Practical exercises and questions:

In the following section there will be six exercises about Cross Site Scripting and SQL injection. We suggest you to follow the exercises in the order in which they are presented.

Exercise 1: In the following exercise you will experience what happens if countermeasures are not implemented in the correct way.
You can find the lab here https://portswigger.net/web-security/cross-site-scripting/dom-based/lab-dom-xss-stored

Objective: The goal of the exercise is to exploit the vulnerability to call an alert function. Keep in mind that this lab contains a stored DOM vulnerability in the comment functionality.

A suggestion in case you are lost: *As in the previous labs, errors might be useful. If you do not succeed the first time, try again! You might be able to bypass the countermeasure…*

Exercise 2: In the following exercise you will experience what happens if a vulnerability is used to perform another attack. In this particular exercise the XSS vulnerability is used to perform a CSRF attack.
You can find the lab here: https://portswigger.net/web-security/cross-site-scripting/exploiting/lab-perform-csrf

Objective: The goal of the exercise is to exploit the Cross Site Scripting vulnerability to perform a Cross Site Request Forgery attack. Keep in mind that this lab contains a stored XSS vulnerability in the comment function. Use these credentials to access the application: Username: wiener – Password: peter

A suggestion in case you are lost: *Try to understand how a legitimate request works. You have to construct a script to perform the CSRF attack and insert it in the comment section which is vulnerable to XSS.*

Exercise 3: In the following exercise event handlers and href attributes are blocked but these countermeasures can be bypassed.
You can find the lab here: https://portswigger.net/web-security/cross-site-scripting/contexts/lab-event-handlers-and-href-attributes-blocked

Objective: The goal of the exercise is to perform a Cross Site Scripting attack that injects a vector that, when clicked calls the alert function. Keep in mind that this lab contains a reflected XSS vulnerability whit some whitelisted tags but all events and href attributes are blocked.

A suggestion in case you are lost: *You need to find a way to bypass the countermeasures. Encoding and animations might be useful.*

Exercise 4: Use the techniques learned in the previous labs to attack the application. You need to log yourself in as an administrator.
You can find the lab here: https://portswigger.net/web-security/sql-injection/union-attacks/lab-retrieve-multiple-values-in-single-column

Objective: The goal of the exercise is to perform a SQL injection UNION attack that retrieves all usernames and passwords and use the information to log in as the administrator user. Keep in mind that this lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables. The only thing you know is that the database contains a different table called users with columns named username and password.

A suggestion in case you are lost: *You need to discover how many columns is the query returning and how many of those can contain text. Using the discovered information you might be able to make the application return what you need.*

Exercise 5: In the following exercise you will perform a time based attack on the database.
You can find the lab here: https://portswigger.net/web-security/sql-injection/blind/lab-time-delays

Objective: The goal of the exercise is to perform a SQL attack to cause a 10 second delay. This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.
The results of the SQL query are not returned, and the application does not respond any differently based on whether the query returns any rows or causes an error. However, since the query is executed synchronously, it is possible to trigger conditional time delays to infer information.

A suggestion in case you are lost: *You might be able to modify some parameters to suit your needs.*

Exercise 6: In the following exercise you will perform a time based attack on the database and retrieve useful information.
You can find the lab here: https://portswigger.net/web-security/sql-injection/blind/lab-time-delays-info-retrieval

Objective: The goal of the exercise is to perform a SQL injection attack that makes you able to log in as the administrator user. Keep in mind that this lab contains a SQL injection vulnerability. The results from the query are not returned and the application does not respond any differently based on whether the query return any rows or causes errors. However, since the query is executed synchronously, it is possible to trigger conditional time delays to infer information.

A suggestion in case you are lost: *Use time delays to test Boolean conditions. You need to find out the name of the administrator account and the length of the password. Once you have done that you need to find out the characters of the password. Assume*

the password only contains lowercase alphanumeric characters. Burp Intruder should help you perform this attack.

Questions:

In this little section you will find some simple questions to test your comprehension of the topic.

1. What are the best strategies to use in order to defend against injection attacks?
2. Explain how XSS vulnerabilities can be detected.
3. Explain how you can protect from XSS attacks.
4. Describe the different types of SQL injection.
5. Explain how you can defend against SQL injection attacks.