

April 26, 2023

# **PHP Password Manager Report**

**Titas Lukaitis**  
Vilnius Univeristy

# Contents

1 UML Diagrams .....	3
2 Responses .....	3
2.1 Index .....	3
2.1.1 Logged Out .....	3
2.1.2 Logged In .....	3
2.2 Registration .....	4
2.2.1 Missing Fields .....	5
2.2.2 Password and Repeated Password Don't Match .....	5
2.2.3 Username Length More Than 30 Characters .....	5
2.2.4 Username Already Exists .....	6
2.2.5 Successful Registration .....	6
2.3 Login .....	7
2.3.1 Missing Fields .....	8
2.3.2 Username Does Not Exist .....	8
2.3.3 Invalid Password .....	9
2.3.4 Successful Login .....	9
2.4 Logout .....	9
2.5 Password Generator .....	10
2.5.1 Invalid Char Length .....	12
2.5.2 No Options Selected .....	12
2.5.3 Options .....	12
2.6 New Password .....	14
2.6.1 Missing Fields .....	14
2.6.2 Success .....	15
2.7 Home Page .....	16
2.7.1 No Passwords .....	18
2.7.2 Passwords .....	19
2.7.3 Change Password Details .....	19
2.7.4 Delete Password .....	20
2.8 Change Password .....	20
2.8.1 Missing Fields .....	22
2.8.2 Passwords Do Not Match .....	22
2.8.3 Success .....	22

## 1 UML Diagrams

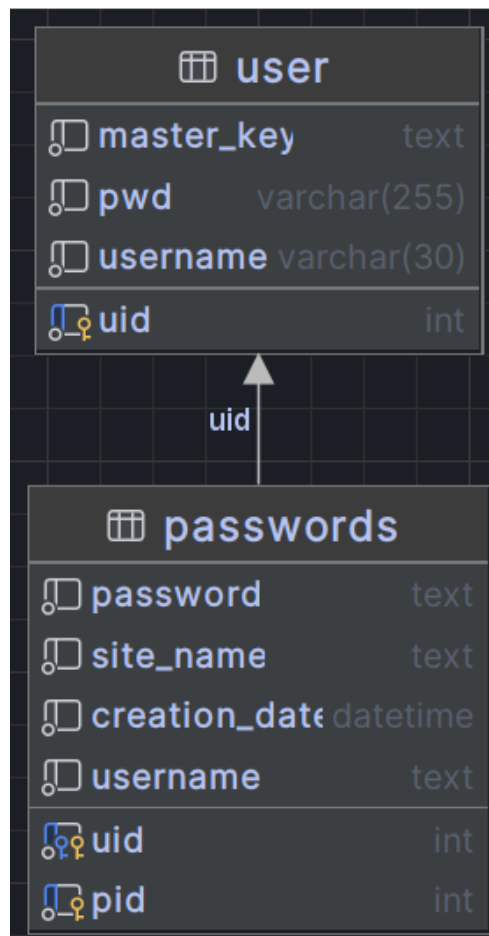


Figure 1: *db model* of the ppm table

## 2 Responses

For more detailed code snippets please view the attached source code.

### 2.1 Index

```
<?php
if (isset($_SESSION['uid'])) {
    echo '<h1 class="text-center">You are logged in! 👍</h1>';
} else {
    echo '<h1 class="text-center">You should log in or register to access the app!</h1>';
}
```

#### 2.1.1 Logged Out

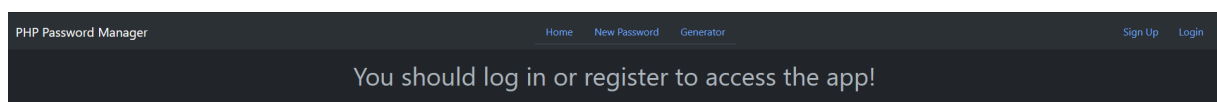


Figure 2: *index* when logged out

#### 2.1.2 Logged In

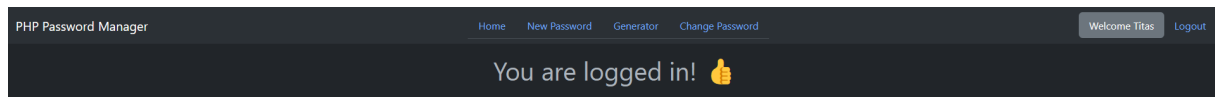


Figure 3: *index* when logged in

## 2.2 Registration

```
<?php
if (isset($_POST['submit'])) {
    if (empty($_POST['username'])) {
        print_messages::printError("Empty Username");
    } elseif (empty($_POST['password'])) {
        print_messages::printError("Empty Password");
    } elseif (empty($_POST['repeatPassword'])) {
        print_messages::printError("Empty Repeated Password");
    } elseif ($repeatPassword !== $password) {
        print_messages::printError("Passwords do not match");
    } elseif (strlen($name) > 30) {
        print_messages::printError("Username too long");
    } elseif ($db_utils->usernamesExists($name)) {
        print_messages::printError("Username already exists");
    } else {
        register_user($password, $name);
    }
}

/**
 * Registers a new user in the database
 * @param string $password Their plain-text password
 * @param string $name Username
 * @return void
 */
function registerUser(string $password, string $name): void {
    // DB vars
    $password_hash = password_hash($password, PASSWORD_DEFAULT);
    $master_key_encode = PwdUtils::generateMasterKey();
    $master_key = PwdUtils::encryptMaster($master_key_encode, $password);

    // Database stuff
    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("INSERT INTO user (username, pwd, master_key) VALUES
(:name, :pwd, :master)");
        $stmt->bindParam(":name", $name);
        $stmt->bindParam(":pwd", $password_hash);
        $stmt->bindParam(":master", $master_key);

        // execute
        $stmt->execute();
        unset($stmt);
        unset($db);
    } catch (PDOException $e) {
        // Silently fail
        PrintMessages::printError("Database Error");
        die();
    }
}
```

```
PrintMessages::printInfo("Registration Complete");  
}
```

### 2.2.1 Missing Fields

The screenshot shows the 'PHP Password Manager' interface with a dark theme. At the top, there are navigation links: 'Home', 'New Password', and 'Generator'. On the right, there are 'Sign Up' and 'Login' buttons. A red error message 'Empty Username' is displayed at the top of the form area. Below the error, there are three input fields: 'Username' (with an '@' icon), 'Password', and 'Repeat Password'. A blue 'Create Account' button is at the bottom of the form.

Figure 4: *register* missing username

The screenshot shows the same 'PHP Password Manager' interface. The red error message now reads 'Empty Password'. The 'Username' field is filled, but the 'Password' and 'Repeat Password' fields are empty. The 'Create Account' button remains at the bottom.

Figure 5: *register* missing password

The screenshot shows the same 'PHP Password Manager' interface. The red error message now reads 'Empty Repeated Password'. Both the 'Username' and 'Password' fields are filled, but the 'Repeat Password' field is empty. The 'Create Account' button remains at the bottom.

Figure 6: *register* missing repeated password

### 2.2.2 Password and Repeated Password Don't Match

Inputs: Username: hello123, Password: 123, Repeated Password: 1234

The screenshot shows the same 'PHP Password Manager' interface. The red error message now reads 'Passwords do not match'. All three input fields ('Username', 'Password', and 'Repeat Password') are filled. The 'Create Account' button remains at the bottom.

Figure 7: *register* password and repeated password don't match

### 2.2.3 Username Length More Than 30 Characters

Inputs:

- Username: 1234567890\_1234567890\_1234567890
- Password: 123
- Repeated Password: 123

PHP Password Manager

Home New Password Generator Sign Up Login

Username too long

@ Username

Password

Repeat Password

Create Account

Figure 8: *register* Username exceeds 30 characters

## 2.2.4 Username Already Exists

uid	master_key	pwd	username
1	4 KfNFZcwXn42nyJnKEdLwLcwnYpHYW0rUHQLRJ08JHqzYCLJg432AhZ3Wb0L4QyB11Tck7...	\$2y\$10\$H.YNkr.SCJXXVdWgyQwt90x8ZUhfFrGxqepypw.azyGhLL8.76cBe	tray
2	5 DPRTI3HEilmmNL41sycSZXDvambvh8ypzdIfQHznSm4YR0QnGy7hxvjoYEbkCPJgGy+gb...	\$2y\$10\$koLzE8j4h.nnXaDHy42fh0qcyuKpLQZtY0ekJrP89ZhCEFrw4nCNo	kipras
3	6 IGc0s1w4WTY0jan4RMiKW8AozpS11j+fZN9A5kzFvjIfqgo/oGdb2UI4csX5600QKbL...	\$2y\$10\$V4iZLopvT3vIwysHxR.ksu03Km5F6E04NxxkAXZg260Imoojav0c2	123
4	7 ma++K2j1k5Zj9CQzopW5IEy428STBFjaqoERjdgBcYR2pVowIwpA3llmBqc38ZbNG+afe...	\$2y\$10\$0a.eaCs8bzc3Pzt6dAQcnuRb90.Rp0i5nmwKRt0hZiY55ScQXPSv0	asya
5	8 EkUaoEzJG51y+8+LxpYLVxVqgVvOPJrHsTseB3TN0do6K7W3CedbX5be4n0WzE3iS0ww+	\$2y\$10\$T1xy/BZuZ/0n13N659vF8uIsscPzTMBGp96N0Lcj06UYbXJN5MYG0	alexei
6	9 hhKHXRR7nc5mZi8Tj0PwbNnPg109dbobUoRHimfqo6XsRktnmidOnfL01CjbQNY0+Rxerg...	\$2y\$10\$Ahr69ed5gosnuZkzwMYp0uDG0D.JAwBgw5gaSQc5Wa9SRMca.FXQK	Titas
7	10 TdqhmZ6an8LRgmr6/Lkfs6dxK6prEyJ+aYSD5jWJX3sQ35MML5Zjuo90yJC3zmxYKd5e5...	\$2y\$10\$gZ9A7sdNmyDhkSEW15e9RuKDN4gLcQpQd2nc9sfduWfE0b1Tb1rc2	test
8	11 GYmLnZDMV5gB/fJpGcncs6szvxEx+0EaKltYzTe3hL0vVSMDeoksPSQzWqTwvaReunrvdh...	\$2y\$10\$KzHFuF/DwnBR30EzHG7XU0pvt0GU44UuankFpemtKcXYiVuMRbgvc	test123
9	12 UgnjeIGrSRqcywmy/cx6rd1Qe5VGckZC2986gDfyYUjVoX01MyuuV46wc4pEPyL6509vu...	\$2y\$10\$pmQ5j1sJorJVNtDrSh6RuabfB3mRy90WZpRw7WBeq7Q0mm5zvRF.	123456

Figure 9: *register* Existing usernames

Inputs:

- Username: 123
- Password: 123
- Repeated Password: 123

PHP Password Manager

Home New Password Generator Sign Up Login

Username already exists

@ Username

Password

Repeat Password

Create Account

Figure 10: *register* Username exists

## 2.2.5 Successful Registration

uid	master_key	pwd	username
1	4 KfNFZcwXn42nyJnKEdLwLcwnYpHYW0rUHQLRJ08JHqzYCLJg432AhZ3Wb0L4QyB11Tck7...	\$2y\$10\$H.YNkr.SCJXXVdWgyQwt90x8ZUhfFrGxqepypw.azyGhLL8.76cBe	tray
2	5 DPRTI3HEilmmNL41sycSZXDvambvh8ypzdIfQHznSm4YR0QnGy7hxvjoYEbkCPJgGy+gb...	\$2y\$10\$koLzE8j4h.nnXaDHy42fh0qcyuKpLQZtY0ekJrP89ZhCEFrw4nCNo	kipras
3	6 IGc0s1w4WTY0jan4RMiKW8AozpS11j+fZN9A5kzFvjIfqgo/oGdb2UI4csX5600QKbL...	\$2y\$10\$V4iZLopvT3vIwysHxR.ksu03Km5F6E04NxxkAXZg260Imoojav0c2	123
4	7 ma++K2j1k5Zj9CQzopW5IEy428STBFjaqoERjdgBcYR2pVowIwpA3llmBqc38ZbNG+afe...	\$2y\$10\$0a.eaCs8bzc3Pzt6dAQcnuRb90.Rp0i5nmwKRt0hZiY55ScQXPSv0	asya
5	8 EkUaoEzJG51y+8+LxpYLVxVqgVvOPJrHsTseB3TN0do6K7W3CedbX5be4n0WzE3iS0ww+	\$2y\$10\$T1xy/BZuZ/0n13N659vF8uIsscPzTMBGp96N0Lcj06UYbXJN5MYG0	alexei
6	9 hhKHXRR7nc5mZi8Tj0PwbNnPg109dbobUoRHimfqo6XsRktnmidOnfL01CjbQNY0+Rxerg...	\$2y\$10\$Ahr69ed5gosnuZkzwMYp0uDG0D.JAwBgw5gaSQc5Wa9SRMca.FXQK	Titas
7	10 TdqhmZ6an8LRgmr6/Lkfs6dxK6prEyJ+aYSD5jWJX3sQ35MML5Zjuo90yJC3zmxYKd5e5...	\$2y\$10\$gZ9A7sdNmyDhkSEW15e9RuKDN4gLcQpQd2nc9sfduWfE0b1Tb1rc2	test
8	11 GYmLnZDMV5gB/fJpGcncs6szvxEx+0EaKltYzTe3hL0vVSMDeoksPSQzWqTwvaReunrvdh...	\$2y\$10\$KzHFuF/DwnBR30EzHG7XU0pvt0GU44UuankFpemtKcXYiVuMRbgvc	test123
9	12 UgnjeIGrSRqcywmy/cx6rd1Qe5VGckZC2986gDfyYUjVoX01MyuuV46wc4pEPyL6509vu...	\$2y\$10\$pmQ5j1sJorJVNtDrSh6RuabfB3mRy90WZpRw7WBeq7Q0mm5zvRF.	123456

Figure 11: *register* DB before registration

PHP Password Manager

Home New Password Generator Sign Up Login

@ hello123

...

...

Create Account

Figure 12: *register* Before registration

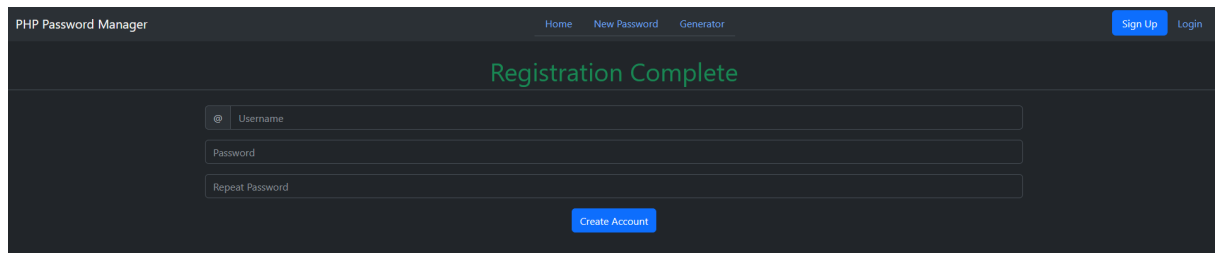


Figure 13: *register* After registration

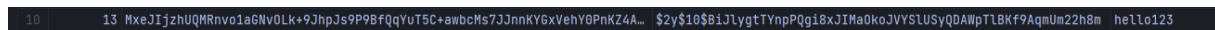


Figure 14: *register* New db entry

## 2.3 Login

```
<?php
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/PrintMessages.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/Database.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/PwdUtils.php");

if (isset($_POST['submit'])) {
    if (empty($_POST['username'])) {
        PrintMessages::printError("Empty Username");
    } elseif (empty($_POST['password'])) {
        PrintMessages::printError("Empty Password");
    } else {
        $name = $_POST['username'];
        $password = $_POST['password'];
        loginUser($name, $password);
        header("Location: /app/home.php");
    }
}

/**
 * Logs the user in, and sets the session variables
 * @param string $username The username of the user
 * @param string $password Plain text password of the user
 * @return void
 */
function loginUser(string $username, string $password): void {
    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("SELECT * FROM user WHERE username = ? LIMIT 1;");
        $stmt->bindParam(1, $username);

        $stmt->execute();
        if ($stmt->rowCount() != 1) {
            PrintMessages::printError("Invalid Username Or Password");
            return;
        }

        $result = $stmt->fetch();

        $uid = $result['uid'];
        $master_key_result = $result['master_key'];
        $pwd = $result['pwd'];
```

```

$username = $result['username'];

if (!password_verify($password, $pwd)) {
    PrintMessages::printError("Invalid Username Or Password");
    return;
}

// Apparently sessions are actually pretty secure already, though
// session_regenerate_id() should be called after each login attempt
session_regenerate_id();
$master_key_decrypted = PwdUtils::decryptMaster($master_key_result,
$password);

$_SESSION['uid'] = $uid;
$_SESSION['master_key'] = $master_key_decrypted;
$_SESSION['username'] = $username;

PrintMessages::printInfo("Login Successful");
} catch (PDOException $e) {
    // Silently fail
    PrintMessages::printError("Database Error");
    die();
}

unset($stmt);
unset($db);
}

```

### 2.3.1 Missing Fields

Input: Nothing

The screenshot shows the login interface of the 'PHP Password Manager'. At the top, there are navigation links: 'Home', 'New Password', 'Generator', 'Sign Up', and 'Login'. A red error message 'Empty Username' is displayed above the 'Username' input field. The 'Password' input field is also visible, and a 'Sign in' button is at the bottom of the form.

Figure 15: *login* Missing username

Input:

- Username: 123

The screenshot shows the login interface of the 'PHP Password Manager'. The 'Username' input field is filled with '123'. A red error message 'Empty Password' is displayed above the 'Password' input field. The 'Sign in' button is at the bottom of the form.

Figure 16: *login* Missing password

### 2.3.2 Username Does Not Exist

Input:

- Username: DoesNotExist
- Password: 123



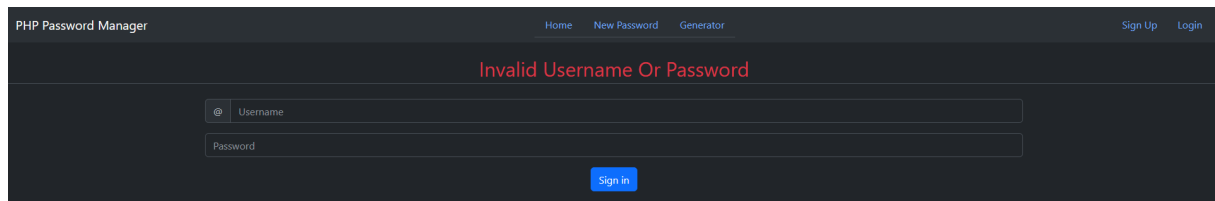


Figure 17: *login* Username does not exist

### 2.3.3 Invalid Password

Input:

- Username: Titas
- Password: Titas

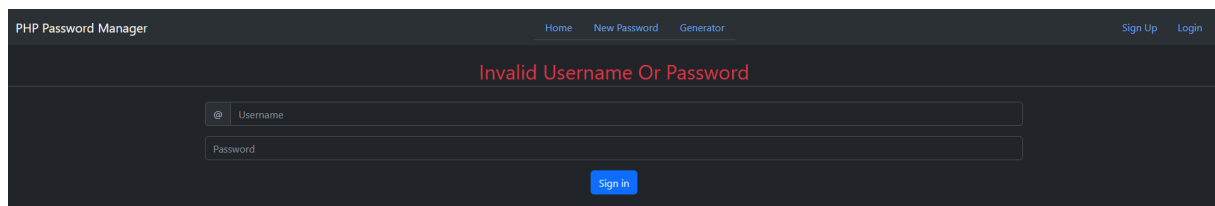


Figure 18: *login* Password is incorrect

### 2.3.4 Successful Login

Input:

- Username: Titas
- Password: Titas2003

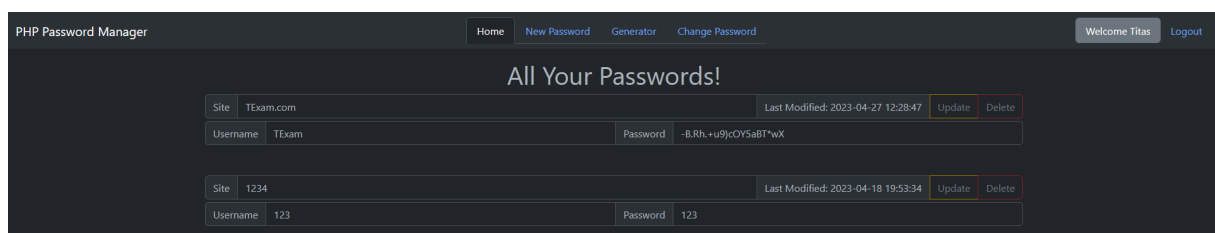


Figure 19: *login* Success!

You are automatically redirected to the Home page.

## 2.4 Logout

Before:

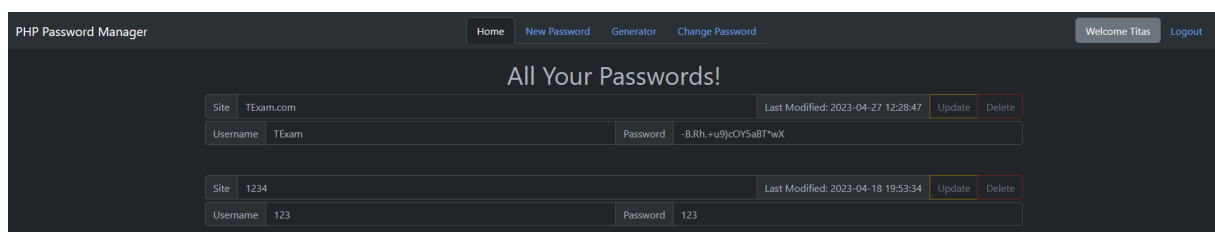


Figure 20: *logout* Before pressing button

After:

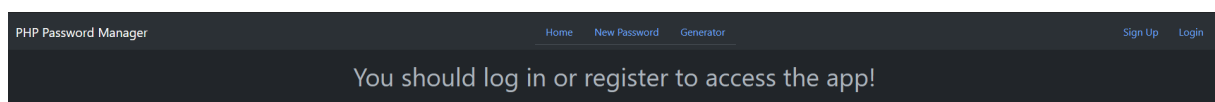


Figure 21: *logout* After pressing button

## 2.5 Password Generator

### generator.php

```
<?php
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/PrintMessages.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/generator/PasswordGenerator.php");

$chars = 20;
if (isset($_POST['chars'])) {
    $chars = $_POST['chars'];
}

$specialChars = true;
if (!isset($_POST['specialChars'])) {
    $specialChars = false;
}

$numbers = true;
if (!isset($_POST['numbers'])) {
    $numbers = false;
}

$uppercase = true;
if (!isset($_POST['uppercase'])) {
    $uppercase = false;
}

$lowercase = true;
if (!isset($_POST['lowercase'])) {
    $lowercase = false;
}

if (isset($_POST['submit'])) {
    print("<h2 class='text-center'>" . generatePassword($chars, $specialChars,
$numbers, $uppercase, $lowercase) . "</h2>");
}

/**
 * Generates a password according to the given parameters. Returns an error message
if the parameters are invalid.
 * @param int $chars A number between 1 and 128
 * @param bool $specialChars True if special characters should be included
 * @param bool $numbers True if numbers should be included
 * @param bool $uppercase True if uppercase letters should be included
 * @param bool $lowercase True if lowercase letters should be included
 * @return string The generated password
 */
function generatePassword(int $chars, bool $specialChars, bool $numbers, bool
$uppercase, bool $lowercase): string {
    $gen = new PasswordGenerator($chars, $numbers, $specialChars, $uppercase,
$lowercase);
    try {
        return $gen->generate();
    } catch (Exception $e) {
        return "<div class='text-danger'><p>" . $e->getMessage() . "</p></div>";
    }
}
```

```

    }
}

```

## PasswordGenerator.php

```

<?php
class PasswordGenerator {
    private int $passwordLength;
    private bool $hasUpperCase;
    private bool $hasLowerCase;
    private bool $hasSymbols;
    private bool $hasNumbers;

    public function __construct(int $passwordLength, bool $hasNumbers, bool
$hasSymbols, bool $hasUpperCase, bool $hasLowerCase) {
        $this->passwordLength = $passwordLength;
        $this->hasNumbers = $hasNumbers;
        $this->hasSymbols = $hasSymbols;
        $this->hasUpperCase = $hasUpperCase;
        $this->hasLowerCase = $hasLowerCase;
    }

    /**
     * Generates a password based on the previous parameters
     * @return string The password
     * @throws Exception
     */
    public function generate(): string {
        if ($this->passwordLength <= 0) {
            throw new Exception("Password length cannot be less than 1");
        } elseif ($this->passwordLength > 128) {
            throw new Exception("Password length cannot be more than 128");
        }

        if (!$this->hasLowerCase && !$this->hasUpperCase && !$this->hasSymbols && !
$this->hasNumbers) {
            throw new Exception("At least 1 option must be enabled");
        }

        $pwd = "";

        for ($i = 0; $i < $this->passwordLength; ) {
            $switch = rand(0, 3);

            // lowercase character
            if (($switch == 0) && $this->hasLowerCase) {
                $pwd .= chr(rand(97, 122));
                $i++;
            }

            // uppercase character
            if (($switch == 1) && $this->hasUpperCase) {
                $pwd .= chr(rand(65, 90));
                $i++;
            }
        }
    }
}

```

```

// number
if (($switch == 2) && $this->hasNumbers) {
    $pwd .= chr(rand(48, 57));
    $i++;
}

// symbol
if (($switch == 3) && $this->hasSymbols) {
    $pwd .= chr(rand(37, 47));
    $i++;
}
}

return $pwd;
}
}

```

### 2.5.1 Invalid Char Length

The screenshot shows the 'Password Generator' section of the 'PHP Password Manager' application. The 'Number of characters' input field contains '-1'. A tooltip message reads: 'Please select a value that is no less than 1.' Below the input field are five checkboxes: 'Include special character', 'Include numbers', 'Include Upper-case letters', and 'Include Lower-case letters'. All checkboxes are currently unchecked. A blue 'Generate' button is located at the bottom right of the form.

Figure 22: *password generator* Length less than 0

The screenshot shows the 'Password Generator' section of the 'PHP Password Manager' application. The 'Number of characters' input field contains '129'. A tooltip message reads: 'Please select a value that is no more than 128.' Below the input field are five checkboxes: 'Include special character', 'Include numbers', 'Include Upper-case letters', and 'Include Lower-case letters'. All checkboxes are currently unchecked. A blue 'Generate' button is located at the bottom right of the form.

Figure 23: *password generator* Length more than 128

### 2.5.2 No Options Selected

The screenshot shows the 'Password Generator' section of the 'PHP Password Manager' application. The 'Number of characters' input field contains '20'. A red error message is displayed above the checkboxes: 'At least 1 option must be enabled'. Below the input field are five checkboxes: 'Include special characters', 'Include numbers', 'Include Upper-case letters', and 'Include Lower-case letters'. All checkboxes are currently unchecked. A blue 'Generate' button is located at the bottom right of the form.

Figure 24: *password generator* No options selected with valid length

### 2.5.3 Options

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

### Password Generator

%'!.-\_+~%+!,-(+

Number of characters 20

Include special characters ☒

Include numbers ☐

Include Upper-case letters ☐

Include Lower-case letters ☐

Generate

Figure 25: *password generator* Only special characters

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

### Password Generator

52018525649316359246

Number of characters 20

Include special characters ☐

Include numbers ☒

Include Upper-case letters ☐

Include Lower-case letters ☐

Generate

Figure 26: *password generator* Only numbers

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

### Password Generator

CWJUUVWIRTBMBNWIFWYWC

Number of characters 20

Include special characters ☐

Include numbers ☐

Include Upper-case letters ☒

Include Lower-case letters ☐

Generate

Figure 27: *password generator* Only uppercase letters

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

### Password Generator

jpunzrnygzfpwzfzofbh

Number of characters 20

Include special characters ☐

Include numbers ☐

Include Upper-case letters ☐

Include Lower-case letters ☒

Generate

Figure 28: *password generator* Only lowercase letters

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

### Password Generator

f8Q(65/0pLm352lW7hq.\*S%s6eC0).

Number of characters 30

Include special characters ☒

Include numbers ☒

Include Upper-case letters ☒

Include Lower-case letters ☒

Generate

Figure 29: *password generator* All options selected with a length of 30

## 2.6 New Password

```
<?php
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/PrintMessages.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/Database.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/PwdUtils.php");

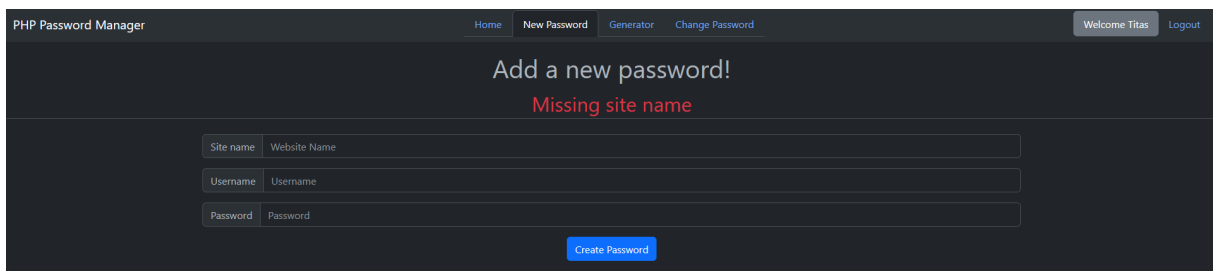
if (isset($_POST["submit"])) {
    if (empty($_POST["site"])) {
        PrintMessages::printError("Missing site name");
    } elseif (empty($_POST["username"])) {
        PrintMessages::printError("Missing username");
    } elseif (empty($_POST["password"])) {
        PrintMessages::printError("Missing password");
    } else {
        createNewPassword($_POST["site"], $_POST["username"], $_POST["password"]);
        unset($_POST["site"]);
        unset($_POST["username"]);
        unset($_POST["password"]);
    }
}

function createNewPassword(string $site, string $username, string $password): void {
    $current_time = date("Y-m-d H:i:s");
    $encrypted_password = PwdUtils::encryptPassword($password);
    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("INSERT INTO passwords (uid, password, username,
site_name, creation_date) VALUES (:uid, :pwd, :user, :site, :date)");
        $stmt->bindParam(':uid', $_SESSION['uid']);
        $stmt->bindParam(':pwd', $encrypted_password);
        $stmt->bindParam(':user', $username);
        $stmt->bindParam(':site', $site);
        $stmt->bindParam(':date', $current_time);

        $stmt->execute();

        PrintMessages::printInfo("Password created!");
        unset($stmt);
        unset($db);
    } catch (PDOException $e) {
        PrintMessages::printError("Database Error");
    }
}
```

### 2.6.1 Missing Fields



The screenshot shows the 'New Password' page of the 'PHP Password Manager'. The page has a dark theme. At the top, there's a navigation bar with links: 'Home', 'New Password', 'Generator', and 'Change Password'. On the right, it says 'Welcome Titus' and 'Logout'. The main heading is 'Add a new password!'. Below it, a red error message says 'Missing site name'. There are three input fields: 'Site name' (with a placeholder 'Website Name'), 'Username' (with a placeholder 'Username'), and 'Password' (with a placeholder 'Password'). A blue 'Create Password' button is at the bottom.

Figure 30: *new password* Missing site name

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

Add a new password!

Missing username

Site name site

Username Username

Password Password

Create Password

Figure 31: *new password* Missing username

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

Add a new password!

Missing password

Site name site

Username user

Password Password

Create Password

Figure 32: *new password* Missing password

## 2.6.2 Success

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

All Your Passwords!

Site	TEExam.com	Last Modified: 2023-04-27 12:28:47	Update	Delete
Username	TEExam	Password	-B.Rh.+u9jcOY5aB1*wX	
Site	1234	Last Modified: 2023-04-18 19:53:34	Update	Delete
Username	123	Password	123	

Figure 33: *new password* Before entering a new password

Input:

- Site: site
- Username: user
- Password: pass

PHP Password Manager

Home New Password Generator Change Password

Welcome Titus Logout

Add a new password!

Password created!

Site name Website Name

Username Username

Password Password

Create Password

Figure 34: *new password* Success on new password

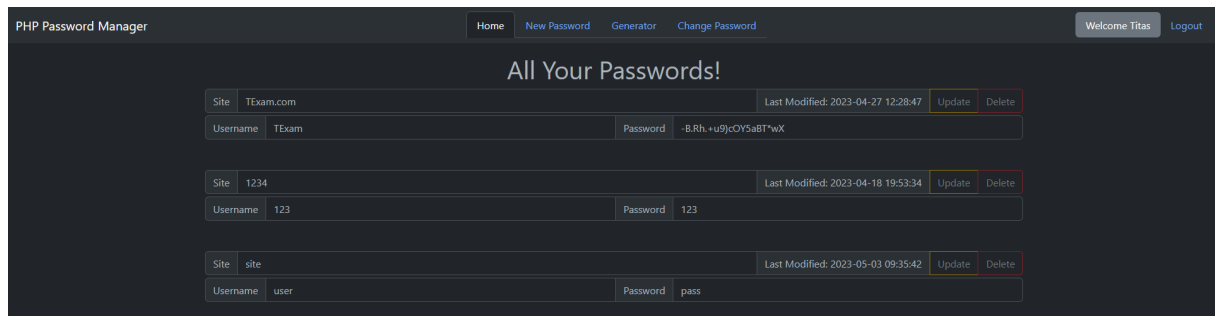


Figure 35: *new password* Home with the new password

19	9	25 gzMSDtmqKqm9LB1taRt5w==	site	2023-05-03 09:35:42	user
----	---	----------------------------	------	---------------------	------

Figure 36: *new password* Database with new password

## 2.7 Home Page

```
<?php
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/PrintMessages.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/Database.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utils/PwdUtils.php");

if (isset($_POST['update'])) {
    $pid = $_POST['pid'];
    $site = $_POST['site'];
    $username = $_POST['username'];
    $password = PwdUtils::encryptPassword($_POST['password']);

    updatePassword($pid, $site, $username, $password);
} elseif (isset($_POST['delete'])) {
    $pid = $_POST['pid'];
    deletePassword($pid);
}

loadPasswords();

/**
 * Loads all of the passwords of the specific user
 * @return void
 */
function loadPasswords(): void {
    $uid = $_SESSION['uid'];

    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("SELECT * FROM ppm.passwords WHERE uid = :uid");
        $stmt->bindParam(":uid", $uid);

        $stmt->execute();

        if ($stmt->rowCount() <= 0) {
            print("<h2 class='text-center'>You don't have any passwords yet!</h2>");
            return;
        }

        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
```



```

        $pid = $row['pid'];
        $site = $row['site_name'];
        $username = $row['username'];
        $password = PwdUtils::decryptPassword($row['password']);
        $creation_date = $row['creation_date'];
        createPasswordForm($pid, $site, $username, $password, $creation_date);
    }
} catch (PDOException $e) {
    PrintMessages::printError("Database Error");
}
}

/**
 * Generates a form for a password that can be edited or deleted
 * @param int $pid Password ID
 * @param string $site Site or app name
 * @param string $username Username for the password
 * @param string $password Password
 * @param string $creation_date Date the password was created or modified
 * @return void
 */
function createPasswordForm(int $pid, string $site, string $username, string
$password, string $creation_date): void {
    echo '
        <div class="container mb-5">
            <form role="form" method="post" id="passwordForm" action="" .
htmlspecialchars($_SERVER["PHP_SELF"]) . ">
                <input type="hidden" id="pid" name="pid" value="" . $pid . ">
                <div class="input-group mb-1">
                    <span class="input-group-text">Site</span>
                    <input type="text" aria-label="Website" class="form-control"
id="site" name="site"
                        value="" . $site . ">
                    <span class="input-group-text" id="lastModified">Last Modified:
' . $creation_date . '</span>
                    <button class="btn btn-outline-secondary border-warning-subtle"
type="submit" name="update">Update</button>
                    <button class="btn btn-outline-secondary border-danger-subtle"
type="submit" name="delete">Delete</button>
                </div>
                <div class="input-group mb-1">
                    <span class="input-group-text">Username</span>
                    <input type="text" aria-label="Username" class="form-control"
id="username"
                        name="username" value="" . $username . ">
                    <span class="input-group-text">Password</span>
                    <input type="text" aria-label="password" class="form-control"
id="password"
                        name="password" value="" . $password . ">
                </div>
            </form>
        </div>';
}

/**
 * Updates an existing password in the database

```

```

* @param int $pid The password ID
* @param string $site The site or app name
* @param string $username The username for the password
* @param string $password The password
* @return void
*/
function updatePassword(int $pid, string $site, string $username, string $password):
void {
    $current_time = date("Y-m-d H:i:s");

    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("UPDATE passwords SET site_name = :site, username
= :user, password = :pwd, creation_date = :date WHERE uid = :uid AND pid = :pid");
        $stmt->bindParam(":uid", $_SESSION['uid']);
        $stmt->bindParam(":pid", $pid);
        $stmt->bindParam(":site", $site);
        $stmt->bindParam(":user", $username);
        $stmt->bindParam(":pwd", $password);
        $stmt->bindParam(":date", $current_time);

        $stmt->execute();
        unset($stmt);
        unset($db);
    } catch (PDOException $e) {
        PrintMessages::printError("Database Error");
    }
}

/**
 * Deletes a password from the database
 * @param int $pid The password ID
 * @return void
 */
function deletePassword(int $pid): void {
    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("DELETE FROM passwords WHERE pid = :pid AND uid
= :uid");
        $stmt->bindParam(":uid", $_SESSION['uid']);
        $stmt->bindParam(":pid", $pid);

        $stmt->execute();
        unset($stmt);
        unset($db);
    } catch (PDOException $e) {
        PrintMessages::printError("Database Error");
    }
}

```

### 2.7.1 No Passwords

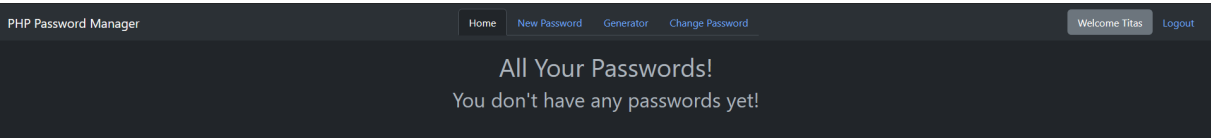


Figure 37: *home* Home with no passwords

2.7.2 Passwords

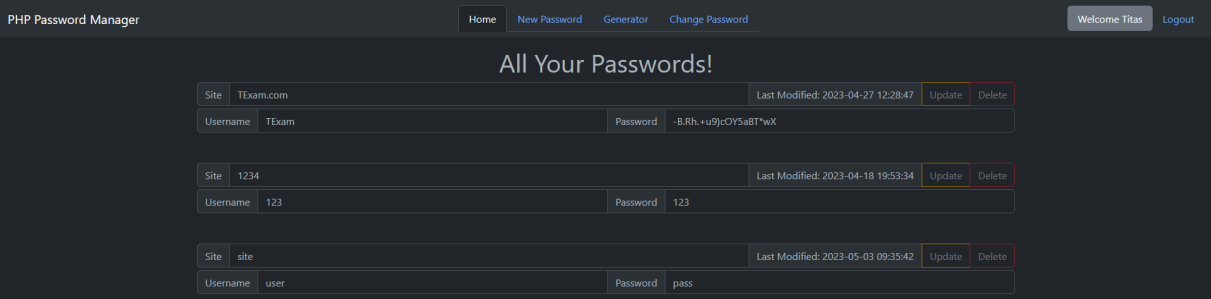


Figure 38: *home* Home with passwords

2.7.3 Change Password Details

Changing site and username first.

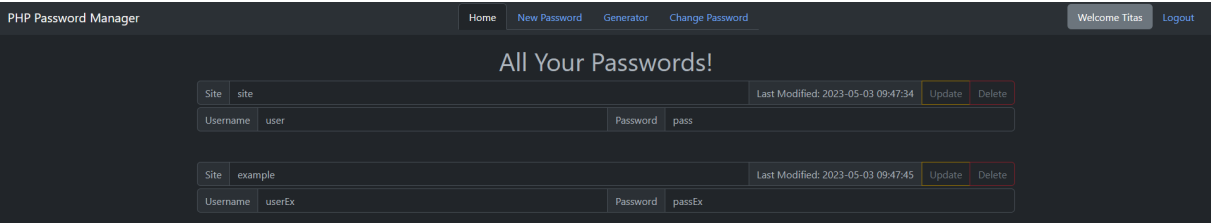


Figure 39: *home* Home before change first password

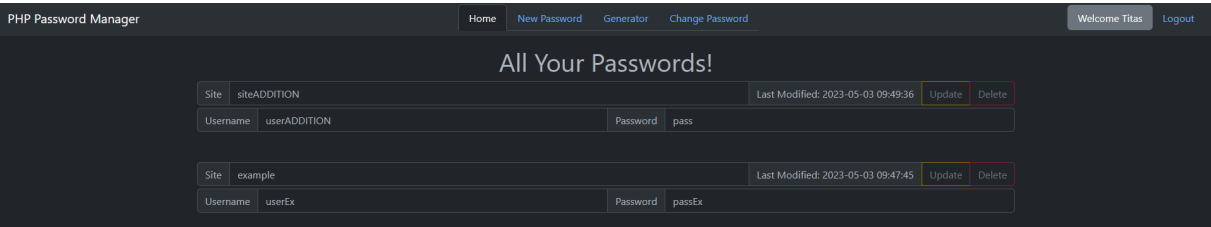


Figure 40: *home* Home after change first password

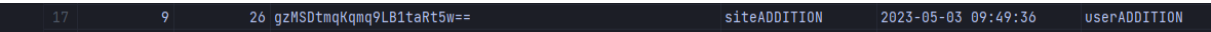


Figure 41: *home* Home after change first password in db

Changing password.

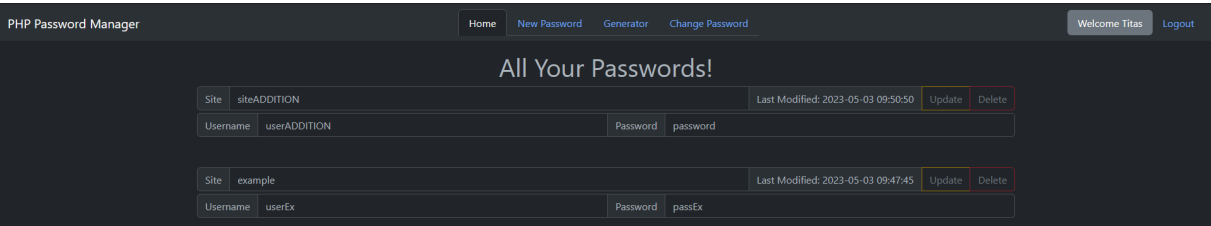


Figure 42: *home* Home after change password from 'pass' to 'password'



Figure 43: *home* Home after change password from 'pass' to 'password' in the db

## 2.7.4 Delete Password

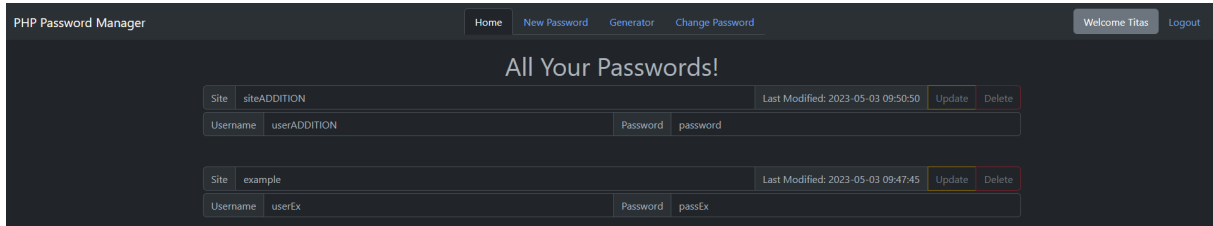


Figure 44: *home* Home before deleting first password

WHERE uid=9 ORDER BY

	uid	pid	password	site_name	creation_date	username
1	9	26	TaEa32eTZHCehsgzu6je+Q==	siteADDITION	2023-05-03 09:50:50	userADDITION
2	9	27	PLi60r8X2cZtMzJuLKAiQw==	example	2023-05-03 09:47:45	userEx

Figure 45: *home* Home before deleting first password in the db

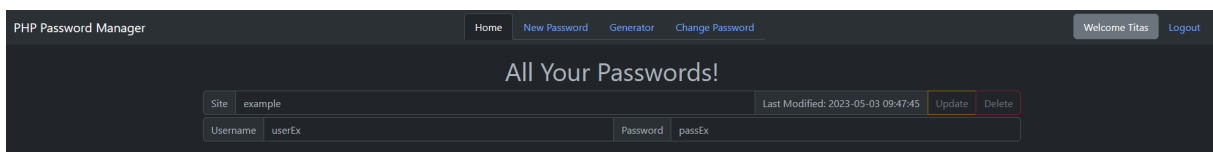


Figure 46: *home* Home after deleting first password

WHERE uid=9 ORDER BY

	uid	pid	password	site_name	creation_date	username
1	9	27	PLi60r8X2cZtMzJuLKAiQw==	example	2023-05-03 09:47:45	userEx

Figure 47: *home* Home after deleting first password in the db

## 2.8 Change Password

```
<?php
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/PrintMessages.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/Database.php");
include_once($_SERVER['DOCUMENT_ROOT'] . "/app/utills/PwdUtils.php");

if (isset($_POST['submit'])) {
    if (empty($_POST['password'])) {
        PrintMessages::printError("Empty Password");
    } elseif (empty($_POST['repeatPassword'])) {
        PrintMessages::printError("Empty Repeated Password");
    } elseif ($_POST['repeatPassword'] !== $_POST['password']) {
        PrintMessages::printError("Passwords do not match");
    } else {
        $password = $_POST['password'];
        changePassword($password);
    }
}

function changePassword($password): void {
    $new_master = PwdUtils::generateMasterKey();
    $passwords = array();

    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("SELECT pid, password FROM passwords WHERE uid
= :uid");
```

```

$stmt->bindParam(":uid", $_SESSION['uid']);

$stmt->execute();

while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
    $pid = $row['pid'];
    $old_password = PwdUtils::decryptPassword($row['password']);
    $passwords[$pid] = $old_password;
}
} catch (PDOException $e) {
    PrintMessages::printError("Database Error");
}

$_SESSION['master_key'] = $new_master;

foreach ($passwords as $pid => $old_password) {
    updatePassword($pid, $old_password);
}

try {
    $new_master_encrypted = PwdUtils::encryptMaster($new_master, $password);
    $new_password_hashed = password_hash($password, PASSWORD_DEFAULT);

    $db = new Database();
    $conn = $db->getConnection();
    $stmt = $conn->prepare("UPDATE user SET master_key = :master, pwd = :pwd
WHERE uid = :uid");
    $stmt->bindParam(":uid", $_SESSION['uid']);
    $stmt->bindParam(":master", $new_master_encrypted);
    $stmt->bindParam(":pwd", $new_password_hashed);

    $stmt->execute();
} catch (PDOException $e) {
    PrintMessages::printError("Database Error");
}
}

/**
 * Updates the password in the database
 * @param string $pid Password ID
 * @param string $password Plain-text Password
 * @return void
 */
function updatePassword(string $pid, string $password): void {
    $current_time = date("Y-m-d H:i:s");
    $encrypted_password = PwdUtils::encryptPassword($password);

    try {
        $db = new Database();
        $conn = $db->getConnection();
        $stmt = $conn->prepare("UPDATE passwords SET password = :pwd, creation_date
= :date WHERE uid = :uid AND pid = :pid");
        $stmt->bindParam(":uid", $_SESSION['uid']);
        $stmt->bindParam(":pid", $pid);
        $stmt->bindParam(":pwd", $encrypted_password);
        $stmt->bindParam(":date", $current_time);
    }
}

```

```

$stmt->execute();
} catch (PDOException $e) {
    PrintMessages::printError("Database Error");
}
}

```

### 2.8.1 Missing Fields

Figure 48: *change password* Missing password

Figure 49: *change password* Missing repeated password

### 2.8.2 Passwords Do Not Match

Figure 50: *change password* Passwords do not match

### 2.8.3 Success

	uid	master_key	pwd	username
1	9	hhKHXr7nC5mZi8Tj0PwbNnPg109dbobUoRHImfqo6XsRktnmid0nfl01CjbQNY0+Rxerg...	\$2y\$10\$AhrG9ed5gosnuZkzwMYp0u0G0D.JAmBgw5gaSq5Wa9SRMcA.FXQK	Titas

Figure 51: *change password* User table before changing password

	uid	pid	password	site_name	creation_date	username
1	9	27	PLi60r8X2cZtMzJuLKAiQw==	example	2023-05-03 09:47:45	userEx

Figure 52: *change password* Passwords table before changing password

Input:

- Password: Titas
- Repeated Password: Titas

Figure 53: *change password* After successful change

WHERE uid=9 ORDER BY

	uid	master_key	pwd	username
1	9	388TgGiSfoDihL0DooVVAYcRuF4dQC63zxc0QB15Lg1/H0fthNa7rgvC/1PtPoIlmL5aK...	\$2y\$10\$Yk4FY0yPQi.rKdA9tbiT..hhD00oDWV7zt8CzK/8v.y0vn426x.9C	Titas

Figure 54: *change password* User table after changing password

WHERE uid=9 ORDER BY

	uid	pid	password	site_name	creation_date	username
1	9	27	DqSCWXk+1YiQ2EpE0Pbegw==	example	2023-05-03 10:02:05	userEx

Figure 55: *change password* Passwords table after changing password