OS-Level AI Phishing Detection System

1. Project Overview

Project Title

OS-Level Real-Time Phishing Detection System using Groq LLM

Problem Statement

Phishing attacks today are not limited to browsers. They appear in emails, documents, text editors, PDFs, chat applications, and even system dialogs. Most existing solutions are browser-based or rule-based and fail to detect phishing content that is hidden, scrolled out of view, or present in non-browser applications.

Proposed Solution

Build an **OS-level background application** that continuously monitors user-visible and accessible textual content across **any application**, analyzes it using a **Groq-powered LLM**, and warns the user in real time via a floating widget when phishing content is detected.

The system works even when phishing content is: - At the bottom of a page - Inside emails, documents, or editors - Spread across multiple UI regions

---

2. Core Idea (What This Project Is)

This project is: - An **always-running OS-level security assistant** - Powered by **Groq LLM via API key** - Uses **prompt-based intelligence (not classical ML training)** - Uses **screen OCR + accessibility APIs** to extract text - Provides **real-time phishing risk analysis with explanation**

This project is NOT: - A browser-only extension - A static dataset-based ML classifier - A signature/rule-based system

---

3. High-Level System Architecture

```
┌─────────────────────────┐
│   User Applications     │
│ (Browser, Notepad, PDF) │
└───────────┬─────────────┘
            │
            ▼
┌─────────────────────────┐
```

```
│ Content Acquisition Layer│
│ • Screen Capture (OCR)   │
│ • Accessibility APIs     │
└───────────┬──────────────┘
            │
            ▼
┌──────────────────────────┐
│ Text Processing Layer    │
│ • Cleaning               │
│ • Deduplication          │
│ • Chunking               │
└───────────┬──────────────┘
            │
            ▼
┌──────────────────────────┐
│ Groq LLM Analysis Layer  │
│ • Prompt-based detection │
│ • Risk scoring           │
└───────────┬──────────────┘
            │
            ▼
┌──────────────────────────┐
│ UI & Alert Layer         │
│ • Floating widget        │
│ • Risk explanation       │
└──────────────────────────┘
```

---

4. Complete End-to-End Flow

Step 1: Application Startup

- The OS-level application starts automatically (or manually).
- A floating widget appears indicating **“Protection Active”**.
- Background services are initialized.

Step 2: Active Window Monitoring

Purpose: Know which application the user is currently interacting with.

- Detect the currently active window using OS APIs.
- Capture window metadata (process name, title, position).

This ensures: - Only relevant content is analyzed - Reduced CPU usage

---

Step 3: Content Acquisition (MOST IMPORTANT)

This is done via **two parallel pipelines**.

3A. Screen Capture Pipeline (Visual Content)

- Capture the active window or screen area
- Used for applications that do not expose internal text
- Examples: PDFs, custom apps, images

Output: Screenshot image

3B. Accessibility Pipeline (Logical Content)

- Uses OS accessibility/UI automation APIs
- Reads full text from apps like:
- Notepad
- Browsers
- Email clients

Key Advantage: - Can read content **not currently visible or scrolled**

Output: Full document text

Step 4: Text Extraction

OCR-Based Extraction

- OCR converts screenshots into raw text
- Handles fonts, layouts, UI elements

Image → Text

Accessibility Text Extraction

- Directly reads clean text
- No OCR errors

Step 5: Text Aggregation & Cleaning

- Merge OCR text + accessibility text
- Remove:
- Duplicate lines
- UI noise (menus, buttons)
- Non-informative tokens

Result: **High-quality unified text corpus**

Step 6: Chunking Strategy

Why chunking is needed: - LLM token limits - Phishing often appears in small sections

Chunk Rules: - 300–700 words per chunk - Logical grouping (paragraphs, sections) - Preserve context

---

Step 7: Groq LLM Analysis

Each chunk is sent to Groq LLM with a structured prompt.

The LLM analyzes: - Phishing intent - Social engineering signals - Suspicious URLs - Urgency and fear language

LLM Output (JSON): - is_phishing - risk_level - reasons - suspicious_phrases

---

Step 8: Decision Engine

- Aggregate results from all chunks
- Compute final risk level

Decision logic: - Any high-risk chunk → HIGH ALERT - Multiple medium-risk → MEDIUM ALERT

Step 9: UI Alert & Explanation

- Floating widget changes color/status
- User receives:
- Risk level
- Explanation
- Highlighted suspicious text

---

5. Technology Stack

OS-Level Application (Frontend)

Component	Technology
UI Widget	PyQt / Tkinter
Always-on-top window	OS window APIs

Component	Technology
Notifications	Native OS APIs

Backend / Core Engine

Component	Technology
Language	Python
Screen Capture	MSS / PIL
OCR	EasyOCR
Accessibility	Windows UI Automation (UIA)
Text Processing	Python NLP utils

LLM Layer

Component	Technology
LLM Provider	Groq
Models	LLaMA / Mixtral (via Groq)
Detection Method	Prompt Engineering
Output	Structured JSON

---

6. Groq LLM – How It Is Used

Why Groq?

- Ultra-low latency
- High throughput
- Suitable for near real-time analysis

Role of Groq in This Project

Groq acts as: - A **reasoning engine** - A **phishing expert** - An **explainability generator**

What You Are NOT Doing

- No model training
- No weight fine-tuning

What You ARE Doing

- Prompt tuning
- Structured outputs
- Context-aware reasoning

---

7. Prompt Design Strategy

The prompt defines: - Phishing rules - Output format - Severity thresholds

Groq returns machine-readable decisions used by the system.

---

8. Security & Privacy Considerations

- No keystroke logging
- No permanent storage of screenshots or text
- All processing done in-memory
- User-controlled ON/OFF
- Transparent permissions

---

9. Limitations (Honest & Important)

- Cannot read text that is neither rendered nor exposed
- OCR accuracy depends on screen quality
- Accessibility varies by application
- API latency and cost considerations

---

10. Project Value

Academic Value

- OS concepts
- AI + systems integration
- Applied cybersecurity

Industry Value

- Real-time threat detection
- LLM-powered security
- Cross-application intelligence

---

11. Future Enhancements

- Local lightweight fallback model
- URL reputation integration
- Browser deep hooks
- Enterprise policy engine

---

12. Final Summary

This project implements a **system-wide AI phishing detection mechanism** that operates beyond browsers, leverages **Groq LLM for intelligent reasoning**, and demonstrates a **deep understanding of OS internals, AI, and cybersecurity**.

It is a **high-impact, advanced, and portfolio-defining project**.