# SOFTWARE REQUIRED SPECIFICATION

## Executive Summary

This document details the Software Requirements Specification (SRS) for our automated OSINT web/app, which is designed to collect and analyze publicly available domain information and generate a secure, password-protected PDF report. The SRS serves as the comprehensive technical blueprint, meticulously outlining both the functional requirements—such as precise methods for WHOIS queries, DNS lookups, and subdomain enumeration, along with data visualization using libraries like Matplotlib and secure PDF generation—and critical non-functional requirements, including security protocols, performance metrics, and scalability. This specification is indispensable for guiding the development, testing, and deployment phases, ensuring the delivery of a robust, secure, and highly functional application that accurately provides domain intelligence.

### PURPOSE

### OVERVIEW OF THE SRS DOCUMENT

This SRS document outlines the complete software requirements for the OSINT web application. The document is divided into the following parts:

- **Functional Requirements**
- **Non-Functional Requirements**
- **Purpose Of The Application**
- **Scope Of The Application**

### PURPOSE OF THE APPLICATION

This application is a **web-based Open Source Intelligence (OSINT) tool** developed to help users collect and analyze publicly available information about websites and domains. It is particularly useful for cybersecurity analysts, ethical hackers, investigators, and researchers who need quick, structured insights into domain ownership, DNS configurations, and other web infrastructure details. The application provides this data in a user-friendly format, with options for graphical representation and exporting secure PDF reports.

### SCOPE OF THE APPLICATION

The application will support the following functionalities:

- **WHOIS Lookup** – To extract domain registration details (registrar, owner, expiry date).

- **DNS Records Retrieval** – To obtain A, MX, NS, and TXT records of the domain.
- **Subdomain Enumeration** – Using tools like Sublist3r to identify related subdomains.
- **IP Geolocation** – To determine the physical location of servers.
- **HTTP Header Analysis** – To gather metadata from HTTP responses.
- **Email Breach Check** – To find if domain-associated emails have been compromised in known data breaches.
- **Data Visualization** – Generate graphs and pie charts based on the collected data.
- **Export to PDF** – All results can be exported into a password-protected PDF report.

## OUT-OF-SCOPE FEATURES

This version of the application will not support:

• Deep Web or Dark Web scanning.

• Malware detection or sandbox analysis. • Real-time alerts or monitoring.

### Definitions, Acronyms, and Abbreviations

| Term/Acronym | Meaning |
|---|---|
| **OSINT** | Open Source Intelligence – information collected from publicly available sources. |
| **WHOIS** | A query and response protocol used to obtain domain registration and ownership details. |
| **DNS** | Domain Name System – translates domain names into IP addresses. |
| **IP Address API** | Application Programming Interface – used to communicate with external data sources or services. |

**FUNCTIONAL FEATURES**

This tool helps users find public information about websites like who owns them, where they are hosted, what subdomains exist, and whether any related emails were leaked — all neatly presented in charts and downloadable reports.

**WHOIS Lookup :**

WHOIS Inquiry.Feature:
Domain Info look up by WHOIS.
**Input:** Domain name (e.g. somerandomname.com).

**Output**: Registrar, owner, contact, date of creation and date of expiry. **Tool/API**: Python whois module or WHOIS API.

**Workflow :**

1. User inputs a domain (eg, openai.com) into a text field. 2. Pressing Search or Enter.

3. Backend which uses regex [a-z0-9.-.[a-z]{2,6}.

4. Runs WHOIS query.

5. Returns plain text response. 6. Formats and displays:

**STRUCTURES AND PRESENTS:**

**The Registrar**: Name (the registration date is mighty): January 1, 2021

**Expiration date** : January 1, 2025 * Admin thats email and nation, etc.

**Provides the option to Export / Save :**

Managing Errors Invalid/Unregistered Domain → Shows " Domain."

**HaveIBeenPwned Lookup Feature:**

Checker for Email Breach Enter your or any email address (for ex : `anynameoraddress@gmail.com`).

**Results:**

Clean status or breach list Tool/API: HaveIBeenPwned

**API Workflow:**

1. The user inputs their email

2..user regex is verfifed as `^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}$`

3.Email and `hibp-api-key` are included in the headers of the API request ([haveibeenpwned.com]).

4.When successful, shows: Adobe (October 2013): Password, Email Email on LinkedIn (May 2016)

5.Display "No known breaches detected" if no breach was discovered

6. Provides the option to Save / Export Handling Errors Invalid email → Input error displayed Website tech stack report

**Input:**

A URL of a website, such as `https:/openai.com`

**Output:**

CMS, server, frameworks and analytics libraries used

**Tool/API: Whatweb**

CLI via Python's subprocess module.

**Workflow**

1. The user types full URL (must include http or https)

2. Runs subprocess to execute whatweb URL command

3. Gets back tags (e.g., WordPress, Apache, jQuery)

4. Tags are shown as badges in the UI and thus displayed to the user

**IntelligentX –**

Threat Intelligence Aggregation
**Input**: Domain, IP, or hash of the file.
**Output**: Malicious actions and suspicious activity reports from open source intelligence feeds.

**Steps of the process:.**

1. Check input.

2. Query threat info APIs.

3. Parse out intelligence (e.g. breach data, malware flags). 4. Add to MongoDB.

5. Present severity ratings in summary.

**Error Handling**: Error Correction:. Invalid entry -- we ask that you check your input .No threats identified “No threat info found. API down -- “Threat feed unavailable.

**Error Handling**
Network error → Show “Network error” Missing Protocol → Invalid domain message

**Workflow :**

1. The user enters their username

2. Verifies that alphanumeric and underscore characters are permitted.

3. Uses a subprocess to run Sherlock

4. Parses output for platforms that have been located

5. Presents findings as cards or a clickable grid.

**Handling Errors**

Invalid characters → "Only underscores, numbers, and alphabets are permitted." "No social profiles found" → "No profiles found."

**Typical Error and Exception Management**

Type of Error Message Behaviour User Interface Feedback

**Invalid Input:**

"Please enter a valid domain, email address, or username" Input field turns red, also a toast notification is presented

**API Limit Exceeded**:|

"Please try again later. An API we hit the API limit. Button disabled; try again in ten seconds

**Timeout:**

“The request has ended. Check your connection. We have auto retry included. Built-in.

**.No Data Found:**

"We did not find any data for what you entered. Display an example of an empty state

**App Crash:**

"Something went wrong. Try again please. Error recorded;
we have a safe fail out mechanism in place. Non-Functional Considerations Must have valid

headers in your API requests like `User-Agent` and `hibp-api-key` to avoid issues.

## NON FUNCTIONAL REQUIREMENTS

### Purpose

This section outlines the non-functional requirements that govern the operational characteristics of the OSINT Reconnaissance Web Application. These characteristics include performance, security, usability, availability, maintainability, and reporting capabilities. Each of these attributes is critical to ensuring the reliable and secure execution of the application’s

core functionalities. These functions include accepting user-submitted domain-related data—potentially including sensitive or password-protected content—conducting open-source intelligence tasks such as WHOIS lookups and DNS enumeration, visualizing gathered intelligence through interactive graphs, and exporting the results in the form of a secure, password-protected PDF report. The defined requirements aim to uphold system integrity, user trust, and operational resilience.

### Performance Requirements

The application must be capable of processing user-submitted domain queries and generating a complete OSINT report, inclusive of WHOIS records, subdomain enumeration, DNS metadata, and visual graph outputs, within a maximum of fifteen seconds under standard scanning conditions. The user interface must deliver a responsive experience, ensuring that all user-triggered events—such as initiating a scan, downloading a report, or rendering a graph—complete within three to five seconds under average network latency.

To support asynchronous task execution and maintain responsiveness, OSINT scanning operations, including the DNS resolver, WHOIS data collector, and SSL certificate scrapers, must be managed using a distributed task queue architecture, preferably employing tools such

as Celery in conjunction with Redis. The system must accommodate a minimum of fifty simultaneous scanning tasks without degradation in performance, enabling horizontal scalability to support peak operational loads.

**Security Requirements**

All communication between the client and the server must be secured through the implementation of HTTPS, using Transport Layer Security (TLS) version 1.2 or higher. Password-protected domain credentials and user tokens shall be stored exclusively in volatile memory and must be encrypted in-memory using the AES-256 encryption standard. These credentials must never be logged or stored in any persistent format.

All generated reports must be exported as password-protected PDF files using AES-256 encryption. Additional security features such as embedded watermarks or expiry timestamps should be supported to prevent unauthorized reuse. User authentication for report access must be enforced using token-based or one-time-password (OTP) based mechanisms.

Operational logs must exclude any sensitive input or output data and instead retain only essential metadata, such as timestamps of scan initiation and report generation outcomes. The application must implement Role-Based Access Control (RBAC) to manage user privileges effectively. Under this model, administrative users shall have unrestricted access to analytics, system configuration settings, and monitoring dashboards, whereas analyst users shall be limited to executing scans and downloading associated reports.

**Usability Requirements**

The application’s user interface must offer a seamless and intuitive workflow, guiding users from the initial domain input stage through OSINT scanning, and concluding with the option to download a structured PDF report. Visual progress indicators and embedded tooltips should be provided to enhance user guidance throughout the process.

Interactive graphical outputs, including domain ownership maps, subdomain trees, and ASN relationship graphs, must be rendered using scalable vector graphics (SVG) or Canvas-based libraries such as D3.js. These visualizations must be designed for accessibility, incorporating features such as high-contrast color palettes and compatibility with screen readers to accommodate users with disabilities.

For modules that involve complex data manipulation, such as WHOIS filtering, metadata selection, or graph customization, in-context help, documentation modals, and tooltips must be readily available. The system must also provide real-time feedback messages to users, communicating statuses such as data retrieval in progress, PDF generation status, or error notifications in cases of third-party service outages.

## Availability Requirements

The system must maintain an operational uptime of at least 99 percent, excluding scheduled maintenance periods. Each functional module of the application must be containerized and configured to support automatic restarts in the event of a failure, utilizing orchestration platforms such as Docker Swarm or Kubernetes.

In scenarios where a third-party dependency such as a WHOIS API becomes unavailable, the application must implement a retry mechanism that attempts failed requests up to three times using exponential backoff. If all retries fail, the system must inform the user through a clear and informative fallback notification, thereby preserving the user experience. To ensure continued reliability, the overall system architecture must be capable of horizontal scaling, with integrated load balancing mechanisms that dynamically distribute requests across available services.

## Report Generation and Export Requirements

Reports generated by the OSINT Web Application must be structured and comprehensive, including sections that detail raw WHOIS metadata, subdomain enumeration results, DNS records, and graph-based visualizations that reflect ownership relationships and domain infrastructure.

Each report must include timestamped insights and references to data sources used during the scan. These reports must be exportable as password-protected PDF documents. Optional export enhancements such as watermarks indicating report expiry and secure download links that expire within ten minutes must be supported.

The PDF rendering engine used—whether WeasyPrint, wkhtmltopdf, or any equivalent—must be capable of producing consistent output that is fully compatible with all modern web browsers and devices.

## Monitoring and Logging Requirements

The system must maintain an internal logging framework capable of capturing critical application-level events, such as the initiation of new scans, completion of exports, and system errors. These logs must strictly omit all sensitive content, including user-submitted inputs, access tokens, or PDF report data.

An administrative dashboard must be provided to allow monitoring of key operational metrics such as average scan duration, report generation success and failure rates, and user activity related to downloads. Automated alerting mechanisms must be configured to flag anomalies, including persistent task queue delays, error rates exceeding predefined thresholds (e.g., five percent failure rate over ten minutes), and unusual response times, such as scans that exceed thirty seconds in duration.