

10<sup>th</sup>  
May

2017

# Intrusion Detection System for Malicious Email

## Technical Report

BSHC (Honours) in Computing – Software Project  
National College of Ireland

Supervisor: Sara Kadry

## Table of Contents

Executive Summary .....	4
1 Introduction.....	5
1.1 Background .....	5
1.2 Aims .....	6
1.3 Scope.....	6
1.4 Technologies.....	7
2 User Classes and Characteristics .....	8
3 Requirements Specification .....	9
3.1 Functional Requirements .....	9
3.1.1 Use Case Diagram .....	10
3.1.2 Requirement 1: Register .....	10
3.1.3 Requirement 2: Upload Photo (Create Album).....	15
3.1.4 Requirement 3: Invite Users .....	19
3.1.5 Requirement 4: Share Photo .....	21
3.1.6 Requirement 5: Edit Friend List (Trusted User).....	22
3.2 Non-Functional Requirements .....	29
3.2.1 Performance/Response time requirement .....	29
3.2.2 Safety Requirement .....	29
3.2.3 Security Requirement .....	29
3.2.4 Requirement Attributes .....	30
3.2.5 Business Rules.....	30
3.2.6 User requirement .....	31
3.2.7 Maintainability requirement.....	31
3.2.8 Portability requirement.....	31
3.2.9 Extendibility requirement.....	31
3.3 Design and Architecture.....	32
3.4 Graphical User Interface (GUI) .....	33
3.5 Testing .....	42
3.6 Evaluation .....	43
4 Conclusion .....	43
5 Future Development.....	43

6	References .....	44
7	Appendix .....	44
7.1	Monthly Journals .....	44
7.2	Project Proposal.....	54

## Executive Summary

Nowadays, users all around the world use email as their fundamental method to share information over the web. The network providers allow all types of email for the purpose of communication. During this transfer of information some malicious emails are received which can cause problems either at the server side or at the client side. In this project, we propose an intrusion detection system designed to detect these malicious emails.

In recent times, some of the most dangerous security threats against private user data at home and in the workplace is phishing. Phishing has become an extremely common form of cyber attack. It consists of defrauding people by luring them to fake websites where users unknowingly provide personal details such as login information and credit card details. These fraudsters appear as a trusted third party, like a well-known bank. The most common methods of phishing are done by email. Once these details are acquired they can be used in the practice of identity theft or credit card fraud. In the past, efforts have been made to stop these attacks by identifying phishing sites using plug-ins, but these efforts have been made in vain by emerging blocking techniques, which render them useless. There is an abundance of these types of attacks, so much so, that the everyday user will be in danger whether they know it or not. In this project we propose an Intrusion Detection System for identifying these types of malicious emails and root them to their source to evaluate. This will be made possible by using a data capture facility that will categorize a number of incoming emails as potentially malicious actions and an evaluation system that will send crawlers to websites related to these detected emails to determine their true intentions. By detecting malicious emails in incoming traffic, this filters a user's inbox and removes the requirement of a user being trained in the practice of secure web browsing. As most users are not trained in this manner, this system will prove quite useful. The Intrusion Detection System (IDS) will detect malicious emails and ensure that all of the incoming emails/data is not harmful. When a malicious email is detected, the next step is to send crawlers to these phishing websites that are linked in these emails and also the website that it is trying to impersonate. An algorithm then strips both sites down and compares them using a scoring system for the difference between the two – ultimately deciding whether or not it is a phishing website. This Intrusion Detection System will be implemented into a photo sharing web application with email functionality.

# 1 Introduction

## 1.1 Background

I am a current employee of ACIA (Aon Centre for Innovation and Analytics) working part-time during my studies. I have been an employee of ACIA since the summer of 2015 when I applied for an internship. I have not been working with this company for long but I have seen my fair share of misleading and malicious emails in the workplace. Even in a technology-based company such as this, employees were still the victims of these phishing attacks.

It just goes to show how complex and deceitful these attacks are becoming. This was the basis of my decision to design and develop an Intrusion Detection System for my final year project. It was one attack in particular that was successful among a small number of my colleagues. It occurred during my six-month placement for third year, around the month of July. A number of staff began to receive fake emails purporting to come from the Revenue Commissioners and were warned soon afterwards that although the email address seems valid, it is just a piece of text and can easily be faked. Email addresses are not verified and cannot be relied on as proof of the identity of the sender organization. These emails seemed very professionally made and tried to trick users into believing that they were due a tax refund, mostly in the range of around €160, and enticed them to enter bank details. It wasn't until the very next day that similar emails started to roll in. Colleagues were warned to remain extremely vigilant and never to click on suspicious links or provide any sensitive information. This time the phishing emails were in disguise as invoices from Apple and again, looked very legitimate. An office wide email was sent out from IT to warn that proceeds from such scams are quickly transferred offshore, usually through multiple countries/banks and prove impossible to retrieve, even where amounts are greatly in excess of €25.99.

Although these types of attacks are very unfortunate, it was good to experience it firsthand in the workplace, and to see how the office dealt with the situation. However I felt that it wasn't enough and I wanted to create something that would tackle this issue head on, eradicating the problem on the user's end.

## 1.2 Aims

The purpose of this project is to provide an easy and user-friendly way to allow users to have a safe working environment when using an online web application with email functionality. The main objective of this project is to maintain a safe and user-friendly environment and to eradicate any incoming threats via email. This product will consist of two main components: an Intrusion Detection System and a web application that will allow users to share photos with friends.

## 1.3 Scope

The web application will allow users to sign up to a service which will provide a safe environment for them to share photos with their friends. To sign up to this service, each user must have a valid email address. Once a user has become a registered member, they can invite other users via email to join the service. They do this by sending an email, which will include an invitation. When the receiver of the invitation email accepts the invite, they will be brought to the web application to register as a member. Once completed, they will then become a friend (trusted user) and will appear on a trusted user friend's list. This will connect both users and allow them to share photos on the website.

To ensure the safety and security of the process of sending emails, we have implemented an intrusion detection system that will detect malicious emails. When sharing photos, any photos received from an untrusted user will be marked as malicious email and be placed in a junk folder. This system will co-exist with the web application and improve the overall security for its members. The Intrusion Detection System uses a data capture facility that will categorize some of the incoming emails as potential malicious activities. By doing so, we create a user-friendly environment where people can share photos without any concern.

By detecting malicious emails in incoming traffic, this system will filter a user's inbox and eradicate the need and cost of requiring users to be trained in the practice of secure web browsing. The Intrusion Detection System (IDS) will detect phishing emails and ensure that all incoming traffic is not harmful. The project is specifically designed for the use of users who sign up to our photo-sharing site with a valid email address. The product will work as a complete user interface for the site.

Anybody with a valid email address and an Internet connection can use this system. It is especially useful for anyone who needs a secure and safe option to share photos. The project can very easily be modified given different scenarios. New features will be able to be added when needed. This makes the aspect of reusability possible. The language that was used for the development of this project is PHP as it is very beneficial over other programming languages in the areas of performance, tools available, cross platform compatibility, libraries and cost.

## 1.4 Technologies

This system will consist of three main components: a Spam filter to detect unsolicited emails, a Simple Mail Transfer Protocol (SMTP) server which is the de facto standard for sending email over the internet, and Apache web server which is an open source web server. The IDS scanner will scan all received e-mails and will mark some of the messages as malicious activities and then enters them into a database. Emails are classified as malicious if they contain embedded HTML or if they are received from an untrusted user. MySQL will be used to store information regarding the detected malicious emails, where data entered into the database consists of the content of the email and time of receipt. A background process will run to scan all URL's listed on marked emails, strips HTML tags and determines whether or not they have been earlier observed. The spam filter will mark email as phishing emails if they carry some of the following attributes:

- Received from untrusted user
- URL including IP addresses
- HTML masked URLs
- Emails containing encoded HTML
- Cross site images

## 2 User Classes and Characteristics

Different services are provided by the IDS secured web application based on the type of user. In this case we have a number of different users: Photo Owner and Trusted User. A photo owner is a member who uploads a photo to the site and invites a user to join the service to view the photo. A trusted user is anyone who receives this email invitation and accepts, placing them in the sender's trusted user friend's list.

### **The features that are available to a Photo Owner are:**

- Log in/ log out
- Change Profile Picture
- Upload photo
- Delete Photo
- Request as friend
- Block user
- Share/Send Photo
- Invite user
- Create Album/Gallery
- Tag photo
- Add comment

### **The features that are available to a Trusted User are:**

- Log in/log out
- View/Download Photo
- Edit trusted user friend list
- Comment on a photo
- Can delete/block other members/friends
- Tag friends/places in photos

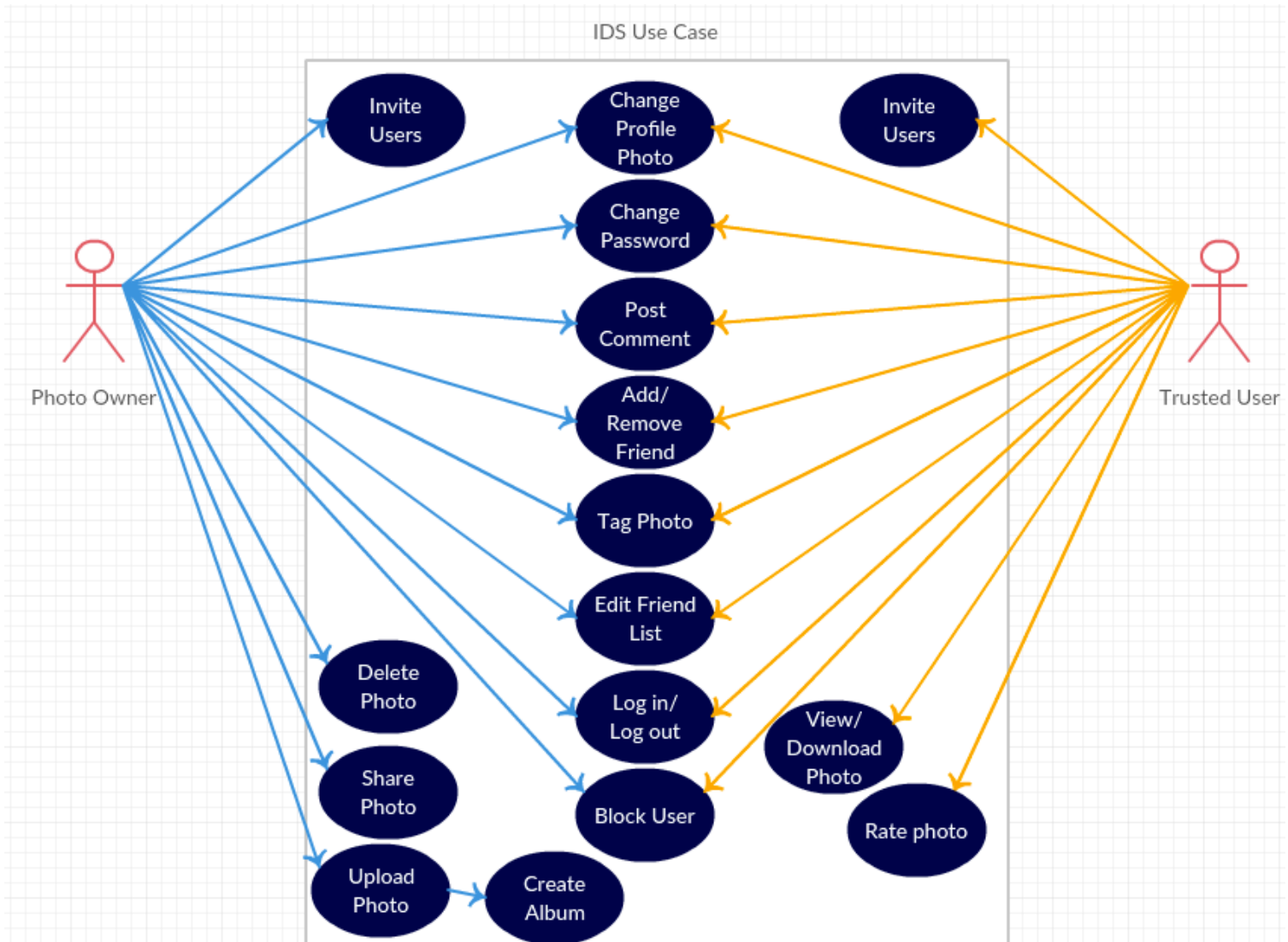


## 3 Requirements Specification

### 3.1 Functional Requirements

1. **Registration (Create Account)** – Users must be able to create an account for this service by registering with a valid email address.
2. **Change Password** – Users will have the option to request a temporary password if they have forgotten theirs, this temporary password will take the first 4 letters of their email and add 5 randomly selected numbers onto that and will be emailed to the user. Users will then also have the option to change their password to something more familiar to them.
3. **Upload Photo** – Users must have the functionality to be able to upload photos into predefined galleries on the website.
4. **Delete Photo** – Users will have the option to delete a photo from their gallery. This function will only belong to the photo owner.
5. **Invite Users** – Users must have the ability to invite other people to join the service via email invitation.
6. **Share Photo** – Users must be able to share photos amongst other users. Photos will be shared over email. This process will be secured by our Intrusion Detection System.
7. **Edit Friends List (Friend/Unfriend User)** – Users must be allowed to modify their trusted users, giving them the power to decide who they can share photos with – sending or receiving.
8. **Block User** – Users must have the functionality to actively block any other member of the web application, disabling them from viewing/sharing any photos.
9. **Comment** – Users will have the ability to leave messages/comments on user profiles and pictures.
10. **Change Profile Picture** – Users will have the ability to change their profile picture at any time from their user profile page.

### 3.1.1 Use Case Diagram



### 3.1.2 Requirement 1: Registration

#### 3.1.2.1 Description & Priority

This requirement describes the actions taken by a user to register as a member on the photo sharing web application. A user must provide a username, valid email address, password, confirm password, gender and country. This requirement is vital to the system as it is the only means of access to the system. It also ensures that only registered users can access the system. Upon registering an automated email will be sent to the user with an activation link. Only activated users will be allowed to access the site, ensuring that valid emails are being used.

### 3.1.2.2 Use Case

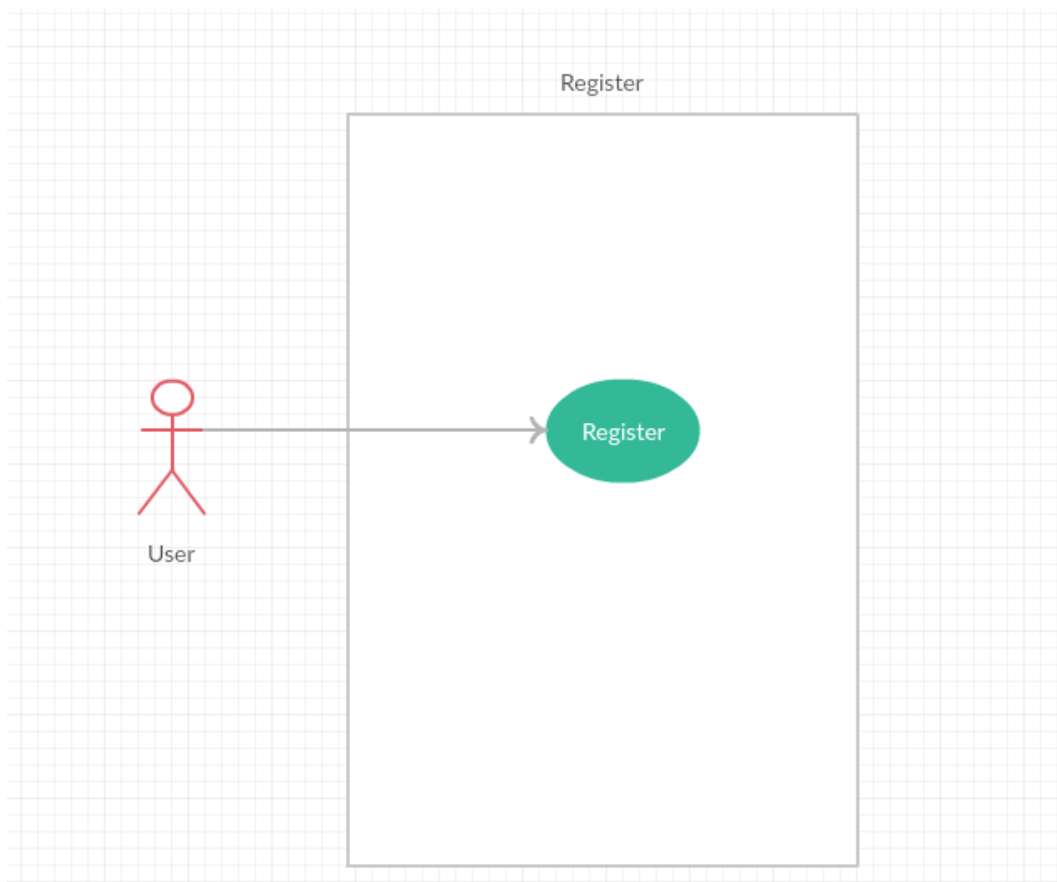
#### Scope

The scope of this use case is to ensure that only registered users can access the system and ensures that all users have a valid email address.

#### Description

This use case describes the actions taken by a user in setting up an account by providing a unique username, password, valid email address, gender and country.

#### Use Case Diagram



#### Flow Description

#### Precondition

The system is in stand-by mode

#### Activation

This use case starts when a user clicks the “Sign Up” button

### **Main flow**

1. The system identifies the clicking of the “Sign Up” button
2. The user enters a username (See A1)
3. The system accepts username
4. The user enters a password (See E1)
5. The system accepts the password
6. The user confirms password
7. The system accepts the password
8. The user enters a valid email address
9. The system accepts email address
10. The user selects gender
11. The system accepts gender
12. The user selects their country
13. The system accepts country
14. User clicks “Create Account”

### **Alternate flow**

A1: Invalid username

1. The system rejects the username as it already exists in the database and requests user to enter a new one
2. The user enters a valid username
3. The use case continues at position 3 of the main flow

### **Exceptional flow**

E1: Invalid password – too short

4. The system rejects the password and requests user to enter a longer one
5. The user enters a valid password
6. The use case continues at position 5 of the main flow

### **Termination**

The system sends an automated email with an activation link to activate users account

### **Post condition**

The system goes into a wait state

### 3.1.3 Requirement 2: Change Password

#### 3.1.3.1 Description & Priority

This requirement describes the actions taken by a user to change their password for logging into the site. Users will have the option to request a temporary password if they have forgotten theirs, this temporary password will take the first 4 letters of their email and add 5 randomly selected numbers onto that and will be emailed to the user. Users will then also have the option to change their password to something more familiar to them.

#### 3.1.3.2 Use Case

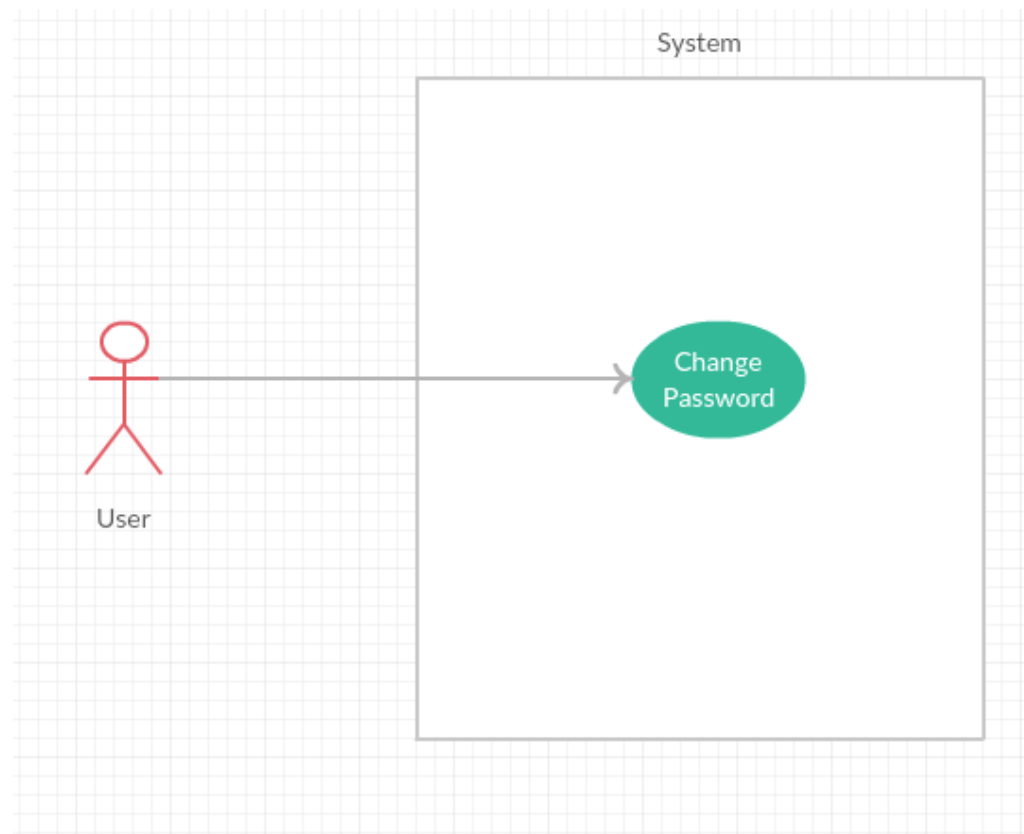
##### Scope

The scope of this use case is to ensure that users have the option of changing their password.

##### Description

This use case describes the actions taken by a user in order to change their password.

##### Use Case Diagram



## **Flow Description**

### **Precondition**

The system is in stand-by mode

### **Activation**

This use case starts when a user clicks the “Change Password” button

### **Main flow**

15. The system identifies the clicking of the “Change Password” button
16. The user enters their current password (See A1)
17. The system accepts password
18. The user enters their new password (See E1)
19. The system accepts the password
20. The user confirms password
21. The system accepts the password
22. User clicks “Make Changes” button

### **Exceptional flow**

E1: Invalid password – too short

7. The system rejects the password and requests user to enter a longer one
8. The user enters a valid password
9. The use case continues at position 5 of the main flow

### **Termination**

The system sends new details to database; new password is encrypted in database.

### **Post condition**

The system goes into a wait state

### 3.1.4 Requirement 3: Upload Photo

#### 3.1.4.1 Description & Priority

This requirement describes the actions taken by a photo owner to upload a photo to the web application and by doing so, enters it into one of five galleries that the user can choose from. From the “Photos” homepage, a member can access the “Upload Photo Now” button. This requirement holds great importance as it is the very basis for this web application. Without the functionality to upload a photo, our photo sharing website is rendered useless.

#### 3.1.4.2 Use Case

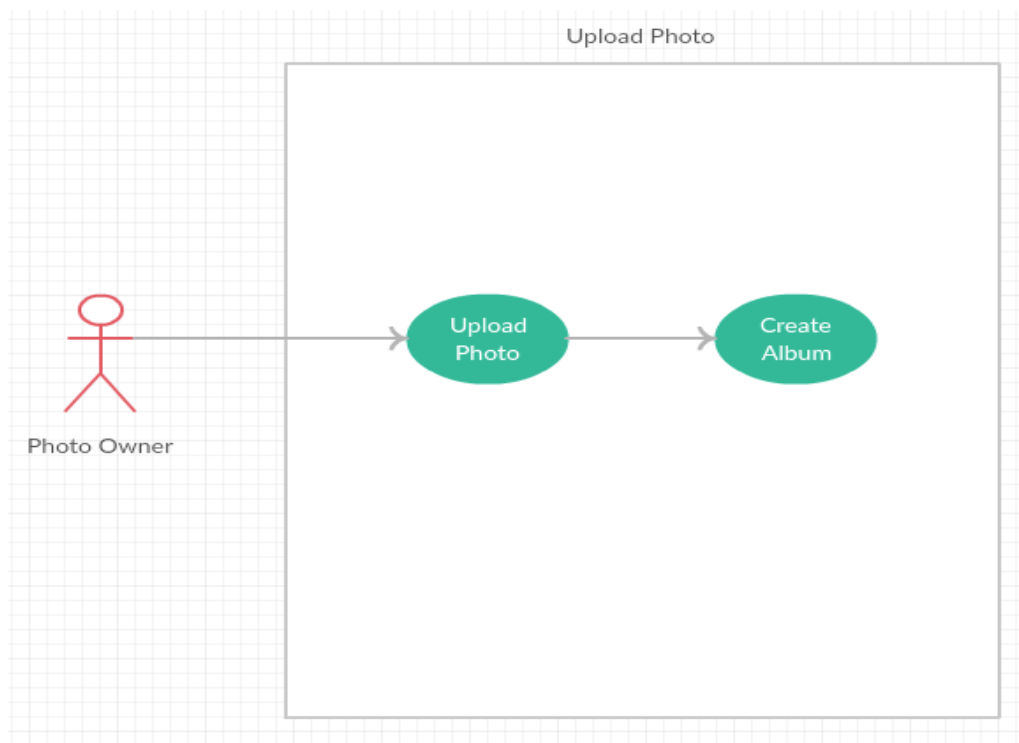
##### Scope

The scope of this use case is to allow a member to upload a photo to this web application into one of 5 galleries.

##### Description

This use case describes the actions taken by a registered member of navigating the “Photos” home page and selecting the “Upload Photo Now” button and following the necessary instructions to successfully upload an image to the website.

##### Use Case Diagram



### **Flow Description**

#### **Precondition**

The system is in stand-by mode

#### **Activation**

This use case starts when a registered member chooses what gallery they wish their photo to be uploaded to.

#### **Main flow**

1. The system identifies the clicking of the “Choose Gallery” dropdown button
2. The member selects an image to upload (See A1)
3. The system accepts the image
4. The member clicks the “Upload Photo Now” button]
5. Photo gets uploaded to selected gallery and is stored in the database

#### **Alternate flow**

A1: Format Rejected

10. The system rejects the image on the basis that it is not the correct format
11. The member converts the image to the correct format
12. The use case continues at position 3 of the main flow

#### **Termination**

The photo is added into selected gallery

#### **Post condition**

The system goes into a wait state



### 3.1.5 Requirement 4: Delete Photo

#### 3.1.5.1 Description & Priority

This requirement describes the actions taken by a photo owner to delete a photo off the web application. When viewing photos in a gallery, a member can access the “Delete this photo” button. This requirement holds great importance as it is important for users to be able to delete any unwanted photos from their page.

#### 3.1.5.2 Use Case

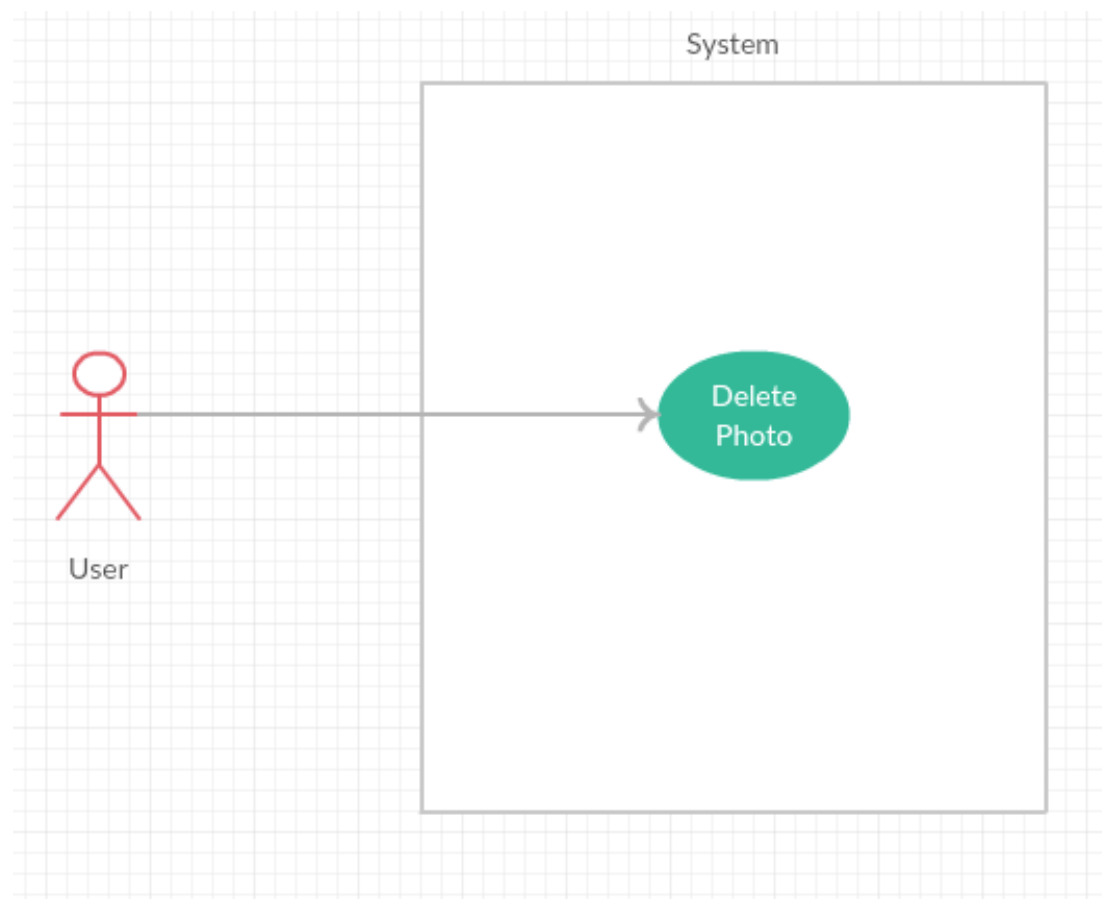
##### Scope

The scope of this use case is to allow a member to delete a photo from this web application from one of the galleries.

##### Description

This use case describes the actions taken by a photo owner of navigating the photos in a gallery and selecting the “Delete this photo” button.

##### Use Case Diagram



### **Flow Description**

#### **Precondition**

The system is in stand-by mode

#### **Activation**

This use case starts when a photo owner selects a photo they wish delete.

#### **Main flow**

1. The system identifies photo owner viewing owned photo
2. The user selects "Delete this photo" button (See A1)
3. System displays pop up message asking to confirm deletion
4. The system deletes the photo
5. The system removes photo from gallery

#### **Alternate flow**

A1: Not owned photo

13. The system does not display the delete button if user does not own image

#### **Termination**

The photo is added into selected gallery

#### **Post condition**

The system goes into a wait state

### 3.1.6 Requirement 5: Invite Users

#### 3.1.6.1 Description & Priority

This requirement describes the actions taken by a member to send an invitation to join the service via email. When a user accepts this invitation, they will be brought to the website and be placed in the sender's trusted user friend list once they have signed up. This will allow the two users to connect and share photos with each other. This requirement is vital as it is a means of populating the website and also allows users to view each others photos.

#### 3.1.6.2 Use Case

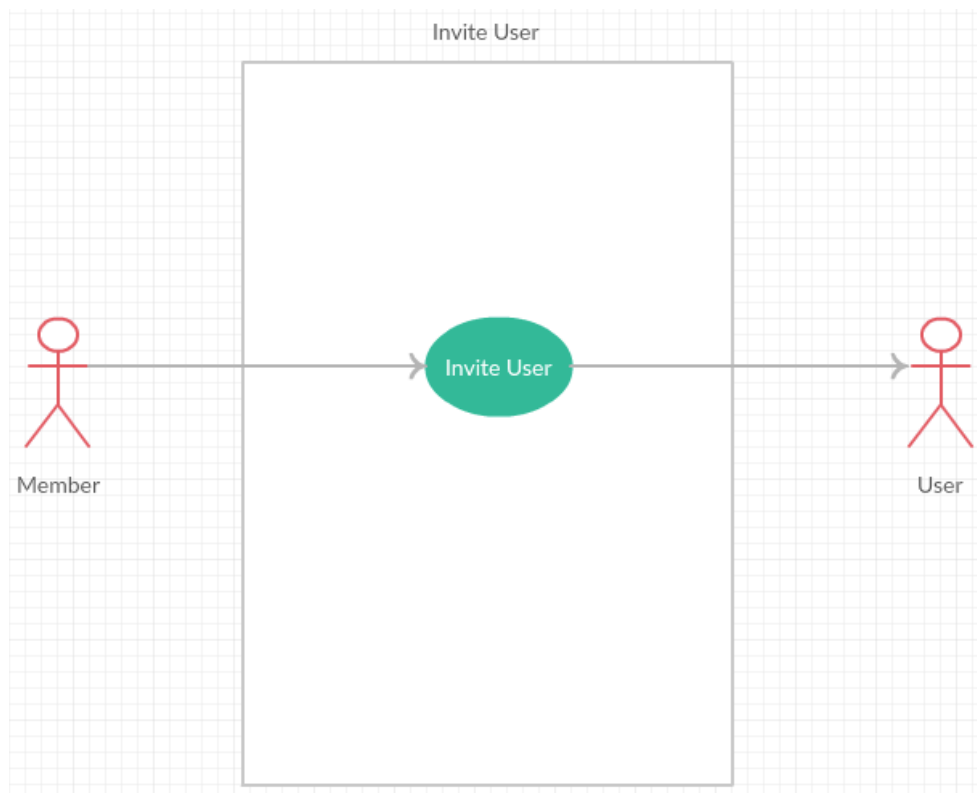
##### Scope

The scope of this use case is to allow a member to send an invitation to another user via automated email

##### Description

This use case describes the actions taken by a member to send an invitation over email

##### Use Case Diagram



### **Flow Description**

#### **Precondition**

The system is in stand-by mode

#### **Activation**

This use case starts when a member clicks the “Invite User” Button

#### **Main flow**

1. The system identifies the clicking of the “Invite” Button
2. The member enters an email address (See A1)
3. The system accepts the email address
4. The member clicks “Send Invitation”
5. The system sends email
6. User receives email
7. User accepts invitation (See E1)
8. User becomes friend on the web application and is added to member’s friend list

#### **Alternate flow**

A1: Invalid Email Address

9. The system rejects the email address as it already exists in the database
10. The member enters a valid email address
11. The use case continues at position 3 of the main flow

#### **Exceptional flow**

E1: User Declines Invite

12. The user declines invitation

#### **Termination**

The system places new friend in senders trusted friend list

#### **Post condition**

The system goes into a wait state

### 3.1.7 Requirement 6: Share Photo

#### 3.1.7.1 Description & Priority

This requirement describes the actions taken by a photo owner to share a photo with a another user. Photos will be sent over email and will be protected by our Intrusion Detection System. This requirement is vital as it is the very basis of the photo sharing website.

#### 3.1.7.2 Use Case

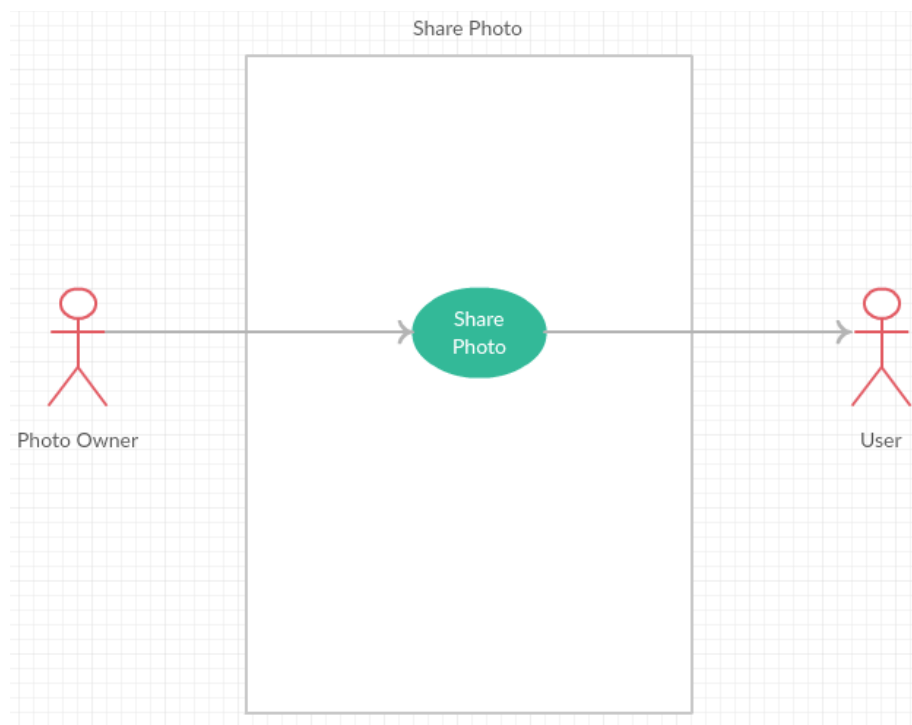
##### Scope

The scope of this use case is to allow a photo owner to share a photo with another user

##### Description

This use case describes the actions taken by a photo owner to share a photo with another user

##### Use Case Diagram



##### Flow Description

##### Precondition

The system is in stand-by mode

##### Activation

This use case starts when a photo owner shares a photo with another user

**Main flow**

1. The system identifies the clicking of the “Share Photo” button
2. The Photo Owner selects a user to share photo with
3. The system accepts request
4. The Photo Owner sends photo (See A1)
5. Users add each other to trusted user list (See E1)
6. Photo Owner sends photo to trusted user

**Alternate flow**

A1: User is not a trusted user

7. The user receives the photo but is marked as malicious email
8. The IDS places email in junk folder
9. The use case continues at position 5 of the main flow

**Exceptional flow**

E1: Edit Trusted User/Friend list

10. Photo Owner adds user to list
11. The system accepts request
12. The use case continues at position 6 of the main flow

**Termination**

The system executes requests

**Post condition**

The system goes into a wait state

### **3.1.8 Requirement 7: Edit Friend List (Friend/Unfriend User)**

#### **3.1.8.1 Description & Priority**

This requirement describes the actions taken by a member to manually modify their trusted users on the system which is displayed as a “Friends List” by adding someone to the list by “friending” them or “Unfriending” them. This requirement proves very important as this gives users the power to choose who they can

share their photos with. Once a user has been removed from the list, any photos received from them will be marked as malicious email by our IDS.

### 3.1.8.2 Use Case

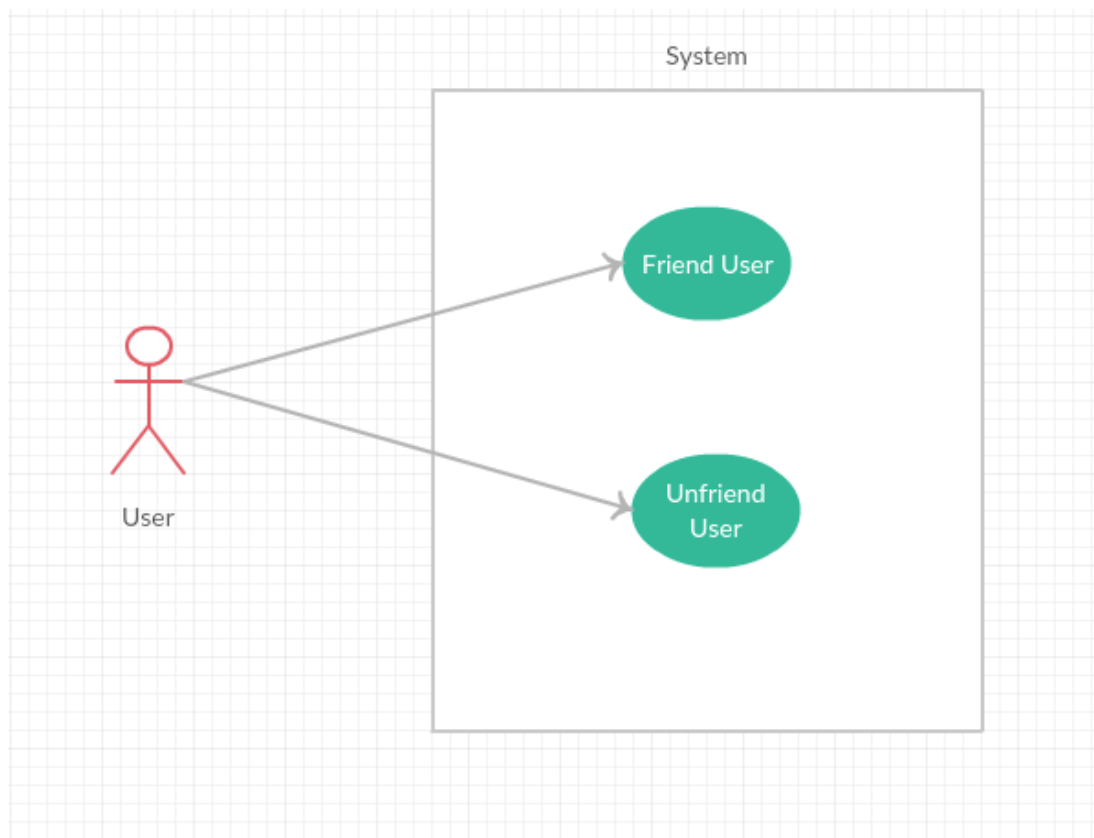
#### Scope

The scope of this use case is to allow a user to modify their friends list

#### Description

This use case describes the actions taken by a user in manually modifying their friend's list

#### Use Case Diagram



#### Flow Description

#### Precondition

The system is in stand-by mode

#### Activation

This use case starts when a user clicks "Request as Friend" button on a user's profile

### **Main flow**

1. The system identifies the clicking of the “Request as Friend” Button
2. The system displays a message confirming the action has been completed
3. System will send a notification to other user about the request
4. Other user accepts request
5. Both users are now friends and will appear in each others friend list
6. User can then select “Unfriend User”
7. This user will no longer be displayed in their friend list
8. The system now displays a “Friend User” button if the user ever wishes to add this user again in the future

### **Termination**

The system returns to the user’s profile

### **Post condition**

The system goes into a wait state

## **3.1.9 Requirement 8: Block User**

### **3.1.9.1 Description & Priority**

Users must have the functionality to actively block any other member of the web application, disabling them from viewing/sharing any photos.

### **3.1.9.2 Use Case**

#### **Scope**

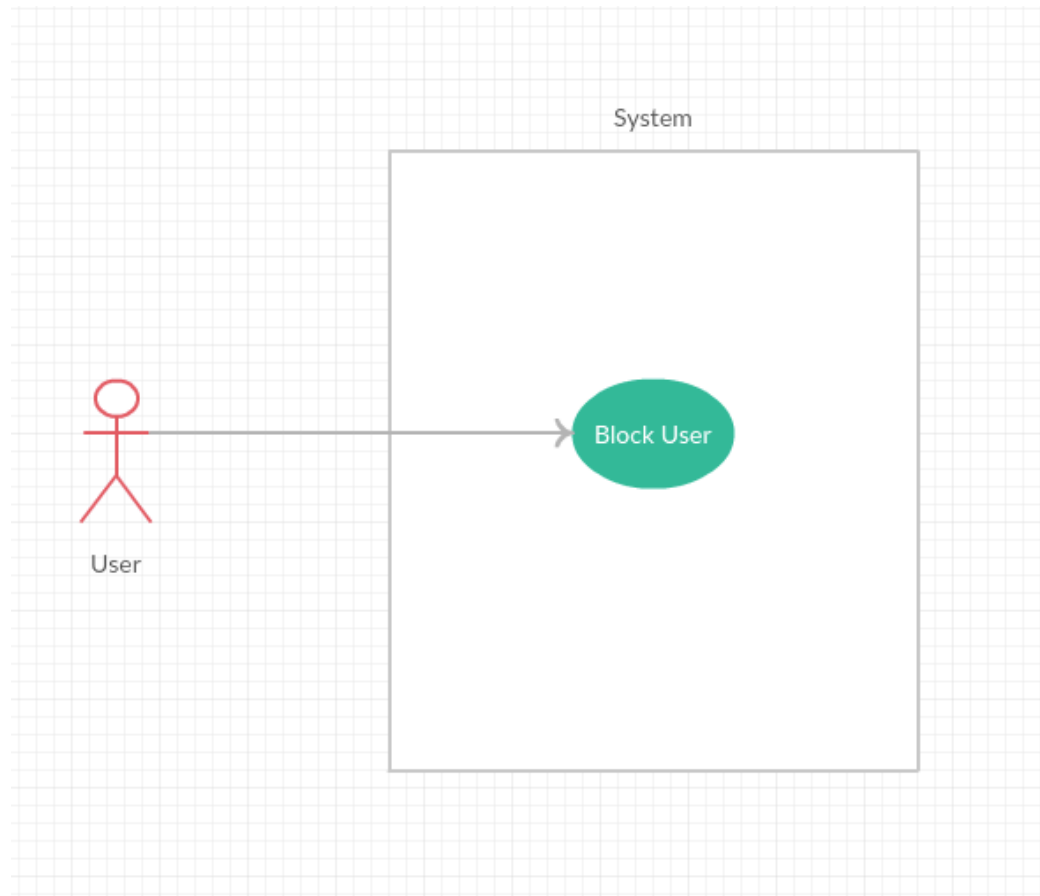
The scope of this use case is to allow a user to block another user

#### **Description**

This use case describes the actions taken by a user to manually block another user



## Use Case Diagram



### Flow Description

#### Precondition

The system is in stand-by mode

#### Activation

This use case starts when a user clicks “Block User” button on a user’s profile

#### Main flow

1. The system identifies the clicking of the “Block User” Button
2. The system displays a message asking user to confirm the block
3. User clicks “ok”
4. System blocks user

**Termination**

The system returns to the user's profile

**Post condition**

The system goes into a wait state

**3.1.10 Requirement 9: Comment**

**3.1.10.1 Description & Priority**

Users must have the functionality to leave comments on user profiles and photos.

**3.1.10.2 Use Case**

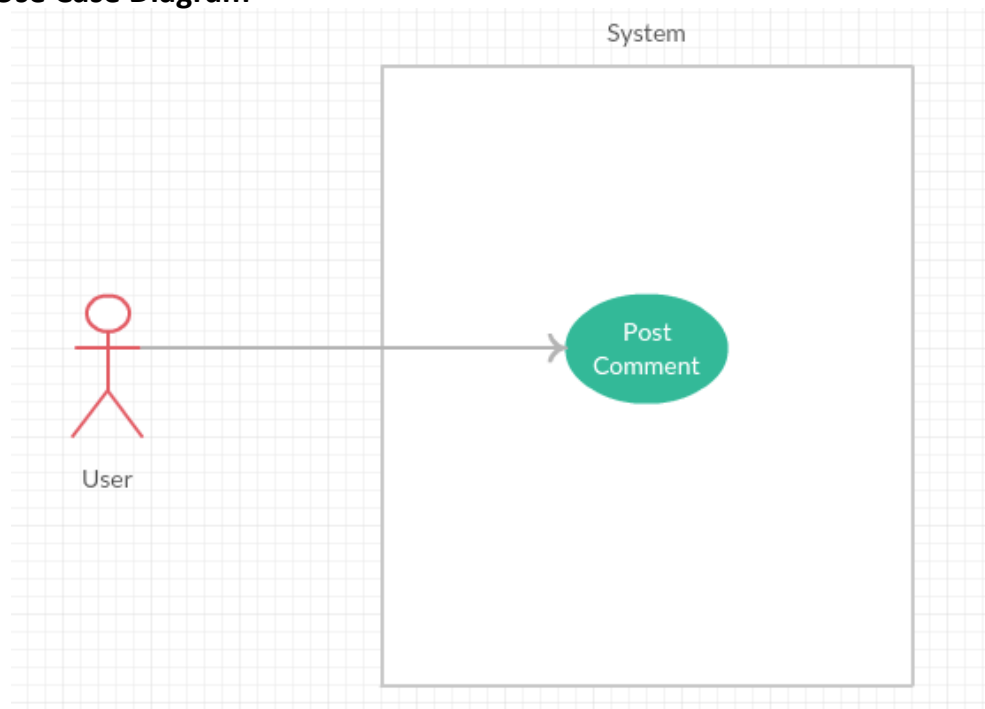
**Scope**

The scope of this use case is to allow a user to post a comment on a user's profile/photo

**Description**

This use case describes the actions taken by a user to write and post a comment on a user's profile or photo

**Use Case Diagram**



### **Flow Description**

#### **Precondition**

The system is in stand-by mode

#### **Activation**

This use case starts when a user clicks on the comment box to begin writing

#### **Main flow**

1. The system identifies the clicking of the comment text box
2. Typing cursor displays in comment box
3. User writes message
4. User clicks “post comment” button
5. System posts comment to user’s profile/photo
6. User receives notification of the event in notifications page

#### **Termination**

The system returns to the user’s profile

#### **Post condition**

The system goes into a wait state

### **3.1.11 Requirement 10: Change Profile Picture**

#### **3.1.11.1 Description & Priority**

This requirement describes the actions taken by a user to change/upload a new profile picture.

#### **3.1.11.2 Use Case**

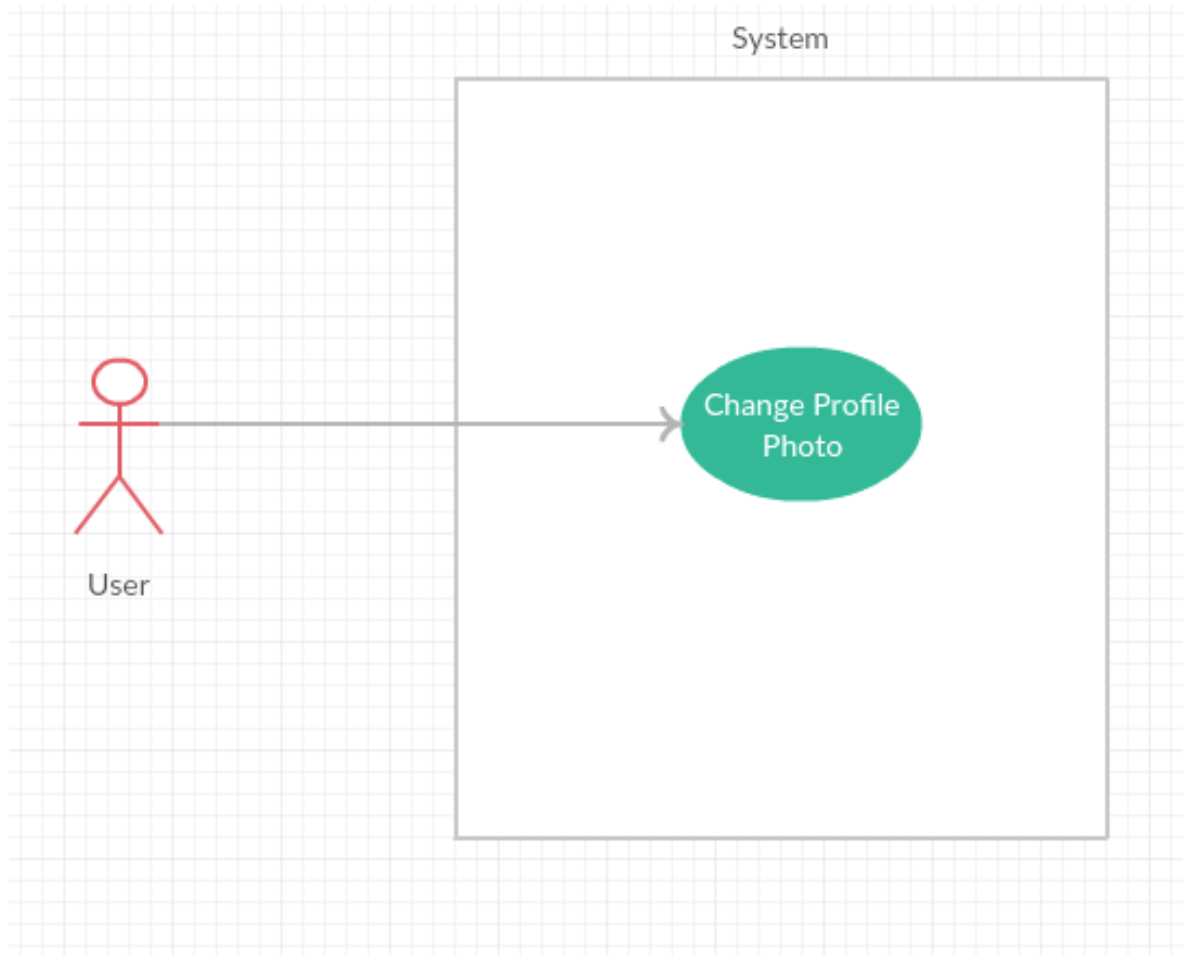
##### **Scope**

The scope of this use case is to allow a photo owner to change their profile photo

##### **Description**

This use case describes the actions taken by a photo owner to change the photo used as their profile picture

## Use Case Diagram



### Flow Description

#### Precondition

The system is in stand-by mode

#### Activation

This use case starts when a photo owner clicks “toggle avatar form” on their current profile photo.

#### Main flow

1. The system identifies the clicking of the “toggle avatar form” button
2. The Photo Owner then selects “Choose file”
3. The user then selects the photo he/she desires
4. The user then selects “upload photo”
5. The system accepts photo
6. The system changes profile picture

### **Termination**

The system executes requests

### **Post condition**

The system goes into a wait state

## **3.2 Non-Functional Requirements**

### **3.2.1 Performance/Response time requirement**

This IDS photo gallery that we are developing will be used as the main performance system which interacts with members and administrators. Therefore, it is expected that the system would perform all the requirements that are specified.

- The system should be fast and accurate
- System will handle expected and non-expected errors in a manner that will prevent information loss and long downtime period.
- System should have error testing to identify invalid username or password and email
- System should be able to handle large amounts of data
- System should accommodate high number of photos and users without any fault

### **3.2.2 Safety Requirement**

It has to be taken into account that at any given time the database may crash due to a virus or operating system failure. Therefore, it is required to take the database backup to prevent losing it. Appropriate UPS facility should be installed in case of power supply failure.

### **3.2.3 Security Requirement**

- Appropriate user authentication should be provided

- Security questions will need to be setup when a user registers as a user
- Security questions will need to be answered when a user forgets their password
- The system will have a number of unique users and each user has access constraints
- The system will use a secured database, photos and passwords will be encrypted in the database
- User details such as username and passwords will be protected with Md5 hash
- Function 'evalLoggedUser' checks a user's session data to see if it matches something on file, system double checks to make sure user is who they say they are. Protects against Cross Site Scripting attacks.
- The system is designed to prevent any user from entering malicious javascript code to conduct Cross Site scripting or SQL Injection.
- Users can read info but cannot edit anything except for their personal details and photos
- Users will have the functionality of blocking a user, disabling them from viewing any of their photos
- When users share a photo, they will be protected with an encrypted form (short 4 digit password)
- A .htaccess file will be established to prevent directory file listing as well as specify a more user friendly 404 error page through htaccess.
- Separate account types for admin and members so that no member can access the database and only admin can update the database

### **3.2.4 Requirement Attributes**

- The project should be open source
- Admins can create changes to the system, but members or other users cannot make any changes.
- The quality of the website and database is maintained in such a way that it can be very user friendly to all users
- The user should be able to download and install the system very easily

### **3.2.5 Business Rules**

A business rule is anything that captures and implements business policies and practices. A rule can enforce business policy, make a

decision, or infer new data from existing data. This includes the rules and regulations that the System users should abide by. This includes the cost of the project and the discount offers provided. The users should avoid illegal rules and protocols. Neither admin nor member should cross the rules and regulations.

### **3.2.6 User requirement**

There are two main types of users of this system. The member, which is a registered user that has logged in to the system and the admin. The members are assumed to have basic knowledge of the computers and internet browsing. The admins of the system are expected to have more knowledge of the internals of the system and should be able to rectify the small problems that may arise due to disk crashes, power failures and other catastrophes that may occur to maintain the system's well being. The UI, user manual, online help and installation guide must be sufficient enough to educate the users on how to use the system without any problems. Admin can provide help to the users in the following areas:

- Backup and Recovery
- Forgot Password
- Log in issues
- Reporting Photos
- Modify profile/photos
- Maintaining files

### **3.2.7 Maintainability requirement**

In order to maintain our database, we will continue to delete unwanted or unnecessary data as they can become very large as we continuously add data. We will also be backing up our databases and sampling database scripts.

### **3.2.8 Portability requirement**

This service will be available on laptops, tablets and mobile devices.

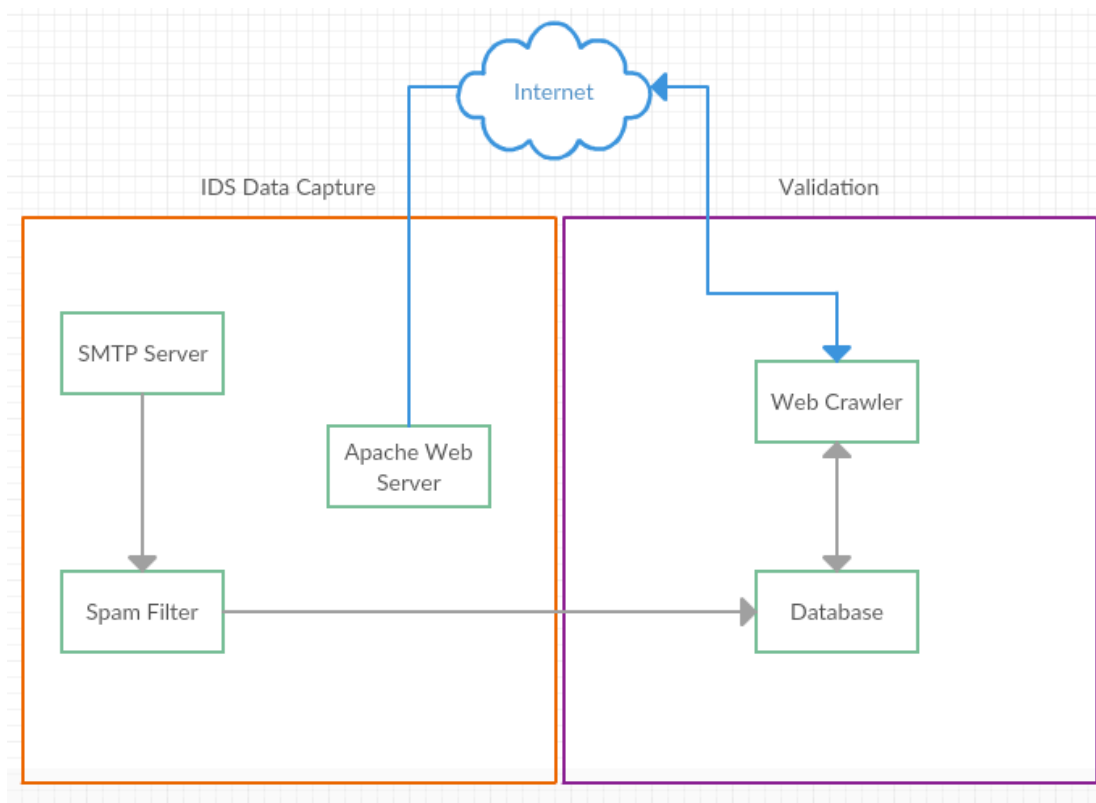
### **3.2.9 Extendibility requirement**

We will extend this system to incorporate Facebook images.

### 3.3 Design and Architecture

This system will consist of three main components: a Spam filter to detect unsolicited emails, a Simple Mail Transfer Protocol (SMTP) server which is the de facto standard for sending email over the internet, and Apache web server which is an open source web server. The IDS scanner will scan all received e-mails and will mark some of them as potential phishing messages or an email that is received from an untrusted user and then enters them into a database. Emails are classified as malicious if they contain embedded HTML or if they are received from an untrusted user. MySQL will be used to store information regarding the detected malicious emails, where data entered into the database consists of the content of the email and time of receipt. A background process will run to scan all URL's listed on marked emails, strips HTML tags and determines whether or not they have been earlier observed.

Prototype Architecture Diagram:

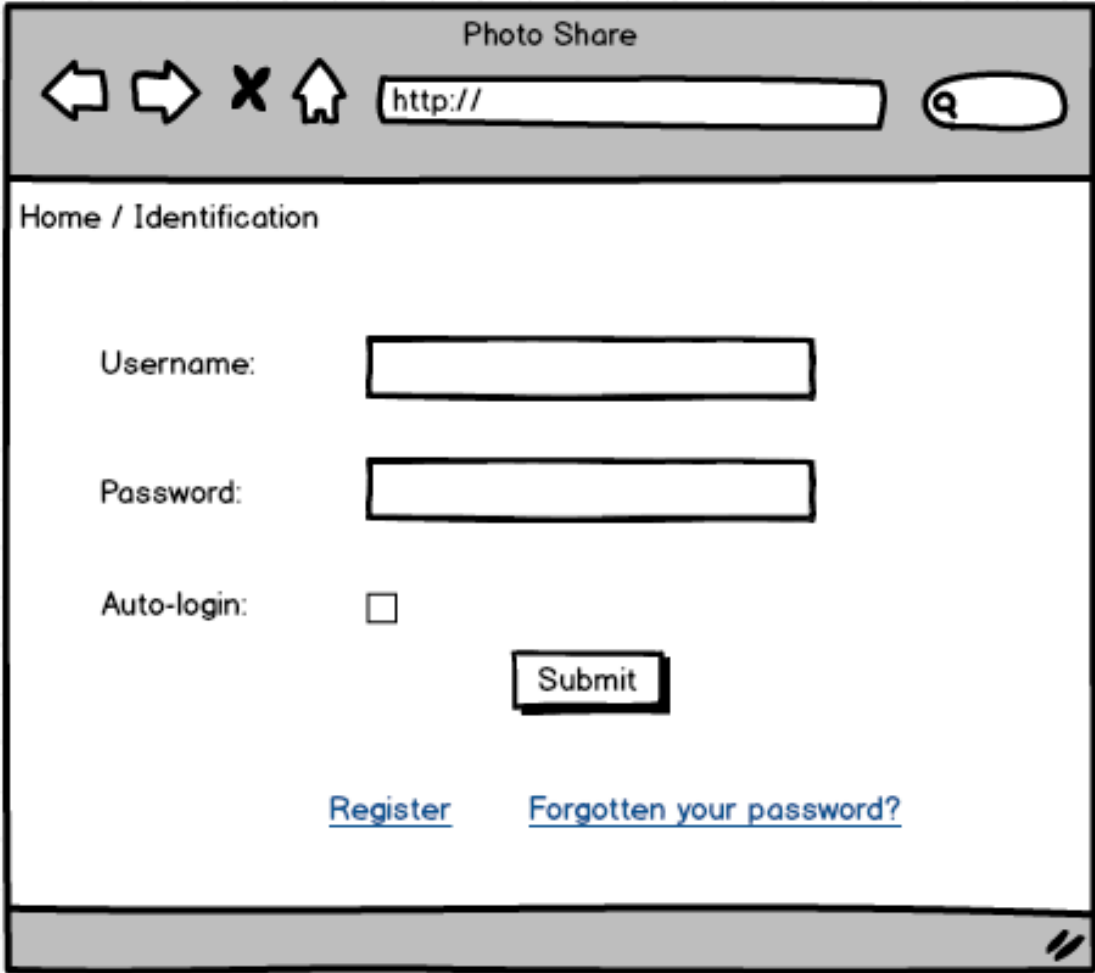




### 3.4 Graphical User Interface (GUI)

For both user and administrator, this software provides good graphical interface. Both users can operate the system performing different tasks such as login, register, upload photo, invite user, share photo and more.

**Log in:** Below is the login page for the photo gallery; users are required to provide a username and password to successfully login. If a user forgets their password they can click the “Forgot your password?” link, which will send necessary instructions via email to recover the password. There is also an auto login feature, which exists for the convenience of frequent members. If a user is a first time visitor they can click the “Register” link, which will bring them to the Register page.

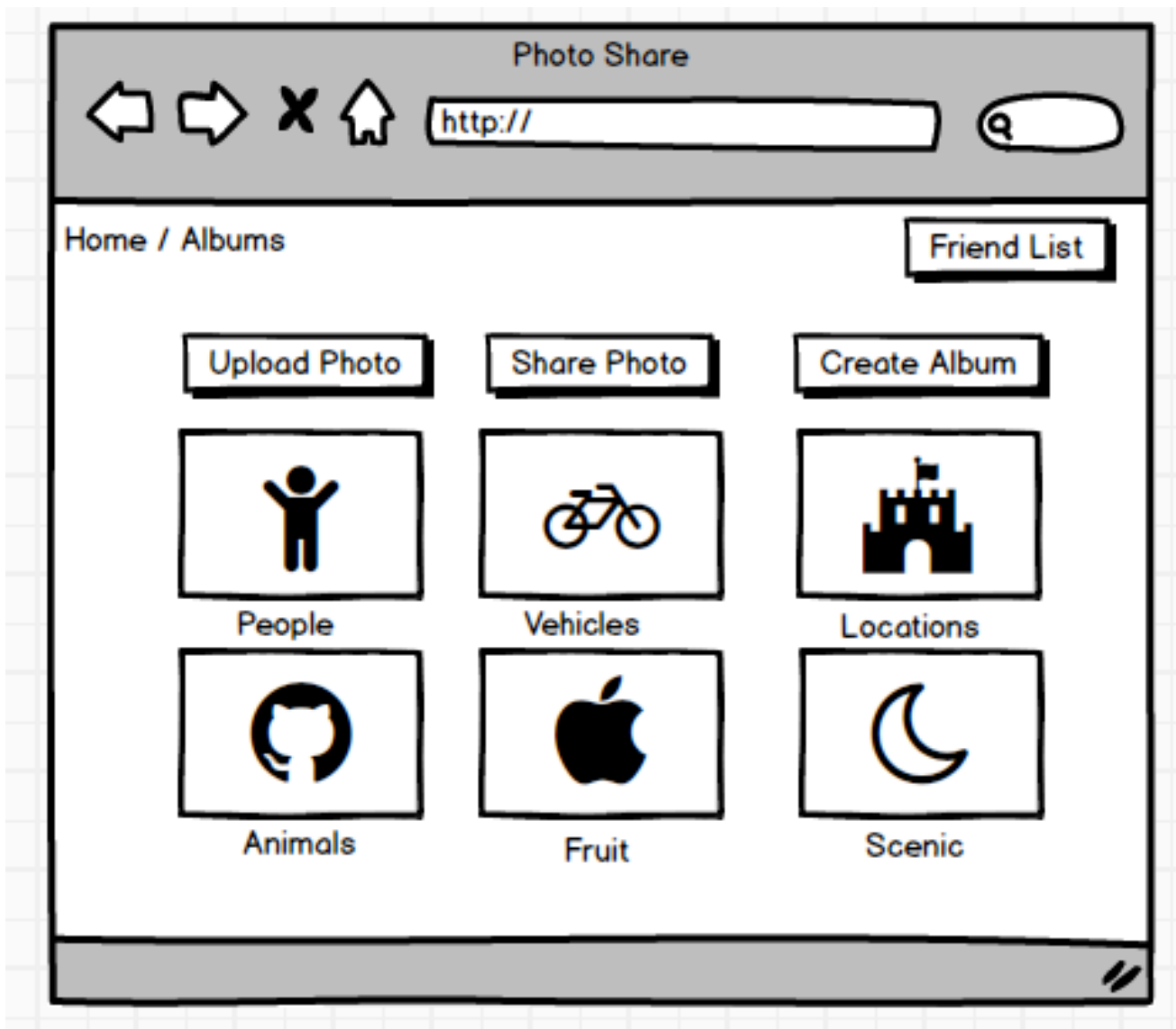


The image shows a hand-drawn sketch of a web browser window titled "Photo Share". The browser's address bar contains "http://". The page content includes a breadcrumb "Home / Identification", a "Username:" label with an input field, a "Password:" label with an input field, and an "Auto-login:" label with an unchecked checkbox. A "Submit" button is centered below the password field. At the bottom, there are two blue underlined links: "Register" and "Forgotten your password?".

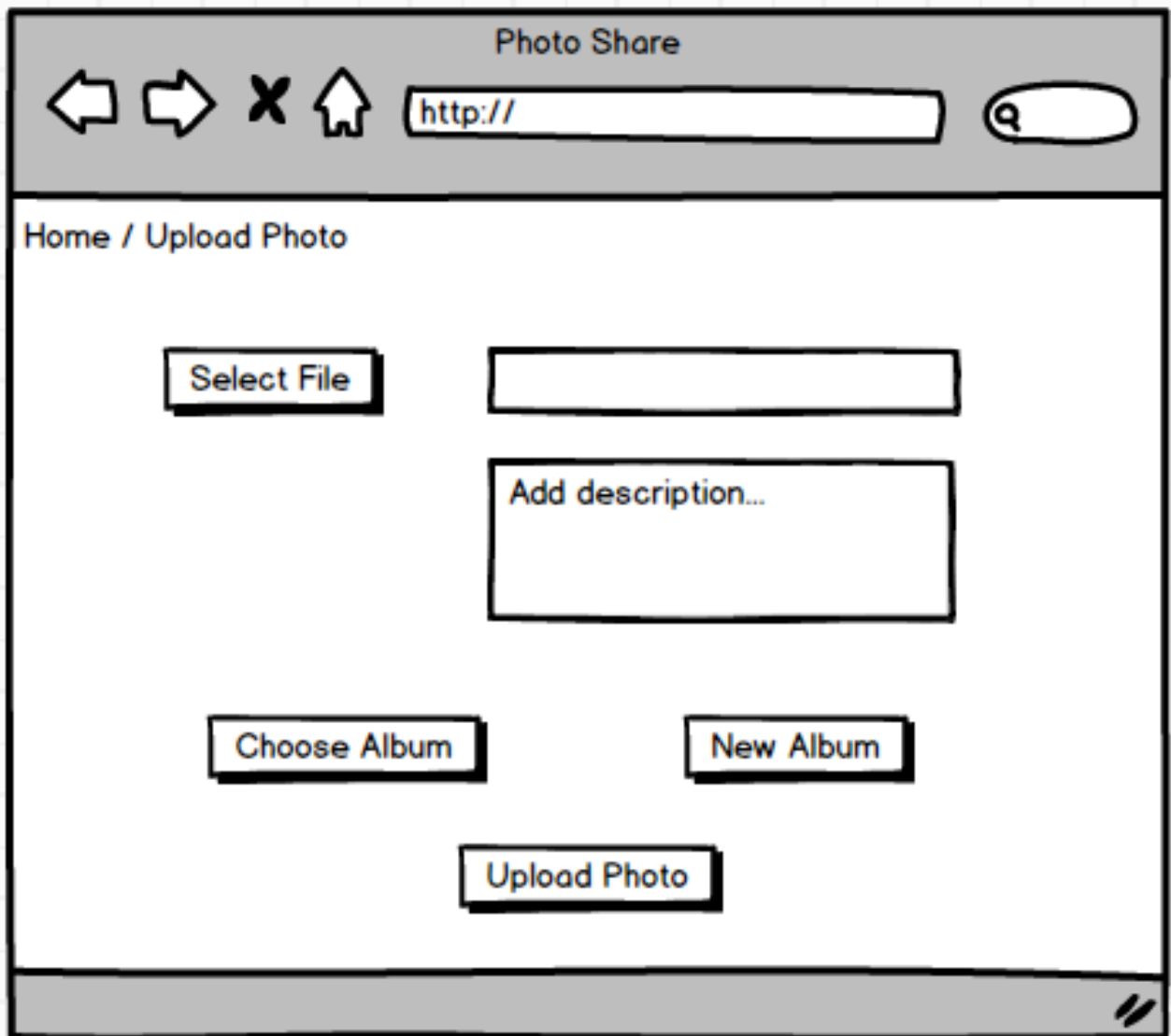
**Register:** Below we can see all credentials that are necessary for a registered member. A member will need a username, password and valid email address. When a user clicks the “Register” button, a confirmation email will be sent to the user’s inbox. Once confirmed, the user is now a member of the photo gallery website.

The image shows a hand-drawn wireframe of a web browser window. The browser's title bar is labeled "Photo Share". The address bar contains "http://". The page content includes a breadcrumb "Home / Registration", four input fields for "Username:", "Password:", "Confirm Password:", and "Email Address:", and two buttons labeled "Register" and "Reset".

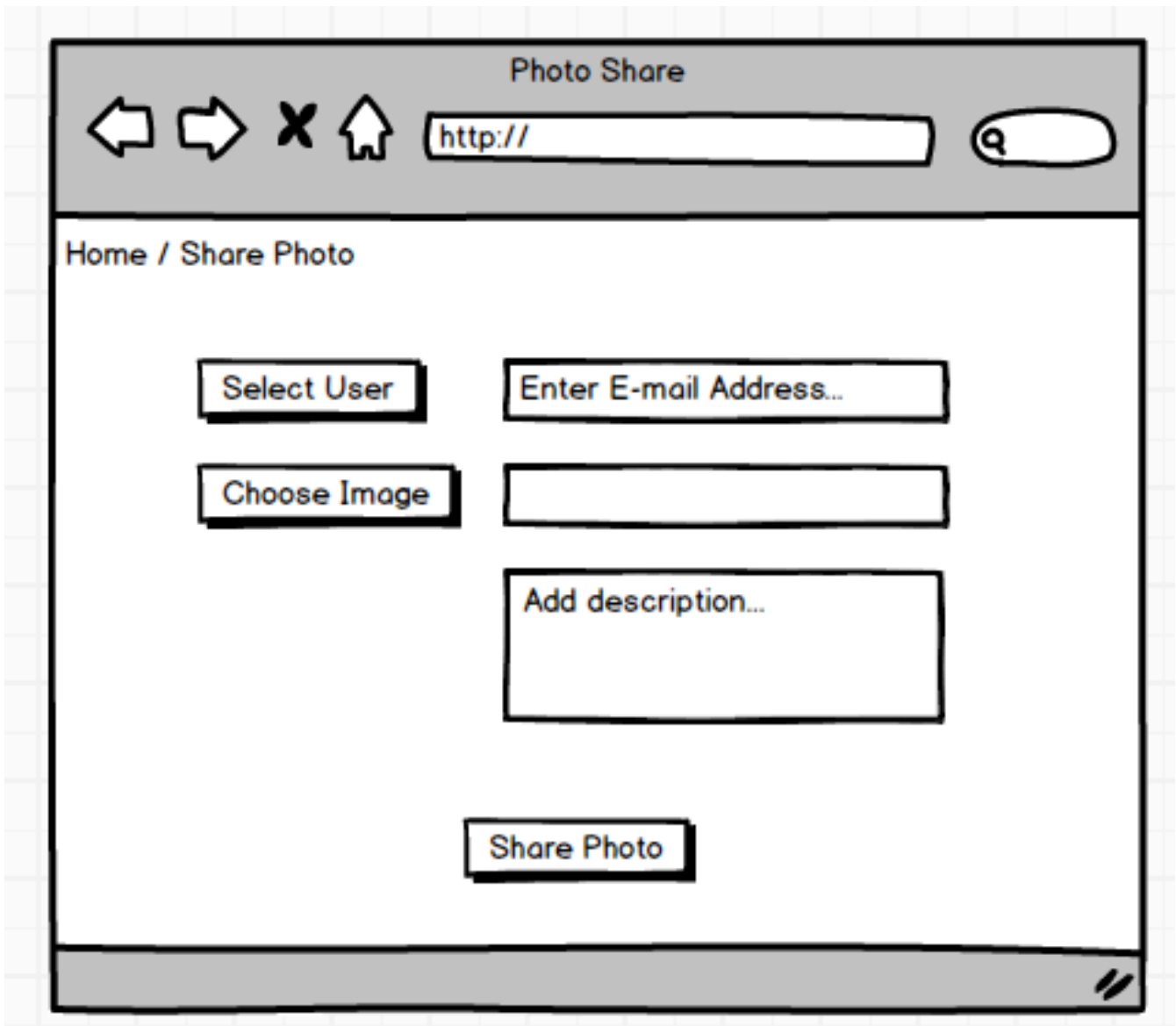
**Albums:** Below we can see the home page, which displays different categories of photos. Some examples below are animals, people and landscapes. When a user chooses a category to browse, all photos in that category will be viewable. From here, a user can also access pages related to uploading photos, sharing photos and friend list, which we will touch on later.



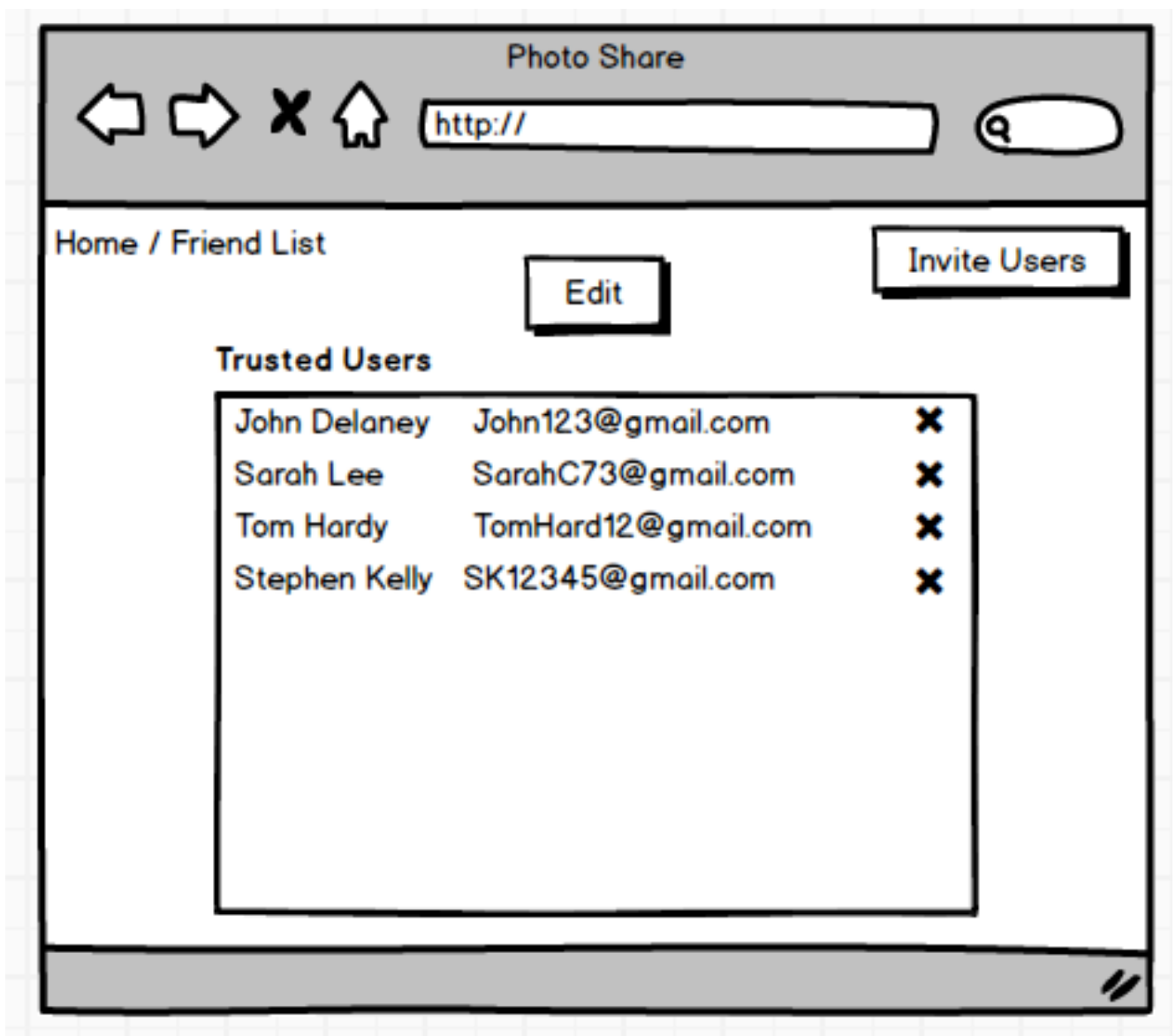
**Upload Photo:** Below we can see how a user can upload a photo to the web application. It is a very simple process. All a user has to do is select an image that is stored locally on their computer or external hard drive, add a description (optional) and select which album they want to upload it to. If this is the first time uploading a photo, or they simply want to create a new album, they can do so by clicking one of the two buttons provided on this page. Once all details have been entered, the user hits “Upload Photo” and the process is complete. The user can then go on to share this photo with trusted users, which we will look at next.



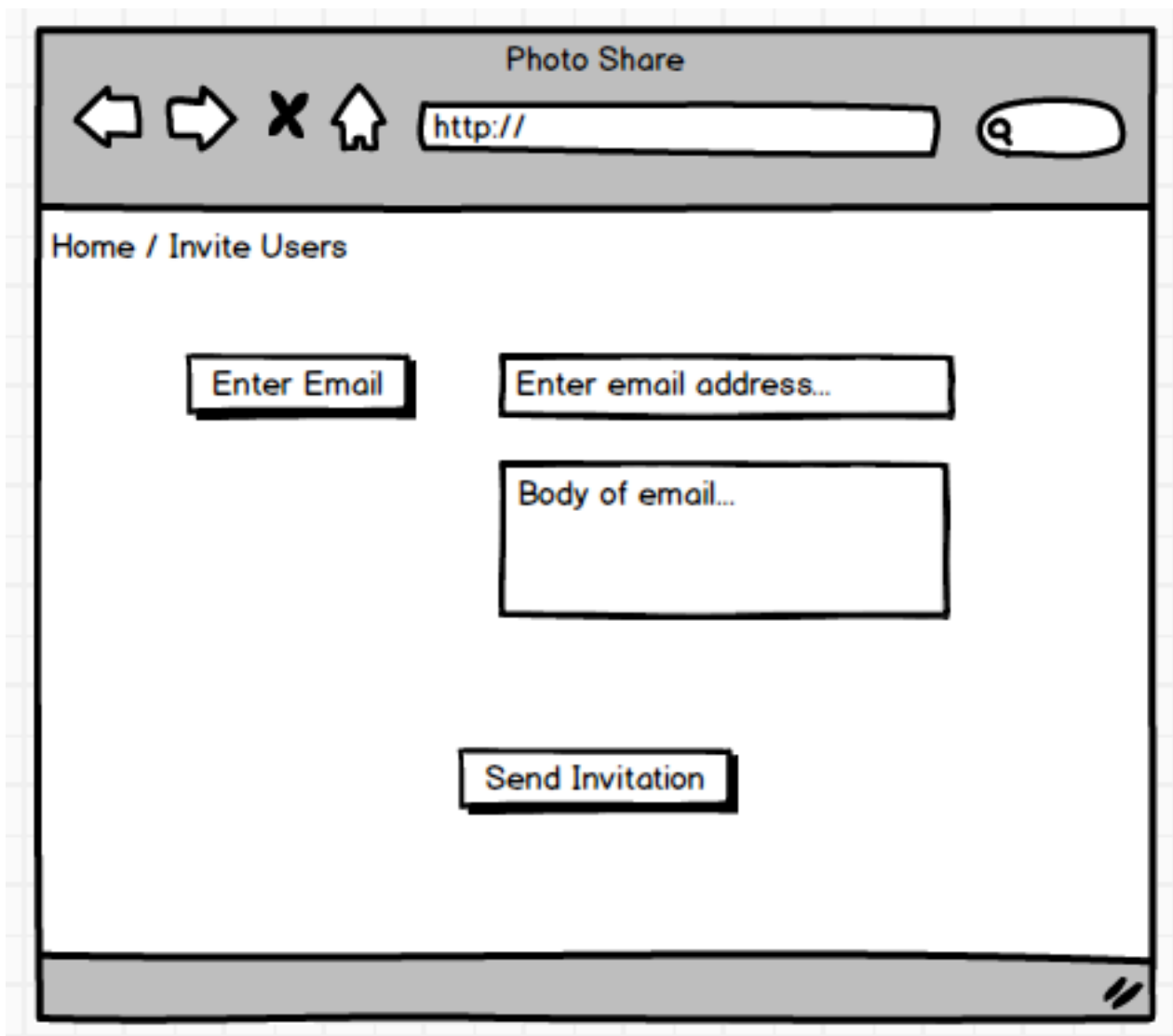
**Share Photo:** Below we can see how a photo owner can share an image with any user with a valid email address. All that is necessary to share a photo is to enter the email of the receiver, select the image you want to share, add a description (optional) and hit the “share photo” button. Note that if a user receives a photo from an untrusted user, our IDS will mark that email as malicious and it will be placed in a junk folder.



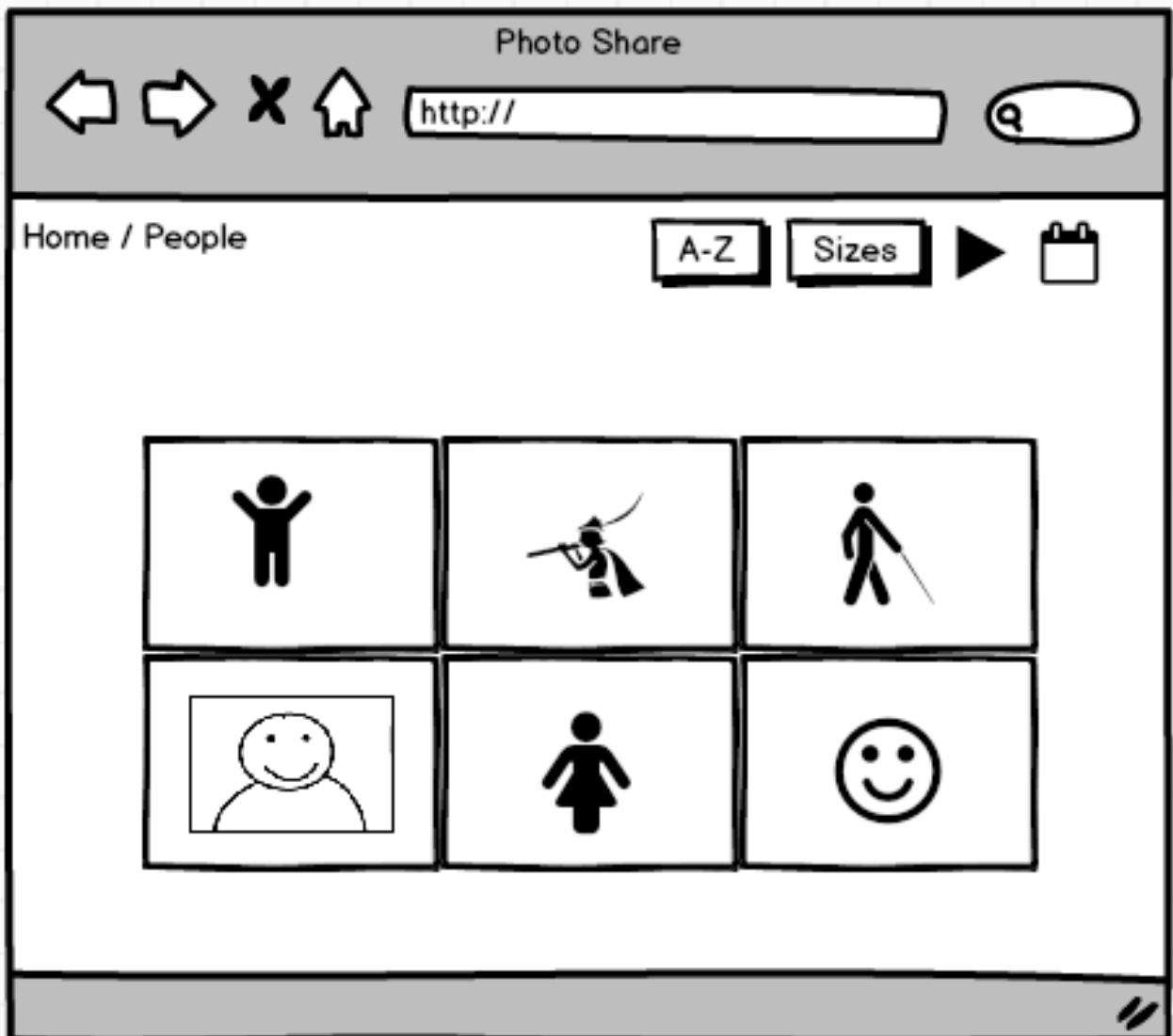
**Edit Friend List (Trusted User):** In the screenshot below we can see how a user can manually edit his/her friend list. It is a very simple process. This page displays the user's friend list, when a user clicks the "Edit" button, x's will appear beside the names of each user in the list. All a user has to do to edit this list is click the x and that user will be removed as a trusted user. The system will ask for confirmation upon clicking an x to ensure it was not clicked by mistake. In the top left hand corner we can see the "Invite User" button, which we will now take a look at.



**Invite Users:** In the screenshot below, we can see how a user can go about inviting other users to join the photo sharing web application. The user enters an email address of the person they want to join. A link will be provided in the email to find the web application but a user may also add a body to the email if they so wish. By clicking “Send Invitation”, an email will be sent housing all the information the receiver needs to become a registered member. By following this link, the new user will automatically be assigned as a trusted user to the sender and vice versa.

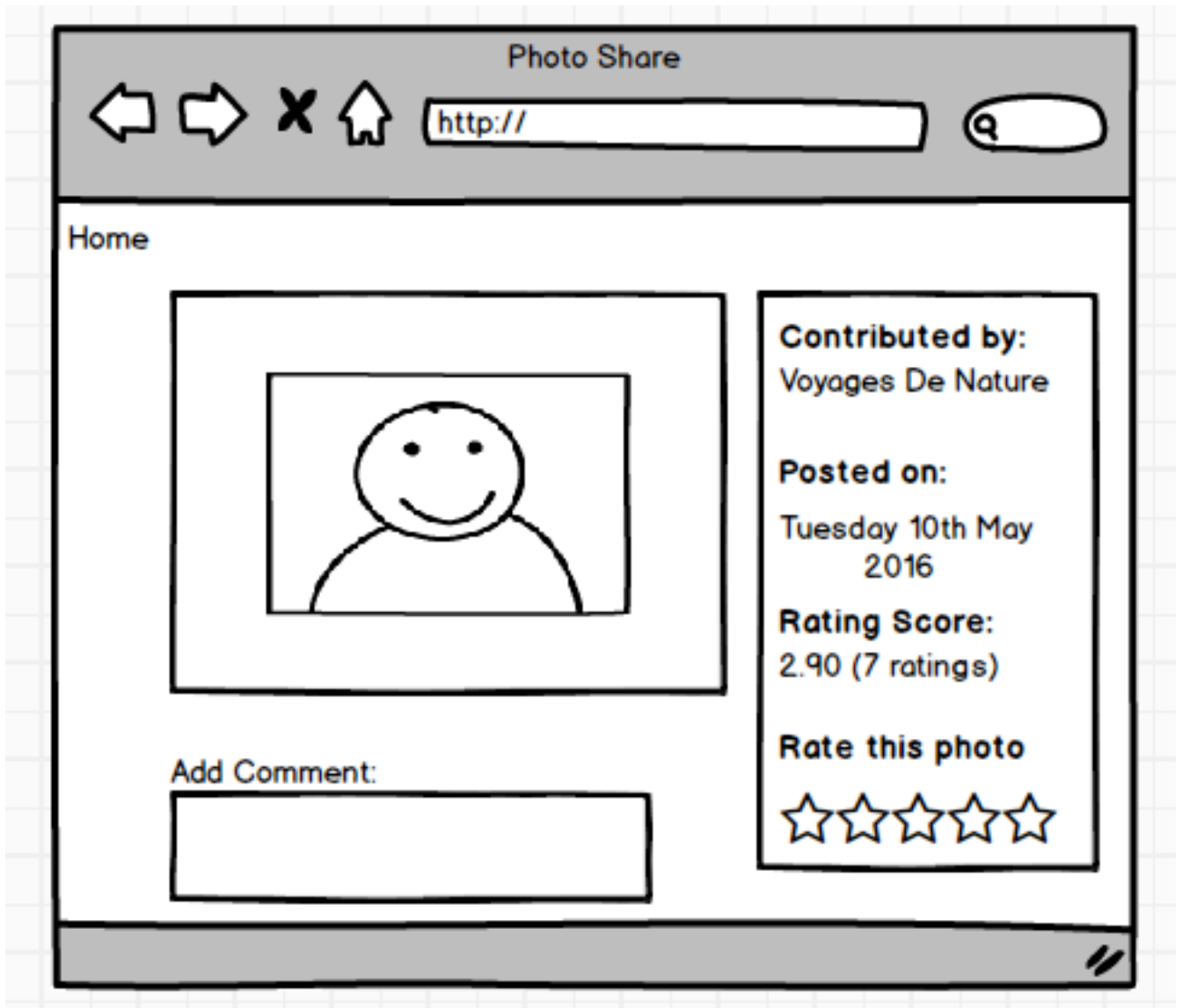


**Album Category – People:** Below we can see the display page when a user chooses an album category. The below category is “People”. There are a number of features available to a user from this page. In the top right hand corner there are four symbols: sort A to Z, photo sizes, slideshow and a calendar to display creation date. To the right of these symbols you will see a dropdown box; this allows users to change the interface theme. An example of a different theme is down below. From here, a user can select an individual photo to view.





**View Photo:** Below we can see the display page for viewing an individual photo. Here a user can view details such author, upload date and rating. Users can also leave a comment or rate the photo using a 5 star system.



## 3.5 Testing

To ensure that our software is functioning correctly, we will conduct the following types of testing:

### White Box Testing

White box testing is a test case design philosophy that uses the control structure as part of component level design to derive test cases.

- **Basis Path Testing**  
Basis path method enables designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set of execution paths.
- **Control Structure Testing**  
It tests control structures of program.

### Black Box Testing

Black box testing enables the software designer to derive sets of input conditions that will fully exercise all functional requirements of a program.

- Graph Based Testing Method
- Equivalence Partitioning
- Boundary Value Analysis
- Orthogonal Array Testing

### Integration Testing

Integration testing is a systematic technique for constructing software architecture while at the same time conducting tests to uncover errors associated with interfacing.

### Regression Testing

Each time a new model is added as part of integration testing, the software changes. At this time we will apply regression testing.

### 3.6 Evaluation

We plan to evaluate this system by collecting a data set that consist of malicious sites and use them with our implementation. This will allow us to calculate the efficiency of detection, which is an extremely important parameter of our Intrusion Detection System. The outcome will be analyzed for accuracy and timeliness.

## 4 Conclusion

Phishing has already become a very popular form of malicious email cyber attack resulting in the theft of personal information. Using existing security technologies to completely prevent security breaches is un-realistic. This is why an IDS is an important component in security as it can offer the advantages of reducing man power that is necessary for monitoring, increasing detection efficiency, providing data that would in other cases not be present, helping the information security community learn about new vulnerabilities and provide legal evidence.

## 5 Future Development

- This system has the potential to evolve into video sharing and implement more social media features, extending its reach to a larger customer base.
- This system could be used in future instances as a core information extraction system.
- Better algorithms could be implemented to increase efficiency and quality of results.

## 6 References

Dofree (2013) *Black Box vs. White Box Testing*. Available at: <https://technologyconversations.com/2013/12/11/black-box-vs-white-box-testing/> [Accessed 5/12/16]

Hasika Pamunuwa (2007) *An Intrusion Detection System for Detecting Phishing Attacks* at: [https://link.springer.com/chapter/10.1007%2F978-3-540-752486\\_13](https://link.springer.com/chapter/10.1007%2F978-3-540-752486_13) [Accessed 2/11/16]

Fette, I (2007) *Learning to Detect Phishing Emails*. Available at: <http://www.cs.cmu.edu/~tomasic/doc/2007/FetteSadehTomasicWWW2007.pdf>

Vanderavero, N (2004) *Scalable Approach to collect malicious Internet Traffic*. Available at: <http://www.info.ucl.ac.be/people/OBO/papers/honeytank.pdf>

## 7 Appendix

### 7.1 Monthly Journals

**- September 2016 -**

#### **Initial Planning**

It is September 3<sup>rd</sup>, and as my final year of college approaches I find myself struggling to formulate an idea for my project that I am truly happy with. As a Cyber Security student, I am extremely interested in developing a security based project as I have a huge interest in this area, but deciding on a certain topic for my project has proved quite difficult – something I did not take into account in my management of time. When I start back in NCI on the 19<sup>th</sup>, I want to be fully prepared with a great security based project idea giving me the push I need to get to work right away! Meanwhile, I have started brushing up on my java skills as I have been out of practice for quite sometime now. During my work placement

with ACIA (Aon Centre for Innovation and Analytics) I was visited by my supervisor Eugene McLaughlin. He gave me great advice, which was to practice programming again as some students return to 4<sup>th</sup> year very ill prepared. I appreciated the advice and acted upon it right away.

It is only a week until we return to class as I enter the week of September 12<sup>th</sup>, 2016. I have to say, I am very much looking forward to starting class again. It feels like such a long time as we have been kept busy with our work placement for the past six-to-seven months. I have been keeping myself busy/preparing for college by looking over old assignments and recoding some old projects. I am actually very surprised at how fast I have been able to pick it back up, its just something that gets burned into your brain I guess – not a bad thing! With this in mind I feel well prepared. Still have not yet decided on a topic for my project but I know it will come to me soon. If I was to make any improvements to my work ethic towards this project, I would have definitely started research on my project a lot earlier.

## Back to Class

It is September 19<sup>th</sup> and all fourth years have returned to NCI. Our very first lecture is of course Software Project and there is great excitement in the room. Eamon gave us an excellent run down of what was expected from us in the coming weeks and I came away from that class with a heightened sense of confidence and support from the college. As Eamon asked for a show of hands around the classroom as to who had already thought of an idea for their project I was a little relieved as it seemed to be a 50/50 difference. As relieved as I was I knew I had to finalize an idea and I had to do it soon. Each student has to prepare a 5 minute pitch presentation to a group of three lecturers who will then either accept your project, deny your project, or accept your project with some nice little improvements and advice. It was our first week back to college but the work had already very much begun.

It wasn't until my second week of college that I finally decided on an idea for my project. I decided to design and implement an Intrusion Detection System with Self Programming Functionality. Basically what its going to be able to do is detect any intrusion on the system and has the ability to create a fake program that will help an attacker get access to with complete analysis on the pattern of attack. It will also create fake counter measures to prevent the attack on the fake program and secure the original application and track the activity and pattern of attack. The system will also log every moment. However, from looking at past projects and hearing about other student's ideas I have lost some confidence in

this idea. It seems that these projects need to have a customer and have a certain appeal i.e. most projects seem to be aimed at facilitating everyday needs or hobbies. I feel like my idea lacks this sense of usefulness and has made me feel like I'm not on the right track at all. I don't know who my customer would be for something like this? However, the presentations are next week, October 4<sup>th</sup> and I have begun thinking about picking a project from the list provided by the college. I am currently just trying to weigh out my options.

## - October 2016 -

### Pitch Presentation

On October 4<sup>th</sup> we had our pitch presentations where students had to pitch their ideas to a group of three lecturers where it would either be: accepted, accepted with revisions or rejected. This process was a very good idea as it improves the overall quality of all final year projects. After much thought and debate with fellow classmates, I decided to go with a project from the approved list from the lecturers, as I didn't have enough confidence in my own idea. During my pitch presentation I informed the lecturers of my idea for an Intrusion Detection System although I had already requested to choose from the list. To my dismay, they told me that the idea (although quite vague) was one of the best they had heard for the Cyber Security stream. I was very disappointed that I didn't put more thought into my idea as I was under the impression that it wasn't good enough or not the right type of project. My own lacks of confidence lead me astray, but my luck hadn't run out quite yet. Sara Kadry, my Secure Applications Programming lecturer, had proposed an Intrusion Detection System for Malicious Email as one of the projects that students were able to choose from an approved list. As I was extremely interested in working in this area I pounced on this opportunity right away. All the doubt that I had with my own idea had been washed away and I couldn't wait to start. I began my research into filtering emails and had a look at how Gmail and other big emailing services done this job already by looking at source code. I wanted to have basic knowledge when meeting Sara for the first time to ensure that I would be given the opportunity to work on this project.

## Supervisor Meetings

On October 13<sup>th</sup> I had my first meeting with Sara. I felt I had learned enough through my initial research to appear ready to take on this project. We discussed different types of approaches that could be taken to complete the project. We conversed on the subject for about 30 minutes and it was an extremely helpful and learning experience. I came away from this meeting with a better feeling of confidence that I was ready for the job. Upon the advice from Sara, I furthered my research and began to look at different types of Spam filters. One such spam filter was SpamAssassin – an open Source mail filter, written in Perl, to identify spam using a wide range of heuristic tests on mail headers and body text. The idea to implement this IDS (Intrusion Detection System) onto an existing web application of mine was suggested. As one of my third year projects we developed a charity website where users could make donations to a charity of their choice. At first I thought this was a good idea, but later decided against it, as you will read later on. I also began looking at all the different features of Gmail, and picked out what features were similar in my own project, as I wanted to create something new, not something that has already been done before. I had a lot of work cut out for me after my first meeting but was extremely excited to make some progress.

With our Project Proposal deadline on the 21<sup>st</sup> of October it was time to put a concrete idea together. My final idea was to create an IDS that specifically targeted phishing emails. This is something that I had experienced myself in the workplace so I thought it might be a good idea to apply this experience into my project. The proposal took quite some time to complete but I was very happy with the outcome. I had been in contact with Sara during this time via email. I would send her my first draft and she would reply with some very helpful feedback. Although the IDS had been thought out, I was still unsure of what to implement it onto. As said previously, I wasn't very pleased with the idea of implementing it to my charity donation web application. But during another one of my meetings with Sara, she suggested the idea of creating a web application that allowed users to share photos. This was a great starting point and I hopped on the idea right away and began to look at source code of existing applications. With our Requirements Specification deadline approaching I have placed most of my time on this document, which is due on the 11<sup>th</sup> of November.

## **- November 2016 -**

### **Initial Prototype – Week 1**

On November 4th I began creating my initial prototype for my final year project. It was easier to start now that I had gathered a lot of my requirements, as our requirements specification document is due on the 14th. I researched a lot of open source photo sharing websites and tried to get a feel for the type of website I wanted to build. This took more time than anticipated and with a lot of assignments and projects due in the next few weeks things were starting to pile on the workload started to get bigger and bigger. During my first week of November I had to put my prototype aside to focus on my other work.

### **Requirements Specification – Week 2**

I spent most of week 2 focusing on deadlines and upcoming CA's for Artificial Intelligence and Web Services & API Development, and continued my work on my Requirements Specification. I spent a lot of time during these few days working to get the initial version of my Requirements Specification for my Software Project completed in time for the submission date. Upon completing my RS document I started to gain a better image of my application after creating a mockup GUI, this had me excited and I wanted to get to work right away. I put more time in with my prototype and made a simple login. It wasn't much, but it was a start. Other assignments/projects started to flood in and once again I had to put the prototype aside to focus on these.

### **ShareSafe Prototype – Week 3**



I spent more time this week to work on my prototype, I added security to my web application that is deployed to the Oracle GlassFish Open Source Edition. I configured security authentication using a login form in a web page. After I created the users, I then created the security roles by setting the security properties in the deployment descriptor. I also gave my application a name: ShareSafe.

### Technical Report – Week 4

My final week of November was mainly spent focusing on my Technical Report. This is a huge document and will be used for our midpoint presentation in December. The document isn't due until the 9th of December but with so many other assignments/projects I wanted to get an early start. Once I am happy with the Technical Report I will continue work on my working prototype, which I will demo during this presentation.

**- December 2016 -**

### Presentation Preparation – Week 1

December was a very busy month with numerous project deadlines, it was an extremely stressful period couple of weeks but I was able to get everything submitted on time. On top of this we all had our mid-point presentation to prepare for. Unlike most students, I had my presentation in week 12 while we were still working on our projects, which was a bit of a disadvantage but nothing too major. I was slightly disappointed with how my presentation went due to being so busy with other work but I felt my documentation (Technical Report) was strong enough to pull me through. The presentation was a good learning experience, as I now know what is expected when it comes to our final presentation in the coming months. My supervisor (Sara Kadry) and second marker (Joe Molumby) were extremely helpful on the day with their feedback and advice on what to improve for the next presentation. They were extremely friendly and made the whole experience much more pleasant and less stressful.

Overall I feel happy with the work I put in with my project but I need to improve on my presentation skills and include much more information in my slides for the next one.

## Exam Preparation

For the rest of the month I need to focus on my exams, which commence on the 5<sup>th</sup> of January 2017. They start very early this year so much of the Christmas break will have to be sacrificed in order to study. I will have to put my progress on this project on pause so that I can focus more on my studies but I look forward to continuing my work when I return for semester 2.

### - January 2017 -

#### 1<sup>st</sup> to 14<sup>th</sup>

The first two weeks of January were primarily focused on studying for my exams so very little work has been put towards my project. I am very pleased with how my exams went which has given me a lot of motivation to work on my project before we return back to college for our second semester.

#### 15<sup>th</sup> to 21<sup>st</sup>

During this time I put a lot of effort into improving my algorithm for sharing photos with trusted users. I also spent some time improving my prototype and giving it a much sleeker and user-friendly interface. I am happy with the progress being made so far.

## 22<sup>nd</sup> to 28<sup>th</sup>

During this time I was on a family holiday in Dubai so no work was done. I unfortunately had to miss our first week back at college but I will put in more effort to make up for lost time. I contacted my supervisor to let her know that I would not be present during this time.

## - February 2017 -

### 1<sup>st</sup> to 8<sup>th</sup>

During one of my supervisor meetings with Sara, she proposed a fantastic idea. As I am a current employee at ACIA (Aon Centre for Innovation and Analytics), Sara suggested that I ask permission to implement my project on their website. It would look great to implement the project to a real life business and look very impressive during my presentation and showcase. I thought this was a fantastic idea so I organised a meeting with my manager in work. Unfortunately this was not something that Aon was prepared to help with as they already have an Intrusion detection system implemented in the business. But it was all worth a try. With this in mind I began to work on the development of my project.

### 9<sup>th</sup> to 15<sup>th</sup>

After much thought, I decided to use PHP for my project instead. Bit of a risky move so late in the college year but I feel it is a good decision. PHP is very popular; therefore there are a number of references and guidelines available on the net. One can also find many support groups, forums, and teams supporting PHP. As I am not the strongest programmer, I feel this will help my progress tremendously as there's always enough online library support to get you through. PHP, being very fast to develop ensures that there is a quick turnaround time. It also has multiple extensions and is extremely scalable.

Before I began to create my index.php page I downloaded and installed XAMPP to store the project on a local server. XAMPP is a free open source cross-platform web server solution developed by Apache Friends. It consists of the Apache HTTP server, MariaDB database and interpreters for scripts written in PHP and Perl programming languages. I was excited to be finally starting development on my

project but I have a lot of work ahead of me. I created the logo for my website using Adobe Fireworks and also created some other icons such as a search bar, search button, home button, header sliver and so on. This took more time than it should of as I was learning the ropes of fireworks, but online tutorials helped me get through it. I am quite pleased with the design and aesthetic of my website. I now need to focus on Database Creation and Table Structure and from there move on to adding in some of the functionality needed. A lot of work ahead but it is exciting and I look forward to progressing further.

### 16<sup>th</sup> to 28<sup>th</sup>

All students were required to get their photos taken in the Atrium for our project showcase profiles (photos could also be used for LinkedIn). I updated my showcase profile by entering a 40-word profile for the project and also a 200-word profile, which was a more detailed description. I look forward to developing my project further in the next few weeks with the kind help and support from Sara.

## - March 2017 -

### 1<sup>st</sup> to 8<sup>th</sup>

Created the database on my host environment and wrote a modular PHP script for establishing connection to the database using the improved MySQLi extension. After doing some research I found that PHP and MySQL are the core data processing technologies of some of the most popular social web applications. I then scripted table creation into my MySQL database to establish all of the software system's tables using PHP and SQL syntax. To add a level of security to my web application, I established a .htaccess file and learned to prevent directory file listing as well as specify a more user friendly 404 error page through htaccess. I used the php.ini file to adjust my PHP configuration settings on the apache server. I then created the main JavaScript module, which is the file that will be called into every page of the web application. I spent the rest of this week to work on other CA's and started preparing for my upcoming exams.

## 9<sup>th</sup> to 16<sup>th</sup>

Began progress on programming a user sign up form and corresponding email activation script. I used HTML, CSS and JavaScript for user interfacing. The server side scripting is PHP connected to a MySQL database. This section of the project includes real time field restricting, user must view terms of use, real time username checking and more. Ajax is in use for all server side data transmissions. User must activate their account through email in order to interact on the application. I used PHPMailer for automated emails being sent out to users for signing up and sending temporary passwords, as you will read further down. PHPMailer is a class library for PHP that provides a collection of functions to build and send email messages. PHPMailer supports several ways of sending email: mail(), Sendmail, qmail & direct to SMTP servers. You can use any feature of SMTP-based e-mail. The email activation process verifies and confirms that they have given their real email address. After the sign up form was complete I programmed the HTML/PHP/Ajax login form, the log out script, and started the user profile page. I also created a module for checking user session and authentication against the MySQL database. Once this functionality was completed I began research on site maintenance automation using Cron Jobs. Upon my research I discovered that I could set them to execute at any interval. In my application I created a PHP cron job that will delete all non-activate users after 10 days. The reason I implemented this was to clear the database of these users periodically so that the system doesn't become clogged with non-activated users. I spent the rest of this week to work on other CA's and started preparing for my upcoming exams.

## 17<sup>th</sup> to 25<sup>th</sup>

I programmed an Ajax PHP MySQL Forgot Password script for the users who may happen to forget their password. This logic generates a temporary password for the user. Also implemented in their account editing functionality is an option for the user to change their password to something they desire as often as they like. Once that was completed I focused my attention on creating web development logic behind creating "Friend" systems and "Block User" systems using PHP, MySQL, JavaScript Ajax, and simple dynamic HTML rendering. I then programmed PHP and mysqli logic into my header file that renders dynamic links for the users. If the user is not logged in they will see a different set of links than a logged in user. I then implemented PHP, MySQL, JavaScript and Ajax programming

logic behind scripting friend lists and notification lists. This also included accepting and rejecting friend requests, which gets placed into the friend\_system.php module that can be called by Ajax anywhere in the web application.

## 26<sup>th</sup> to 31<sup>st</sup>

Implemented the functionality to allow users to change their profile photo and began building the photo upload application. I had to familiarize myself with the PHP and MySQL side of creating image upload applications and created a simple front-end form. I created a PHP and Ajax photo gallery, the initial gallery thumbnails are loaded using PHP, everything else is brought into view using Ajax that requests images from PHP and MySQL. It also contains a simple file upload script so that the users can populate their galleries with pictures. I installed a delete link only for the photo owner when they view any images that they uploaded themselves.

I wanted to continue working on my project as I felt I was making some good progress but I had to set aside more time for other assignments and exams. Very busy time!!

## 7.2 Project Proposal

### - Objectives -

Nowadays, users all around the world use email as their fundamental method to share information over the web. The network providers allow all types of email for the purpose of communication. During this transfer of information some malicious emails are received which can cause problems either at the server side or at the client side. In this project, we propose an intrusion detection system for detecting these malicious emails.

In recent times, one of the most dangerous threats against personal data security at home and in the workplace is phishing. Phishing has become an extremely common form of cyber attack. It consists of defrauding people by luring them to fake websites where users unknowingly provide personal details such as login information and credit card details. These fraudsters appear as a trusted third party, like a well-known bank. The most common methods of phishing are done by email. Once these details are acquired they can be used in the practice of identity theft or credit card fraud. In the past, efforts have been made to stop these attacks by identifying phishing sites using plug-ins, but these efforts have been made in vain by emerging blocking techniques, which render them useless. There is an abundance of these types of attacks, so much so, that the everyday user will be in danger whether they know it or not. I will also include a personal experience of my own later in this document, which brought this topic to my attention. In this project we propose an Intrusion Detection System for identifying these types of malicious emails and root them to their source to evaluate. This will be made possible by using a data capture facility that will categorize some of the incoming emails as potential phishing activities and an evaluation system that will send crawlers to websites related to these detected emails to determine their true intentions. By detecting malicious emails in incoming traffic, this filters a user's inbox and removes the requirement of a user being trained in the practice of secure web browsing. As most users are not trained in this manner, this system will prove quite useful. The Intrusion Detection System (IDS) will detect phishing emails and ensure that all incoming traffic is not harmful. Upon detection of a malicious email, the next step is to send crawlers to these phishing websites that are linked in these emails and also the website that it is trying to impersonate. An algorithm then strips both sites down and compares them using a scoring system for the difference between the two – ultimately deciding whether or not it is a phishing website.

## **- Background -**

I am a current employee of ACIA (Aon Centre for Innovation and Analytics) working part-time during my studies. I have been an employee of ACIA since the summer of 2015 when I applied for an internship. I have not been working with this company for long but I have seen my fair share of misleading and malicious emails in the workplace. Even in a technology-based company such as this, employees were still the victims of these phishing attacks. It just goes to show

how complex and deceitful these attacks are becoming. This was the basis of my decision to design and develop an Intrusion Detection System for my final year project. It was one attack in particular that was successful among a small number of my colleagues. It occurred during my six-month placement for third year, around the month of July. A number of staff began to receive fake emails purporting to come from the Revenue Commissioners and were warned soon afterwards that although the email address seems valid, it is just a piece of text and can easily be faked. Email addresses are not verified and cannot be relied on as proof of the identity of the sender organization. These emails seemed very professionally made and tried to trick users into believing that they were due a tax refund, mostly in the range of around €160, and enticed them to enter bank details. It wasn't until the very next day that similar emails started to roll in. Colleagues were warned to remain extremely vigilant and never to click on suspicious links or provide any sensitive information. This time the phishing emails were in disguise as invoices from Apple and again, looked very legitimate. On office wide email was sent out from IT to warn that proceeds from such scams are quickly transferred offshore, usually through multiple countries/banks and prove impossible to retrieve, even where amounts are greatly in excess of €25.99. Although these types of attacks are very unfortunate, it was good to experience it firsthand in the workplace, and to see how the office dealt with the situation. However I felt that it wasn't enough and I wanted to create something that would tackle this issue head on, eradicating the problem on the user's end. There are a number of current solutions out there that use strong spam filters to segregate phishing solicitations. A few examples would include:

- Spoof Guard
- Spam Assassin
- Phishwish

### **Spoof Guard**

Spoof Guard is a browser based plug-in that monitors user's Internet activity and displays a warning message if it suspects a website to be a phishing site. The tool determines this by observing if the page was loaded from an email and whether or not the URL was visited before. Spoof Guard uses three methods to determine impersonation. 1 – a stateless method that determines whether a downloaded page is suspicious. 2 – a stateful method that evaluates a downloaded page



seeing user's previous activity. 3 – a method, which evaluates outgoing post data. Spoof Guard uses an aggregate function to calculate the total spoof score (TSS).

### **Spam Assassin**

Spam Assassin is a tool that recognizes spam containing phishing email. It has a false negative of 15% for spam e-mails, and performs worst when tested with 10 fold cross validation. It uses a range of heuristic tests on the headers of mail to identify spam and can be an IDS for detecting phishing attacks. In its algorithm, it uses text analysis, Bayesian filtering, DNS block list and a Stochastic Gradient Descent method in training a neural network. This is used for its scoring based on perception. Spam Assassin does not delete emails but can route classified emails to mail boxes or folders.

### **Phishwish**

The primary goal of Phishwish is to maximize the number of phishing emails detected while minimizing the number of false positives. It is applicable to emails that instruct the recipient to log into a website. It processes text based and HTML formatted emails, although some rules are only applicable to HTML. Each of the rules are assigned a configurable weight.

This is where I believe there is a significant research gap as these examples are all either client side, plug-ins or just spam filter. There is a gap here with room for improvement.

### **Existing Software**

In this section we will discuss some of the current generation intrusion detection systems:

Network IDS – The network IDS usually has two logical components: the sensor and the management station. The sensor sits on a network segment, monitoring it for suspicious traffic. The management station receives alarms from the sensor and displays them to an operator. The sensors are usually dedicated systems that exist only to monitor the network. They have a network interface in promiscuous mode, which means they receive all network traffic not just that destined for their IP address and they capture passing network traffic for analysis. If they detect something that looks unusual, they pass it back to the analysis station.

Host IDS – The host-based IDS looks for signs of intrusion on the local host system. These frequently use the host system’s audit and logging mechanism as a source of information for analysis. They look for unusual activity that is confined to the local host such as logins, improper file access, unapproved privilege escalation, or alterations on system privileges. This IDS architecture generally uses rule-based engines for analyzing activity; an example of such a rule might be, “super user privilege can only be attained through the su command”. Therefore successive login attempts to the root account might be considered an attack.

Our system has a number of advantages over some of the current systems you can find today:

- The detection of a phishing email is independent of individuals’ settings.
- Because our system is server-based, it allows uniform security enforcement within an organization and reduces the efforts of performing updates.
- It is not vulnerable to an attack that would bypass client browser based defenses.

## **- Technical Approach -**

This system will consist of three main components: a Spam filter to detect unsolicited emails, a Simple Mail Transfer Protocol (SMTP) server which is the de facto standard for sending email over the internet, and Apache web server which is an open source web server. The IDS scanner will scan all received e-mails and will mark some of them as potential phishing messages and then enters them into a database. Emails are classified as phish if they contain embedded HTML. MySQL will be used to store information regarding the detected malicious emails, where data entered into the database consists of the content of the email and time of receipt. A background process will run to scan all URL’s listed on marked emails, strips HTML tags and determines whether or not they have been earlier observed. If the sites have not been previously evaluated, the crawlers will try to determine if they are phishing websites. Known characteristics of phishing emails

will be used to detect them. The spam filter will mark email as phishing emails if they carry some of the following attributes:

- URL including IP addresses
- HTML masked URLs
- Emails containing encoded HTML
- Cross site images

## - Special Resources Required -

The following are highly valued resources in my learning of secure programming:

**Website:** [https://www.owasp.org/index.php/Main\\_Page](https://www.owasp.org/index.php/Main_Page)

**Book:** J. Manico 2014, Iron-Clad Java: Building Secure Web Applications, McGraw-Hill Education.

## - Technical Details -

The language used for developing the project is Java as it is quite advantageous than other languages in terms of performance, tools available, cross platform compatibility, libraries, cost (freely available), and development process.

## - Evaluation -

This project will be evaluated by a number of factors:

- How well it runs upon completion
- How many specified goals of the project have been reached
- Demo this project to a test group and receive feedback and reviews on their experiences