

🐙 Gabrysiewicz Update README.md 2d78ce1 · now ⌚

514 lines (468 loc) · 19.8 KB

Preview

Code

Blame

Raw



## Web Applications Security



Kamil Gabrysiewicz

Index: 95400

Grupa: 2.1

Wtorek 11:45-13:15

Semestr 2

Laboratorium 2

# Tasks to do:

## Task 2.1.

Create a "news" database. Inside, create a "message" table and fill it with data. Use the script in Listing 3.1 to create the table.

→ ~ docker ps



CONTAINER ID	IMAGE	COMMAND	CREATED	
STATUS	PORTS		NAMES	
19cb67f88ac5	php:8.2-apache	"docker-php-entrypoi..."	8 minutes ago	Up
8 minutes	0.0.0.0:8080->80/tcp, [::]:8080->80/tcp		php-app	
7a69e8d62d1a	mysql:8.0	"docker-entrypoint.s..."	8 minutes ago	Up
8 minutes	3306/tcp, 33060/tcp		mysql-db	

mysql> SHOW TABLES;



```
+-----+
| Tables_in_mydb |
+-----+
| message        |
| user           |
+-----+
```

mysql> SELECT id, login, email, hash FROM user LIMIT 3;



```
+-----+-----+-----+-----+
| id | login | email          | hash                               |
+-----+-----+-----+-----+
| 1  | john  | johny@gmail.com | 552d29f9290b9521e6016c2296fa4511 |
| 2  | susie | susie@gmail.com | 8c90f286786c7f3b96564e1e88e0ddab |
| 3  | anie  | anie@gmail.com  | dcb710a566c2a24c8bfaf83618e728f7 |
+-----+-----+-----+-----+
```

mysql> SELECT \* FROM message LIMIT 3;



```
+-----+-----+-----+-----+
| id | name                               | type   | message                               |
| deleted |
+-----+-----+-----+-----+
| 1  | New Intel technology               | public | Intel has announced a new           |
processor for desktops              | 0      |
| 2  | Intel shares raising               | private | brokers announce: Intel             |
shares will go up!                  | 0      |
| 3  | New graphic card from NVidia       | public  | NVidia has announced a new         |

```

1. Intel has announced a new processor for desktops
2. brokers announce: Intel shares will go up!
3. NVidia has announced a new graphic card for desktops
4. A passenger plane has crashed in Europe
5. A new version of virus was found!
6. Price of bitcoin reaches new record.
7. A new version of windows was announced. Present buyers of Widows 10 can update the system to the newest version for free.
8. test
9. test
10. anie@gmail.com
11. johny@gmail.comjohnsF5%gR552d29f9290b9521e6016c2296fa4511
12. johny@gmail.comjohn | sF5%gR | 552d29f9290b9521e6016c2296fa4511
13. susie@gmail.comsusie | j67R | 8c90f286786c7f3b96564e1e88e0ddb

## Task 2.4.

Add the ability to edit messages to the application. Verify what SQLi attacks are possible against the added module.

Message updated successfully.

### Messages

New Intel technology	Lorem Ipsum	<a href="#">Edit</a>
Intel shares raising	brokers announce: Intel shares will go up!	<a href="#">Edit</a>
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	<a href="#">Edit</a>
Airplane crash	A passenger plane has crashed in Europe	<a href="#">Edit</a>
Coronavirus	A new version of virus was found!	<a href="#">Edit</a>
Bitcoin price raises	Price of bitcoin reaches new record.	<a href="#">Edit</a>
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.	<a href="#">Edit</a>
test	test	<a href="#">Edit</a>
(SELECT email FROM `user` WHERE 1 LIMIT 1) test	test	<a href="#">Edit</a>
test	anie@gmail.com	<a href="#">Edit</a>
test	johny@gmail.comjohnsF5%gR552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	johny@gmail.comjohn   sF5%gR   552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	susie@gmail.comsusie   j67R   8c90f286786c7Bb96564e1e88e0ddab	<a href="#">Edit</a>

### Navigation

[index](#)  
[messages](#)  
[add new message](#)

## Edit Message

Name

Type

Message Content

```
test','public',(SELECT email FROM `user`  
WHERE 1 LIMIT 1),0)#
```

### Navigation

[index](#)  
[messages](#)  
[add new message](#)

Message updated successfully.

#### Messages

New Intel technology	test,'public',(SELECT email FROM `user` WHERE 1 LIMIT 1),0)#	<a href="#">Edit</a>
Intel shares raising	brokers announce: Intel shares will go up!	<a href="#">Edit</a>
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	<a href="#">Edit</a>
Airplane crash	A passenger plane has crashed in Europe	<a href="#">Edit</a>
Coronavirus	A new version of virus was found!	<a href="#">Edit</a>
Bitcoin price raises	Price of bitcoin reaches new record.	<a href="#">Edit</a>
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.	<a href="#">Edit</a>
test	test	<a href="#">Edit</a>
(SELECT email FROM `user` WHERE 1 LIMIT 1)	test	<a href="#">Edit</a>
test	anie@gmail.com	<a href="#">Edit</a>
test	johny@gmail.comjohnsF5%gR552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	johny@gmail.comjohn   sF5%gR   552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	susie@gmail.comsusie   j67R   8c90f286786c7f3b96564e1e88e0ddab	<a href="#">Edit</a>

#### Navigation

[index](#)  
[messages](#)  
[add new message](#)

In my case, The update edit made in unsuccessful to perform SQL Injection. Its probably due to prepared statements

```
$stmt = $this->mysqli->prepare("UPDATE message SET name = ?, type = ?,  
message = ? WHERE id = ?");
```



inside of

```
public function updateMessage($id, $name, $type, $content) {  
    $stmt = $this->mysqli->prepare("UPDATE message SET name = ?, type =  
    ?, message = ? WHERE id = ?");  
  
    if ($stmt) {  
        $stmt->bind_param("sssi", $name, $type, $content, $id);  
        $stmt->execute();  
        $stmt->close();  
        return true;  
    } else {  
        printf("Error updating message: %s\n", $this->mysqli->error);  
        return false;  
    }  
}
```



Its creating a query that separates the SQL code structure from the user input data.

## Task 2.5.

---

Add a new database user. Define the minimum required set of permissions for it (show what set it is in the report). Use the newly created account to connect the application to the database. Verify what has changed in terms of application security. Tip. Try the attack in Figure 3.10 again

### Creation of an user

```
mysql> CREATE USER 'new_user'@'172.20.0.3' IDENTIFIED BY 'user_password';  
Query OK, 0 rows affected (0.01 sec)
```



### Granting privileges

```
mysql> GRANT SELECT, INSERT, UPDATE ON mydb.* TO 'new_user'@'172.20.0.3';  
Query OK, 0 rows affected (0.01 sec)
```



### Apply privileges

```
mysql> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.01 sec)
```



## Add message

Name

Type

Message content

## Navigation

[index](#)  
[messages](#)  
[add new message](#)

INSERT INTO message (name, type, message, deleted) VALUES ('test',(SELECT 'public' WHERE (SELECT @@version) LIKE '10.4.22-MariaDB' LIMIT 1),'test,0','public','version',0)  
**Fatal error:** Uncaught mysqli\_sql\_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'test,(SELECT 'public' WHERE (SELECT @@version) LIKE '10.4.22-MariaDB' LIMIT 1),' at line 1 in /var/www/html/classes/Db.php:34 Stack trace: #0 /var/www/html/classes/Db.php(34): mysqli->query('INSERT INTO mes...') #1 /var/www/html/messages.php(12): Db->addMessage('test,(SELECT ...','public','version') #2 (main) thrown in /var/www/html/classes/Db.php on line 34

In my case I get an error. Which on one hand is bad but on the other does not reveals the version of my database version, which in this case should return:

```
mysql> SELECT @@version;
+-----+
| @@version |
+-----+
| 8.0.39    |
+-----+
```

or

```
mysql> SELECT @@version LIKE '10.4.22-MariaDB' LIMIT 1;
+-----+
| @@version LIKE '10.4.22-MariaDB' |
+-----+
|                                0 |
+-----+
1 row in set (0.00 sec)
```

But even if it would pass, it probably should be fixed with the use of prepared statements

```
$stmt = $this->mysqli->prepare("INSERT INTO message (`name`, `type`,  
`message`, `deleted`) VALUES (?, ?, ?, ?)");  
$stmt->bind_param("sssi", $name, $type, $content, $deleted);  
$stmt->execute();
```



## Task 2.6.

Modify the application. Use only PDO to connect to the database. Place the code for handling the database in the Db class. In each case, the data should be retrieved by a dedicated function that is called in the PHP page code. The function should return the page a set of data to display

```
apt-get update
```



```
apt-get install -y libpng-dev libjpeg-dev libfreetype6-dev libzip-dev && \  
docker-php-ext-configure gd --with-freetype --with-jpeg && \  
docker-php-ext-install gd pdo pdo_mysql
```



```
class Db {  
    private $pdo;  
    private $select_result;  
  
    public function __construct($server, $user, $pass, $db) {  
        try {  
            // Create a new PDO instance  
            $this->pdo = new  
PDO("mysql:host=$server;dbname=$db;charset=utf8", $user, $pass);  
            $this->pdo->setAttribute(PDO::ATTR_ERRMODE,  
PDO::ERRMODE_EXCEPTION);  
        } catch (PDOException $e) {  
            echo "Connection failed: " . $e->getMessage();  
            exit();  
        }  
    }  
  
    public function select($sql) {  
        $results = [];  
        try {  
            $stmt = $this->pdo->query($sql);  
            while ($row = $stmt->fetch(PDO::FETCH_OBJ)) {  
                $results[] = $row;  
            }  
        }  
    }  
}
```





```

    }
    $this->select_result = $results;
    return $results;
} catch (PDOException $e) {
    echo "Select failed: " . $e->getMessage();
    return false;
}
}

public function addMessage($name, $type, $content) {
    $sql = "INSERT INTO message (`name`, `type`, `message`, `deleted`)
VALUES (:name, :type, :content, 0)";
    try {
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(':name', $name);
        $stmt->bindParam(':type', $type);
        $stmt->bindParam(':content', $content);
        return $stmt->execute();
    } catch (PDOException $e) {
        echo "Add message failed: " . $e->getMessage();
        return false;
    }
}

public function getMessage($message_id) {
    foreach ($this->select_result as $message) {
        if ($message->id == $message_id) {
            return $message->message;
        }
    }
}

public function updateMessage($id, $name, $type, $content) {
    $sql = "UPDATE message SET name = ?, type = ?, message = ? WHERE id
= ?";
    try {
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(1, $name);
        $stmt->bindParam(2, $type);
        $stmt->bindParam(3, $content);
        $stmt->bindParam(4, $id);
        $stmt->execute();
        return true;
    } catch (PDOException $e) {
        echo "Error updating message: " . $e->getMessage();
        return false;
    }
}

```

```

public function __destruct() {
    $this->pdo = null; // Close the PDO connection
}
}
?>

```

Message updated successfully.

#### Messages

New Intel technology	' OR '1'=1	<a href="#">Edit</a>
Intel shares raising	brokers announce: Intel shares will go up!	<a href="#">Edit</a>
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	<a href="#">Edit</a>
Airplane crash	A passenger plane has crashed in Europe	<a href="#">Edit</a>
Coronavirus	A new version of virus was found!	<a href="#">Edit</a>
Bitcoin price raises	Price of bitcoin reaches new record.	<a href="#">Edit</a>
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.	<a href="#">Edit</a>
test	test	<a href="#">Edit</a>
(SELECT email FROM `user` WHERE 1 LIMIT 1) test		<a href="#">Edit</a>
test	anie@gmail.com	<a href="#">Edit</a>
test	johny@gmail.comjohnsF5%gR552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	johny@gmail.comjohn   sF5%gR   552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	susie@gmail.comsusie   j67R   8c90f286786c7f3b96564e1e88e0ddab	<a href="#">Edit</a>
Test	Test	<a href="#">Edit</a>
PDO TEST 2	PDO TEST 2	<a href="#">Edit</a>

#### Navigation

[index](#)  
[messages](#)  
[add new message](#)

## Task 2.7.

### Include user input filtering in your application

To filter user input, I decided to add the `htmlspecialchars` and `trim` functions to remove unnecessary characters and convert potentially harmful characters such as `'`, `"`, `/`, and `\` into their harmless equivalents.

#### Messages.php


```

// Adding new message
if (isset($_POST['add_message'])) {
    $name = htmlspecialchars(trim($_POST['name']));
    $type = htmlspecialchars(trim($_POST['type']));
    $content = htmlspecialchars(trim($_POST['content']));

    if (!$db->addMessage($name, $type, $content)) {
        echo "<p style='color:red;'>Adding new message failed.</p>";
    }
}

```

#### Messages.php



```
// Editing existing message
if (isset($_POST['update_message'])) {
    $id = intval($_POST['id']);
    $name = htmlspecialchars(trim($_POST['name']));
    $type = htmlspecialchars(trim($_POST['type']));
    $content = htmlspecialchars(trim($_POST['content']));

    if ($db->updateMessage($id, $name, $type, $content)) {
        echo "<p>Message updated successfully.</p>";
    } else {
        echo "<p style='color:red;'>Updating message failed.</p>";
    }
}
```

message\_edit.php



```
<form method="post" action="messages.php">
    <input type="hidden" name="id" value="<?php echo $message_id; ?>" />
    <table>
        <tr>
            <td>Name</td>
            <td>
                <input type="text" name="name" value="<?php echo
htmlspecialchars($message->name); ?>" size="56" required />
            </td>
        </tr>
        <tr>
            <td>Type</td>
            <td>
                <select name="type">
                    <option value="public" <?php if ($message->type ==
"public") echo "selected"; ?>>Public</option>
                    <option value="private" <?php if ($message->type ==
"private") echo "selected"; ?>>Private</option>
                </select>
            </td>
        </tr>
        <tr>
            <td>Message Content</td>
            <td>
                <textarea required name="content" rows="10" cols="40"><?php
echo htmlspecialchars($message->message); ?></textarea>
            </td>
        </tr>
    </table>
```

```
<input type="submit" value="Update Message" name="update_message"/>
</form>
```

## Task 2.8.

---

Apply whitelist to filter message type. Tip: Notice that the message type is selected by the user from a list. This does not mean, however, that you cannot insert any other string of characters there. This is possible using developer tools that modify the content of the page (in the Chrome browser by pressing the F12 button). Therefore, you need to verify whether the value transferred from this field is included in the set (public, private)

---

Messages.php

```
// Adding new message
if (isset($_POST['add_message'])) {
    $name = htmlspecialchars(trim($_POST['name']));
    $type = htmlspecialchars(trim($_POST['type']));
    $content = htmlspecialchars(trim($_POST['content']));

    // whitelist
    $allowed_types = ['public', 'private'];

    if (in_array($type, $allowed_types)) {
        if (!$db->addMessage($name, $type, $content)) {
            echo "<p style='color:red;'>Adding new message failed.
        </p>";
        }
    } else {
        // Handle invalid type
        echo "<p style='color:red;'>Invalid message type. Please select
a valid option.</p>";
    }
}
```

Invalid message type. Please select a valid option.		
Messages		
New Intel technology	' OR '1'=1	<a href="#">Edit</a>
Intel shares raising	brokers announce: Intel shares will go up!	<a href="#">Edit</a>
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	<a href="#">Edit</a>
Airplane crash	A passenger plane has crashed in Europe	<a href="#">Edit</a>
Coronavirus	A new version of virus was found!	<a href="#">Edit</a>
Bitcoin price raises	Price of bitcoin reaches new record.	<a href="#">Edit</a>
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.	<a href="#">Edit</a>
test	test	<a href="#">Edit</a>
(SELECT email FROM `user` WHERE 1 LIMIT 1) test		<a href="#">Edit</a>
test	anie@gmail.com	<a href="#">Edit</a>
test	johny@gmail.comjohnsF5%gR552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	johny@gmail.comjohn   sF5%gR   552d29f9290b9521e6016c2296fa4511	<a href="#">Edit</a>
test	susie@gmail.comsusie   j67R   8c90f286786c7f3b96564e1e88e0ddab	<a href="#">Edit</a>
Test	Test	<a href="#">Edit</a>
PDO TEST 2	PDO TEST 2	<a href="#">Edit</a>
Navigation		
<a href="#">index</a> <a href="#">messages</a> <a href="#">add new message</a>		

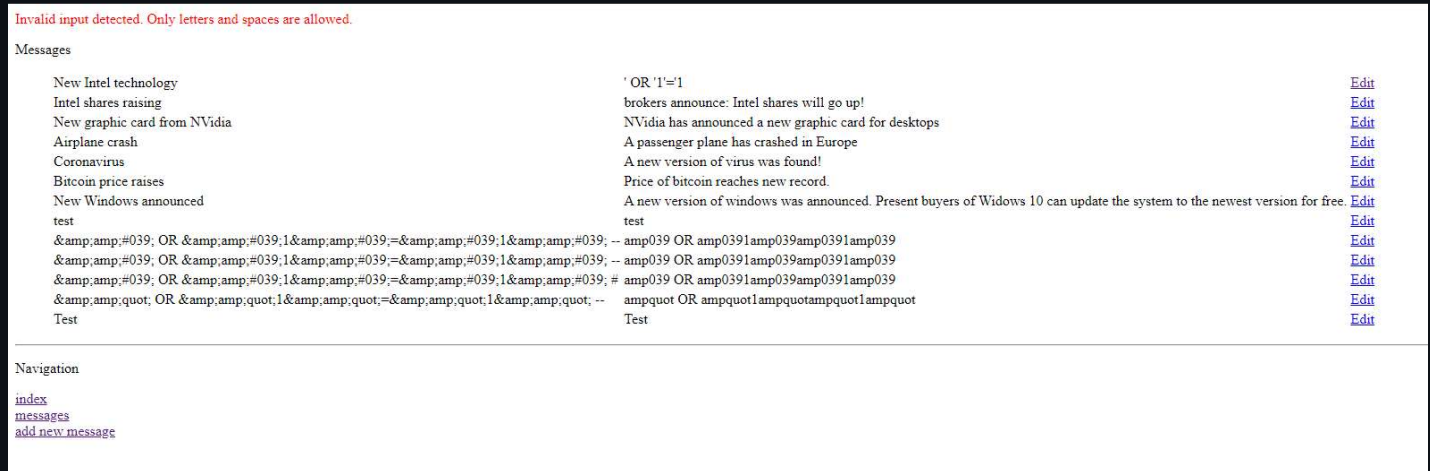
Modify the application. Develop comprehensive application security against SQLi attacks. Create an additional Filter class. Include all the functionalities necessary to filter data. Add filtering functions for data downloaded from the form. In the Db class, modify the functions for adding data to the database so that each database function filters the data before using it. The purpose of these modifications is to protect against programmer mistakes. He will not have to remember to filter data. Each time the database function is called, the filter function will be automatically invoked. Tip: The presented approach to filtering and inserting data into the database may require rebuilding database functions. The name, email address and URL will be filtered differently. This problem can be solved in two ways:

- ## 1. Develop independent functions for different types of data:

- addName
- addURL
- addEmail

2. Develop one generic function for inserting data and pass the type of filter that should be used to the function as a parameter.

At first, I wanted to allow characters like ' ', but I really didn't like the outcome and decided it would be best to forbid these characters.



Filter class has a filter function for each type:

- name
  - Ensures it's a string and doesn't allow characters that are not letters or spaces.
- email
  - Ensures it's a string and uses a built-in validation function.
- type
  - Ensures it's a string and is either `public` or `private`.
- url
  - Ensures it's a string and uses a built-in validation function.
- general
  - Removes any character that is not a word character `\w`, which includes letters, digits, and underscores. `\s` removes whitespace.

Also I am not sure why there are email and url functions, I assumed it had to be type and content so I made type filter and general filter for content.

Filter.php



```
<?php
class Filter {
    public static function filter_name($name) {
        if (!is_string($name)) {
            throw new InvalidArgumentException("Name must be a string.");
        }
        if (preg_match('/^[^a-zA-Z\s]/', $name)) {
            throw new InvalidArgumentException('Invalid input detected.
Only letters and spaces are allowed.');
```

```
        }
        return htmlspecialchars(trim($name));
    }
    // ??
    public static function filter_email($email) {
        if (!is_string($email)) {
            throw new InvalidArgumentException("Email must be a string.");
        }
        // Built-in validation
        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
            throw new InvalidArgumentException("Invalid email format.");
        }
        return htmlspecialchars(trim($email));
    }
    // ??
    public static function filter_url($url) {
        if (!is_string($url)) {
            throw new InvalidArgumentException("URL must be a string.");
        }
        // Built-in validation
        if (!filter_var($url, FILTER_VALIDATE_URL)) {
            throw new InvalidArgumentException("Invalid URL format.");
        }
        return htmlspecialchars(trim($url));
    }

    public static function filter_general($input) {
        if (!is_string($input)) {
            throw new InvalidArgumentException("Input must be a string.");
        }
        // Remove potentially harmful characters
        return preg_replace('/[^\w\s]/', '',
htmlspecialchars(trim($input)));
    }

    public static function filter_type($type) {
        if (!is_string($type)) {
            throw new InvalidArgumentException("Type must be a string.");
```

```

    }
    // Ensure the type is either 'public' or 'private'
    $type = strtolower(htmlspecialchars(trim($type)));
    if (!in_array($type, ['public', 'private'])) {
        throw new InvalidArgumentException("Type must be either
'public' or 'private'.");
    }
    return $type;
}
}
?>

```

Db.php makes use of the Filter class functions, so before inserting or updating data, it throws an exception if necessary. Additionally, SQL queries are **prepared statements** to increase security.

## Db.php

```

<?php
class Db {
    private $pdo; // PDO instance
    private $select_result; // result

    public function __construct($server, $user, $pass, $db) {
        try {
            // Create PDO instance
            $this->pdo = new
PDO("mysql:host=$server;dbname=$db;charset=utf8", $user, $pass);
            $this->pdo->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
        } catch (PDOException $e) {
            echo "Connection failed: " . $e->getMessage();
            exit();
        }
    }

    public function select($sql) {
        $results = [];
        try {
            $stmt = $this->pdo->query($sql);
            while ($row = $stmt->fetch(PDO::FETCH_OBJ)) {
                $results[] = $row;
            }
            $this->select_result = $results;
            return $results;
        }
    }
}

```



```

    } catch (PDOException $e) {
        echo "Select failed: " . $e->getMessage();
        return false;
    }
}

public function addMessage($name, $type, $content) {
    $filtered_name = Filter::filter_name($name);
    $filtered_type = Filter::filter_type($type);
    $filtered_content = Filter::filter_general($content);

    $sql = "INSERT INTO message (`name`, `type`, `message`, `deleted`)
VALUES (:name, :type, :content, 0)";
    try {
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(':name', $filtered_name);
        $stmt->bindParam(':type', $filtered_type);
        $stmt->bindParam(':content', $filtered_content);
        return $stmt->execute();
    } catch (PDOException $e) {
        echo "Add message failed: " . $e->getMessage();
        return false;
    }
}

public function updateMessage($id, $name, $type, $content) {
    $filtered_name = Filter::filter_name($name);
    $filtered_type = Filter::filter_type($type);
    $filtered_content = Filter::filter_general($content);

    $sql = "UPDATE message SET name = ?, type = ?, message = ? WHERE id
= ?";
    try {
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(1, $filtered_name);
        $stmt->bindParam(2, $filtered_type);
        $stmt->bindParam(3, $filtered_content);
        $stmt->bindParam(4, $id);
        $stmt->execute();
        return true;
    } catch (PDOException $e) {
        echo "Error updating message: " . $e->getMessage();
        return false;
    }
}

public function __destruct() {
    $this->pdo = null; // Close the PDO connection

```

```
}  
}  
?>
```

After an update of the application, I also tested it with ZAP, and as a result, I received some warnings but no High risk errors.

### Alert counts by alert type

This table shows the number of alerts of each alert type, together with the alert type's risk level.

(The percentages in brackets represent each count as a percentage, rounded to one decimal place, of the total number of alerts included in this report.)

Alert type	Risk	Count
<a href="#">Content Security Policy (CSP) Header Not Set</a>	Medium	24 (342.9%)
<a href="#">Missing Anti-clickjacking Header</a>	Medium	22 (314.3%)
<a href="#">Information Disclosure - Debug Error Messages</a>	Low	17 (242.9%)
<a href="#">Private IP Disclosure</a>	Low	17 (242.9%)
<a href="#">Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)</a>	Low	22 (314.3%)
<a href="#">Server Leaks Version Information via "Server" HTTP Response Header Field</a>	Low	24 (342.9%)
<a href="#">X-Content-Type-Options Header Missing</a>	Low	22 (314.3%)
Total		7

## Task 2.10.

Verify the vulnerability of the secured application to SQLI attacks. Conduct several selected attacks on the application and present their results..

I tried basic payloads of:

- ' OR '1'='1' --

- ' OR '1'='1' #
- " OR "1"="1" --
- " OR "1"="1" #

Creating a new message and editing existing ones resulted in the following error:

Invalid input detected. Only letters and spaces are allowed.

as it did in earlier task:

Invalid input detected. Only letters and spaces are allowed.

Messages

New Intel technology	' OR '1'=1	<a>Edit</a>
Intel shares raising	brokers announce: Intel shares will go up!	<a>Edit</a>
New graphic card from NVidia	NVidia has announced a new graphic card for desktops	<a>Edit</a>
Airplane crash	A passenger plane has crashed in Europe	<a>Edit</a>
Coronavirus	A new version of virus was found!	<a>Edit</a>
Bitcoin price raises	Price of bitcoin reaches new record.	<a>Edit</a>
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.	<a>Edit</a>
test	test	<a>Edit</a>
&amp;#039; OR &amp;#039;1&amp;#039;=&amp;#039;1&amp;#039; --	amp039 OR amp0391amp039amp0391amp039	<a>Edit</a>
&amp;#039; OR &amp;#039;1&amp;#039;=&amp;#039;1&amp;#039; --	amp039 OR amp0391amp039amp0391amp039	<a>Edit</a>
&amp;#039; OR &amp;#039;1&amp;#039;=&amp;#039;1&amp;#039; #	amp039 OR amp0391amp039amp0391amp039	<a>Edit</a>
&amp;quot; OR &amp;quot;1&amp;quot;=&amp;quot;1&amp;quot; --	ampquot OR ampquot1ampquotampquot1ampquot	<a>Edit</a>
Test	Test	<a>Edit</a>

Navigation

[index](#)  
[messages](#)  
[add new message](#)