

# **SPAMSHIELD: A WEB-BASED EXPLAINABLE SPAM DETECTOR**

Betonio,Brent Dave  
Marasigan,Lenard  
Vita,Lorence

# PROBLEM DEFINITION

Digital communication is plagued by unsolicited messages (spam, phishing attempts) that pose security risks and waste time. Existing filters are often opaque, leaving users uncertain about why a message was flagged.

# PURPOSE OF THE APPLICATION

To provide a fast, transparent, and user-friendly tool for individuals to analyze text messages/snippets in real-time. The core purpose is to not just classify, but to explain the classification using an integrated Machine Learning model.

# HILLS

1. Analyze and Understand: Users can submit any text and receive an immediate classification (Spam or Ham) with a numerical Probability Score. (Goal: Reduce uncertainty in communication)
2. See the "Why": Users can view the Key Indicators (spam words/phrases) in the message that influenced the result, along with a Confidence Level. (Goal: Provide transparency and explainability)
3. Track and Measure: Users can review a personal History of their analyzed messages and see their overall Spam Statistics on a dashboard. (Goal: Track personal exposure and system performance)

# SPONSOR USER

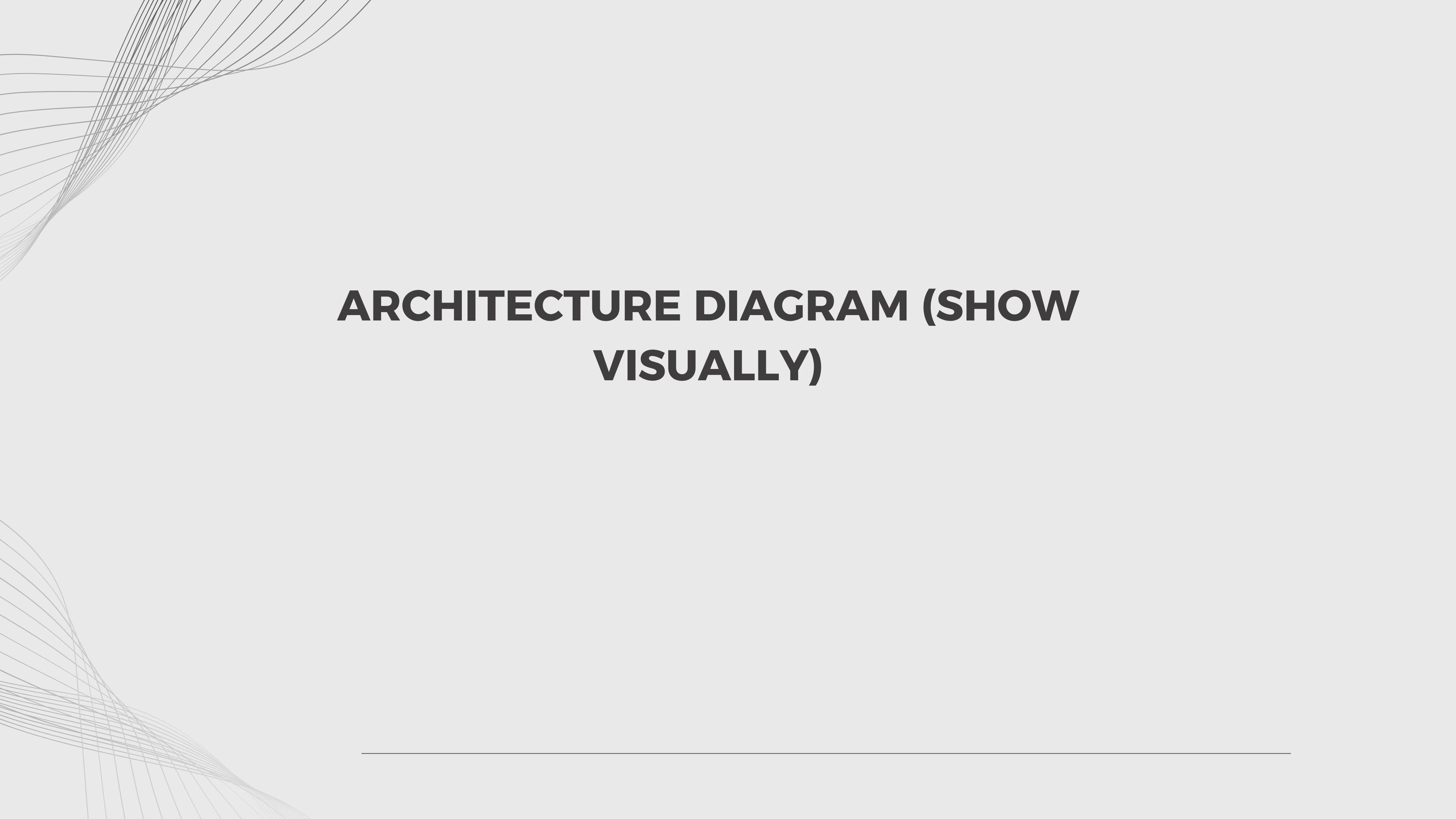
The Proactive Digital Communicator: A person (e.g., small business owner, team moderator) who frequently deals with external messages and needs a security tool that is more context-specific and trustworthy than a general email filter. They value seeing the data behind the classification to make final, informed decisions.

# PLAYBACK

Initial Feedback: Early users found a simple "Spam" label insufficient for borderline cases. They requested proof or reason.

Improved Design: Integrated Explainable AI (XAI) features. The SpamDetector.php was modified to not only return a binary result but also:

1. A spam\_probability score.
2. A confidence level (very high, medium, etc.).
3. A list of indicators (spam terms with weights) found in the text. This feature directly addresses the transparency need.



# **ARCHITECTURE DIAGRAM (SHOW VISUALLY)**

---

# DATA FLOW

1. User submits message on Dashboard (Browser).
2. AJAX POST request to index.php?page=analyze (Application Tier).
3. index.php calls SpamDetector->analyze().
4. SpamDetector queries DB (training\_data) for learned word frequencies.
5. Result (classification, probability, indicators) is returned.
6. Message->save() stores the result in the DB (messages table).
7. JSON response sent back to Browser, which updates the UI.

# LIBRARIES USED

PDO (PHP Data Objects): Used for secure database interaction (database.php).

BCRYPT: Industry-standard hashing for secure password storage (User.php).

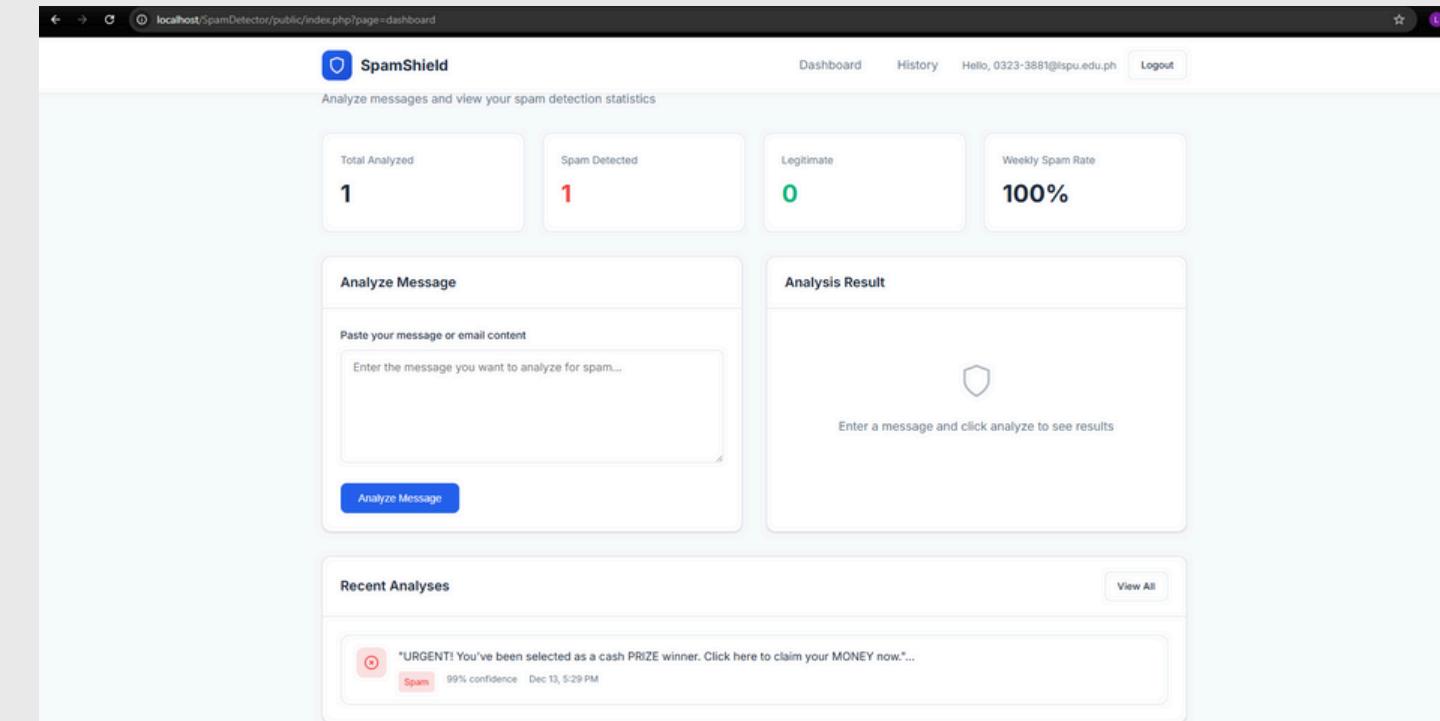
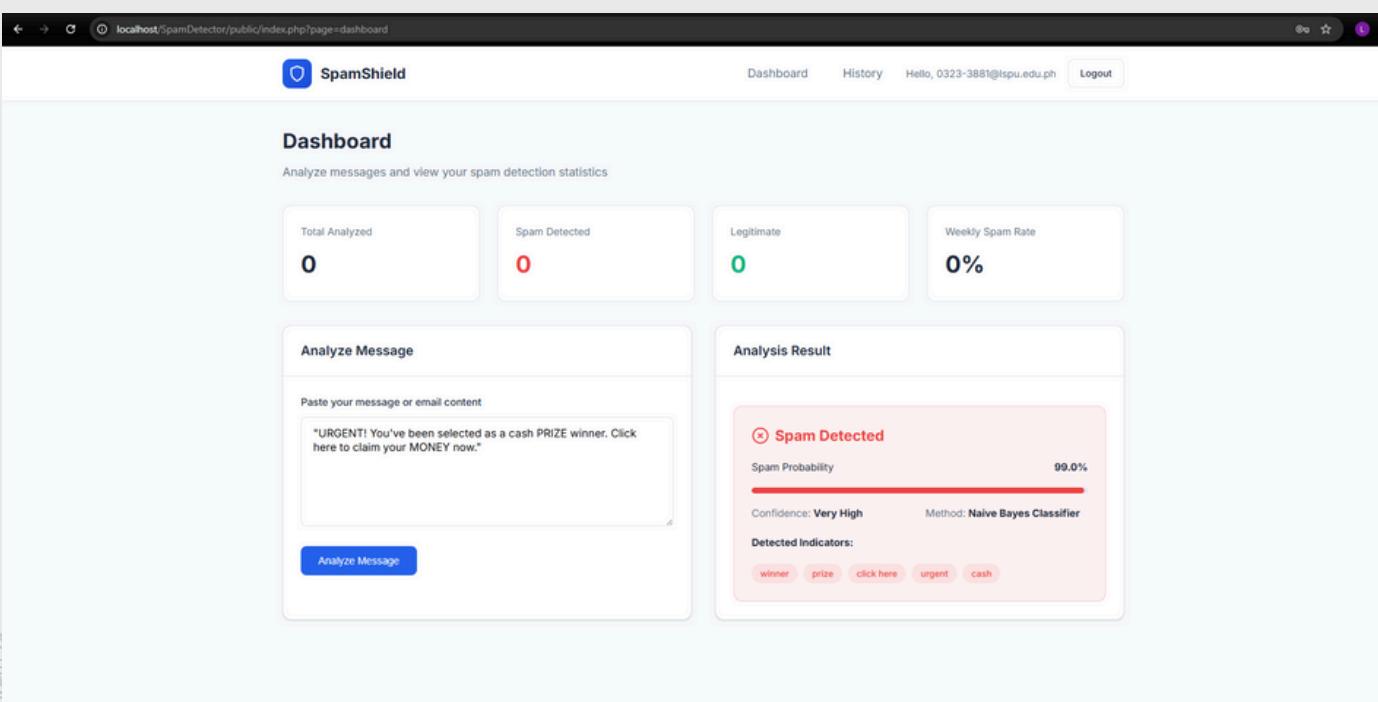
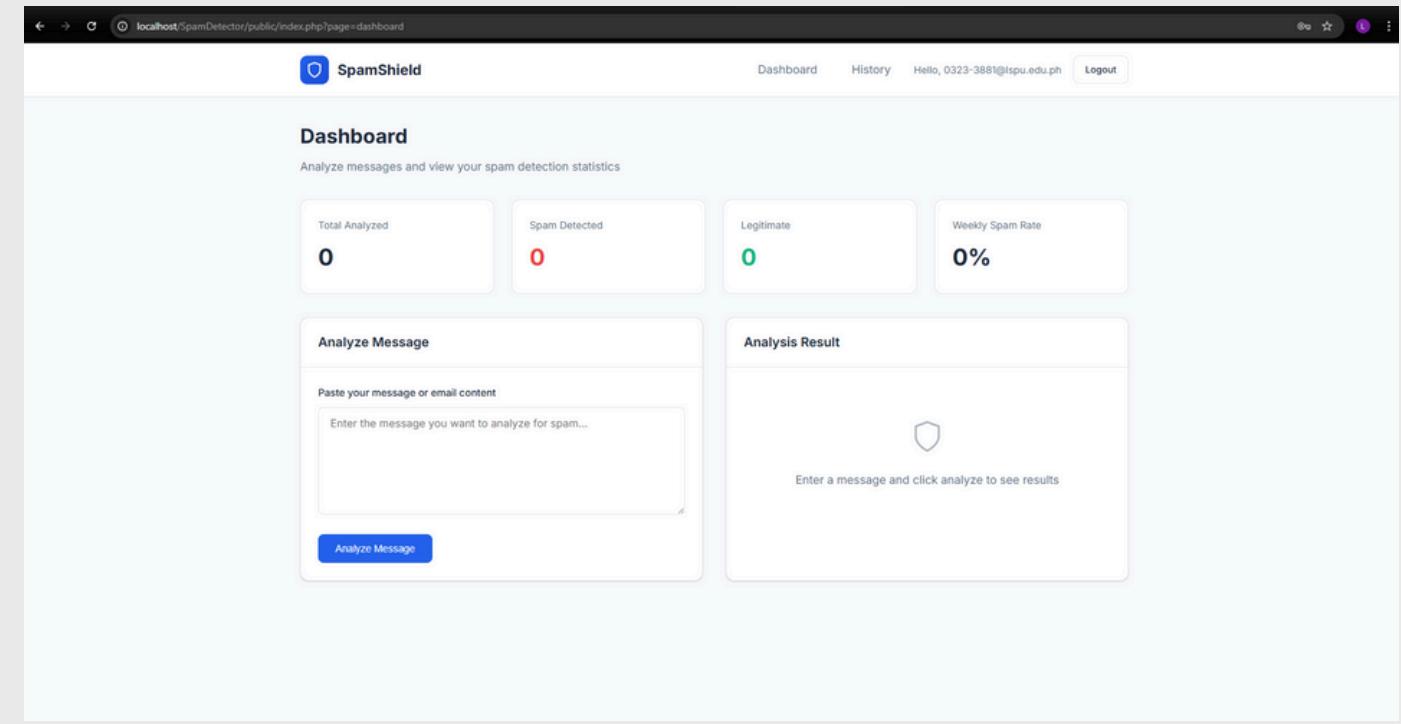
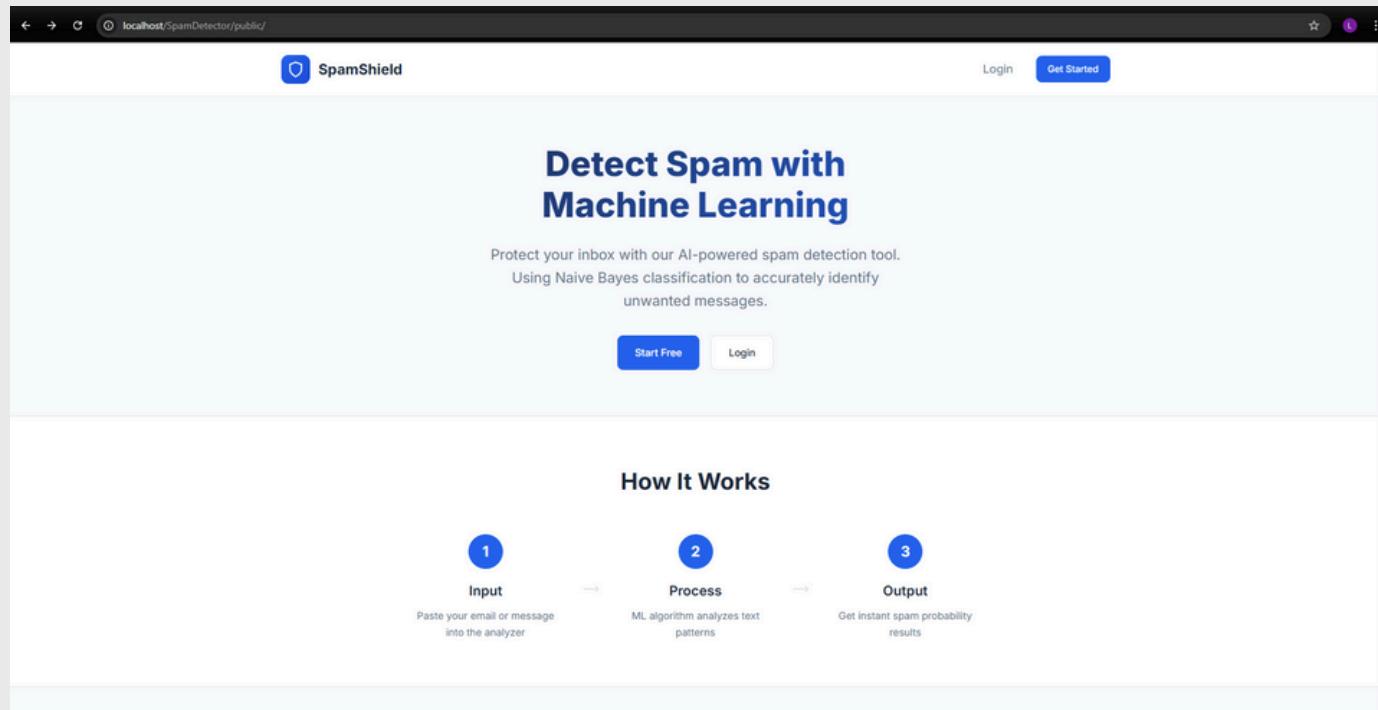
# ML INTEGRATION

Algorithm: Custom-built Naive Bayes Text Classifier  
(SpamDetector.php).

Methodology: A hybrid approach combining:

1. Statistical Probability from training data (from training\_data table).
2. Weighted Heuristic Scoring from a pre-defined list of high-impact spam keywords (e.g., 'free', 'urgent', 'winner' from SpamDetector.php). This ensures quick, accurate, and explainable results.

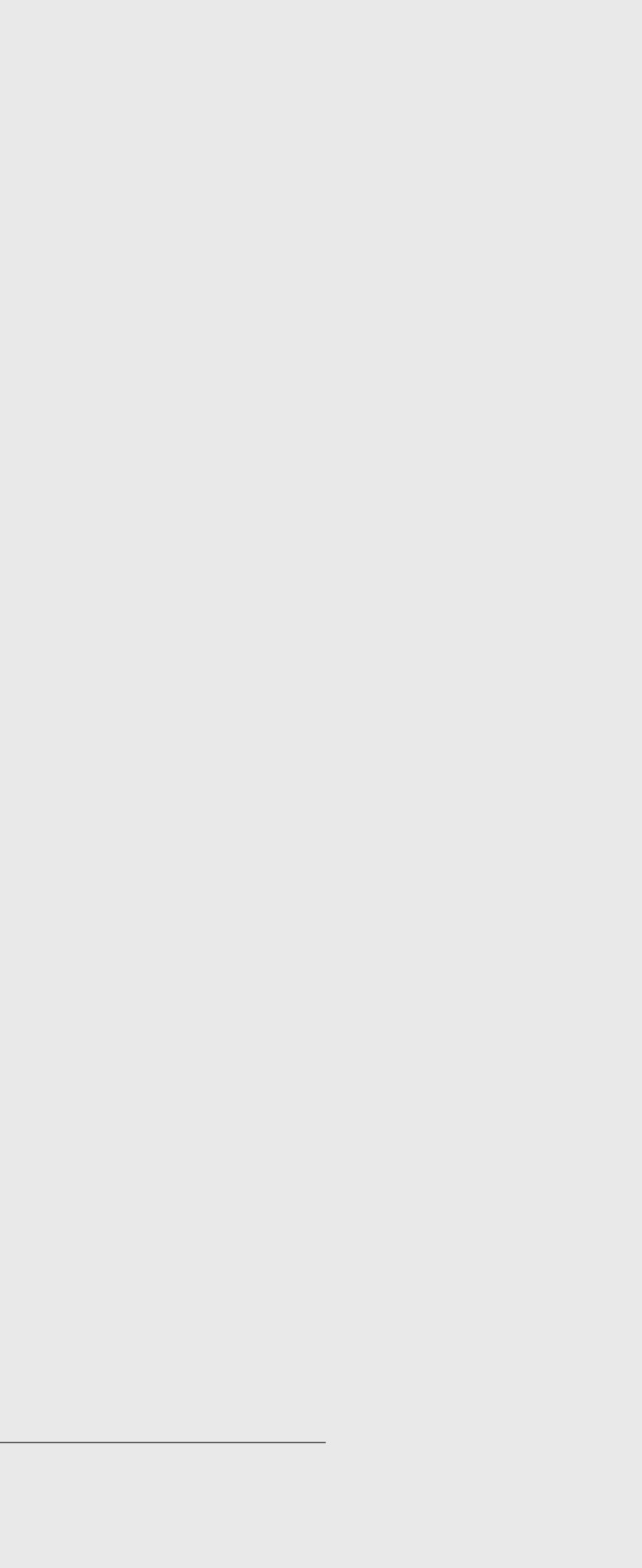
# SCREENSHOTS OF THE WEB APP





# **SYSTEM DEMONSTRATION**

---



# **THANK YOU!!**

---