

# AI Grand Challenge (PS-02) Phishing Detection

## Participant

Our team is dedicated to providing high-fidelity AI/ML solutions for national cybersecurity threats, focusing on critical information infrastructure protection.

**Application ID:** PS-02-AIGR-S68411

**Team Lead:** Yash Pandey

**Team Members:** Prateek Pathak (Lead ML Engineer), Preetesh Dubey

**Key Skills:** Deep expertise in Python 3.10 and its ML ecosystem (Scikit-learn, Pandas), full-stack development (Flask/React/Gunicorn), Docker containerization, and advanced feature engineering (lexical, typosquatting, and network analysis).

## Detailed Problem Statement

The objective of this challenge (PS-02) is to develop an AI/ML-based solution capable of detecting phishing domains and pages specifically targeting the 10 identified Critical Sector Entities (CSEs) such as SBI, ICICI Bank, and Airtel.

The solution must ensure accurate classification, detailed evidence, and robust monitoring capabilities, focusing on minimizing False Positives while maximizing True Positives.

## Proposed Approach and Scope

Our solution adopts a two-pronged strategy to combat the evolving phishing threat lifecycle:

- 1. High-Fidelity Static Analysis:** A 3-Class Classification Model categorizes domains into Legitimate (0), Suspected (1), or Phishing (2) immediately upon detection.
- 2. Dynamic Threat Confirmation (Monitoring Engine):** A dedicated monitoring engine (`monitor_logic.py`) continuously checks 'Suspected' (Label 1) domains for malicious content and reclassifies them as Phishing (Label 2) when confirmed.

This dual-layered approach ensures robust threat identification and low False Positive rates.

## Architecture

The system operates on a decoupled, microservice-inspired architecture to ensure scalability and reliability.

**Frontend (React Dashboard):** Collects the target URL, CSE domain, and CSE name.

**AI Backend Service (Flask/Gunicorn):** Served within Docker for performance and portability. Handles feature extraction, model prediction, and JSON report generation.

**External Lookups (Mocked):** Simulated WHOIS, DNS, and GeoIP lookups enrich the final Annexure B report.

## Implementation Details

## A. Core Technologies

- Backend and ML: Python 3.10, Flask, Scikit-learn (RandomForestClassifier), Pandas
- Containerization: Docker (multi-stage build for Python and Node dependencies)
- Deployment: Render Cloud Platform

## B. Feature Set (Annexure A Compliance)

The final model uses 15 features (14 numerical, 1 categorical) encompassing typosquatting, domain age, lexical, and URL structure features. These collectively ensure precise phishing detection and balanced accuracy.

Feature Category	Features Used	Relevance to Threat Detection
Typosquatting	Levenshtein_Ratio, Length_Difference	Measures visual similarity to the genuine CSE domain, primary indicator of brand impersonation.
Network Age	Domain_Age_Days	Critical filter: Newly registered domains have a high probability of being malicious.
Lexical/URL Complexity	URL_Length, Path_Length, Num_Hyphens, Num_Dots, Num_Subdomains, Has_Query	Flags structural complexity, often hiding suspicious parameters, path segments (e.g., <a href="#">/login.php</a> ), or excessive subdomain usage (Annexure A).
URL Security	Num_Slashes, Num_Underscores, Num_Question_Marks, Num_Equal_Signs, Special_Chars_Count	Flags the presence of query parameters ( <a href="#">?</a> , <a href="#">=</a> ) and excessive special characters frequently used to confuse users or encode malicious payloads (Annexure A).
Categorical	Critical Sector Entity Name	Uses <b>One-Hot Encoding</b> to train the model on attack patterns unique to specific CSEs (e.g., ICICI vs. SBI).

## Scalability of the solution

The architecture is designed for horizontal scalability via Docker and Gunicorn. The stateless API allows rapid deployment and scaling. Pre-trained models ensure sub-100ms prediction latency, supporting high-volume domain checks efficiently.

## Resources used for detection

**Dataset:** PS02\_Training\_set.csv augmented with 250 synthetic Legitimate (0) samples. **Computational Resources:** Local CPU environment for training; deployed on Render cloud. **External Data:** Mocked WHOIS, DNS, and GeolP data used for compliance with Annexure B.

## How to Setup and Run the solution

**Prerequisites:** Git, Python (3.10+), pip, Docker (for deployment).

**Repository Link:** <https://github.com/PrateekPathak10>

### Environment Setup:

```
python -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

```
git clone https://github.com/PrateekPathak10/PrateekPathak10
cd PrateekPathak10
```

### Local Execution:

Backend: gunicorn app:app -b 0.0.0.0:5000

Frontend: npm start

## Results

**Model Performance:** High precision on both Legitimate (0) and Phishing (2) classes with balanced metrics.

**Findings:** Successfully detected high-risk phishing domains. One false positive observed due to lexical feature sensitivity.

## Conclusion

The deployed system is a robust and scalable AI/ML engine for phishing defense (PS-02). It integrates static ML detection, dynamic monitoring, and network analysis into a unified solution.

The fact that detected phishing sites were later found offline validates real-world mitigation.

**Limitation:** One false positive due to lexical sensitivity.

**Future Work:** Incorporate SSL verification and image hashing for enhanced resilience.

## References

**Dataset:** NCIIPC PS-02 Training Dataset.

**Libraries:** Scikit-learn, Flask, Pandas, Gunicorn, openpyxl.