SQL Injection Attacks: Prevention and Security Measures

Hypothesis:

"The implementation of robust security measures can effectively mitigate the risks associated with SQL injection attacks, safeguarding databases from unauthorized access and potential data breaches."

First, let's understand what is SQL.

SQL stands for Structured Query Language, a language used to interact with structured databases.

DataBase! What is it?

A database is a set of data stored permanently in a structured way that ensures high efficiency in storing searching and updating data.

we have 2 types of databases relational databases and non-relational databases.

as the name says `relational` this type of database is characterized by relationships between tables.

so let's recap...

A Relational Database is a set of data stored permanently in tables that have relationships(Foreignkey).

Now we understand the meaning of DataBase and we know it's the structure that will save the data but how we are going to write and read data from it?

you guessed it it’s SQL

SQL is the language used to write, read, update, and delete in a database.

Let’s move to our topic SQL INJECTION.

Imagine we have a website and the first page is authentication for users to log in

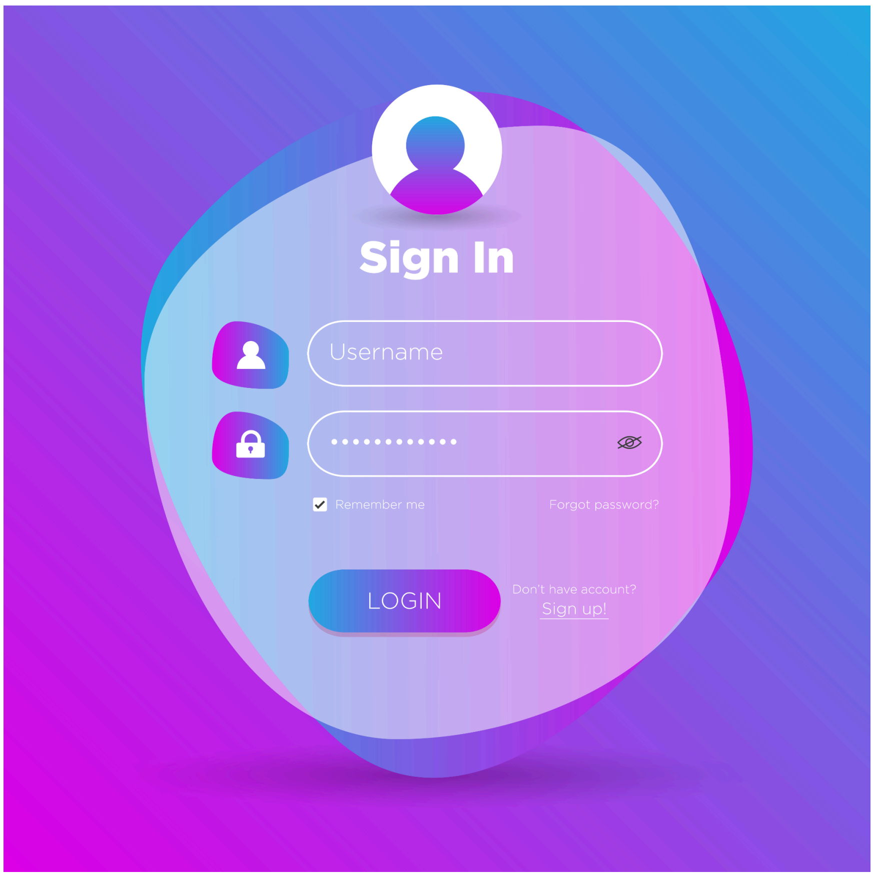

table: users

id	email	password
1	**user1@gmail.com**	**securePassword**
2	**johndoe@gmail.com**	**bestPassword**

when the user clicks Log in this is the SQL query sent:

```
SELECT *

FROM users

WHERE email = ’johndoe@gmail.com’

AND password = ’bestPassword’ LIMIT 1
```

And Yes it will work correctly, so where’s the problem?

Bad people exist .. and they will try to break things

so imagine if they used << `’ OR 1 == 1   - -`>> in the password input

we will send a request like this:

```
SELECT *

FROM users

WHERE email = ’johndoe@gmail.com’

AND password = ’ ’ OR 1 == 1   - -’ LIMIT 1
```

🔺 ‘–’ means all the text after is a comment

so the first quote will end the existing quote then ‘OR 1 == 1’ is a condition that always returns True, when we use OR in a condition, if 1 of the conditions statements returns True, the condition also returns True, and the ‘--’ will turn the last part to a comment, so we can log in without using any password TARAA.

This is SQL injection, it is the most dangerous method hackers can use to attack your website

There are a ton of other ways this is just an example **BUT PLEASE NOTE that SQL INJECTION IS A CYBER CRIME, NEVER TRY IT**

But if you want to try it for education purposes here's a legit way to use this test website:

https://demo.testfire.net/index.jsp

So let's move to WHAT'S THE SOLUTION?

as we have seen SQL Injections are so dangerous but the protection is not that hard...

1- Use Prepared statements with parameterized queries

2- Use stored procedures

3- Filter and validate user's input before using it in SQL query

There is always more.

Let's detail one of those solutions, Prepared statements.

using the same login example:

```
-- Assuming you have a MySQL database connection


SET @email = 'user@gmail.com';

SET @password = 'password';


-- Using a parameterized query to prevent SQL injection

PREPARE stmt FROM 'SELECT * FROM users WHERE email = ? AND password = ? LIMIT 1';

EXECUTE stmt USING @email, @password;

DEALLOCATE PREPARE stmt;
```

Using this method even if user input contains those malicious characters it will not break the system.