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

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.	**ssword policy 10marks** Password entropy, encrypted storage, security questions and recovery of password Implemented all the required features
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.	**ulnerabilities 10 marks** SQL injection, XSS, CSRF, File Upload and any other obvious vulnerability. Implemented counter-measures for all the listed vulnerabilities
All the requirements are implemented to authenticate users. Implementation quality is excellent.	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.	**uthentication 10 marks_** User identity management (registration and login etc), Email verification for registration, 2 factor authentications (PIN and or email), Implemented all the required features
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.	**bfuscation/Common attacks 10 marks** Brute force attack – Number of attempts Botnet attack – Captcha ----

				Dictionary attack/Rainbow table attack Implemented Attempts Restriction, Google reCaptcha, and Dictionary Attack Prevention
Implementation of other security features has no flaws. No obvious security feature is ignored.	Several security features implemented. Implementation has flaws.	Other security features are implemented 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 important 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/implementation could have been better.	Claimed features are somewhat complex and implementation 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/Database-Task1	**atures of webs 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

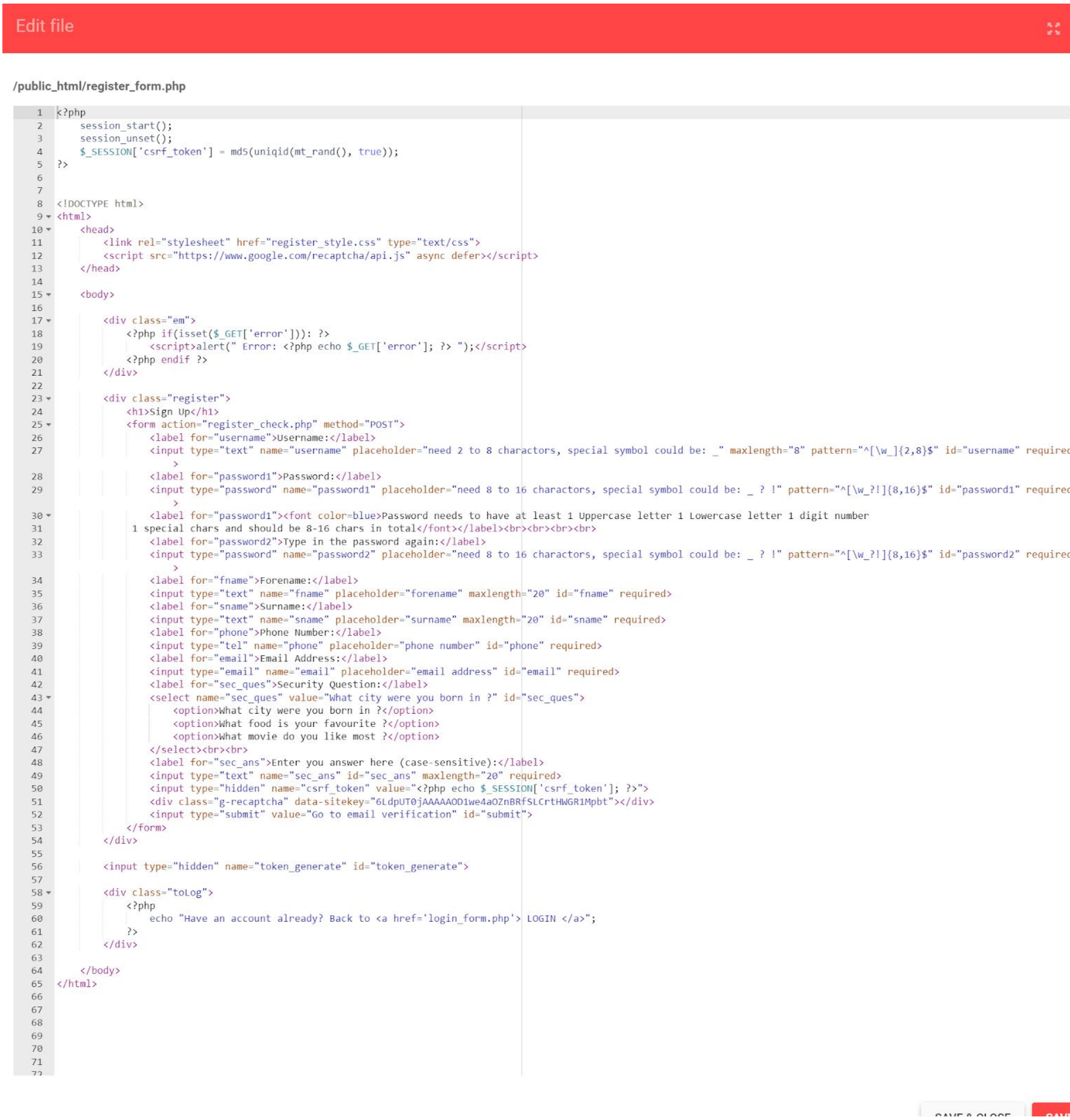
Marking grid filled up by you.

See above.

Task 1 – User registration

Registration feature code screenshots

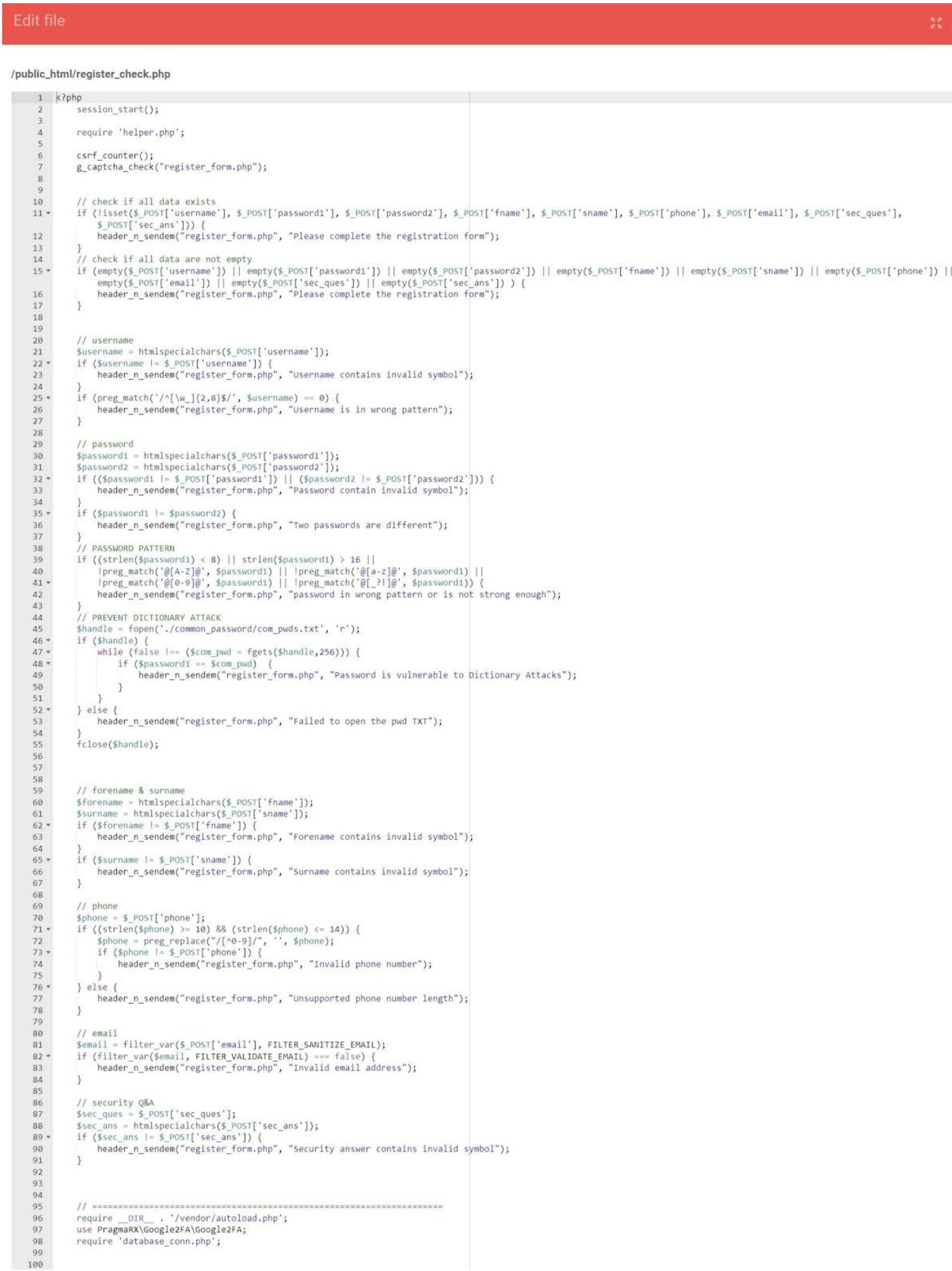
Register Form



/public_html/register_form.php

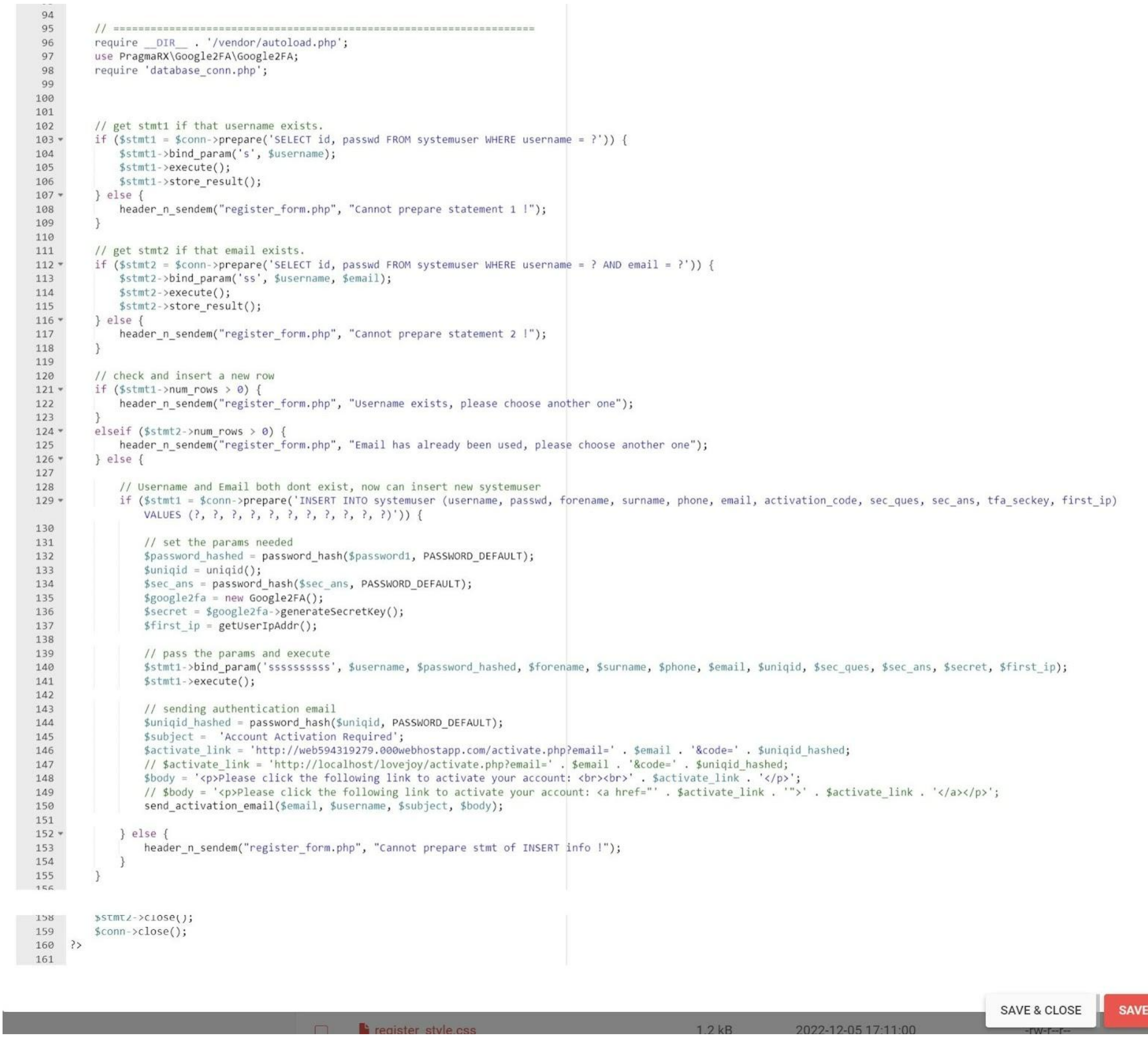
```
<?php
    session_start();
    session_unset();
    $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));
?>


<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" href="register_style.css" type="text/css">
        <script src="https://www.google.com/recaptcha/api.js" async defer></script>
    </head>

    <body>

        <div class="em">
            <?php if(isset($_GET['error'])): ?>
                <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
            <?php endif ?>
        </div>

        <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 charactors, special symbol could be: _" maxlength="8" pattern="^[\w_]{2,8}$" id="username" required
                    >
                <label for="password1">Password:</label>
                <input type="password" name="password1" placeholder="need 8 to 16 charactors, special symbol could be: _ ? !" pattern="^[\w_?!]{8,16}$" 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 charactors, 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="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                <input type="submit" value="Go to email verification" id="submit">
            </form>
        </div>

        <input type="hidden" name="token_generate" id="token_generate">

        <div class="toLog">
            <?php
                echo "Have an account already? Back to <a href='login_form.php'> LOGIN </a>";
            ?>
        </div>

    </body>
</html>
```

Register Check



/public_html/register_check.php

```php
<?php
    session_start();

    require 'helper.php';

    csrf_counter();
    g_captcha_check("register_form.php");


    // 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");
    }


    // username
    $username = htmlspecialchars($_POST['username']);
    if ($username != $_POST['username']) {
        header_n_sendem("register_form.php", "Username contains invalid symbol");
    }
    if (preg_match('/^[\w_]{2,8}$/', $username) == 0) {
        header_n_sendem("register_form.php", "Username is in wrong pattern");
    }

    // password
    $password1 = htmlspecialchars($_POST['password1']);
    $password2 = htmlspecialchars($_POST['password2']);
    if (($password1 != $_POST['password1']) || ($password2 != $_POST['password2'])) {
        header_n_sendem("register_form.php", "Password contain invalid symbol");
    }
    if ($password1 != $password2) {
        header_n_sendem("register_form.php", "Two passwords are different");
    }
    // PASSWORD PATTERN
    if ((strlen($password1) < 8) || strlen($password1) > 16 ||
        !preg_match('@[A-Z]@', $password1) || !preg_match('@[a-z]@', $password1) ||
        !preg_match('@[0-9]@', $password1) || !preg_match('@[_?!]@', $password1)) {
        header_n_sendem("register_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 ($password1 == $com_pwd)  {
                header_n_sendem("register_form.php", "Password is vulnerable to Dictionary Attacks");
            }
        }
    } else {
        header_n_sendem("register_form.php", "Failed to open the pwd TXT");
    }
    fclose($handle);



    // forename & surname
    $forename = htmlspecialchars($_POST['fname']);
    $surname = htmlspecialchars($_POST['sname']);
    if ($forename != $_POST['fname']) {
        header_n_sendem("register_form.php", "Forename contains invalid symbol");
    }
    if ($surname != $_POST['sname']) {
        header_n_sendem("register_form.php", "Surname contains invalid symbol");
    }

    // phone
    $phone = $_POST['phone'];
    if ((strlen($phone) >= 10) && (strlen($phone) <= 14)) {
        $phone = preg_replace("/[^0-9]/", '', $phone);
        if ($phone != $_POST['phone']) {
            header_n_sendem("register_form.php", "Invalid phone number");
        }
    } else {
        header_n_sendem("register_form.php", "Unsupported phone number length");
    }

    // email
    $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
    if (filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
        header_n_sendem("register_form.php", "Invalid email address");
    }

    // security Q&A
    $sec_ques = $_POST['sec_ques'];
    $sec_ans = htmlspecialchars($_POST['sec_ans']);
    if ($sec_ans != $_POST['sec_ans']) {
        header_n_sendem("register_form.php", "Security answer contains invalid symbol");
    }



    // ===================================================================
    require __DIR__ . '/vendor/autoload.php';
    use PragmaRX\Google2FA\Google2FA;
    require 'database_conn.php';

```



```php
    // ==================================================================
    require __DIR__ . '/vendor/autoload.php';
    use PragmaRX\Google2FA\Google2FA;
    require 'database_conn.php';



    // get stmt1 if that username exists.
    if ($stmt1 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ?')) {
        $stmt1->bind_param('s', $username);
        $stmt1->execute();
        $stmt1->store_result();
    } else {
        header_n_sendem("register_form.php", "Cannot prepare statement 1 !");
    }

    // get stmt2 if that email exists.
    if ($stmt2 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ? AND email = ?')) {
        $stmt2->bind_param('ss', $username, $email);
        $stmt2->execute();
        $stmt2->store_result();
    } else {
        header_n_sendem("register_form.php", "Cannot prepare statement 2 !");
    }

    // check and insert a new row
    if ($stmt1->num_rows > 0) {
        header_n_sendem("register_form.php", "Username exists, please choose another one");
    }
    elseif ($stmt2->num_rows > 0) {
        header_n_sendem("register_form.php", "Email has already been used, please choose another one");
    } else {

        // Username and Email both dont exist, now can insert new systemuser
        if ($stmt1 = $conn->prepare('INSERT INTO systemuser (username, passwd, forename, surname, phone, email, activation_code, sec_ques, sec_ans, tfa_seckey, first_ip)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)')) {

            // set the params needed
            $password_hashed = password_hash($password1, PASSWORD_DEFAULT);
            $uniqid = uniqid();
            $sec_ans = password_hash($sec_ans, PASSWORD_DEFAULT);
            $google2fa = new Google2FA();
            $secret = $google2fa->generateSecretKey();
            $first_ip = getUserIpAddr();

            // pass the params and execute
            $stmt1->bind_param('sssssssss', $username, $password_hashed, $forename, $surname, $phone, $email, $uniqid, $sec_ques, $sec_ans, $secret, $first_ip);
            $stmt1->execute();

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

        } else {
            header_n_sendem("register_form.php", "Cannot prepare stmt of INSERT info !");
        }
    }

    $stmt2->close();
    $conn->close();
?>
```

Database Table

Three tables in total

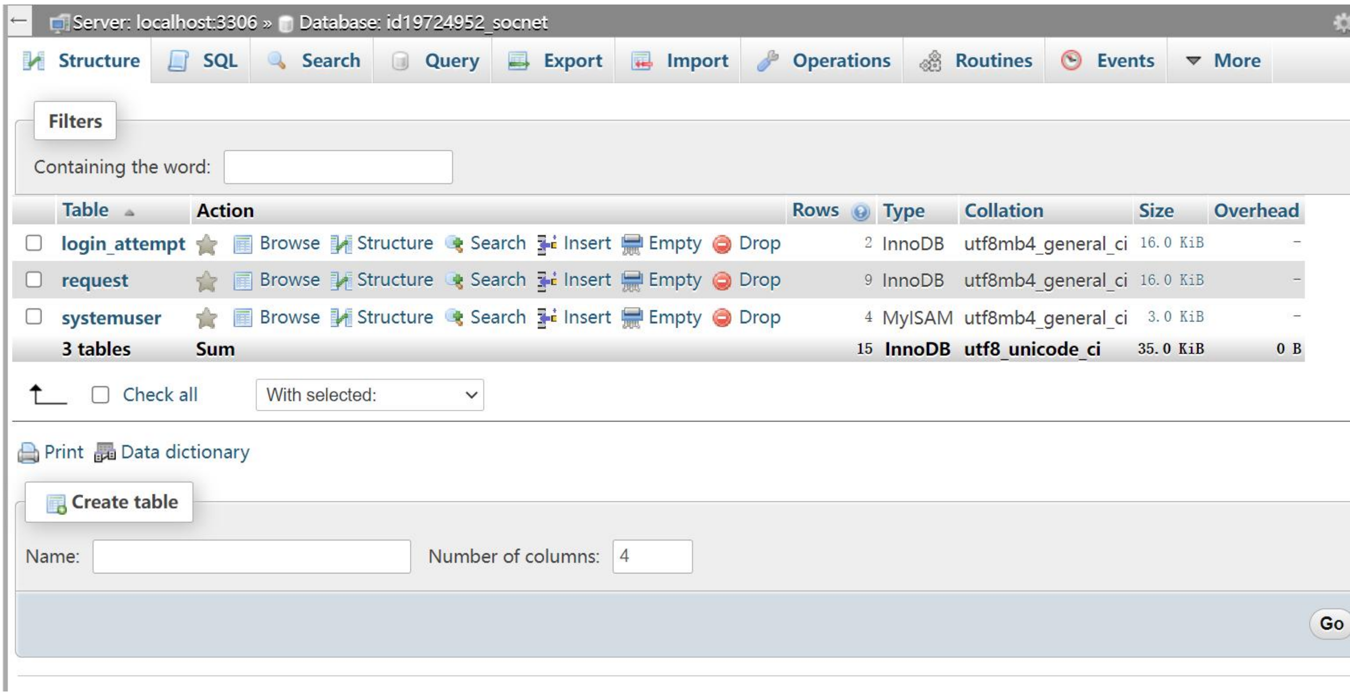
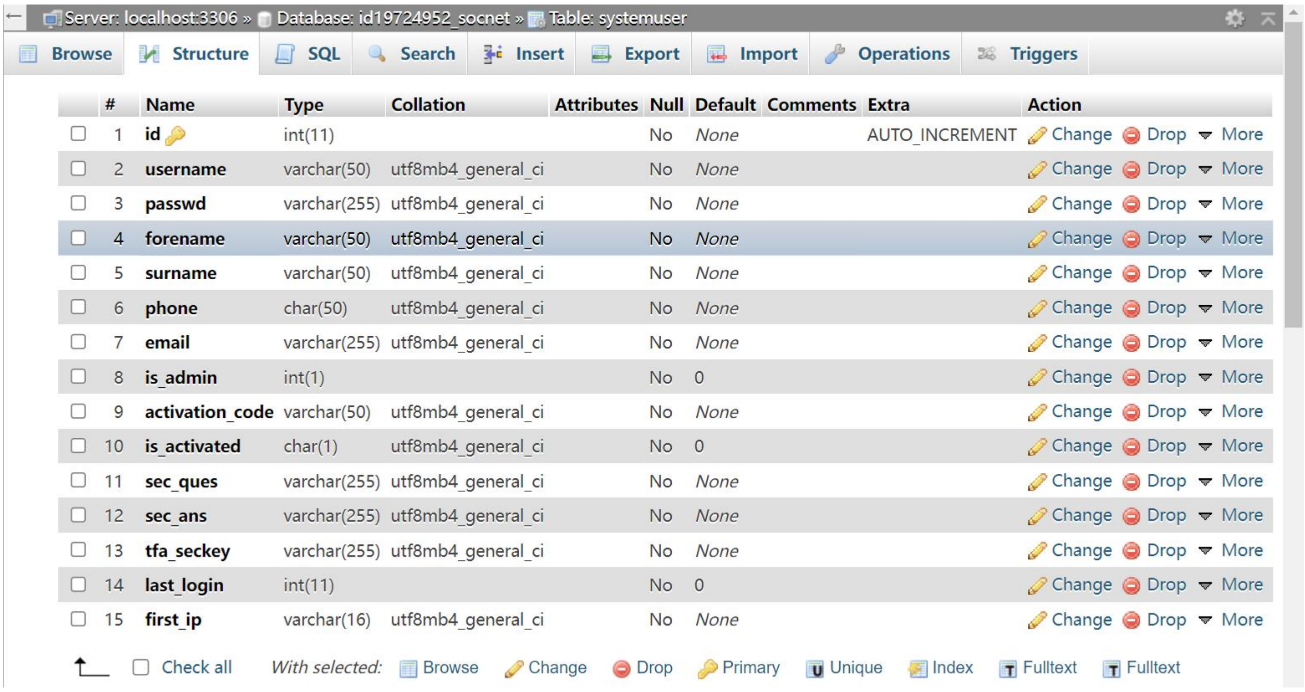


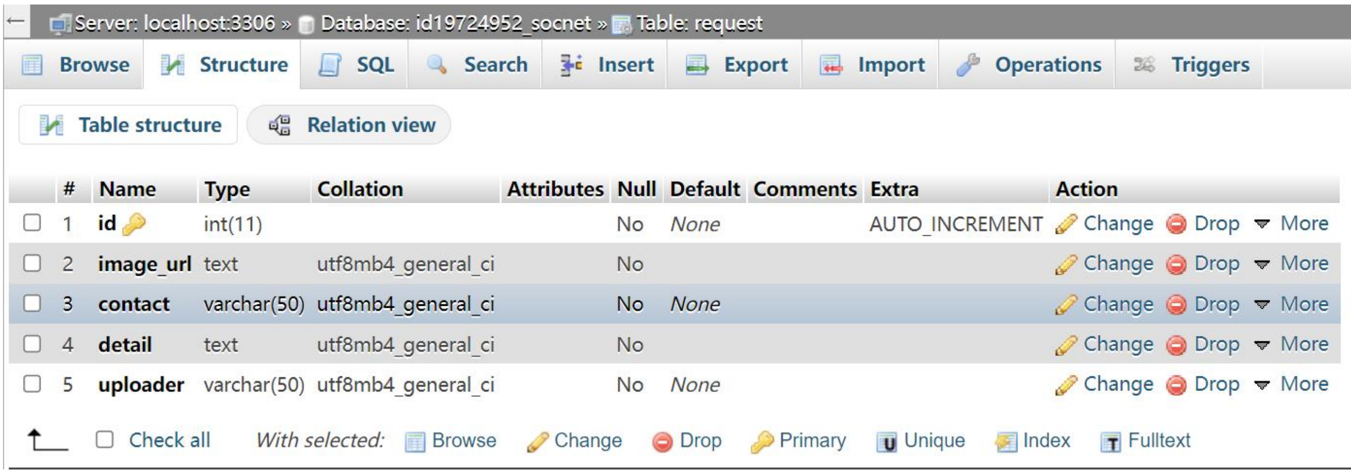
	Table	Action	Rows	Type	Collation	Size	Overhead
□	login_attempt	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	request	Browse Structure Search Insert Empty Drop	9	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	systemuser	Browse Structure Search Insert Empty Drop	4	MyISAM	utf8mb4_general_ci	3.0 KiB	-
	3 tables	Sum	15	InnoDB	utf8_unicode_ci	35.0 KiB	0 B

Table: systemuser


Server: localhost:3306 » Database: id19724952_socnet » Table: systemuser
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Triggers

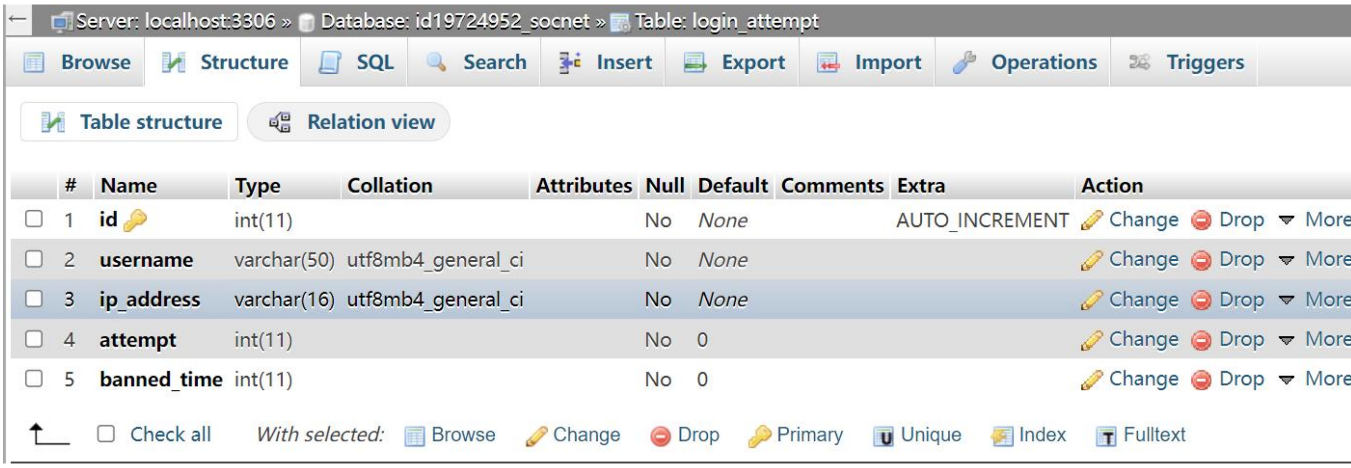
	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
□	2	username	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
□	3	passwd	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
□	4	forename	varchar(50)	utf8mb4_general_ci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

Table: request



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**image_url**	text	utf8mb4_general_ci		No				Change Drop More
□	3	**contact**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
□	4	**detail**	text	utf8mb4_general_ci		No				Change Drop More
□	5	**uploader**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More

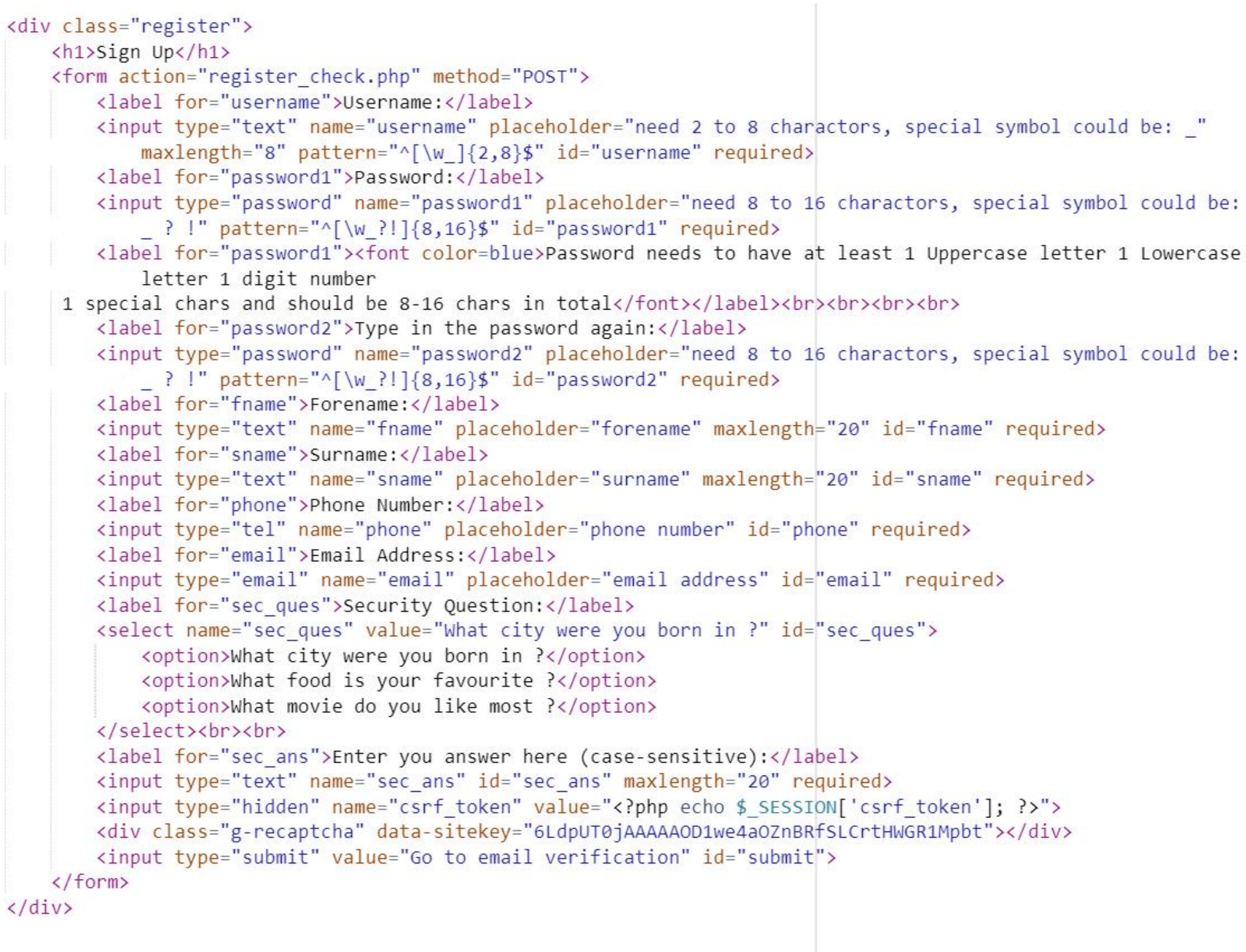
Table: login_attempt



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**username**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
□	3	**ip_address**	varchar(16)	utf8mb4_general_ci		No	*None*			Change Drop More
□	4	**attempt**	int(11)			No	0			Change Drop More
□	5	**banned_time**	int(11)			No	0			Change Drop More

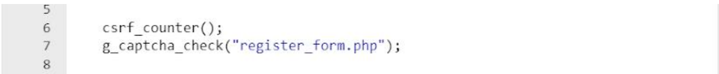
Why do you think it is secure? Use bullet points to provide your reasons and back it up with code snippet from your application.

- use “pattern”, “required” attributes in html form

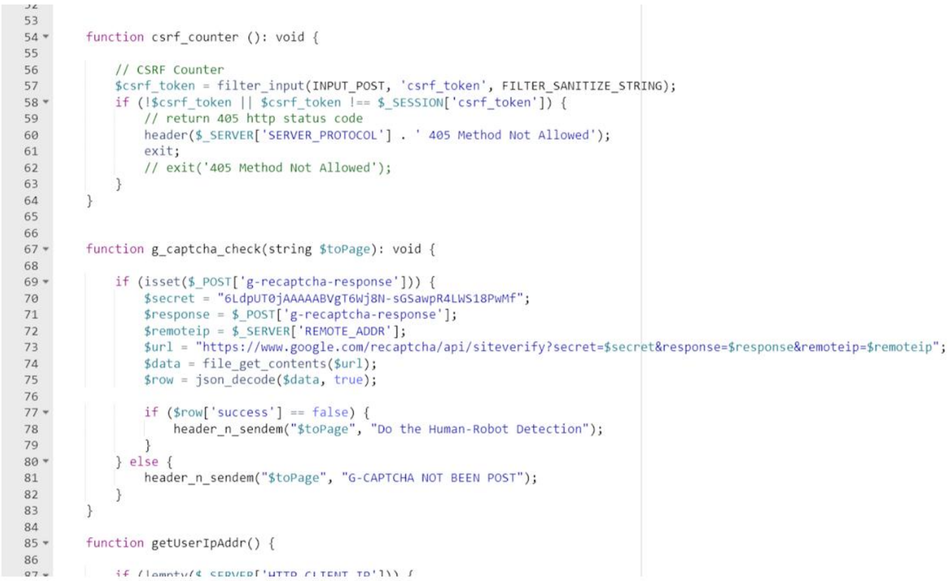


```
<div class="register">
    <h1>Sign Up</h1>
    <form action="register_check.php" method="POST">
        <label for="username">Username:</label>
        <input type="text" name="username" placeholder="need 2 to 8 charactors, special symbol could be: _"
            maxlength="8" pattern="^[\w_]{2,8}$" id="username" required>
        <label for="password1">Password:</label>
        <input type="password" name="password1" placeholder="need 8 to 16 charactors, special symbol could be:
            _ ? !" pattern="^[\w_?!]{8,16}$" 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 charactors, 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="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
        <input type="submit" value="Go to email verification" id="submit">
    </form>
</div>
```

- csrf counter-measure implemented, Google reCaptcha implemented



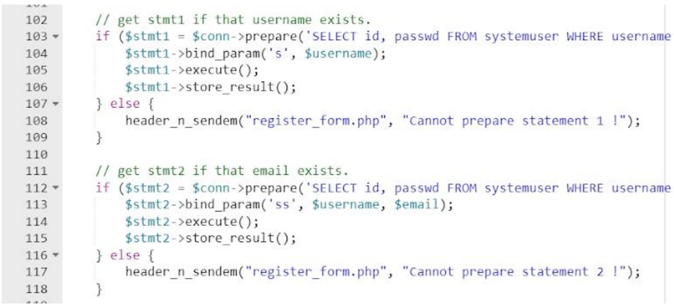
```
    csrf_counter();
    g_captcha_check("register_form.php");
```



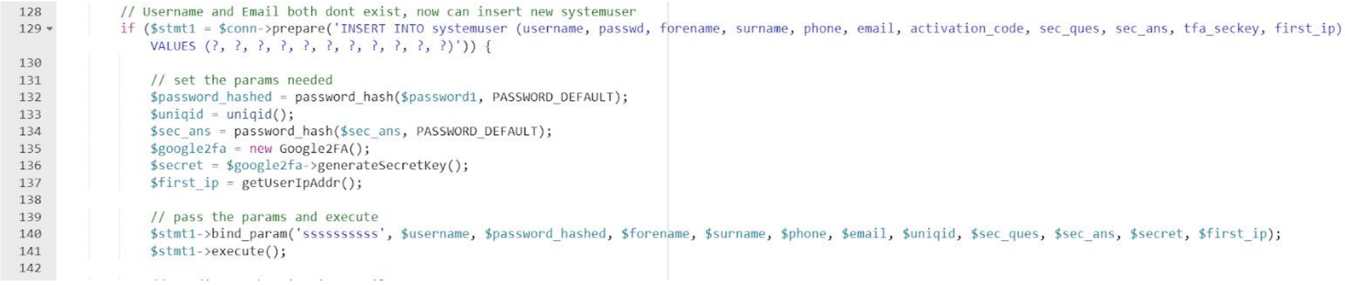
```
    function csrf_counter (): void {

        // CSRF Counter
        $csrf_token = filter_input(INPUT_POST, 'csrf_token', FILTER_SANITIZE_STRING);
        if (!$csrf_token || $csrf_token !== $_SESSION['csrf_token']) {
            // return 405 http status code
            header($_SERVER['SERVER_PROTOCOL'] . ' 405 Method Not Allowed');
            exit;
            // exit('405 Method Not Allowed');
        }
    }


    function g_captcha_check(string $toPage): void {

        if (isset($_POST['g-recaptcha-response'])) {
            $secret = "6LdpUT0jAAAAABVgT6Wj8N-sGSawpR4LWS18PwMf";
            $response = $_POST['g-recaptcha-response'];
            $remoteip = $_SERVER['REMOTE_ADDR'];
            $url = "https://www.google.com/recaptcha/api/siteverify?secret=$secret&response=$response&remoteip=$remoteip";
            $data = file_get_contents($url);
            $row = json_decode($data, true);

            if ($row['success'] == false) {
                header_n_sendem("$toPage", "Do the Human-Robot Detection");
            }
        } else {
            header_n_sendem("$toPage", "G-CAPTCHA NOT BEEN POST");
        }
    }

    function getUserIpAddr() {

        if (!empty($_SERVER['HTTP_CLIENT_IP'])) {
```

- prepare() used to prevent the SQL injection

Just an example, implemented it everywhere it fits



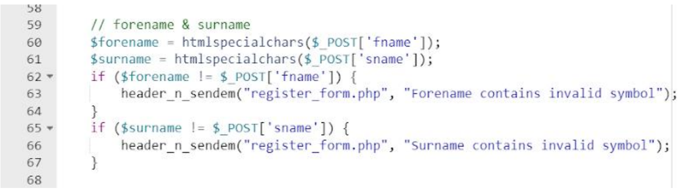
```php
// get stmt1 if that username exists.
if ($stmt1 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ?')) {
    $stmt1->bind_param('s', $username);
    $stmt1->execute();
    $stmt1->store_result();
} else {
    header_n_sendem("register_form.php", "Cannot prepare statement 1 !");
}

// get stmt2 if that email exists.
if ($stmt2 = $conn->prepare('SELECT id, passwd FROM systemuser WHERE username = ? AND email = ?')) {
    $stmt2->bind_param('ss', $username, $email);
    $stmt2->execute();
    $stmt2->store_result();
} else {
    header_n_sendem("register_form.php", "Cannot prepare statement 2 !");
}
```



```php
    // Username and Email both dont exist, now can insert new systemuser
    if ($stmt1 = $conn->prepare('INSERT INTO systemuser (username, passwd, forename, surname, phone, email, activation_code, sec_ques, sec_ans, tfa_seckey, first_ip)
        VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)')) {

        // set the params needed
        $password_hashed = password_hash($password1, PASSWORD_DEFAULT);
        $uniqid = uniqid();
        $sec_ans = password_hash($sec_ans, PASSWORD_DEFAULT);
        $google2fa = new Google2FA();
        $secret = $google2fa->generateSecretKey();
        $first_ip = getUserIpAddr();

        // pass the params and execute
        $stmt1->bind_param('ssssssssss', $username, $password_hashed, $forename, $surname, $phone, $email, $uniqid, $sec_ques, $sec_ans, $secret, $first_ip);
        $stmt1->execute();

```

- htmlspecialchars() used to skip script-inserting risks

Just an example, implemented it everywhere it fits

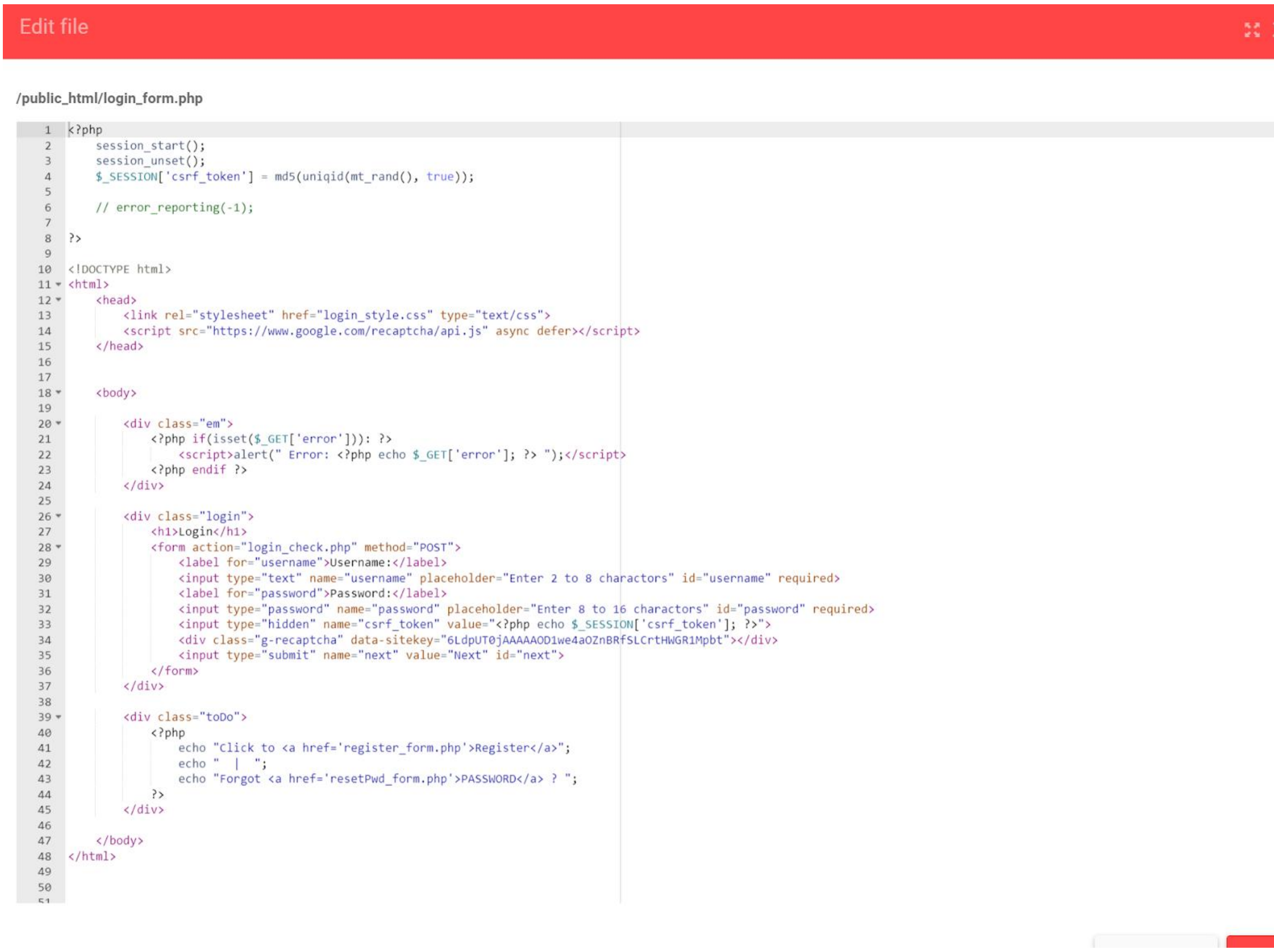


```php

// forename & surname
$forename = htmlspecialchars($_POST['fname']);
$surname = htmlspecialchars($_POST['sname']);
if ($forename != $_POST['fname']) {
    header_n_sendem("register_form.php", "Forename contains invalid symbol");
}
if ($surname != $_POST['sname']) {
    header_n_sendem("register_form.php", "Surname contains invalid symbol");
}

```

Task 2 - Develop a secure login feature.

Login feature code screenshots

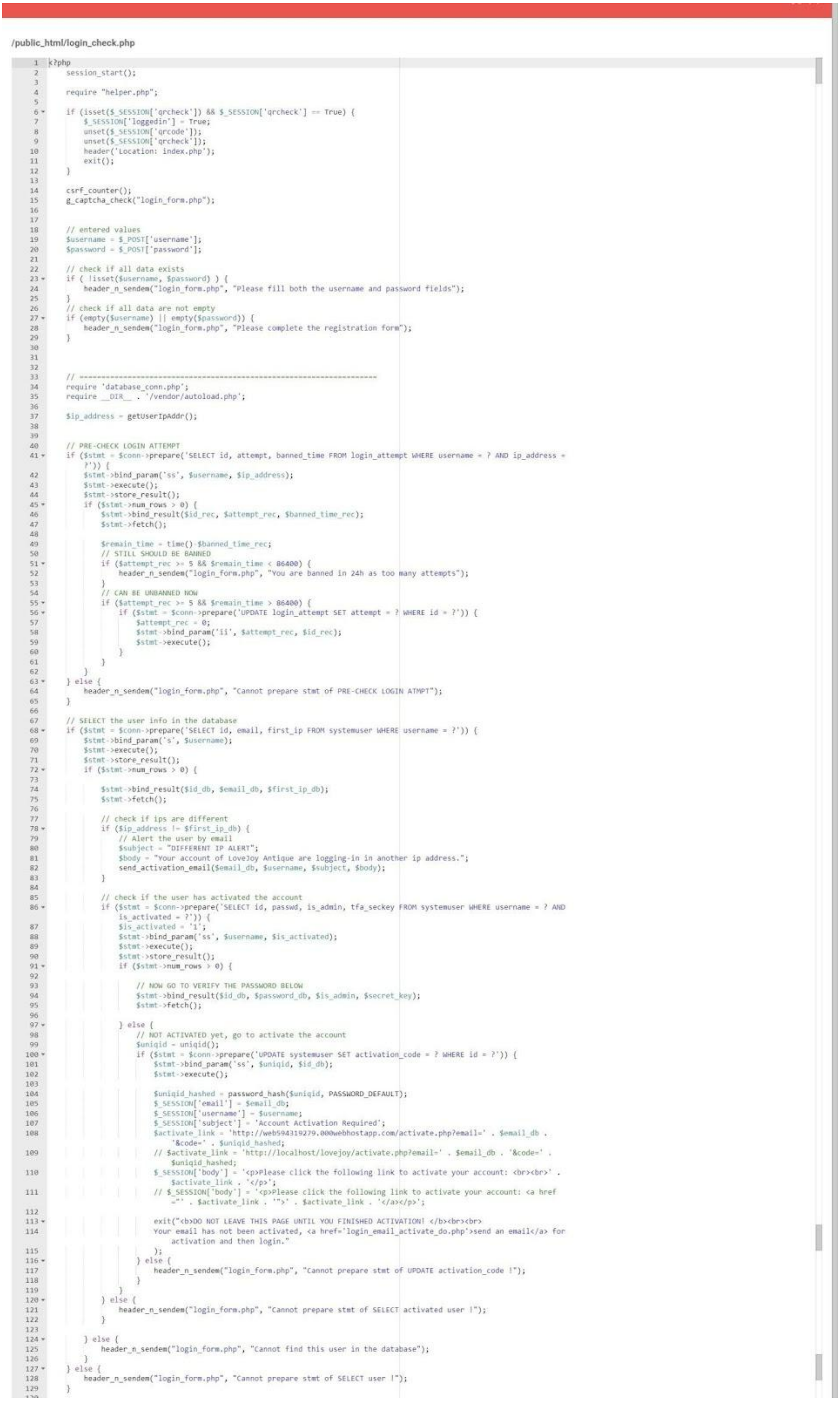
Login Form



/public_html/login_form.php

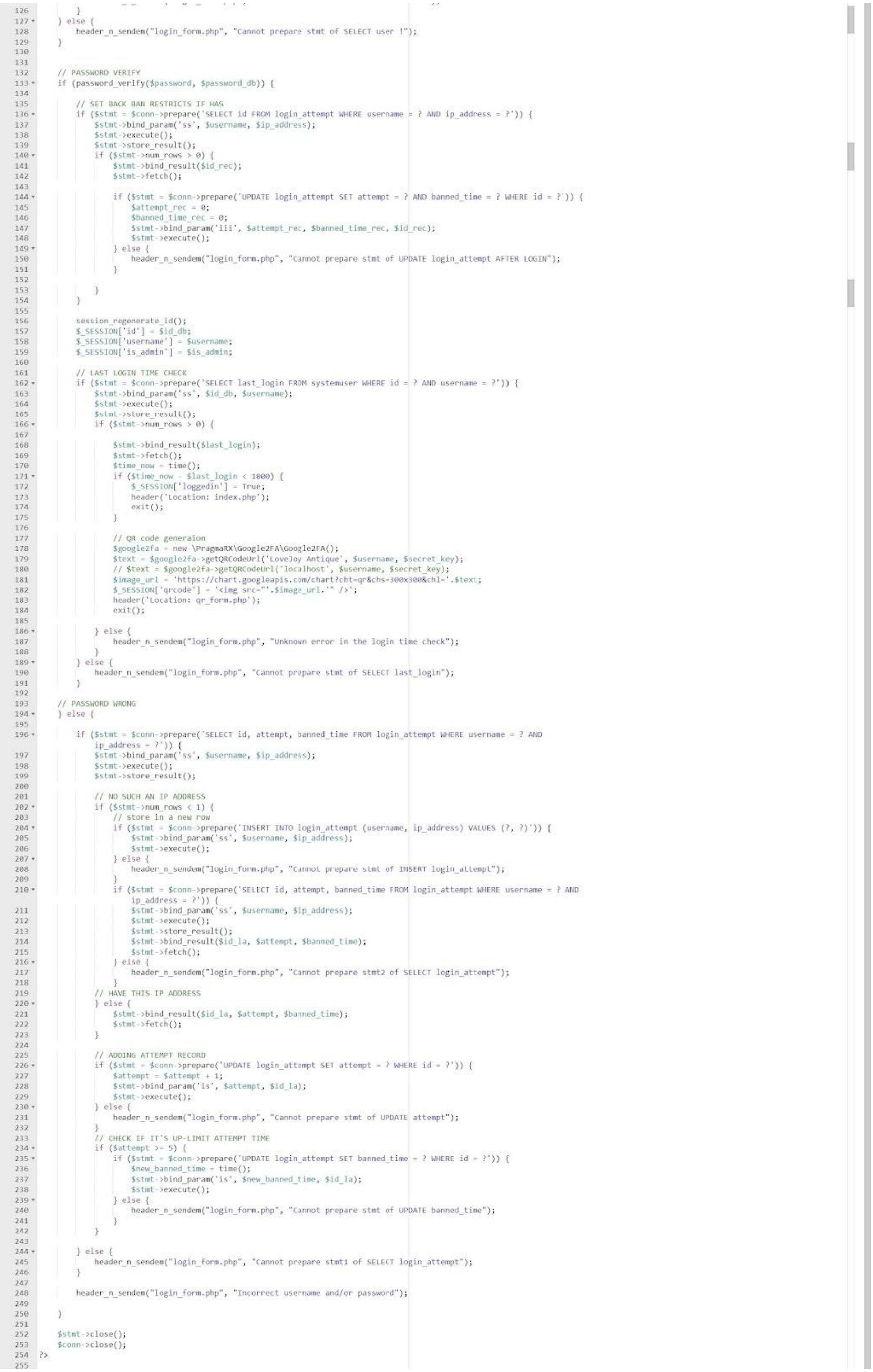
```
<?php
    session_start();
    session_unset();
    $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));

    // error_reporting(-1);

?>

<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" href="login_style.css" type="text/css">
        <script src="https://www.google.com/recaptcha/api.js" async defer></script>
    </head>


    <body>

        <div class="em">
            <?php if(isset($_GET['error'])): ?>
                <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
            <?php endif ?>
        </div>

        <div class="login">
            <h1>Login</h1>
            <form action="login_check.php" method="POST">
                <label for="username">Username:</label>
                <input type="text" name="username" placeholder="Enter 2 to 8 charactors" id="username" required>
                <label for="password">Password:</label>
                <input type="password" name="password" placeholder="Enter 8 to 16 charactors" id="password" required>
                <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                <input type="submit" name="next" value="Next" id="next">
            </form>
        </div>

        <div class="toDo">
            <?php
                echo "Click to <a href='register_form.php'>Register</a>";
                echo "  |  ";
                echo "Forgot <a href='resetPwd_form.php'>PASSWORD</a> ? ";
            ?>
        </div>

    </body>
</html>
```

Login Check



/public_html/login_check.php

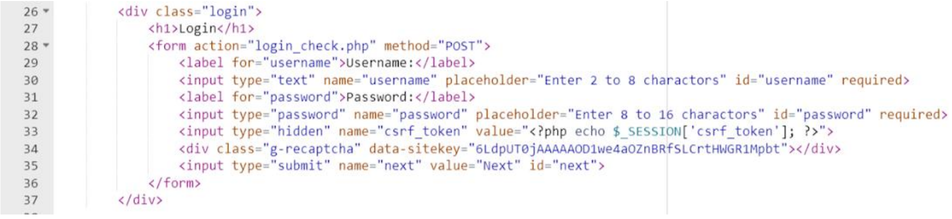
```php
<?php
    session_start();

    require "helper.php";

    if (isset($_SESSION['qrcheck']) && $_SESSION['qrcheck'] == True) {
        $_SESSION['loggedin'] = True;
        unset($_SESSION['qrcode']);
        unset($_SESSION['qrcheck']);
        header('Location: index.php');
        exit();
    }

    csrf_counter();
    g_captcha_check("login_form.php");


    // entered values
    $username = $_POST['username'];
    $password = $_POST['password'];

    // check if all data exists
    if ( !isset($username, $password) ) {
        header_n_sendem("login_form.php", "Please fill both the username and password fields");
    }
    // check if all data are not empty
    if (empty($username) || empty($password)) {
        header_n_sendem("login_form.php", "Please complete the registration form");
    }



    // ====================================================================
    require 'database_conn.php';
    require __DIR__ . '/vendor/autoload.php';

    $ip_address = getUserIpAddr();


    // PRE-CHECK LOGIN ATTEMPT
    if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address =
        ?')) {
        $stmt->bind_param('ss', $username, $ip_address);
        $stmt->execute();
        $stmt->store_result();
        if ($stmt->num_rows > 0) {
            $stmt->bind_result($id_rec, $attempt_rec, $banned_time_rec);
            $stmt->fetch();

            $remain_time = time()-$banned_time_rec;
            // STILL SHOULD BE BANNED
            if ($attempt_rec >= 5 && $remain_time < 86400) {
                header_n_sendem("login_form.php", "You are banned in 24h as too many attempts");
            }
            // CAN BE UNBANNED NOW
            if ($attempt_rec >= 5 && $remain_time > 86400) {
                if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
                    $attempt_rec = 0;
                    $stmt->bind_param('ii', $attempt_rec, $id_rec);
                    $stmt->execute();
                }
            }
        }
    } else {
        header_n_sendem("login_form.php", "Cannot prepare stmt of PRE-CHECK LOGIN ATMPT");
    }

    // SELECT the user info in the database
    if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
        $stmt->bind_param('s', $username);
        $stmt->execute();
        $stmt->store_result();
        if ($stmt->num_rows > 0) {

            $stmt->bind_result($id_db, $email_db, $first_ip_db);
            $stmt->fetch();

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

            // check if the user has activated the account
            if ($stmt = $conn->prepare('SELECT id, passwd, is_admin, tfa_seckey FROM systemuser WHERE username = ? AND
                is_activated = ?')) {
                $is_activated = '1';
                $stmt->bind_param('ss', $username, $is_activated);
                $stmt->execute();
                $stmt->store_result();
                if ($stmt->num_rows > 0) {

                    // NOW GO TO VERIFY THE PASSWORD BELOW
                    $stmt->bind_result($id_db, $password_db, $is_admin, $secret_key);
                    $stmt->fetch();

                } 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></p>';

                        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 !");
                    }
                }
            } else {
                header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT activated user !");
            }

        } else {
            header_n_sendem("login_form.php", "Cannot find this user in the database");
        }
    } else {
        header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT user !");
    }
```



```php
        }
    } else {
        header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT user !");
    }


    // PASSWORD VERIFY
    if (password_verify($password, $password_db)) {

        // SET BACK BAN RESTRICTS IF HAS
        if ($stmt = $conn->prepare('SELECT id FROM login_attempt WHERE username = ? AND ip_address = ?')) {
            $stmt->bind_param('ss', $username, $ip_address);
            $stmt->execute();
            $stmt->store_result();
            if ($stmt->num_rows > 0) {
                $stmt->bind_result($id_rec);
                $stmt->fetch();

                if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? AND banned_time = ? WHERE id = ?')) {
                    $attempt_rec = 0;
                    $banned_time_rec = 0;
                    $stmt->bind_param('iii', $attempt_rec, $banned_time_rec, $id_rec);
                    $stmt->execute();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE login_attempt AFTER LOGIN");
                }

            }
        }

        session_regenerate_id();
        $_SESSION['id'] = $id_db;
        $_SESSION['username'] = $username;
        $_SESSION['is_admin'] = $is_admin;

        // LAST LOGIN TIME CHECK
        if ($stmt = $conn->prepare('SELECT last_login FROM systemuser WHERE id = ? AND username = ?')) {
            $stmt->bind_param('ss', $id_db, $username);
            $stmt->execute();
            $stmt->store_result();
            if ($stmt->num_rows > 0) {

                $stmt->bind_result($last_login);
                $stmt->fetch();
                $time_now = time();
                if ($time_now - $last_login < 1800) {
                    $_SESSION['loggedin'] = True;
                    header('Location: index.php');
                    exit();
                }

                // QR code generaion
                $google2fa = new \PragmaRX\Google2FA\Google2FA();
                $text = $google2fa->getQRCodeUrl('LoveJoy Antique', $username, $secret_key);
                // $text = $google2fa->getQRCodeUrl('localhost', $username, $secret_key);
                $image_url = 'https://chart.googleapis.com/chart?cht=qr&chs=300x300&chl='.$text;
                $_SESSION['qrcode'] = '<img src="'.$image_url.'" />';
                header('Location: qr_form.php');
                exit();

            } else {
                header_n_sendem("login_form.php", "Unknown error in the login time check");
            }
        } else {
            header_n_sendem("login_form.php", "Cannot prepare stmt of SELECT last_login");
        }

    // PASSWORD WRONG
    } else {

        if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND
            ip_address = ?')) {
            $stmt->bind_param('ss', $username, $ip_address);
            $stmt->execute();
            $stmt->store_result();

            // NO SUCH AN IP ADDRESS
            if ($stmt->num_rows < 1) {
                // store in a new row
                if ($stmt = $conn->prepare('INSERT INTO login_attempt (username, ip_address) VALUES (?, ?)')) {
                    $stmt->bind_param('ss', $username, $ip_address);
                    $stmt->execute();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt of INSERT login_attempt");
                }
                if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND
                    ip_address = ?')) {
                    $stmt->bind_param('ss', $username, $ip_address);
                    $stmt->execute();
                    $stmt->store_result();
                    $stmt->bind_result($id_la, $attempt, $banned_time);
                    $stmt->fetch();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt2 of SELECT login_attempt");
                }
            // HAVE THIS IP ADDRESS
            } else {
                $stmt->bind_result($id_la, $attempt, $banned_time);
                $stmt->fetch();
            }

            // ADDING ATTEMPT RECORD
            if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
                $attempt = $attempt + 1;
                $stmt->bind_param('is', $attempt, $id_la);
                $stmt->execute();
            } else {
                header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE attempt");
            }
            // CHECK IF IT'S UP-LIMIT ATTEMPT TIME
            if ($attempt >= 5) {
                if ($stmt = $conn->prepare('UPDATE login_attempt SET banned_time = ? WHERE id = ?')) {
                    $new_banned_time = time();
                    $stmt->bind_param('is', $new_banned_time, $id_la);
                    $stmt->execute();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE banned_time");
                }
            }

        } else {
            header_n_sendem("login_form.php", "Cannot prepare stmt1 of SELECT login_attempt");
        }

        header_n_sendem("login_form.php", "Incorrect username and/or password");

    }

    $stmt->close();
    $conn->close();
?>
```

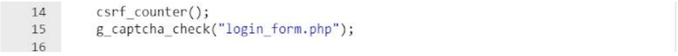
Why do you think it is secure? Use bullet points to provide your reasons and back it up code snippet from your application.

- use “pattern”, “required” attributes in html form



```
<div class="login">
    <h1>Login</h1>
    <form action="login_check.php" method="POST">
        <label for="username">Username:</label>
        <input type="text" name="username" placeholder="Enter 2 to 8 charactors" id="username" required>
        <label for="password">Password:</label>
        <input type="password" name="password" placeholder="Enter 8 to 16 charactors" id="password" required>
        <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
        <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
        <input type="submit" name="next" value="Next" id="next">
    </form>
</div>
```

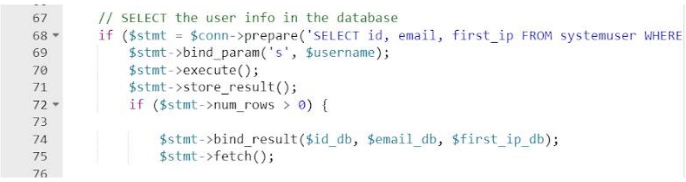
- csrf counter-measure implemented, Google reCaptcha implemented



```
csrf_counter();
g_captcha_check("login_form.php");
```

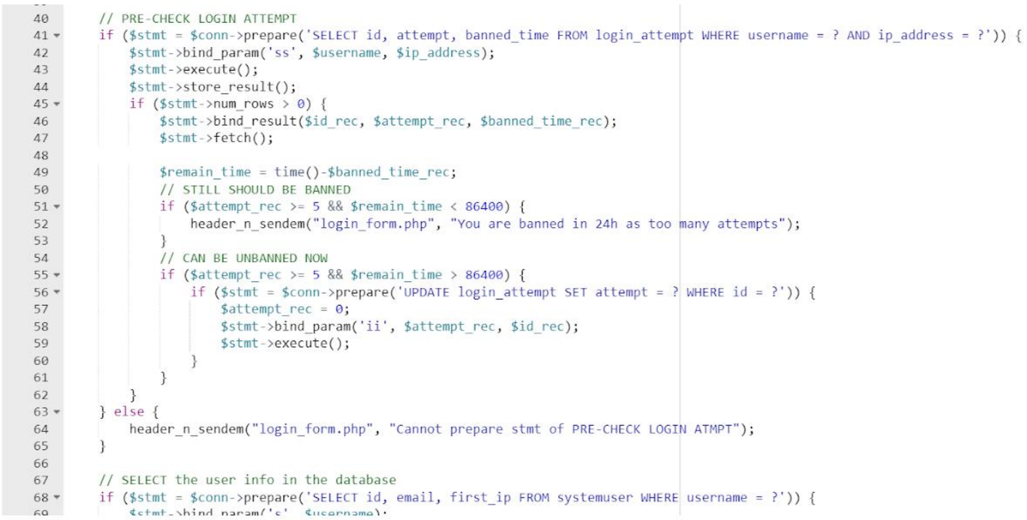
- prepare() used to prevent the SQL injection

Just an example, implemented it everywhere it fit



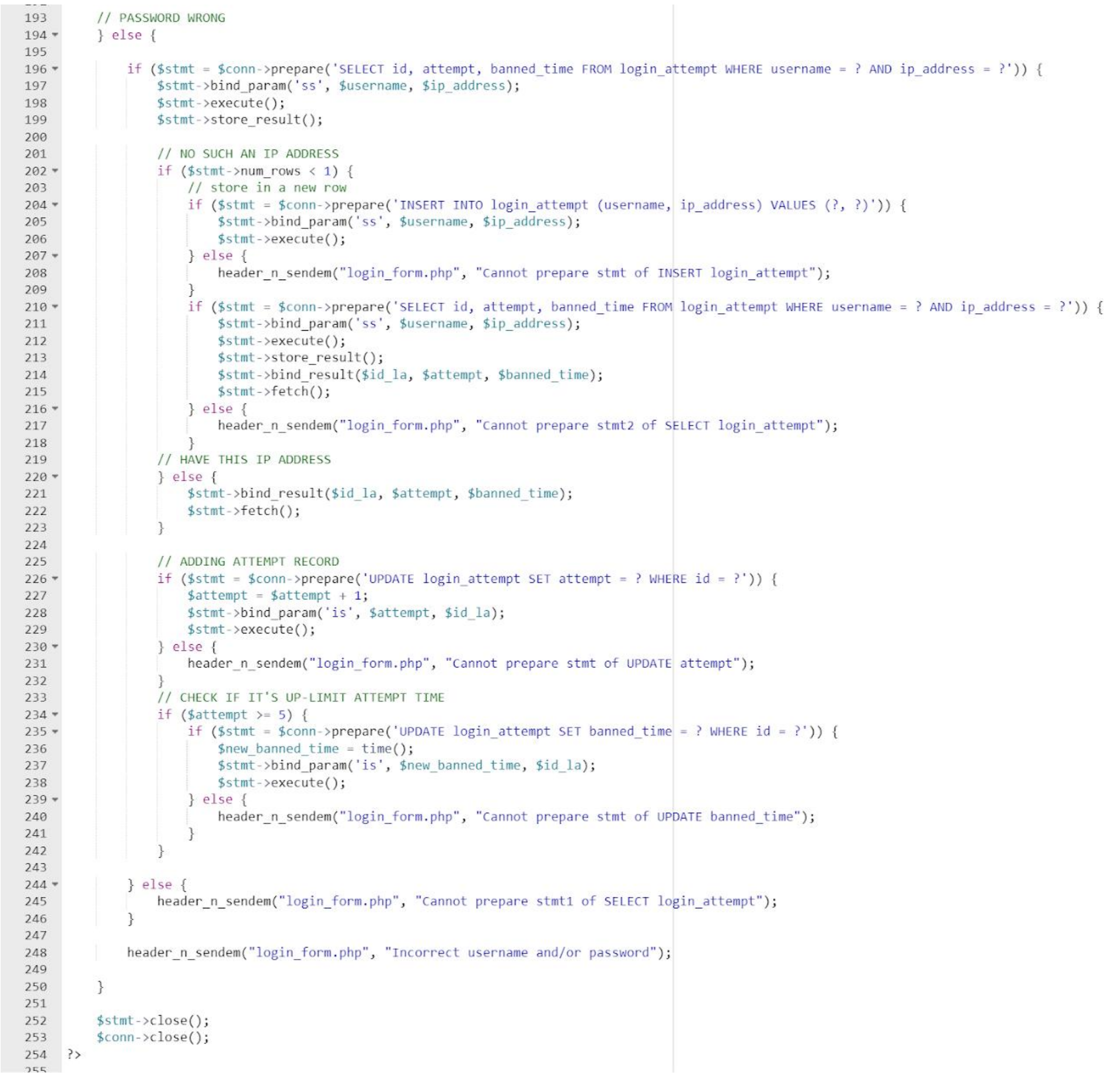
```
// SELECT the user info in the database
if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
    $stmt->bind_param('s', $username);
    $stmt->execute();
    $stmt->store_result();
    if ($stmt->num_rows > 0) {

        $stmt->bind_result($id_db, $email_db, $first_ip_db);
        $stmt->fetch();
```

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



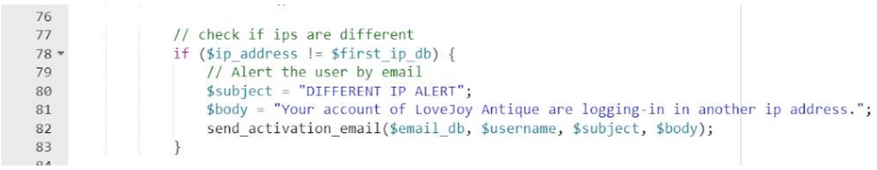
```
// PRE-CHECK LOGIN ATTEMPT
if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
    $stmt->bind_param('ss', $username, $ip_address);
    $stmt->execute();
    $stmt->store_result();
    if ($stmt->num_rows > 0) {
        $stmt->bind_result($id_rec, $attempt_rec, $banned_time_rec);
        $stmt->fetch();

        $remain_time = time()-$banned_time_rec;
        // STILL SHOULD BE BANNED
        if ($attempt_rec >= 5 && $remain_time < 86400) {
            header_n_sendem("login_form.php", "You are banned in 24h as too many attempts");
        }
        // CAN BE UNBANNED NOW
        if ($attempt_rec >= 5 && $remain_time > 86400) {
            if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
                $attempt_rec = 0;
                $stmt->bind_param('ii', $attempt_rec, $id_rec);
                $stmt->execute();
            }
        }
    }
} else {
    header_n_sendem("login_form.php", "Cannot prepare stmt of PRE-CHECK LOGIN ATMPT");
}

// SELECT the user info in the database
if ($stmt = $conn->prepare('SELECT id, email, first_ip FROM systemuser WHERE username = ?')) {
    $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



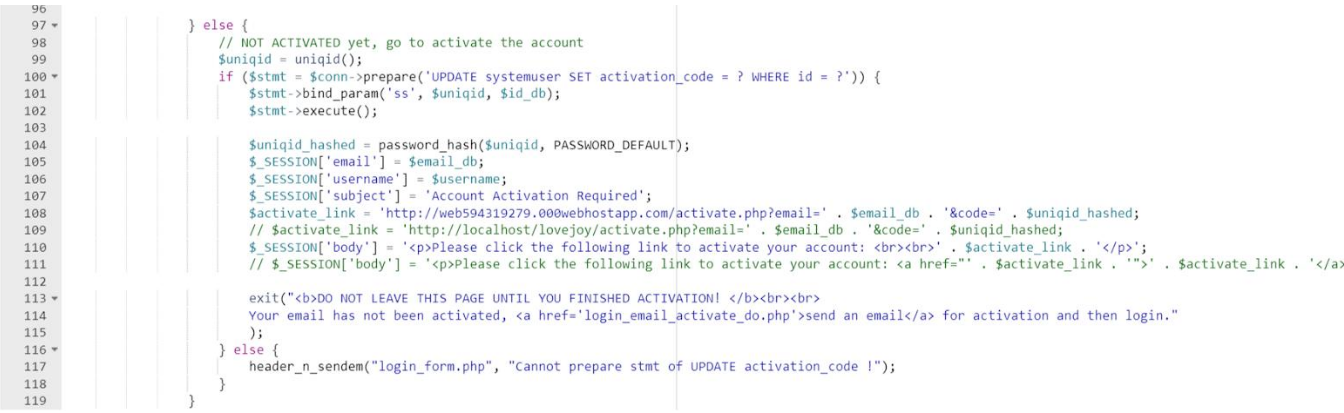
```php
    // PASSWORD WRONG
    } else {

        if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
            $stmt->bind_param('ss', $username, $ip_address);
            $stmt->execute();
            $stmt->store_result();

            // NO SUCH AN IP ADDRESS
            if ($stmt->num_rows < 1) {
                // store in a new row
                if ($stmt = $conn->prepare('INSERT INTO login_attempt (username, ip_address) VALUES (?, ?)')) {
                    $stmt->bind_param('ss', $username, $ip_address);
                    $stmt->execute();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt of INSERT login_attempt");
                }
                if ($stmt = $conn->prepare('SELECT id, attempt, banned_time FROM login_attempt WHERE username = ? AND ip_address = ?')) {
                    $stmt->bind_param('ss', $username, $ip_address);
                    $stmt->execute();
                    $stmt->store_result();
                    $stmt->bind_result($id_la, $attempt, $banned_time);
                    $stmt->fetch();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt2 of SELECT login_attempt");
                }
            // HAVE THIS IP ADDRESS
            } else {
                $stmt->bind_result($id_la, $attempt, $banned_time);
                $stmt->fetch();
            }

            // ADDING ATTEMPT RECORD
            if ($stmt = $conn->prepare('UPDATE login_attempt SET attempt = ? WHERE id = ?')) {
                $attempt = $attempt + 1;
                $stmt->bind_param('is', $attempt, $id_la);
                $stmt->execute();
            } else {
                header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE attempt");
            }
            // CHECK IF IT'S UP-LIMIT ATTEMPT TIME
            if ($attempt >= 5) {
                if ($stmt = $conn->prepare('UPDATE login_attempt SET banned_time = ? WHERE id = ?')) {
                    $new_banned_time = time();
                    $stmt->bind_param('is', $new_banned_time, $id_la);
                    $stmt->execute();
                } else {
                    header_n_sendem("login_form.php", "Cannot prepare stmt of UPDATE banned_time");
                }
            }

        } else {
            header_n_sendem("login_form.php", "Cannot prepare stmt1 of SELECT login_attempt");
        }

        header_n_sendem("login_form.php", "Incorrect username and/or password");

    }

    $stmt->close();
    $conn->close();
?>
```

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



```php
            // check if ips are different
            if ($ip_address != $first_ip_db) {
                // Alert the user by email
                $subject = "DIFFERENT IP ALERT";
                $body = "Your account of LoveJoy Antique are logging-in in another ip address.";
                send_activation_email($email_db, $username, $subject, $body);
            }
```

- user have to activated their email before log in



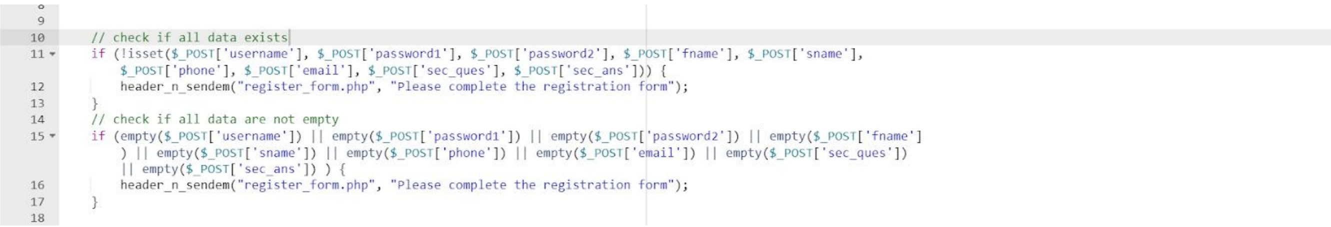
```php
            } else {
                // NOT ACTIVATED yet, go to activate the account
                $uniqid = uniqid();
                if ($stmt = $conn->prepare('UPDATE systemuser SET activation_code = ? WHERE id = ?')) {
                    $stmt->bind_param('ss', $uniqid, $id_db);
                    $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



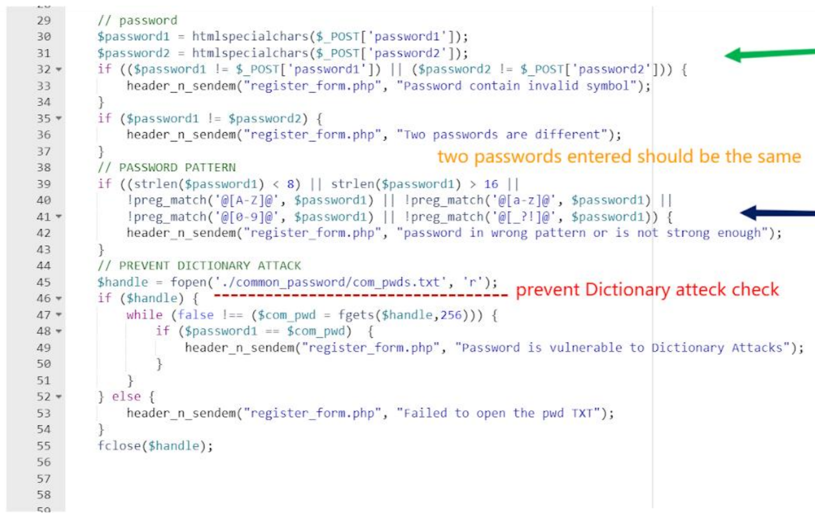
```php
    // check if all data exists
    if (!isset($_POST['username'], $_POST['password1'], $_POST['password2'], $_POST['fname'], $_POST['sname'],
        $_POST['phone'], $_POST['email'], $_POST['sec_ques'], $_POST['sec_ans'])) {
        header_n_sendem("register_form.php", "Please complete the registration form");
    }
    // check if all data are not empty
    if (empty($_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



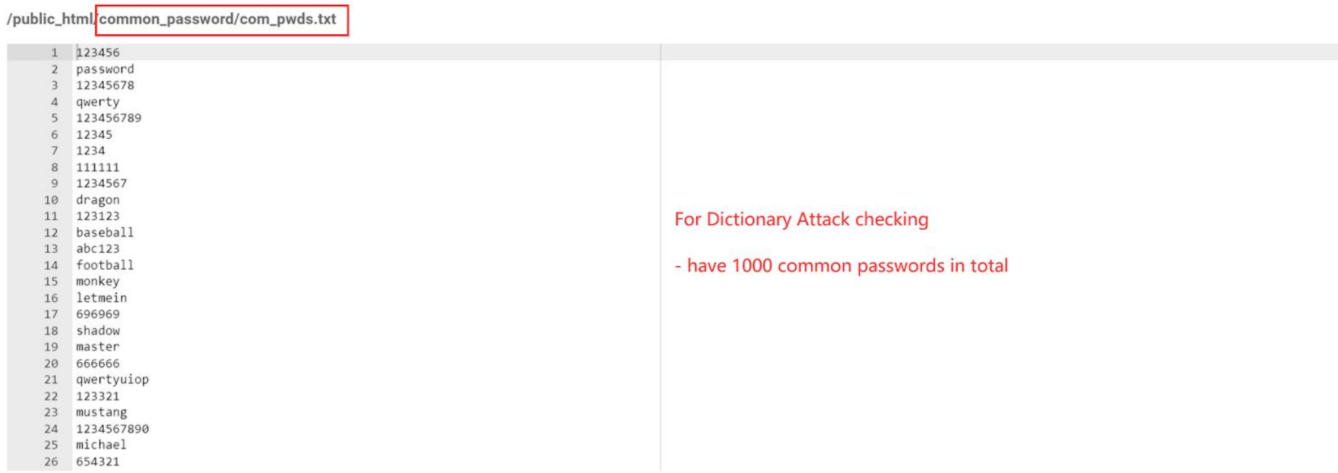
```
    // password
    $password1 = htmlspecialchars($_POST['password1']);
    $password2 = htmlspecialchars($_POST['password2']);
    if (($password1 != $_POST['password1']) || ($password2 != $_POST['password2'])) {
        header_n_sendem("register_form.php", "Password contain invalid symbol");
    }
    if ($password1 != $password2) {
        header_n_sendem("register_form.php", "Two passwords are different");
    }
    // PASSWORD PATTERN
    if ((strlen($password1) < 8) || strlen($password1) > 16 ||
        !preg_match('@[A-Z]@', $password1) || !preg_match('@[a-z]@', $password1) ||
        !preg_match('@[0-9]@', $password1) || !preg_match('@[_?!]@', $password1)) {
        header_n_sendem("register_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 ($password1 == $com_pwd)  {
                header_n_sendem("register_form.php", "Password is vulnerable to Dictionary Attacks");
            }
        }
    } else {
        header_n_sendem("register_form.php", "Failed to open the pwd TXT");
    }
    fclose($handle);
```

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_pwds.txt

```
123456
password
12345678
qwerty
123456789
12345
1234
111111
1234567
dragon
123123
baseball
abc123
football
monkey
letmein
696969
shadow
master
666666
qwertyuiop
123321
mustang
1234567890
michael
654321
```

For Dictionary Attack checking

- have 1000 common passwords in total

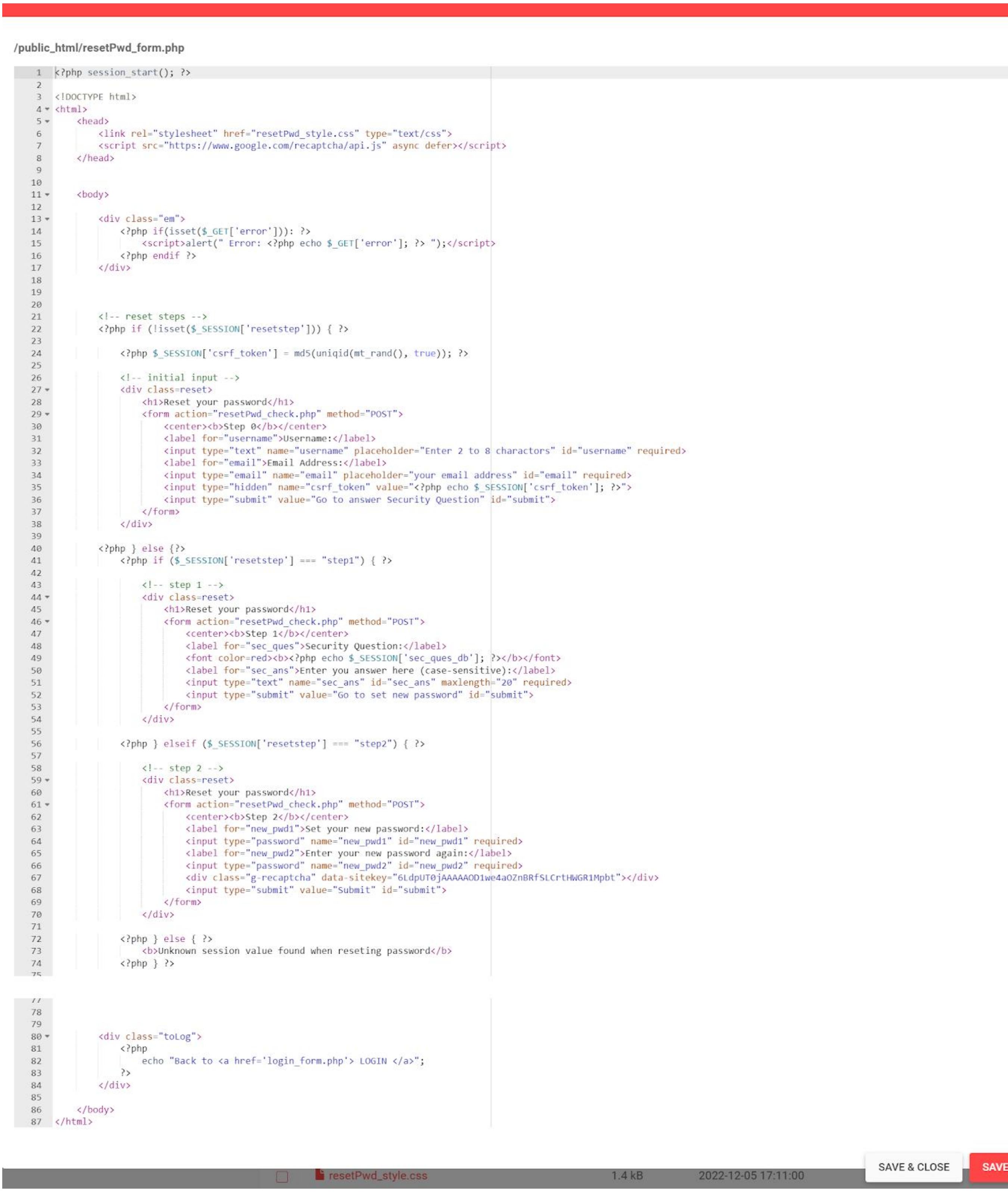
Password Recovery

- Include 3 steps:

- - enter username and email

- - answer the security question

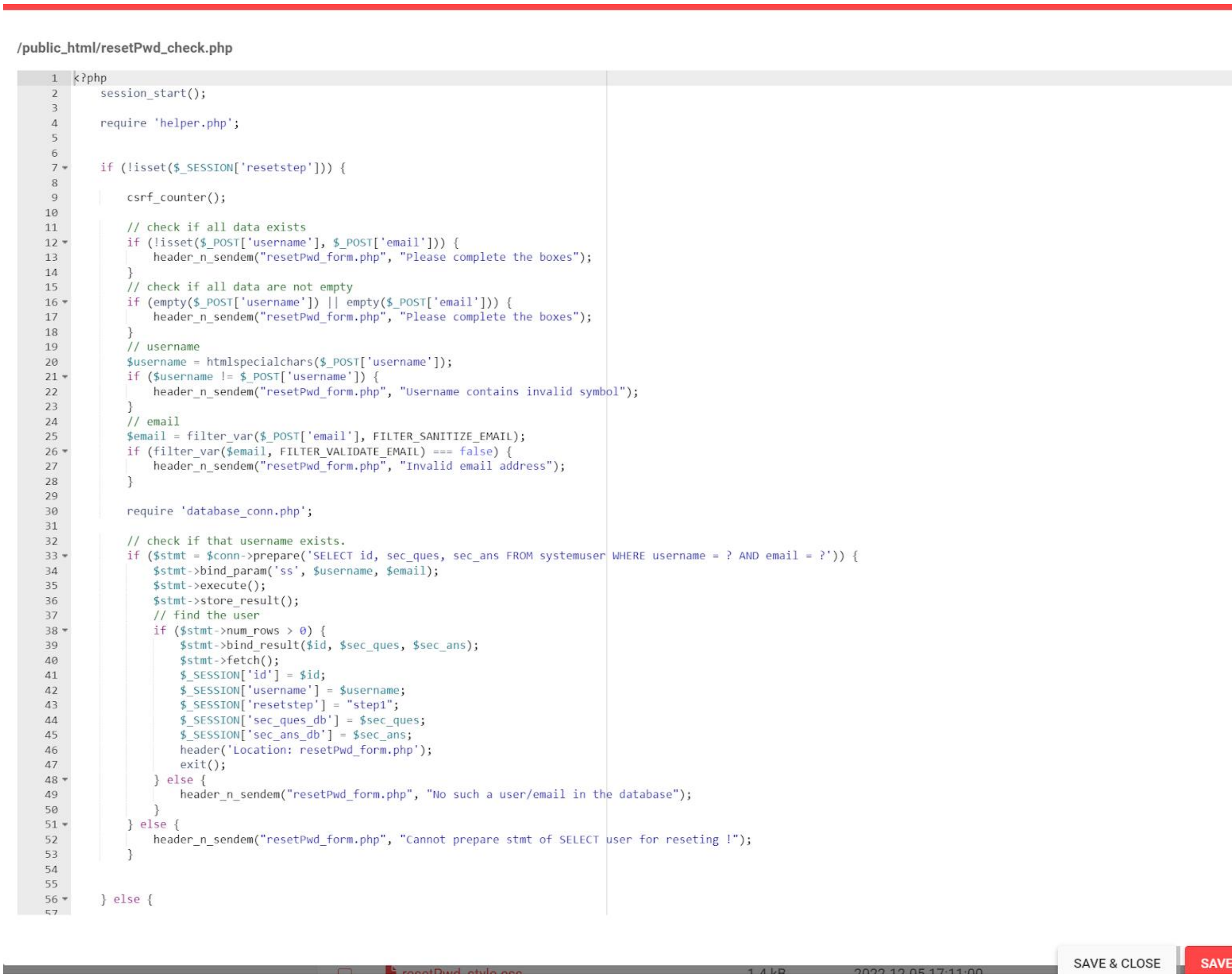
- - set the new password



/public_html/resetPwd_form.php

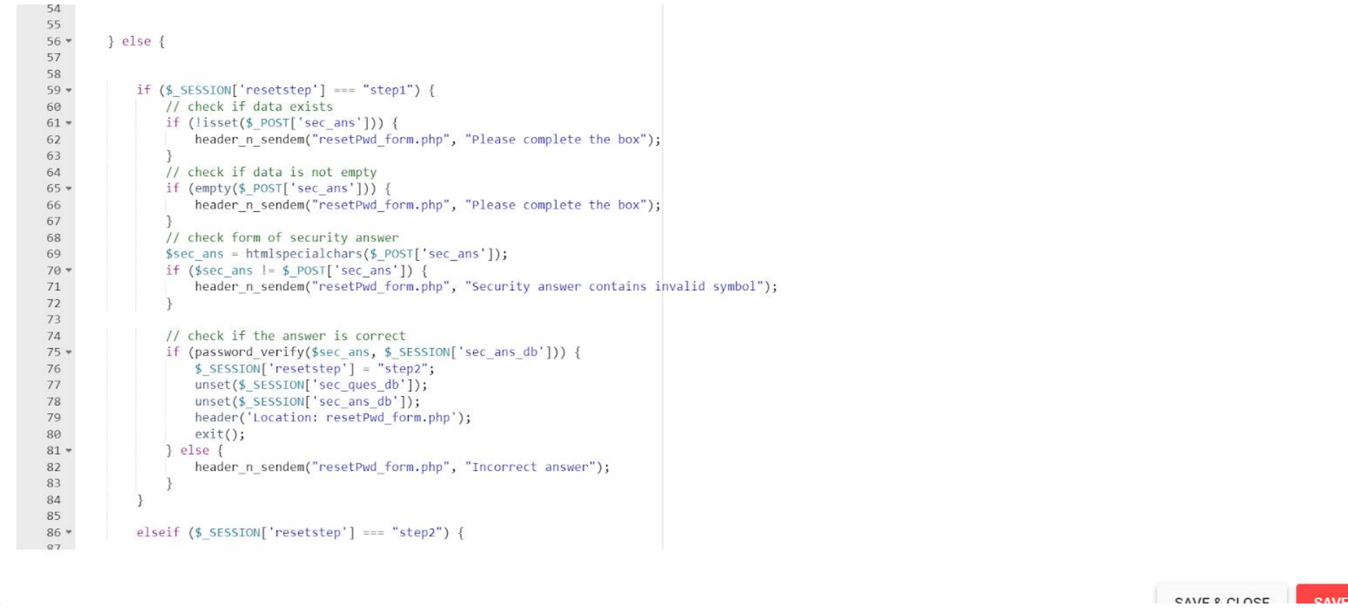
```
<?php session_start(); ?>

<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" href="resetPwd_style.css" type="text/css">
        <script src="https://www.google.com/recaptcha/api.js" async defer></script>
    </head>


    <body>

        <div class="em">
            <?php if(isset($_GET['error'])): ?>
                <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
            <?php endif ?>
        </div>



        <!-- reset steps -->
        <?php if (!isset($_SESSION['resetstep'])) { ?>

            <?php $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true)); ?>

            <!-- initial input -->
            <div class=reset>
                <h1>Reset your password</h1>
                <form action="resetPwd_check.php" method="POST">
                    <center><b>Step 0</b></center>
                    <label for="username">Username:</label>
                    <input type="text" name="username" placeholder="Enter 2 to 8 charactors" id="username" required>
                    <label for="email">Email Address:</label>
                    <input type="email" name="email" placeholder="your email address" id="email" required>
                    <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                    <input type="submit" value="Go to answer Security Question" id="submit">
                </form>
            </div>

        <?php } else {?>
            <?php if ($_SESSION['resetstep'] === "step1") { ?>

                <!-- step 1 -->
                <div class=reset>
                    <h1>Reset your password</h1>
                    <form action="resetPwd_check.php" method="POST">
                        <center><b>Step 1</b></center>
                        <label for="sec_ques">Security Question:</label>
                        <font color=red><b><?php echo $_SESSION['sec_ques_db']; ?></b></font>
                        <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="submit" value="Go to set new password" id="submit">
                    </form>
                </div>

            <?php } elseif ($_SESSION['resetstep'] === "step2") { ?>

                <!-- step 2 -->
                <div class=reset>
                    <h1>Reset your password</h1>
                    <form action="resetPwd_check.php" method="POST">
                        <center><b>Step 2</b></center>
                        <label for="new_pwd1">Set your new password:</label>
                        <input type="password" name="new_pwd1" id="new_pwd1" required>
                        <label for="new_pwd2">Enter your new password again:</label>
                        <input type="password" name="new_pwd2" id="new_pwd2" required>
                        <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                        <input type="submit" value="Submit" id="submit">
                    </form>
                </div>

            <?php } else { ?>
                <b>Unknown session value found when reseting password</b>
            <?php } ?>




        <div class="toLog">
            <?php
                echo "Back to <a href='login_form.php'> LOGIN </a>";
            ?>
        </div>

    </body>
</html>
```

- check of entering username and email


/public_html/resetPwd_check.php

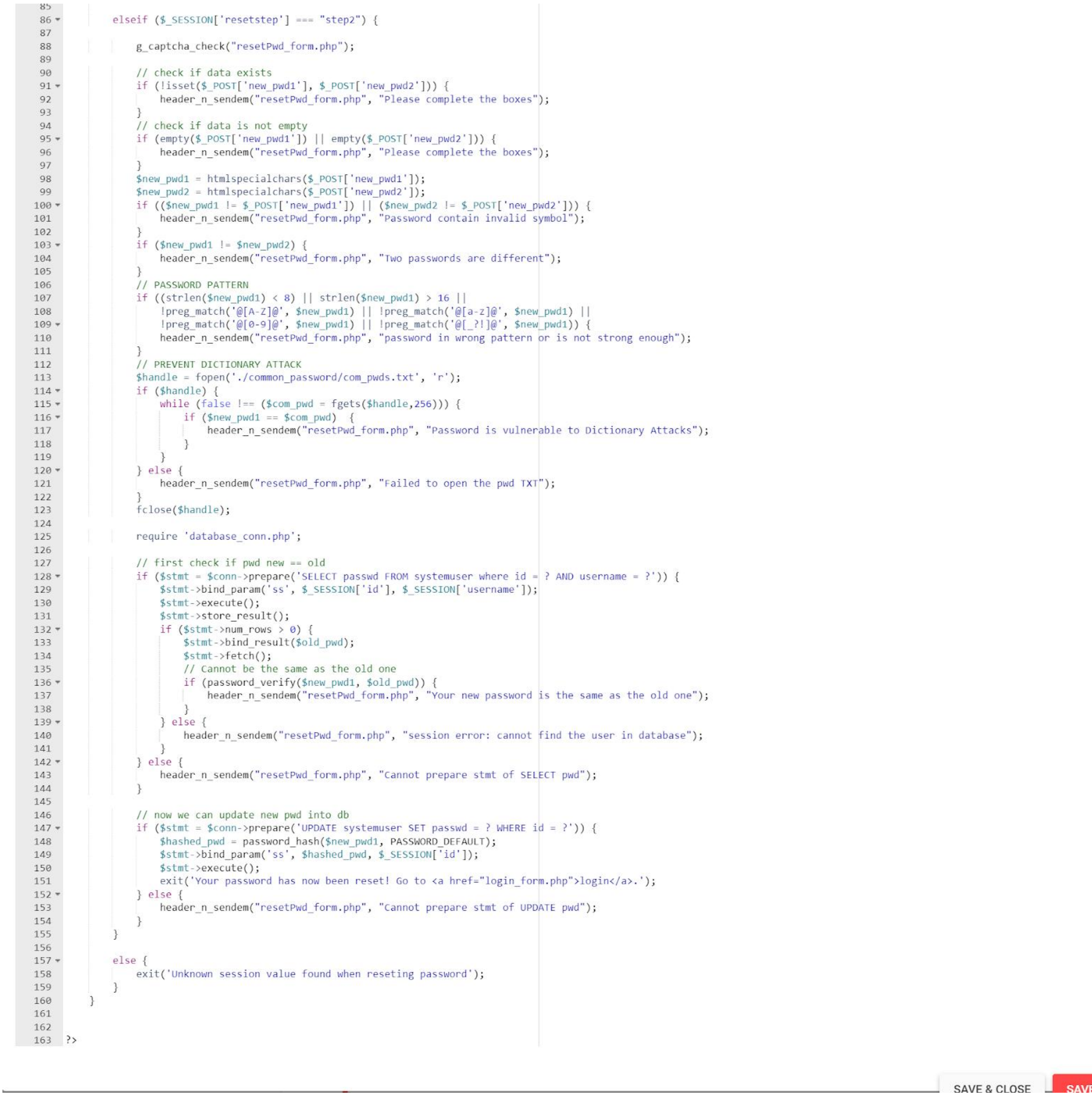
```php
<?php
    session_start();

    require 'helper.php';


    if (!isset($_SESSION['resetstep'])) {

        csrf_counter();

        // check if all data exists
        if (!isset($_POST['username'], $_POST['email'])) {
            header_n_sendem("resetPwd_form.php", "Please complete the boxes");
        }
        // check if all data are not empty
        if (empty($_POST['username']) || empty($_POST['email'])) {
            header_n_sendem("resetPwd_form.php", "Please complete the boxes");
        }
        // username
        $username = htmlspecialchars($_POST['username']);
        if ($username != $_POST['username']) {
            header_n_sendem("resetPwd_form.php", "Username contains invalid symbol");
        }
        // email
        $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
        if (filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
            header_n_sendem("resetPwd_form.php", "Invalid email address");
        }

        require 'database_conn.php';

        // check if that username exists.
        if ($stmt = $conn->prepare('SELECT id, sec_ques, sec_ans FROM systemuser WHERE username = ? AND email = ?')) {
            $stmt->bind_param('ss', $username, $email);
            $stmt->execute();
            $stmt->store_result();
            // find the user
            if ($stmt->num_rows > 0) {
                $stmt->bind_result($id, $sec_ques, $sec_ans);
                $stmt->fetch();
                $_SESSION['id'] = $id;
                $_SESSION['username'] = $username;
                $_SESSION['resetstep'] = "step1";
                $_SESSION['sec_ques_db'] = $sec_ques;
                $_SESSION['sec_ans_db'] = $sec_ans;
                header('Location: resetPwd_form.php');
                exit();
            } else {
                header_n_sendem("resetPwd_form.php", "No such a user/email in the database");
            }
        } else {
            header_n_sendem("resetPwd_form.php", "Cannot prepare stmt of SELECT user for reseting !");
        }


    } else {
```

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



```php


    } else {


        if ($_SESSION['resetstep'] === "step1") {
            // check if data exists
            if (!isset($_POST['sec_ans'])) {
                header_n_sendem("resetPwd_form.php", "Please complete the box");
            }
            // check if data is not empty
            if (empty($_POST['sec_ans'])) {
                header_n_sendem("resetPwd_form.php", "Please complete the box");
            }
            // check form of security answer
            $sec_ans = htmlspecialchars($_POST['sec_ans']);
            if ($sec_ans != $_POST['sec_ans']) {
                header_n_sendem("resetPwd_form.php", "Security answer contains invalid symbol");
            }

            // check if the answer is correct
            if (password_verify($sec_ans, $_SESSION['sec_ans_db'])) {
                $_SESSION['resetstep'] = "step2";
                unset($_SESSION['sec_ques_db']);
                unset($_SESSION['sec_ans_db']);
                header('Location: resetPwd_form.php');
                exit();
            } else {
                header_n_sendem("resetPwd_form.php", "Incorrect answer");
            }
        }

        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

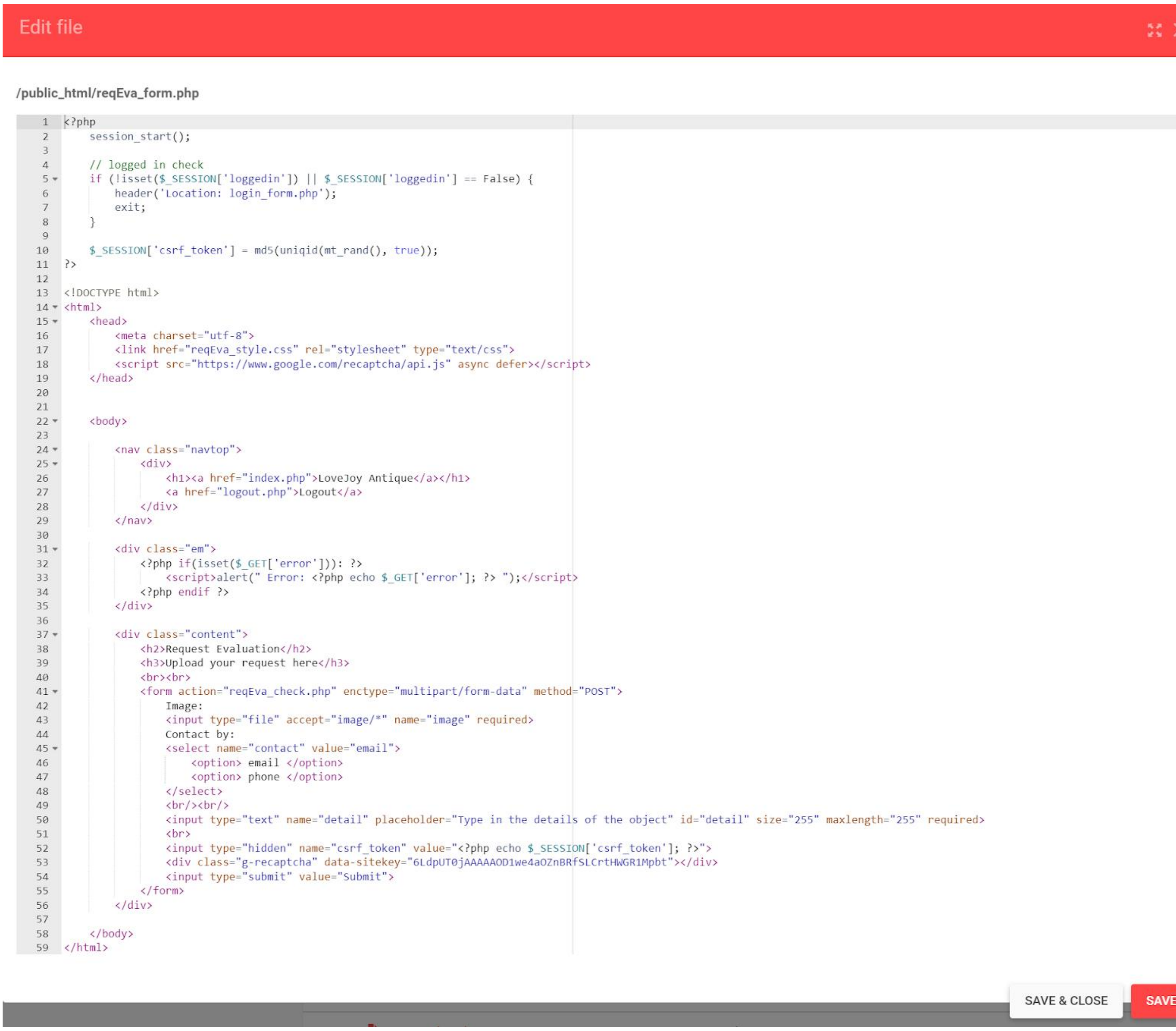


```
    elseif ($_SESSION['resetstep'] === "step2") {

        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 reseting password');
    }
}


?>
```

Task 4 - Implement a “Evaluation Request” web page.

Request Evaluation feature screenshot

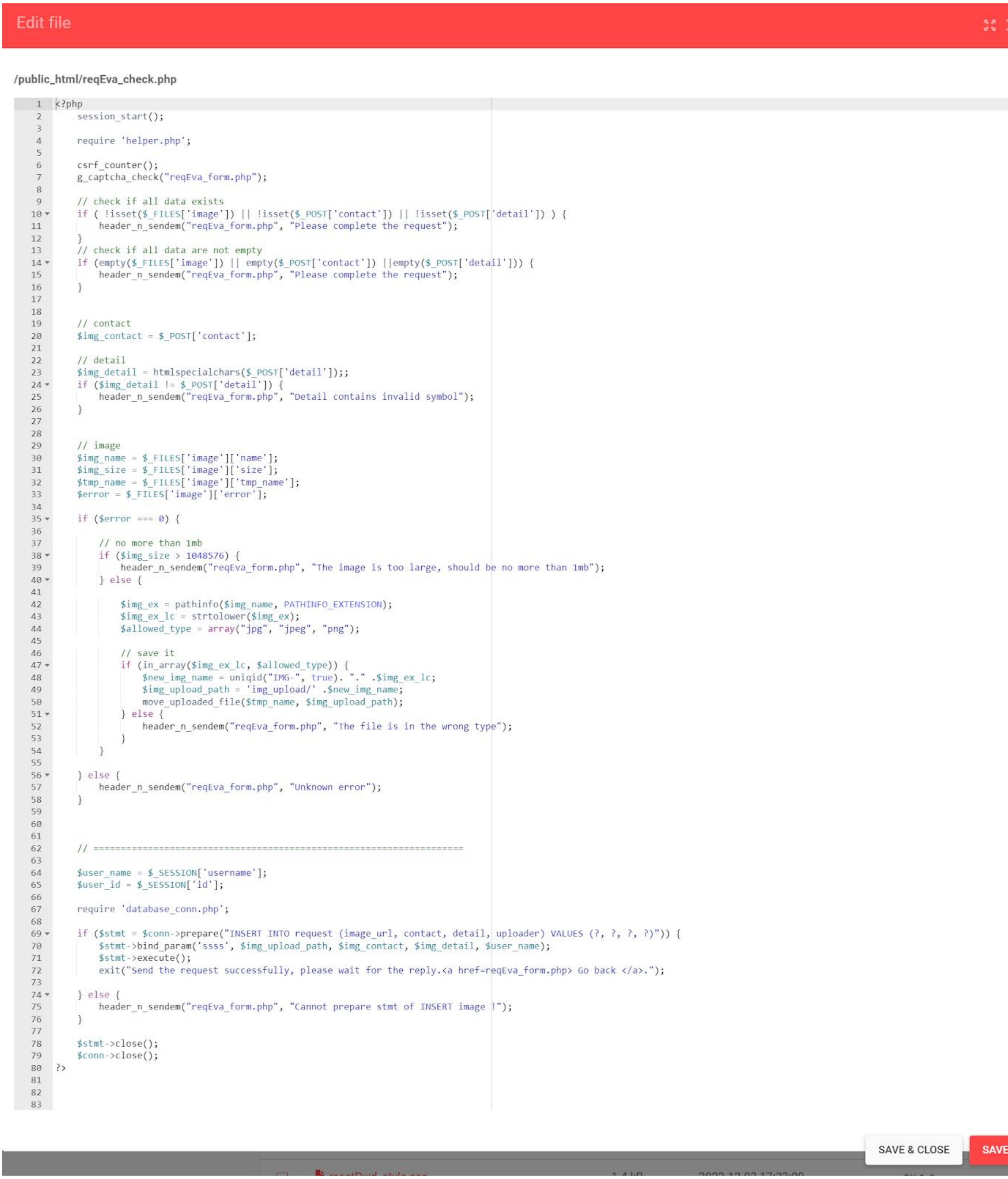
Evaluation Request form



/public_html/reqEva_form.php

```
<?php
    session_start();

    // logged in check
    if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
        header('Location: login_form.php');
        exit;
    }

    $_SESSION['csrf_token'] = md5(uniqid(mt_rand(), true));
?>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <link href="reqEva_style.css" rel="stylesheet" type="text/css">
        <script src="https://www.google.com/recaptcha/api.js" async defer></script>
    </head>


    <body>

        <nav class="navtop">
            <div>
                <h1><a href="index.php">LoveJoy Antique</a></h1>
                <a href="logout.php">Logout</a>
            </div>
        </nav>

        <div class="em">
            <?php if(isset($_GET['error'])): ?>
                <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
            <?php endif ?>
        </div>

        <div class="content">
            <h2>Request Evaluation</h2>
            <h3>Upload your request here</h3>
            <br><br>
            <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
                Image:
                <input type="file" accept="image/*" name="image" required>
                Contact by:
                <select name="contact" value="email">
                    <option> email </option>
                    <option> phone </option>
                </select>
                <br/><br/>
                <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255" maxlength="255" required>
                <br>
                <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                <input type="submit" value="Submit">
            </form>
        </div>

    </body>
</html>
```

Evaluation Request check

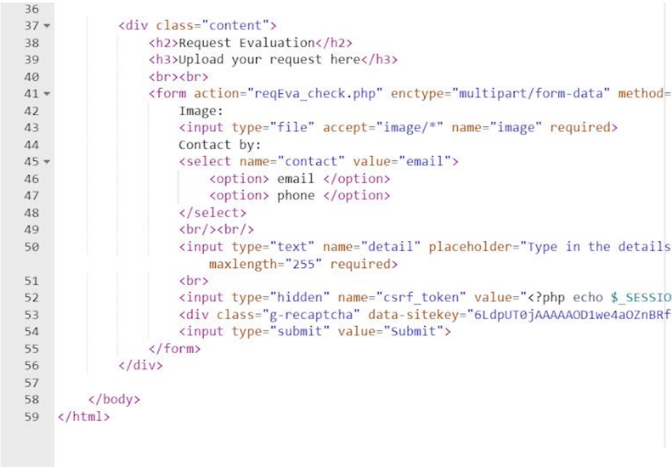


/public_html/reqEva_check.php

```php
<?php
    session_start();

    require 'helper.php';

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

    // check if all data exists
    if ( !isset($_FILES['image']) || !isset($_POST['contact']) || !isset($_POST['detail']) ) {
        header_n_sendem("reqEva_form.php", "Please complete the request");
    }
    // check if all data are not empty
    if (empty($_FILES['image']) || empty($_POST['contact']) ||empty($_POST['detail'])) {
        header_n_sendem("reqEva_form.php", "Please complete the request");
    }


    // contact
    $img_contact = $_POST['contact'];

    // detail
    $img_detail = htmlspecialchars($_POST['detail']);;
    if ($img_detail != $_POST['detail']) {
        header_n_sendem("reqEva_form.php", "Detail contains invalid symbol");
    }


    // image
    $img_name = $_FILES['image']['name'];
    $img_size = $_FILES['image']['size'];
    $tmp_name = $_FILES['image']['tmp_name'];
    $error = $_FILES['image']['error'];

    if ($error === 0) {

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

            $img_ex = pathinfo($img_name, PATHINFO_EXTENSION);
            $img_ex_lc = strtolower($img_ex);
            $allowed_type = array("jpg", "jpeg", "png");

            // save it
            if (in_array($img_ex_lc, $allowed_type)) {
                $new_img_name = uniqid("IMG-", true). "." .$img_ex_lc;
                $img_upload_path = 'img_upload/' .$new_img_name;
                move_uploaded_file($tmp_name, $img_upload_path);
            } else {
                header_n_sendem("reqEva_form.php", "The file is in the wrong type");
            }
        }

    } else {
        header_n_sendem("reqEva_form.php", "Unknown error");
    }



    // ====================================================================

    $user_name = $_SESSION['username'];
    $user_id = $_SESSION['id'];

    require 'database_conn.php';

    if ($stmt = $conn->prepare("INSERT INTO request (image_url, contact, detail, uploader) VALUES (?, ?, ?, ?)")) {
        $stmt->bind_param('ssss', $img_upload_path, $img_contact, $img_detail, $user_name);
        $stmt->execute();
        exit("Send the request successfully, please wait for the reply.<a href=reqEva_form.php> Go back </a>.");

    } else {
        header_n_sendem("reqEva_form.php", "Cannot prepare stmt of INSERT image !");
    }

    $stmt->close();
    $conn->close();
?>
```

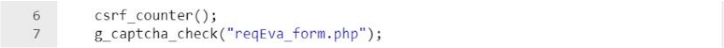
Why do you think it is secure?

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



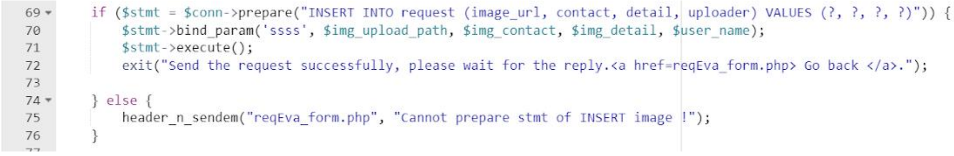
```
        <div class="content">
            <h2>Request Evaluation</h2>
            <h3>Upload your request here</h3>
            <br><br>
            <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
                Image:
                <input type="file" accept="image/*" name="image" required>
                Contact by:
                <select name="contact" value="email">
                    <option> email </option>
                    <option> phone </option>
                </select>
                <br/><br/>
                <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255"
                    maxlength="255" required>
                <br>
                <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                <input type="submit" value="Submit">
            </form>
        </div>

    </body>
</html>
```

- csrf counter-measure implemented, Google reCaptcha implemented



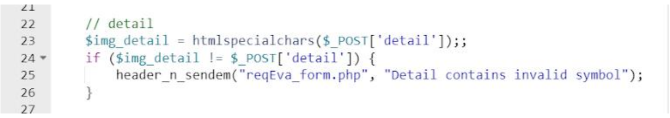
```
    csrf_counter();
    g_captcha_check("reqEva_form.php");
```

- prepare() used to prevent the SQL injection



```
    if ($stmt = $conn->prepare("INSERT INTO request (image_url, contact, detail, uploader) VALUES (?, ?, ?, ?)")) {
        $stmt->bind_param('ssss', $img_upload_path, $img_contact, $img_detail, $user_name);
        $stmt->execute();
        exit("Send the request successfully, please wait for the reply.<a href=reqEva_form.php> Go back </a>.");

    } else {
        header_n_sendem("reqEva_form.php", "Cannot prepare stmt of INSERT image !");
    }
```

- htmlspecialchars() used to skip script-inserting risks

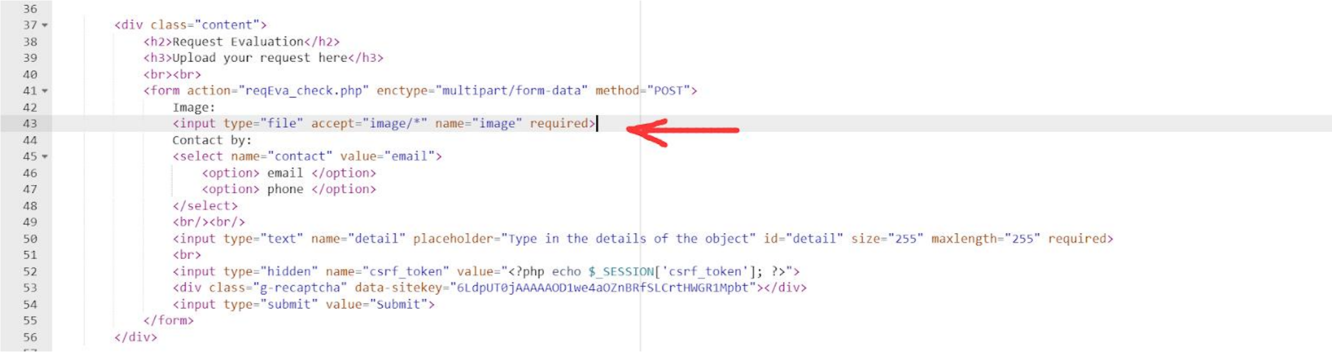


```
    // detail
    $img_detail = htmlspecialchars($_POST['detail']);;
    if ($img_detail != $_POST['detail']) {
        header_n_sendem("reqEva_form.php", "Detail contains invalid symbol");
    }
```

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

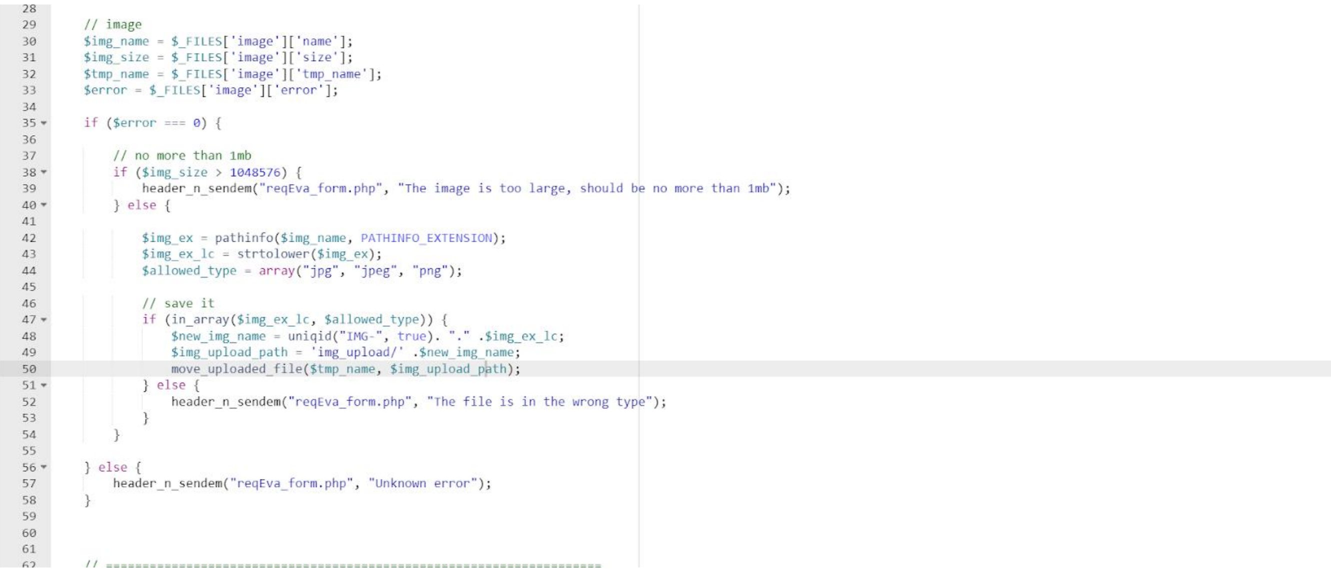
Code of the feature

Form



```
        <div class="content">
            <h2>Request Evaluation</h2>
            <h3>Upload your request here</h3>
            <br><br>
            <form action="reqEva_check.php" enctype="multipart/form-data" method="POST">
                Image:
                <input type="file" accept="image/*" name="image" required>
                Contact by:
                <select name="contact" value="email">
                    <option> email </option>
                    <option> phone </option>
                </select>
                <br/><br/>
                <input type="text" name="detail" placeholder="Type in the details of the object" id="detail" size="255" maxlength="255" required>
                <br>
                <input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
                <div class="g-recaptcha" data-sitekey="6LdpUT0jAAAAAOD1we4aOZnBRfSLCrtHWGR1Mpbt"></div>
                <input type="submit" value="Submit">
            </form>
        </div>
```

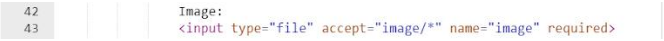
Check



```
// image
$img_name = $_FILES['image']['name'];
$img_size = $_FILES['image']['size'];
$tmp_name = $_FILES['image']['tmp_name'];
$error = $_FILES['image']['error'];

if ($error === 0) {

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

        $img_ex = pathinfo($img_name, PATHINFO_EXTENSION);
        $img_ex_lc = strtolower($img_ex);
        $allowed_type = array("jpg", "jpeg", "png");

        // save it
        if (in_array($img_ex_lc, $allowed_type)) {
            $new_img_name = uniqid("IMG-", true). "." .$img_ex_lc;
            $img_upload_path = 'img_upload/' .$new_img_name;
            move_uploaded_file($tmp_name, $img_upload_path);
        } else {
            header_n_sendem("reqEva_form.php", "The file is in the wrong type");
        }
    }

} else {
    header_n_sendem("reqEva_form.php", "Unknown error");
}



// =================================================================
```

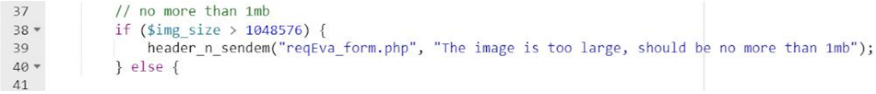
Why do you think it is secure?

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



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

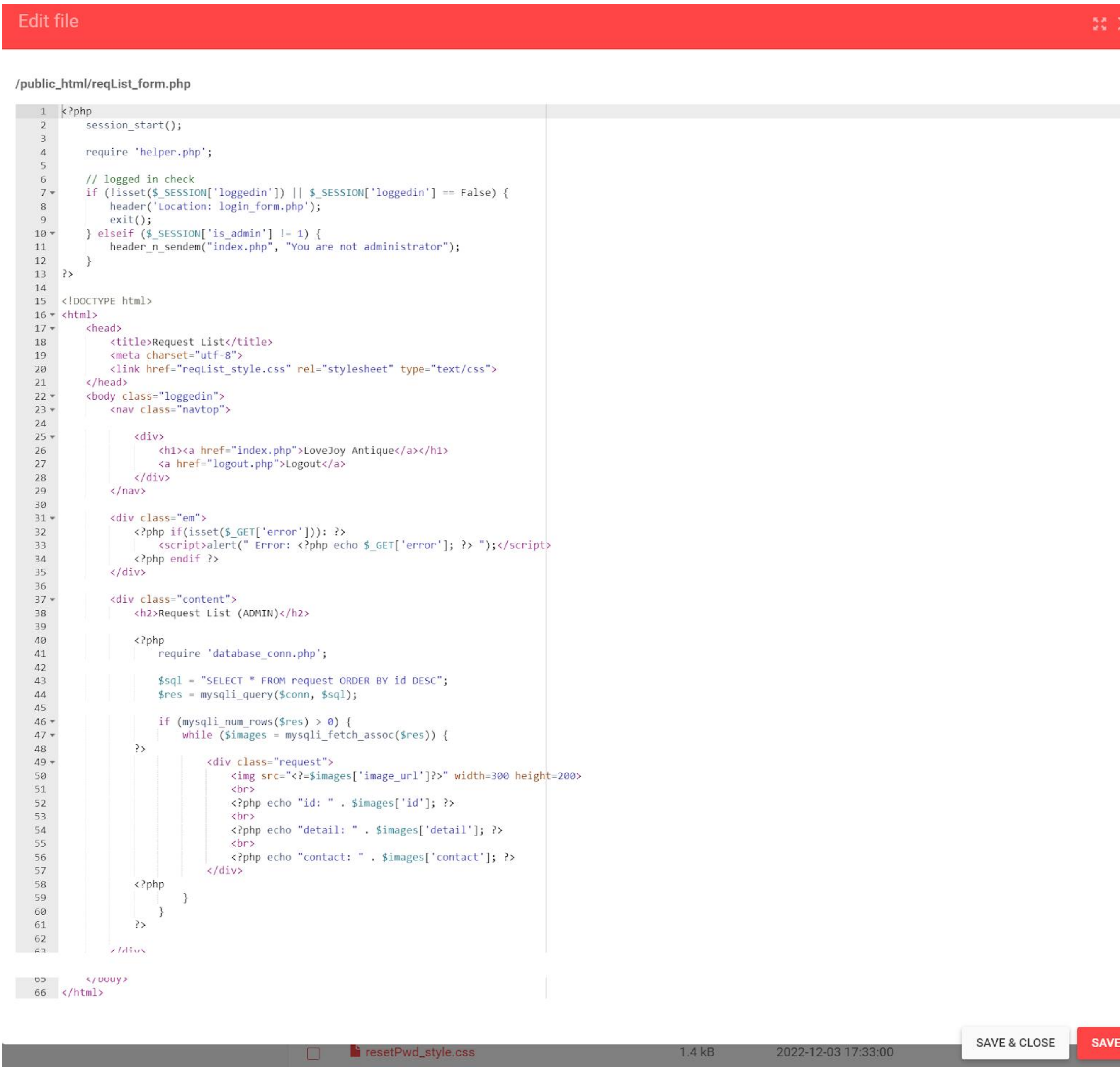
- not allow image more than 1mb



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

Task 6 – Request Listing Page

Code of the feature

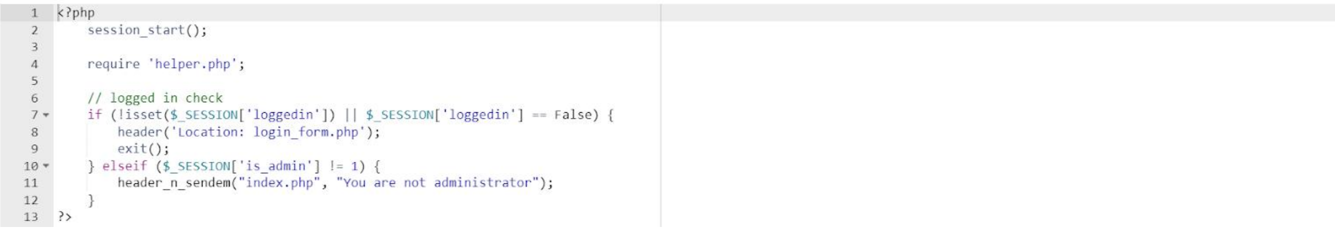


/public_html/reqList_form.php

```
<?php
    session_start();

    require 'helper.php';

    // logged in check
    if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
        header('Location: login_form.php');
        exit();
    } elseif ($_SESSION['is_admin'] != 1) {
        header_n_sendem("index.php", "You are not administrator");
    }
?>

<!DOCTYPE html>
<html>
    <head>
        <title>Request List</title>
        <meta charset="utf-8">
        <link href="reqList_style.css" rel="stylesheet" type="text/css">
    </head>
    <body class="loggedin">
        <nav class="navtop">

            <div>
                <h1><a href="index.php">LoveJoy Antique</a></h1>
                <a href="logout.php">Logout</a>
            </div>
        </nav>

        <div class="em">
            <?php if(isset($_GET['error'])): ?>
                <script>alert(" Error: <?php echo $_GET['error']; ?> ");</script>
            <?php endif ?>
        </div>

        <div class="content">
            <h2>Request List (ADMIN)</h2>

            <?php
                require 'database_conn.php';

                $sql = "SELECT * FROM request ORDER BY id DESC";
                $res = mysqli_query($conn, $sql);

                if (mysqli_num_rows($res) > 0) {
                    while ($images = mysqli_fetch_assoc($res)) {
            ?>
                        <div class="request">
                            <img src="<?=$images['image_url']?>" width=300 height=200>
                            <br>
                            <?php echo "id: " . $images['id']; ?>
                            <br>
                            <?php echo "detail: " . $images['detail']; ?>
                            <br>
                            <?php echo "contact: " . $images['contact']; ?>
                        </div>
            <?php
                    }
                }
            ?>

        </div>

    </body>
</html>
```

Why do you think it is secure?

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



```
<?php
    session_start();

    require 'helper.php';

    // logged in check
    if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] == False) {
        header('Location: login_form.php');
        exit();
    } elseif ($_SESSION['is_admin'] != 1) {
        header_n_sendem("index.php", "You are not administrator");
    }
?>
```

- 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.