**CyberCube Vulnerabilities and CVE Changes API Documentation**

**Overview**

This API provides endpoints to fetch data from the vulnerabilities and cve\_changes tables in the cybercube database. The API is built using Flask and MySQL, and it supports querying based on cve\_id and product\_id.

**Prerequisites**

* Python 3.x
* Flask
* MySQL database

**Installation**

1. Clone the repository or download the code.
2. Install the required packages using pip:

Bash - pip install flask mysql-connector-python

1. Ensure you have a MySQL database running and the cybercube database with the required tables (vulnerabilities and cve\_changes).

**Configuration**

Update the MySQL database connection parameters in the code as needed:

db\_config = {

'user': 'root',

'password': 'admin',

'host': 'localhost',

'database': 'cybercube'

}

**Running the API**

To run the Flask application, execute the following command:

Bash - python app.py

The API will be available at http://127.0.0.1:5000.

**API Endpoints**

**1. Get Vulnerabilities**

**URL**

GET /vulnerabilities

**Parameters**

* cve\_id (optional): Filter by specific CVE ID.
* product\_id (optional): Filter by specific Product ID.

**Example Requests**

* Get all vulnerabilities:

GET http://127.0.0.1:5000/vulnerabilities

* Get vulnerabilities by CVE ID:

GET http://127.0.0.1:5000/vulnerabilities?cve\_id=CVE-2021-12345

* Get vulnerabilities by Product ID:

GET http://127.0.0.1:5000/vulnerabilities?product\_id=Product-123

* Get vulnerabilities by both CVE ID and Product ID:

GET http://127.0.0.1:5000/vulnerabilities?cve\_id=CVE-2021-12345&product\_id=Product-123

**Response**

The response is a JSON array of vulnerabilities. Each vulnerability contains various fields such as CVE ID, source identifier, published date, last modified date, etc.

**Example Response**

[

{

"CVE\_ID": "CVE-2021-12345",

"SOURCE\_IDENTIFIER\_CODE": 1,

"PUBLISHED\_DATE": "2021-05-15T00:00:00",

"LAST\_MODIFIED\_DATE": "2021-05-20T00:00:00",

"VULNERABILITY\_STATUS\_CODE": 2,

"DESCRIPTION\_EN": "Sample vulnerability description.",

"CVSS\_BASE\_SCORE": 7.5,

"ACCESS\_VECTOR\_CODE": 3,

"CONFIDENTIALITY\_IMPACT\_CODE": 2,

"INTEGRITY\_IMPACT\_CODE": 2,

"AVAILABILITY\_IMPACT\_CODE": 2,

"BASE\_SEVERITY\_CODE": 4,

"EXPLOITABILITY\_SCORE": 8.6,

"IMPACT\_SCORE": 6.4,

"WEAKNESS\_DESCRIPTION\_CODE": 1,

"CONFIGURATION\_CRITERIA": "Sample criteria",

"PRODUCT": "Sample Product",

"PUBLISHED\_DATE\_DATE": "2021-05-15",

"PUBLISHED\_DATE\_TIME": "00:00:00",

"LAST\_MODIFIED\_DATE\_DATE": "2021-05-20",

"LAST\_MODIFIED\_DATE\_TIME": "00:00:00",

"RECORD\_ADDED\_DATE": "2021-05-15"

}

]

**2. Get CVE Changes**

**URL**

GET /cve\_changes

**Parameters**

* cve\_id (optional): Filter by specific CVE ID.

**Example Requests**

* Get all CVE changes:

GET http://127.0.0.1:5000/cve\_changes

* Get CVE changes by CVE ID:

GET http://127.0.0.1:5000/cve\_changes?cve\_id=CVE-2021-12345

**Response**

The response is a JSON array of CVE changes. Each CVE change contains various fields such as CVE ID, event name code, CVE change ID, source identifier code, created date, etc.

**Example Response**

[

{

"CVE\_ID": "CVE-2021-12345",

"EVENT\_NAME\_CODE": 1,

"CVE\_CHANGE\_ID": "Change-001",

"SOURCE\_IDENTIFIER\_CODE": 2,

"CREATED\_DATE": "2021-05-15T00:00:00",

"CREATED\_DATE\_DATE": "2021-05-15",

"CREATED\_DATE\_TIME": "00:00:00",

"RECORD\_ADDED\_DATE": "2021-05-15"

}

]

**Helper Functions**

**get\_db\_connection()**

Creates and returns a MySQL database connection.

**convert\_to\_serializable(data)**

Converts non-serializable data types (like datetime.date, datetime.datetime, and datetime.timedelta) to serializable formats.

**Code Explanation**

* The get\_db\_connection function establishes a connection to the MySQL database using the provided configuration.
* The convert\_to\_serializable function converts non-serializable data types to string or ISO format to ensure the data can be serialized to JSON.
* The /vulnerabilities endpoint retrieves data from the vulnerabilities table based on optional cve\_id and product\_id query parameters.
* The /cve\_changes endpoint retrieves data from the cve\_changes table based on the optional cve\_id query parameter.
* Both endpoints build the SQL query dynamically based on the provided parameters, execute the query, and return the results as JSON.

**Running the Application**

To run the application, execute the following command in your terminal:

Bash - python app.py

The application will be available at http://127.0.0.1:5000.

**Conclusion**

This API provides endpoints to fetch data from the vulnerabilities and cve\_changes tables in the cybercube database, supporting filtering by cve\_id and product\_id. The API returns the data in JSON format, making it easy to integrate with other systems or applications.