GoPhishFree

Architecture Document

Course:	EECS 582 - Capstone Project
Team Number:	24
Team Members:	Ty Farrington, Brett Suhr, Andrew Reyes, Nicholas Holmes, Kaleb Howard
Project Name:	GoPhishFree
Date:	February 8, 2026

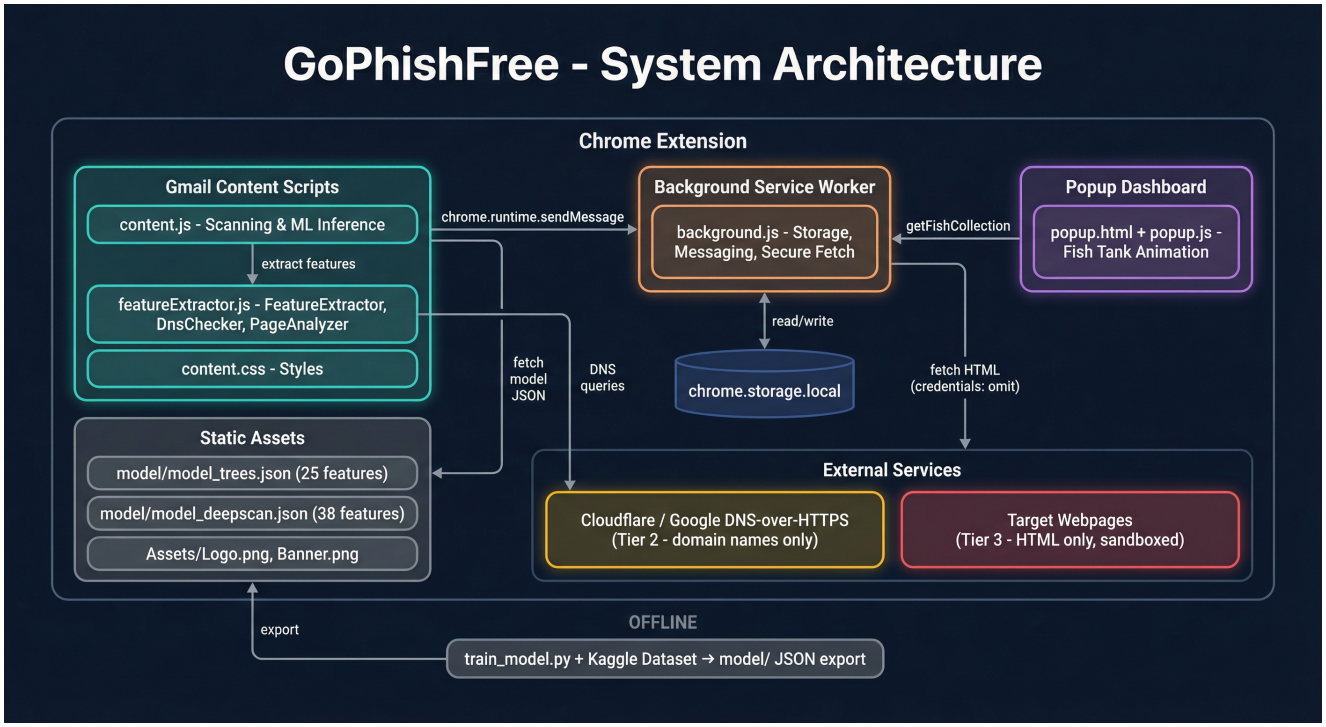
Project Synopsis

A privacy-first Chrome extension that detects phishing emails in Gmail using a locally-run Random Forest machine learning model and a three-tier risk scoring pipeline.

1. System Overview

GoPhishFree is a Chrome Manifest V3 extension that scans emails in real time as users read them in Gmail. The entire detection pipeline - feature extraction, machine-learning inference, and risk scoring - executes locally inside the browser, ensuring that no email content ever leaves the user's device. The system is composed of four major components: a Gmail content script that orchestrates scanning and renders the UI, a feature extraction module that derives numerical signals from email metadata and URLs, a background service worker that manages storage and proxies network requests, and a popup dashboard that presents a gamified "fish tank" collection of scanned emails.

Figure 1 - System Architecture



High-level component map showing content scripts, service worker, popup, and external services.

2. Three-Tier Detection Pipeline

Detection is organized into three progressive tiers, each adding deeper analysis at the cost of additional permissions or latency.

Tier 1 - Email Analysis (always active)

When a user opens an email, the content script extracts the sender address, display name, body text, attachment metadata, and every hyperlink. The FeatureExtractor class computes 25 numerical features from this data - covering URL structure (dot count, subdomain depth, path length, query parameters), special-character frequency (@ symbol, tildes, underscores, percent-encoding), security indicators (missing HTTPS, IP-based URLs, double slashes in paths), suspicious patterns (domain appearing in subdomains or paths, link-text mismatches), and content signals (sensitive-word count, query complexity). An additional 10 custom features capture higher-level behavioural cues such as punycode usage, header-address mismatch, URL-shortener links, urgency language,

and credential-request phrases. The 25 ML features are Z-score normalized using the training scaler's mean and standard deviation, then fed into the Random Forest model for probability estimation.

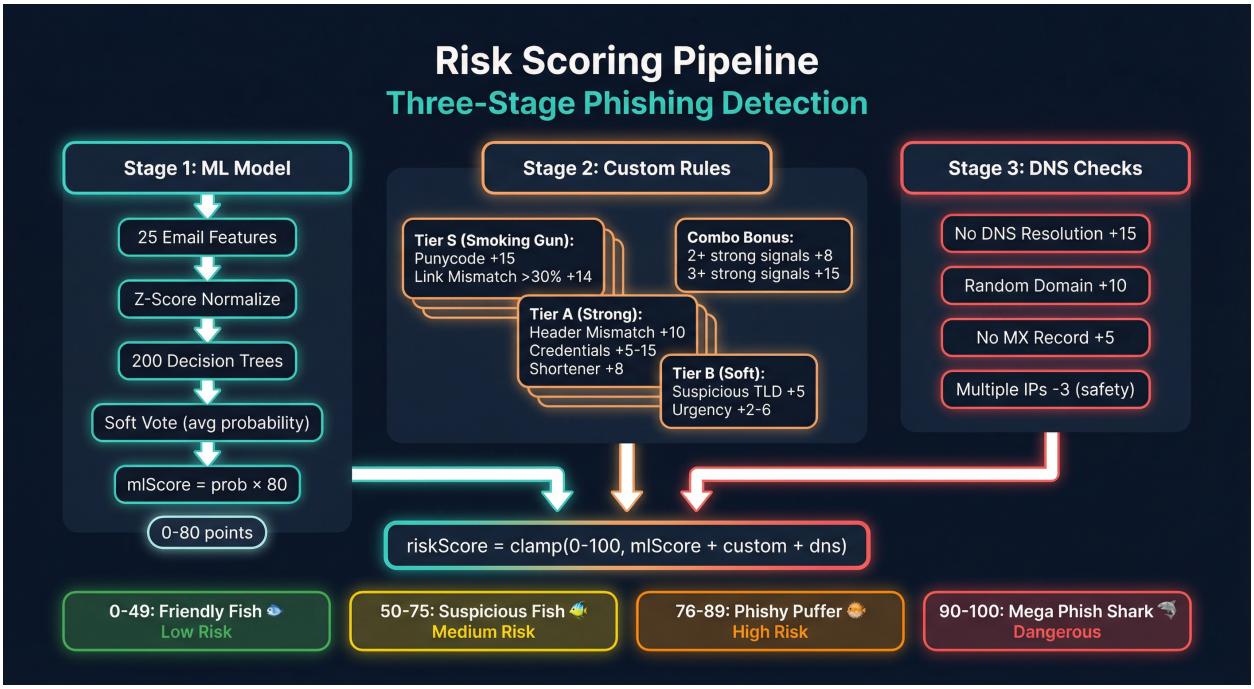
Tier 2 - DNS Validation (enabled by default)

If Enhanced Scanning is toggled on (the default), GoPhishFree queries Cloudflare's DNS-over-HTTPS resolver (with Google DNS as a fallback) for the sender's domain and every linked domain. Four DNS features are derived: whether the domain resolves at all, whether it has MX records, whether the A-record returns multiple IPs (a sign of legitimate load-balanced infrastructure), and whether the domain label is a random string based on Shannon entropy analysis. DNS results are cached with a 10-minute TTL to minimize latency on subsequent scans.

Tier 3 - Deep Scan (user-initiated)

Users may optionally trigger a Deep Scan, which fetches the raw HTML of up to 10 linked pages through the background service worker. The fetch is sandboxed: credentials are omitted, responses are capped at 2 MB, content-type is validated, redirects are checked, and no JavaScript is executed. The PageAnalyzer class extracts 13 page-structure features - insecure forms, external form actions, abnormal action URLs, mailto submissions, external hyperlink and resource ratios, external favicons, null/self-redirect links, iframes, missing page titles, image-only forms, and embedded brand names. These 13 features are appended to the original 25 to form a 38-element vector, which is run through a separate Deep Scan Random Forest model for a refined probability.

Figure 2 - Risk Scoring Pipeline



Three-stage pipeline: ML probability, custom rule adjustments, and DNS adjustments converge into a 0-100 risk score.

3. Machine Learning Model

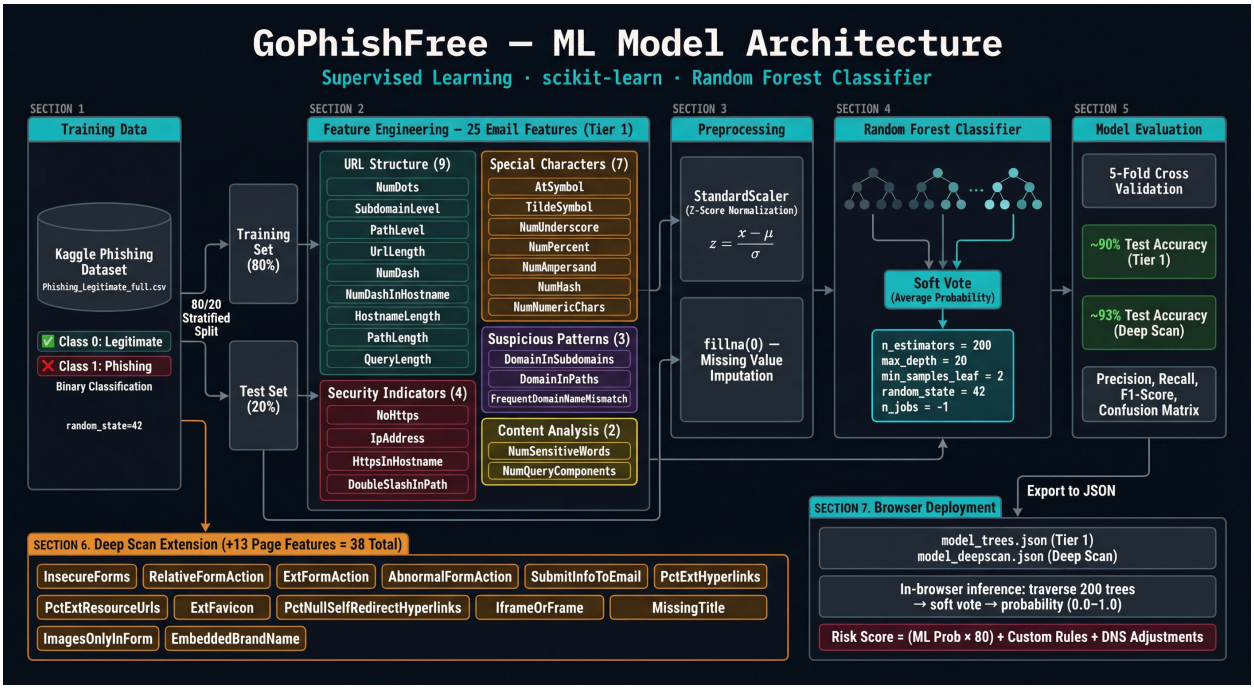
GoPhishFree uses supervised learning with scikit-learn's RandomForestClassifier for binary classification (legitimate vs. phishing). The training dataset is a publicly available Kaggle corpus (Phishing_Legitimate_full.csv)

containing labelled URL and email samples. Data is split 80/20 with stratified sampling (random_state=42) to preserve class balance.

Preprocessing consists of selecting the relevant feature columns, imputing missing values with zero, and applying StandardScaler Z-score normalization. The classifier is configured with 200 estimators, a maximum tree depth of 20, a minimum of 2 samples per leaf, and parallel training across all CPU cores (n_jobs=-1). Model evaluation uses 5-fold cross-validation, a classification report (precision, recall, F1-score per class), and a confusion matrix. The Tier 1 model achieves approximately 90% test accuracy, while the Deep Scan model reaches approximately 93%.

After training, the full Random Forest structure - every tree's split features, thresholds, child pointers, and leaf probabilities - is exported to JSON along with the scaler's mean and scale arrays. At runtime the content script loads this JSON and performs inference by traversing all 200 trees in JavaScript, averaging the leaf probabilities (soft voting) to produce a phishing probability between 0.0 and 1.0.

Figure 3 - ML Model Architecture



End-to-end ML pipeline: training data, 25/38 features, preprocessing, Random Forest (200 trees), evaluation, and browser deployment.

4. Risk Score Composition

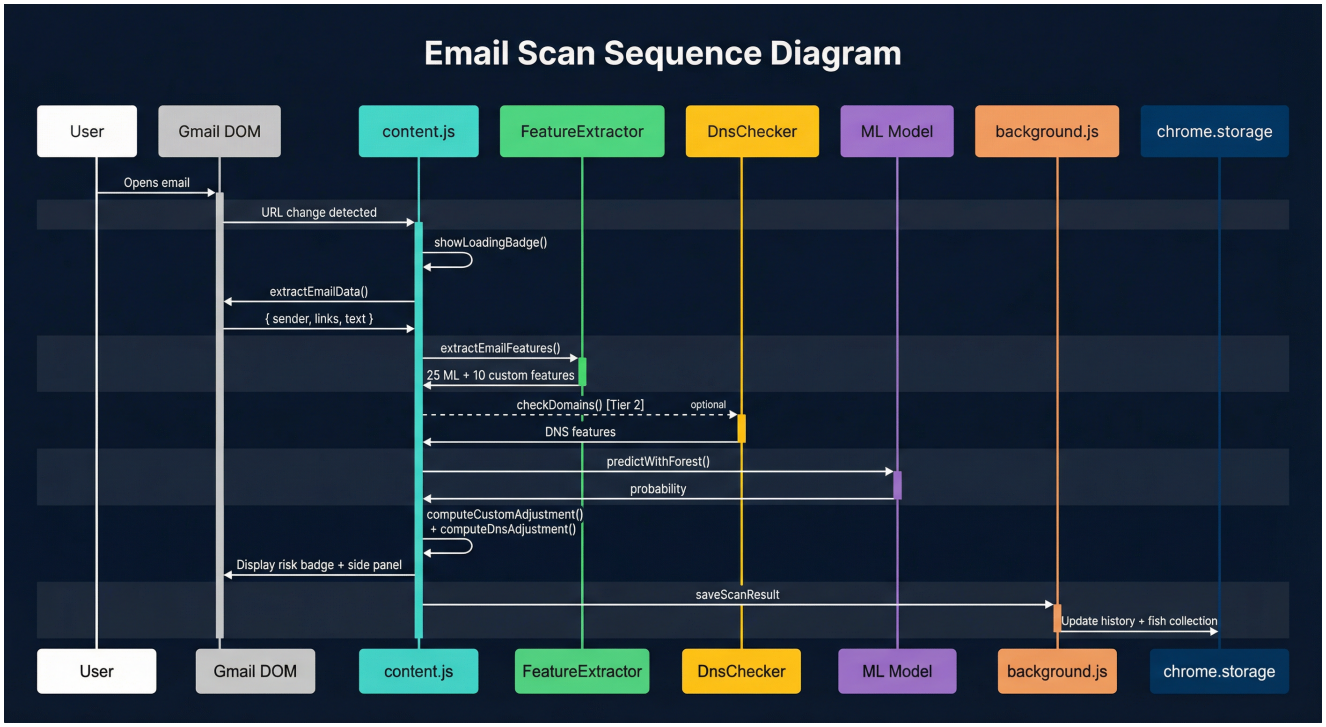
The final risk score is computed as: Risk Score = (ML Probability x 80) + Custom Adjustment + DNS Adjustment, clamped to the range 0-100. The ML component contributes up to 80 points. Custom rules add points through a tiered weighting scheme: Tier S ("smoking gun") signals such as punycode domains (+15) or link-text mismatch above 30% (+14); Tier A ("strong") signals like header-address mismatch (+10) or credential-request language (up to +15); and Tier B ("soft") signals such as suspicious TLDs (+5) or urgency language (up to +6). A combination bonus adds +8 for two or more strong signals, or +15 for three or more. DNS adjustments add up to +15 for non-resolving domains, +10 for random-string domains, or subtract 3 points for domains with multiple IPs (indicating legitimate CDN infrastructure).

The resulting score maps to four risk levels displayed as themed fish: Friendly Fish (0-49, Low), Suspicious Fish (50-75, Medium), Phishy Pufferfish (76-89, High), and Mega Phish Shark (90-100, Dangerous). The fish type, score, human-readable reasons, and timestamp are persisted to chrome.storage.local and displayed in the popup fish tank.

5. Runtime Scan Flow

The diagram below illustrates the full sequence from the moment a user opens an email to the final risk badge display. A MutationObserver in the content script detects Gmail navigation events, triggers feature extraction, runs ML inference, applies custom and DNS adjustments, renders the badge and side panel, and persists the result via the background service worker.

Figure 4 - Email Scan Sequence



Sequence diagram: User opens email -> DOM detection -> feature extraction -> ML inference -> risk badge -> storage.

6. Security & Privacy

GoPhishFree enforces strict privacy guarantees. All ML inference and feature extraction execute entirely within the browser - no email content, URLs, or metadata are transmitted to any external server. DNS-over-HTTPS queries send only domain names (never email bodies or user data) to Cloudflare or Google public resolvers. Deep Scan fetches omit all credentials and cookies, cap response sizes at 2 MB, validate content types, enforce an 8-second timeout, and parse HTML via DOMParser without executing any scripts. The extension requires no API keys, no backend server, and stores all data locally via the Chrome Storage API.