Project report

GitHub Dependency visualization

# Version Table.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Date | Changes |
| 1.0 | Lokesh Agnihotri | 3/10/2023 | Initial Draft |
|  |  |  |  |

Contents

[Version Table. 1](#_Toc129946774)

[Abstract: 2](#_Toc129946775)

[1. Introduction 2](#_Toc129946776)

[2. Problem Statement: 3](#_Toc129946777)

[3. Relevance of the Problem: 3](#_Toc129946778)

[4. Methodology: 3](#_Toc129946779)

[4.1 Retrieving repositories: 3](#_Toc129946780)

[4.2 Identifying dependencies: 3](#_Toc129946781)

[4.3 Identifying sub-dependencies: 4](#_Toc129946782)

[4.4 Creating a JSON file: 4](#_Toc129946783)

[4.5 Visualizing the data in Splunk Cloud: 4](#_Toc129946784)

[5. Results 4](#_Toc129946785)

[6. Quality Validation: 4](#_Toc129946786)

[7. Conclusion and Next Steps: 5](#_Toc129946787)

# Abstract:

Managing dependencies is a crucial task in software development, particularly for large organizations with multiple repositories. Identifying all the dependencies and sub-dependencies of a software application can be a challenging task. In this research paper, I present a solution that uses the GitHub API to retrieve all the repositories of a given GitHub username, identifies the dependencies and sub-dependencies across multiple repositories using a Python script, and visualizes the data in the form of a dashboard in Splunk Cloud. The solution can help developers better understand the complexity of their software and identify potential issues before they become major problems.

1. Introduction:

In the world of software development, many applications are built on top of other software components. These components are typically referred to as dependencies. However, as the number of dependencies grows, it becomes increasingly difficult to manage them. This is particularly true for large organizations with many repositories, where dependencies can quickly become a complex web of interrelated components. To address this problem, I have developed a solution that pulls all the dependencies and sub-dependencies of all the repositories of a given GitHub username, and then creates a JSON file in Python, which is sent to a Splunk Cloud instance where the data is visualized in the form of a dashboard.

# Problem Statement:

The problem I aim to solve is the increasing complexity of managing dependencies in software development. The challenge is to identify all the dependencies and sub-dependencies across multiple repositories and visualize the data in a way that is easily understandable.

# Relevance of the Problem:

As software development becomes more complex, managing dependencies becomes increasingly difficult. The ability to visualize dependencies across multiple repositories can help developers better understand the complexity of their software and identify potential issues before they become major problems.

# Methodology:

To solve this problem, I developed a Python script that uses the GitHub API to retrieve all the repositories of a given GitHub username. For each repository, I use a combination of regular expressions and parsing to identify all the dependencies listed in the repository's package or requirements files. I then use recursion to identify all the sub-dependencies of each dependency. Finally, I create a JSON file that contains all the dependencies and sub-dependencies and send it to a Splunk Cloud instance where it is visualized in the form of a dashboard.

The Following Steps were taken for the development of the solution.

## 4.1 Retrieving repositories:

The first step in the process is to retrieve all the repositories of a given GitHub username using the GitHub API. The API provides a simple and easy way to retrieve repository information for any user.

## 4.2 Identifying dependencies:

For each repository, I identify the dependencies listed in the repository's package or requirements files. I used regular expressions to parse the files and identify the dependencies. For example, in a Python project, the requirements.txt file lists all the dependencies required by the project. Similarly, in a Node.js project, the package.json file lists all the dependencies required by the project.

## 4.3 Identifying sub-dependencies:

Once the direct dependencies are identified, I use recursion to identify all the sub-dependencies of each dependency. This process continues until all sub-dependencies are identified.

## 4.4 Creating a JSON file:

Finally, I create a JSON file that contains all the dependencies and sub-dependencies. The JSON file is created in Python and contains a list of dependencies, where each dependency is represented as a dictionary with the following keys:

Name: The name of the dependency

Version: The version of the dependency

Sub-dependencies: A list of sub-dependencies (if any)

The JSON file is then sent to a Splunk Cloud instance where it is visualized in the form of a dashboard.

## 4.5 Visualizing the data in Splunk Cloud:

Splunk Cloud is a cloud-based service that enables users to search, analyze, and visualize data from various sources. I used Splunk's HTTP Event Collector (HEC) to send the JSON file to the Splunk Cloud instance. Once the data is received, I used the Splunk Web Framework to create a dashboard that visualizes the data.

1. Results:

The solution I developed successfully retrieves all the repositories of a given GitHub username, identifies all the dependencies and sub-dependencies across multiple repositories, and visualizes the data in the form of a dashboard in Splunk Cloud. The dashboard provides developers with a clear and easy-to-understand view of the dependencies in their software, allowing them to identify potential issues before they become major problems.

# Quality Validation:

To validate the quality of the solution, I conducted several tests using different GitHub usernames and repositories. The solution was able to successfully retrieve all the repositories and dependencies for each username and repository tested. The JSON file created by the solution was also validated using several online JSON validators, which confirmed that the file is valid and well-formed.

# Conclusion and Next Steps:

In conclusion, I have developed a solution that provides developers with a clear and easy-to-understand view of the dependencies in their software. The solution successfully retrieves all the repositories of a given GitHub username, identifies all the dependencies and sub-dependencies across multiple repositories, and visualizes the data in the form of a dashboard in Splunk Cloud.

The next steps for this solution include adding additional features to the dashboard to provide even more insights into software dependencies, such as identifying potential security vulnerabilities or highlighting dependencies that are no longer maintained. Additionally, the solution could be extended to work with other source code management systems such as Bitbucket or GitLab. The code can be extended to run the script automatically at a certain interval of time and user has to click once on a provided dashboard link to see the updated visuals.

In summary, the solution I developed provides a valuable tool for software developers to manage and understand the complexity of their software dependencies.