OWASP Juice Shop

OWASP Juice Shop

This machine uses the OWASP Juice Shop vulnerable web application to learn how to identify and exploit common web application vulnerabilities.
This room has been designed for beginners, but can be completed by anyone.

[Task 1] Connect To Our Network

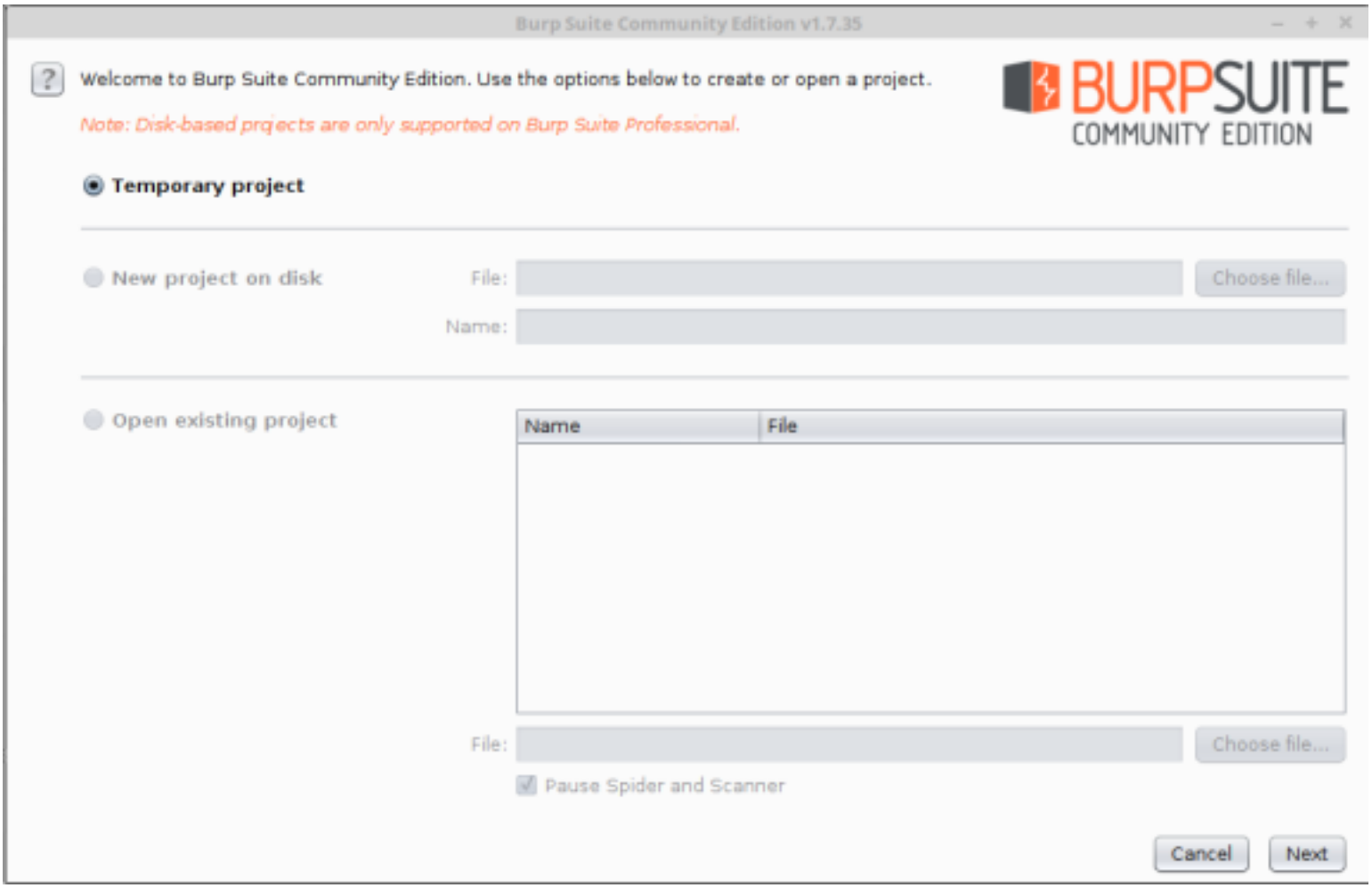
To access the OWASP Juice Shop machine, you need to a connect to our network. If you haven't connected to our network before, you can use the guide here.
Please note that the machine may take up to a few minutes to instantiate and run the juice shop application.
Credits to OWASP and Bjorn Kimminich

#1
Connect to our Network.

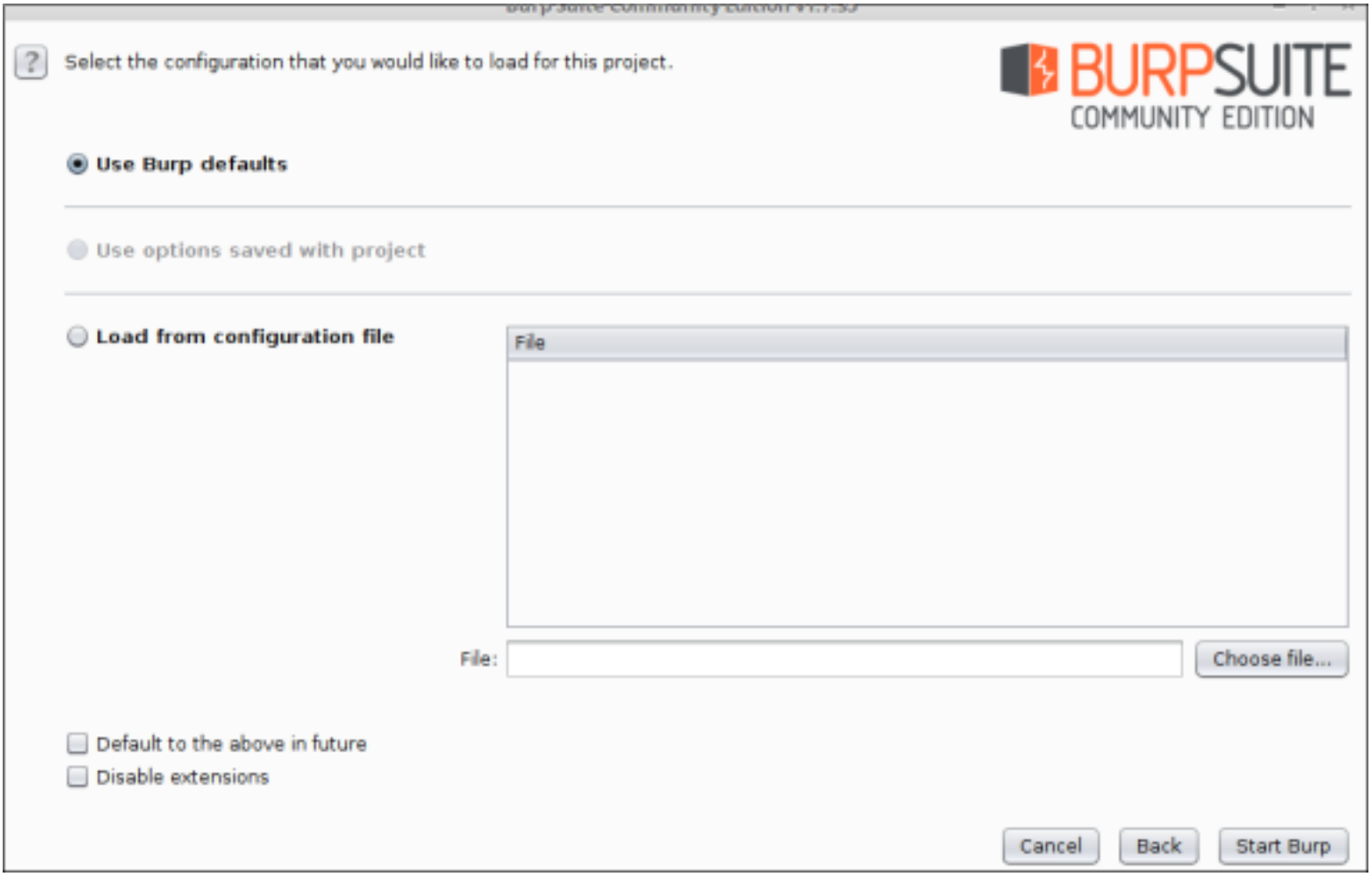
No answer needed

[Task 2] Configure Burp(If you haven't already)

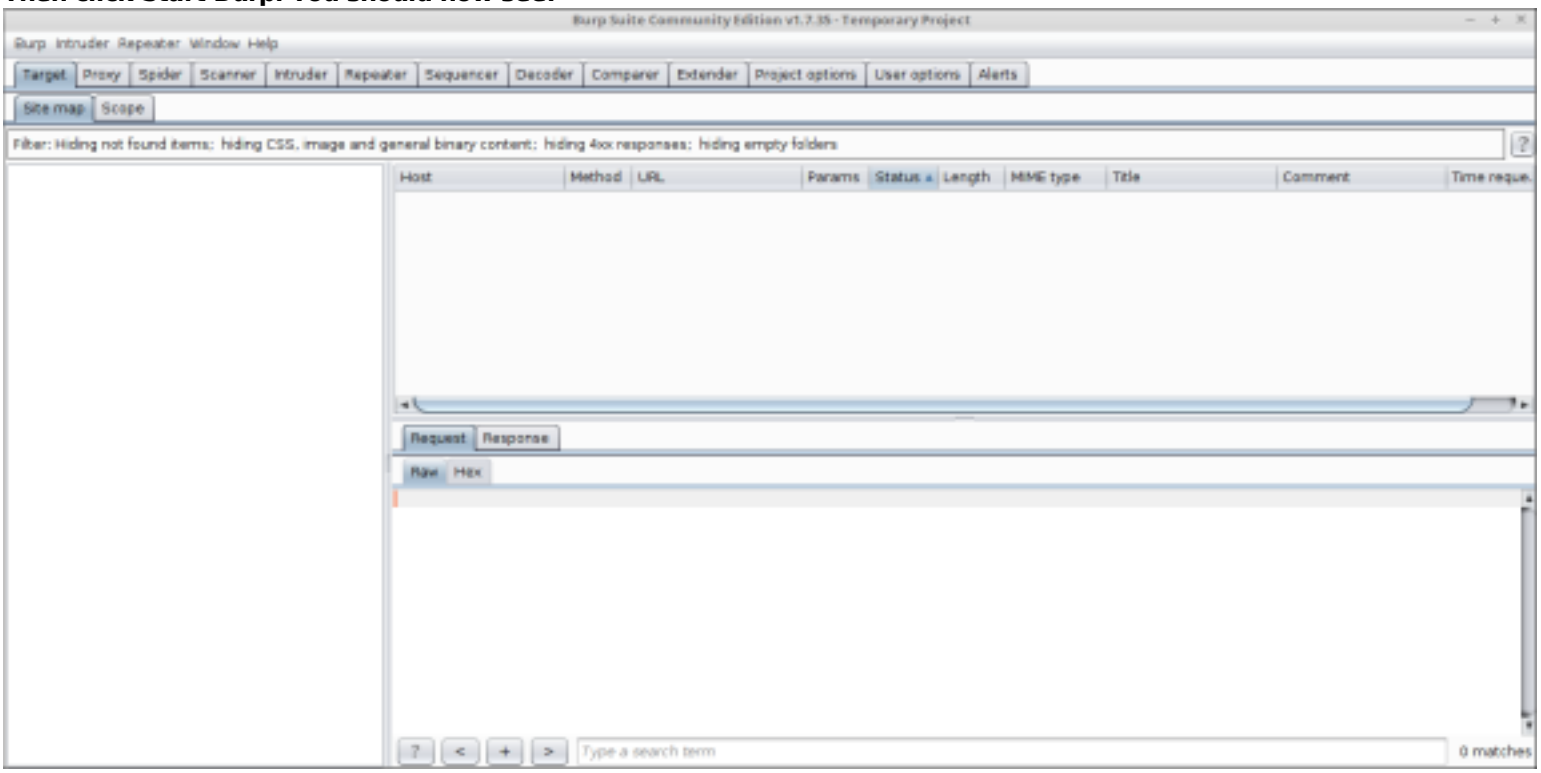
Before actually testing the application, it's important to have Burp set up(if you haven't already). You can follow these instructions, or check out the instruction on this room.
Download Burp from here (make sure you have Java installed too).
One you have Burp installed open the application. You should be presented with the following interface:



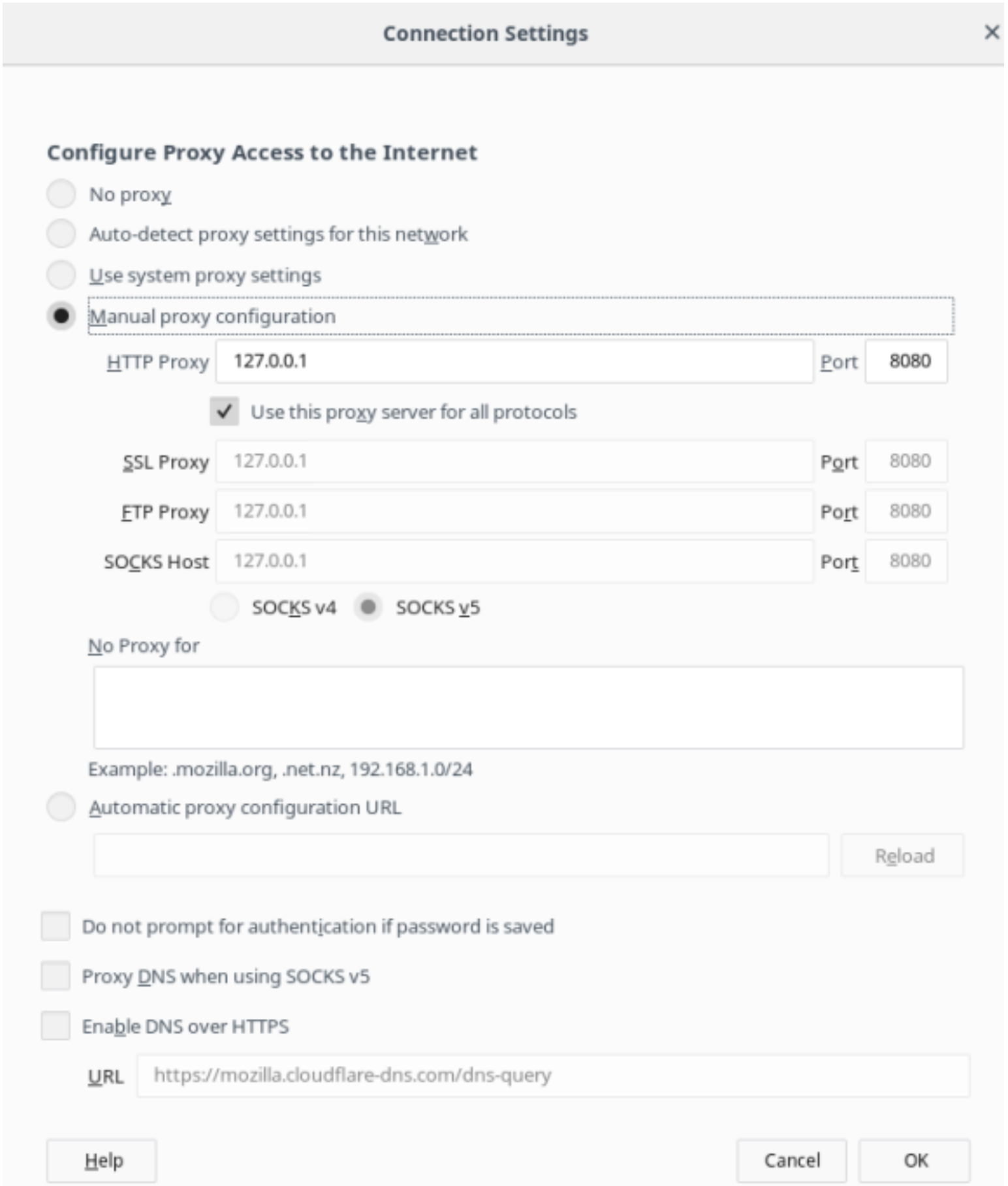
If you want to save a project, do so. Otherwise click Next. Which will bring you to:



Then click Start Burp. You should now see:

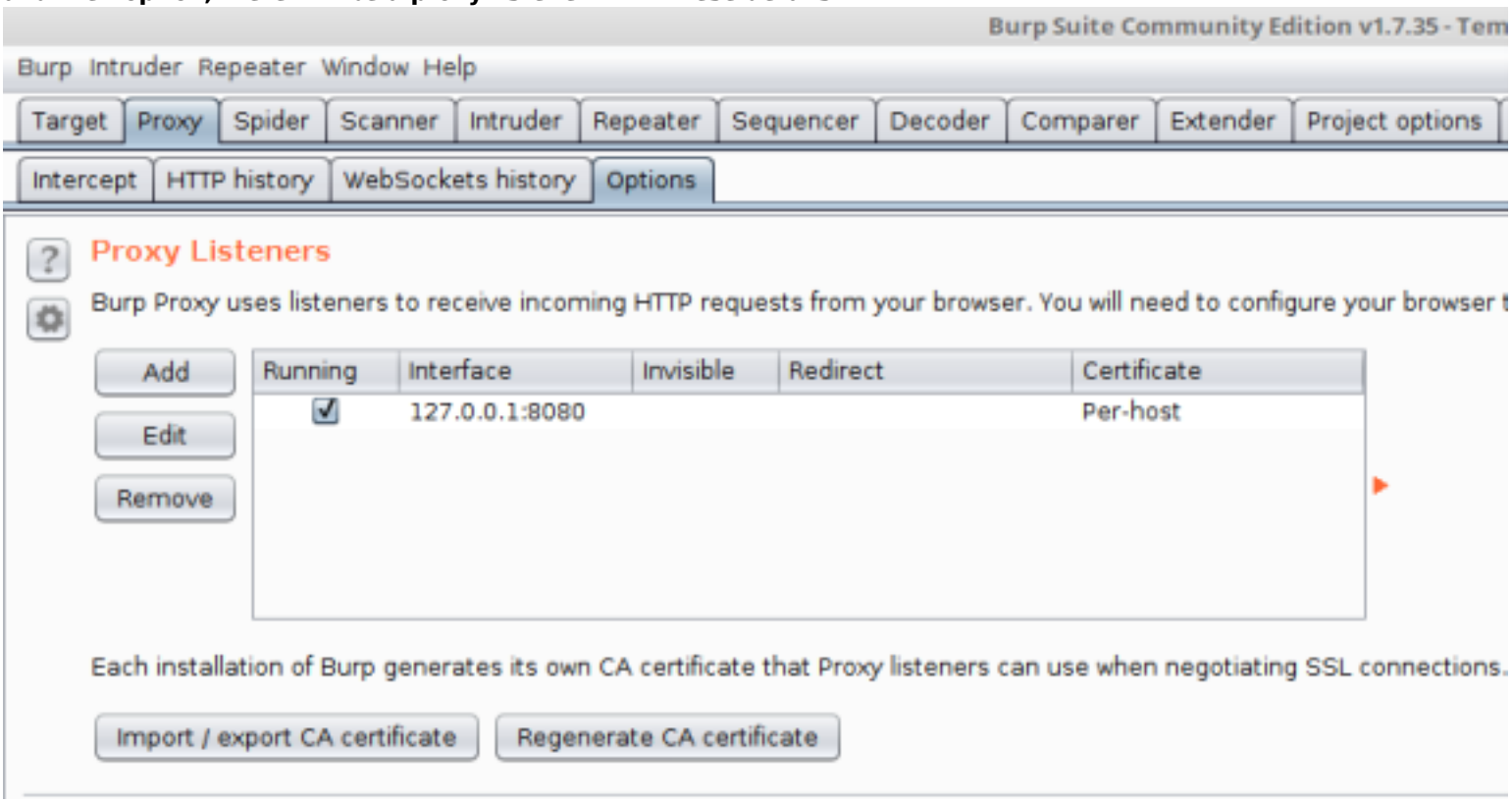


Now we have Burp installed we need to get it to intercept our traffic. I will be doing this with Firefox. Other browsers will work, just have to find the correct browser setting.
On Firefox, open the preferences (about:preferences#general) and scroll to the bottom where you can see Network Settings then click on Settings. You will see the following:

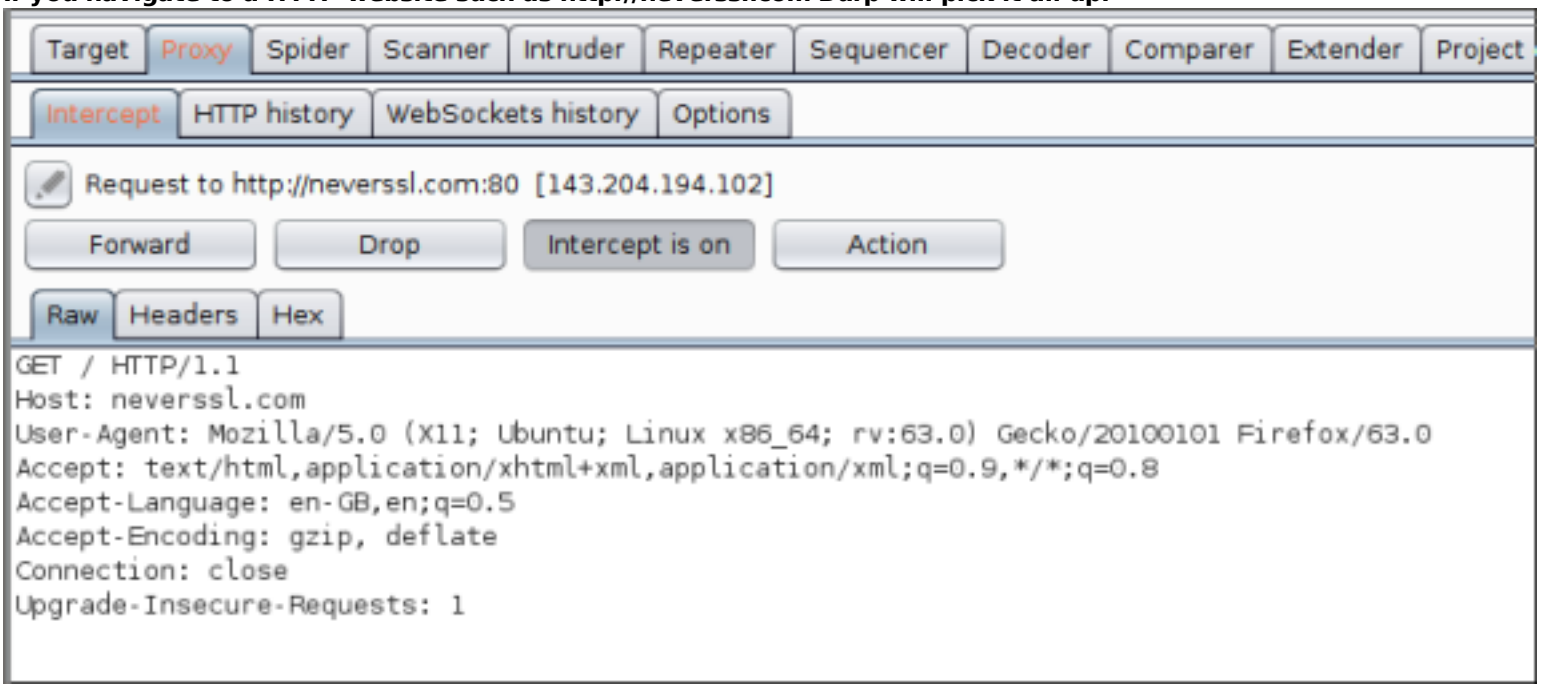


Select Manual proxy configuration and copy the same config as me. You should simply have to type in 127.0.0.1 in the HTTP proxy, select the checkbox with "Use this proxy for all protocols" and type in Port 8080. Once this is complete, click ok.
To stop your browser from tunneling everything through to your machine first, open up your firefox network settings again and click "No proxy"

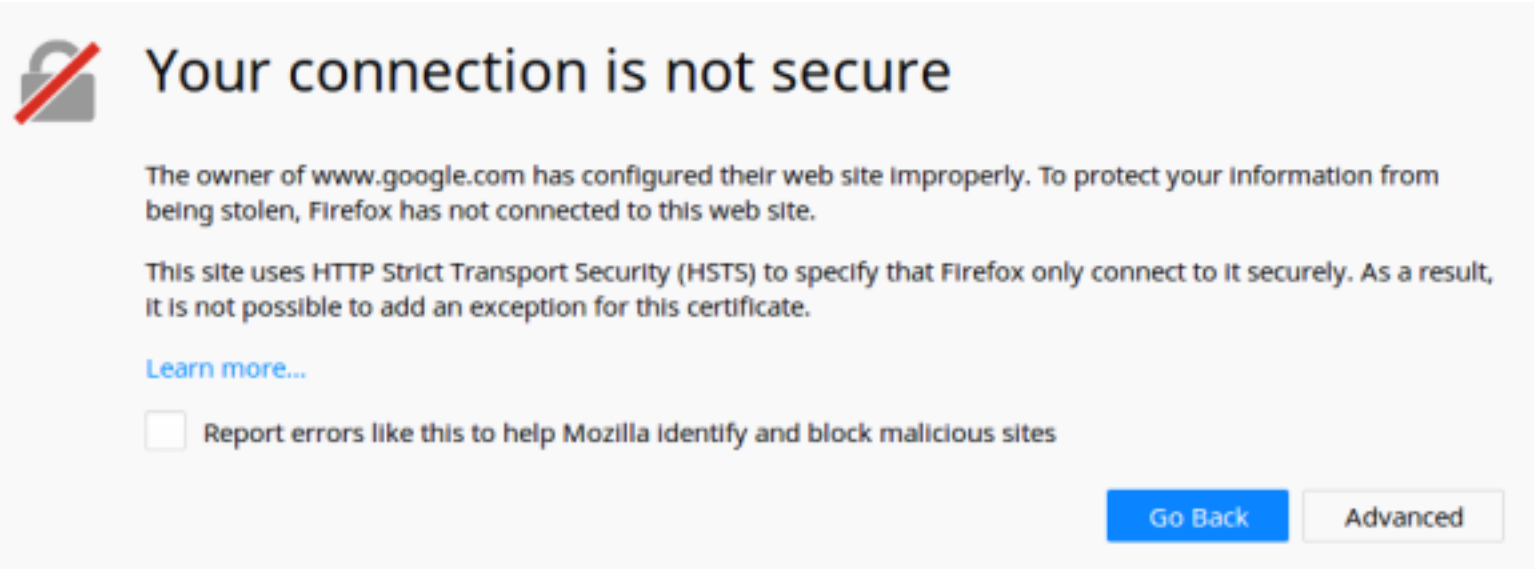
What you are doing now is proxying all of your web traffic through your local machine that is being intercepted by anything that is listening. Which in this case, will be Burp Suite. You will find if you open Burp Suite, click Proxy and then option, there will be a proxy listener with these details:



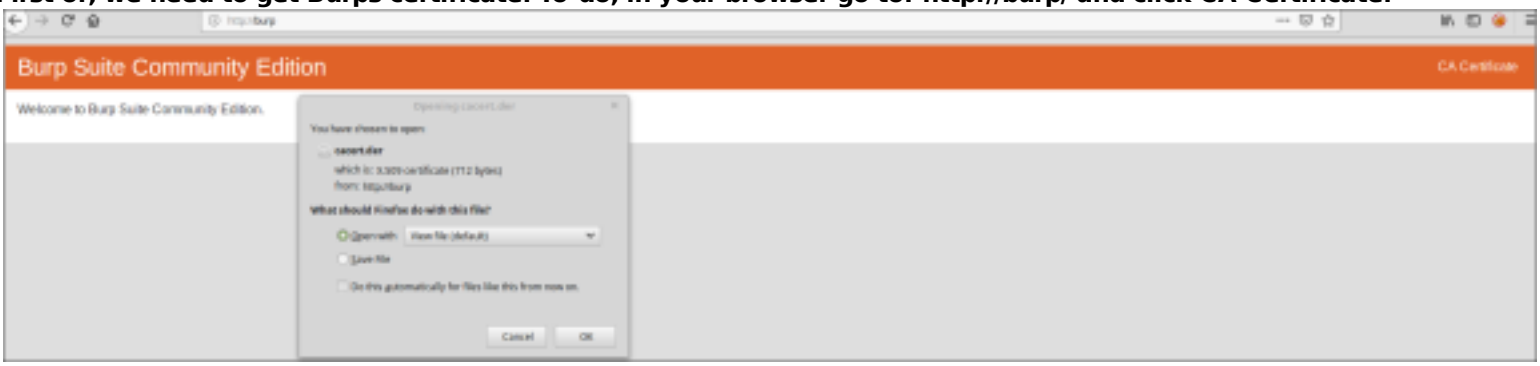
Make sure your checkbox for running is ticked.
***Hooorrraaaayyy*, we now have Burp Suite intercepting any traffic we generate through the browser... on HTTP.**
If you navigate to a HTTP website such as http://neverssl.com Burp will pick it all up:



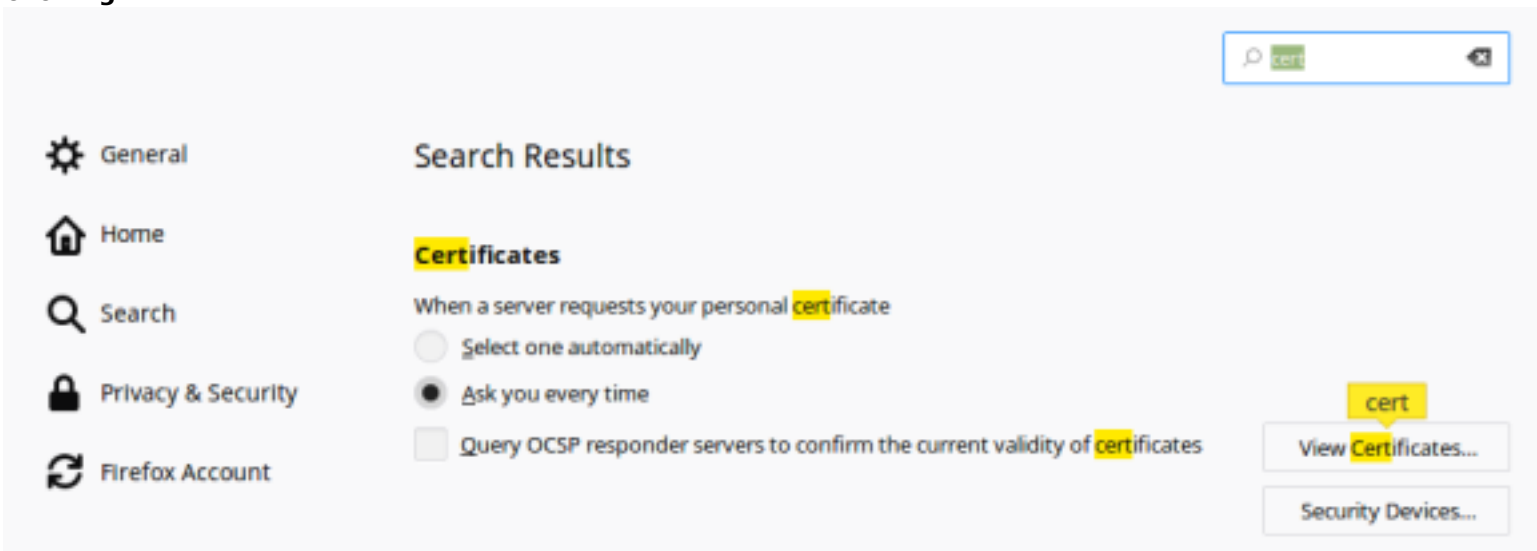
Burp will hold the proxied request until you either stop intercepting or click the forward button. You may a;sp see lots of other request Burp picks up. This is all traffic your browser is generating.
However, if we visit a HTTPS site such as: https://google.com we will get a horrible TLS error:



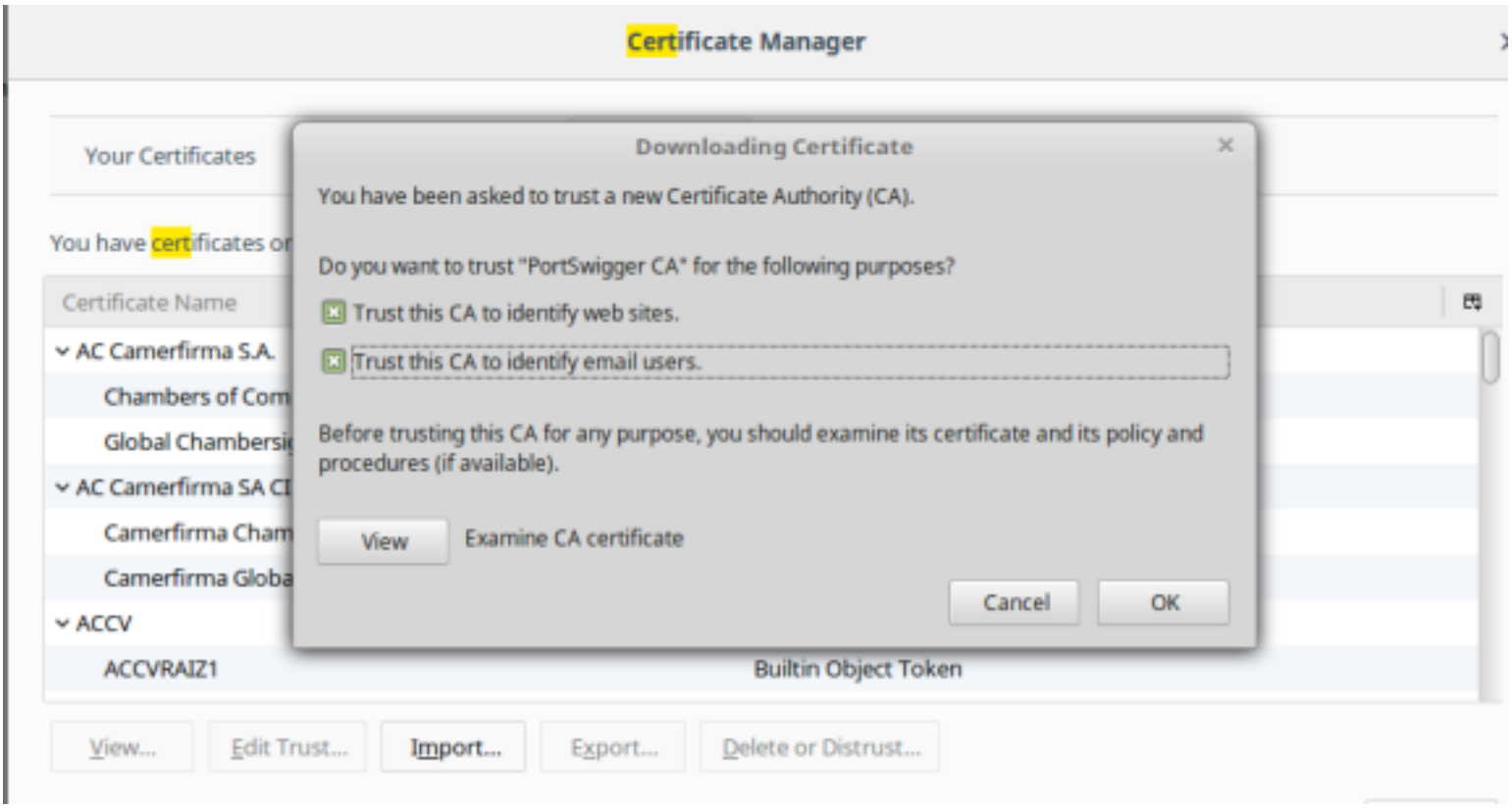
Lots of sites have TLS (HTTPS) to encrypt the data from the client to their server. As we are a man (or in this case a proxy) in the middle (MITM), the browser will think there is something wrong and will throw an error as seen above. To get around this, we have Burp sign our traffic with its certificate and tell out browser to make Burps TLS singing a Certificate Authority (basically just telling the browser that anything signed by Burp is all good).
First of, we need to get Burps certificate. To do, in your browser go to: http://burp/ and click CA Certificate.



Select Save File and download it.
Once downloaded, go to your browser preferences (about:preferences) and search "Cert", you should see the following:



Click View Certificates, then Authorities then Import. From here, go to where you downloaded Burps file (and select it). Select the both trust checkboxes (this is important otherwise it will not work) and then click ok. Like so:



Then click ok to navigate out of the Certificate Manager. You have now successfully installed Burps CA certificate allowing you to navigate to HTTPS sites.
Try and re-open https://google.com - Your TLS error will have gone away and you can intercept it with Burp!

#1
Set up Burp

No answer needed

[Task 3] Walk through the application

The first step to identifying vulnerabilities in a web application is *actually* using the web application. Use the web application like the actual user would:
• create an account
• click on the links you can see what the application does(and to identify an attack surface i.e. what parts of the application have functionality that you can attack)
• use the different functionality(e.g. making transactions)
Through this it is important to identify common themes such as:
• What languages/frameworks did the developer use to create the application.
• What version of the server/language did the developer use(if specified in the application)
During the walk through, it's important to think like a developer. During this process try and think of the design/-implementation of a particular feature, and using these features in a way that the developer did not intend for them to be used.

#1
Walk through the application and use the functionality available.

No answer needed

Javascript Frameworks:
Angular 7.1.1
Zone.js

Font Scripts:
Google Font API

Web Frameworks:
Express

Miscellaneous:
webpack

Web Severs:
Express

Programming Languages:
Node.js

Javascript Libraries:
Hammer.js

[Task 4] Injection

This section will focus on injection vulnerabilities. Injection vulnerabilities can cover a lot of different vulnerabilities including but not limited to:
• (no)SQL Injection - Depending on the database used, an attacker can enter a malicious or malformed query to either retrieve or tamper data from the database
• Command Injection - In applications that take user input or user controlled data and run them as system commands, a user may tamper with this data to execute their own system commands
In this case, it will just be basic SQL Injection

#1
Log in with the administrator's user account using SQL Injection

No answer needed

---

email found **admin@juice-sh.op**

Think of how SQL queries are written to check if a user existsx2F;has the right password. How would you break out of this query?
----------Create Fake Account----------
email: **test@xyz**
pass: **12345**
pass: **12345**
security question: **Company you first work for as an adult? CIA**
---
http://10.10.140.209/ftp/order_2ec7-2cf60f285da7950a.pdf found ftp path /home/ubuntu/juice-shop_8.2.0/ftp/order_2ec7-2cf60f285da7950a.pdf'

OWASP Juice Shop - Order Confirmation
Customer: test@xyz
Order #: 2ec7-2cf60f285da7950a
1x Apple Juice (1000ml) ea. 1.99 = 1.99
1x Banana Juice (1000ml) ea. 1.99 = 1.99
Total Price: 3.98
Thank you for your order!
---
on Contact Us link hit Complain-
ONLY PDFs can be uploaded

Logged in with:
Email = **' or 1=1--**
Password = **abcd**e

[Task 5] Broken Authentication

This task will involve looking at exploiting authentication through different logic flaws.
When we talk about logic flaws within authentication, we include:
• forgotten password mechanisms
• exploiting bugs in the authentication process

#1
reset Jim's password using the forgotten password mechanism - what was the answer to the secret question?

jim@juice-sh.op
samuel

#2
What is the administrator password?

admin123

[Task 6] Sensitive Data Exposure

When creating an application, it's important to store and transmit sensitive data carefully. In some cases, developer may not correctly protect sensitive data so it would be easy to gain access to personal information. In cases where this is done correctly, this protection isn't applied consistently. This task will involve identifying and extracting sensitive data from the application.

#1
Access a confidential document and enter the name of the first file with the extension ".md"

acquisitions.md
----------**found in ftp directory**----------
http://10.10.140.209/ftp/

[Task 7] Broken Access Control

Most systems are designed to be used with multiple users. Users can either have the same privileges, or users can have separate privileges(e.g. having administrators or moderators that have more actions/permissions on the website). Users should not be able to access data from other users(with the same or different privilege). Broken access control involves identifying different bugs or incorrect implementations that allow an attacker to access data that they should not have access to.

#1
Access the administration section of the store - What is the name of the page?

administration

#2
Access someone else's basket

No answer needed

#3
Get rid of all 5 star customer feedback

No answer needed

[Task 8] Cross Site Scripting(XSS)

XSS is a vulnerability that involves injecting malicious javascript in trusted websites. Once an attacker has injected malicious javascript,
sometimes a browser will not know whether to trust it, and it will run the script. Using this exploit an attacker can:
• steal session information through cookies
• arbitrarily redirect users to their own pages(for phishing)

There are a few different types of XSS attacks:
◇ Persistent/Non-Reflected - Here the XSS payload has been stored in the database, and once the server/-framework passes the data from
the database into the webpage, the script/payload is executed
◇ Non Persistent/Reflected - Here the XSS payload is usually crafted using a malicious link. It is not stored.
You can cause javascript to execute using different payloads and HTML tags- this is a good list of resources for payloads.

#1
Carry out reflected XSS using Tracking Orders

No answer needed

#2
Carry out XSS using the Search field?

No answer needed
