**System Requirements Specification**

**for**

AutoPen

**Version 1.0 approved**

**Prepared by Michael Allen**

**AutoPen**

**09/29/2023**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 4

3.3 Software Interfaces 4

3.4 Communications Interfaces 4

**4. System Features 5**

4.1 System Feature 1 5

4.2 System Feature 2 5

**5. Other Nonfunctional Requirements 6**

5.1 Performance Requirements 6

5.2 Safety Requirements 6

5.3 Security Requirements 7

5.4 Software Quality Attributes 7

5.5 Business Rules 8

**6. Other Requirements 8**

**Appendix A: Glossary 9**

**Appendix B: Analysis Models 9**

**Appendix C: To Be Determined List 9**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Michael Allen | 9/21/23 | Section 1 and 2 | 1.0 |
| Michael Allen | 9/26/23 | Section 4 and 5 | 1.1 |
| Michael Allen | 9/28/23 | Section 3 and 6 | 1.2 |
| Caleb Hall | 9/29/23 | Editing and formatting | 1.3 |

# **Introduction**

## **Purpose**

The purpose of this product is to use machine learning to allow users to apply penetration testing to their website in order to locate possible vulnerabilities in design. This SRS covers both the machine learning application to allow for automated penetration testing, and the website that will be used to access said application.

## **Document Conventions**

1. Document section titles will be in Times New Roman size 14 font. Document non-title writing will be in Arial size 11 font.
2. The requirements have their own authority

## **Intended Audience and Reading Suggestions**

This documentation on AutoPen is designed specifically for web developers and web testers as users of this system.

The rest of the document is organized as follows. Section 1 provides an initial overview of the document, explaining its purpose, conventions, and scope. Section 2 delves into a comprehensive understanding of the product, its functions, characteristics, and constraints. Section 3 details the specifications for how the system interacts with its users, hardware, other software, and communications networks. Section 4 describes individual features or functionalities of the system in depth. Section 5 enumerates the system's performance, safety, security, quality, and business rules ensuring it meets established standards and stakeholder expectations. Section 6 contains additional information in appendixes.

## **Product Scope**

With the need for stronger, faster security with the increase of cybersecurity threats, AutoPen aims to revolutionize the field of penetration testing by leveraging artificial intelligence to conduct automated, comprehensive, and adaptive penetration tests. The primary objective is to provide businesses and organizations with a quick, cost-effective and thorough way to identify and rectify potential vulnerabilities in their networks and systems.

## **References**

The URL of the web application is autopentest.net

# **Overall Description**

## **Product Perspective**

The integration of automated penetration testing represents a new paradigm shift in the field of cybersecurity. By replacing manual systems and innovating with advanced technologies, organizations can enhance their security posture, reduce vulnerabilities, and stay ahead of potential cyber threats. The increased efficiency, accuracy, and scalability offered by automated penetration testing make it an indispensable tool in today's digital landscape. Embracing automation in penetration testing is not only a step towards better security but also a strategic investment in the long-term success and resilience of an organization's IT infrastructure.

## **Product Functions**

1. The product must allow users to test a website for vulnerabilities
2. The product must be a functional website, allowing any user to visit using the internet

## **User Classes and Characteristics**

The users fall into two categories: end user and internal developer. End users are the web app developers who are testing their own site. Internal developers are AutoPen employees. End users only have read/write permissions on their own profiles, while internal developers have admin access across the site. Both classes are equally important to satisfy.

## **Operating Environment**

To effectively utilize the required system, the following operating environment is necessary:

1. Computer
2. Internet Connection
3. Website/Web Browser
   1. Independent of user’s computer operating system

## **Design and Implementation Constraints**

The design and implementation of Auto Pen must consider the potential legal and ethical constraints associated with the device. By addressing these concerns, such as intellectual property rights, privacy, consumer protection, plagiarism, and human interaction, Auto Pen can be developed and used responsibly. It is crucial to prioritize legal compliance and ethical considerations to ensure the successful integration and acceptance of Auto Pen in various industries and domains.

## **User Documentation**

An instruction panel and video tutorial will be implemented into our website. The AutoPen website aims to enhance user experience, provide clear guidance, and support users throughout their penetration testing journey. This addition will ensure that users can effectively utilize the platform's capabilities and maximize the benefits of automated penetration testing.

## **Assumptions and Dependencies**

The development of machine learning algorithms for penetration testing is a complex and time-consuming process. As a team, we need to be aware of the main constraint that we may face, which is the limitation of time. Time is a critical factor when it comes to developing sophisticated algorithms for penetration testing. The process involves collecting and analyzing large amounts of data, training the machine learning models, and validating their effectiveness. Each of these steps requires significant time and effort to ensure the accuracy and reliability of the algorithms.

Another constraint is tooling, as many professional tools useful for such a project are often locked behind large paywalls or corporate secrecy. So for this document and for the included requirements, it is assumed we are able to obtain the necessary tooling. This also demonstrates a dependency on external tooling for algorithm training and penetration testing analysis.

Software dependencies include Python for the machine learning algorithm. The project depends on Sitegrounds web hosting service for application hosting and user interaction.

The AI algorithm is dependent on Python’s AI and Cybersecurity libraries.

This project assumes the legality of such a tool.

This project assumes the hosting platform has the power to host the backend algorithm tasks.

# **External Interface Requirements**

## **User Interfaces**

**Sample Screens:** The main interface will have input fields for target specifications, a "Start Test" button, progress bar, and a results display area.

**GUI Standards:** The interface will follow modern web application design principles, ensuring ease of use and intuitive navigation.

**Layout Constraints:** The site should be responsive to fit various screen sizes, including mobile devices.

**Standard Buttons:** Every screen will have "Help," "Home," and "Settings" buttons.

Keyboard Shortcuts: "CTRL + H" for help, "CTRL + S" to start a test.

**Error Messages:** Errors will appear as red pop-up banners at the top of the screen with concise explanations.

**Components with UI:** Main dashboard, settings, results page, and user profile.

## **Hardware Interfaces**

**Supported Devices:** Users can access the website using PCs, laptops, tablets, and mobile phones.

**Data & Control Interactions:** As the application is hosted on Siteground's web hosting, interaction is primarily through the website's servers with 20 GB web space.

**Communication Protocols:** Secure HTTP (HTTPS) for web communications and WebSockets for real-time updates.

## **Software Interfaces**

**Connected Software:**

Database: Hosted on Sitegrounds, specific database details to be provided.

Operating Systems: Sitegrounds' web hosting environment.

Backend Language & Libraries: Python 3.11 with its standard and third-party libraries.

**Data In/Out:**

Incoming: User credentials, target specifications, user configurations.

Outgoing: Test results, progress updates, error messages.

**Services & Communications:** RESTful API for communication between frontend and backend. API documentation available in a separate document.

**Shared Data Mechanism:** Given the constraints of web space, optimizing data storage and retrieval will be crucial. Database caching strategies will be implemented for efficiency.

## **Communications Interfaces**

**Functions:** Email notifications, web browser alerts, server-to-server communications.

**Message Formatting:** JSON for data interchange.

**Communication Standards:** Secure FTP (SFTP) for file transfers, HTTPS for secure web communications, specific to Sitegrounds' configurations.

**Security/Encryption:** All communications will be encrypted using TLS 1.3 or as supported by Sitegrounds.

**Data Transfer Rates:** Optimized for broadband connections, minimum recommended speed of 5Mbps.

**Synchronization:** WebSocket for real-time data synchronization between client and server, depending on Sitegrounds' support.

# **System Features**

## System Feature: Automated Penetration Testing

**4.1.1 Description and Priority**

Benefits: 9 (Significant user value)

Penalty: 4 (Users may opt for competitors if not implemented well)

Cost: 7 (Complex to implement, but feasible)

Risk: 8 (Potential for misuse or false results)

**4.1.2 Stimulus/Response Sequences**

1. User inputs target specifications.
2. System validates input and prompts confirmation.
3. User initiates the test.
4. System displays progress and provides real-time updates.
5. System presents results upon completion.

**4.1.3 Functional Requirements**

REQ-1: System shall allow users to input target specifications.

REQ-2: System shall validate the target specifications for correctness and safety.

REQ-3: System should provide real-time feedback on the penetration testing progress.

REQ-4: System shall display a summary of vulnerabilities detected and potential fixes.

REQ-5: System must handle potential errors, such as unreachable targets, gracefully by notifying the user.

REQ-6: Ensure user authentication before initiating any penetration tests.

## **System Feature**: User Profile Management

**4.1.1 Description and Priority**

Allows users to create, edit, and manage their profiles. Priority: Medium

Benefits: 7 (Enhances user experience)

Penalty: 5 (Users might be dissatisfied with static profile settings)

Cost: 3 (Standard feature in web applications)

Risk: 7 (Data privacy concerns)

**4.1.2 Stimulus/Response Sequences**

1. Users will log in or sign up.
2. System presents a dashboard or user profile page.
3. User edits profile information.
4. System validates and saves changes, then confirms with the user.

**4.1.3 Functional Requirements**

REQ-1: System shall provide user registration and login functionality.

REQ-2: System should allow users to edit their profile information, including name, email, and contact details.

REQ-3: System must securely store and encrypt user passwords.

REQ-4: System must allow password recovery through a secure method, such as email verification.

# **Other Nonfunctional Requirements**

## **Performance Requirements**

1. The system shall respond to user input, such as starting a penetration test, within 2 seconds under typical load conditions.
2. During the execution of a penetration test, progress updates should be real-time, with no lag exceeding 3 seconds.
3. The system should support simultaneous penetration tests from 100 users without performance degradation.
4. The website should load within 3 seconds under typical network conditions.

Rationale: Quick response and real-time updates are essential for user satisfaction and to maintain trust in the system's effectiveness.

## **Safety Requirements**

1. The system must strictly adhere to the target specified by the user. Under no circumstances should the system perform penetration tests on websites or domains other than the one specified.
2. The system shall incorporate rate limiting to prevent accidental or intentional flooding of a target, ensuring it adheres to ethical testing norms.
3. The system should provide clear disclaimers and guidance to users on ethical use, ensuring they have the necessary permissions to test the target website.

Reference: Adherence to international cybersecurity guidelines and standards, such as the OWASP testing guide, is mandatory to ensure the product's safety.

## **Security Requirements**

1. All user data, especially login credentials, must be encrypted using industry-standard protocols like bcrypt (cryptographic hash function) for password hashing.
2. User sessions should be protected against session hijacking or cookie theft.
3. The system shall implement multi-factor authentication for enhanced user account security.
4. Any machine learning models or algorithms associated with the system should be hosted on a separate server, isolated from the primary website. Direct access to these models by external entities must be strictly prohibited.
5. Regular security audits should be conducted to identify and mitigate potential vulnerabilities.
6. All data communication, especially involving user information or test results, should be encrypted using protocols like TLS.

Reference: Adherence to the General Data Protection Regulation (GDPR) and other regional data protection laws is vital to ensure user data privacy and security. The system should also aim for security certifications like ISO/IEC 27001 to establish trustworthiness.

## **Software Quality Attributes**

1. Adaptability: The software should be able to accommodate changes in penetration testing methodologies or emerging threats with minimal modifications.
2. Availability: Aiming for 99.9% uptime, the system should always be accessible for users when they require penetration testing.
3. Correctness: The penetration testing results should have an accuracy rate of at least 95%, ensuring reliable insights for users.
4. Flexibility: It should be feasible to integrate new tools or functionalities without major architectural changes.
5. Interoperability: The system should be able to seamlessly interact with common third-party platforms or services, if needed, such as vulnerability databases.
6. Maintainability: Developers should be able to fix bugs, make enhancements, or address issues with clear documentation in place and modular code architecture.
7. Portability: While the core application is web-based, any auxiliary tools or scripts should work across different OS platforms.
8. Reliability: During any penetration test, the system should not crash and should handle errors gracefully, providing useful feedback to the user.
9. Reusability: Components of the system, especially the AI models, should be designed in a way that they can be used in different contexts or projects.
10. Robustness: The system should handle unexpected inputs or situations without failing, especially considering varied targets for penetration testing.
11. Testability: The system should have provisions to be tested easily, both for individual units and end-to-end functionality.
12. Usability: The user interface should be intuitive with a preference towards ease of use, even if it comes with a slight learning curve.

Preference: Emphasis on usability and correctness for the primary user experience, ensuring they trust the system and can navigate it effortlessly.

## **Business Rules**

1. Machine Learning Application Access: Only the creators or designated supervisors of the penetration testing website/product have the privilege to access, modify, or manage the underlying machine learning application.
2. Report Access: Only registered and authorized users can view penetration test reports. Unauthorized access attempts should be logged and reported.
3. Role-Based Access Control: Depending on the user role (admin, user, guest, etc.), different levels of system functionalities and data should be accessible. For instance, only admins or creators can modify system configurations, while regular users can only initiate tests and view their own reports.

# **Other Requirements**

Reuse Objectives

1. Modular Design: Components of the system, especially the machine learning models and data processing units, should be modularly designed for potential reuse in other related projects.
2. API Development: Develop RESTful APIs that can be used internally and potentially opened up for third-party integrations or other projects.

Database Requirements

1. Scalability: The database should be scalable to handle an increasing number of user records and penetration test results.
2. Backup: Regular backups, both incremental and full, should be taken to prevent any data loss.
3. Access Control: Strict role-based access to ensure that only authorized personnel can view or modify the database.

Internationalization Requirements

1. Localization: The system should support multiple languages, starting with English, Spanish, and French.
2. Currency: If there's a payment system in place, it should be able to handle and convert multiple currencies.
3. Time Zones: User profiles and scheduling features should consider user time zones for accurate timing and notifications.

Legal Requirements

1. Data Protection: Adhere to international data protection regulations like GDPR (General Data Protection Regulation) for European users and CCPA (California Consumer Privacy Act) for California residents.
2. Ethical Use: The tool's capabilities must not be misused, and users must have necessary permissions to conduct penetration tests on the target.
3. Licensing: Any third-party tools, libraries, or services used should be properly licensed, and their terms and conditions should be followed.

Accessibility Requirements

1. Web Standards: The website should be designed following W3C Web Content Accessibility Guidelines (WCAG) to ensure that it's usable by people with disabilities.
2. Contrast and Font: Adequate contrast ratios and readable fonts should be used to cater to users with vision impairments.
3. Keyboard Navigation: The site should be navigable using just the keyboard, aiding users with mobility impairments.

**Appendix A: Glossary**

1. AI: Artificial Intelligence. It refers to the capability of a machine to imitate intelligent human behavior.
2. Penetration Testing (Pentest): The practice of testing a computer system, network, or web application to find vulnerabilities that attackers could exploit.
3. SRS: Software Requirements Specification. A document that describes the features, behaviors, and attributes of a software system.
4. TLS: Transport Layer Security. A protocol ensuring privacy and data security between two communicating applications.
5. RESTful API: Representational State Transfer. A set of rules that developers follow when they create their API, allowing for interaction between systems using HTTP.

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**

1. Database Selection: Specific database technology to be used hasn't been finalized.
2. Third-Party Integration: Deciding on whether to integrate third-party vulnerability databases directly.
3. Pricing Model: How users will be charged for the service is TBD.
4. Notification System: How and when users receive notifications about test results or system updates.
5. User Data Retention Policy: Duration and conditions under which user data and reports will be stored.

(These are placeholders and might change based on the actual pending decisions in a real-world project.)