CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation

```
1. Serial.print(F("Password: "));
2.   int kunci,cipher;
3.   char cipher2;
4.   kunci=3;
5.   cipher=0;
6.   char temp[20];
7. String pass0, pass1, pass2, pass3, pass4, pass5, pass6, pass7,
8. pass8, pass9, pass10, pass11, pass12, pass13, pass14, pass15,
9. pass16;
10.  for (uint8_t i = 0; i < 16; i++) {
11.    if(buffer2[i] != 32){
12.     pass0 = char(buffer2[0]);
13.     pass0.toCharArray(temp,20);
14.     cipher = ((temp[0] + kunci - 'a') % 26) + 'a';
15.     cipher = char(cipher);

16.     cipher2=cipher;
17.     pass0=cipher2;

18.     pass1 = char(buffer2[1]);
19.     pass1.toCharArray(temp,20);
20.     cipher = ((temp[0] + kunci - 'a') % 26) + 'a';
21.     cipher = char(cipher);
22.     cipher2=cipher;
23.     pass1=cipher2;

24.     pass2 = char(buffer2[2]);
25.     pass2.toCharArray(temp,20);
26.     cipher = ((temp[0] + kunci - 'a') % 26) + 'a';
```

```
1. postData = "uid=" + pass + "&username=" + user0 + user1 + user2
2.  + user3 + user4 + user5 + user6 + user7 + user8 + user9 +
      user10 + user11 + user12 + user13 + user14 + user15 +
      user16;
3.     http.begin("http://192.168.43.206/sufendi/bisa1.2.php");
```

Line 1 to get all the data that ready by the RFID into one variable called postData. Line 3 is to request destination where data will be used.

```
1. <?php
2.     $servername = "localhost";
3.    $username = "root";
4.    $password = "";
5.     $dbname = "dbfinal";
6.
7.     // Create connection
8.        $conn = new mysqli($servername, $username, $password,
$dbname);
9.     // Check connection
10.    if ($conn->connect_error) {
11.               die("Database Connection failed: " . $conn-
>connect_error);
12.     }
13.    if(!empty($_POST['uid']))
14.     {
15.         $uid = $_POST['uid'];
16.         $uid = rtrim($uid);
17.         $ukuran = strlen($uid);
18.         $arr1 = str_split($uid);
```

```
19.         for($i = 0; $i<$ukuran; $i++){
20.                     $temp[$i] = ord($arr1[$i])-3;
21.                     $karakter[$i] =chr($temp[$i]);
22.                     echo $karakter[$i].",";
23.         }
24.
25.         $gabung = implode("",$karakter);
26.
27.         $username = $_POST['username'];
28.         $username = rtrim($username);
29.
30.         //$station = $_POST['station'];
31.
32.        $sql = "INSERT INTO tbltemp (uid, username)
33.
34.         VALUES ('".$gabung."','".$username."')";
35.
36.
37.         if ($conn->query($sql) === TRUE) {
38.             echo "OK";
39.         } else {
40.             echo "Error: " . $sql . "<br>" . $conn->error;
41.         }
42.   }
43.
44.
45.   $conn->close();
46. ?>
```

Line 2 until 5 is to declare variable that used for access the database. Line 8 is to make connection to database. Line 10 is to stop process when failed to access the database. Line 13 to check if there is data that can be posted. Line15 and 16 is to declare uid variable. Ine 16 until line 18 is to decrale variables that will be used to decrypt data. Line 19 until line 23 is to decrypt the data, it will first

ready the length of data and then will split it to char and input each char to array then it will decrypt it.Line 25 is variables to combine all char that already decrypted. Line 27 to declare variable username. Line 28 to remove space at the end of the data.

```
1. <?php
2. class User
3. {
4.     private $conn;
5.
6.     function __construct($cn)
7.     {
8.         $this->conn = $cn;
9.     }
10.
11.    function rfidlogin($username)
12.    {
13.        $select = $this->conn->prepare("SELECT * FROM tbltemp
WHERE username = '$username'");
14.        $select->execute();
15.        $data = $select->fetchAll(PDO::FETCH_BOTH);
16.
17.        if(count($data) != 0)
18.        {
19.            $select = $this->conn->prepare("SELECT uid FROM
tbltemp WHERE username = '$username'");
20.            $select->execute();
21.            $data1 = $select->fetchAll(PDO::FETCH_BOTH);
22.
23.            $select = $this->conn->prepare("SELECT uid FROM
tblrfid WHERE username = '$username'");
24.            $select->execute();
25.            $data2 = $select->fetchAll(PDO::FETCH_BOTH);
26.
27.            echo $data1[0][0];
```

```
28.                 echo $data2[0][0];


29.                 if($data1[0][0] == $data2[0][0])
30.                 {
31.                       $delete   =   $this->conn->prepare("DELETE
FROM tbltemp WHERE username = '$username'");
32.                       $delete->execute();
33.
34.                       $_SESSION['rfid'] = "ADA";
35.                       header("Location: indexuser.php");
36                  }
37.                 if($data1[0][0] != $data2[0][0])
38.                 {
39.                       $delete   =   $this->conn->prepare("DELETE
FROM tbltemp WHERE username = '$username'");
40.                       $delete->execute();
41.                       $_SESSION['report'] = 'RFID Keliru!!!';
42.                       header("Location: user.php");
43.                 }
44.           }
45.           else
46.           {
47.                 $_SESSION['report'] = 'Anda Belum Scan RFID!!!';
48.                 header("Location: user.php");
49.           }
50.     }
51.}
52.?>
```

Line 2 to declare class name. Line 11 to declare function called rfidlogin. Line 13 until 15 to get data in database where it match each other and store it in variable data. Line 17 until 25 if there is data it will continue the process to input uid in tbltemp where it has same username, and it also select uid in tblrfid where it also has same username. Line 28 is to compare whether data in tbltemp and tbl rfid is match. Line 31 is to delete the data in tbltemp after the comparison. Line 34 and 35 is to direct user to main page if the data is correct. Line 37 until 42 executed if the data is not matched and it will go back to user.php (line 42). Line 47 and 48 is when user push next button but not yet scan the card.

5.2 Testing

LOGIN TESTING

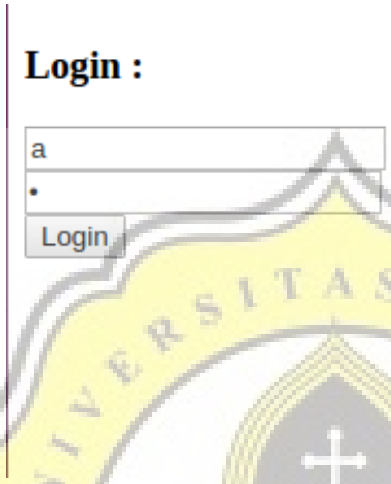


Illustration 5.1: Login Page

This is the login page that user need to fill. The user need to fill correct id and password and it will directed to card verification page after click submit button.

Illustration 5.2: Card Verification Page

This is the card verification page. It contains what user id that need to be verified. User need to scan the card to RFID. The data that read by the RFID will be sent to database.

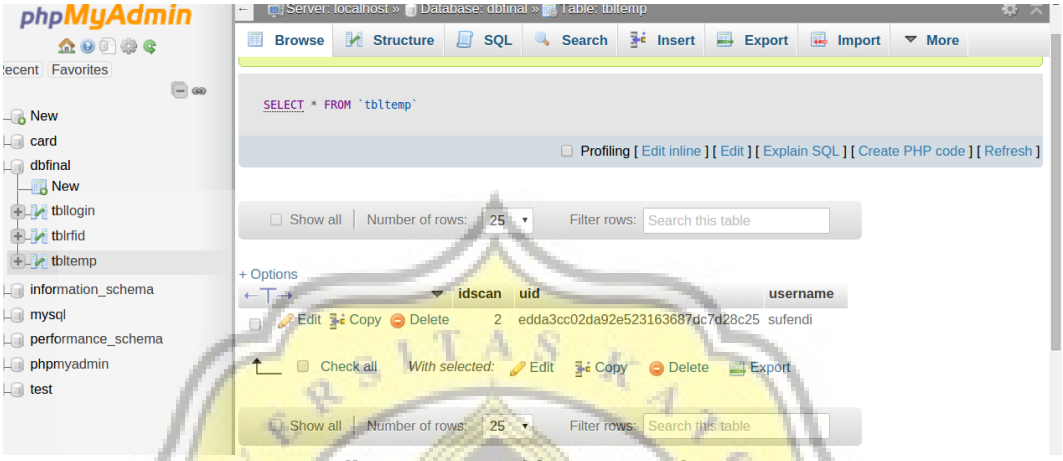


Illustration 5.3: Tbltemp in Database

Tha data that RFID read will be saved temporarily in table tbltemp and will be encrypted using MD5 method. After the user scan the card, it just need to click the next button and it will directed to the next page if thE card match with the user id.

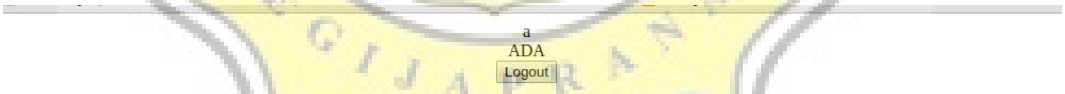


Illustration 5.4: Main Page

After the user success to login it will directed to the main page of the website.

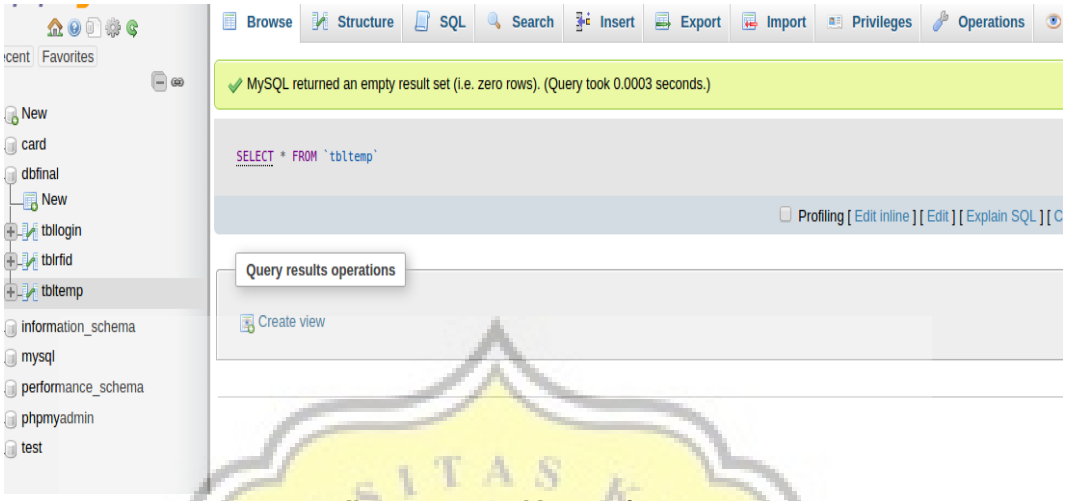


Illustration 5.5: Tbltemp After Login

The data in tbltemp will be deleted whether the user success or not to login to the main page. It is to prevent the data of the card being hacked.

INJECTION TESTING

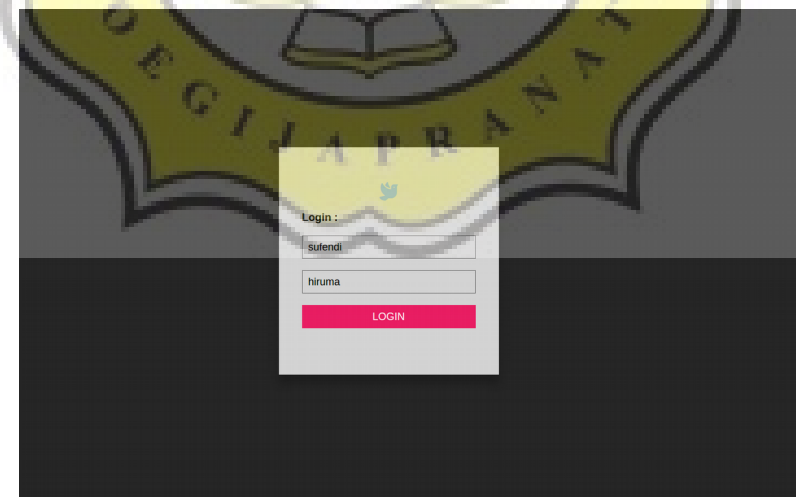
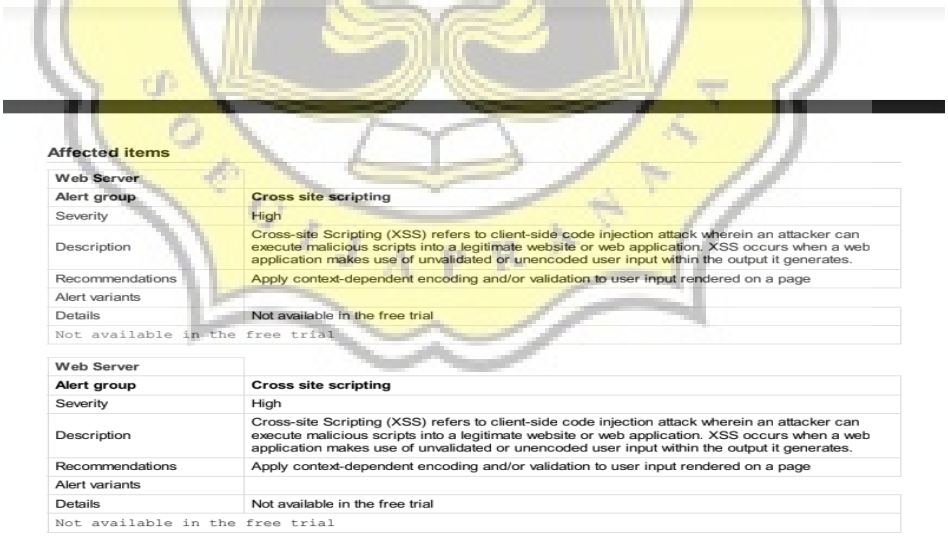


Illustration 5.6: Injection Form

Illustration 5.7: Injection Input

This test will try to hijacking the card verification page. As shown above it cannot to be hijacked because if hacker input the card's passwrod it will be encrypted and the data that store in database will not match for login usage.

ACUNETIX TESTING



Affected items

Web Server	
Alert group	Cross site scripting
Severity	High
Description	Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.
Recommendations	Apply context-dependent encoding and/or validation to user input rendered on a page
Alert variants	
Details	Not available in the free trial
Not available in the free trial	

Web Server	
Alert group	Cross site scripting
Severity	High
Description	Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.
Recommendations	Apply context-dependent encoding and/or validation to user input rendered on a page
Alert variants	
Details	Not available in the free trial
Not available in the free trial	

Illustration 5.8: Acunetix Result

There are 2 high risk after do analysis with acunetix.This system quite vulnerable with cross site scripting. Cross site scripting is one of code injection

attack. This is carried out by the attacker using HTML code or client script code. So, the attacker can bypass security system on client side.