**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Drawing with Simple Brush**

|  |  |
| --- | --- |
| **Group 6** | |
| **Group members** | Trương Nguyễn Hồng Huân – Team Leader – SE61577  Đỗ Quốc Bảo – Team Member – SE61657  Nguyễn Minh Quang – Team Member – SE61650  Lê Hoàng Long – Team Member – SE61599 |
| **Supervisor** | Mr. Kiều Trọng Khánh |
| **Ext. Supervisor** | N/A |
| **Capstone Project**  **code** | DWSB |

-Ho Chi Minh City, May 8th, 2017-

*This page is intentionally left blank*

## Table of Contents

[Table of Contents 3](#_bookmark0)

[List of Tables 4](#_bookmark1)

[Definitions, Acronyms, and Abbreviations 6](#_bookmark2)

1. [Report No. 1 Introduction 7](#_bookmark3)
   1. [Project Information 7](#_bookmark4)
   2. [Introduction 7](#_bookmark5)
   3. [Current Situation 7](#_bookmark6)
   4. [Problem Definition 7](#_bookmark7)
   5. [Proposed Solution 8](#_bookmark8)
      1. [Feature functions 8](#_bookmark9)
      2. [Advantages and disadvantages 8](#_bookmark10)
   6. [Functional Requirements 9](#_bookmark11)

6.1 Name Card Management .................................................**Error! Bookmark not defined.**

6.2 Event ................................................................................**Error! Bookmark not defined.**

6.3 Searching..........................................................................**Error! Bookmark not defined.**

6.4 Suggestion ........................................................................**Error! Bookmark not defined.**

6.5 User Management ...........................................................**Error! Bookmark not defined.**

* 1. [Role and Responsibility 9](#_bookmark12)

## List of Tables

[Table 1: Roles and Responsibilities 9](#_bookmark13)

## List of Figures

Figure 1: Modified Waterfall Development Model..............................**Error! Bookmark not defined.**

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| DWSB | Drawing With Simple Brush |
| UI | User Interface |
| UX | User Experience |

# Introduction

## Project Information

* Project name: **Drawing with Simple Brush**
* Project Code: **DWSB**
* Product Type: **Mobile Application, Website Application**
* Start Date: **May 8th, 2017**
* End Date: **<Ngày kết thúc>**

## Introduction

In this document, we introduce a solution to help children drawing easily with a tablet or smartphone.

Children usually draw some things that they like or they just see. Many researches have also explained why children love drawing and confirmed the benefits of drawing for children. However, most of them don’t know how to draw and they need someone to guide. Some parents do not have much time to guide their kids and they must choose another solution.

Recognize these problems; we build a system which help children to solve current problems. The system intends to provide the experience of a virtual drawing tutor for children. In the process of analysis, we believe our algorithm will help children easy to imagine the step of drawing in which they input. The inputed pictures are converted to simple brush, and then show the strokes in step by step to make sure the children will be able the draw easily.

## Current Situation

Most of children love drawing but they do not know how to draw. Some parents do not have much time to guide their kids and they must choose another solution.

* Take children to drawing classes. This solution is very effective but it quite expensive.
* Drawing Books is the most common solution. Howerver, the problem is inconvenient for erasing and reusing. In addition, the problem of drawing patterns is not diversified because of the limitations of drawing books.
* Application on mobile/ tablet can teach children in the market. This solution is quite new with many people. But it can solve problems better more than others.

## Problem Definition

Below are advantages and disadvantages of other current application in the market:

|  |  |  |
| --- | --- | --- |
| **App Name** | **Advantages** | **Disadvantages** |
| DrawYourPet | * Have guide function in step by step. * Pictures close to children. | * Category is not diversified. * Only allow drawing from available sample pictures. * Don’t have comparision function. |
| Draw4Kids | * Have guide function in step by step. | * Don’t have option to choose level. |
| Drawing Desk | * Nice design. * Support other mode instead of only drawing. | * Drawling Desk doesn’t have guide function in step by step. |

## Proposed Solution

Our proposed solution is to build an algorithm to transfer photos or pictures available in the phone to “simple brush” photos. Then, the system will seperate strokes and displays them in step by step so that children can draw easily.

### Feature functions

* Web application:
* Manage sample pictures of application.
* Manage accounts of application.
* Provides diversity for the samples.
* Mobile application:
* Search sample pictures with appropriate criteria.
* Draw free picture.
* Draw pictures under the guidance step by step.
* Take a picture.
* Choose picture in device to draw
* Compare the result picture and sample picture.
* Share picture.

### Benefits and drawbacks

The benefits and drawbacks of the proposed solution:

* Benefits:
* Provide the diversity of samples to draw.
* Allow children to draw their love pictures.
* Allow children to learn step by step drawing.
* Check the drawing results.
* Drawbacks:
* Only online application.

## Functional Requirements

Function requirements of the system are listed as below:

* Administrator component:
* Manage sample pictures of application
* Staff component:
* Manage licenses of user.
* Manage accounts of application.
* Provides diversity for the samples.
* User Applications:
* Search sample pictures with appropriate criteria.
* Draw pictures under the guidance step by step.
* Take a picture.
* Choose picture in device to draw.
* Compare the result picture and sample picture.
* Share picture.

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| 1 | Kiều Trọng Khánh | Project Manager | Supervisor | [khanhkt@fpt.edu.vn](mailto:baodqse61657@fpt.edu.vn) |
| 2 | Trương Nguyễn Hồng Huân | Developer | Leader | [huantnhse61577@fpt.edu.vn](mailto:huantnhse61577@fpt.edu.vn) |
| 3 | Đỗ Quốc Bảo | Developer | Member | [baodqse61657@fpt.edu.vn](mailto:baodqse61657@fpt.edu.vn) |
| 4 | Nguyễn Minh Quang | Developer | Member | [quangnmse61650@fpt.edu.vn](mailto:quangnmse61650@fpt.edu.vn) |
| 5 | Lê Hoàng Long | Developer | Member | [longlhse61599@fpt.edu.vn](mailto:longlhse61599@fpt.edu.vn) |

Table 1: Roles and Responsibilities

# Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

|  |  |
| --- | --- |
| **Official name** | Drawing with Simple Brush. |
| **Vietnamese name** | Ứng dụng hỗ trợ trẻ em tập vẽ hình. |
| **Abbreviation** | DWSB |

### Problem Abstract

We are developing new algorithm to handle pictures in the software field and our team is not familiar with it so it might require time and effort for the whole team to be able to understand and use it.

On the other hands, the algorithm can separate picture into arrays is not existed in the market; we need time to research about it. Besides, comparision algorithm needs high precision which waste much time to improve.

Our team also does not have UI and UX design background so our application and web design may not be easy to use or the main features may not attract enough attention from customers.

### Project Overview

#### Current Situation

* Lack of handle image knowledge: Most our members do not have experience to handle image. Therefore, our team need significant amount of time to research.
* Develop new algorithm to convert to simple brush: Refer to available algorithm in the market.
* Develop new algorithm to convert to single brush step by step: This algorithm is not existed in the market, so we need much time to research.
* Develop new algorithm comparision algorithm: This algorithm needs high precision, so we need much time to improve the accuracy.
* Lack of UI knowledge: Our team members all study Information System and no one have good understanding about UI and UX of a standard E-Commerce application. That is the reason why spending time to study popular websites UI, UX is quite important.

#### The Proposed System

* Our system includes two main subsystems: an online website for company’s staffs, a mobile application for users.
* DBWS apply some tools to support drawing in application.
* DBWS use OpenCV library to support basic functions about handle image.
* Develop an algorithm about analysis and processing picture by apply filtering and erosion, DBWS will solve about the converting any pictures to simple brush
* Develop another algorithm to separate picture into arrays by apply conner detection and flooding. Then arrange to logical steps.
* DBWS build an algorithm that use Euclidian distance to the similarity of the picture from the drawer with the original picture at a relative level to check the results.
* We also build a high available web server to maintain the main system to work 24/7 to make sure that if mobile applications need access to the information there will be always available.

##### Web Site

* For Administrator:
  + Manage Account
* For Staff:
  + Manage License
  + Manage Picture

##### Mobile Application

* For User:
  + Search Picture
  + Draw Picture
  + Share Picture
  + Compare Picture
  + Contribute Picture
  + Breakdown Step

### Boundaries of the System

* The system do:
  + Allow users to search picture.
  + Allow users to draw picture.
  + Allow users to buy license.
  + Allow users to share picture.
  + Allow users to contribute picture.
  + Allow users to breakdown picture.
  + Allow users to compare the result picture and sample picture.
* The system not do:
  + Not manage revenue.

### Future Plans

Add to time-mode to increase the challenge.

Expand the community into a social network for interactive children.

### Development Environment

## Hardware requirements

|  |  |  |
| --- | --- | --- |
| **Windows** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window Server 2003 | Window Server 2008 |
| **Computer Processor** | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz) |
| **Computer Memory** | 4GB RAM | 8GB or more |

Table 2: Hardware Requirement for Server

|  |  |  |
| --- | --- | --- |
| **Windows** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | 4 Mbps | 8 Mbps |
| **Operating System** | Android 5 | Android 5 or above |
| **Hardware** | Camera 5 MP | Camera 8 MP |
| **Computer Memory** | 1GB RAM | 2GB or more |

Table 4: Hardware Requirement for Mobile

## Software requirement

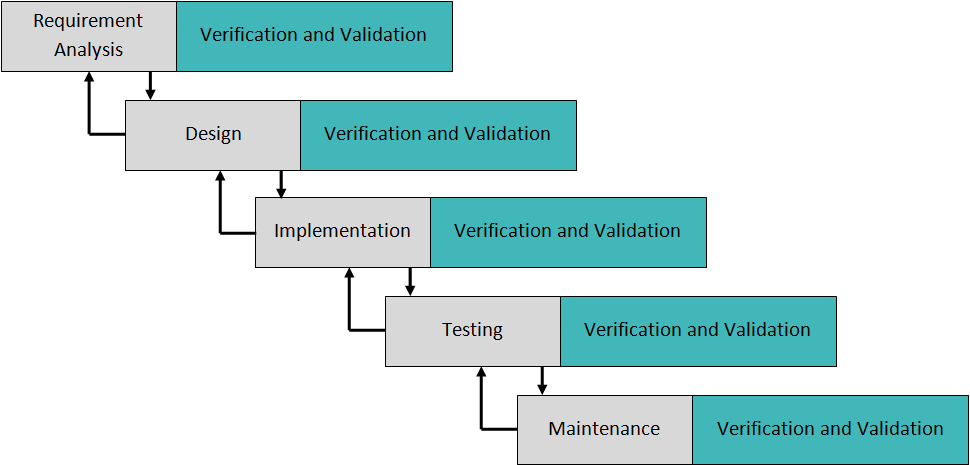
|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| IDE | Android Studio, Intellij IDEA 2017.1.2, PyCharm 2017.1.2 | Used to implement mobile, website and web service. |
| DBMS | MySQL 5.7 | Used to create and manager the database for system. |
| Source control | SourceTree 2.0.20.1 | Used for source control. |
| Modeling tool | StartUML 2.8 | Used to create models and diagrams |

## Project organization

### Software Process Model

This project is developed under waterfall model. We apply modified waterfall model to capable with current situation in our team. We choose this model because the following reasons:

* This is a project with clear requirement.
* We get a clear picture of their requirements, current situation and what have to be done.
* We have developed an algorithm and know quite clear the steps so no need communicate many with customers.
* Based on researches and clarify standard regimen of diseases are stable, clear, fixed and well understood by all team members.
* This Modified Waterfall model involves VERIFICATION or VALIDATION between the phases, so any deviations can be corrected immediately, providing the customer satisfaction, so this is preferred.
* We decide to modify the waterfall model (as picture below) so we can mix Requirement Analysis, Design, Implementaion, Testing and Maintenance phase in order to fit with out team’s working process.



**Figure 1: Waterfall Model**

For more information:

<http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm>

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Kiều Trọng Khánh | Project manager | * Specify user requirement * Control the development process * Give out technique and business analysis support |
| **2** | Trương Nguyễn Hồng Huân | Team Leader, Developer, Tester | * Managing process * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **3** | Đỗ Quốc Bảo | Developer, Tester | Designing database  Clarifying requirements  Prepare documents  GUI Design  Create test plan  Coding  Testing |
| **4** | Nguyễn Minh Quang | Developer, Tester | Designing database  Clarifying requirements  Prepare documents  GUI Design  Create test plan  Coding  Testing |
| **5** | Lê Hoàng Long | Developer, Tester | Designing database  Clarifying requirements  Prepare documents  GUI Design  Create test plan  Coding  Testing |

Table 4: Roles and Responsibilities Details

### Tools and Techniques

|  |  |
| --- | --- |
| **Tool / Technique** | **Name / Version** |
| Front-end | HTML5, CSS3, JavaScript, jQuery, AJAX, Bootstrap |
| Back-end | Python 3, JavaEE 8, Spring Framwork 4 |
| Web Server | Apache Tomcat 8 |
| Developing Tool | IntelliJ IDEA 2017.1.2, Android Studio, PyCharm 2017.1.2, Spyder 3 |
| Database Management System | MySQL 5.7 |
| Source Control | SourceTree 2.0.20.1 |
| Modeling tool | StartUML 2.8 |

## Project Management Plan

### Software development life cycle

#### Phase 1: Requirement Analysis

|  |  |
| --- | --- |
| **Phase** | Requirement Analysis |
| **Descriptions** | Collect requirements from customer.  Identify and clarify requirements for the system in general. |
| **Deliverables** | Introduction of proposed system.  Software requirement specification.  Project Task Plan.  Prototypes |
| **Resources needed** | All team members; 6 days |
| **Dependencies and constraints** | N/A |
| **Risks** | Missing requirement  Unclear scope of project  Lack of member share of understand |

#### Phase 2: Designs

|  |  |
| --- | --- |
| **Phase** | Design |
| **Descriptions** | Architecture design for the system  Detail design using top-down break down  Choose Architecture style |
| **Deliverables** | Software Design Document  Base code structure  Technology notes |
| **Resources needed** | All team members; 10 days |
| **Dependencies and constraints** | Depend on “Requirement Analysis” |
| **Risks** | Lack of experience.  Not fulfil requirement. |

#### Phase 3: Implementation

|  |  |
| --- | --- |
| **Phase** | Implementation |
| **Descriptions** | Implement algorithms and processing API  Implement mobile application  Implement administrator website  Unit Testing |
| **Deliverables** | Processing API  Mobile application  Administrator website  Unit Testing document |
| **Resources needed** | All team members; 30 days |
| **Dependencies and constraints** | Depend on “Designs” |
| **Risks** | Lack of experience.  Implementation does not match with design.  Human mistake |

#### Phase 4: Testing

|  |  |
| --- | --- |
| **Phase** | Testing |
| **Descriptions** | Testing  Verify code with Software Design Document  Validate with Software requirement specification |
| **Deliverables** | Defect list  Verification checklist  Validation checklist |
| **Resources needed** | All team members; 10 days |
| **Dependencies and constraints** | Depend on “Testing” |
| **Risks** | Lack of experience.  Missing test case. |

#### Phase 5: Maintenance

|  |  |
| --- | --- |
| **Phase** | Maintaince |
| **Descriptions** | Deploying API server and website  Deploy mobile application for use  Maintaince |
| **Deliverables** | Fully accessible website from domain  Download and useable mobile application on Google Play |
| **Resources needed** | All team members; 10 days |
| **Dependencies and constraints** | Depend on “Testing” |
| **Risks** | Lack of experience. |

### Phase Detail

#### Phase 1: Requirement Analysis

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Collect requirements** | Find which systems currently provide similar service, their strengths and weakness. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **2. Identify and clarify main functions.** | Define which main functions system should provide. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **3. Create System Introduce.** | Complete Introduce Report. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **4. Software Project Management Plan** | Prepare Project Management Plan | HuanTNH, BaoDQ, QuangNM, LongLH |
| **5. Prototype** | Build prototype of proposed system (Website/Mobile) | HuanTNH, BaoDQ, QuangNM, LongLH |
| **6. SRS** | Create SRS document. | HuanTNH, BaoDQ, QuangNM, LongLH |

Table 5: Phase 1: Requirement Analysis

#### Phase 2: Design

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Detailed Design** | Compare new document with existed documents of system. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **2. Database Design** | Based on parsed data to recommendation.  Based on other needs to recommendation. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **3. Technology research** | Android. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **4. Design Document** | Prepare Project Management Plan | HuanTNH, BaoDQ, QuangNM, LongLH |

Table 5: Phase 2: Design

#### Phase 3: Implementation

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Front-End web functions** | Implement front-end functions on web. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **2. Front-End mobile functions** | Implement front-end functions on mobile. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **3. Back-End web functions** | Implement back-end functions on web. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **4. Mobile functions** | Implement mobile application. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **5. Unit Testing** | Write test case and testing for web functions. | HuanTNH, BaoDQ, QuangNM, LongLH |

Table 5: Phase 3: Implementation

#### Phase 4: Testing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Integration testing** | Write testcaseandtesting systems | HuanTNH, BaoDQ, QuangNM, LongLH |
| **2. Alpha testing** | Doalpha test with customer. | HuanTNH, BaoDQ, QuangNM, LongLH |

Table 5: Phase 4: Testing

#### Phase 5: Maintenance

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Installation guide** | Write installation guide. | HuanTNH, BaoDQ, QuangNM, LongLH |
| **2. User Manual** | Write user manual. | HuanTNH, BaoDQ, QuangNM, LongLH |

Table 5: Phase 5: Maintenance

### All Meeting Minutes

## Coding Convention

Python: Using for develop algorithm.

Summary:

* Naming convention:
* Use snake-case style for variables and functions name.
* Use snake-case style for files and directories name.
* Use camel-case for classes.
* Indentation:
* Use four spaces per indentation level.
* Do not use “;” for separate statements and expressions.
* Avoid single-line method.
* Classes:
* Put underscore before name for private instance variables.
* Use self to refer current class instance.

Java: Using to develop website and mobile application.

Summary:

* Naming Convention:
* Use camel-case style for variables and functions name (Lowercase for firs letter).
* Use camel-case for classes (Uppercase for first letter).
* Indentation:
* Use four spaces per indentation level.
* User eight spaces to continue indent.
* Use “;” for separate statements and expressions.
* Classes:
* Classes should be declared using class keyword.
* Using public, private, protected to declare protection level of instance variable and function.
* Use this to refer current class instance.

References:

[1] Python: <https://www.python.org/dev/peps/pep-0008/>

[2] Java: <http://www.oracle.com/technetwork/java/codeconvtoc-136057.html>

# Software Requirement Specification

## User Requirement Specification

### Guest Requirement

Guest is person who does not have access the system. Guest can use some functions in the system. To use all functions, guest must login. These are some functions guest can use:

* Sign up
* Sign in
* Search

### User Requirement

User is person who signs in into the system. These are some functions user can use:

* Search
* Draw Picture
  + - Share Picture
    - Compare Picture
* Buy License

### Authorized User

Authorized User is person who signs in into the system. These are some functions user can use:

* Sign out
* Edit Profile

### Premium User Requirement

Premium User is person who buys license. These are some functions user can use:

* Handle Picture

### Administrator Requirement

Admin is person who manages all the staffs in the system. Admin can do following functions:

* Manage Account
  + - Add Account
    - Enable Account

### Staff Requirement

Staff is person who manages all the pictures and licenses in the system. Staff can do following functions:

* Manage Picture
  + - Add Picture
    - Update Picture
    - Approve Picture
    - Import Picture
* Manage License Type
  + - Add License Type
    - Update License Type

### Auto Handler Requirement

* Convert to Simple Brush
* Notify

## System Requirement Specification

### External Interface Requirement

#### User Interface

* The user interface uses Vietnamese as main language for users on Mobile application.
* The user interface uses English as main language for Staff and Admin on Web application.
* The user interface displays best on 1920x1080.

#### Hardware Interface

* N/A

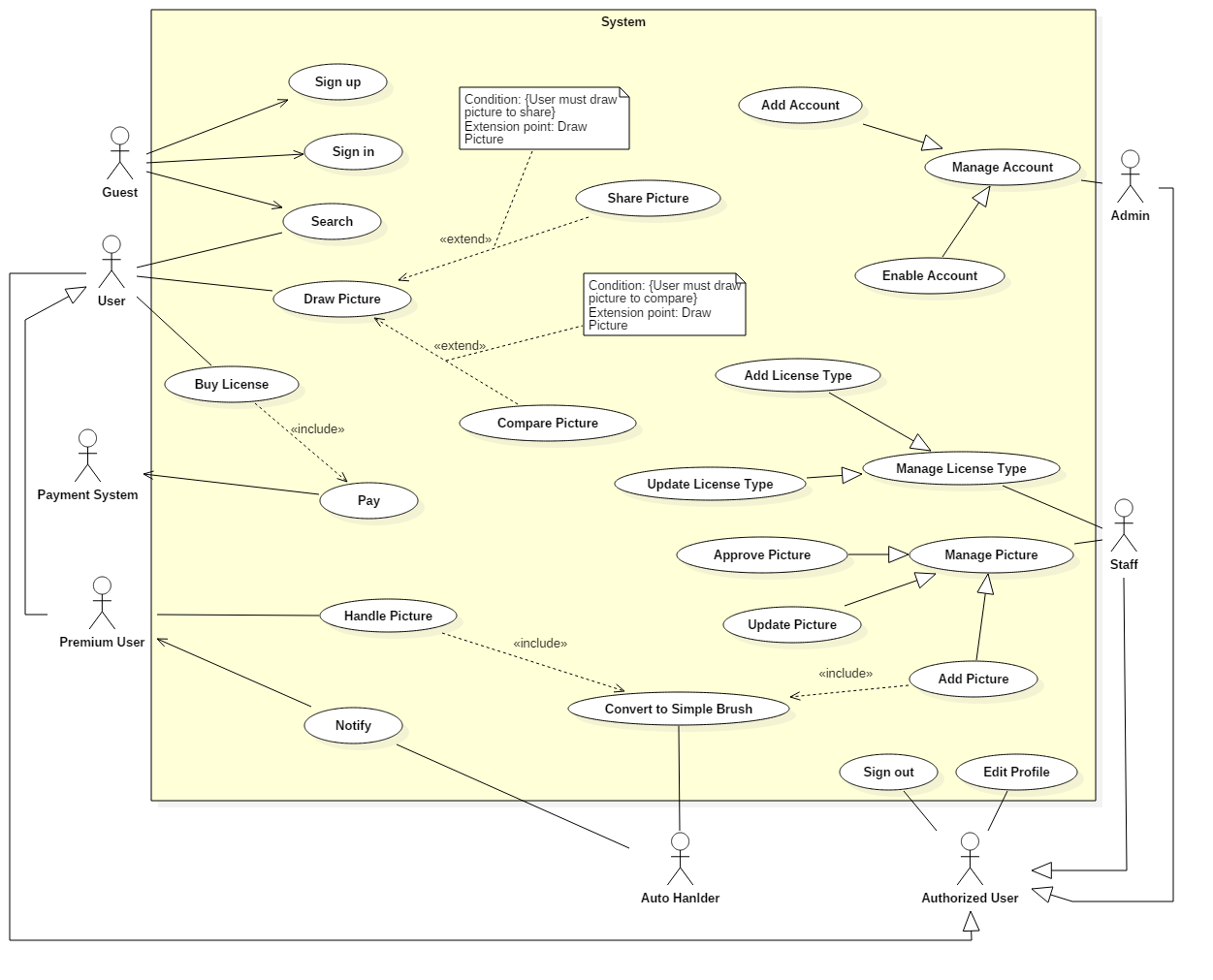
#### Software Interface

* N/A

#### Communication Protocol

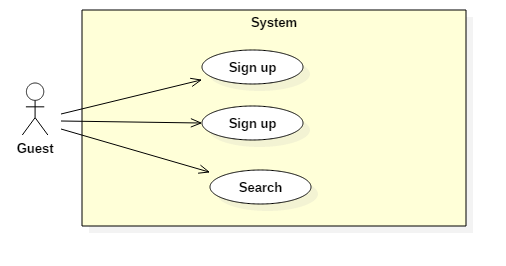
* Use HTTP protocol 1.1 for communication between the web browser and the web server.
* Use HTTP protocol 1.1 for communication between the mobile application and the web service.
* Use HTTP protocol 1.1 for communication between the server and the Microsoft service.

### System Overview Use Case

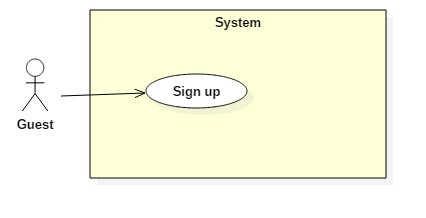


### List of Use Case

#### <Guest> Overview Use Case

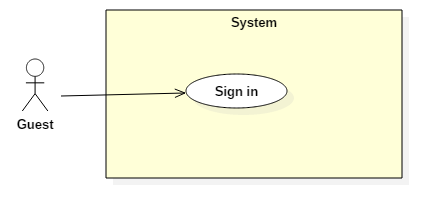


##### <Guest> Sign up (UC\_G01)



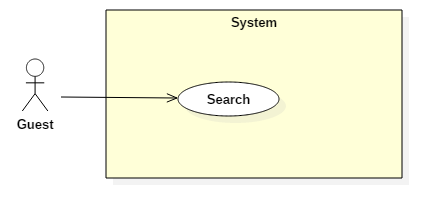
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_G01** | | | |
| **Use Case No.** | UC\_G01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Sign up | | |
| **Author** | QuangNM | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest to sign up new account to the system.   **Goal:**   * This function helps guest sign up new account to the system. Using this account, guest can login to the system to use some functions with the user’s permission.   **Triggers:**   * Actor sends command to sign up new account.   **Preconditions:**   * N/A   **Post Conditions:**   * **Success:** New account has been signed up to the system. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to sign up view. | System requires some information from actor:   * **Email**: free text input, required, regex: /\A([\w+\-].?)+@[a-z\d\-]+(\.[a-z]+)\*\.[a-z]+\z/i, length 8 – 40 characters. | | 2 | Guest inputs required information. |  | | 3 | Guest sends command to sign up. | System validates inputted information.  [Exception 1, 2, 3, 4] | | 4 |  | System creates new account. |   **Alternative:** N/A  **Exceptions:**  **Relationships:** N/A  **Business Rules:**   * System does not allow duplicate “Email”. * The initial status of account will be set to “Active”. | | | |
| Table 39: Use Case UC\_G01 - <Guest> Sign Up | | | |

##### <Guest> Sign in (UC\_G02)



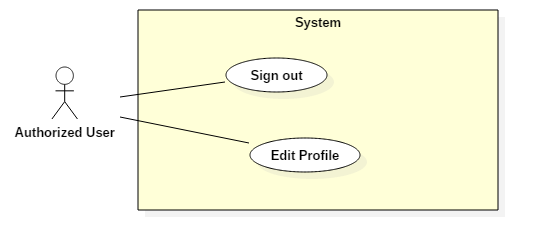
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_G02** | | | |
| **Use Case No.** | UC\_G02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Sign in | | |
| **Author** | QuangNM | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows Guest to authenticate specified role of user such as Admin, Staff and Premium User.   **Goal:**   * Guest can access system and use features with their specific role.   **Triggers:**   * Actor sends command to sign in.   **Preconditions:**   * N/A   **Post Conditions:**   * **Success:** Actor goes to main view with their specified role. * **Fail:** Show error message: “Email or password is not correct”.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to sign in view. | System requires identify information from Guest:   * **Email**: free text input. * **Password**: free text input. | | 2 | Guest inputs information. |  | | 3 | Guest sends command to sign in the system. | Guest will sign in the system with their specified role.  [Alternative 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Guest enters wrong identify information. | System shows error message: “Email or password is not correct”. |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * Password is encrypted before being sent to server. * System will compare actor’s encrypted string with the one stored in the system. If matched, actor will be signed in with their related role. | | | |
| Table 39: Use Case UC\_G02 - <Guest> Sign In | | | |

##### <Guest> Search (UC\_G03)

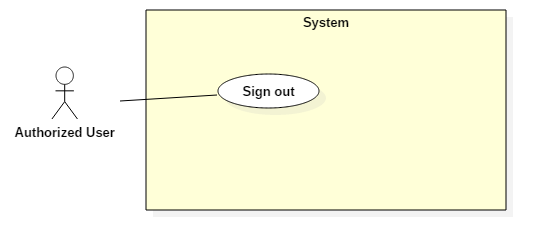


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_G03** | | | |
| **Use Case No.** | UC\_G03 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search | | |
| **Author** | QuangNM | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows Guest to search pictures via tag.   **Goal:**   * Display result for searched pictures.   **Triggers:**   * Actor enters keywords or tags and sends command to seach.   **Preconditions:**   * N/A   **Post Conditions:**   * Success: Show results of possible pictures that match guest input. * Fail: N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor enters tag. |  | | 2 | Actor sends command to search. | System based on tag to get picture list from database.  System shows the list of pictures that matching with the searched input.  [Alternative 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | There is no result have the same info with inputted information. | System shows message “No result found!” |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * Inputed tag will be breakdown into array of tag. * For each keyword or tag, find all the pictures with that keyword or tag then push all found pictures to the shared list. * After done searching for each keyword or tag, sort the shared list by most occurrences first, then added date. * Display the shared list to the user with that order. | | | |
| Table 39: Use Case UC\_G03 - <Guest> Search | | | |

#### <Authorized User> Overview Use Case

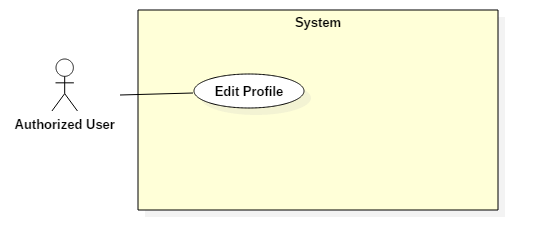


##### <Authorized User> Sign out (UC\_AU01)



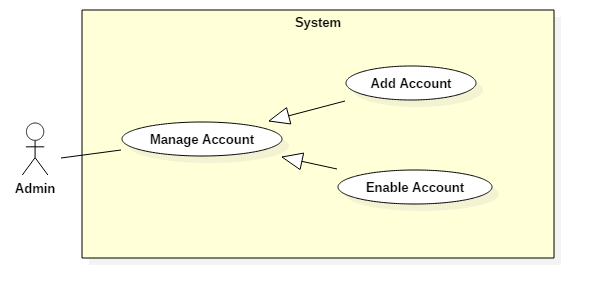
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_AU01** | | | |
| **Use Case No.** | UC\_AU01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Sign out | | |
| **Author** | QuangNM | | |
| **Date** | 18/05/2017 | **Priority** | High |
| **Actor:**   * Authorized User (include User, Premium User, Staff, Admin)   **Summary:**   * This use case allows Authorized User to sign out of the system.   **Goal:**   * Authorized User can sign out of the system.   **Triggers:**   * Authorized User sends command to sign out. * Authorized User sends command after not available time too long.   **Preconditions:**   * Actor signed in into the system.   **Post Conditions:**   * **Success:** User goes to sign in view. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends command to sign out.  [Alternative 1] | System clears session and sign user out of the system.  System displays sign in view. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | User sends command after not available time too long. | System checks the session state then takes user out of system.  System display sign in view. |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * If user is not available longer than 20 minutes, they will see the sign in view when they are back. | | | |
| Table 39: Use Case UC\_AU01 - <Authorized User> Sign out | | | |

##### <Authorized User> Edit Profile (UC\_AU02)

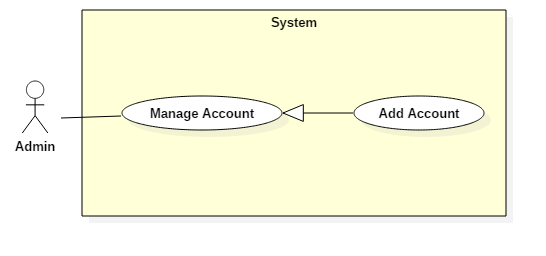


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_AU02** | | | |
| **Use Case No.** | UC\_AU02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Edit Profile | | |
| **Author** | QuangNM | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Authorized User (include User, Premium User, Staff, Admin)   **Summary:**   * This use case allows Authorized User to edit their profile information.   **Goal:**   * This function help Authorized User edit their profile with new information.   **Triggers:**   * Authoried User sends command to edit profile.   **Preconditions:**   * Actor signed in into the system.   **Post Conditions:**   * **Success:** Profile information will be updated. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to edit profile view. | System requires some information from User:   * **Full Name**: free text input, required, length 1- 20 characters. * **Gender:** * **Phone**: free text input, Integer, required, length 9 – 11 digits. * **Date of Birth:** * **Old Password**: free text input. * **New Password**: free text input, required, length 6 – 20 characters. * **Confirm password**: free text input, required, length 6 – 20 characters. | | 2 | User inputs information. |  | | 3 | User sends command to edit profile. | System validates inputted information.  [Exception 1, 2, 3, 4, 5] | | 4 |  | System edits profile with new information. |   **Alternative Scenario:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Invalid full name format. | System shows error message: “Full name cannot be blank”. | | 3 | Invalid phone format. | System shows error message: “Phone cannot blank or contain alphabetical characters and must be 9 – 11 digits”. | | 4 | Invalid old password format. | System shows error message: “Password cannot be blank and must be 6 – 20 characters long”. | | 5 | Invalid confirm password doest not match. | System shows error message: “These passwords don’t match. Try again?”. |   **Relationships:** N/A  **Business Rules:**   * User’s profile will be updated in the system with inputted information. * Password will be encrypted before storing into the system. | | | |
| Table 39: Use Case UC\_AU02 - <Authorized User> Edit Profile | | | |

#### <Admin> Overview Use Case

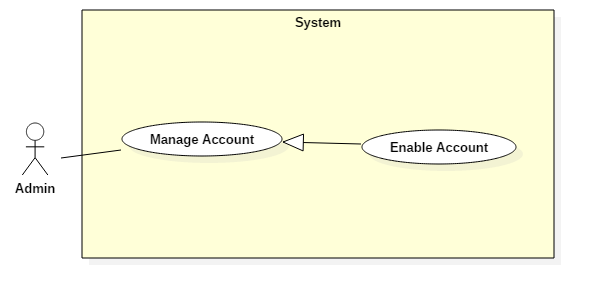


##### <Admin> Add Account (UC\_A01)



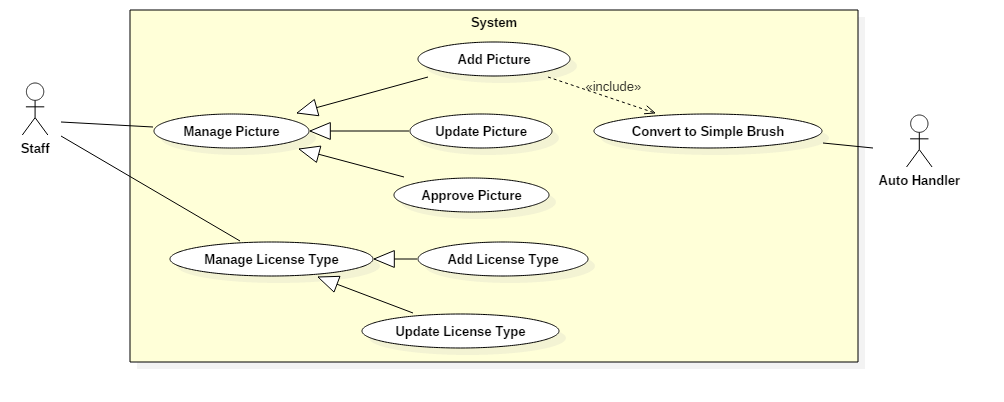
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S01** | | | |
| **Use Case No.** | UC\_A01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add Account | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Admin   **Summary:**   * This use case allows admin to create new account.   **Goal:**   * This function helps admin create new account to the system. Using this account, staff can login to the system to use some functions with the staff’s permission.   **Triggers:**   * Admin sends creating account command.   **Preconditions:**   * Actor signed in at role “Admin”.   **Post Conditions:**   * **Success:** New account has been added to the system. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin goes to create account view. | System requires some information from Admin:   * **Full Name**: free text input, required, length 1 – 40 charactes. * **Email**: free text input, required, length 8 – 40 characters, regex: /\A([\w+\-].?)+@[a-z\d\-]+(\.[a- * z]+)\*\.[a-z]+\z/i * **Gender:** select * **Phone:** free text input, required, length 1 – 20 charactes * **Date of Birth:** | | 2 | Admin inputs required information. |  | | 3 | Admin sends command to create new account.  [Alternative 1, 2] | System validates inputted information.  [Exception 1, 2, 3] | | 4 |  | System adds new account and shows success message: “Add new account successfully”. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Admin sends command to reset. | System resets all required information fields. |   [Alternative 2]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Admin sends command to cancel. | System closes add new account form. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Email had already existed in the system. | System shows error message: “Email had already existed”. | | 2 | Invalid Email format. | System shows error message: “Not a valid email format”. | | 3 | Invalid First Name format. | System shows error message: “First Name cannot be blank and must be 1 – 40 characters long”. | | 4 | Invalid Last Name format. | System shows error message: “First Name cannot be blank and must be 1 – 40 characters long”. |   **Relationships:** N/A  **Business Rules:**   * The initial status of account will be set to “Active”. * System does not allow duplicate “Email”. * Password will be randomly generated. Length of generated password is 8 characters. * Randomly generated password includes alphanumerics, uppercase and lowercase. * Password will be encrypted before storing to the system. | | | |
| Table 39: Use Case UC\_S01 - <Admin> Add Staff | | | |

##### <Admin> Enable Account (UC\_A02)

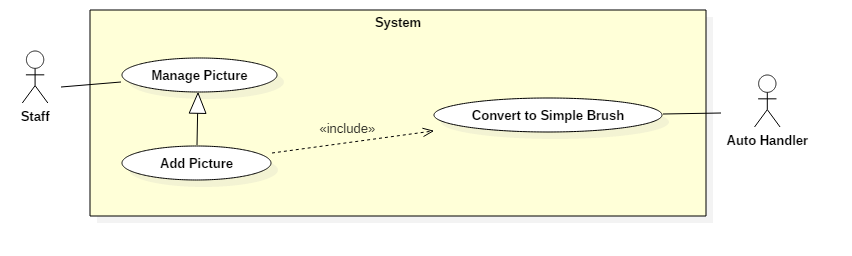


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_AU02** | | | |
| **Use Case No.** | UC\_AU02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Enable Account | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Admin   **Summary:**   * This use case allows admin to update account.   **Goal:**   * This function helps admin update an available account when they want to change status of account.   **Triggers:**   * Admin sends command to enable account.   **Preconditions:**   * Actor signed in at role “Admin”. * The account must be available before.   **Post Conditions:**   * **Success:** Status account has been updated. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin goes to account’s profile. | System list out information of account in the system and components:   * **Full Name**: text, read only. * **Email**: text, read only. * **Gender**: text, read only. * **Phone**: text, read only. * **Date of Birth**: text, read only. * **Status:** select one of the options. | | 2 | Admin changes status of account. |  | | 3 | Admin sends command to enable account.  [Alternative 1] | System enables account and shows success message: “Enabled account successfully”. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Admin sends process. | System redirects to previous page. |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * Account’s status will be changed to “Deactive”. * If account’s status is changed to “Deactive”, the account won’t be able to access the system. | | | |
| Table 39: Use Case UC\_A02 - <Admin> Edit Staff | | | |

#### <Staff> Overview Use Case

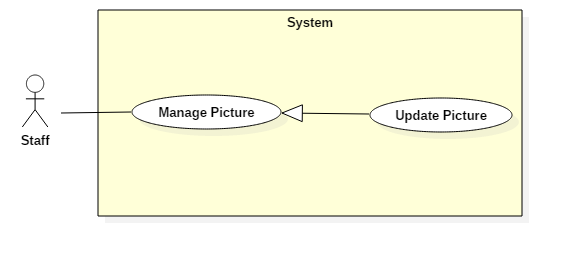


##### <Staff> Add Picture (UC\_S01)



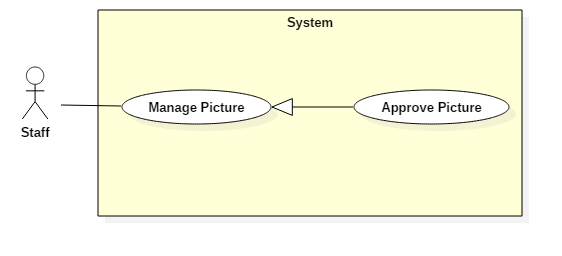
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S01** | | | |
| **Use Case No.** | UC\_S01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add Picture | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Staff   **Summary:**   * This use case allows staff to add new picture.   **Goal:**   * Staff can add new picture to the system.   **Triggers:**   * Staff sends adding picture command.   **Preconditions:**   * Actor signed in at role “Staff”.   **Post Conditions:**   * **Success:** New picture has been added to the system. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to add picture view. | System requires some information from Staff:   * **Picture name**: free text input, required, length 1-40 characters. * **Tag**: free text input, required, length 1-40 characters. * **Picture**: hidden free text input, required, length 1-40 characters. | | 2 | Staff inputs required information. |  | | 3 | Staff sends command to add a new picture.  [Alternative 1, 2] | System validates inputted information.  [Exception 1, 2, 3] | | 4 |  | System adds new picture and shows success message: “Add new picture successfully”. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to reset. | System resets all required information fields. |   [Alternative 2]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to cancel. | System closes add new picture form. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Picture is empty. | System shows error message: “Picture cannot be blank”. | | 2 | Invalid picture format. | System shows message: “Picture must be in jpg/jpeg/png format”. | | 3 | Invalid picture name format. | System shows error message: “Picture name cannot be blank and must be 1-40 characters long”. | | 4 | Invalid tag format. | System shows error message: “Tag cannot be blank and must be 1-40 characters long”. |   **Relationships:**   * Include Convert to Simple Brush: Auto Handler has to convert to simple brush when staff adds new picture.   **Business Rules:**   * The initial status of picture will be set to “Active”. * After adding picture, Auto Handler will convert picture to Simple Brush then breakdown into step by step to store in the system. | | | |
| Table 39: Use Case UC\_S01 - <Staff> Add Picture | | | |

##### <Staff> Update Picture (UC\_S02)



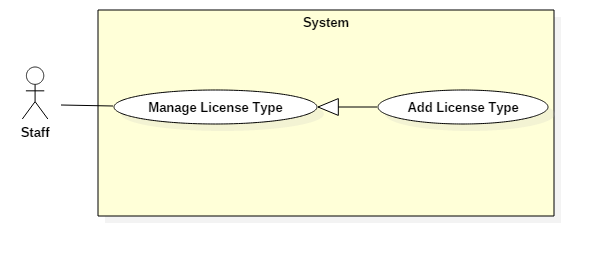
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S02** | | | |
| **Use Case No.** | UC\_S02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Update Picture | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Staff   **Summary:**   * This use case allows Staff to update information’s picture.   **Goal:**   * This functions helps Staff update an available picture.   **Triggers:**   * Staff sends updating picture command.   **Preconditions:**   * Actor signed in at role “Staff”. * The picture must be available before.   **Post Conditions:**   * **Success:** Show message: “”. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to update picture view. | System requires information from staff:   * **Tag**: free text input, required, length 1-40 characters. | | 3 | Staff changes information of picture. |  | | 4 | Staff sends command to update picture.  [Alternative 1] | System validates inputted information.  [Exception 1, 2] | | 5 |  | System updates picture and shows success message: “Update picture successfully”. |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to cancel. | System closes update picture form. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Invalid picture name format. | System shows error message: “Picture name cannot be blank and must be 1-40 characters long”. | | 2 | Invalid tag format. | System shows error message: “Tag cannot be blank and must be 1-40 characters long”. |   **Relationships:** N/A  **Business Rules:**   * Picture will be updated with inputed information. * If staff updates the status of picture to “Deactive”, the user will not be able to draw. | | | |
| Table 39: Use Case UC\_S02 - <Staff> Edit Product | | | |

##### <Staff> Approve Picture (UC\_S03)



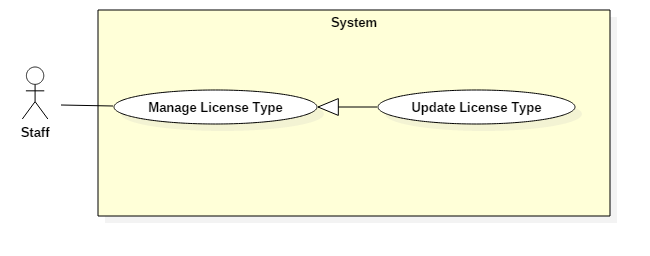
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S03** | | | |
| **Use Case No.** | UC\_S03 | **Use Case Version** | 2.0 |
| **Use Case Name** | Approve Picture | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Staff   **Summary:**   * This use case allows Staff to approve a specific picture for Premium User.   **Goal:**   * Staff can approve a specific picture for Premium User.   **Triggers:**   * Staff sends approving picture command.   **Preconditions:**   * Actor signed in at role “Staff”.   **Post Conditions:**   * **Success:** * **Fail:**   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to approve picture view. | System display a list of existing picture:   * **No**: text, read only. * **Picture**: hidden free text input. * **Tag:** text, read only. * **Created\_date:** text, read only. * **Status:** select one of the options. | | 2 | Staff selects a specific picture to approve.  [Alternative 1, 2] | System requires information from Staff:   * **Picture name:** free text input, required, length 1-40 characters. * **Tag:** free text input, required, length 1-40 characters. | | 3 | Staff inputs information. |  | | 4 | Staff sends command to approve picture. | System validates inputted information.  [Exception 1] | | 5 |  | System adds a picture to the system. |   **Alternative Senario:**   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to cancel. | System shows message options to confirm command. | | 2 | Staff confirms to delete picture. | System deletes selected picture. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | |  | Invalid tag format. | System shows error message: “Tag cannot be blank and must be 1 - 40 characters long”. |   **Relationships:**  **Business Rules:**   * Picture’s status will be changed to “Active”. * Picture will be approved with inputed information. * After picture is approved, system will notify for premium user. | | | |
| Table 39: Use Case UC\_S03 - <Staff> Approve Picture | | | |

##### <Staff> Add License Type (UC\_S04)



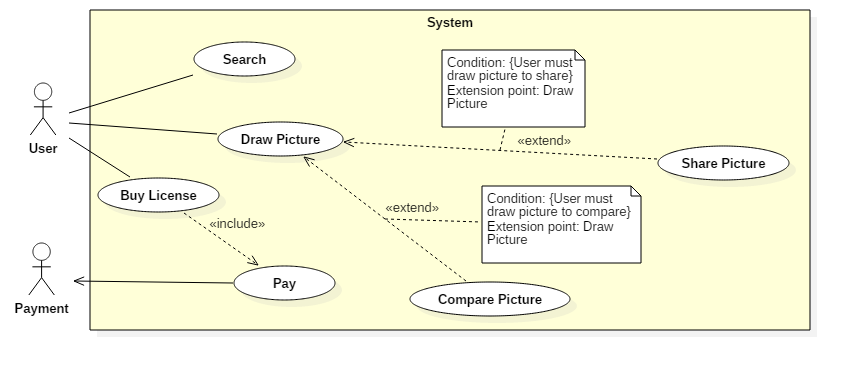
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S04** | | | |
| **Use Case No.** | UC\_S04 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add License Type | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Staff   **Summary:**   * This use case allows Staff to add new license type.   **Goal:**   * Staff can add new license type to the system.   **Triggers:**   * Staff sends adding license type command.   **Preconditions:**   * Actor signed in at role “Staff”.   **Post Conditions:**   * **Success:** New picture has been added to the system. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to create account view. | System requires some information from Staff:   * **License Name**: free text input, required, length 1 – 40 charactes. * **Day**: free text input, required, length 1 – 40 charactes. * **Price:** free text input, required, length 1 – 40 charactes. | | 2 | Staff inputs required information. |  | | 3 | Staff sends command to create new license type.  [Alternative 1, 2] | System validates inputted information.  [Exception 1, 2, 3] | | 4 |  | System adds new license type and shows success message: “Add new license type successfully”. |   **Alternative Senario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | 1 | Staff sends command to reset. | System resets all required information fields. |   [Alternative 2]   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to cancel. | System closes add new license type form. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Invalid license name format. | System shows error message: “License name cannot be blank and must be 1-40 characters long”. | | 2 | Invalid day format. | System shows error message: “Day cannot be blank and must be 1-40 characters long”. | | 3 | Invalid price format. | System shows error message: “Price cannot be blank and must be 1-40 characters long”. |   **Relationships:** N/A  **Business Rules:**   * The initial status of license type will be set to “Active”. | | | |
| Table 39: Use Case UC\_S04 - <Staff> Add License Type | | | |

##### <Staff> Update License Type (UC\_S05)

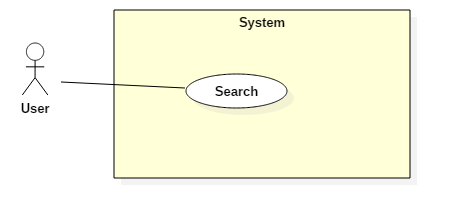


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S05** | | | |
| **Use Case No.** | UC\_S05 | **Use Case Version** | 2.0 |
| **Use Case Name** | Update License Type | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Staff   **Summary:**   * This use case allows Staff to update license type.   **Goal:**   * This function helps staff update an available license type when they want to change status of license type.   **Triggers:**   * Staff sends updating license type command.   **Preconditions:**   * Actor signed in at role “Staff”.   **Post Conditions:**   * **Success:** * **Fail:**   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to license type view. | System display a list of existing picture:   * **No**: text, read only. * **License Name**: free text input. * **Day**: free text input. * **Price**: free text input. * **Status:** select one of the options. | | 2 | Staff selects a specific license type to update.  [Alternative 1, 2] | System requires information from Staff:   * **License Name:** free text input, required, length 1-40 characters. * **Day:** free text input, required, length 1-40 characters. * **Price:** free text input, required, length 1-40 characters. | | 3 | Staff inputs information. |  | | 4 | Staff sends command to approve picture. | System validates inputted information.  [Exception 1] | | 5 |  | System adds a picture to the system. |   **Alternative Senario:**   |  |  |  | | --- | --- | --- | | No. | Actor Action | System Response | | 1 | Staff sends command to cancel. | System shows message options to confirm command. | | 2 | Staff confirms to delete picture. | System deletes selected picture. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Invalid license name format. | System shows error message: “License name cannot be blank and must be 1-40 characters long”. | |  | Invalid day format. | System shows error message: “Day cannot be blank and must be 1-40 characters long”. | |  | Invalid price format. | System shows error message: “Price cannot be blank and must be 1-40 characters long”. |   **Relationships:**  **Business Rules:**   * License type will be approved with inputed information. * If staff updates the status of license type to “In-Active”, the user will not be able to buy. | | | |
| Table 39: Use Case UC\_S05 - <Staff> Update License Type | | | |

#### <User> Overview Use Case

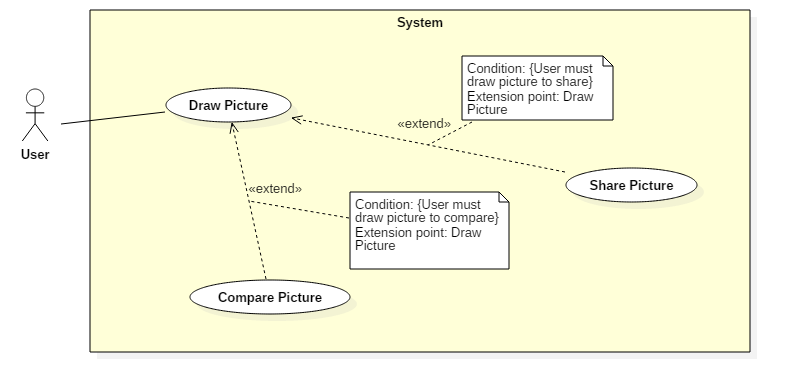


##### <User> Search (UC\_U01)



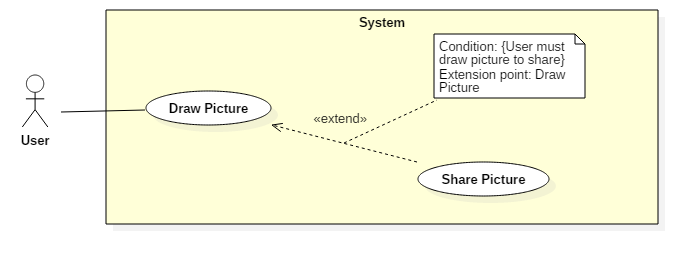
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_U01** | | | |
| **Use Case No.** | UC\_U01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search | | |
| **Author** | LongLH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows User to search pictures via keywords or tags.   **Goal:**   * Display result for searched pictures.   **Triggers:**   * User enters keywords or tags and sends command to seach.   **Preconditions:**   * Actor signed in at role “User”.   **Post Conditions:**   * Success: Show results of possible pictures that match guest input. * Fail: N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor enters keyword or tag. |  | | 2 | Actor sends command to search. | System based on keyword or tag to get picture list from database.  System shows the list of pictures that matching with the searched input.  [Alternative 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | There is no result have the same info with inputted information. | System shows message “No result found!” |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * Inputed tags will be breakdown into array of tag. * For each keyword or tag, find all the pictures with that keyword or tag then push all found pictures to the shared list. * After done searching for each keyword or tag, sort the shared list by most occurrences first, then added date. * Display the shared list to the user with that order. | | | |
| Table 39: Use Case UC\_U01 - <User> Search | | | |

##### <User> Draw Picture (UC\_U02)



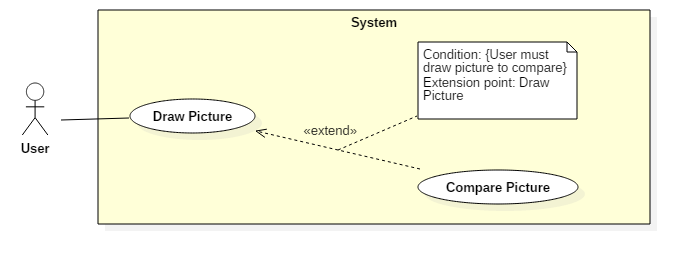
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_U02** | | | |
| **Use Case No.** | UC\_U02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Draw Picture | | |
| **Author** | LongLH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows User to draw picture in the application.   **Goal:**   * Picture will be displayed with Simple Brush.   **Triggers:**   * User sends drawing picture command.   **Preconditions:**   * Actor signed in into as role “User”. * Picture must be converted to Simple Brush.   **Post Conditions:**   * **Success:** Single strokes of the user will be displayed on the screen. * **Fail:** Single strokes of the user will not be displayed on the screen.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to picture view. | System displays a list of existing picture. | | 2 | Actor selects a specific picture to draw. | System displays 2 modes.   * Single screen mode * Dual screen mode   [Alternative 1] | | 3 | Actor selects single screen mode. | System will display sample picture with dot brush. | | 4 | Actor draws following sample picture.  [Exception 1] |  |   **Alternative Senario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Actor setects dual screen mode. | System will display two screens.   * Left Screen: show sample picture in step by step with simple brush. * Right Screen: show white page for user draw. |   **Exceptions:** N/A  **Relationships:**   * Extend from Share Picture. When user selects share picture. * Extend from Compare Pictuce. When user selects compare picture.   **Business Rules:**   * For single screen mode: * System will check to see if the dotted version of choosen picture is available. * If it is available, system will load the picture to display. * If it is not available, system will create a dotted version of the choosen picture by relative comparing and spacing the simple brush picture. Then system will load the dotted picture. * For dual screen mode: * System will check to see if the folder contains steps is already created. * If it is created, system will load the pictures by step number from 1 to all pictures in that folder. * If it is not created, the step folder will be created in the system, then system will invoke the Breakdown step by auto handler | | | |
| Table 39: Use Case UC\_U02 - <Staff> Draw Picture | | | |

##### <User> Share Picture (UC\_U03)



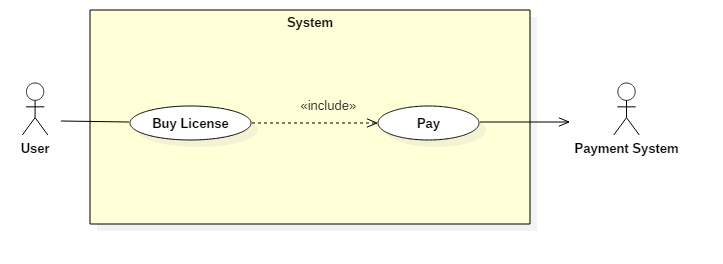
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_U03** | | | |
| **Use Case No.** | UC\_U03 | **Use Case Version** | 2.0 |
| **Use Case Name** | Share Picture | | |
| **Author** | LongLH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows user to share picture.   **Goal:**   * User can share picture.   **Triggers:**   * User sends command to share picture.   **Preconditions:**   * Actor signed in into as role “User”.   **Post Conditions:**   * **Success:** N/A. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor shares picture. | System displays dialog such as Facebook, Instagram. | | 2 | Actor selects |  |   **Alternative Senario:** N/A  **Exceptions:** N/A  **Relationships:**   * Extend to Share Picture: When user shares picture.   **Business Rules:**   * API | | | |
| Table 39: Use Case UC\_S03 - <Staff> Remove Product | | | |

##### <User> Compare Picture (UC\_U04)



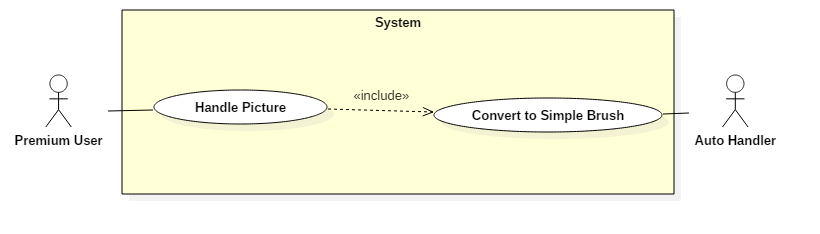
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_U05** | | | |
| **Use Case No.** | UC\_U04 | **Use Case Version** | 2.0 |
| **Use Case Name** | Compare Picture | | |
| **Author** | LongLH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows User to compare picture.   **Goal:**   * This function helps user compare the drawn picture with sample picture.   **Triggers:**   * User sends comparing picture command.   **Preconditions:**   * Actor signed in into as role “User”.   **Post Conditions:**   * **Success:** Similar percentages will be shown on the screen. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends command to compare picture. | System shows how much similar percentage on the screen. |   **Alternative Senario:** N/A  **Exceptions:** N/A  **Relationships:**   * Extend to Draw Picture: When user draws picture.   **Business Rules:**   * System save the user’s drawing to temporary folder. * System will then load the user’s drawing and the system’s picture, rescale to the same size and compare them using Euclidian distance. * After comparing, system will display similarity by a percentage number. | | | |
| Table 39: Use Case UC\_S03 - <Staff> Remove Product | | | |

##### <User> Buy License (UC\_U05)

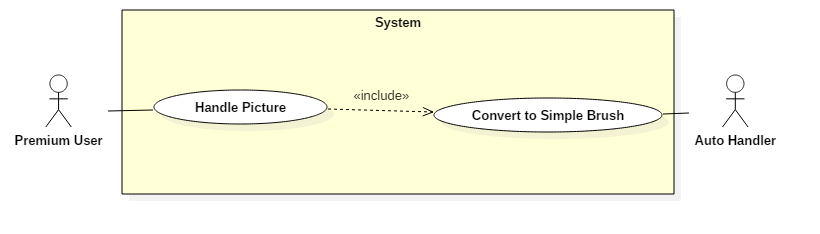


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_S05** | | | |
| **Use Case No.** | UC\_S05 | **Use Case Version** | 2.0 |
| **Use Case Name** | Buy License | | |
| **Author** | LongLH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows user to buy license.   **Goal:**   * This function helps user buy license in the application. After buying license, user is upgraded to Premium User.   **Triggers:**   * Actor sends buying license command.   **Preconditions:**   * Actor signed in into as role “User”.   **Post Conditions:**   * **Success:**.Payment successfully * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to license view. | System display a list of existing license:   * **No**: text, read only. * **License**: text, read only. * **Used Day**: text, read only. * **Price**: text, read only. | | 2 | User selects a specific license to buy. | System requires some information from User:   * **First name**: free text input, required, length 1- 20 characters. * **Last name**: free text input, required, length 1 – 20 characters. * **Card Number:** free text input, Integer, required. * **Zip Code:** free text input, Integer, required. * **Expiration Date:** required, date format: “DD/MM/YYYY”. * **Security Code:** free text input, Integer, required. | | 3 | User inputs required information. |  | | 4 | User sends command to buy license. | System validates inputted information.  [Exception 1, 2, 3, 4, 5, 6] | | 5 |  | System upgrades user to Premium User. |   **Alternative Senario:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No. | Cause | System Response | | 1 | Invalid first name format. | System shows error message: “First name cannot be blank and must be 1 – 20 characters long”. | | 2 | Invalid last name format. | System shows error message: “Last name cannot be blank and must be 1 – 20 characters long”. | | 3 | Invalid card number format. | System shows error message: “Card number cannot be blank”. | | 4 | Invalid zip code format. | System shows error message: “Zip Code cannot be blank”. | | 5 | Invalid expiration day format. | System shows error message: “Expiration day cannot be blank or not a valid date format”. | | 6 | Invalid security code format. | System shows error message: “Security Code cannot be blank”. |   **Relationships:**   * Include Pay.   **Business Rules:**   * After buying license, user is upgraded to Premium User. | | | |
| Table 39: Use Case UC\_S03 - <Staff> Remove Product | | | |

#### <Premium User> Overview Use Case

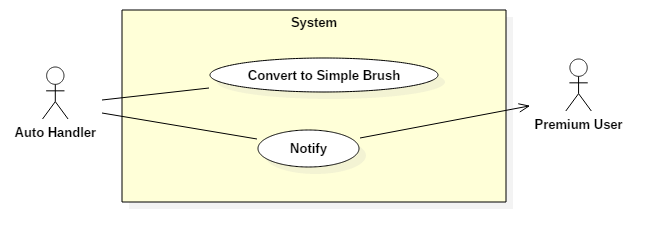


##### <Premium User> Handle Picture (UC\_PU01)

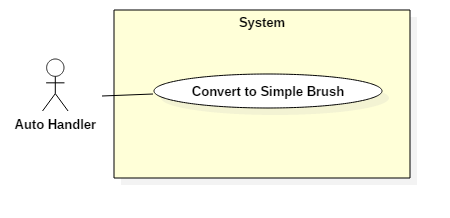


|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_PU01** | | | |
| **Use Case No.** | UC\_PU01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Handle Picture | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Premium User   **Summary:**   * This use case allows Premium User to handle picture.   **Goal:**   * .   **Triggers:**   * Premium User sends contributing picture command.   **Preconditions:**   * Actor signed in at role “Premium User”.   **Post Conditions:**   * **Success:** Picture will be converted to simple crush. * **Fail:** Show error message: “Picture is not converted”.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends command to contribute picture. | System opens Camera. | | 2 | User captures new picture or selects from device. |  | | 3 | User sends command to upload. | System uploads picture to server.  [Exception 1, 2] | | 4 |  | System converts specify picture to the 2D with simple brush.  System returns converted picture.  System shows message: “Picture is contributed successfully. We will verify your picture and notify to you when picture is approved”. |   **Alternative Scenario:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Invalid picture format. | System shows message: “Uploaded picture must be in jpg/jpeg/png format and must have at least 600x600px and less than 160Kb.” | | 2 | Unable to upload pictures. | System shows message: “System timeout, please try again later.” |   **Relationships:**   * Include Convert to Simple Brush: Auto Handler has to convert to Simple Brush whenever picture is contributed.   **Business Rules:**   * Picture will be compressed before being stored in the system. | | | |
| Table 39: Use Case UC\_PU01 - <Premium User > Handle Picture | | | |

#### <Auto Handler> Overview Use Case

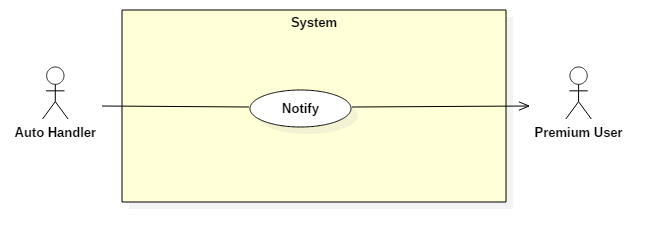


##### <Auto Handler> Convert to Simple Brush



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_AH01** | | | |
| **Use Case No.** | UC\_AH01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Convert to Simple Brush | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Auto Handler   **Summary:**   * Auto Handler will convert specify picture to the 2D with simple brush.   **Goal:**   * Auto Handler will converts specify picture to the 2D with simple brush.   **Triggers:**   * Auto Hander sends command to convert to simple brush.   **Preconditions:**   * Premium User must contribute picture or breakdown step.   **Post Conditions:**   * **Success:** Picture will be converted to simple crush. * **Fail:** Show error message: “Picture is not converted”.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Auto Hander converts to simple brush. |  |   **Alternative Scenario:** N/A  **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:** | | | |
| Table 39: Use Case UC\_AH01 - <Auto Handler> Convert to Simple Brush | | | |

##### <Auto Hander> Notify



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC\_PU01** | | | |
| **Use Case No.** | UC\_AH02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Notify | | |
| **Author** | HuanTNH | | |
| **Date** | 17/05/2017 | **Priority** | High |
| **Actor:**   * Auto Handler   **Summary:**   * Auto Handler will notify to Premium User.   **Goal:**   * .   **Triggers:**   * .   **Preconditions:**   * .   **Post Conditions:**   * **Success:** User’s smartphone receives notification. * **Fail:** System error log: “Send notification failed”.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User selects “Add Picture” button. | System opens Camera.  [Alternative 1, 2] |   **Alternative Scenario:** N/A  **Exceptions:** N/A  **Relationships:**   * Include Convert to Simple Brush: Picture will be converted to Simple Brush whenever a new picture is added by premium user.   **Business Rules:**   * Picture will be compressed before being stored in the system. * Picture will be converted to Simple Brush when Premium User adds picture. | | | |
| Table 39: Use Case UC\_AH02 - <Auto Handler> Notify | | | |

## Software System Attribute

### Usability

#### Graphic User Interface

* All the texts, labels and notifications must be written in Vietnamese.

#### Usability

* The system needs no more than 1 day of training for staffs.

### Reliability

* The system will notify premium user in case license’s user is nearly expired.

### Availability

* Server should have back-up method to make sure that if it having problems, all necessary data can be protected and restore easily.
* In online mode, system provides all features for user.

### Security

* Each role of user has a specific permission to interact with system.
* System always checks authorization and authentication before doing anything.
* Only admin can grant permission to other roles.

### Maintainability

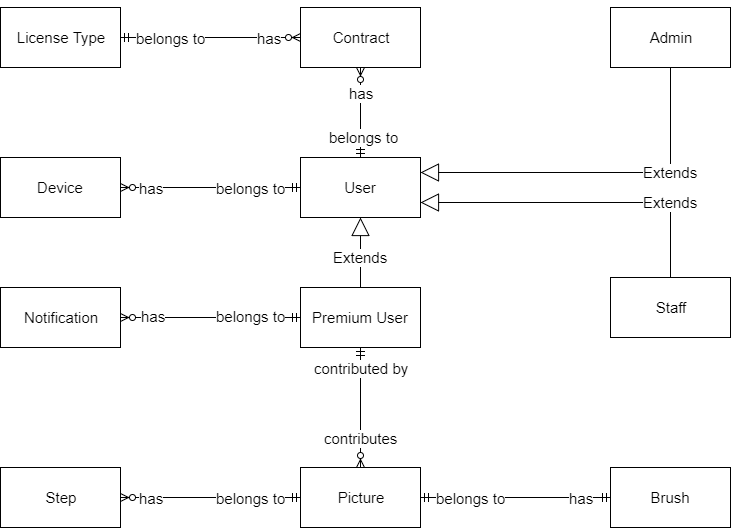
* The system is divided into separated modules.

### Portability

* Web application can be run on Chrome browser version 58 or later.
* User can use the mobile application on devices running Android 4 or later.

### Performance

## Conceptual Diagram

****

**Data Dictionary**

|  |  |
| --- | --- |
| **Entity Name** | **Description** |
| Admin | Person who manages the system. |
| Staff | Person who trains the system. |
| User | Contains information of all users in the system. |
| Premium User | Person who users the system with some extended features. |
| Picture | Contains information of all pictures in the system. |
| Step |  |
| Brush |  |
| License Type | Contains information of all license types in the system. |
| Notification | Contains information of all notifications in the system. |
| Device | Contains information of all devices in the system. |

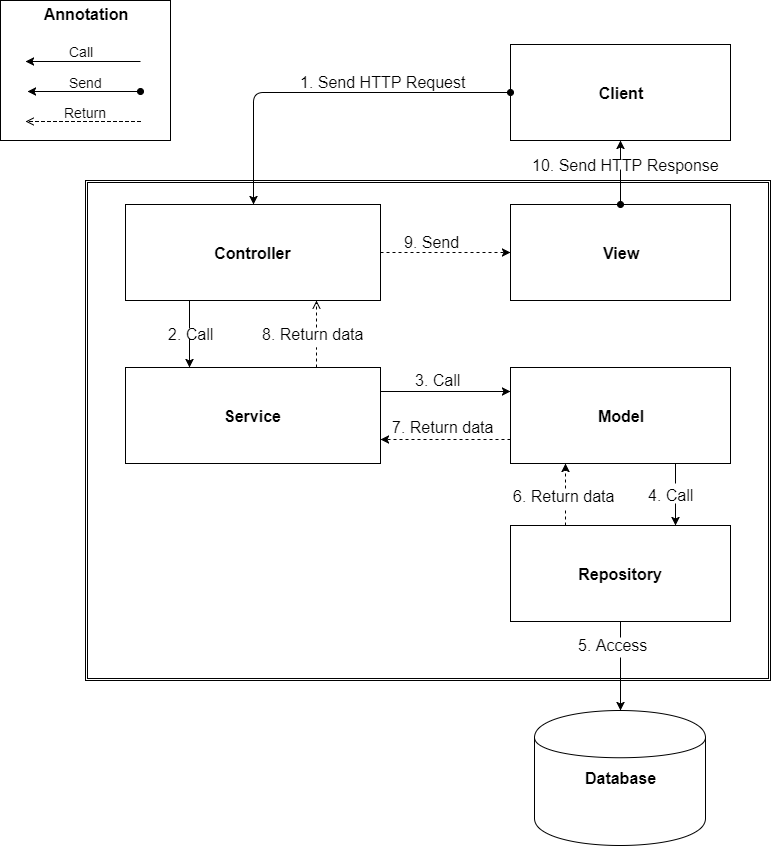
# Software Design Description

## Design Overview

* 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 1: gives an overall description of the system architecture design.
* Section 2: gives component diagrams that describe the connection and integration of the system.
* Section 3: gives the detail design description, which includes class diagram, class explanation, and sequence diagram to details the application functions.
* Section 4: gives the interface design description, which includes component interface, web application interface and mobile application design.
* Section 5: describe a fully attributed Entity Relationship Diagram.
* Section 6: describe the algorithms that apply to the system.

## System Architectural Design

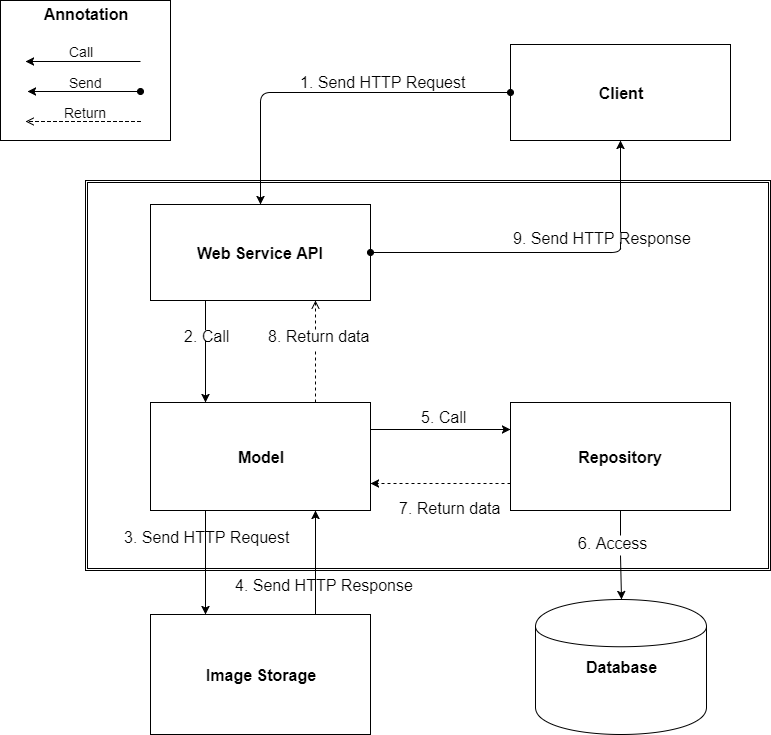
### Web Application Architecture Description



The web application uses MVC as architecture. We choose this architecture for a couple of reasons:

* The application will be divided into three clear parts.
* Testing each component in the architecture become easier.
* The existing of Model or Business Layer leverage testability and decoupling to the View.
* Repositoty is a pattern to abstract the data layer and manipulate the database; it also acts as an ORM. It combines both data and domain logic into object so that user can easily know how to update the data.

### Web Service Application Architecture Description



In Web Service, the system is developed under Restful architecture style. We choose this model for Web Service because of following advantages:

* In the future, our system will expand for other platform as well. Web service provides reusable API that can be called by any platform, so our system will be expand easily.
* We use Python because it provides comprehensive infrastructure support for developing algorithms applications. Python helps developer to create high performing, easily testable, reusable and loose coupling.

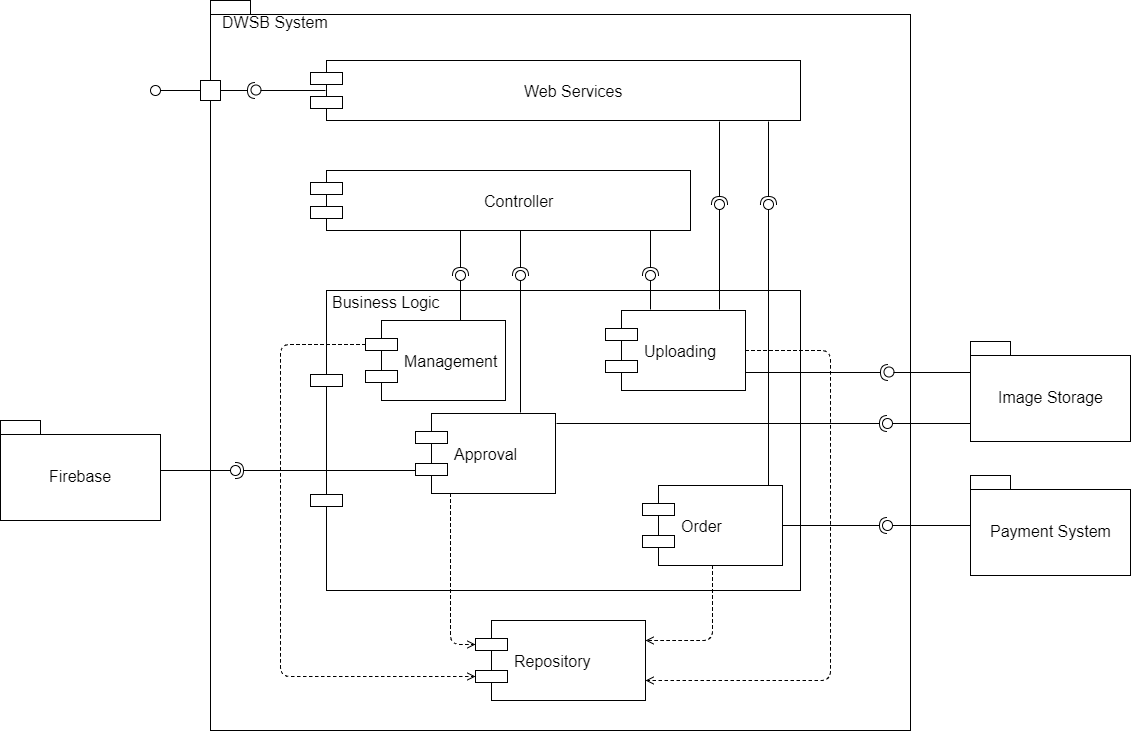
### Mobile Application Architecture Description

Mobile Architecture

Mobile application architecture will use MVVM. There is couple of reasons choosing this architect:

* It minimizes the number of dependency, helps increase loose coupling between application
* The View is passive and will be notified to update data
* The View Model will contain complex view logic; it also has no information about View. This help increase reusability of View Model
* The Model is where business logic sit in; it also performs API requests to backend application
* Develop Mobile Application using MVVM have increase productivity and reduce side effects

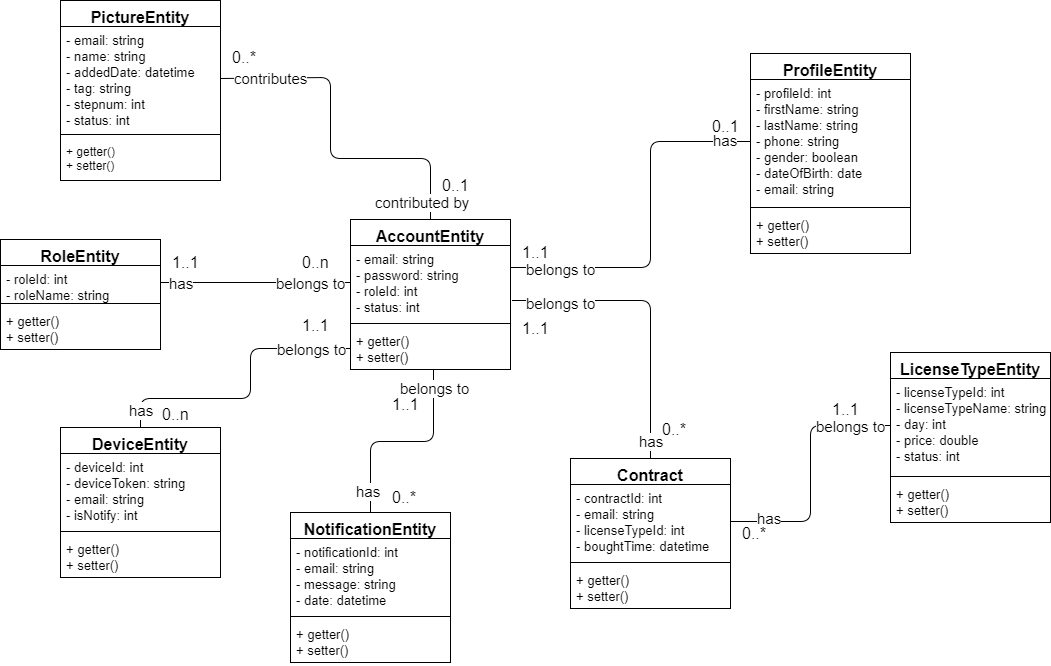
## Component Diagram



|  |  |
| --- | --- |
| **Name** | **Description** |
| DWSB System | Drawing with Simple Brush system. |
| Web Service | Component is used to provide API for mobile application to interact with the system. |
| Controller | Component to handle HTTP request. |
| Business Logic | Common services are used to handle system’s business operations. |
| Management | Component to manage system data. |
| Uploading | Component to upload picture. |
| Approval | Component to approve picture. |
| Order | Component to handle making order and interact with payment system. |
| Repository | Abstract data layer to interact with database system. |
| Firebase | External Component to send notification to user. |
| Payment System | External component for handle purchasing. |
| Image Storage | External component to storing image. |

## Detailed Description

### Class Diagram



|  |  |  |
| --- | --- | --- |
| **Class name** | **Mapping entity with conceptual diagram** | **Description** |
| AccountEntity | Account | Contains the information of account. |
| PictureEntity | Picture | Contains the information of picture. |
| LicenseTypeEntity | LicenseType | Contains the information of license type. |
| ContractEntity | Contract | Contains the information of contract. |
| DeviceEntity | Device | Contains the informtaion of device. |
| NotificationEntity | Notification | Contains the information of notification. |
| RoleEntity | N/A | Not exist in conceptual diagram. But needed in class diagram to contain the information of role. |
| ProfileEntity | N/A | Not exist in conceptual diagram. But needed in class diagram to contain the information of profile. |
| Not Implement | Brush | Exist in conceptual diagram. But not existed in class diagram because it had declare in configuration file. |
| Not Implement | Step | Exist in conceptual diagram. But not existed in class diagram because it had declare in configuration file. |

### Class Diagram Explanation

#### Role

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| RoleId | Integer | Private | Unique id of role. |
| RoleName | String | Private | Name of role. |

**Methods**

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

#### Profile

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ProfileId | Integer | Private | Unique id of profile. |
| Email | String | Private | Email in account’s profile. |
| FullName | String | Private | Full name in account’s profile. |
| Phone | String | Private | Phone in account’s profile. |
| Gender | Boolean | Private | Gender in account’s profile. |
| DateOfBirth | Date | Private | Date of Birth in account’s profile. |

**Methods**

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

#### Account

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Email | String | Private | Email of account. |
| Password | String | Private | Password of each account. |
| RoleId | Integer | Private | Role of account in the system. |
| Status | Integer | Private | Status of account. |

**Methods**

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

#### Picture

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Email | String | Private | Email of picture. |
| Name | String | Private | Name of picture. |
| Tag | String | Private | Tag of picture. |
| AddedDate | DateTime | Private | Added Date of picture. |
| Stepnum | Integer | Private | Number the steps of pictures. |
| Status | Integer | Private | Status of picture |

**Methods**

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

#### License Type

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| LicenseTypeId | Integer | Private | Unique id of license type. |
| LicenseTypeName | String | Private | Name of license type. |
| Price | Double | Private | Price of license type. |
| Day | Integer | Private | Number the use days of license type. |
| Status | Integer | Private | Status of license type. |

**Methods**

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

#### Contract

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ContractId | Integer | Private | Unique id of contract. |
| LicenseTypeId | Integer | Private | Unique id of license type. |
| Email | String | Private | Email in account’s contract. |
| BoughtTime | DateTime | Private | Bought time in account’s contract |

**Methods**

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

#### Device

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| DeviceId | Integer | Private | Unique id of device. |
| DeviceToken | String | Private | Token of device. |
| Email | String | Private | Email in account’s device. |
| IsNotify | Integer | Private | Status of device’s notification. |

**Methods**

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

#### Notification

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| NotificationId | Integer | Private | Unique id of notification. |
| Email | String | Private | Email of each account. |
| Message | String | Private | Message of each notification. |
| Date | DateTime | Private | Date of each notification. |

**Methods**

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

### Interaction Diagram

#### Mobile Application

##### Unauthorized User

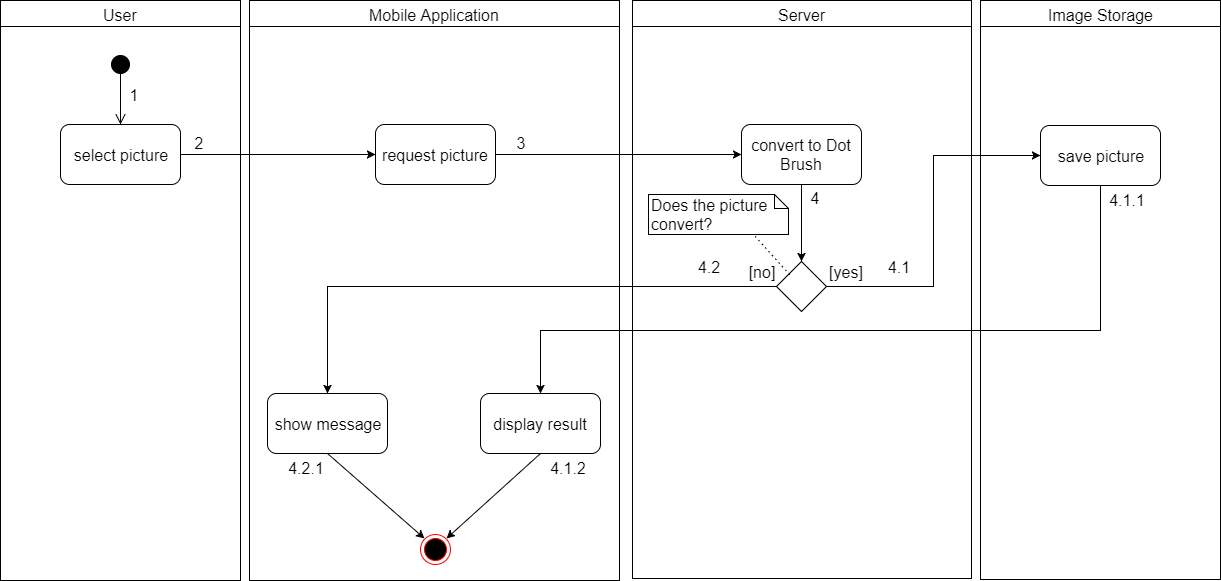
###### < Unauthorized User> Login

Summary: This diagram shows the process of unauthorized user login into the system using mobile application.

##### Authorized User

###### <Authorized User> Dot Brush

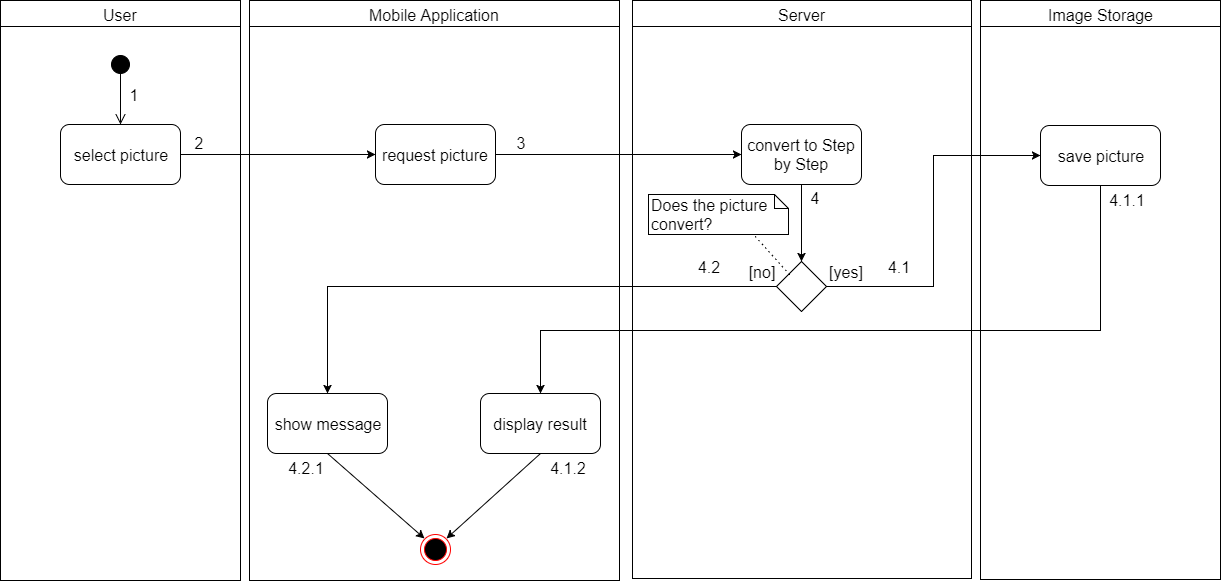
Summary: This diagram shows the process of authorized user draw sample picture with dot brush using mobile application.



|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Select picture | 3. Request picture |
| 4. Convert to Dot Brush | 4.1. Able to convert picture | 4.1.1. Save picture |
| 4.1.2. Display result and finish | 4.2. Unable to convert picture | 4.2.1. Show message and finish |

###### <Authorized User> Step By Step

Summary: This diagram shows the process of authorized user draw sample picture in step by step using mobile application.



|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Select picture | 3. Request picture |
| 4. Convert to Step by Step | 4.1. Able to convert picture | 4.1.1. Save picture |
| 4.1.2. Display result and finish | 4.2. Unable to convert picture | 4.2.1. Show message and finish |

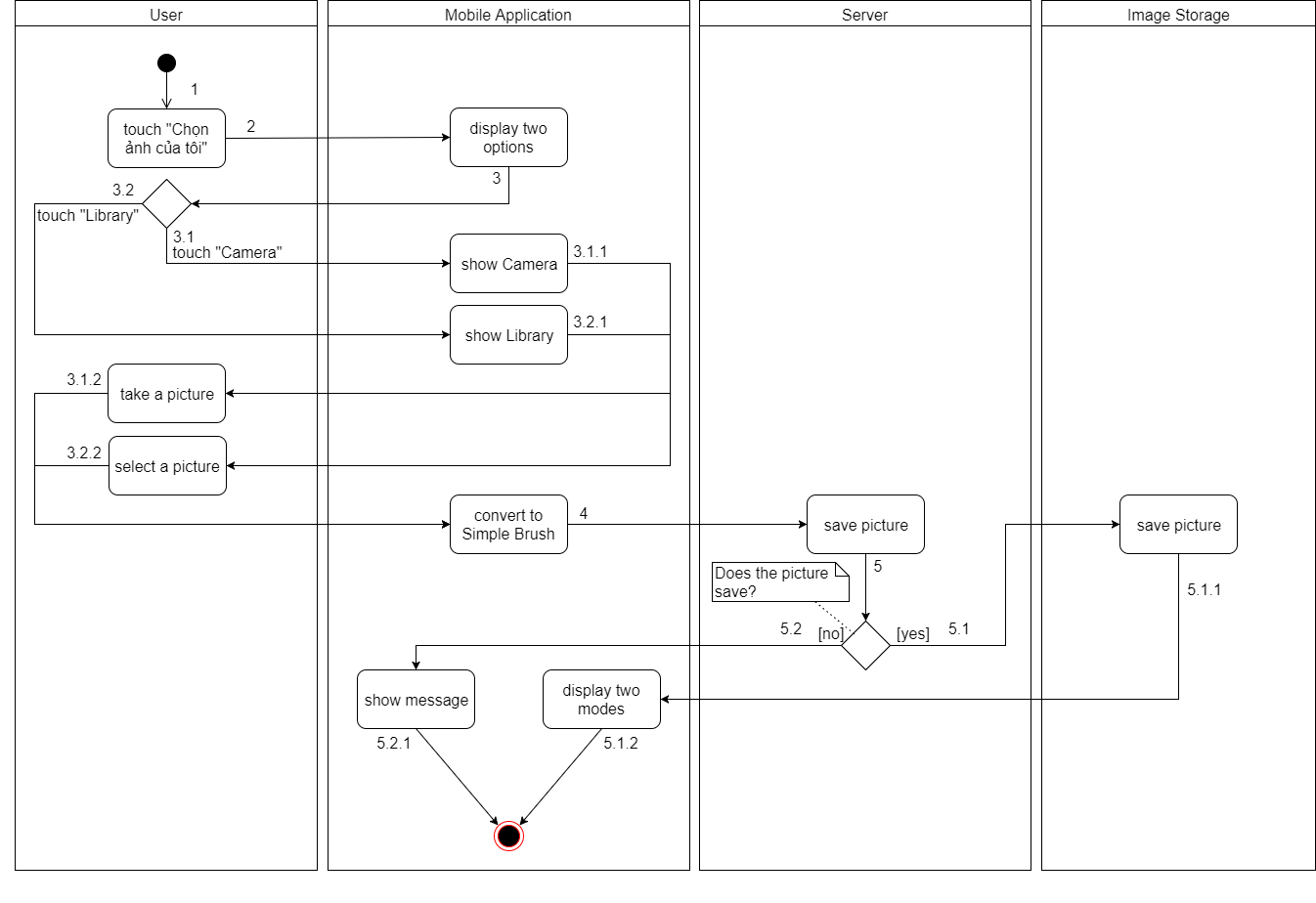
###### <Authorized User> Compare Picture

Summary: This diagram shows the process of authorized user compare picture using mobile application.

##### Premium User

###### <Premium User> Upload Picture

Summary: This diagram shows the process of premium user upload picture using mobile application.



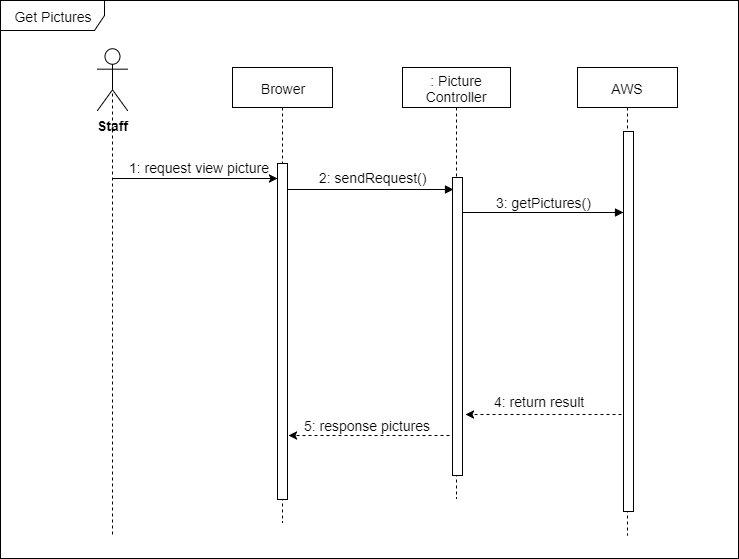
|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Touch “Chọn ảnh của tôi” | 3. Display two options |
| 3.1. Touch “Camera” | 3.1.1. Show Camera | 3.1.2. Take a picture |
| 3.2. Touch “Library” | 3.2.1. Show Library | 3.2.2. Select a picture |
| 4. Convert to Simple Brush | 5. Save picture | 5.1. Able to save picture |
| 5.1.1. Save picture | 5.1.2. Display two modes and finish | 5.2. Unable to save picture |
| 5.2.1. Show message and finish |  |  |

#### Web Application

##### Staff

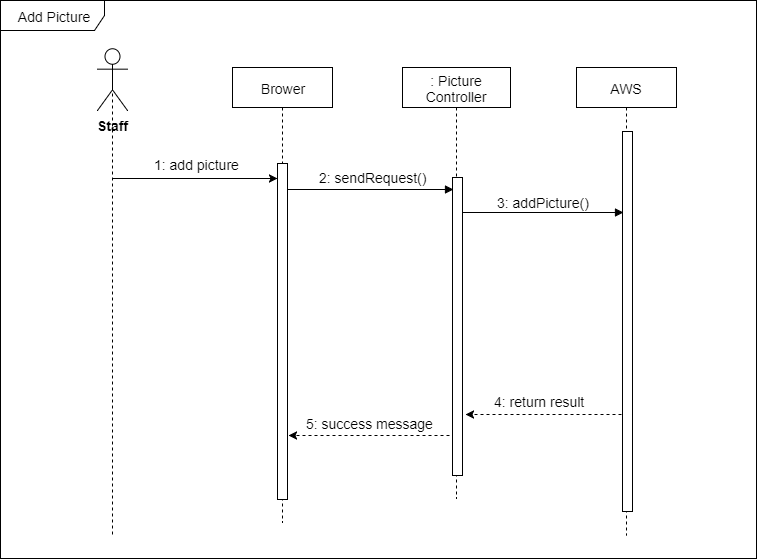
###### <Staff> Get Pictures

Summary: This diagram shows the process of staff getting pictures using the web application.



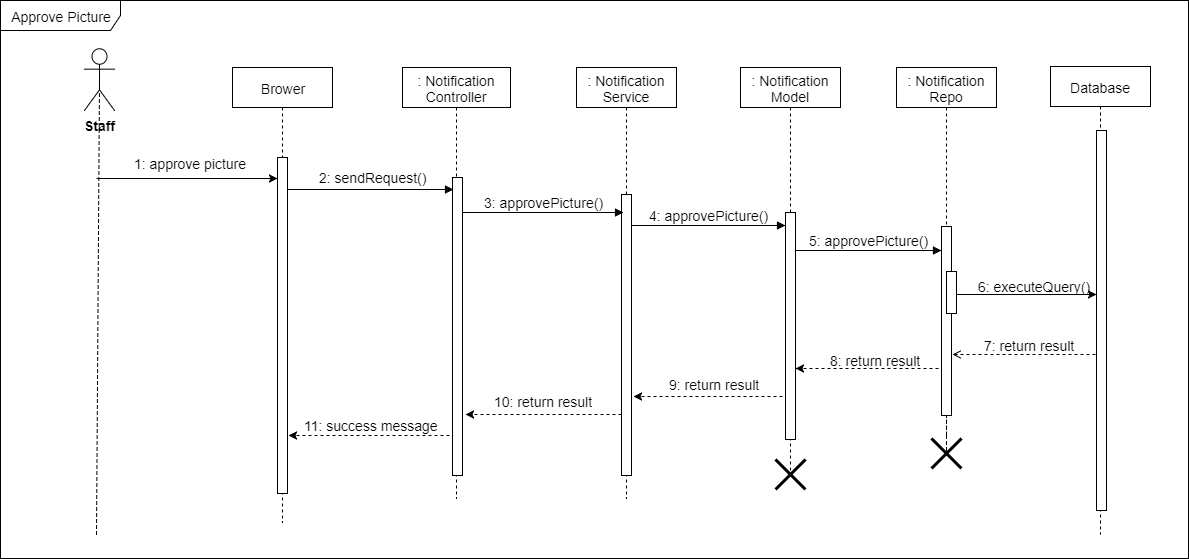
###### <Staff> Add Picture

Summary: This diagram shows the process of staff adding a picture using the web application.



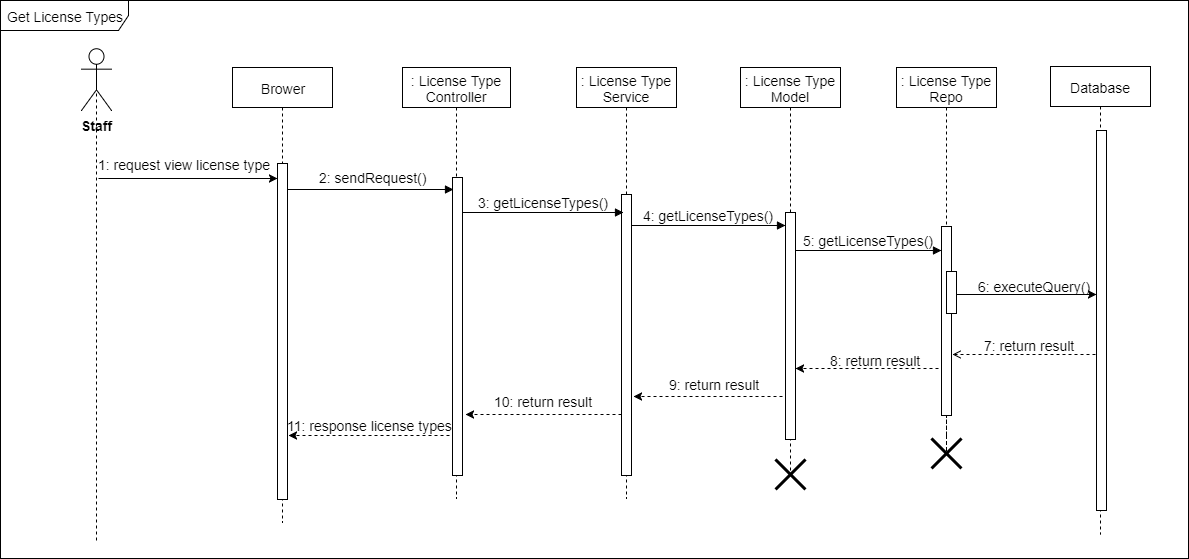
###### <Staff> Approve Picture

Summary: This diagram shows the process of staff approving the picture after user has contributed using the web application.



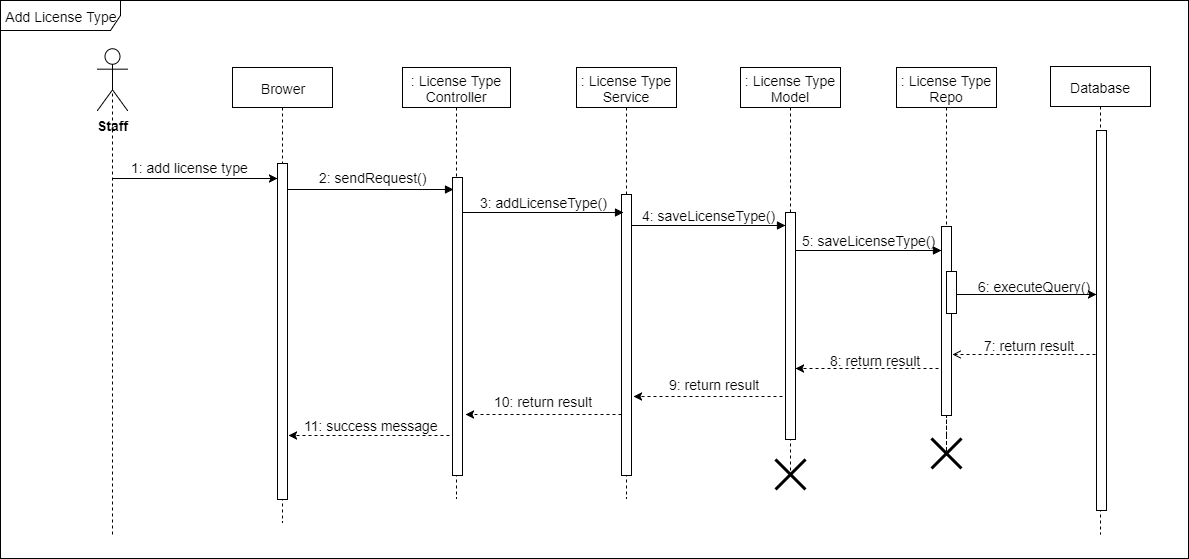
###### <Staff> Get License Types

Summary: This diagram shows the process of staff getting license types using the web application.



###### <Staff> Add License Type

Summary: This diagram shows the process of staff adding a license type using the web application.



## Interface

### Component interface

#### Module of Picture API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Get all pictures | String status | JSON list of pictures |  | JsonProcessingException |
|  | Get dot picture | String name | JSON includes url, name |  | JsonProcessingException |
|  | Get step picture | String name | JSON includes urls, name |  | JsonProcessingException |
|  | Get user’s pictures | String email | JSON list of pictures |  | JsonProcessingException |
|  | Upload picture | String base64 | JSON upload picture |  | JsonProcessingException |
|  | Contribute picture | String email, String name | JSON contribute picture |  | JsonProcessingException |
|  | Compare picture | String name, String base64 | JSON includes name, accuracy |  | JsonProcessingException |
|  | Search picture | String tags | JSON list of pictures |  | JsonProcessingException |

#### Module of Account API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Login with google | String email |  |  | JsonProcessingException |
|  | Register with google | String fullname, String email |  |  | JsonProcessingException |

#### Module of Profile API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Update Profile | String fullName, String phone, String gender, String dateOfBirth |  |  | JsonProcessingException |

#### Module of License Type API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Get all license types |  | JSON list of license types |  | JsonProcessingException |

#### Module of Contract API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Add contract | String email, String licenseTypeId |  |  | JsonProcessingException |

#### Module of Device API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Add device | String deviceToken, String email |  |  | JsonProcessingException |

#### Module of Notification API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
|  | Get notification | String email | JSON list of messages |  | JsonProcessingException |

### User Interface Design

#### Mobile Application Design

##### Unauthorized User

###### <Unauthorized User> Login

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | Đăng nhập với Google | Log-in to the system via Google account | N/A | Redirect user to Google login page. |
| **2** | Free draw | View free draw | N/A | Transfer to view free draw page |
| **3** | Library | View library | N/A | Transfer to view library page |
| **4** | Your picture | View my library | N/A | Transfer to view your picture page |
| **5** | Profile | View profile | N/A | Transfer to view profile page |
| **6** | Notification | View notification | N/A | Transfer to view notification page |
| **7** | Nâng cấp tài khoản | View license type | N/A | Transfer to view upgrade account page |

##### Authorized User

###### <Authorized User> Home Pape



**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | Profile | View profile | N/A | Transfer to view profile page |
| **2** | Đăng xuất |  | N/A |  |
| **3** | Nâng cấp tài khoản | View license type | N/A | Transfer to view upgrade account page |
| **4** | Free Draw | View free draw | N/A | Transfer to view free draw pape |
| **5** | Library | View library | N/A | Transfer to view library pape |
| **6** | Your Library | View your library | N/A | Transfer to view your library pape |

###### <Authorized User> Library

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Search | Fill in tag | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **2** | Select picture | Select picture based on user selection | N/A | Displays 2 modes.   * Một màn hình * Hai màn hình |
| **3** | Vẽ bằng ảnh của tôi | User draws from their uploaded picture | N/A | Display 2 selections:   * Từ máy ảnh * Từ thư viện ảnh |

###### <Authorized User> Taking Picture

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | Một màn hình | Take single screen mode | N/A | Display sample picture with dot brush. |
| **2** | Hai màn hình | Take dual screen mode | N/A | System will display two screens.  - Left Screen: show sample picture in step by step with simple brush.  - Right Screen: show white page for user draw. |

###### <Authorized User> Single Screen Mode

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** |  |  |  |  |

###### <Authorized User> Dual Screen Mode

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** |  |  |  |  |

###### <Authorized User> License Type

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | Choose license type | Select license type based user choice | N/A | Transfer to view payment |

##### Premium User

###### <Premium User> Choose Your Picture

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
|  | Từ máy ảnh | Take new pictures using device camera | N/A | Open device camera |
|  | Từ thư viện ảnh | Select photos in device | N/A | Open photo album |

###### <Premium User> Your Library

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Search | Fill in tag | No | Yes | Textbox | String | N/A |

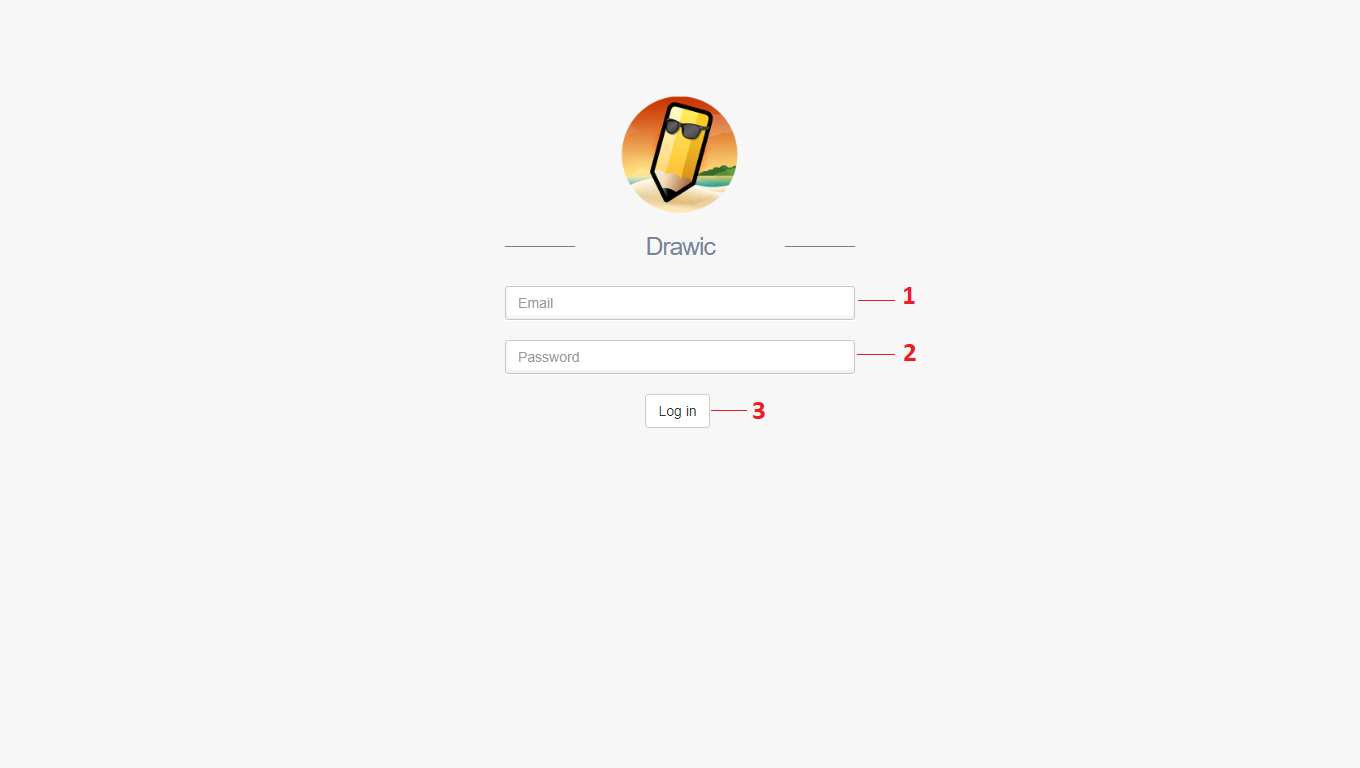
**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **2** | Select picture | Select picture based on user selection | N/A | Displays 2 modes.   * Một màn hình * Hai màn hình |
| **3** | Vẽ bằng ảnh của tôi | User draws from their uploaded picture | N/A | Display 2 selections:   * Từ máy ảnh * Từ thư viện ảnh |

#### Web Application Design

##### Admin

###### <Admin> Login



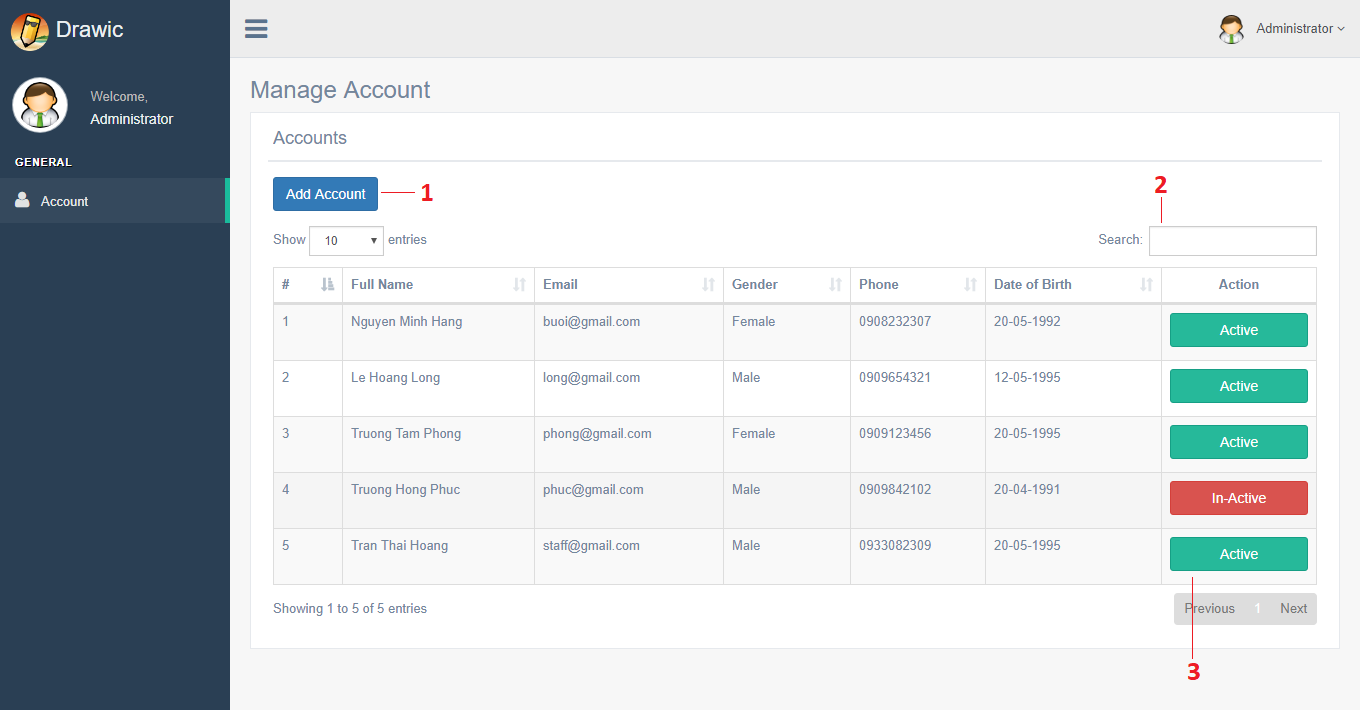
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Email | Fill in email | No | Yes | Textbox | String | N/A |
| **2** | Password | Fill in password | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **3** | login | Log in to the system | N/A | Transfer to specific page with specific account. |

###### <Admin> Get Account



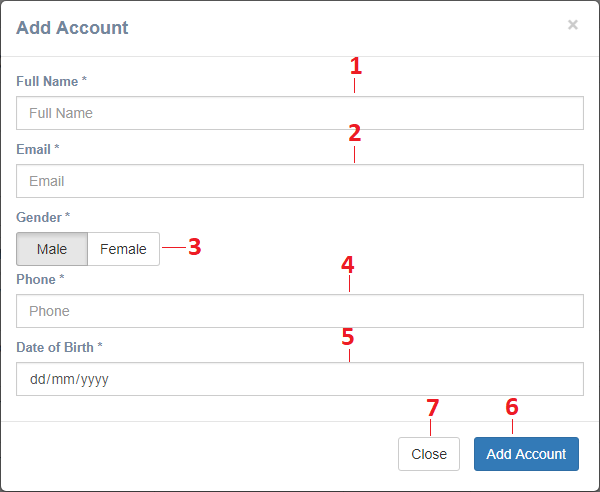
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Search | Fill in keyword | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **2** | addAccount | Open add account modal. | N/A | Show add account modal. |
| **3** | enableAccount | Open enable account modal. | N/A | Show enable account modal. |

###### <Admin> Add Account



**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Full Name | Fill in full name | No | Yes | Textbox | String | N/A |
| **2** | Email | Fill in email | No | Yes | Textbox | String | N/A |
| **3** | Gender | Choose gender | No | Yes | Label | String | N/A |
| **4** | Phone | Fill in phone | No | Yes | Textbox | String | N/A |
| **5** | Date of Birth | Fill in date of birth | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **6** | addAccount | Add a account to the system. | N/A | Account is added to the system. |
| **7** | close | Discard add account. | N/A | Close add account modal. |

##### Staff

###### <Staff> Get Pictures

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Search | Fill in keyword | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **2** | addPicture | Open add picture modal. | N/A | Show add picture modal. |
| **3** | updatePicture | Open update picture modal. | N/A | Show update picture modal. |

###### <Staff> Add Picture

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | Search | Fill in keyword | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

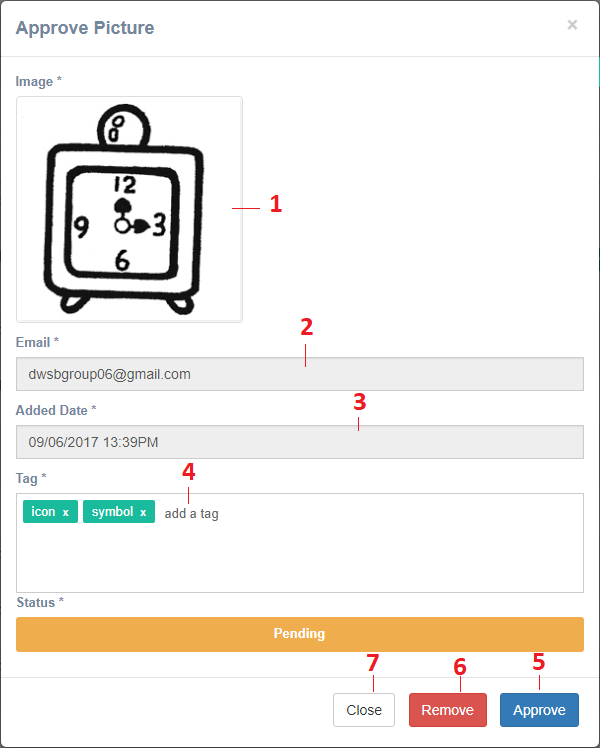
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **2** | addPicture | Add a picture to the system. | N/A | Picture is added to the system. |
| **3** | close | Discard add picture. | N/A | Close add picture modal. |

###### <Staff> Get Notification

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | approvePicture | Open approve picture modal. | N/A | Show approve picture modal. |

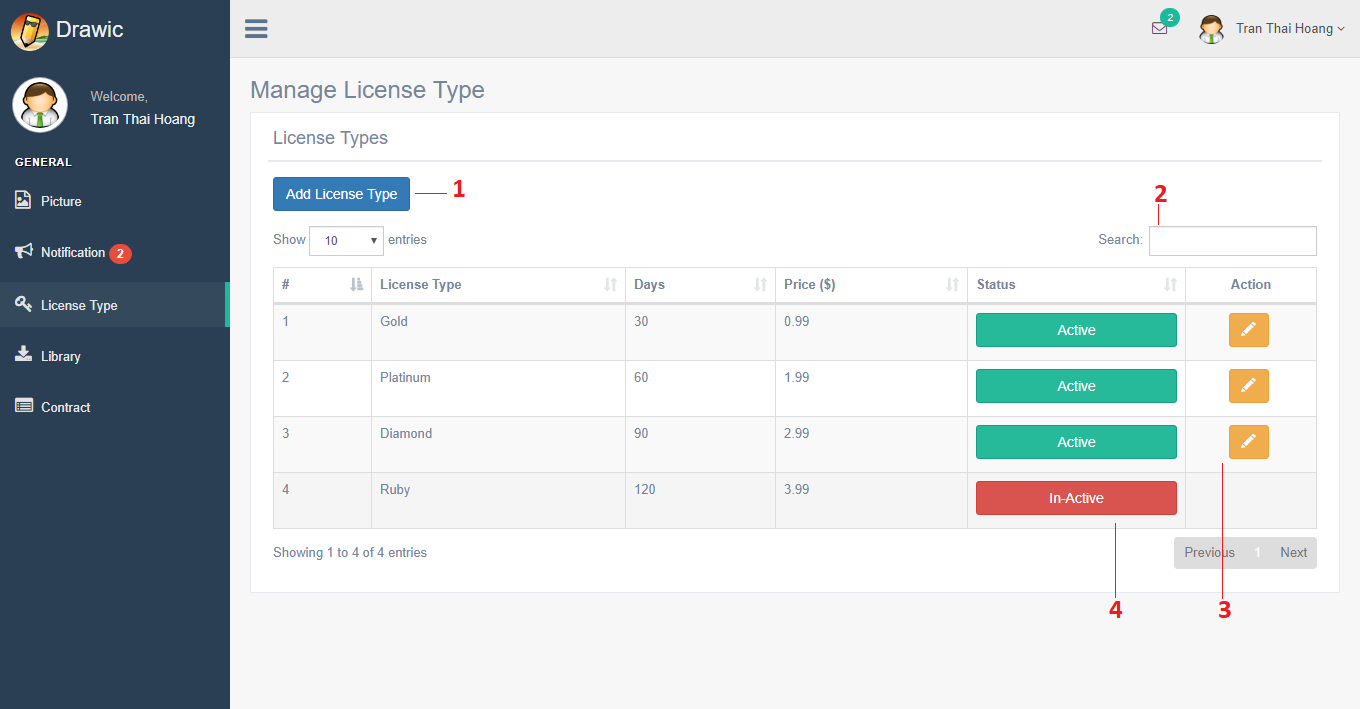
###### <Staff> Approve Picture



**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **5** | approvePicture | Approve the picture. | N/A | Picture is approved and available for users to interact. |
| **6** | removePicture | Remove the picture from the system. | N/A | Picture is removed from system |
| **7** | close | Discard update picture. | N/A | Close update picture modal. |

###### <Staff> Get License Type



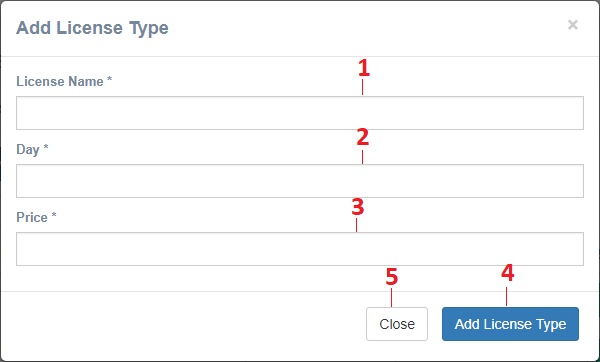
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **2** | Search | Fill in keyword | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | addLicenseType | Open add license type modal. | N/A | Show add license type modal. |
| **3** | updateLicenseType | Open update license type modal. | N/A | Show update license type modal. |
| **4** | enableLicenseType | Open enable license type | N/A | Show enable license type modal. |

###### <Staff> Add License Type



**Fields**

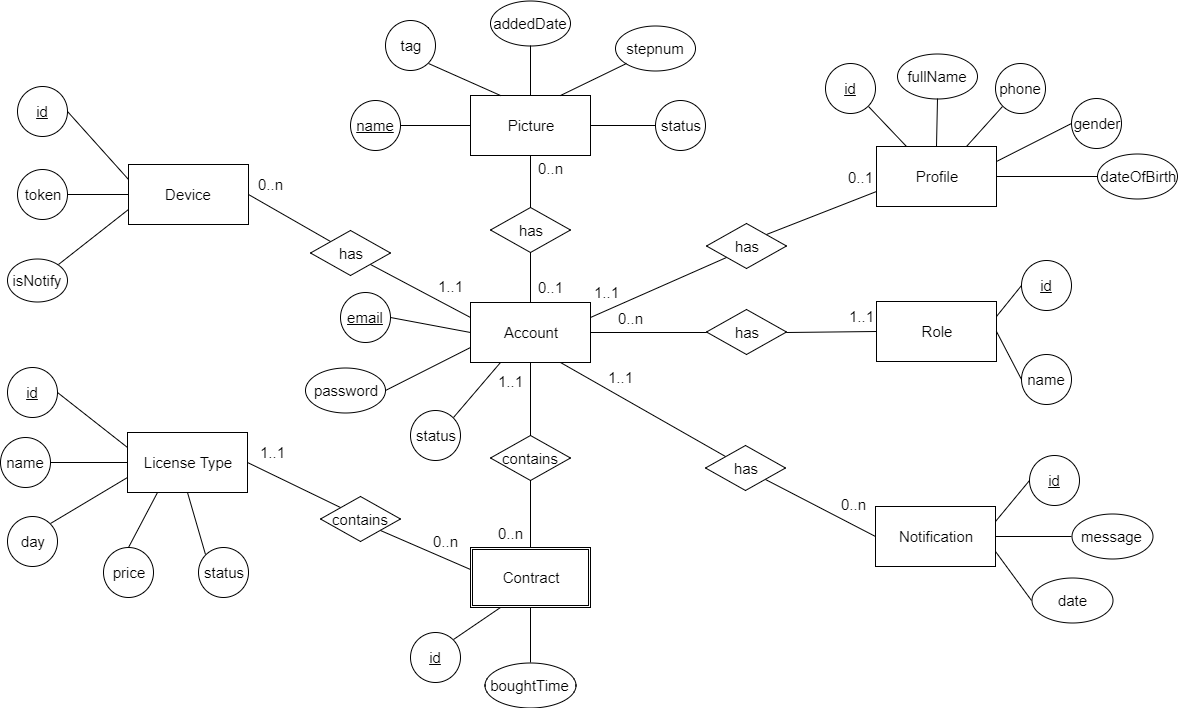
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| **1** | License Name | Fill in license name | No | Yes | Textbox | String | N/A |
| **2** | Day | Fill in day | No | Yes | Textbox | String | N/A |
| **3** | Price | Fill in price | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| **1** | addLicenseType | Add a license type to the system. | N/A | License type is added to the system. |
| **2** | close | Discard add license type. | N/A | Close add license type modal. |

## Database Design

### Entity relationship diagram (ERD)



### Data Dictionary

|  |  |
| --- | --- |
| **Entity Data Dictionary: describe contents of all entities** | |
| **Entity Name** | **Description** |
| Profile | Contains information of all profiles in the system. |
| Role | Contains information of all roles in the system. |
| Account | Contains information of all accounts in the system. |
| Picture | Contains information of all pictures in the system. |
| License Type | Contains information of all license types in the system. |
| Notification | Contains information of all notifications in the system. |
| Device | Contains information of all devices in the system. |
| Contract | Contains information of all contracts in the system. |

## Algorithms

### Web Service Application

#### Adaptive Threshholding using The Integral Image

##### Definition

##### Problem Definition

##### Solution

##### References

#### Make Midpoint - Image

##### Definition

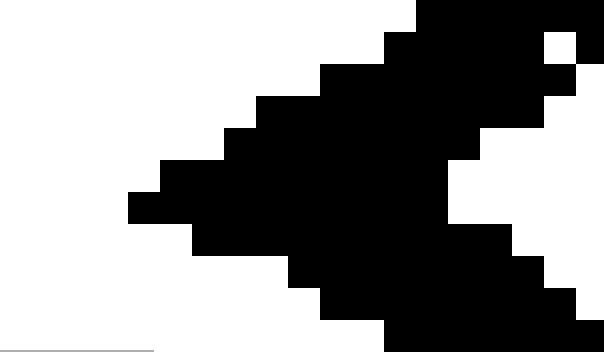
* The algorithm proposed a method to reduce input image to much simpler version for faster and more accurate for making dash image.

##### Problem Definition

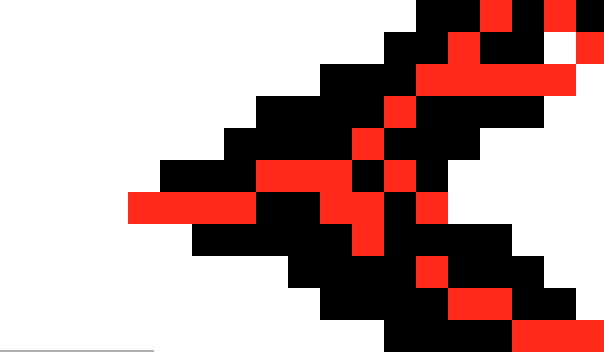
* Making dash image involved a complex and time consuming process for each black point in an image.
* A major problem is how to extract a list of points that:
  + Consisted only black points.
  + Reduced number of point while retaining and/or improving correctness of dash making algorithm.

##### Solution

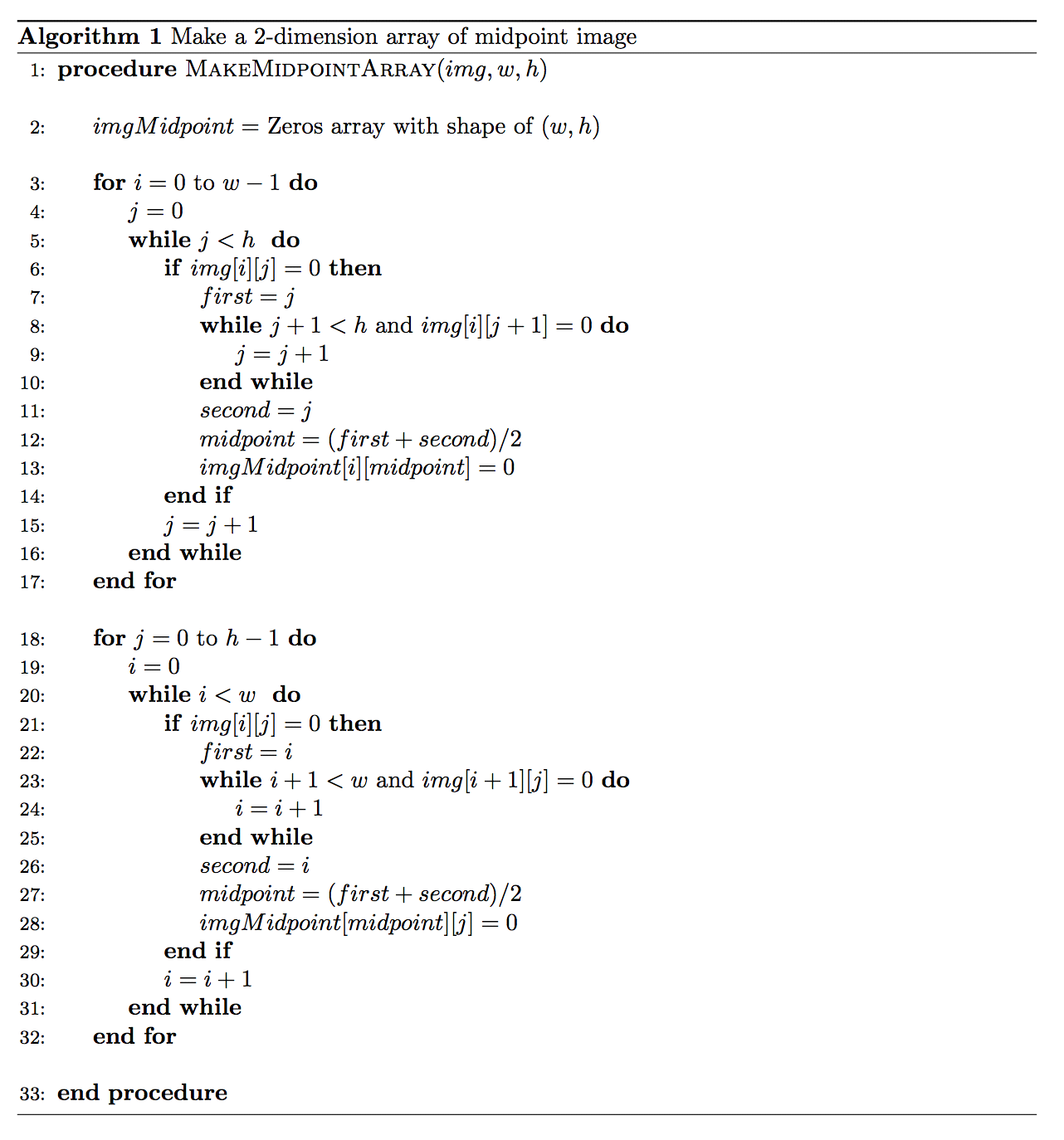
* Our solution is to make a list of midpoints.
* A midpoint is the middle of a connected horizontal or vertical line region:
  + One connected horizontal or vertical line region only have on midpoint.
  + Example: Consider this image (zoomed in to pixels):



* + This image will produce a list of midpoints (in red pixels):



##### Pseudocode

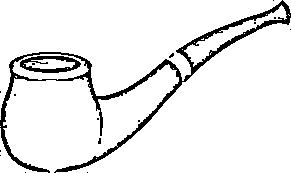


##### Complexity and Effect

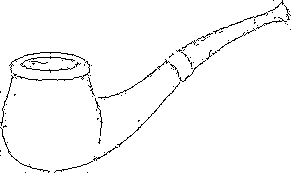
* Complexity:

*.*

* Effect:
  + In the worst-case scenario, the algorithm reduces the number of point from to .
  + We tested our algorithm with 10 different images and it reduces about 20x to 50x times the number of point.
* Sample result:
  + Input image:



* + Output list (draw on blank image):



#### Delete from point with radius

#### Make Dash Image

#### Find Connected List

#### Shi-Tomasi Conner Detector

#### Make Step Images

#### Image Comparison Algorithms

#### Image Auto Algorithms

### Mobile Application

#### Calculates Scale Gestures

##### Definition

* Purpose of method is found appropriate ratio to scale layer refer to gestures of fingers in any point on Canvas.

##### Define the problem

* User pinch to zoom in a point on Canvas by two fingers. There are two values that need to be defined:
  + Spacing between two fingers.
  + Point to pinch zoom.

##### Calculate

* Build equation for calculating distance of 2 fingers:
  + Distance:
* Determine of center Point of two fingers:
  + Center P:

##### Solution

* After calculate values in previous step, we can determine scale gestures at point P:

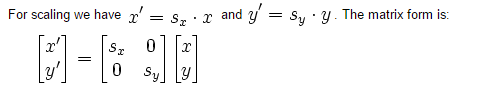
#### Build Transformation Matrix

##### Definition

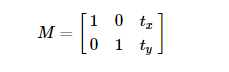
* Transformation matrix is a special matrix that can describe 2d and 3d transformations. Propose this method to applydifferent geometric transformation to images like translation, scalling. Both of them is necessary when pinch to zoom in any point.

##### Define the problem

* Scaling is just resizing of the image.



* Translation is the shifting of object's location. If you know the shift in (x, y) direction, let it be (tx, ty), you can create the transformation matrix M as follows:



##### Solution

* According the result of Calculate Scale Gestures, we can pinch – zoom in any point on Canvas with two value of matrix M:
* M.scale and M.translate to center point P between two fingers of users.

##### Complexity

* Complexity:

##### References

<https://developer.android.com/reference/android/graphics/Matrix.html>

#### Making Sense of Multi Touch

##### Definition

* To track the touch events from multiple pointers user use.Methods to identify the index of the pointer and the touch event which happened for this pointer.
* Such as: Zoom, Draw, Drag...

##### Define the problem

* Built-in draw and zoom:
  + Draw: use single touch.
  + Pinch-zoom: use multitouch.
* Intergrate Drawing and Zooming, the line of drawing path and path dimensions must be calculated according to the appropriate coordinates so that the drawing position is not changed before and after the zoom.

##### Solution

* Because the area around the Point P is zoom in while the size of Screen and Bitmap on Canvas is fixed. So calculate an equation before- after zoom in, zoom out:
* Solve the equation above, we can get the position of Point X in Canvas and can calculate line of XY.

##### Complexity

* Complexity:

##### References

<https://android-developers.googleblog.com/2010/06/making-sense-of-multitouch.html>

#### Equation measure on Position of Multilayers

##### Definition

* Purpose of method is allow to sync all layers including canvas can zoom in – zoom out at the same time, same point. And Canvas still draws on it.

##### Define the problem

* Fit to screen: Each layer have a difference coordinate with a matrix. So need to measure the appropriate values (dimension) so that all layers can pinch to zoom in all point on screen.
* Put the content of layer in the center of Matrix.

##### Solution

* Equation fit to screen:
  + let kx is the scaleLayer in horizontal x
  + let ky is the scaleLayer in vertical y
  + Matrix of layer
* Equation put layer into center of Matrix:

##### References

* <https://developer.android.com/reference/android/graphics/Matrix.ScaleToFit.html>

#### Convert Pencil Image Algorithms

* The algorithm proposed a method thatconvert into simple pencil sketch image from an input standard image.

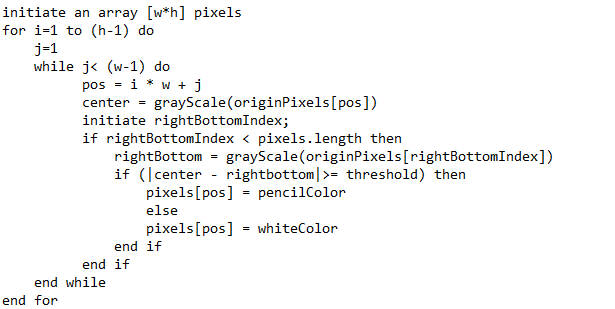
##### Problem Definition

* The image processing algorithm is performed on a black-while color so it is necessary to convert the image to a black-white image before processing.
* Pixel analysis is based on a given threshold. Although the speed of processing is fast but the accuracy is depend on threshold.

##### Solution

* Convert picture to gray- scale: refer to RGB channel color to convert.
* The algorithm calculates over two pixels, one pixel in focus and one pixel in the bottom of right of the pixel. If it is greater than threshold, set the pixel value to the pencil color and set the background color to white.
* Use dynamic variables to make adjustments image density directly.
* Vẽ hình

##### Pseudocode



##### Complexity and Effect

* Complexity:

*.*

* Effect:
  + The processing speed of an image has 600 x 800 dimensions when the threshold change is approximately 0.1 second.
  + We tested our algorithm with 10 different images Samsung Note 5. 4GB Ram, Android 6.0.1 ([Marshmallow](https://www.thegioididong.com/hoi-dap/android-6-marshmallow-co-gi-hot-726146)).
* Sample results:
  + Input image:



* + Output list depend on threshold:



* Threshold is decrease from left to right. In some case, need to increase threshold to maintain stroke for image. Thus, threshold is adjusted by user

# System Implementation & Test

## Introduction

### Overview

This section describes the approach and methodologies used by our group to plan, organize and manage the testing of DWSB system. It provides in detail all necessary information about the implementation and testing procedure of the system included test plans, test cases, test result, test environments, pass/fail criteria and risks estimations as well as a checklist to cover all possible cases.

### Test Approach

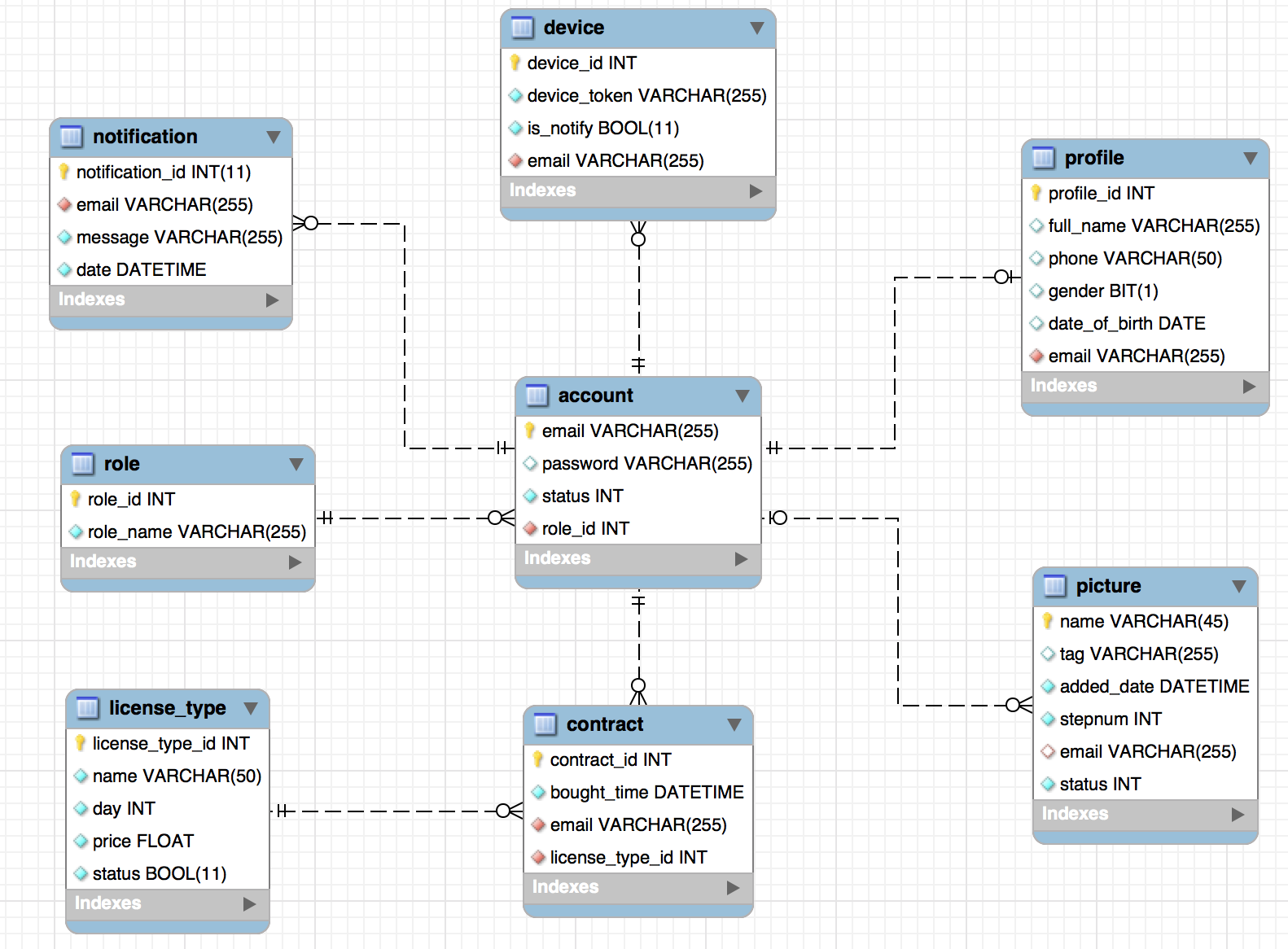
Goal: Test all features in the whole DWSB system based on the core flow.

Method: black-box testing.

Technique: check list.

## Database Relationship Diagram

### Physical Diagram



### Data Dictionary

**Data dictionary**

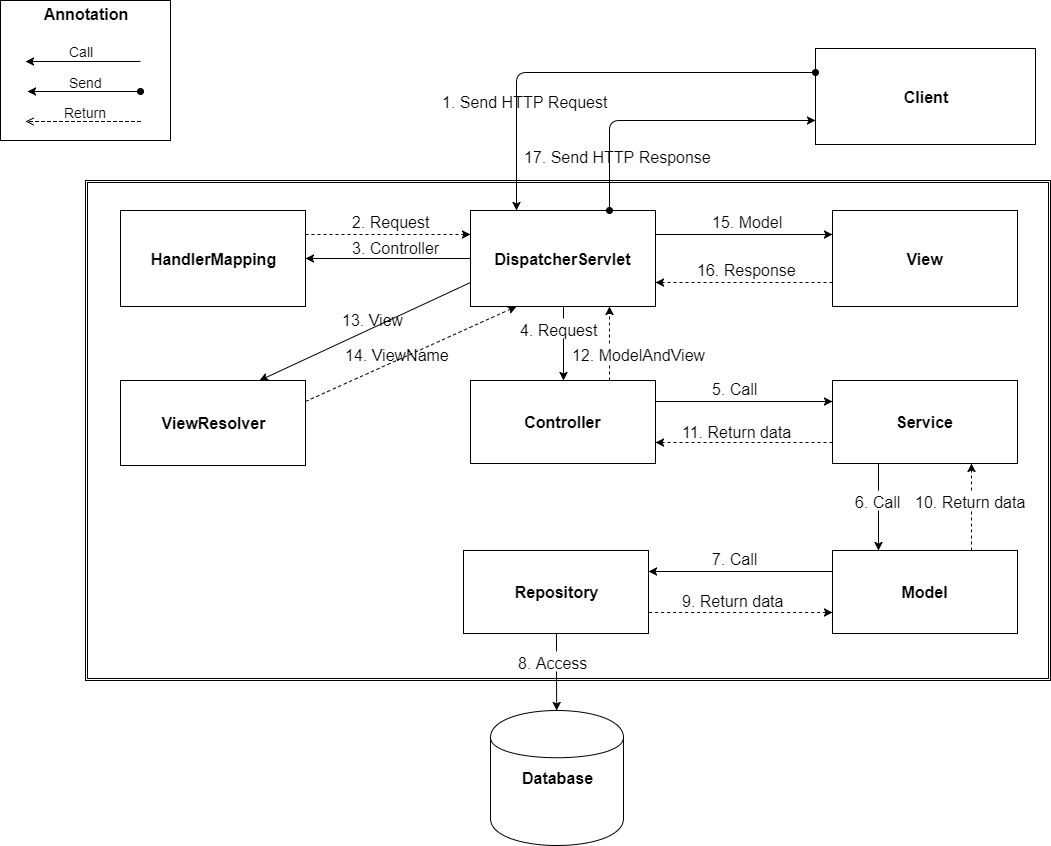
|  |  |  |
| --- | --- | --- |
| **Class Dictionary: Describe content of all table** | | |
| **Class name** | **Description** | **Mapping with ERD diagram** |
| **Account** | Contains the information of account. | Account |
| **Profile** | Contains the information of profile’s account. | Profile |
| **Role** | Contains the information of role. | Role |
| **Picture** | Contains the information of picture. | Picture |
| **License Type** | Contains information of license type. | License Type |
| **Contract** | Contains the information of contract. | Contract |
| **Notification** | Contains the information of notification. | Notification |
| **Device** | Contains the information of device. | Device |

**Attribute Data Dictionary**

| **Table Name** | **Attributes** | **Description** | **Domain** | **Allow Null** |
| --- | --- | --- | --- | --- |
| **account** | email(PK) | Unique identifier for account | VARCHAR (255) | No |
| Password | Password for each account | VARCHAR (255) | Yes |
| Status | Status for each account | INT(11) | No |
| role\_id | FK references to table “role” | INT(11) | No |
| **role** | role\_id{PK} | Unique identifier for role | INT(11) | No |
| role\_name | Name for each role | VARCHAR (255) | No |
| **profile** | profile\_id(PK) | Unique identifier for each profile | INT(11) | No |
| full\_name | Full name for each profile’s account | VARCHAR (255) | Yes |
| phone | Phone for each profile’s account | VARCHAR (255) | Yes |
| gender | Gender for each profile’s account | BIT(1) | Yes |
| date\_of\_birth | Date of birth for each profile’s account | DATE | Yes |
| email{FK} | FK references to table: “account” | VARCHAR (255) | No |
| **picture** | name{PK} | Unique identifier for picture | VARCHAR(45) | No |
| tag | Tag for each picture | VARCHAR(255) | No |
| added\_date | Added Date for each picture | DATETIME | No |
| stepnum | Number of step for each picture | INT(11) | No |
| status | Status for each picture | INT(11) | No |
| email{FK} | FK references to table: “account” | VARCHAR(255) | No |
| **license type** | license\_type\_id{PK} | Unique identifier of license type | INT(11) | No |
| name | Name for each license type | VARCHAR(50) | No |
| day | Used day for each license type | INT(11) | Nos |
| price | Price for each license type | FLOAT | No |
| status | Status for each license type | INT(11) | No |
| **contract** | contract\_id{PK} | Unique identifier for contract | INT(1) | No |
| bought\_time | Bought time for each contract | DATETIME | No |
| email(FK) | FK references to table: “account” | VARCHAR(255) | No |
| license\_type\_id(FK) | FK references to table: “license type” | INT(11) | No |
| **notification** | notification\_id{PK} | Unique identifier for notification | INT(11) | No |
| Message | Message for each notification | VARCHAR (255) | No |
| date | Date for each notification | DATETIME | No |
| email(PK) | FK references to table: “account” | VARCHAR(255) | No |
| **device** | device\_id{PK} | Unique identifier for device | INT(11) | No |
| device\_token | Device token for each device | VARCHAR(255) | No |
| is\_notify | Status for each device | INT(11) | No |
| email{FK} | FK references to table: “account” | VARCHAR(255) | No |

## Implementaion

### Web Architecture Implementation



|  |  |
| --- | --- |
| **Name** | **Description** |
| DispatcherServlet |  |
| HandlerMapping |  |
| Controller |  |
| Service | Responsible for business logic |
| Model | Business Logic Layer |
| Repository | Object relational mapping |
| Database | Contains a collection of schemas, table, queries |
| ViewResolver |  |
| View | Outputs the display of information |

### Web Service Architecture Implementation

|  |  |
| --- | --- |
| **Name** | **Description** |
| Controller | Receives input from client |
| Model | Business Logic Layer |
| Database | Contains a collection of schemas, table, queries |

### Mobile Architecture Implementation

Mobile Architecture

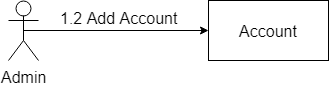
|  |  |
| --- | --- |
| **Name** | **Description** |
| Controller | Control and handle event of a screen |
| View Model | View properties and commands |
| Model | Business Logic Layer |
| Web Services | Interface to communicate with web services |
| View | Outputs the display of information |

## Test Plan

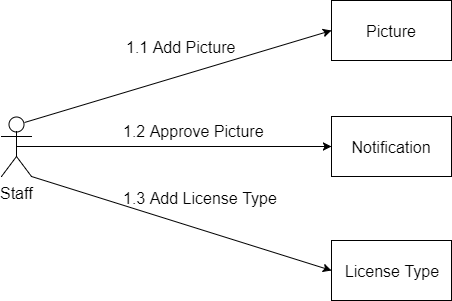
### Communication Diagram

#### Web Application

##### Admin

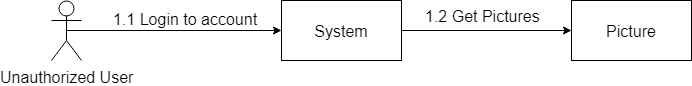


##### Staff

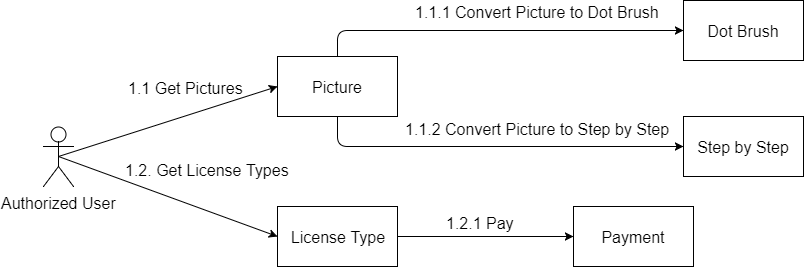


#### Mobile Application

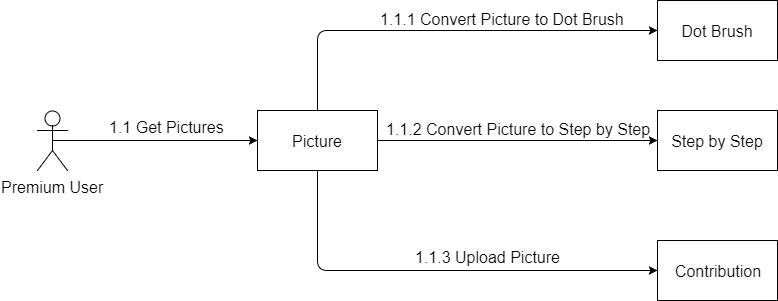
##### Unauthorized User



##### Authorized User



##### Premium User



## System Testing Test Case

### Web Application Test Cases

#### Admin

##### <Admin> Add Account

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **AAA01** | Create an account | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: “Le Hoang Long, Nguyen Minh Quang…” * **Email**: “longlh@gmail.com, quangnm@gmail.com...” * **Gender**: select one of the options Male * **Phone**: 0909654321 * **Date of Birth**: 20/03/1993   4. Click “Add Account” button. | Web application shows message: “Add account successfully”. | Pass: 3  Fail: 7 | From 10/7/2017  To 14/7/2017 |
| **AAA02** | Create account without email | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: “Le Hoang Long, Nguyen Minh Quang…” * **Email**: * **Gender**: select one of the options Male * **Phone**: 0909654321 * **Date of Birth**: 20/03/1993   4. Click “Add Account” button. | Web application shows error message “Required” under Email fields. | Pass: 4  Fail: 10 | From 10/7/2017  To 14/7/2017 |
| **AAA03** | Create account with email has existed | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: “Le Hoang Long, Nguyen Minh Quang…” * **Email**: “longlh@gmail.com, quangnm@gmail.com...” * **Gender**: select one of the options Male * **Phone**: 0909654321 * **Date of Birth**: 20/03/1993   4. Click “Add Account” button. | Web application shows error message “Email existed”. | Pass: 4  Fail: 10 | From 10/7/2017  To 14/7/2017 |
| **AAA04** | Create account without full name | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: * **Email**: “longlh@gmail.com, quangnm@gmail.com...” * **Gender**: select one of the options Male * **Phone**: 0909654321 * **Date of Birth**: 20/03/1993   4. Click “Add Account” button. | Web application shows error message “Required” under Full Name fields. | Pass: 4  Fail: 10 | From 10/7/2017  To 14/7/2017 |
| **AAA05** | Create account without Phone | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: “Le Hoang Long, Nguyen Minh Quang…” * **Email**: “longlh@gmail.com, quangnm@gmail.com...” * **Gender**: select one of the options Male * **Phone**: * **Date of Birth**: 20/03/1993   4. Click “Add Account” button. | Web application shows error message “Required” under Phone fields. | Pass: 3  Fail: 10 | From 10/7/2017  To 14/7/2017 |
| **AAA06** | Create account without Date of Birth | 1. Login successfully with role “Admin”.  2. In “Account” view, click “Add Account” button.  3. Input fields:   * **Full Name**: “Le Hoang Long, Nguyen Minh Quang…” * **Email**: “longlh@gmail.com, quangnm@gmail.com...” * **Gender**: select one of the options Male * **Phone**: 0909654321 * **Date of Birth**:   4. Click “Add Account” button. | Web application shows error message “Required” under Date of Birth fields. | Pass: 5  Fail: 5 | From 10/7/2017  To 14/7/2017 |

#### Staff

##### <Staff> Add Picture

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **SADP01** | Add a picture | 1. Login successfully with role “Staff”.  2. Go to “Picture” view  3. Input fields:   * **Picture**: upload picture.   4. Click “Add Picture” button. | Web application shows message: “Add picture successfully | Pass: 47  Fail: 3 | From 10/7/2017  To 14/7/2017 |
| **SADP02** | Add picture without picture | 1. Login successfully with role “Staff”.  2. Go to “Picture” view  3. Input fields:   * **Picture**:   4. Click “Add Picture” button. | Web application shows error message “Required” under Picture fields. | Pass: 9  Fail: 1 | From 10/7/2017  To 14/7/2017 |

##### <Staff> Approve Picture

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **SAPP01** | Approve a picture | 1. Login successfully with role “Staff”.  2. Go to “Notification” view  3. Input fields:   * **Tag**: “drawing, house, cat…”   4. Click “Approve Picture” button. | Web application shows message: “Add picture successfully | Pass: 47  Fail: 3 | From 10/7/2017  To 14/7/2017 |
| **SAPP02** | Approve picture without tag | 1. Login successfully with role “Staff”.  2. Go to “Notification” view  3. Input fields:   * **Tag**:   4. Click “Approve Picture” button. | Web application shows error message “Required” under Tag fields. | Pass: 9  Fail: 1 | From 10/7/2017  To 14/7/2017 |

##### <Staff> Add License Type

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **SALT01** | Add a license type | 1. Login successfully with role “Staff”.  2. In “License Type” view, click “Add License Type” button.  3. Input fields:   * **License Name**: “Gold, Diamond…” * **Day**: “30, 60…” * **Price**: “0.99, 1.99…”   4. Click “Add License Type” button. | Web application shows message: “Create license type successfully”. | Pass: 7  Fail: 3 | From 10/7/2017  To 14/7/2017 |
| **SALT02** | Add license type with license name has existed | 1. Login successfully with role “Staff”.  2. In “License Type” view, click “Add License Type” button.  3. Input fields:   * **License Name**: “Gold, Diamond…” * **Day**: “30, 60…” * **Price**: “0.99, 1.99…”   4. Click “Add License Type” button. | Web application shows error message “License Name existed”. | Pass: 4  Fail: 1 | From 10/7/2017  To 14/7/2017 |
| **SALT03** | Add license type without license name | 1. Login successfully with role “Staff”.  2. In “License Type” view, click “Add License Type” button.  3. Input fields:   * **License Name**: * **Day**: “30, 60…” * **Price**: “0.99, 1.99…”   4. Click “Add License Type” button. | Web application shows error message “Required” under License Name fields. | Pass: 4  Fail: 1 | From From 10/7/2017  To 14/7/2017 |
| **SALT04** | Add license type without license name | 1. Login successfully with role “Staff”.  2. In “License Type” view, click “Add License Type” button.  3. Input fields:   * **License Name**: “Gold, Diamond…” * **Day**: * **Price**: “0.99, 1.99…”   4. Click “Add License Type” button. | Web application shows error message “Required” under Day fields. | Pass: 4  Fail: 1 | From 10/7/2017  To 14/7/2017 |
| **SALT05** | Add license type without license name | 1. Login successfully with role “Staff”.  2. In “License Type” view, click “Add License Type” button.  3. Input fields:   * **License Name**: “Gold, Diamond…” * **Day**: “30, 60…” * **Price**:   4. Click “Add License Type” button. | Web application shows error message “Required” under Price fields. | Pass: 4  Fail: 1 | From 10/7/2017  To 14/7/2017 |

### Mobile Appplication Test Cases

#### Authorized User

##### <Authorized User> Dot Brush

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **AUDB01** |  |  |  |  |  |
| **AUDB02** |  |  |  |  |  |
| **AUDB03** |  |  |  |  |  |
| **AUDB04** |  |  |  |  |  |
| **AUDB05** |  |  |  |  |  |

##### <Authorized User> Step by Step

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **AUSS01** |  |  |  |  |  |
| **AUSS02** |  |  |  |  |  |
| **AUSS03** |  |  |  |  |  |
| **AUSS04** |  |  |  |  |  |
| **AUSS05** |  |  |  |  |  |

#### Premium User

##### <Premium User> Upload Picture

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Result** | **Test Date** |
| **PUUP01** |  |  |  |  |  |
| **PUUP02** |  |  |  |  |  |
| **PUUP03** |  |  |  |  |  |
| **PUUP04** |  |  |  |  |  |
| **PUUP05** |  |  |  |  |  |

# Software User’s Manual

## Installation Guide

### Setting up environment at server side

The specifications are based on the dependencies requirements and performance test result from previous section of this document.

#### Hardware Requirements

|  |  |
| --- | --- |
| **Hardware** | **Specification** |
| Internet Connection | Cable, Wi-Fi (4 Mbps) |
| Computer Processor | Intel® Xeon 1.4GHz or more |
| Computer Memory | 2GB of RAM or more |
| Hard Drive | 30GB or more |

#### Software Requirements

|  |  |
| --- | --- |
| **Software** | **Application name/version** |
| Operating System | Microsoft Server 2008 |
| Java | Intellij |
| Python | PyCharm |
| DBMS | MySQL |

#### Mobile Requirements

|  |  |
| --- | --- |
| **Software** | **Application name/version** |
| CPU |  |
| Memory | 2 GB |
| Storage | 16 GB |
| Model | Galaxy Note 5 |

### Deployment at server side

#### Prepare deployment package

#### Configure Server before deploy

#### Deploy web application on server

### Setting up the environment at client side

#### Setting up for computer

## User Guide

### Mobile Application

### Web Appication

## Appendix