

Introduction to Computer Security – G6077

Weighting: 50% of marks for the module

Version Information: Version 1, 21st Oct 2022

Submission deadline: Check deadline on Sussex direct. e-submission to Canvas

You must work on this assignment on your own. The standard Informatics rules for collusion, plagiarism and lateness apply. Any cases of potential misconduct discovered will be reported and investigated.

Lovejoy's Antique Evaluation Web Application

In this coursework, you will develop a secure web application for a local antique dealer named Lovejoy. Lovejoy wants a minimum viable product allowing customers to register and then request evaluations of potential antique objects. Lovejoy has many rivals in the antique business who may sometimes resort to underhand tactics and so is very concerned about the security of the application.

Your secure web application will need to have these features for the minimum viable product (MVP) release: user registration and login, a password policy, "request evaluation" page and then an extension of the "request evaluation" page file upload to allow upload of photos. Finally, Lovejoy needs a request listing page.

You should build Lovejoy's MVP focusing on the following features in each task. As well as the code, you should submit a report as described in the appendix below, where you will provide a self-reflection on the security and for each feature. Mark allocation for each task are as described below and in the security analysis grid. You should reflect upon your work and provide estimates of how much you've achieved by filling out the grid, which if completed will be allocated 5 marks. There are thus 35 marks for completing the application reasonably, 60 marks for the security features identified and implemented, and 5 marks for self-reflection.

You have a choice of technologies from which to build the application:

-

PHP (host it for free on 000webhost)

-

-

Java,

-

-

Python

-

No other approach is allowed

Task 1 - Develop a secure web form that allows customers to register in the application. They must register an email address, password, name and contact telephone number. The users' details should be stored in a database.	Code Quality 5 marks
	Database Design 5 marks
Task 2 - Develop a secure login feature.	Code Quality 5 marks
Task 3 – Extend the password management feature to provide password strength recommendations and password recovery.	Code Quality 5 marks
Task 4 - Implement a “Request Evaluation” web page only accessible to logged in users. This web page should have a comment box to type in the details of the object and their request, and a dropdown box for preferred method of contact between phone or email.	Code Quality 5 marks

Task 5 – Extend the “Request Evaluation” page to allow for file upload of a photo of the object	Code Quality 5 marks
Task 6 – Implement a page that displays a list of evaluation requests. This page should only be visible to an administrator role	Code Quality 5 marks

Submission guidance

You are only submitting the report to the Canvas. The report template is provided below.
The template

Report – Use the template provided in the Canvas for the report.

Provide screenshots of all

the marking criteria elements and annotate where necessary. Use

bullet points to

give any explanation. Don't write paragraphs.

Report will have

Excellent (10-9 marks)	Good (8-6 marks)	Average (5-3 marks)	Poor (2-0 marks)	riteria
Student must have gone				

beyond				
Policy has no flaw, and its implementation is excellent. Various mechanisms implemented to ensure password policy is secure.	Policy has no flaws, but implementation of policy is simple.	Password policy has very few flaws. However, different sections of policy are implemented and working.	Policy has many flaws for example password is not encrypted, and no salt applied. Password forgot policy has security flaws.	<p>password policy 10marks</p> <p>Password entropy, encrypted storage, security questions and recovery of password</p> <p>Implemented all the required features</p>
Several countermeasures are implemented, and the quality of countermeasures are excellent.	Countermeasures are implemented in all the pages however quality of implementation is simple.	Implemented countermeasures only in some parts of the application.	Very little effort to implement countermeasures to avoid these vulnerabilities.	<p>vulnerabilities 10 marks</p> <p>SQL injection, XSS, CSRF, File Upload and any other obvious vulnerability.</p> <p>Implemented counter-measures for all the listed vulnerabilities</p>
All the requirements are implemented to authenticate the user. However, quality of implementation is simple.	All requirements are implemented to authenticate the user. However, quality of implementation is simple.	Only some obvious requirements are not implemented.	Lots of obvious authentication's requirements are not implemented.	<p>authentication 10 marks</p> <p>User identity management (registration and login etc), Email verification for registration, 2 factor authentications (PIN and or email),</p> <p>Implemented all the required features</p>
Excellent implementation of countermeasures against these attacks.	No flaws in countermeasures however quality of implementation is simple.	Some flaws in countermeasures	Very little effort against these attacks.	<p>obfuscation/Common attacks 10 marks</p> <p>Brute force attack – Number of attempts</p> <p>Botnet attack – Captcha</p> <p>----</p>

				Dictionary attack/Rainbow table attack Implemented Attempts Restriction, Google reCaptcha, and Dictionary Attack Prevention
Implementat ion of other security features has no flaws. No obvious security feature is ignored.	Several security features implemented. Implementation has flaws.	Other security features are implemente d but obvious ones are ignored.	Very little effort to implement some obvious other security features like storage of confidential information .	ther security features like confidentiality of information 10 marks For example, identify information that needs to be stored as encrypted. Implemented this on password, answer of security question, and activation code which sent to user. Think these are all for this project.
Claimed features are complex. Quality of achievement is excellent.	Claimed features are complex however quality of achievement/implem entation could have been better.	Claimed features are somewhat complex and implementat ion could have been better.	Claimed features are not complex and challenging.	eeper understanding, two extra web security 10 marks Carry out your investigation and implement two more security features. These need to be complex and challenging one. 1. Alert user by email when their account is logging-in in another

				ip address. 2. 3. Ban the ip which reaching attempt limit, but not just ban by username. 4.		
5 marks	5 marks	5 marks	5 marks	5 marks	10 marks	
List evaluation- Task6	Request evaluation – task 5	Request evaluation – task 4	Forgot password-Task3	Login-Task2	User registration/Databases-Task1	Features of web application
Completed	Completed	Completed	Completed	Completed	Completed	

Up to 5 marks	0 marks	
Fully completed	Marking not completed	Self-reflection

Report

You will be submitting this report to the Canvas.

1.

Application URL: <https://web594319279.000webhostapp.com/index.php>

2.

1.

Code file Location

2.

One Drive: https://1drv.ms/u/s!Am_YorlOURcthCBJriaFJ71Bqokt?e=gvCaYX

Google Drive:

https://drive.google.com/drive/folders/1YNSOkDZpcQ1UaVKJIYtM6njtNW6YUhxN?usp=share_link

1.

Test users' detail:

2.

Admin

Username: root

Password: Daq2xgrastqm!

Not Admin

Username: user123

Password: 11aaAA!!

Task 0 – Self-reflection

Marking grid filled up by you.

See above.

Task 1 – User registration

Registration feature code screenshots

Register Form

Edit file

/public_html/register_form.php

```
1 <?php
2 session_start();
3 session_unset();
4 $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));
5 ?>
6
7
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <link rel="stylesheet" href="register_style.css" type="text/css">
12 <script src="https://www.google.com/recaptcha/api.js" async defer></script>
13 </head>
14
15 <body>
16
17 <div class="em">
18 <?php if(isset($_GET['error'])): ?>
19 <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
20 <?php endif ?>
21 </div>
22
23 <div class="register">
24 <h1>Sign Up</h1>
25 <form action="register_check.php" method="POST">
26 <label for="username">Username:</label>
27 <input type="text" name="username" placeholder="need 2 to 8 characters, special symbol could be: _" maxlength="8" pattern="^[a-z_]{2,8}$" id="username" required>
28 <label for="password1">Password:</label>
29 <input type="password" name="password1" placeholder="need 8 to 16 characters, special symbol could be: _ ? !" pattern="^[a-zA-Z0-9_?!]{8,16}$" id="password1" required>
30 <label for="password1"><font color=blue>Password needs to have at least 1 Uppercase letter 1 Lowercase letter 1 digit number
31 special chars and should be 8-16 chars in total</font></label><br><br><br>
32 <label for="password2">Type in the password again:</label>
33 <input type="password" name="password2" placeholder="need 8 to 16 characters, special symbol could be: _ ? !" pattern="^[a-zA-Z0-9_?!]{8,16}$" id="password2" required>
34 <label for="fname">Forename:</label>
35 <input type="text" name="fname" placeholder="forename" maxlength="20" id="fname" required>
36 <label for="sname">Surname:</label>
37 <input type="text" name="sname" placeholder="surname" maxlength="20" id="sname" required>
38 <label for="phone">Phone Number:</label>
39 <input type="tel" name="phone" placeholder="phone number" id="phone" required>
40 <label for="email">Email Address:</label>
41 <input type="email" name="email" placeholder="email address" id="email" required>
42 <label for="sec_ques">Security Question:</label>
43 <select name="sec_ques" value="What city were you born in ?" id="sec_ques">
44 <option>What city were you born in ?</option>
45 <option>What food is your favourite ?</option>
46 <option>What movie do you like most ?</option>
47 </select><br><br>
48 <label for="sec_ans">Enter you answer here (case-sensitive):</label>
49 <input type="text" name="sec_ans" id="sec_ans" maxlength="20" required>
50 <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
51 <div class="g-recaptcha" data-sitekey="6LdpU0jAAAAAD1we4a0ZnBRfSLCrTHMGR1mpbt"></div>
52 <input type="submit" value="Go to email verification" id="submit">
53 </form>
54 </div>
55
56 <input type="hidden" name="token_generate" id="token_generate">
57
58 <div class="toLog">
59 <?php
60 echo "Have an account already? Back to <a href='login_form.php'> LOGIN </a>";
61 ?>
62 </div>
63
64 </body>
65 </html>
66
67
68
69
70
71
72
```

Register Check

/public_html/register_check.php

```

1 <?php
2 session_start();
3
4 require 'helper.php';
5
6 csrf_counter();
7 g_captcha_check("register_form.php");
8
9
10 // check if all data exists
11 if (!isset($_POST['username'], $_POST['password1'], $_POST['password2'], $_POST['fname'], $_POST['sname'], $_POST['phone'], $_POST['email'], $_POST['sec_ques'],
12     $_POST['sec_ans'])) {
13     header_n_sendem("register_form.php", "Please complete the registration form");
14 }
15 // check if all data are not empty
16 if (empty($_POST['username']) || empty($_POST['password1']) || empty($_POST['password2']) || empty($_POST['fname']) || empty($_POST['sname']) || empty($_POST['phone']) ||
17     empty($_POST['email']) || empty($_POST['sec_ques']) || empty($_POST['sec_ans'])) {
18     header_n_sendem("register_form.php", "Please complete the registration form");
19 }
20
21 // username
22 $username = htmlspecialchars($_POST['username']);
23 if ($username != $_POST['username']) {
24     header_n_sendem("register_form.php", "Username contains invalid symbol");
25 }
26 if (preg_match('/^[w_]{2,8}$/', $username) == 0) {
27     header_n_sendem("register_form.php", "Username is in wrong pattern");
28 }
29
30 // password
31 $password1 = htmlspecialchars($_POST['password1']);
32 $password2 = htmlspecialchars($_POST['password2']);
33 if (($password1 != $_POST['password1']) || ($password2 != $_POST['password2'])) {
34     header_n_sendem("register_form.php", "Password contain invalid symbol");
35 }
36 if ($password1 != $password2) {
37     header_n_sendem("register_form.php", "Two passwords are different");
38 }
39 // PASSWORD PATTERN
40 if ((strlen($password1) < 8) || strlen($password1) > 16 ||
41     !preg_match('@[A-Z]@', $password1) || !preg_match('@[a-z]@', $password1) ||
42     !preg_match('@[0-9]@', $password1) || !preg_match('@[_?!]@', $password1)) {
43     header_n_sendem("register_form.php", "password in wrong pattern or is not strong enough");
44 }
45 // PREVENT DICTIONARY ATTACK
46 $handle = fopen('./common_password/com_pwd.txt', 'r');
47 if ($handle) {
48     while (false !== ($com_pwd = fgets($handle, 256))) {
49         if ($password1 == $com_pwd) {
50             header_n_sendem("register_form.php", "Password is vulnerable to Dictionary Attacks");
51         }
52     }
53 } else {
54     header_n_sendem("register_form.php", "Failed to open the pwd TXT");
55 }
56 fclose($handle);
57
58 // forename & surname
59 $forename = htmlspecialchars($_POST['fname']);
60 $surname = htmlspecialchars($_POST['sname']);
61 if ($forename != $_POST['fname']) {
62     header_n_sendem("register_form.php", "Forename contains invalid symbol");
63 }
64 if ($surname != $_POST['sname']) {
65     header_n_sendem("register_form.php", "Surname contains invalid symbol");
66 }
67
68 // phone
69 $phone = $_POST['phone'];
70 if ((strlen($phone) >= 10) && (strlen($phone) <= 14)) {
71     $phone = preg_replace("/[0-9]/", '', $phone);
72     if ($phone != $_POST['phone']) {
73         header_n_sendem("register_form.php", "Invalid phone number");
74     }
75 } else {
76     header_n_sendem("register_form.php", "Unsupported phone number length");
77 }
78
79 // email
80 $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
81 if (filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
82     header_n_sendem("register_form.php", "Invalid email address");
83 }
84
85 // security Q&A
86 $sec_ques = $_POST['sec_ques'];
87 $sec_ans = htmlspecialchars($_POST['sec_ans']);
88 if ($sec_ans != $_POST['sec_ans']) {
89     header_n_sendem("register_form.php", "Security answer contains invalid symbol");
90 }
91
92
93
94
95 // =====
96 require __DIR__ . '/vendor/autoload.php';
97 use PragmaRX\Google2FA\Google2FA;
98 require 'database_conn.php';
99
100

```

```

94
95 // =====
96 require __DIR__ . '/vendor/autoload.php';
97 use PragmaRX\Google2FA\Google2FA;
98 require 'database_conn.php';
99
100
101
102 // get stmt1 if that username exists.
103 if ($stmt1 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ?')) {
104     $stmt1->bind_param('s', $username);
105     $stmt1->execute();
106     $stmt1->store_result();
107 } else {
108     header_n_sendem("register_form.php", "Cannot prepare statement 1 !");
109 }
110
111 // get stmt2 if that email exists.
112 if ($stmt2 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ? AND email = ?')) {
113     $stmt2->bind_param('ss', $username, $email);
114     $stmt2->execute();
115     $stmt2->store_result();
116 } else {
117     header_n_sendem("register_form.php", "Cannot prepare statement 2 !");
118 }
119
120 // check and insert a new row
121 if ($stmt1->num_rows > 0) {
122     header_n_sendem("register_form.php", "Username exists, please choose another one");
123 }
124 elseif ($stmt2->num_rows > 0) {
125     header_n_sendem("register_form.php", "Email has already been used, please choose another one");
126 } else {
127     // Username and Email both dont exist, now can insert new systemuser
128     if ($stmt1 = $conn->prepare('INSERT INTO systemuser (username, passwd, forename, surname, phone, email, activation_code, sec_ques, sec_ans, tfa_seckey, first_ip)
129     VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)')) {
130
131         // set the params needed
132         $password_hashed = password_hash($password1, PASSWORD_DEFAULT);
133         $uniqid = uniqid();
134         $sec_ans = password_hash($sec_ans, PASSWORD_DEFAULT);
135         $google2fa = new Google2FA();
136         $secret = $google2fa->generateSecretKey();
137         $first_ip = getUserIpAddr();
138
139         // pass the params and execute
140         $stmt1->bind_param('sssssssss', $username, $password_hashed, $forename, $surname, $phone, $email, $uniqid, $sec_ques, $sec_ans, $secret, $first_ip);
141         $stmt1->execute();
142
143         // sending authentication email
144         $uniqid_hashed = password_hash($uniqid, PASSWORD_DEFAULT);
145         $subject = 'Account Activation Required';
146         $activate_link = 'http://web594319279.000webhostapp.com/activate.php?email=' . $email . '&code=' . $uniqid_hashed;
147         // $activate_link = 'http://localhost/lovejoy/activate.php?email=' . $email . '&code=' . $uniqid_hashed;
148         $body = '<p>Please click the following link to activate your account: <br><br>' . $activate_link . '</p>';
149         // $body = '<p>Please click the following link to activate your account: <a href="" . $activate_link . "">' . $activate_link . '</a></p>';
150         send_activation_email($email, $username, $subject, $body);
151
152     } else {
153         header_n_sendem("register_form.php", "Cannot prepare stmt of INSERT info !");
154     }
155 }
156
157
158 $stmt2->close();
159 $conn->close();
160 ?>
161

```

SAVE & CLOSE

SAVE

register_style.css

1.2 kB

2022-12-05 17:11:00

Database Table

Three tables in total

Server: localhost:3306 » Database: id19724952_socnet

Structure SQL Search Query Export Import Operations Routines Events More

Filters

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> login_attempt		2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> request		9	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> systemuser		4	MyISAM	utf8mb4_general_ci	3.0 KiB	-
3 tables	Sum	15	InnoDB	utf8_unicode_ci	35.0 KiB	0 B

☐ Check all With selected:

Print Data dictionary

Create table

Name: Number of columns:

Go

Table: systemuser

Server: localhost:3306 » Database: id19724952_socnet » Table: systemuser

Browse Structure SQL Search Insert Export Import Operations Triggers

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	
<input type="checkbox"/> 2	username	varchar(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 3	passwd	varchar(255)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 4	forename	varchar(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 5	surname	varchar(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 6	phone	char(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 7	email	varchar(255)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 8	is_admin	int(1)			No	0			
<input type="checkbox"/> 9	activation_code	varchar(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 10	is_activated	char(1)	utf8mb4_general_ci		No	0			
<input type="checkbox"/> 11	sec_ques	varchar(255)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 12	sec_ans	varchar(255)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 13	tfa_seckey	varchar(255)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 14	last_login	int(11)			No	0			
<input type="checkbox"/> 15	first_ip	varchar(16)	utf8mb4_general_ci		No	None			

☐ Check all With selected:

Table: request

Server: localhost:3306 » Database: id19724952_socnet » Table: request

Browse Structure SQL Search Insert Export Import Operations Triggers

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id 🔑	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/> 2	image_url	text	utf8mb4_general_ci		No				Change Drop More
<input type="checkbox"/> 3	contact	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 4	detail	text	utf8mb4_general_ci		No				Change Drop More
<input type="checkbox"/> 5	uploader	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More

☐ Check all
 With selected: Browse Change Drop Primary Unique Index Fulltext

Table: login_attempt

Server: localhost:3306 » Database: id19724952_socnet » Table: login_attempt

Browse Structure SQL Search Insert Export Import Operations Triggers

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id 🔑	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/> 2	username	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	ip_address	varchar(16)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 4	attempt	int(11)			No	0			Change Drop More
<input type="checkbox"/> 5	banned_time	int(11)			No	0			Change Drop More

☐ Check all
 With selected: Browse Change Drop Primary Unique Index Fulltext

Why do you think it is secure? Use bullet points to provide your reasons and back it up with code snippet from your application.

- use "pattern", "required" attributes in html form


```

<div class="register">
  <h1>Sign Up</h1>
  <form action="register_check.php" method="POST">
    <label for="username">Username:</label>
    <input type="text" name="username" placeholder="need 2 to 8 characters, special symbol could be: _"
      maxlength="8" pattern="^[\\w_]{2,8}$" id="username" required>
    <label for="password1">Password:</label>
    <input type="password" name="password1" placeholder="need 8 to 16 characters, special symbol could be:
      _ ? !" pattern="^[\\w_?!]{8,16}$" id="password1" required>
    <label for="password1"><font color=blue>Password needs to have at least 1 Uppercase letter 1 Lowercase
      letter 1 digit number
1 special chars and should be 8-16 chars in total</font></label><br><br><br><br>
    <label for="password2">Type in the password again:</label>
    <input type="password" name="password2" placeholder="need 8 to 16 characters, special symbol could be:
      _ ? !" pattern="^[\\w_?!]{8,16}$" id="password2" required>
    <label for="fname">Forename:</label>
    <input type="text" name="fname" placeholder="forename" maxlength="20" id="fname" required>
    <label for="sname">Surname:</label>
    <input type="text" name="sname" placeholder="surname" maxlength="20" id="sname" required>
    <label for="phone">Phone Number:</label>
    <input type="tel" name="phone" placeholder="phone number" id="phone" required>
    <label for="email">Email Address:</label>
    <input type="email" name="email" placeholder="email address" id="email" required>
    <label for="sec_ques">Security Question:</label>
    <select name="sec_ques" value="What city were you born in ?" id="sec_ques">
      <option>What city were you born in ?</option>
      <option>What food is your favourite ?</option>
      <option>What movie do you like most ?</option>
    </select><br><br>
    <label for="sec_ans">Enter you answer here (case-sensitive):</label>
    <input type="text" name="sec_ans" id="sec_ans" maxlength="20" required>
    <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
    <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4a0ZnBRfSLCrthWGR1MpbT"></div>
    <input type="submit" value="Go to email verification" id="submit">
  </form>
</div>

```

- csrf counter-measure implemented, Google reCaptcha implemented

```

5
6   csrf_counter();
7   g_captcha_check("register_form.php");
8

```

```

52
53
54 *
55
56   // CSRF Counter
57   $csrf_token = filter_input(INPUT_POST, 'csrf_token', FILTER_SANITIZE_STRING);
58 *
59   if (!isset($csrf_token) || $csrf_token !== $_SESSION['csrf_token']) {
60     // return 405 http status code
61     header($_SERVER['SERVER_PROTOCOL'] . ' 405 Method Not Allowed');
62     exit;
63     // exit('405 Method Not Allowed');
64   }
65
66
67 *
68
69 *
70   if (isset($_POST['g-recaptcha-response'])) {
71     $secret = "6LdpUT0jAAAAABVgT6Wj8N-sGSawpR4LWS18PwMf";
72     $response = $_POST['g-recaptcha-response'];
73     $remoteip = $_SERVER['REMOTE_ADDR'];
74     $url = "https://www.google.com/recaptcha/api/siteverify?secret=$secret&response=$response&remoteip=$remoteip";
75     $data = file_get_contents($url);
76     $row = json_decode($data, true);
77
78     if ($row['success'] == false) {
79       header_n_sendem("$toPage", "Do the Human-Robot Detection");
80     }
81 *
82     } else {
83       header_n_sendem("$toPage", "G-CAPTCHA NOT BEEN POST");
84     }
85
86 *
87 *
88   }
89
90 *
91   function getUserIpAddr() {
92
93     if (!empty($_SERVER['HTTP_CLIENT_IP'])) {

```

- prepare() used to prevent the SQL injection

Just an example, implemented it everywhere it fits

```
102 // get stmt1 if that username exists.
103 if ($stmt1 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ?')) {
104     $stmt1->bind_param('s', $username);
105     $stmt1->execute();
106     $stmt1->store_result();
107 } else {
108     header_n_sendem("register_form.php", "Cannot prepare statement 1 !");
109 }
110
111 // get stmt2 if that email exists.
112 if ($stmt2 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ? AND email = ?')) {
113     $stmt2->bind_param('ss', $username, $email);
114     $stmt2->execute();
115     $stmt2->store_result();
116 } else {
117     header_n_sendem("register_form.php", "Cannot prepare statement 2 !");
118 }
119
120
121 // Username and Email both dont exist, now can insert new systemuser
122 if ($stmt1 = $conn->prepare('INSERT INTO systemuser (username, passwd, forename, surname, phone, email, activation_code, sec_ques, sec_ans, tfa_seckey, first_ip)
123     VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)')) {
124
125     // set the params needed
126     $password_hashed = password_hash($password1, PASSWORD_DEFAULT);
127     $uniqid = uniqid();
128     $sec_ans = password_hash($sec_ans, PASSWORD_DEFAULT);
129     $google2fa = new Google2FA();
130     $secret = $google2fa->generateSecretKey();
131     $first_ip = getUserIPAddr();
132
133     // pass the params and execute
134     $stmt1->bind_param('sssssssss', $username, $password_hashed, $forename, $surname, $phone, $email, $uniqid, $sec_ques, $sec_ans, $secret, $first_ip);
135     $stmt1->execute();
136 }
137
138
139
140
141
142
```

- htmlspecialchars() used to skip script-inserting risks

Just an example, implemented it everywhere it fits

```
58
59 // forename & surname
60 $forename = htmlspecialchars($_POST['fname']);
61 $surname = htmlspecialchars($_POST['sname']);
62 if ($forename != $_POST['fname']) {
63     header_n_sendem("register_form.php", "Forename contains invalid symbol");
64 }
65 if ($surname != $_POST['sname']) {
66     header_n_sendem("register_form.php", "Surname contains invalid symbol");
67 }
68
```

Task 2 - Develop a secure login feature.

Login feature code screenshots

Login Form

/public_html/login_form.php

```
1 <?php
2 session_start();
3 session_unset();
4 $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));
5
6 // error_reporting(-1);
7
8 ?>
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13 <link rel="stylesheet" href="login_style.css" type="text/css">
14 <script src="https://www.google.com/recaptcha/api.js" async defer></script>
15 </head>
16
17 <body>
18
19
20 <div class="em">
21 <?php if(isset($_GET['error'])): ?>
22 <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
23 <?php endif ?>
24 </div>
25
26 <div class="login">
27 <h1>Login</h1>
28 <form action="login_check.php" method="POST">
29 <label for="username">Username:</label>
30 <input type="text" name="username" placeholder="Enter 2 to 8 characters" id="username" required>
31 <label for="password">Password:</label>
32 <input type="password" name="password" placeholder="Enter 8 to 16 characters" id="password" required>
33 <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
34 <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAD1we4a0ZnBRfSLCrTHwGR1Mpbt"></div>
35 <input type="submit" name="next" value="Next" id="next">
36 </form>
37 </div>
38
39 <div class="toDo">
40 <?php
41 echo "Click to <a href='register_form.php'>Register</a>";
42 echo " | ";
43 echo "Forgot <a href='resetPwd_form.php'>PASSWORD</a> ? ";
44 ?>
45 </div>
46
47 </body>
48 </html>
49
50
51
```

Login Check

/public_html/login_check.php

```
1 k?php
2 session_start();
3
4 require "helper.php";
5
6 if (isset($_SESSION['qrcheck']) && $_SESSION['qrcheck'] == True) {
7     $_SESSION['loggedin'] = True;
8     unset($_SESSION['qrcheck']);
9     unset($_SESSION['qrcheck']);
10    header('Location: index.php');
11    exit();
12 }
13
14 csrf_counter();
15 g_captcha_check("login_form.php");
16
17
18 // entered values
19 $username = $_POST['username'];
20 $password = $_POST['password'];
21
22 // check if all data exists
23 if ( !isset($username, $password) ) {
24     header_n_sendem("login_form.php", "Please fill both the username and password fields");
25 }
26 // check if all data are not empty
27 if (empty($username) || empty($password)) {
28     header_n_sendem("login_form.php", "Please complete the registration form");
29 }
30
31
32 // =====
33 require "database_conn.php";
34 require __DIR__ . "/vendor/autoload.php";
35
36 $ip_address = getUserIpAddress();
37
38
39 // PRE-CHECK LOGIN ATTEMPT
40 if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
41     $stmt->bind_param('ss', $username, $ip_address);
42     $stmt->execute();
43     $stmt->store_result();
44     if ($stmt->num_rows > 0) {
45         $stmt->bind_result($id_rec, $attempt_rec, $banned_time_rec);
46         $stmt->fetch();
47
48         $remain_time = time() - $banned_time_rec;
49         // STILL SHOULD BE BANNED
50         if ($attempt_rec >= 5 && $remain_time < 86400) {
51             header_n_sendem("login_form.php", "You are banned in 24h as too many attempts");
52         }
53         // CAN BE UNBANNED NOW
54         if ($attempt_rec >= 5 && $remain_time > 86400) {
55             if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
56                 $attempt_rec = 0;
57                 $stmt->bind_param('ii', $attempt_rec, $id_rec);
58                 $stmt->execute();
59             }
60         }
61     }
62 }
63 } else {
64     header_n_sendem("login_form.php", "Cannot prepare stmt of PRE-CHECK LOGIN ATMP1");
65 }
66
67 // SELECT the user info in the database
68 if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
69     $stmt->bind_param('s', $username);
70     $stmt->execute();
71     $stmt->store_result();
72     if ($stmt->num_rows > 0) {
73         $stmt->bind_result($id_db, $email_db, $first_ip_db);
74         $stmt->fetch();
75
76         // check if ips are different
77         if ($ip_address != $first_ip_db) {
78             // Alert the user by email
79             $subject = "DIFFERENT IP ALERT";
80             $body = "Your account of LoveJoy Antique are logging-in in another ip address.";
81             send_activation_email($email_db, $username, $subject, $body);
82         }
83     }
84
85     // check if the user has activated the account
86     if ($stmt = $conn->prepare('SELECT id, passwd, is_admin, tfa_secret FROM systemuser WHERE username = ? AND is_activated = ?')) {
87         $is_activated = '1';
88         $stmt->bind_param('ss', $username, $is_activated);
89         $stmt->execute();
90         $stmt->store_result();
91         if ($stmt->num_rows > 0) {
92             // NOW GO TO VERIFY THE PASSWORD BELOW
93             $stmt->bind_result($id_db, $password_db, $is_admin, $secret_key);
94             $stmt->fetch();
95
96             } else {
97                 // NOT ACTIVATED yet, go to activate the account
98                 $uniqid = uniqid();
99                 if ($stmt = $conn->prepare('UPDATE systemuser SET activation_code = ? WHERE id = ?')) {
100                     $stmt->bind_param('ss', $uniqid, $id_db);
101                     $stmt->execute();
102
103                     $uniqid_hashed = password_hash($uniqid, PASSWORD_DEFAULT);
104                     $_SESSION['email'] = $email_db;
105                     $_SESSION['username'] = $username;
106                     $_SESSION['subject'] = 'Account Activation Required';
107                     $activate_link = 'http://web594319279.000webhostapp.com/activate.php?email=' . $email_db . '&code=' . $uniqid_hashed;
108                     // $activate_link = 'http://localhost/lovejoy/activate.php?email=' . $email_db . '&code=' . $uniqid_hashed;
109                     $_SESSION['body'] = '<p>Please click the following link to activate your account: <br><br>'. $activate_link . '</p>';
110                     // $_SESSION['body'] = '<p>Please click the following link to activate your account: <a href="' . $activate_link . '>' . $activate_link . '</a></p>';
111
112                     exit("<b>DO NOT LEAVE THIS PAGE UNTIL YOU FINISHED ACTIVATION! </b><br><br>
113                     Your email has not been activated, <a href='login_email_activate_do.php'>send an email</a> for
114                     activation and then login.");
115                 }
116             } else {
117                 header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE activation_code !");
118             }
119         } else {
120             header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT activated user !");
121         }
122     }
123 } else {
124     header_n_sendem("login_form.php", "Cannot find this user in the database");
125 }
126 } else {
127     header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT user !");
128 }
129 }
```



```

126     }
127 } else {
128     header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT user !");
129 }
130
131 // PASSWORD VERIFY
132 if (password_verify($password, $password_db)) {
133     // SET BACK BAN RESTRICTS IF HAS
134     if ($stmt = $conn->prepare('SELECT id FROM login_attempt WHERE username = ? AND ip_address = ?')) {
135         $stmt->bind_param('ss', $username, $ip_address);
136         $stmt->execute();
137         $stmt->store_result();
138         if ($stmt->num_rows > 0) {
139             $stmt->bind_result($id_rec);
140             $stmt->fetch();
141             if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? AND banned_time = ? WHERE id = ?')) {
142                 $attempt_rec = 0;
143                 $banned_time_rec = 0;
144                 $stmt->bind_param('iii', $attempt_rec, $banned_time_rec, $id_rec);
145                 $stmt->execute();
146             } else {
147                 header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE login_attempt AFTER LOGIN");
148             }
149         }
150     }
151     session_regenerate_id();
152     $_SESSION['id'] = $id_db;
153     $_SESSION['username'] = $username;
154     $_SESSION['is_admin'] = $is_admin;
155
156     // LAST LOGIN TIME CHECK
157     if ($stmt = $conn->prepare('SELECT last_login FROM systemuser WHERE id = ? AND username = ?')) {
158         $stmt->bind_param('ss', $id_db, $username);
159         $stmt->execute();
160         $stmt->store_result();
161         if ($stmt->num_rows > 0) {
162             $stmt->bind_result($last_login);
163             $stmt->fetch();
164             $time_now = time();
165             if ($time_now - $last_login < 1800) {
166                 $_SESSION['logged_in'] = True;
167                 header('Location: index.php');
168                 exit();
169             }
170
171             // QR code generation
172             $google2fa = new \PragmarX\Google2FA\Google2FA();
173             $text = $google2fa->getQRCodeUrl('LoveJoy Antique', $username, $secret_key);
174             // $text = $google2fa->getQRCodeUrl('localhost', $username, $secret_key);
175             $image_url = 'https://chart.googleapis.com/chart?cht=qr&chs=300x300&chl='.$text;
176             $_SESSION['qr_code'] = '';
177             header('Location: qr_form.php');
178             exit();
179         } else {
180             header_n_sendem("login_form.php", "Unknown error in the login time check");
181         }
182     } else {
183         header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT last_login");
184     }
185 }
186
187 // PASSWORD WRONG
188 } else {
189     if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND
190     ip_address = ?')) {
191         $stmt->bind_param('ss', $username, $ip_address);
192         $stmt->execute();
193         $stmt->store_result();
194
195         // NO SUCH AN IP ADDRESS
196         if ($stmt->num_rows < 1) {
197             // store in a new row
198             if ($stmt = $conn->prepare('INSERT INTO login_attempt (username, ip_address) VALUES (?, ?)')) {
199                 $stmt->bind_param('ss', $username, $ip_address);
200                 $stmt->execute();
201             } else {
202                 header_n_sendem("login_form.php", "Cannot prepare stmt of INSERT login_attempt");
203             }
204         }
205         if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND
206         ip_address = ?')) {
207             $stmt->bind_param('ss', $username, $ip_address);
208             $stmt->execute();
209             $stmt->store_result();
210             $stmt->bind_result($id_la, $attempt, $banned_time);
211             $stmt->fetch();
212         } else {
213             header_n_sendem("login_form.php", "Cannot prepare stmt2 of SELECT login_attempt");
214         }
215     }
216     // HAVE THIS IP ADDRESS
217 } else {
218     $stmt->bind_result($id_la, $attempt, $banned_time);
219     $stmt->fetch();
220
221     // ADDING ATTEMPT RECORD
222     if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
223         $attempt = $attempt + 1;
224         $stmt->bind_param('is', $attempt, $id_la);
225         $stmt->execute();
226     } else {
227         header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE attempt");
228     }
229     // CHECK IF IT'S UP-LIMIT ATTEMPT TIME
230     if ($attempt > 5) {
231         if ($stmt = $conn->prepare('UPDATE login_attempt SET banned_time = ? WHERE id = ?')) {
232             $new_banned_time = time();
233             $stmt->bind_param('is', $new_banned_time, $id_la);
234             $stmt->execute();
235         } else {
236             header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE banned_time");
237         }
238     }
239 }
240
241 } else {
242     header_n_sendem("login_form.php", "cannot prepare stmt1 of SELECT login_attempt");
243 }
244
245 header_n_sendem("login_form.php", "Incorrect username and/or password");
246
247 }
248
249 $stmt->close();
250 $conn->close();
251
252 }
253
254 }
255

```

Why do you think it is secure? Use bullet points to provide your reasons and back it up code snippet from your application.

- use "pattern", "required" attributes in html form

```
26 <div class="login">
27 <h1>Login</h1>
28 <form action="login_check.php" method="POST">
29 <label for="username">Username:</label>
30 <input type="text" name="username" placeholder="Enter 2 to 8 characters" id="username" required>
31 <label for="password">Password:</label>
32 <input type="password" name="password" placeholder="Enter 8 to 16 characters" id="password" required>
33 <input type="hidden" name="csrf_token" value="php echo $_SESSION['csrf_token']; ?">
34 <div class="g-recaptcha" data-sitekey="6LdpU0jAAAAA0D1we4a0ZnBRfSLCrthWGR1MpbT"></div>
35 <input type="submit" name="next" value="Next" id="next">
36 </form>
37 </div>
```

- csrf counter-measure implemented, Google reCaptcha implemented

```
14 csrf_counter();
15 g_captcha_check("login_form.php");
16
```

- prepare() used to prevent the SQL injection

Just an example, implemented it everywhere it fit

```
67 // SELECT the user info in the database
68 if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
69     $stmt->bind_param('s', $username);
70     $stmt->execute();
71     $stmt->store_result();
72     if ($stmt->num_rows > 0) {
73
74         $stmt->bind_result($id_db, $email_db, $first_ip_db);
75         $stmt->fetch();
76     }
```

- check if this ip is under ban before any other database-related thing

```
40 // PRE-CHECK LOGIN ATTEMPT
41 if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
42     $stmt->bind_param('ss', $username, $ip_address);
43     $stmt->execute();
44     $stmt->store_result();
45     if ($stmt->num_rows > 0) {
46         $stmt->bind_result($id_rec, $attempt_rec, $banned_time_rec);
47         $stmt->fetch();
48
49         $remain_time = time() - $banned_time_rec;
50         // STILL SHOULD BE BANNED
51         if ($attempt_rec >= 5 && $remain_time < 86400) {
52             header_n_sendem("login_form.php", "You are banned in 24h as too many attempts");
53         }
54         // CAN BE UNBANNED NOW
55         if ($attempt_rec >= 5 && $remain_time > 86400) {
56             if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
57                 $attempt_rec = 0;
58                 $stmt->bind_param('ii', $attempt_rec, $id_rec);
59                 $stmt->execute();
60             }
61         }
62     }
63 } else {
64     header_n_sendem("login_form.php", "Cannot prepare stmt of PRE-CHECK LOGIN ATMPMT");
65 }
66
67 // SELECT the user info in the database
68 if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
69     $stmt->bind_param('s', $username);
```

- attempts will be recoreded when password is wrong, after 5 time, this ip cannot log in this account in 24 hours

```

193 // PASSWORD WRONG
194 } else {
195
196     if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
197         $stmt->bind_param('ss', $username, $ip_address);
198         $stmt->execute();
199         $stmt->store_result();
200
201         // NO SUCH AN IP ADDRESS
202         if ($stmt->num_rows < 1) {
203             // store in a new row
204             if ($stmt = $conn->prepare('INSERT INTO login_attempt (username, ip_address) VALUES (?, ?)')) {
205                 $stmt->bind_param('ss', $username, $ip_address);
206                 $stmt->execute();
207             } else {
208                 header_n_sendem("login_form.php", "Cannot prepare stmt of INSERT login_attempt");
209             }
210             if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
211                 $stmt->bind_param('ss', $username, $ip_address);
212                 $stmt->execute();
213                 $stmt->store_result();
214                 $stmt->bind_result($id_la, $attempt, $banned_time);
215                 $stmt->fetch();
216             } else {
217                 header_n_sendem("login_form.php", "Cannot prepare stmt2 of SELECT login_attempt");
218             }
219             // HAVE THIS IP ADDRESS
220         } else {
221             $stmt->bind_result($id_la, $attempt, $banned_time);
222             $stmt->fetch();
223         }
224
225         // ADDING ATTEMPT RECORD
226         if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
227             $attempt = $attempt + 1;
228             $stmt->bind_param('is', $attempt, $id_la);
229             $stmt->execute();
230         } else {
231             header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE attempt");
232         }
233         // CHECK IF IT'S UP-LIMIT ATTEMPT TIME
234         if ($attempt >= 5) {
235             if ($stmt = $conn->prepare('UPDATE login_attempt SET banned_time = ? WHERE id = ?')) {
236                 $new_banned_time = time();
237                 $stmt->bind_param('is', $new_banned_time, $id_la);
238                 $stmt->execute();
239             } else {
240                 header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE banned_time");
241             }
242         }
243
244     } else {
245         header_n_sendem("login_form.php", "Cannot prepare stmt1 of SELECT login_attempt");
246     }
247
248     header_n_sendem("login_form.php", "Incorrect username and/or password");
249
250 }
251
252 $stmt->close();
253 $conn->close();
254
255 ?>

```

- alert user when a different ip is logging-in their account

```

76
77 // check if ips are different
78 if ($ip_address != $first_ip_db) {
79     // Alert the user by email
80     $subject = "DIFFERENT IP ALERT";
81     $body = "Your account of LoveJoy Antique are logging-in in another ip address.";
82     send_activation_email($email_db, $username, $subject, $body);
83 }
84

```

- user have to activated their email before log in

```

96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
} else {
    // NOT ACTIVATED yet, go to activate the account
    $uniqid = uniqid();
    if ($stmt = $conn->prepare('UPDATE systemuser SET activation_code = ? WHERE id = ?')) {
        $stmt->bind_param('ss', $uniqid, $id_db);
        $stmt->execute();

        $uniqid_hashed = password_hash($uniqid, PASSWORD_DEFAULT);
        $_SESSION['email'] = $email_db;
        $_SESSION['username'] = $username;
        $_SESSION['subject'] = 'Account Activation Required';
        $activate_link = 'http://web594319279.000webhostapp.com/activate.php?email=' . $email_db . '&code=' . $uniqid_hashed;
        // $activate_link = 'http://localhost/lovejoy/activate.php?email=' . $email_db . '&code=' . $uniqid_hashed;
        $_SESSION['body'] = '<p>Please click the following link to activate your account: <br><br>' . $activate_link . '</p>';
        // $_SESSION['body'] = '<p>Please click the following link to activate your account: <a href=' . $activate_link . '>' . $activate_link . '</a>';

        exit("<b>DO NOT LEAVE THIS PAGE UNTIL YOU FINISHED ACTIVATION! </b><br><br>
        Your email has not been activated, <a href='login_email_activate_do.php'>send an email</a> for activation and then login."
        );
    } else {
        header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE activation_code !");
    }
}

```

Task 3 - Implement password strength and password recovery

List each password policy element that you implemented and back it up with code snippets from your application

Password Strength

- check if password has been post and is not empty

```

9
10
11
12
13
14
15
16
17
18
// check if all data exists
if (!isset($_POST['username'], $_POST['password1'], $_POST['password2'], $_POST['fname'], $_POST['sname'],
    $_POST['phone'], $_POST['email'], $_POST['sec_ques'], $_POST['sec_ans'])) {
    header_n_sendem("register_form.php", "Please complete the registration form");
}
// check if all data are not empty
if (empty($_POST['username']) || empty($_POST['password1']) || empty($_POST['password2']) || empty($_POST['fname'])
    || empty($_POST['sname']) || empty($_POST['phone']) || empty($_POST['email']) || empty($_POST['sec_ques'])
    || empty($_POST['sec_ans'])) {
    header_n_sendem("register_form.php", "Please complete the registration form");
}

```

- skip html special chars

- two passwords entered should be the same

- Password should be in specific pattern

- Could not be the common-used password

```

29 // password
30 $password1 = htmlspecialchars($_POST['password1']);
31 $password2 = htmlspecialchars($_POST['password2']);
32 if (($password1 != $_POST['password1']) || ($password2 != $_POST['password2'])) {
33     header_n_sendem("register_form.php", "Password contain invalid symbol");
34 }
35 if ($password1 != $password2) {
36     header_n_sendem("register_form.php", "Two passwords are different");
37 }
38 // PASSWORD PATTERN
39 if ((strlen($password1) < 8) || strlen($password1) > 16 ||
40     !preg_match('@[A-Z]@', $password1) || !preg_match('@[a-z]@', $password1) ||
41     !preg_match('@[0-9]@', $password1) || !preg_match('@[_?!]@', $password1)) {
42     header_n_sendem("register_form.php", "password in wrong pattern or is not strong enough");
43 }
44 // PREVENT DICTIONARY ATTACK
45 $handle = fopen('./common_password/com_pwd.txt', 'r');
46 if ($handle) {
47     while (false !== ($com_pwd = fgets($handle, 256))) {
48         if ($password1 == $com_pwd) {
49             header_n_sendem("register_form.php", "Password is vulnerable to Dictionary Attacks");
50         }
51     }
52 } else {
53     header_n_sendem("register_form.php", "Failed to open the pwd TXT");
54 }
55 fclose($handle);
56
57
58
59

```

← skip html special characotrs and check

two passwords entered should be the same

← Password needs to have at least:
1 Uppercase letter,
1 Lowercase letter,
1 digit number,
1 special chars,
and should be 8-16 chars in total

prevent Dictionary atteck check

/public_html/common_password/com_pwd.txt

```

1 123456
2 password
3 12345678
4 qwerty
5 123456789
6 12345
7 1234
8 111111
9 1234567
10 dragon
11 123123
12 baseball
13 abc123
14 football
15 monkey
16 letmein
17 696969
18 shadow
19 master
20 666666
21 qwertyuiop
22 123321
23 mustang
24 1234567890
25 michael
26 654321

```

For Dictionary Attack checking

- have 1000 common passwords in total

Password Recovery

- Include 3 steps:

- - enter username and email

- - answer the security question

- - set the new password

/public_html/resetPwd_form.php

```
1  <?php session_start(); ?>
2
3  <!DOCTYPE html>
4  <html>
5  <head>
6      <link rel="stylesheet" href="resetPwd_style.css" type="text/css">
7      <script src="https://www.google.com/recaptcha/api.js" async defer></script>
8  </head>
9
10
11 <body>
12
13     <div class="em">
14         <?php if(isset($_GET['error'])): ?>
15             <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
16         <?php endif ?>
17     </div>
18
19
20     <!-- reset steps -->
21     <?php if (!isset($_SESSION['resetstep'])) { ?>
22
23         <?php $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true)); ?>
24
25         <!-- initial input -->
26         <div class="reset">
27             <h1>Reset your password</h1>
28             <form action="resetPwd_check.php" method="POST">
29                 <center><b>Step 0</b></center>
30                 <label for="username">Username:</label>
31                 <input type="text" name="username" placeholder="Enter 2 to 8 charactors" id="username" required>
32                 <label for="email">Email Address:</label>
33                 <input type="email" name="email" placeholder="your email address" id="email" required>
34                 <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
35                 <input type="submit" value="Go to answer Security Question" id="submit">
36             </form>
37         </div>
38
39     <?php } else { ?>
40     <?php if ($_SESSION['resetstep'] === "step1") { ?>
41
42         <!-- step 1 -->
43         <div class="reset">
44             <h1>Reset your password</h1>
45             <form action="resetPwd_check.php" method="POST">
46                 <center><b>Step 1</b></center>
47                 <label for="sec_ques">Security Question:</label>
48                 <font color=red><b><?php echo $_SESSION['sec_ques_db']; ?></b></font>
49                 <label for="sec_ans">Enter you answer here (case-sensitive):</label>
50                 <input type="text" name="sec_ans" id="sec_ans" maxlength="20" required>
51                 <input type="submit" value="Go to set new password" id="submit">
52             </form>
53         </div>
54
55     <?php } elseif ($_SESSION['resetstep'] === "step2") { ?>
56
57         <!-- step 2 -->
58         <div class="reset">
59             <h1>Reset your password</h1>
60             <form action="resetPwd_check.php" method="POST">
61                 <center><b>Step 2</b></center>
62                 <label for="new_pwd1">Set your new password:</label>
63                 <input type="password" name="new_pwd1" id="new_pwd1" required>
64                 <label for="new_pwd2">Enter your new password again:</label>
65                 <input type="password" name="new_pwd2" id="new_pwd2" required>
66                 <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAD1w4a0ZnBRfSLCrthWGR1MpbT"></div>
67                 <input type="submit" value="Submit" id="submit">
68             </form>
69         </div>
70
71     <?php } else { ?>
72         <b>Unknown session value found when resetting password</b>
73     <?php } ?>
74
75
76
77
78
79
80 <div class="toLog">
81     <?php
82         echo "Back to <a href='login_form.php'> LOGIN </a>";
83     ?>
84 </div>
85
86 </body>
87 </html>
```



resetPwd_style.css

1.4 kB

2022-12-05 17:11:00

SAVE & CLOSE

SAVE

- check of entering username and email

/public_html/resetPwd_check.php

```
1 <?php
2 session_start();
3
4 require 'helper.php';
5
6
7 if (!isset($_SESSION['resetstep'])) {
8
9     csrf_counter();
10
11     // check if all data exists
12     if (!isset($_POST['username'], $_POST['email'])) {
13         header_n_sendem("resetPwd_form.php", "Please complete the boxes");
14     }
15     // check if all data are not empty
16     if (empty($_POST['username']) || empty($_POST['email'])) {
17         header_n_sendem("resetPwd_form.php", "Please complete the boxes");
18     }
19     // username
20     $username = htmlspecialchars($_POST['username']);
21     if ($username != $_POST['username']) {
22         header_n_sendem("resetPwd_form.php", "Username contains invalid symbol");
23     }
24     // email
25     $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
26     if (filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
27         header_n_sendem("resetPwd_form.php", "Invalid email address");
28     }
29
30     require 'database_conn.php';
31
32     // check if that username exists.
33     if ($stmt = $conn->prepare('SELECT id, sec_ques, sec_ans FROM systemuser WHERE username = ? AND email = ?')) {
34         $stmt->bind_param('ss', $username, $email);
35         $stmt->execute();
36         $stmt->store_result();
37         // find the user
38         if ($stmt->num_rows > 0) {
39             $stmt->bind_result($id, $sec_ques, $sec_ans);
40             $stmt->fetch();
41             $_SESSION['id'] = $id;
42             $_SESSION['username'] = $username;
43             $_SESSION['resetstep'] = "step1";
44             $_SESSION['sec_ques_db'] = $sec_ques;
45             $_SESSION['sec_ans_db'] = $sec_ans;
46             header('Location: resetPwd_form.php');
47             exit();
48         } else {
49             header_n_sendem("resetPwd_form.php", "No such a user/email in the database");
50         }
51     } else {
52         header_n_sendem("resetPwd_form.php", "Cannot prepare stmt of SELECT user for resetting !");
53     }
54
55 } else {
```

- check of answering the security question. The correct answer in database is hashed

```
54
55
56 } else {
57
58     if ($_SESSION['resetstep'] === "step1") {
59         // check if data exists
60         if (!isset($_POST['sec_ans'])) {
61             header_n_sendem("resetPwd_form.php", "Please complete the box");
62         }
63         // check if data is not empty
64         if (empty($_POST['sec_ans'])) {
65             header_n_sendem("resetPwd_form.php", "Please complete the box");
66         }
67         // check form of security answer
68         $sec_ans = htmlspecialchars($_POST['sec_ans']);
69         if ($sec_ans != $_POST['sec_ans']) {
70             header_n_sendem("resetPwd_form.php", "Security answer contains invalid symbol");
71         }
72
73         // check if the answer is correct
74         if (password_verify($sec_ans, $_SESSION['sec_ans_db'])) {
75             $_SESSION['resetstep'] = "step2";
76             unset($_SESSION['sec_ques_db']);
77             unset($_SESSION['sec_ans_db']);
78             header('Location: resetPwd_form.php');
79             exit();
80         } else {
81             header_n_sendem("resetPwd_form.php", "Incorrect answer");
82         }
83     }
84
85     elseif ($_SESSION['resetstep'] === "step2") {
```

- reset the new password. The new password cannot be the same as the old one. And also the password pattern check, dictionary attack prevention...like the one in registration part

```

85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
?>

elseif ($_SESSION['resetstep'] === "step2") {

    g_captcha_check("resetPwd_form.php");

    // check if data exists
    if (!isset($_POST['new_pwd1'], $_POST['new_pwd2'])) {
        header_n_sendem("resetPwd_form.php", "Please complete the boxes");
    }

    // check if data is not empty
    if (empty($_POST['new_pwd1']) || empty($_POST['new_pwd2'])) {
        header_n_sendem("resetPwd_form.php", "Please complete the boxes");
    }

    $new_pwd1 = htmlspecialchars($_POST['new_pwd1']);
    $new_pwd2 = htmlspecialchars($_POST['new_pwd2']);
    if (($new_pwd1 != $_POST['new_pwd1']) || ($new_pwd2 != $_POST['new_pwd2'])) {
        header_n_sendem("resetPwd_form.php", "Password contain invalid symbol");
    }

    if ($new_pwd1 != $new_pwd2) {
        header_n_sendem("resetPwd_form.php", "Two passwords are different");
    }

    // PASSWORD PATTERN
    if ((strlen($new_pwd1) < 8) || strlen($new_pwd1) > 16 ||
        !preg_match('@[A-Z]@', $new_pwd1) || !preg_match('@[a-z]@', $new_pwd1) ||
        !preg_match('@[0-9]@', $new_pwd1) || !preg_match('@[_?!]@', $new_pwd1)) {
        header_n_sendem("resetPwd_form.php", "password in wrong pattern or is not strong enough");
    }

    // PREVENT DICTIONARY ATTACK
    $handle = fopen('./common_password/com_pwds.txt', 'r');
    if ($handle) {
        while (false !== ($com_pwd = fgets($handle, 256))) {
            if ($new_pwd1 == $com_pwd) {
                header_n_sendem("resetPwd_form.php", "Password is vulnerable to Dictionary Attacks");
            }
        }
    } else {
        header_n_sendem("resetPwd_form.php", "Failed to open the pwd TXT");
    }
    fclose($handle);

    require 'database_conn.php';

    // first check if pwd new == old
    if ($stmt = $conn->prepare('SELECT passwd FROM systemuser where id = ? AND username = ?')) {
        $stmt->bind_param('ss', $_SESSION['id'], $_SESSION['username']);
        $stmt->execute();
        $stmt->store_result();
        if ($stmt->num_rows > 0) {
            $stmt->bind_result($old_pwd);
            $stmt->fetch();
            // Cannot be the same as the old one
            if (password_verify($new_pwd1, $old_pwd)) {
                header_n_sendem("resetPwd_form.php", "Your new password is the same as the old one");
            }
        } else {
            header_n_sendem("resetPwd_form.php", "session error: cannot find the user in database");
        }
    } else {
        header_n_sendem("resetPwd_form.php", "Cannot prepare stmt of SELECT pwd");
    }

    // now we can update new pwd into db
    if ($stmt = $conn->prepare('UPDATE systemuser SET passwd = ? WHERE id = ?')) {
        $hashed_pwd = password_hash($new_pwd1, PASSWORD_DEFAULT);
        $stmt->bind_param('ss', $hashed_pwd, $_SESSION['id']);
        $stmt->execute();
        exit('Your password has now been reset! Go to <a href="login_form.php">login</a>');
    } else {
        header_n_sendem("resetPwd_form.php", "Cannot prepare stmt of UPDATE pwd");
    }
}

else {
    exit('Unknown session value found when resetting password');
}

```

SAVE & CLOSE

Task 4 - Implement a "Evaluation Request" web page.

Request Evaluation feature screenshot

Evaluation Request form

Edit file

/public_html/reqEva_form.php

```
1 <?php
2     session_start();
3
4     // logged in check
5     if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
6         header('Location: login_form.php');
7         exit;
8     }
9
10    $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));
11 ?>
12
13 <!DOCTYPE html>
14 <html>
15 <head>
16     <meta charset="utf-8">
17     <link href="reqEva_style.css" rel="stylesheet" type="text/css">
18     <script src="https://www.google.com/recaptcha/api.js" async defer></script>
19 </head>
20
21
22 <body>
23
24     <nav class="navtop">
25         <div>
26             <h1><a href="index.php">LoveJoy Antique</a></h1>
27             <a href="logout.php">Logout</a>
28         </div>
29     </nav>
30
31     <div class="em">
32         <?php if(isset($_GET['error'])): ?>
33         <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
34         <?php endif ?>
35     </div>
36
37     <div class="content">
38         <h2>Request Evaluation</h2>
39         <h3>Upload your request here</h3>
40         <br><br>
41         <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
42             Image:
43             <input type="file" accept="image/*" name="image" required>
44             Contact by:
45             <select name="contact" value="email">
46                 <option> email </option>
47                 <option> phone </option>
48             </select>
49             <br><br>
50             <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255" maxlength="255" required>
51             <br>
52             <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
53             <div class="g-recaptcha" data-sitekey="6LdpU0jAAAAAD1we4a0ZnBRfSLCrtHwGRImpbt"></div>
54             <input type="submit" value="Submit">
55         </form>
56     </div>
57
58 </body>
59 </html>
```

SAVE & CLOSE

SAVE

Evaluation Request check

/public_html/reqEva_check.php

```

1  <?php
2      session_start();
3
4      require 'helper.php';
5
6      csrf_counter();
7      g_captcha_check("reqEva_form.php");
8
9      // check if all data exists
10     if ( !isset($_FILES['image']) || !isset($_POST['contact']) || !isset($_POST['detail']) ) {
11         header_n_sendem("reqEva_form.php", "Please complete the request");
12     }
13     // check if all data are not empty
14     if (empty($_FILES['image']) || empty($_POST['contact']) || empty($_POST['detail'])) {
15         header_n_sendem("reqEva_form.php", "Please complete the request");
16     }
17
18
19     // contact
20     $img_contact = $_POST['contact'];
21
22     // detail
23     $img_detail = htmlspecialchars($_POST['detail']);
24     if ($img_detail != $_POST['detail']) {
25         header_n_sendem("reqEva_form.php", "Detail contains invalid symbol");
26     }
27
28
29     // image
30     $img_name = $_FILES['image']['name'];
31     $img_size = $_FILES['image']['size'];
32     $tmp_name = $_FILES['image']['tmp_name'];
33     $error = $_FILES['image']['error'];
34
35     if ($error == 0) {
36
37         // no more than 1mb
38         if ($img_size > 1048576) {
39             header_n_sendem("reqEva_form.php", "The image is too large, should be no more than 1mb");
40         } else {
41
42             $img_ex = pathinfo($img_name, PATHINFO_EXTENSION);
43             $img_ex_lc = strtolower($img_ex);
44             $allowed_type = array("jpg", "jpeg", "png");
45
46             // save it
47             if (in_array($img_ex_lc, $allowed_type)) {
48                 $new_img_name = uniqid("IMG-", true). ".$img_ex_lc;
49                 $img_upload_path = 'img_upload/' . $new_img_name;
50                 move_uploaded_file($tmp_name, $img_upload_path);
51             } else {
52                 header_n_sendem("reqEva_form.php", "The file is in the wrong type");
53             }
54         }
55     } else {
56         header_n_sendem("reqEva_form.php", "Unknown error");
57     }
58
59
60
61
62     // =====
63
64     $user_name = $_SESSION['username'];
65     $user_id = $_SESSION['id'];
66
67     require 'database_conn.php';
68
69     if ($stmt = $conn->prepare("INSERT INTO request (image_url, contact, detail, uploader) VALUES (?, ?, ?, ?)") {
70         $stmt->bind_param('ssss', $img_upload_path, $img_contact, $img_detail, $user_name);
71         $stmt->execute();
72         exit("Send the request successfully, please wait for the reply.<a href=reqEva_form.php> Go back </a>.");
73     } else {
74         header_n_sendem("reqEva_form.php", "Cannot prepare stmt of INSERT image !");
75     }
76
77     $stmt->close();
78     $conn->close();
79
80     ?>
81
82
83

```

SAVE & CLOSE

SAVE

Why do you think it is secure?

- use "pattern", "required" attributes in html form, also use "maxlength" to restrict the text which is too long

```

36
37
38 <h2>Request Evaluation</h2>
39 <h3>Upload your request here</h3>
40 <br><br>
41
42 <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
43     Image:
44     <input type="file" accept="image/*" name="image" required>
45     Contact by:
46     <select name="contact" value="email">
47         <option> email </option>
48         <option> phone </option>
49     </select>
50     <br/><br/>
51     <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255"
52         maxlength="255" required>
53     <br>
54     <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
55     <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAD1we4a0ZnBRfSLCrthWGR1Mpbt"></div>
56     <input type="submit" value="Submit">
57 </form>
58 </div>
59 </body>
60 </html>

```

- csrf counter-measure implemented, Google reCaptcha implemented

```

6 csrf_counter();
7 g_captcha_check("reqEva_form.php");

```

- prepare() used to prevent the SQL injection

```

69 if ($stmt = $conn->prepare("INSERT INTO request (image_url, contact, detail, uploader) VALUES (?, ?, ?, ?)") {
70     $stmt->bind_param('ssss', $img_upload_path, $img_contact, $img_detail, $user_name);
71     $stmt->execute();
72     exit("Send the request successfully, please wait for the reply.<a href=reqEva_form.php> Go back </a>.");
73 }
74 else {
75     header_n_sendem("reqEva_form.php", "Cannot prepare stmt of INSERT image !");
76 }
77

```

- htmlspecialchars() used to skip script-inserting risks

```

21 // detail
22 $img_detail = htmlspecialchars($_POST['detail']);
23 if ($img_detail != $_POST['detail']) {
24     header_n_sendem("reqEva_form.php", "Detail contains invalid symbol");
25 }
26
27

```

Task 5 - Develop a feature that will allow customers to submit photographs

Code of the feature

Form

```

36
37
38 <h2>Request Evaluation</h2>
39 <h3>Upload your request here</h3>
40 <br><br>
41
42 <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
43     Image:
44     <input type="file" accept="image/*" name="image" required>
45     Contact by:
46     <select name="contact" value="email">
47         <option> email </option>
48         <option> phone </option>
49     </select>
50     <br/><br/>
51     <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255"
52         maxlength="255" required>
53     <br>
54     <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
55     <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAD1we4a0ZnBRfSLCrthWGR1Mpbt"></div>
56     <input type="submit" value="Submit">
57 </form>
58 </div>
59 </body>
60 </html>

```

Check

```
28 // image
29 $img_name = $_FILES['image']['name'];
30 $img_size = $_FILES['image']['size'];
31 $tmp_name = $_FILES['image']['tmp_name'];
32 $error = $_FILES['image']['error'];
33
34
35 if ($error == 0) {
36
37     // no more than 1mb
38     if ($img_size > 1048576) {
39         header_n_sendem("reqEva_form.php", "The image is too large, should be no more than 1mb");
40     } else {
41
42         $img_ex = pathinfo($img_name, PATHINFO_EXTENSION);
43         $img_ex_lc = strtolower($img_ex);
44         $allowed_type = array("jpg", "jpeg", "png");
45
46         // save it
47         if (in_array($img_ex_lc, $allowed_type)) {
48             $new_img_name = uniqid("IMG-", true). "." . $img_ex_lc;
49             $img_upload_path = 'img_upload/' . $new_img_name;
50             move_uploaded_file($tmp_name, $img_upload_path);
51         } else {
52             header_n_sendem("reqEva_form.php", "The file is in the wrong type");
53         }
54     }
55 } else {
56     header_n_sendem("reqEva_form.php", "Unknown error");
57 }
58
59
60
61
62 // =====
```

Why do you think it is secure?

- use "accept" attribute in html form to let user only can upload image

```
42 Image:
43 <input type="file" accept="image/*" name="image" required>
```

- not allow image more than 1mb

```
37 // no more than 1mb
38 if ($img_size > 1048576) {
39     header_n_sendem("reqEva_form.php", "The image is too large, should be no more than 1mb");
40 } else {
41
```

Task 6 – Request Listing Page

Code of the feature

/public_html/reqList_form.php

```

1  <?php
2      session_start();
3
4      require 'helper.php';
5
6      // logged in check
7  *  if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
8      header('location: login_form.php');
9      exit();
10 *  } elseif ($_SESSION['is_admin'] != 1) {
11      header_n_sendem("index.php", "You are not administrator");
12  }
13  ?>
14
15  <!DOCTYPE html>
16  *  <html>
17  *      <head>
18          <title>Request List</title>
19          <meta charset="utf-8">
20          <link href="reqList_style.css" rel="stylesheet" type="text/css">
21      </head>
22  *      <body class="loggedin">
23  *          <nav class="navtop">
24
25              <div>
26                  <h1><a href="index.php">LoveJoy Antique</a></h1>
27                  <a href="logout.php">Logout</a>
28              </div>
29          </nav>
30
31          <div class="em">
32              <?php if(isset($_GET['error'])): ?>
33                  <script>alert(" Error: <?php echo $_GET['error']; ?>");</script>
34              <?php endif ?>
35          </div>
36
37  *      <div class="content">
38          <h2>Request List (ADMIN)</h2>
39
40          <?php
41              require 'database_conn.php';
42
43              $sql = "SELECT * FROM request ORDER BY id DESC";
44              $res = mysqli_query($conn, $sql);
45
46              if (mysqli_num_rows($res) > 0) {
47  *                  while ($images = mysqli_fetch_assoc($res)) {
48
49                      <div class="request">
50                          
51                          <br>
52                          <?php echo "id: " . $_images['id']; ?>
53                          <br>
54                          <?php echo "detail: " . $_images['detail']; ?>
55                          <br>
56                          <?php echo "contact: " . $_images['contact']; ?>
57                      </div>
58                  <?php
59                      }
60              }
61          ?>
62      </div>
63
64  </body>
65  </html>

```



resetPwd_style.css

1.4 kB

2022-12-03 17:33:00

SAVE & CLOSE

SAVE

Why do you think it is secure?

- not allow user who is not logged-in / administrator to go in

```

1  <?php
2      session_start();
3
4      require 'helper.php';
5
6      // logged in check
7  *  if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
8      header('location: login_form.php');
9      exit();
10 *  } elseif ($_SESSION['is_admin'] != 1) {
11      header_n_sendem("index.php", "You are not administrator");
12  }
13  ?>

```

- no input box to have the SQL injection risk

- for data shown on this page, they have been sanitized and validated before saving into the database.