



 slington college
(इस्लिङ्टन कलेज)

Module Code & Module

Title: API Security Tester

CS6P05NI Project Proposal

5% Individual Coursework

Submission: Final Submission

AY 2025 2026

Academic Semester: Autumn Semester 2025

Credit: 30 Credit Year Long Module

Student Name: Samriddhi Poudel

London Met ID: 23047345

College ID: NP01NT4A230037

Assignment Due Date: Wednesday, December 3, 2025

Assignment Submission Date: Wednesday, December 3, 2025

Internal Supervisor: Anuj Shilpakar

External Supervisor: Rabindra Khadka

I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Acknowledgement

I would like to express my sincere gratitude to my internal supervisor, Mr. Anuj Shilpakar, and my external supervisor, Mr. Rabindra Khadka, for their invaluable guidance, encouragement, and support throughout the development of this project. Their expertise and insightful feedback helped me stay focused and improve the quality of my work.

I am also grateful to my faculty, colleagues, and friends for their continuous motivation and assistance during this journey. Finally, I thank my family for their unwavering support and understanding, which made completing this project possible.

Abstract

APIs (Application Programming Interfaces) are now a requisite to the contemporary software systems, and it facilitates the seamless integration, automation, and enhances user experiences. Nevertheless, their accessibility and openness bring them to high levels of security breaches such as failed authentication, broken authorization and misconfigurations. The increasing rate of API-related attack, along with the lack of monitoring and control, aids the necessity of the proactive approach in security measures. This project will overcome such challenges by creating an API Security Tester and Vulnerability Scanner that improves the API security posture within an organization.

The suggested tool will be able to detect typical vulnerabilities, track APIs destined of suspicious behavior, create warnings, and elaborate reports. The system provides end-to-end protection of RESTful and third-party APIs by incorporating automated security testing, real-time touring, and interactive dashboards. The incremental Agile process will see to it that the tool is developed in functional modules to ensure that there is constant testing, refinement, and adjustment to the changing security threats.

Finally, the proposed project is expected to provide a convenient web-based application that ensures the ease of managing API security and mitigates the risks of operations. The capabilities of the tool such as vulnerability identification, automated monitoring, and reporting enable both the technical and non-technical users to have a strong API security. The project makes software environments safer and more reliable through automatized repetitive checks and actionable insights, therefore, helping organizations in the digital transformation process.

Table of Contents

1.	Introduction.....	1
1.1.	Problem Scenario	2
1.2.	Project as a Solution	3
2.	Aim and Objectives	4
2.1.	Aim	4
2.2.	Objectives	4
3.	Expected outcomes and Deliverables	5
3.1.	Deliverables	5
4.	Project Risks, Threats, and Contingency Plans	7
5.	Methodologies	8
5.1.	Considered Methodologies	8
5.1.1.	Agile Methodology	8
5.1.1.1.	Incremental Framework	9
5.1.2.	Spiral Methodology	10
5.1.3.	Prototype Methodology	11
5.2.	Selected Methodology	12
5.2.1.	Agile Methodology - Incremental Framework	12
6.	Resource Requirements	14
6.1.	Hardware Requirements	14
6.2.	Software Requirements	14
7.	Work Breakdown Structure(WBS)	15
8.	Milestones	16
9.	Gantt Chart.....	17
10.	Conclusion	18
11.	References	19

Table Of Figures

Figure 1 Common API Security Vulnerabilities (Shah H. , 2020).....	2
Figure 2 API Security Risks (Mehta, 2025).....	3
Figure 3 Agile Methodology (Perin, 2024)	9
Figure 4 Incremental Model (Framework) (More, 2023)	10
Figure 5 Spiral Methodology (intellectsoft, 2019)	11
Figure 6 Prototype Methodology (Benett, 2024)	12
Figure 7 Work Breakdown Structure (WBS).....	15
Figure 8 Milestones	16
Figure 9 Gantt Chart.....	17

Table Of Tables

Table 1: Project Risks, Threats, and Contingency Plan

1. Introduction

Modern software systems rely upon APIs (Application Programming Interfaces), which are described as the connective tissue of an application and a service. They facilitate data flow, automate processes, integrate microservices and cloud-native platforms, improve operations and innovate. Nowadays, APIs are used by more than 85 percent of enterprises to provide customers with smooth experiences and achieve digital transformation (Astera, 2024).

Even though APIs are critically important, they are vulnerable since they are open and accessible by nature. Authentication, authorization, encryption, input validation, rate limiting, logging, monitoring, and secure coding practices are some of the measures that can be applied in API security to reduce unauthorized access and data breaches. The API security in the complex IT ecosystems, including microservices and third-party integrations, requires some special strategies besides the traditional web application security (Chinnasamy, 2023).

The incidents involving T-Mobile (2022), LinkedIn (2021), Facebook (2019), Strava (2018) and Equifax (2017) among others are all examples of highly advanced API gateways that are not immune to attacks. Moreover, the growth of API-related attacks in India grew over 3000% in Q3 2024, and 30% of critical vulnerabilities have not been patched in more than six months (Chinnasamy, 2023).

All this speaks of the urgency of proactive and comprehensive API security measures.

1.1. Problem Scenario

APIs are still very prone to misuse due to a number of reasons. Some of the common vulnerabilities are broken object-level authorization, failed authentication, broken function-level authorization, and inadequate input validation, among others, enabling attackers to access unauthorized services or disrupt services (OWASP, 2023).

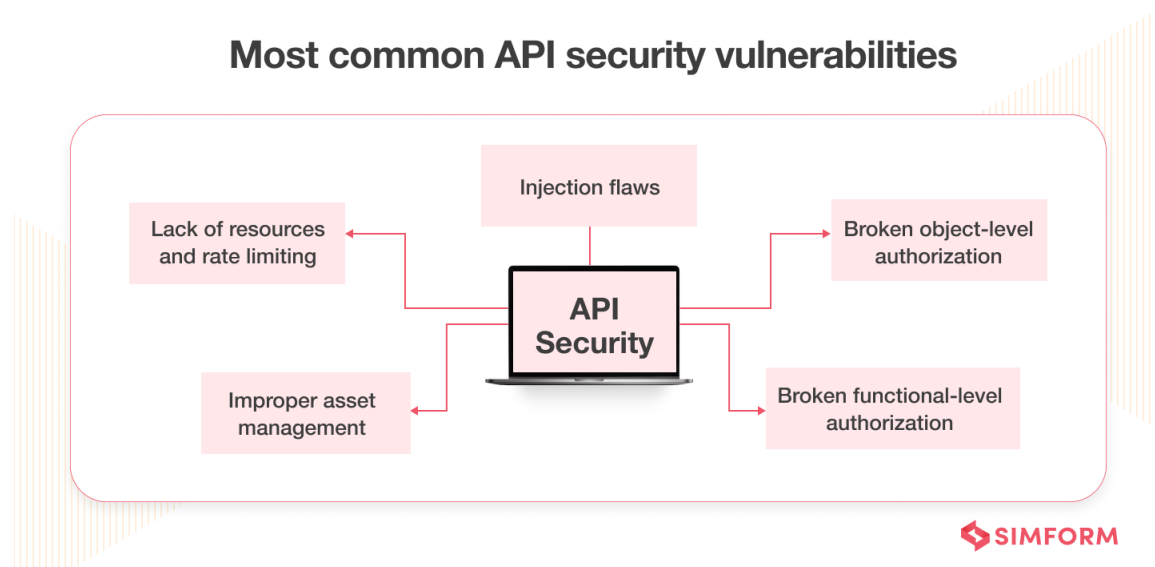


Figure 1 Common API Security Vulnerabilities (Shah H. , 2020)

Also, inadequate inventory control and documentation, unmanaged third-party APIs, and unlimited resource usage are one of the factors that make the attack surface bigger. Most organizations have no complete access to API endpoints and integrations and have blind spots that attackers can exploit. The security issue is increasing exponentially as APIs spread to facilitate the process of digital transformation (Samuels, 2024).

API Security Risks

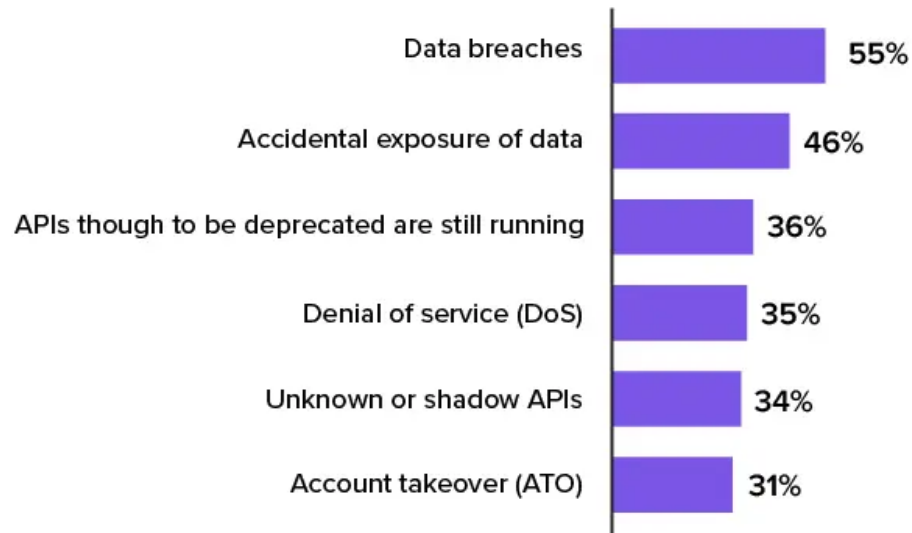


Figure 2 API Security Risks (Mehta, 2025)

1.2. Project as a Solution

In order to overcome these issues, this project suggests creation of API Security Tester and Vulnerability Scanner. The tool will offer a single platform to:

- Identify weaknesses like failed authentication, authorization, SSRF, and misconfigurations.
- Keep an eye on APIs and track their abnormal activity.
- Create warning and reports so that it mitigates possible threats in advance.
- Implement best practice in API security in a wide range of environments, such as RESTful and third-party API.

The project will help to enhance the organizational API security posture by automating testing, continuously monitoring, and providing comprehensive reporting of the results and reducing the operational risk and availability of services.

2. Aim and Objectives

2.1. Aim

This project's goal is to create and implement an integrated API Security Tester and Vulnerability Scanner that protects APIs by finding vulnerabilities, enforcing security best practices, and offering real-time monitoring and alerting.

2.2. Objectives

- Create a tool to assess APIs for prevalent vulnerabilities, such as broken authentication, authorization issues, SSRF, and security misconfigurations.
- Set up automated surveillance to identify unusual API behavior and possible attacks.
- Incorporate instant alerts and notifications for essential API vulnerabilities.
- Deliver user-friendly dashboards and reports to illustrate API security status for both technical and non-technical audiences.

3. Expected outcomes and Deliverables

The goal of this project is to provide an automated, lightweight API Security Testing and Vulnerability Scanner that can find common flaws in API endpoints. A working web-based solution that enables users to input or upload API data and obtain a thorough analysis of potential security flaws based on industry-standard procedures is the anticipated result.

By automating repetitious checks, displaying risk levels, and producing concise, useful vulnerability reports, the tool is anticipated to simplify API security testing for beginners as well as professionals.

3.1. Deliverables

The key deliverables of the API Security Testing Tool are as follows:

- Dashboard on the Web
An easy-to-use interface that allows users to view scan findings, enter API URLs, monitor status, and obtain reports.
- Module for Automated Security Testing
- A backend engine that runs tests like:
 - Checks for missing or invalid authentication tokens
 - Attempts to bypass authorization
 - Stress testing with rate limits
 - Tests for input validation (oversized payloads, special characters, and corrupted data)
 - Response analysis to find critical data exposure, stack traces, or error leaks
- Visualization of Results
 - Interactive summaries and charts that show:
 - Severity of vulnerability (Low, Medium, High, Critical)
 - Tests that failed vs those that passed
 - Anomalies and response patterns
- Generator for Vulnerability Reports

automatic creation of a structured report in PDF or HTML that includes: Every vulnerability found Rating of severity Impact and description Suggested fixes API metadata and the timestamp • Notification and Alert System alerts produced by the system for: Important weaknesses Failures in authentication Rate-limit problems Possible disclosure of private information

4. Project Risks, Threats, and Contingency Plans

Risk	Description	Likelihood	Impact	Contingency Plan
API Variability	Different APIs behave differently, generating conflicting scan results.	Medium	Medium	Allow custom configs for different API types.
False Positives/Negatives	Scanner may misidentify vulnerabilities.	Medium	High	Add manual review mode and refine rules.
Testing Risks	Stress tests may disrupt weak APIs.	Low	High	Use secure scans and limit risky tests.
Performance Issues	Multiple scans may slow down the system.	Medium	Medium	Reduce redundant scans and optimize the backend.
Limited Coverage	Advanced vulnerabilities may not be detected.	Medium	Medium	Update rules regularly

Table 1: Project Risks, Threats, and Contingency Plans

5. Methodologies

A stepwise process that ensures the development, implementation, support and disposal of information systems is the System Development Life Cycle (SDLC), and NIST points out that security needs to be incorporated at all stages to provide robust protection throughout the cycle. Through risk assessment, determining the right security controls, performing ongoing monitoring and proper disposal of systems, organizations are able to mitigate the vulnerabilities at an early stage as well as ensuring that confidentiality, integrity and availability of information within the system is maintained throughout the life of the system. After standards like FIPS 199, FIPS 200 and NIST SP 800-53 can be used in ensuring that security is not an afterthought but an in-built requirement which makes the systems more resilient, affordable and reliable over time (Radack, 2009).

5.1. Considered Methodologies

5.1.1. Agile Methodology

Agile methodology refers to an iterative approach of project management which involves splitting the work into short flexibilities to enable the teams to adjust easily when the requirements change. It places a high emphasis on working software, close customer working collaboration and never-ending improvement instead of inflexible planning. Agile makes software projects responsive and efficient by promoting collaboration, constant feedback, and continual delivery. This is what renders it optimal in dynamic and rapid systems such as the contemporary API security tools (Paulk, 2021).

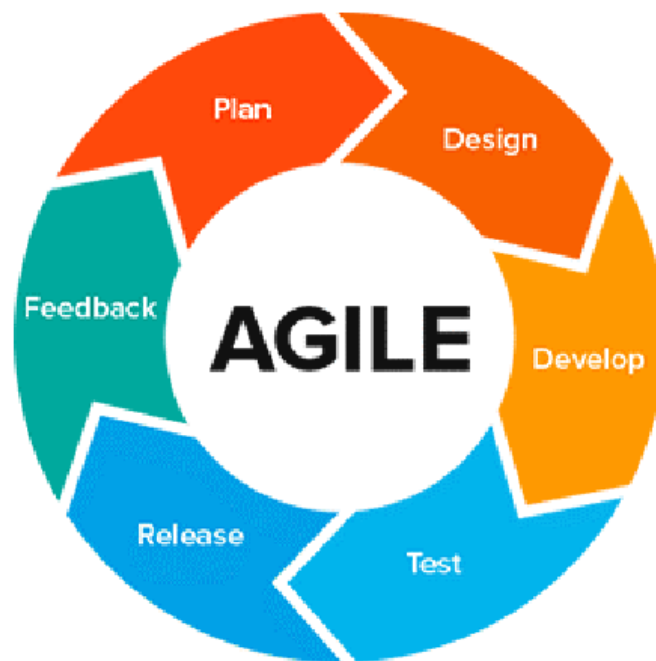


Figure 3 Agile Methodology (Perin, 2024)

5.1.1.1. Incremental Framework

Incremental framework Software development model through which the system is developed and delivered in small and manageable chunks known as increments. Each module is built, tested and implemented one step at a time rather than all in one go and with each increment, a new functionality is added on the project. This will enable teams to receive feedback at an earlier stage, modify requirements fast and mitigate risks during the project. It is specially useful in complicated systems as the workload is split and divided into distinct phases, making the work more adaptive and the progress becoming visible at each step (More, 2023).

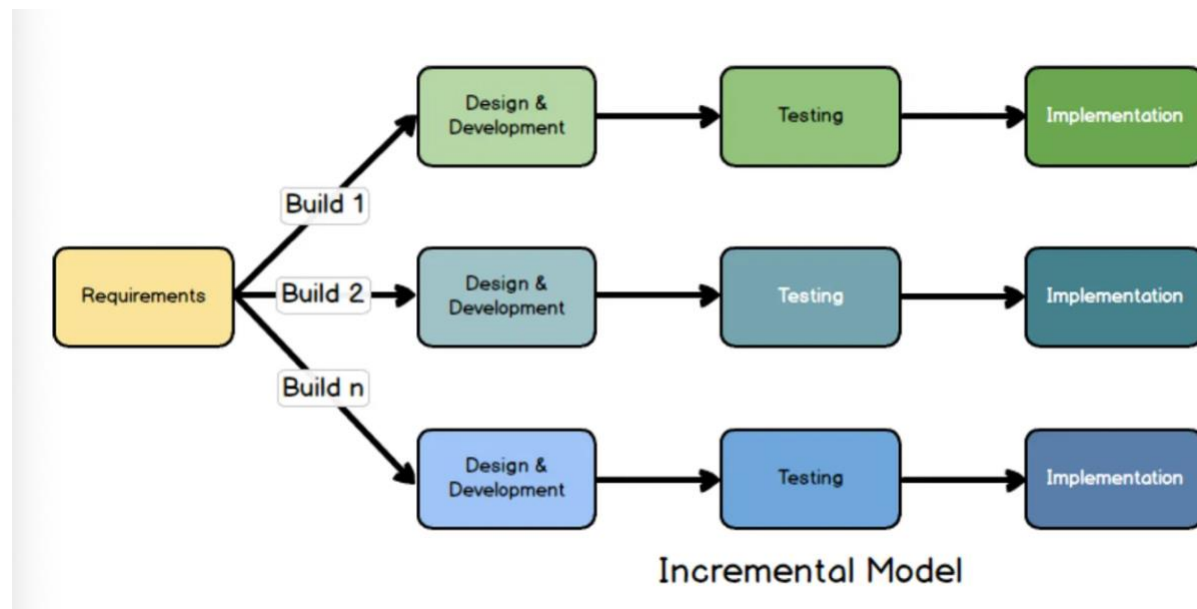


Figure 4 Incremental Model (Framework) (More, 2023)

5.1.2. Spiral Methodology

Spiral methodology is a community-driven approach created by the Council of Europe to understand well-being from the people's own perspective. Instead of using fixed indicators, it asks individuals directly what well-being and ill-being mean to them, then organizes those responses into shared values. It's open, participatory, and designed to help municipalities build policies based on collective needs and co-responsibility (Inga Jēkabsone, 2013).

Spiral model

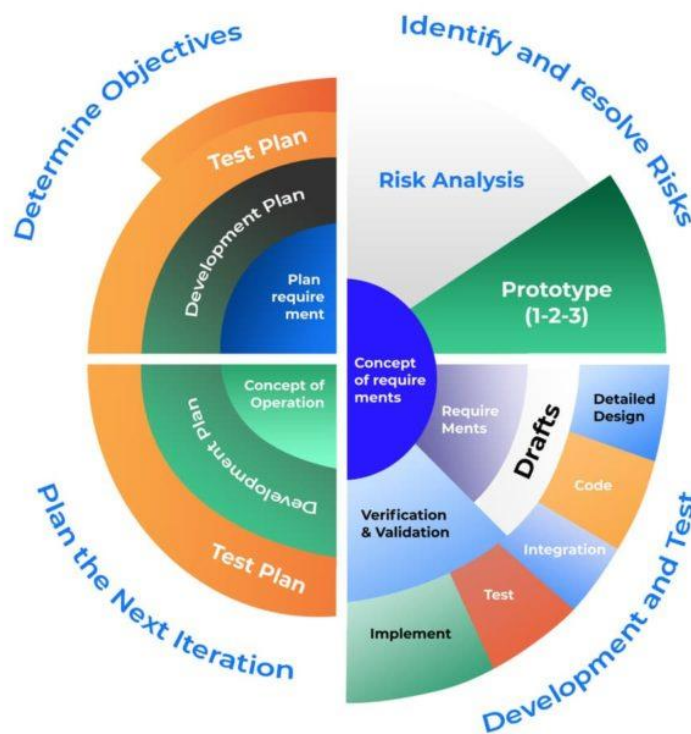


Figure 5 Spiral Methodology (intellectsoft, 2019)

5.1.3. Prototype Methodology

Prototype-based methodology is a software development approach where the system is built through quick, iterative prototypes instead of relying on fixed, predefined classes or structures. Instead of defining everything upfront, developers create working models that evolve as the project progresses. This makes it flexible for applications whose requirements change frequently or can't be fully understood in the beginning. By continuously refining the prototype based on feedback, the final system becomes more aligned with real user needs and behaviors (Shah, 2001).

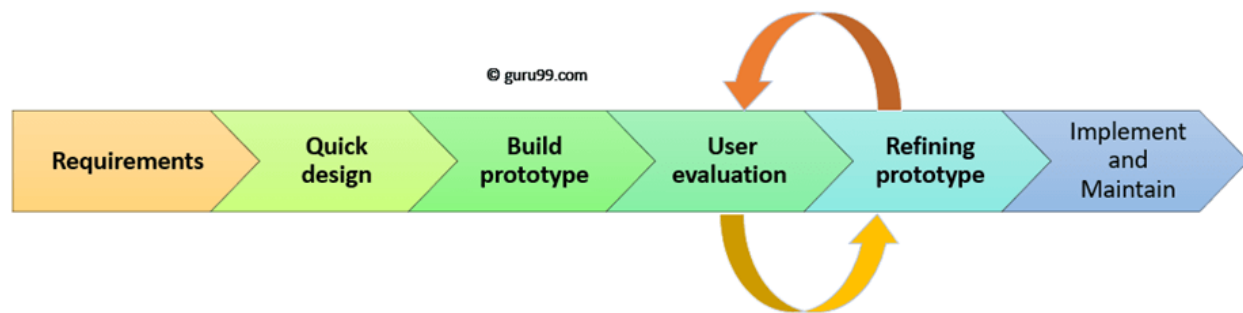


Figure 6 Prototype Methodology (Benett, 2024)

5.2. Selected Methodology

Among all the methodologies researched the Agile Incremental Methodology was selected for the development of the API Security Tester.

5.2.1. Agile Methodology - Incremental Framework

The Agile Incremental Framework is an iterative development approach where the system is built and delivered in small, functional increments. Each increment adds new features or enhancements to the existing system, allowing continuous feedback, early validation, and progressive improvement. This framework emphasizes flexibility, frequent evaluation, and structured progress to ensure the product aligns with user needs and evolving requirements.

Key Reasons for Choosing the Incremental Framework:

i. Iterative Development Suits Security Tools

API security testing requires constant refinement; incremental delivery allows testing modules to evolve gradually while adapting to new vulnerabilities.

ii. Simplified Documentation

Each increment produces clear deliverables and progress records, making FYP documentation organized and easier to compile.

iii. High Flexibility

As new security requirements or threats emerge, the incremental framework allows adjustments without affecting the entire system.

iv. **Progressive Feature Validation**

Core testing features, monitoring modules, and reporting components can be developed and validated in stages, ensuring quality and early feedback.

6. Resource Requirements

6.1. Hardware Requirements

- i. Stable Internet Connection for API Testing
- ii. A personal laptop

6.2. Software Requirements

- i. Operating System: Windows 10/11 or Unix
- ii. Development Environment: Visual Studio Code / PyCharm
- iii. Programming Languages: Python / JavaScript
- iv. Database: MySQL / PostgreSQL
- v. Documentation: MS-Word
- vi. Web Browser: Google Chrome/ Brave

7. Work Breakdown Structure(WBS)



Figure 7 Work Breakdown Structure (WBS)

8. Milestones

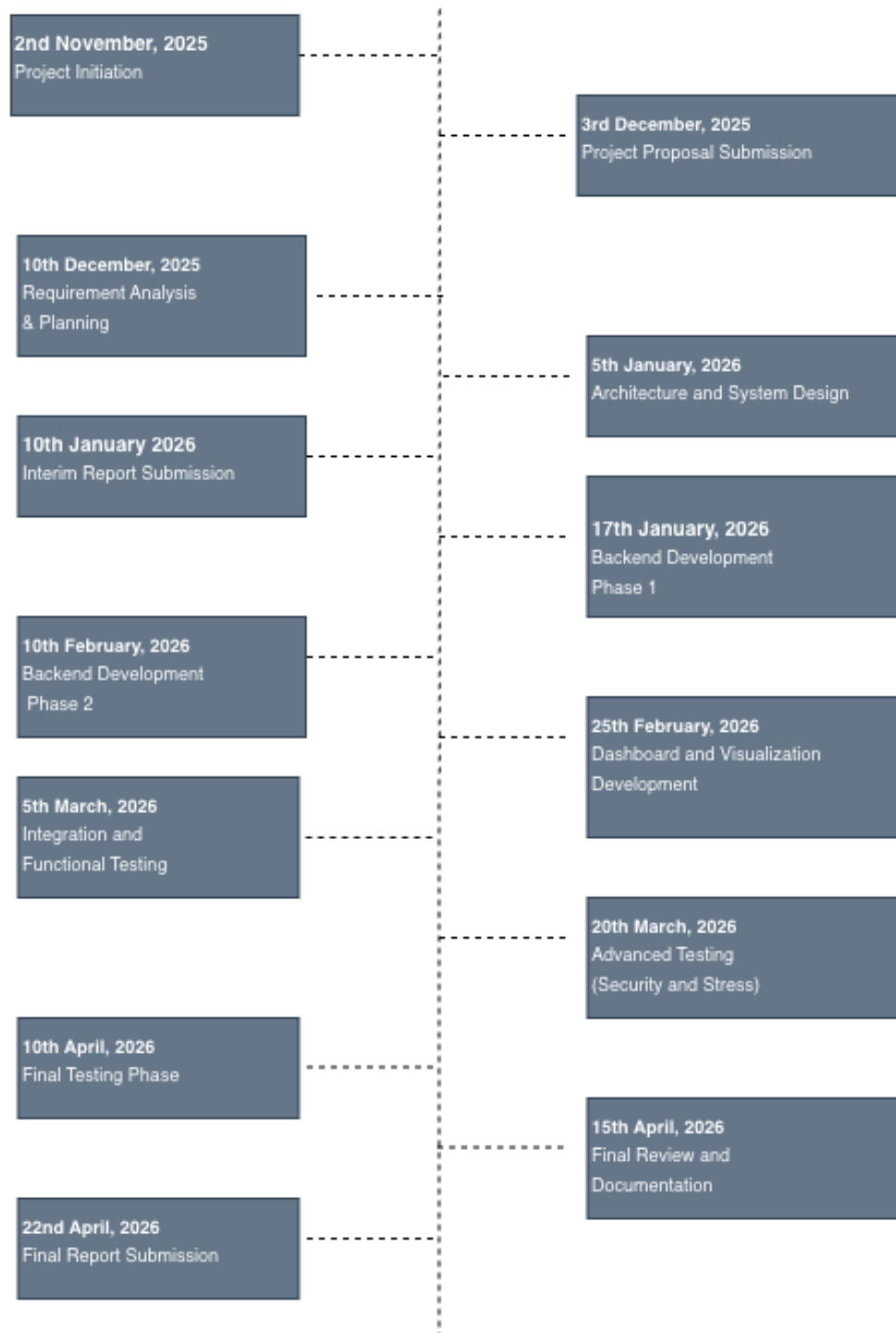


Figure 8 Milestones

9. Gantt Chart

teamgantt
Created with Free Edition

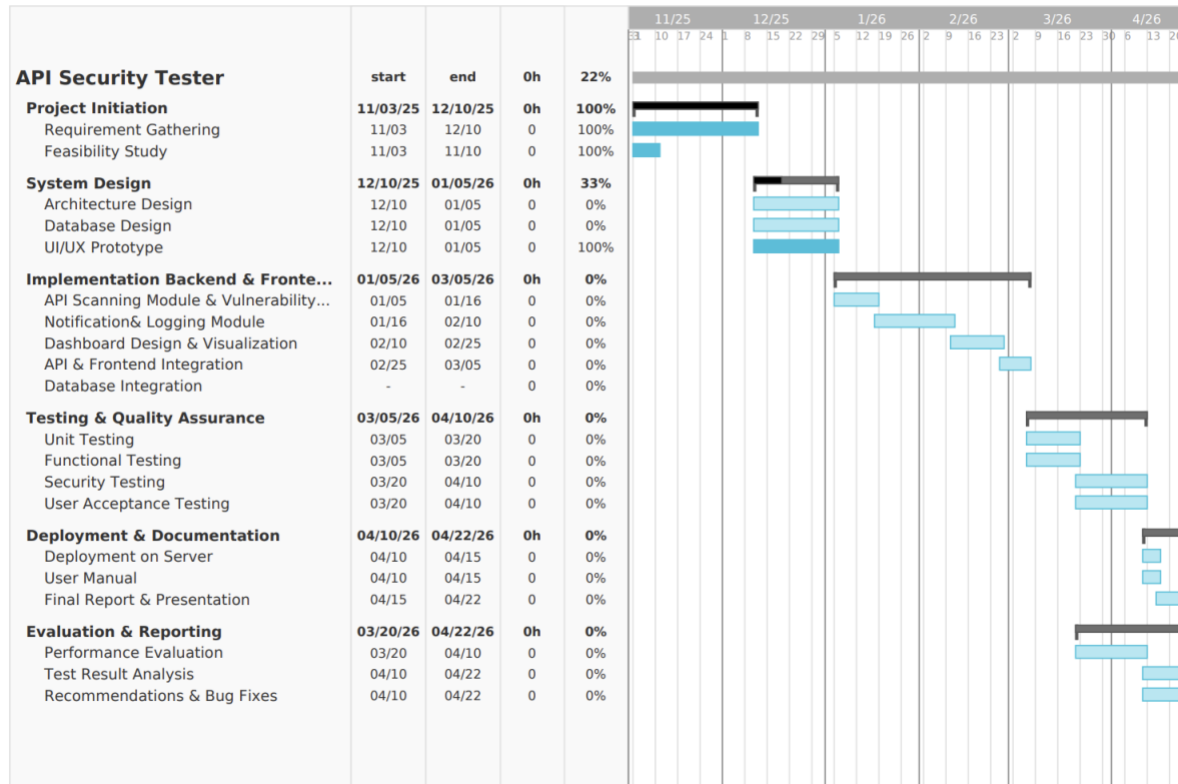


Figure 9 Gantt Chart

10. Conclusion

The suggested API Security Tester and Vulnerability Scanner will be able to fill the emerging threat landscape related to APIs by offering an all-encompassing, automated, and user-friendly tool. The tool will enhance the API security posture of an organization, monitor suspicious behavior, generate actionable reports, and present real-time notifications, which will mitigate operational risk.

The implementation of Agile Incremental Framework will provide the flexibility of the development process, iterativeness, and responsiveness to the changing security threats, which will help to continuously improve and verify the system. The software-based work in the project embraces the current software practices and technologies to provide an accessible web-based platform that is both user-friendly and non-technical to facilitate effective management of the API security measures.

Finally, the project will lead to safer and more resilient software environments, increase the resiliency of organizations in case of cyber attacks, and advance the digital transformation process of organizations by guaranteeing safe and reliable API operations.

11. References

- Astera. (2024, June 27). *Everything You Need to Know about API Adoption*. From Astera: <https://www.astera.com/type/blog/api-adoption/>
- Chinnasamy, V. (2023, December 15). *API Security 101: Understanding the Risks and Implementing Best Practices*. From Indusface: <https://www.indusface.com/blog/what-is-api-security-and-why-is-it-important/>
- OWASP. (2023). <https://owasp.org/API-Security/editions/2023/en/0x11-t10/>. From owasp.org: <https://owasp.org/API-Security/editions/2023/en/0x11-t10/>
- Samuels, R. (2024, September 24). *API Security: The 6 biggest challenges AppSec teams face, and how to solve them*. From Portswigger: <https://portswigger.net/blog/api-security-the-6-biggest-challenges-appsec-teams-face-and-how-to-solve-them#:~:text=and%20knowledge%20sharing.-,Limitations%20of%20current%20testing%20and%20tools,tools%20can't%20simulate%20them.>
- Radack, S. (2009). *THE SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)* . National Institute of Standards and Technology . From <https://csrc.nist.gov/csrc/media/publications/shared/documents/itl-bulletin/itlbul2009-04.pdf>
- Paulk, M. C. (2021). *Agile Methodologies and Process Discipline*. Software Engineering Institute.
- Inga Jēkabsone, S. T. (2013). *CHALLENGES OF THE SPIRAL METHODOLOGY FOR WELL-BEING STUDIES*. Latvia: University Of Latvia .
- Shah, A. (2001). A Framework for the Prototype-based Software Development Methodologies. *Journal of King Saud University - Computer and Information Sciences*, 13, 111-131.
- Perin, M. A. (2024, June). From Research Gate: https://www.researchgate.net/figure/Agile-Development-Methodology-Figure-3-depicts-the-Agile-Development-Methodology-applied_fig1_381116391
- intellectsoft. (2019, March 19). *Guide to Spiral Model SDLC*. From intellectsoft: <https://www.intellectsoft.net/blog/spiral-model-sdlc/>
- Benett, L. (2024, August 13). *Prototype Model in Software Engineering*. From guru99: <https://www.guru99.com/software-engineering-prototyping-model.html>

Shah, H. (2020, December 10). *API Security Best Practices to Protect Data* . From simform: <https://www.simform.com/blog/api-security-best-practices/>

Mehta, A. (2025, September 10). *The top API security risks and how to mitigate them*. From appinventiv: <https://appinventiv.com/blog/how-to-mitigate-api-security-risks/>

More, J. (2023, Aug 30). *Incremental Model in Software Development Life Cycle (SDLC): Phases, Advantages & Disadvantages*. From medium: <https://medium.com/design-bootcamp/dlc-software-development-model-incremental-development-7598ae6500e6>