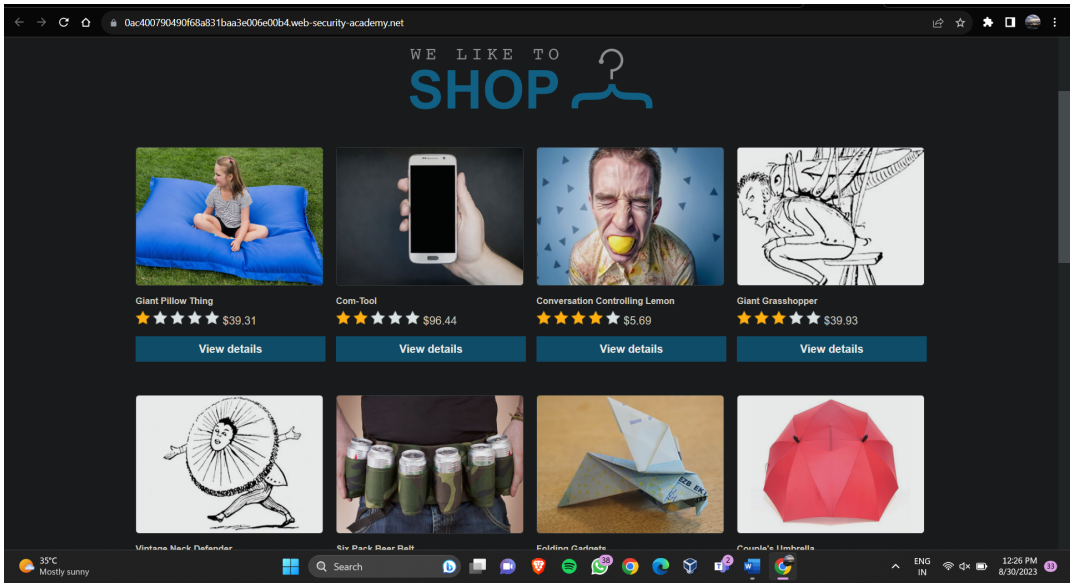
Rohit Patiballa
21BCI0278

Assignment - 1

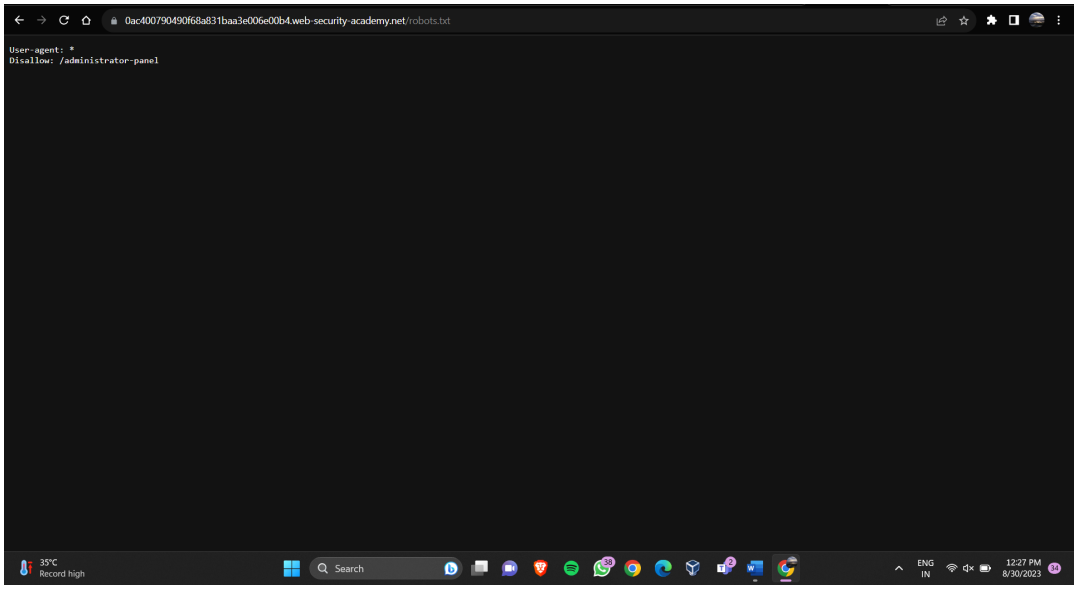
Checking the top 5 CWE vulnerabilities(2021)

1. **Broken Access Control -**
   CWE : CWE - 862: Missing Authorization
   OSWAP CATEGORY: A01:2021- Broken Access Control
   Description -
   The product does not perform an authorization check when an actor attempts to access a resource or perform an action.
   Business Impact -
   Assuming a user with a given identity, authorization is the process of determining whether that user can access a given resource, based on the user's privileges and any permissions or other access-control specifications that apply to the resource.
   When access control checks are not applied, users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information exposures, denial of service, and arbitrary code execution.
   Practical implementation -
   Visit an example website.



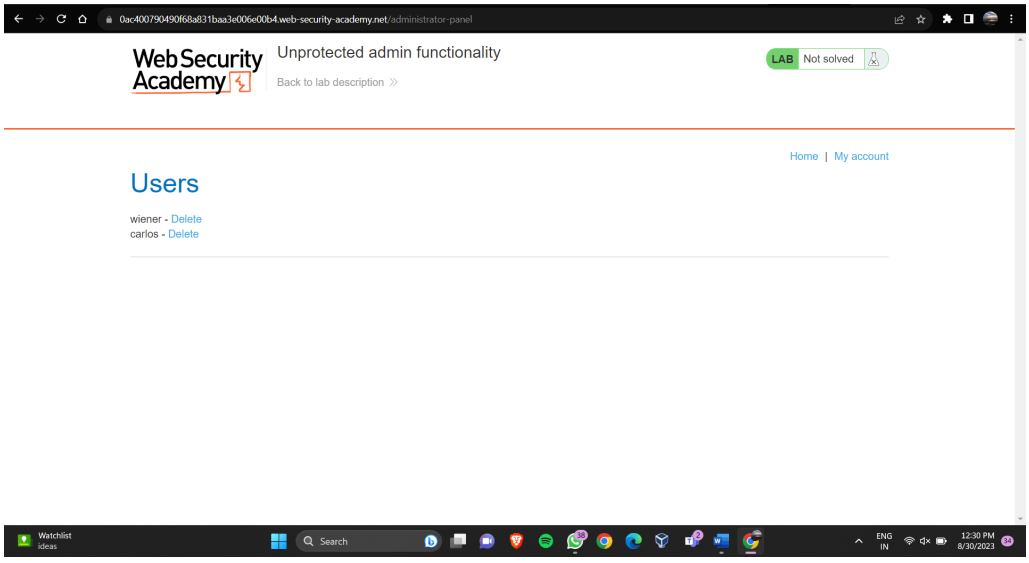
Rohit Patiballa
21BCI0278

Write robot.txt in the URL.



The above given site has vulnerability of broken access control because the Admin panel can be used using .txt in the URL

We can access admin panel.

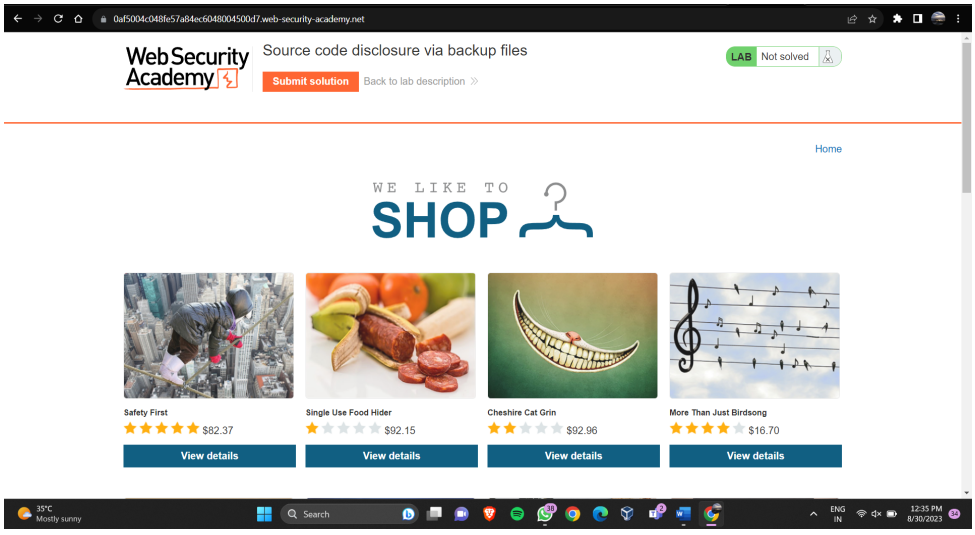


Therefore the above image shows that website has broken access control vulnerability.

Rohit Patiballa
21BCI0278

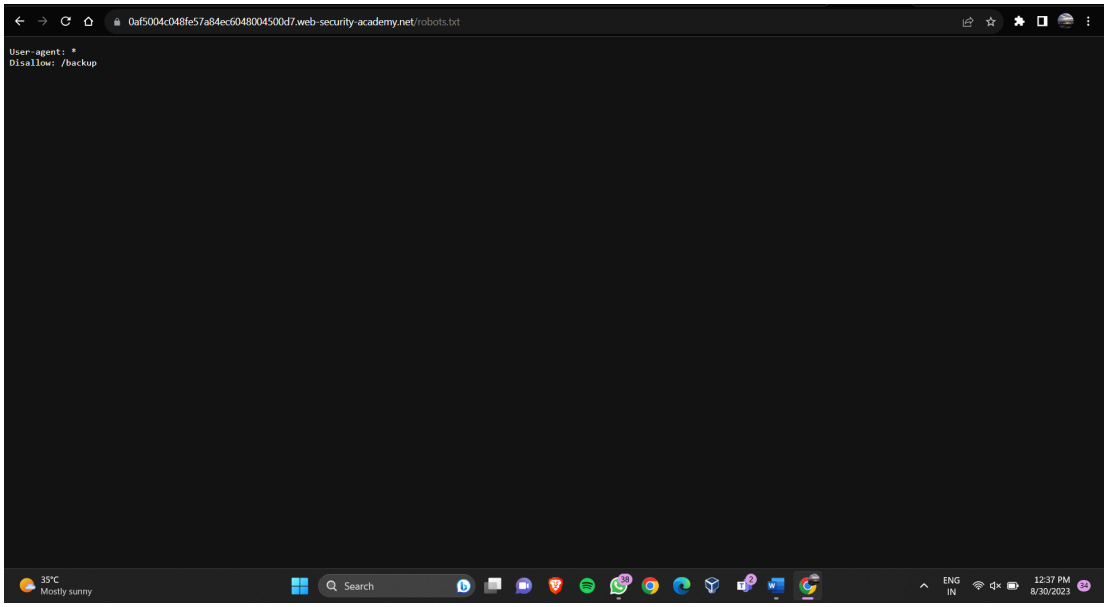
2. **Cryptographic Failures -**
   CWE: CWE - 324 :Use of a key past its expiration date
   OSWAP CATEGORY: A02:2021- Cryptographic Failures
   Description -
   The product uses a cryptographic key or password past its expiration date, which diminishes its safety significantly by increasing the timing window for cracking attacks against that key.
   Business Impact -
   While the expiration of keys does not necessarily ensure that they are compromised, it is a significant concern that keys which remain in use for prolonged periods of time have a decreasing probability of integrity. For this reason, it is important to replace keys within a period of time proportional to their strength.

   Practical Implementation -
   Visit an example website.

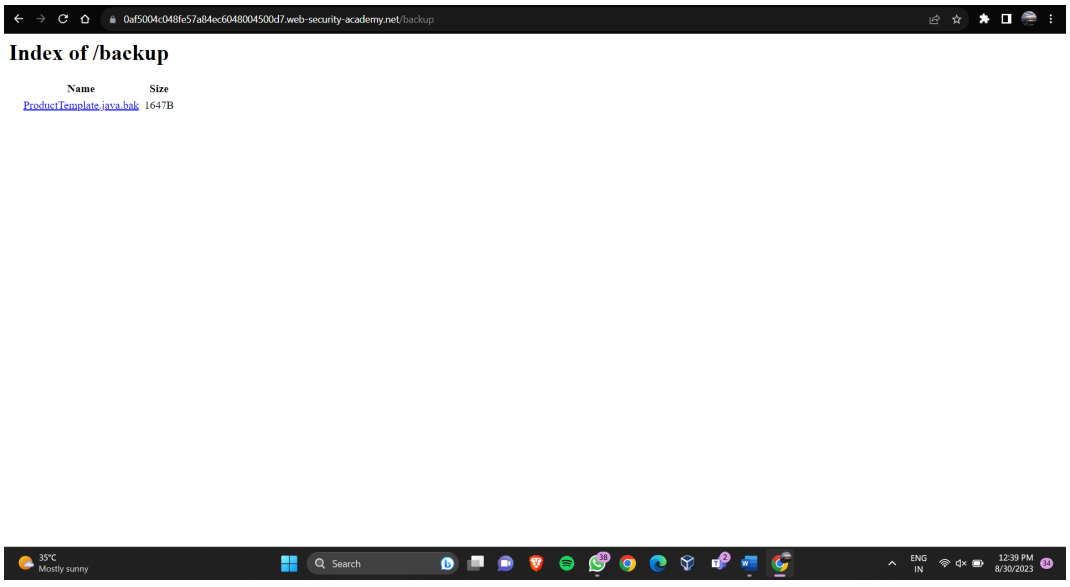


   Upon adding /robots.txt to the URL of the website , we get the disallowed link.

Rohit Patiballa
21BCI0278

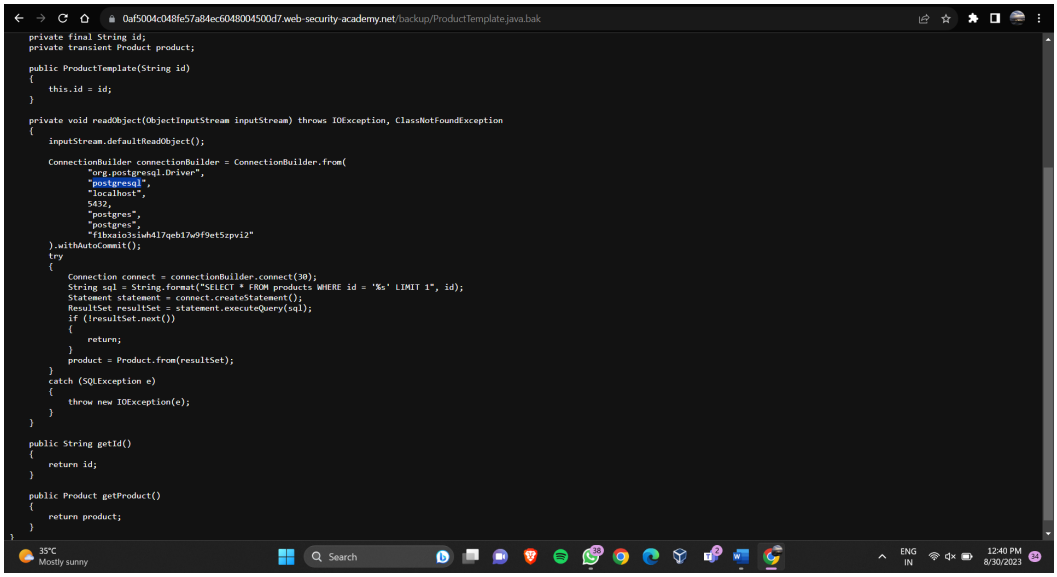


Adding the disallowed link to URL of the website.



Use the file , open it to get the decrypted password for the backup storage of the website.

Rohit Patiballa
21BCI0278



Therefore , website has vulnerability in cryptographic failures as the password was not encrypted before storing.

3. **Injection -**
   CWE-94: Improper Control of Generation of Code ('Code Injection')
   OSWAP CATEGORY: A03:2021- Injection
   Description -
   The product constructs all or part of a code segment using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the syntax or behavior of the intended code segment.

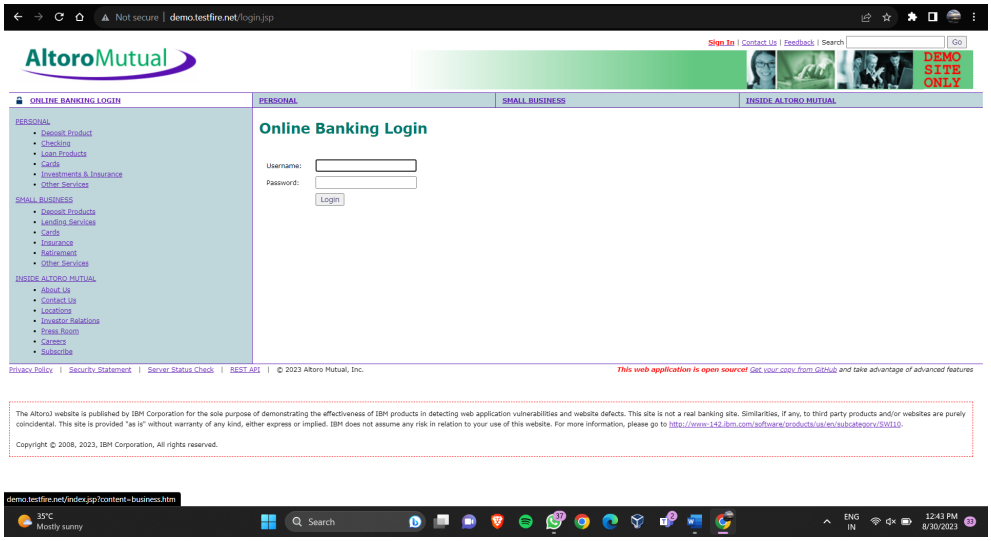
   Business Impact -
   When a product allows a user's input to contain code syntax, it might be possible for an attacker to craft the code in such a way that it will alter the intended control flow of the product. Such an alteration could lead to arbitrary code execution.

   Injection problems encompass a wide variety of issues -- all mitigated in very different ways. For this reason, the most effective way to discuss these weaknesses is to note the distinct features which classify them as injection

weaknesses. The most important issue to note is that all injection problems share one thing in common -- i.e., they allow for the injection of control plane data into the user-controlled data plane. This means that the execution of the process may be altered by sending code in through legitimate data channels, using no other mechanism. While buffer overflows, and many other flaws, involve the use of some further issue to gain execution, injection problems need only for the data to be parsed. The most classic instantiations of this category of weakness are SQL injection and format string vulnerabilities.

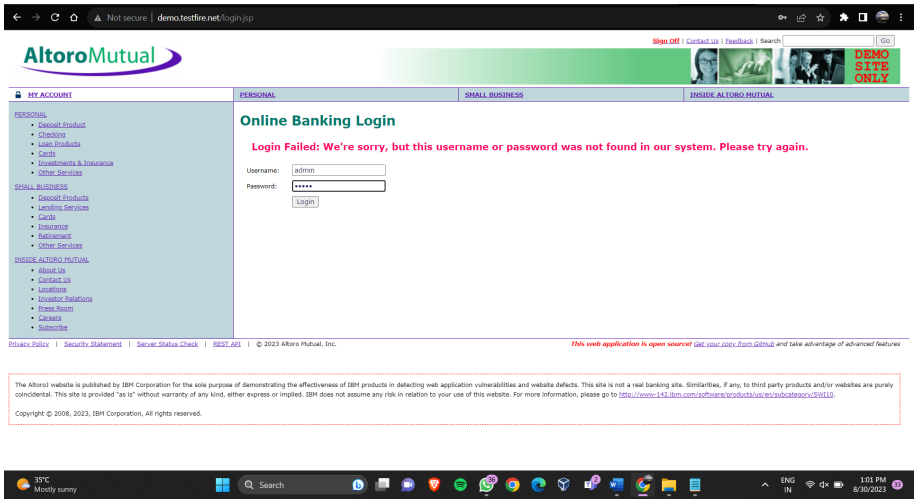
Practical Implementation -

Visit any sample website



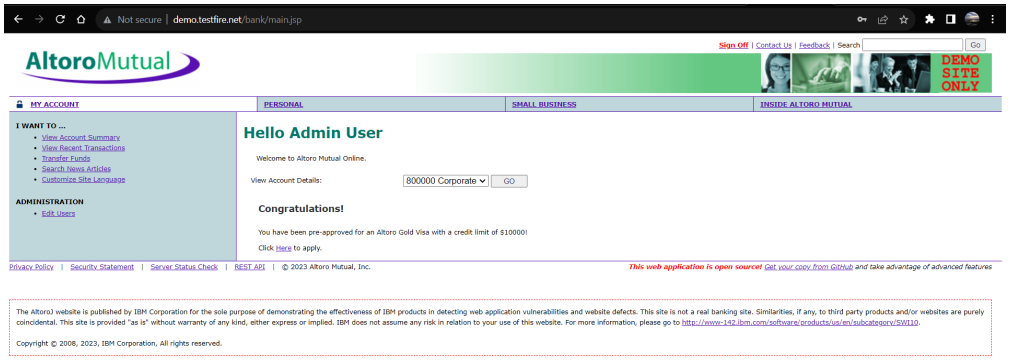
Try admin and any random password.

Rohit Patiballa
21BCI0278



We get , wrong password message.

Now try admin'-- and password as same as your first one.



As you can see , login is successful !

4. **Insecure Design -**
   CWE-539: Use of Persistent Cookies Containing Sensitive Information
   OSWAP CATEGORY: A04:2021- Insecure Design
   <u>Description -</u>
   The web application uses persistent cookies, but the cookies contain sensitive information.

   <u>Business Impact -</u>

Cookies are small bits of data that are sent by the web application but stored locally in the browser. This lets the application use the cookie to pass information between pages and store variable information. The web application controls what information is stored in a cookie and how it is used. Typical types of information stored in cookies are session identifiers, personalization and customization information, and in rare cases even usernames to enable automated logins. There are two different types of cookies: session cookies and persistent cookies. Session cookies just live in the browser's memory and are not stored anywhere, but persistent cookies are stored on the browser's hard drive. This can cause security and privacy issues depending on the information stored in the cookie and how it is accessed.

Practical Implementation -

Consider the above example , since there is a vulnerability and SQL injection can be executed there is an issue with the design. There for the above website has Insecure design vulnerability as well.

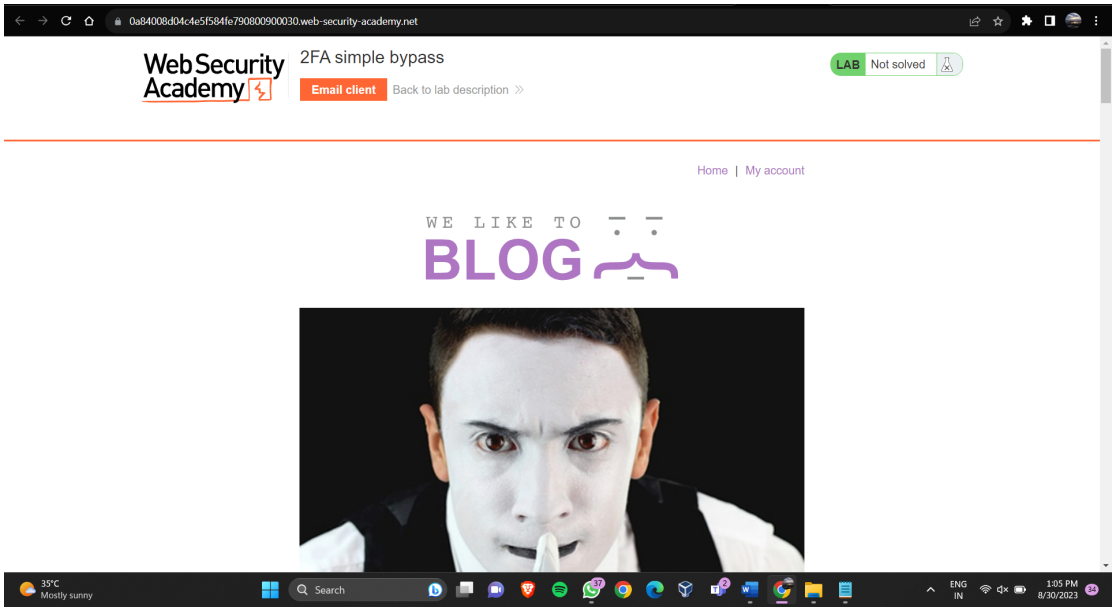
5. **Security Misconfiguration**
   CWE-547: Use of Hard-coded, Security-relevant Constants
   OSWAP CATEGORY: A05:2021- Security Misconfiguration
   Description -
   The product uses hard-coded constants instead of symbolic names for security-critical values, which increases the likelihood of mistakes during code maintenance or security policy change.

   Business Impact -
   If the developer does not find all occurrences of the hard-coded constants, an incorrect policy decision may be made if one of the constants is not changed. Making changes to these values will require code changes that may be difficult or impossible once the system is released to the field. In addition, these hard-coded values may become available to attackers if the code is ever disclosed.

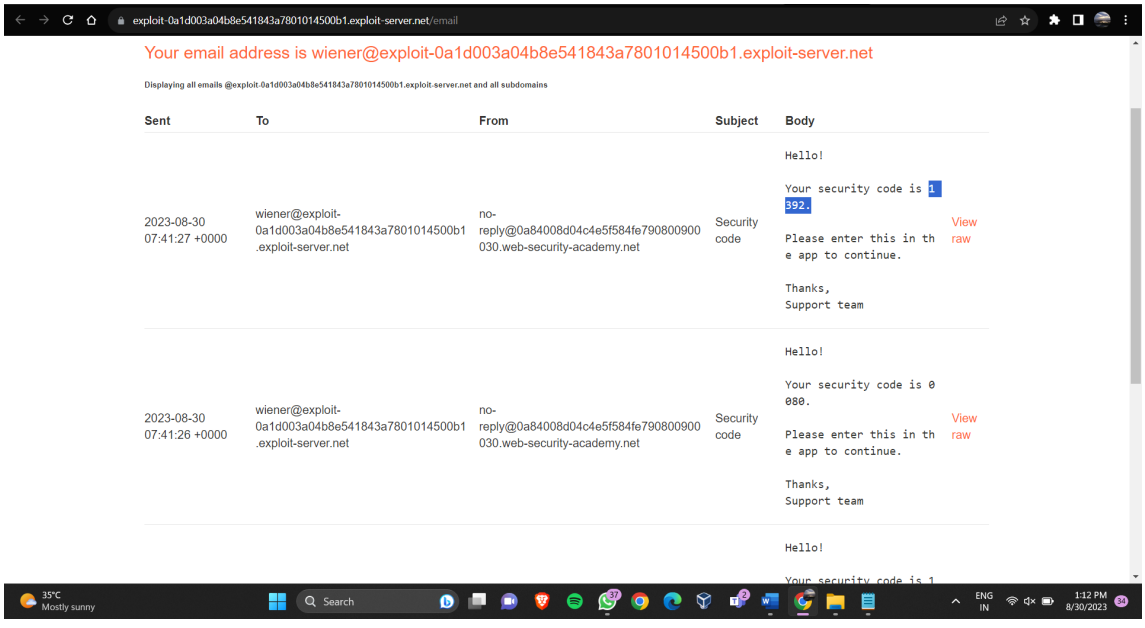
Rohit Patiballa
21BCI0278

Practical Implementation -

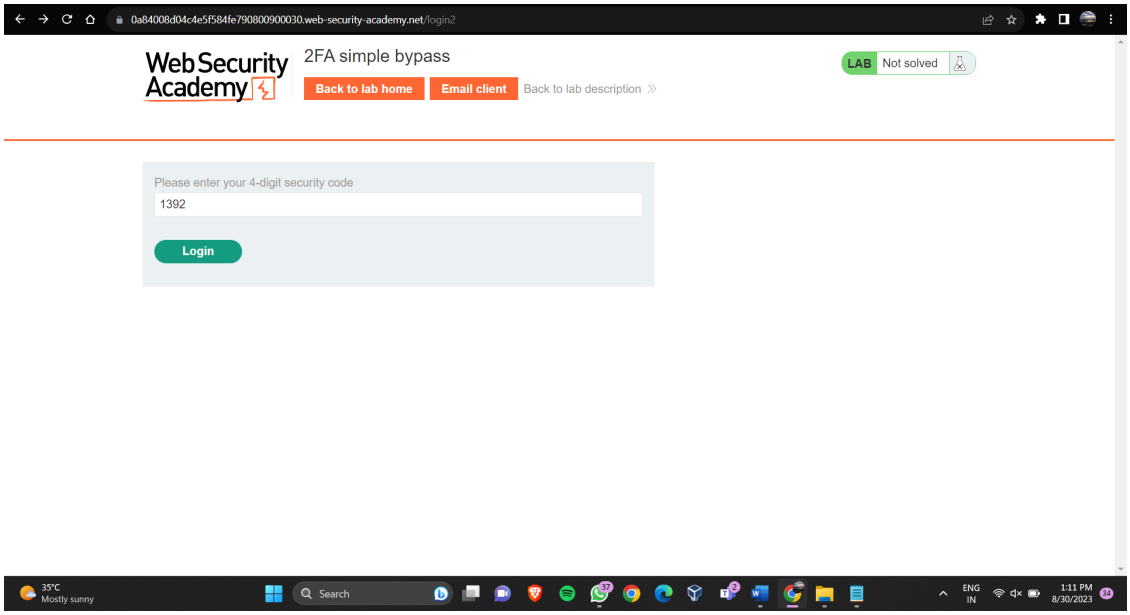
Visit any sample website

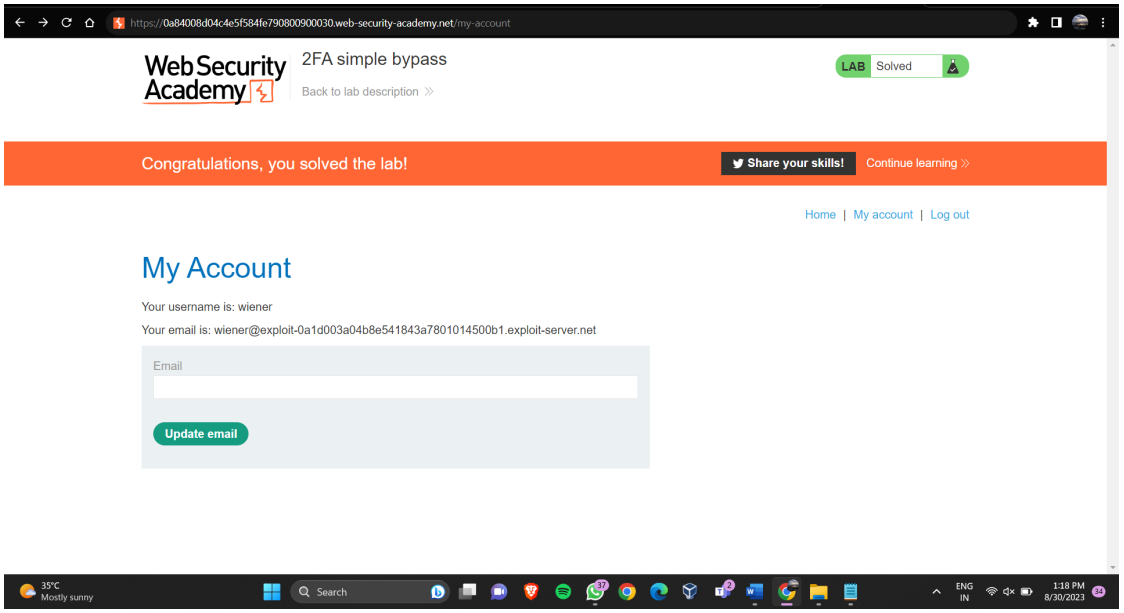


Log in with given credentials ie , username - wiener and ps - peter
You get a 2FA verification to the mail.



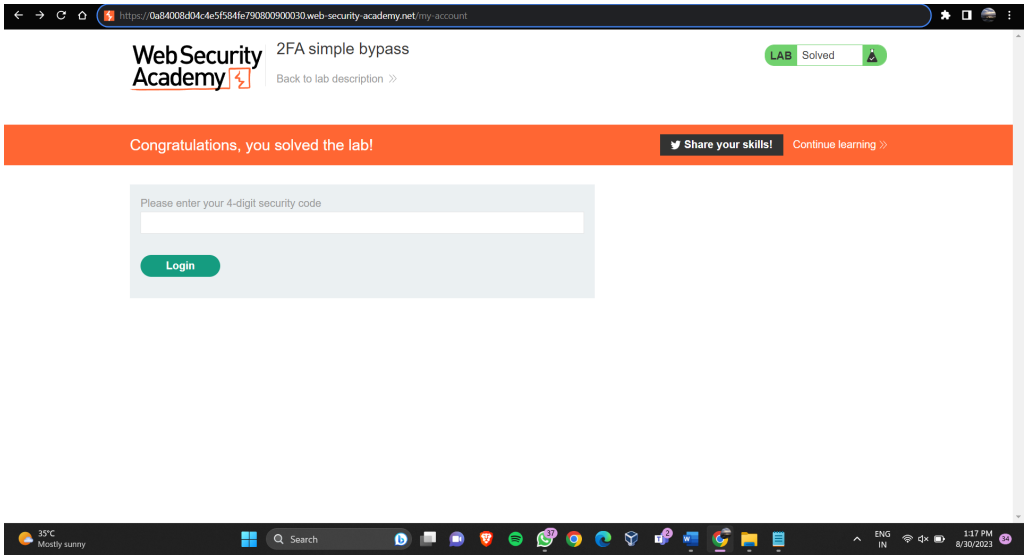
Rohit Patiballa
21BCI0278



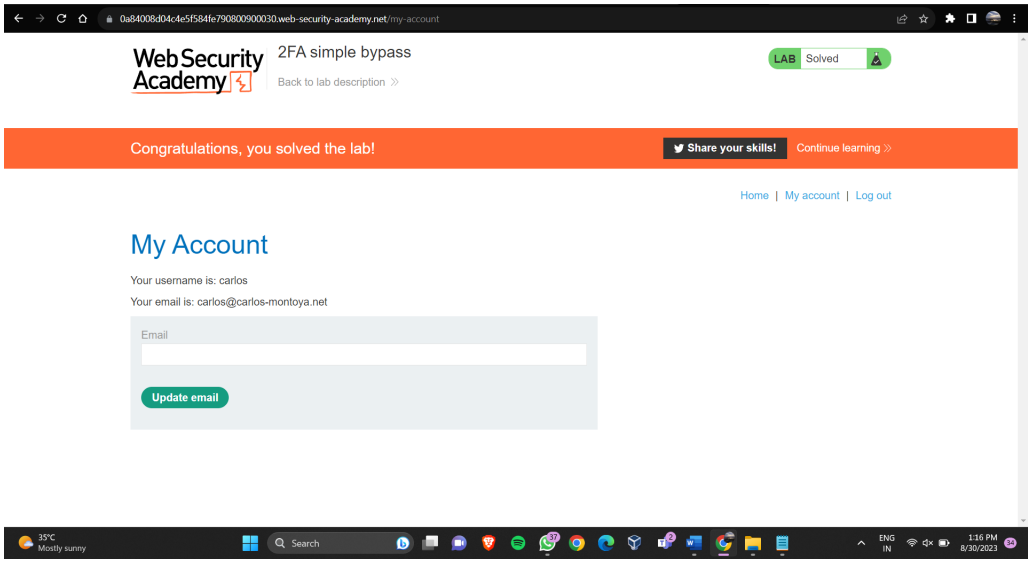


Login using the 2FA code and login gets successful !
Now logout and try logging in to the victims account using given credentials ie , username - carlos and ps - montoya

Rohit Patiballa
21BCI0278



Update url to /my-account and you can access the account !



We logged in successfully !

Therefore the sample website has Security Misconfiguration Vulnerability.