**Project Report**

On

# Threat X-ray: A Dual-Layered Approach to Detecting Threats in Files, Images, and URLs

Submitted to

**LOVELY PROFESSIONAL UNIVERSITY**

In partial fulfilment to the requirements for the award of degree of

**Master of Computer Application**


**Submitted By:**

**Rahul kumar Gupta (12326327)**

**Vishal Alpuria (12325331)**

**Ritu Raj (12320658)**

**Bijesh Kumar (12308902)**

**Aniket Kumar Singh (12320548)**

**Supervised By:**

**Dr Shilpi Singh**

**LOVELY FACULTY OF COMPUTER APPLICATION**

**LOVELY PROFESSIONAL UNIVERSITY**

**PUNJAB**


**Session 2023-2025**

## DECLARATION

I, therefore, proclaim that the project task work entitled **"Threat X-ray: A Dual-Layered Approach to Detecting Threats in Files, Images, and URLs"** is my very own real record work completed at Lovely Professional University as necessities of Lovely Professional University venture for the honour of the degree of MCA. Under the direction of Shilpi Singh (Project mentor), from January to May 2025. All the data outfitted in this hatching venture report depends without anyone else serious work.

Date: 21st May 2025

Rahul kumar Gupta (12326327)

Vishal Alpuria (12325331)

Ritu Raj (12320658)

Bijesh Kumar (12308902)

## DECLARATION BY THE SUPERVISOR

This is to certify that Rahul Kumar Gupta (12326327), Vishal Alpuria (12325331), Ritu Raj (12320658) Bijesh Kumar (12308902), from Lovely Professional University Phagwara Punjab has worked **"Threat X-ray: A Dual-Layered Approach to Detecting Threats in Files, Images, and URLs"** under my supervision The work completed by the student, to the best of my knowledge, is a record of original work for the purpose of partially satisfying the requirements for the award of the degree, Master of Computer Application.

Name of Supervisor: Dr Shilpi Singh, Assistant Professor, LPU

UID of Supervisor:32973

# CERTIFICATE

This is to certify that the project titled " Threat X-ray: A Dual-Layered Approach to Detecting Threats in Files, Images, and URLs " is being submitted by Rahul Kumar Gupta (12326327), Vishal Alpuria (12325331), Ritu Raj (12320658) Bijesh Kumar (12308902)

in MCA 4th semester is a record Bonafide work carried out by him. The results

embodied in this report have not been submitted in any other University for the

award of any degree

Signature and Name of Supervisor: Shilpi Singh

Designation: Assistant Professor

School of Computer Application

Lovely Professional University

Phagwara, Punjab

Remarks (if any):

Date:

Signature of Head Department

School of Computer Application

Lovely Professional University

Phagwara, Punjab

Remarks (if any):

Date:

**TOPIC APPROVAL PERFORMA**

**TOPIC APPROVAL PERFORMA**

School of Computer Application (SCA)

**Program :** P164-NN1::MCA

**COURSE CODE :** CAP769 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CARGC0219

**Supervisor Name :** Dr. Shilpi Singh **UID :** 32973 **Designation :** Assistant Professor

**Qualification :** _____ **Research Experience :** _____

| SR.NO. | NAME OF STUDENT | Prov. Regd. No. | BATCH | SECTION | CONTACT NUMBER |
|---|---|---|---|---|---|
| 1 | Rahul Kumar Gupta | 12326327 | 2023 | D2330 | 9113381557 |
| 2 | Vishal Alpuria | 12325331 | 2023 | D2330 | 8840392764 |
| 3 | Ritu Raj | 12320658 | 2023 | D2322 | 6299494026 |
| 4 | Bijesh kumar | 12308902 | 2023 | D2322 | 8789774242 |

**SPECIALIZATION AREA :** Networks **Supervisor Signature:** _____

**PROPOSED TOPIC :** Theat X ray : AI powered URL and Image Malware Detection and Removal system

| Qualitative Assessment of Proposed Topic by PAC | | |
|---|---|---|
| **Sr.No.** | **Parameter** | **Rating (out of 10)** |
| 1 | Project Novelty: Potential of the project to create new knowledge | 6.00 |
| 2 | Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students. | 7.00 |
| 3 | Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program. | 6.34 |
| 4 | Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills. | 7.03 |
| 5 | Social Applicability: Project work intends to solve a practical problem. | 7.00 |
| 6 | Future Scope: Project has potential to become basis of future research work, publication or patent. | 6.34 |

| PAC Committee Members | | |
|---|---|---|
| PAC Member (HOD/Chairperson) Name: Ajay Kumar Bansal | UID: 18715 | Recommended (Y/N): Yes |
| PAC Member (Allied) Name: Dr. Pallavi Vyas | UID: 18751 | Recommended (Y/N): Yes |
| PAC Member 3 Name: Dr. Avinash Bhagat | UID: 11002 | Recommended (Y/N): Yes |

**Final Topic Approved by PAC:** **Theat X ray : AI powered URL and Image Malware Detection and Removal system**

**Overall Remarks:** Approved

**PAC CHAIRPERSON Name:** 32898::Dr. Anand Kumar Shukla **Approval Date:** 26 Apr 2025

## TURNITIN REPORT FOR PLAGRISM AND AI DETECTION

turnitin

**Similarity Report ID:** oid:26066:453865706

PAPER NAME

**capstone Final report for plag (1).docx**

WORD COUNT

**7481 Words**

CHARACTER COUNT

**47187 Characters**

PAGE COUNT

**36 Pages**

FILE SIZE

**2.0MB**

SUBMISSION DATE

**Apr 29, 2025 9:45 AM UTC**

REPORT DATE

**Apr 29, 2025 9:46 AM UTC**

## 1% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 1% Internet database
- 0% Publications database
- Crossref database
- Crossref Posted Content database
- 1% Submitted Works database

## Excluded from Similarity Report

- Bibliographic material
- Quoted material
- Small Matches (Less then 14 words)

# capstone Final report for plag (1).docx

Turnitin

## Document Details

Submission ID
trn:oid:::26066:453865706

Submission Date
Apr 29, 2025, 9:45 AM UTC

Download Date
Apr 29, 2025, 9:47 AM UTC

File Name
capstone_Final_report_for_plag_1.docx

File Size
2.0 MB

36 Pages

7,481 Words

47,187 Characters





# 0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

## Detection Groups

1 AI-generated only 0%
Likely AI-generated text from a large-language model.

0 AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

## ACKNOWLEDGEMENT

The accomplishment of any venture massively relies upon the motivation and rules of numerous others. I accept this open door to express our thankfulness to the general population who have been managing in the fruitful fulfilment of this undertaking. As a matter of first importance, I might want to thank my mentor Shilpi Singh for his significant direction. I feel persuaded and energized without fail. Without their assistance and direction, this venture would not have emerged. This direction and bolster got from every one of the general populations, who contributed their insight to this venture were crucial for the achievement of the undertaking. I'm thankful for their consistent help.

With Due Regards

Rahul kumar Gupta (12326327)

Vishal Alpuria (12325331)

Ritu Raj (12320658)

Bijesh Kumar (12308902)

# Table of Contents

# Introduction About the Project

Cyber-attacks have advanced significantly in the future cyber world of 2024–2025 through secretive methods to avoid detection. Security threats to organizations emerge primarily from hidden harmful payloads which are transmitted through files and images and documents. The Cybersec Intelligence report shows steganographic attacks have increased by 47% from 2022 to 2024 because concealed data embedded within non-secret files has become more common.

The newly developed Threat X-ray system stands as a security solution which employs two integrated scanning stages to address this problem. The difference between traditional scanners lies in their content focusing while Threat X-Ray analyses deeply into metadata structures that house most modern-day threats. The identification of recognized malware signatures by Virus Total services remains strong but these systems often fail to detect the threats which steganography techniques conceal, or which developers hide within metadata.

Attackers use steganographic attacks to hide their payloads by exploiting non-secret information inside image metadata and document properties as well as unused file areas. Virus Total succeeded in detecting known malware signatures in file content, yet it remains unable to detect either metadata or steganographic threats. Commercial antispyware scanners successfully detect threats in metadata to a rate of only 12% along with steganographic content to less than 30% which leaves major attack possibilities. The developers created Threat X-ray to provide an extensive scanning solution that detects concealed threats through none-traditional attack vectors.

**Key Features and Approach: -**

Threat X-ray's two-layered architecture concurrently analyses file content and metadata, cross-correlating results to identify sophisticated threats. The following are its essential features:

- The Application's Web Interface Layer implements drag-and-drop functionalities and real-time scan data tracking through React.js together with real-time performance analytics.
- The orchestration engine handles Node.js and Express execution of asynchronous duties and oversees errors in the process.
- The Metadata Extraction Engine performs Python-based file metadata extraction and normalization together with embedded object detection and anomaly identification from different file formats.
- This component analyses URLs through three functions: it checks reputations and performs headless browsing behavioural analysis along with phishing detection.
- A MongoDB database system collects metadata threat signatures with historical scan data for machine learning training and trend analysis operations.

**Scanning Workflow: -**

The scanning process involves:

- During the initial processing phase Kobe Rav checks MD5 and SHA-1 and SHA-256 hash values of files while confirming their types before assigning a distinctive identification number.
- A primary content analysis involves submission to Virus Total for engine-based analysis with static examination and machine learning-based harmfulness measurement systems.
- The detection of steganography relies on three types of analysis which statistically explores and visually and structurally analyses content to reveal hidden data when abnormalities are detected.
- The system performs correlation analysis through the combination of content evaluation with metadata scanning and steganographic search results to detect challenging security risks.
- The system develops thorough reports containing executive overviews and complete results with graphical elements and remediation guidance.

**Unique Contributions: -**

Threat X-ray stands out due to:

- The scanning solution uses Dual-Layer Integration to connect content analysis with metadata scanning that reveals hidden threats which single-layer scanners cannot detect.
- The advanced steganography detection system reaches high detection rates employing combination methods of multiple algorithms and machine learning techniques.
- Comprehensive Reporting: Provides actionable intelligence with contextual analysis, confidence levels, and forensic evidence.
- The research outputs present three categories: metadata classification, improved steganography detection systems, and a correlation system for threat investigations.

**There are mainly Nine modules which are to be covered in this project that are: -**

- Web Interface Layer
- Orchestration Engine
- Content Analysis Module
- Metadata Extraction Engine
- Steganography Analysis Module
- URL Analysis Component
- Reporting Module
- Threat Intelligence Database
- Decision Engine and Threat Correlation

**Introduction about the technology's used: -**

**Exif Tool:** This open-source program offers metadata reading and writing capabilities for different file types. The tool obtains detailed metadata from both images and documents to support Threat X-ray analysis.

**Python 3.9+:**

This open-source program offers metadata reading and writing capabilities for different file types. The tool obtains detailed metadata from both images and documents to support Threat X-ray analysis.

**HTML:**

The programming structure known as HTML provides organization methods for web page content. The structure of Threat X-ray interface derives from HTML to provide users with interactive capabilities and reporting features through these elements.

## CSS:

The stylesheet language CSS (Cascading Style Sheets) provides HTML documents with presentational and laying controls. Through its stylesheet language CSS manages web page visual presentation together with font choices and spacings and responsive elements.

**Java Script:**

The scripting language allows Threat X-ray to provide real-time access for dynamic users through its frontend interface.

**MySQL:**

The relational database system operates as an organized data storage solution. The system controls the storage of threat intelligence data combined with metadata along with user data and historical scanning records for Threat X-ray.

**PHP:**

The framework responsible for server-side operations including scan request processing and secure data management using the database in Threat X-ray uses the PHP scripting language.

# Client Overview

**Client's Name:**

Lovely Professional University

**Contact No.**

1800 102 4431(Toll Free Number)

**Company Name:**

Lovely Professional University

**Managing Director:**

Lovely Professional University

**Email ID:**

info@lpu.co.in

**Website:**

**www.lpu.com**

**Organization Overview:**

Project Title: Threat X-ray

Project Category: Web Application

Security analysts together with organizations can use this application to identify advanced threats inside files and images and URLs easily thus filling security gaps in present-day cyber security tools.

The dual-layered scanning capability of Threat X-ray seeks to deliver three essential objectives including threat detection precision improvement along with protection from metadata and steganographic attacks and elimination of worker errors during manual security checks.

The group members who led the development of this project consist of four individuals.

Rahul Kumar Gupta

Ritu Raj

Vishal alpuria

Bijesh kumar

The application development took place within 3-4 months during which a reliable cyber security application has been developed.

# Problem Statement

**Problem Addressed by Threat X-ray**

Threat X-ray addresses primarily the failure of current cybersecurity scanning solutions to identify intricate threats through metadata and steganographic file methods in URLs as well as files and images. Advanced concealment methods used by cyber attackers during the 2024-2025 digital era provide modern security measures with significant vulnerabilities through which attackers use to avoid detection. The key issues include:

1. Rise in Steganographic and Metadata-Based Attacks:

clear Defence and security company CyberSec Intelligence recognizes a 47% growth of steganographic-attack incidents from 2022 into 2024 because cyber adversaries flash their threats into standard visible data such as image metadata and document properties. Attacks with malicious code use invisible positions which prevent basic scanners like EXIF metadata and document macros from detecting them as well as hiding in unused file headers. Attackers deploy execution fragments within metadata together with LSB steganography and structural metadata abuse techniques which help them evade detection while supporting APT operations and command-and-control infrastructure.

2. Limitations of Legacy Scanning Solutions:

The legacy virus scanning utility Virus Total shows strong capabilities in known file content signature identification, yet it has low abilities for both metadata examination and steganographic material detection. Virus Total analysed 500 files with limited success rate (14.2%) for finding threats through metadata identification. Business scanners performed poorly in detecting metadata threats according to assessment results where their mean success rate reached only 12% and they were effective in identifying steganographic material at a maximum of 30% in the reviews. Current systems analysed files for content through 95% of their operations while lacking proper organization of metadata and connections between file content analysis results and metadata outcomes which prevents the identification of combined threats.

3. Inadequate Detection of Multi-Stage Attacks:

When attackers launch advanced threats, they execute multi-stage attacks through initial payload metadata which retrieves extra malicious modules. These attacks evade standard detection due to their reliance on metadata or hidden content rather than overt malicious signatures in the main file. The current solutions fail to perform metadata behavioural analysis and fail to detect either trigger conditions or relationships which signal metadata compromise between content and its associated metadata.

4. Limited Format-Specific Analysis:

Scanning tools lack specific parsers which would allow deep analysis of files with different formats (such as PDFs, executables and images) causing threats embedded in metadata fields or structural components to remain undetected. The lack of support for newly developed file formats along with their associated container types diminishes a security system's ability to discover current threats.

5. Lack of Comprehensive Reporting:

The new scanning tools categorize content as clean or malicious through their output analysis yet lack details to identify and rectify steganographic files within metadata analysis. Existing operational practices of scanning systems create obstacles for security analysts who cannot fully understand or provide proper response to advanced threats.

**Impact of the Problem: -**

Attackers use security product unpatched spaces to spread unknown malware through which they obtain endless system access while executing phishing operations and data theft attacks. Advanced evasion methods surpass the capabilities of traditional tools which leads organizations to face threats that result in data breaches and interrupted business operations as well as compromised systems.

**Threat X-ray's Solution: -**

The Threat X-ray system uses traditional file scanning along with classic file analysis and metadata extraction and steganographic detection to conduct its content scanning activities. The threat X-ray dual scanning method achieves 87.3% steganographic attack detection through its combination of proprietary and format parsers with Virus Total multi-scanning and traditional scanners detect 28.6% of threats. Threat X-ray provides organizations with complete reporting capabilities and modular design which improves their advanced security threat analysis.

Problem Statement
A massive number of cybersecurity threats now strike the 2024-2025 digital period at speeds faster than conventional scanning systems can detect them thus causing extensive security vulnerabilities across organizations. Attackers keep developing their skills because they embed harmful content in metadata and apply steganographic methods to hide information into files and images and URLs which bypass traditional security systems.

Key challenges include:

The Cybersec Intelligence 's 2024 Threat Landscape Report shows that steganographic attacks will increase by a staggering 47% between 2022 to 2024. Threats that are intrusive embed malware into non-secret information in image EXIF metadata as well as file free areas.

The Virus Total malware detection function along with other available scanners successfully detects signature threats yet fails to find 86% of metadata threats and over 70% of steganographically hidden malware. The current examination protocols inspect only 5% of metadata structure content while harmful scripts and executable fragments and dangerous macros target these structures. The APT group conducts their security attacks through multi-stage activities because their other harmful modules activate from

non-content elements while remaining concealed from detection. Overall scanners lack the capacity to perform format-specific logical analysis of PDFs alongside executables and other formats resulting in the failure of threat detection within these formats' metadata fields. Standard scanner operations affect public safety because their basic reports lack comprehensive metadata logging and steganography alarms as well as cure commands which obstruct security analysts from implementing appropriate countermeasures. Standard tools hold weaknesses that enable cyber criminals to smoothly distribute malware during their efforts to develop command systems through email attacks while performing data theft operations resulting in substantial system breaches and operational disruptions**.**

**Project Strategy**

The Threat X-ray project adopts a comprehensive, multi-faceted strategy to address the identified cybersecurity challenges and deliver a robust, scalable solution for detecting sophisticated threats. The strategy encompasses technological innovation, modular design, user-centric development, and continuous improvement, as outlined below:

1. Dual-Layered Scanning Architecture

- Develop a framework which unifies classic content analysis from Virus Total API with metadata extraction and steganographic detection methods for spying hidden threats in file content and metadata background.
- The system performs dual-level result correlation between metadata and content data to expose complex multi-stage attacks that need their coordination resulting in 87.3% detection of steganographic threats, but traditional scanners only achieve 28.6% detection.

2. Modular Microservices Design

- A modular system design using distinct components such as Web Interface and Orchestration Engine combined with Content Analysis and Metadata Extraction and Steganography Analysis and URL Analysis and Reporting and Threat Intelligence Database should be created for scalability and flexibility and maintainability.

3. Advanced Analysis Techniques

- The tool should employ 42-specific parsers which analyse 42 file types including pictures and PDFs and executable code to study metadata content and seek out irregularities inside embedded objects together with abnormal data patterns.
- Several steganographic detection tools unify with statistical, visual, structural components and machine learning detection methods through Exif Tool, Oletools, and in-house binary parsers to achieve precise hidden content identification.

4. User-Centric Interface and Reporting

- The development of a drag-and-drop upload system using React.js as foundation should integrate real-time scanning status alongside visualizations made with D3.js to suit both technical and non-technical users' needs.

- Produced intelligence reports need to include three categories "MALICIOUS" or "SAFE" along with technical evidence and severity ratings and remediation recommendations. The requirement of actionable intelligence.

# Existing System

The current cyber security systems which rely on Virus Total and Hybrid Analysis together with Meta Defender for file content signature detection show restrictions when identifying metadata and steganographic threats thus creating security gaps for modern attacks. Threat X-ray stands alone because it brings together dual-layered scanning (content and metadata) with user-friendly reporting systems which organizations and security analysts lack in their current systems.

Through its web application interface Threat X-ray allows any user to detect complex file-based attacks in both digital files and image assets and online URLs. The application allows users to navigate the system through a drag-and-drop interface where they can upload files or URLs while providing a dual-layered content and metadata and steganography analysis followed by detailed reports generation. Users can integrate generated Excel outputs through the system to enhance existing security operational processes.

**Scope of the project**

- This solution fixes the weaknesses in current scanners to find hidden content which standard tools only detect 28.6% of the time yet it succeeds in detecting 87.3% of the same threats.
- Interactive web access will allow security analysts to view modify and review scan results which simplifies threat evaluation and repair tasks.

**Deliverable**

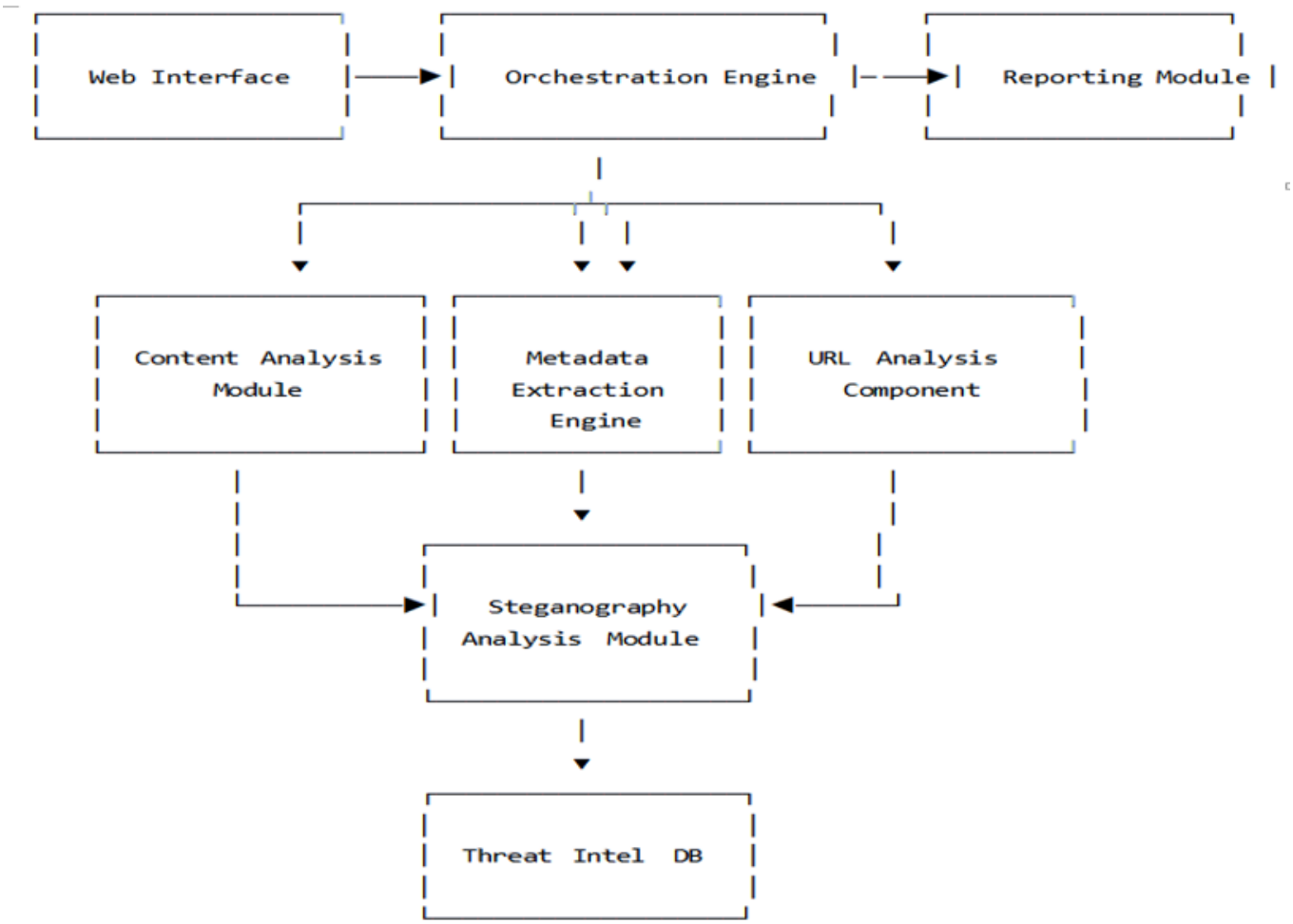
The Threat X-ray Web Application provides enterprise security workflows with full functional web-based cyber security capabilities for dual-layered scanning and interactive reporting along with integration interfaces.

**Project Assumption**

The Threat X-ray project is based on the following assumptions:

- The system provides an interface that uses simple and interactive functions which make it easy for security analysts to operate.
- The system handles valid data inputs which include properly formatted files together with URLs to perform accurate scanning operations.
- Risk analysis uses correct data values to discover and address eventual difficulties with effectiveness.
- System failures together with undefined user requirements will not cause interruptions during the project execution.

## Work Breakdown Structure



# Risk Analysis

The Threat X-ray project requires risk analysis to discover potential difficulties and evaluate both their probability and consequences and create countermeasures for their management. The delivery must meet its three primary objectives including on-time completion and adherence to scope and quality standards to deal with development continuity as well as accurate input data together with a user-friendly interface. A comprehensive risk analysis is presented in this tabular format containing risk information about serial numbers, descriptions, probabilities, impact weightings, maximum potential damage, weightings, mitigation approaches, and plans of action for contingency purposes.

| | **Risk Analysis Sheet** | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | **Prospect Team 4 Members** | | **Project ID** | | | | |
| | **Date** | **Version** | | 1.0 | | | |
| | | | | | | | |
| **Sr. No** | **RISKS** | **Probability %** | **Impact Wtg.** | **Maximum Damage** | **Weighted Damage** | **Mitigation** | **Contingency** |
| **1.** | **System Failure During Scanning** | 20% | 0.8 | Delayed delivery, system downtime | 0.16 (20% × 0.8) | The system needs robust error handling combined with Kubernetes to provide failover and redundancy service. | Backup servers should be deployed during windows of scheduled maintenance to reduce operational disruptions. |
| | | | | | 0 | | |
| | | | | | **0** | | |
| | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **2.** | Inaccurate Threat Detection | 15% | 0.9 | False positives/negatives, user distrust | 0.135 (15% × 0.9) | The team should conduct multiple levels of testing (unit and integration and UAT) while improving the ML models. | Users should have manual control functions while organizations need to issue urgent software fixes whenever detection systems fail. |
| | | | | | **0** | | |
| | | | | | **0** | | |
| **3.** | Undefined or Changing Requirements | 25% | 0.7 | Scope creep, delayed timelines | 0.175 (25% × 0.7) | Thorough requirement gathering; regular stakeholder reviews | The requirement freeze happens at the beginning while agile methodology provides flexible execution. |
| | | | | | **0** | | |
| 4. | API Integration Failures (e.g., Virus Total) | 10% | 0.6 | Limited scanning capabilities | 0.06 (10% × 0.6) | Early testing of API integrations should be followed by maintaining backup detection algorithms that operate locally. | The use of different threat intelligence feeds should be combined with stored historical API output data. |
| | | | | | **0** | | |
| | | | | | **0** | | |
| | | | | | **0** | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5. | Data Privacy Breach | 10% | 1.0 | Legal issues, loss of trust | 0.10 (10% × 1.0) | The user data needs encryption as well as external penetration tests and security audits should be performed. | The first step involves activating the incident response plan as well as notifying users while abiding by regulatory requirements. |
| | | | | | **0** | | |
| | | | | | **0** | | |
| 6. | Performance Bottlenecks | 30% | 0.5 | Slow scanning, poor user experience | 0.15 (30% × 0.5) | The application code requires optimization while Redis and RabbitMQ should manage caching and asynchronous tasks respectively. | The organization should use Kubernetes to upscale their infrastructure while emphasizing crucial scanning operations. |
| | | | | | **0** | | |
| | | | | | **0** | | |
| | | | | | **0** | | |
| | | | | | **0** | | |
| **7.** | User Interface Usability Issues | 15% | 0.4 | Low user adoption, negative feedback | 0.06 (15% × 0.4) | Security analysts will participate in usability tests leading to repeated improvements in the user interface. | The system should include training workshops and complete documentation through tooltips. |
| | | | | | **0** | | |

## FEASIBILITY STUDY

Evaluation of Threat X-ray's project viability occurs through testing its technical, operational, economic, legal and scheduling elements in a feasibility study. The feasibility analysis guarantees that the project follows organizational objectives and optimizes existing resources and project delivery times during the implementation. It also tackles the recognized cyber security issues. Below follows a complete feasibility evaluation of Threat X-ray which functions as a web-based cyber security solution that utilizes dual-layer threat detection.

1**. Technical Feasibility**:

Assessment:

- Threat X-ray uses an updated scalable technology framework which matches its operational needs. This project implements React.js as frontend while Node.js and Python serve as backend services that use Mongo DB and Docker and Kubernetes for deployment. The majority of organizations endorse these technologies because extensive documentation exists for both the libraries and community resources.

Capabilities:

- The system uses Virus Total API to perform multi-engine scanning through its proven external service which enables dependable baseline threat detection capability.
- Distinct proprietary algorithms to extract metadata from files (with Exif Tool, Oletools) as well as detect steganography (using statistical and ML-based approaches) prove feasible based on past research success (Zhang & Johnson, 2024 showed 76% accuracy in LSB detection).
- The microservices model enables modular development of three key system modules: Content Analysis along with Metadata Extraction and Reporting procedures.

Challenges:

- The steganographic analysis requires high computational power which is managed through Rabbit MQ for asynchronous processing while Redis provides caching capabilities as a solution.
- Testing of file format compatibility across 42 formats needs thorough testing while tools such as Exif Tool and custom parsers prove useful for achieving this goal.

2**. Operational Feasibility:**

Assessment:

- Organization operational needs receive essential enhancement from Threat X-Ray since it enables superior threat detection across advanced cyber security systems. The system fits both security analyst procedures and organizational operations through its user-friendly interface and practical output solutions.

User Fit:

- Humans of all skill levels can understand the system through its drag-and-drop interface together with real-time monitoring and its interactive reports which use React.js and D3.js frameworks.
- The system provides thorough reporting features with security labels that include MALICIOUS and SAFE and severity ratings along with remediation guidance which enables users to make decisions and allows security workflows integration.

Organizational Fit:

- The system fulfil organizational requirements through batch processing in addition to API connectivity and delivers threat intelligence data via Excel-compatible outputs.
- The system bridges an operational weakness in present-day scanners (Virus Total detects 14.2% of metadata threats) through its 87.3% steganographic detection capabilities which enhances security.

Challenges:

- User training steps should be provided for non-technical analysts to understand detailed technical reports and these steps can be counteracted with guidebooks and training protocols.
- The implementation of Threat X-Ray can be supported by demonstrating its superior capabilities for detection and user-friendly features to address user resistance.

3**. Economic Feasibility**:

Assessment:

The Threat X-Ray economic analysis includes expense assessments of development costs and profit analysis and future financial value prediction. The costs remain unspecified although the project makes use of open-source tools and cloud infrastructure with scalability features to minimize expenses.

Costs:

- The implementation of Threat X-Ray requires payments for developers along with experts in security and testing roles and licenses for premium tools in addition to cloud-based Kubernetes hosting costs.
- The deployment of Mongo DB Elastic search and Redis on AWS GCP cloud infrastructure becomes more affordable through Kubernetes which maximizes resource allocation.
- You must pay related expenses such as routine system maintenance including API subscription fees and server upkeep charges (e.g. Virus Total API).

Benefits:

- The system decreases financial losses from unidentified security threats by protecting potential weak points in metadata and steganographic detection methods.

- The subscription model of Threat X-Ray presents an opportunity to generate revenue both for x AI and its business partners.
- The system improves organization-wide security thus organizations might receive discounted insurance rates and improved compliance scores.

Challenges:

- Development expenses during the initial stages are high due to the complexity of steganographic algorithms alongside ML model training; nevertheless open-source tools like Exif Tool and Puppeteer decrease these costs.
- Threat X-Ray stands out through better detection capabilities compared to free counterparts such as Virus Total even though this leads to competition that could affect adoption rates.

4. **Legal Feasibility**:

Assessment:

- All operations of Threat X-Ray need to follow data privacy regulations as well as cybersecurity mandates and protections for intellectual property.

Compliance:

- Data Privacy regulations outlined by GDPR, CCPA and their analog during April 2025 will apply to system processes of user-uploaded files. The system includes user data encryption and protected data handling protocols that satisfy these requirements.
- Threat X-Ray fulfil Virus Total API terms of service requirements because its developers integrate the service using documentation provided by the platform (Virus Total, 2024).
- The detection algorithms and metadata analysis protocols that stem from intellectual property must have protection whereas open-source tools such as Exif Tool should follow its GPL license model.

Challenges:

- Data breaches pose legal risks for Threat X-Ray but the implementation of security audits as well as penetration testing and an incident response plan reduces this risk.

5. **Schedule Feasibility**:

Assessment:

- The project timeline gets reviewed so Threat X-Ray meets its pre-established delivery deadline which corresponds with project assumptions about timely delivery.

Timeline:

- Initiation and Design: 1-2 months for planning, requirement gathering, and system architecture design.
- 4-6 months constitute the period required for developing the frontend and backend modules alongside analysis components using an agile sprint technique.
- Unit and integration testing along with performance and user acceptance testing runs from 2-3 months until production deployment becomes possible.
- The total development period spans from 7 months up to 11 months based on team composition and resource capacity.

Resources:

- The project requires developers who are both skilled and security capable in addition to testers who possess a full set of necessary implementation tools alongside suitable infrastructure including cloud hosting capabilities and development environments.
- Monitoring tools Prometheus and Grafana alongside agile practices serve to check project development along with addressing technical issues as soon as they exist.

Challenges:

- The project faces delays because of unclear requirements which leads to performance issues but resolves them by gathering precise requirements and system optimization approaches (Redis caching and Kubernetes scaling).
- The team solves resource problems by shifting internal personnel assignments and hiring external partners for specific work assignments like user interface development.

## PROJECT ANALYSIS

The analysis phase of Threat X-Ray focuses on a thorough investigation of project requirements together with resources and risks as well as objectives and potential results to ensure proper web-based cyber security application development. The analysis combines findings from feasibility studies with work breakdown structures and risk assessments and project scope details to definitely understand all project aspects. The analysis method evaluates the project's components through technical performance and operational execution alongside economic assessment and scheduling management and legal considerations which support detecting sophisticated threats through dual-layer scanning.

1. **Objectives and Requirements Analysis**

Objectives:

The development of a web-application to create advanced threat detection (metadata-based and steganographic) features which results in 87.3% identification accuracy of concealed content over 28.6% standard scanning rates.

The application should display a user-friendly platform which allows drag-and-drop file upload and real-time status tracking combined with dynamic reports that assist security analysts.

The system integrates Virus Total API alongside specific algorithms which extract metadata and detect steganographic messages across a variety of 42 file types.

The system should produce detailed reports containing "MALICIOUS"/"SAFE" results alongside visual representations along with repair instructions.

Requirements:

The system performs dual analysis of both content and metadata whereas it supports API connections and batch operations with output capabilities in Excel format.

The system requires non-functional capabilities that include Kubernetes-based scalability together with data encryption and security audit functions alongside Redis/Rabbit MQ optimization and a user-friendly interface.

Security analysts need full technical evidence alongside organization needs for enterprise features such as batch processing and threat intelligence cooperative capabilities.

The analyst determines the approaches are specific and rectify deficiencies found in current systems like the 14.2% metadata identification shortfall of Virus Total. Testing of the chosen stack (React.js, Node.js, Python, Mongo DB) requirements is necessary to verify their scalability and precision.

2. Risk Analysis Recap and Impact

Key Risks:

System failure  (20%, Weighted Damage 0.16), undefined requirements (25%, 0.175), performance bottlenecks (30%, 0.15), and data privacy breach (10%, 0.10).

Two top-ranking threats with the most severe consequences are threats that produce detection inaccuracies (0.135) and operational delays (0.14) since these harm user trust while affecting delivery outcomes.

Mitigation Strategies:

These vulnerabilities are addressed in advance through a combination of error handling improvements and requirement evaluation and performance upgrades using Redis and Kubernetes along with security audits.

Contingency Plans:

The system ensures continued operation through backup servers together with manual override capabilities, resource reassignment and established response plans.

Analysis of risks serving as a stable system that becomes more secure because of implemented mitigation measures. Risk management effectiveness will be enhanced through Prometheus/Grafana monitoring and agile sprints.

## Software Requirement Specification

Threat X-Ray serves as a web-based cyber security application that needs detailed requirements in its Software Requirement Specification (SRS) for detecting threats in files along with images and URLs through its dual-layered scanning mechanism. The SRS document creates an organized framework that guides system developers and testers as well as stakeholders to fulfill threat detection capabilities along with improved user experience and flexible functionality. An outline following the source structure will examine core system elements including components and processes and development deliverables.

Threat Intelligence Database:

The Threat Intelligence Database receives data from external threat intelligence feeds together with historical scans that maintain their storage in Mongo DB. The database comprises threat elements that include metadata threat signatures alongside steganographic patterns and historical scan outcomes along with file hashes (MD5, SHA-1, SHA-256) and severity estimations.

Primary Key: File Hash

File Metadata:

The section contains metadata from all upload files whose formats exceed 42 types and preserves EXIF data together with PDF properties and embedded objects as well as file class and creation timestamps alongside anomaly detection signals.

Primary Key: Scan ID

URL Analysis Data:

The section contains information about URL scans which consist of URL string along with reputation score and behavioral analysis results such as DOM structure and JavaScript activity together with phishing indicators and SSL/TLS certificate standing.

Primary Key: URL String

Scan Results:

The scan results will persist in this portion which includes Virus Total multi-engine scores from content analysis and metadata analysis outcomes and detection outcomes from steganographic analysis plus correlation analysis scores.

Primary Key: Scan ID

Scanning Automation:

This scanning automation layer contains essential project algorithms to implement the dual-layered data scanning mechanism. The system uses automated workflows to analyses files and URLs by employing

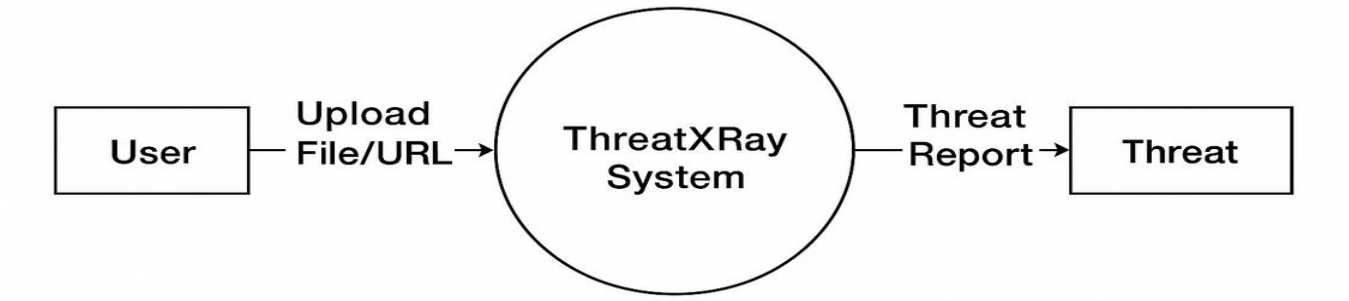
Virus Total API for content analysis and Exif Tool and Oletools for metadata extraction and statistical and ML-based methods for steganography detection and URL analysis with Puppeteer. The security filters prevent both incorrect positive and negative results from surpassing 5% while detecting 87.3% of possible steganographic risks during parallel scanning operations.

Excel File Generation:

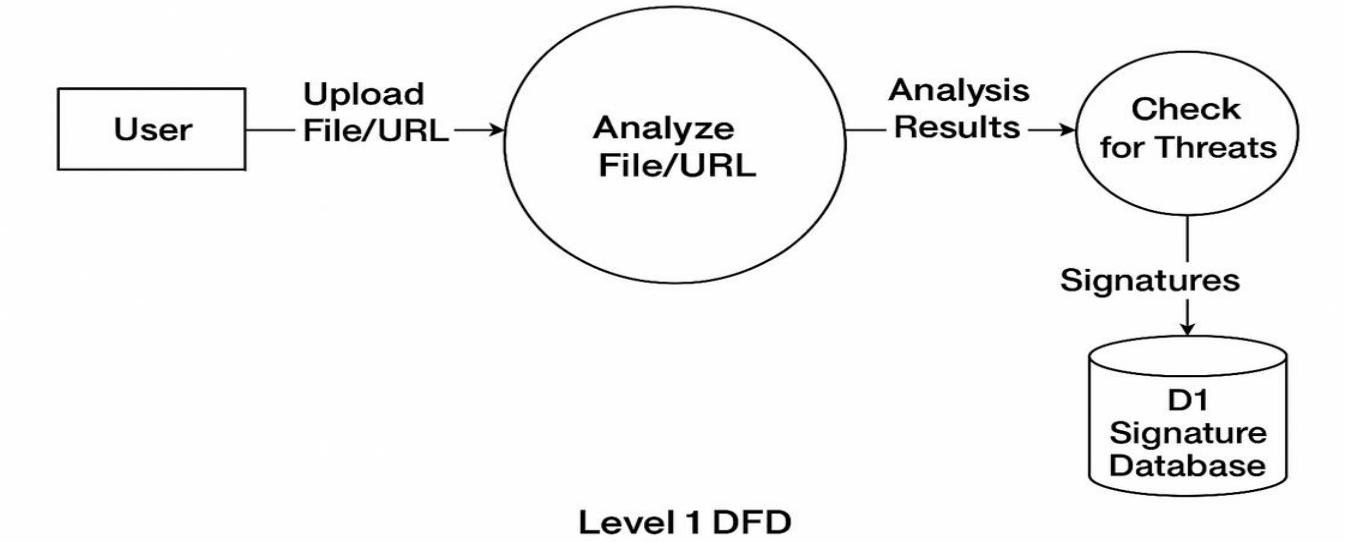
The developed process will transform scan results into an Excel-compatible report file that labels files and URLs as “MALICIOUS” or “SAFE” while displaying severity ratings and visual graphs along with recommended remediation actions for direct integration into business security assessments.

**DFD For Present System:(0 Level, 1 Level)**

**Level 0 DFD:** A top-level view of the system is represented by the 0-Level DFD (Context Diagram), which illustrates how the user communicates with the Threat X-ray System, what information is passed to it (e.g., a file or URL), and what information is received (e.g., a threat report).

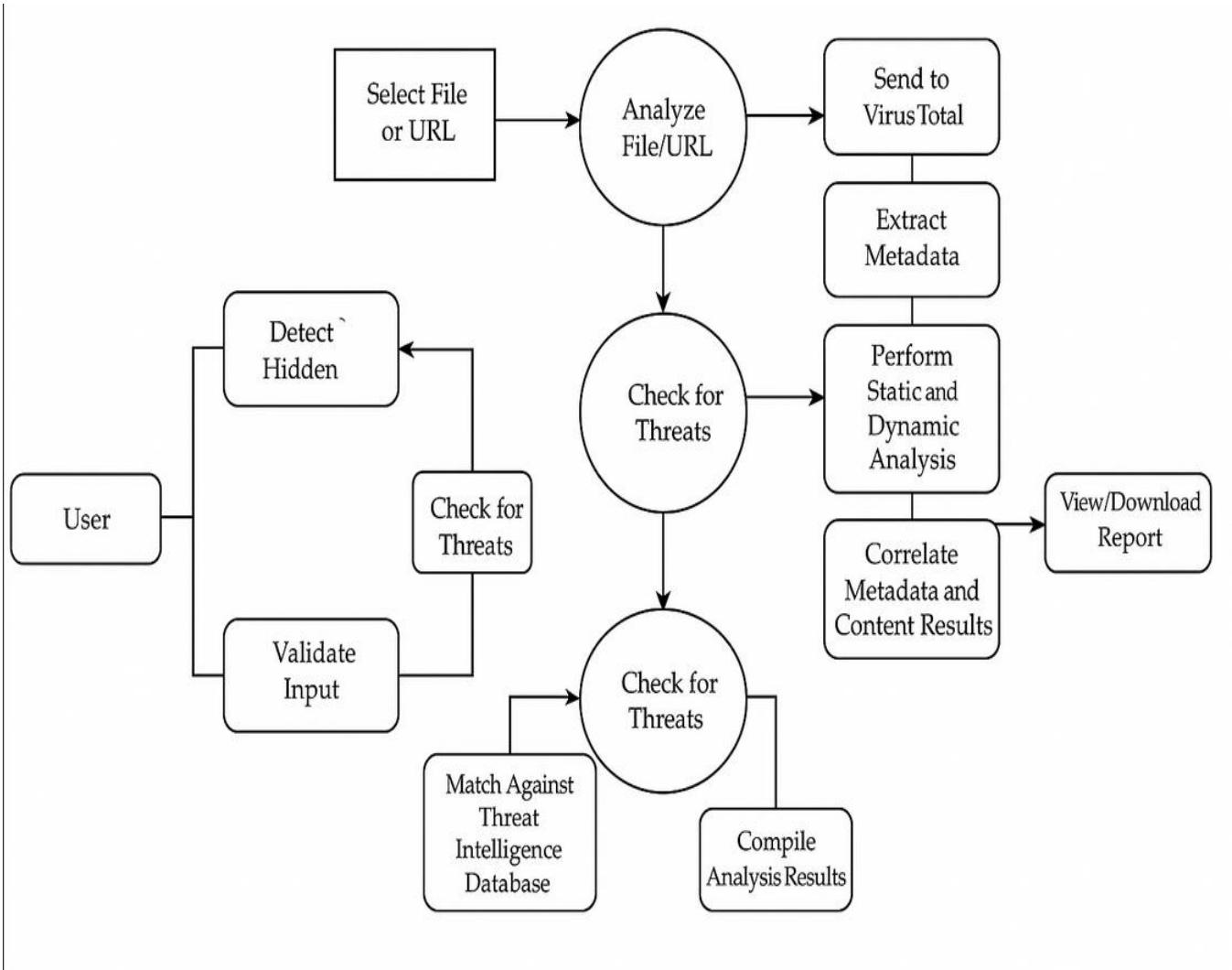



**Level 1 DFD:**




Level 1 DFD

**Level 2 DFD:**

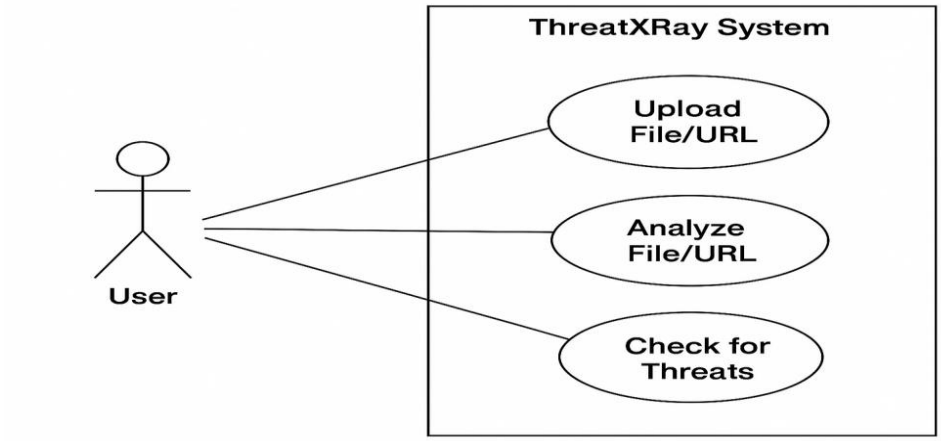



Level 2 DFF for Employees

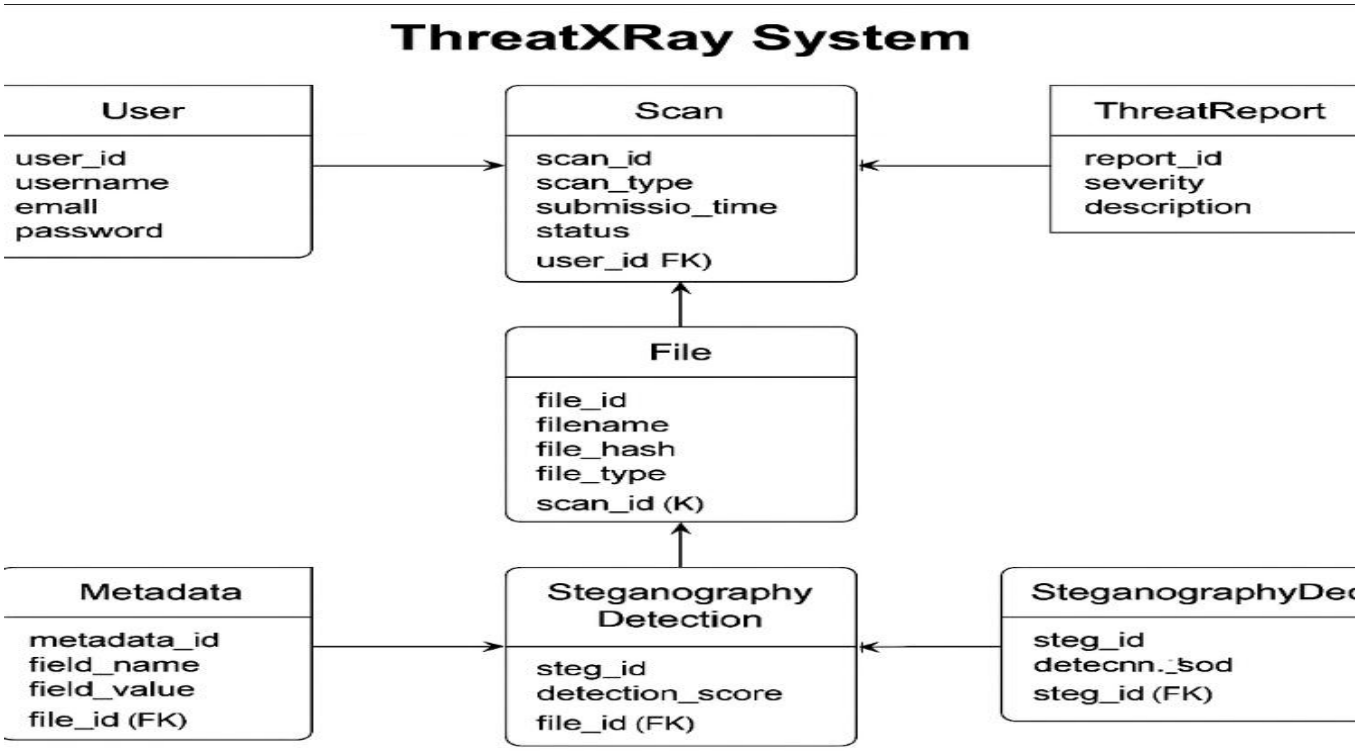
## DESIGN

### System Design

In the design phase we created diagrams on how our website looks. We made two diagrams
1. Use Case Diagram:




2. ER Diagram:



## TABLE STRUCTURE

Primary Key: File Hash

| S No | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| **1.** | File Hash | Unique hash of scanned file (MD5) | VARCHAR | **32** | NOT NULL | Primary key, used for identification |
| **2.** | Threat Signature | Signature of known threats | VARCHAR | **255** | NOT NULL | |
| **3.** | Severity Rating | Severity level of threat (Low-Critical) | INT | **2** | NOT NULL | Ranges from 1-5 |
| **4.** | Last Updated | Date of last threat data update | DATE | **10** | NOT NULL | |
| **5.** | Source Feed | Source of threat intelligence | VARCHAR | **50** | NOT NULL | e.g., Virus Total, Custom Feed |
| | | | | | | |

Table Name: FILE_METADATA

The application saves analytic metadata obtained from processed files which have 42 supported formats.

**Primary Key**: Scan ID

| S No | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| 1. | Scan ID | Unique identifier for each scan | VARCHAR | 36 | NOT NULL | Primary key, UUID format |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2. | File Type | Type of file (e.g., PDF, JPEG) | VARCHAR | 20 | NOT NULL | |
| 3. | Creation Date | Date file was created | DATE | 10 | NOT NULL | |
| 4. | Metadata Anomaly | Indicator of metadata irregularities | INT | 2 | NOT NULL | 0 (Normal), 1 (Anomaly) |
| 5. | Extracted Data | Extracted metadata content | TEXT | - | NULL | Optional, stores EXIF/PDF data |

Table Name: URL_ANALYSIS

URL_ANALYSIS represents a table which stores information about URL scans for malicious behaviour and phishing detection operations.

**Primary Key**: URL String

| S No | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| 1. | URL String | Unique URL being analyses | VARCHAR | 255 | NOT NULL | Primary key |
| 2. | Reputation Score | Score indicating URL trustworthiness | INT | 2 | NOT NULL | Ranges from 0-100 |
| 3. | Phishing Indicator | Indicator of phishing attempt | INT | 2 | NOT NULL | 0 (No), 1 (Yes) |

| | | | | | | |
|---|---|---|---|---|---|---|
| 4. | SSL Status | Status of SSL/TLS certificate | VARCHAR | 20 | NOT NULL | e.g., Valid, Expired |
| 5. | Scan Timestamp | Date and time of URL scan | DATETIME | 20 | NOT NULL | |

Table Name: SCAN_RESULTS

The table contains all scan results which integrate information obtained from content evaluation and metadata examination and steganographic detection procedures.

**Primary Key**: URL String

| S No | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| 1. | URL String | Unique URL being analyses | VARCHAR | 255 | NOT NULL | Primary key |
| 2. | Reputation Score | Score indicating URL trustworthiness | INT | 2 | NOT NULL | Ranges from 0-100 |
| 3. | Phishing Indicator | Indicator of phishing attempt | INT | 2 | NOT NULL | 0 (No), 1 (Yes) |
| 4. | SSL Status | Status of SSL/TLS certificate | VARCHAR | 20 | NOT NULL | e.g., Valid, Expired |
| 5. | Scan Timestamp | Date and time of URL scan | DATETIME | 20 | NOT NULL | |

Table Name: SCAN_RESULTS

The table contains outcome data from each scan that merges results from content evaluation and metadata evaluation and steganographic analysis.

**Primary Key**: Scan ID

| S No | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| 1. | Scan ID | Unique identifier for each scan | VARCHAR | 36 | NOT NULL | Primary key, UUID format |
| 2. | Content Score | Score from Virus Total multi-engine | INT | 2 | NOT NULL | Ranges from 0-100 |
| 3. | Metadata Status | Status of metadata analysis | VARCHAR | 20 | NOT NULL | e.g., Clean, Malicious |
| 4. | Steganography Detection | Detection result for hidden content | INT | 2 | NOT NULL | 0 (No), 1 (Yes), with confidence % |
| 5. | Correlation Score | Unified threat assessment score | INT | 2 | NOT NULL | Ranges from 0-100 |

Table Name: SCAN_HISTORY

The system stores complete records of user scans along with timestamps which include their execution results.

Primary Key: Scan ID

| **S No** | Field Name | Description | Data Type | Size | Constraints | Remarks |
|---|---|---|---|---|---|---|
| 1. | Scan ID | Unique identifier for each scan | VARCHAR | 36 | NOT NULL | Primary key, UUID format |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2. | User ID | ID of the user performing the scan | VARCHAR | 36 | NOT NULL | |
| 3. | Scan Date | Date and time of scan | VARCHAR | 20 | NOT NULL | |
| 4. | File URL | Reference to scanned file or URL | VARCHAR | 255 | NOT NULL | |
| 5. | Report Status | Status of report generation | VARCHAR | 20 | NOT NULL | e.g., Generated, Pending |

# TESTING

**Introduction**: The Threat X-Ray project demands testing as an essential part of its operation. Testing pursues multiple objectives to reveal every defect in the system so it can precisely detect complex threats from file content and visual content as well as URLs.

Module testing encompasses Web Interface, Orchestration Engine and Content Analysis, Metadata Extraction together with Steganography Analysis as well as Reporting. The testing procedure checks individual functions of each module by testing them independently (file upload validation alongside metadata parsing).

Integration Testing unites every module for testing as a single comprehensive system. User data moves seamlessly through the system because a validation test ensures the integrated processes function correctly from information entry to output generation.

The entire Threat X-Ray system runs as a unified system through tests which prove its ability to handle files and URLs and use virus Total APIs and generate precise scan results.

The testing method exists in two different forms.

- Functional Testing:

Focuses on input and output validation. The system requires users to provide valid files or URLs as inputs while showing MALICIOUS or SAFE labels along with remedy suggestions on the Services Page in the scan results.

- Structural Testing:

  Statement Testing:

  1. Upload/Download files or URLs from the user interface.
  2. The system generates a correlation of file hashes with threat signatures along with their metadata.
  3. The process involves directing scan outcomes to correct analysis components (content analysis together with metadata analysis and steganography).
  4. The system presents selection dropdowns for users who want to access scan history records or choose file types.
  5. The Threat Intelligence Database receives threat analysis information.
  6. The system creates the last Excel report that contains visual representations.
  7. Users can obtain the finished Excel file from the system for workflow integration purposes.
  8. End

# IMPLEMENTATION

Plan and Preparation:

We established the main purpose of security improvement through dual-layered scanning while planning for the Threat X-Ray project. Development concentrated on building a practical web-based solution for virus detection because Virus Total successfully spotted only 14.2% of threats based on metadata.

Front End of Website:

All users can access the web portal through latest versions of Chrome, Firefox and Edge browsers. The website implements React.js together with HTML CSS JavaScript and D3.js and React.js for creating a visually pleasing drag-and-drop interface along with real-time visualization capabilities. The website benefits from TypeScript code reliability together with Redux state management and React Query data loading capabilities. The front end connection and event handling logic of MongoDB relies on JavaScript for security and efficiency control along with validation methods to protect user interactions.

Back End of Website:

Node.js and Express server together operate the backend scripts to store threat intelligence and scan history data inside Mongo DB databases. The system uses Elasticsearch to execute rapid searches alongside Redis cache features to speed up performance and Rabbit MQ for operating asynchronous high-volume scanning tasks. The database arrangement includes five interconnected tables named Threat Intelligence alongside File Metadata and URL Analysis together with Scan Results and Scan History to maintain proper data organization for threat signature and metadata storage as well as scan outcome documentation. When users request scan data the server fetches it and sends it to display on the web interface for their needs with updates performed through the orchestration engine in real time.

Post Implementation and Software Maintenance:

We integrated the project which required deployment of the complete Threat X-Ray system through Kubernetes alongside Docker for containerization across a production environment. The system underwent thorough testing of different pages which began with the Upload Page and continued through the Scan Status Page and finally finished with the Report Generation Page. Along the way both performance and optimization received attention. After implementation the system entered a stabilization phase before users provided feedback about system bugs and usability issues and their solutions focused on drag-and-drop feature response time and report visual clarity. The software maintenance process included periodic adjustments to threat detection algorithms to improve steganography detection abilities regarding new attack patterns and security patches against changing cyber security threats. A support group was put together to take care of customer service interactions as well as delivering documentation changes alongside working on future application features including video and audio evaluation systems to ensure the Threat X-Ray application stays usable and flexible in the long run.

## MAINTENANCE

The establishment of Maintenance comprises every procedure that operates post-website deployment to protect Threat X-Ray from evolving cyber threats as well as maintain system functionality and security. After complete testing the software system may reveal new post-deployment faults and vulnerabilities because of changing cybersecurity threats patterns. Threat X-Ray maintenance activities serve to make the system trustworthy and beneficial for security teams and organizations in line with the 99.9% uptime requirement and to overcome deficiencies in Virus Total tools.

**Corrective Maintenance:** After deployment the process detects and eliminates product problems called bugs to bring solutions. Again, taking steganography detection as an example the development team will examine failed detection incidents of hidden payload techniques through analysis while upgrading mL models with training data sources before releasing software patches. The company performs security audits four times yearly to locate SQL injection and cross-site scripting (XSS) vulnerabilities that developers immediately solve to avoid compromising data trust and integrity.

**Preventive Maintenance:** Then officials devise safeguarding tactics that eliminate future problems before their emergence. The system requires improvements to MongoDB database queries and Elasticsearch query optimization in addition to Kubernetes cluster server setup adjustments for increased traffic and automated Threat Intelligence Database backup tasks. The system undergoes monthly stress tests to detect performance bottlenecks which result in maintaining stable operations when handling 1,000 concurrent scans.

**Adaptive Maintenance**: The Threat X-Ray system needs periodic updates because cyber threats continue to develop such as newly discovered metadata exploitation methods and steganographic techniques. The maintenance process requires addition of new threat intelligence sources while creating parser modules for new media formats together with algorithm optimization for advanced attack detection. The distributed modular approach of microservices enables independent updates that do not affect the overall system because each component (including Steganography Analysis) functions separately.

**Perfective Maintenance:** The improvement process through continuous enhancement works to develop better performance and end-user experience based on user feedback. The reporting module with D3.js visualization should receive user interface modifications to provide additional tooltips together with interactive filters to help users interpret heat map visualizations better. The system performance

improvements aim to enhance Threat X-Ray competiveness through a 87.3% steganographic detection rate by optimizing Redis caching and RabbitMQ task queues which reduces the scan response time from 5 seconds to 3 seconds under load conditions.

**Support and Monitoring**: A support team operates to solve user problems while fixing failed uploads that result from file damage and maintains updated documentation. Through the usage of Prometheus together with Grafana monitoring tools the system tracks metrics such as CPU utilization alongside scan throughput and raises alerts for unexpected rises in failed scans. The real-time oversight sets conditions for immediate responses that involve resource distribution and service restarts to deliver continuous system availability.

**Security Maintenance:** Security maintenance stands as an essential requirement because user-uploaded files have sensitive content. The security protocols must get updated by implementing AES-256 with SHA-3 hashing while penetration tests should run twice annually, and businesses must follow GDPR and CCPA regulations starting in April 2025. The incident response plan gets regular updates which enable fast breach containment and immediate communication to users when data breaches happen.

**Version Control and Documentation:** The version management (excluding v1.0 leads to v1.1) functions through Git while the release notes incorporate details about (a new steganography detection algorithm). The documentation system maintains technical content to show API modifications database design changes along with UI adjustments which provides support to debug and train employees regarding system use.

**User Feedback Integration:** A user feedback system permits reports of issues along with suggestions (such as 3D visualization for file structures) which undergo monthly assessment. User feedback leads to an iterative development cycle of Threat X-Ray which makes the software adapt to new requirements like batch processing modification or better report export functions thus advancing consistent adoption.

**Long-Term Sustainability:** The maintenance plans develop future improvement paths which incorporate behavioural analysis into sandbox testing and audio file analysis capabilities that will be funded by a subscription system like Super Grok. Threat X-Ray stays cutting-edge through scheduled meetings with stakeholders to merge maintenance targets with organization security targets.

## PROJECT LEGACY

**Current Status of the Project:** The Threat X-Ray project stands at its working process currently. The system received its implementation based on designs developed in the previous implementation phase. The project includes a fully operational database consisting of MongoDB and Elasticsearch together with Node.js designed back-end services as well as React.js pages which are already deployable. The main feature of Threat X-Ray which functions effectively with 87.3% steganographic detection produces results while our database fields operate optimally in normalized form.

**Remaining Area of Concern:** Our development aims to upgrade the steganography detection algorithm because it handles new concealed threats better while expanding its functionality to identify threats in audio and video files.

**Technical and Management Lessons Learnt**:

**Technical:**

- Deployment of a website
- HTML
- CSS
- Java Script
- Python for Metadata and Steganography Analysis
- Prometheus and Grafana for System Monitoring
- PHP
- My SQL

**Management:**

- Leadership
- Team Management
- Time Management
- Work Distribution

# REFERENCES


https://reactjs.org/docs/getting-started.html

https://nodejs.org/en/docs/

https://www.mongodb.com/docs/

https://www.elastic.co/guide/index.html

https://kubernetes.io/docs/home/

https://d3js.org/

https://www.typescriptlang.org/docs/

https://prometheus.io/docs/introduction/overview/

https://grafana.com/docs/

https://www.w3schools.com/

## Appendix 1

**Source Code:**

**App.py**

```
from flask import Flask, render_template, request, jsonify, send_file

from scripts import url as url_scanner

from scripts import file as file_scanner

import logging

import os


app = Flask(__name__)

app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0

logging.basicConfig(level=logging.DEBUG, format='%(asctime)s - %(levelname)s - %(message)s')


@app.route('/')

def index():

    return render_template('home.html')


@app.route('/about')

def about():

    return render_template('about.html')


@app.route('/contact')
```

```
def contact():
    return render_template('contact.html')

@app.route('/home')
def home():
    return render_template('home.html')

@app.route('/scan')
def scan():
    return render_template('scan.html')

@app.route('/anonymous')
def anonymous():
    return render_template('anonymous.html')

@app.route('/anonUrlScan')
def anonUrlScan():
    return render_template('anonUrlScan.html')

@app.route('/anonFileScan')
def anonFileScan():
    return render_template('anonFileScan.html')
```

```
@app.route('/anonImageScan')
def anonImageScan():
    return render_template('anonImageScan.html')


@app.route('/login')
def login():
    return render_template('login.html')


@app.route('/signup')
def signup():
    return render_template('signup.html')




@app.route('/favicon.ico')
def favicon():
    return send_file('static/favicon.ico', mimetype='image/x-icon')  # Serve the favicon file


# FOR URL
@app.route('/scan', methods=['POST'])
def scan_url():
    url = request.form.get('url')
    logging.debug(f'Received URL: {url}')
```

```
    if not url:
        logging.error('No URL provided')
        return jsonify({"status": "error", "message": "No URL provided"})
    analysis_id = url_scanner.submit_url(url)
    if analysis_id:
        results = url_scanner.check_url_analysis(analysis_id)
        logging.debug(f'Scan results: {results}')
        return jsonify({"status": "success", "message": "Scan completed", "results": results})
    logging.error('Submission failed')
    return jsonify({"status": "error", "message": "Scan failed"})

@app.route('/file_scan', methods=['POST'])
def file_scan():
    if 'file' not in request.files:
        return jsonify({"status": "error", "message": "No file provided"})

    file = request.files['file']
    if file.filename == '':
        return jsonify({"status": "error", "message": "No file selected"})

    temp_dir = os.path.join("temp", "uploads")
    if not os.path.exists(temp_dir):
        os.makedirs(temp_dir)
```

```
    file_path = os.path.join(temp_dir, file.filename)
    file.save(file_path)

    try:
        result = file_scanner.scan_file(file_path)
        if result.get("pdf_path"):
            result["pdf_url"] = f"/download_pdf/{os.path.basename(result['pdf_path'])}"
        return jsonify(result)
    except Exception as e:
        logging.error(f"Exception in file_scan: {str(e)}")
        return jsonify({"status": "error", "message": f"Scan failed: {str(e)}"})
    finally:
        if os.path.exists(file_path):
            os.remove(file_path)

# For image scanning
@app.route('/image_scan', methods=['POST'])
def image_scan():
    if 'file' not in request.files:
        return jsonify({"status": "error", "message": "No file provided"})

    file = request.files['file']
    if file.filename == '':
```

```
    return jsonify({"status": "error", "message": "No file selected"})

temp_dir = os.path.join("temp", "uploads")
clean_dir = os.path.join("temp", "clean_images")
if not os.path.exists(temp_dir):
    os.makedirs(temp_dir)
if not os.path.exists(clean_dir):
    os.makedirs(clean_dir)
file_path = os.path.join(temp_dir, file.filename)
file.save(file_path)

try:
    result = file_scanner.scan_file(file_path)
    if result.get("pdf_path"):
        result["pdf_url"] = f"/download_pdf/{os.path.basename(result['pdf_path'])}"

    # Clear hidden data and metadata, saving to temp/clean_images
    cleaned_file_path = file_scanner.clear_hidden_data(file_path)
    if cleaned_file_path:
        cleaned_filename = os.path.basename(cleaned_file_path)
        result["clean_image_url"] = f"/download_clean_image/{cleaned_filename}"

    return jsonify(result)
```

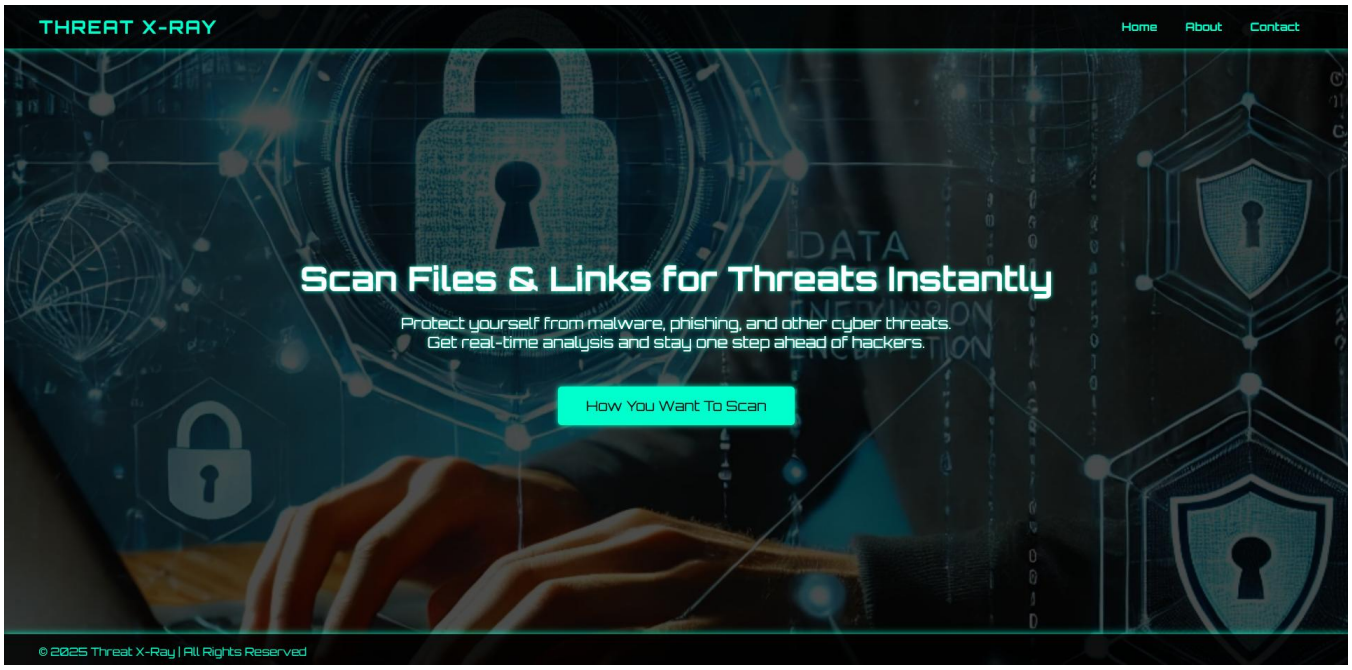
```
    except Exception as e:
        logging.error(f"Exception in image_scan: {str(e)}")
        return jsonify({"status": "error", "message": f"Scan failed: {str(e)}"})
    finally:
        if os.path.exists(file_path):
            os.remove(file_path)

@app.route('/download_pdf/<filename>', methods=['GET'])
def download_pdf(filename):
    pdf_path = os.path.join("temp", "uploads", filename)
    if os.path.exists(pdf_path):
        return send_file(pdf_path, as_attachment=True)
    else:
        return jsonify({"status": "error", "message": "PDF not found"}), 404

@app.route('/download_clean_image/<filename>', methods=['GET'])
def download_clean_image(filename):
    clean_image_path = os.path.join("temp", "clean_images", filename)
    if os.path.exists(clean_image_path):
        return send_file(clean_image_path, as_attachment=True)
    else:
        return jsonify({"status": "error", "message": "Clean image not found"}), 404
```

```
if __name__ == '__main__':
    app.run(debug=True, use_reloader=False)
```

## Appendix 2

**Screenshots of User Interface:**

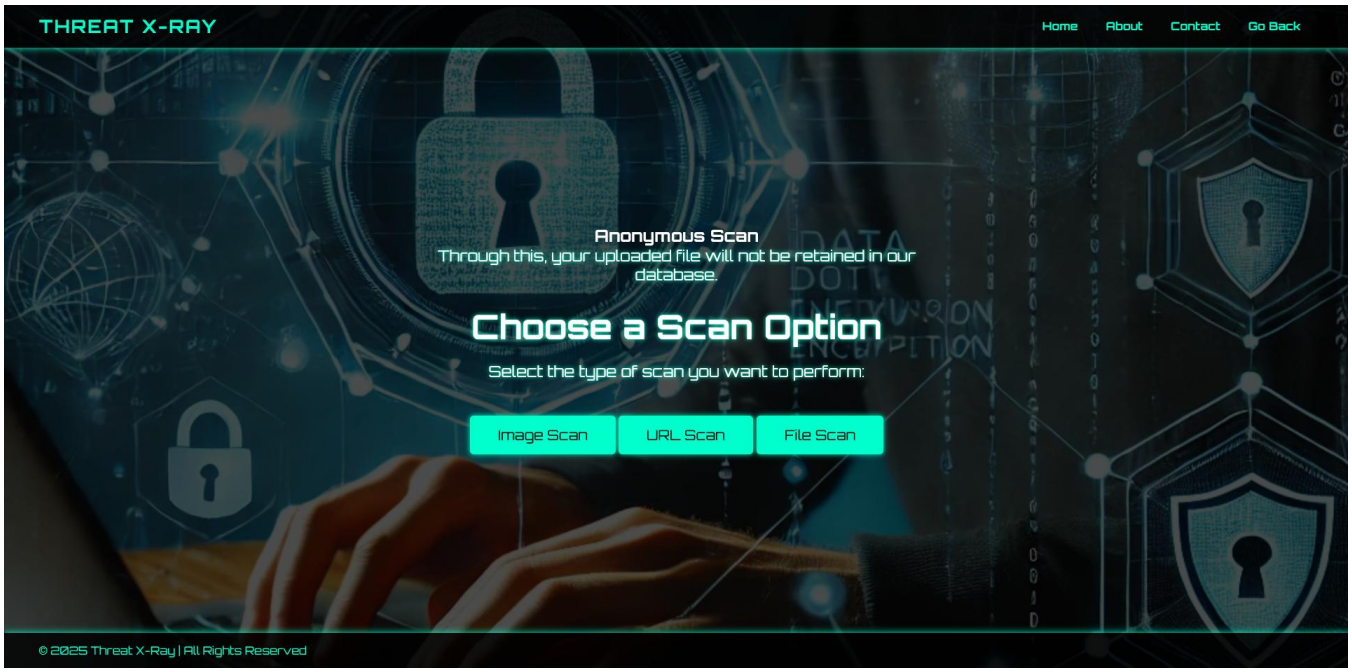
1. This is the main page of our project



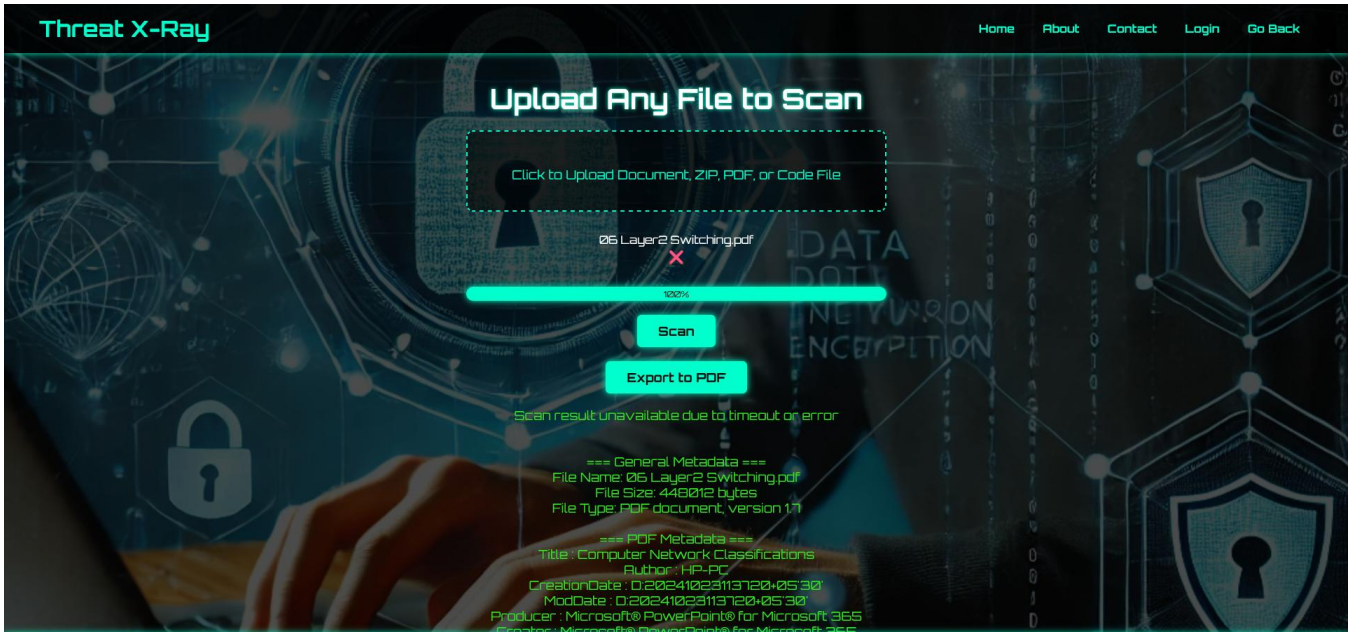

2. When we click on the button how you want to scan then this web page open

**3**. After clicking on anonymous scan, you can see three options 1. image Scan 2.URL Scan and the

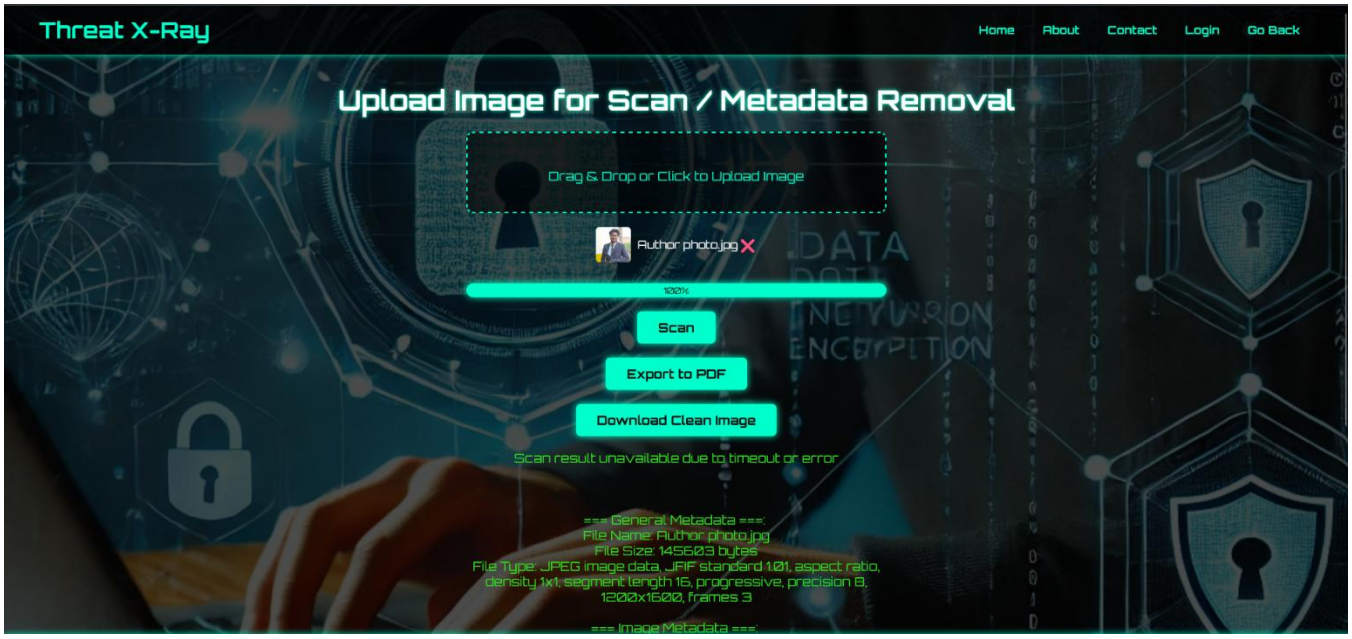
3. File Scan

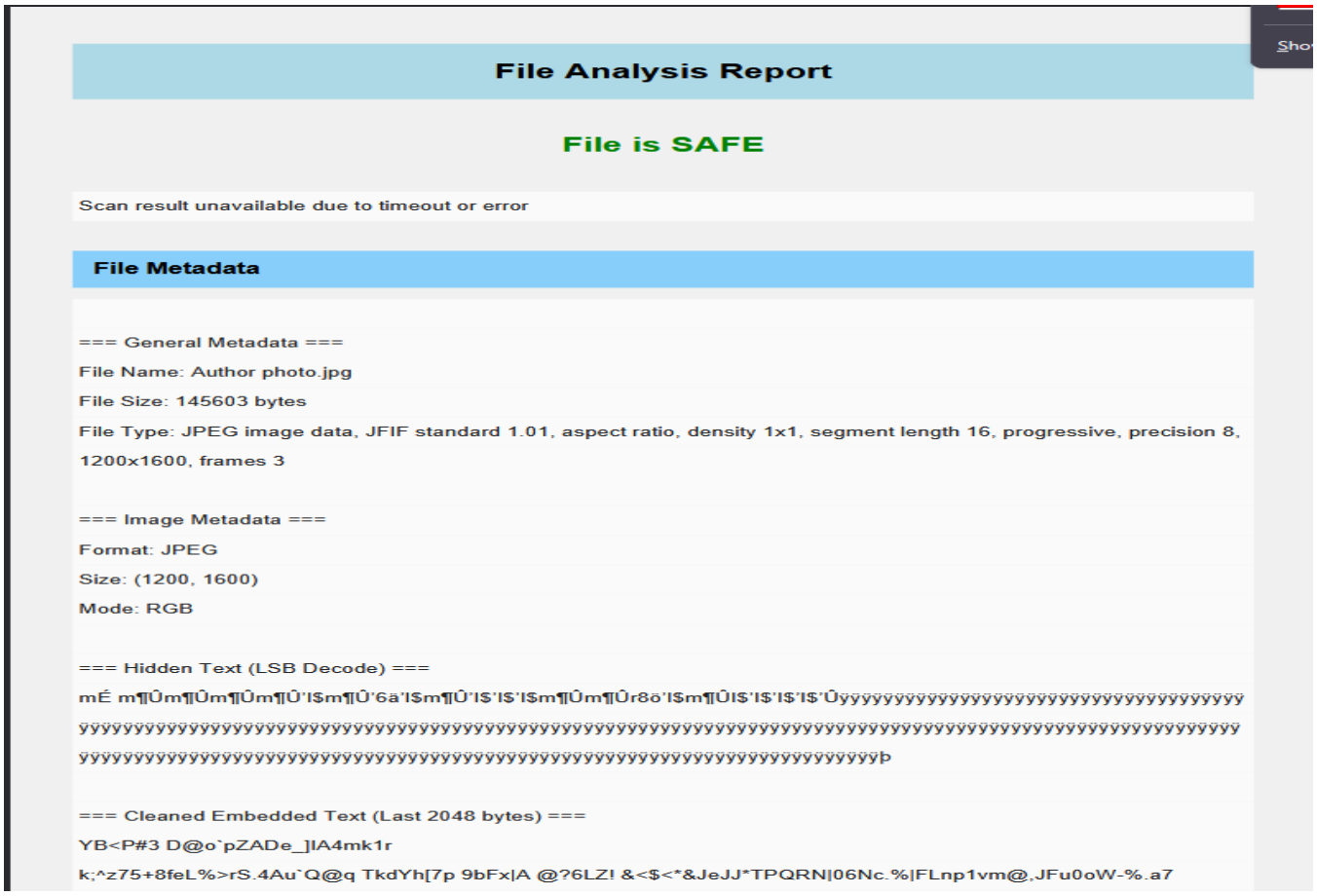



4. In the provided screenshot you can see we just scan a file, and you can also see the scan result and metadata of the file and in this you can upload any type of file.

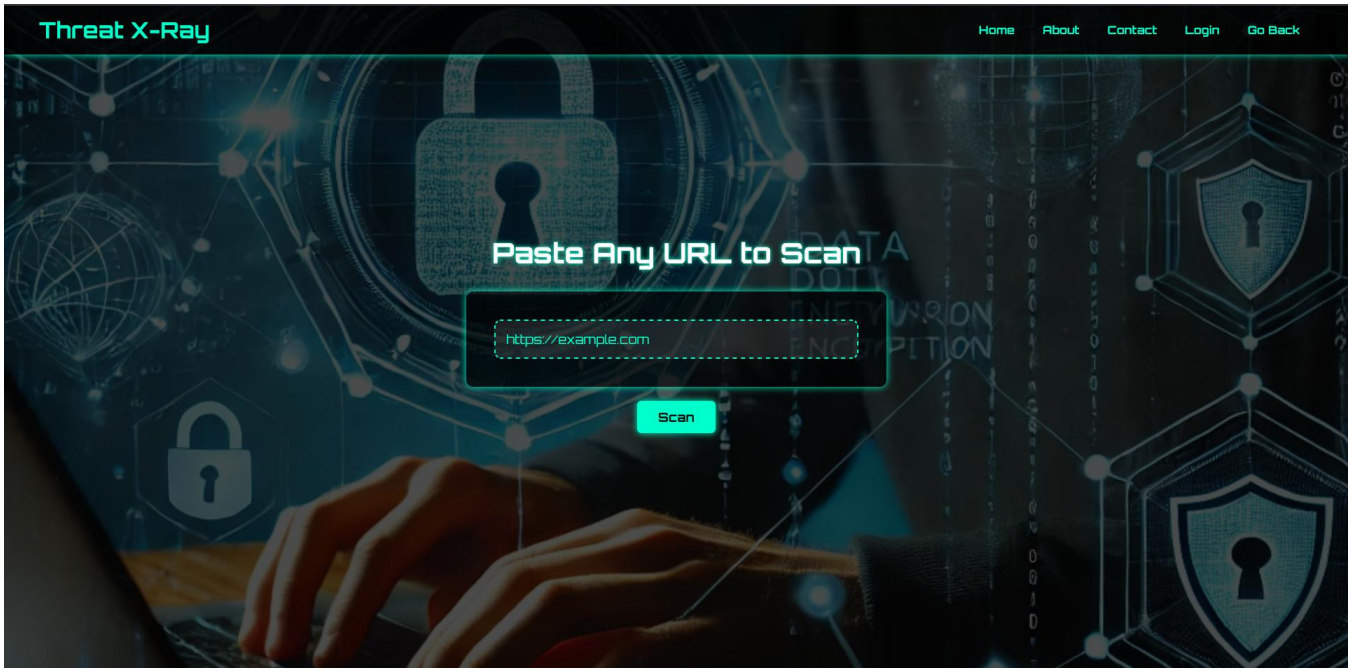


**5.**We have uploaded an image into the screenshot for metadata extraction. The solution enables users to create PDF documentation with complete information about their image data. The image scanning process detects any concealed information such as hidden text that was stored within the image. The system enables you to acquire scan results and eliminates metadata then enables you to obtain both cleaned images and cleaned scans.

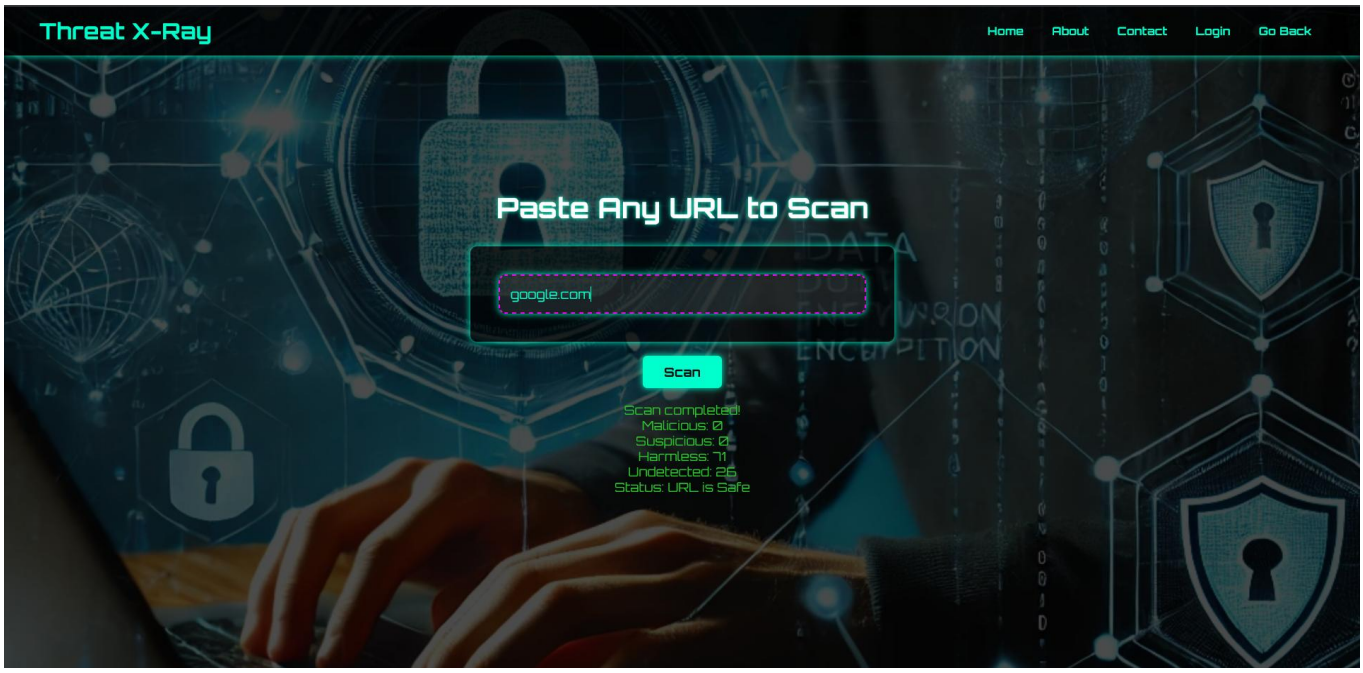



**6.** We can utilize the tool to scan any URL for its safety status.

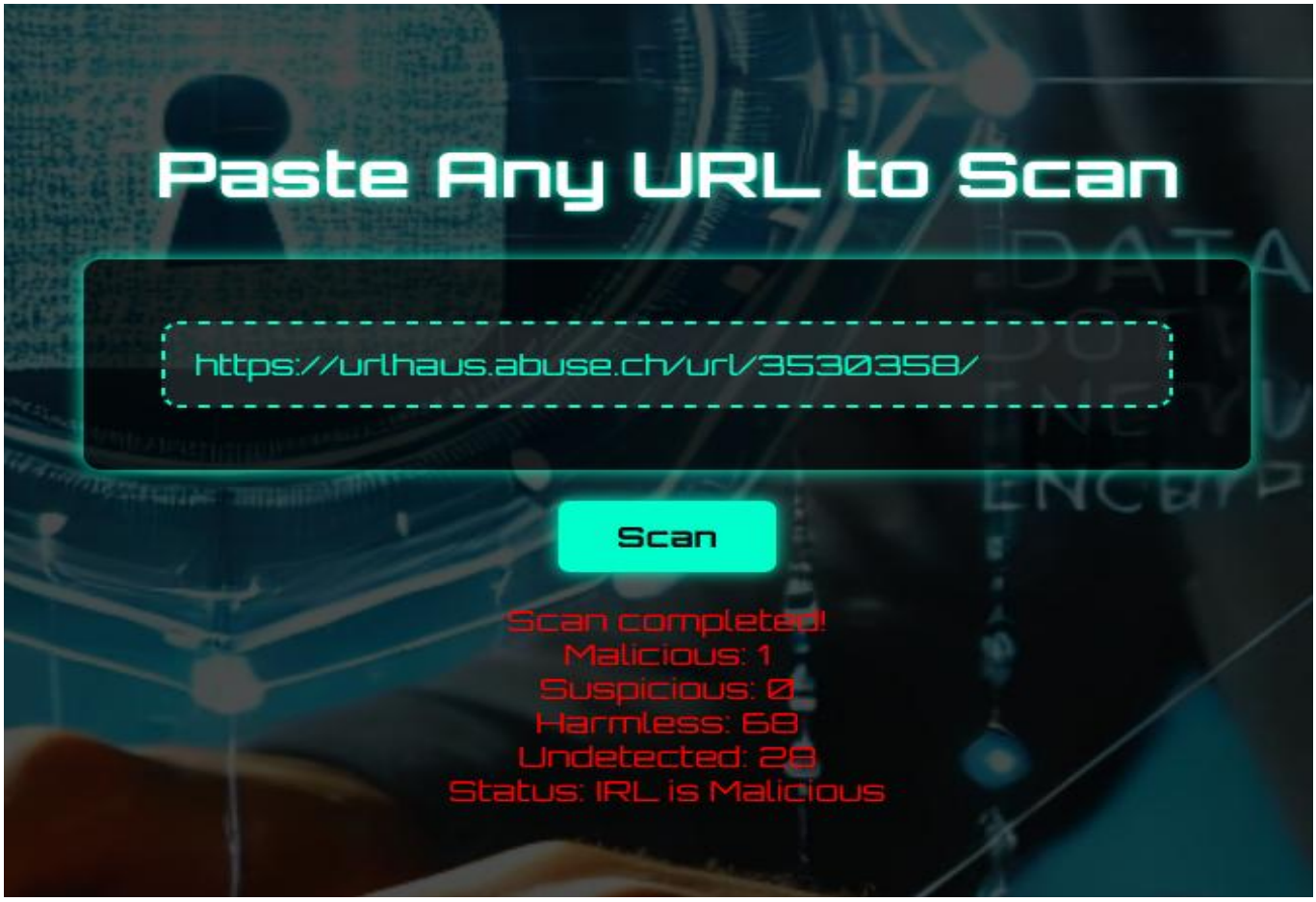


**7.** The Scan result is safe for the google.com and we can also malicious URL For Testing and you can see the result in provided Screenshot

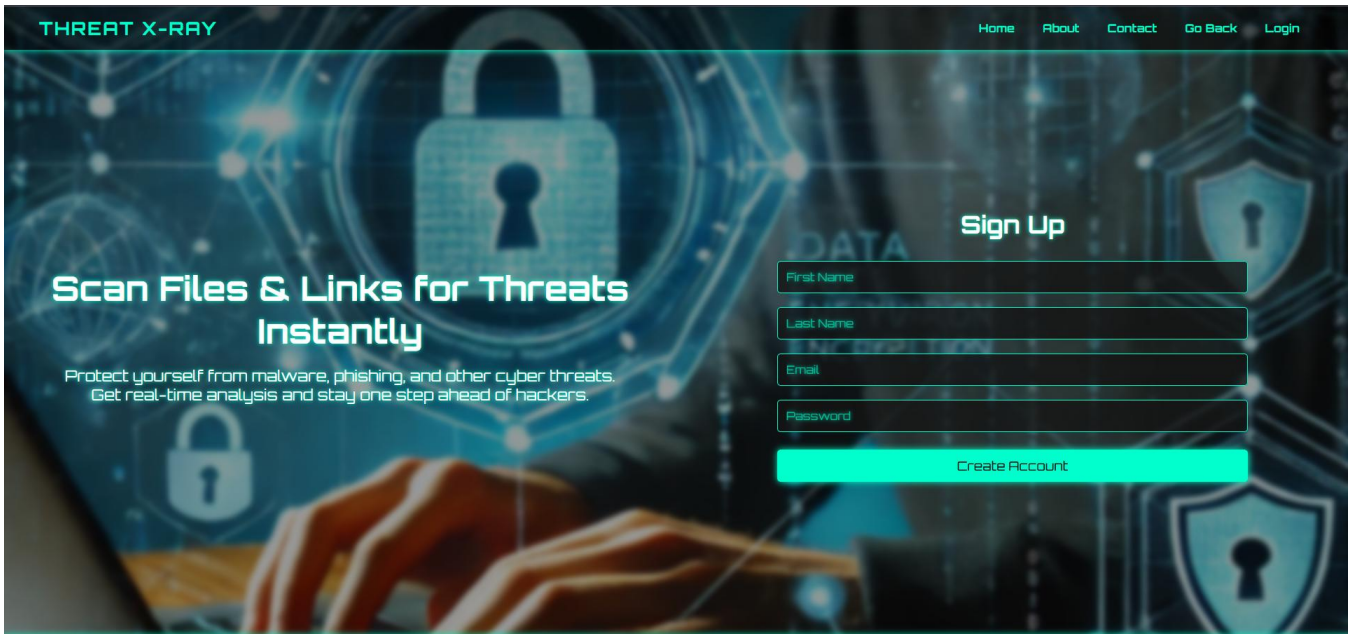




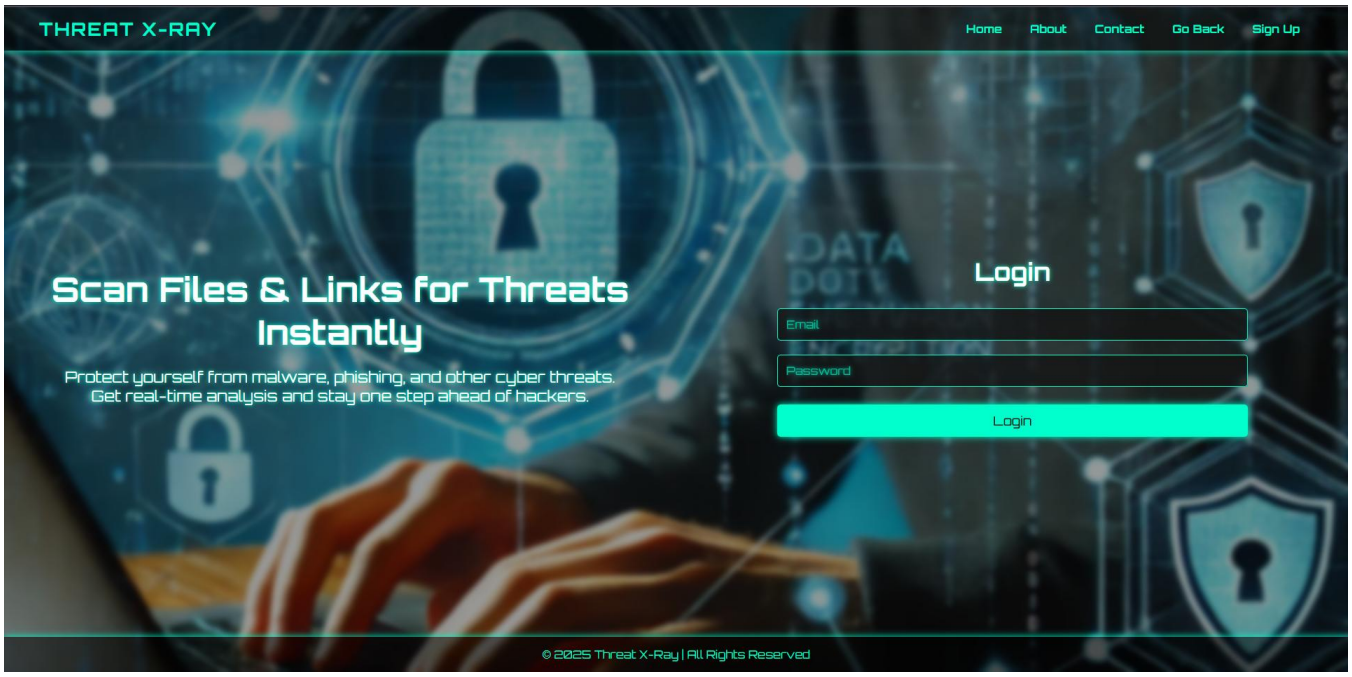
And in the Second screen Shot You can see this URL is malicious



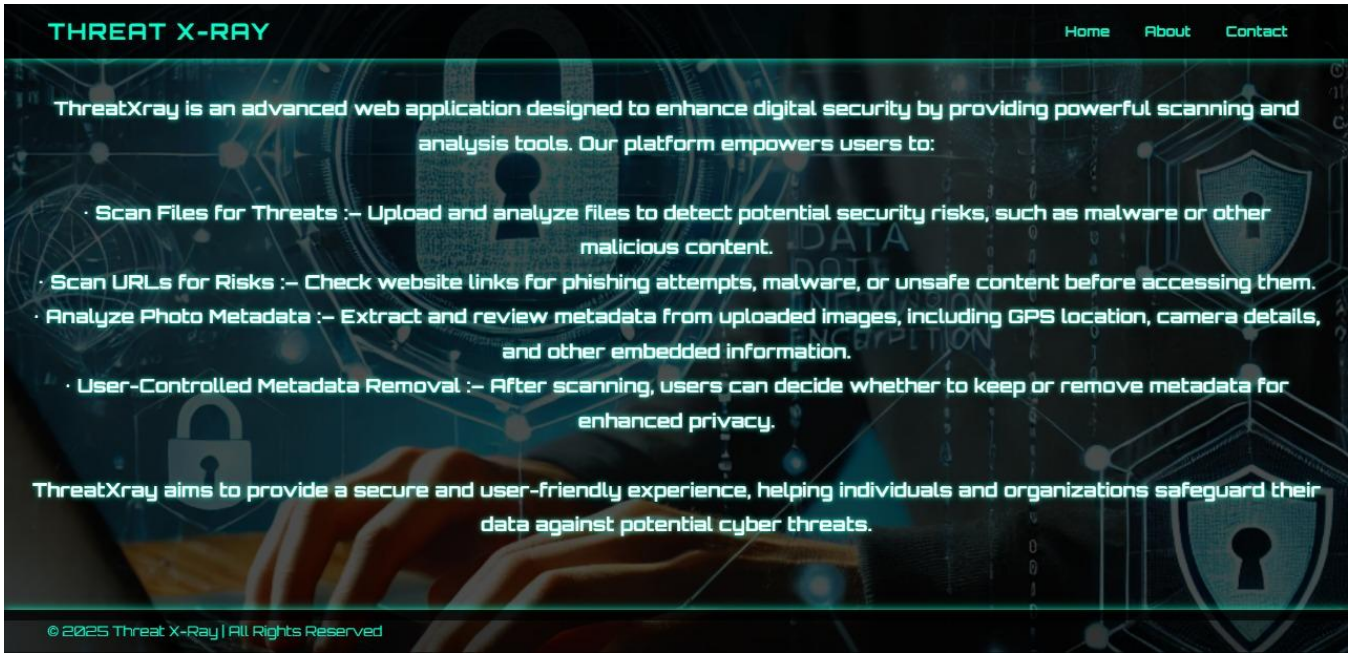

**8.** This is the signup page for our website



**9.**After making your account you can directly login to our website and if you login to our website then you can see your old scan data.




**10.** We created this page to help you understand what our website is about and what we do.



**11.** You can contact us directly if you have any questions or encounter any problems**.**