Documentation for Securein_Assesment

1. Introduction

I have created a web application designed to fetch and display detailed information about Common Vulnerabilities and Exposures (CVEs). It provides users with an interface to view critical details about vulnerabilities, including descriptions, CVSS metrics (severity, score, vector string), CPE (Common Platform Enumeration) matches, and more. The application interacts with a RESTful API to fetch CVE data and present it in a user-friendly format.

The project is developed using **React.js** for the frontend, with **Node.js** (Express.js) for the backend, ensuring a smooth and scalable user experience.

2. Features

- **CVE List Display**: The application retrieves and displays a list of CVEs, showcasing their ID, identifier, published date, last modified date, and status.
- **Detailed CVE Information**: Users can click on any CVE ID to view detailed information, including:
  - Descriptions (in different languages)
  - CVSS V2 metrics (severity, score, vector string, and other related metrics)
  - CPE (Common Platform Enumeration) matches
- **Responsive Interface**: The app ensures an optimal experience across devices with a minimal and clean UI layout.
- **Error Handling**: Proper error messages and loading states are displayed to improve the user experience during data fetching.

3. Technology Stack

The application is built using the following technologies:

- **Frontend**:
  - **React.js**: A JavaScript library for building user interfaces.
  - **CSS**: Inline styling to style the app’s components.
- **Backend** (for local development, optional if needed):
  - **Node.js**: A runtime environment to build the server-side logic.
  - **Express.js**: A minimal web framework for building the API.
- **API**:

- The app interacts with a RESTful API to fetch the CVE data. The data source is hosted locally at http://localhost:5000/cves/list?page=1&limit=10.

---

4. Installation and Setup

Prerequisites

- Node.js (version 14 or higher)
- npm (Node package manager) or yarn (for managing dependencies)

Steps to Run the Application Locally

1. **Clone the Repository**:
2. git clone https://github.com/yourusername/Securein_Assesment.git
3. cd Securein_Assesment
4. **Install Dependencies**: Run the following command to install all the necessary packages:
5. npm install
6. **Run the Application**: To start the development server and run the app:
7. npm start

This will run the app at http://localhost:3000 by default.

8. **API Setup**: Ensure that the API is also running on http://localhost:5000/cves/list?page=1&limit=10 for the app to fetch the data correctly.

---

5. API Details

The application fetches data from a RESTful API endpoint:

- **URL**: http://localhost:5000/cves/list?page=1&limit=10

Response Example:

The API responds with a list of CVE details, including:

- CVE ID
- Published Date
- Last Modified Date
- Status
- Descriptions (in different languages)
- CVSS metrics (base severity, base score, vector string)
- CPE matches for vulnerable platforms

6. Application Flow

1. **CVE List Page**:
   - When the app loads, it fetches a list of CVEs and displays them in a table format. The columns include CVE ID, Identifier, Published Date, Last Modified Date, and Status.
2. **CVE Details Page**:
   - Clicking on a CVE ID navigates to the detailed view of that CVE, showing descriptions, CVSS V2 metrics (severity, score, vector string), and CPE matches.

The **CVE Details Page** includes the following sections:

- **CVE Descriptions**: Displays different language descriptions of the CVE.
- **CVSS Metrics**: Shows the severity, score, and vector string.
- **CPE Matches**: Lists vulnerable platforms with their respective CPE URIs.

7. User Interface

The application uses a simple and minimalistic layout to display the data. The interface includes:

- **CVE List Table**: A table that lists CVEs with key details such as CVE ID, Identifier, Published Date, Last Modified Date, and Status.
- **CVE Details Page**: A detailed view of a selected CVE, showing descriptions, metrics, and CPE matches.

All data is displayed in a clean and easy-to-read format, with tables and lists properly aligned.

8. Error Handling

The application includes basic error handling:

- If the API fetch fails, an error message is displayed ("Failed to fetch data from server.").
- If a CVE ID is not found, a "CVE not found" message appears.
- In case of unexpected data structure, an "Invalid API response" message is shown.

9. Future Improvements

The following features could be added in future updates:

- Pagination for the CVE list (to display more CVEs).
- Authentication and authorization (to secure data and restrict access).

- Search and filter functionality for the CVE list.
- Integration with more CVE data sources for a broader range of vulnerabilities.

---

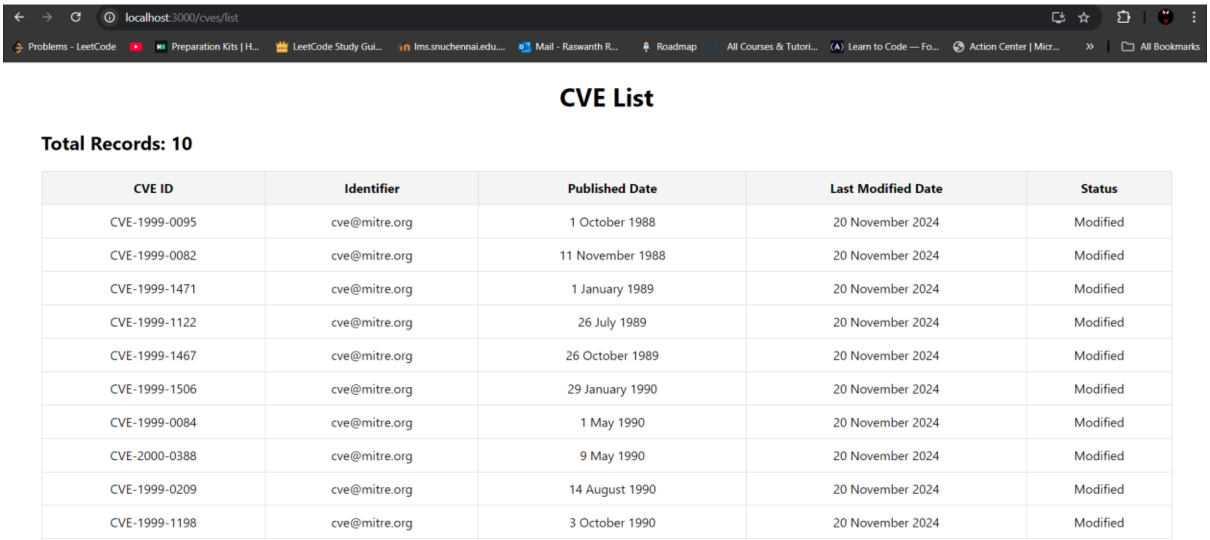
10. Conclusion

This web application provides an efficient and user-friendly way to explore CVE details, CVSS metrics, and vulnerable platforms, with the aim of increasing security awareness. The app ensures ease of access to essential vulnerability data, helping security professionals and developers identify and address potential vulnerabilities in their systems.

---

11. Screenshots

1. **CVE List Page**:

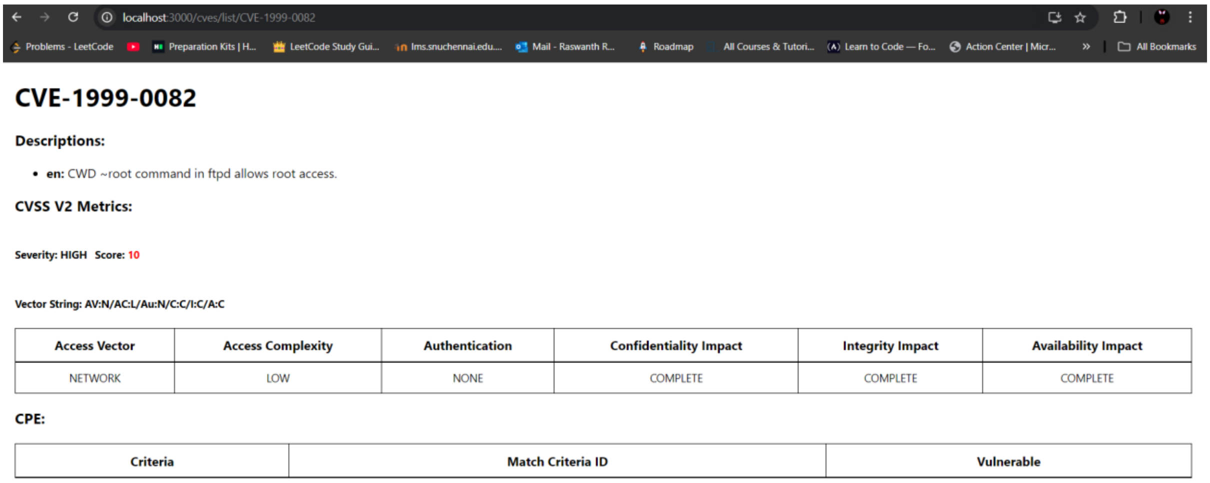


CVE List

Total Records: 10

CVE ID	Identifier	Published Date	Last Modified Date	Status
CVE-1999-0095	cve@mitre.org	1 October 1988	20 November 2024	Modified
CVE-1999-0082	cve@mitre.org	11 November 1988	20 November 2024	Modified
CVE-1999-1471	cve@mitre.org	1 January 1989	20 November 2024	Modified
CVE-1999-1122	cve@mitre.org	26 July 1989	20 November 2024	Modified
CVE-1999-1467	cve@mitre.org	26 October 1989	20 November 2024	Modified
CVE-1999-1506	cve@mitre.org	29 January 1990	20 November 2024	Modified
CVE-1999-0084	cve@mitre.org	1 May 1990	20 November 2024	Modified
CVE-2000-0388	cve@mitre.org	9 May 1990	20 November 2024	Modified
CVE-1999-0209	cve@mitre.org	14 August 1990	20 November 2024	Modified
CVE-1999-1198	cve@mitre.org	3 October 1990	20 November 2024	Modified

2. **CVE Details Page**:



CVE-1999-0082

Descriptions:

- **en:** CWD ~root command in ftpd allows root access.

CVSS V2 Metrics:

Severity: HIGH Score: 10

Vector String: AV:N/AC:L/Au:N/C:C/I:C/A:C

Access Vector	Access Complexity	Authentication	Confidentiality Impact	Integrity Impact	Availability Impact
NETWORK	LOW	NONE	COMPLETE	COMPLETE	COMPLETE

CPE:

Criteria	Match Criteria ID	Vulnerable

12. Contributing

1. **Fork the repository**: Create your own fork of the repository.
2. **Create a branch**: Create a feature branch for your changes.
3. **Commit your changes**: Commit your changes with meaningful messages.
4. **Push to your fork**: Push the changes to your forked repository.
5. **Create a Pull Request**: Open a pull request from your fork to the original repository.