# Software Specification Document

Version = 5.2

# Author: Project Developer: Puya Pakshad

## 1. Introduction

This project involves collecting and integrating software vulnerability data from multiple sources, including NVD, CVE, OSV, and MITRE. The collected data is cleaned, standardized, and stored in a centralized database repository for analysis. The primary goal is to identify trends, such as the most vulnerable projects of 2024 or the prevalence of specific bug types like buffer overflows, and provide actionable insights through user-friendly dashboards. These tools empower security experts to make informed decisions and enhance software resilience.

## 2. System Overview

The system collects and processes data from the following primary sources using their respective APIs:  
1. CVE: Using cve-search/api to gather detailed vulnerability entries.  
2. NVD: Utilizing nvd.nist.gov/api for vulnerability severity scores and metadata.  
3. OSV: Accessing osv.dev/api for open-source vulnerability insights.  
4. MITRE: Fetching data through mitre.org/cve-api for curated vulnerability summaries.  
  
The data collected is analyzed to identify trends, such as specific bug types or high-risk projects, and visualized for better understanding and decision-making.

## 3. Software Features

1. 1. Presenting a well-organized table of vulnerabilities extracted from local and online data sources, with search functionality and filters based on various vulnerability attributes.

2. Supporting pagination for better navigation through large datasets in the vulnerability table.

3. Utilizing APIs to collect detailed data from CVE, NVD, OSV, and MITRE repositories.

4. Visualizing trends and patterns using interactive charts like pie, bar, and area charts.

5. Performing statistical analysis, including metrics like mean, standard deviation, and severity scores.

6. Predicting future vulnerabilities using a machine learning model based on historical data.

7. Providing customizable insights to analyze vulnerabilities by type, year, and severity.

8. Offering a user-friendly, responsive design for seamless interaction on various devices.

## 4. Technical Details

The system is built using the following technologies and tools:  
1. Backend: Python/Flask  
2. Database: PostgreSQL for centralized data storage  
3. Frontend: HTML, CSS, JavaScript (with Chart.js for visualizations)  
4. APIs: Integration with CVE, NVD, OSV, and MITRE APIs  
5. Machine Learning: Scikit-learn for predictive modeling  
6. Deployment: AWS for hosting and scalability

## 5. Predictive Modeling

The system utilizes a Linear Regression model implemented using Scikit-learn to predict future vulnerabilities based on historical data. The predictions provide actionable insights for cybersecurity teams to anticipate and mitigate risks effectively. The model is trained on past data, cleaned, and standardized before processing.