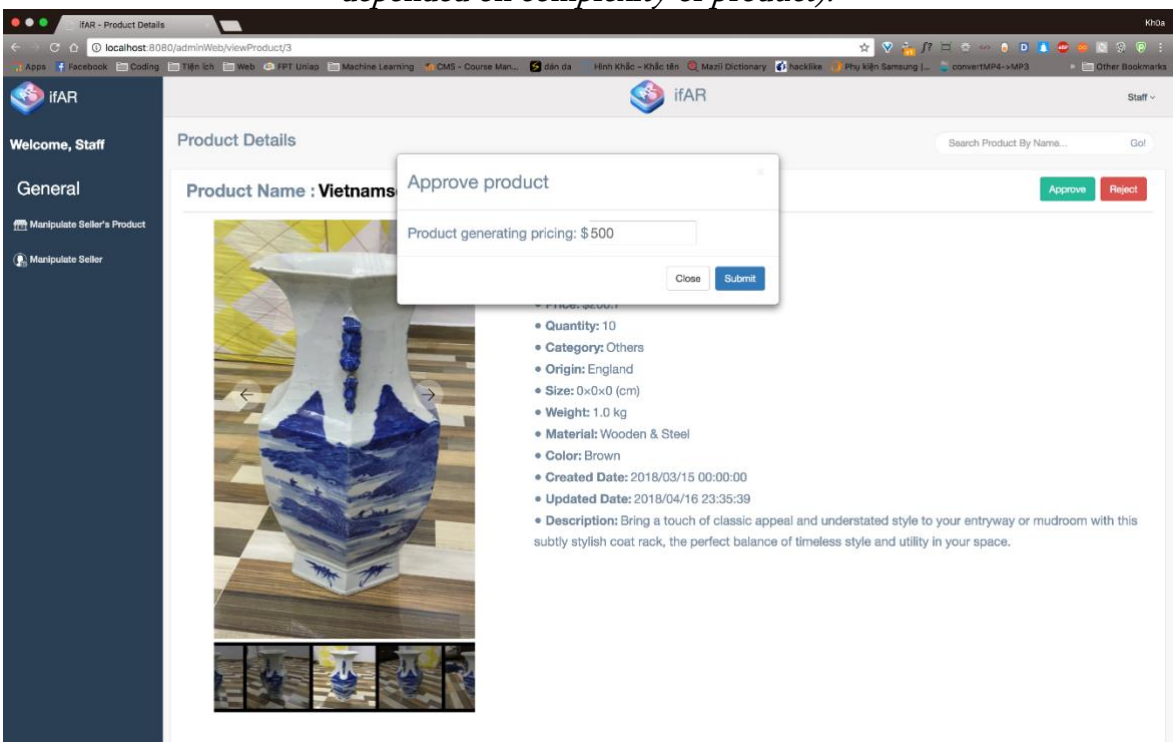
Choose "Detail" at the end of each record to see full information of product.



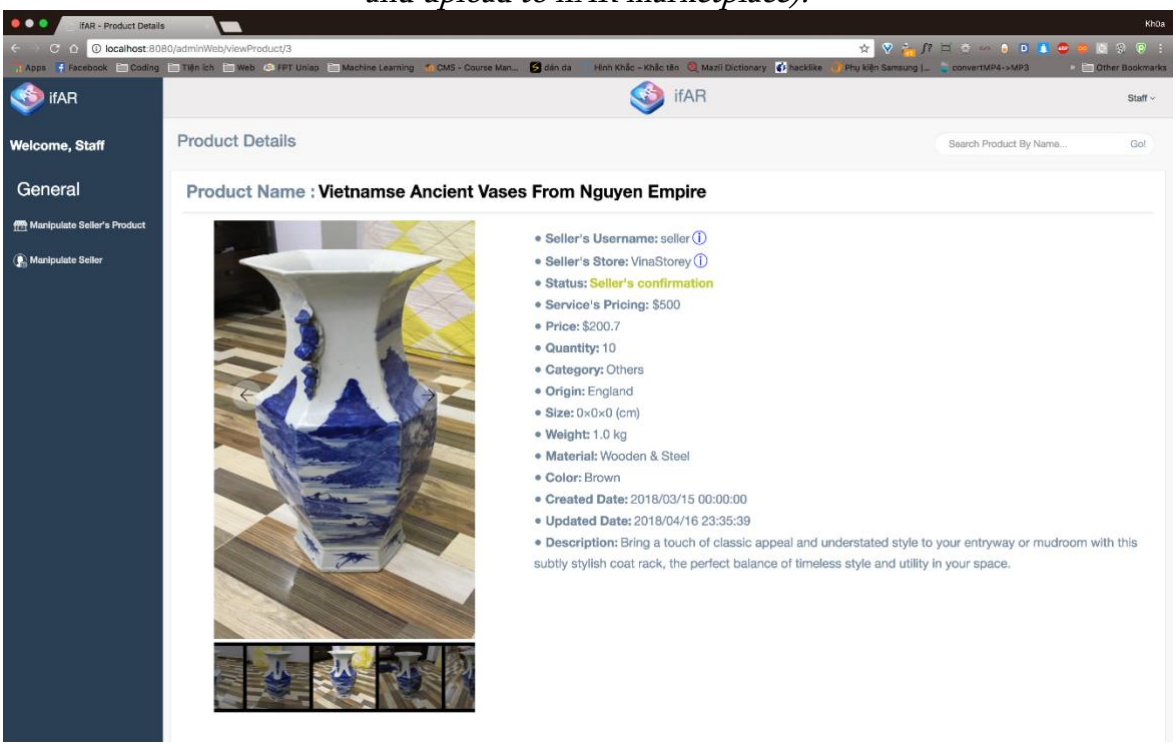
Product's detail is shown below



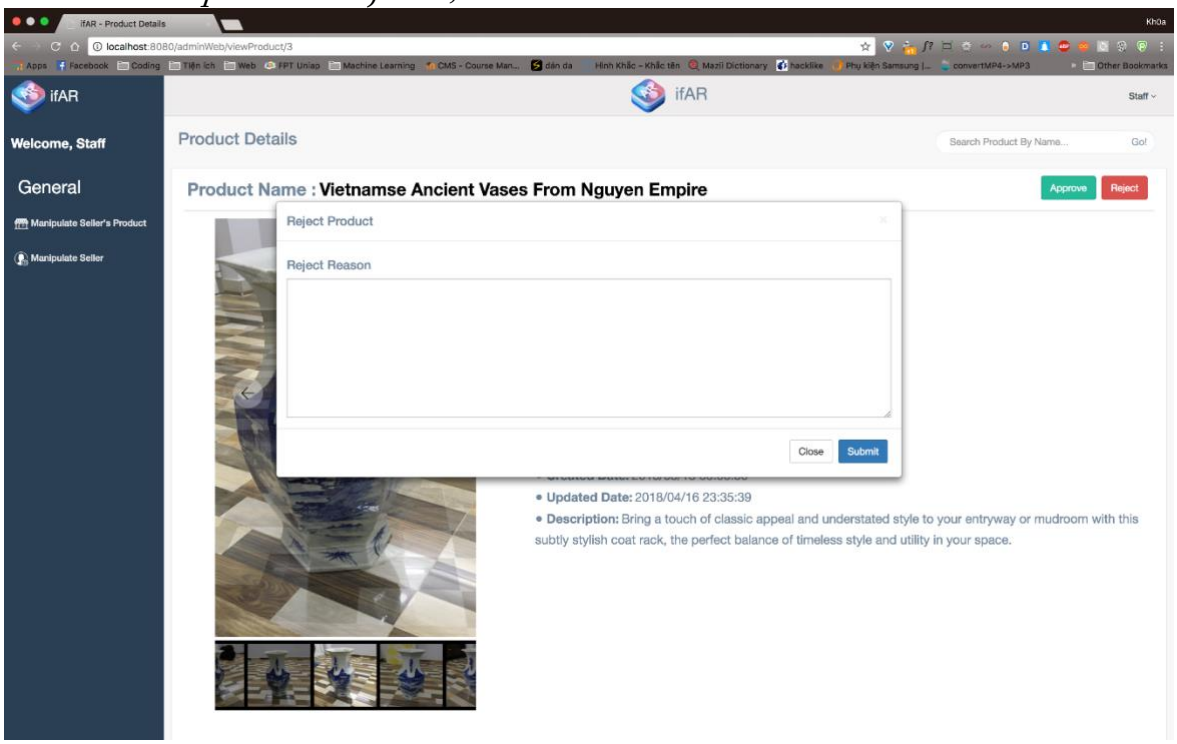
If product's status is "Pending", Staff can "Approve" or "Reject" this product. If product is approved, Staff will announce pricing service of this product (pricing is depended on complexity of product).



After being approved, this product change status from "Pending" to "Waiting for Seller's confirmation" that means seller will approve the pricing service or not. If seller approves the pricing, product will be started processes (generate 3D model from images and upload to ifAR marketplace).



If product is rejected, Staff will announce the reason for seller.

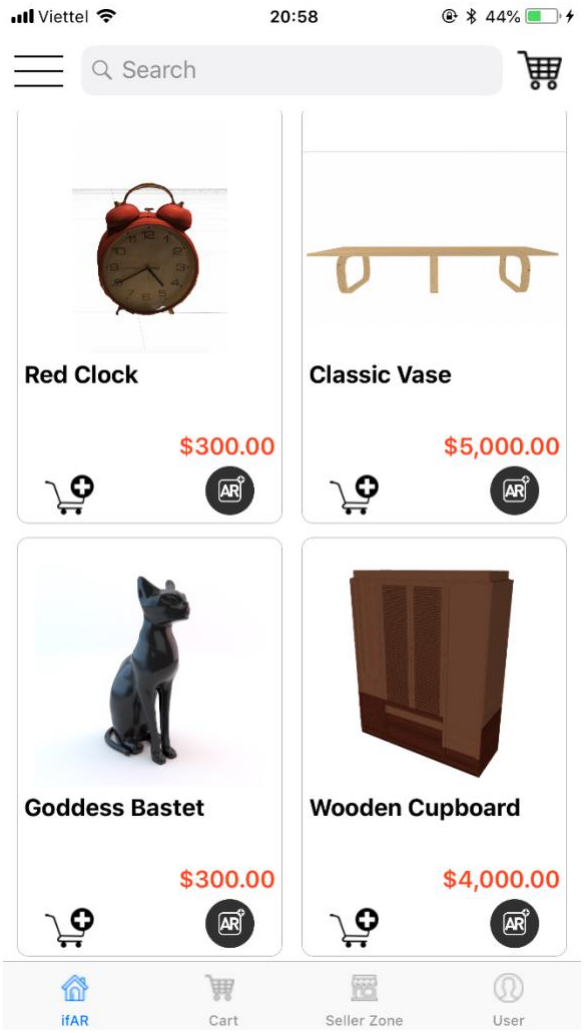


2.2 Mobile Application

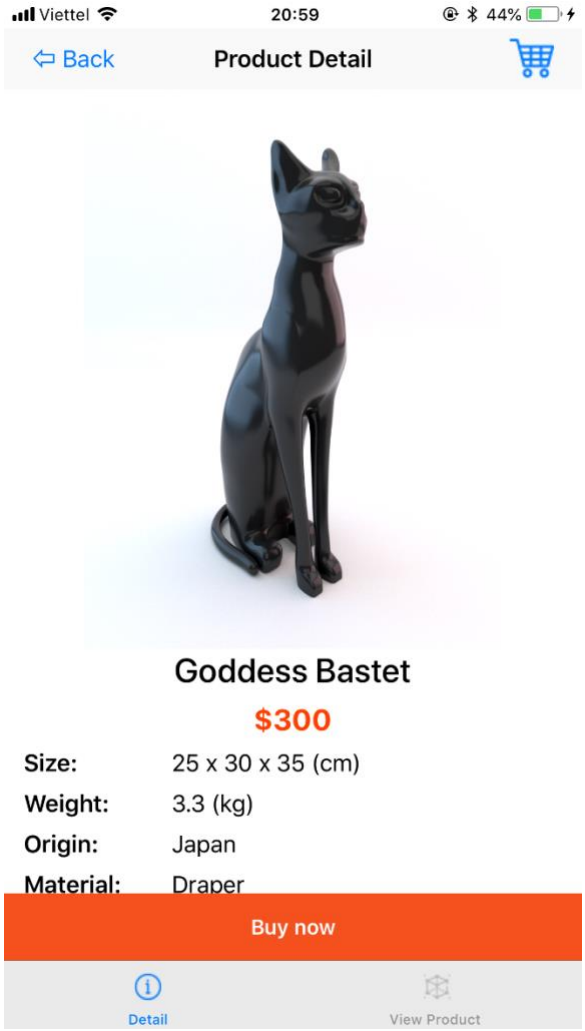
2.2.1 Customer

2.2.1.1 Simulate Product in AR view and 3D view

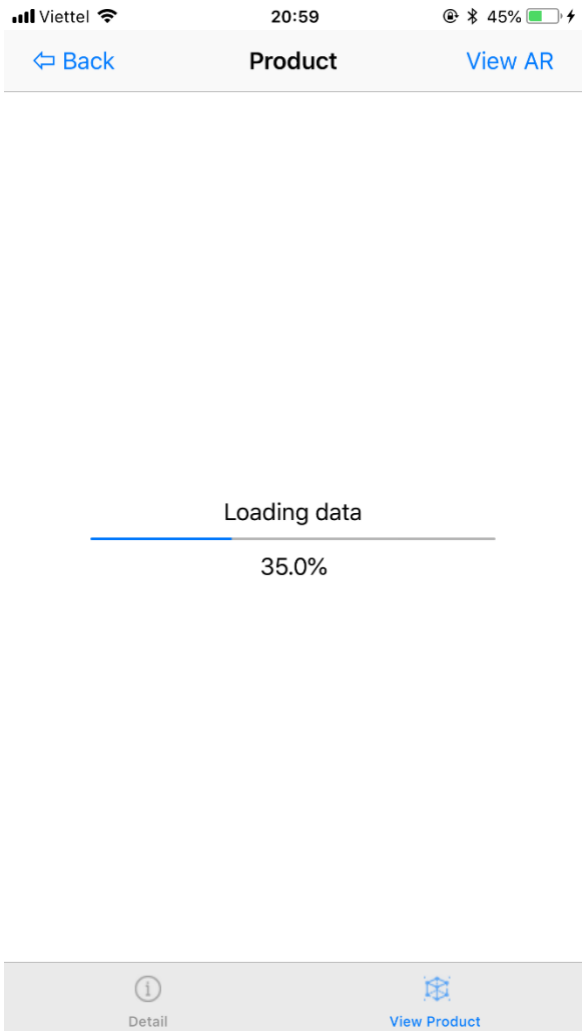
Choose product "Goddess Bastest"



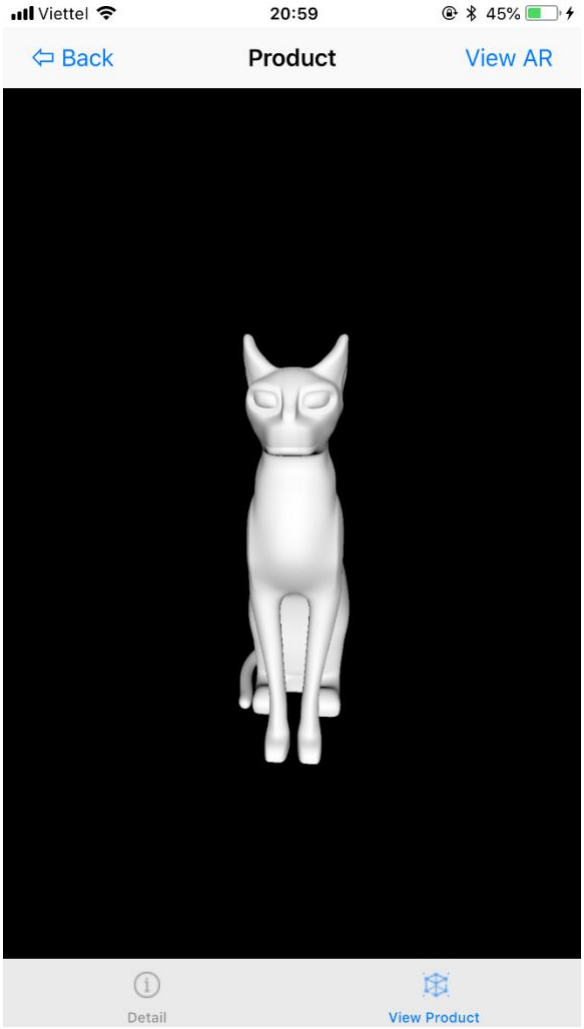
Here's information of product. Choose "View product" to view product in 3D view.



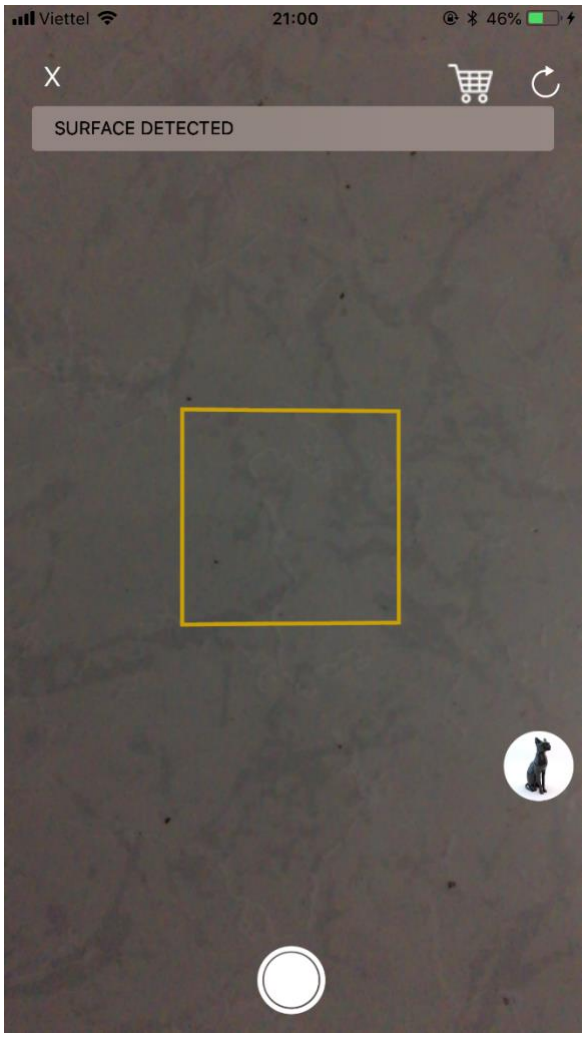
Wait for download 3D model process finish.



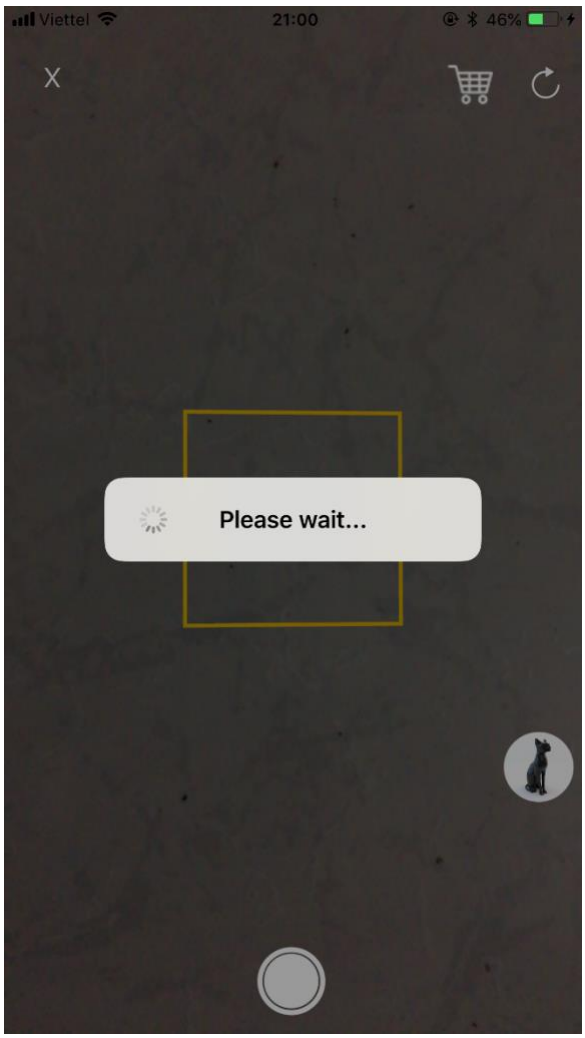
3D model is loaded. You can rotate it in any direction.
You can also view this product in AR view by choose "View AR" in top right corner.



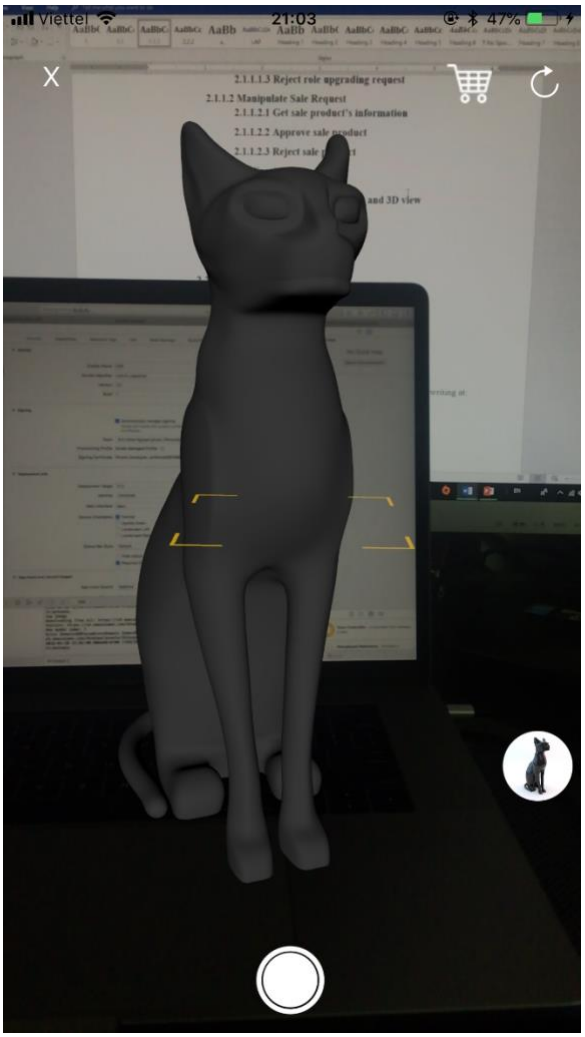
Here's AR view, move camera near to surface to detect the surface. When yellow square appears, you can tap on it to place product.



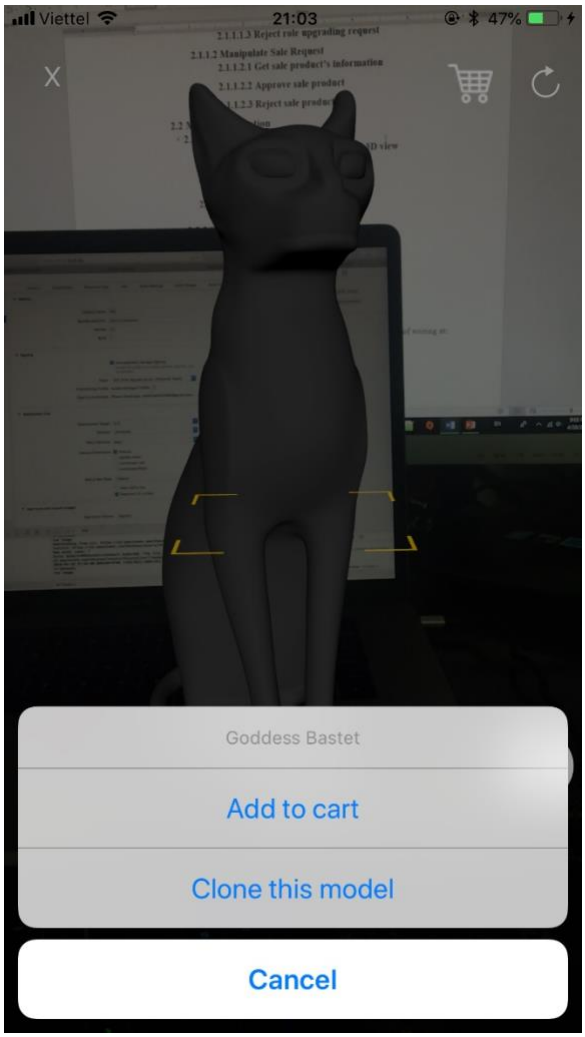
Wait for downloading model from server



By tap the icon of product on right edge, you can perform some action such as: "Add to cart", "Clone this model".

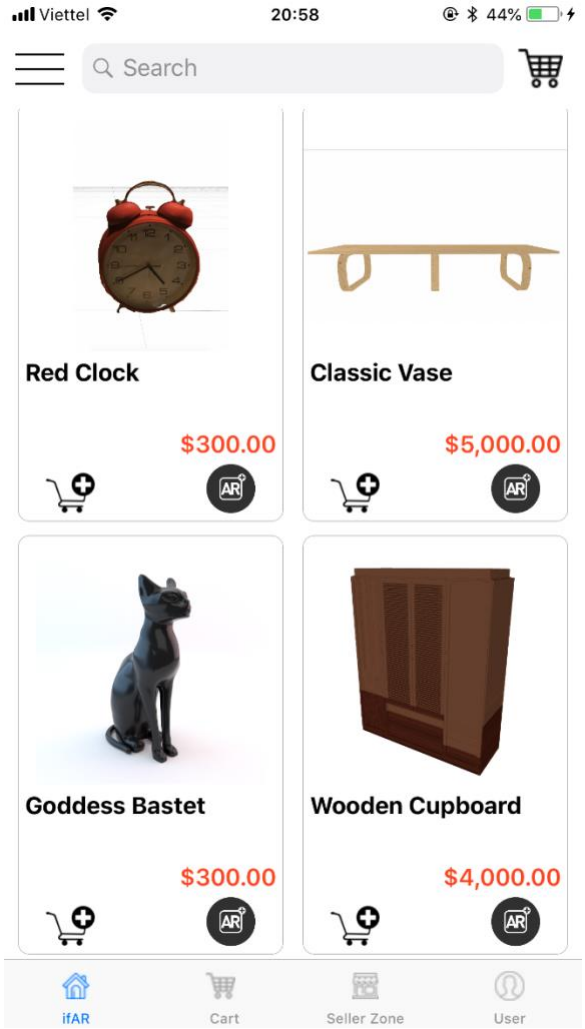


Choose suitable action, "Add to cart": add this product to cart, "Clone this model": add one more 3D model to AR View.

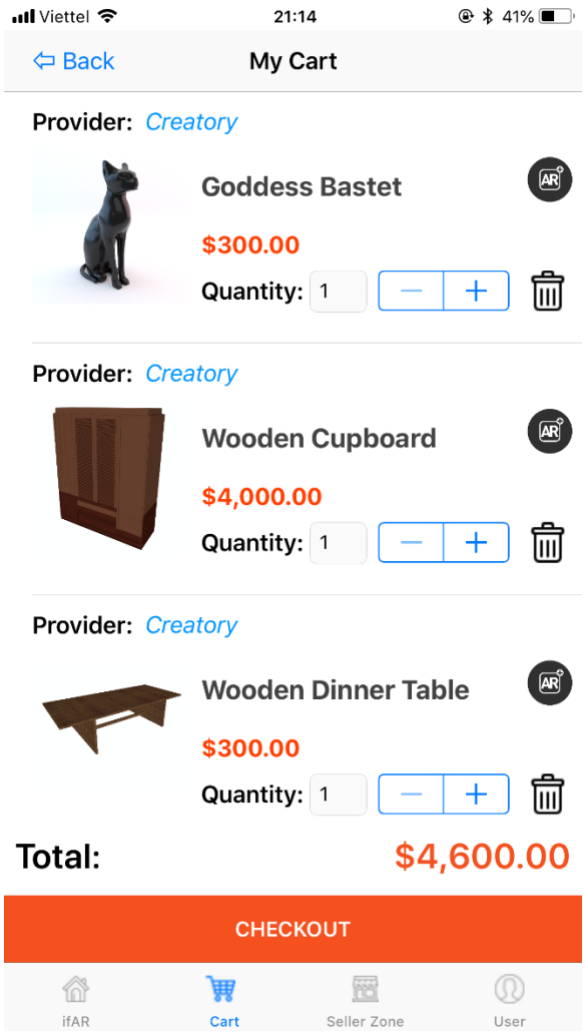


2.2.1.2 Go Online Shopping

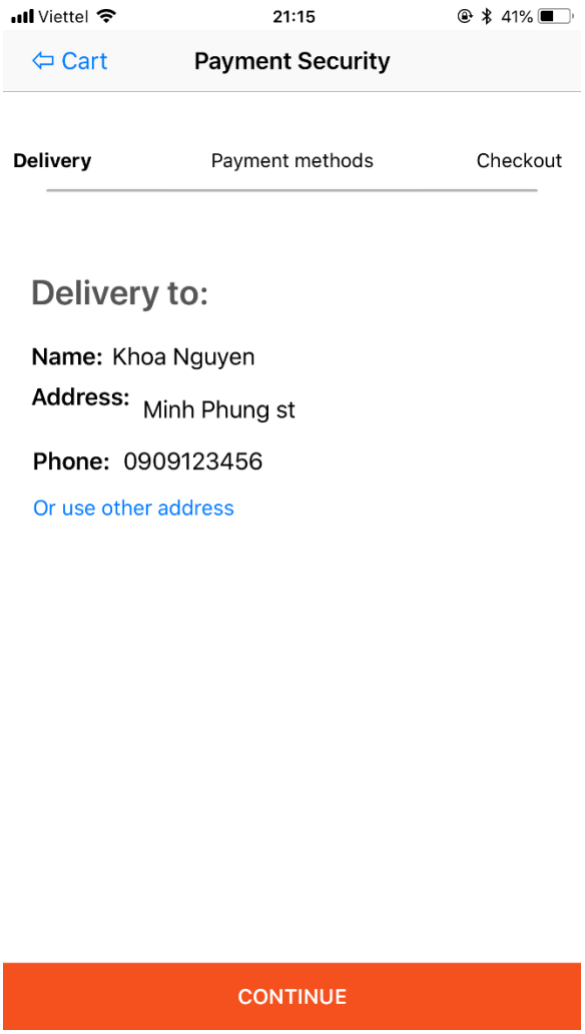
Add some products to cart, by tap the icon "add" on bottom left corner of product cell. Then click Cart icon on top right corner of screen to or on tab bar at bottom to view cart.



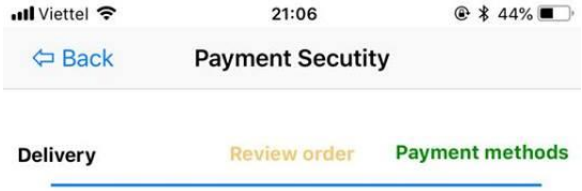
Choose "Checkout" to check out your cart



Here's delivery address in your profile, you can also change address by choose "Or use other address".
Choose "Continue" to move to next section.

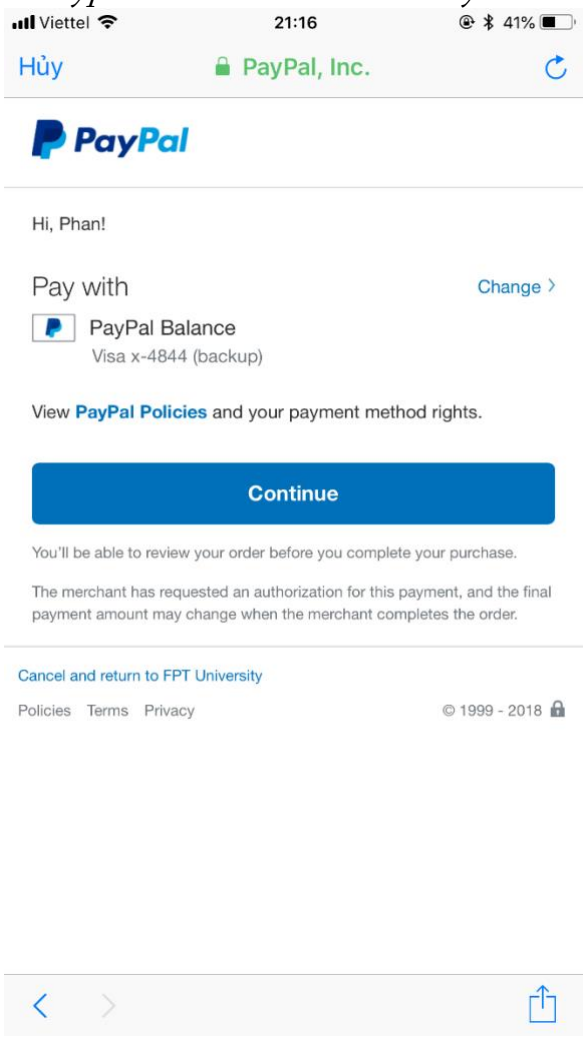


We provide 2 methods of payment: "Cash on Delivery" and "PayPal". In this example, we'll use PayPal

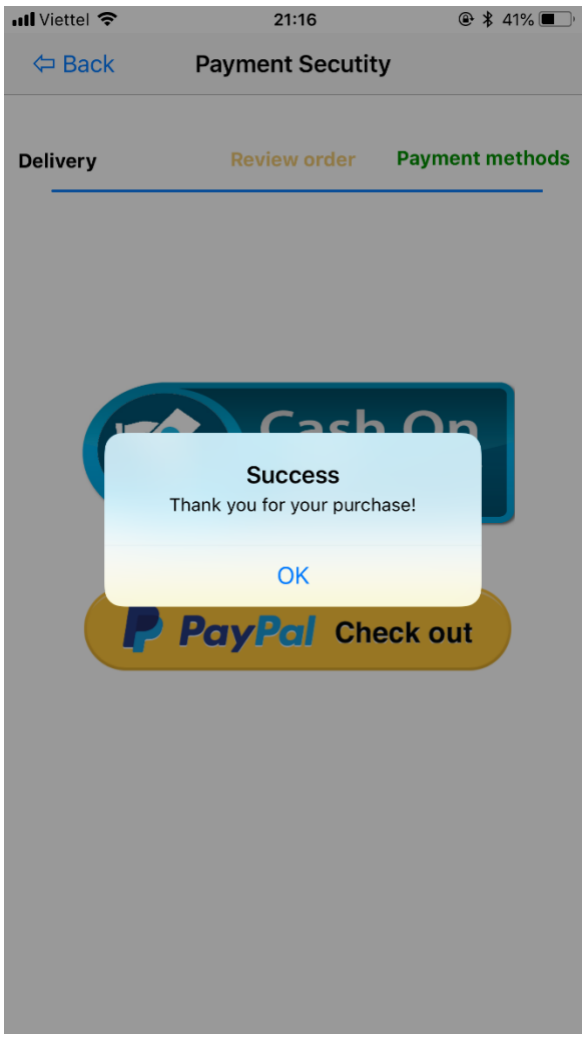


.

Login to your Paypal Account and checkout by choose "Continue"

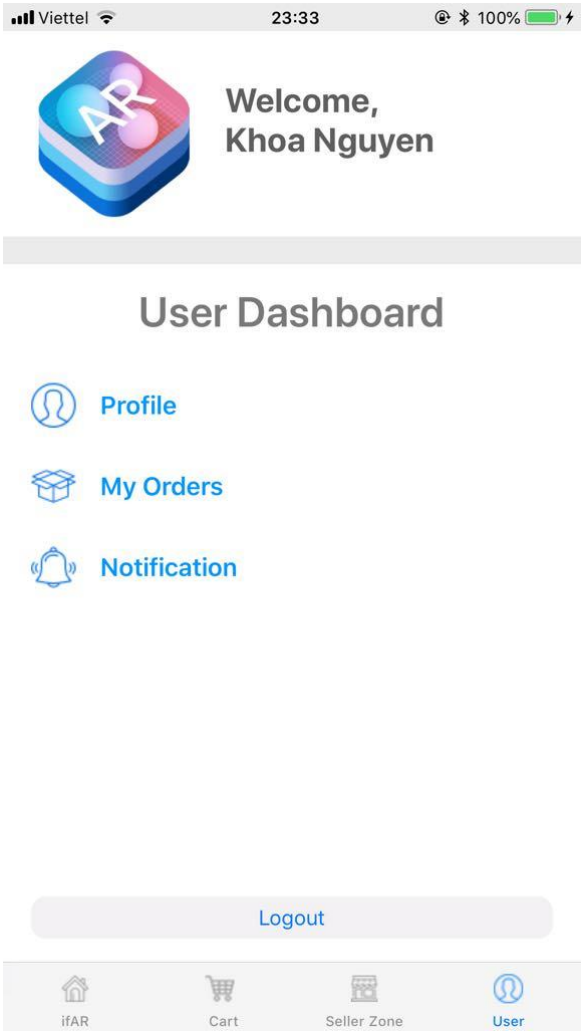


Check out successfully.

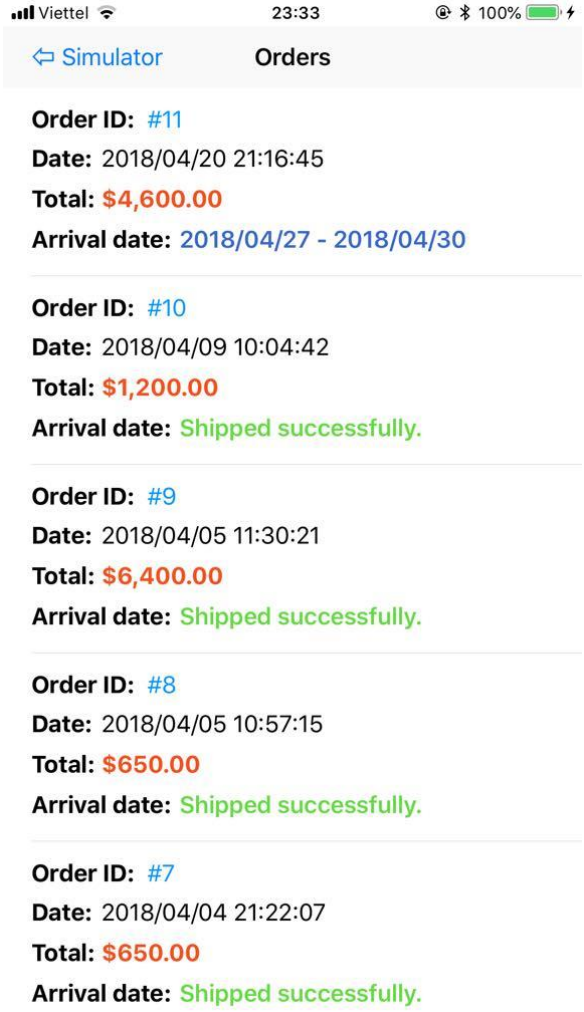


2.2.1.3 Track Orders

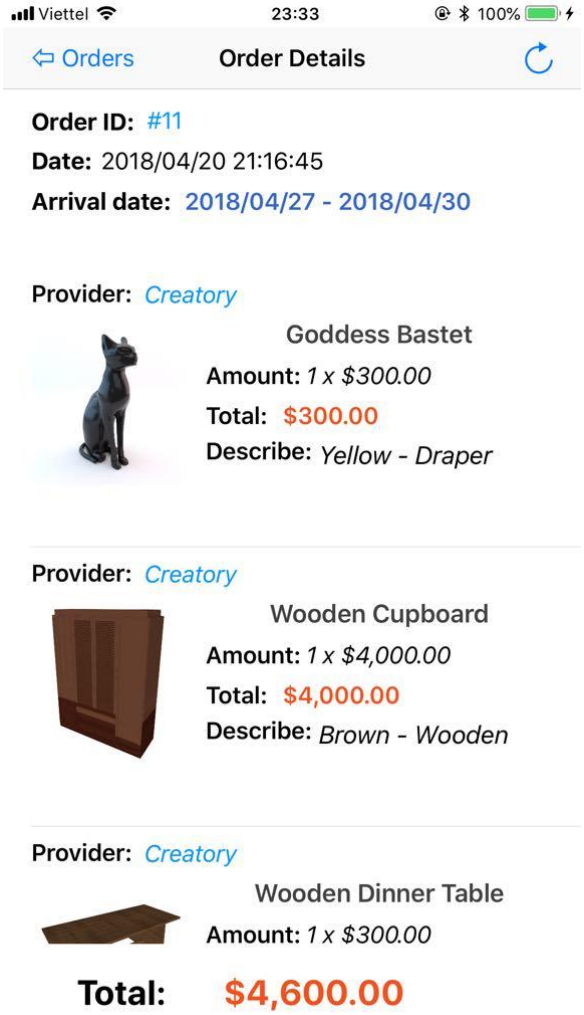
Move to User tab on Tab bar at bottom. Then choose "My Orders" to view all your orders.



By choose any order cell, you can view order details of order.



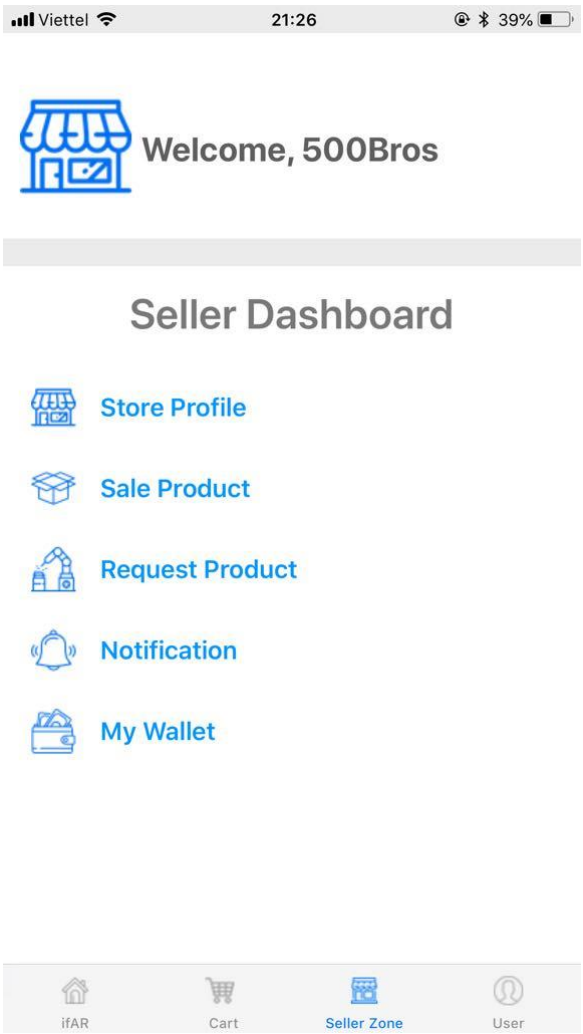
Review order details' information



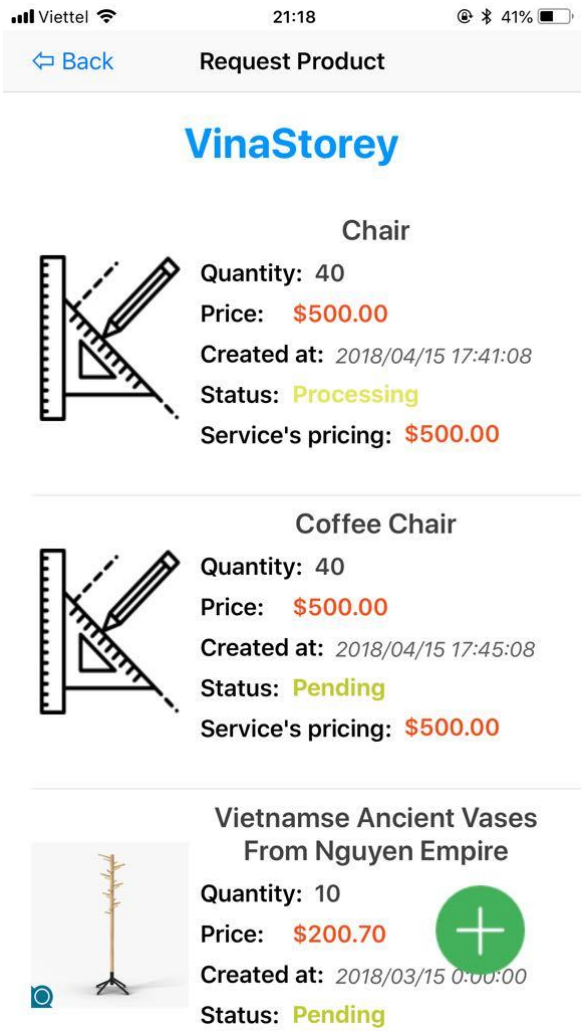
2.2.2 Seller

2.2.2.1 Request to sell product

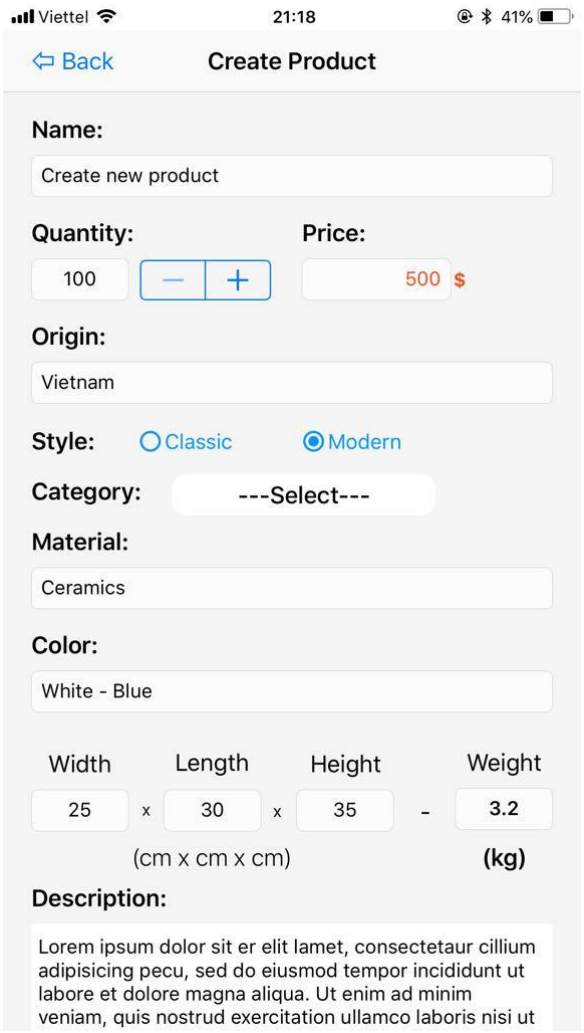
Choose "Seller Zone" tab on tab bar at bottom. Then choose "Requset Product" to request new sale product.



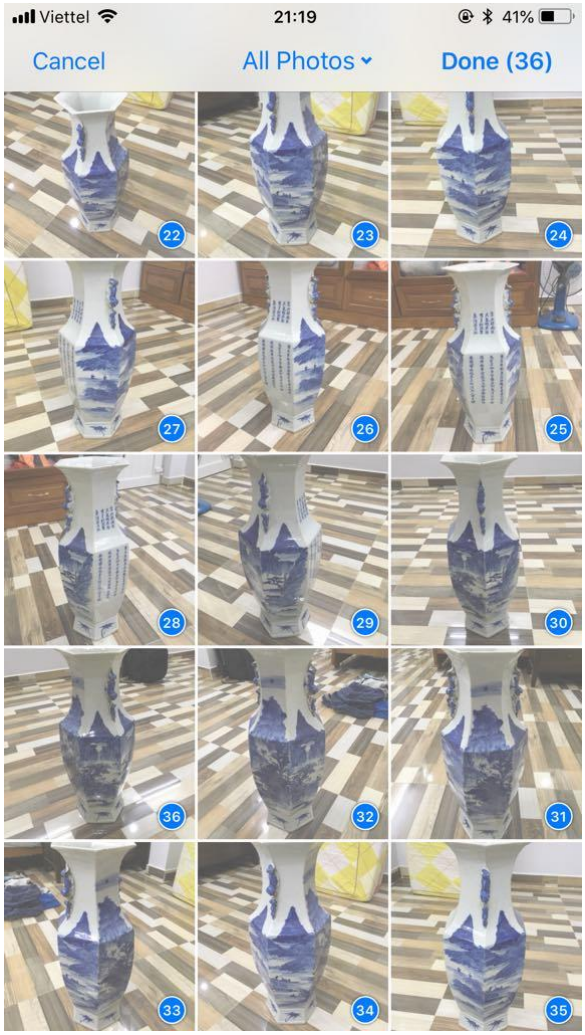
Tap "+" icon to create new request



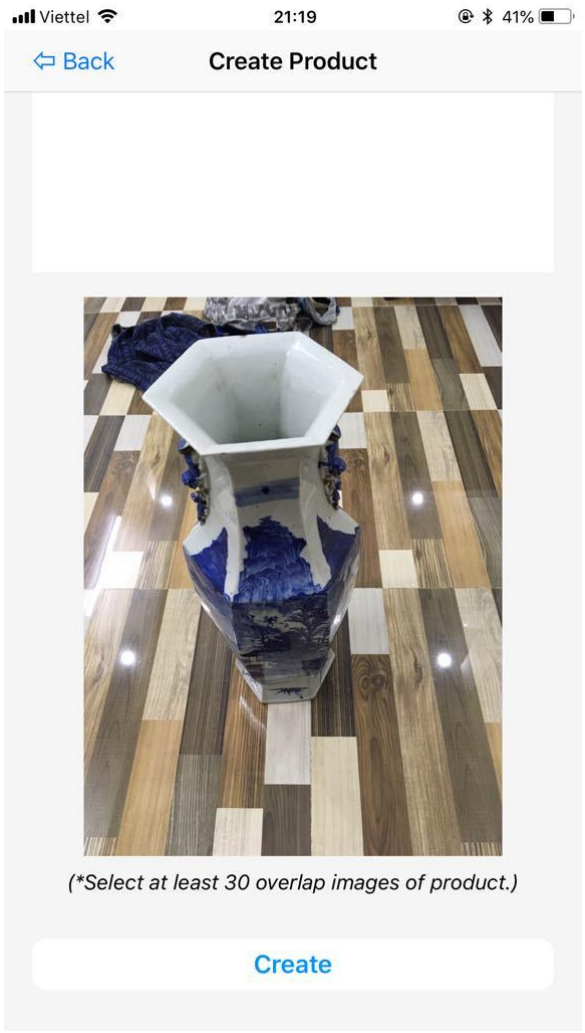
Fill information of your product



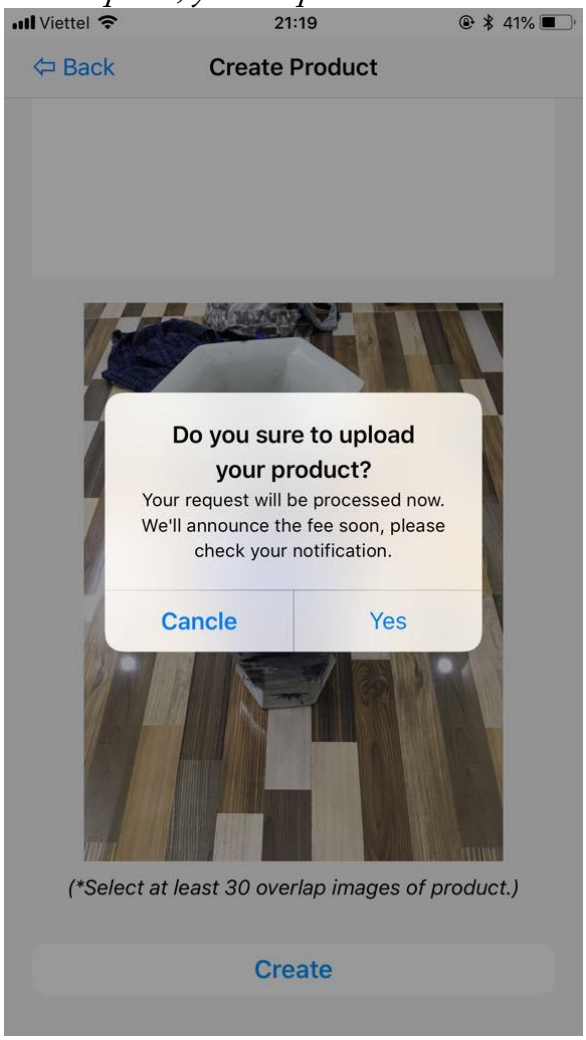
And choose images of the product



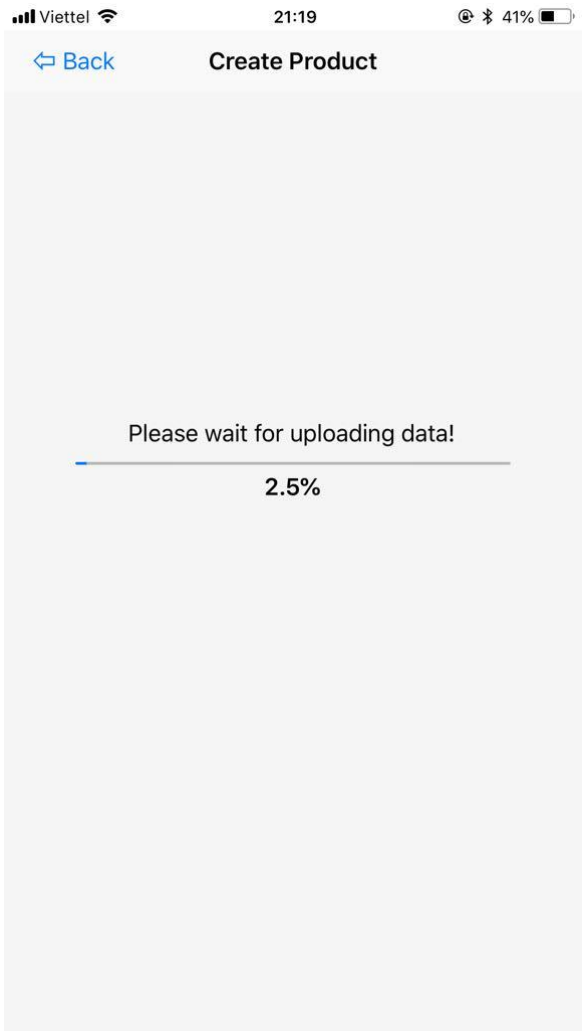
Choose "Create" to send the request



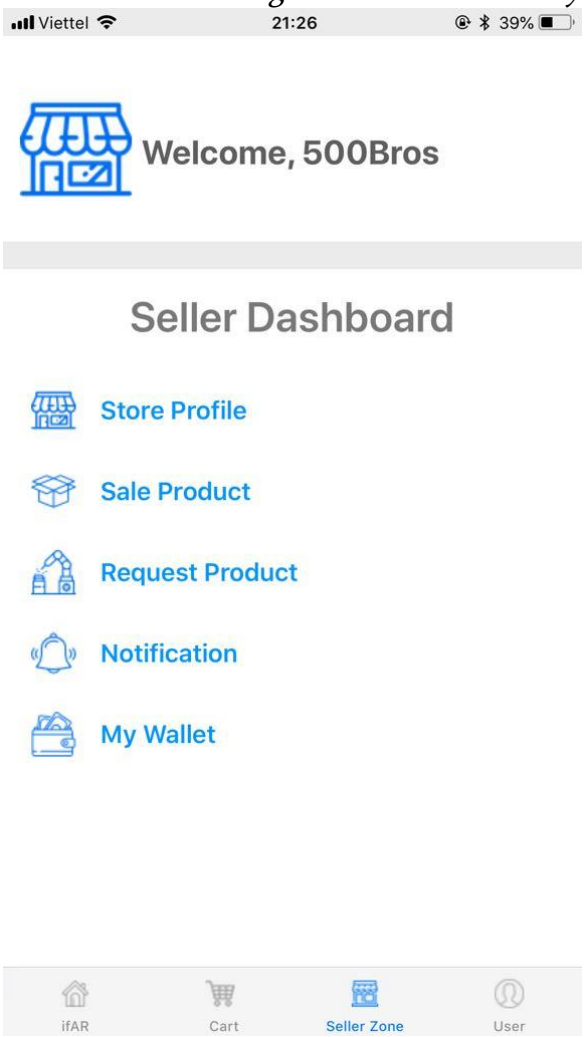
By choose "Yes" option, your request will be sent to our system.



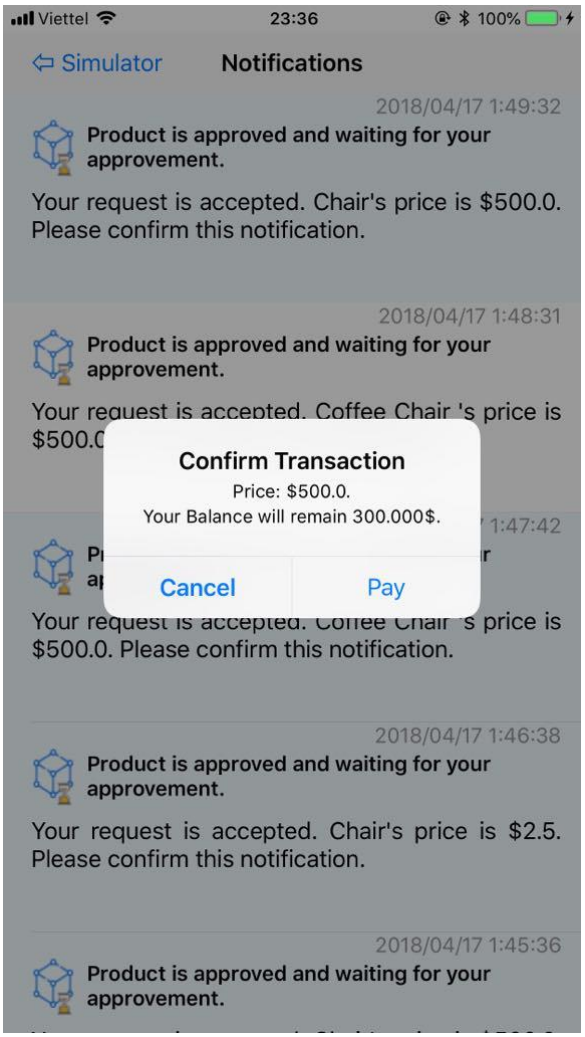
Wait for uploading process finish.



Choose "Notification" to get notification from system

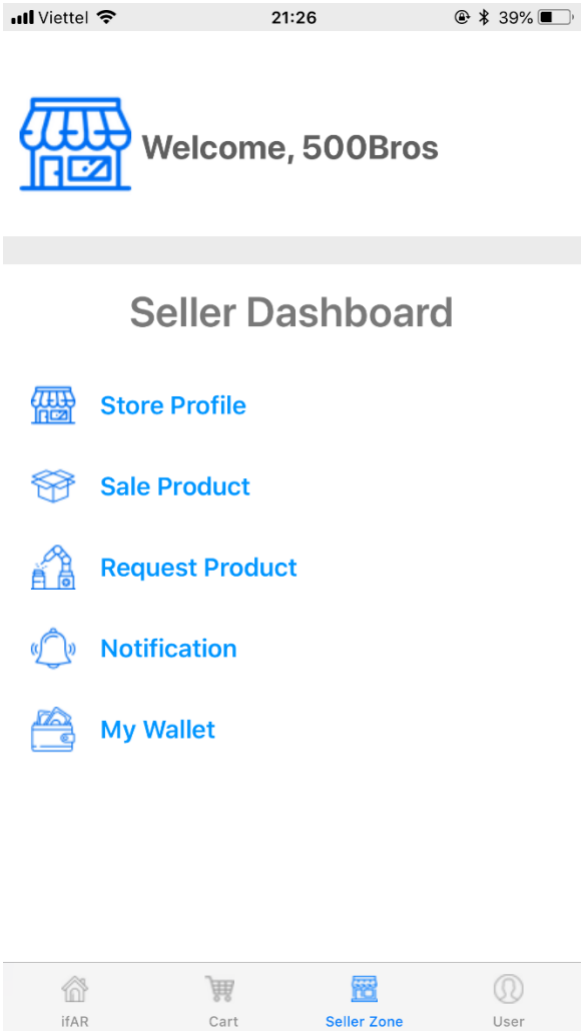


Confirm the notification from system and pay service pricing to start create sale product on marketplace by choose "Pay" option.

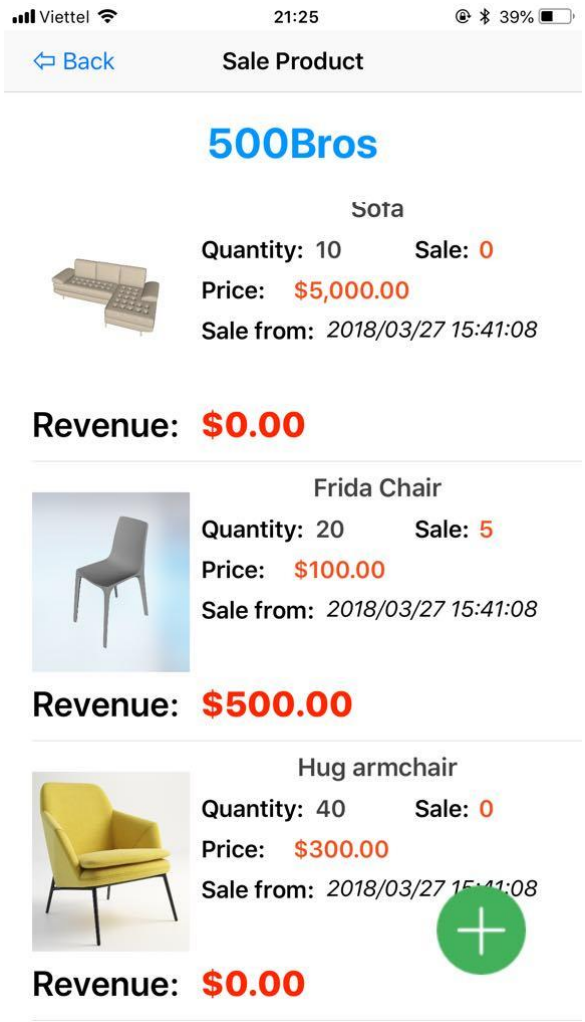


2.2.2.2 Manipulate products

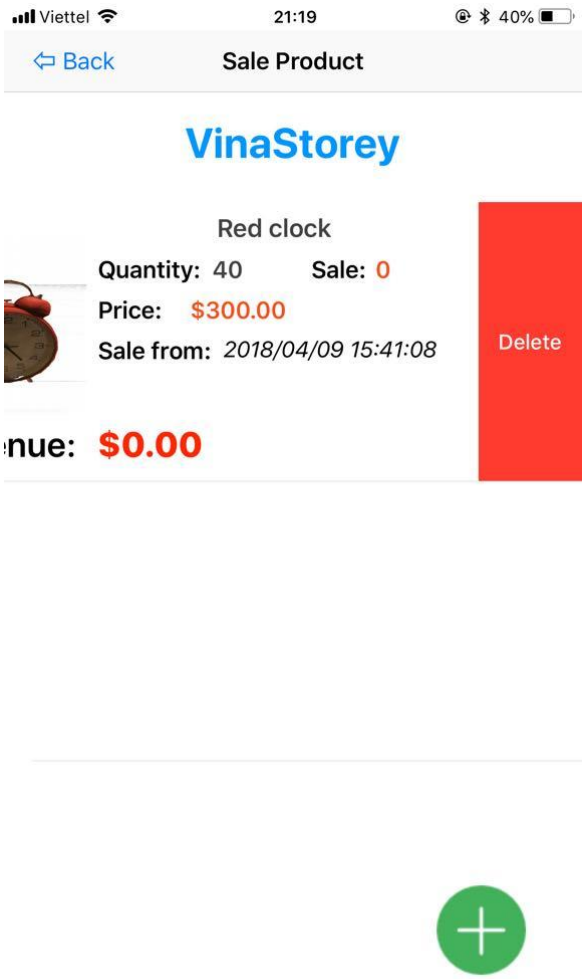
Choose "Seller Zone" on tab bar at bottom, then choosing "Sale product" to manipulate your products which is sold on marketplace.



Here's all products are now selling on marketplace. By choose any product, you can view product's detail.

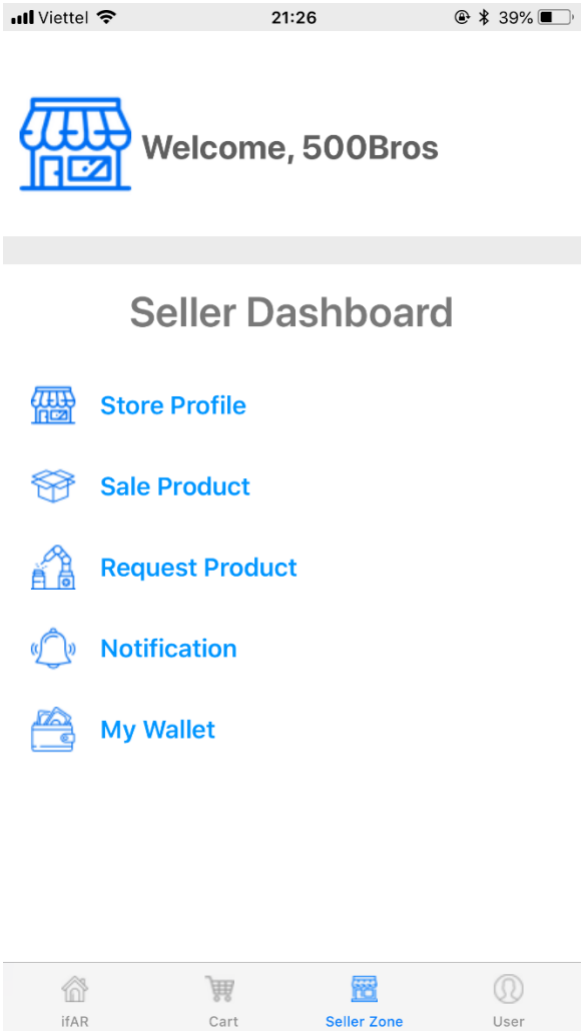


Swipe left on product to remove it from marketplace.

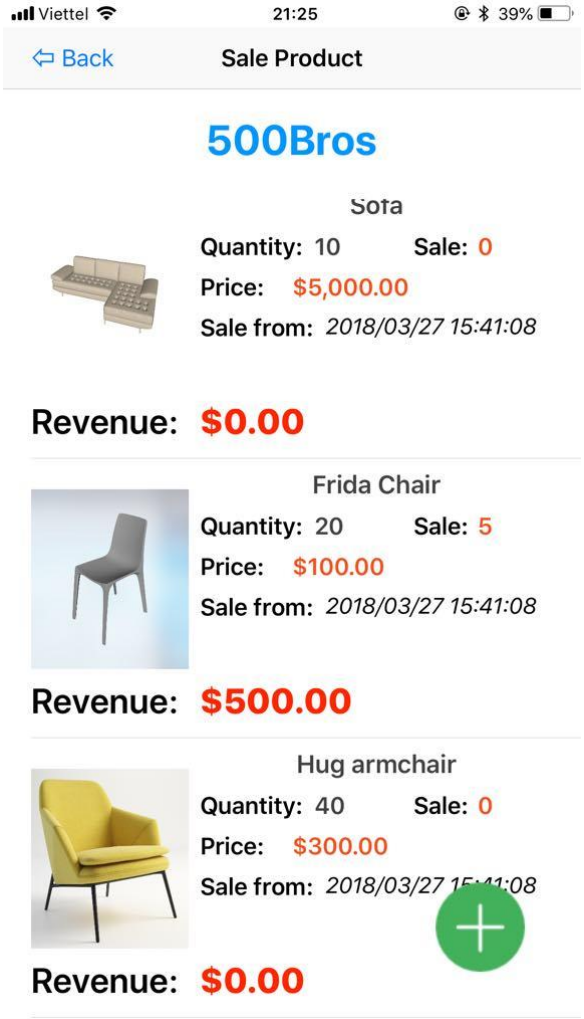


2.2.2.3 Manipulate orders

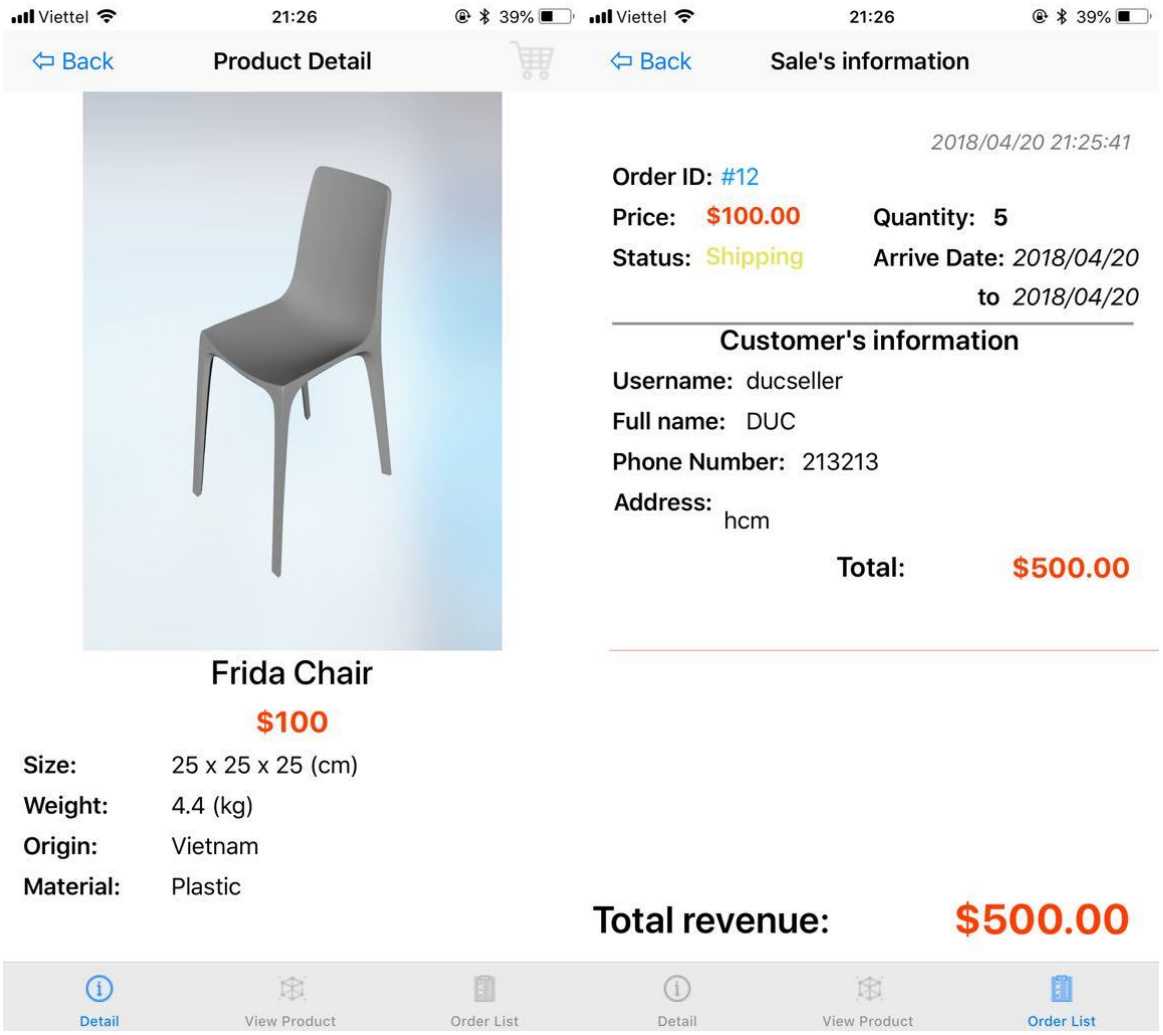
Choose "Seller Zone" on tab bar at bottom, then choosing "Sale product" to manipulate your products which is sold on marketplace.



Here's all products are now selling on marketplace. By choose any product, you can view product's detail.



By moving to last tab on tab bar, you can manipulate orders on this product.



G. Appendix

The reference components of the reference material refer to the way of writing at: http://www.khoahocviet.info/meresci/vi/meresci03d4.html