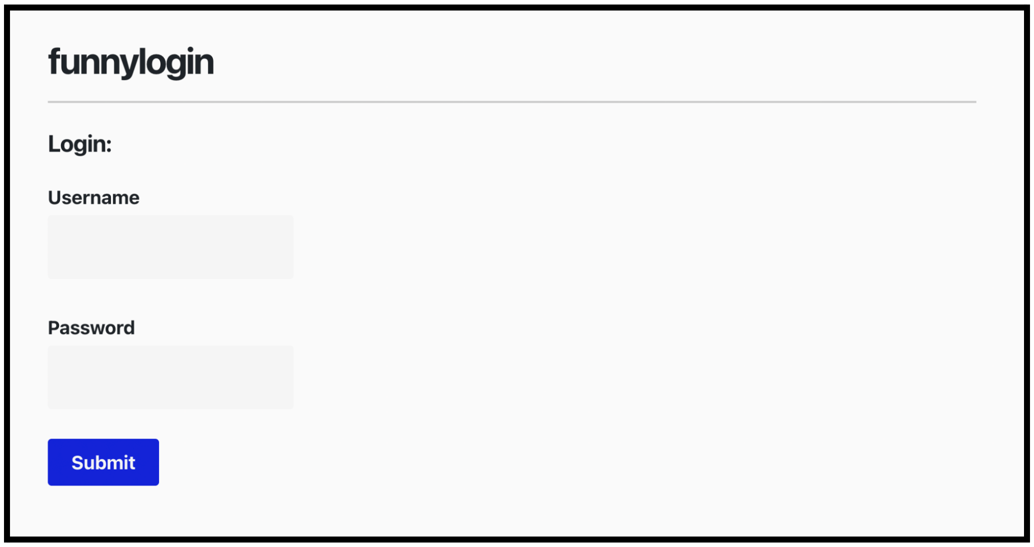
Website Interface:



Getting Started

So we have a typical login setup here where the user inputs a username and a password. Let's jump to the source code and take a look at what they are checking for.

On lines 29-30, we can see the SQL query being sent to the web server where the user inputs two values – the username (user) and password (pass) – with no string sanitization

```
const { user, pass } = req.body;
const query = `SELECT id FROM users WHERE username = '${user}' AND password = '${pass}';`;
```

This suggests a possible SQLi attack.

Further down, on lines 37-38 we can see that to get the flag, we need to fulfill 2 conditions. 1) users[id] = True and 2) isAdmin[user] = True.

```
if (users[id] && isAdmin[user]) {
    return res.redirect("/?flag=" + encodeURIComponent(FLAG));
}
```

Let's delve deeper into what these 2 conditions mean.

Solving the first condition: users[id] = True

To make users[id] true, we need to make sure that id returned from the SQL query is a value in [0, 100,000] because of how the database is initially filled in:

```
const users = [...Array(100_000)].map(() => ({ user: `user-${crypto.randomUUID()}`, pass:
crypto.randomBytes(8).toString("hex") }));
db.exec(`INSERT INTO users (id, username, password) VALUES ${users.map((u,i) => `(${i}, '${u.user}',
'${u.pass}')`).join(", ")}`);
```

To do this, we can write an SQLi attack and input malicious code in the either variable so that the query variable is set like this:

```
SELECT id FROM users WHERE username = '${user}' AND password = '' UNION SELECT 1;--
```

This will return id as 1 which is part of [0, 100,000]. Thus, we have fulfilled the first part of the condition (setting users[id]=True).

Solving the second condition: isAdmin[user] = True

To set isAdmin[user] = True, we need to make sure that the user is the administrator. This is rather difficult at first glance because the administrator is chosen at random from 100,000 users on lines 20-22.

```
const isAdmin = {};
const newAdmin = users[Math.floor(Math.random() * users.length)];
isAdmin[newAdmin.user] = true;
```

Thus, we cannot guess which user was given Administrator access very easily. However, there is a work around which comes from one of JavaScript's many interesting properties.

__proto__ is a property that allows you to access and modify an object's prototype. Most objects in JavaScript have a __proto__ field – (any objects created with Object.create(null) does not initialize with [Prototype]). Thus, isAdmin[__proto__] will return True.