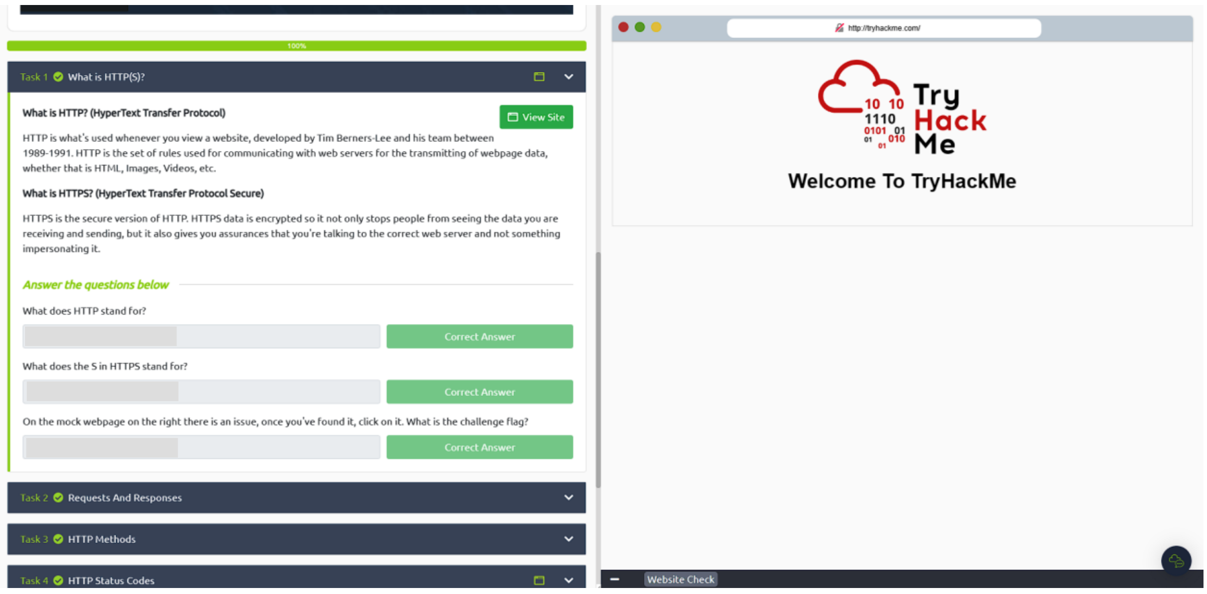
TryHackMe: HTTP in detail

As you start this room by TryHackMe (THM), you will start to learn all about HTTP and HTTPS protocols that we use in our every day lives. Whether you are browsing the internet on a mobile device or a computer, HTTP/HTTPS will be in play within your browser. This could be Microsoft Edge, Google, Safari or Firefox to name some of the most used browsers.

Task 1: What is HTTP(S)?
When you open the room to HTTP in detail, the site automatically loads in split view to display a TryHackMe.com webpage. If this does not happen, click on the **View Site** button in the top right of Task 1 to load the webpage.



What is HTTP? (HyperText Transfer Protocol)

HTTP is what's used whenever you view a website, developed by Tim Berners-Lee and his team between 1989-1991. HTTP is the set of rules used for communicating with web servers for the transmitting of webpage data, whether that is HTML, Images, Videos, etc.

What is HTTPS? (HyperText Transfer Protocol Secure)

HTTPS is the secure version of HTTP. HTTPS data is encrypted so it not only stops people from seeing the data you are receiving and sending, but it also gives you assurances that you're talking to the correct web server and not something impersonating it.

Answer the questions below

What does HTTP stand for?

What does the S in HTTPS stand for?

On the mock webpage on the right there is an issue, once you've found it, click on it. What is the challenge flag?

Read the section that tells you about HyperText Transfer Protocol (HTTP) and HyperText Transfer Protocol Secure (HTTPS), then answer the following questions:

1. What does HTTP stand for?

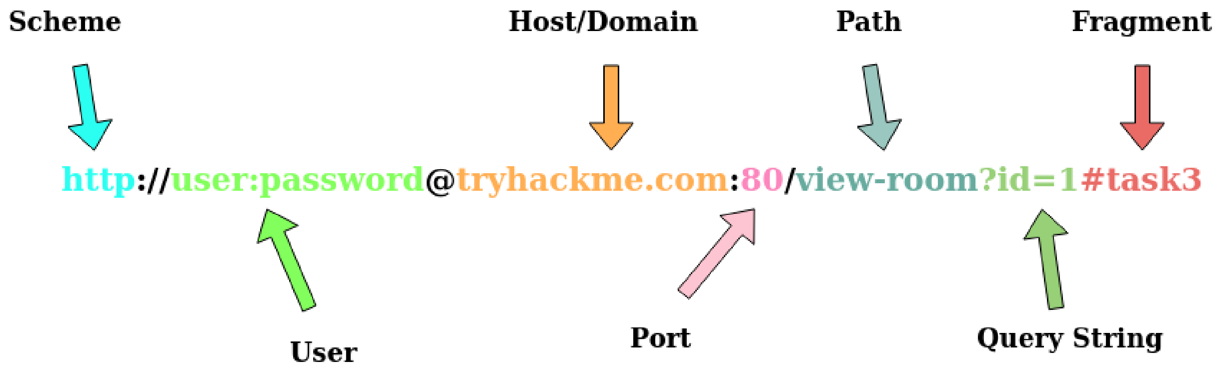
2. What dopes the S in HTTPS stand for?

3. On the mock webpage on the right there is an issue, once you've found it, click on it. What is the challenge flag?

The first two answers can be found by reading through the information at the top of this task. The final answer for this task is the challenge flag which can be found in the webpage. If you look carefully, you can see that the page is not https. If you click on the padlock, you will reveal the challenge flag.

Once you have completed this task, you no longer need the Website Check webpage open so you can close the window by clicking on the **–** sign at the bottom left of the webpage.

Task 2: Requests and Responses
This task will take you through how the web address or Uniform Resource Locator (URL) is made up. You will learn what each part does for the URL. Read through the information provided in this task and answer the questions at the bottom.



Scheme: This instructs on what protocol to use for accessing the resource such as HTTP, HTTPS, FTP (File Transfer Protocol).

User: Some services require authentication to log in, you can put a username and password into the URL to log in.

Host: The domain name or IP address of the server you wish to access.

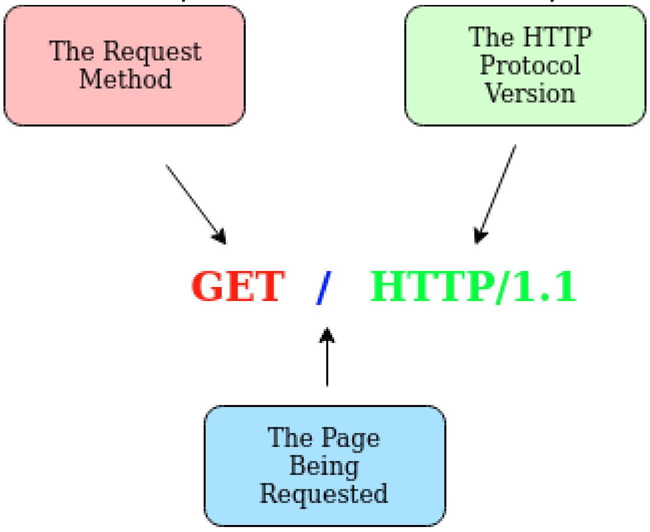
Port: The Port that you are going to connect to, usually 80 for HTTP and 443 for HTTPS, but this can be hosted on any port between 1 - 65535.

Path: The file name or location of the resource you are trying to access.

Query String: Extra bits of information that can be sent to the requested path. For example, /blog?id=1 would tell the blog path that you wish to receive the blog article with the id of 1.

Fragment: This is a reference to a location on the actual page requested. This is commonly used for pages with long content and can have a certain part of the page directly linked to it, so it is viewable to the user as soon as they access the page.

The second part of the task will take you through making a request to a webpage.



Example Request:

```
GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Referer: https://tryhackme.com/
```

To breakdown each line of this request:

Line 1: This request is sending the GET method (more on this in the HTTP Methods task), request the home page with / and telling the web server we are using HTTP protocol version 1.1.

Line 2: We tell the web server we want the website tryhackme.com

Line 3: We tell the web server we are using the Firefox version 87 Browser

Line 4: We are telling the web server that the web page that referred us to this one is https://tryhackme.com

Line 5: HTTP requests always end with a blank line to inform the web server that the request has finished.

Example Response:

```
HTTP/1.1 200 OK
Server: nginx/1.15.8
Date: Fri, 09 Apr 2021 13:34:03 GMT
Content-Type: text/html
Content-Length: 98

<html>
<head>
    <title>TryHackMe</title>
</head>
<body>
    Welcome To TryHackMe.com
</body>
</html>
```

To breakdown each line of the response:

Line 1: HTTP 1.1 is the version of the HTTP protocol the server is using and then followed by the HTTP Status Code in this case "200 Ok" which tells us the request has completed successfully.

Line 2: This tells us the web server software and version number.

Line 3: The current date, time and timezone of the web server.

Line 4: The Content-Type header tells the client what sort of information is going to be sent, such as HTML, images, videos, pdf, XML.

Line 5: Content-Length tells the client how long the response is, this way we can confirm no data is missing.

Line 6: HTTP response contains a blank line to confirm the end of the HTTP response.

Lines 7-14: The information that has been requested, in this instance the homepage.

Once you have read through this and gone through the examples as shown above, have a go at the questions below:

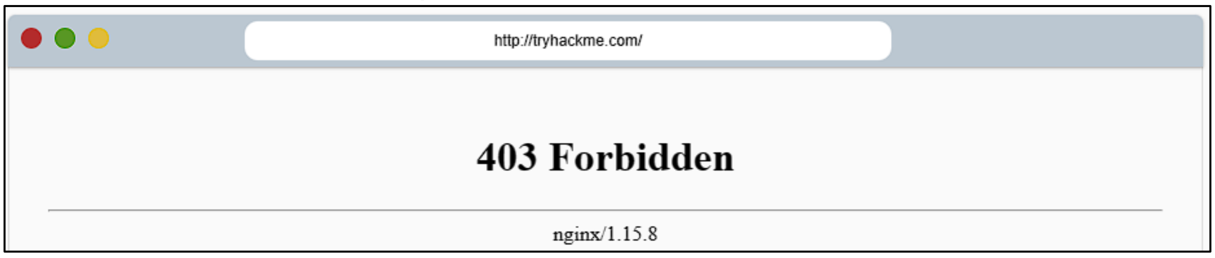
1. What HTTP protocol is being used in the above example?

2. What response header tells the browser how much data to expect?

Task 3: HTTP Methods
Task 3 talks about the methods used in HTTP. You will briefly cover the **GET, POST, PUT** and **DELETE** request methods. Then answer the following questions:

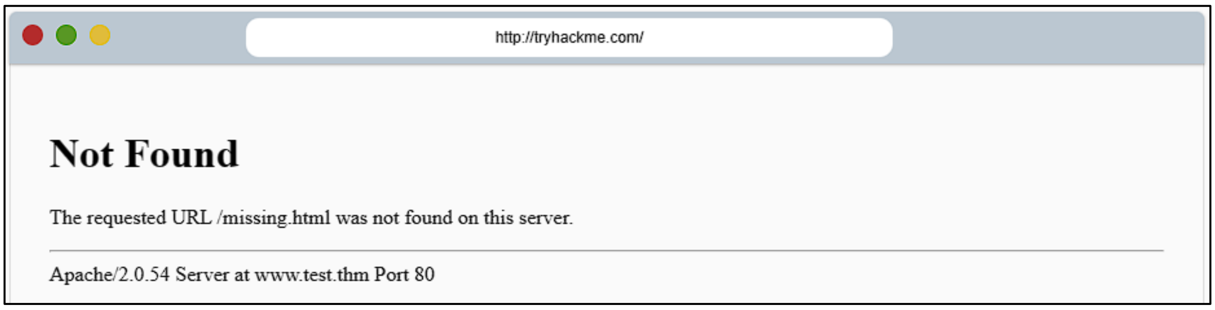
1. What method would be used to create a new user account?
2. What method would be used to update your email address?
3. What method would be used to remove a picture you've uploaded to your account?
4. What method would be used to view a news article?

Task 4: HTTP Status Codes
Task 4 will take you through the HTTP Codes that you may encounter. We have all seen the 403 Error at some point.


http://tryhackme.com/

403 Forbidden

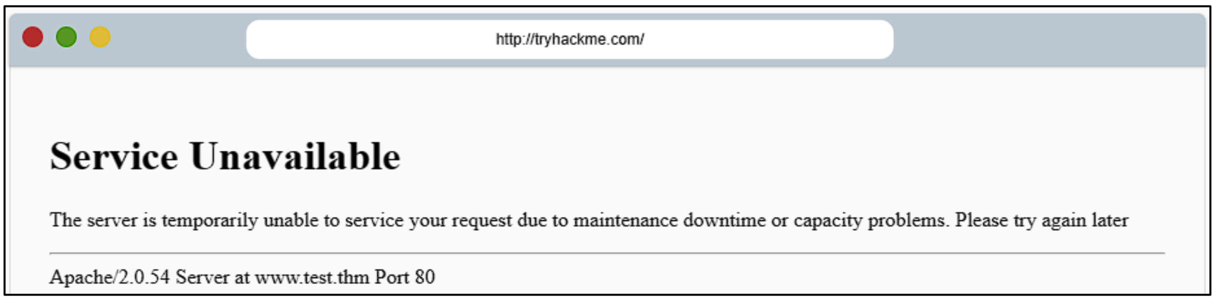
nginx/1.15.8


http://tryhackme.com/

Not Found

The requested URL /missing.html was not found on this server.

Apache/2.0.54 Server at www.test.thm Port 80


http://tryhackme.com/

Service Unavailable

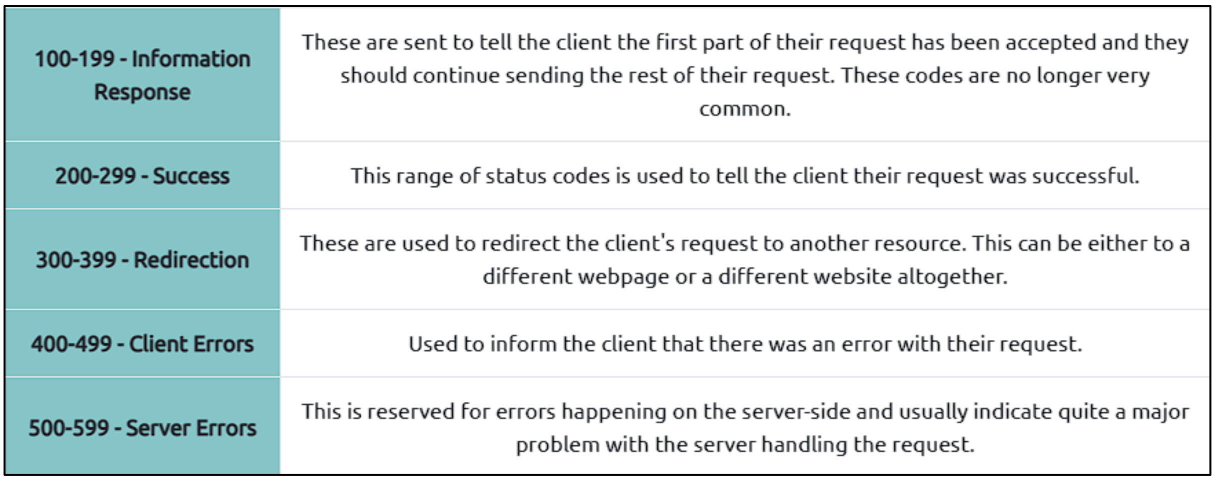
The server is temporarily unable to service your request due to maintenance downtime or capacity problems. Please try again later

Apache/2.0.54 Server at www.test.thm Port 80

But there are other codes we may not see as often and can sometimes through you off if you do see them. THM will tell you what the codes mean should you come across them.

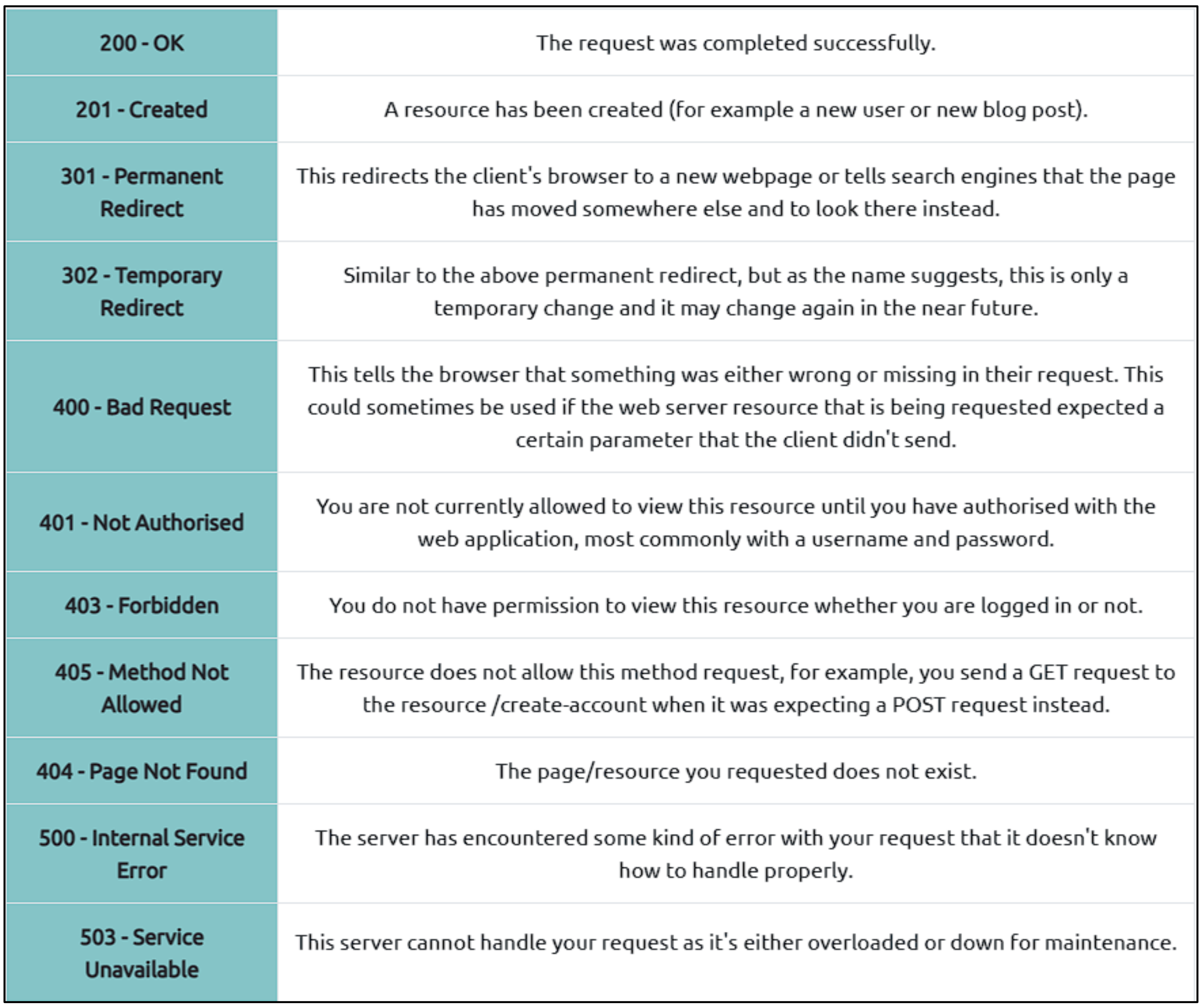
Read through the information about these codes and their meanings.

HTTP status codes and their meanings:



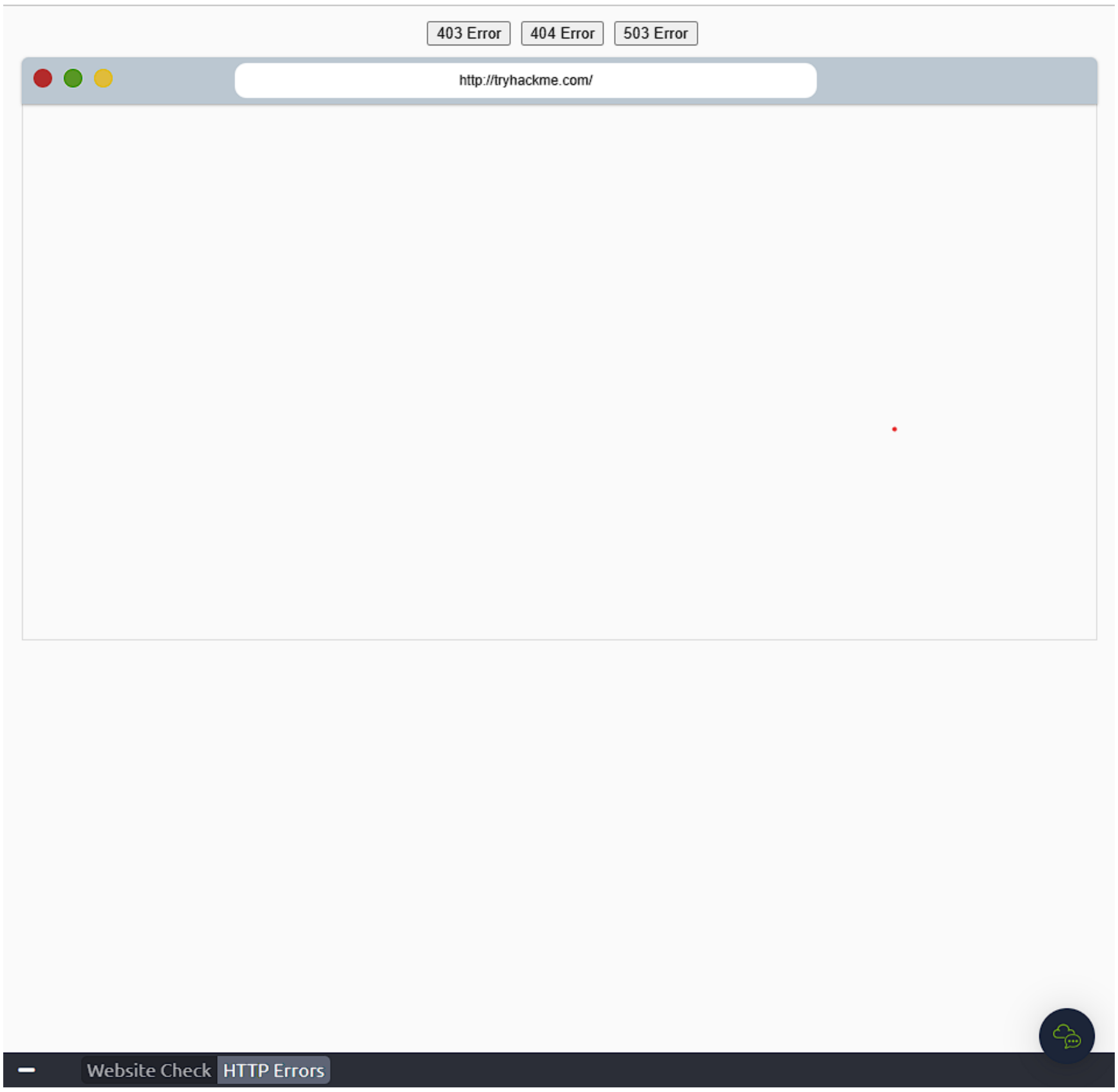
100-199 - Information Response	These are sent to tell the client the first part of their request has been accepted and they should continue sending the rest of their request. These codes are no longer very common.
200-299 - Success	This range of status codes is used to tell the client their request was successful.
300-399 - Redirection	These are used to redirect the client's request to another resource. This can be either to a different webpage or a different website altogether.
400-499 - Client Errors	Used to inform the client that there was an error with their request.
500-599 - Server Errors	This is reserved for errors happening on the server-side and usually indicate quite a major problem with the server handling the request.

Common HTTP status codes we may come across when we are browsing the internet:



200 - OK	The request was completed successfully.
201 - Created	A resource has been created (for example a new user or new blog post).
301 - Permanent Redirect	This redirects the client's browser to a new webpage or tells search engines that the page has moved somewhere else and to look there instead.
302 - Temporary Redirect	Similar to the above permanent redirect, but as the name suggests, this is only a temporary change and it may change again in the near future.
400 - Bad Request	This tells the browser that something was either wrong or missing in their request. This could sometimes be used if the web server resource that is being requested expected a certain parameter that the client didn't send.
401 - Not Authorised	You are not currently allowed to view this resource until you have authorised with the web application, most commonly with a username and password.
403 - Forbidden	You do not have permission to view this resource whether you are logged in or not.
405 - Method Not Allowed	The resource does not allow this method request, for example, you send a GET request to the resource /create-account when it was expecting a POST request instead.
404 - Page Not Found	The page/resource you requested does not exist.
500 - Internal Service Error	The server has encountered some kind of error with your request that it doesn't know how to handle properly.
503 - Service Unavailable	This server cannot handle your request as it's either overloaded or down for maintenance.

When you get to the bottom of the code table, click on the **View Site** button at the top of Task 4 to load a new webpage which should look like below:



Take a look at the top of the webpage. You will see there are some of the common codes we see! Click on each one to see what they are for, then answer the following questions:

1. What response code might you receive if you've created a new user or blog post?

2. What response code might you receive if you've tried to access a page that doesn't exist?

3. What response code might you receive if the web server cannot access its database and the application crashes?

4. What response code might you receive if you try to edit your profile without logging in first?

You no longer need the HTTP Errors webpage open for the next task so you can close the window by clicking on the **–** sign at the bottom left of the webpage.

Task 5: Headers
This task will tell you more about the Headers. Headers are additional bits of data you can send to the web server when making requests.

Read through the information provided here about the different headers and the information they provide. Then answer the following questions:

1. What header tells the web server what browser is being used?

2. What header tells the browser what type of data is being returned?

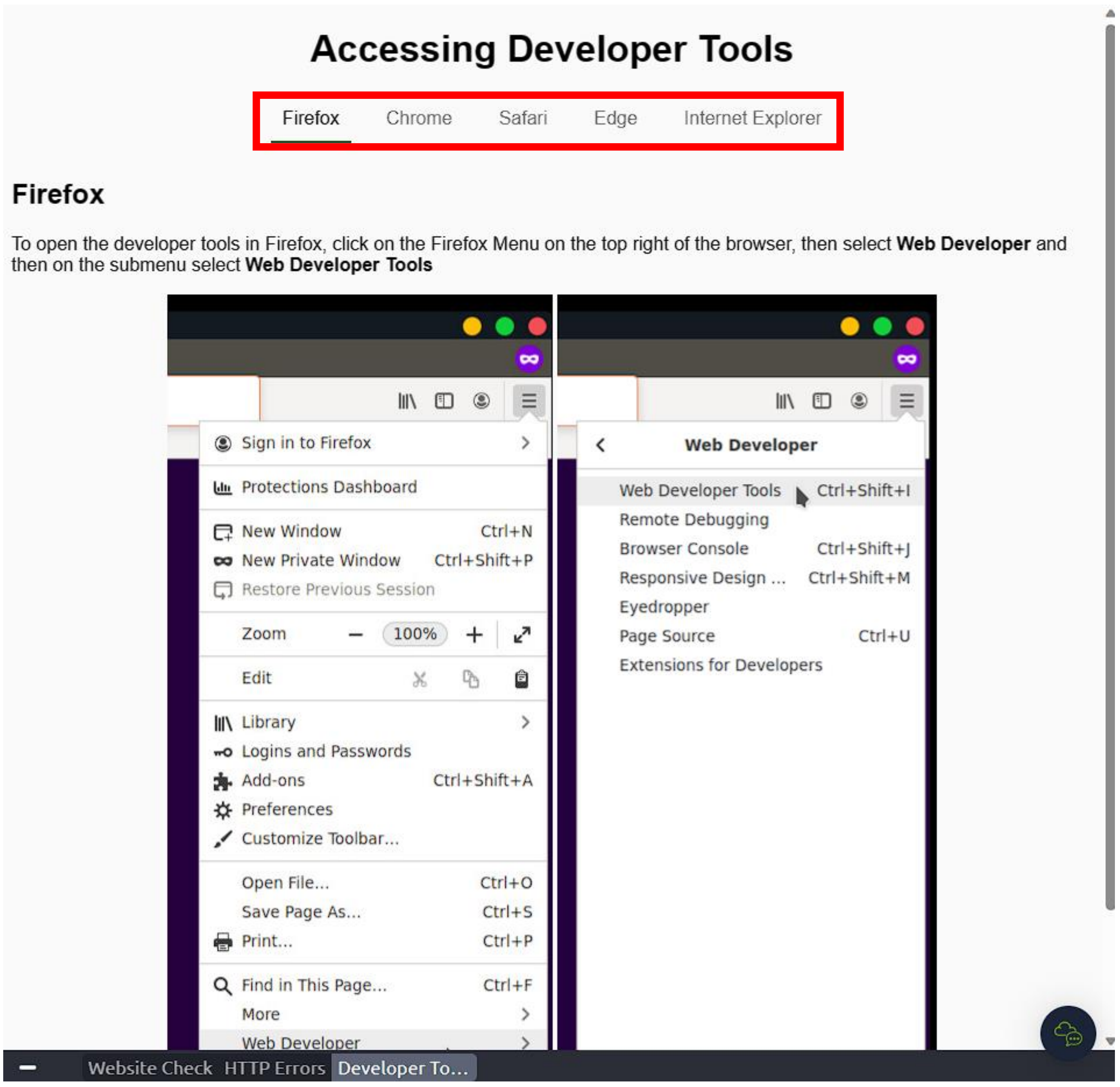
3. What header tells the web server which website is being requested?

Task 6: Cookies
In this task we will cover what cookies are (and I don't mean the delicious ones we all want to eat). Every time you visit a website, cookies will be saved to your device/computer. If it's the first time you have visited the website, or you have cleared down your cache and cookies from your settings, you may see a pop-up ask you to accept cookies or warn you that the site uses cookies and gives you options to accept or decline the usage of cookies.

Click on **View Site** and read through the information provided on THM about cookies and view the website that has been loaded in the split view. Click on each tab and read through the information.

Click on:

Firefox
Chrome
Safari
Edge
Internet Explorer (this browser has been deprecated but is still in use by some older systems – those systems would be running Operating Systems such as Windows XP, Vista and Windows 7. Although you may not see these systems in use, some companies/organisations may still use them)


Accessing Developer Tools

Firefox Chrome Safari Edge Internet Explorer

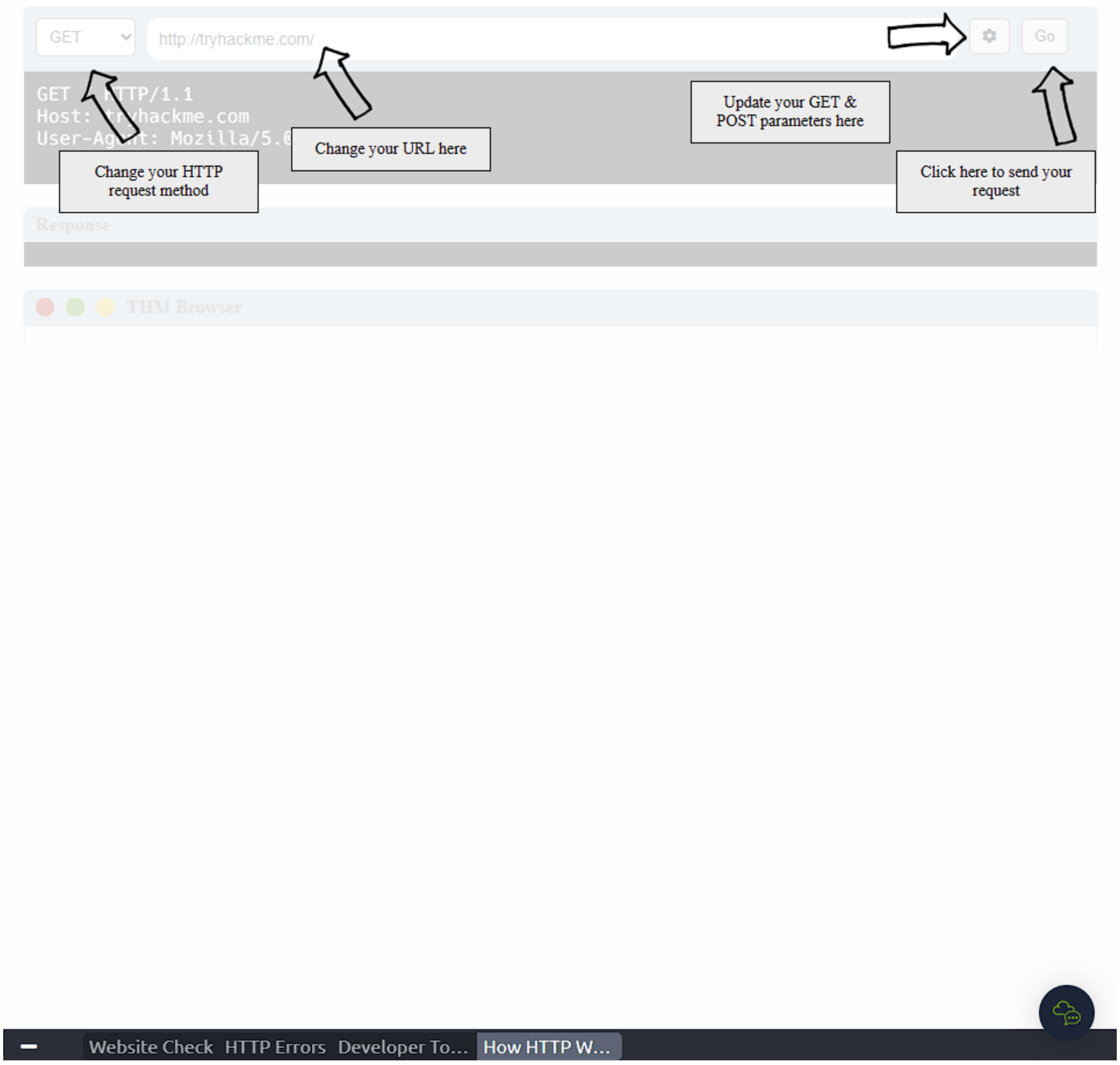
Firefox

To open the developer tools in Firefox, click on the Firefox Menu on the top right of the browser, then select **Web Developer** and then on the submenu select **Web Developer Tools**

Once you have read all the information on cookies and clicked on each tab in the Developer Tools window, answer the following question:

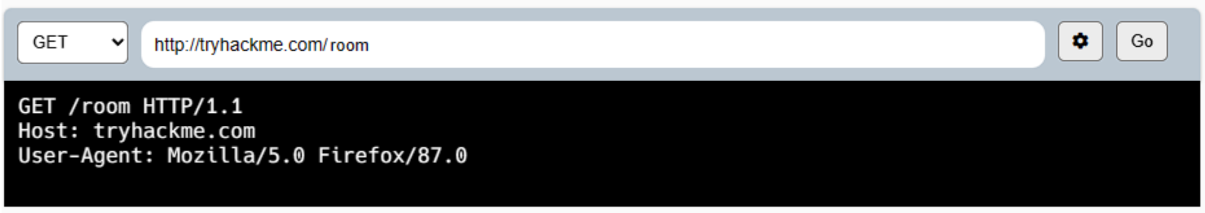
1. Which header is used to save cookies to your computer?

Task 7: Making Requests
You're almost done. This is the final task in the **HTTP in detail** room. Click on the **View Site** and use it to answer the following questions to capture 5 challenge flags:



```
GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0
```

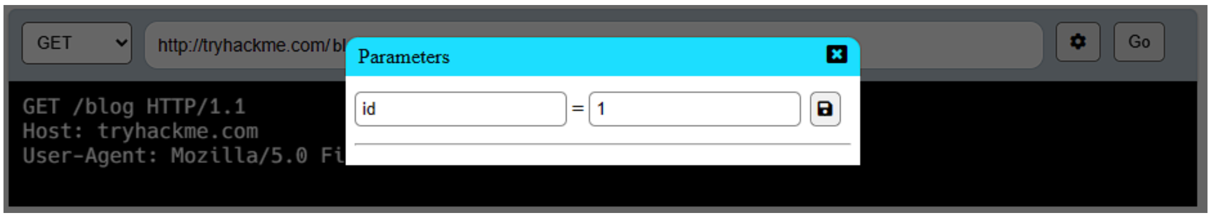
1. Make a GET request to /room.



```
GET /room HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
```

Type room at the end of the URL and click Go, then insert Flag which can be seen on the screen.

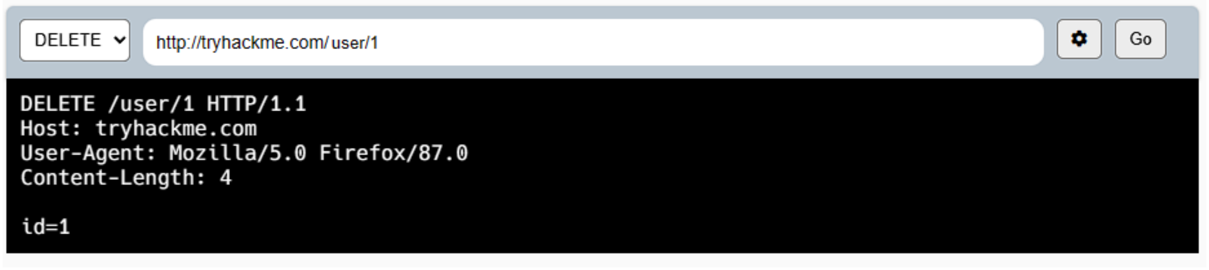
2. Make a GET request to /blog and using the gear icon set the id parameter to 1 in the URL field.



```
GET /blog HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Fi
```

Type blog at the end of the URL and then click on the cog. Enter id and 1 as shown and click the save button, then click Go. Insert Flag which can be seen on the screen.

3. Make a DELETE request to /user/1

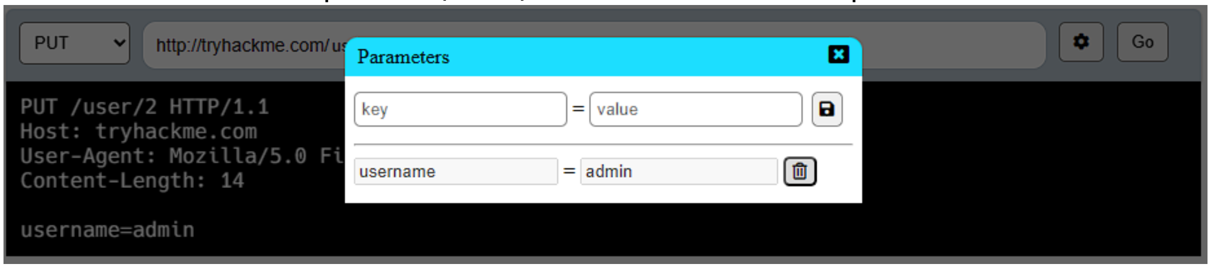


```
DELETE /user/1 HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 4

id=1
```

Select DELETE from the dropdown menu. Type user/1 at the end of the URL, then click Go. Insert Flag which can be seen on the screen.

4. Make a PUT request to /user/2 with the username parameter set to admin



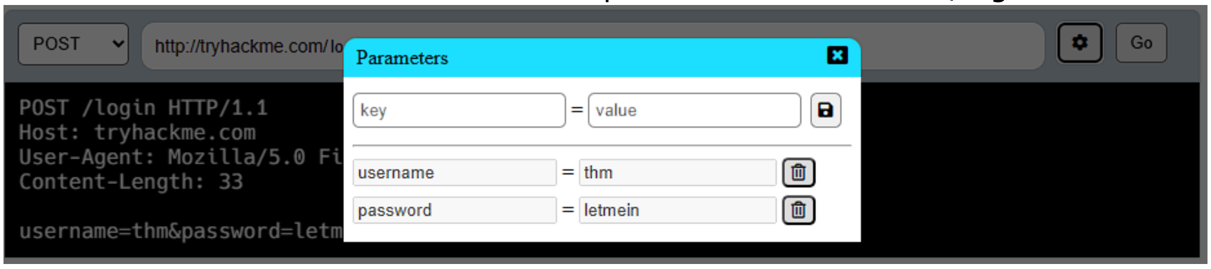
username = admin

```
PUT /user/2 HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Fi
Content-Length: 14

username=admin
```

Select PUT from the dropdown menu, enter user/2 at the end of the URL. Click the cog and enter username and admin and click save, then click Go to view the flag. Insert Flag which can be seen on the screen.

5. POST the username of thm and a password of letmein to /login



username = thm
password = letmein

```
POST /login HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Fi
Content-Length: 33

username=thm&password=letm
```

Select POST from the dropdown menu, enter login at the end of the URL. Click the cog and enter username and thm and click save, repeat for password and letmein and click save. Then click Go to view the flag. Insert Flag which can be seen on the screen.

CONGRATULATIONS, you have successfully completed the HTTP in detail room on TryHackMe.com.

Written by: Alan Rowe
Username: Guzz747
https://tryhackme.com/p/Guzz747