Web Applications Security		
Kamil Gabrysiewicz	Index: 95400	Grupa: 2.1
Wtorek 11:45-13:15	Semestr 2	Laboratorium 4

Task 4.1. & Task 4.2.

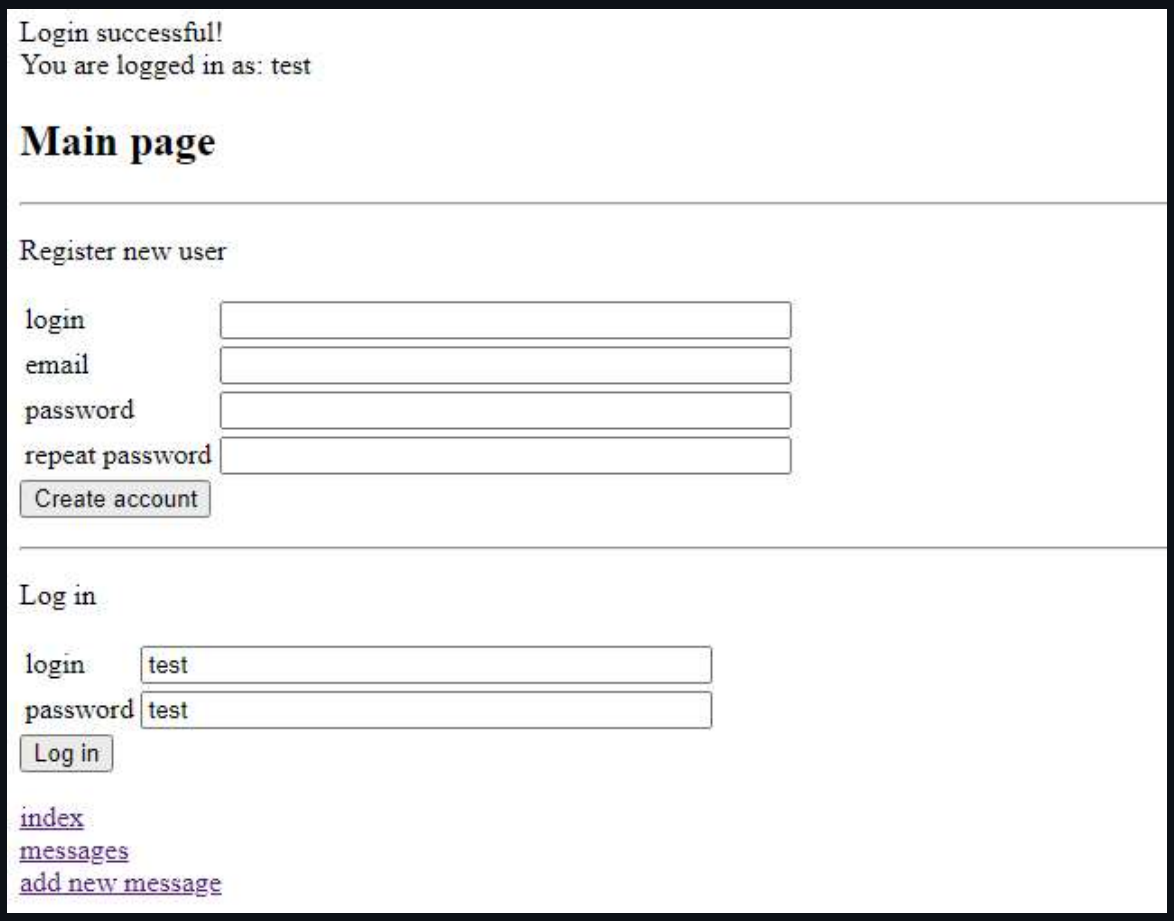
Implement the ability to create user accounts and log in in the application. Check out how you can use different hash functions to store your passwords.

Modify login and user account creation to use salt to store passwords.

```
<?php
require './htmlpurifier-4.15.0/library/HTMLPurifier.auto.php';
```

```
class Pdo_
{
    private $pdo;
    private $purifier;

    public function __construct($server, $user, $pass, $db) {
        $config = HTMLPurifier_Config::createDefault();
        $this->purifier = new HTMLPurifier($config);
        try {
            $this->pdo = new PDO("mysql:host=$server;dbname=$db;charset=utf8", $user, $pass);
            $this->pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        } catch (PDOException $e) {
            echo "Connection failed: " . $e->getMessage();
            die();
        }
    }
    public function add_user($login, $email, $password){
        $login=$this->purifier->purify($login);
        $email=$this->purifier->purify($email);
        $salt = bin2hex(random_bytes(8)); // 16 characters
        try {
            $sql = "INSERT INTO `user`( `login`, `email`, `hash`, `salt`, `id_status`,
`password_form`) VALUES (:login, :email, :hash, :salt, :id_status, :password_form)";
            //hash password
            $hashedPassword = hash('sha512', $salt . $password);

            $data = [
                'login' => $login,
                'email' => $email,
                'hash' => $hashedPassword,
                'salt' => $salt,
                'id_status'=>'1',
                'password_form'=>'1'
            ];
            $this->pdo->prepare($sql)->execute($data);
        } catch (Exception $e) {
            echo "Adding new user failed: " . $e->getMessage();
        }
    }
    public function log_user_in($login, $password){
        $login = $this->purifier->purify($login);

        try {
            $sql = "SELECT id, hash, salt, login FROM user WHERE login = :login";
            $stmt = $this->pdo->prepare($sql);
            $stmt->execute(['login' => $login]);
            $user_data = $stmt->fetch(PDO::FETCH_ASSOC);

            if ($user_data) {
                // Retrieve the stored salt and hash
                $salt = $user_data['salt'];
                $storedHash = $user_data['hash'];

                // Hash the entered password with the stored salt
                $hashedPassword = hash('sha512', $salt . $password);

                // Compare the hashed password to the stored hash
                if ($hashedPassword === $storedHash) {
                    echo 'Login successful! <br/>';
```

```
                    echo 'You are logged in as: ' . htmlspecialchars($user_data['login']) .
'<br/>';
                } else {
                    echo 'Login FAILED<br/>';
                }
            } else {
                echo 'Login FAILED: User not found<br/>';
            }

        } catch (Exception $e) {
            echo "Login attempt failed: " . $e->getMessage();
        }
    }
}
```



Task 4.3.

Implement the ability to change the user's password in the application. Remember that changing the password involves generating a new salt value.

`Pdo_.php`

```
// Method to change the user's password
    public function change_password($login, $old_password, $new_password) {
        $login = $this->purifier->purify($login);

        try {
```

```
            // Verify old password
            $sql = "SELECT id, hash, salt FROM user WHERE login = :login";
            $stmt = $this->pdo->prepare($sql);
            $stmt->execute(['login' => $login]);
            $user_data = $stmt->fetch(PDO::FETCH_ASSOC);

            if ($user_data) {
                $storedHash = $user_data['hash'];
                $storedSalt = $user_data['salt'];

                // Hash the old password with the stored salt
                $hashedOldPassword = hash('sha512', $storedSalt . $old_password);

                // Verify the old password
                if ($hashedOldPassword === $storedHash) {
                    // Generate new salt and hash for the new password
                    $newSalt = bin2hex(random_bytes(8)); // Generate a new salt
                    $newHash = hash('sha512', $newSalt . $new_password);

                    // Update the database with the new hash and salt
                    $updateSql = "UPDATE user SET hash = :newHash, salt = :newSalt WHERE id =
:id";
                    $updateStmt = $this->pdo->prepare($updateSql);
                    $updateStmt->execute([
                        'newHash' => $newHash,
                        'newSalt' => $newSalt,
                        'id' => $user_data['id']
                    ]);

                    echo 'Password changed successfully!';
                } else {
                    echo 'Password change FAILED: Incorrect current password.';
                }
            } else {
                echo 'Password change FAILED: User not found.';
            }

        } catch (Exception $e) {
            echo "Password change attempt failed: " . $e->getMessage();
        }
    }
}
```

Change password of

```
user: test
old password:test
new password: test2
new password repeat: test2
```

Main page

Register new user

login
email
password
repeat password
Create account

Log in

login test123
password student
Log in

Change Password

Login test
Current Password ••••
New Password •••••
Repeat New Password •••••
Change Password

index
messages
add new message

Successfull change of user's password

Password changed successfully!

Main page

Register new user

login
email
password
repeat password
Create account

Log in

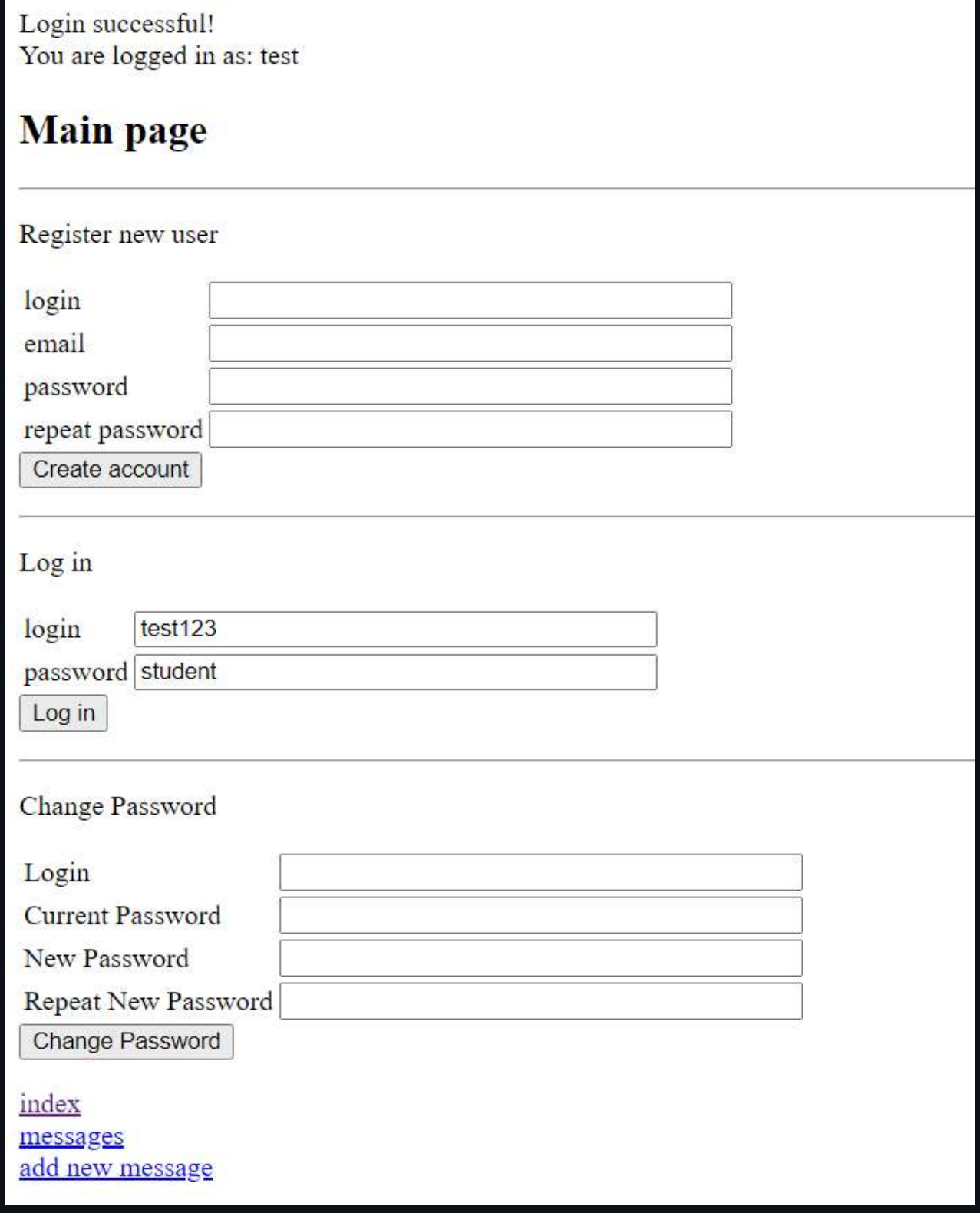
login test123
password student
Log in

Change Password

Login
Current Password
New Password
Repeat New Password
Change Password

index
messages
add new message

Login to user: test with new credentials



Task 4.4.

Add functionality to your application to encrypt password hashes before saving them to the database. Analyze the security of the Aes.php class in Listing 5.3. Were you correct in storing the cryptographic key and initialization vector in the code?

In my application, password hashing before inserting it into the database was implemented from the beginning. Also, listing 5.3 does not exist, while listing 4.3 discusses lifetimes and OTPs. Therefore, I assume the task is about adding password hashing before executing the query to the database.

```
public function add_user($login, $email, $password){
        $login=$this->purifier->purify($login);
        $email=$this->purifier->purify($email);
        $salt = bin2hex(random_bytes(8)); // 16 characters
        try {
            $sql = "INSERT INTO `user`( `login`, `email`, `hash`, `salt`, `id_status`,
`password_form`) VALUES (:login, :email, :hash, :salt, :id_status, :password_form)";
            //hash password
            $hashedPassword = hash('sha512', $salt . $password);

            $data = [
                'login' => $login,
                'email' => $email,
                'hash' => $hashedPassword,
                'salt' => $salt,
                'id_status'=>'1',
                'password_form'=>'1'
            ];
            $this->pdo->prepare($sql)->execute($data);
        } catch (Exception $e) {
            echo "Adding new user failed: " . $e->getMessage();
        }
    }
```

Task 4.5.

Add the mechanisms presented in this lab to your application and implement two-factor authentication. In addition to verifying the login and password, send the user a one-time code. Use an independent communication channel to submit.

Main page

Register new user

login
email
password
repeat password
Create account

Log in

login test
password test2
Log in

Code
Verify

Change Password

Login
Current Password
New Password
Repeat New Password
Change Password

index
messages
add new message

Main page

Register new user

login
email
password
repeat password
Create account

Log in

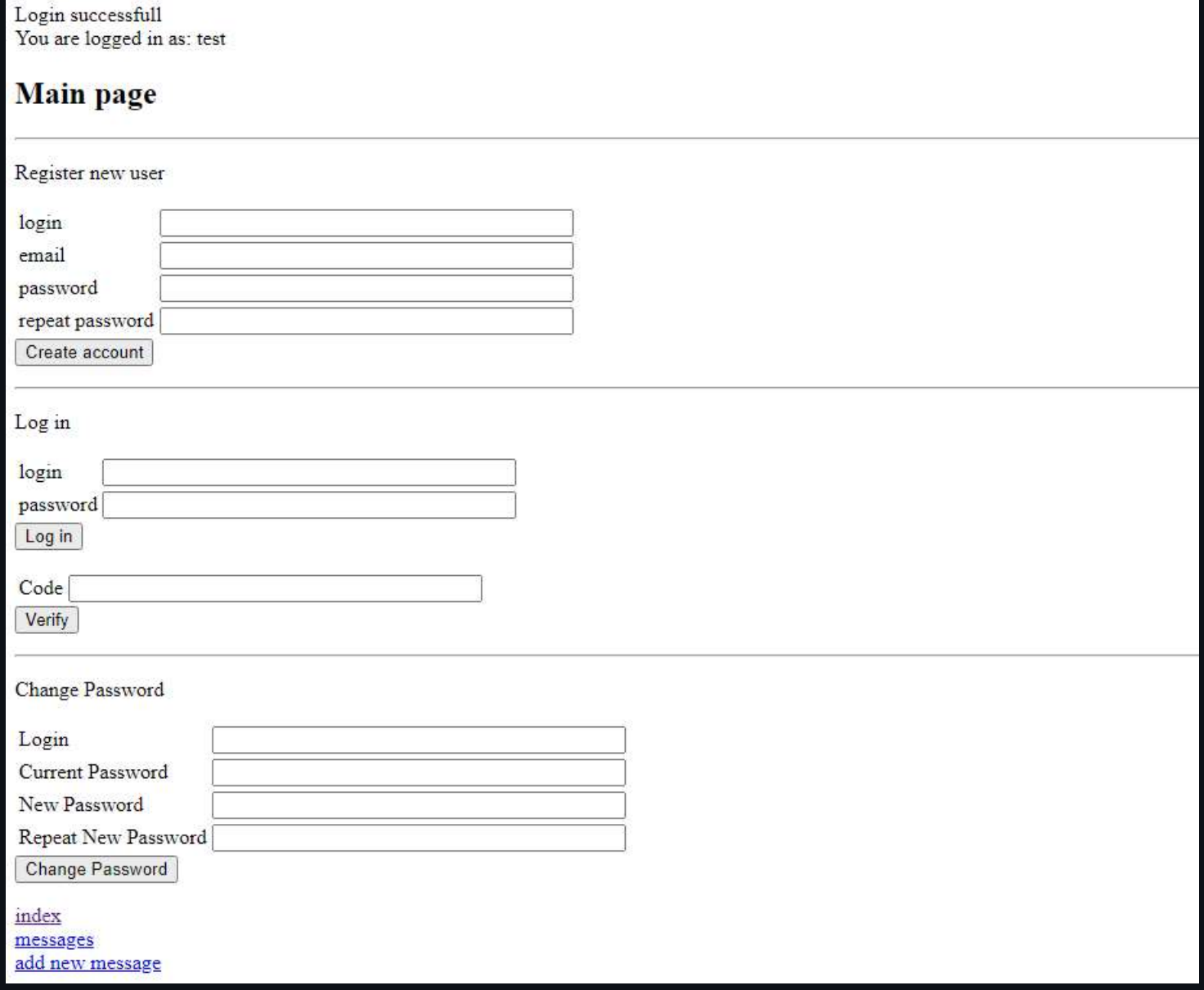
login
password
Log in

Code 191114
Verify

Change Password

Login
Current Password
New Password
Repeat New Password
Change Password

index
messages
add new message



Task 4.6. & Task 4.7.

A good level of security offered by the login mechanism is not everything. It happens that the user forgets to log out of the application after finishing work. Then the next computer user has a chance to use the started session. Protect your application against this possibility. Inside the session, set a variable that determines the session expiration time. Let this time be 5 minutes. When the user logs in, set the session expiration time to now()=5 minutes. When switching to the next page or any other operation in the session, perform the same operation before checking whether the session has not expired. If the session has expired, set the session status to "not logged in" and redirect the user to the login page by displaying an appropriate message.

Implement the session mechanism on all pages of the application. Display the user's login information on the page or show 'Not logged in' if the user is not authenticated.

After logging in with 2FA, a session expiration date is set for 5 minutes. After this time, the user is automatically logged out. Each page now manages the expiration date by either refreshing it or verifying it.

session_expiration: 1731324349
Logged in: 1
Login successful!
You are logged in as: test
You are logged in as: test

Main page

Register new user

login
email
password
repeat password
Create account

Log in

login
password
Log in

Code
Verify

Change Password

Login
Current Password
New Password
Repeat New Password
Change Password

index
messages
add new message

session_expiration: 1731325918
Logged in: 1

Messages

New Intel technology	' OR '1'='1	Edit
Intel shares raising	brokers announce: Intel shares will go up!	Edit
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	Edit
Airplane crash	A passenger plane has crashed in Europe	Edit
Coronavirus	A new version of virus was found!	Edit
Bitcoin price raises	Price of bitcoin reaches new record.	Edit
New Windows announced	A new version of windows was announced. Present buyers of Widows 10 can update the system to the newest version for free.	Edit
test	test	Edit
XSS 1	XSS 1 Test	Edit
XSS 1	Shady website	Edit
nothing	nothing	Edit
nothing	nothing	Edit
nothing	nothing	Edit
XSS 2 b	XSS 1 Test	Edit
XSS 2 B	Shady website	Edit
XSS 2 C		Edit

Navigation

index
messages
add new message

session_expiration: 1731325758
Logged in: 1

Add Message

Name

Type Public

Message Content

Add Message

Navigation

index
messages
add new message

session_expiration: 1731325857
Logged in: 1

Edit Message

Name	New Intel technology
Type	Public
Message Content	' OR '1'='1

Update Message

Navigation

index
messages
add new message

If the session expires, the user is logged out, and the session is destroyed.

session_expiration: null
Logged in: null

Warning: Undefined array key "login" in **/var/www/html/index.php** on line **67**

Warning: Trying to access array offset on value of type bool in **/var/www/html/classes/Pdo_.php** on line **199**
login FAILED

Main page

Register new user

login
email
password
repeat password
Create account

Log in

login
password
Log in

Code
Verify

Change Password

Login
Current Password
New Password
Repeat New Password
Change Password

index
messages
add new message