AutoPen System Design Document Version 5.1

Calla Robison

Caleb Hall

Michael Allen

Myles Scott

Joshua Buscher

Version 1.0 / Joshua + Michael	10/24/2023
Version 2.1 / Joshua + Myles	10/31/2023
Version 2.2 / Caleb + Calla + Michael	10/31/2023
Version 3.1 /Michael + Caleb + Joshua + Calla + Myles	11/21/2023
Version 4.0 / Joshua + Myles	2/2/2024
Version 4.1 / Michael	2/5/2024
Version 5.0 / Joshua	2/13/2024
Version 5.1 / Joshua	3/2/2024
Version 5.1 / Calla + Myles + Michael + Joshua + Caleb	3/3/2024

TABLE OF CONTENT

SYSTEM DESIGN DOCUMENT

1 INTRODUCTION

Cybersecurity threats are rapidly evolving, requiring more agile and thorough security systems. AutoPen aims to revolutionize the field of penetration testing by leveraging Burp Suite to conduct automated, comprehensive penetration tests. The primary objective is to provide businesses and organizations with a quicker, cost-effective, and more thorough method of identifying and rectifying potential vulnerabilities in their networks and systems.

AutoPen is designed to:
- Be used by small and medium enterprises (SMEs).
- Cover common vulnerability areas, including OWASP Top 10.
- Option to run tests periodically or on demand.
- Compliance with standard cybersecurity frameworks and regulations.

1.1 Purpose and Scope

Section 1 provides an introduction to the system. The following sections contain details regarding the system architecture, human-machine interfaces, detailed design, and external interfaces.

1.2 Project Executive Summary

This section provides a managerial overview of AutoPen, specifically the system design.

1.2.1 System Overview

With the increasing complexity and sophistication of cyber threats, organizations are in need of robust security measures to protect their assets and sensitive information. Traditional manual penetration testing methods can be time-consuming, costly, and prone to human error. AutoPen aims to streamline and optimize the penetration testing process, providing organizations with accurate and efficient assessments of their security posture.

The automated penetration test app leverages advanced technologies such as Burp Suite to simulate real-world cyber-attacks. By automating the testing process, our application can rapidly scan and analyze large volumes of data, identifying vulnerabilities, misconfigurations, and potential entry points that could be exploited by malicious actors.

The web application will be hosted on the cloud, allowing users to access the system. The web app will contain multiple pages, each with a different function such as configuring tests, launching tests, and viewing test results.

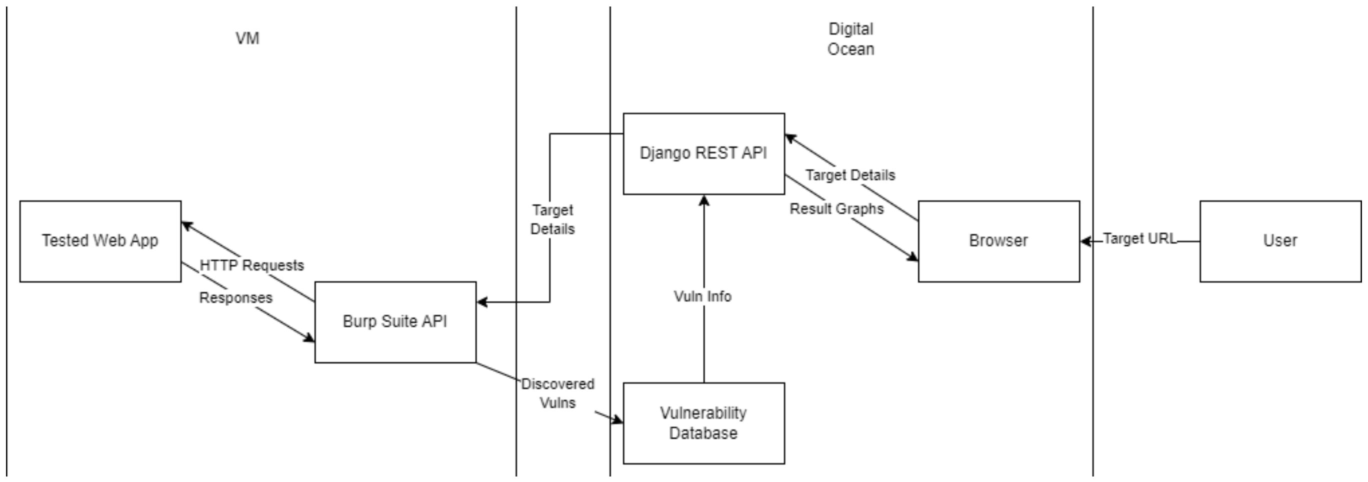


Figure 1: System Diagram

Figure 1 (shown above) is a basic system overview for Autopen. The VM is responsible for hosting the tested application and the Burp Suite API that performs the testing. DigitalOcean is responsible for connectinging the user to the AutoPen website and consists of the user's browser, the Django REST API that prints the pentest results, and the vulnerability database that stores the found vulnerabilities for the REST API to print out.

Area of Penetration Testing: Web application pen-testing

Primary programming languages:

- Client-side programming: Presentation layers and user interaction. Client-side code is responsible for rendering web pages, handling user input, and making the user experience dynamic and responsive.
  - Javascript: React - It serves as the frontend that handles the UI and getting and setting data via requests to the Django backend. Works in conjunction with HTML and CSS for a dynamic web application
- Server-side programming: Handles server logic, data processing, and database interactions. Server-side code is responsible for generating dynamic content, handling user authentication, and managing data.
  - Python: Django framework - It handles database migration for user account information, it renders the Burp Suite API data into a json file for front-end (React) usage, and it provides built-in user authentication with an admin dashboard for debugging
  - Django REST Framework - API built to allow communication between the react front-end and Django backend.

1.2.2 Design Constraints

In order to ensure the effectiveness and reliability of AutoPen, there are several design constraints that need to be considered. The primary focus lies in developing a web application that is capable of using Burp Suite to identify vulnerabilities while simultaneously ensuring its own security and preventing unauthorized access.

Management:

- Time Limitation: The project must be fully designed, developed, and tested within a single academic year. This period includes all stages of development, from initial planning to final presentation.
- Skill Constraints: The team consists of students with intermediate-level knowledge in software development and cybersecurity. Therefore, the project will avoid the development of custom penetration testing tools or highly complex algorithms that are beyond the team's current skill level.
- Financial Constraints: The project's budget is strictly limited to resources already provided by the academic institution or are accessible for a low fee. This includes software tools, development environments, and hosting platforms.

Technical:

- Burp Suite: The project will use the professional version of Burp Suite for web vulnerability scanning. Therefore the project is limited in abilities to the offerings of this product.
- DigitalOcean Hosting: The project will be deployed on a basic tier of DigitalOcean Hosting, which provides essential services within the project's budget. The project will not utilize higher-tier services that incur additional costs, limiting the compute power of the assets.

Legal:

- United States' Computer Fraud and Abuse act: The project must follow the United States' Computer Fraud and Abuse act that makes unauthorized access to protected computers a crime.
- Electronic Communications Privacy Act: The project must follow the Electronic Communications Privacy Act that regulates the disclosure and interception of communications related to electronics.
- EU General Data Protection Regulation: The project must follow the EU's General Data Protection Regulations that gives strict requirements for the protection and processing of personal data.
- General liability: The product must be liable for any damages caused by the penetration test.
- Documentation: The product must create detailed documentation that lists the scope of the pentest, the methods and its results, the latter being needed to ensure proper procedures were met.

Assumptions:

- Legality: The team assumes that this project is legal to be made and used in its current scope.
- Hosting platform resources: The team assumes that the hosting platform dedicating enough resources to ensure efficient operation of our programs

There are two main assumptions, that being the legality of this tool and the hosting platform dedicating enough resources to ensure efficient operation of our programs.

1.2.3 Future Contingencies

The design and implementation of "AutoPen" comes with potential contingencies that may shift its developmental trajectory. Anticipating these challenges and devising alternate strategies ensures a smoother progression of the project. The formatting of this section is the name of the issue that needs a contingency, followed by bullet points describing the scenario of the issue, and the workaround to solve it.

Unintended Style Disruption:

- Scenario: A user mistakenly selects a penetration test style or configuration, causing disruption or breakdown of the target site.
- Possible Workaround: Implement a pre-test configuration validation mechanism. Restrict user choices to predefined, vetted penetration test configurations. Offer clear guidelines and warnings for configurations known to be aggressive.

Accidental Targeting of External Websites:

- Scenario: Due to misinterpretation of user input, "AutoPen" mistakenly tests a website other than the intended target, leading to possible ethical and legal concerns.
- Possible Workaround: Introduce a two-step validation procedure. Before starting a test, users are required to reenter the web address of the site they aim to test. "AutoPen" verifies this against the original target URL to ensure alignment.

Web Hosting Limitations:

- Scenario: The current web host, for reasons such as inadequate hosting capabilities, legal issues, or strategic decisions, becomes unsuitable for hosting "AutoPen".
- Possible Workaround: Investigate alternative, more robust web hosting providers. Begin initial groundwork for transitioning to another provider to ensure minimal disruption. Concurrently, explore the feasibility of using a privately maintained physical server as an alternative or backup. Regularly evaluate hosting needs versus the host's capabilities to preemptively address limitations.

Transition to a Different Hosting Provider:

- Scenario: A strategic decision is made to migrate "AutoPen" to another hosting provider for enhanced performance, cost-efficiency, or other benefits.
- Possible Workaround: Ensure that the new hosting provider meets all the requirements of "AutoPen". Begin the transition process during off-peak hours to minimize disruption. Implement a robust backup and migration strategy to prevent data loss.

Consideration of a Private Physical Server:

- Scenario: Due to growing needs or for enhanced control, there's a proposition to move "AutoPen" onto a private server managed by the development team.
- Possible Workaround: Conduct a thorough feasibility study weighing the pros and cons of maintaining a private server. Address challenges such as maintenance costs, security considerations, and scalability. If pursued, establish a transition roadmap and backup strategies.

1.3 Document Organization

This design document offers an in-depth look into the architecture and blueprint of

"AutoPen". Beginning with the product's functionalities, it delves into its limitations, interactions, interfaces, hardware and software designs, and wraps up with security measures.

1.4 Project References

The following are external resources used by the AutoPen project:

1.4.1 Documentation for Burp Suite, the api used to run penetration tests, can be found at: https://portswigger.net/burp

1.4.2 AutoPen Github can be found at: https://github.com/Caleb-Hall-1015/AutoPen

1.5 Glossary

Penetration Testing (Pentest): The practice of simulating an attacker and testing a computer system, network, or web application to find vulnerabilities.

SRS: Software Requirements Specification. A document that describes the features, behaviors, and attributes of a software system.

TLS: Transport Layer Security. A protocol ensuring privacy and data security between two communicating applications.

RESTful API: Representational State Transfer. A set of rules that developers follow when they create their API, allowing for interaction between systems using HTTP.

2 SYSTEM ARCHITECTURE

Section 2 concerns the general architecture of AutoPen. The formatting of Section 2 is as follows: Section 2.1 is about the hardware architecture of the system, Section 2.2 covers the software architecture of the system, and Section 2.3 details the internal communications architecture of the system.

2.1 System Hardware Architecture

AutoPen is designed to operate entirely within the cloud, with all assets securely stored and managed in cloud-based servers. Unlike traditional hardware architectures, AutoPen does not require any physical infrastructure to function effectively. This cloud-based approach offers several advantages, including increased flexibility, scalability, and accessibility.

2.2 System Software Architecture

The Burp Suite API-powered Penetration Testing system is designed with a three-tier architecture: Presentation Layer (Web UI), Business Logic Layer (Backend Server), and API (Burp Suite vulnerability analyzation tool). As for how the user interacts with the software, The user interacts with the Presentation Layer, where they can register, log in, initiate tests, and view results. This communicates with the Business Logic Layer, where the main functionalities like Burp Suite invocations, test configurations, and report generations take place. The results, user data, and test configurations are stored and fetched from the Data Layer which is the database.

2.2.1 Software Modules

The software modules for the product includes the User Management Module, the testing Initialization module, the reporting module and the notification module. The User Management module is in charge of handling user registration, login, and profile management. The test initialization module handles initiating and configuring penetration tests. The reporting module handles the generation and displaying of test results to the user. Notification module helps to alert users about system updates and test completions.

2.2.2 Coding languages

The coding languages can be split into two sections, front end and back end languages. Front end languages refer to languages that allow areas of the system that are in the immediate view of users, like the website, to function. The front end language for the product includes HTML, CSS and JavaScript, specifically React.js. Back end languages refer to languages that function behind the scenes to perform tasks outside the view of users, like cloud services. The back end languages for the product are Python 3.11 and Python Django.

2.2.3 Tools

Some of the tools for that are used for the product are Git for version control, Docker for containerizing application components for consistent deployment and testing, DigitalOcean Iaas for integration of the pentesting API Burp Suite into a web application, DigitalOcean droplets which is used in conjunction with DigitalOcean Iaas for Cloud storage, and Github Actions which help with the integration and deployment of the product. Other tools include Kali Linix which is a version of linux with pre built tools for penetration testing, VMWare which is a virtualization software for hosting VMs, Metasploit, a current unused penetration testing api, and Burp Suite, the penetration testing API that is currently being used instead of Metasploit.

2.3 Internal Communications Architecture

Our system primarily relies on a web-based communication system, utilizing a combination of HTTP/HTTPS requests for standard interactions and WebSockets for real-time updates.In addition, when a user accesses the web application, their browser communicates with the backend server using HTTP/HTTPS requests. For tasks like initiating a penetration test, where real-time feedback is essential, a WebSocket connection is established. This ensures users receive immediate updates about the test's progress. The backend, in turn, communicates with the database to store or fetch data as required. The formatting of this section is a list of all communication architectures and a diagram reference describing a diagram of said communication architectures.

Communication Architectures:

- HTTP/HTTPS: Used for standard request-response interactions between the frontend and backend.
- WebSocket: A protocol providing full-duplex communication channels over a single TCP connection, facilitating real-time updates.
- Diagram Reference: Figure 2, shown below, shows how Django Data Models interacts with the Database that stores queries, Serializers that Create Json data objects, and REACT that stores static files. REACT and Serializers also interact with Django REST API CRUD, which obtains data from databases.

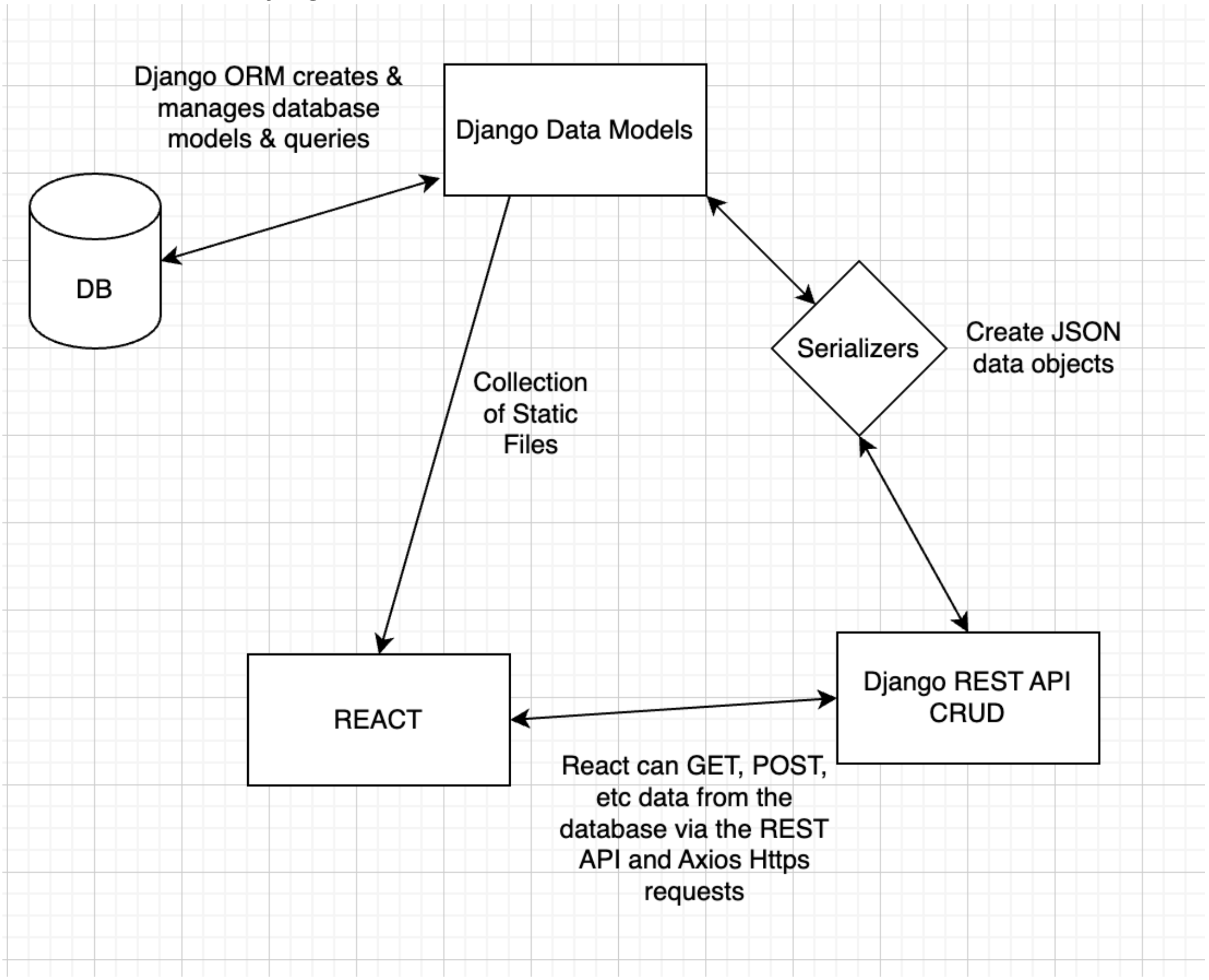

Django ORM creates & manages database models & queries
Django Data Models
DB
Serializers
Create JSON data objects
Collection of Static Files
REACT
Django REST API CRUD
React can GET, POST, etc data from the database via the REST API and Axios Https requests

Figure 2: Internal Communication Flow

3 HUMAN-MACHINE INTERFACE

Section 3 is about the inputs users give to the product and the outputs the users receive from the product. The formatting of Section 3 is as follows: Section 3.1 describes the inputs of the product, Section 3.2 describes the outputs of the product.

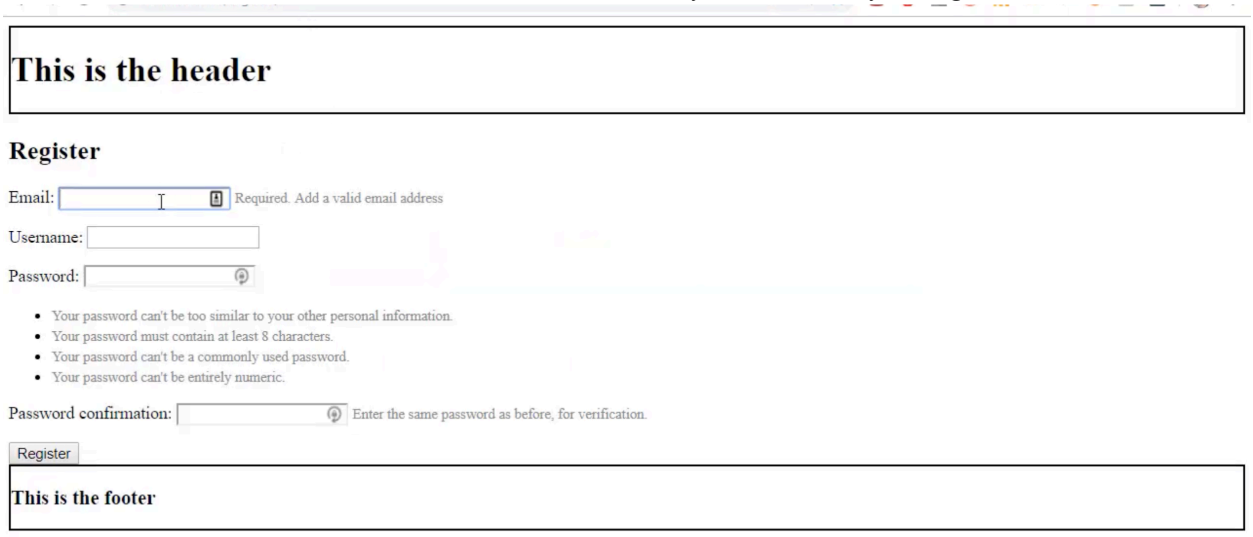
3.1 Inputs

The main input methods are through data entry screens on the web application. These input mechanisms directly map to the high-level data flows described in the System Overview section.

3.1.1 Data Entry Screens:

3.1.1.1 User Registration Screen:

- Data Elements: Username, Email, Password, Confirm Password, optional user profile details like contact number
- Mandatory fields: Username, Email, Password.
- Constraints:
  - Passwords should match the Confirm Password field and have a minimum of 8 characters.
  - Emails must adhere to standard email formats.
- Controls: Users are barred from proceeding without completing mandatory fields. Passwords undergo strength validation.
- Access Restrictions: Exclusively accessible by unregistered users.



This is the header

Register

Email: Required. Add a valid email address

Username:

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation: Enter the same password as before, for verification.

This is the footer

Figure 3 : User Registration Page

- Figure 3 depicts a screenshot of an example user interface for a registration page. The interface includes a header at the top and a footer at the bottom, framing the registration form. The form fields prompt the user for an email address, with a note indicating that a valid email address is required, a username, and a password, along with a password confirmation field. Accompanying the password field are instructions specifying the password criteria, such as a minimum length and restrictions on similarity to other personal information. The layout is simple and user-centric, designed to facilitate an easy and secure registration process.

3.1.1.2 Login Screen:

- Data Elements: Username/Email, Password.
- Mandatory fields: Username/Email and Password.
- Controls: Users are barred from proceeding without correct credentials.
- Access Restrictions: Exclusively accessible by unregistered users or logged-out users.

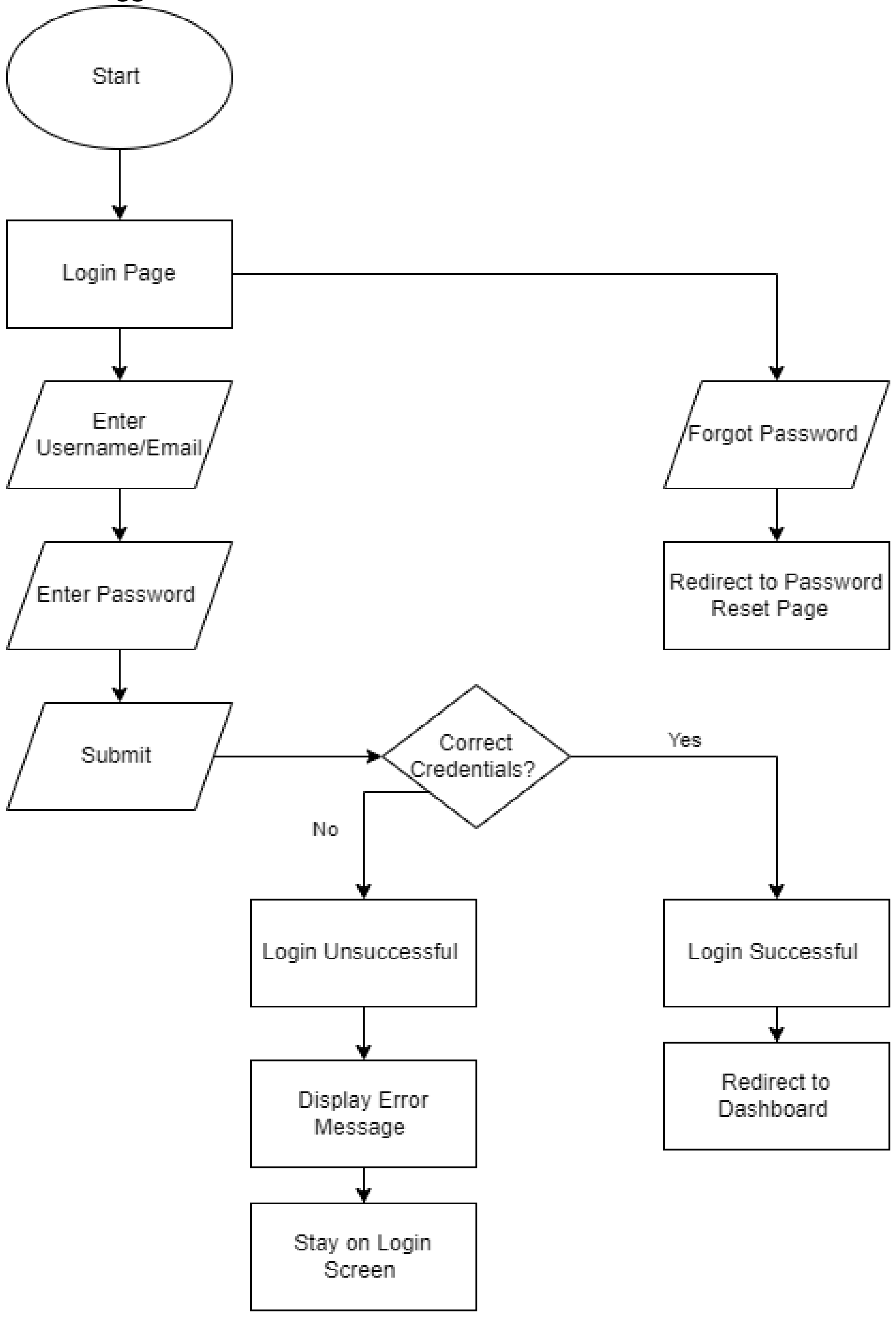


Figure 4: Login Workflow

- Figure 4 illustrates the login workflow for a system or application. The process begins at 'Start', leading to a 'Login Page' where the user is prompted to enter their username or email, followed by their password. After submitting the credentials, the system checks for their correctness. If the credentials are incorrect, the workflow directs to 'Login Unsuccessful' where an error message is displayed and the user remains on the login screen. If the user has forgotten their password, there is a branch from the 'Enter Username/Email' step to a 'Forgot Password' option, which leads to a 'Redirect to Password Reset Page'. If the credentials are correct, the user is directed to 'Login Successful' and then redirected to the dashboard of the application.

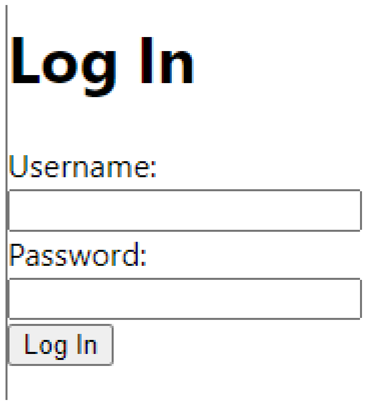
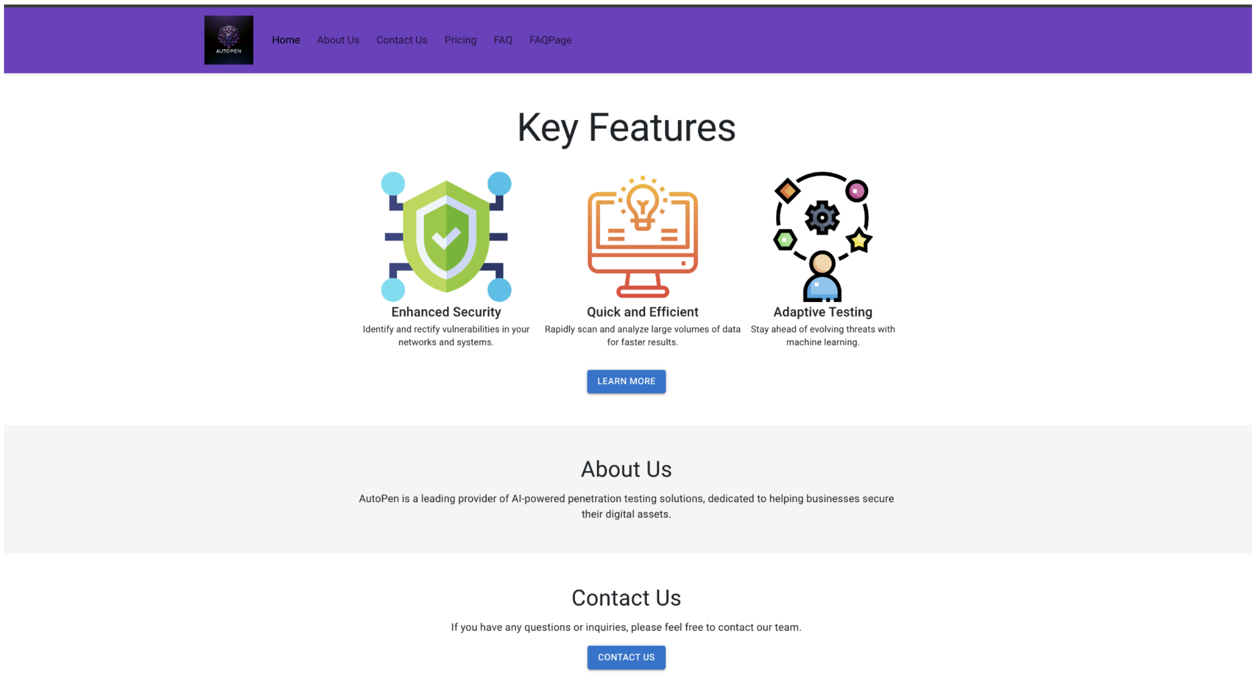


Figure 5: Login Page

- Figure 5 showcases a simplistic user interface for a login page. It features a straightforward layout with two input fields, one for the 'Username' and another for the 'Password'. Below the input fields is a 'Login' button to submit the credentials. The page is titled 'Log In' to clearly indicate its purpose to the user. The design emphasizes clarity and ease of use, with a minimalistic approach that avoids any unnecessary distractions.



Key Features

Enhanced Security
Identify and rectify vulnerabilities in your networks and systems.

Quick and Efficient
Rapidly scan and analyze large volumes of data for faster results.

Adaptive Testing
Stay ahead of evolving threats with machine learning.

About Us
AutoPen is a leading provider of AI-powered penetration testing solutions, dedicated to helping businesses secure their digital assets.

Contact Us
If you have any questions or inquiries, please feel free to contact our team.

Figure 6: Home Page

- Figure 6 displays a screenshot of a home page from a website, which is designed to provide a clean and informative user experience. The top portion of the page includes a navigation bar with links to 'Home', 'About us', 'Contact Us', 'Pricing', and 'FAQ/Help' sections. The main content area highlights the 'Key Features' of the service, which are 'Enhanced Security', 'Quick and Efficient', and 'Adaptive Testing', each accompanied by an icon and a brief description. A 'Learn More' button is present under the features for further user engagement. The lower part of the page includes sections for 'About Us', giving a brief introduction to the company, and 'Contact Us', inviting user inquiries with a button for contact initiation. The overall design employs a clean, modern aesthetic with a focus on usability and easy access to information.

3.1.1.3 Payment Screen:

- Data Elements: Credit/Debit Card Number, CVV, Expiry Date, Billing Address.
- All fields are mandatory, and include:
  - Credit/Debit Card Number should adhere to card standards.
  - Card expiration date
  - Billing address
  - CVV should be 3 or 4 digits.
- Controls: Payment cannot be processed without valid payment details.
- Access Restrictions: Only registered users access premium features or services.

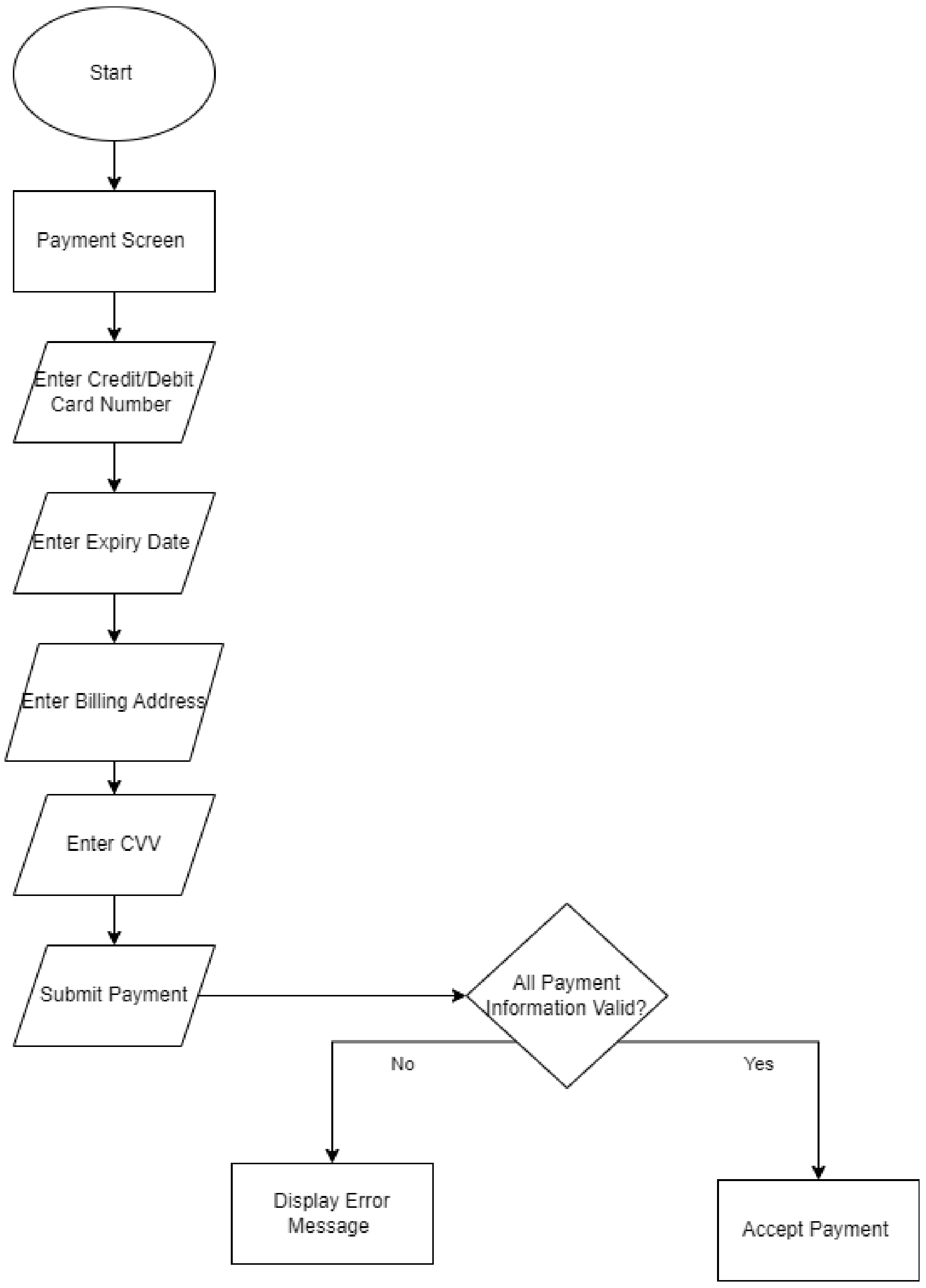
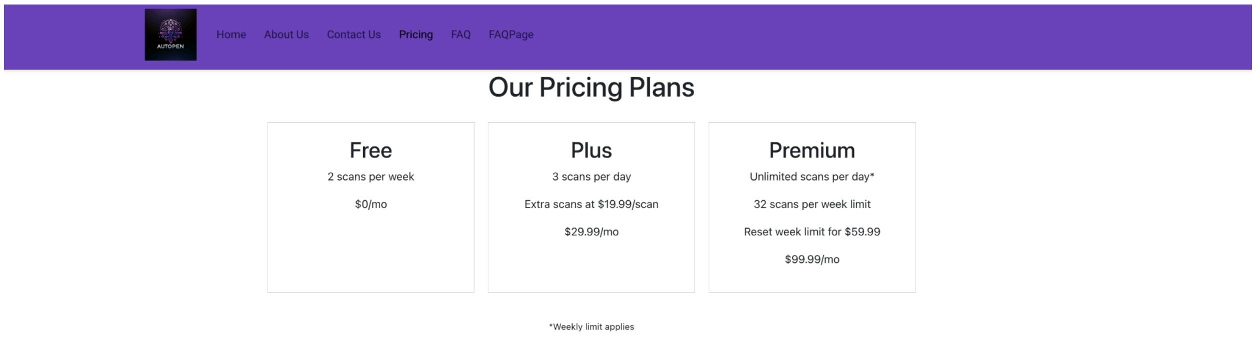


Figure 7: Payment Workflow

- Figure 7 depicts a payment workflow diagram, beginning with the 'Start' node and leading to a 'Payment Screen'. The process involves sequential steps where the user is required to enter their credit/debit card number, expiration date, billing address, and CVV. After these details are submitted, the system evaluates whether all payment information is valid. If any information is incorrect or incomplete, the workflow progresses to 'Display Error Message', prompting the user to correct the input. Conversely, if all the information is correct, the payment is accepted, completing the transaction process.



Our Pricing Plans

Free	Plus	Premium
2 scans per week	3 scans per day	Unlimited scans per day*
$0/mo	Extra scans at $19.99/scan	32 scans per week limit
	$29.99/mo	Reset week limit for $59.99
		$99.99/mo

*Weekly limit applies

Figure 8: Pricing Page

3.1.1.4 Launch/Progress Screen:

- Data Elements: Target URL, Test depth (choices: shallow, medium, deep), Optional notes or instructions.
- Mandatory field: Target URL.
- The target URL must adhere to standard URL formats.
- Test depth is selectable via a dropdown menu.
- Controls: Tests cannot be started without a valid target URL.
- Access Restrictions: Accessible only to registered and authenticated users.

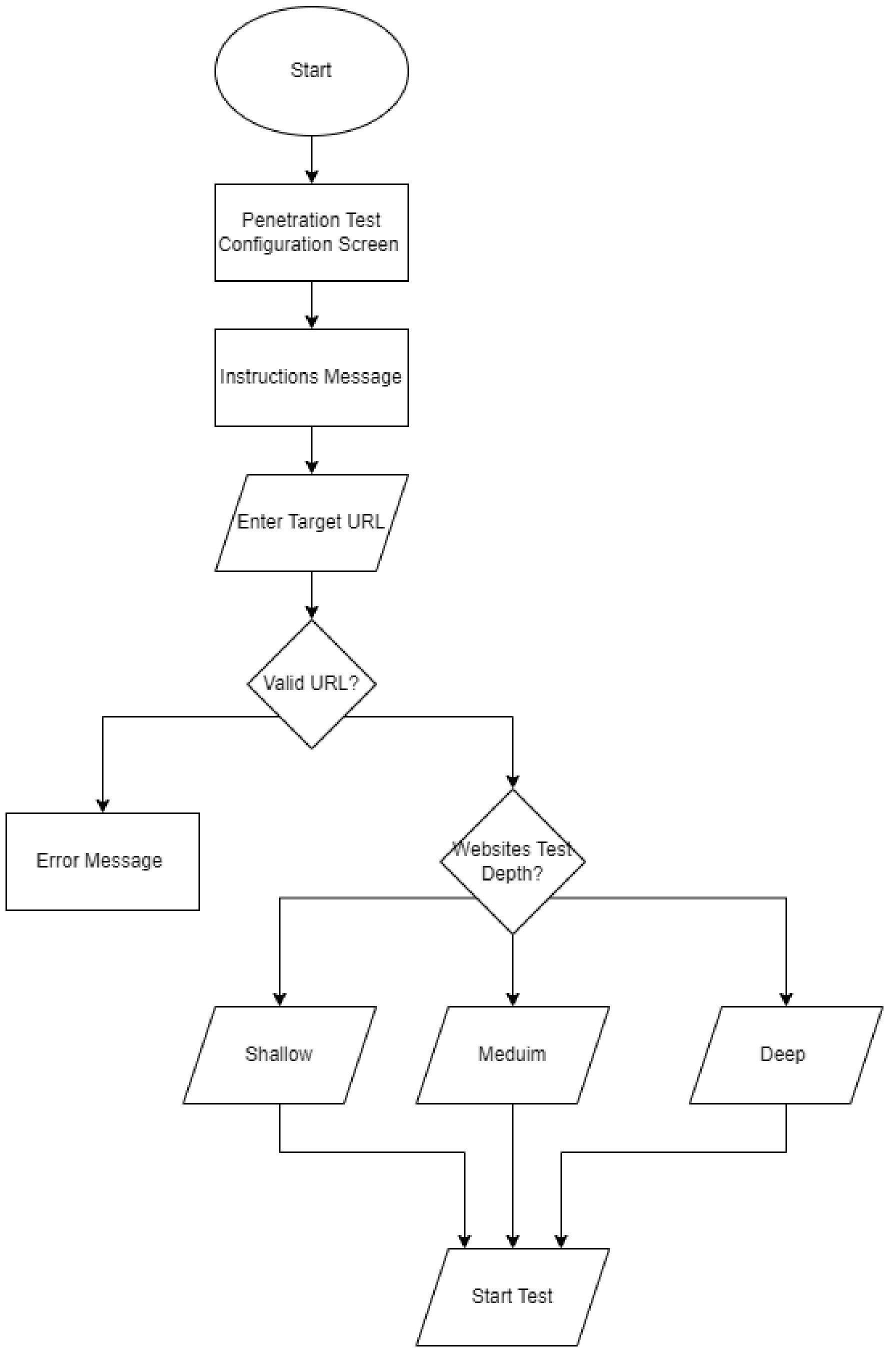


Figure 9: Penetration Configuration Workflow

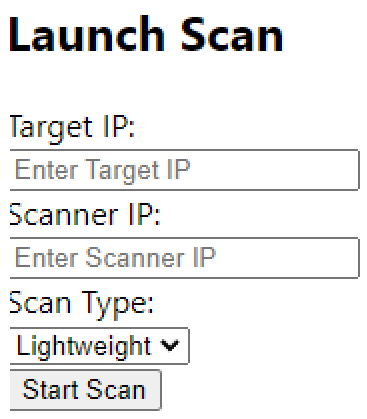


Figure 10: Launch Scan Page

3.1.1.5 Generate Report Screen:

- Data Elements: Report contents, Vulnerabilities Found, Suggested Fixes, Download Report Button.
- Controls: Users can download or print the generated report.
- Access Restrictions: Available post-completion of a penetration test.

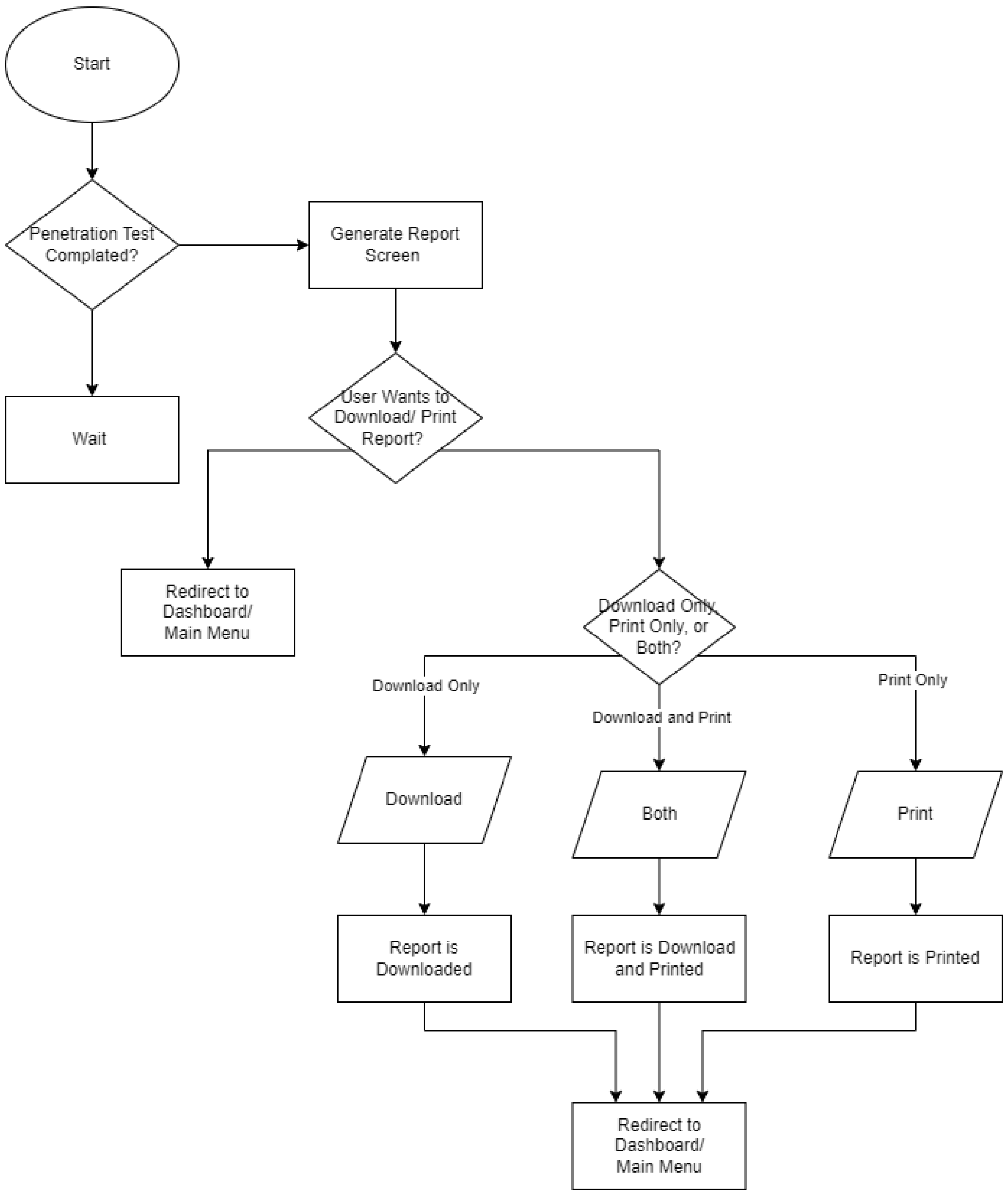
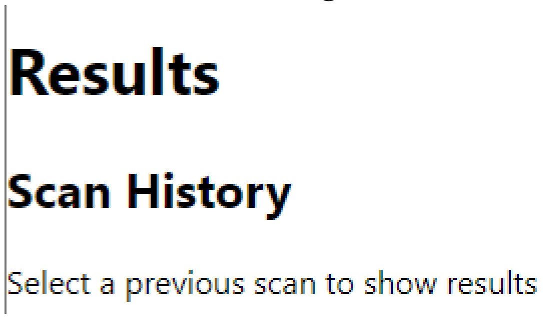

Start
Penetration Test Complated?
Generate Report Screen
Wait
User Wants to Download/ Print Report?
Redirect to Dashboard/ Main Menu
Download Only, Print Only, or Both?
Download Only
Download and Print
Print Only
Download
Both
Print
Report is Downloaded
Report is Download and Printed
Report is Printed
Redirect to Dashboard/ Main Menu

Figure 11: Generate Report Flow



Results

Scan History

Select a previous scan to show results

Figure 12: Results Page

3.1.2 Miscellaneous Messages:

- Success messages upon successful registration or test initiation.
    - Error messages detailing the nature of the error, e.g., "Invalid email format" or "Passwords do not match."

3.1.3 Transaction Codes:

REG_USER: User registration
INIT_TEST: Initiate penetration test

3.2 Outputs

System outputs primarily consist of data display screens showing test results, user profiles, and system notifications.

3.2.1 Test Result Screen:

- Identification Code: **TEST_RES**
- Contents: Vulnerabilities detected, severity levels, potential fixes, and an overall security rating. This will be represented in a graphical format using line graphs and pie charts.
- Purpose: To provide users with insights into their system's vulnerabilities and guide them on mitigation.
- Primary Users: Registered users who initiated the tests.
- Access Restrictions: Only the user who initiated the test can view its results.

3.2.2 User Profile Screen:

- Identification Code: **USER_PROF**
- Contents: User's registered details, past test history, and configurations.
- Purpose: Allows users to view and edit their details and review past tests.
- Primary Users: Registered users.
- Access Restrictions: Users can only view their own profiles.

3.2.3 System Notification Screen:

- Identification Code: **SYS_NOTIF**
- Contents: System-related notifications, updates, or alerts.
- Purpose: To keep users informed about system-related updates or changes.
- Primary Users: All users of the platform.
- Access Restrictions: Varies based on the nature of the notification. Some might be general, while others might be user-specific.

4 DETAILED DESIGN

This section is about the detailed design of the product, giving more details about the hardware and software going into it. The formatting for Section 4 is as follows: Section

4.1 is about the design of the products hardware, and Section 4.2 is about the design of the products software.

4.1 Hardware Detailed Design

At the core of Autopen is a Python-based backend that orchestrates and drives the penetration tests, while users interact through a responsive web interface. This detailed design focuses on the underlying cloud infrastructure specifications essential for "AutoPen" and the user device requirements to interact with the system optimally.

Cloud Server Component for "AutoPen":
"AutoPen" primarily functions as a web application with its core logic and backend operations developed in Python. Its hosting on cloud infrastructure ensures scalability, redundancy, and efficiency. Here are the specifications tailored for the application:

Instance Type:
- High-compute cloud instance optimized for tasks such as Burp Suite operations

Memory:
- Website host does not specify an exact number.

Storage:
- High-speed SSD storage, at 20GB, ensuring swift data read/write operations which are crucial for real-time penetration testing analytics.

Processor:
- Multi-core CPUs that cater well to Python's processing needs, ensuring rapid response times for user requests.

User Device Requirements for Accessing "AutoPen" Web Interface:
Users access "AutoPen" via its web interface. Optimal experience demands:

Desktop Systems/Laptops:
- Browser: Up-to-date web browser such as Chrome, Firefox, Safari, or others capable of supporting modern web standards and JavaScript.
- Memory: At least 4GB RAM for smooth functioning.
- Processor: Dual-core processor or better to ensure quick rendering and processing of web content.
- Monitor resolution: 1366x768 pixels or higher for the best display experience.

Tablets/Smartphones:
- Browser: Default mobile browsers, like Safari for iOS and Chrome for Android.
- Memory: 2GB RAM or more for fluid navigation and interaction.
- Screen resolution: A responsive design of "AutoPen" ensures adaptability to various screen sizes, from smartphones to tablets.

4.2 Software Detailed Design

"AutoPen" is structured around multiple software modules that collectively facilitate its primary function: automated penetration testing via a web-based platform. Each module focuses on distinct functionalities, ensuring optimal performance and streamlined operations.

4.2.1 User Management Module:

- Narrative Description: Manages user registration, authentication, profile management, and session handling. It ensures secure and seamless user interactions.
- Interfaces: Interacts with the PostgreSQL Database Module for user data storage and retrieval and the Notification Module for alerts.
- Data Elements: Username, Password (hashed and salted), Email, User profile details.
- Graphic Representation/Reference: <Explanation of what the diagram is about goes here>
- Visual Representation/Reference: <picture of module and Explanation of what the picture is about goes here>

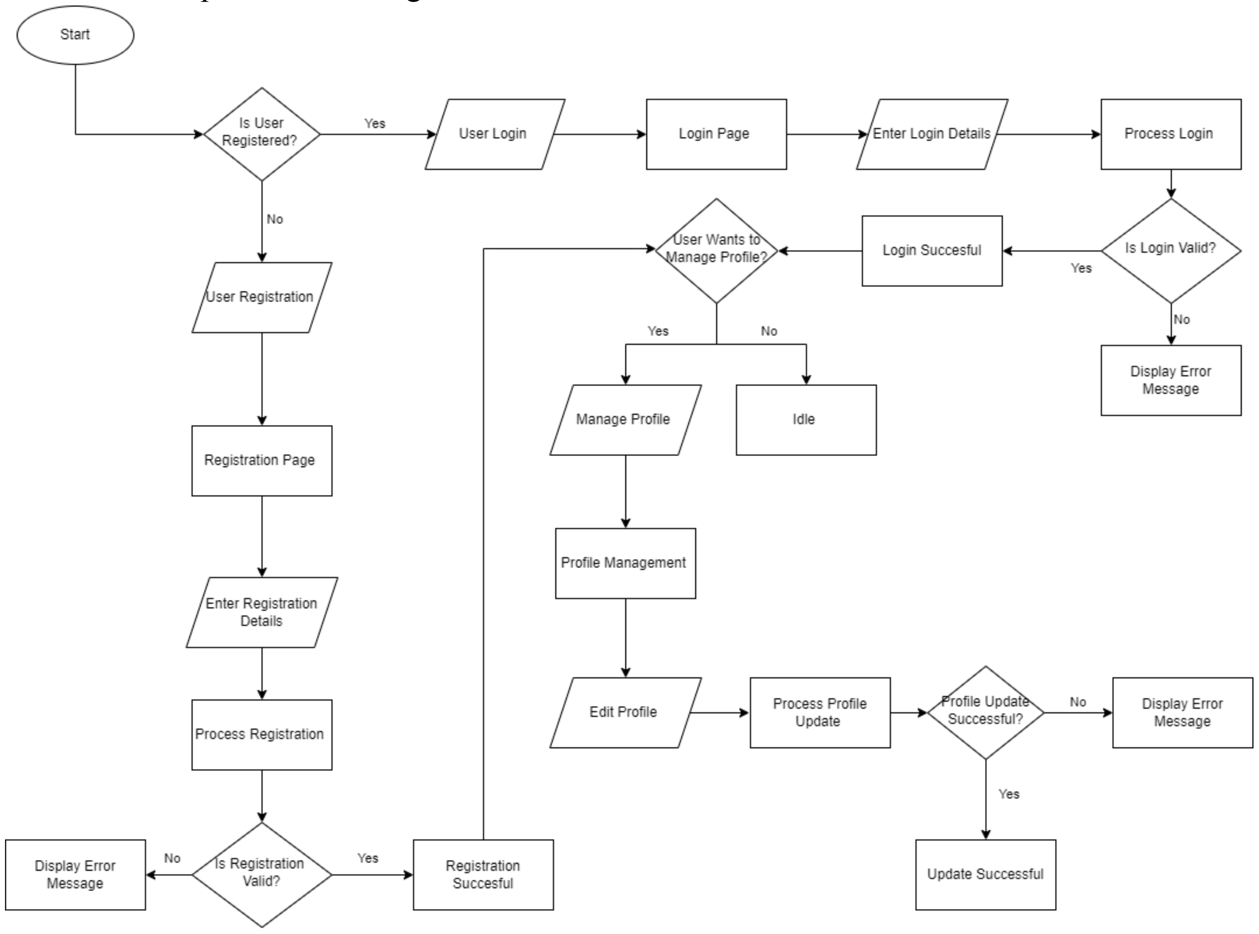


Figure 13: User Management Module Flowchart

4.2.2 Burp Suite Penetration Testing Module:

- Narrative Description: The heart of "AutoPen", this module handles the Burp Suite driven penetration tests. It deploys Burp Suite to simulate and conduct tests.

- Interfaces: Communicates with the PostgreSQL Database Module for storing test configurations and results.
- Data Elements: Target URL, Test parameters, Burp Suite outputs, Vulnerability reports.

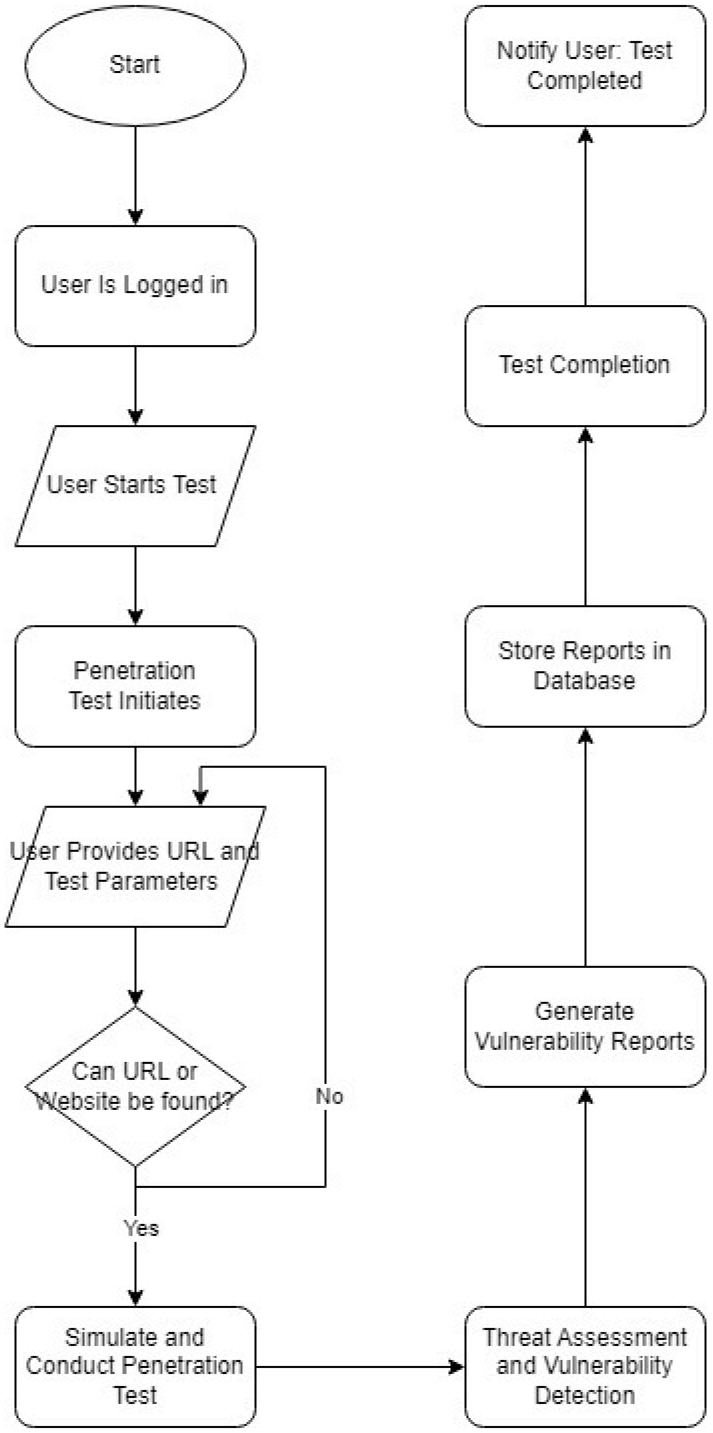
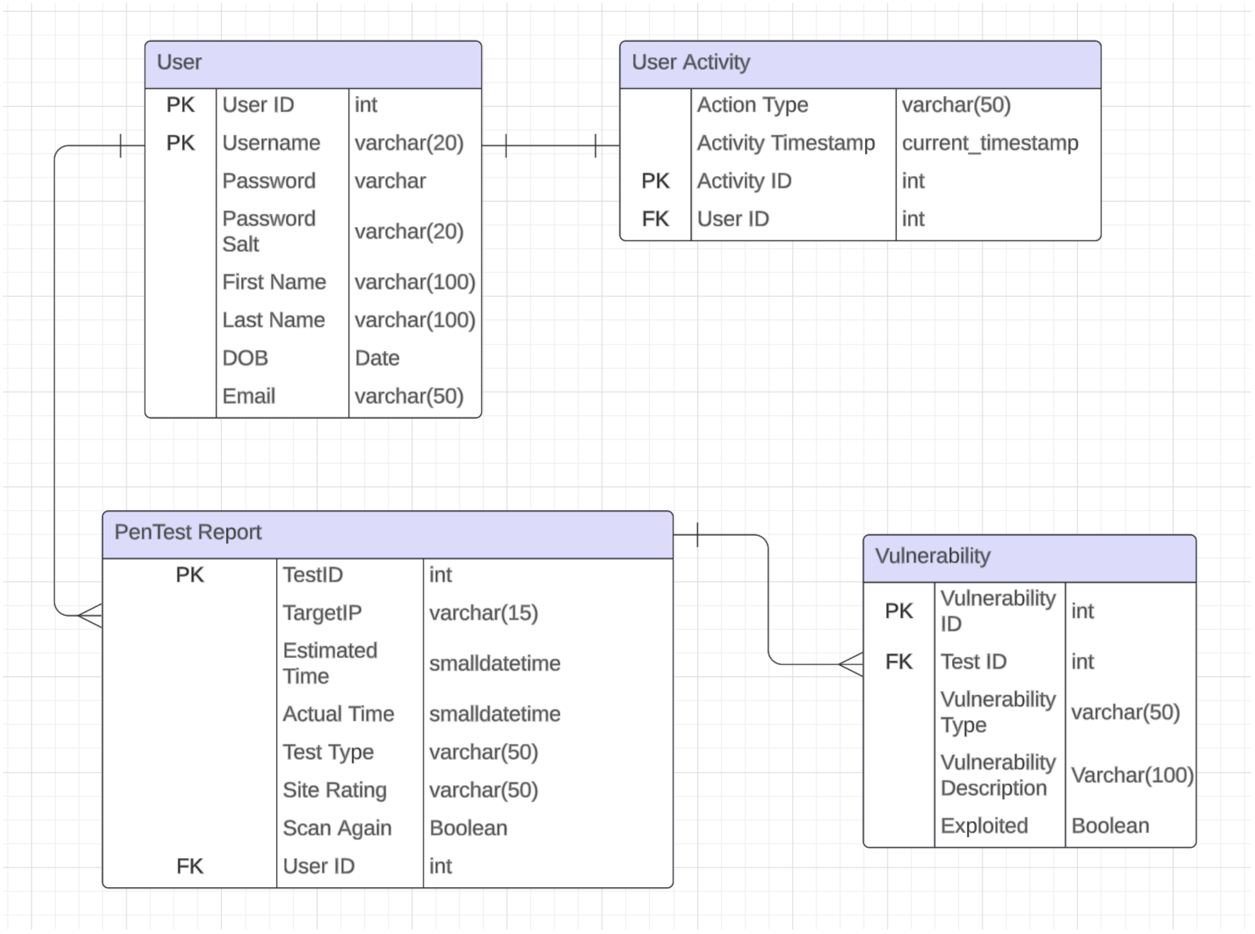

Start
User Is Logged in
User Starts Test
Penetration Test Initiates
User Provides URL and Test Parameters
Can URL or Website be found?
No
Yes
Simulate and Conduct Penetration Test
Threat Assessment and Vulnerability Detection
Generate Vulnerability Reports
Store Reports in Database
Test Completion
Notify User: Test Completed

Figure 14: Burp Suite Penetration Testing Module Flowchart

4.2.3 Database Module:

- Narrative Description: Centrally manages all data storage and retrieval operations, ensuring data consistency, integrity, and security.
- Interfaces: Serves all other modules, providing them with necessary data storage and access functionalities.
- Data Elements: User data, test configurations, system logs.



User		
PK	User ID	int
PK	Username	varchar(20)
	Password	varchar
	Password Salt	varchar(20)
	First Name	varchar(100)
	Last Name	varchar(100)
	DOB	Date
	Email	varchar(50)

User Activity		
	Action Type	varchar(50)
	Activity Timestamp	current_timestamp
PK	Activity ID	int
FK	User ID	int

PenTest Report		
PK	TestID	int
	TargetIP	varchar(15)
	Estimated Time	smalldatetime
	Actual Time	smalldatetime
	Test Type	varchar(50)
	Site Rating	varchar(50)
	Scan Again	Boolean
FK	User ID	int

Vulnerability		
PK	Vulnerability ID	int
FK	Test ID	int
	Vulnerability Type	varchar(50)
	Vulnerability Description	Varchar(100)
	Exploited	Boolean

Figure 15: Web Database Schema

4.2.4 Notification Module:

- Narrative Description: Responsible for sending real-time updates, alerts, and notifications to users.
- Interfaces: Mainly communicates with the User Management and Burp Suite Penetration Testing modules.
- Data Elements: Notification content, User contact details, Notification logs.

4.3 Internal Communications Detailed Design

Given that "AutoPen" is a web-based application, internal communications play a pivotal role, particularly when discussing backend-server to database interactions, real-time data exchange, and synchronous/asynchronous task handling. The listing of this section includes the main points about the topic of internal communication of the product, and a bit of text and/or a diagram going into more detail about said topic.

- Number of Servers and Clients: Given it's cloud-hosted, there's a dynamic allocation of servers based on demand. Every user accessing the platform is considered a client.
- Specifications for Bus Timing and Control: Standard cloud infrastructure communication protocols, optimized for minimal latency.
- Data Exchange Formats: Primarily JSON for API interactions, ensuring lightweight and structured data transfers.
- Graphical Representation: (A network diagram showcasing the cloud server clusters, data centers, and potential CDNs would be illustrated, providing clarity on data flow and infrastructure layout.)
- LAN Topology: In a cloud environment, virtual LANs (VLANs) ensure data segregation and traffic optimization.

5 EXTERNAL INTERFACES

This section is about the external interfaces of the product. The format of Section 5 is as follows: Section 5.1 is about the architecture of the external interfaces, and Section 5.2 goes into detail about the design of the external interfaces.

5.1 Interface Architecture

For "AutoPen" to function efficiently, it interfaces with external systems. These could range from payment gateways for subscription services to external databases for vulnerability definitions or cloud storage solutions for report storage. This section elaborates on the electronic interplay between "AutoPen" and these external systems.

Interface Architecture:
"AutoPen" interfaces primarily via web-based APIs for real-time interactions and scheduled batch transfers for periodic data syncing or backup. These interfaces ensure seamless data exchange and functional interplay.

Architecture:
Wide Area Networks (WAN) for expansive data transfers.
Gateway interfaces for payment processes or other third-party services.
Diagram: A diagram would illustrate the connection pathways between "AutoPen" and external systems, emphasizing data flow directionality and potential protocols. This would correlate with context diagrams presented earlier.

5.2 Interface Detailed Design

This section describes each kind of interface that the product uses and Payment Gateway Interface:

- Data Format: Standard Payment Data Format which includes user billing details, transaction IDs, payment status, and timestamps.
- Hand-shaking Protocols: Initial request from "AutoPen" to payment gateway, acknowledgment receipt from gateway, followed by transaction status.
- Error Reports: Payment failures or discrepancies trigger error reports. These are

logged in "AutoPen" and potentially forwarded to users via notifications.
- Query and Response: Transaction ID query fetches transaction status.

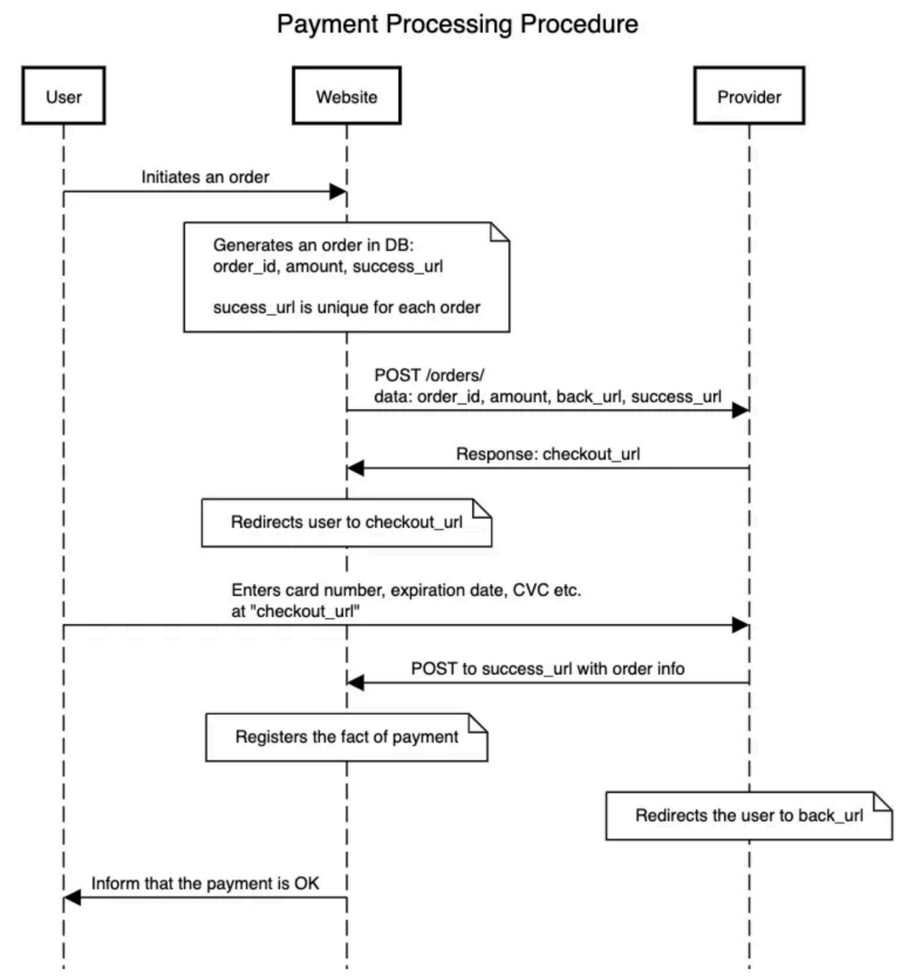

Payment Processing Procedure
User
Website
Provider
Initiates an order
Generates an order in DB:
order_id, amount, success_url
sucess_url is unique for each order
POST /orders/
data: order_id, amount, back_url, success_url
Response: checkout_url
Redirects user to checkout_url
Enters card number, expiration date, CVC etc.
at "checkout_url"
POST to success_url with order info
Registers the fact of payment
Redirects the user to back_url
Inform that the payment is OK

Figure 16: Payment Process through Django

External Vulnerability Database Interface:
- Data Format: Vulnerability definitions, metadata, timestamps, severity rankings, and suggested mitigation measures.
- Hand-shaking Protocols: Initiation request, acknowledgment, data transfer, and completion acknowledgment.
- Error Reports: Any discrepancies or data transfer failures result in error logs.
- Query and Response: Query by vulnerability ID returns specific vulnerability details.

Cloud Storage Interface for Report Storage:
- Data Format: User ID, timestamp, report data, file format, and encryption details.
- Hand-shaking Protocols: File upload initiation, acknowledgment, upload progress, and completion status.

- Error Reports: Failed uploads or data mismatches lead to error notifications.
- Query and Response: File retrieval by user ID and timestamp fetches specific reports.

6 SYSTEM INTEGRITY CONTROLS

The AutoPen system enforces a stringent protocol where access to critical data items is exclusively reserved for the creators and developers of the system, a measure that guarantees only those with authorized credentials are empowered to view and manipulate sensitive information. This restrictive approach should substantially lower the risk of unauthorized access or potential data breaches. Within the framework of Internal Security for Critical Data Access, the application adopts a Role-Based Access Control (RBAC) strategy, designed to allocate access and functional capabilities in strict accordance with the user's assigned role, effectively minimizing the chances of unauthorized data disclosure. The application delineates clear roles in alignment with operational requirements and security policies, ensuring operational functionality is maintained without compromising security integrity. These roles are crafted to reflect the specific responsibilities within the system, such as administrators who are given system access for user management, test configurations, and viewing all penetration test results, managers with permissions to configure tests, view results for their team's projects, and manage project settings, and users, specifically penetration testers, who are allowed to configure and initiate penetration tests and view their own test results. Each role is endowed with permissions that precisely mirror their responsibilities, a system that ensures the security measures do not impede the operational efficiency of the application.

Implementing authentication and authorization mechanisms, such as Multi-Factor Authentication (MFA), are critical components of this structure. MFA requires users to undergo multiple verification steps during login, which includes the use of a strong, unique password that meets industry-standard complexity requirements and token-based authentication. Following the authentication phase, the application conducts an authorization process to verify if the user holds the necessary permissions, as defined by their role, to carry out the requested action or access certain data, ensuring that users' interactions with the application are confined to the privileges associated with their roles. To maintain the integrity and security of the application, audit procedures are put in place. These procedures are designed to flag any unauthorized access attempts immediately, generating instant notifications to designated administrators to allow for rapid response to any potential security incidents, allowing "AutoPen" systems to preserve a competent security posture.

7 SYSTEM DISCLAIMERS

To protect both users and owners of "AutoPen", this document includes a list of disclaimers concerning the use of the AutoPen application. For users to operate within “AutoPen” terms of service they must first review and confirm their understanding of all “AutoPen” disclaimers.

Disclaimer 1: AutoPen test results that are generated are intended solely for the users who have requested the specific assessment. Please keep in mind that these results are confidential and should not be shared without proper authorization. If you have any questions or concerns regarding the test results, we encourage you to contact our support team for further assistance."

Disclaimer 2: Parts of document contain statements written with the help of ChatGPT

Disclaimer 3: Information included in this document is subject to change during the process of system design.