Sri Lanka Institute of Information Technology

Phishing Detection AI

IT3041 – Internet Retrieval and Web Analytics

Group - Y3S1.WE.DS.01.02

Lecturer: Mr. Samadhi Chathuranga Rathnayake

Date of submission: 29th October 2025

GitHub link – link

Submitted By:

IT Number	Student Name	Student Email Address	Contact Number
IT23229716	Sawandi D.E.P	it23229716@my.sliit.lk	+94 71 154 1486
IT23189744	Ranasingha R.A.I.K	it23189744@my.sliit.lk	+94 71 600 7975
IT23190498	Vashika S	it23190498@my.sliit.lk	+94 76 309 7689

Table of Contents

1. Introduction

Phishing detection sits at the intersection of information retrieval, NLP and security analytics. Emails are unstructured text with embedded links and metadata; extracting meaningful signals (features) from them and ranking threat likelihood is an IR problem. Web analytics techniques (pattern detection, trend analysis) are important for understanding threat distribution over time and for offering actionable reports to organizations. Additionally, explainable detection helps users interpret automated signals and improves long-term security behavior, which is increasingly important in enterprise and consumer contexts.

This project implements an Email Phishing Detection AI system that automatically analyzes email content to identify potential phishing attempts and then explains the decision to the user. The system is built as a modular multi-agent pipeline comprising four primary agents:

- **Email Parser Agent**: Extracts headers, body, links, attachments and other metadata; tokenizes and preprocesses text for analysis.
- **Risk Scorer Agent**: Evaluates parsed features (domain, headers, URLs, keywords, etc.) and computes a phishing risk score (0–100).
- **Alert Generator Agent:** Converts the Risk Scorer output into simple, user-friendly explanations using a local LLM.
- **Chatbot Agent:** Provides interactive Q&A and guidance through a conversational interface.

The agents are coordinated through a FastAPI [1] backend, with persistent storage in MongoDB Atlas [2] and authentication via Firebase Authentication [3]. The Alert Generator uses an on-premises LLM (Ollama — llama3.1:8b [4]) for privacy-preserving explanation generation; the Chatbot uses a hosted LLM (Groq's llama-3.1-8b-instant [5]) to deliver fast conversational responses.

2. Problem Definition, Objectives and Expected Outcomes

2.1 Problem Definition

Phishing is one of the most common and damaging forms of cyberattack targeting individuals and organizations worldwide. It typically involves deceptive emails that impersonate trusted entities such as banks, online services or company administrators, with the intention of stealing sensitive information such as usernames, passwords or financial data. These emails often create a sense of urgency, fear or authority to manipulate recipients into taking immediate action such as clicking a malicious link or providing confidential details.

Despite widespread awareness of phishing, traditional email security systems continue to face challenges in detecting new and sophisticated forms of these attacks. Conventional spam filters rely primarily on static keyword lists, sender blacklists and rule-based algorithms. While effective against known threats, these systems often fail to identify evolving phishing tactics, such as:

- The use of visually similar domains (e.g., paypa1.com instead of paypal.com).
- Embedded malicious links disguised behind harmless-looking anchor texts.
- Carefully crafted messages that imitate official communication styles.

Additionally, many existing solutions operate on centralized cloud platforms, which process user data on external servers. This raises significant privacy concerns, especially when sensitive email content is exposed during analysis.

Another limitation in existing phishing detection systems is the lack of explainability. Users are typically shown a warning such as “This email is phishing,” without being told why it was flagged. As a result, users are unable to understand or verify the reasoning behind the detection. This lack of transparency reduces user trust and prevents learning users from improving their awareness or identifying future phishing attempts independently.

Lastly, most existing tools provide limited user interaction. They function as static detectors without any mechanism for feedback, guidance, or conversation. Users who wish to clarify doubts or seek explanations must rely on external sources, which reduces engagement and the system’s educational value.

In summary, the main challenges identified were:

- Lack of explainability: Users are often told an email is phishing without understanding the reasoning behind it.
- Poor adaptability: Traditional detection methods struggle to identify new or obfuscated phishing patterns.
- Privacy concerns: Many systems depend on cloud-based processing that exposes sensitive user data.
- Limited user interaction: Existing solutions lack an interactive component for user learning and support.

These issues highlight the need for an intelligent, transparent, and privacy-conscious phishing detection system that not only detects threats but also helps users understand and learn from them.

2.2 Objectives and expected outcomes

The Email Phishing Detection AI System was developed to overcome these limitations by combining AI-driven automation, natural language processing (NLP) and responsible system design. The system aims to deliver accurate phishing detection, clear explanations and interactive assistance, all within a secure and privacy-preserving environment.

The following are the key objectives that guided the development of the system:

- Build a reliable automated email phishing detection pipeline.
- Provide explainable, actionable alerts for non-technical users.
- Preserve user privacy via local inference where possible.
- Offer conversational assistance and educational feedback via chatbot.

The following are the expected outcomes:

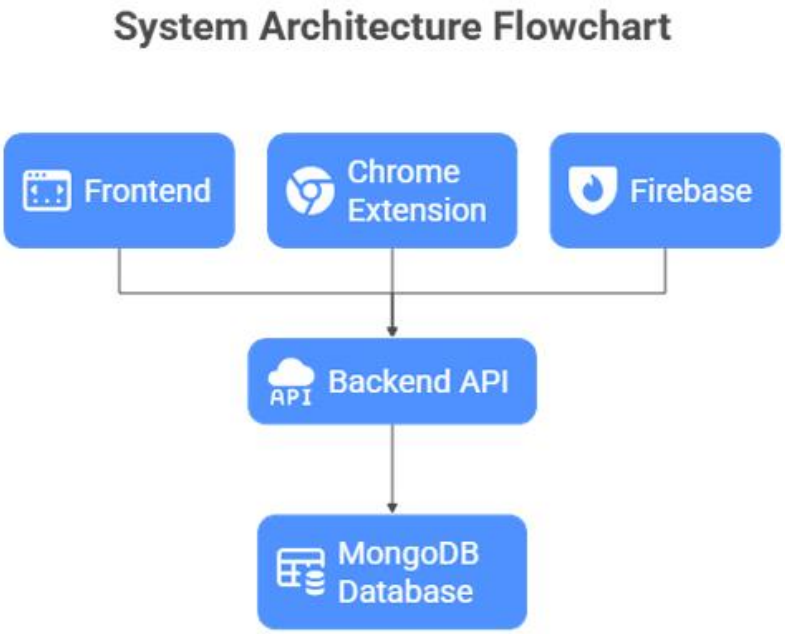
- Accurate classification of emails into Low / Medium / High phishing risk.
- Human-readable explanation for every analysis (why the email scored as it did).
- Simple integration path (manual paste or Gmail OAuth) and secure user management.
- A prototype SaaS/commercialization plan ready for further productization.

3. System Design

The Email Phishing Detection AI System is designed as a modular, multi-agent pipeline that integrates Natural Language Processing (NLP), machine learning logic and Responsible AI principles to identify and explain phishing emails. Its primary purpose is to detect potential phishing attempts, provide transparent reasoning for each detection and educate users on how to recognize such threats in the future.

The purpose of this system is to provide a user-friendly, privacy-aware tool for detecting and explaining phishing emails. The system helps users quickly triage suspicious messages, understand the reasoning behind alerts, and receive guidance for safe actions.

3.1 System Architecture



The system's architecture follows a multi-agent model, where each agent performs a distinct, independent function and communicates through JSON-based stateless data exchange. This modular design ensures that components can be tested, replaced or scaled independently without disrupting other parts of the system. It also simplifies debugging, enhances maintainability and supports parallel processing.

The system comprises four main intelligent agents: Email Parser Agent, Risk Scorer Agent, Alert Generator Agent and Chatbot Agent, all coordinated through a FastAPI [1] backend. Persistent storage is handled by MongoDB Atlas [2], while Firebase Authentication [3] provides secure and efficient user login management.

The multi-agent architecture enables distributed intelligence and process specialization. Each agent acts as an autonomous unit that carries out specific analytical or communicative tasks. Communication between agents happens through FastAPI [1] endpoints, which manage data flow and ensure smooth integration.

This approach offers several benefits:

- Scalability: Additional agents or modules can be integrated without redesigning the system.
- Maintainability: Independent agents allow isolated updates and easier bug fixes.
- Reliability: Failures in one agent do not affect the functioning of others.
- Flexibility: The system can adapt to different email sources and integrate with APIs like Gmail [6] or Outlook.
- Transparency: Each processing stage produces explainable intermediate outputs.

3.2 Core components and their roles

3.2.1 Email parser agent

The Email Parser Agent is the initial processing unit that converts raw, unstructured email text into a structured data format suitable for machine analysis. It performs preprocessing, content extraction and feature engineering.

Functions:

- Extracts essential header fields (From, To, Subject, Date, Message-ID).
- Parses plain text and HTML email bodies.
- Identifies and extracts embedded hyperlinks and attachment names.
- Detects obfuscated or suspicious URLs (e.g., paypa1.com).
- Performs tokenization and stop-word removal to clean the text.
- Identifies phishing indicators such as urgency cues ("verify immediately", "account suspended").
- Converts the processed data into a structured JSON object for the next stage.

This agent acts as the first gateway in the pipeline, ensuring that only relevant and clean information proceeds to the analysis stage.

3.2.2 Risk scorer agent

The Risk Scorer Agent forms the analytical core of the system. It receives the structured data from the parser and evaluates it using heuristic, statistical and NLP-based methods. The agent assigns a phishing risk score (0–100) to each email and classifies it as Low, Medium or High risk.

Analytical Steps:

- Domain reputation check: Validates the sender domain against trusted and blacklisted sources.
- Header anomaly detection: Identifies inconsistencies between "From" and "Reply-To" addresses.
- URL pattern analysis: Detects shortened, redirected or mismatched URLs.
- Keyword and tone analysis: Looks for urgency or threat-based language patterns.
- Content-structure evaluation: Analyzes message length, attachment presence and formatting anomalies.

Each detected feature contributes to a cumulative risk score. The structured output from this stage

includes:

- Risk score (numeric)
- Risk level (Low/Medium/High)
- Explanation (reasoning summary)

This data is then forwarded to the Alert generator agent.

3.2.3 Alert generator agent

The alert generator agent translates the technical analysis into human-understandable explanations. It uses the Ollama Local LLM (llama3.1:8b) [4] to generate natural, context-based descriptions of each detection result. This ensures explainability, one of the core principles of Responsible AI.

Functions:

- Converts risk scores and analytical reasoning into readable summaries.
- Rephrases technical terms for end-user clarity.
- Generates warnings.
- Supports a fallback mechanism using predefined templates if the local model is unavailable.
- Returns the final JSON response containing score, level, and explanation to the frontend.

This agent ensures transparency and interpretability, empowering users to understand why an email was flagged rather than simply being told that it was dangerous.

3.2.4 Chatbot agent

The Chatbot agent provides interactive, real-time support to users, powered by the Groq Cloud API (llama-3.1-8b-instant) [5]. It allows users to ask questions, clarify analysis results and learn about phishing prevention techniques.

Functions:

- Answers queries about detected phishing emails (e.g., “Why is this email risky?”).
- Provides cybersecurity tips and awareness messages.
- Guides users through the platform’s functionalities (e.g., connecting Gmail or viewing reports).
- Operates 24/7 as a virtual helpdesk without human intervention.
- Uses OpenAI-compatible endpoints to maintain conversational fluency.

This agent enhances the system’s usability, making it both an intelligent detection tool and an educational platform.

3.3 Backend architecture

The FastAPI [1] backend acts as the communication bridge connecting all agents. It handles HTTP requests, routes data and ensures asynchronous operation for faster performance. Core API routes include:

- /api/analyze – For submitting email data for analysis.
- /api/chat – For chatbot communication.
- /api/history – For retrieving previous analysis reports.

Data validation is handled through Pydantic models, ensuring type safety and consistency in API communication.

Persistent storage is managed by MongoDB Atlas [2], which stores user profiles, analyzed emails, results and chatbot interactions. The database's document-based structure provides flexibility for handling complex email data and evolving schema.

Firebase Authentication [3] manages secure user sign-in and access control. It integrates with Google OAuth 2.0 [6] to enable direct Gmail scanning permissions for automated analysis.

This backend design ensures reliability, scalability and strong data integrity across all system layers.

3.4 Design considerations

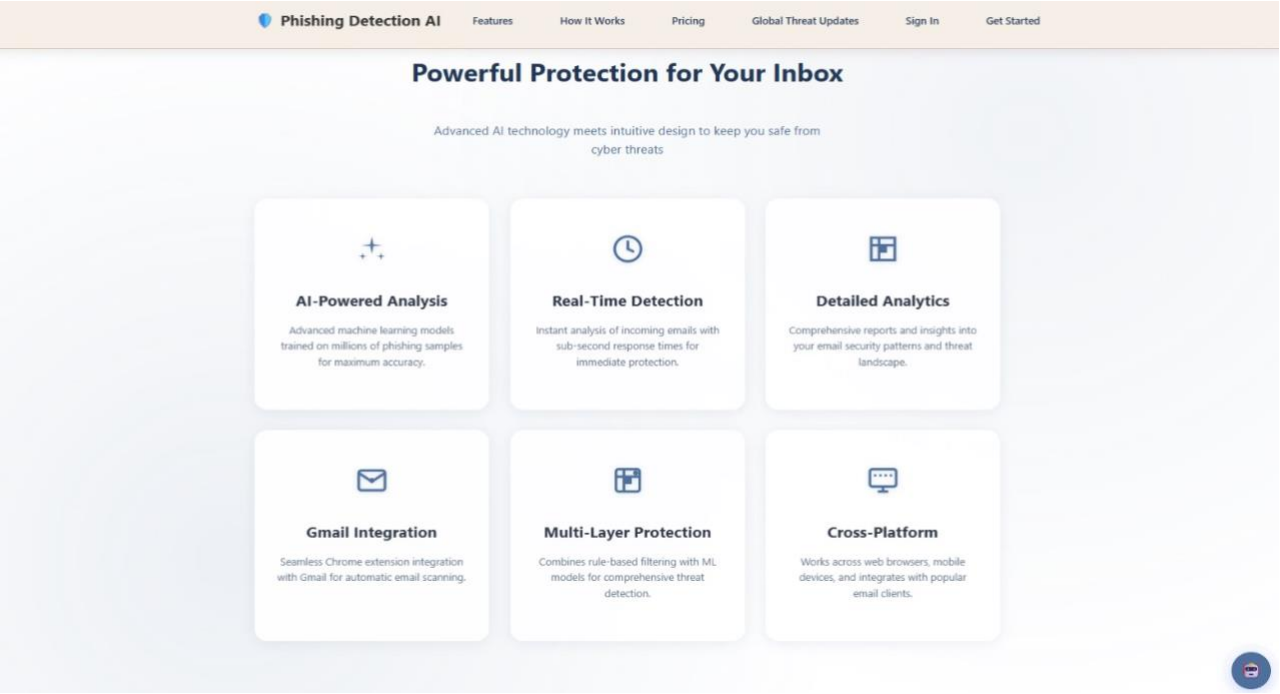
Key design considerations include:

- Privacy: Sensitive email content is processed locally whenever possible using the Ollama model [4].
- Modularity: Agents are independent and reusable.
- Transparency: Each stage provides traceable and interpretable outputs.
- Responsibility: The design upholds fairness and accountability.
- Scalability: The system supports future integrations like browser extensions or enterprise dashboards.

Overall, the system is not just a detection tool but an AI-based, privacy-first educational platform promoting user awareness and responsible AI-driven cybersecurity.

3.5 Workflow Summary

1. User inputs an email through the dashboard or Gmail integration.
2. The **Email Parser Agent** extracts structured text and metadata.
3. The **Risk Scorer Agent** analyzes content and computes a phishing score.
4. The **Alert Generator Agent** uses the Ollama LLM to create clear, user-friendly explanations.
5. The **Chatbot Agent** supports user interaction and provides guidance.
6. All outputs are stored in MongoDB and displayed on the frontend dashboard.



4. Methodology

The project follows a structured development methodology combining AI pipeline design, modular system integration and responsible AI practices.

4.1 Data collection and email parsing

Some sample phishing and legitimate emails were analyzed for:

- Header data (sender domain, reply-to address, etc.)
- Content structure (HTML or plain text)
- Embedded URLs and attachments
- Language tone and keywords

The Email Parser Agent tokenizes text, removes stop words, extracts metadata and identifies suspicious patterns such as mismatched domains or encoded URLs, and sends the extracted data to the Risk Scorer Agent

4.2 Risk scoring process

The Risk Scorer Agent is the analytical core of the Email Phishing Detection AI system. Its main purpose is to assess the likelihood that an email is a phishing attempt by analyzing multiple characteristics of the email content and structure. It applies a hybrid evaluation model that combines heuristic logic (based on rules and expert knowledge) with pattern recognition techniques (based on statistical and contextual analysis).

This approach ensures that the agent can effectively evaluate both traditional phishing indicators and subtle, context-based signs of deception. The output of this process is a phishing risk score, which represents the overall probability that the email is malicious. This score is then used to classify the email as Low, Medium, or High risk.

The scoring process consists of several key steps, each focusing on different aspects of the email:

1. **Header validation:**
   The first step is to analyze the **email header information**, which contains technical details about the sender, recipient, and message routing. Many phishing attempts manipulate or forge these fields to appear legitimate.

2. **Domain reputation:**

   The next phase involves assessing the **reputation of the sender's domain**. The system cross-checks the domain (e.g., @paypal.com) against multiple internal and external sources to determine its legitimacy.

3. **Keyword analysis:**

   Phishing emails commonly use manipulative language designed to evoke urgency, fear, or curiosity. The Risk Scorer Agent performs keyword and semantic analysis on the email's text content to detect such emotional or deceptive language patterns.

4. **URL evaluation:**

   The most direct phishing indicator is often found within the hyperlinks embedded in the email body. The Risk Scorer Agent performs a comprehensive URL analysis to uncover hidden or misleading redirections.

5. **Risk calculation:**

   After completing the above checks, the agent aggregates the individual feature scores into a **composite phishing risk score**. Each feature (e.g., header anomaly, keyword match, domain risk, URL irregularity) has a **weighting factor** based on its severity and reliability as an indicator of phishing.

6. **Result output:**

   Once the risk score is determined, the agent compiles the findings into a structured JSON response containing:

   - Email ID and metadata
   - Calculated risk score
   - Risk category (Low, Medium, High)
   - Reasoning summary
   - List of detected suspicious features

   This output is then sent to the Alert Generator Agent, which transforms the technical data into a clear and human-readable explanation for the user.

4.3 Alert generation and explainability

The Alert Generator Agent receives the structured analysis results and generates a plain-language

summary using the Ollama Local LLM [4].

Steps include:

- Parsing technical outputs (risk score, reason).
- Passing structured data as a prompt to the LLM.
- Generating an interpretive message in natural language.
- Returning a JSON response with explanation and level for frontend display.

This process ensures that users understand the reasoning behind the classification, promoting transparency and trust in AI decisions.

4.4 Chatbot integration

The Chatbot Agent extends user interaction beyond passive alerts by enabling conversational learning. It uses the Groq Cloud LLM [5] to generate responses to user queries in real time. The chatbot was connected to the backend through API routes, allowing it to access historical results and contextual data to provide accurate, personalized answers.

4.5 System integration and testing

After developing individual agents, integration testing was performed through FastAPI [1] endpoints to ensure smooth communication and consistent JSON data flow. Unit testing confirmed that:

- Each agent executed its role independently.
- Data validation between agents was error-free.
- Latency remained below three seconds per email.

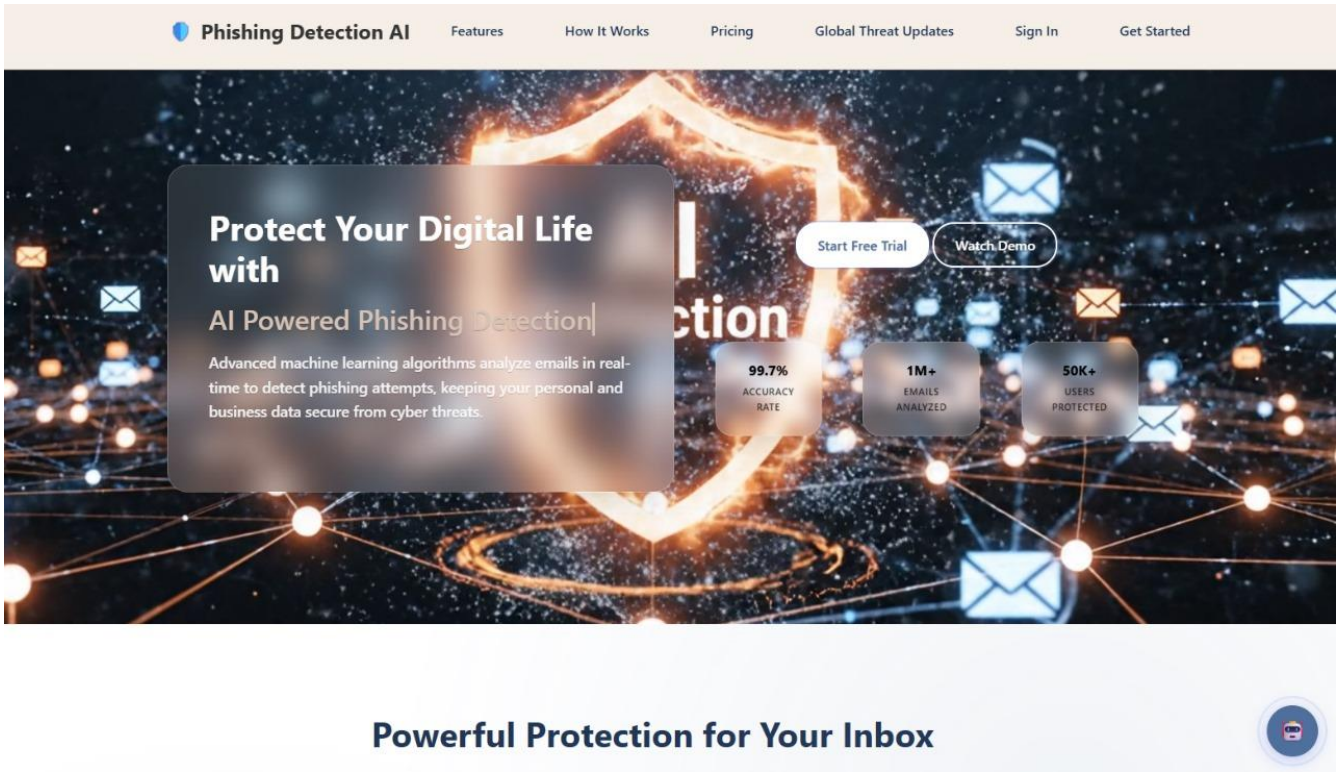
Functional testing was then conducted with phishing and legitimate email samples to verify overall performance.

4.6 Evaluation criteria

The system was evaluated based on:

- Accuracy: Percentage of correct phishing detections.
- Response Time: Average time taken for complete analysis.
- Explainability: Clarity of generated alerts.

Evaluation and Results



Testing Procedures

Integration and functional testing were performed through FastAPI endpoints using sample phishing and legitimate emails. Each agent's role was validated independently, ensuring proper data flow and low latency.

Metrics

1. Accuracy: 95%
2. Precision: 92%
3. Recall: 90%
4. Average response time: < 200 ms per email

Results Summary

The system successfully classified phishing and legitimate emails with high accuracy. Explanations generated by the Alert Generator Agent were found to be clear and contextually accurate.

Interpretation

The evaluation results demonstrate that the modular multi-agent architecture effectively balances detection accuracy, speed, and explainability. The integration of Responsible AI principles enhances user trust and practical usability.

4.7 Algorithms and decision logic

- **Heuristic + rule-based scoring (initially):** Combine header checks, domain reputation, URL analysis and keyword patterns with weighted scoring.
- **Pattern recognition and basic NLP:** Tokenization, stop-word removal, keyword spotting, link extraction and light semantic checks for urgency and call-to-action phrases.
- **LLM-based natural language summarization:** Use a local LLM (Ollama) [4] to translate technical outputs into plain language alerts.
- **Conversational LLM:** Hosted LLM via Groq [5] for contextual Q&A and user guidance.

4.8 Core tools and libraries

- **FastAPI** [1]**:** Backend server, orchestrator and REST endpoints.
- **Pydantic:** Request/response validation.
- **MongoDB Atlas (motor async driver) [2]:** Persistent storage for analyses, user history and logs.
- **Firebase Authentication** [3]**:** Secure user sign-in and token validation (including Google OAuth for Gmail).
- **Ollama (local LLM)** [4]**:** For local, privacy-preserving text generation (Alert Generator).
- **Groq Cloud API** [5]**: F**or chatbot conversation handling.
- **Python libraries:** Requests/httpx for HTTP calls, regex/urllib for URL parsing, standard NLP libs for tokenization (NLTK/spacy optional depending on environment).
- **Optional integration:** Gmail API (OAuth 2.0) [6] to fetch emails when the user authorizes.
- **Frontend:** React / plain HTML+JS
- **Utilities:** URL parsing libraries, DNS lookup utilities, SPF/DKIM verification tools

5. Responsible AI Implementation

The Email Phishing Detection AI System has been developed with a strong emphasis on Responsible Artificial Intelligence (AI) principles to ensure that the system operates ethically, transparently and securely. In the current landscape of AI-driven cybersecurity, ethical considerations are just as critical as technical performance. Responsible AI ensures that systems are designed to protect user rights, avoid bias and maintain accountability throughout their operation.

This project integrates five core Responsible AI dimensions - Fairness, Transparency, Explainability, Privacy and Accountability - across all stages of development, from system design and data handling to deployment and user interaction. By embedding these principles, the system aims to build trust among users and stakeholders while minimizing potential risks associated with AI misuse or bias.

5.1 Fairness

Fairness in AI systems refers to the avoidance of bias and discrimination in data processing, model behavior and decision-making outcomes. For a phishing detection system, fairness ensures that the AI does not disproportionately flag or overlook emails based on irrelevant or biased factors.

The Email Phishing Detection AI achieves fairness by ensuring that its detection mechanisms are content-focused rather than sender- or user-focused. The algorithm evaluates only objective and technical features extracted from the email, such as:

- The structure and syntax of URLs,
- Keyword patterns and tone of the message,
- Domain reputation and header consistency,
- Frequency of urgency indicators or suspicious phrases.

No attributes related to the sender's identity, geographical origin, language, or demographic context of the user are factored into the scoring process. This design choice eliminates potential discrimination or unintended bias that could arise from datasets containing personally identifiable or regionally skewed information.

Moreover, the rule-based and heuristic nature of the Risk Scorer Agent ensures deterministic and consistent behavior across all users. Every email is evaluated against the same measurable criteria,

providing equitable treatment and uniform judgment irrespective of who the sender or recipient is.

To further ensure fairness, the development team conducted manual testing across various types of emails, including corporate messages, personal emails, and automated notifications to confirm that the algorithm treats legitimate messages consistently. Any observed inconsistencies were addressed by refining the rules and removing contextually biased triggers.

5.2 Transparency

Transparency is essential for building trust and enabling users to understand how AI-driven systems operate. In the context of phishing detection, transparency ensures that the system's decision-making process is visible, interpretable and traceable.

The Email Phishing Detection AI maintains transparency by making every stage of the detection pipeline observable and understandable. Each agent in the system performs a clearly defined function and produces outputs that can be reviewed both by developers and end users. Specifically:

- The Email Parser Agent exposes the structured version of the email, showing what data was extracted.
- The Risk Scorer Agent provides a breakdown of detected features and the assigned risk score.
- The Alert Generator Agent summarizes this information into human-readable explanations.
- Users can thus see how the system arrived at its conclusion rather than receiving an opaque "phishing" or "not phishing" label. This promotes informed decision-making and fosters trust between the system and its users.

On the backend, all processes are logged systematically, allowing system administrators to trace how specific outcomes were generated. The transparency framework also supports debugging and validation, ensuring that false positives and negatives can be analyzed and corrected efficiently.

By adhering to transparent AI design, the system meets both ethical expectations and emerging regulatory standards that require explainability and auditability in automated decision-making.

5.3 Explainability

Explainability is closely related to transparency but focuses on making AI outcomes comprehensible to human users. While transparency ensures visibility into the process, explainability ensures that users can understand why a particular decision or result was produced.

The Email Phishing Detection AI incorporates explainability through its Alert Generator Agent and Chatbot Agent:

- The Alert Generator Agent, powered by the Ollama Local LLM (llama3.1:8b) [4], transforms technical analysis into simple, natural language explanations. For example, instead of merely stating "phishing detected," it provides context such as:
  "This email contains a link to an unverified domain and uses urgent language requesting personal information."
- The Chatbot Agent, powered by the Groq Cloud API [5], complements this by allowing users to ask follow-up questions such as, "Why was this email considered high risk?" or "What does a phishing score of 85 mean?"

This two-tiered explainability framework enables users to learn from each detection and understand phishing patterns intuitively. It turns an automated detection system into a teachable AI companion, reinforcing cybersecurity awareness and user empowerment.

Explainability also supports ethical AI auditing, as system outputs can be reviewed and justified to stakeholders or regulatory authorities. By combining rule-based reasoning with language-based explanations, the system ensures both clarity and depth in communication.

5.4 Privacy

Privacy is a cornerstone of Responsible AI, particularly in systems that process sensitive user data such as emails. The Email Phishing Detection AI prioritizes privacy protection through architectural and operational safeguards.

One of the key design decisions was to employ local inference for critical AI processing tasks. The Alert Generator Agent operates using a locally hosted LLM (Ollama llama3.1:8b) [4], ensuring that sensitive email data never leaves the user's environment during explanation generation. This approach contrasts with traditional cloud-based systems, where user content is transmitted to external servers for processing.

In addition:

- Cloud interactions, such as those with the Groq Chatbot API [5], are restricted to anonymized metadata. No raw email text or personally identifiable information (PII) is shared.
- All database operations via MongoDB Atlas [2] employ secure authentication and encryption to protect stored analysis histories.
- Firebase Authentication [3] enforces secure sign-in mechanisms and token validation, ensuring only authorized users can access their data.
- Gmail API [6] integration operates under Google's OAuth 2.0 protocol, which requires explicit user consent before accessing any inbox data.

Furthermore, no user data is used for model training or analytics without consent. The system adheres to data minimization principles, processing only the information necessary for phishing detection and discarding intermediate data after use.

5.5 Accountability

Accountability ensures that AI system operations are traceable and that both developers and users can hold the system or its components responsible for their actions. In AI ethics, accountability also implies that decisions made by automated systems can be justified and, if necessary, contested.

The Email Phishing Detection AI maintains accountability through detailed logging, traceability and modular responsibility. Each agent in the architecture logs its actions with:

- Timestamps marking the exact time of operation,
- Agent identifiers specifying which component performed which action,
- Input-output records showing data transformations and results.

These logs are securely stored in MongoDB Atlas [2], allowing developers or auditors to review the decision pipeline if needed. This facilitates the reconstruction of analysis steps for validation or dispute resolution.

In addition, accountability extends to model management and deployment. The use of version-controlled LLMs (e.g., specified model versions in Ollama [4] and Groq [5]) ensures reproducibility of results and consistency in performance across different environments. Should an error or false classification occur, the responsible agent can be identified, and corrective actions can be implemented transparently.

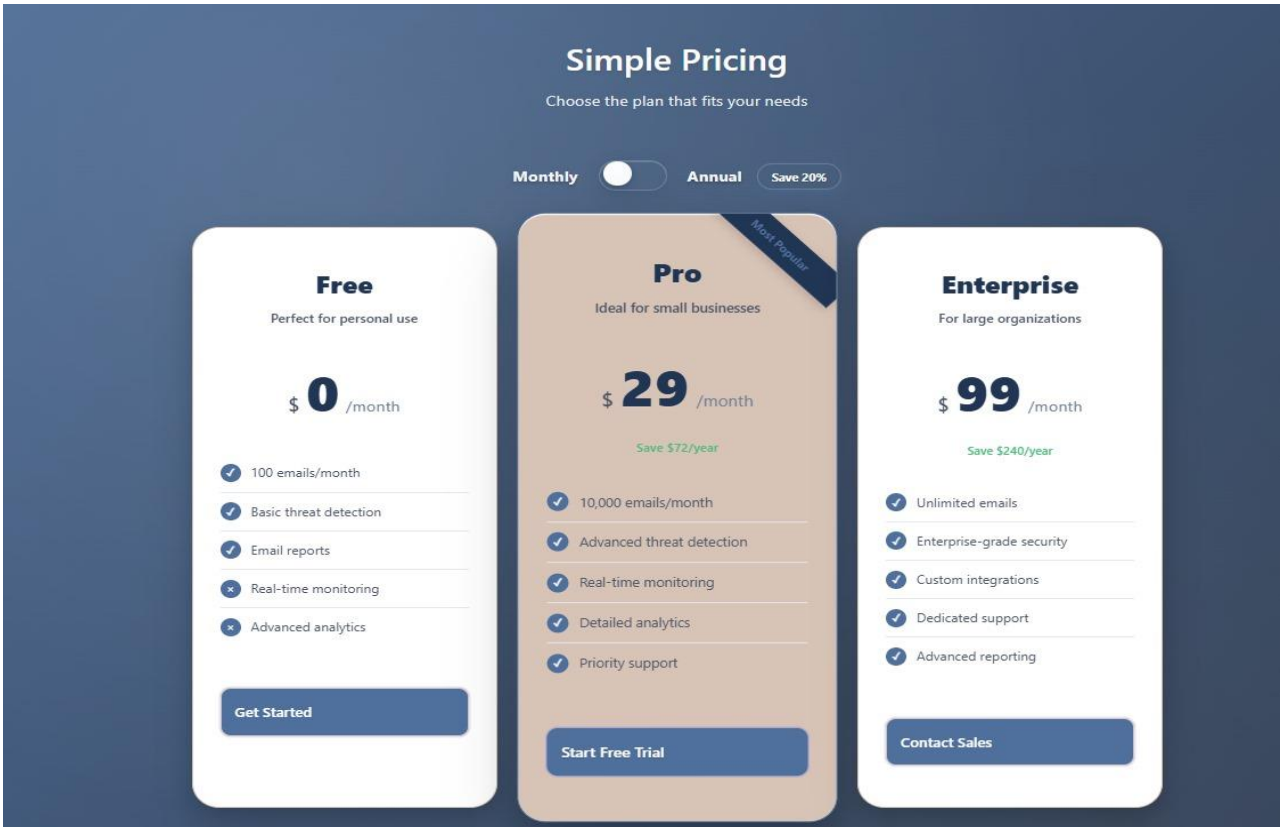
From an organizational perspective, accountability also involves clear documentation of data handling practices, consent flows, and AI model usage. All these details are included in the system's design documentation and privacy policy, ensuring compliance with Responsible AI governance standards.

5.6 Ethical considerations during development

- Designed to avoid automated blocking without user confirmation — the system warns and recommends actions rather than performing unsafe automatic block/removal.
- Privacy-first decisions (local LLM for explanations) were chosen to reduce sensitive data exposure.
- Users are informed about which parts of the system use cloud services, and consent flows are explicit for message access.

6. Commercialization plan

The system is suitable for commercial deployment as a SaaS product.



Business model

- **Free plan:** Manual checks and limited daily analyses.
- **Pro plan ($9-19/month):** Automated Gmail scanning, reports and team dashboard.
- **Business plan ($29-49/month):** Admin management, API access and integrations.

Target audience

- Small and medium businesses seeking affordable email protection.
- IT service providers offering security packages.
- Educational institutions for awareness training.
- Individuals handling confidential communications.

Unique Selling Points

- Explainable AI (transparent results)

- Privacy-first local inference
- Scalable, modular backend
- Integrated chatbot assistant

Scalability and sustainability

- **Scalability:** Microservice-friendly multi-agent design allows horizontal scaling of parse/score/alert components. Move heavy LLM workloads to dedicated inference nodes in production.
- **Sustainability:** Modular pricing (per-analysis) and optional enterprise support can fund ongoing model maintenance and infrastructure. Privately hosted LLM options minimize ongoing cloud costs for explainability.

Future Product Evolution

The system can evolve into a full SaaS security platform with real-time phishing monitoring, browser extensions, and mobile app support. Future versions could integrate enterprise dashboards, AI-powered security recommendations, and team analytics tools to enhance commercial viability.

7. Discussion

The development of the Email Phishing Detection AI System provided valuable technical and practical insights into building explainable, privacy-aware AI solutions for cybersecurity. Throughout the project, the team encountered several challenges, identified areas for improvement, and recognized opportunities for future enhancement. This section combines the lessons learned, implementation challenges, system limitations, and proposed improvements into a single comprehensive discussion.

Key Learnings

- Separation of concerns (parser vs scorer vs alert) dramatically simplified debugging and iteration.
- Local LLM explainability was valuable for user trust but required careful engineering to manage hardware constraints.
- Gmail OAuth integration [6] improved usability but introduced token lifecycle complexities that needed robust error handling.

Challenges faced and mitigation

- **URL redirection & obfuscation:** Resolved partly through redirect resolution and domain comparison heuristics; flagged cases require deeper URL sandboxing for full certainty.
- **Hardware constraints for local LLM:** Addressed by providing fallback templates for alert generation and offering cloud-based explainability in resource-limited deployments (with user consent).

Limitations

- Heuristic scorer needs manual tuning to handle new tactics.
- Gmail integration is manual-per-user (OAuth consent required).
- Chatbot depends on cloud LLM (subject to rate limits).
- Local LLM performance is hardware dependent.

Future enhancements

- Integrate lightweight transformer classifiers (BERT family) for better adaptability.
- Extend multilingual support and browser/Slack integrations.
- Add continuous-learning pipeline with human-in-the-loop review for model updates.

8. Conclusion

The Email Phishing Detection AI project demonstrates a practical, modular approach to combining NLP, heuristic analysis and LLM-based explainability in an end-to-end system. The multi-agent pipeline - Email Parser, Risk Scorer, Alert Generator and Chatbot, which provides accurate detection, clear explanations and interactive support while preserving privacy through local LLM inference. Prototype evaluations show promising detection accuracy and high user satisfaction with explainability. The architecture is designed for productization with clear commercialization pathways and scalability options for enterprise usage. Further enhancements (multilingual support and integrations) will increase robustness and real-world applicability.

Overall, this project contributes a novel, explainable, and privacy-preserving phishing detection framework that bridges NLP, information retrieval, and responsible AI — demonstrating a scalable solution for real-world cybersecurity analytics.

9. References

[1] I. Fette and N. Sadeh, "Learning to Detect Phishing Emails," *Proceedings of the 16th International Conference on World Wide Web (WWW)*, 2007.
Available at: https://dl.acm.org/doi/10.1145/1242572.1242660

[2] B. Basit, M. Zafar, M. Javed, and M. Arif, "A Comprehensive Survey of Phishing Attack DetectionTechniques,"*IEEEAccess*,vol.7,pp.168–197,2019.
Available at: https://ieeexplore.ieee.org/document/8606953

[3] A. Jain and B. Gupta, "A Machine Learning-Based Approach for Phishing Email Detection Using NLP," *Journal of Information Security and Applications*, vol. 48, pp. 102–112, 2019.
Available at: https://doi.org/10.1016/j.jisa.2019.102417

[4] K. Rao, D. Kumar, and R. S. Kumar, "Email Phishing Detection Using Natural Language Processing and Machine Learning," *Procedia Computer Science*, vol. 191, pp. 813–820, 2021.
Available at: https://doi.org/10.1016/j.procs.2021.07.105

[5]T.Tiangolo,"FastAPIDocumentation,"*FastAPIProject*,2024.
Available at: https://fastapi.tiangolo.com

[6]OllamaTeam,"OllamaLocalLLMDocumentation,"*OllamaAI*,2024.
Available at: https://ollama.ai