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

**Name : Keyur Patel , Roll No: 16010421073, 16010421074 Batch: A2**

**Krishiv Patel**

**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.

1. **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.

1. **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.

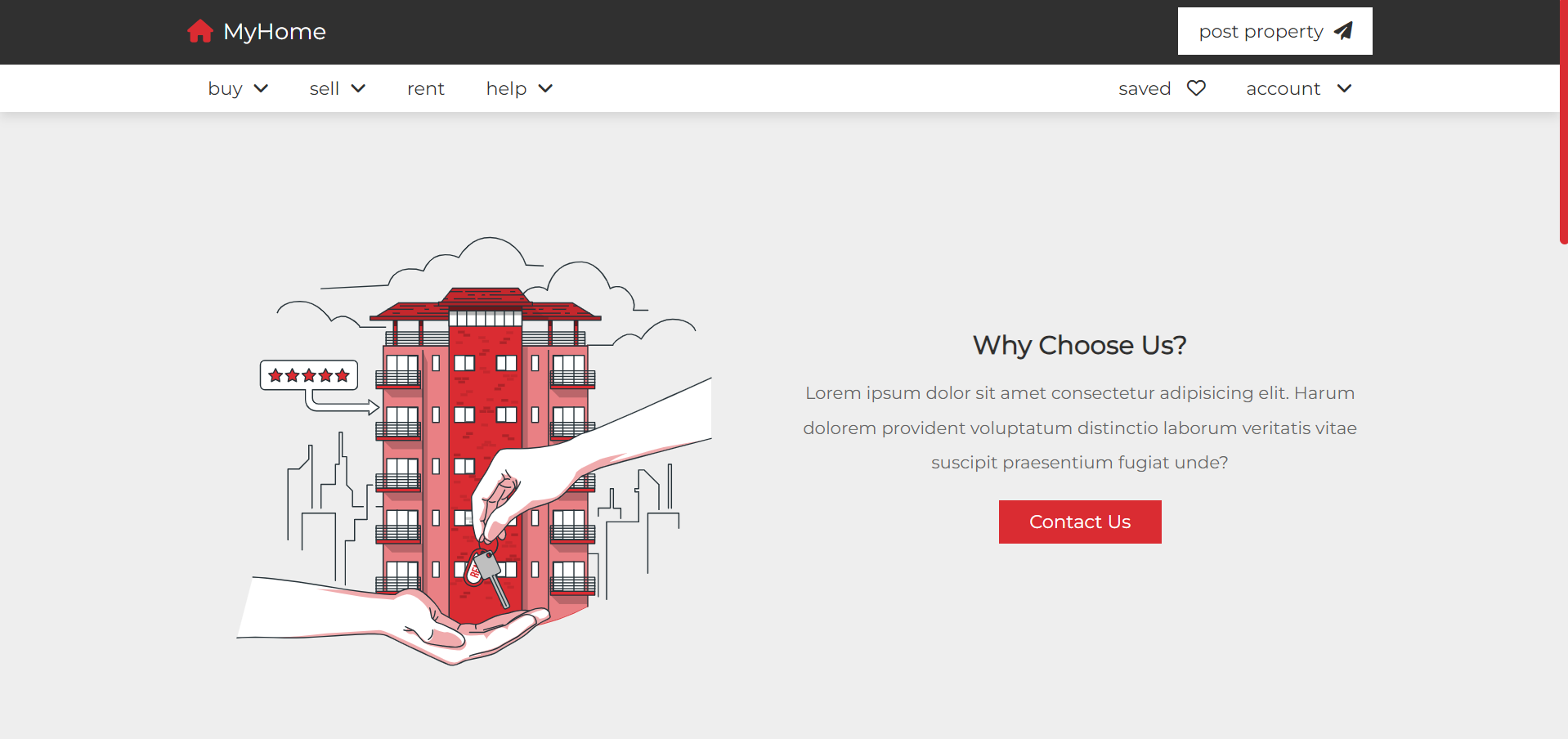
**Procedure & Output(Screenshot):**

**Real Estate Listing Platform**

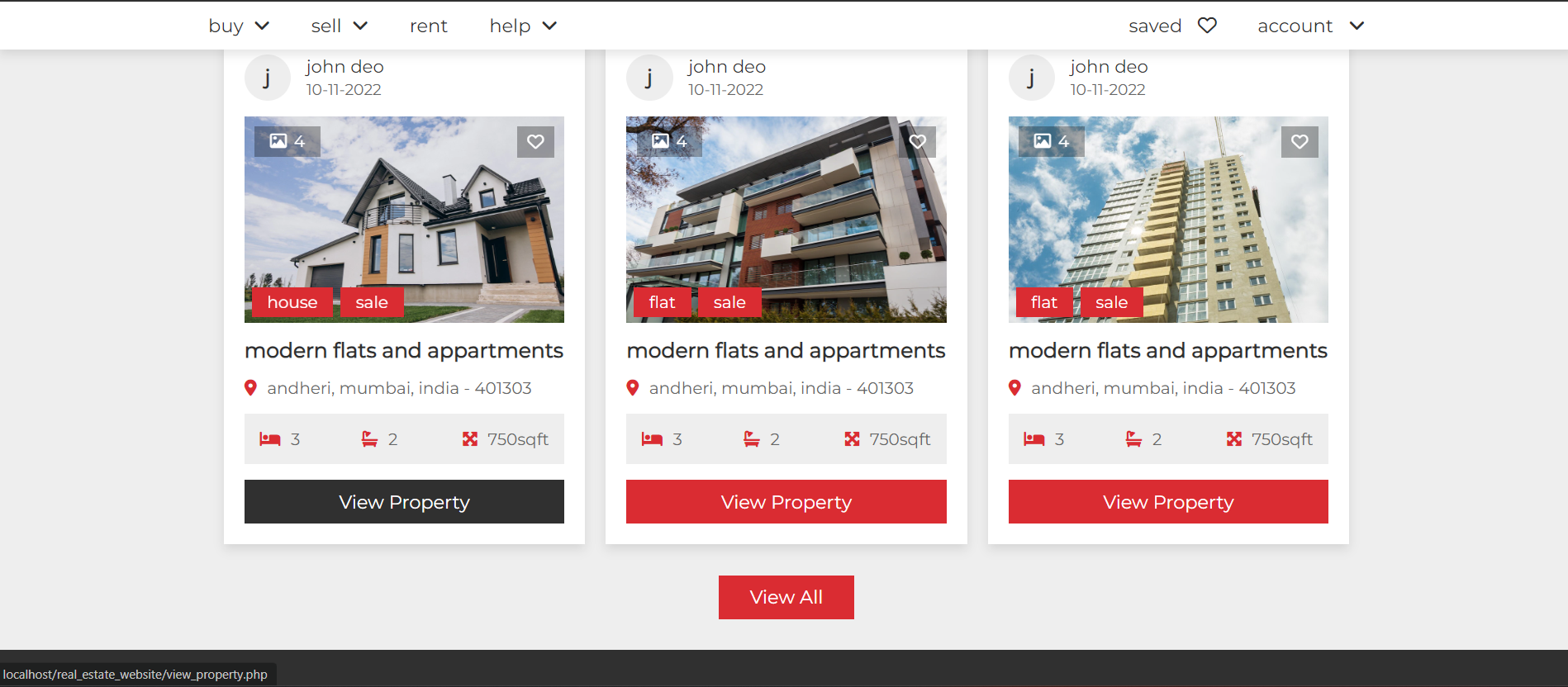
Home Page of website

****

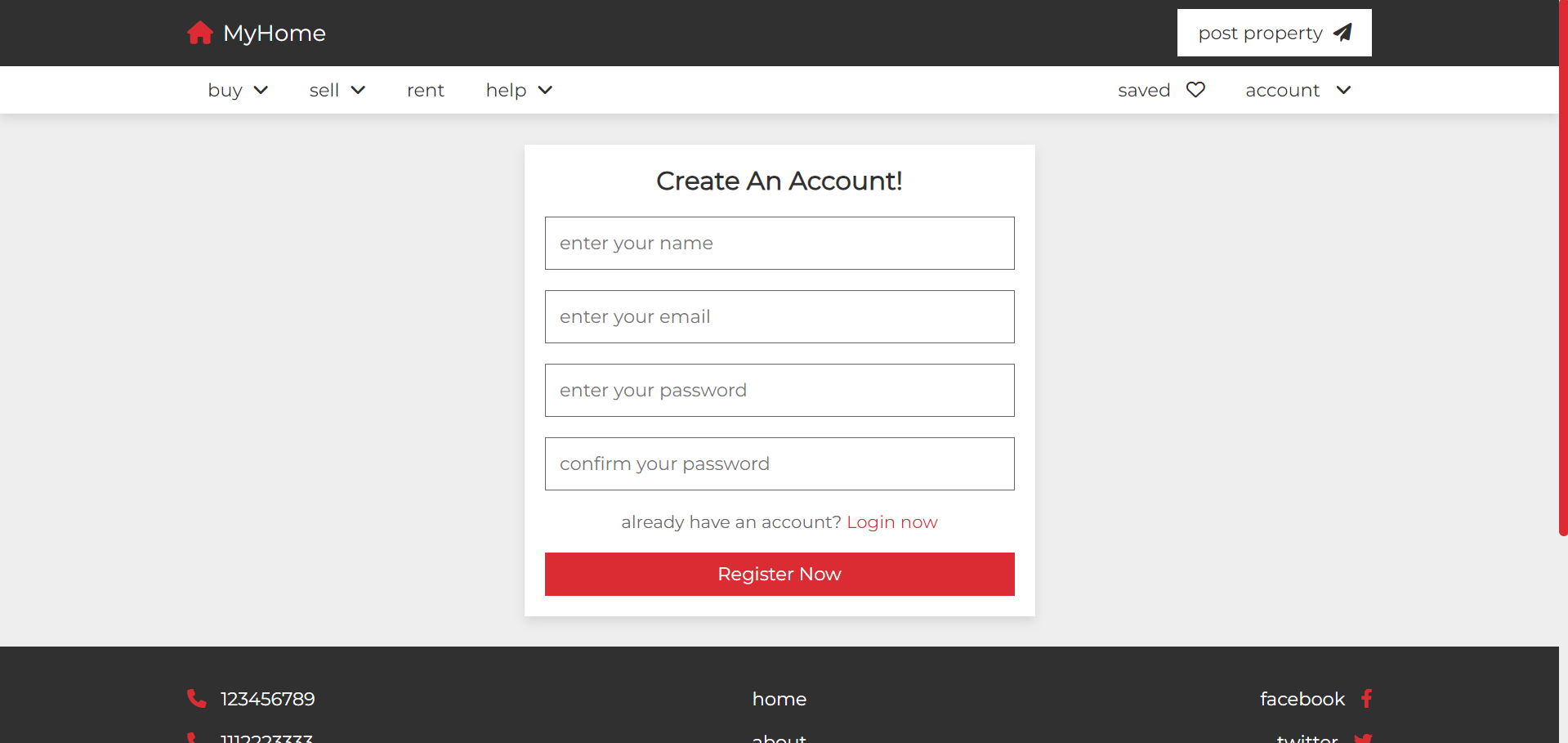
About us

****

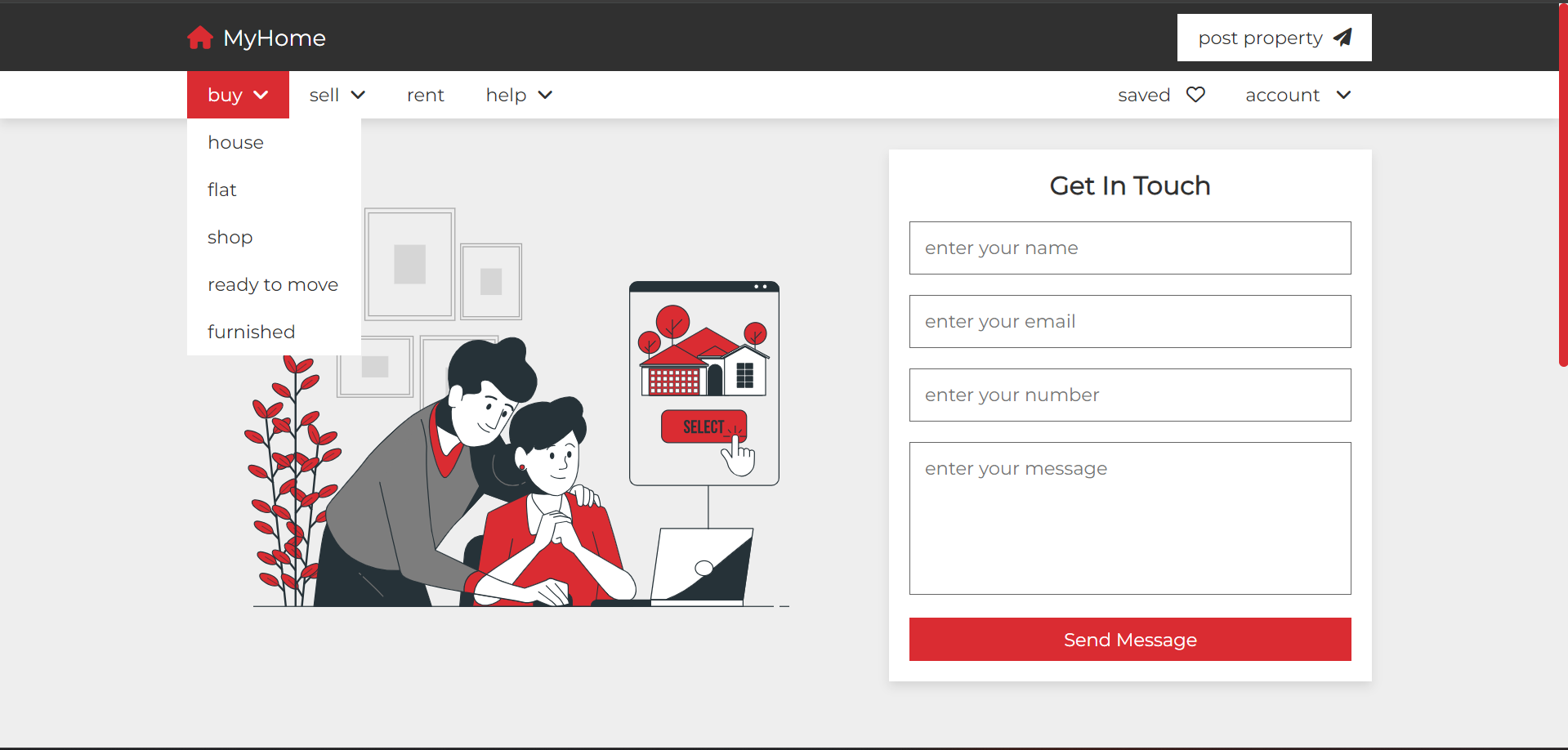
View Property of website

****

Create Account

****

**Contact Us**

****

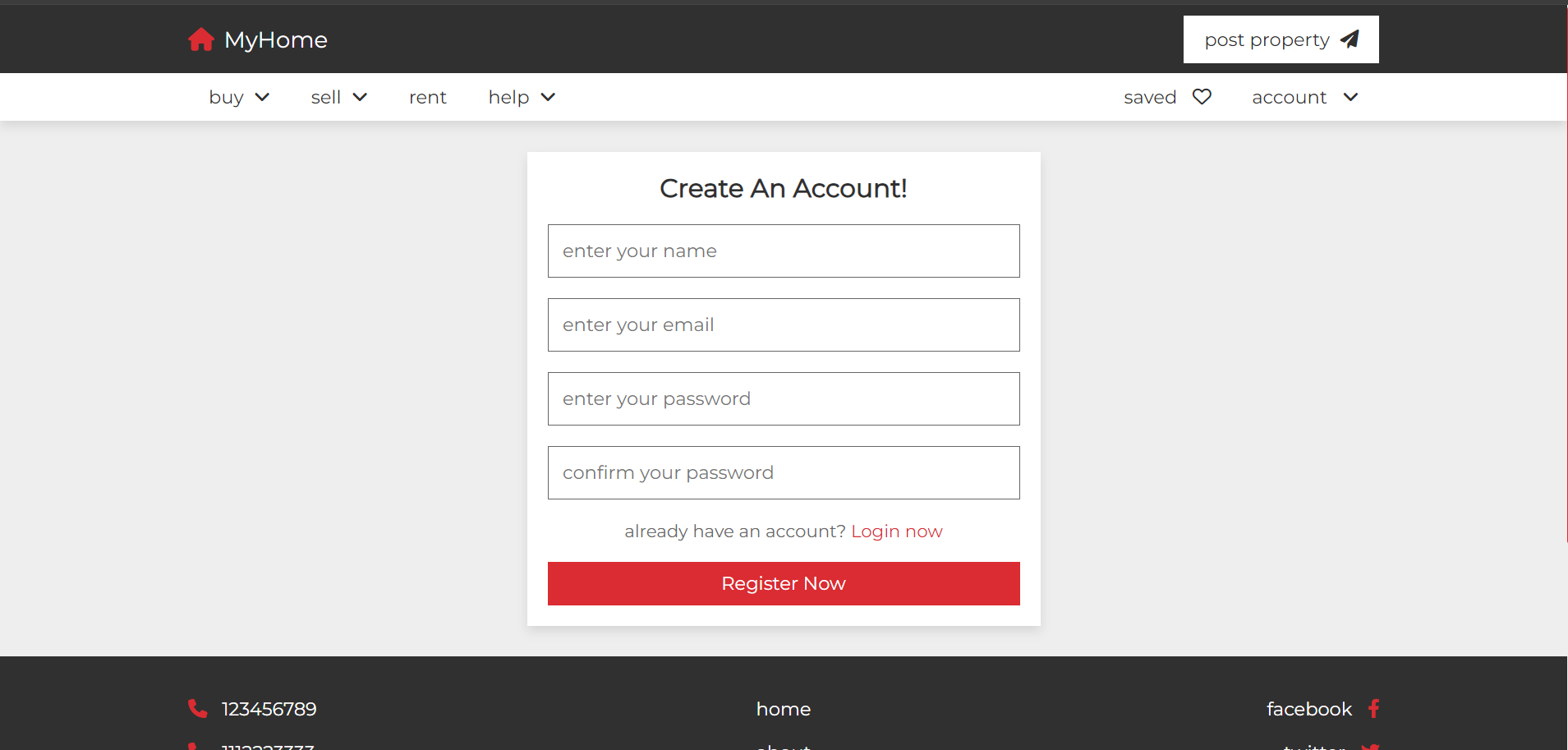
**Vulnerability - 1 (CWE – 639 Insecure Direct Object Reference)**

**Insecure Direct Object Reference:-**

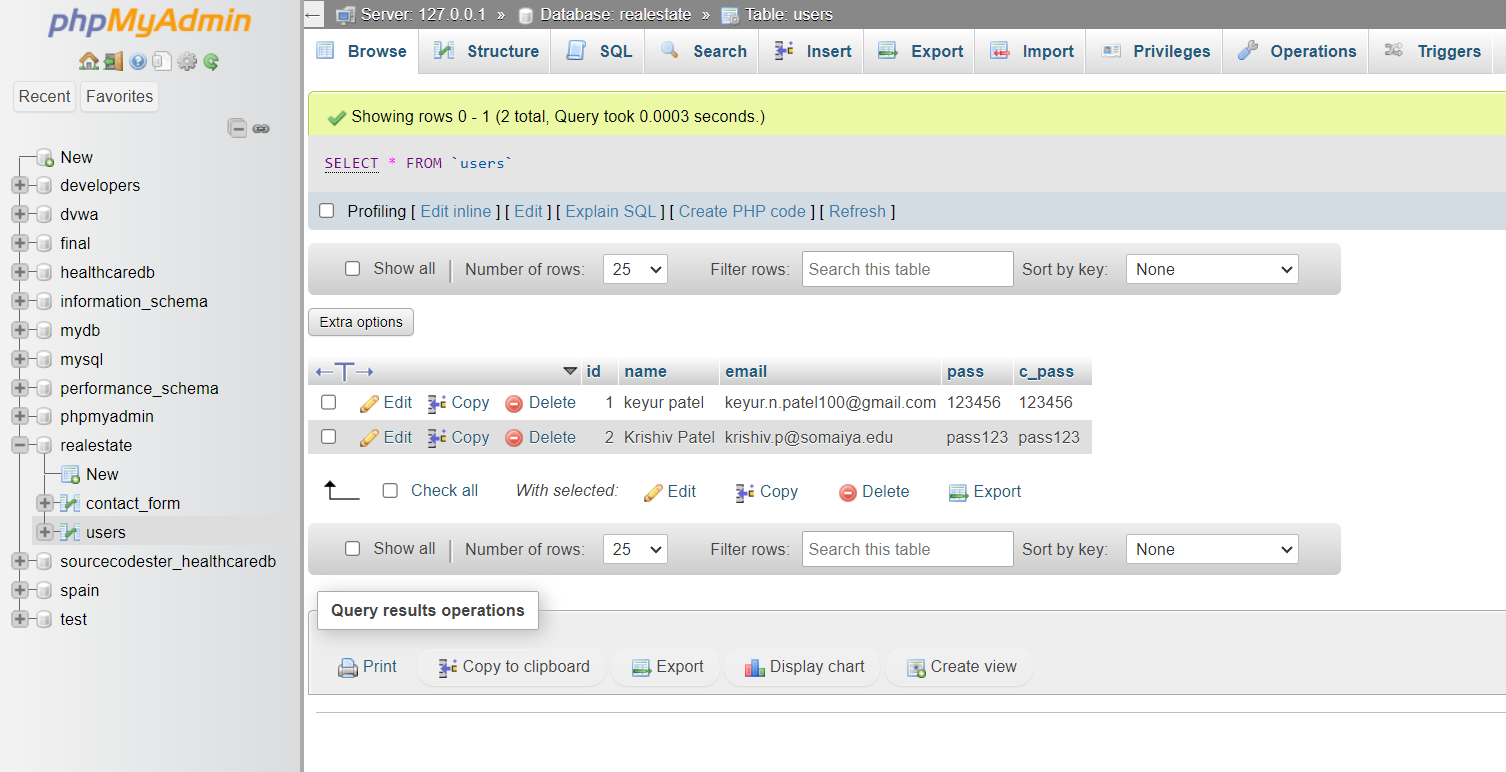
Insecure Direct Object Reference refers to a vulnerability that occurs when a web application exposes a reference to an internal implementation object, such as a file, directory, database record, or key, in a way that allows unauthorized access to that object.

**Procedure to Expose Vulnerability:**

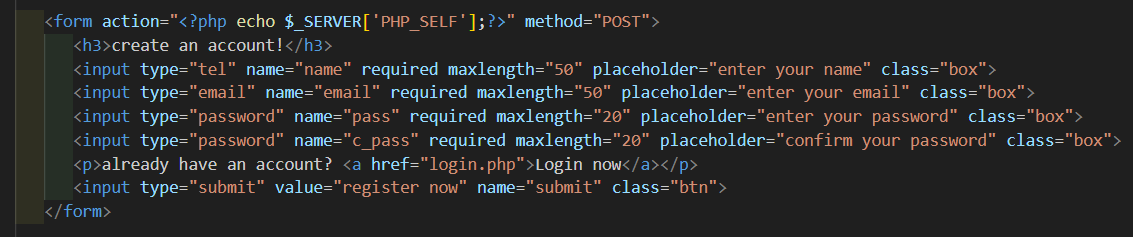
Step-1) This is the Create Account page where we are going to exposing the modification of user credential using burpsuite.

****

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

****

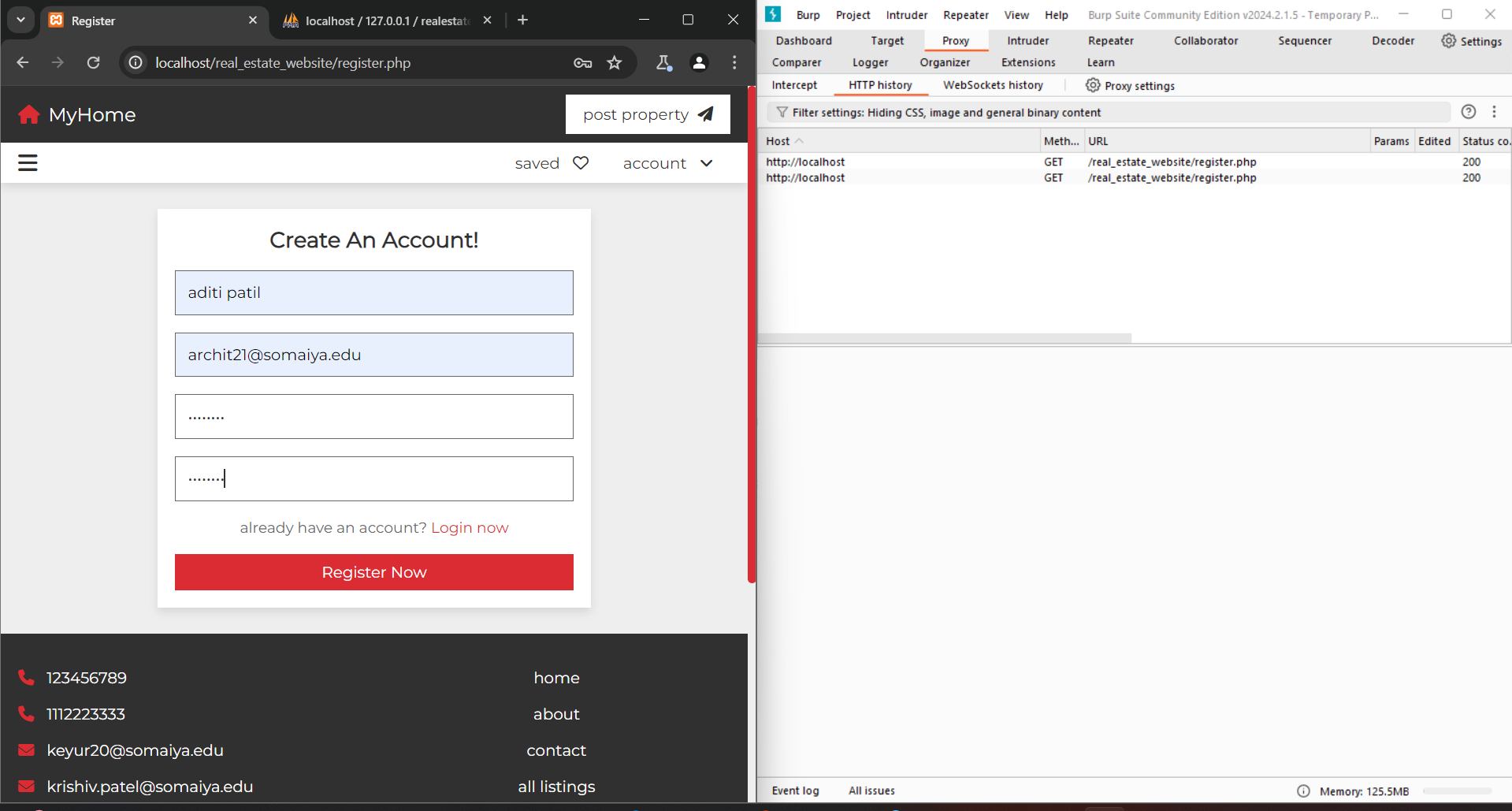
Step-3) Form code for account.

****

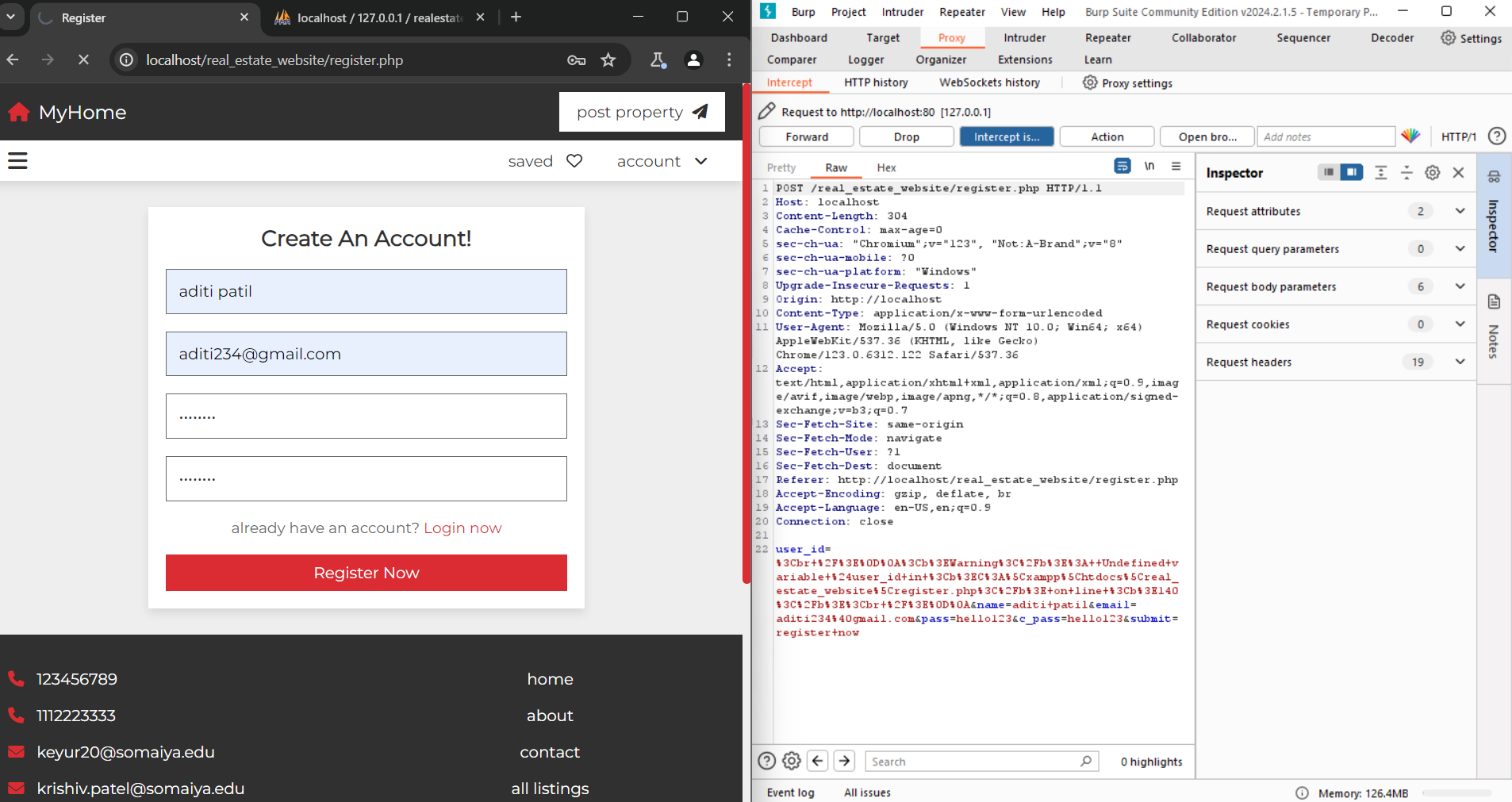
Step-4) Form is submitted using post method in users table.

****

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

****

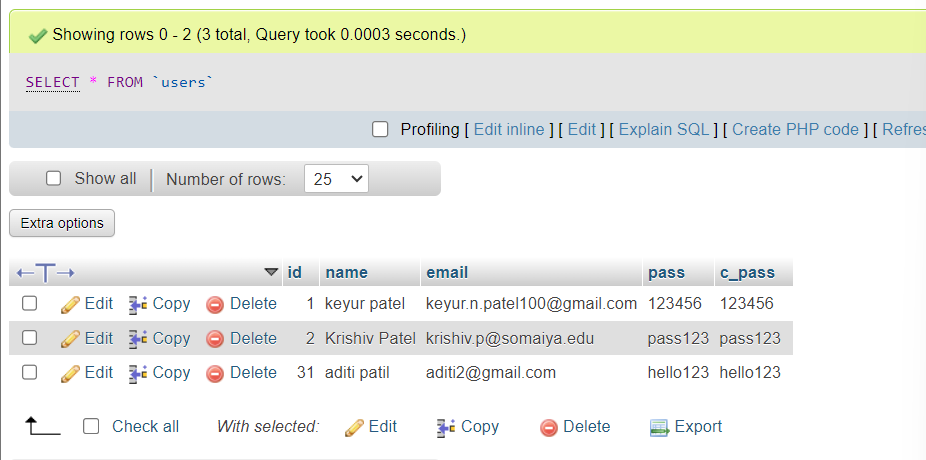
Step-6) Switch on the intercept in proxy tab ,after then click on ‘Register now’ button you will see that request is intercepted by burpsuite.

****

Step-7) Modify the email for user and click on forward option at intercept to send modified data to database.

****

Step-8) Email is modified at phpMyAdmin which was done by unauthorized person.

****

**Mitigation:**

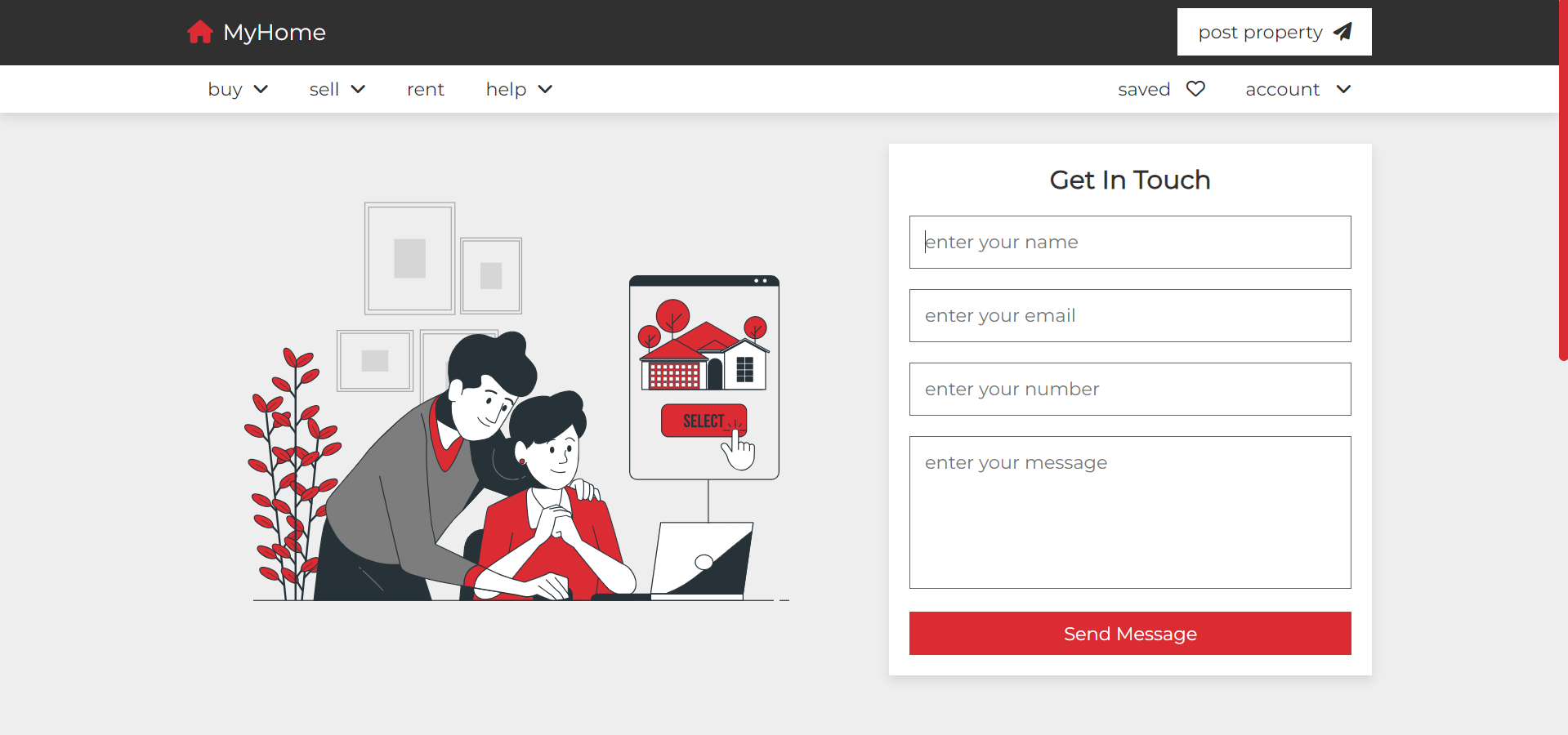
* ***Implement Access Control Checks:*** Ensure that every object a user tries to access is checked against the user's permissions. This means verifying that the user has the right to view or modify the object before allowing the action. This can be achieved by checking the user's session information 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)**

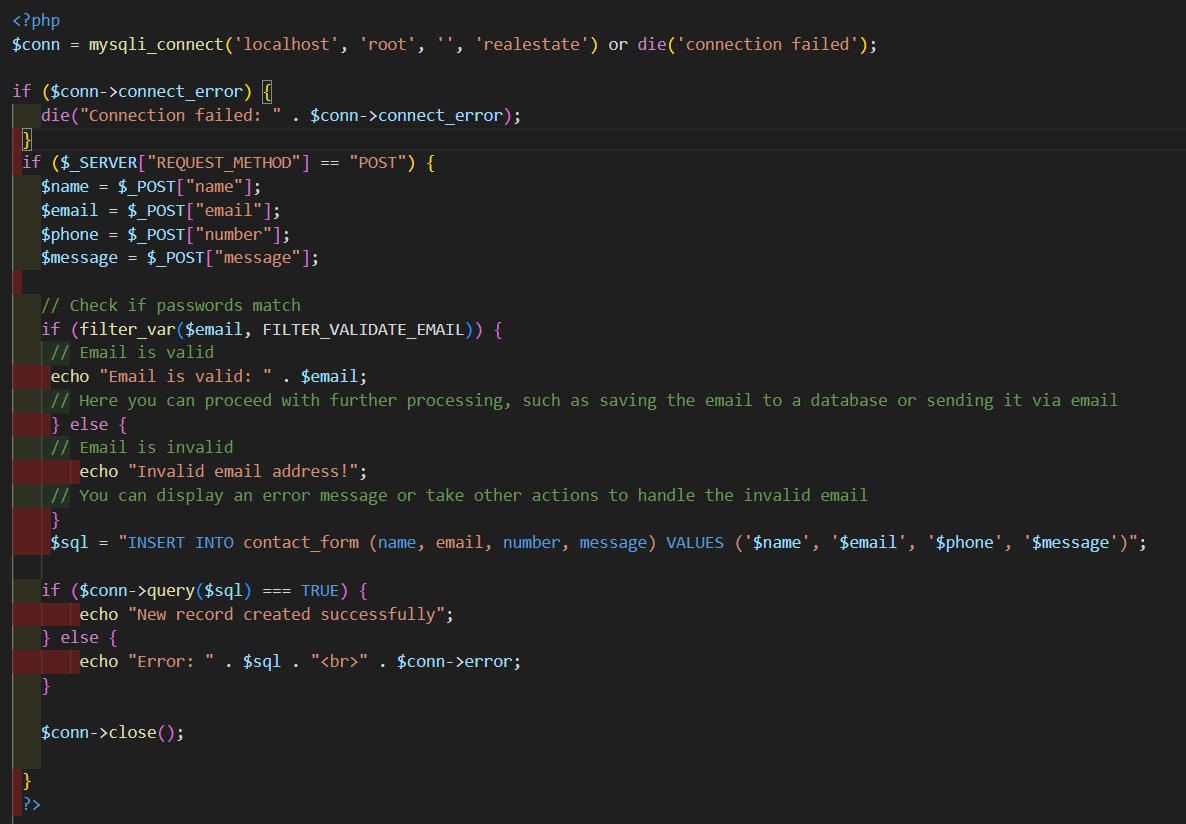
If integrity check values or "checksums" are omitted from a protocol, there is no way of determining if data has been corrupted in transmission. The lack of checksum functionality in a protocol removes the first application-level check of data that can be used. The end-to-end philosophy of checks states 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:**

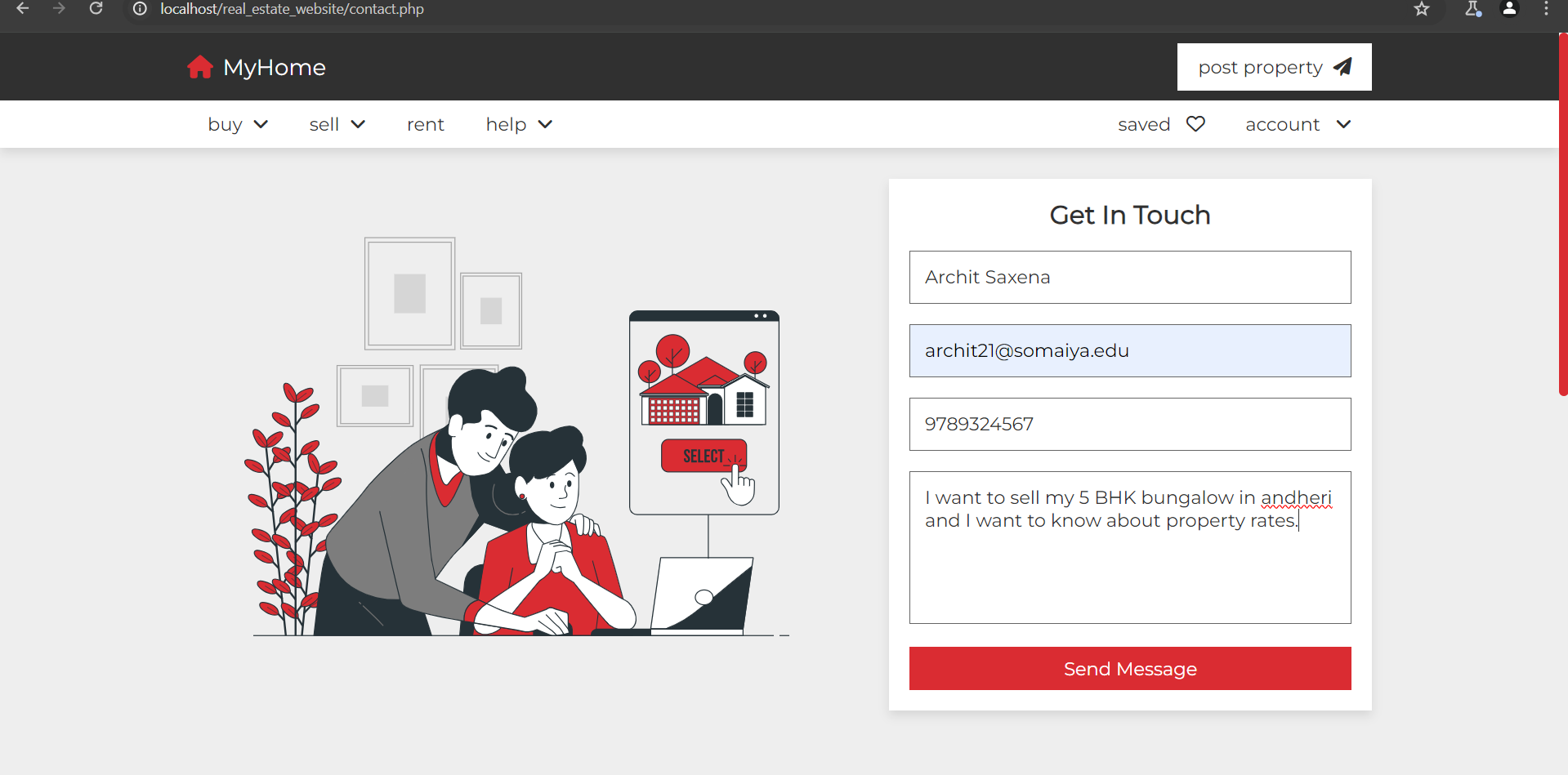
Step-1) Go to Contact us webpage for users. Here we are going to expose integrity of data being tampered by attacker without any checksum provided.



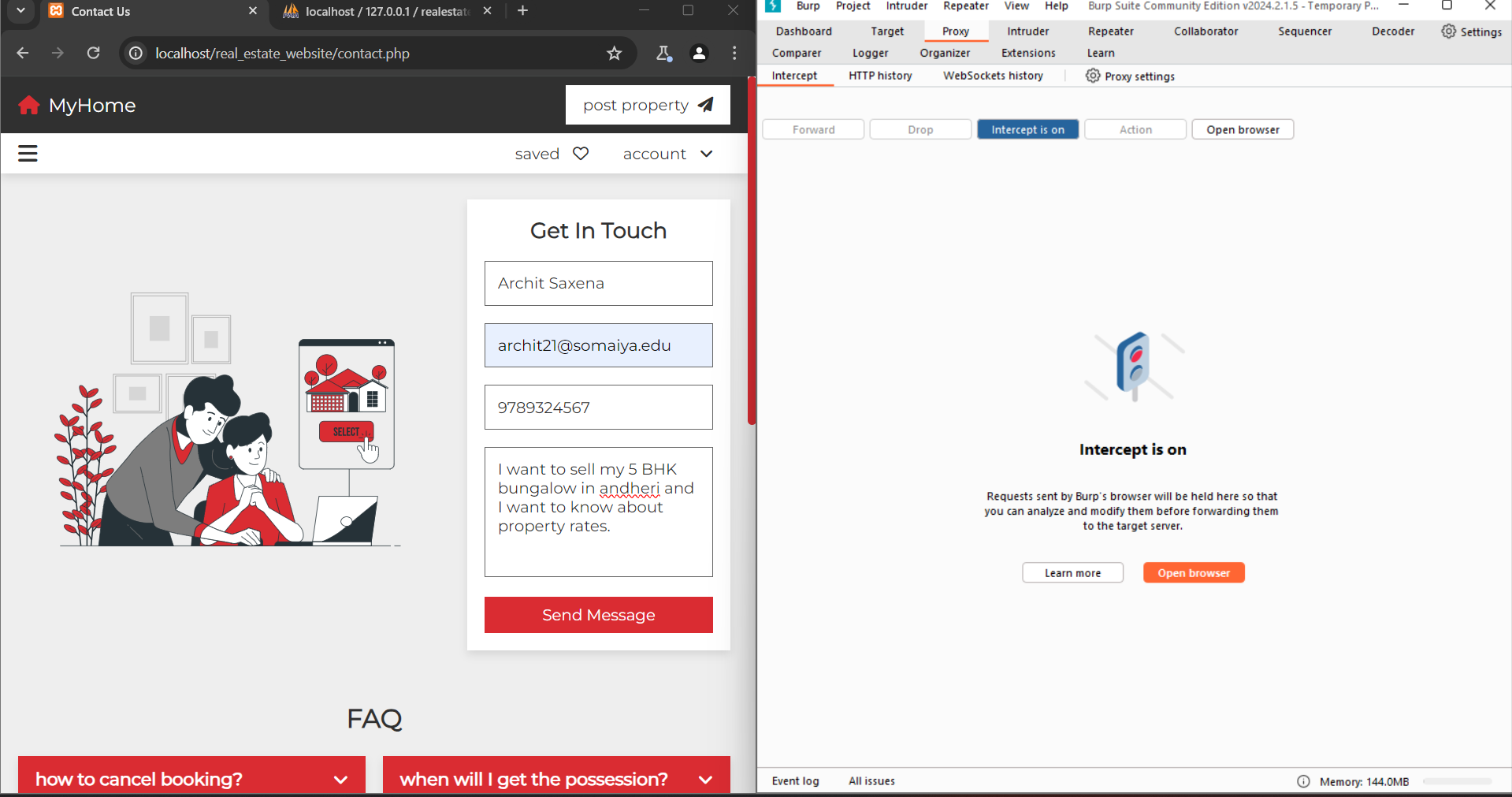
Step-2) PHPCode for submission after entering the message which is then sent to database in phpMyAdmin using post method.

****

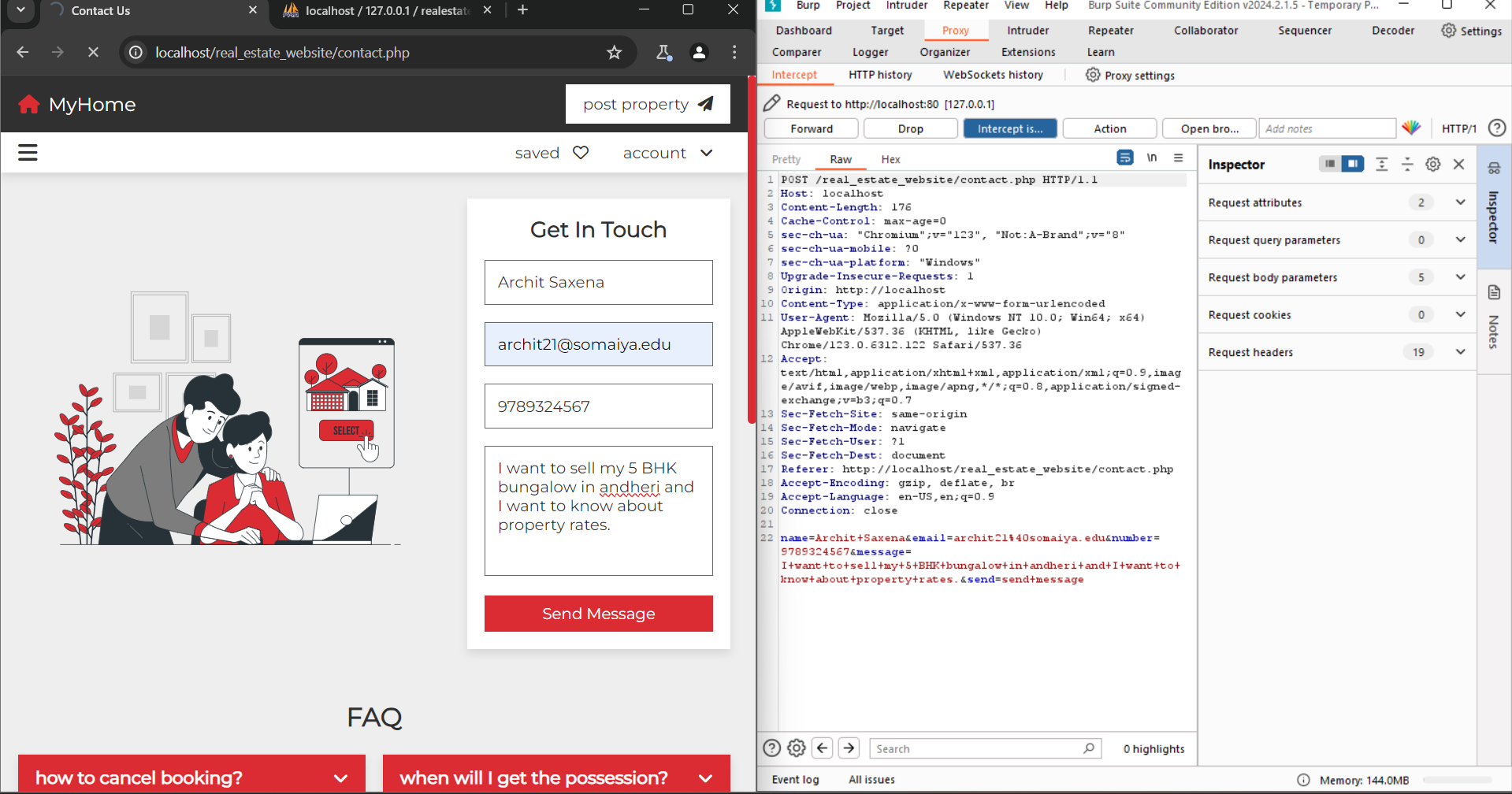
Step-3) Enter the form details when intercept is off.

****

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

****

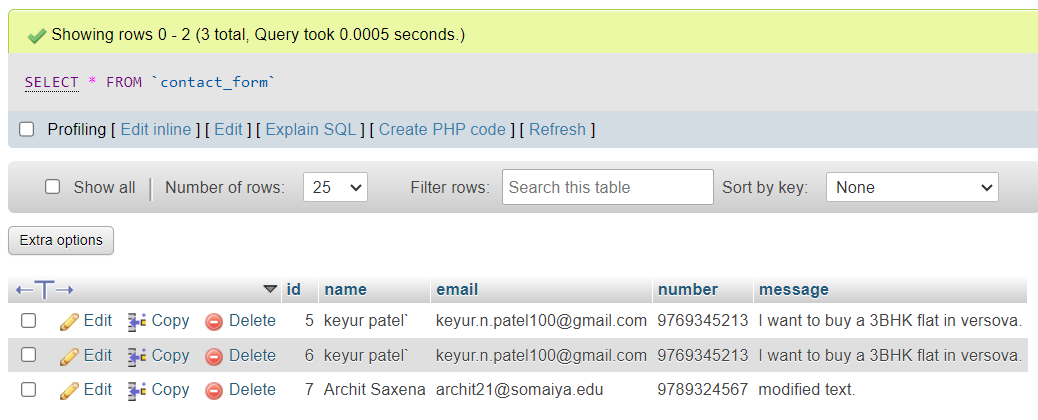
Step-5) The Request is then redirected to burpsuite which shows what is sent by user.

****

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

****

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

****

****

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

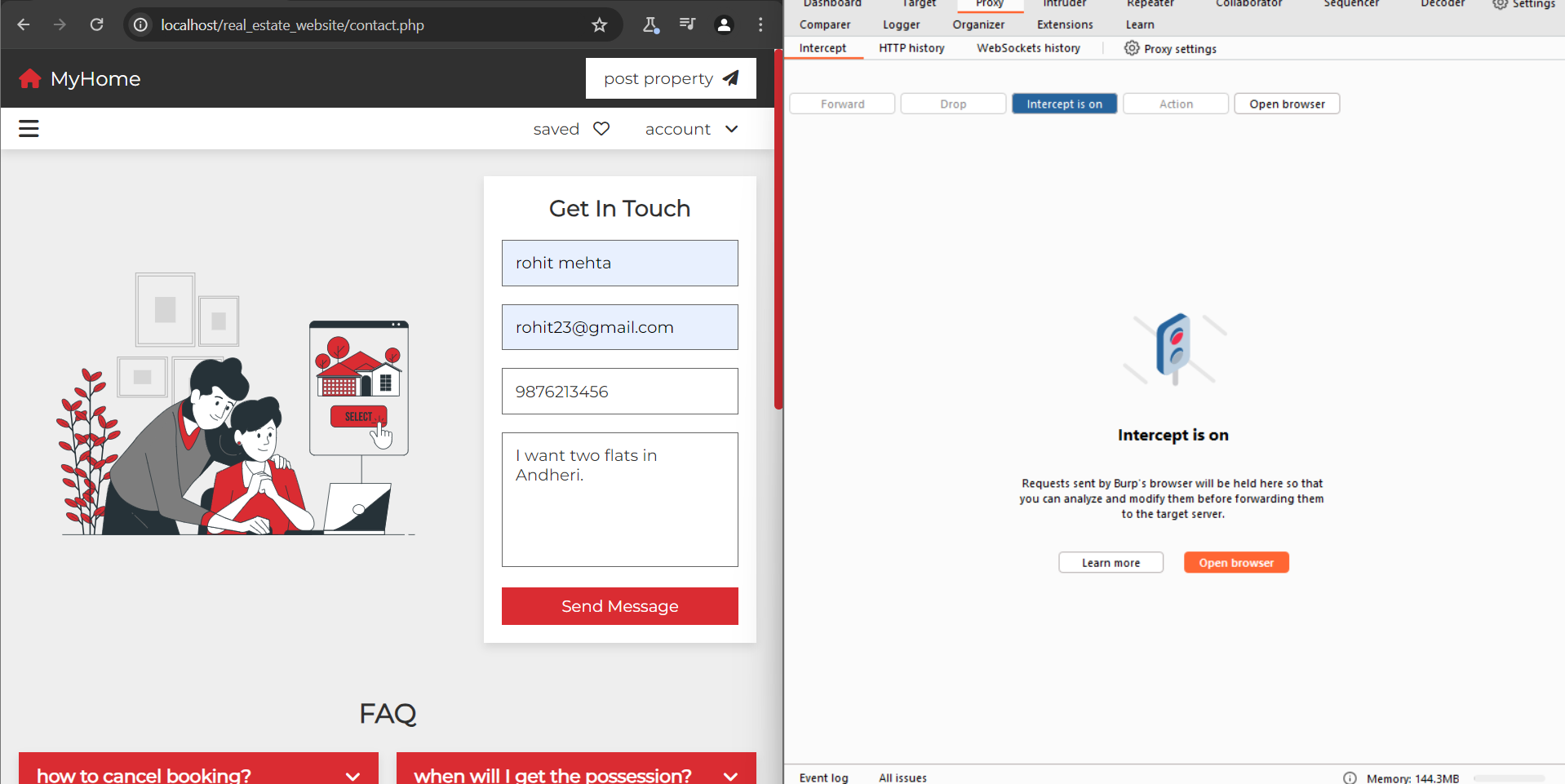
****

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

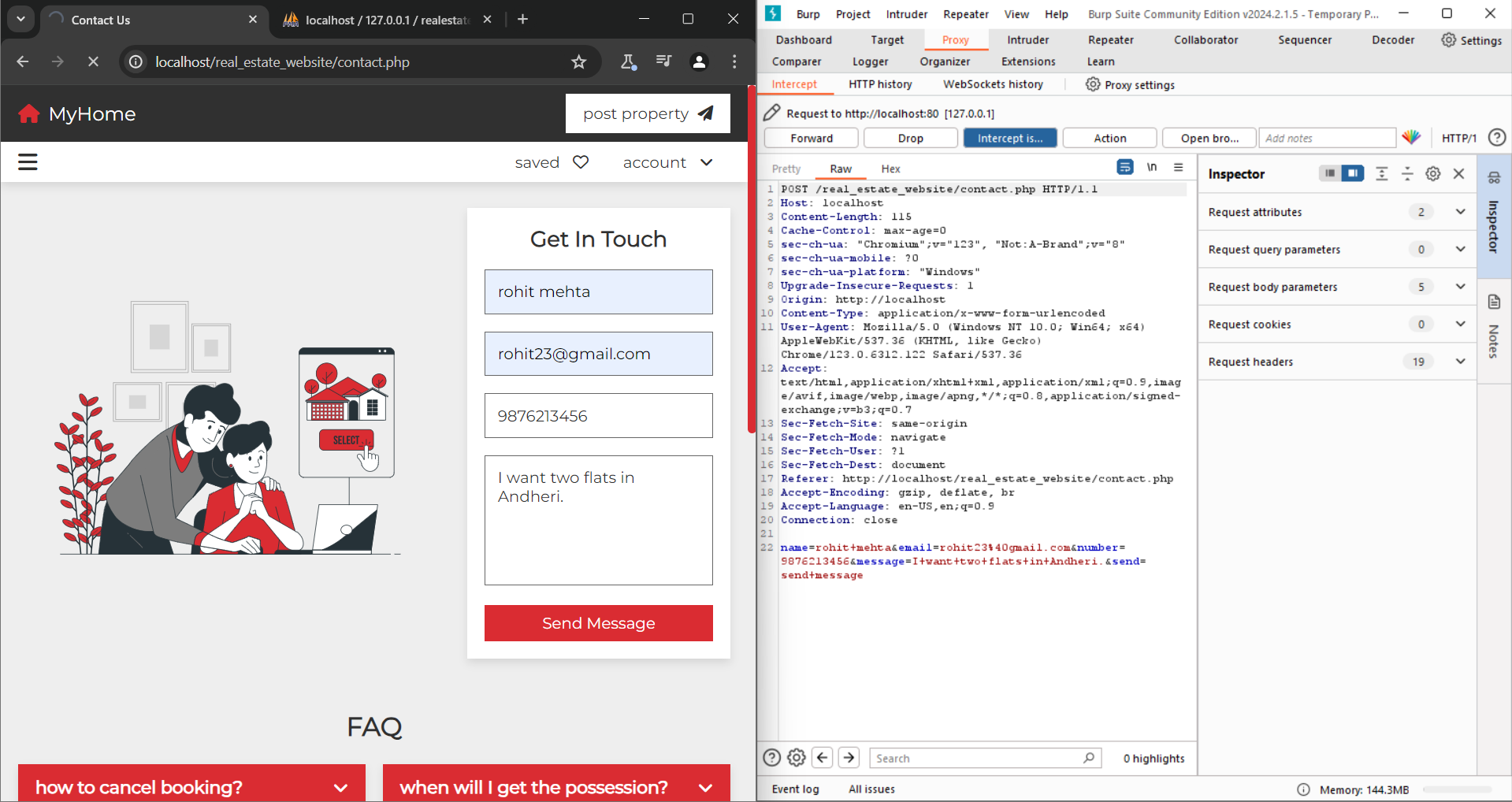
**Checksum:** 8c89d822e1797b82b011b2b60a3ea377d9c43ab360805f25413171f008bc43c8

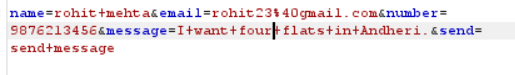
****

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

****

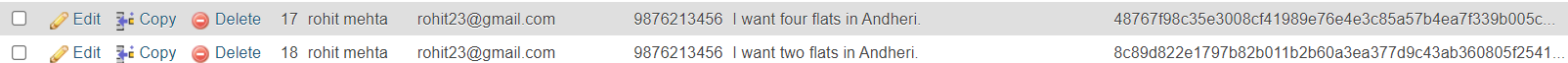
Step-4) Then the request is redirected to burpsuite where we will modify the text “I want two flats in Andheri” to “I want four flats in Andheri”.

****

****

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

**Checksum:** **48767f98c35e3008cf41989e76e4e3c85a57b4ea7f339b005c61dd748fc023aa**

****

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.