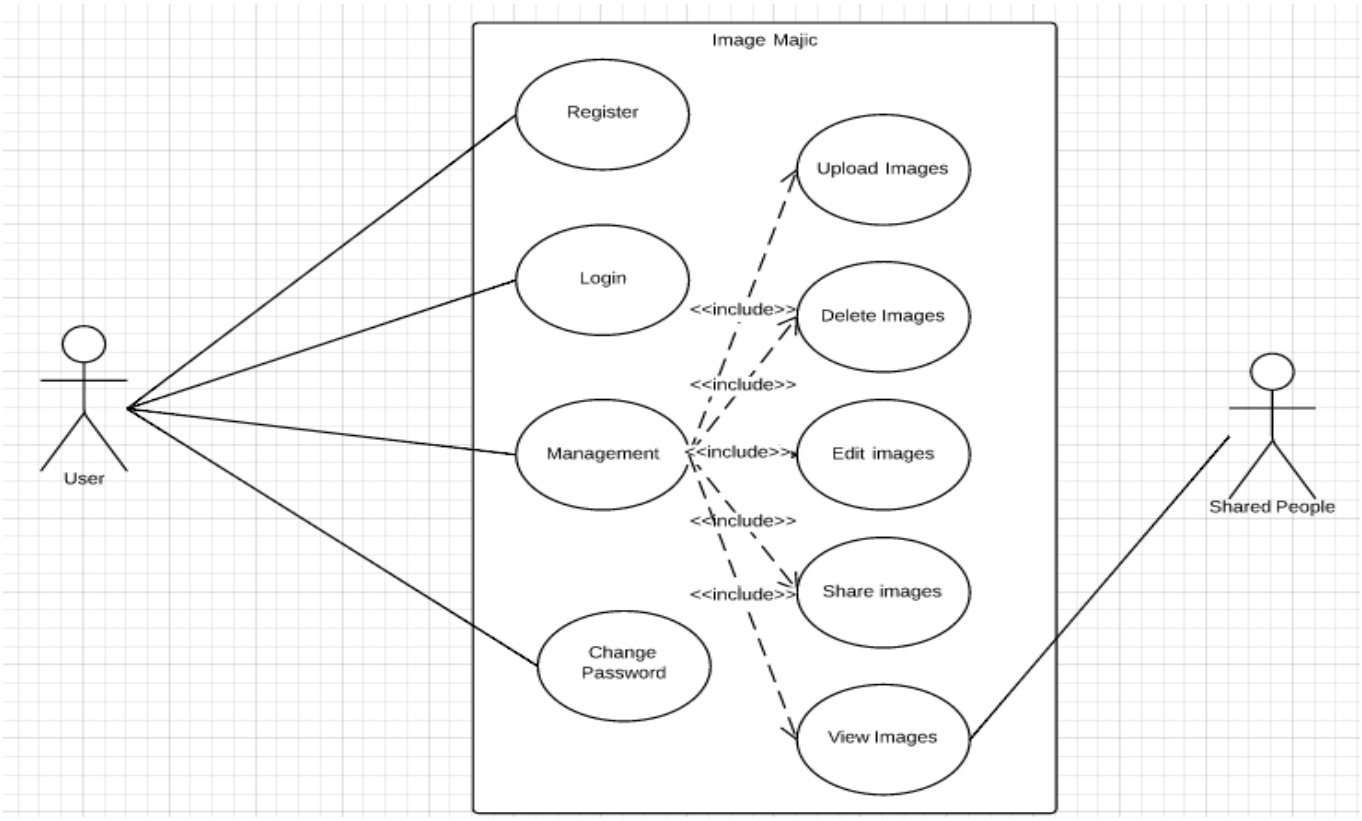
Image Majik System Design

Keran Wang(686976)

Lindong Li(655251)

Content

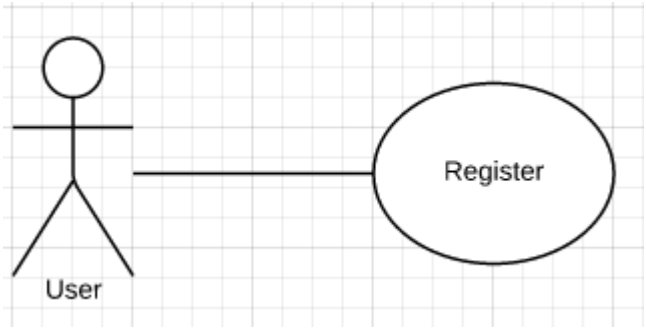
Part A Usecase Diagram



Overview

This is the overview of the use cases, we will describe these with use cases in isolation as below:

1. Register

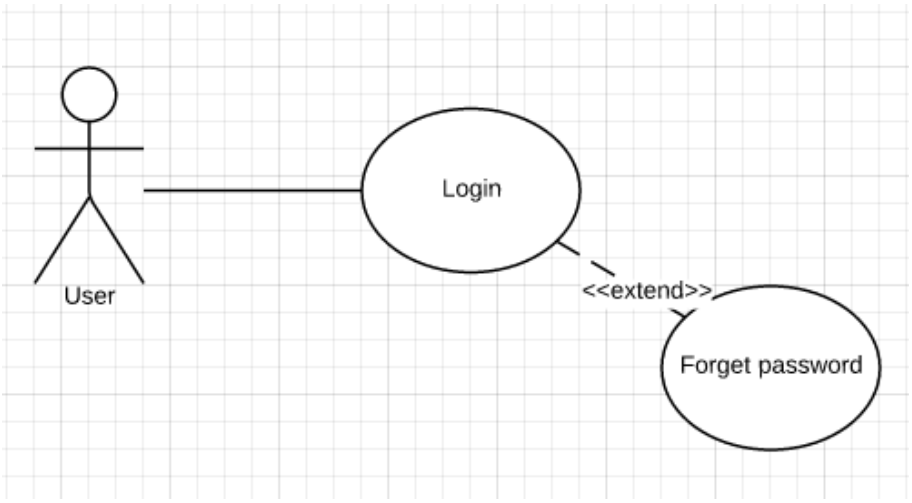


Use case Description

Use Case Name	**Register**
Trigger	A new person wants to be the user of the system
Preconditions	none
Basic Path	1. A new person selects a link to register 2. A new User form is displayed 3. A new person enter the User's details 4. A new person submits the form

Alternate Paths	5a. If User's details are incorrect, then go back to step 2
Postconditions	The new person has been a registered user of the system
Exception Paths	Register fails for down or network problem

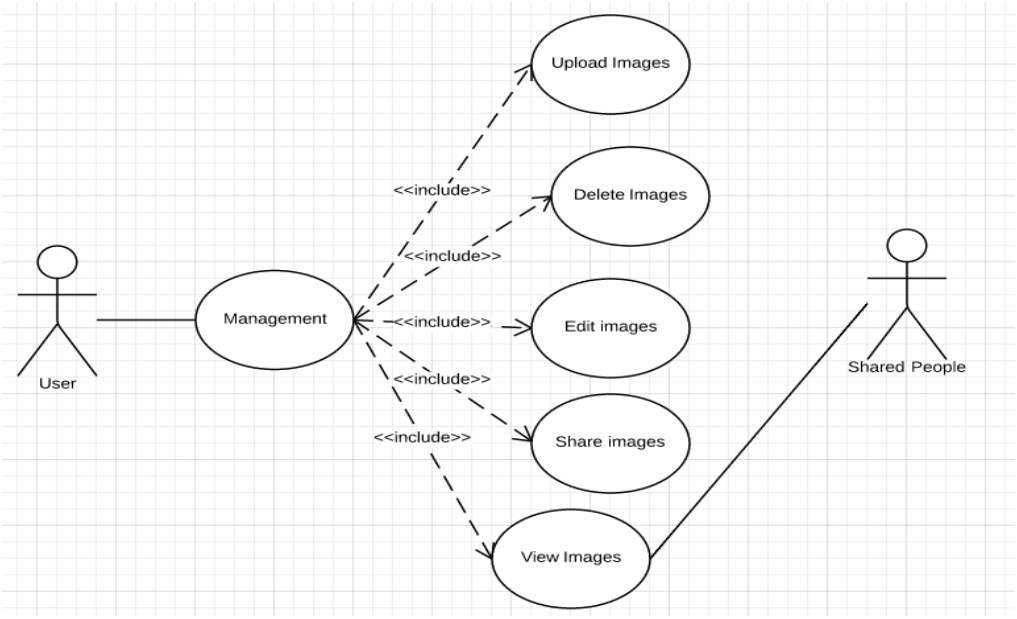
2. Login



Use case Description

Use Case Name	**Login**
Trigger	A User wants to log on the system
Preconditions	The user has registered
Basic Path	1. User selects a link to login 2. User supplies identification details 3. Extension Point Forget password: user who forgets password could get the password back here 4. If the user has validated identification details, then website is redirected to the personal center
Alternate Paths	4a. If the identification details are not validated then go back to step 2
Postconditions	The user has logged
Exception Paths	Login fails for server or network problem

3. Manage



Use case Description

Use Case	**Upload image**
Preconditions	The user has logged in the program
Trigger	The user clicks the upload button
Basic Path	1. The user click “upload” button 2. The program requests a photo from the user 3. The user select a photo in his or her computer 4. If the picture is valid, upload the photo to the server
Alternative Paths	4a. If the picture is invalid, the program tells the user that it is invalid and the program back to step2
Postconditions	An image has been uploaded
Exception Paths	Upload fails for server or network problem

Use Case	**Delete image**
Preconditions	The user has logged in the program There is at least one photo in the account to delete
Trigger	The user clicks the delete button
Basic Path	1. The user select the photos he or she wants to delete by version number 2. The program sends an alert to ensure that the user really wants to delete 3. The user clicks “yes” to delete 4. The program deletes the photo from the database
Alternative Paths	3a. The user click “no” button to cancel .
Postconditions	An image has been deleted
Exception Paths	Deleting fails for server breaks down or network problem

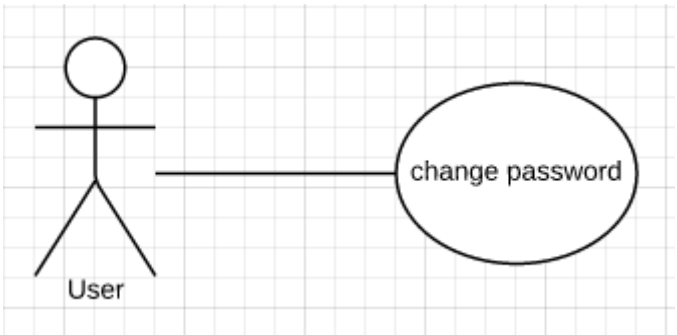
Use Case	**Share image**

Preconditions	The user has logged in the program There is at least one photo in the account to share
Trigger	The user clicks the share button
Basic Path	1. The user select the photos he or she wants to share by version number 2. The program asks the user who he or she wants to share with 3. The user enter the user names 4. The program check whether these accounts are valid 5. If the accounts are valid, share the photo with these accounts
Alternative Paths	5a. If there is an invalid account, tell the user which one is invalid and program back to 3
Postconditions	Images has been shared with another user
Exception Paths	Sharing fails for server breaks down or network problem

Use Case	**Editing image**
Trigger	A User wants to use filters to edit images
Preconditions	The user has logged and has at least one photo to edit
Basic Path	1. User selects a image 2. User selects a filter 3. A modified image is displayed 4. Extension point Save modified images: user chooses to save modified images
Alternate Paths	4a. The user cancel than change and the changed image won't be saved
Postconditions	A modified image has been displayed
Exception Paths	Edit fails for server breaks down or network problem

Use Case	**View image**
Trigger	A User wants to view images
Preconditions	The user has logged and has at least one photo to view
Basic Path	1. User clicks an image 2. The selected view displays the image
Alternate Paths	node
Postconditions	The image is displayed in a selected image page
Exception Paths	View fails for server breaks down or network problem

4. Change Password



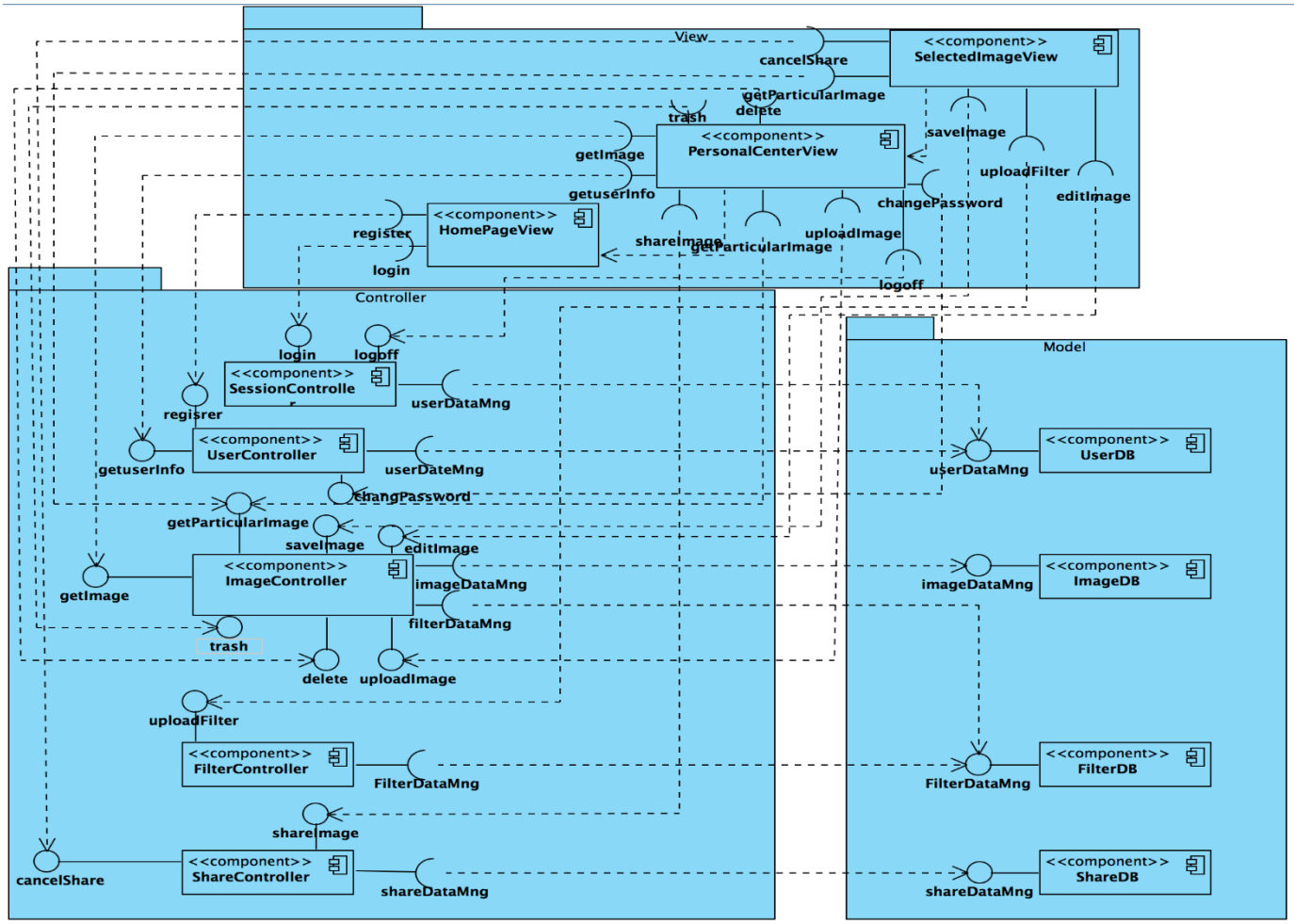
Use case Description

Use Case	**Change password**

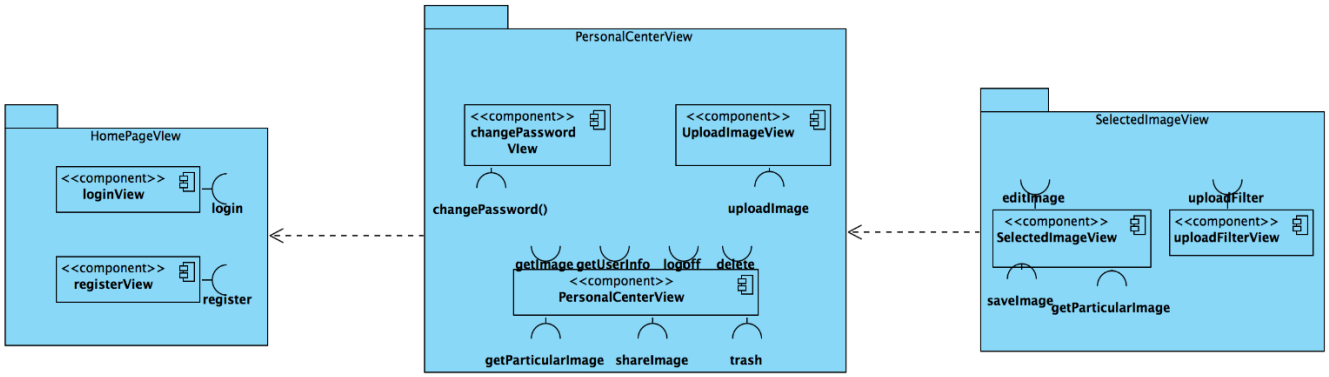
Preconditions	The user has logged in the program
Trigger	The user clicks the “change password” button
Basic Path	1. The user click the “change password” button 2. The program asks the user to enter his or her current password 3. The user enter his or her current password 4. The program check the password 5. If the password is correct, website will be redirected to the page to enter new password 6. The user enters his or her new password 7. The user clicks save button to save the new password 8. The program saves the new password
Alternative Paths	5a. If the password is wrong, ask for current password again 7a. The user click “cancel” button to cancel the change. 8a. If the new password format is wrong, back to step5
Postconditions	none
Exception Paths	Change fails for server breaks down or network problem

Part B Component Diagram

Conponent Diagram



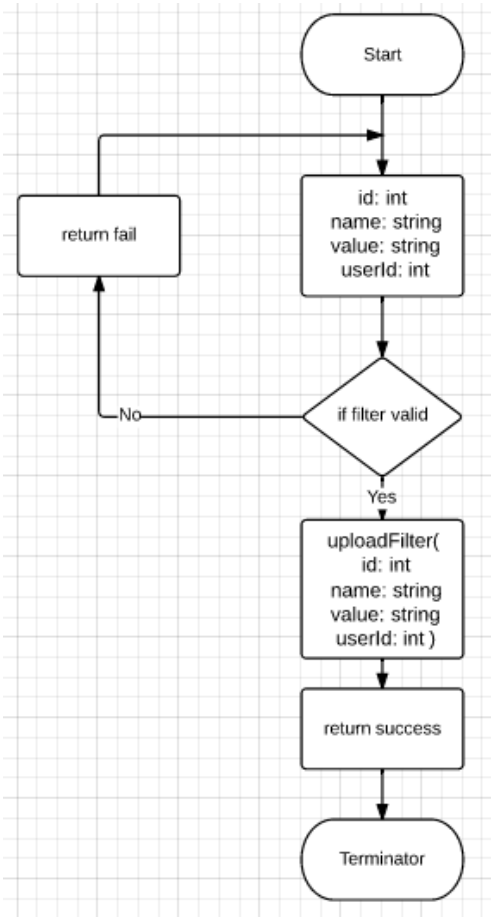
This is the description of the view part in detail.



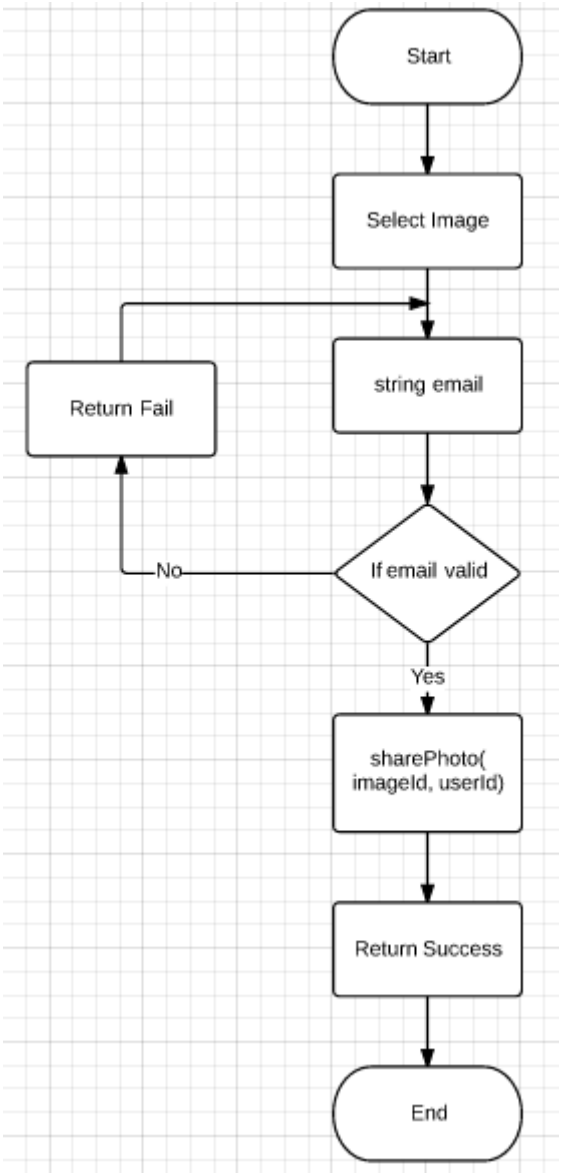
Control Flow Diagram

The rest of this part is the control flow diagram.

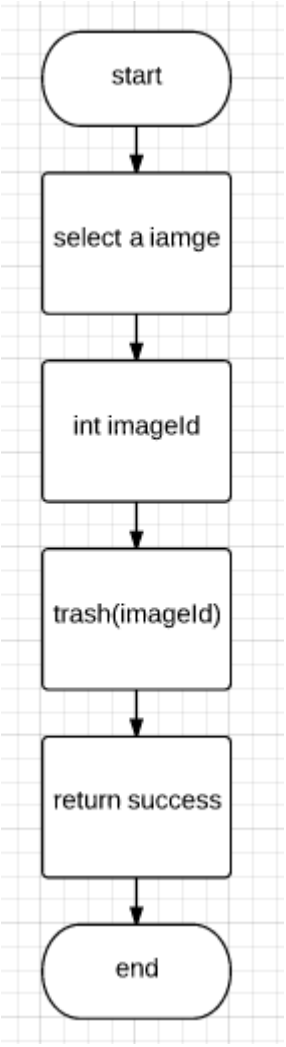
1.Update Filter



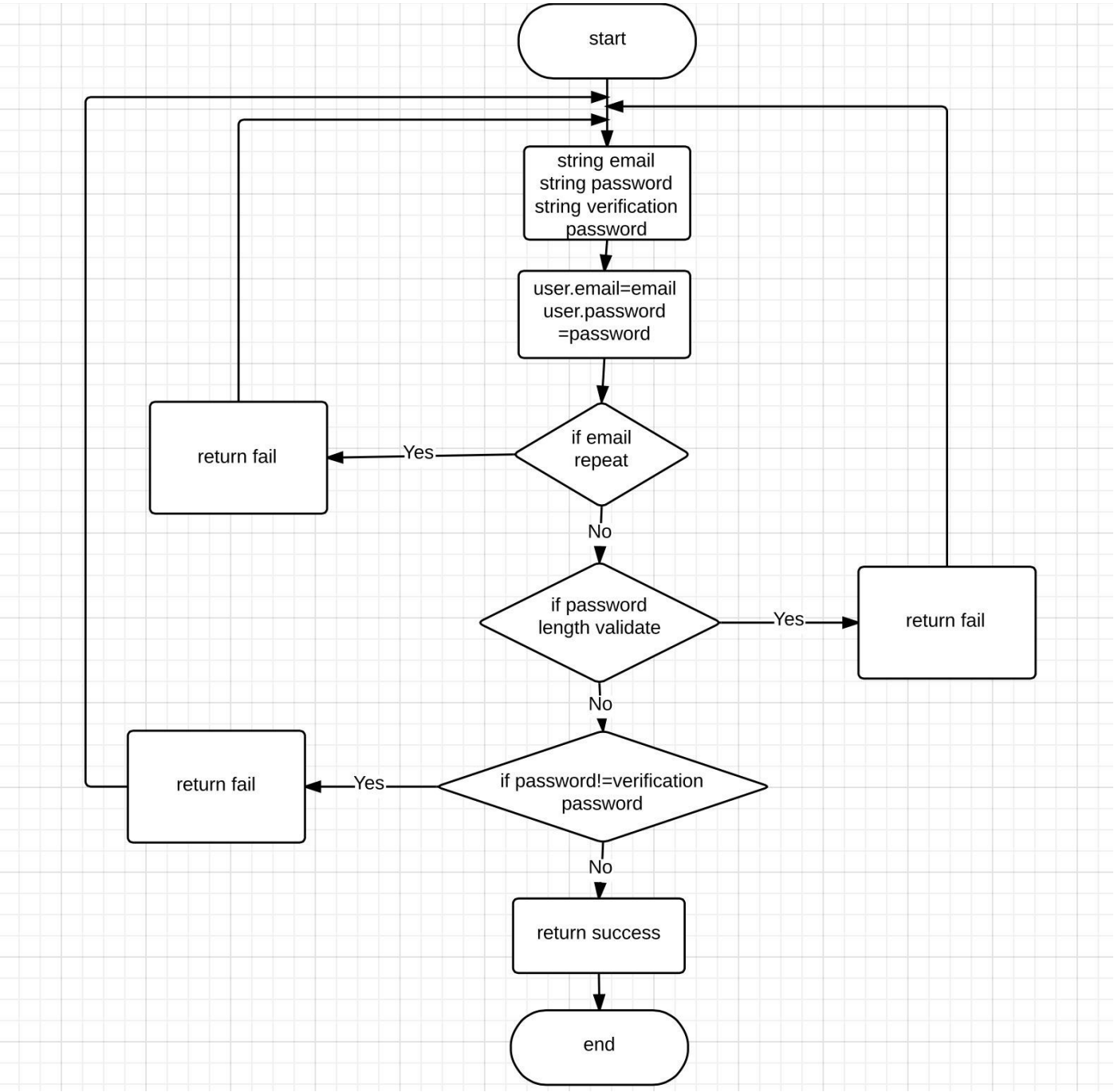
2.Share


Start
Select Image
string email
If email valid
No
Return Fail
Yes
sharePhoto(
imageId, userId)
Return Success
End

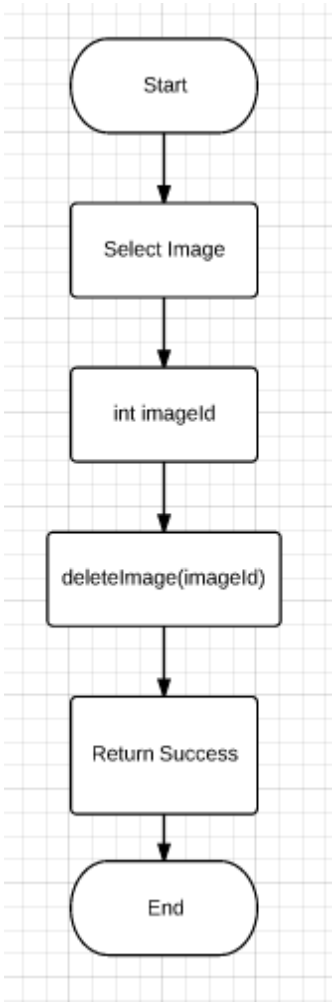
3.Trash


start
select a iamge
int imageId
trash(imageId)
return success
end

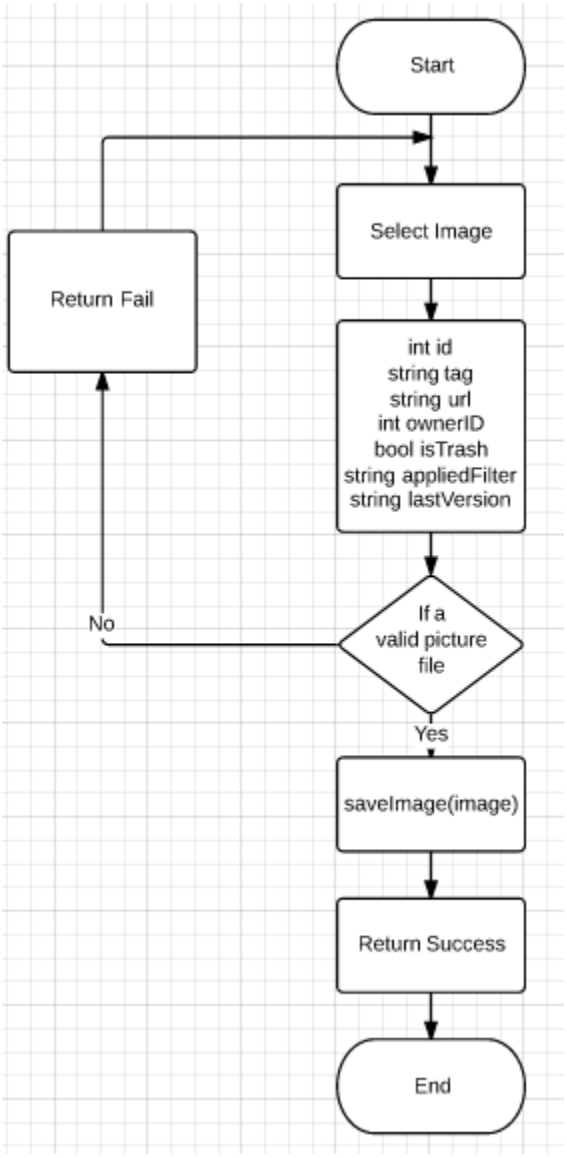
4.Register


start
string email
string password
string verification password
user.email=email
user.password =password
if email repeat
Yes
return fail
No
if password length validate
Yes
return fail
No
if password!=verification password
Yes
return fail
No
return success
end

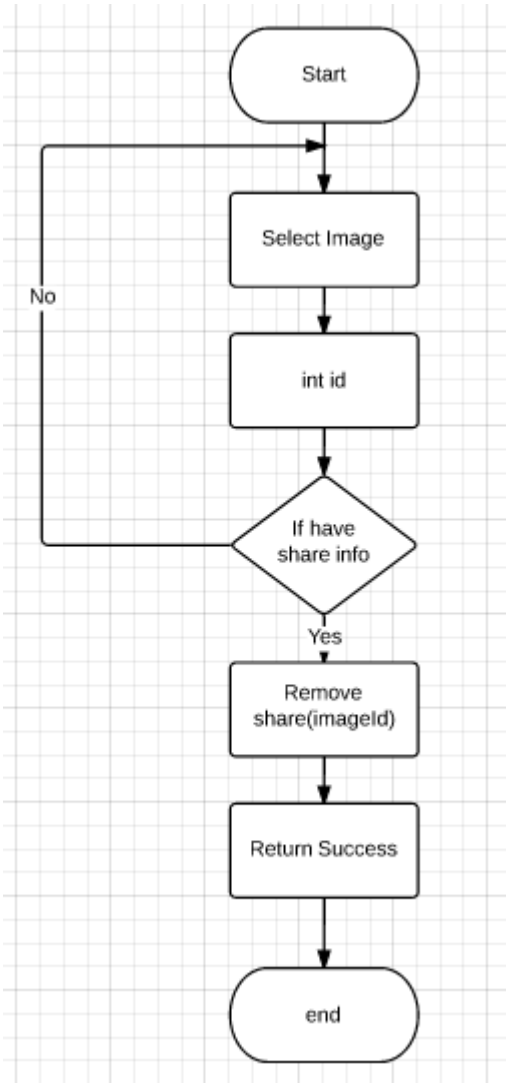
5.Delete Image



6.Upload Image

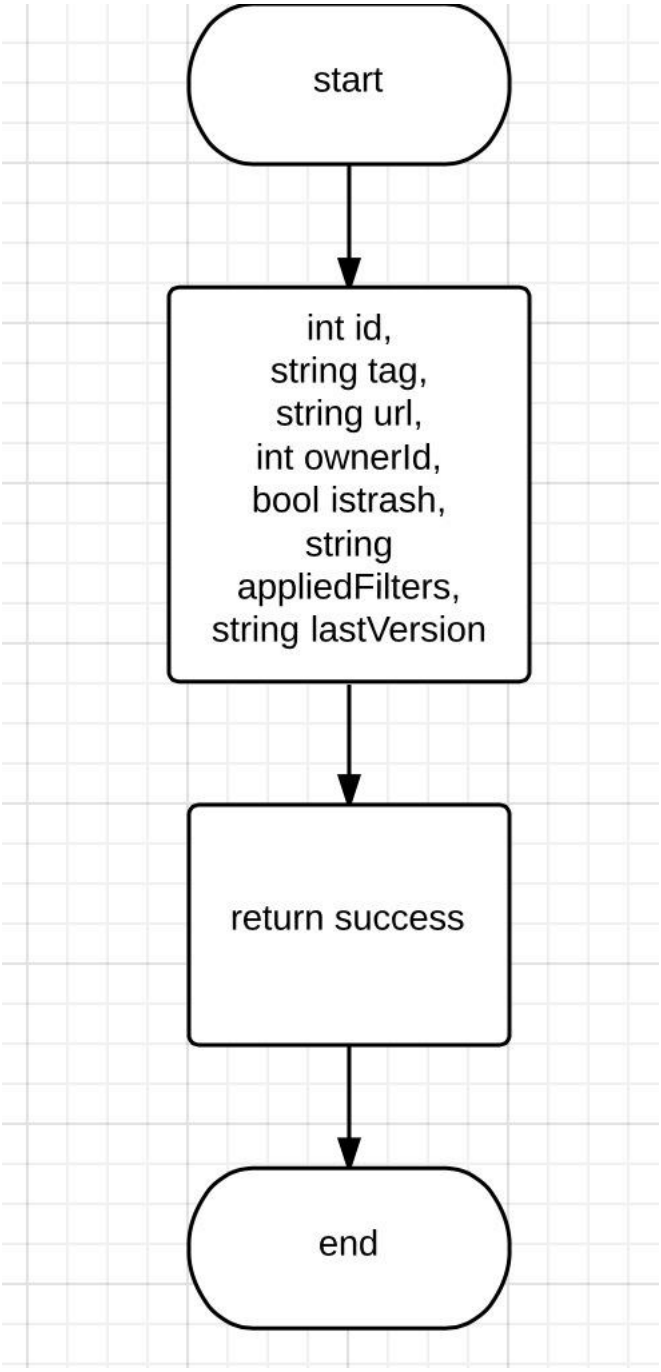


7.Cancel Share

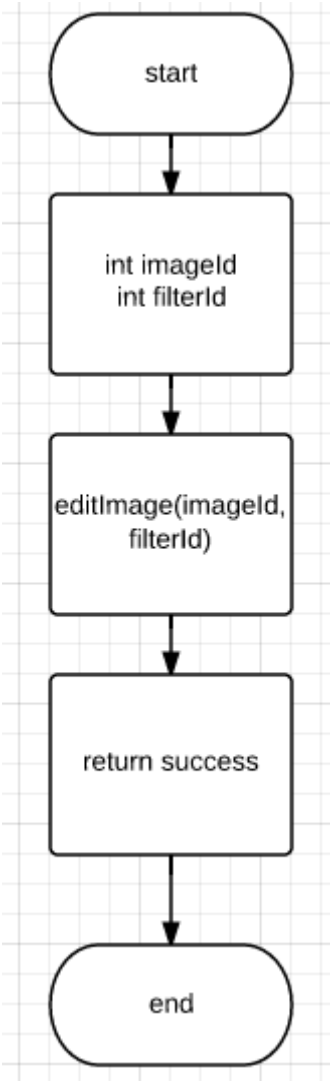


8.Change Password

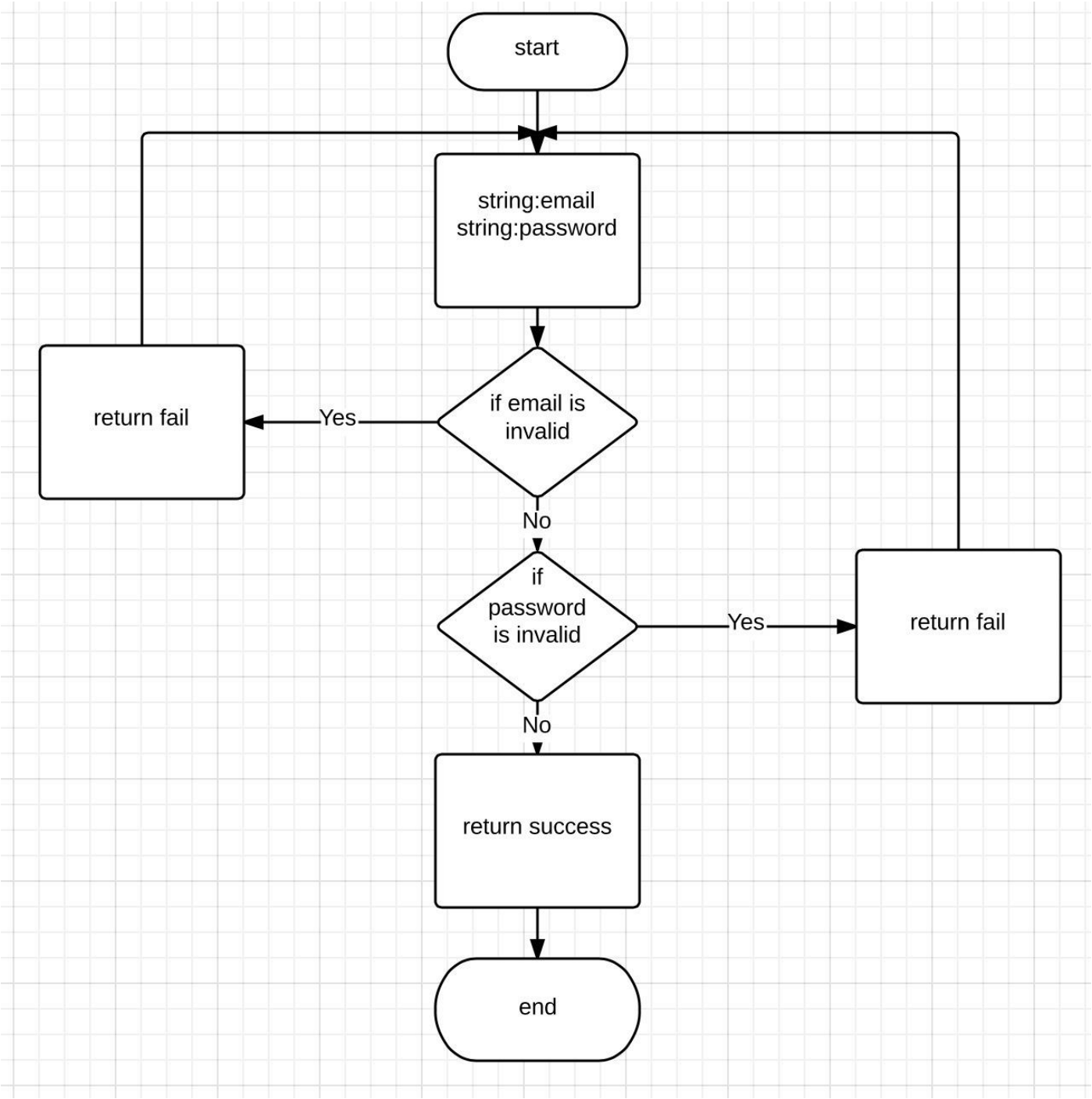
9.Save


start
int id,
string tag,
string url,
int ownerId,
bool istrash,
string
appliedFilters,
string lastVersion
return success
end

10.Edit


start
int imageId
int filterId
editImage(imageId,
filterId)
return success
end

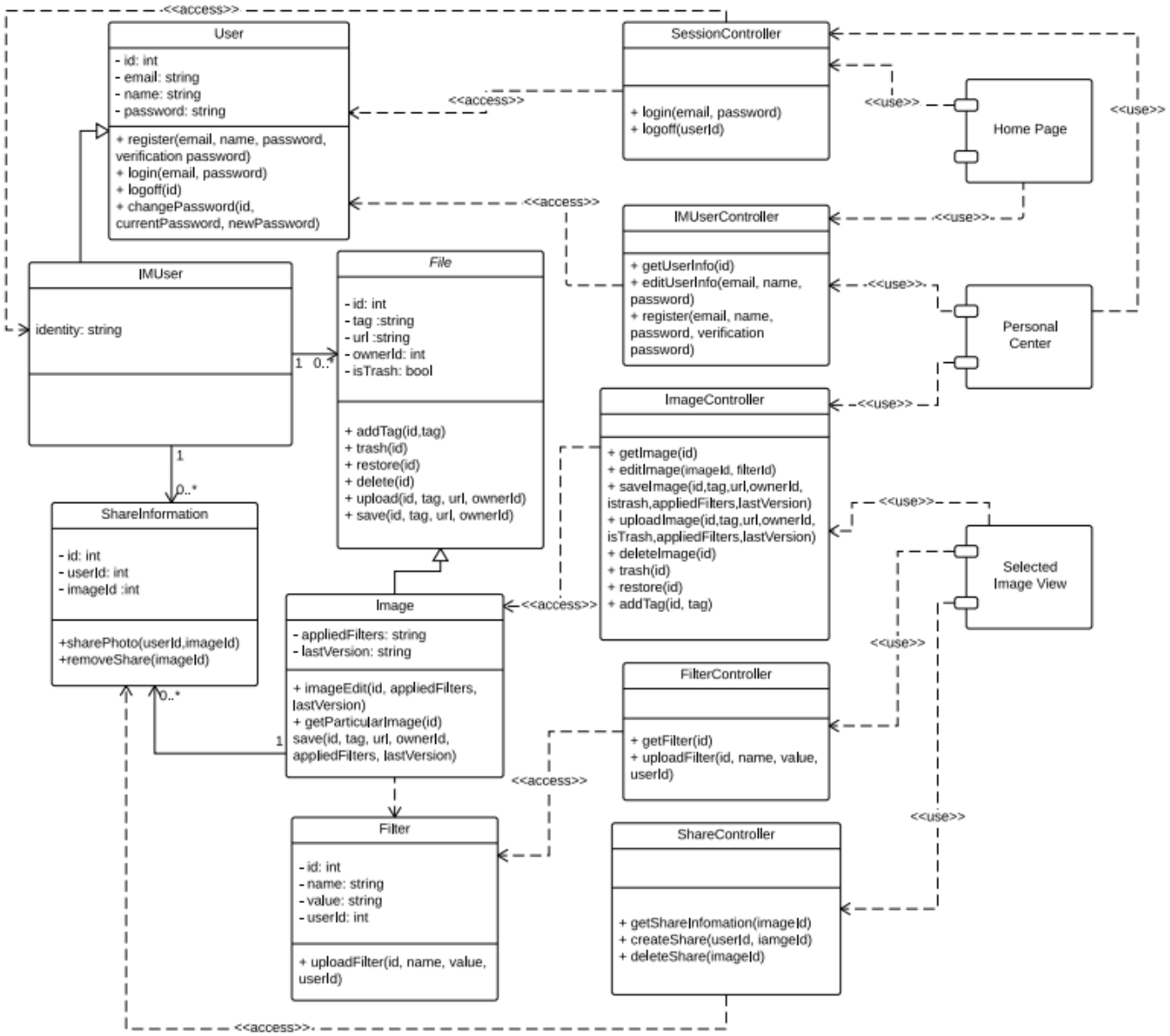
11.Login


start
string:email
string:password
if email is invalid
Yes
return fail
No
if password is invalid
Yes
return fail
No
return success
end

Part C Class Diagram

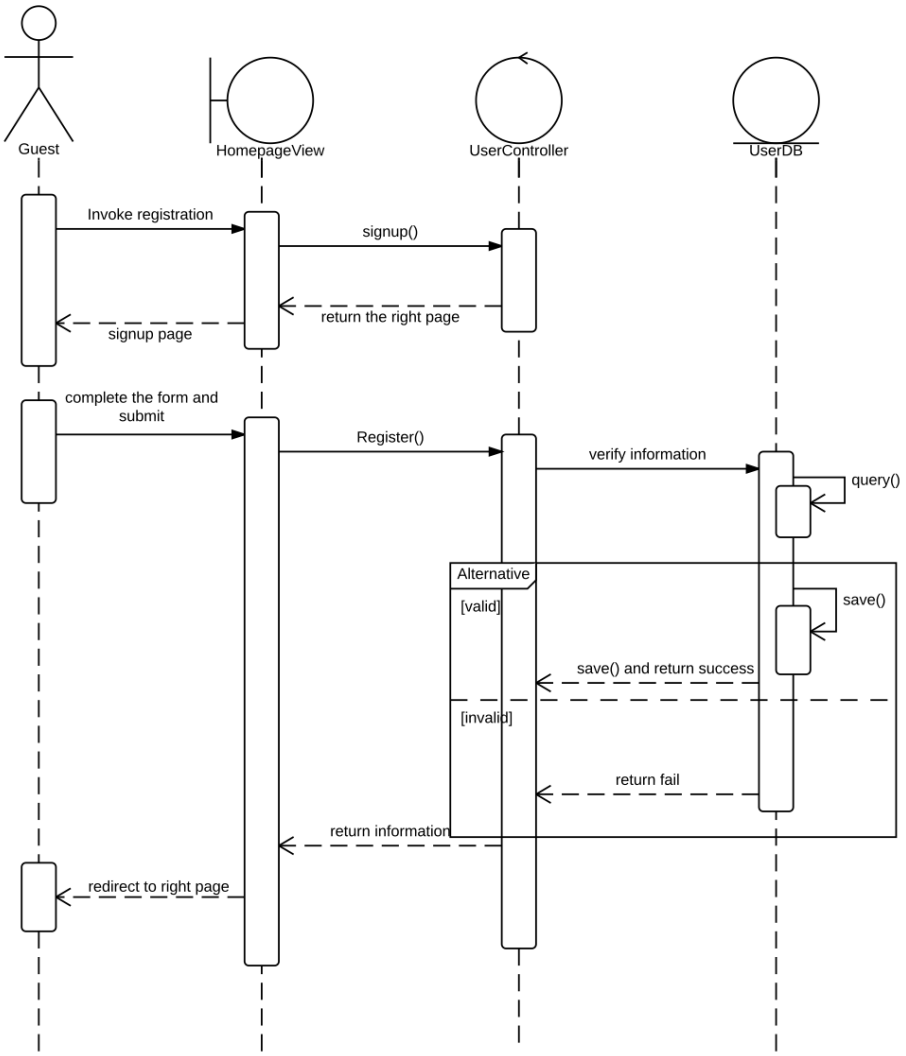
Class Diagram

This is the class diagram we design with MVC pattern and factory pattern. We isolate the program in three parts for MVC pattern, because this pattern makes the program high cohesion and low coupling. We add user class and file class for factory pattern, because this pattern makes it easier to maintain and future enhancements.



Sequence Diagram

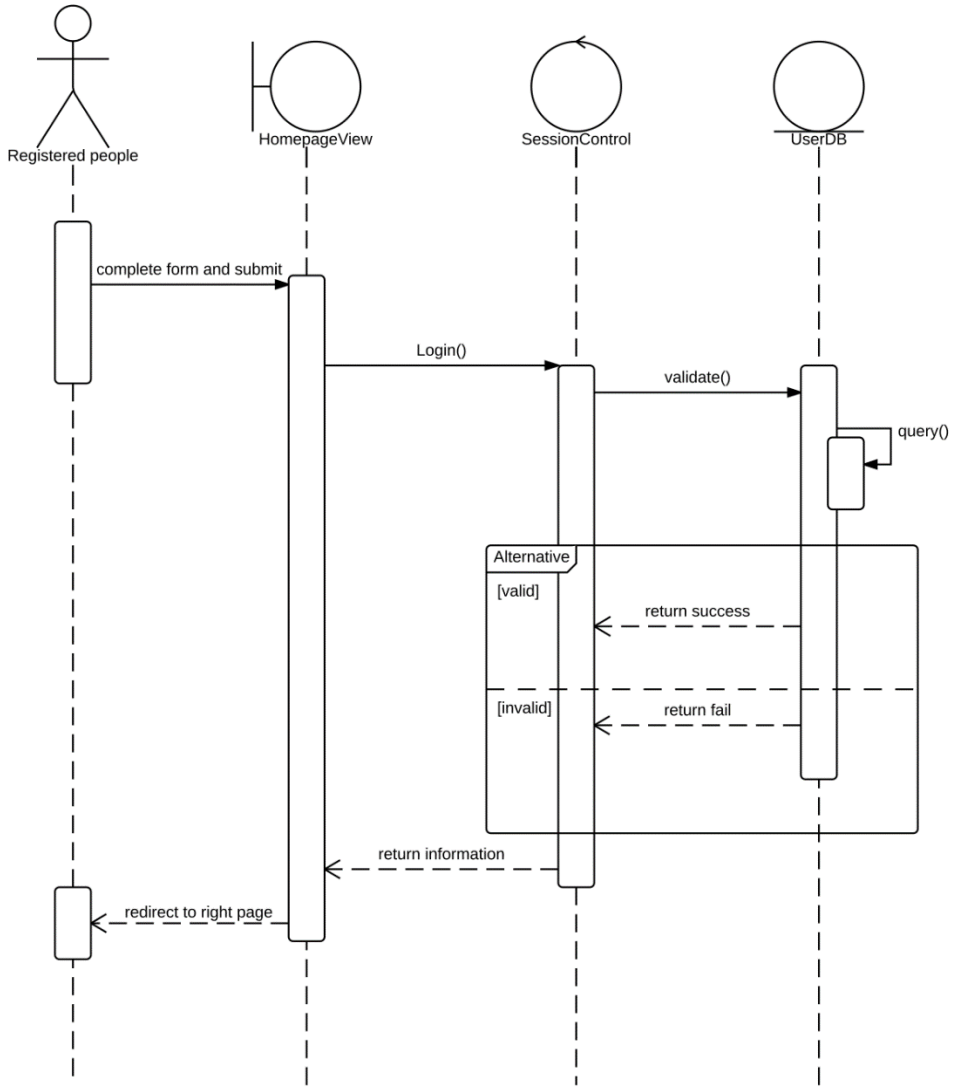
1.Register Sequence Diagram



Register

1. The user invoke registration, the user controller will return the signup page to the user.
2. The user complete the form and submit.
3. The user controller will verify if form details are valid by user model.
   3.1 If the form details are valid, a new user would be created and the user controller will return the personal center page to the user.
   3.2 If the form details are invalid, the user controller will return the sign up page to the user with the wrong message.

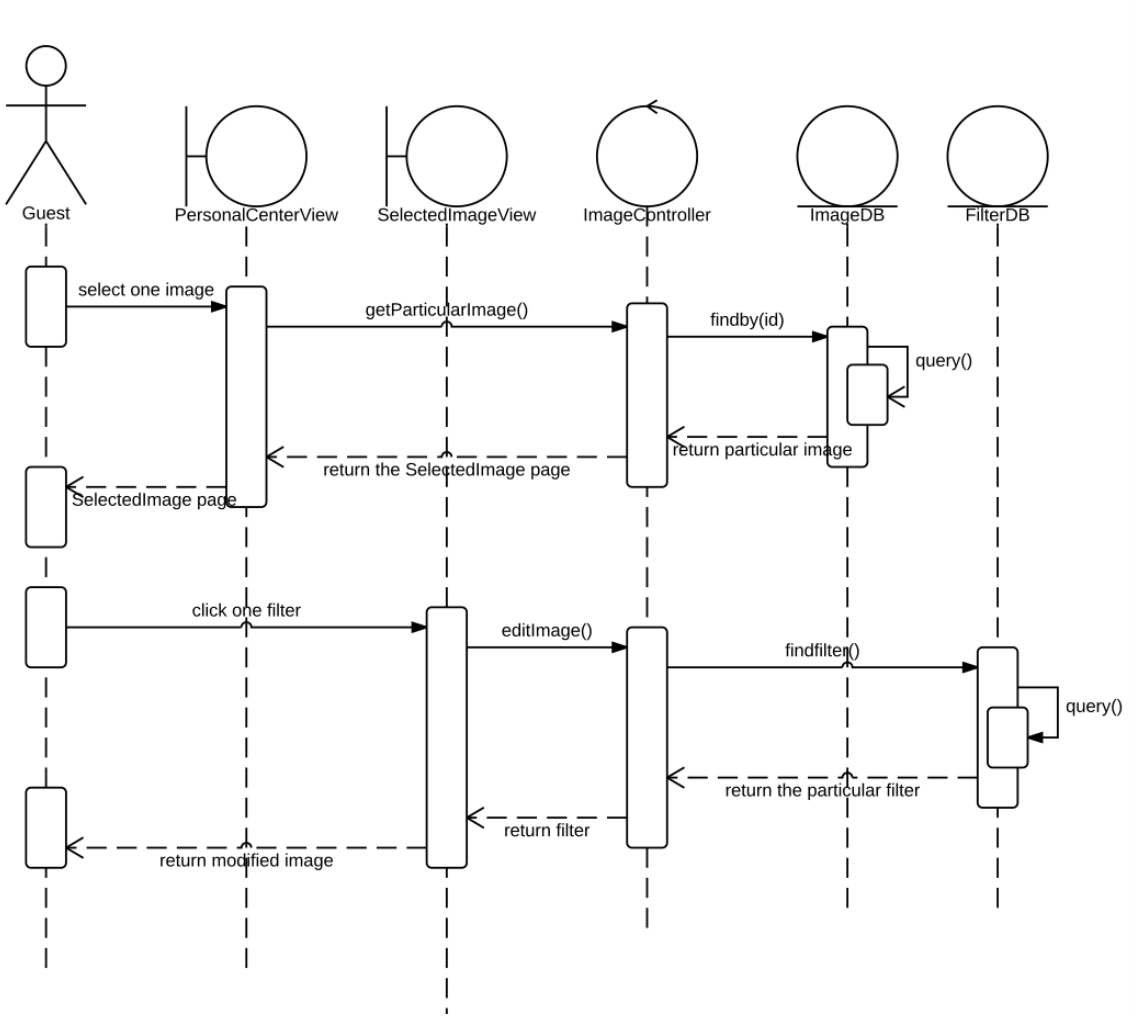
2.Login Sequence Digram



Login

1. The user complete the form and submit.
2. The session controller will verify if form details are valid by user model.
   2.1 If the form details are valid, the user controller will return the personal center page to the user.
   2.2 If the form details are invalid, the user controller will return the login page to the user with the wrong message.

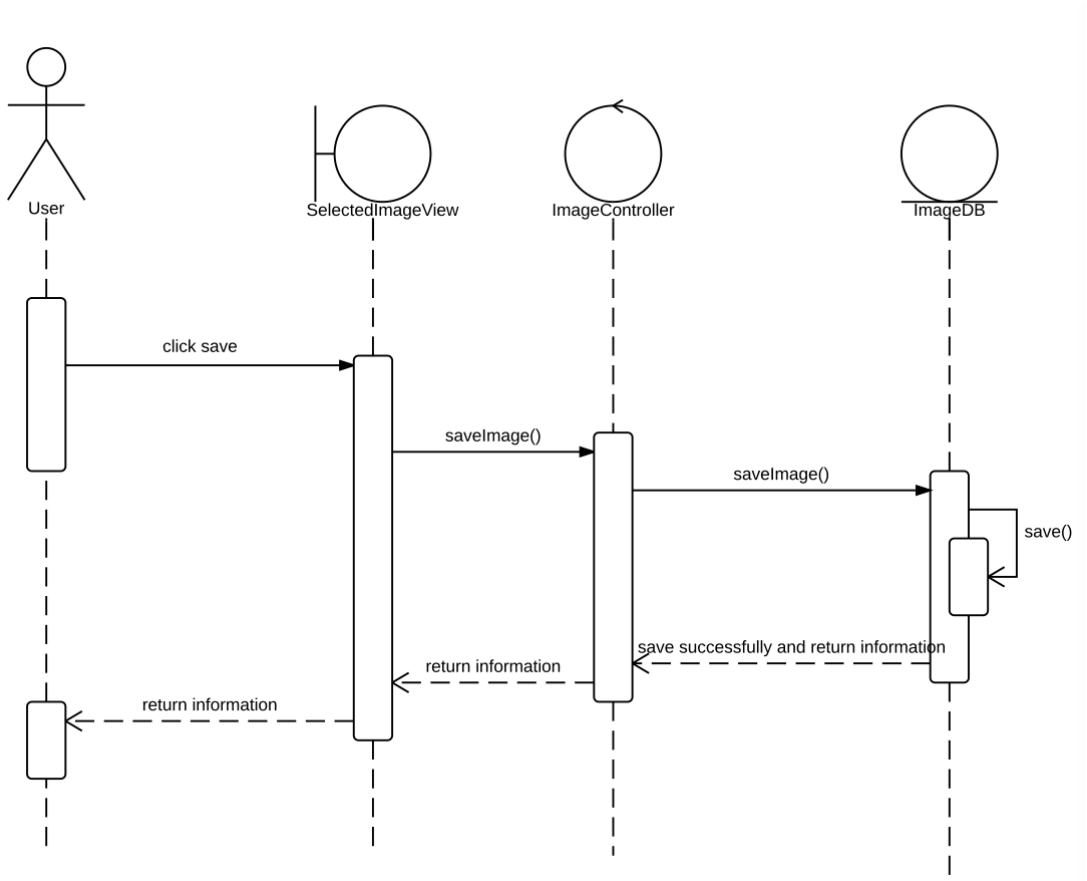
3.Edit image Sequence Digram



Edit image

1. The user select one image on the personal center page.
2. The image controller would access to the image model to get the particular image and return the selected image page to the user.
3. The user click one filter on the selected image page.
4. The image controller would access to the filter model to get the filter.
5. The image controller would edit the image with the filter and return a selected image with modified image to the user.

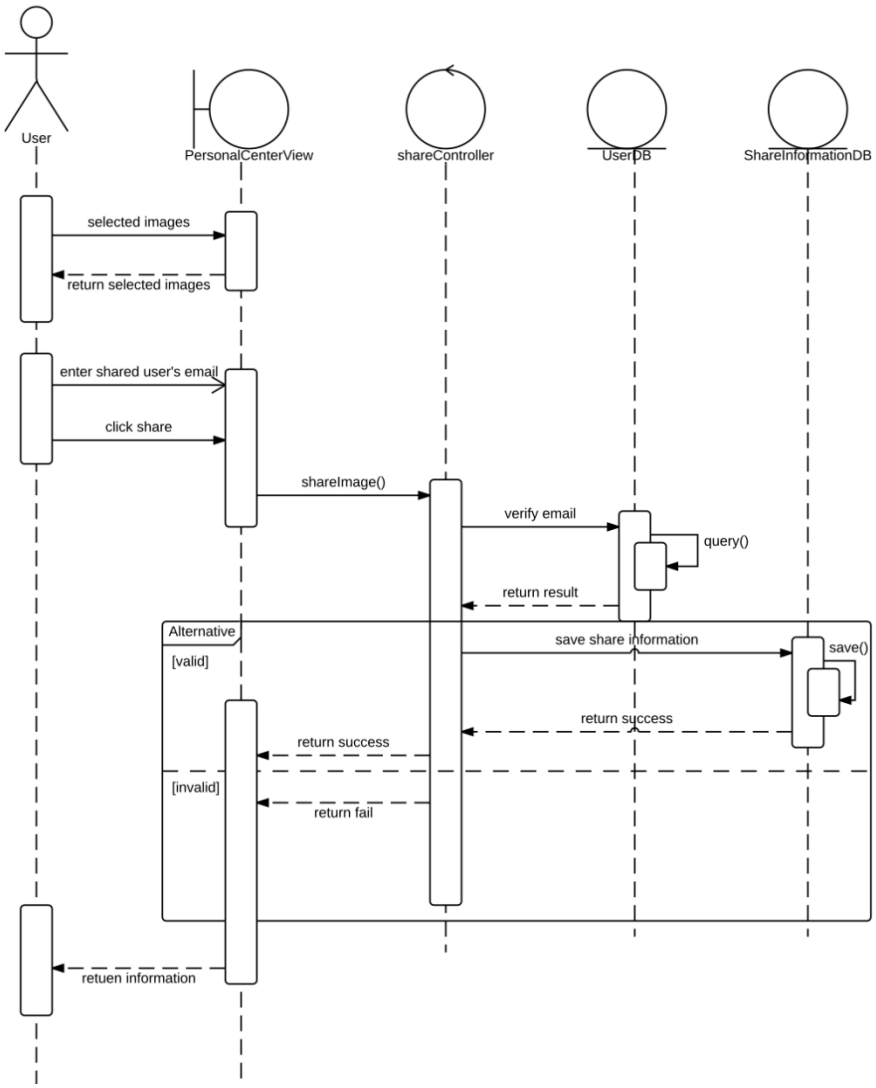
4.Save image Sequence Digram



Save image

1. The user click save on the selected image page.
2. The image controller would save the image by the image model.
3. The image controller would return the save success information.

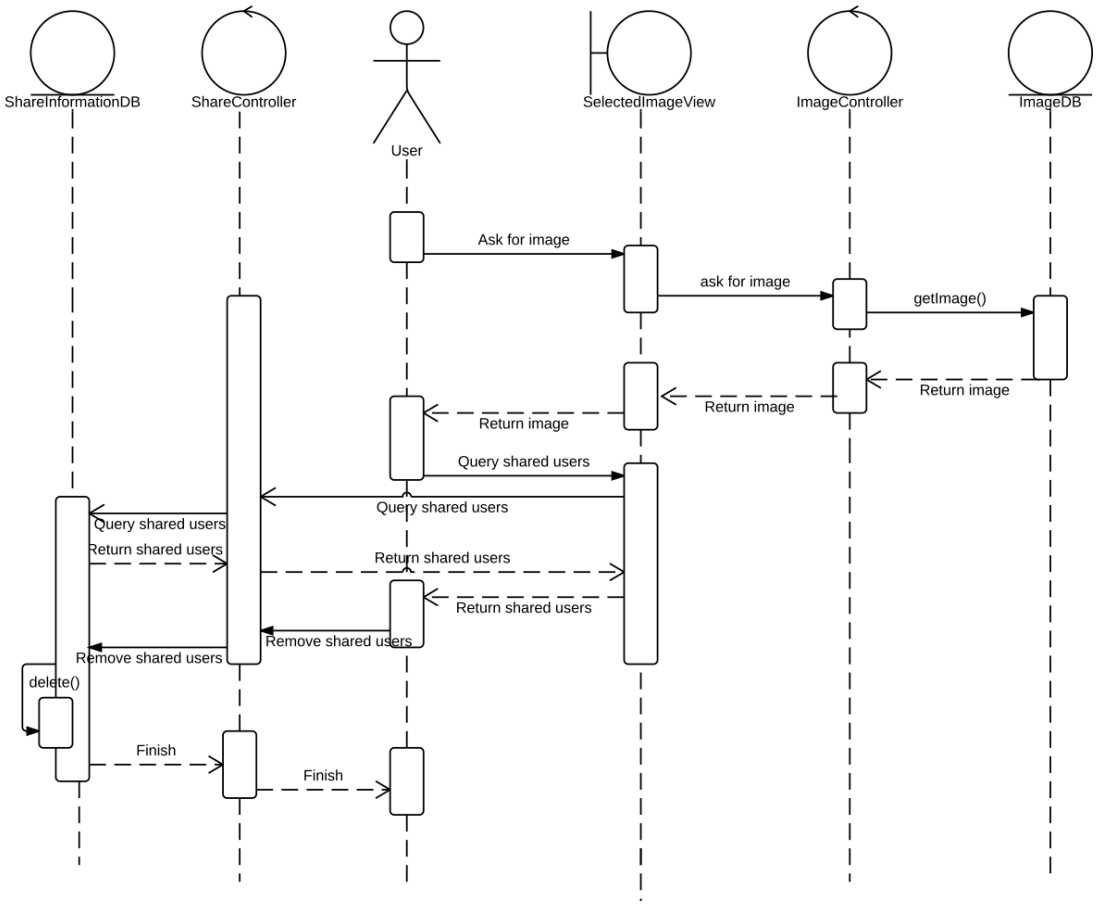
5.Share Sequence Diagram



Share Image

1. The user selects images on the personal center page and the page will return the selected images.
2. The user enters the email of person who the user want to share images with.
3. The user clicks share button.
4. The share controller would verify the email by querying in the user model.
   4.1 If the email is valid, the share controller would save the share information by the shareInformation model and then return success information to the user.
   4.2 If the email is invalid, the share controller would return the wrong information to the user.

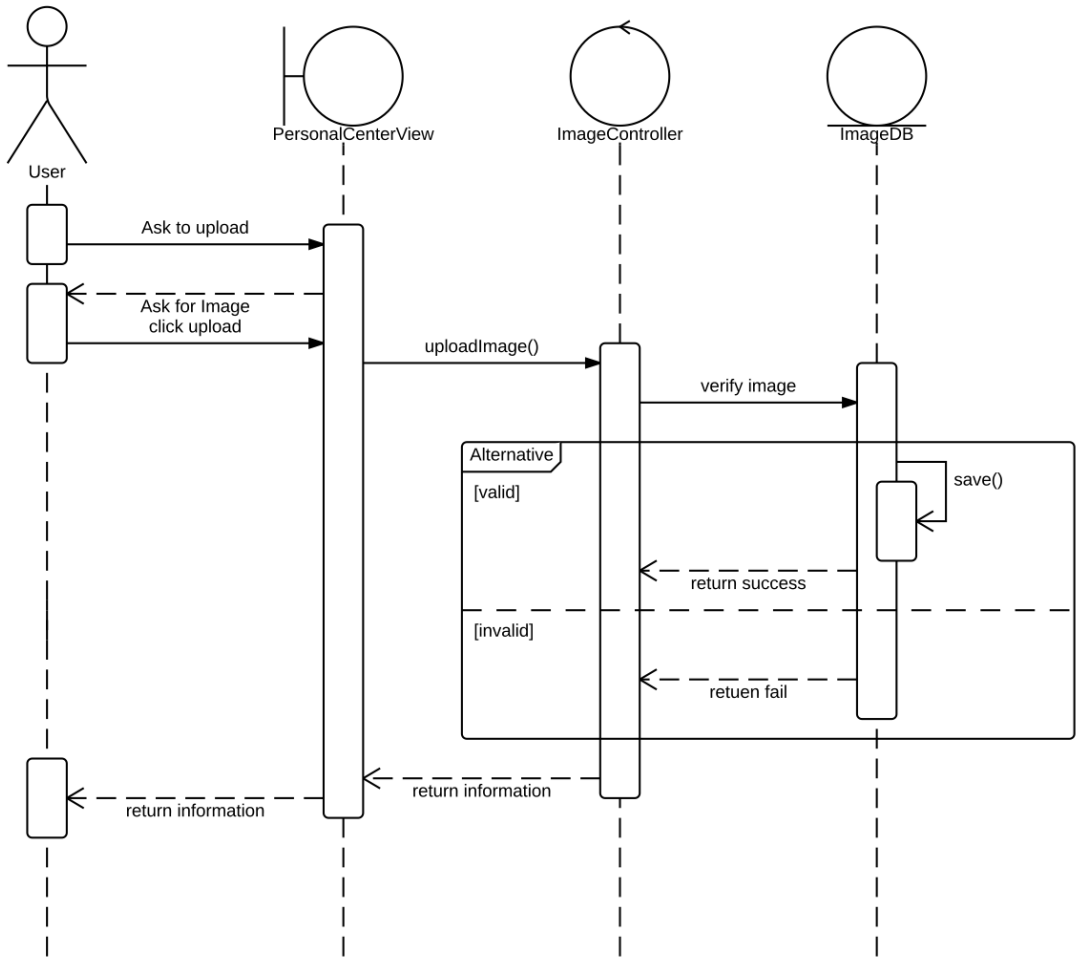
6.Cancel share Sequence Digram



Cancel share

1. The user calls for pictures with the selected image page, the request transfer through the image controller to the image model.
2. The image model returns the picture through the image controller to the selected image page and display it to the user.
3. The user queries for the shared user information about this picture.
4. The share controller receives the request and sends that request to the shareInformation model
5. The shareInformation model returns the information through the controller to the selected image page and the page displays the information.
6. The user makes the delete request through the controller.
7. The shareInformation model receives the request from the controller, remove the information and return notification.

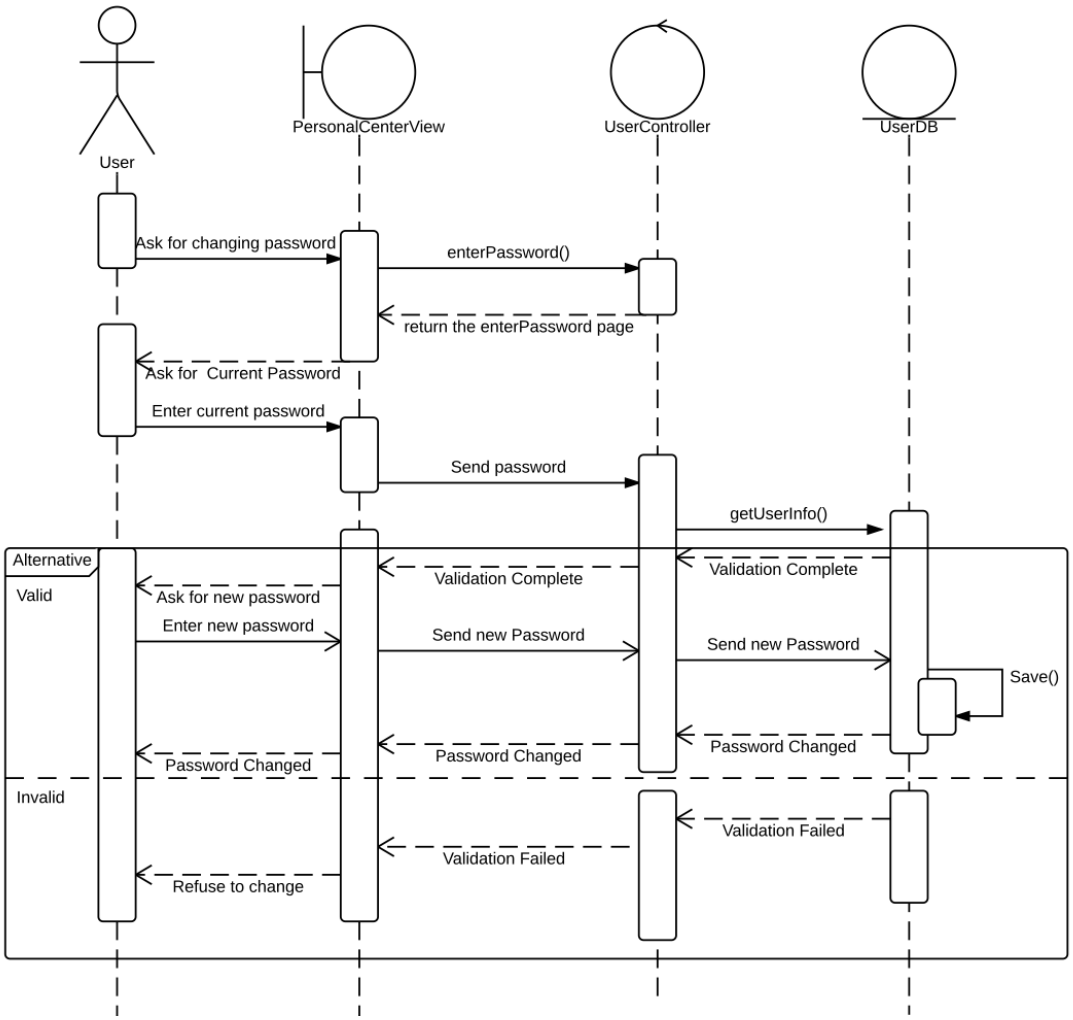
7.Upload image Sequence Diagram



Upload image

1. The user clicks the upload button to make a request for uploading
2. The personal center displays the upload page and asks the user for the image the use wants to upload.
3. The user selects one image and clicks upload button.
4. The image controller would verify if the image is valid.
   4.1 if the image is valid, the image controller would save the image by the image model and return success information to the user.
   4.2 if the image is invalid, the image controller would return the wrong information to the user.

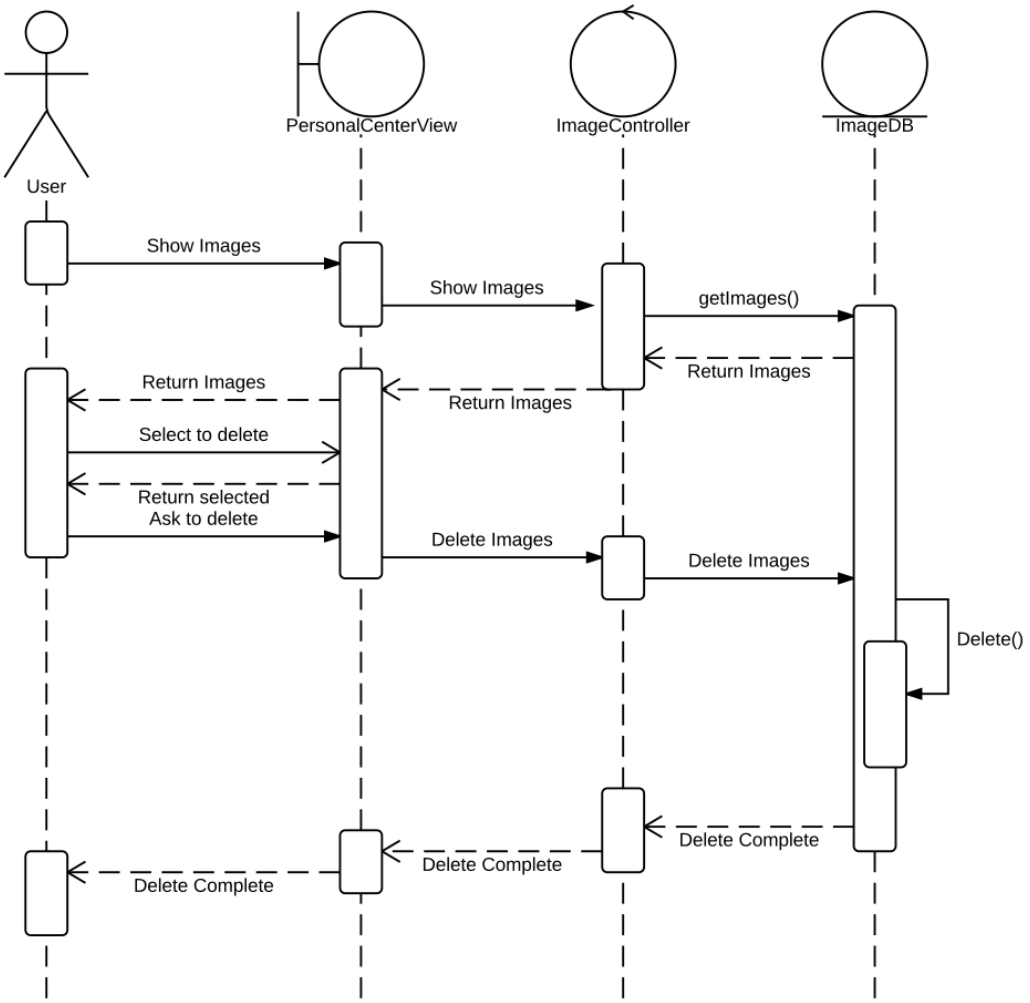
8.Change password Sequence Diagram



Change password

1. The user clicks “change password” button to make request to change the password.
2. The personal center receives the request and the user controller would return the enterPassword page to the user.
3. The user enter his or her current password, and the password is transferred by the user controller to the user model.
   3.1 If the user enters the right current password, the model returns the identification information through the controller to allow the personal center display the change password page
   3.2 If the use enters a wrong password, the model returns a failure notification through the user controller and redirect the user to enter his or her current password.
4. The user enter the new password and verification new password and then submit, the user controller would verify if the new password is valid.
   4.1 If the new password is valid, the user controller would save the new password by the user model and return the success information to the user.
   4.2 If the new password is invalid, the user controller would return the wrong information to the user.

9.Delete image Sequence Diagram



Delete Image

1. The user asks the image data on the selected image page.
2. The view sends the request to the image controller and the controller send the request to the image model.
3. The image model return the image information which is asked through the controller and display with the selected image view.
4. The user selects the images and then the selected image page would return the selected images.
5. The user makes the remove request with the selected image view
6. The request transfer from the image controller to the image model.
7. The image model would delete the images and the image controller would return success information to the user.