Defenses against SQL Injection

Use parametrized queries

- Ensure that user input is treated as data, not command
- `cursor.execute("SELECT * FROM Accounts WHERE name= ?“, (name))`

Object Relational Mapper (ORM)

- Abstract SQL generation and reduce risk of injection

```
class user(DBObject) {
  name =  Column(String(255));
  age =  Column(Integer);
  passsword =  Column(String(255));
}
```

Input inspection

- Sanitization: escape dangerous characters
- Validate and reject malformed input.
- Whitelist: only choose from allowed values

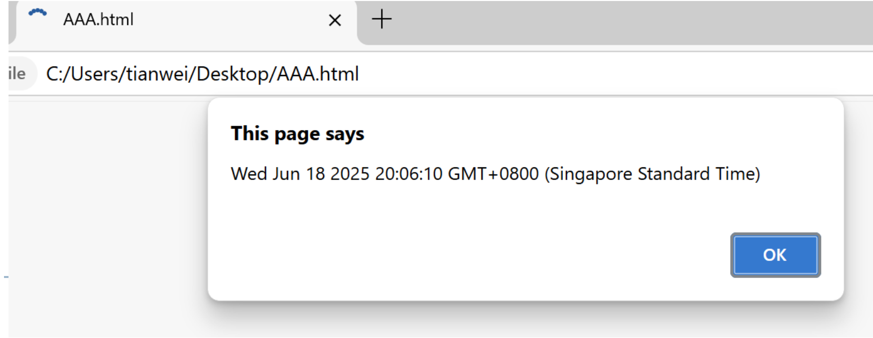
Example 3: Cross-Site Scripting (XSS)

JavaScript

- A programming language for web applications.
- The server sends the JavaScript code to the client, and the browser runs it.
- It makes the website more interactive.

JavaScript can be directly embedded in HTML with `<script>`

```
<html>
  <body>
    <script>alert(Date())</script>
  </body>
</html>
```



This page says

Wed Jun 18 2025 20:06:10 GMT+0800 (Singapore Standard Time)

Example 3: Cross-Site Scripting (XSS)

Basic idea of XSS

- The attacker injects malicious JavaScript code to a legitimate website
- When victim clients visit the website, the malicious code will be sent to their browsers, and executed on their local computers.
- The malicious code could insert malware to the victims' computers, or collect private information and send it to the remote attacker.

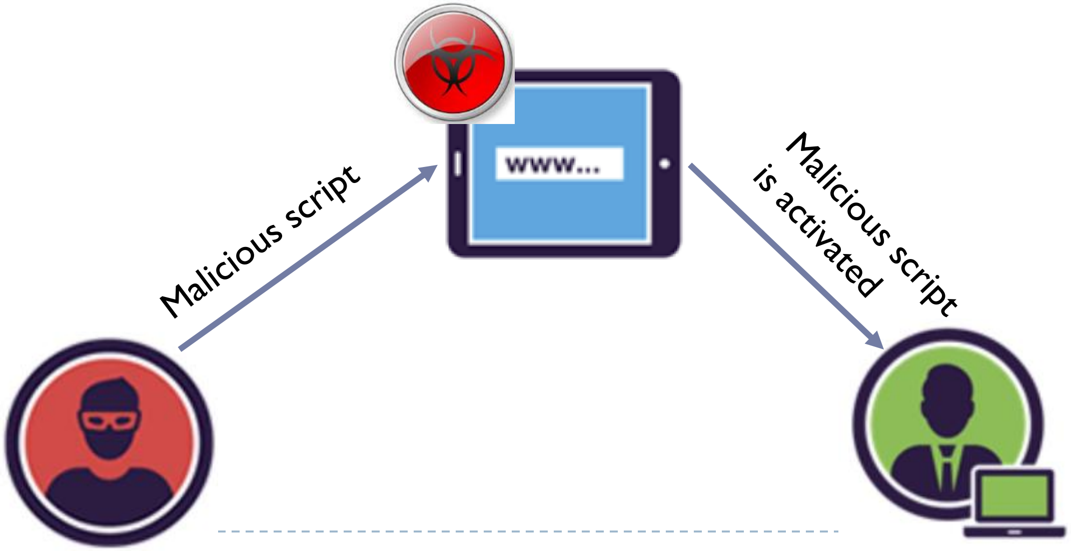
Two types of XSS

- Stored XSS
- Reflected XSS

Stored XSS Attack (Persistent)

Attacker’s code is stored persistently on the website

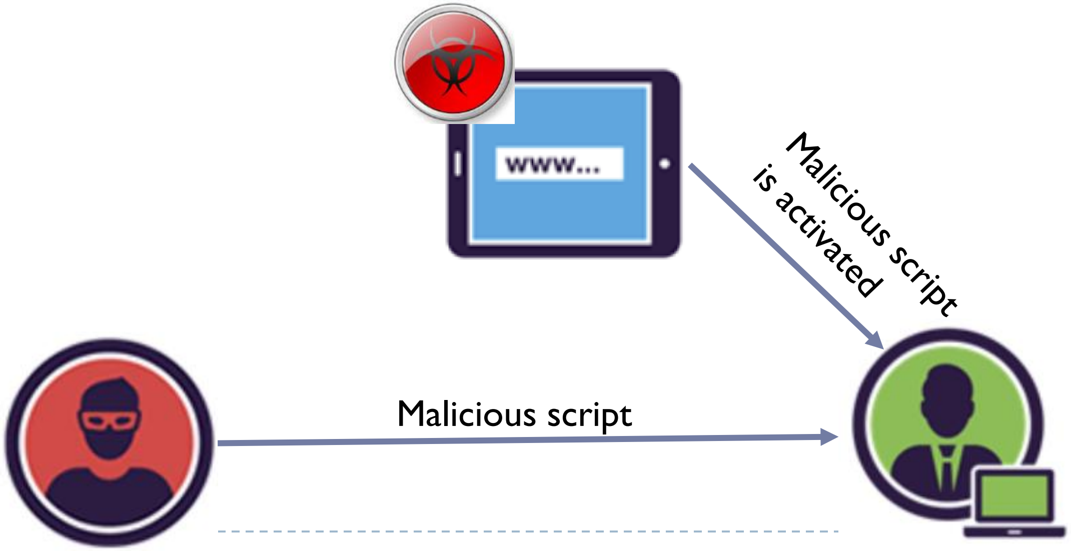
- The attacker discovers a XSS vulnerability in a website
- The attacker embeds malicious commands inside the input and sends it to the website.
- Now the command has been injected to the website.
- A victim browses the website, and the malicious command will run on the victim’s computers.



Reflected XSS Attack (Non-persistent)

The attacker tricks the victim to put the code in the request and reflected from the server

- The attacker discovers a XSS vulnerability in a website
- The attacker creates a link with malicious commands inside.
- The attacker distributes the link to victims, e.g., via emails, phishing link.
- A victim accidently clicks the link, which actives the malicious commands.



Defenses against XSS

Content Security Policy (CSP)

- Instruct the browser to only use resources loaded from specific places.
- Policies are enforced by the browser.
- Examples of policies
  - Disallow all inline scripts
  - Only allow scripts from specific domains

Input inspection

- Sanitization: escape dangerous characters
- Validate and reject malformed input.