yazzlkja

Photographer Matching Platform

System and Software Architecture Description (SSAD)

Team 2

- 2014210011 고나현 rhskgus@korea.ac.kr
- 2013210086 김영훈 2013210086@korea.ac.kr
- 2014210101 김원경 kwk2392@korea.ac.kr
- 2013210081 박진호 onlylove6913@kroea.ac.kr
- 2013210039 이민섭 alexn0@korea.ac.kr
- 2014210065 이상진 eesj@korea.ac.kr
- 2015410005 이정섭 james319@korea.ac.kr
- 2013210020 임원준 rhimjun@korea.ac.kr
- 2014210059 정희석 poco2889@korea.ac.kr

Table of Contents

Version control

Date	Author	Revise	Version
11/24/2018	고나현, 이정섭, 임원준, 정희석	Initial draft	0.1
11/26/2018	All	Overall revise	0.2

List of Figures

1 Introduction

The section covers the purpose of the system and software architecture description document as well as all the references used in the preparation of this document.

1.1 Purpose of SSAD

The System and Software Architecture Description describes how the Photographer Matching Platform will be implemented. The document contains two major sections, system analysis and architectural design of the proposed system.

The system analysis section focuses on the high level architectural components and interactions that satisfy the system requirements. It will describe how the system will be decomposed into three main components. It also analyzes the behaviors of the photographer(s) and photographee(s).

The architectural design section focuses on the how the software is implemented using specific software framework, components and technologies. The section describes to the implementers such that it is clear for them to proceed to construction phase. The architecture details presented here will be the committed architecture and design that will be built in the construction phase.

1.2 Standards and Conventions

MBASE framework is used for overall software engineering process.

Unified Modeling Language (UML) is used for diagram illustrations. Use case diagram, class diagram, sequence diagrams are used for graphical presentation in a standard notation.

1.3 References

1. UML Documentation (10/22/2018)
2. WinWin Negotiation Results (10/14/2018)
3. Project Developer Meeting Minutes: 가자쏘공 0920.docx, 가자쏘공 0927.docx 가자쏘공 1010.docx 가자쏘공 1014.docx
4. System and Software Requirements Description (SSRD) v0.5

2 Architectural Analysis

The intention of the architectural analysis is to identify the high-level software components and interactions from different perspectives.

2.1 Component Model

The component model in figure 2-1 is derived and refined from Entity Model of OCD 4.5.2. Components are identified by rejecting non-component entity classes, such as Photo DB, Message DB, etc. Those entities will be described in SSAD 3.2 Object Model.

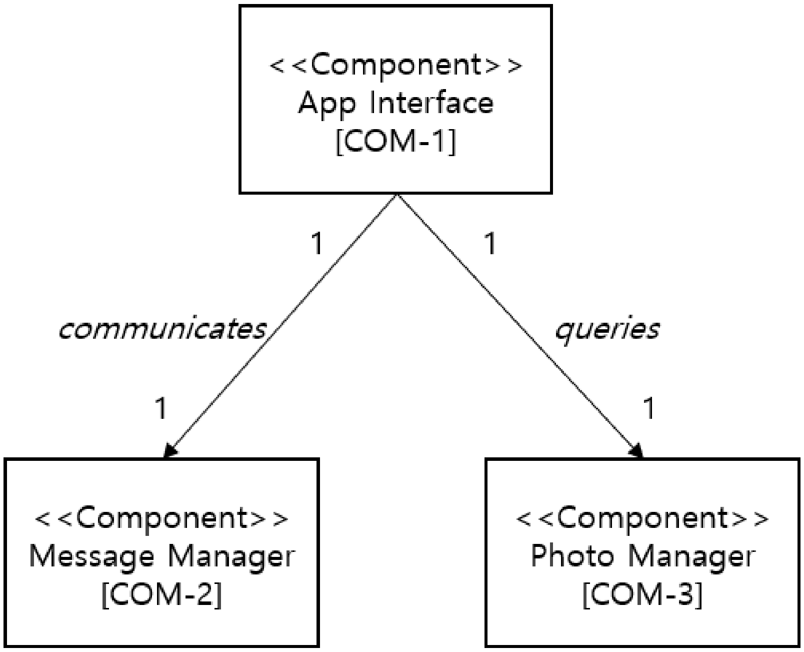
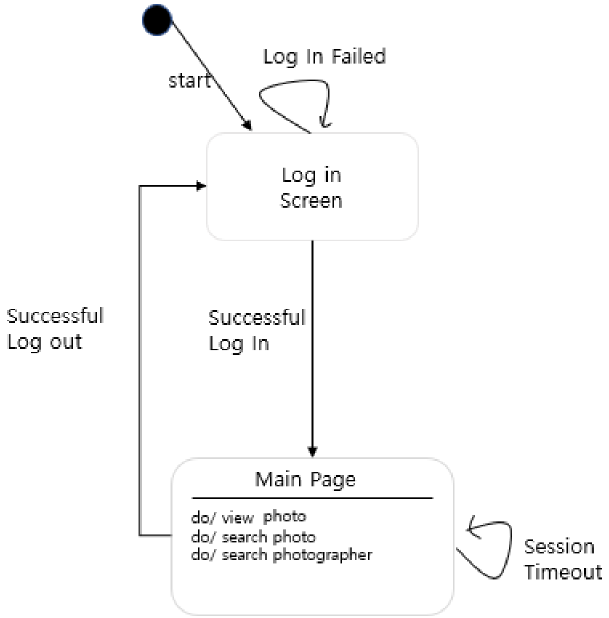
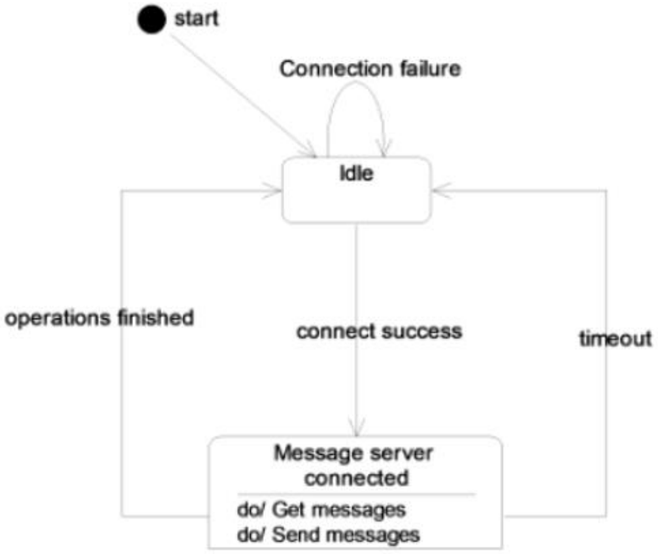
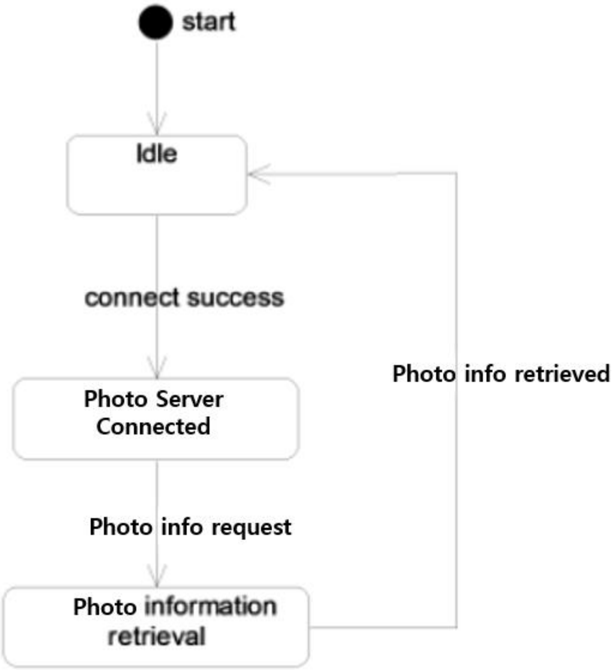


Figure 2-1 Component Model Diagram

Component Specification:

<table>
<tr><th>Identifier</th><th colspan="2">COM-1</th></tr>
<tr><td>Defining quality</td><td colspan="2">A user interface that allows to display photo and photographer's profile and message exchange.</td></tr>
<tr><td>Name</td><td colspan="2">App Interface</td></tr>
<tr><td>Attributes</td><td colspan="2">a) Customer ID
b) Customer Profile
c) Display Photo
d) Message index</td></tr>
<tr><td>Behaviors</td><td colspan="2">Refer to SSRD v0.4,
Figure 2-2 Behavior Model – Access through</td></tr>
<tr><td>Relationships</td><td colspan="2">a) COM-2 Message Manager
b) COM-3 Photo Manager</td></tr>
<tr><td>Roles</td><td colspan="2">Request initiator</td></tr>
<tr><td>State Groups</td><td colspan="2"> </td></tr>
<tr><td>Constraints</td><td>Dependencies</td><td>COM-2 Message Manager
COM-3 Photo Manager</td></tr>
<tr><td></td><td>Candidate Key</td><td></td></tr>
<tr><td></td><td>Cardinality</td><td>1, each app interface serves one user</td></tr>
</table>

Identifier	**COM-2**	
Defining quality	A service to provide client to access and exchange messages between photographee(s) and photographer(s). Messages are stored in DB.	
Name	Message Manager	
Attributes	a) Message server address b) Message protocol	
Behaviors	a) Response to customer's request b) Read messages from the message server c) Send messages through the message server	
Relationships	COM-1 App Interface	
Roles	Middle-man between App Interface and existing message server(s)	
State Groups	 start Connection failure Idle operations finished connect success timeout Message server connected do/ Get messages do/ Send messages	
Constraints	Dependencies	Nil.
	Candidate Key	
	Cardinality	1, one app interface interacts 1 message manager component

Identifier	**COM-3**	
Defining quality	A service providing access to search for pictures and find the pictures you want. This is done by connecting to message server.	
Name	Photo Manager	
Attributes	a) Photo server address b) Retrieval photo	
Behaviors	Lookup photo which customer wants	
Relationships	COM-1 App Interface	
Roles	Middle-man to existing photo server	
State Groups	 start Idle connect success Photo Server Connected Photo info request Photo information retrieval Photo info retrieved	
Constraints	Dependencies	Nil.
	Candidate Key	
	Cardinality	1, each app interface component has photo manager component

2.2 Behavior Model

The behavior model describes the behaviors that will be carried out in our system from the actors' perspective.

2.2.1 Behaviors breakdown

CAP-01 Anytime and anywhere access the system

- BH-01 Access through application
- BH-02 Login <<policy>>

CAP-02 Search and View Photo and Photographer

- BH-03 Search Photo
- BH-04 View Photo
- BH-05 Search Photographer

CAP-03 Exchange Message between photographer and photographee

- BH-06 Send Message

2.2.2 Diagram illustration

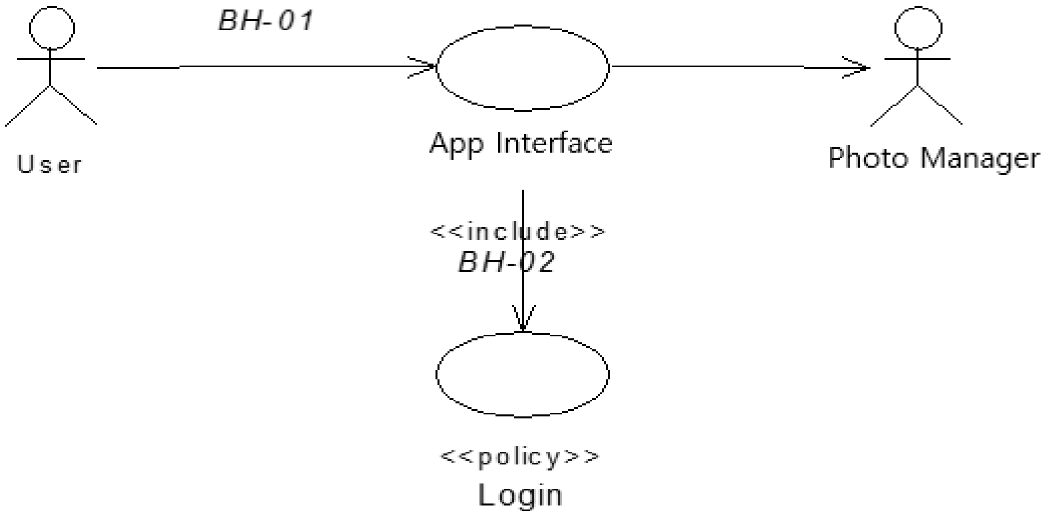


Figure 2-2 Behavior Model – Access through application

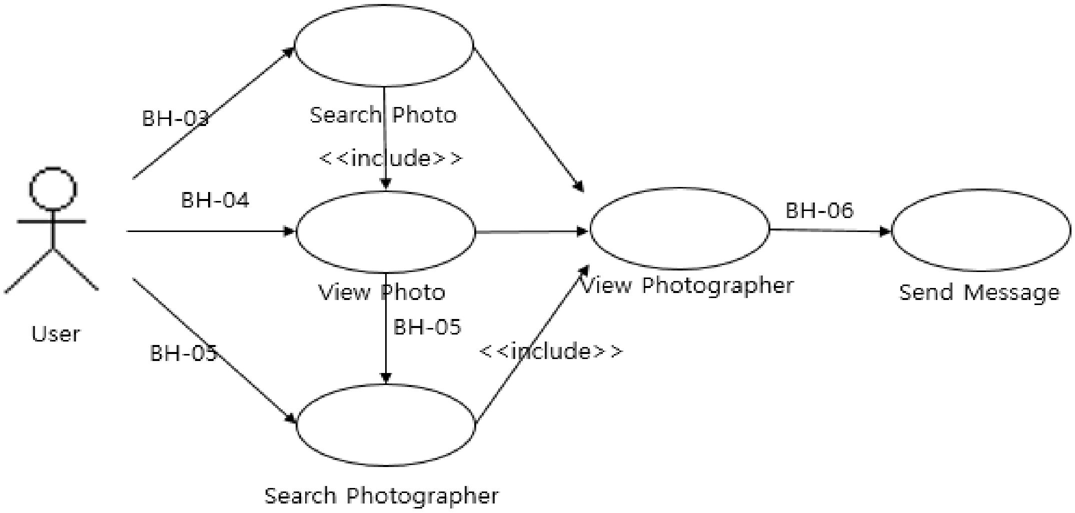


Figure 2-3 Behavior Model – User

2.2.3 Behavior Specification

Identifier	**BH-01**
Name	Access app interface
Trigger	User motivation of using platform service
Preconditions	Open a application
Postconditions	Open application main page
Input	UserID, password
Output	Main page of application displayed
Exceptions	Invalid user login
Use Case	Refer to SSRD Figure 3-2

Identifier	**BH-02**
Name	Login
Trigger	Signin button clicked
Preconditions	Login page shown
Postconditions	Open application main page
Input	Username, password
Output	Main page of application displayed
Exceptions	Invalid user login
Use Case	Refer to UML_Diagram Sequence Diagram 2. Login

Identifier	**BH-03**
Name	Search Photo
Trigger	Click search button and type keyword(s)
Preconditions	A photographer uploads a photo, he/she can include a photo-taken location information and keyword(s).
Postconditions	
Input	Keyword(s), location
Output	Show photos related to keyword(s) or a location
Exceptions	a) There is no photo matched with a keyword b) There is no photo matched with a location
Use Case	Refer to UML_Diagram Sequence Diagram 6. Search Photo

Identifier	**BH-04**
Name	View Photo
Trigger	Success Login
Preconditions	Only logged-in users can view uploaded photo(s)
Postconditions	
Input	Photo ID
Output	A photo with description
Exceptions	
Use Case	Refer to UML_Diagram Sequence Diagram 14. View Photo

Identifier	**BH-05**
Name	Search Photographer
Trigger	Click search button and put location or name
Preconditions	Information of photographers and locations should be stored in their profile.
Postconditions	
Input	Location, Name
Output	Show the list of photographers
Exceptions	a) When there is no photographer in the location b) In case you search a photographer by his/her name and there is no one having the name
Use Case	Refer to UML_Diagram Sequence Diagram 5. Search Photographer

Identifier	**BH-06**
Name	Send Message
Trigger	Click message button
Preconditions	Service user(s) who register the app is given an authority to send and receive messages
Postconditions	The Matching between photographee(s) and

	photographer(s)
Input	What to say to a photographer or a photographee
Output	Chat room with history of sender/recipient
Exceptions	Message server is down
Use Case	Refer to SSRD RQ-7

2.3 Enterprise Model

The enterprise model describes the overall inter-relationship of application structures including components and behaviors. Relationships such as generalization will be shown here.

2.3.1 Component Classifications

Classification hierarchy of components based on SSAD 0 The intention of the architectural analysis is to identify the high-level software components and interactions from different perspectives.

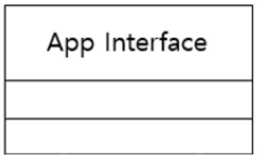


Figure 2-4 Enterprise model – App Interface

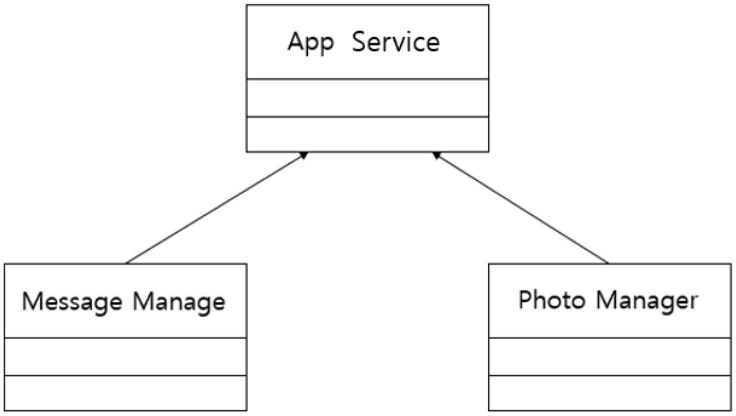


Figure 2-5 Enterprise model – Photographer Matching Service

2.3.2 Behavior Classifications

Classification hierarchy of the behaviors presented in SSAD 2.2 Behavior Model

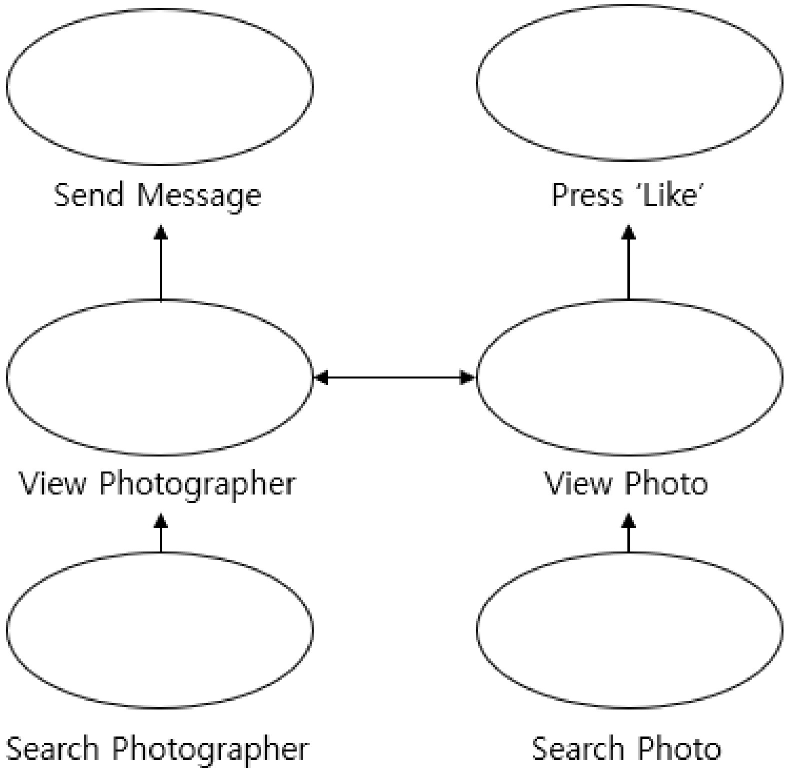


Figure 2-6 Enterprise model - behaviors

3 System Design

The system design section of the document describes how the system will be implemented in software. Specific technology choices such as software framework, protocols and implementation objects are mentioned such that the implementers receive clear and precise architectural design proposed.

3.1 Architectural Views

The architectural views provide a high level blueprint for implementation from different perspectives. They act as a bridge that transforms from system analysis result to implementation.

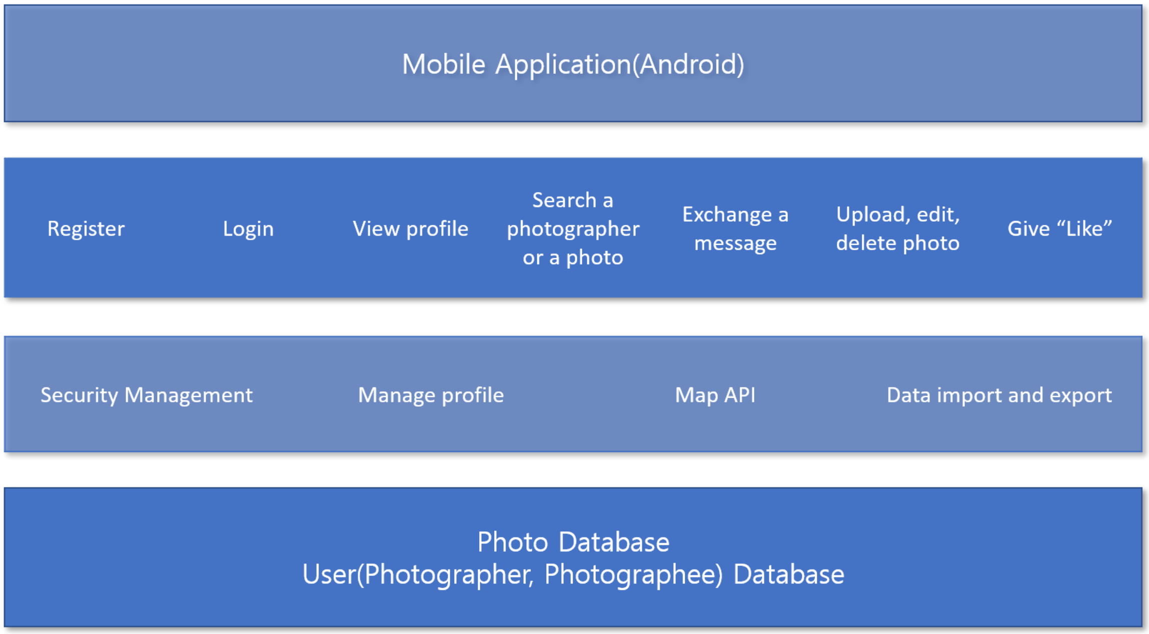


Figure 3-1 Layered Architecture

3.1.1 Layered Architecture

Description	Organizes the system into layers with related functionality associated with each layer. A layer provides services to the layer above it so the lowest-level layers represent core services that are likely to be used throughout the system.
Advantages	It is simple that it is easy to learn and visible in the project at first. It is consistent across different projects that the layers. Separation of concerns is guaranteed that concerns having a layer and to the point that you stick to the rules of Layered Architecture, but it is easy with the code organization it implies.
Disadvantages	It is confusing to find the right levels of abstraction. For example, ‘Exchange a message’, which is on 2nd layer, includes some functionality factors. As it is normally on 3rd layer, ‘Exchange a message’ should be placed on 3rd layer. But messaging is too ambiguous to put 3rd layer because its properties are also into 2nd layer. Also, it is very hard to tell what a project is doing by looking at the code organization. To understand the overview you should read the class names and even the implementation.

Component Specification:

Layered architecture is appropriate to our project because of its simplicity. It is easy to understand that a layer communicates with an adjacent layer. For instance, we can find out at first grasp that ‘Login’ which is on user interface management layer(2nd layer) is related to ‘Security Management’ which is on functionality layer(3rd layer). Also, we divide the whole project into various small ones, consistency is highly required in order to prevent any confusion. It helps to keep the overall code organization is the same in every layered project.

3.2 Class Model

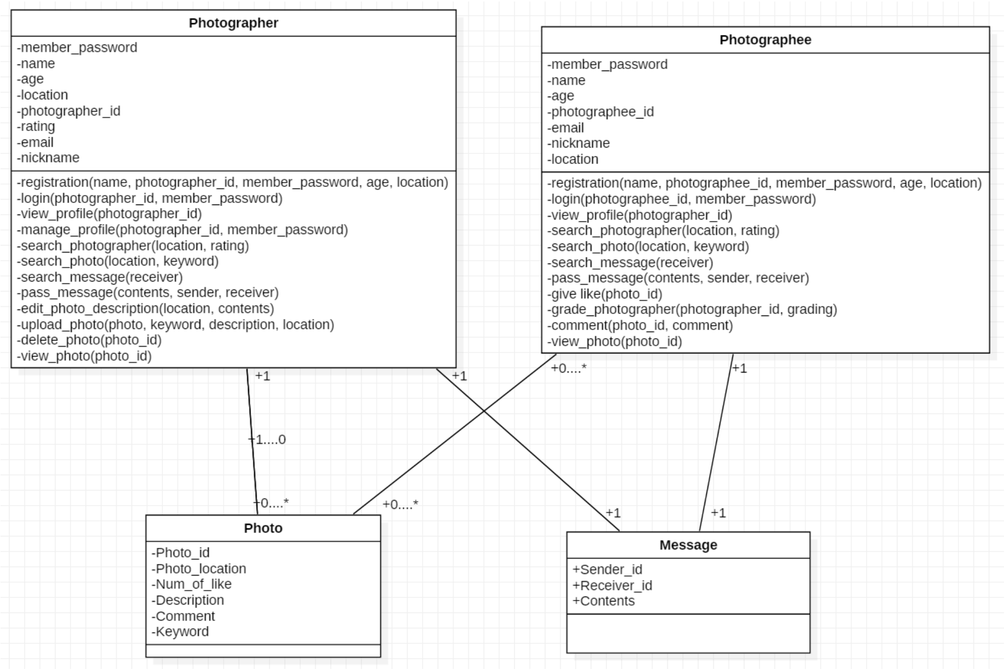

Photographer
-member_password
-name
-age
-location
-photographer_id
-rating
-email
-nickname
-registration(name, photographer_id, member_password, age, location)
-login(photographer_id, member_password)
-view_profile(photographer_id)
-manage_profile(photographer_id, member_password)
-search_photographer(location, rating)
-search_photo(location, keyword)
-search_message(receiver)
-pass_message(contents, sender, receiver)
-edit_photo_description(location, contents)
-upload_photo(photo, keyword, description, location)
-delete_photo(photo_id)
-view_photo(photo_id)
Photographee
-member_password
-name
-age
-photographee_id
-email
-nickname
-location
-registration(name, photographee_id, member_password, age, location)
-login(photographee_id, member_password)
-view_profile(photographer_id)
-search_photographer(location, rating)
-search_photo(location, keyword)
-search_message(receiver)
-pass_message(contents, sender, receiver)
-give like(photo_id)
-grade_photographer(photographer_id, grading)
-comment(photo_id, comment)
-view_photo(photo_id)
+1
+1
+0....*
+1
+1....0
+0....*
+0....*
+1
+1
Photo
-Photo_id
-Photo_location
-Num_of_like
-Description
-Comment
-Keyword
Message
+Sender_id
+Receiver_id
+Contents

Figure 3-2 Class Model Diagram

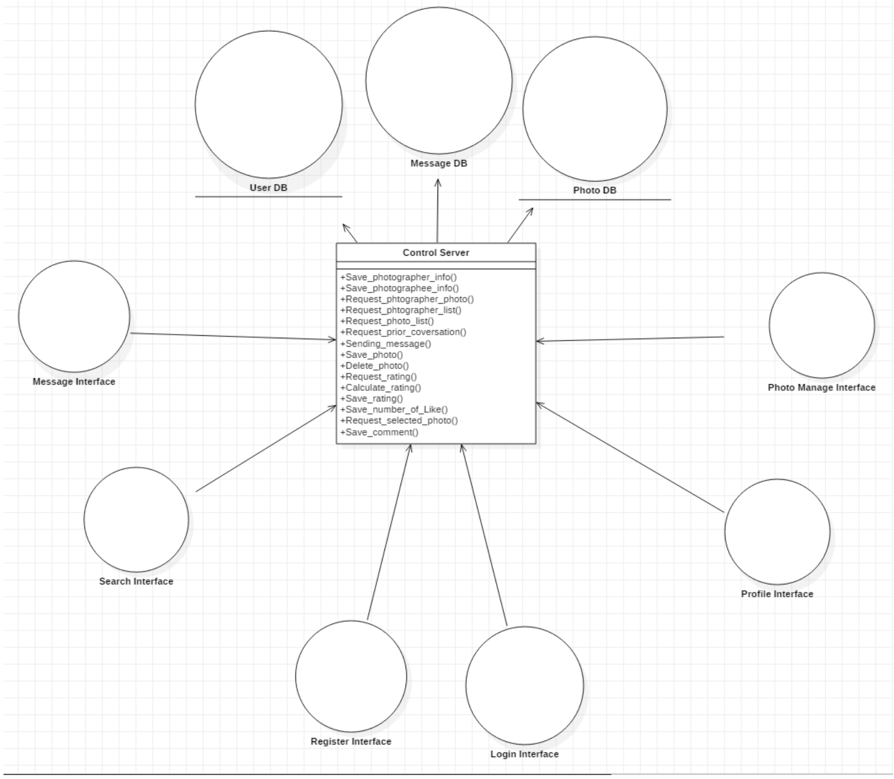


Figure 3-3 Object Model Diagram

3.3 Operations Model

3.3.1 Critical Algorithms

No critical algorithm used in this Photographer matching platform project.

3.3.2 Operation Specifications

3.3.2.1 Registration

Identifier	**OP-01**
Initiator	SSAD 2.2.3 Registration <Behavior>
Passed parameters	a) Username b) Password c) Account type(photographee, photographer)
Return values	Main page of the photographer matching platform with welcoming message
Exception handling	Registration failed, a pop-up message “Fail” shows up.
Guards	Entries cannot be empty
Validation	Connect to server to validate the user registration
Messages	Registration request {Request}
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-01

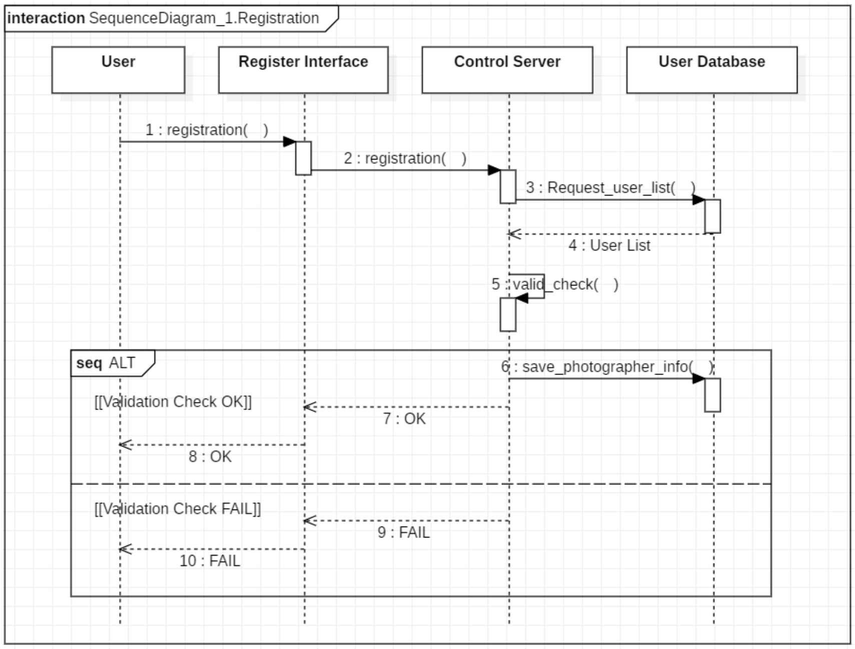


Figure 3-4 Registration OP-01

3.3.2.2 Login

Identifier	**OP-02**
Initiator	SSAD 2.2.3 Login <Behavior>
Passed parameters	a) Username b) Password c) Account type
Return values	Main page of the photographer matching platform
Exception handling	a) Login failed, a pop-up message “Fail” shows up.
Guards	Entries cannot be empty
Validation	Connect to server to validate the user registration
Messages	Login request {Request}
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-02

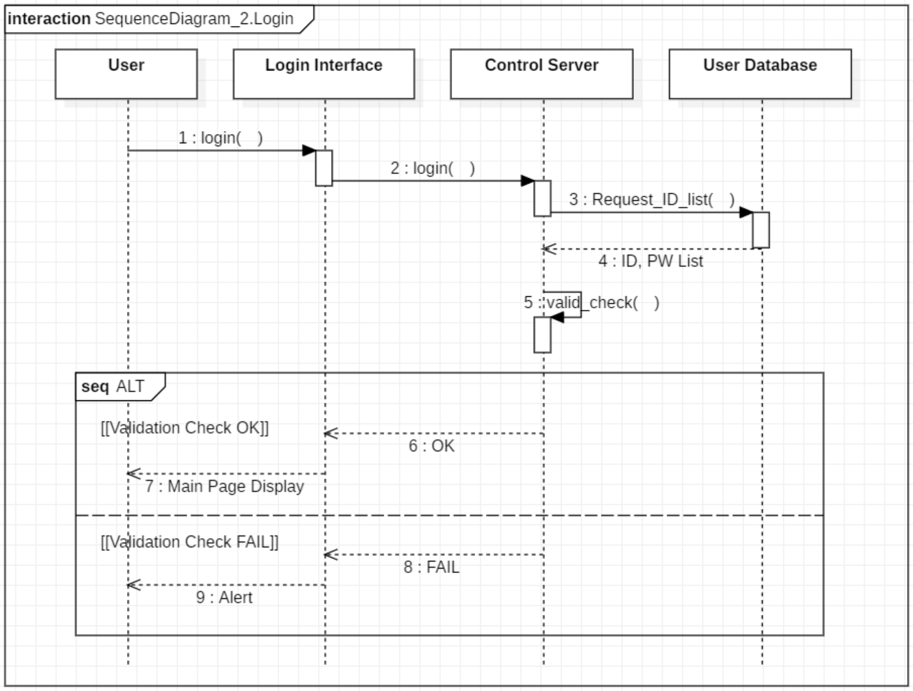


Figure 3-5 Login OP-02

3.3.2.3 View Profile

Identifier	**OP-03**
Initiator	SSAD 2.2.3 View Profile <Behavior>
Passed parameters	a) Photographer
Return values	photographer's profile page
Exception handling	Failing to get response from server, a pop-up message "Fail" shows up.
Guards	Information of a photographer's profile should be contained in DB.
Validation	The photographer that a user intends to see should be identical to information of a photographer's profile saved in DB.
Messages	View profile request {request}
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-03

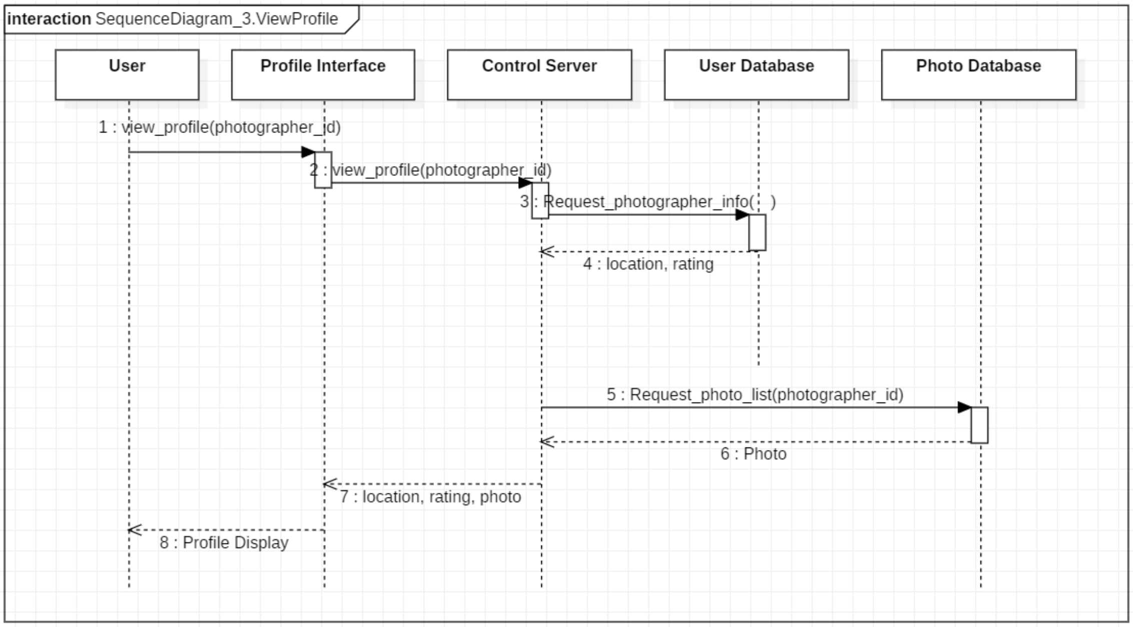

interaction SequenceDiagram_3.ViewProfile
User
Profile Interface
Control Server
User Database
Photo Database
1 : view_profile(photographer_id)
2 : view_profile(photographer_id)
3 : Request_photographer_info( )
4 : location, rating
5 : Request_photo_list(photographer_id)
6 : Photo
7 : location, rating, photo
8 : Profile Display

Figure 3-6 View Profile OP-03

3.3.2.4 Manage Profile

Identifier	**OP-04**
Initiator	SSAD 2.2.3 Manage Profile <Behavior>
Passed parameters	a) modified profile
Return values	revised profile page
Exception handling	Failing to get response from server, a pop-up message "Fail" shows up.
Guards	Information of a user's profile should be contained in DB.
Validation	Revised information in profile should be saved in his/her own DB.
Messages	Manage profile request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-04

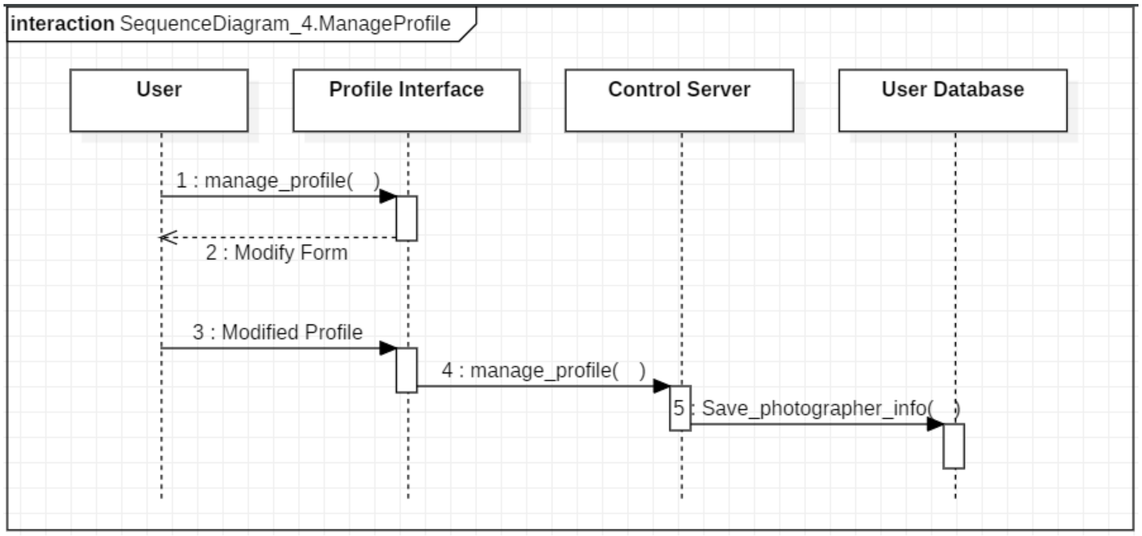


Figure 3-7 Manage Profile OP-04

3.3.2.5 Search Photographer

Identifier	**OP-05**
Initiator	SSAD 2.2.3 Search Photographer <Behavior>
Passed parameters	a) search term(s)
Return values	The list of photographers matched by search term(s)
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	Input string is not empty
Validation	A search term should be identical to the resulted data in DB.
Messages	Search photographer request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-05

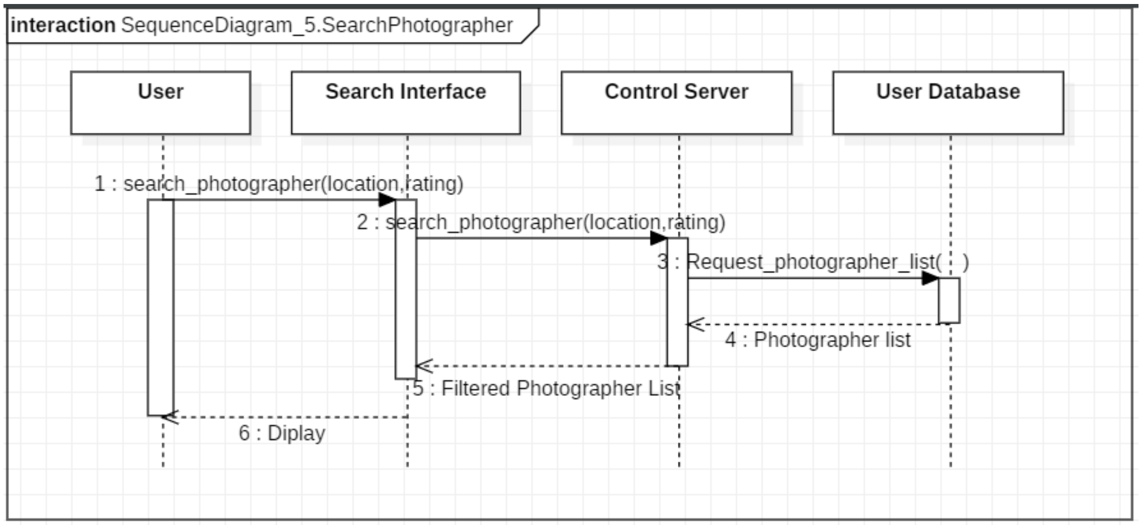

interaction SequenceDiagram_5.SearchPhotographer
User
Search Interface
Control Server
User Database
1 : search_photographer(location,rating)
2 : search_photographer(location,rating)
3 : Request_photographer_list( )
4 : Photographer list
5 : Filtered Photographer List
6 : Diplay

Figure 3-8 Search Photographer OP-05

3.3.2.6 Search Photo

Identifier	**OP-06**
Initiator	SSAD 2.2.3 Search Photo <Behavior>
Passed parameters	a) search term(s)
Return values	The list of photographers matched by search term(s)
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	Input string is not empty
Validation	A search term should be identical to the resulted data in DB.
Messages	Search photo request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-06

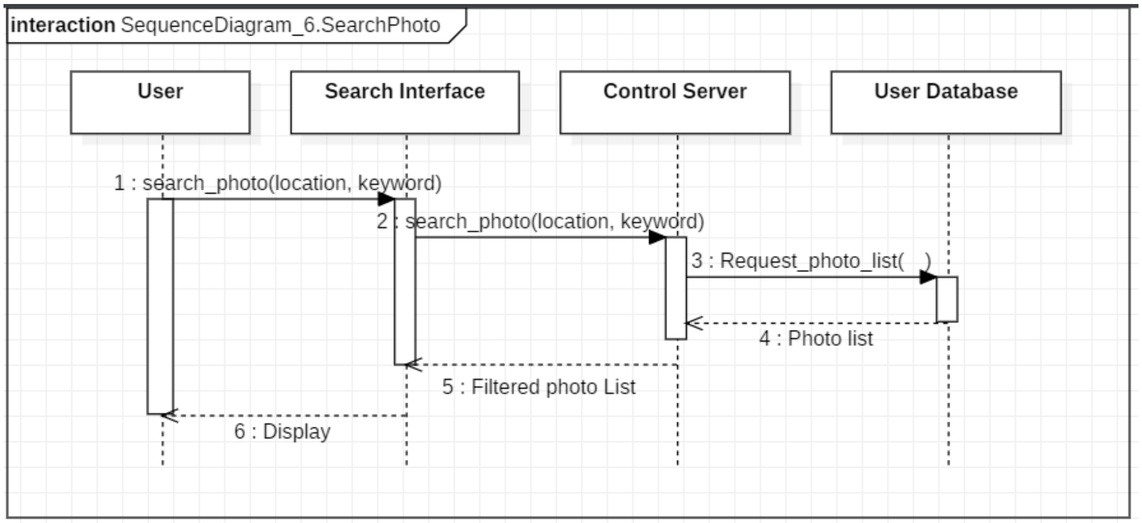

interaction SequenceDiagram_6.SearchPhoto
User
Search Interface
Control Server
User Database
1 : search_photo(location, keyword)
2 : search_photo(location, keyword)
3 : Request_photo_list( )
4 : Photo list
5 : Filtered photo List
6 : Display

Figure 3-9 Search Photo OP-06

3.3.2.7 Exchange Message

Identifier	**OP-07**
Initiator	SSAD 2.2.3 Exchange Message <Behavior>
Passed parameters	a) Message content
Return values	Chat room with history of sender/recipient
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	Message content is not empty
Validation	Letters should be shown identically in the chat room as they are typed in.
Messages	Exchange message request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-07

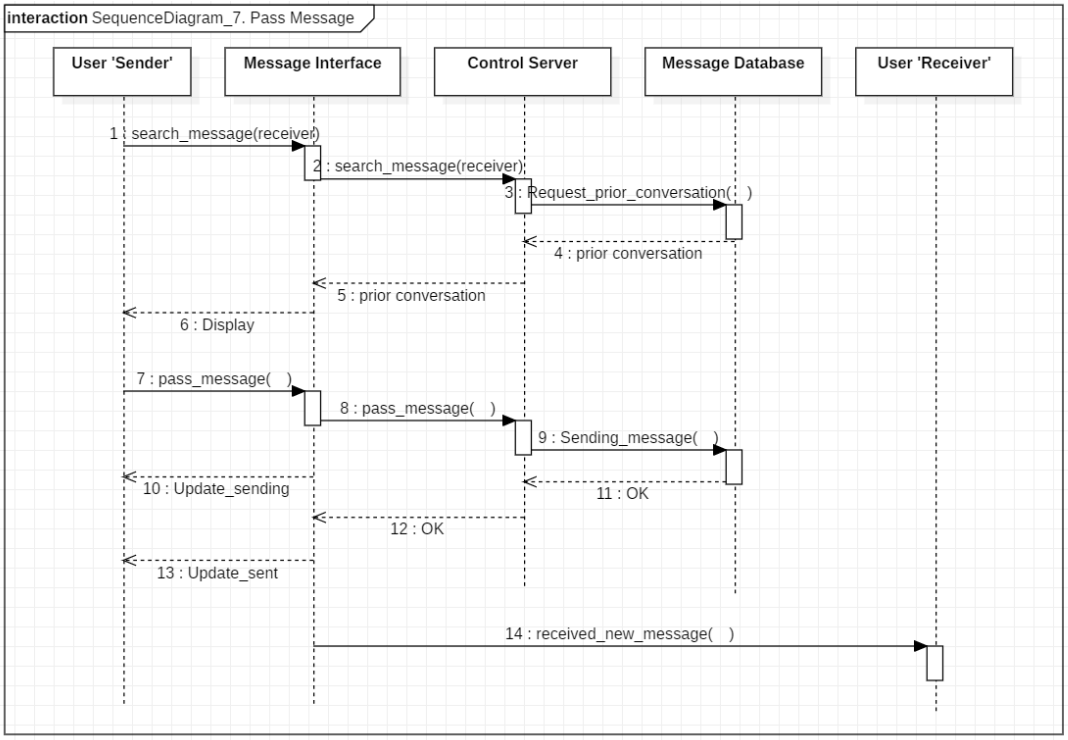


Figure 3-10 Exchange message OP-07

3.3.2.8 Upload Photo

Identifier	**OP-08**
Initiator	SSAD 2.2.3 Upload Photo <Behavior>
Passed parameters	a) photo b) description of photo
Return values	The page which shows uploaded photo(s)
Exception handling	Failing to get response from server, a pop-up message "Fail" shows up.
Guards	Input photo should exist. The size of photo(s) should be less than the limit of a file size.
Validation	Uploaded photo should be saved in his/her own DB.
Messages	Upload photo request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-08

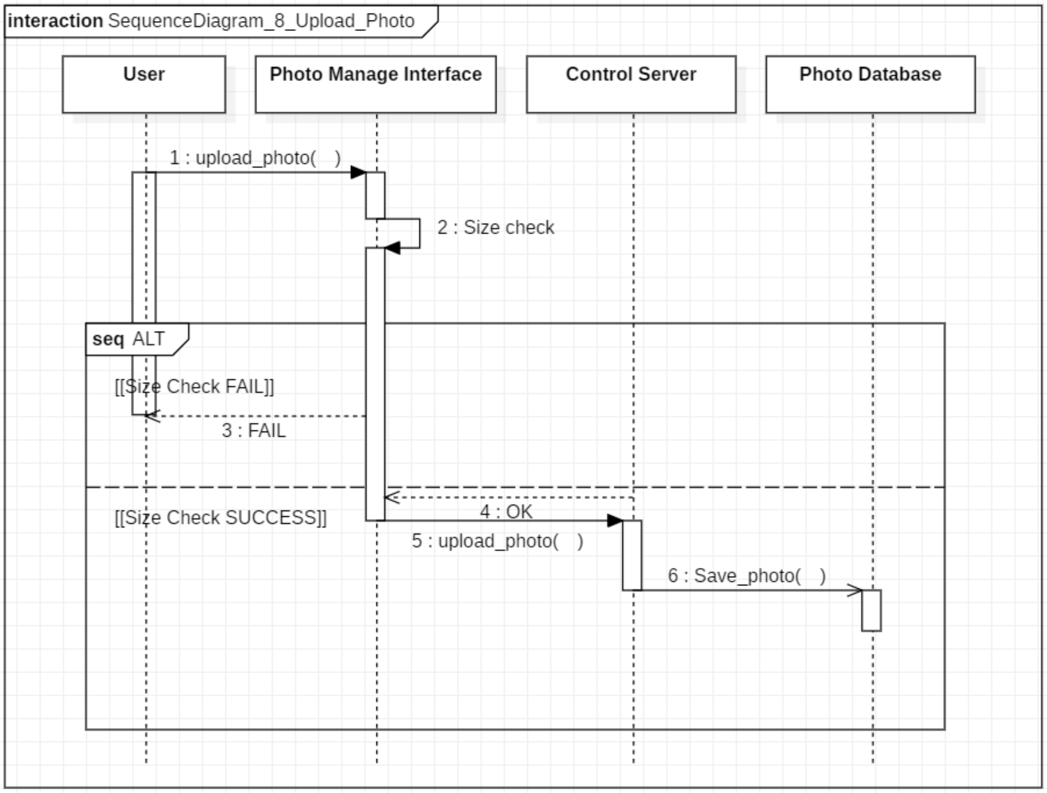


Figure 3-11 Upload photo OP-08

3.3.2.9 Edit Photo Description

Identifier	**OP-09**
Initiator	SSAD 2.2.3 Edit Photo Description <Behavior>
Passed parameters	a) description of photo
Return values	The page which shows edited description of photo(s)
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	Photos cannot be replaced.
Validation	Edited description of photo should be saved in his/her own DB.
Messages	Edit photo description request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-09

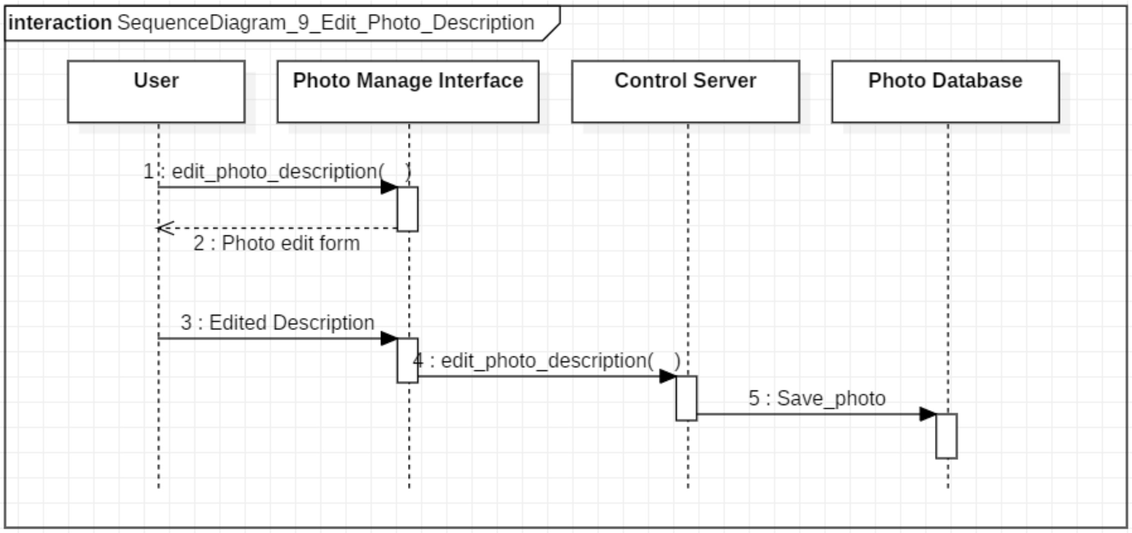

interaction SequenceDiagram_9_Edit_Photo_Description
User
Photo Manage Interface
Control Server
Photo Database
1 : edit_photo_description( )
2 : Photo edit form
3 : Edited Description
4 : edit_photo_description( )
5 : Save_photo

Figure 3-12 Edit photo OP-09

3.3.2.10 Delete Photo

Identifier	**OP-10**
Initiator	SSAD 2.2.3 Delete photo <Behavior>
Passed parameters	a) photo(s)
Return values	Pop-up message “Success!” shows up.
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	
Validation	Photo should be successfully deleted from his/her own DB.
Messages	Delete photo request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-10

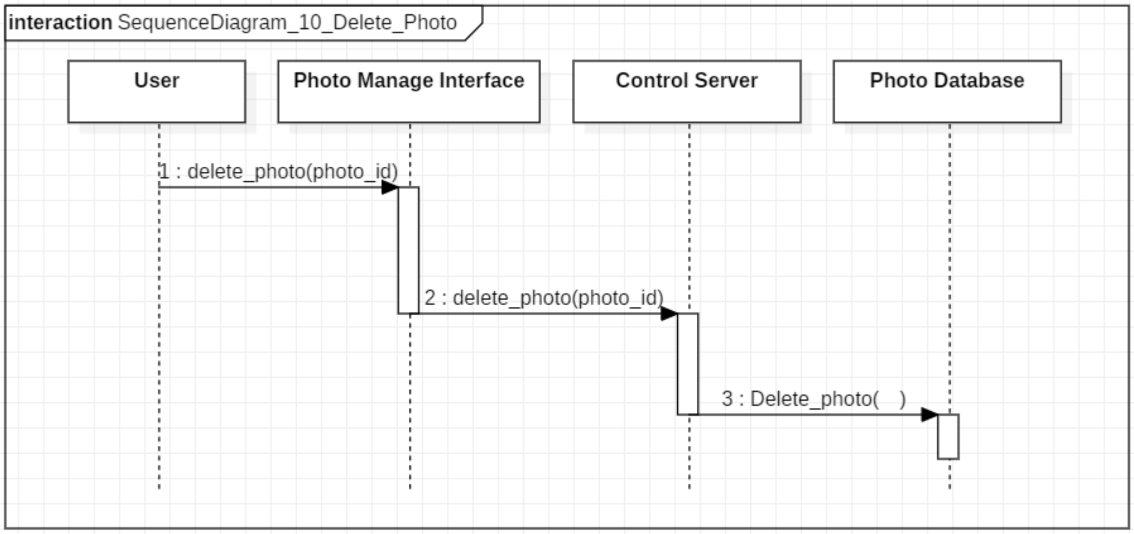

interaction SequenceDiagram_10_Delete_Photo
User
Photo Manage Interface
Control Server
Photo Database
1 : delete_photo(photo_id)
2 : delete_photo(photo_id)
3 : Delete_photo(   )

Figure 3-132 Delete photo OP-10

3.3.2.11 Give “Like”

Identifier	**OP-11**
Initiator	SSAD 2.2.3 Give “Like <Behavior>
Passed parameters	a) a click of “Like”
Return values	Updated page which shows the change of the number of “Like”
Exception handling	
Guards	
Validation	The change of the number of “Like” should be reflected in intended photographer’s DB.
Messages	Give “Like” request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-11

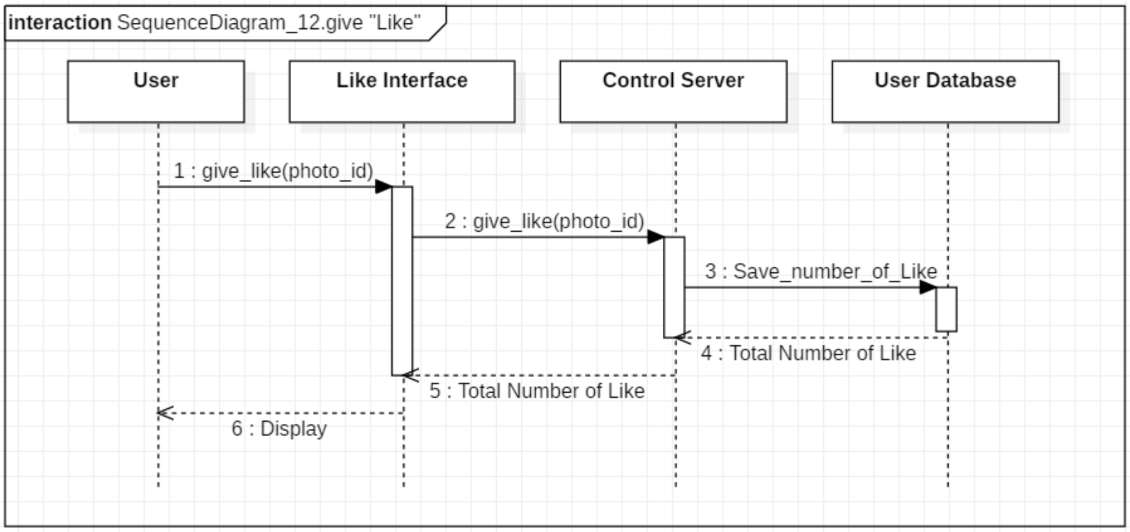


Figure 3-14 Give “Like” OP-11

3.3.2.12 View Photo

Identifier	**OP-12**
Initiator	SSAD 2.2.3 View Photo <Behavior>
Passed parameters	a) Photographer ID
Return values	photographer’s profile page
Exception handling	Failing to get response from server, a pop-up message “Fail” shows up.
Guards	Information of a photographer’s profile should be contained in DB.
Validation	The photographer that a user intends to see should be identical to information of a photographer’s profile saved in DB.
Messages	View photo request <request>
Exits	
Constraints	
Relates to	SSAD 2.2.3 BH-12

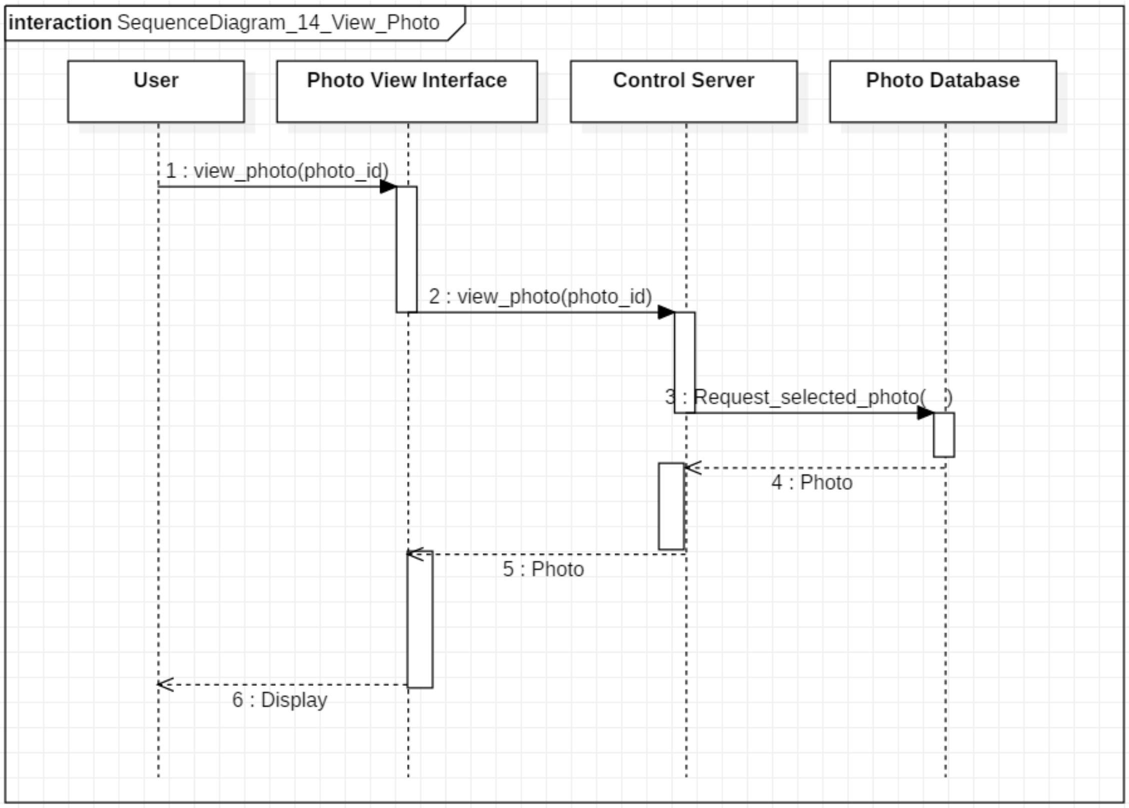


Figure 3-15 View Photo OP-12

4 Common Definition Language

Naver map API

A Naver map service draws visualized maps on website or mobile apps. It consists of a map JavaScript API and Static Map API.

Java

Java is a language developed by Sun Microsystems. Programs written in Java can run on many platforms without any changes or re-compilation.

Servlet

Java programs that runs on the Server-side. Can replace CGI scripts and requires fewer resources.

Photographer

One who is taking a photograph

Photographee

One who is photographed.

Retouch

Change original photo to be an edited photo by controlling some attributes like contrast, sharpness, tone, etc.

Keyword

Some specific words for searching

Review

Comment or feedback about a uploaded photo

Rating

Represent a photographer's rank in the platform by reflecting the number of “Like”

Like

A feature that allows users to show their support for specific comments or photos.

Profile

Photographer's information such as name, sex, location, photos, etc.

Platform

Such an operating system, software, etc. having a standard design for use with compatible programs, applications, etc.

User

Representing both of photographer and photographee

User Interface

General user interface(for mobile)