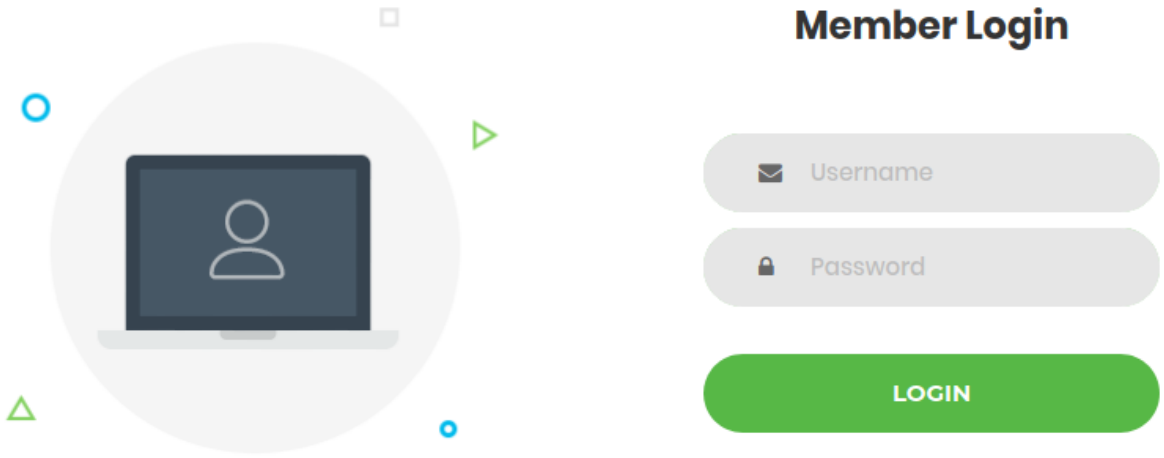
Baby Auth

Analysis

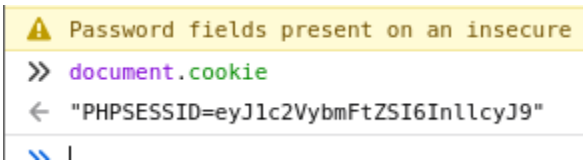
We are first greeted by a login page. Let's, once again, try `admin` with password `admin`:
Looks like we'll have to create an account - let's try those credentials.
This is great, because now we know we need a user called `admin`. Let's create another user - I'll use username and password `yes`, because I doubt that'll be used.



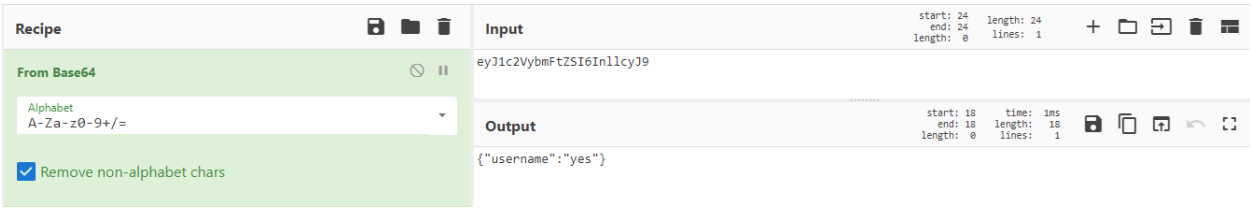
We're redirected to the login, which makes it seem like it worked. Let's log in with the credentials we just created:

we're not an admin!

When it comes to accounts, one very common thing to check is **cookies**. Cookies allow, among other things, for users to authenticate without logging in every time. To check cookies, we can right-click and hit **Inspect Element** and then move to the **Console** tab and type `document.cookie`.



Well, we have a cookie called `PHPSESSID` and the value `eyJ1c2VybmFtZSI6InllcyJ9`. Cookies are often base64 encoded, so we'll use a tool called CyberChef to decode it. Once we decode the base64, we see that the contents are simply `{"username":"yes"}`.



Exploitation

So, the website knows our identity due to our cookie - but what's to stop us from forging a cookie? Since we control the cookies we send, we can just edit them. Let's create a fake cookie!

Note that we're URL encoding it as it ends in the special character `=`, which usually has to be URL encoded in cookies. Let's change our cookie to `eyJ1c2VybmFtZSI6ImFkbWluIn0%3D`!

```
» document.cookie = "PHPSESSID=eyJ1c2VybmFtZSI6ImFkbWluIn0%3D"
⚠ Cookie “PHPSESSID” will be soon rejected because it has the “SameSite” attribute
  /SameSite
← "PHPSESSID=eyJ1c2VybmFtZSI6ImFkbWluIn0%3D"
```

Ignore the warning, but we've now set `document.cookie`. Refresh the page to let it send the cookies again.

And there you go - we successfully authenticated as an admin!
`HTB{s3ss10n_1nt3grity_1s_0v3r4tt3d_4nyw4ys}`