1.1 Project problem  
Because of vulnerabilities like SQL injection, cross-site scripting (XSS), unsecured APIs, and incorrectly setup servers, cyberattacks are increasingly targeting modern web applications. Manual vulnerability identification takes a lot of effort, is prone to mistakes, and calls for specific knowledge. Many businesses, particularly small-to-medium-sized businesses (SMEs), lack the funding necessary to put strong security procedures in place, leaving their systems vulnerable to intrusions. In order to proactively identify and reduce risks, this project tackles the urgent need for an automated, easily available, and effective web vulnerability scanner.

1.2 Project Goals  
This project's primary objectives are to: 1. Create an automated web vulnerability scanner that finds and reports common security flaws.   
2. To enhance web applications' security posture by making it possible to identify and fix problems early.   
3. To offer an easy-to-use solution that companies, security teams, and developers of all technical backgrounds can employ.

## What is an automated web vulnerability ?

An automated online vulnerability scanner is a software program that combines pre-established rules, machine learning (ML), and simulated attack patterns to systematically find security flaws in servers, web apps, and APIs. By doing away with human penetration testing, it makes security assessments quick, scalable, and repeatable.

## Why are vulnerability scanners Important?

1. Cost-effectiveness: Lower the costs related to manual security audits.   
2. Proactive Defense: Find weaknesses before they are used by attackers.   
3. Compliance: Adhere to legal requirements (such as GDPR and PCI-DSS).   
4. Protection of Reputation: Avoid data breaches that undermine corporate confidence.   
5. Continuous Monitoring: Make DevOps pipelines (shift-left security) capable of real-time scanning.

## What the project covers

Traversal of the Path   
• Definition: Gains access to unauthorized files (such as /../../etc./passwd) by taking advantage of inadequate input sanitization.   
• Impact: System breach and data theft.   
• Detection: Check server responses for file disclosures by injecting traversal sequences (such as../, %2e%2e%2f).   
XSS, or cross-site scripting   
• Definition: Inserts harmful scripts, such as , into web pages.   
• Types include DOM-based (client-side), stored (permanent), and reflected (URL-based).   
• Detection: In HTML/JS contexts, submit payloads and look for unencoded output.

Request Forgery on the Server Side (SSRF)   
In other words, it compels a server to send unapproved internal queries, as to AWS metadata endpoints.   
• Impact: Theft of cloud credentials and internal network reconnaissance.   
• Detection: Send internal IP-based URLs (such as http://169.254.169.254) and track the replies.   
1.5.4 SSTI, or server-side template injection, is defined as the process of inserting malicious code into templating engines, such as Jinja2 and Smarty.   
• Impact: Data breaches and remote code execution (RCE).   
• Finding vulnerabilities: Use template syntax to test (for example, {{7\*7}} → 49 indicates vulnerability).

## Beneficiaries

• Developers: Make sure that CI/CD pipelines incorporate security.   
• Penetration testers: Quickly identify vulnerabilities.   
• Organizations: Lower audit expenses and breach risks.   
• End Users: Protect private information from fraud.