

**Title: Insecure Direct Object Reference (IDOR) vulnerability in a Real Estate Listing Platform.**

**Name :** Keyur Patel ,  
Krishiv Patel

**Roll No:** 16010421073, 16010421074

**Batch:** A2

**Aim:** Explain and fix an Insecure Direct Object Reference (IDOR) vulnerability in a Real Estate Listing Platform, specifying the relevant CWE: Insecure Direct Object Reference (CWE-639) and (CWE-353) Missing Support for Integrity Check..

**Resources needed:** Real Estate website , Burpsuite , Xampp

**Pre Lab/ Prior Concepts:**

Students should have prior knowledge of Burpsuite,SQL Injection,Input Validation,Access control mechanism,Data Encryption.

**Theory:**

**1. Insecure Direct Object Reference:-**

Insecure Direct Object Reference refers to a vulnerability that occurs when a web application exposes a reference to an internal implementation object, such as a file, directory, database record, or key, in a way that allows unauthorized access to that object.

**2. Missing Support For Integrity Check:-**

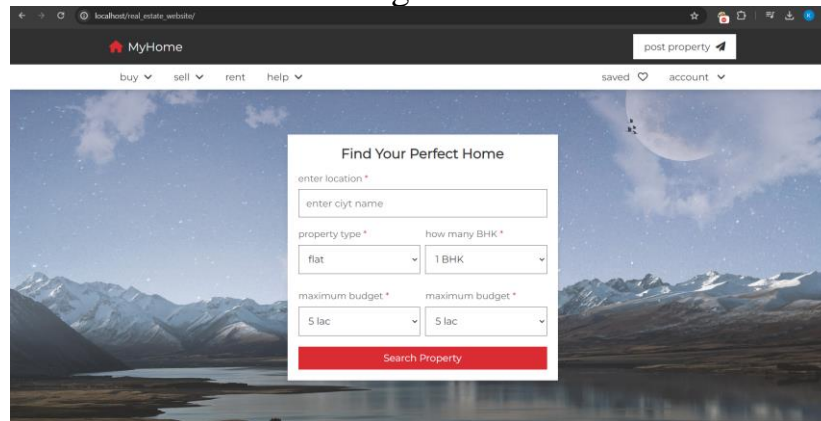
If integrity check values or "checksums" are omitted from a protocol, there is no way of determining if data has been corrupted in transmission. The lack of checksum functionality in a protocol removes the first application-level check of data that can be used. The end-to-end philosophy of checks states that integrity checks should be performed at the lowest level that they can be completely implemented. Excluding further sanity checks and input validation performed by applications, the protocol's checksum is the most important level of checksum, since it can be performed more completely than at any previous level and takes into account entire messages, as opposed to single packets.

### 3. Burpsuite:-

Burp Suite is a powerful cybersecurity tool used primarily for web application security testing and analysis. It's developed by PortSwigger, a UK-based cybersecurity company. Burp Suite is widely recognized and extensively used by security professionals, including penetration testers, security researchers, and web developers, to identify and address vulnerabilities in web applications.

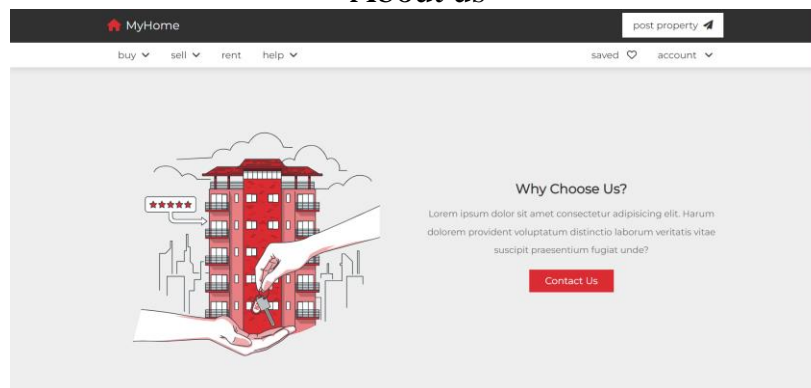
#### Procedure & Output(Screenshot):

##### Real Estate Listing Platform Home Page of website

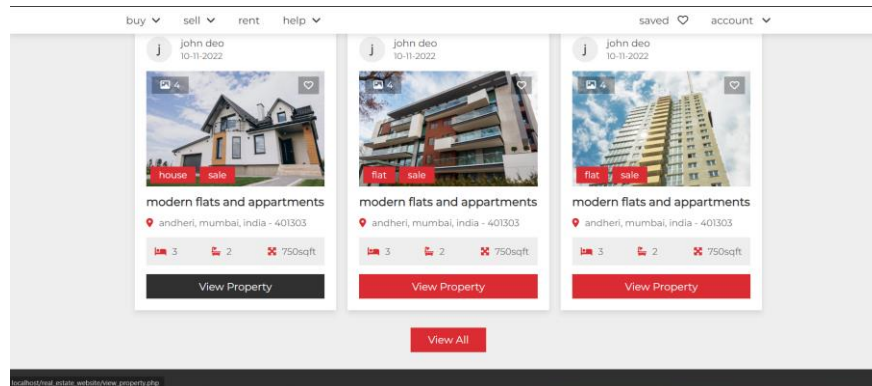


The screenshot shows the home page of a website called 'MyHome'. The browser address bar shows 'localhost/real\_estate\_website/'. The website has a dark header with the 'MyHome' logo and a 'post property' button. Below the header is a navigation bar with links for 'buy', 'sell', 'rent', and 'help', and user links for 'saved', 'account', and 'help'. The main content area features a large background image of a mountain landscape with a lake. Overlaid on this is a white search form titled 'Find Your Perfect Home'. The form contains the following fields: 'enter location \*' (text input), 'enter city name' (text input), 'property type \*' (dropdown menu with 'flat' selected), 'how many BHK \*' (dropdown menu with '1 BHK' selected), 'maximum budget \*' (dropdown menu with '5 lac' selected), and another 'maximum budget \*' (dropdown menu with '5 lac' selected). A red 'Search Property' button is at the bottom of the form.

##### About us



##### View Property of website



## Create Account

## Contact Us

## Vulnerability - 1 (CWE – 639 Insecure Direct Object Reference)

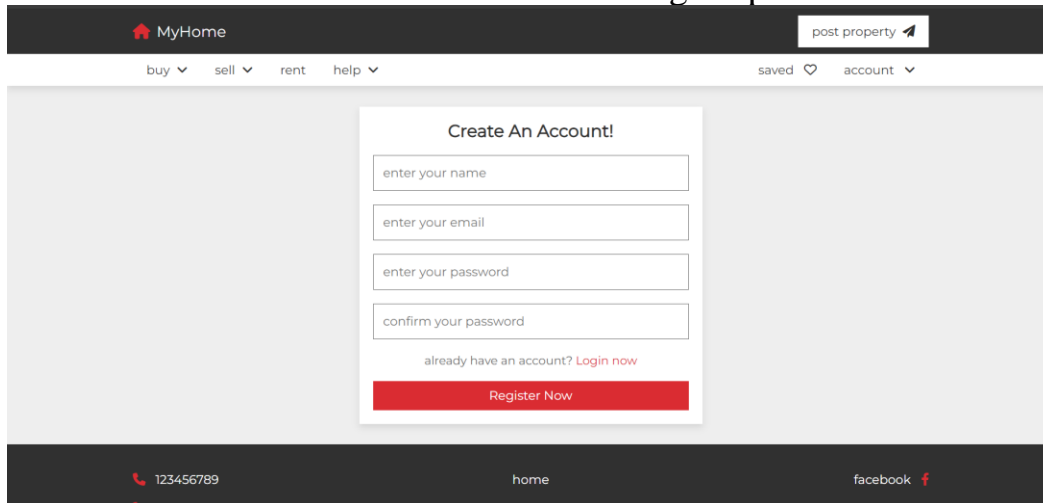
### Insecure Direct Object Reference:-

Insecure Direct Object Reference refers to a vulnerability that occurs when a web application exposes a reference to an internal implementation object, such as a file,

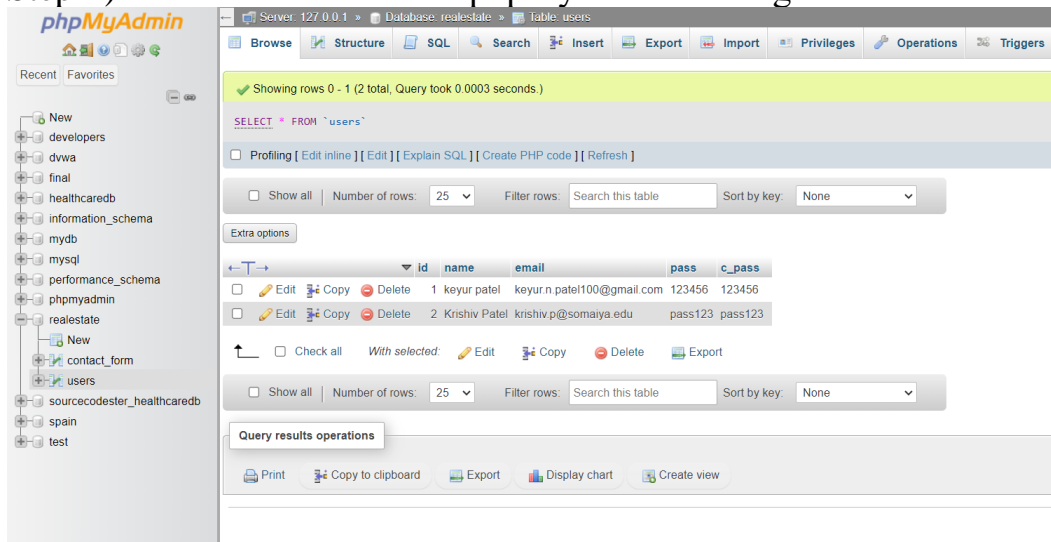
directory, database record, or key, in a way that allows unauthorized access to that object.

### Procedure to Expose Vulnerability:

Step-1) This is the Create Account page where we are going to exposing the modification of user credential using burpsuite.



Step-2) Data stored for user in phpMyAdmin having database as 'realestate'.



	id	name	email	pass	c_pass
<input type="checkbox"/>	1	keyur patel	keyur.n.patel100@gmail.com	123456	123456
<input type="checkbox"/>	2	Krishiv Patel	krishiv.p@somaiya.edu	pass123	pass123

Step-3) Form code for account.

```
<form action="<?php echo $_SERVER['PHP_SELF'];?>" method="POST">
<h3>create an account!</h3>
<input type="text" name="name" required maxlength="50" placeholder="enter your name" class="box">
<input type="email" name="email" required maxlength="50" placeholder="enter your email" class="box">
<input type="password" name="pass" required maxlength="20" placeholder="enter your password" class="box">
<input type="password" name="c_pass" required maxlength="20" placeholder="confirm your password" class="box">
<p>already have an account? <a href="login.php">login now</a></p>
<input type="submit" value="register now" name="submit" class="btn">
</form>
```

Step-4) Form is submitted using post method in users table.

```

$conn = mysqli_connect('localhost','root','','realestate') or die('connection failed');

// Check connection
if ($conn->connect_error) {
    die("connection failed: " . $conn->connect_error);
}
?>

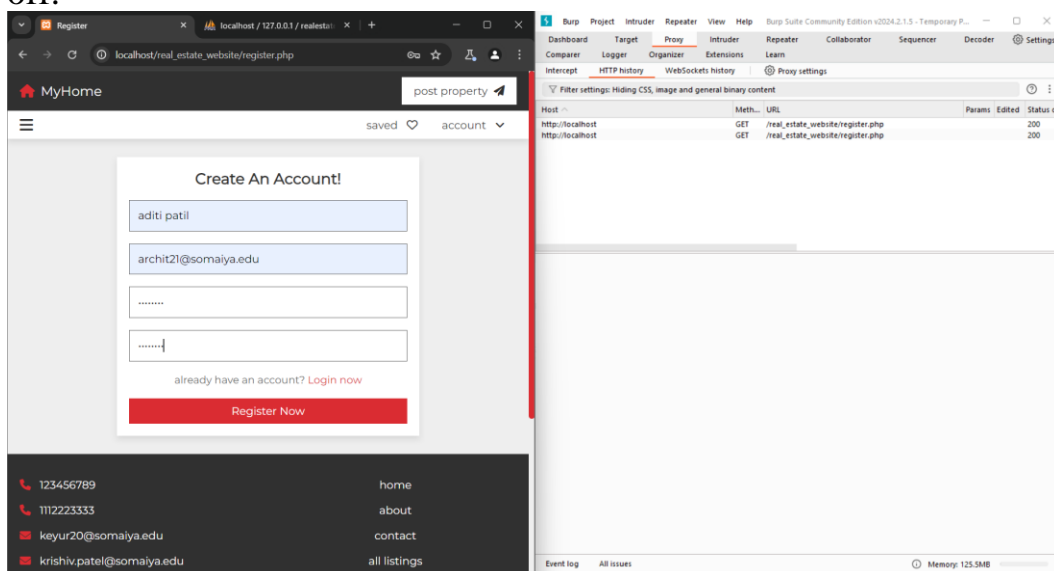
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST["name"];
    $email = $_POST["email"];
    $pass = $_POST["pass"];
    $c_pass = $_POST["c_pass"];

    // Check if passwords match
    if ($pass != $c_pass) {
        echo "Passwords do not match";
    } else {
        // Insert data into database
        $sql = "INSERT INTO users (name, email, pass,c_pass) VALUES ('$name', '$email', '$pass', '$c_pass')";

        if ($conn->query($sql) === TRUE) {
            echo "New record created successfully";
        } else {
            echo "Error: " . $sql . "<br>" . $conn->error;
        }
    }
}
?>

```

Step-5) Entering the credentials in burpsuite browser when intercept is switched off.



Step-6) Switch on the intercept in proxy tab ,after then click on 'Register now' button you will see that request is intercepted by burpsuite.



action. This can be achieved by checking the user's session information against the object's ownership or permissions.

- ***Verify User Permissions:*** Every time an access attempt is made, verify the user's permission. This can be implemented structurally using the recommended approach for your web framework. For example, when looking up objects based on primary keys, ensure that the datasets users have access to are used.
- ***Use Prepared Statements for Database Queries:*** To prevent SQL Injection, which is a related vulnerability, use prepared statements for database queries. This ensures that user input is treated as data and not.

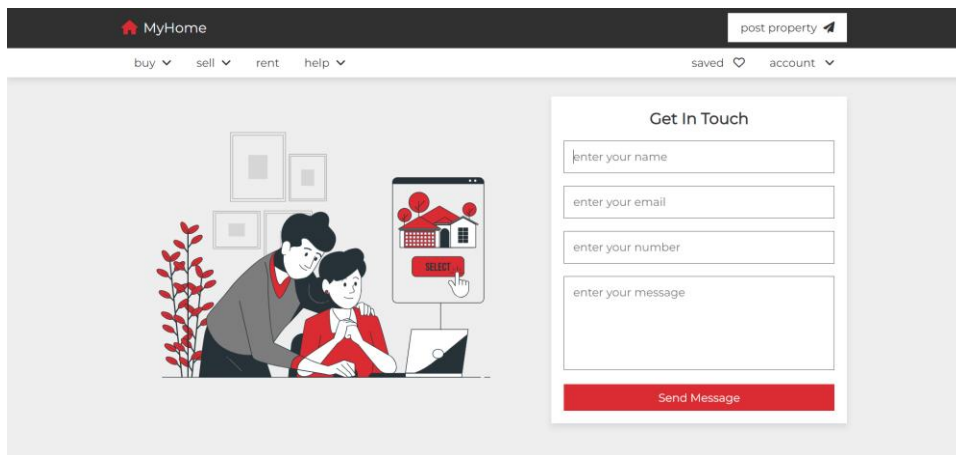
### **Vulnerability - 2 (CWE – 353 Missing Support for Integrity Check)**

If integrity check values or "checksums" are omitted from a protocol, there is no way of determining if data has been corrupted in transmission. The lack of checksum functionality in a protocol removes the first application-level check of data that can be used. The end-to-end philosophy of checks states that integrity checks should be performed at the lowest level that they can be completely implemented. Excluding further sanity checks and input validation performed by applications, the protocol's checksum is the most important level of checksum, since it can be performed more completely than at any previous level and takes into account entire messages, as opposed to single packets.

#### **Procedure to Expose Vulnerability:**

Step-1) Go to Contact us webpage for users. Here we are going to expose integrity of data being tampered by attacker without any checksum provided.





Step-2) PHPCode for submission after entering the message which is then sent to database in phpMyAdmin using post method.

```
<?php
$conn = mysqli_connect('localhost', 'root', '', 'realestate') or die('connection failed');

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

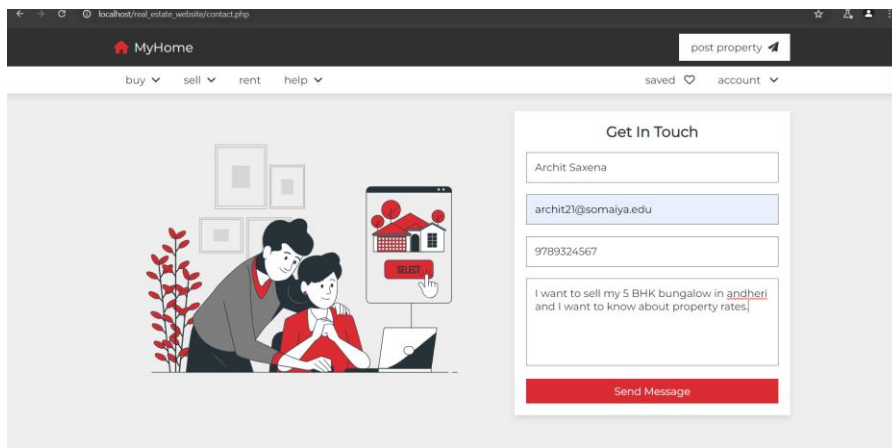
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST["name"];
    $email = $_POST["email"];
    $phone = $_POST["number"];
    $message = $_POST["message"];

    // Check if passwords match
    if (filter_var($email, FILTER_VALIDATE_EMAIL)) {
        // Email is valid
        echo "Email is valid: " . $email;
        // Here you can proceed with further processing, such as saving the email to a database or sending it via email
    } else {
        // Email is invalid
        echo "Invalid email address!";
        // You can display an error message or take other actions to handle the invalid email
    }
    $sql = "INSERT INTO contact_form (name, email, number, message) VALUES ('$name', '$email', '$phone', '$message')";

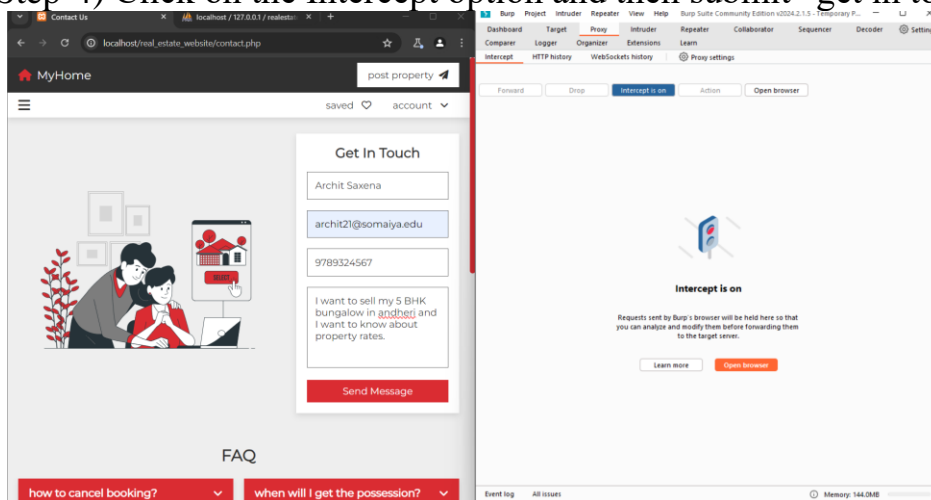
    if ($conn->query($sql) === TRUE) {
        echo "New record created successfully";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

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

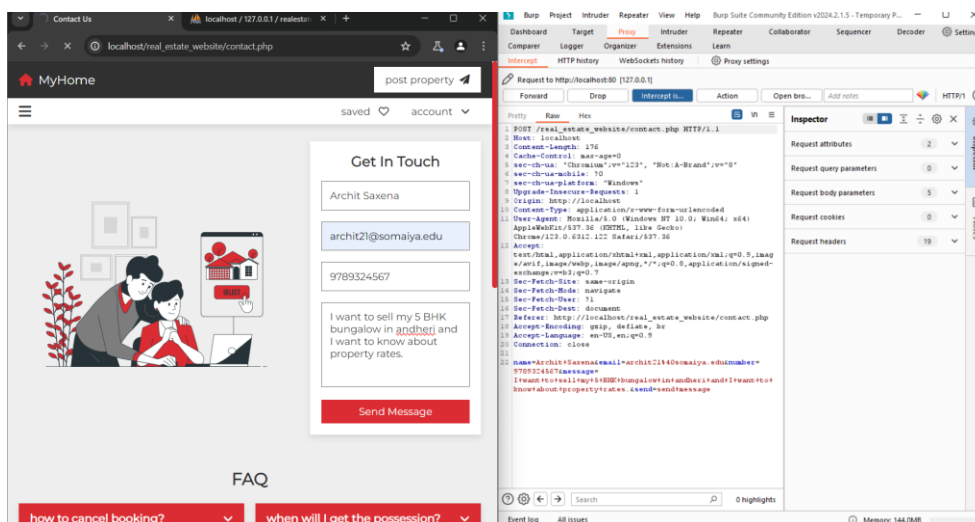
Step-3) Enter the form details when intercept is off.



Step-4) Click on the Intercept option and then submit “get in touch” form.



Step-5) The Request is then redirected to burpsuite which shows what is sent by user.



Step-4) The Attacker modifies the message and tampers the integrity of message.

```

1 |
2 | name=Archit+Saxena&email=archit21%40somaiya.edu&number=
   | 9789324567&message=modified+text.&send=send+message

```

Step-5) Then forward the modified message to database. You will see that data is modified as “*modified text*”.

✓ Showing rows 0 - 2 (3 total, Query took 0.0005 seconds.)

`SELECT * FROM `contact_form``

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Show all | Number of rows:  | Filter rows:  | Sort by key:

Extra options

		id	name	email	number	message
<input type="checkbox"/>	Edit  Copy  Delete	5	keyur patel	keyur.n.patel100@gmail.com	9769345213	I want to buy a 3BHK flat in versova.
<input type="checkbox"/>	Edit  Copy  Delete	6	keyur patel	keyur.n.patel100@gmail.com	9769345213	I want to buy a 3BHK flat in versova.
<input type="checkbox"/>	Edit  Copy  Delete	7	Archit Saxena	archit21@somaiya.edu	9789324567	modified text.

## Mitigation:

Step-1) The modified PHP code having function for fetch and verify data which includes checksum and hashes the text for ‘message’.

```

<?php
$conn = mysqli_connect('localhost', 'root', '', 'realestate') or die('connection failed');

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST["name"];
    $email = $_POST["email"];
    $phone = $_POST["number"];
    $message = $_POST["message"];

    // Generate checksum
    $data = $name . $email . $phone . $message;
    $checksum = hash('sha256', $data);

    // Prepare SQL statement
    $sql = "INSERT INTO contact_form (name, email, number, message, checksum) VALUES (?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("sssss", $name, $email, $phone, $message, $checksum);

    if ($stmt->execute()) {
        echo "New record created successfully";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

    $stmt->close();
}

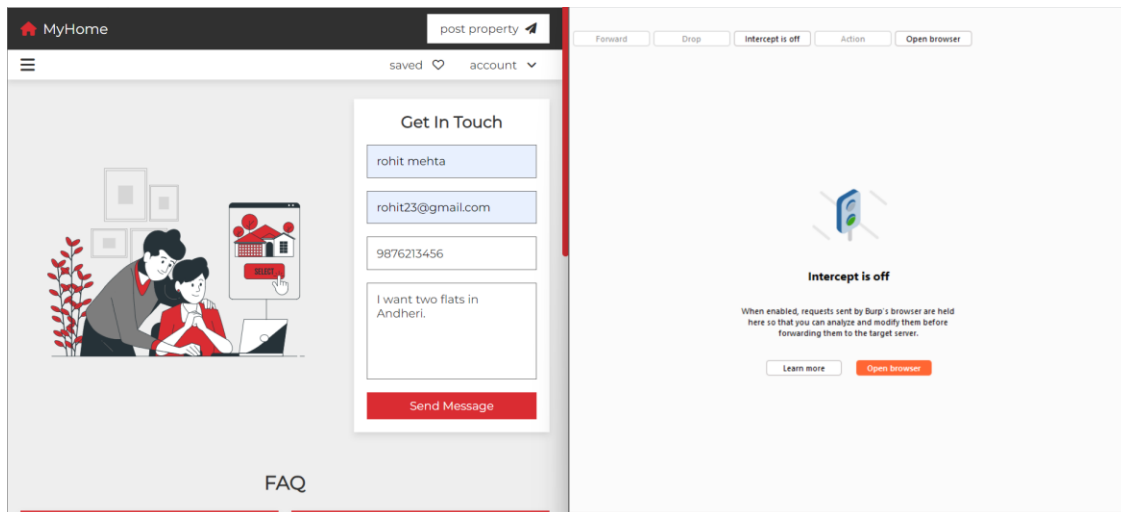
function fetchAndVerifyData($conn, $id) {
    $sql = "SELECT * FROM contact_form WHERE id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("i", $id);
    $stmt->execute();
    $result = $stmt->get_result();
    $retrievedData = $result->fetch_assoc();

    // Recalculate checksum
    $recalculatedChecksum = hash('sha256', $retrievedData['name'] . $retrievedData['email'] . $retrievedData['number'] . $retrievedData['message']);

    if ($recalculatedChecksum === $retrievedData['checksum']) {
        // Data is intact
        return $retrievedData;
    } else {
        // Data may have been tampered with
        return false;
    }
}

```

Step-2) Entering The form details when intercept is off.



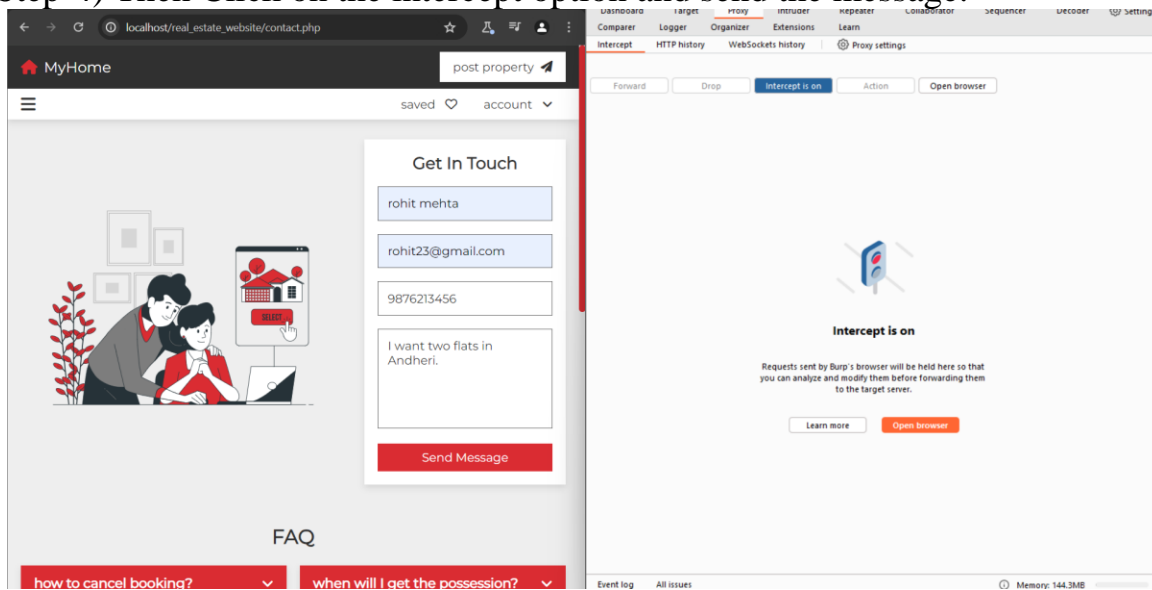
Step-3) The checksum generated for “I want two flats in Andheri” is

**Checksum:**

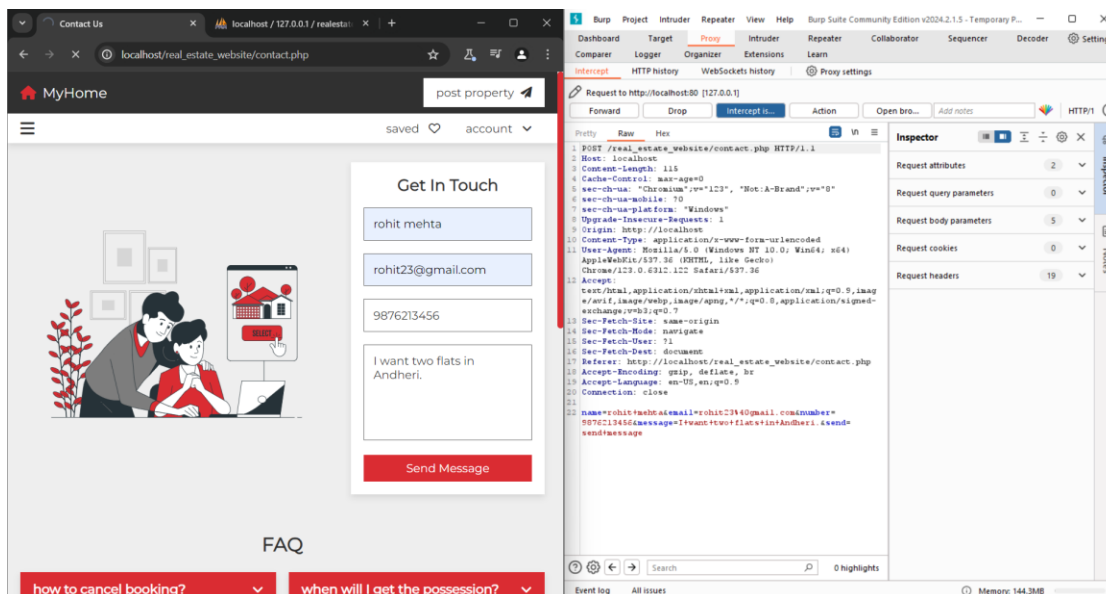
8c89d822e1797b82b011b2b60a3ea377d9c43ab360805f25413171f008bc43c8

✖ Edit Copy Delete 16 rohit mehta rohit23@gmail.com 9876213456 I want two flats in Andheri. 8c89d822e1797b82b011b2b60a3ea377d9c43ab360805f2541...

Step-4) Then Click on the intercept option and send the message.



Step-4) Then the request is redirected to burpsuite where we will modify the text “I want two flats in Andheri” to “I want four flats in Andheri”.



```
name=rohit+mehta&email=rohit23@gmail.com&number=
9876213456&message=I+want+four+flats+in+Andheri.&send=
send+message
```

Step-5) The checksum generated for “I want four flats in Andheri” is

**Checksum:**

**48767f98c35e3008cf41989e76e4e3c85a57b4ea7f339b005c61dd748fc023aa**

<input type="checkbox"/>				17	rohit mehta	rohit23@gmail.com	9876213456	I want four flats in Andheri.	48767f98c35e3008cf41989e76e4e3c85a57b4ea7f339b005c61dd748fc023aa
<input type="checkbox"/>				18	rohit mehta	rohit23@gmail.com	9876213456	I want two flats in Andheri.	8c89d822e1797b82b011b2b60a3ea377d9c43ab360805f2541...

Thus we can determine if data integrity is modified through change in hash values and compare it with original text.

**Outcomes:**

**Conclusion:**

In this IA we exposed the vulnerability in Real Estate listing platform using Burpsuite for intercepting and modifying the request and redirecting it to database. For data integrity we used checksum to see if data is tampered by attacker.