



UNIVERSITY OF  
WOLLONGONG  
AUSTRALIA



# **School of Computer Science & Software Engineering**

Bachelor of Computer Science (Digital Systems Security)

CSCI321- Project

Technical Manual

Group: SS18-4B

Hidir Ibrahim	5920620	muhammad683@mymail.sim.edu.sg
Nitisha Venkatachari	5283048	vp001@mymail.sim.edu.sg
Eustacia Lim You Yan	5922173	yyelim001@mymail.sim.edu.sg

Supervisor: Dr Ta Nguyen Binh Duong

Assessor: Prof Tan Kheng Teck

## Table of Contents

DOCUMENT CONTROL.....	4
ROLES.....	4
RECORD OF REVISION.....	4
EXECUTIVE SUMMARY.....	5
INTRODUCTION.....	5
PROJECT DESCRIPTION.....	6
<i>Background</i> .....	6
<i>Solution</i> .....	6
<i>Objectives</i> .....	6
VISUAL CRYPTOGRAPHY ALGORITHM.....	8
<i>Arguments</i> .....	8
<i>Initialization</i> .....	8
<i>Pixel Traversing &amp; New Image Transformation</i> .....	8
<i>Algorithm of New Image Transform</i> .....	8
LIMITATIONS.....	10
<i>Limitation 1: Short of Manpower</i> .....	10
<i>Limitation 2: No mainstream uses of Visual Cryptography</i> .....	10
<i>Limitation 3: Application Reliant on User's Email Security</i> .....	10
APPLICATION REQUIREMENTS.....	11
<i>Component 1: Frontend Website</i> .....	11
<i>Component 2: Backend Website</i> .....	12
<i>Component 3: Image Processor</i> .....	13
<i>Component 4: Database</i> .....	14
RISK MANAGEMENT.....	15
GANTT CHART.....	16
PROJECT WEBSITE.....	16
USE CASE DIAGRAM.....	17
<i>Iteration 1</i> .....	17
<i>Iteration 2</i> .....	18
DETAILED USE CASE.....	19
WORKFLOW DIAGRAM.....	32
<i>Add Admin</i> .....	32
<i>Delete Admin</i> .....	33
<i>Edit Admin</i> .....	34
<i>Create Subject</i> .....	35
<i>Delete Subject</i> .....	36
<i>Edit Subject</i> .....	37
<i>Create New Critical Section</i> .....	38
<i>Delete Critical Section</i> .....	39
<i>Edit Critical Section</i> .....	40
SEQUENCE DIAGRAM.....	41
<i>Iteration 1</i> .....	41
USER INTERFACE.....	42
<i>Front End</i> .....	42
<i>Admin Menu</i> .....	43
<i>Manage Website Content</i> .....	44
<i>Manage Subject</i> .....	45

<i>Edit Subject</i> .....	46
<i>Create Subject</i> .....	47
<i>Subject Created</i> .....	48
<i>Subject Deleted</i> .....	49
<i>Manage Admins</i> .....	50
<i>Create New Admins</i> .....	50
<i>Update Admin's Username &amp; Password</i> .....	51
<i>Delete Admin</i> .....	51
<i>Verification</i> .....	52
<i>Manage Critical Section</i> .....	55
<i>Add New Critical Section</i> .....	55
<i>Update New Critical Section</i> .....	56
<i>Delete Critical Section</i> .....	56
DEVELOPMENT METHODOLOGY .....	57
TEST CASES .....	58
DEVELOPMENT TOOLS.....	68
<i>Main Stack</i> .....	68
<i>External Libraries Used</i> .....	69

## Document Control

Title: Secure Remote Authentication using Visual Cryptography

Owner	Current Version	Last Modified	
Hidir Ibrahim	1.0	Date	Time
Nitisha Venkatachari		05/12/2018	0100
Eustacia Lim You Yan			

## Roles

Name	Role
Hidir Ibrahim	Project Leader
Nitisha Venkatachari	Lead Designer
Eustacia Lim You Yan	Lead Developer

## Record of Revision

Revision Date	Description	Section Affected	Changes Made By	Version
5/12/2018	Document Created	All	Everyone	1.0
30/12/2018	Modification of document	Detailed Use Case Gantt Chart	Everyone	1.1
20/2/2019	Added Test Cases & Updated Stack List	Development Tools Test Cases	Everyone	1.2
27/2/2019	Modification of Document	Workflow Diagram, Use Case Diagram, Interfaces	Everyone	1.3
1/3/2019	Modification of Document	Standardize the document	Everyone	1.4

## Executive Summary

The aim of this project is to increase security to a website/application by providing an additional authentication method on top of the traditional password protocol for logging in the user by utilizing Visual Cryptography.

## Introduction

Our team feels that the current mainstream method of a single password-based authentication is not enough to fulfil the security needs of websites today. Our team proposes an additional authentication-method on top of a password to further improve security.

With this documentation, we aim to make clear:

- Problems with a single-authentication based protocol
- Our proposed way of providing authentication using Visual Cryptography

This documentation will also cover these topics in the following order

- Reasons for proposing this project
- Existing applications/proposals of Visual Cryptography
- Our Application Objectives & Requirements
- Development Methods
- Roles & Timetable

## Project Description

### *Background*

In today's high technology environment, IT has become an indispensable part of any enterprise or organization. Accurate business planning, effective marketing, and long-term business growth at the optimum level is impossible to achieve without IT. As such, organizations are becoming more and more dependent on their information systems.

People are increasingly concerned about the proper use of information, especially personal data. Many organizations will identify information as an area of their operations that needs to be protected as part of their system of internal control. Unfortunately, cyber criminals like unethical hackers are now more common, and the threat of organization getting crucial and confidential data stolen is very real.

On the other side of things, single-factor authentication is the simplest form of authentication methods. Most verification today uses only password as an authentication method.

However, there are many users who are still using weak passwords for their work accounts and email accounts. Many even use the same password for all their accounts, as it is more convenient for them to remember. This makes it much easier for cyber criminals to access their accounts and all the information that is available. If one's get compromised, it could cost thousands of dollars and irreversible damage for the organization.

### *Solution*

To solve the current situation, we have come up with the idea of enhancing authentication by implementing two-factor authentication using Visual Cryptography. Visual Cryptography hides secret within the images, images transformed into multiple shares and decoding is done by stacking the share which will reveal the secret image or text. For the user to login into the account, the user required password and registered email access to retrieve the shares that is sent to the user to complete the login process.

Notable critical sections of the systems are registered in the database by name. If authorized users wish to access one of these pages, they are required to access their linked email to retrieve the shares sent to them. Only by uploading the legitimate shares would the user be able to view the correct challenge to submit.

### *Objectives*

Our goal of the project is to provide an additional form of authentication by utilizing the power of Visual Cryptography to entities who have websites that require more than a password-based protocol, saving them thousands to millions of dollars in damages from potential attacks a stand-alone password-based authentication is susceptible to.

In traditional single-authentication systems, a brute force attack would give attackers easy access to very powerful operations or content should access only be relying on a password. Damage from this attack could cost thousands of dollars and irreversible damage for some businesses.

With our product, it will require an additional challenge for an attacker to do damage. Having cracked the password would not be enough, as the attacker would be required to access the email linked to the user to access critical sections in the system. This is done using splitting of images into shares known as Visual Cryptography.

Some examples of uses various entities would benefit from our product:

- School systems that allows teachers remote access for grading and timetables.

- A private membership-based service providing valuable content to its members such as access to research papers to registered professors.

## Visual Cryptography Algorithm

We use a python script which is responsible for converting the original generated challenge image into 2 shares.

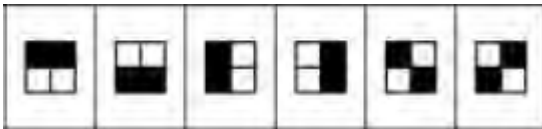
### Arguments

The script takes in 3 arguments.

1. Image to split
2. Admin ID
  - Used for renaming of file
3. Path to save created images

### Initialization

1. Check if the file exists.
2. Script opens file into an Image Class.
3. Converts image class into 1 bit
  - This converts the image into black and white.
4. System prepares image class for the 2 output images: Share 1 & Share 2
5. System sets width and height of new shares to be double of the original as every 1 pixel is now 4 pixels.
6. System defines the 6 possible patterns:



### Pixel Traversing & New Image Transformation

After initializing is complete, script start traversing each pixel one by one. At each pixel, depending on its color, transform the 2 new images based on value of this pixel. Move on to the next pixel.

#### Algorithm of New Image Transform

1. Get the current value of pixel on current coordinate of original image
  - Either 1 or 0,
  - 1 = White Pixel
  - 0 = Black Pixel
2. Randomly choose from 1 of the 6 available patterns. (from step6 of initialization)
3. From this pattern, we will transform Share 1 and Share 2



## If pixel was white or 1

- Share 1 follows pattern chosen in Step 2
- Share 2 follows the exact same pattern.

**Result:** Combining the 2 shares will not reveal any changes/information and only show black and white noise.

## If pixel was black or 0

- Share 1 follows chosen pattern in Step b
- Share 2 follows the complete opposite pattern.

**Result:** Combining the 2 shares will now reveal a distinct black pixel. A combination of distinct black pixels will reveal the hidden challenge.

The pixels of the	white □						black ■					
share A												
share B												
Stacking												

## Algorithm of Pixel Traversing of Original Image.

1. Starts from (0,0) to (0, **a**) where **a** is the max height of the image
2. Moves on to (1,0) to (1, a)
3. Repeat until we reach (b,0) to (b, a) where **b** is the max width of the image.
4. In simple terms: Go through column by column from left to right.

## Limitations

### ***Limitation 1: Short of Manpower***

Our team faces a unique problem of having the least possible members (3) for a FYP group, as compared to traditional groupings with 4/5 member.

To overcome this, it more important for every member to be comfortable with different specialization including those outside our role as even a designer will be expected to take part programming of the product.

Everyone in our team should not only be fluent in coding in Java but also be able to do designs such as constructing of diagrams such as Use Case & Class Diagrams.

### ***Limitation 2: No mainstream uses of Visual Cryptography***

Since there is little mainstream use for Visual Cryptology, we predict we will struggle more in finding solutions should we face problems when starting work on the Visual Cryptography aspect of our product.

To overcome this, we have started reading more on the available research papers on IEEE early on to better understand the topic.

All of us should be familiar with at least 2 different schemes of Visual Cryptography before the start of Phase 2.

### ***Limitation 3: Application Reliant on User's Email Security***

The entire authentication product is based off the assumption that the user logging in is the only one who has access to his email address.

To overcome this, we will remind the users to make sure their own email providers have a form of 2-factor authentication to further reinforce security.

To make sure this system is secure, registration requires user to be registered to an email provider that supports 2FA.

## Application Requirements

Our system will consist of 4 main components

- Front-end Website
- Back-end Website
- Image Processor
- Database

### *Component 1: Frontend Website*

The frontend website will consist of the public website that is accessible to anyone to view. It will be coded in PHP & HTML.

#### Functional Requirements

- Website should allow any users to access the website.
- Website should display content of pages dynamically based on the data from database.

#### Non-Functional Requirements

- **Ease of Use:** - User should be able to navigate between pages easily.
- **Platform Compatibility:** - Website should work on all major browsers supported on Windows & Mac.

#### Potential Problems

- Converting generated string to image
  - We will need to find a way to convert a string into an image that displays that string.

**Solution:** Find a library that does this securely.

## ***Component 2: Backend Website***

The backend website will consist of an admin login page & an admin panel. It will be coded in PHP & HTML.

### Functional Requirements

- Website should authenticate user through password to access admin panel.
- Website should allow password-authenticated admins to modify the front-end website.
- Website should allow highest-level admins to manage access critical sections of backend
- Websites should authenticate users through VC 2FA to access any pages listed under critical sections.
- Website should allow users to upload their image shares.
- Website should be able to email user their share.
- Website should be able to generate a random string and convert it into an image displaying the string.

### Non-Functional Requirements

- **Documentation:** - A user manual should show users how to use the website.
- **Ease of Use:** - User should be able to manage the front-end website through the backend with ease after reading the User Manual once.
- **Platform Compatibility:** - Website should work on all major browsers supported on Windows & Mac.

### Potential Problems

- Converting generated string to image
  - We will need to find a way to convert a string into an image that displays that string.

**Solution:** Find a library that does this securely.

***Component 3: Image Processor***

This will be our propriety technology that uses Visual Cryptology to provide a second form of authentication. It will split an image of a random string of alphanumeric characters into 2 separate share images. It will be coded in Python. It should also be able to combine 2 different images of same resolution into 1.

**Functional Requirements**

- Image Processor should be able to take in 1 image as input and output 2 separate share images without revealing any information of original image.
- Image Processor should be able to combine 2 different shares into 1.

**Non-Functional Requirements**

- **Documentation:** A user manual to show users on how to use the Image Processor within 1hour of reading.
- **Integrability:** Image Processor should be able to communicate with the backend system.
- **Performance:** Image Processor should not take longer than 10seconds to split or combine an image.

**Potential Problems**

- Combining 2 shares
  - If user uploads a share that is of different resolution from the server-held share, combining the 2 shares might crash the applet.

**Solution:** Check resolution of user-submitted share matches that of server-held share, before proceeding with combining. If not, automatically deny.

## ***Component 4: Database***

Our database will hold user information. We will be using MySQL for our database.

### Functional Requirements

- Database should store information of all pages
- Database should hold names of pages identified as critical sections.
- Database should hold hashed values of any sensitive information.

### Non-Functional Requirements

- **Data Integrity:** - Data provided to the users should be authentic and should not be easily manipulated.
- **Security:** - Users should not have any loss of personal data, nor should their data be misused.

### Risk Management

	Impact Type	Risk Seriousness (%)	Likelihood of Occurrence (%)	Risk Description
1	Change Management	90%	20%	Change requests cannot be fulfilled
		10%	40%	Ambiguous change requests
2	Communication	90%	20%	Project Team misinterpreted the requirements.
		70%	10%	Team members are not updated with the progress of the project.
3	Design	80%	20%	System does not meet the functional requirements.
		50%	30%	System does not provide sufficient usability.
4	Time Management	80%	5%	System not completed on time.

## Gantt Chart

	Task Name	Duration	Start	Finish	Predecessors	Resource Names
GANNT CHART	1 <b>Secure Remote Authentication using Visual Cryptography</b>	<b>103 days</b>	<b>Sat 13/10/18</b>	<b>Sat 2/3/19</b>		
	2 <b>Phase 1 - Inception Phase</b>	<b>15 days</b>	<b>Sat 13/10/18</b>	<b>Wed 31/10/18</b>		
	3 Drafting of Project Plan	14 days	Sat 13/10/18	Tue 30/10/18		Hidir,Nitisha,Eustacia
	4 Setting of Roles and Responsibilities	1 day	Sun 28/10/18	Sun 28/10/18		Hidir,Nitisha,Eustacia
	5 Project Briefing	1 day	Wed 31/10/18	Wed 31/10/18		Hidir,Nitisha,Eustacia
	6 Requirement Gathering	7 days	Mon 15/10/18	Tue 23/10/18		Hidir,Nitisha,Eustacia
	7 Research on Exisiting Projects	7 days	Mon 15/10/18	Tue 23/10/18		Hidir,Nitisha,Eustacia
	8 Research on Visual Cryptography	45 days	Sat 13/10/18	Wed 12/12/18		Eustacia
	9 <b>Phase 2- Elaboration Phase</b>	<b>48 days</b>	<b>Thu 1/11/18</b>	<b>Sat 5/1/19</b>		
	10 Drafting of Use Case Diagram	7 days	Thu 1/11/18	Fri 9/11/18		Eustacia
	11 Drafting of Sequence Diagram	9 days	Thu 13/12/18	Tue 25/12/18		Eustacia
	12 Identify & Analyse risk	3 days	Thu 1/11/18	Mon 5/11/18		Eustacia
	13 Setting up Project Website	3 days	Thu 1/11/18	Mon 5/11/18		Nitisha
	14 Drafting of Website Design	7 days	Fri 7/12/18	Mon 17/12/18		Nitisha
	15 Iteration of Website Design	3 days	Tue 18/12/18	Thu 20/12/18		Nitisha
	16 Prototype Test Cases	5 days	Thu 1/11/18	Wed 7/11/18		Hidir
	17 Coding of Prototype Design	19 days	Tue 11/12/18	Fri 4/1/19		Hidr
GANNT CHART	18 <b>Phase 3 - Construction Phase</b>	<b>35 days</b>	<b>Mon 7/1/19</b>	<b>Fri 22/2/19</b>		
	19 Review Tech Manual	35 days	Mon 7/1/19	Fri 22/2/19		Hidir,Nitisha,Eustacia
	20 Image Comparison Feature	10 days	Mon 7/1/19	Fri 18/1/19		Hidir,Nitisha,Eustacia
	21 Password Encryption Feature	10 days	Mon 21/1/19	Fri 1/2/19		Hidir
	22 Implementation of additional features	15 days	Mon 4/2/19	Fri 22/2/19		Hidir
	23 Enhance Existing Prototype Design	7 days	Thu 10/1/19	Fri 18/1/19		Hidir,Nitisha,Eustacia
	24 Performance Evaluation	7 days	Mon 21/1/19	Tue 29/1/19		Hidir,Nitisha,Eustacia
	25 Integration of additional features to program	7 days	Thu 14/2/19	Fri 22/2/19		Hidir,Nitisha,Eustacia
	26 <b>Phase 4 - Transition Phase</b>	<b>5 days</b>	<b>Mon 25/2/19</b>	<b>Fri 1/3/19</b>		
	27 Deploy Main Program	1 day	Mon 25/2/19	Mon 25/2/19		Hidir,Nitisha,Eustacia
	28 Write up of Test Case Report	1 day	Tue 26/2/19	Tue 26/2/19		Hidir,Nitisha,Eustacia
	29 Conduct Full Risk & Countermeasure Analysis	1 day	Wed 27/2/19	Wed 27/2/19		Hidir,Nitisha,Eustacia
	30 Deploy Completed Project Website	1 day	Thu 28/2/19	Thu 28/2/19		Hidir,Nitisha,Eustacia
	31 User Acceptance Test	1 day	Fri 1/3/19	Fri 1/3/19		Hidir,Nitisha,Eustacia

## Project Website

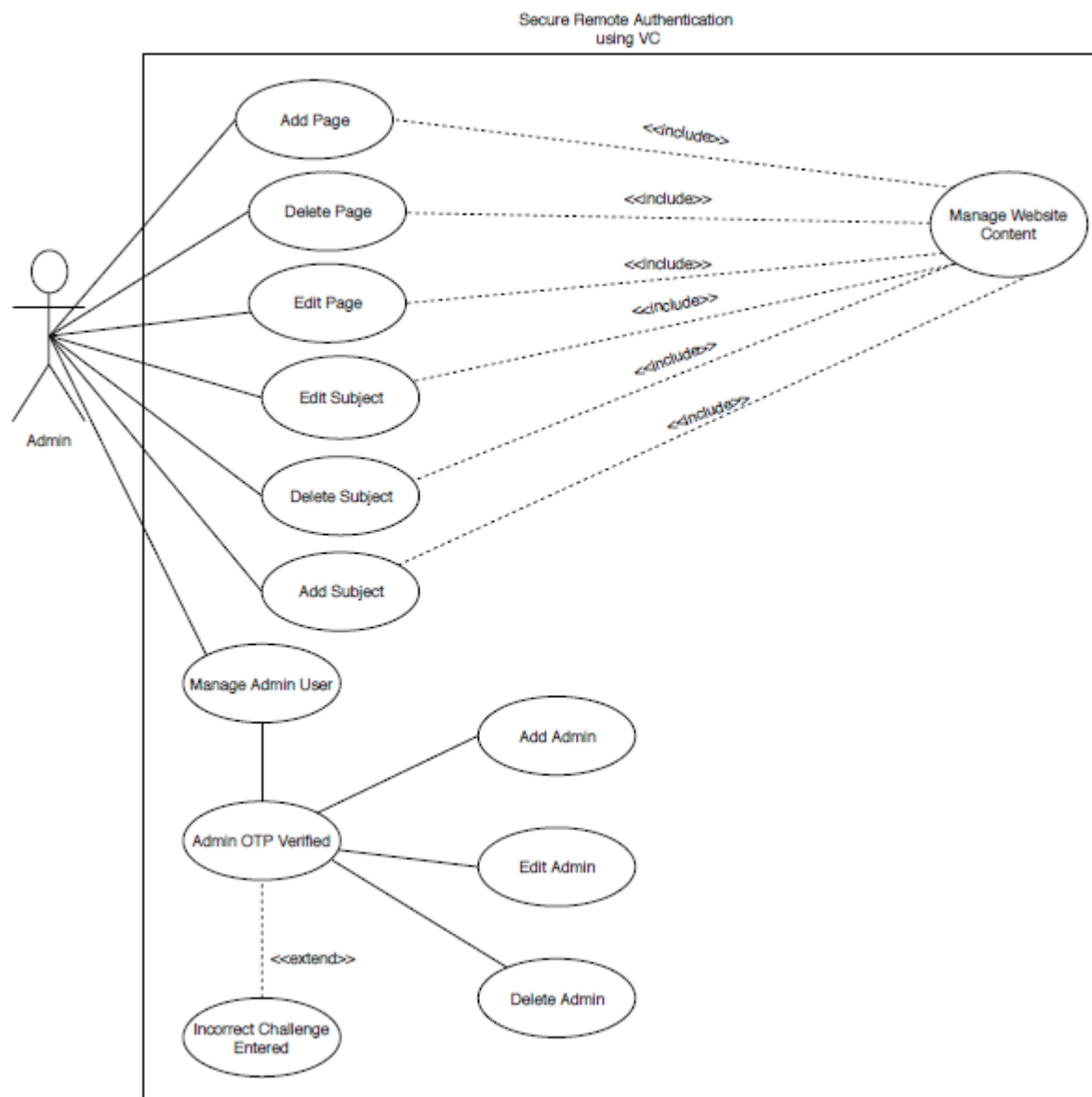
The following is our project website link:

<https://veramorgana.wixsite.com/vcauth/>

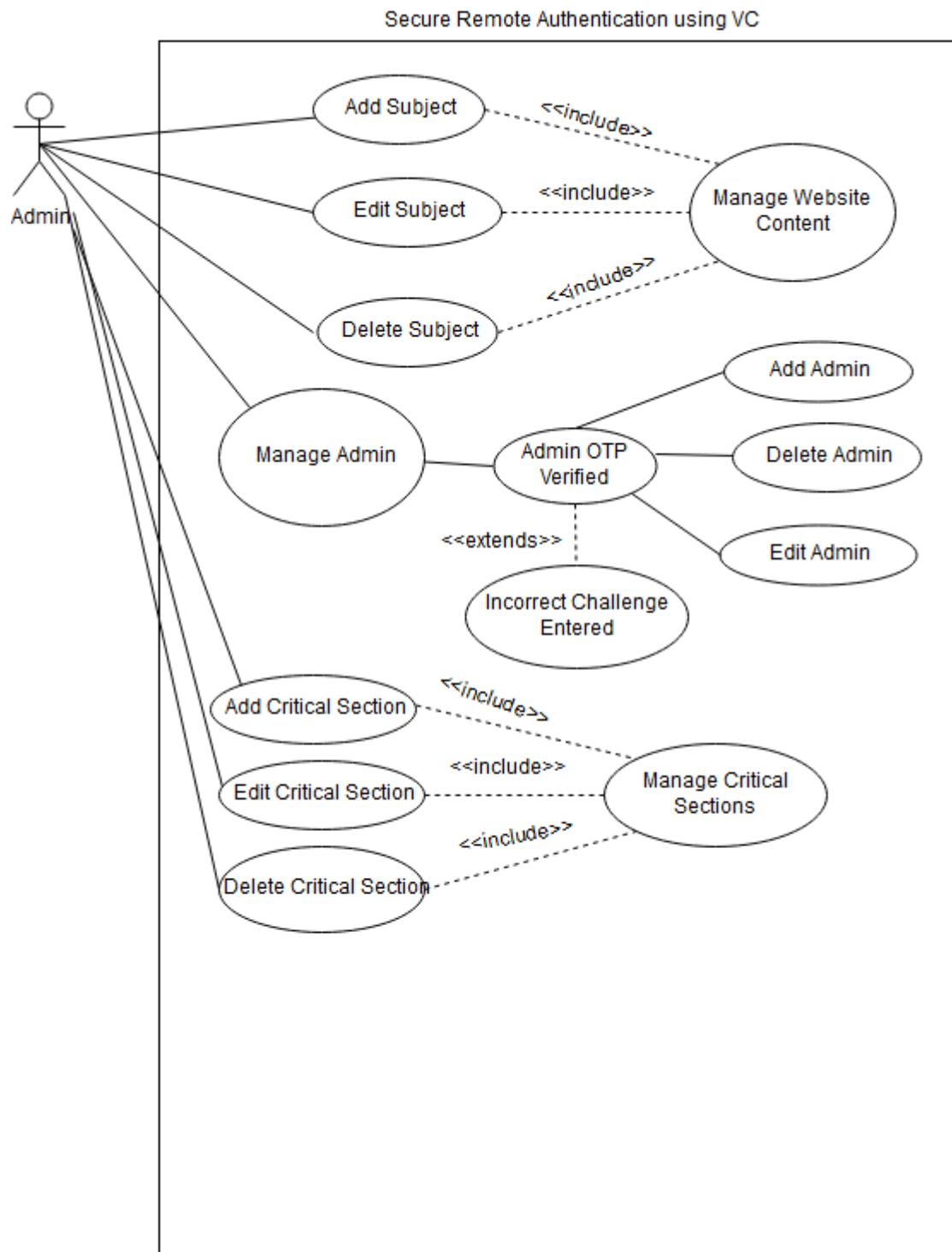


## Use Case Diagram

### Iteration 1



*Iteration 2*



### Detailed Use Case

<b>Use Case ID</b>	UC-0001
<b>Use Case Name</b>	Add Subject
<b>Brief Description</b>	Admin add subject to the website content
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	Subject will be added.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Admin selects “Add New Subject”.</li> <li>2. Admin enter Menu Name.</li> <li>3. Admin selects position and visibility option.</li> <li>4. Admin selects “Create Subject” button to submits the details of the subject.</li> <li>5. Upon completion, the subject should be added under the menu.</li> </ol>
<b>Alternate Flows</b>	<p><b>2a. Menu Name is missing</b></p> <ol style="list-style-type: none"> <li>1. The program prompt for Menu Name.</li> <li>2. Use case resumes at main flow step 3.</li> </ol> <p><b>3a. Visibility option is missing</b></p> <ol style="list-style-type: none"> <li>1. The program prompt for visibility option.</li> <li>2. Use case resumes at main flow step 4.</li> </ol>

<b>Use Case ID</b>	UC-0002
<b>Use Case Name</b>	Delete Subject
<b>Brief Description</b>	Delete on an existing subject from the website content
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	Subject will be deleted.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Admin selects on one of the existing subjects from the menu.</li> <li>2. Admin selects on “Edit Subject”.</li> <li>3. Admin selects on “Delete Subject”.</li> <li>4. System prompts Admin to confirm user’s selection</li> <li>5. Admin selects “OK”.</li> <li>6. Upon completion, the subject will be deleted.</li> </ol>
<b>Alternate Flows</b>	<p><b>5a. When admin deletes a subject with existing pages under it</b></p> <ol style="list-style-type: none"> <li>1. System will display an error message.</li> <li>2. Admin must delete any existing pages linked to the subject first.</li> </ol>

<b>Use Case ID</b>	UC-0003
<b>Use Case Name</b>	Edit Subject
<b>Brief Description</b>	Admin edit on an existing subject
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	Subject will be edited.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Admin selects on one of the existing subjects from the menu.</li> <li>2. Admin selects on “Edit Subject”.</li> <li>3. Admin edit the Menu Name, position or visibility option.</li> <li>4. Admin selects “Edit Subject” button to submits the edited details of the subject.</li> <li>5. Upon completion, the subject of the menu name, position or visibility option should be edited according to the admin preference.</li> </ol>

<b>Use Case ID</b>	UC-0004
<b>Use Case Name</b>	Admin 2FA Verified
<b>Brief Description</b>	User to be 2FA verified before accessing critical section of system.
<b>Actor(s)</b>	User (Admin)
<b>Pre-condition(s)</b>	User accesses a page identified as a critical section by the system.
<b>Post-condition(s)</b>	User granted access to critical section until logged out.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. User attempts to access page</li> <li>2. System check if page is a critical section.</li> <li>3. System generates 6-long random string as a challenge string and converts it into image displaying string.</li> <li>4. System hashes the challenge string and links it to the user.</li> <li>5. System calls VC Script and divides image into 2 shares: User &amp; Server Share</li> <li>6. System emails User Share to email linked to User.</li> <li>7. System keeps Server share.</li> <li>8. System removes base image and user share from server.</li> <li>9. System prompts user to upload share sent to their email.</li> <li>10. User uploads the share they received.</li> <li>11. System calls VC Script and combines user-uploaded image to server share.</li> <li>12. System displays combined image which will display the same challenge string generated in Step 3.</li> <li>13. System prompts user to enter challenge displayed.</li> <li>14. User enters challenge as seen in combined image.</li> <li>15. System hashes user input and checks if same as hash linked to user in Step 4.</li> <li>16. User granted temporary access to critical section of system until logged off.</li> </ol>
<b>Alternate Flows</b>	<p><b>2a. System detects page as Non-Critical</b></p> <ol style="list-style-type: none"> <li>1. User will have direct access to the page and no 2FA is required.</li> </ol> <p><b>10a. User Uploads Incorrect Share File</b></p>

	<ol style="list-style-type: none"> <li>1. Displayed combined image in Step 12 will display gibberish. User will thus be unable to proceed as they are not able to enter the correct challenge.</li> </ol>
--	---

<b>Use Case ID</b>	UC-0005
<b>Use Case Name</b>	Add Critical Section
<b>Brief Description</b>	Admin can add pages that require additional security as critical sections.
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	A page will be added as a critical section
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Admin selects “Add New Critical Sections”.</li> <li>2. Admin enter Menu Name.</li> <li>3. Admin selects security level and decide if they want to use Fast Auth</li> <li>4. Admin selects “Create Critical Section” button to submits the details of the critical section</li> <li>5. Upon completion, the page should be added under the menu.</li> </ol>
<b>Alternate Flows</b>	<p><b>2a. Any Field is missing</b></p> <ol style="list-style-type: none"> <li>1. The program prompt for missing field</li> </ol>

<b>Use Case ID</b>	UC-0006
<b>Use Case Name</b>	Edit Critical Section
<b>Brief Description</b>	Admin can edit existing critical sections and their options
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	A page listed as critical section will be updated
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. Admin selects “Edit Critical Sections” from the list.</li> <li>2. Admin updates Menu Name.</li> <li>3. Admin update security level and decide if they want to use Fast Auth</li> <li>4. Admin selects “Update” button to submits the changes of the critical section</li> <li>5. Upon completion, the page should be added under the menu.</li> </ol>
<b>Alternate Flows</b>	<p><b>2a. Any Field is missing</b></p> <ol style="list-style-type: none"> <li>1. The program prompt for missing field</li> </ol>



<b>Use Case ID</b>	UC-0007
<b>Use Case Name</b>	Delete Critical Section
<b>Brief Description</b>	Admin can delete existing critical sections
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin has already logged in
<b>Post-condition(s)</b>	A page listed as critical section will be deleted.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>Admin selects “Delete Critical Sections” from the list.</li> <li>Admin confirms deletion.</li> </ol>
<b>Alternate Flows</b>	<p><i>2a. Any Field is missing</i></p> <ol style="list-style-type: none"> <li>The program prompt for missing field</li> </ol>

<b>Use Case ID</b>	UC-0008
<b>Use Case Name</b>	Admin 2FA Verified <b>Level 1</b>
<b>Brief Description</b>	User to be 2FA verified before accessing critical section of system by uploading 1 share image sent to his main email.
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin accesses a page identified as a critical section level 1 by the system.
<b>Post-condition(s)</b>	Admin granted access to critical section until logged out.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. User attempts to access page</li> <li>2. System check if page is a critical section.</li> <li>3. System generates 6-long random string as a challenge string and converts it into image displaying string.</li> <li>4. System hashes the challenge string and links it to the user in Database</li> <li>5. System calls VC Script and divides image into 2 shares: User &amp; Server Share</li> <li>6. System emails User Share to email linked to User.</li> <li>7. System keeps Server share.</li> <li>8. System removes base image and user share from server.</li> <li>9. System prompts user to upload share sent to their email.</li> <li>10. User uploads the share they received.</li> <li>11. System calls VC Script and combines user-uploaded image to server share.</li> <li>12. System displays combined image which will display the same challenge string generated in Step 3.</li> <li>13. System prompts user to enter challenge displayed.</li> <li>14. User enters challenge as seen in combined image.</li> <li>15. System hashes user input and checks if same as hash linked to user in Step 4.</li> <li>16. User granted temporary access to critical section of system until logged off.</li> </ol>
<b>Alternate Flows</b>	<i>2a. System detects page as Non-Critical</i>

	<ol style="list-style-type: none"> <li>1. User will have direct access to the page and no 2FA is required.</li> </ol> <p><b><i>12a. Fast Auth Enabled</i></b></p> <ol style="list-style-type: none"> <li>1. System will use Image Recognition Module to allow user to skip the step of having to enter Challenge manually.</li> </ol> <p><b><i>10a. User Uploads Incorrect Share File</i></b></p> <ol style="list-style-type: none"> <li>1. Displayed combined image in Step 12 will display gibberish. User will thus be unable to proceed as they are not able to enter the correct challenge.</li> </ol>
--	---

<b>Use Case ID</b>	UC-0009
<b>Use Case Name</b>	Admin 2FA Verified <b>Level 2</b>
<b>Brief Description</b>	User to be 2FA verified before accessing critical section of system by uploading 2 share images, 1 <sup>st</sup> to his main email, 2 <sup>nd</sup> to his secondary email.
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin accesses a page identified as a critical section level 2 by the system.
<b>Post-condition(s)</b>	Admin granted access to critical section until logged out.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. User attempts to access page</li> <li>2. System check if page is a critical section.</li> <li>3. System generates 6-long random string as a challenge string and converts it into image displaying string.</li> <li>4. System hashes the challenge string and links it to the user in Database</li> <li>5. System calls VC Script and divides image into 3 shares: User Share 1, User Share 2 &amp; Server Share</li> <li>6. System emails User Share 1 to email1 linked to User.</li> <li>7. System emails User Share 2 to email2 linked to User.</li> <li>8. System keeps Server share.</li> <li>9. System removes base image and user share from server.</li> <li>10. System prompts user to upload share sent to their email.</li> <li>11. User uploads both the share they received.</li> <li>12. System calls VC Script and combines both user-uploaded images to server share.</li> <li>13. System displays combined image which will display the same challenge string generated in Step 3.</li> <li>14. System prompts user to enter challenge displayed.</li> <li>15. User enters challenge as seen in combined image.</li> <li>16. System hashes user input and checks if same as hash linked to user in Step 4.</li> <li>17. User granted temporary access to critical section of system until logged off.</li> </ol>

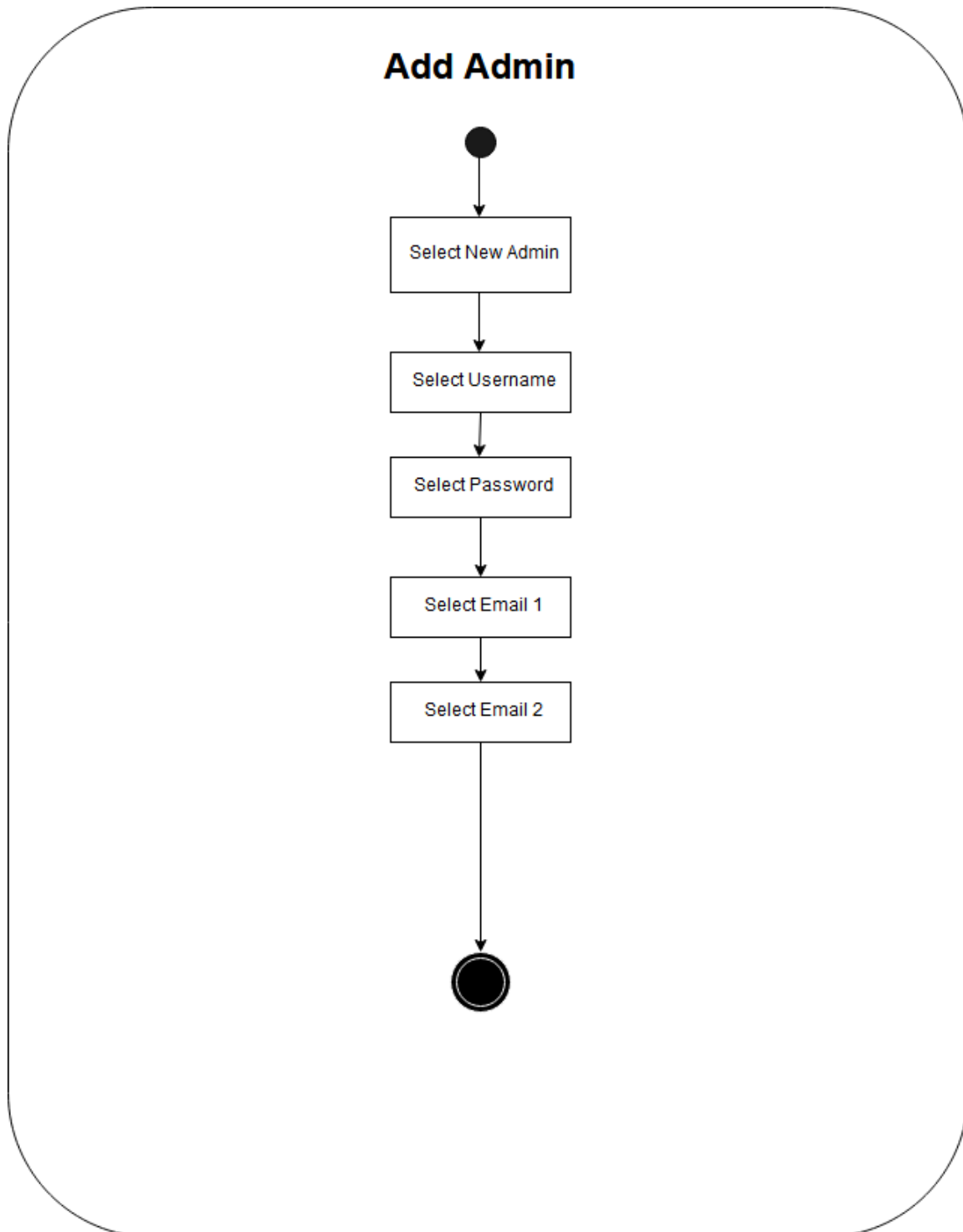
Alternate Flows	<p><b><i>2a. System detects page as Non-Critical</i></b></p> <ol style="list-style-type: none"> <li>1. User will have direct access to the page and no 2FA is required.</li> </ol> <p><b><i>12a. Fast Auth Enabled</i></b></p> <ol style="list-style-type: none"> <li>1. System will use Image Recognition Module to allow user to skip the step of having to enter Challenge manually.</li> </ol> <p><b><i>10a. User Uploads Incorrect Share File</i></b></p> <ol style="list-style-type: none"> <li>1. Displayed combined image in Step 12 will display gibberish. User will thus be unable to proceed as they are not able to enter the correct challenge.</li> </ol>
-----------------	--

<b>Use Case ID</b>	UC-0010
<b>Use Case Name</b>	Admin 2FA Verified <b>Level 3</b>
<b>Brief Description</b>	User to be 2FA verified before accessing critical section of system by uploading 2 share images, 1 <sup>st</sup> to his main email, 2 <sup>nd</sup> to page master
<b>Actor(s)</b>	Admin
<b>Pre-condition(s)</b>	Admin accesses a page identified as a critical section level 2 by the system.
<b>Post-condition(s)</b>	Admin granted access to critical section until logged out.
<b>Main Flow</b>	<ol style="list-style-type: none"> <li>1. User attempts to access page</li> <li>2. System check if page is a critical section.</li> <li>3. System generates 6-long random string as a challenge string and converts it into image displaying string.</li> <li>4. System hashes the challenge string and links it to the user in Database</li> <li>5. System calls VC Script and divides image into 3 shares: User Share 1, User Share 2 &amp; Server Share</li> <li>6. System emails User Share 1 to email1 linked to User.</li> <li>7. System emails User Share 2 to Page Master linked to the Critical Section.</li> <li>8. System keeps Server share.</li> <li>9. System removes base image and user share from server.</li> <li>10. System prompts user to upload share sent to their email.</li> <li>11. User goes to his email to get Share1.</li> <li>12. User await page master to forward him the second share as sign of approval to access the page.</li> <li>13. User uploads both the share they received.</li> <li>14. System calls VC Script and combines both user-uploaded images to server share.</li> <li>15. System displays combined image which will display the same challenge string generated in Step 3.</li> <li>16. System prompts user to enter challenge displayed.</li> <li>17. User enters challenge as seen in combined image.</li> <li>18. System hashes user input and checks if same as hash linked to user in Step 4.</li> </ol>

	<p>19. User granted temporary access to critical section of system until logged off.</p>
Alternate Flows	<p><b><i>2a. System detects page as Non-Critical</i></b></p> <p>1. User will have direct access to the page and no 2FA is required.</p> <p><b><i>12a. Fast Auth Enabled</i></b></p> <p>1. System will use Image Recognition Module to allow user to skip the step of having to enter Challenge manually.</p> <p><b><i>10a. User Uploads Incorrect Share File</i></b></p> <p>1. Displayed combined image in Step 12 will display gibberish. User will thus be unable to proceed as they are not able to enter the correct challenge.</p>

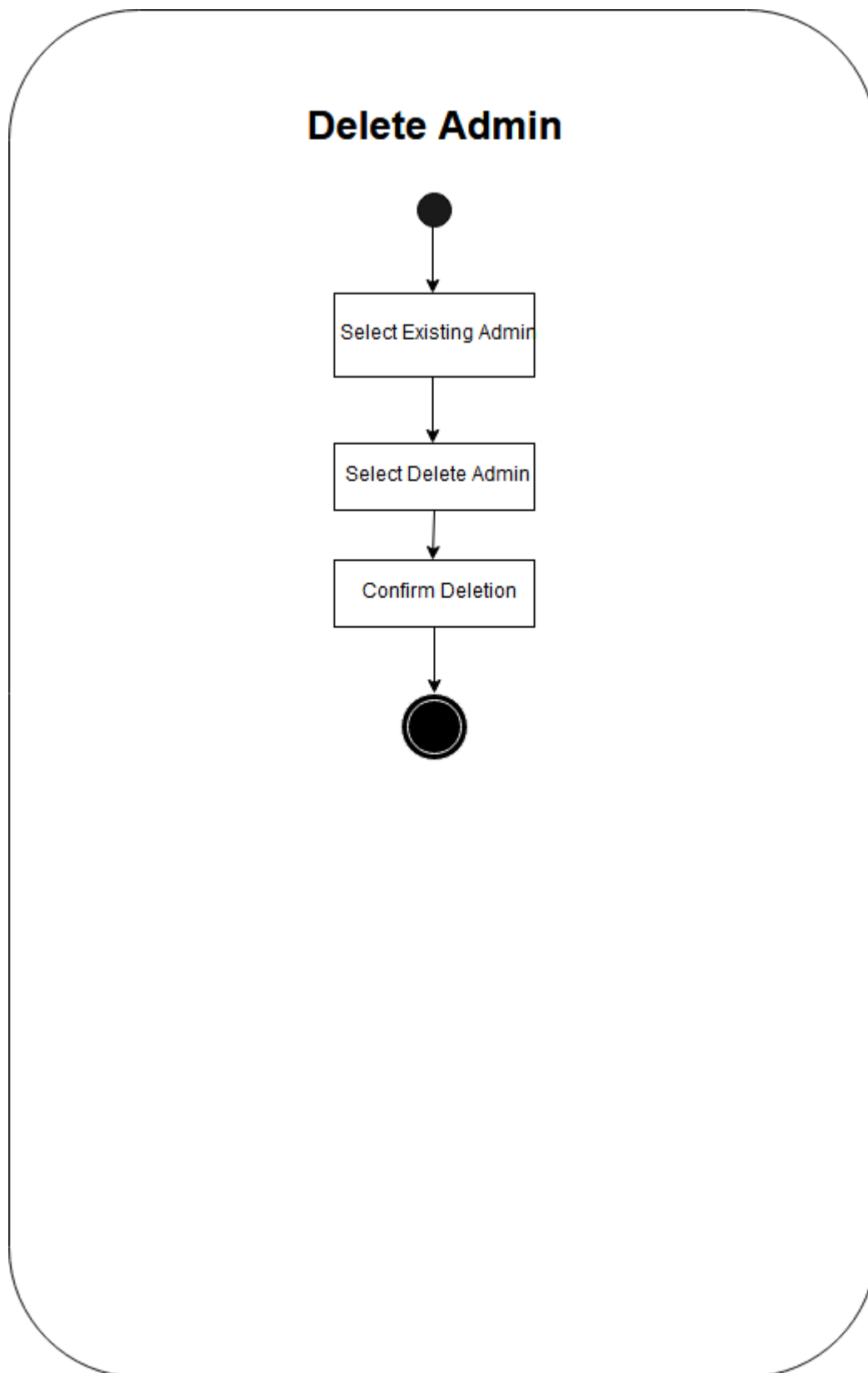
## Workflow Diagram

### Add Admin



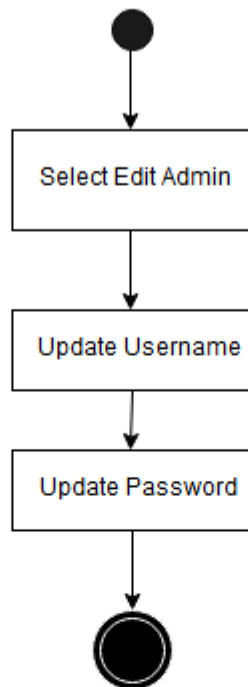


*Delete Admin*

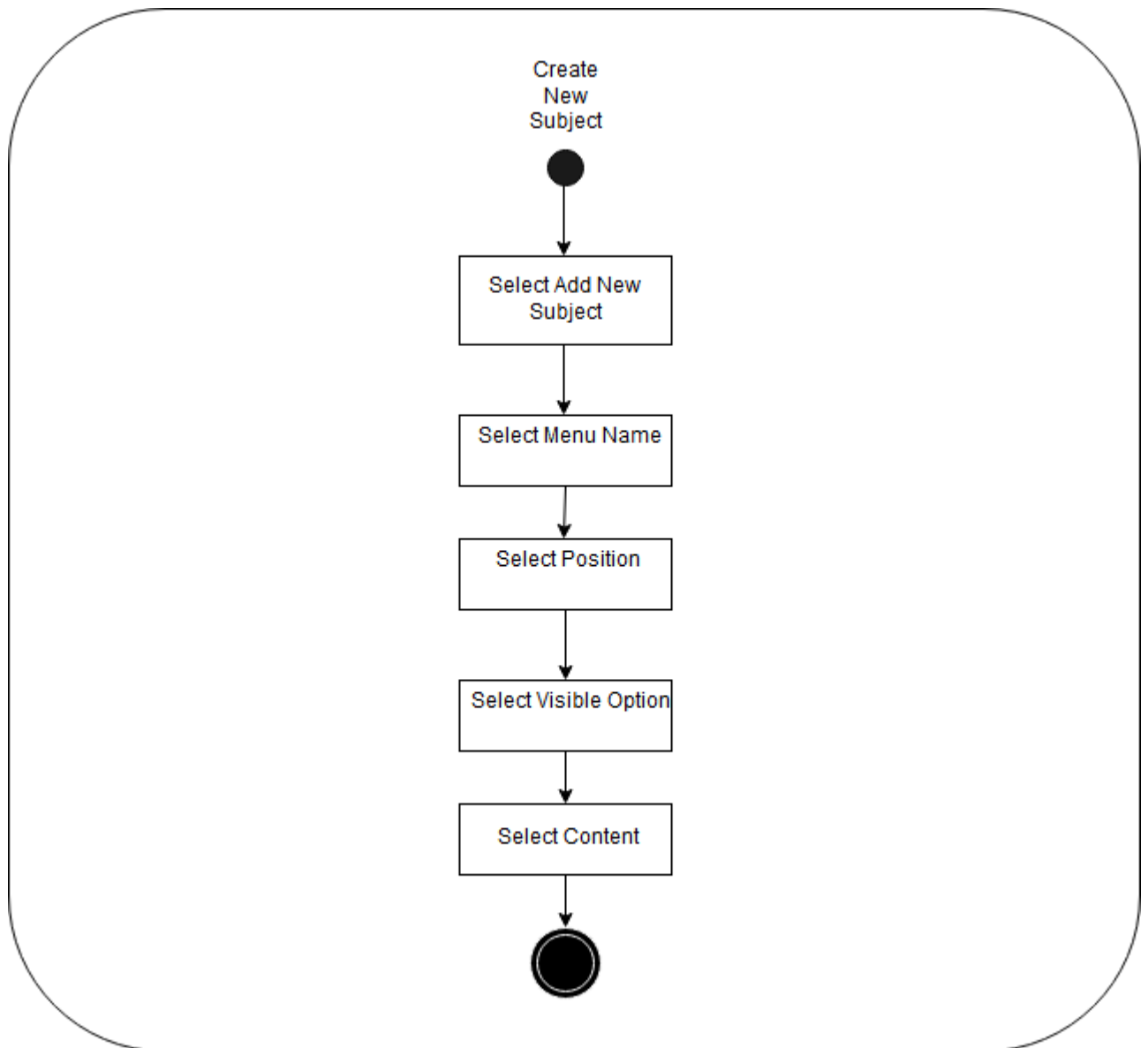


*Edit Admin*

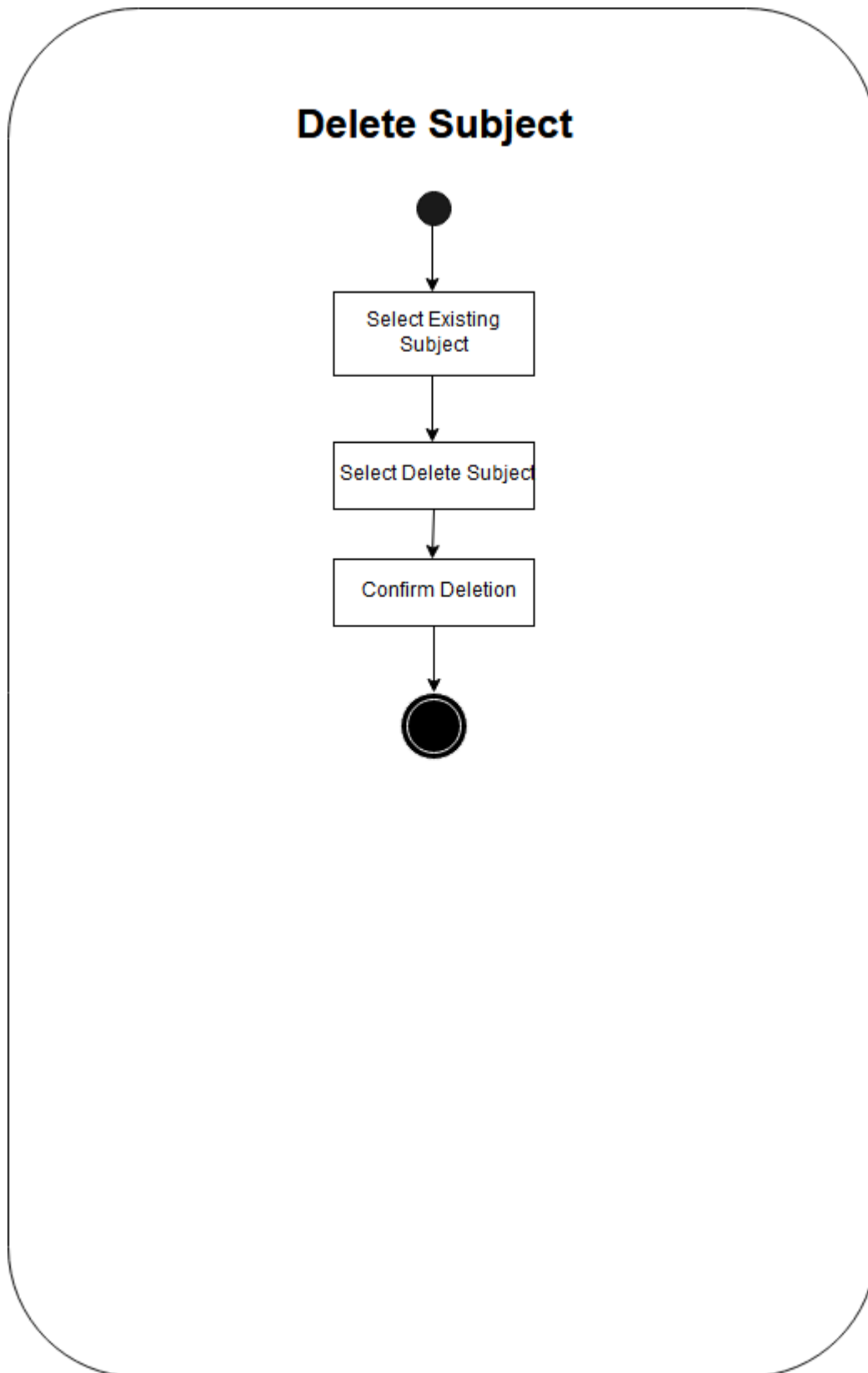
## Edit Admin



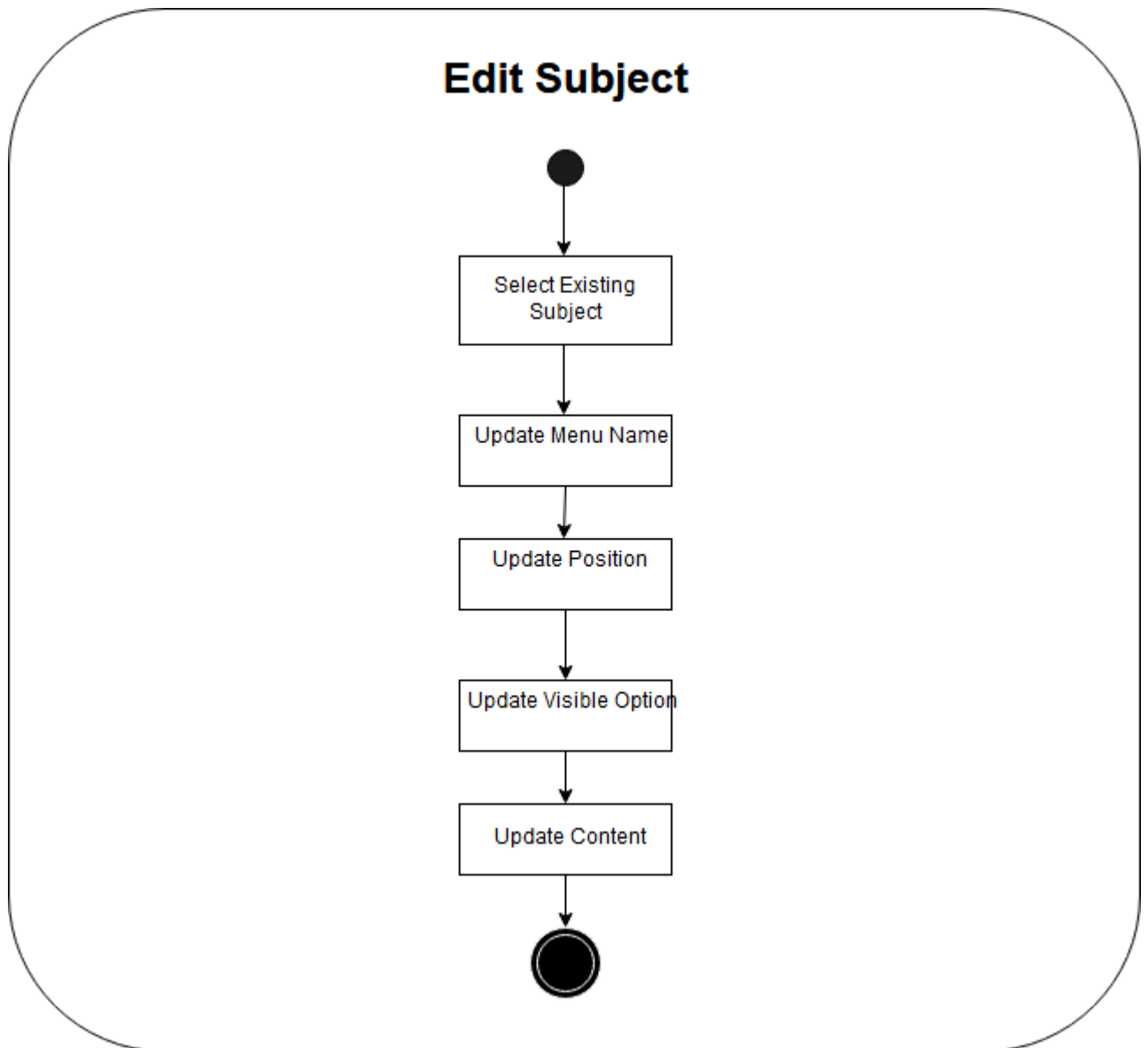
*Create Subject*



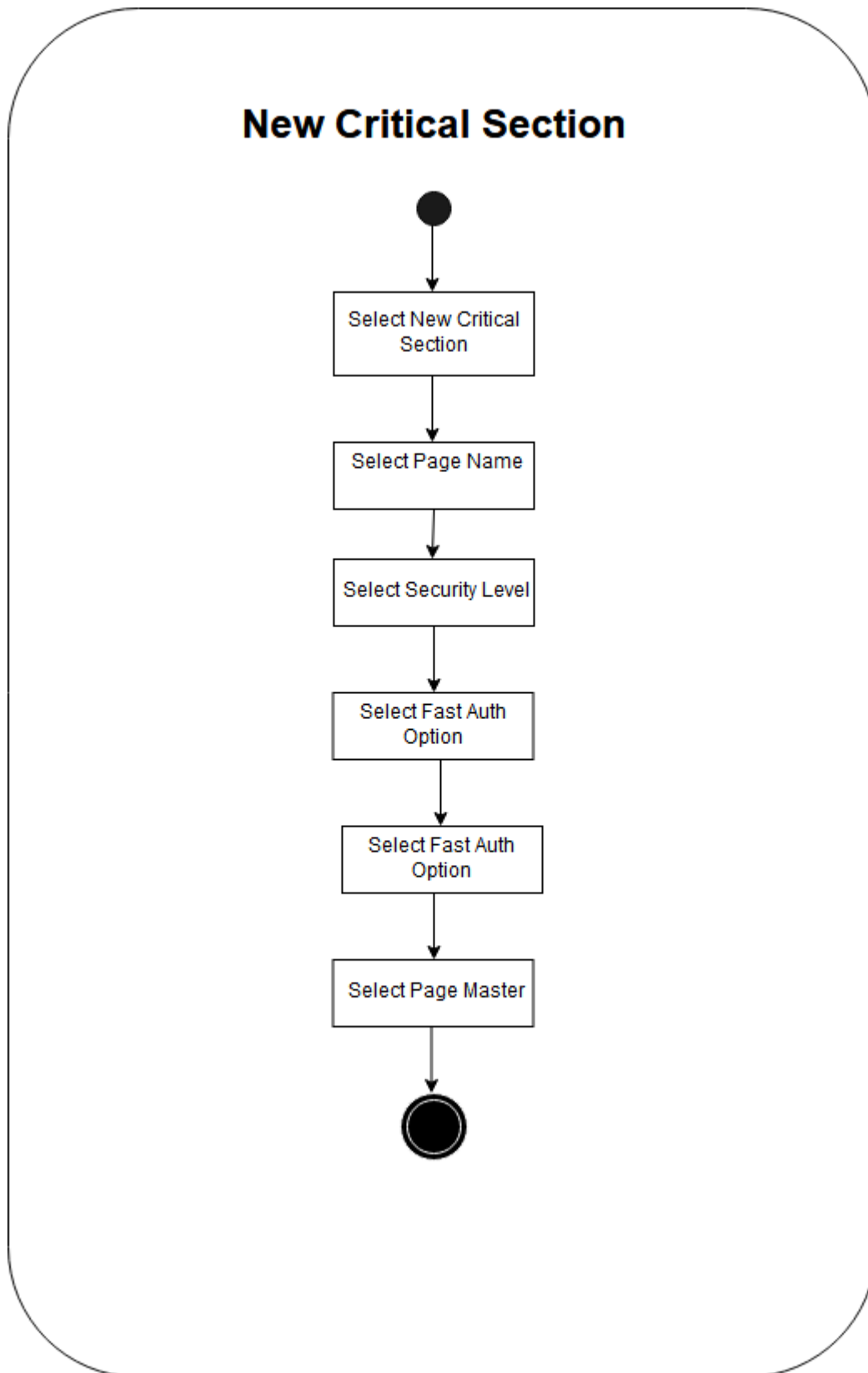
*Delete Subject*



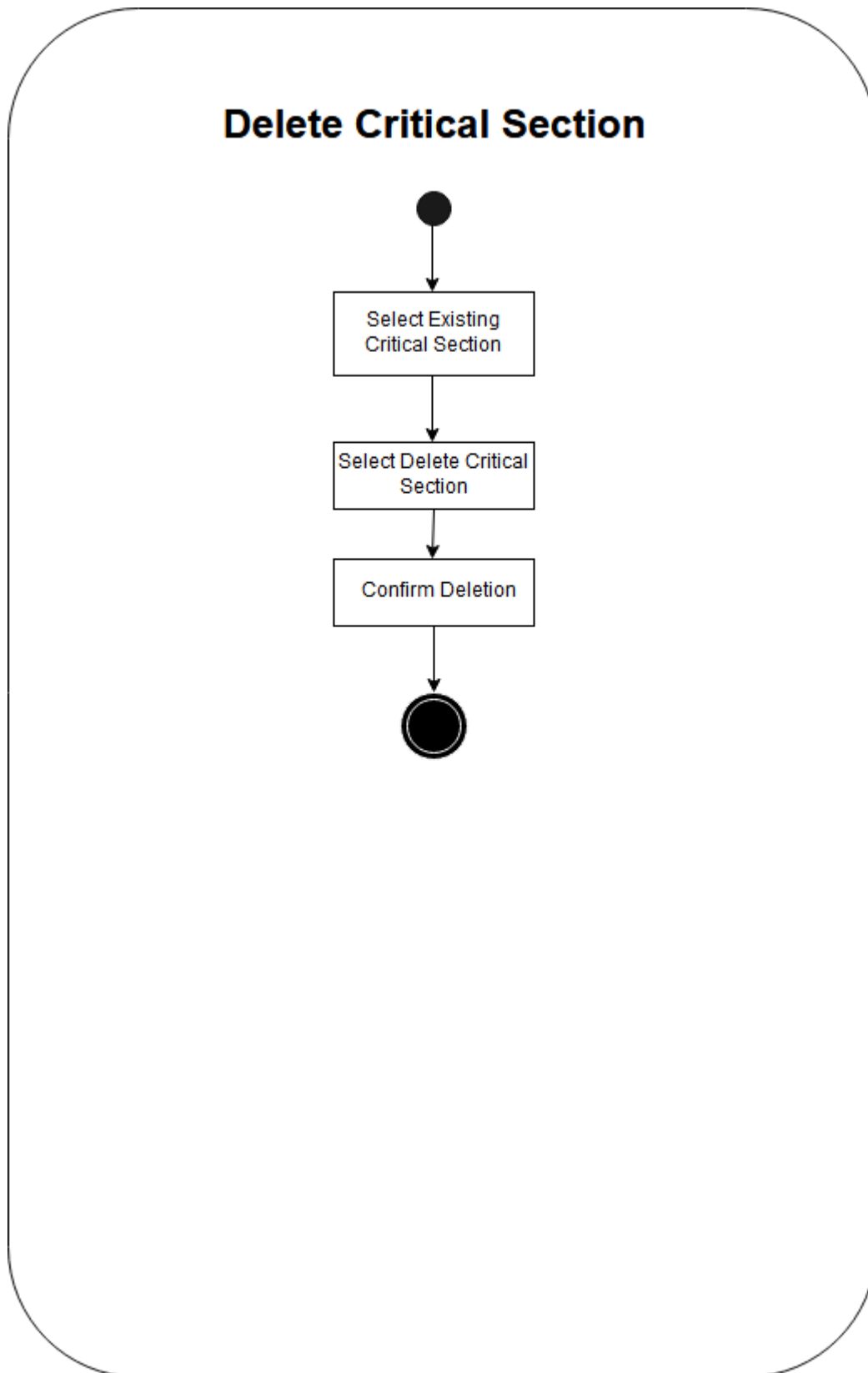
## Edit Subject



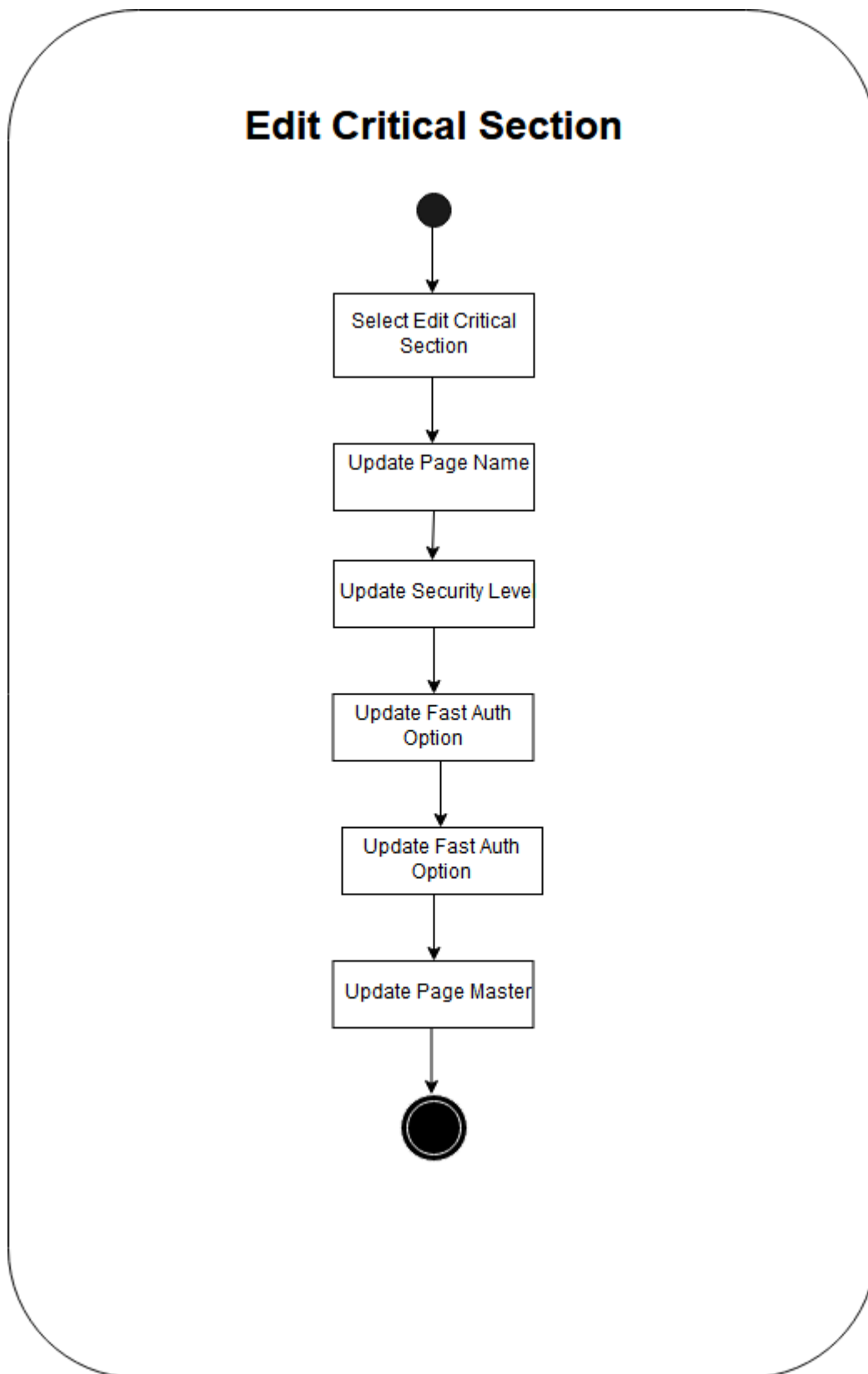
*Create New Critical Section*



*Delete Critical Section*



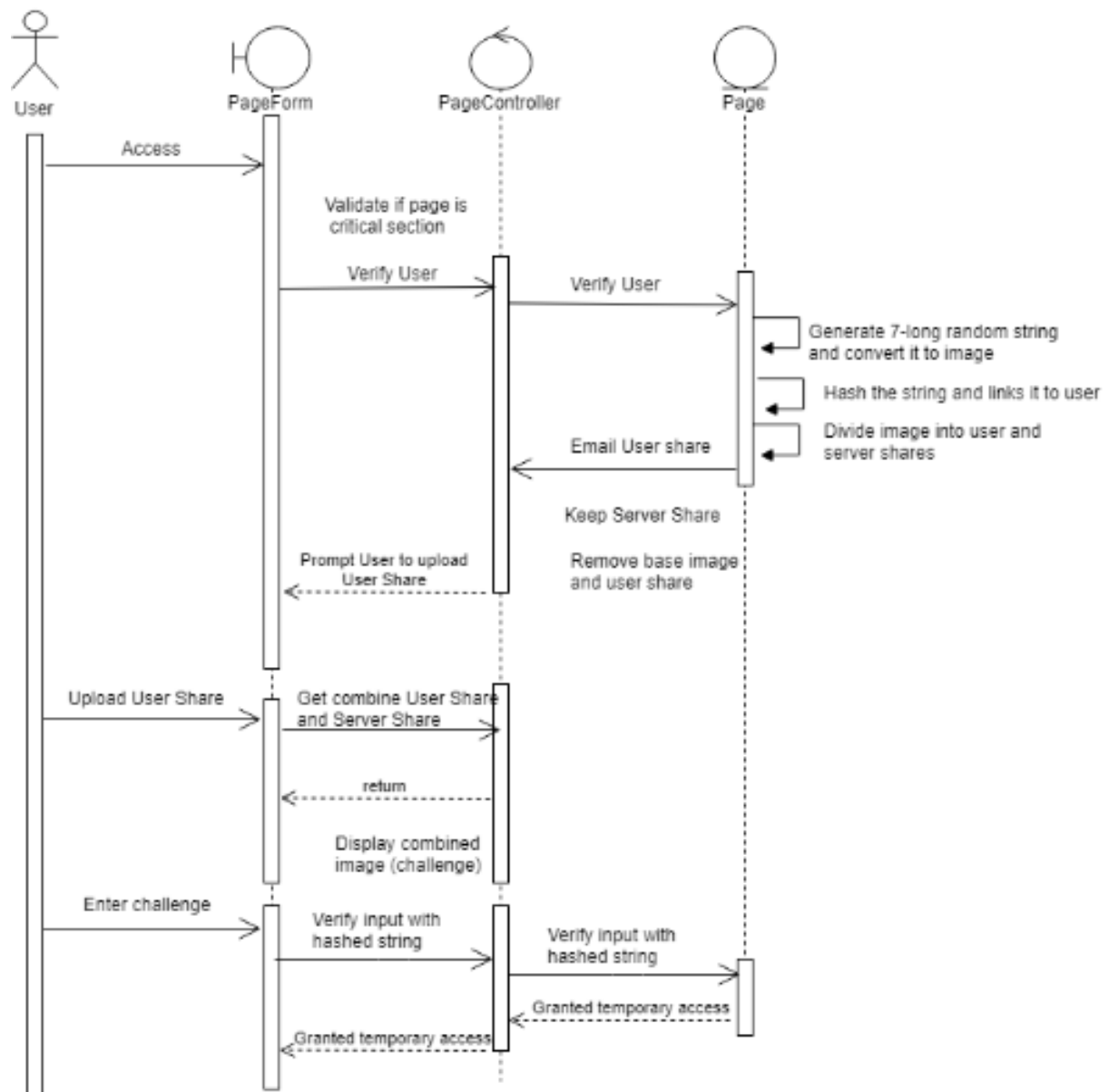
*Edit Critical Section*





## Sequence Diagram

### Iteration 1



## User Interface

### Front End

#### Iteration 1

## Visual Cryptography Project

- About Visual Crypto
- Our Members
- Misc
- Others
- dscfsvdsc

### About Visual Crypto

This is about visual crypto.

The best way to visualize the visual cryptographic scheme is to consider a concrete example. At the end of this extended abstract we enclose two random looking dot patterns. To decrypt the secret message, the reader should photocopy each pattern on a separate transparency, align them carefully, and project the result with an overhead projector.

This basic model can be extended into a visual variant of the  $k$  out of  $n$  secret sharing problem: Given a written message, we would like to generate  $n$  transparencies so that the original message is visible if any  $k$  (or more) of them are stacked together, but totally invisible if fewer than  $k$  transparencies are stacked together (or analysed by any other method). The original encryption problem can be considered as a 2 out of 2 secret sharing problem.

The main results of this paper (besides introducing this new paradigm of cryptographic schemes) include practical implementations of a  $k$  out of  $n$  visual secret sharing scheme for small values of  $k$  and  $n$ , as well as efficient asymptotic constructions which can be proven optimal within certain classes of schemes.

Copyright 2019, FYP

#### Iteration 2

Visual Cryptography

[About Visual Crypto](#)
[Our Members](#)
[Misc](#)
[dscfsvdsc](#)

# About Visual Crypto

This is about visual crypto.

The best way to visualize the visual cryptographic scheme is to consider a concrete example. At the end of this extended abstract we enclose two random looking dot patterns. To decrypt the secret message, the reader should photocopy each pattern on a separate transparency, align them carefully, and project the result with an overhead projector.

This basic model can be extended into a visual variant of the  $k$  out of  $n$  secret sharing problem: Given a written message, we would like to generate  $n$  transparencies so that the original message is visible if any  $k$  (or more) of them are stacked together, but totally invisible if fewer than  $k$  transparencies are stacked together (or analysed by any other method). The original encryption problem can be considered as a 2 out of 2 secret sharing problem.

The main results of this paper (besides introducing this new paradigm of cryptographic schemes) include practical implementations of a  $k$  out of  $n$  visual secret sharing scheme for small values of  $k$  and  $n$ , as well as efficient asymptotic constructions which can be proven optimal within certain classes of schemes.

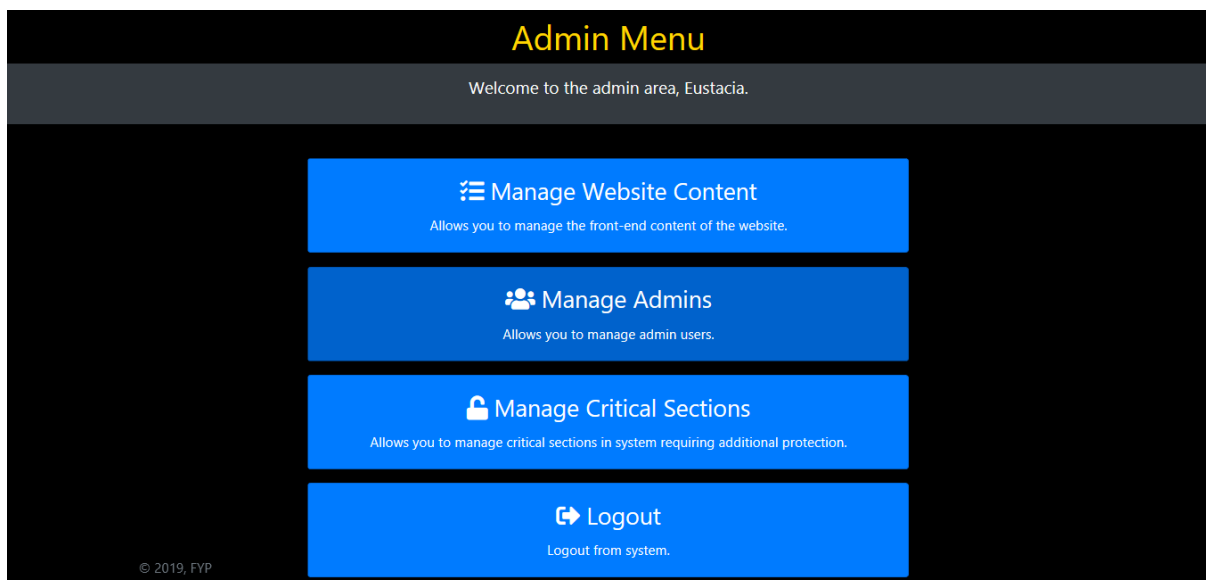
© 2019, FYP

## Admin Menu

### Iteration 1



### Iteration 2

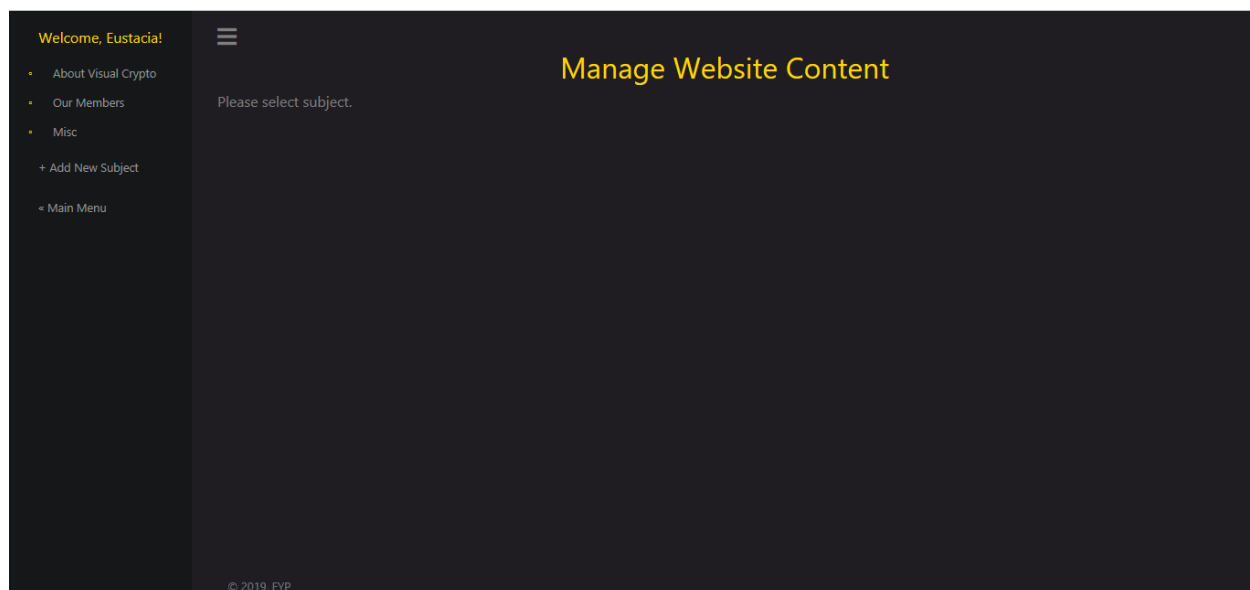


## Manage Website Content

### Iteration 1



### Iteration 2



## Manage Subject

### Iteration 2

Welcome, Eustacia!

- About Visual Crypto
- Our Members
- Misc
- + Add New Subject
- < Main Menu

## Manage Subjects

**Menu Name:** About Visual Crypto

**Position:** 1

**Visible:** Yes

**Content:** This is about visual crypto. The best way to visualize the visual cryptographic scheme is to consider a concrete example. At the end of this extended abstract we enclose two random looking dot patterns. To decrypt the secret message, the reader should photocopy each pattern on a separate transparency, align them carefully, and project the result with an overhead projector. This basic model can be extended into a visual variant of the  $k$  out of  $n$  secret sharing problem: Given a written message, we would like to generate  $n$  transparencies so that the original message is visible if any  $k$  (or more) of them are stacked together, but totally invisible if fewer than  $k$  transparencies are stacked together (or analysed by any other method). The original encryption problem can be considered as a 2 out of 2 secret sharing problem. The main results of this paper (besides introducing this new paradigm of cryptographic schemes) include practical implementations of a  $k$  out of  $n$  visual secret sharing scheme for small values of  $k$  and  $n$ , as well as efficient asymptotic constructions which can be proven optimal within certain classes of schemes.

[Edit Subject](#)

© 2019, FVP

## Edit Subject

### Iteration 1



**Visual Cryptography Project**

- About Visual Crypto
  - General Idea
  - Existing Algorithms
- Our Members**
- Misc
- Others
  - New Page

### Edit Subject: Our Members

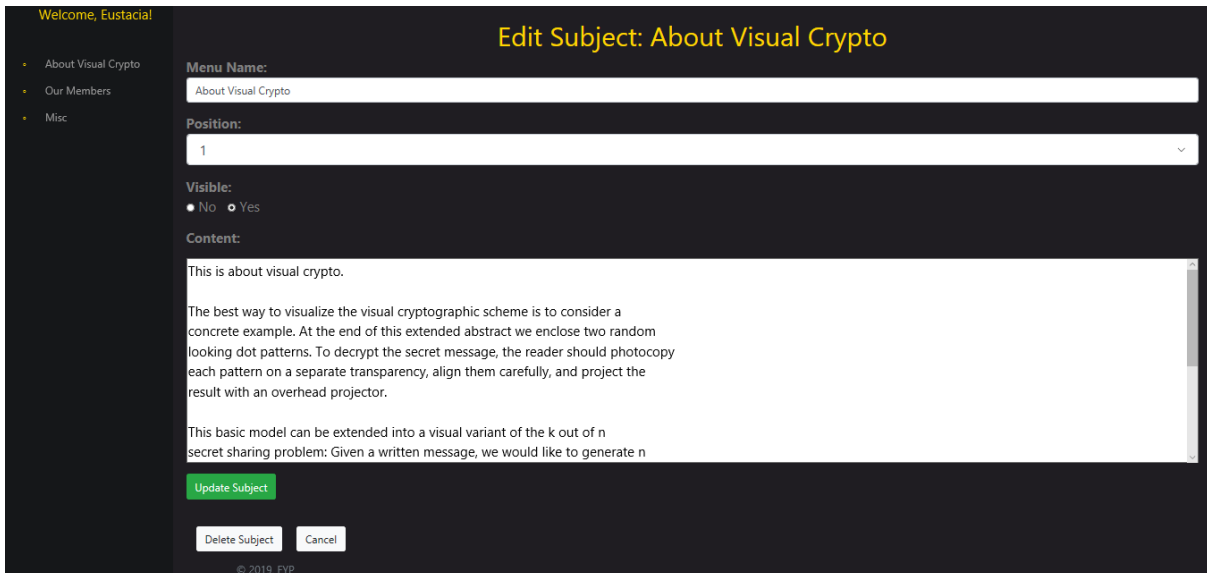
Menu Name:

Position:

Visible: ☐ No ☒ Yes

[Cancel](#) [Delete Subject](#)

### Iteration 2



Welcome, Eustacial

- About Visual Crypto
- Our Members
- Misc

### Edit Subject: About Visual Crypto

Menu Name:

Position:

Visible: ☒ No ☐ Yes

Content:

This is about visual crypto.

The best way to visualize the visual cryptographic scheme is to consider a concrete example. At the end of this extended abstract we enclose two random looking dot patterns. To decrypt the secret message, the reader should photocopy each pattern on a separate transparency, align them carefully, and project the result with an overhead projector.

This basic model can be extended into a visual variant of the k out of n secret sharing problem: Given a written message, we would like to generate n

© 2019, FYP

## Create Subject

### Iteration 1

**Visual Cryptography Project**

**Create Subject**

Menu Name:

Position:

Visible: ☐ No ☐ Yes

[Cancel](#)

### Iteration 2

**Create Subject**

Menu Name:

Position:

Visible: ☐ No ☐ Yes

Content:

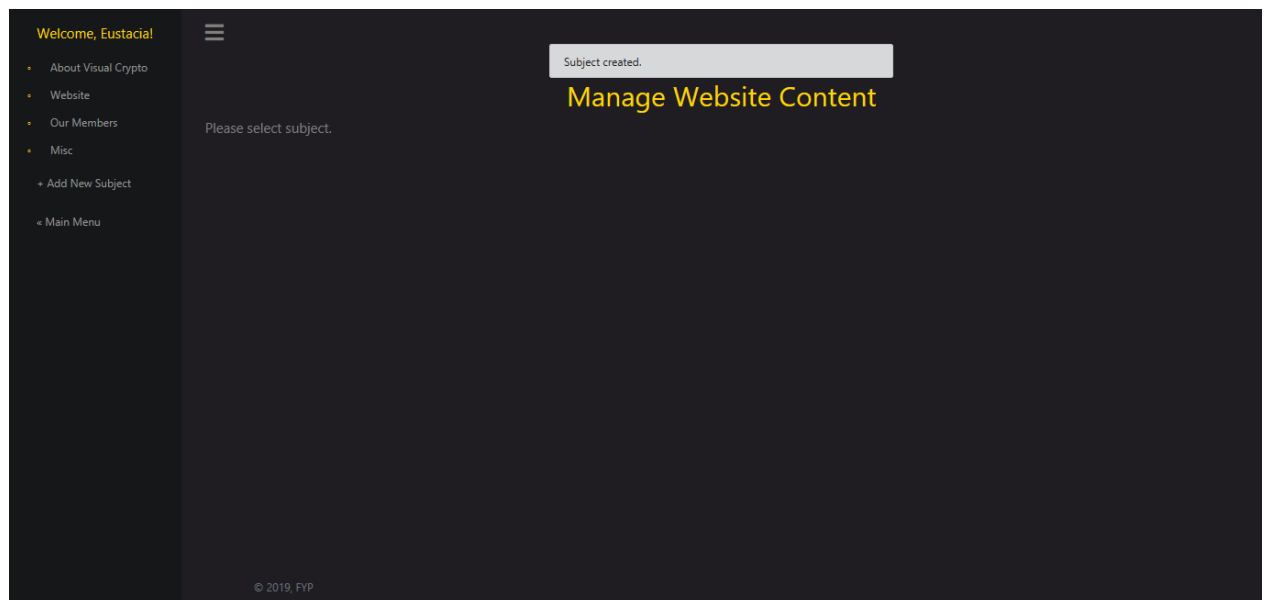
© 2019, FYP

*Subject Created*

Iteration 1



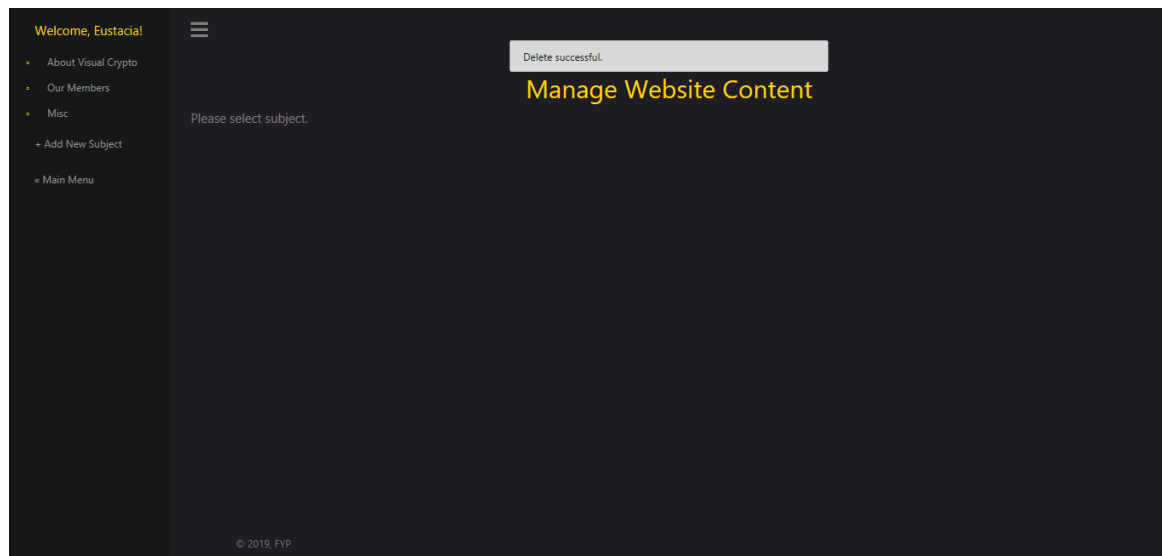
Iteration 2





## *Subject Deleted*

### Iteration 2



## Manage Admins

### Iteration 2

Main Menu

Manage Admins

Username	Actions	
hidir	<div>Edit</div>	<div>Delete</div>
test	<div>Edit</div>	<div>Delete</div>
eustacia	<div>Edit</div>	<div>Delete</div>

Add

© 2019, FYP

## Create New Admins

### Iteration 2

Admin Menu : Create Admin

Username:

Password:

First Email:

Second Email:

Create Admin

Cancel

© 2019, FYP

### Update Admin's Username & Password

#### Iteration 2

Admin Menu : Update

Username:

hidir

New Password:

Login

Cancel

© 2019, FYP

### Delete Admin

#### Iteration 2

◀ Main Menu

Manage Admins

Delete successful.

Username	Actions
hidir	<div>Edit</div> <div>Delete</div>
eustacia	<div>Edit</div> <div>Delete</div>

Add

© 2019, FYP

## Verification

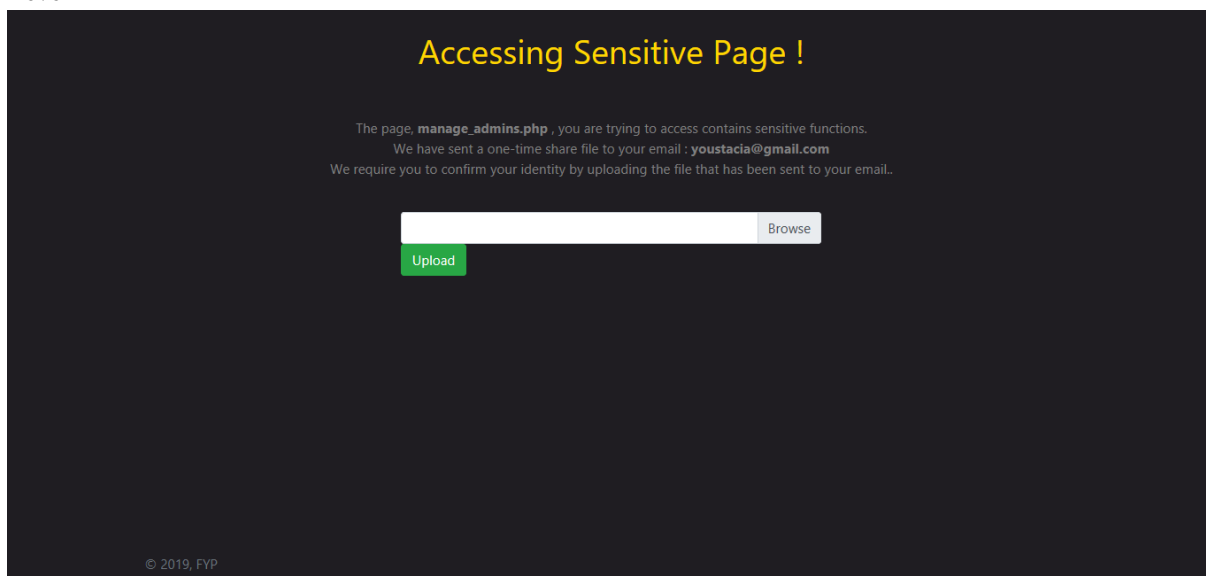
### Stage 1

#### Iteration 1



#### Iteration 2

#### Level 1



## Level 2

### Accessing Sensitive Page !

The page, `manage_admins.php` , you are trying to access contains sensitive functions.

We have sent a one-time share file to both your emails :

File 1 sent to : `youstacia@gmail.com`  
File 2 sent to : `youstacia@gmail.com`

We require you to confirm your identity by uploading **BOTH** files that has been sent to your emails.

File 1 :

File 2 :

© 2019, FYP

## Level 3

### Accessing Sensitive Page !

The page, `manage_admins.php` , you are trying to access contains sensitive functions.

You will require 2 share files to access this page.

File 1 sent to : `youstacia@gmail.com`  
File 2 sent to page master : `dcdda`

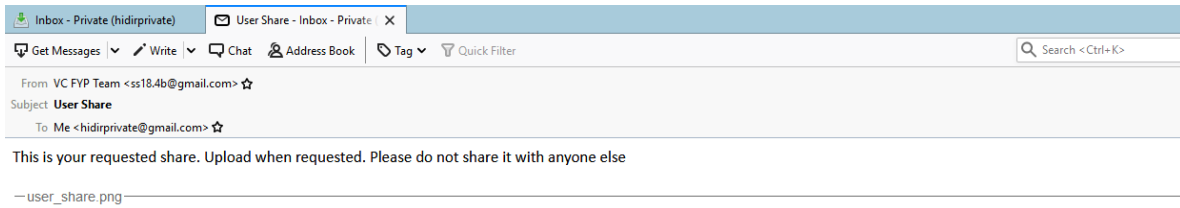
We require you to confirm your identity by uploading **BOTH** files.

File 1 :

File 2 :

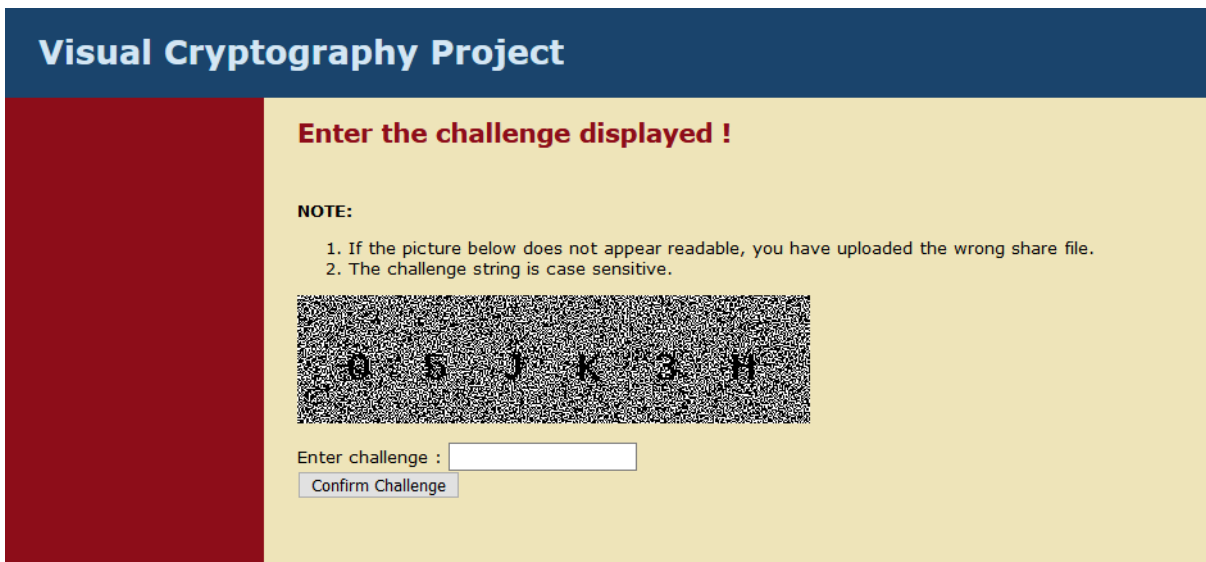
© 2019, FYP

## Verification Email Sent

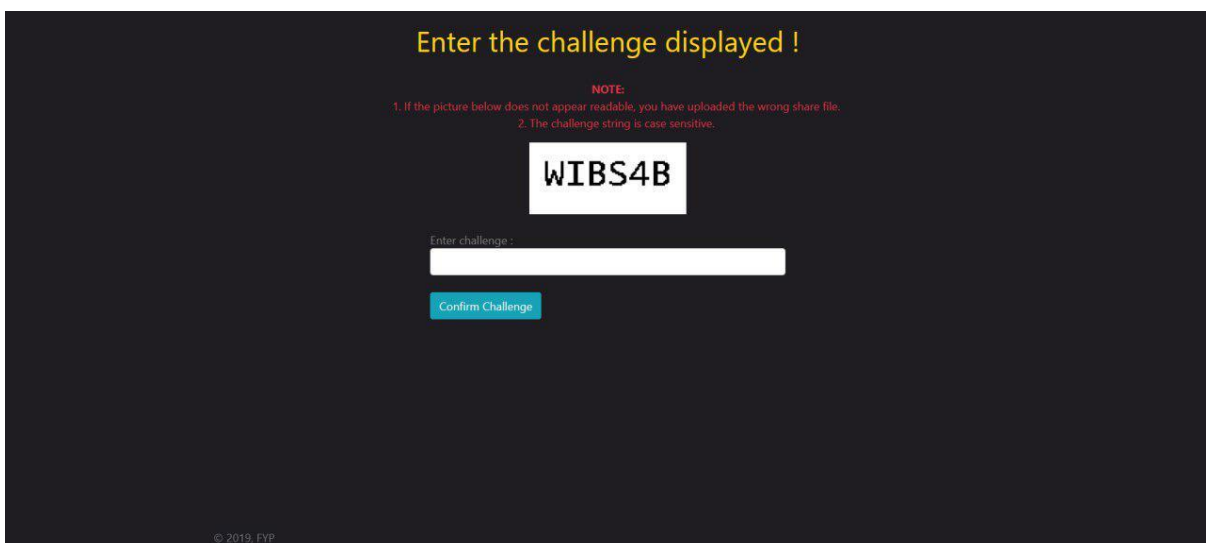


## Stage 2

### Iteration 1

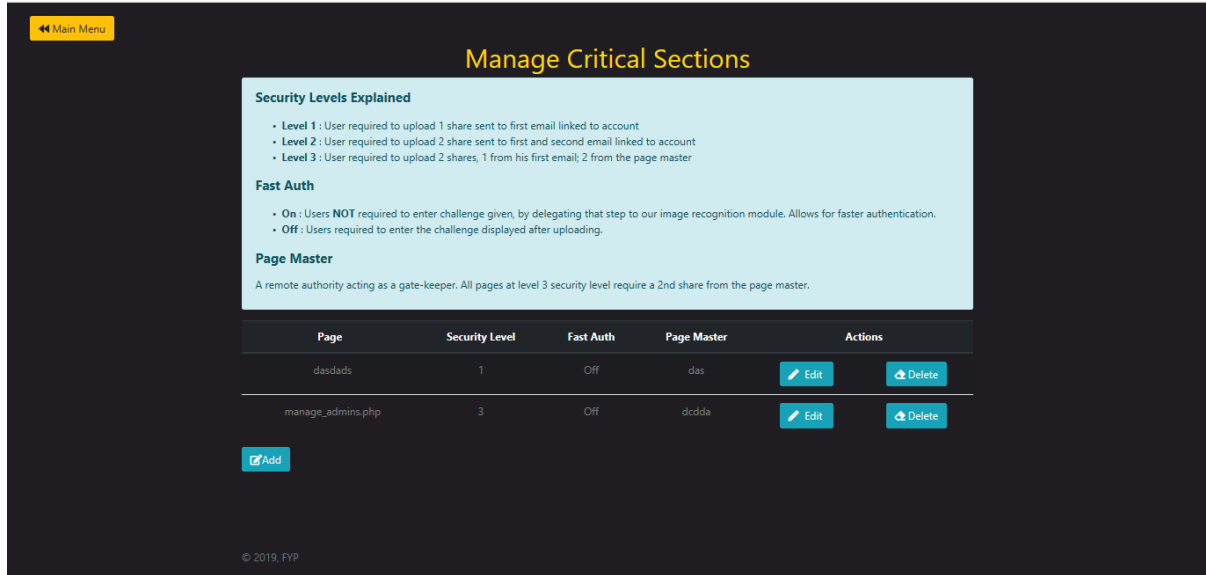


### Iteration 2



## Manage Critical Section

### Iteration 2



## Add New Critical Section

### Iteration 2

Admin Menu : New Critical Section

Page Name:

Security Level:

1

Fast Auth:

Off

Page Master:

Create Critical Section

Cancel

© 2019, FYP

## Update New Critical Section

### Iteration 2

Admin Menu : Update

Page Name:

dasdads

Security Level:

1

Fast Auth:

Off

Page Master:

das

Submit

Cancel

© 2019, FYP

## Delete Critical Section

### Iteration 2

Main Menu

Delete successful.

Manage Critical Sections

Security Levels Explained

- Level 1 : User required to upload 1 share sent to first email linked to account
- Level 2 : User required to upload 2 share sent to first and second email linked to account
- Level 3 : User required to upload 2 shares, 1 from his first email; 2 from the page master

Fast Auth

- On : Users NOT required to enter challenge given, by delegating that step to our image recognition module. Allows for faster authentication.
- Off : Users required to enter the challenge displayed after uploading.

Page Master

A remote authority acting as a gate-keeper. All pages at level 3 security level require a 2nd share from the page master.

Page	Security Level	Fast Auth	Page Master	Actions
<a href="#">Add</a>				

© 2019, FYP



## **Development Methodology**

Rational Unified Process (RUP) is an iterative software development process framework. It is adaptive and not based on a single concrete perspective. RUP is intended to be tailored by software project teams whereby they will select elements of the process that are appropriate to their needs.

There are three disciplines that encompass modelling activities for a single project in the RUP - Business Modelling, Requirements, and Analysis & Design

The development process is divided into four distinct phases

1. Inception

Project idea is stated. The development team determines if the project is worth pursuing and the resources needed.

2. Elaboration

Further evaluation of the project's architecture and required resources. Possible applications of the software and costs associated with the development are taken into consideration by developers.

3. Construction

The project is developed and completed. The software is designed, written, and tested.

4. Transition

The software is released to the public. Final adjustments or updates are made based on feedback from end users.

## Test Cases

We have done some testing on common features. Due to vigorous testing during implementation, we had 18 out of 19 successful test cases, with only 1 test cases. We have successfully rectified the test case that failed.

<b>Test Case ID: 1</b>				
<b>Description:</b> Admin Logging In with Correct Password				
<b>Precondition:</b> Admin not logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User enters username and password. Clicks "Submit"	Page name "login.php"  Username: test  Password: test	User should be able to proceed to admin.php.	User proceeds to admin.php	Pass
<b>Post-conditions:</b>				

<b>Test Case ID: 2</b>				
<b>Description:</b> Admin Logging In with Wrong Password				
<b>Precondition:</b> Admin not logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User enters username and password. Clicks "Submit"	Page name "login.php"  Username: test  Password: test	User should be given message of incorrect password and login failed.	User given message of incorrect password and failed.	Pass
<b>Post-conditions:</b> User will be asked to re-enter login details (____ attempts)				

<b>Test Case ID:</b> 3				
<b>Description:</b> Admin Logging Out				
<b>Precondition:</b> Admin currently logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Click “Log Out”	Page name “logout.php”	Admin should get logged out and no longer have access to admin.php	Admin logged out and no longer have access to admin.php	Pass
<b>Post-conditions:</b> Admin no longer has access to admin pages.				

<b>Test Case ID:</b> 4.1				
<b>Description:</b> Adding New Subject				
<b>Precondition:</b> Admin already logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User enters menu name, position, visibility and content  Clicks “Create Subject”	Page name “Test Page”  Position 5  Visibility On  Content “Hello 1 2 3”	New subject should be created	Subject failed. Returning exception error	Fail
<b>Post-conditions:</b> New subject now available to all users				

<b>Test Case ID:</b> 4.2				
<b>Description:</b> Adding New Subject				
<b>Precondition:</b> Admin already logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User enters menu name, position, visibility and content  Clicks “Create Subject”	Page name “Test Page”  Position 5  Visibility On  Content “Hello 1 2 3”	New subject should be created	Subject created successfully. Previous failed because forgot to include validation_functions.php at the top of new_subject.php	Pass
<b>Post-conditions:</b> New subject now available to all users				

<b>Test Case ID:</b> 5				
<b>Description:</b> Editing Existing Subject				
<b>Precondition:</b> Admin already logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User change menu name from “Test Page” to “Tests Page 2”, position from 5 to 4, visibility to off and content from “Hello 1 2 3” to “Hello 1 2 3 4”.  Clicks “Edit Subject”	Page name “Test Page”  Position 4  Visibility Off  Content “Hello 1 2 3 4”	All values of “Test Page” should change accordingly to Test Data Values	All values of “Test Page” successfully changed accordingly to Test Data Values	Pass
<b>Post-conditions:</b> Subject has new values.				

<b>Test Case ID: 6</b>				
<b>Description:</b> Deleting Existing Subject				
<b>Precondition:</b> Admin already logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User picks a subject that he wants to delete.  Click confirmation "Yes"	Page name "Test Page"	Test page should be deleted from database and list of subjects.	Test page deleted from database and list of subjects	Pass
<b>Post-conditions:</b> Subject no longer shown.				

<b>Test Case ID: 7</b>				
<b>Description:</b> Create admin				
<b>Precondition:</b> Admin already logged in; Admin must have necessary privileges				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Admin enters username, password, first email, second email  Clicks "Create Admin"	Page name "new_admin.php"  Username "test"  Password "test123"  Email <a href="mailto:thistest@gmail.com">thistest@gmail.com</a>  Second email: <a href="mailto:thistest2@gmail.com">thistest2@gmail.com</a>	New admin should be created and stored in the database	New admin created successfully, and admin details stored in the database	Pass
<b>Post-conditions:</b> User will need to ask Admin if modification of details is required				

<b>Test Case ID:</b> 8				
<b>Description:</b> Edit Admin				
<b>Precondition:</b> Admin already logged in; Admin has passed VC Auth				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Admin change value of username and password of existing user.	Username from “test” change to “test2”  Password from “test123” to “test1234”	Username should now be shown as test2 instead of test.	Username shown as test2.	Pass
<b>Post-conditions:</b> Re login as that user to see if can [ PASS]				

<b>Test Case ID:</b> 9				
<b>Description:</b> Delete Admin				
<b>Precondition:</b> User currently logged in; must me an Admin				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Delete an existing user using the Delete button.  Click Yes for Confirmation	User test2	Existing admin should be deleted	Admin is deleted successfully	Pass
<b>Post-conditions:</b> Former Admin no longer has any access to sensitive pages				

<b>Test Case ID:</b> 10				
<b>Description:</b> Admin Adds New Critical Section				
<b>Precondition:</b> Admin already logged in				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
User enters name of page to secure, level of security and whether to enable fast auth. Clicks "Submit"	Page name "test.php"  Level of Security 1  Fast Auth Off	Page will be created with level 1 security and fast auth off	Page created with level 1 security a fast auth off.	Pass
<b>Post-conditions:</b> User will be redirected to manage critical section page.				

<b>Test Case ID:</b> 11				
<b>Description:</b> Edit Critical Section				
<b>Precondition:</b> User already logged in; User must have necessary privileges				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Admin enters page name, security level Clicks "Submit"	Page name "edit_critical_section.php"	Security level of pages should be modified, providing more security to pages with higher security levels	Security level of pages are modified correctly	Pass
<b>Post-conditions:</b> Nil				

<b>Test Case ID:</b> 12				
<b>Description:</b> Delete Critical Section				
<b>Precondition:</b> User currently logged in; must be an Admin				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Go to “Critical Sections”  Select the subject  Click “Delete”	Page name “manage_critical_section.php”	Existing page should be deleted	Page is deleted successfully	Pass
<b>Post-conditions:</b> Deleted subject is no longer accessible (non-existent)				

<b>Test Case ID:</b> 13				
<b>Description:</b> Accessing Page with Security Level 1 and uploading correct file.				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level1 security.  Upload correct file sent to email  Enter the correct captcha image	Page with level1 security.  File “user_share.png” received in email.	Access will be granted to page	Access granted to page.	Pass
<b>Post-conditions:</b> If logged out, need to re-authenticate				



<b>Test Case ID:</b> 14				
<b>Description:</b> Accessing Page with Security Level 1 and uploading correct file and wrong captcha				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level 1 security.  Upload file sent to email  Enter the WRONG captcha image	Page with level1 security.  File “user_share.png” received in email.	Access will be denied	Access denied to page.	Pass
<b>Post-conditions:</b>				

<b>Test Case ID:</b> 15				
<b>Description:</b> Accessing Page with Security Level 1 and uploading wrong file				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level1 security.  Upload random file	Page with level 1 security  File “user_share.png” received in email.	Captcha shown will be jibberish, making it impossible to enter right challenge	Challenge image shown jibberish.	Pass
<b>Post-conditions:</b>				

<b>Test Case ID:</b> 16				
<b>Description:</b> Accessing Page with Security Level 2 and uploading correct file				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level 2 security.  Upload correct file sent to email  3. Enter the correct captcha image	Page with level2security.  File “user_share1.png” received in email1.  File “user_share2.png” received in email2.	Captcha shown will be shown with the string, an authorize user after correct challenge. entered	User authorized successfully.	Pass
<b>Post-conditions:</b>				

<b>Test Case ID:</b> 17				
<b>Description:</b> Accessing Page with Security Level 2 and uploading wrong file				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level2 security.  Upload wrong file sent to email  3. Enter whatever since can’t see the challenge	Random file for share1.  File “user_share2.png” received in email2.	Captcha shown will be as jibberish, can’t enter challenge.	User denied access as wrong challenge.	Pass
<b>Post-conditions:</b>				

<b>Test Case ID:</b> 18				
<b>Description:</b> Accessing Page with Security Level 3 and uploading correct file				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level 3 security.  Upload correct file from email1 and from page master.  Enter correct captcha.	File “user_share1.png” received in email1.  File “user_share2.png” received from page master.	Captcha shown will be shown with the string, an authorize user after correct challenge. entered.	User authorized successfully.	Pass
<b>Post-conditions:</b>				

<b>Test Case ID:</b> 19				
<b>Description:</b> Accessing Page with Security Level 3 and uploading wrong file				
<b>Precondition:</b> Currently logged in; Page must be added as critical section				
Test Steps	Test Data	Expected Results	Actual Result	Test Result (Pass/Fail)
Access a page that is listed as critical section with level3 security.  Upload wrong file sent to email  3. Enter whatever since can’t see the challenge	Random file for share1.  File “user_share2.png” received from page master.	Captcha shown will be as gibberish, can’t enter challenge.	User denied access as wrong challenge.	Pass
<b>Post-conditions:</b>				

## **Development Tools**

### ***Main Stack***

#### **PHP 7.2.1**

PHP is an open-sourced, service-sided scripting language commonly used for web-development. As our application requires the use of an external database, HTML will not be enough as it only allows for client-sided execution.

#### **MYSQL 5.7.23**

MYSQL is an open-sourced relational database management system, commonly used in combination with a server-sided scripting language such as PHP or ASP to provide easy interaction between webpage and the database.

#### **APACHE 2.4.35**

Apache is an open-sourced and most widely used webserver software.

#### **SSL**

SSL is a technology that encrypts communications between the end-user and the server. It helps to prevent hacker attacks that are based on eavesdropping. SSL protected sites are commonly identified by having https instead of http in its URL.

#### **Python 3.7.0**

Python is used in our Visual Cryptography functions due to its powerful image manipulation library called Pillow. Its quick and easy processing for our Visual Cryptography requirements combined with easy integration with PHP made it an ideal choice for our PHP website.

#### **OpenSSL 1.1.0**

OpenSSL is a software library for applications that secure communications over computer networks against eavesdropping or need to identify the party at the other end. Using OpenSSL allows us to have secure connection between browser and server using SSL and HTTPS.

### ***External Libraries Used***

#### **PHPMailer**

<https://github.com/PHPMailer/PHPMailer>

Although PHP provides a built-in mail function, it only delivers very basic functionality such as sending an email.

This means additional functions such as attaching attachments had to be coded from scratch.

With PHPMailer, we get a widely-used and supported library that handles many common functionalities such as attaching images to emails without the clutter.

#### **BulletProof**

<https://github.com/samayo/bulletproof>

Bulletproof is a single-class library to upload images in PHP with security. From our research, we found there are many dangers of allowing external users to upload files to our website. Some dangers we found were:

Users attempting to upload files with malicious content

Users attempting to upload files that are not images

Users attempting to upload files with malicious file names

We researched on all precautions we can take to prevent all the above from happening. (more info about precautions in future Possible Attacks Section) and found BulletProof provided all the necessary checks to prevent any of the above attacks we identified.

#### **Tesseract OCR 4.0.0**

<https://github.com/tesseract-ocr/>

Tesseract is an optical character recognition engine for various operating systems. It is free software, released under the Apache License, and development has been sponsored by Google since 2006. In 2006, Tesseract was considered one of the most accurate open-source OCR engines available. We used Tesseract for reading strings after system has combined the 2 shares, allowing for smoother and faster auth.

#### **Pytesseract 0.2.6**

<https://pypi.org/project/pytesseract/>

Pytesseract allow us to access the Tesseract OCR program through the web using Python.

#### **OpenCV 4.0.1**

<https://opencv.org/>

OpenCV is a library of programming functions mainly aimed at real-time computer vision. We used OpenCV for improving image quality before passing it to Tesseract for more accurate readings.

## **2x2 Visual Cryptography Script**

<https://github.com/LessonStudio/VisualCryptography>

This is an old script coded by Robert Donovan that allows splitting of image for Visual Cryptography of a 2x2 scheme. It was not working due to deprecated syntax, but we managed to fix it. We will be rewriting the script as we will be aiming to support 3x3 scheme in the future.