Software Requirement Specifications

Version: [1.8]

Project ID	*F25-119*
Supervisor	*Sir Minhal Raza*
Co Supervisor	*Sir Usama Antuley, Miss Saeeda Kanwal*
Project Team	*Zehra Jabeen Mirza(22K-4781), Muhammad Shehryar Zubair Khan(22K-4736), Anas Ghazi(21L-5081)*
Submission Date	*Dec 11, 2025*

Document History

Version	Name of Person	Date	Description of change
1.0	Zehra Jabeen Mirza	*11/19/25*	*Document Created*
1.1	Anas Ghazi	*11/19/25*	*Intro section completed*
1.2	Anas Ghazi	*11/21/25*	*Scope and Not in Scope*
1.3	Zehra Jabeen Mirza	*11/21/25*	*System Constraints and Interfaces*
1.4	Zehra Jabeen Mirza	*11/24/25*	*Use Case Diagram plus Functional and Non-Functional Requirements*
1.5	Zehra Jabeen Mirza	*11/24/25*	*Use Case Table for PZ-01 and PZ-02*
1.6	Muhammad Shehryar Zubair Khan	*11/25/25*	*Use Case table for PZ-03 and PZ-04*
1.7	Muhammad Shehryar Zubair Khan	*11/26/25*	*use case table for PZ-05*
1.8	Muhammad Shehryar Zubair Khan	*11/29/25*	*References and over all personal review of the document*

Distribution List

Name	Role
Sir Minhal Raza	Supervisor
Sir Usama Antuley	Co- Supervisor
Miss Saeeda Kanwal	Co- Supervisor

Document Sign-Off

Version	Sign-off Authority	Sign-off Date
1.8	*Supervisor:Minhal Raza*	*30/11/2025*
1.8	*Co supervisor: Sir Usama Antuley*	*30/11/2025*
1.8	*Co supervisor: Miss Saeeda Kanwal*	*30/11/2025*

Table of Contents

1. 7

1.1. *7*
1.2. *7*
1.3 Abbreviations ……...8
1.4. *7*

2. 8

2.1. *8*
2.2. *8*
2.3. *9*
2.4. *9*
2.5. ***Error! Bookmark not defined.***
2.6. ***Error! Bookmark not defined.***
2.7. *9*
2.8. *10*
2.9. Dependencies 11

3. 12

3.1. ***Error! Bookmark not defined.***
3.2. *12*
3.3. *13*

4. 15

4.1. 15

4.1.1 **Error! Bookmark not defined.**

5. 22

5.1. *22*
5.2. *22*
5.3. *22*
5.4. ***Error! Bookmark not defined.***

6. 24

7. 25

1. Introduction

1.1. Purpose of Document

*The purpose of this Software Requirements Specification (SRS) is to present a clear, detailed, and structured description of all functional and non-functional requirements for **Phisher Zero**, an AI-powered phishing email detection system. This document ensures that all stakeholders, including technical teams, evaluators, and end-users, share a unified understanding of the system's objectives, scope, and constraints.*

The SRS aims to:

- *Provide a clear communication framework among all project stakeholders.*
- *Serve as a reference document for developers during system design, development, testing, and deployment.*
- *Ensure alignment between the system architecture and the core goal of detecting phishing attempts using AI agents.*
- *Establish a foundation for requirement traceability across the entire development lifecycle.*

1.2. Intended Audience

- *Fast NU*
- *Jury*
- *Supervisor (Sir Minhal Raza)*
- *Co Supervisors (Sir Usama Antuley / Ms Saeeda Kanwal)*
- *Students of FAST NU*
- *Our Team (Designer, Developer, Tester)*
- *Potential users of Phisher Zero, including individuals and organizations seeking secure email screening.*

1.3 Abbreviations

- ***LLM** = Large Language Model*
- ***NLP** = Natural Language Processing*
- ***URL** = Uniform Resource Locator*
- ***SRS** = Software Requirements Specification*

1.4 Document Convention

- *Font Family = Arial*
- *Font Size = 12 for headings, 10 for the rest of the content*

2. Overall System Description

2.1. Project Background

Phishing continues to be one of the most common and damaging cybersecurity threats faced by individuals, organizations, and institutions. Attackers exploit social engineering, deceptive URLs, forged email content, and malicious attachments to trick users into revealing sensitive information. Traditional spam filters and rule-based detection methods often fail against modern phishing attacks that use AI-generated text and sophisticated obfuscation techniques.

To address these challenges, ***Phisher Zero*** *leverages advanced AI agents capable of analyzing email content, URLs, and metadata to detect phishing attempts. The system aims to minimize human error, reduce the burden on security teams, and provide an accessible, accurate, and explainable phishing detection platform for regular users.*

Problem Statement:

- *Increasing sophistication of phishing emails makes traditional rule-based filters inadequate.*
- *Manual identification of phishing attempts is time-consuming and prone to human error.*
- *Users cannot reliably assess email safety due to deceptive links and convincing text.*
- *Lack of transparency in many existing detection tools reduces user trust.*

2.2. Project Scope

Phisher Zero *aims to provide an intelligent, automated phishing email detection system using multiple AI agents to analyze email content, URLs, and metadata. The system focuses on accessibility, explainability, and high-accuracy detection.*

Included Functionalities:

- *Chrome Extension interface for scanning emails.*
- *NLP Agent for analyzing email text, tone, and suspicious language patterns.*
- *URL Agent for evaluating links using rule-based and ML-based heuristics.*
- *Ensemble Agent to combine outputs from all agents for final classification.*
- *Explainer Agent to provide human-readable justifications for detection results.*
- *Dashboard for users to view analysis history and flagged emails.*
- *Backend API for secure agent execution and result aggregation.*

Excluded Functionalities:

- *Attachment malware analysis (beyond URL or metadata analysis).*
- *Integration with enterprise-level SIEM/SOC systems (out of scope for FYP).*
- *Automated email sending or email client replacement features.*

2.3. Not In Scope

*Phisher Zero does **not** independently verify or open attachments, perform deep malware scans, or simulate sandboxed execution of files.*
*The system only analyzes **email text, URLs, and metadata**, not full attachment forensics.*

2.4. Project Objectives

***User Functions**:*

- *Scan emails for phishing indicators using the Chrome extension.*
- *View detailed explanation of why an email was flagged.*
- *Review history of previously scanned emails.*

System (Agents) Functions:

- ***NLP Agent:** Evaluate email content for phishing indicators (urgency, threats, impersonation, etc.).*
- ***URL Agent:** Inspect embedded links for obfuscation, domain spoofing, and redirection risks.*
- ***Ensemble Agent:** Aggregate results from individual agents for improved accuracy.*
- ***Explainer Agent:** Generate a clear explanation to help users understand the classification outcome.*

***Admin Functions**:*

● *Maintenance and updates: These cover maintenance, model updates, and dataset refresh cycles..*

● *Api Management: API keys and configuration management are handled internally.*

● *Simple: No dedicated admin panel is required at this stage.*

2.5. Stakeholders

- *End Users (students, faculty, employees, general internet users)*
- *Developers (FYP team)*
- *Jury / Evaluation Committee*
- *Supervisor (Sir Minhal Raza)*
- *Cybersecurity educators and researchers*

2.6. Operating Environment

Phisher Zero will operate within a browser-based environment using a Chrome extension and a cloud/local backend API. The system assumes:

- *Chrome Browser (latest version)*
- *Backend server running Node.js or Python APIs*
- *AI models deployed on local server or edge cloud*
- *Reliable internet connection for URL resolution and API calls*

- *Secure HTTPS communication between extension and backend*

The system may also use third-party safe-browsing APIs (if needed), which must be available and functional.

2.7. System Constraints

- ***Browser Environment:*** *Must run on Google Chrome; compatibility for other browsers is future work.*
- ***Backend Requirements:*** *Node.js or Python server (subject to change in accordance with newer updates) with GPU/CPU resources sufficient for model inference.*
- ***AI Model Constraints:***
  - *NLP agent must avoid bias and adhere to safe AI usage guidelines.*
  - *Detection models must remain explainable and transparent.*
- ***Network Dependency:***
  - *URL agent requires internet access to resolve and analyze links.*
- ***Noisy Environments:***
  - *Extension must use visual cues instead of audible alerts.*
- ***User Interface:***
  - *Needs to be simple, intuitive, and accessible for non-technical users.*
  - *Graphical visualizations (e.g., color-coded risk levels) preferred over heavy textual descriptions.*
- ***Legal & Ethical Constraints:***
  - *Must respect privacy—emails are analyzed locally or securely transmitted with encryption.*
  - *URL verification must avoid violating website terms or triggering harmful actions.*

2.8. Assumptions & Dependencies

- *Users will submit emails or scan messages through the Chrome extension without attempting to bypass or manipulate the detection system.*
- *Users will rely on the system's analysis and will not intentionally provide incorrect or misleading data about emails.*
- *Users have basic familiarity with using browser extensions and simple web interfaces.*
- *The AI agents (NLP, URL, Ensemble, Explainer) operate on the assumption that email text and URLs are available and readable in the scanned content.*
- *Phisher Zero assumes that users have stable internet access when performing URL checks, as certain URL evaluations require external lookups.*
- *The backend environment is properly configured with all required packages, models, and security settings.*

2.9. Dependencies

- ***Browser Environment (Chrome Extension):***
  The system depends on the Chrome extension API to capture email content and trigger agent-based analysis. Any browser update or deprecation of relevant APIs may affect functionality.
- ***Backend Frameworks:***
  The backend relies on frameworks such as ***Node.js/Python****, supporting libraries, and the server environment required to host AI models.*

- ***AI Model Availability:***
  The NLP, URL, and Ensemble models must be available and correctly loaded at runtime (e.g., PyTorch/TensorFlow dependencies).
- ***URL Resolution Services:***
  URL evaluation depends on external DNS resolution or safe-browsing checks, which require uninterrupted network connectivity.
- ***Database System:***
  A database (e.g., MySQL, MongoDB, or Firebase) is required to store logs, previous detections, and system metadata. Its availability and query performance impact the overall system.
- ***Third-Party APIs (Optional):***
  If external safe-browsing or threat-intelligence APIs are used, their availability, rate limits, or downtimes may affect URL risk evaluation.

3. External Interface Requirements

This section outlines the interface requirements for Phisher Zero, ensuring that the system interacts properly with hardware, software, external APIs, communication channels, and browser components. As a browser-integrated AI system, these requirements are crucial for seamless execution across the Chrome extension and backend multi-agent pipeline.

3.1. Hardware Interfaces

Phisher Zero is lightweight and does not require dedicated hardware. The following interfaces apply:

User Device (Browser Client)

- *Must support **Google Chrome (latest version)**.*
- *Minimum hardware specifications:*
    - *4GB RAM*
    - *Dual-core processor*
    - *Windows 10 / macOS 10.13 / Linux kernel 4.x or later*
- *Device must support:*
    - *Local extension storage (scan logs, cached metadata)*
    - *Secure internet connectivity for API calls*

Backend Server / Cloud Environment (Developers end mostly)

- *Hosted locally or on cloud (AWS/GCP/Azure)*
- *Minimum configuration:*
    - *8GB RAM*
    - *Quad-core CPU*
    - *GPU optional for NLP acceleration*
- *Must support:*
    - *Docker containers (backend/agents)*
    - *FastAPI server*
    - *Model inference (URL classifier, ensemble agent)*

No specialized sensors or peripherals are required since detection is purely email-based.

3.2. Software Interfaces

Phisher Zero interacts with multiple software systems and services(subject to change with newer versions).
Browser Extension Interface
Uses Chrome Manifest V3 APIs:

- *chrome.scripting — Inject content scripts for Gmail scanning*

- *chrome.tabs — Read DOM elements*
- *chrome.runtime — Messaging between extension layers*
- *chrome.identity — Gmail API OAuth (optional)*

Backend API Layer
*The backend is built using **FastAPI**. It exposes REST endpoints such as:*

API Endpoint	***Description***
/nlp/analyze	*Send email body → receive semantic phishing score*
/url/check	*Send URLs → receive URL feature analysis + VirusTotal scan*
/ensemble/predict	*Fuse NLP + URL results into final classification*
/explain/generate	*Generate human-friendly justification*

*All data exchange uses **JSON**.*
AI/ML Model Interfaces

- *Gemini API for:*
  - *NLP Agent*
  - *Ensemble Agent*
  - *Explainer Agent*
- *DistilBERT fallback model for offline NLP analysis*
- *scikit-learn URL model trained on PhishTank + OpenPhish features*

External API Interfaces

- ***Gmail API** for directly fetching email bodies (optional mode)*
- ***VirusTotal API** for real-time URL reputation scanning*
- ***PhishTank** and **OpenPhish** for dataset updates*

3.3. Communications Interfaces

Phisher Zero relies on secure communication between browser, backend, and external threat services.
Communication Protocols

- *HTTPS (TLS 1.2+) between:*
  - *Extension ↔ Backend*
  - *Backend ↔ Gemini API*
  - *Backend ↔ VirusTotal API*

Messaging

- *Chrome extension → Backend:*
  - *REST calls using fetch()*
  - *JSON payloads containing text, URLs, metadata*

- *Backend internal messaging:*
    - *Microservice-like orchestration between agents*
    - *Asynchronous processing supported*

Data Security

- *No raw email is sent outside except:*
    - *URL strings (VirusTotal)*
    - *Embeddings/features (Gemini API)*
- *Sensitive data is hashed/tokenized before leaving device*
- *OAuth tokens encrypted before storage*

Error Handling

- *If backend unreachable → "Detection engine offline."*
- *If VirusTotal throttled → URL marked "Unknown Risk."*
- *If Gemini API fails → fallback model activated*

4. Functional Requirements

This section describes detailed functional requirements for Phisher Zero, covering the Chrome extension, backend agents, detection workflow, and admin/user operations.

4.1. Functional Hierarchy

1. Email Scanning (Browser Extension)

1.1 Retrieve email body (DOM/Gmail API)
1.2 Extract URLs + metadata
1.3 Preprocess & send to backend
1.4 Display phishing classification
1.5 Display explainer agent justification

2. NLP Agent (Content Analysis)

2.1 Detect urgency, fear, impersonation patterns
2.2 Identify linguistic red flags (e.g., grammar anomalies, threats)
2.3 Return phishing content score

3. URL Agent

3.1 Extract all URLs from scanned email
3.2 Perform lexical & structural feature extraction
3.3 Check URL using VirusTotal API
3.4 Predict URL safety using ML classifier
3.5 Return URL risk score

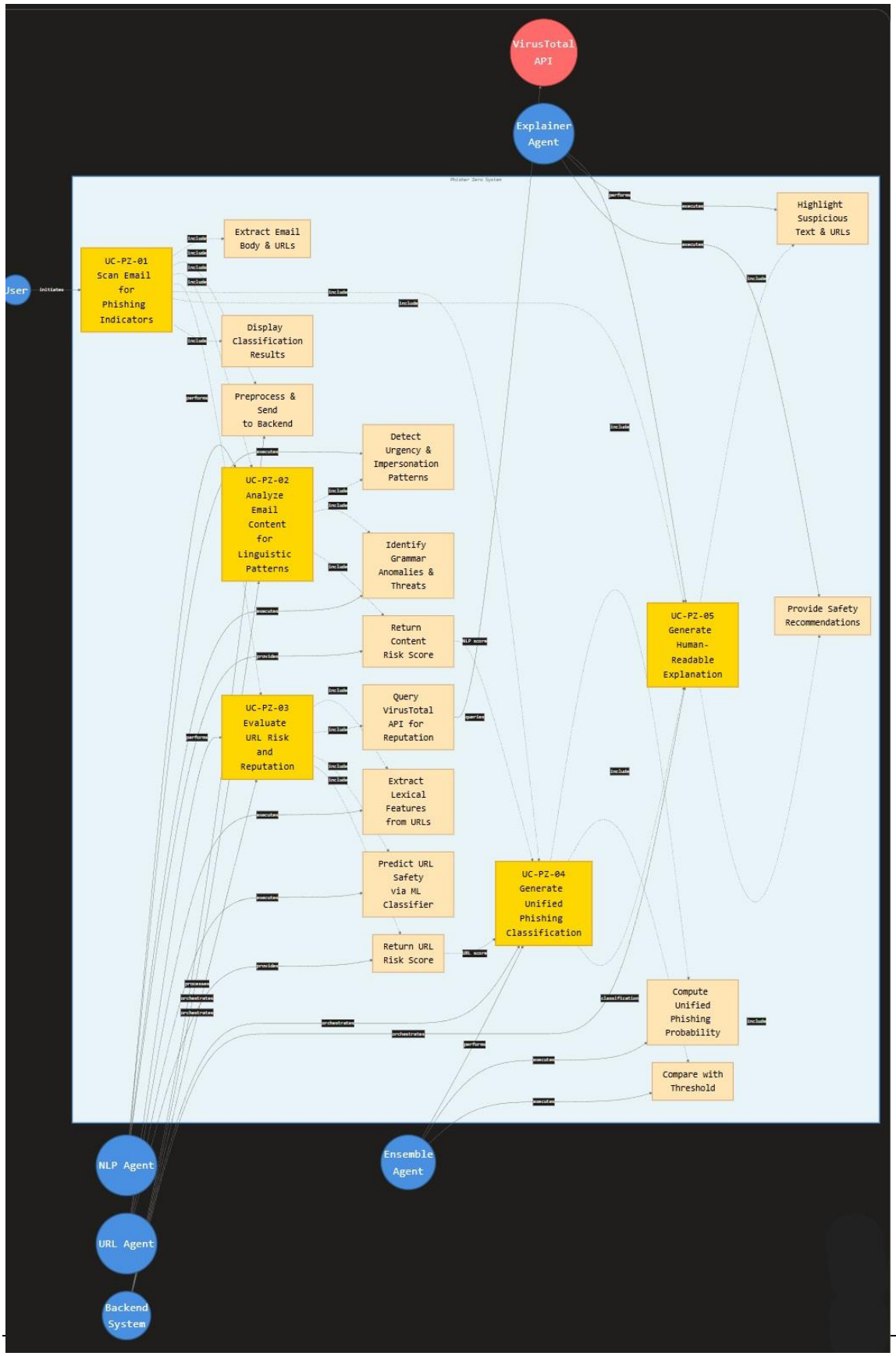
4. Ensemble Agent

4.1 Receive NLP & URL agent scores
4.2 Compute unified phishing probability
4.3 Compare with threshold → final classification

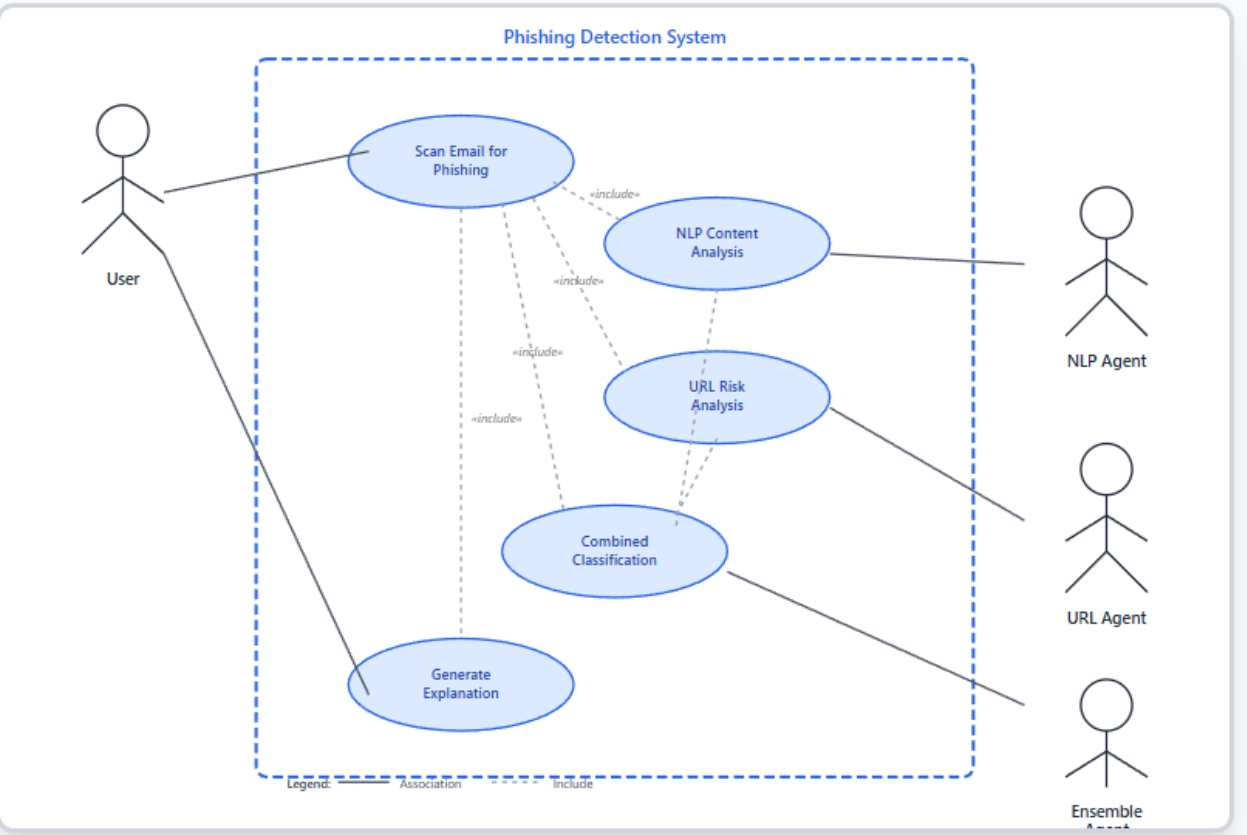
5. Explainer Agent

5.1 Generate natural-language justification
5.2 Highlight specific problematic text/URLs
5.3 Provide user-safe recommendations

4.1.1. Use Case


VirusTotal API
Explainer Agent
Phisher Zero System
User
initiates
UC-PZ-01 Scan Email for Phishing Indicators
Extract Email Body & URLs
Display Classification Results
Preprocess & Send to Backend
UC-PZ-02 Analyze Email Content for Linguistic Patterns
Detect Urgency & Impersonation Patterns
Identify Grammar Anomalies & Threats
Return Content Risk Score
UC-PZ-03 Evaluate URL Risk and Reputation
Query VirusTotal API for Reputation
Extract Lexical Features from URLs
Predict URL Safety via ML Classifier
Return URL Risk Score
UC-PZ-04 Generate Unified Phishing Classification
UC-PZ-05 Generate Human-Readable Explanation
Highlight Suspicious Text & URLs
Provide Safety Recommendations
Compute Unified Phishing Probability
Compare with Threshold
NLP Agent
URL Agent
Backend System
Ensemble Agent

Simplified Use-case Diagram:



<UC-PZ-01>		
Use case Id:	***UC-PZ-01***	
Actors:	***User**: Initiates the use case by installing the extension into the system then scanning emails.*	
Feature:	*Full phishing detection pipeline*	
Pre-condition:	• *Extension installed and active* • *Backend API reachable* • *Email content accessible*	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	*User opens an email*	*Extension detects new email context*
2.	*User clicks "Scan Email"*	*Extension extracts body, URLs, metadata*
3.	*Data sent to backend*	*Backend preprocesses email*
4.	*NLP Agent processes text*	*Returns content-risk score*

5.	*URL Agent analyzes URLs*	*Returns URL-risk score*
6.	*Ensemble Agent merges results*	*Produces final classification*
7.	*Explainer Agent generates narrative*	*Justification created*
8.	*Results shown in UI*	*Warning banner + explanation displayed*

Alternate Scenarios:

- *No internet → Local error notification*
- *3a: Empty email content → Show "Nothing to scan*

Post Conditions

Step#	***Description***
1	*Email is classified as Safe / Suspicious / Phishing*

Use Case Cross referenced	• **UC-PZ-02: NLP Content Analysis** • **UC-PZ-03: URL Risk Analysis** • **UC-PZ-04: Ensemble Prediction** • **UC-PZ-05: Explanation Generation**

< UC-PZ-02>		
Use case Id:	*UC-PZ-02*	
Actors:	***NLP Agent***	
Feature:	*NLP-Based Email Content Analysis*	
Pre-condition:	• Backend receives email body text. • NLP model must be loaded (Gemini or fallback model).	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	*Backend sends email body to NLP Agent.*	*NLP agent receives text input.*
2.	*NLP Agent processes linguistic features (urgency, impersonation, tone).*	*Generates semantic feature vector.*
3.	*NLP Agent predicts phishing content probability.*	*Returns risk score (0–1).*
Alternate Scenarios:		
1a: If input text is empty → returns: "Insufficient content for analysis". ***3a: If Gemini API is down → DistilBERT fallback model is automatically used.***		
Post Conditions		
Step#	***Description***	
1.	*NLP score is forwarded to the Ensemble Agent.*	
Use Case Cross referenced	• **UC-PZ-01: Main Email Scan** • **UC-PZ-04: Ensemble Prediction**	

< UC-PZ-03>	
Use case Id:	*UC-PZ-03*
Actors: ***URL Agent*** ***VirusTotal API (external)***	
Feature:	*URL Risk Analysis*
Pre-condition:	URL(s) extracted from email. VirusTotal API key configured. URL ML classifier must be loaded.
Scenarios	

Step#	Action	Software Reaction
1.	Backend sends list of URLs to URL Agent.	URL Agent begins processing.
2.	URL Agent extracts lexical features (entropy, hyphens, domain length).	Prepares URL feature vector.
3.	URL Agent sends URL to VirusTotal API.	Retrieves reputation (malicious/clean/suspicious).
4.	URL ML model predicts risk.	Returns URL risk score (0–1).
5.	Final URL risk summary sent to Ensemble Agent.	Results used for fusion
Alternate Scenarios:		
1a "No URLs detected" → Skips URL analysis. **3a VirusTotal API unreachable → URL marked as Unknown Risk.** **4a URL classifier fails → URL result defaults to reputation only.**		
Post Conditions		
Step#	**Description**	
1.	URL risk scores forwarded to Ensemble Agent.	
Use Case Cross referenced	**UC-PZ-01** **UC-PZ-04**	

< UC-PZ-04 >		
Use case Id:	UC-PZ-04.	
Actors: Ensemble Agent		
Feature: Combined Phishing Classification		
Pre-condition:	NLP score available. URL score(s) available. Ensemble algorithm configured.	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Backend passes NLP + URL scores to Ensemble Agent.	Fusion module loads inputs.
2.	Ensemble Agent applies weighted aggregation.	Computes final phishing probability.
3.	Generates final classification (Safe/ Suspicious/Phishing).	Sends result to Explainer Agent.
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a If one agent fails → fusion uses available inputs only. **2a If weighting config invalid → fallback to default weights.**		

Post Conditions	
Step#	***Description***
1.	*Final classification stored and returned to user interface.*
Use Case Cross referenced	UC-PZ-01 UC-PZ-02 UC-PZ-03 UC-PZ-05

< UC-PZ-05 >		
Use case Id:	*UC-PZ-05*	
Actors: ***Explainer Agent*** ***User***		
Feature:	*Explanation Generation*	
Pre-condition:	Final phishing classification available. Explanation model (Gemini) must be active.	
Scenarios		
Step#	***Action***	***Software Reaction***
1.	*Ensemble passes classification + agent outputs to Explainer Agent.*	*Explanation model receives structured data.*
2.	*Explainer Agent generates justification.*	*Highlights suspicious text & URLs.*
3.	*System displays explanation to user.*	*Explanation appears in popup/bottom sheet.*
Alternate Scenarios:		
2a ***If explanation generation fails → Displays "Basic Score Only: Explanation Unavailable."***		
Post Conditions		
Step#	***Description***	
1.	*User receives clear reasoning for classification.*	
Use Case Cross referenced	UC-PZ-01 UC-PZ-04	

5. Non-functional Requirements

5.1. Performance Requirements

- *The system must process phishing-detection requests from the Chrome extension with **no noticeable lag**, supporting **at least 10+ concurrent users** during peak usage*
- *The Chrome extension must display the final phishing classification **within 7–10 seconds** after the user clicks "Scan Email," under normal network conditions.*
- *NLP analysis using the Gemini API or the fallback DistilBERT model must return a phishing-intent score within **5 seconds** on average.*
- *URL Agent operations including lexical analysis, model inference, and optional VirusTotal queries must complete within **7 seconds** collectively.*
- *Ensemble Agent fusion (NLP + URL scoring) must compute the final probability within **1 second**.*
- *The Explainer Agent must generate a human-readable justification within **2–4 seconds**, depending on model availability.*
- *System uptime must be maintained at **99% or higher**, ensuring uninterrupted availability for phishing-scan operations.*
- *Backend API response time must remain under **500 ms** for lightweight calls (metadata lookup).*
- *Total end-to-end processing time for a complete email scan must not exceed **10 seconds**, except when third-party APIs (e.g., VirusTotal) introduce external delays.*

5.2. Safety Requirements

- *The system must prevent unauthorized modification, deletion, or tampering of critical data such as scan data, model configurations, or user-sensitive data.*
- *The system must include error-handling mechanisms to avoid crashes caused by malformed emails, broken URLs, or unexpected user actions.*
- *Fail-safe protocols must ensure that if one agent fails (e.g., Gemini API down), the system falls back gracefully—never producing misleading or incomplete results.*
- *The URL Agent must not interact with URLs in a way that triggers harmful actions (e.g., opening malicious sites); analysis must remain static and passive.*
- *The system must not expose sensitive email content to any third-party services except where strictly necessary and privacy-preserving (e.g., hashed/tokens instead of raw data).*
- *Logging must exclude any personally identifiable information (PII) from emails to ensure safety, privacy, and ethical data handling.*
- *Fail-open conditions must not occur; if an analysis component fails, the system should return a "Scan Incomplete" or "Unknown Risk" status rather than marking the email safe.*

5.3. Security Requirements

- *All communication between Chrome extension ↔ Backend ↔ AI models and external APIs must occur over HTTPS (TLS 1.2+).*
- *Sensitive data such as tokens, logs, OAuth credentials, and API keys must be encrypted using industry-standard encryption (e.g., AES-256).*
- *Email content transmitted to the backend must be encrypted in transit and stored only in hashed or tokenized form where possible.*
- *AI model queries (e.g., to Gemini) must send embeddings or sanitized text, avoiding full email transmission whenever feasible.*
- *URL evaluation requests to external services (e.g., VirusTotal) must not include sensitive or user-identifying metadata.*
- *The system must defend against cyber attacks such as Phishing.*
- *The system must strictly follow privacy and ethical AI guidelines to ensure optimal user data protection.*

5.4. User Documentation

- *A detailed **User Manual** must be provided to guide users through installation, scanning, viewing results, understanding risk levels, and using the extension.*
- ***Tutorial Videos will demonstrate end-to-end email scanning, interpreting phishing results, and understanding explanation reports**.*
- ***Context-Sensitive Help** must automatically display relevant guidance based on the user's active screen or action—for example, while viewing analysis details.*
- *A **Troubleshooting Guide** will help users resolve common issues such as backend offline errors, API failures, missing email content, or authentication problems.*
- *Developer documentation. (for internal team use).*

6. References

- ***Journal Article***

  Z. Jafari, S. H. Ghafoori, and M. Ahmadinia, "Smart phishing detection on webpages using multi-agent deep learning and multi-dimensional features," International Journal of Smart Electrical Engineering, vol. 14, no. 1, pp. 33–44, 2025. [Online]. Available: https://oiccpress.com/ijsee/article/view/17130

- ***Conference / Preprints / Technical Papers***

  B. Lim, J. Zhang, S. Thomas, and J. Ko, "EXPLICATE: Enhancing phishing detection through explainable AI and LLM-powered interpretability," arXiv preprint, arXiv:2503.20796, 2025. [Online]. Available: https://arxiv.org/abs/2503.20796

  Y. Xue, W. Chen, R. Lee, and P. Liu, "MultiPhishGuard: An LLM-based multi-agent system for phishing email detection," arXiv preprint, arXiv:2505.23803, 2025.[Online]. Available: https://arxiv.org/abs/2505.23803

  J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in NAACL-HLT 2019, pp. 4171–4186, 2019.[Online]. Available: https://doi.org/10.18653/v1/N19-1423

- ***Government / Policy / Industry Reports***

  Verizon, "2023 Data Breach Investigations Report," Verizon Enterprise, 2023. [Online]. Available: https://www.verizon.com/business/resources/reports/dbir/

- ***Website***

  Google, "Safe Browsing: Protecting users from phishing," 2023. [Online]. Available: https://safebrowsing.google.com

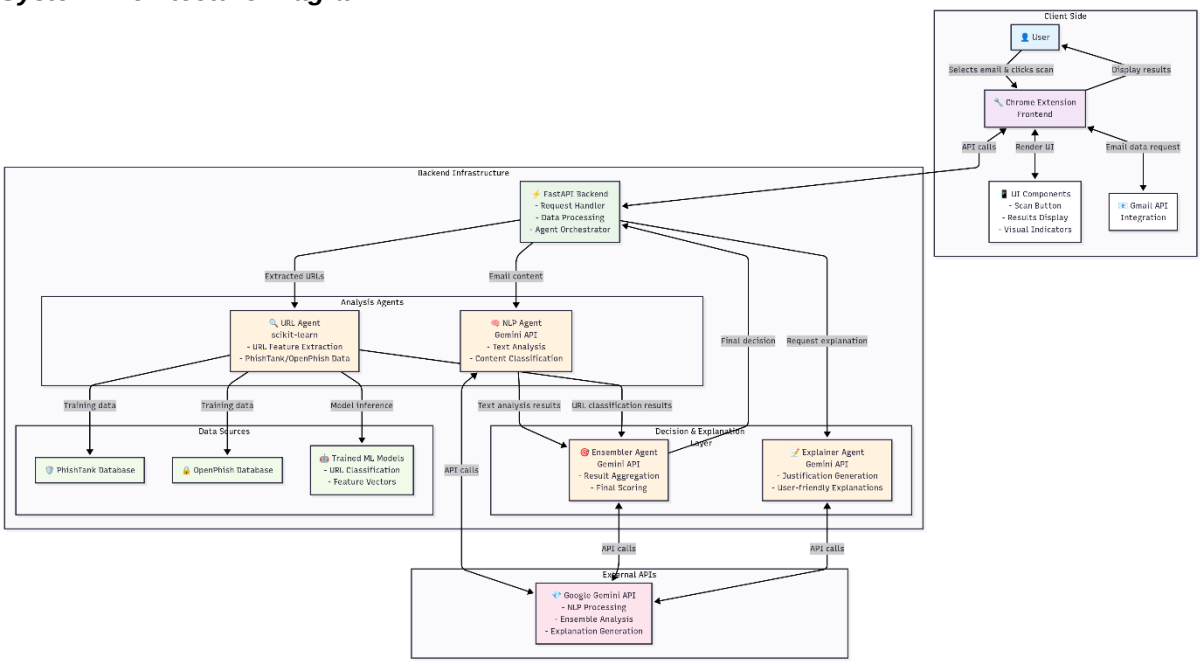
  Google, "Gemini API documentation," 2024.[Online]. Available: https://ai.google.dev/gemini-api/docs

- ***Classic / Foundational Sources***

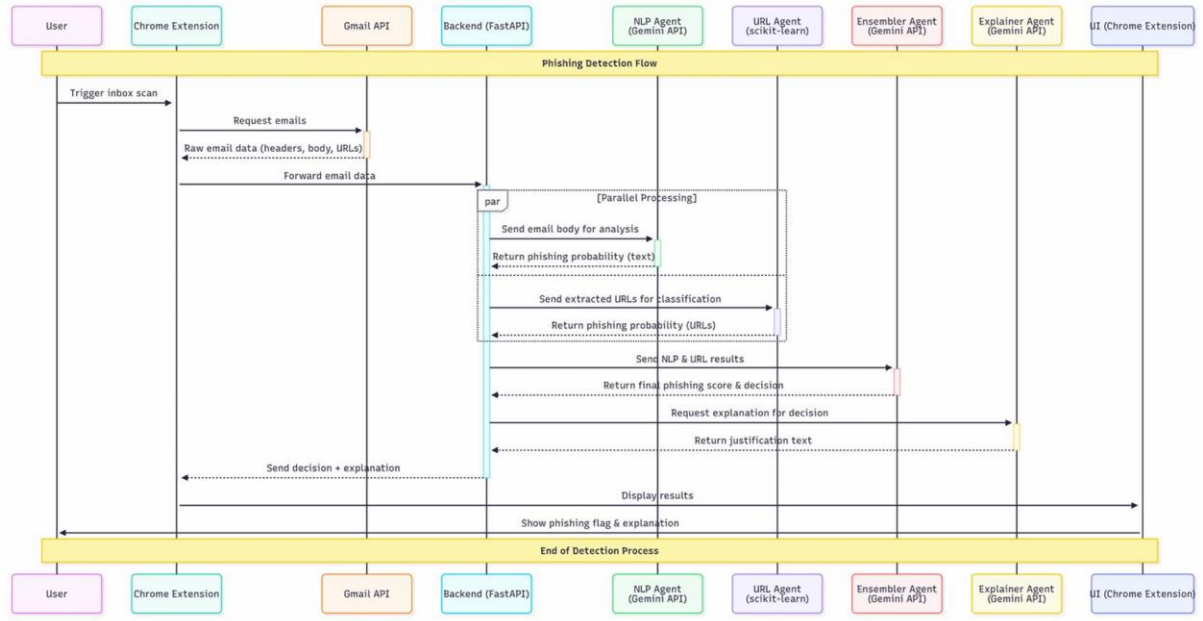
  T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer, "Social phishing," Communications of the ACM, vol. 50, no. 10, pp. 94–100, 2007. [Online]. Available: https://doi.org/10.1145/1290958.1290968

7. Appendices

- **Appendix A**
  System Architecture Diagram



- **Appendix B**
  Sequence Diagram



Appendix C
Activity Diagram: