*Open Source Intelligence Application*

**Project Management Plan**



Department of Information Technology and Management

*November 2024*

**Revision History**

Note: The revision history cycle begins once changes or enhancements are requested after the document has been baselined.

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 12/3/2024 | 3.0 | Final Update | Puya Pakshad |
| 11/25/2024 | 2.0 | Final Project Management Plan | Henil Gandhi |
| 11/11/2024 | 1.0 | Developed an Initial Project Management Plan | Henil Gandhi |

Instructions

| **Activity** | **New Capability (1)** | **Feature Enhancement (2)** |
| --- | --- | --- |
| **Field Deployment (A)** | Yes | No |
| **Cloud/Web Deployment (B)** | Yes | Yes |
| **Mobile Application (C)** | No | No |

**Table Of Content**

1. **Introduction  
   1.1 Project Overview  
   1.2 Scope Statements  
   1.3 Goals and Objectives  
   1.4 Stakeholders and Key Personnel**
2. **Project Organization**
3. **Acquisition Process**
4. **Monitoring and Control Mechanisms**
5. **Systems Security Plans and Requirements**
6. **Work Breakdown Structure (WBS)**
7. **Project Success Criteria**
8. **Communication Management Plan**
9. **Risk Management Plan**
10. **Software Configuration Management (SCM) Plan**
11. **Training Plan**
12. **Quality Assurance Plan**
13. **Project Measurement Plan  
    13.1 Description  
    13.2 Performance Measurements**
14. **Reference Materials**

**1. Introduction**

**1.1 Project Overview**

**The Vulnerability Visualization Application is a web-based platform designed to collect, analyze, and visualize software vulnerability data from four trusted sources: NVD, CVE, OSV, and MITRE. The platform uses APIs to fetch data, processes it to identify trends, and applies machine learning models (e.g., linear regression) to predict future vulnerabilities. This tool provides actionable insights through user-friendly dashboards and visualizations to assist cybersecurity teams in risk prioritization and decision-making.**

**1.2 Scope Statements**

* **In-Scope:**
  + **Integration of APIs from NVD, CVE, OSV, and MITRE.**
  + **Visualization of vulnerabilities using interactive charts such as bar charts, donut charts, and area charts.**
  + **Predictive analysis using machine learning models for forecasting future vulnerabilities.**
  + **Responsive, user-friendly web interface with search and pagination functionalities.**
* **Out-of-Scope:**
  + **Real-time vulnerability monitoring.**
  + **Implementing automated security patches.**

**1.3 Goals and Objectives**

* **Enable security teams to analyze vulnerabilities with dynamic reporting.**
* **Provide predictive insights on future vulnerabilities to enhance planning.**
* **Create a centralized repository for data integration from multiple APIs.**

**1.4 Stakeholders and Key Personnel**

* **Stakeholders:**
  + **Cybersecurity professionals (end-users).**
  + **Researchers and developers.**
  + **API providers (NVD, CVE, OSV, MITRE).**
* **Key Personnel:**
  + **Project Manager: Oversees planning, execution, and deliverables.**
  + **Web Developer: Manages API integration and backend/frontend development.**
  + **Data Scientist: Implements and optimizes machine learning models.**
  + **UI/UX Designer: Designs user-friendly and responsive interfaces.**
  + **QA Engineer: Tests and validates the platform for accuracy and security.**

**2. Project Organization**

* **Development Team: Responsible for API integration, data processing, and visualization features.**
* **QA Team: Ensures functionality and security compliance.**
* **Project Management Office (PMO): Tracks timelines, mitigates risks, and ensures alignment with project goals.**

**3. Acquisition Process**

* **Technologies: Python (Flask) for backend, PostgreSQL for database management, Chart.js for visualizations, and jQuery for interactivity.**
* **Third-Party APIs:**
  + **NVD:** [**https://services.nvd.nist.gov/rest/json/cves**](https://services.nvd.nist.gov/rest/json/cves)
  + **CVE: https://www.cve.org/api**
  + **OSV: https://osv.dev/v1/api**
  + **MITRE: https://cve.mitre.org/api**
* **Hosting Platform: Cloud deployment on AWS for scalability and reliability.**

**4. Monitoring and Control Mechanisms**

* **Weekly sprint reviews to assess progress and address challenges.**
* **Use of Trello for task management and tracking.**
* **Automated CI/CD pipelines to ensure continuous testing and deployment.**

**5. Systems Security Plans and Requirements**

* **Secure all API communications with HTTPS and API key authentication.**
* **Implement Role-Based Access Control (RBAC) for enhanced security.**
* **Regular vulnerability scans and adherence to secure coding practices.**

**6. Work Breakdown Structure (WBS)**

1. **Planning:  
   1.1 Define requirements.  
   1.2 Consult stakeholders.**
2. **Design:  
   2.1 Create UI wireframes.  
   2.2 Design system architecture.**
3. **Development:  
   3.1 Integrate APIs and create backend logic.  
   3.2 Develop visualizations (charts, reports).**
4. **Testing:  
   4.1 Perform unit and integration testing.  
   4.2 Conduct user acceptance testing (UAT).**
5. **Deployment:  
   5.1 Configure hosting and deployment.  
   5.2 Launch final application and gather feedback.**

**7. Project Success Criteria**

* **Successful integration with four vulnerability data APIs.**
* **Accurate visualizations and predictive modeling results.**
* **Positive user feedback on usability and functionality.**

**8. Communication Management Plan**

* **Weekly status meetings with all stakeholders.**
* **Use of Slack for instant communication within the team.**
* **Email updates for external stakeholders.**

**9. Risk Management Plan**

* **Data Availability Risk: Mitigate by caching recent API data.**
* **Security Risks: Conduct regular penetration tests.**
* **Scalability Risks: Ensure cloud hosting can handle peak loads.**

**10. Software Configuration Management (SCM) Plan**

* **Version control using Git and GitHub for code repositories.**
* **Continuous integration with automated testing pipelines.**

**11. Training Plan**

* **Train end-users on how to interpret vulnerability reports.**
* **Provide technical documentation for developers and maintainers.**

**12. Quality Assurance Plan**

* **Automated testing for data accuracy and system functionality.**
* **Manual testing to ensure responsiveness and user experience.**

**13. Project Measurement Plan**

**13.1 Description**

**Track performance metrics for API response times, visualization rendering speeds, and prediction accuracy.**

**13.2 Performance Measurements**

* **API Response Time: < 500ms for data fetching.**
* **Visualization Rendering Speed: < 1 second for standard datasets.**
* **Prediction Accuracy: Maintain a minimum accuracy of 85% for linear regression forecasts.**

**14. Reference Materials**

* **NVD API Documentation:** [**https://nvd.nist.gov/**](https://nvd.nist.gov/)
* **CVE API Reference:** [**https://cve.org/**](https://cve.org/)
* **Chart.js Documentation:** [**https://www.chartjs.org/**](https://www.chartjs.org/)
* **Flask Documentation:** [**https://flask.palletsprojects.com/**](https://flask.palletsprojects.com/)