1. **SQL injection attack**: 🡪

is a type of web attack in which the attacker adds Structured Query Language ([SQL](http://searchsqlserver.techtarget.com/definition/SQL)) code to a Web form input box to gain access to application database. In this m-project :

* 1. Develop a vulnerable web application with form inputs.
  2. Test different SQLI attack types.
  3. Add a code to secure the web application against SQLI attack

1. **Broken Authentication**🡪

● Weak session management

● Credential stuffing

● Brute force

● Forgotten password

● No multi-factor authentication

● Sessions don’t expire

**3. Sensitive Data Exposure**🡪

● Clear-text data transfer

● Unencrypted storage

● Weak crypto or keys

● Certificates not validated

● Exposing PII or Credit Cards

**5. Broken Access Control🡪**

● Access hidden pages

http://site.com/admin/user-management

● Elevate to an administrative account

● View other people’s data

http://site.com/user?id=7

● Modifying cookies or JWT tokens

**6. Security Misconfiguration🡪**

● Security features not configured properly

● Unnecessary features enabled

● Default accounts not removed

● Error messages expose sensitive information🡪

7.**XSS attack**: 🡪

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

In this project :

* 1. Develop a vulnerable web application with form inputs.
  2. Test different XSS attack types.
  3. Add a code to secure the web application against XSS attack

**8. Using Components with Known Vulnerabilities🡪** .

Modern applications contain a *lot* of third-party code.It’s hard to keep it all up to date.

Attackers can enumerate the libraries you use, and develop exploits

1. **Failles de type File Inclusion** : 🡪

La faille LFI tient son nom de **Local File Inclusion** (Inclusion de fichier local). Elle permet à un utilisateur d’inclure des fichiers locaux (appartenant donc au serveur externe) à partir d’une URL. Ces fichiers peuvent très bien être en dehors du répertoire racine du site web. Des fichiers sensibles comme ceux contenant des données personnelles et notamment des mots de passe peuvent donc être inclus et récupérés.

Dans ce mini-projet :

* 1. Réaliser une application vulnérable
  2. Tester les attaques LFI
  3. Sécuriser l’application contre les attaques LFI

1. **Command/ code execution:** .-->

Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell. In this attack, the attacker-supplied operating system commands are usually executed with the privileges of the vulnerable application

**11.File upload** :

Attacking a website or a web application through uploading a file which contains a malicious code inside it may result in  plenty of problems with the website itself.

**12.Web browser vulnerabilitiesm**

* 1. DRIVE-BY DOWNLOADS
  2. PLUGINS AND EXTENSIONS
  3. UI-REDRESS ATTACK
  4. MAN-IN-THE-BROWSER ATTACKS
  5. ADWARE

**13.CSRF vulnerabilities**