SQL INJECTION

SEMINAR PRESENTATION

PRANITH RAO
4SO18CS088

WHAT ARE THOSE TERMS?

- **SQL:** It stands for Structured Query Language and it is used to retrieve and manipulate data from structured databases.
- **SQL Injection Attack:** In these type of attacks SQL statements are injected into the vulnerable spots with a malicious intention.

ATTACK INTENTS

01 DATA EXTRACTION

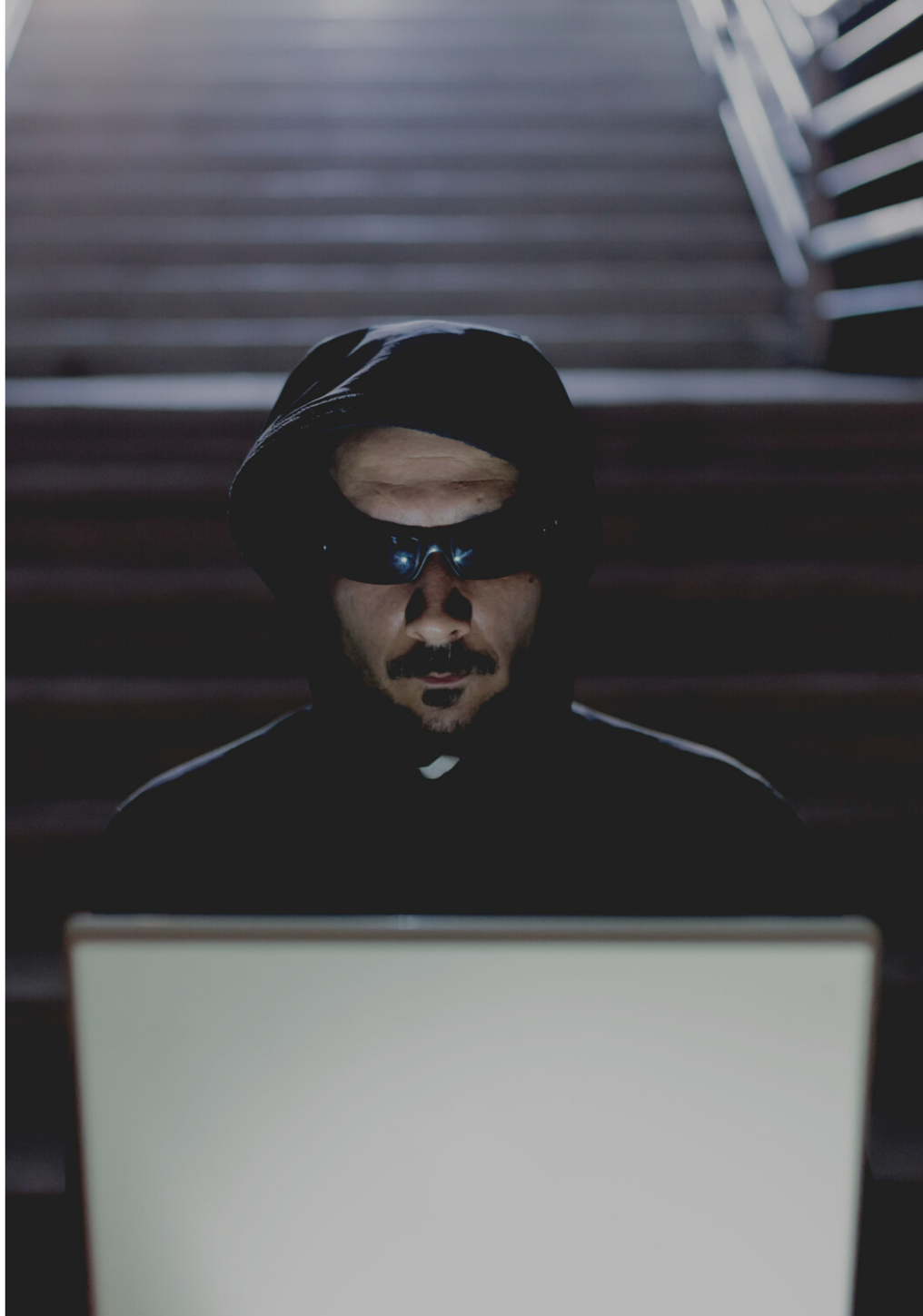
02 ADDING OR MODIFYING DATA

03 FINGERPRINT THE DATABASE

04 PERFORMING DENIAL OF SERVICE ATTACK

05 BYPASSING AUTHENTICATION

06 DATABASE SCHEMA IDENTIFICATION



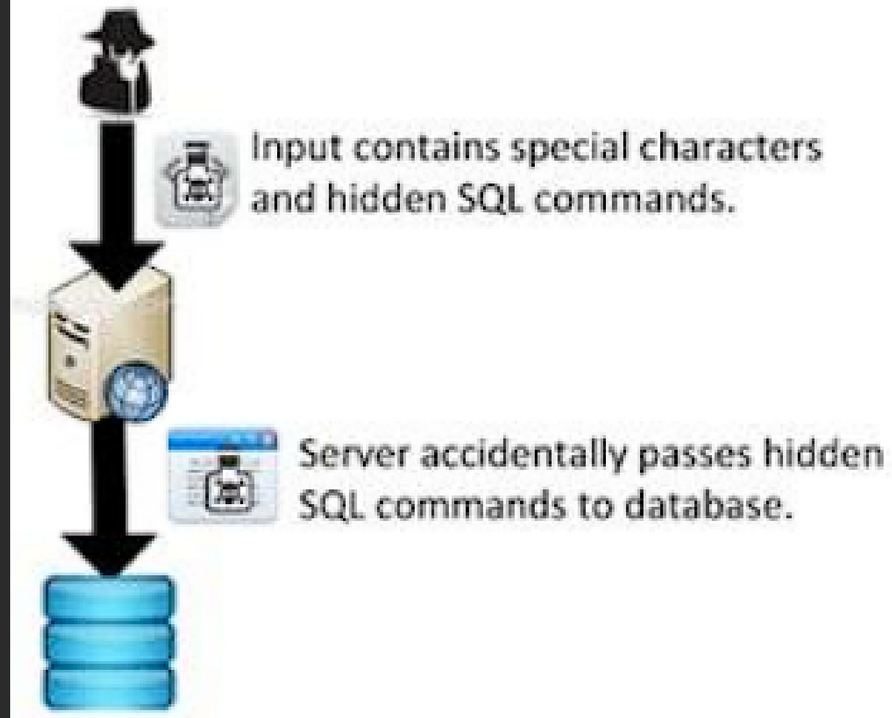


ILLUSTRATION OF SQL INJECTION

User Input

Lack of proper input validation

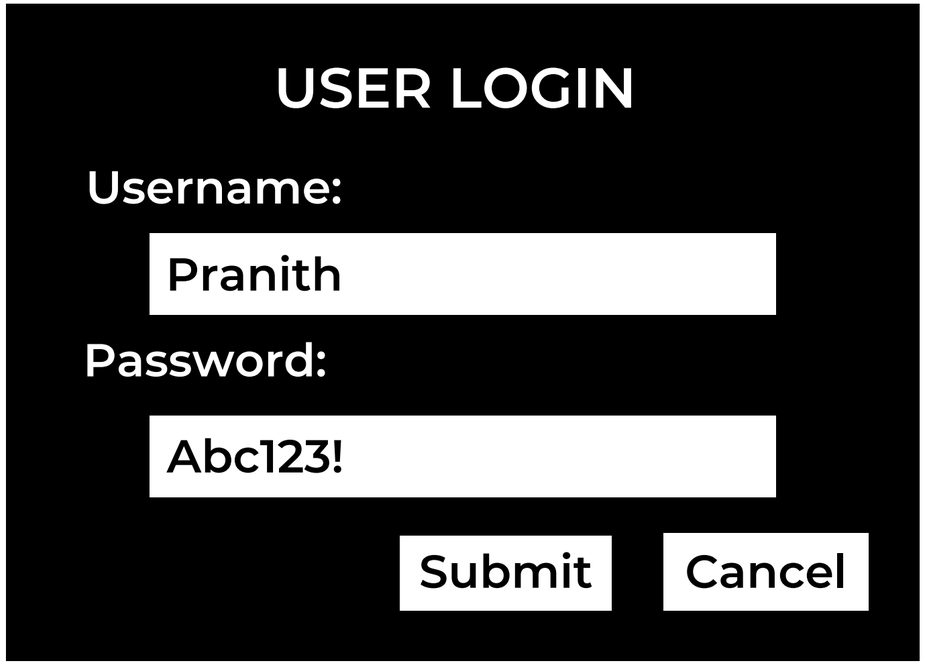
Backend processing

Access to database

DANGEROUS CHARACTERS

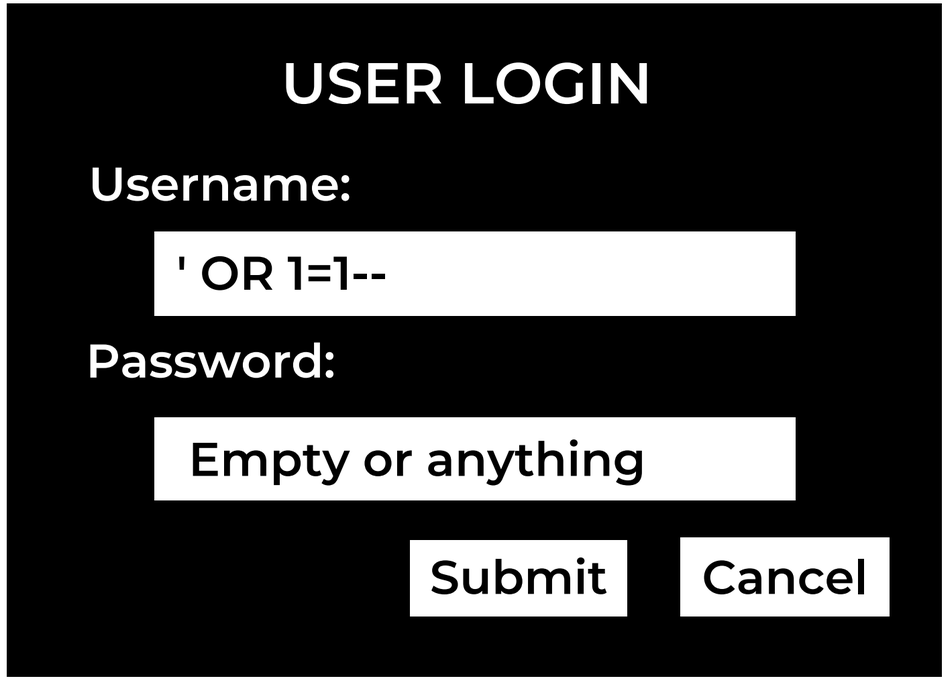
Character	Meaning
;	Query Delimiter.
'	Character Data String Delimiter
--	Single Line Comment.

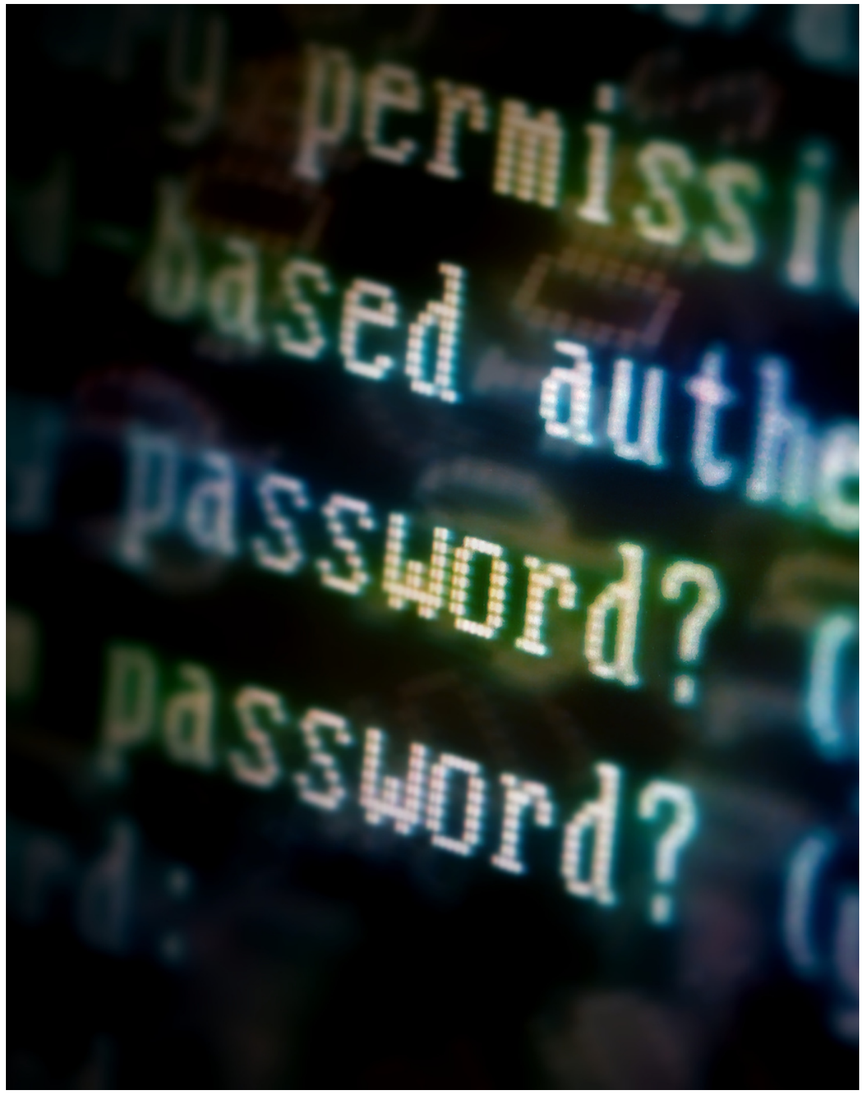
WORKING OF SQL INJECTION



SELECT * FROM people WHERE name = ‘Pranith’ AND password = ‘Abc123!’

SELECT * FROM people WHERE name = ‘’ OR 1=1--’ AND password = ‘ ’;





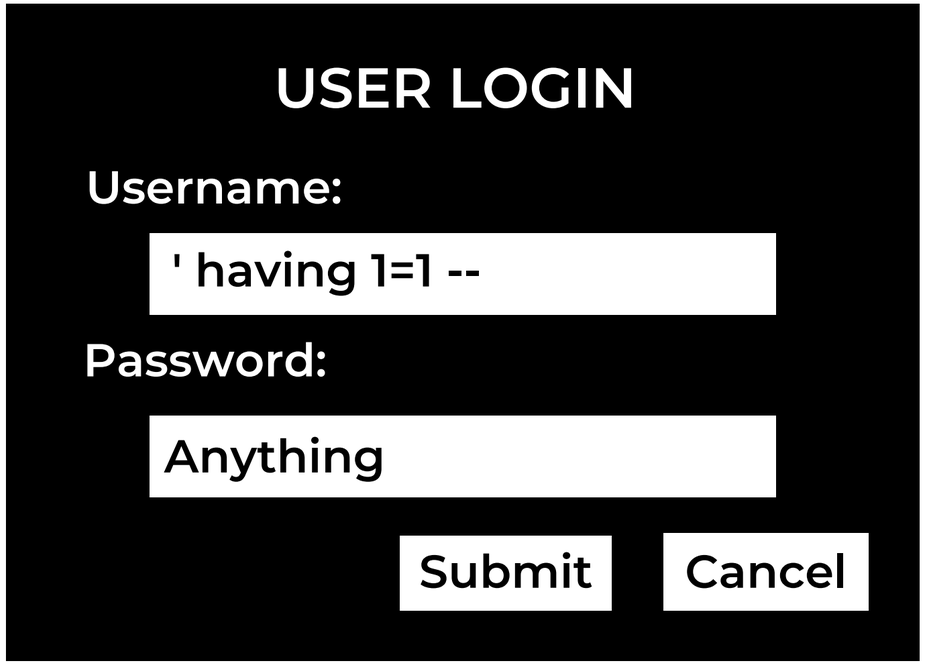
BYPASSING AUTHENTICATION

SELECT * FROM people WHERE name = ‘’ OR 1=1--’ AND password = ‘ ’;

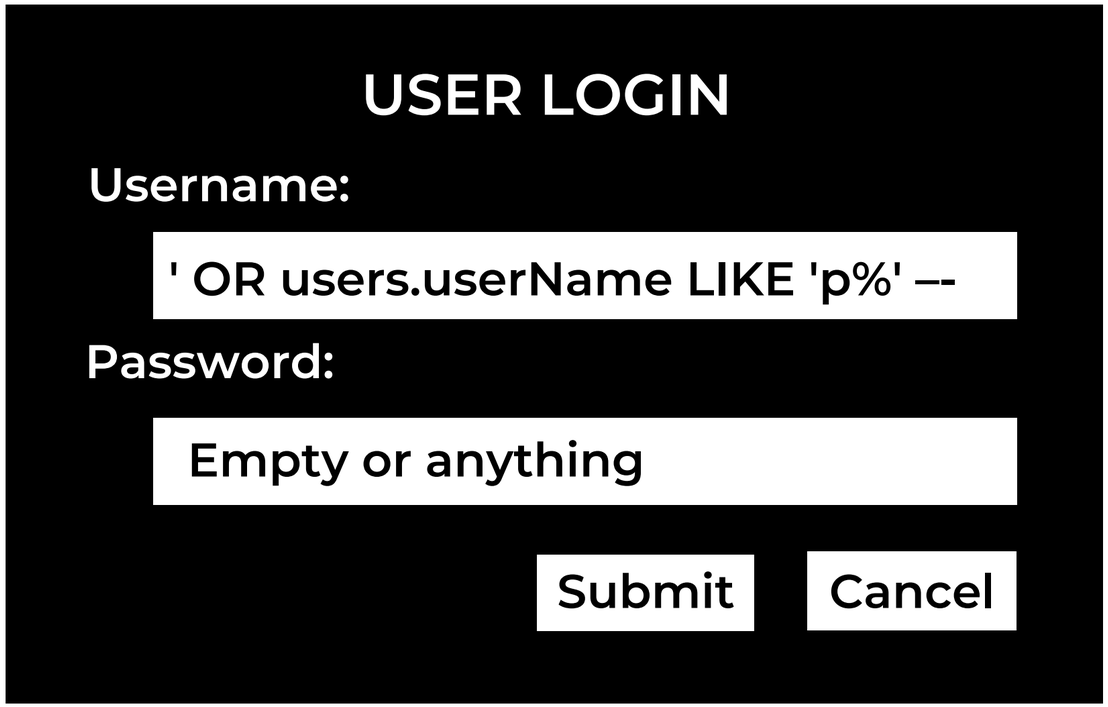
It Works as follows:

- ‘: Closes the user input field.
- OR: Continues the SQL query so that the process executes what comes before OR and what comes after.
- 1=1: A statement which is always true.
- --: Comments outs the rest of the lines so that it won’t be processed.

OBTAINING COLUMN NAME

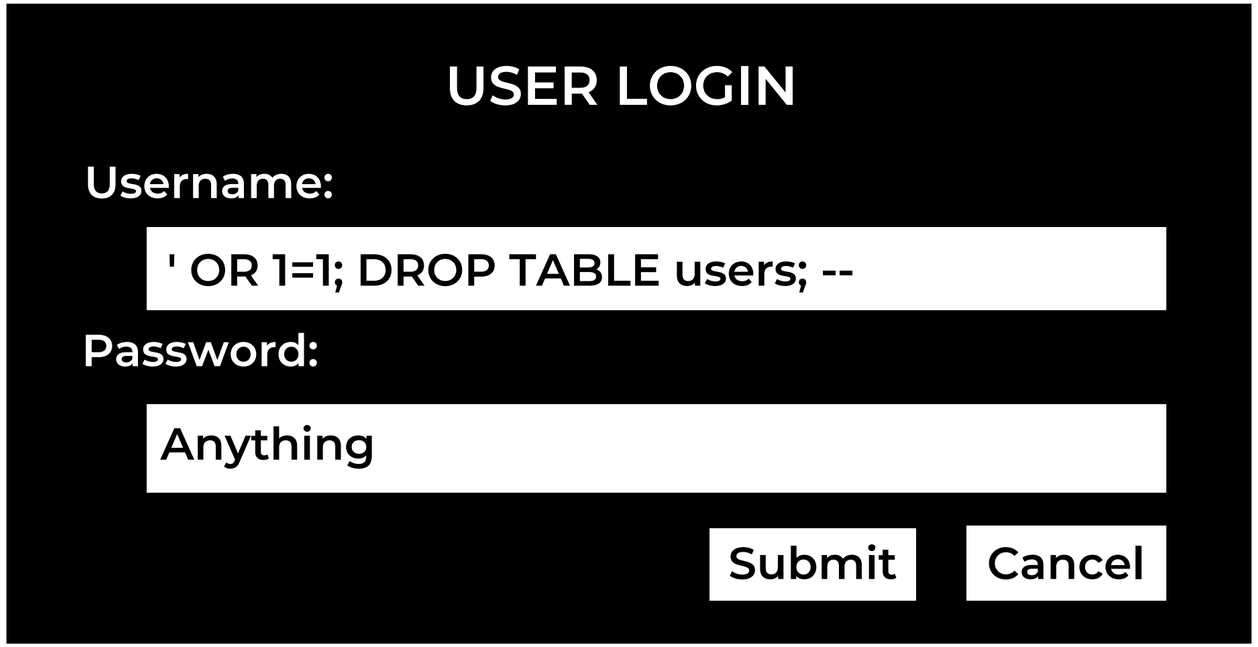


“Column 'users.userName' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause. /filename, line 16”



The query grabs the personName field of the first row whose personName field starts with ‘p’.

DROPPING A TABLE



A semicolon (;) is used to delimit a query in certain databases.
The query:

- First choses the userName field for all users database entries
- Then it would erase the users table

When it's done whenever the new user tries to login, it returns a error message as
Invalid object name 'users'. /filename, line 16

TYPES OF ATTACK

On Basis of Execution Nature of Injection

a. First Order Attacks
b. Second Order Attacks

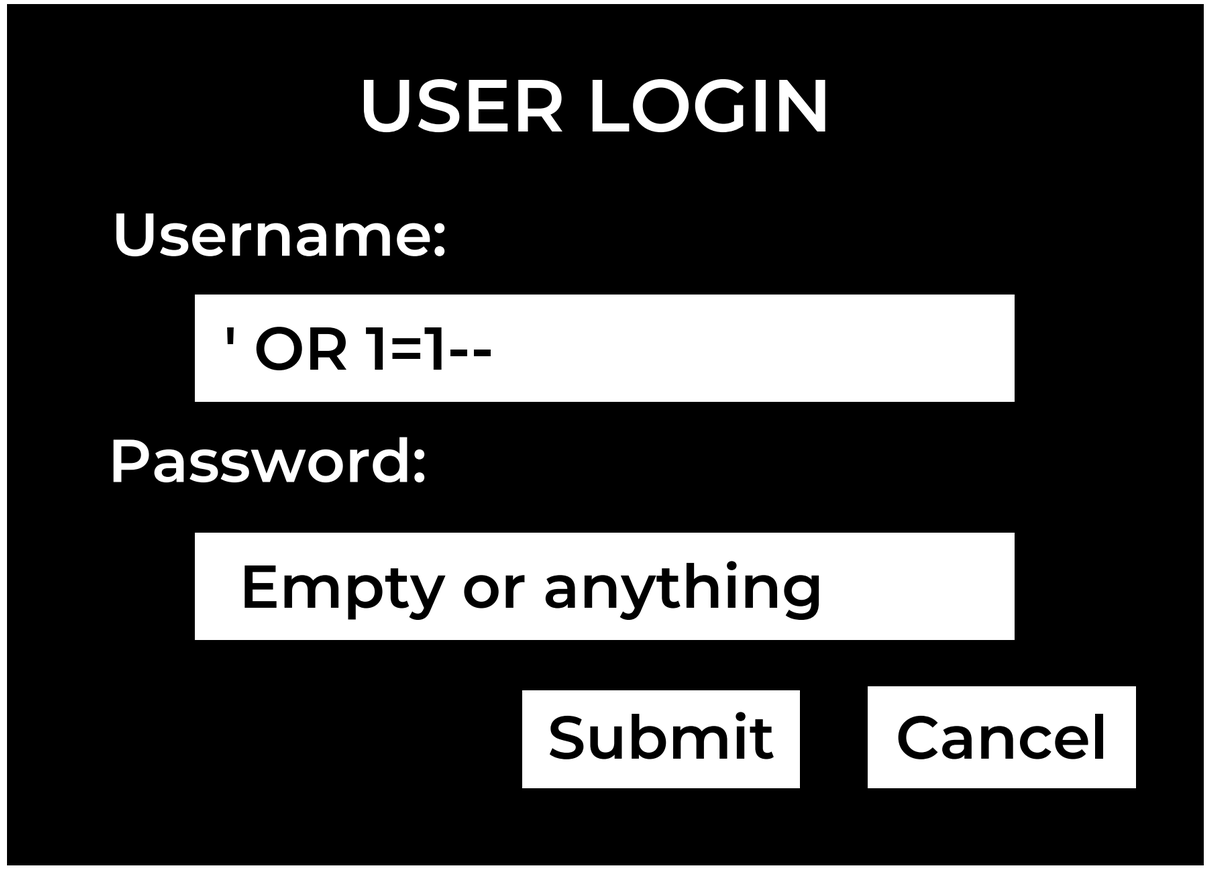
On Basis of Goal or Purpose

a. Tautologies
b. Illegal/Logically Incorrect Queries
c. Union Query
d. Piggy-backed Queries

FIRST ORDER ATTACK

- Single query
- Results obtained immediately

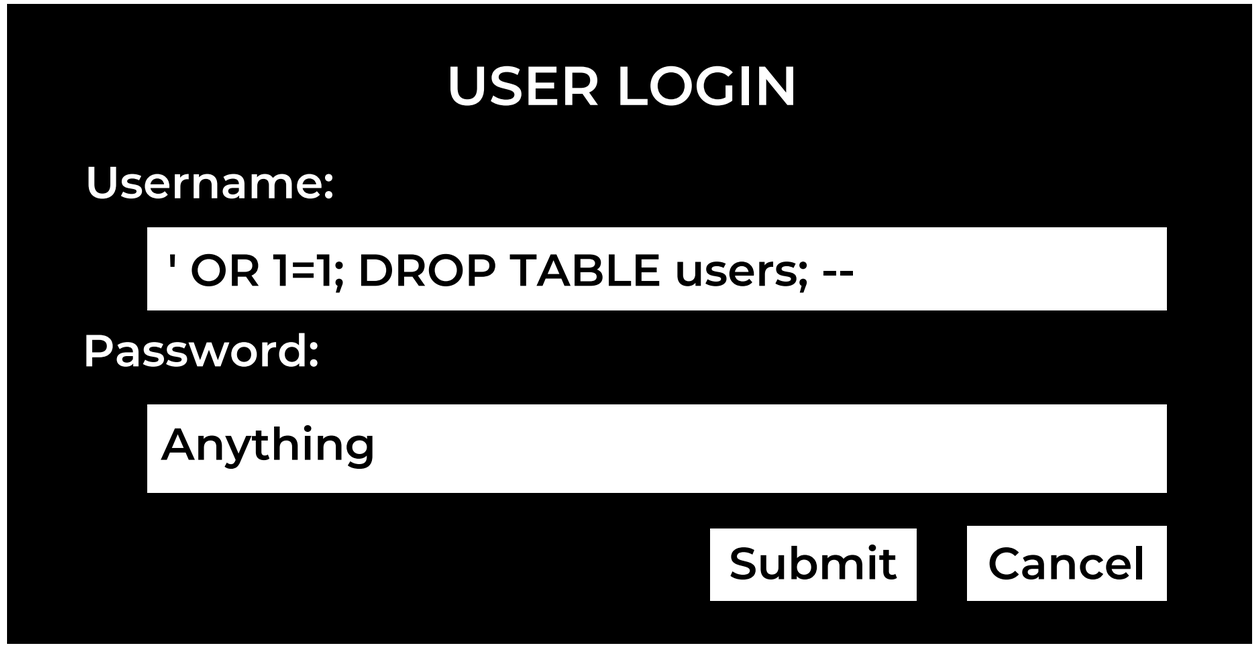
- Purposes:
  Bypass Authentication
  Obtain Column name



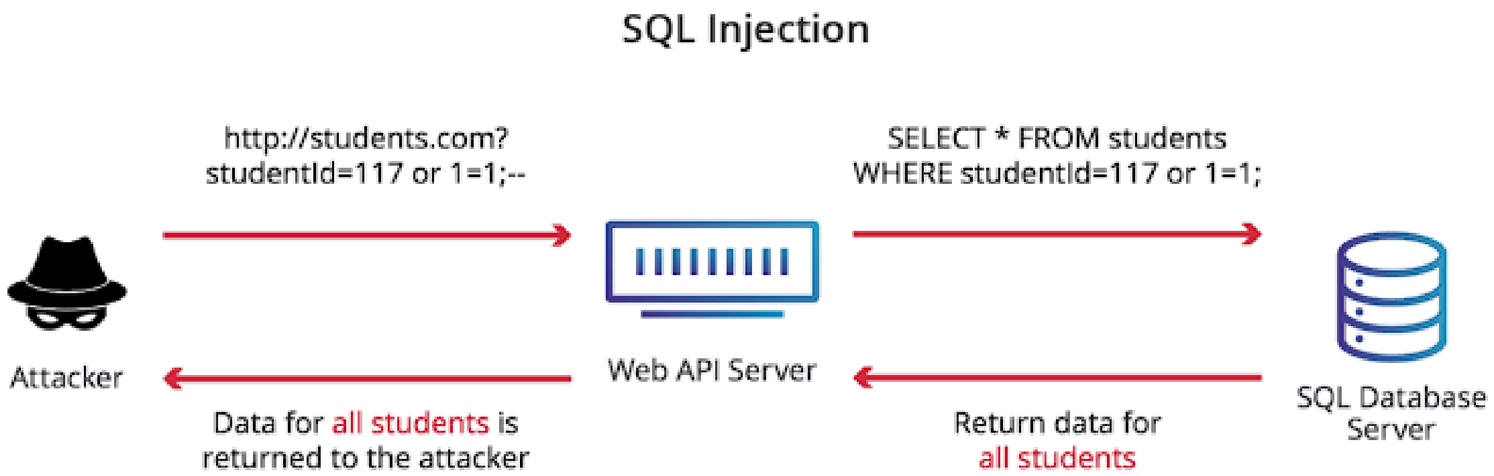
SECOND ORDER ATTACK

- Two queries separated by ;
- Results obtained after the second query is executed

- Purposes:
  DOS - Denial of service



TAUTOLOGIES



- 1' OR '1'='1
- admin' OR 1<4
- admin' OR 4>2
- x' OR 'select' > 's'
- x' OR 'select' < x'

ILLEGAL/LOGICALLY INCORRECT QUERIES

Enter trash/unacceptable data

Cause syntax, type or logical errors

Query fails

Error message with sensitive information

1. Original URL: http://www.pranithrao.com?id=57
2. SQL Injection: http:// www.pranithrao.com?id=57"
3. Error message showed: Unexpected character SELECT name FROM Employee WHERE id =57\"

UNION QUERY

Used to fetch

Datatype of columns

Data from other tables

To find column number

' ORDER BY 1--

' ORDER BY 1--

If there are 3 columns

' UNION SELECT 'a',NULL,NULL--

' UNION SELECT NULL,'a',NULL,--

' UNION SELECT NULL,NULL,'a'--

SELECT * FROM user_details
WHERE userid =' ' UNION
SELECT * FROM
EMP_DETAILS -- ' and
password = 'abcd'

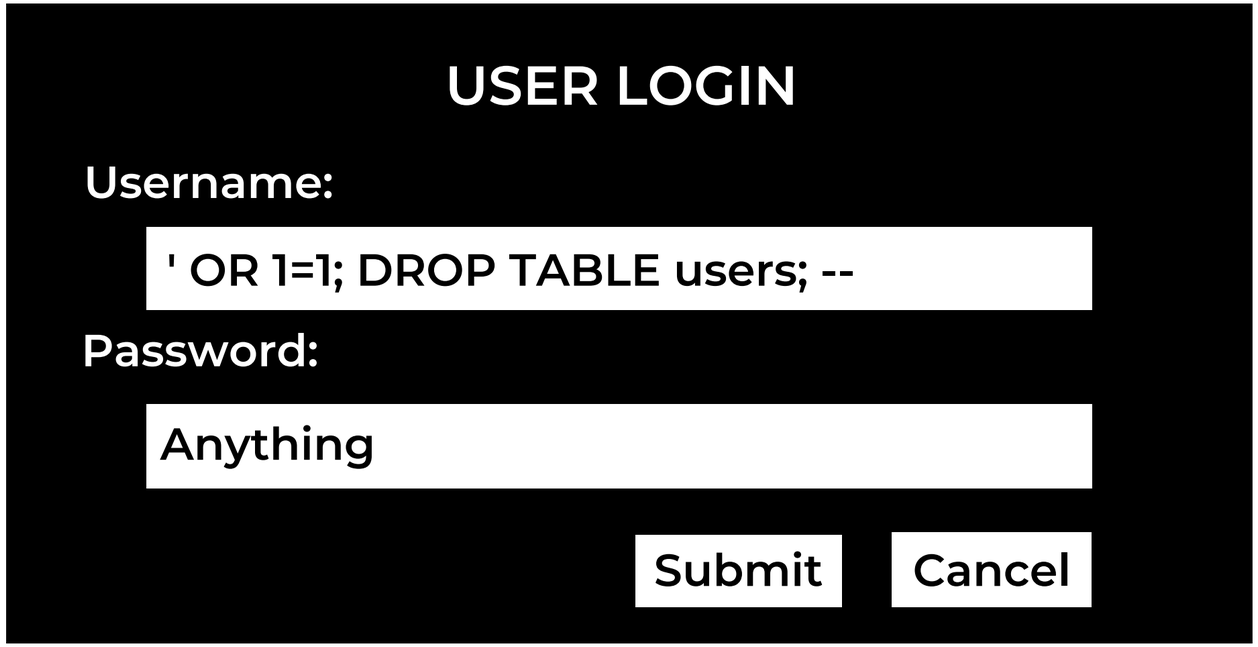
Error message:

Conversion failed when converting the varchar value 'a' to data type int.

PIGGY - BACKED QUERIES

- Similar to second order attack
- Two queries separated by ;
- Results obtained after the second query is executed

- Purposes:
  DOS - Denial of service



PREVENTING SQL INJECTION ATTACKS

- Use Bind Variables
- Validate the input
- Disable errors
- Limit input length
- Limit Characters
- Limit the Open-Ended input

USE BIND VARIABLES

- Most effective defense
- User input supplied as a variable and not as a direct input
- Increases performance

Using MySQL, without bind parameters:

```
$mysqli->query("select first_name, last_name"
              . "  from employees"
              . " where subsidiary_id = " . $subsidiary_id);
```

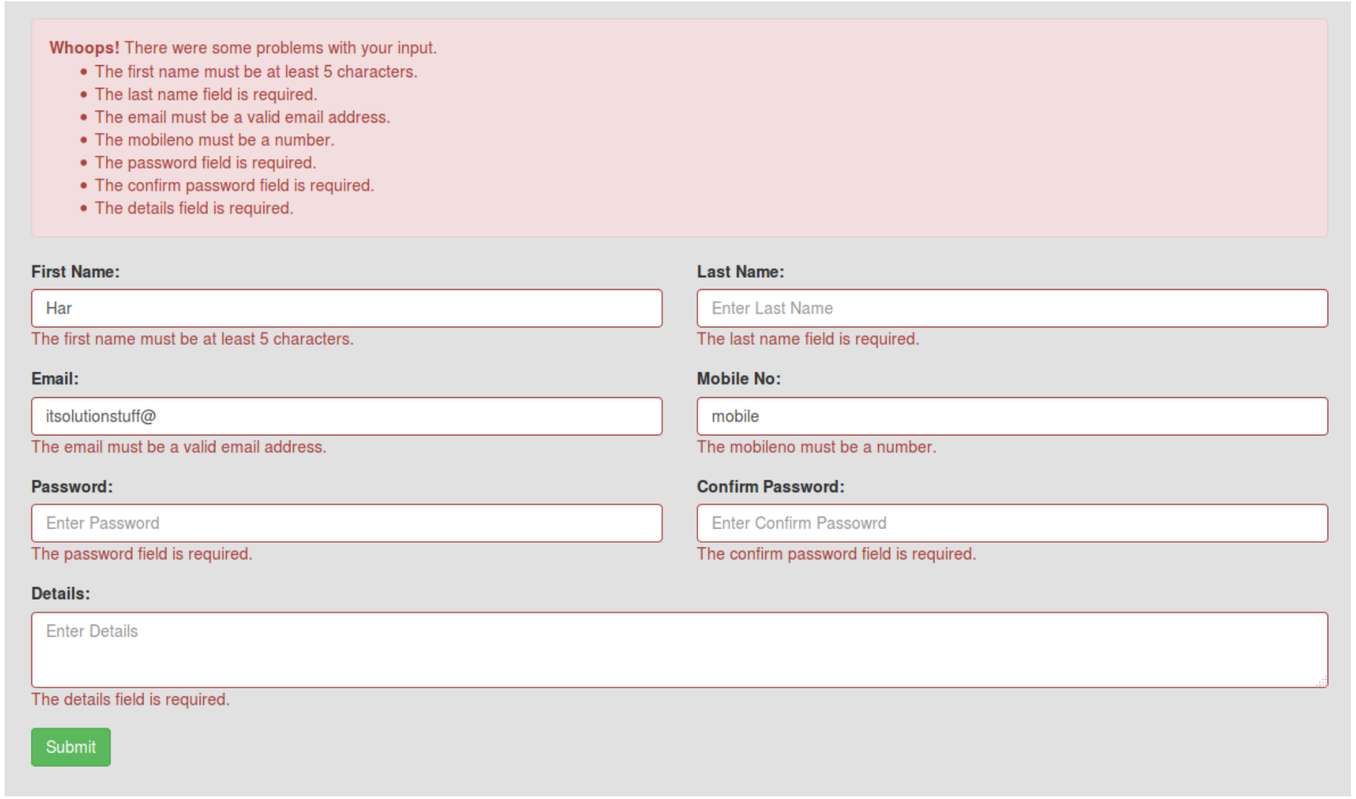
Using a bind parameter:

```
if ($stmt = $mysqli->prepare("select first_name, last_name"
                           . "  from employees"
                           . " where subsidiary_id = ?"))
{
   $stmt->bind_param("i", $subsidiary_id);
   $stmt->execute();
} else {
  /* handle SQL error */
}
```

VALIDATE THE INPUT

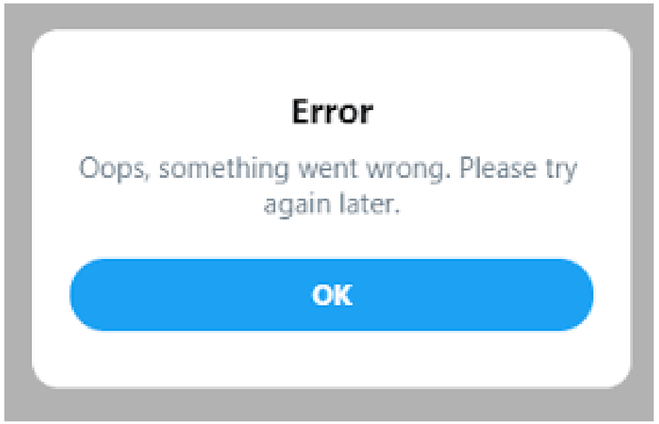
Most of the validations must be done at the client side

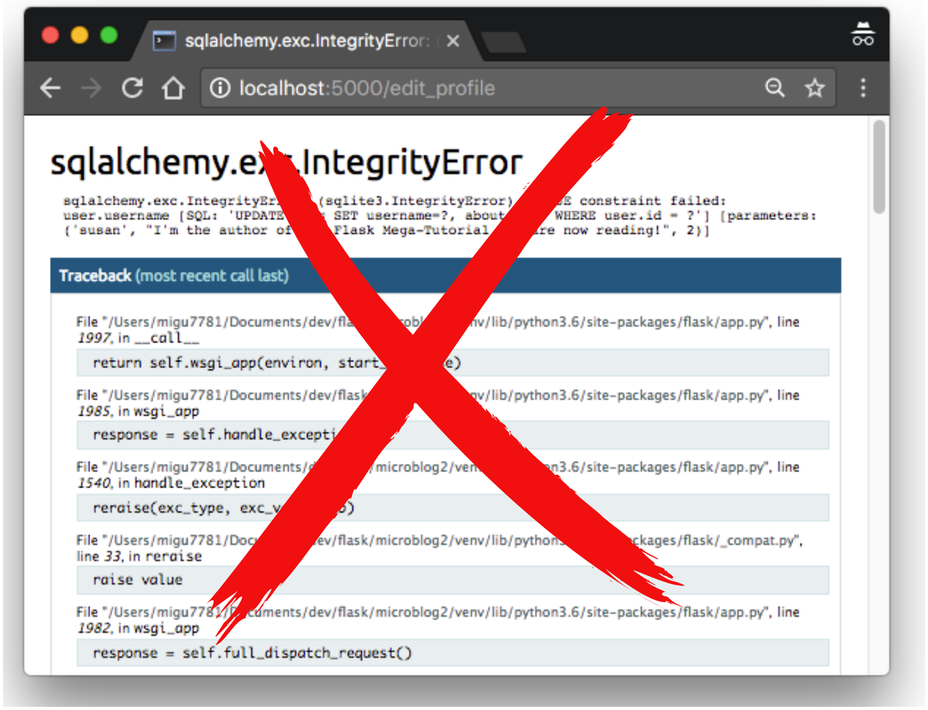
- Input type checking
- Pattern matching
- Remove or replace special database characters



DISABLE ERRORS

- Use TRY and CATCH blocks in backend code.
- Display custom messages instead of debug information.





LIMIT INPUT LENGHT

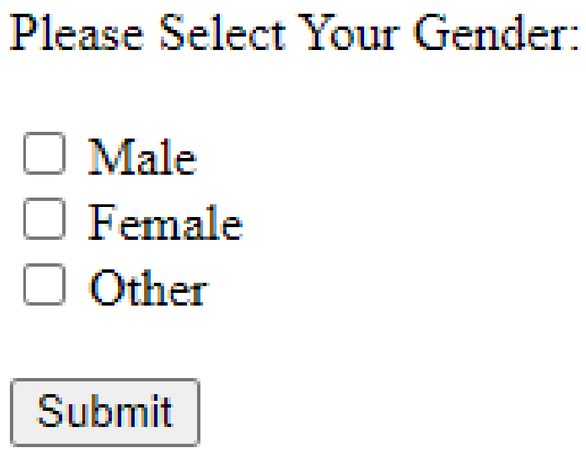
- Restrict the length of the input fields
- Saves memory
- Appending long and multi query strings wont be possible

LIMIT CHARACTERS

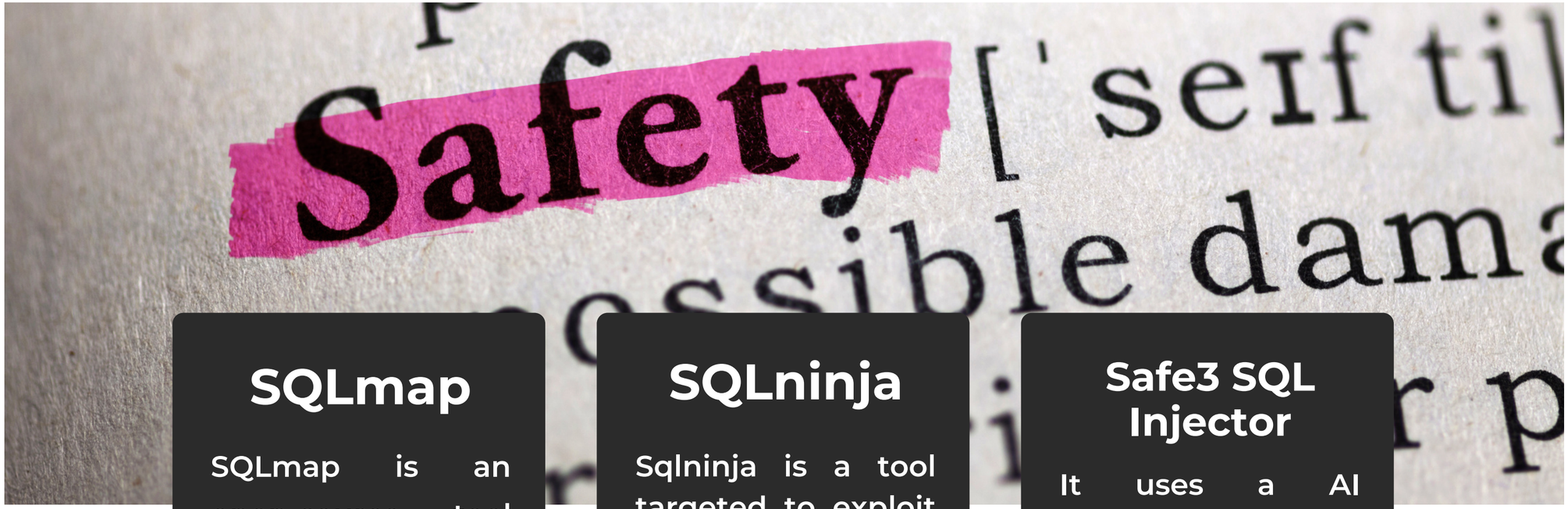
- Allow alphanumeric characters only
- Disallow SQL Injection payloads such as “<>/?*()&”
- Disallow keywords such as SELECT,UPDATE,DELETE,OR,HAVING,AND...

LIMIT THE OPEN-ENDED INPUT

- No text field provided for the user to enter data
- Alternatives like
  Bullets
  Radio buttons
  Tick boxes
  Dropdown menus
  should be used wherever possible.



SQL INJECTION PREVENTION TOOLS



SQLmap

SQLmap is an open-source tool used in penetration testing to detect and exploit SQL injection flaws

SQLninja

Sqlninja is a tool targeted to exploit SQL Injection vulnerabilities on a web application that uses Microsoft SQL Server as its back-end.

Safe3 SQL Injector

It uses a AI detection engine to detect and exploit SQL injection flaws and taking over of database servers


THANK YOU!