**APPROACH:**

**Using MERN STACK**

**Frontend Overview:**

**CVE List Component (CVEList.js):**

**Uses useEffect to fetch CVE data from Server 2 (running on port 8000) when the component mounts or when the pagination state (currentPage or perPage) changes.**

**Constructs an API call to retrieve data based on the current page number and number of results per page.**

**Stores the total number of CVEs (totalDocs) retrieved from the API in a state variable.**

**Display:**

**Renders the retrieved CVE data in a table format, displaying fields like ID, Source Identifier, Published Date, Last Modified Date, and Status.**

**Handles row clicks to pass selected CVE details to the CVEDetails component for detailed view.**

**CVE Details Component (CVEDetails.js):**

**Display Details:**

**Receives selected CVE details as props from CVEList when a row is clicked.**

**Displays specific details of the selected CVE, including Description, CVSS V2 Metrics, and CPE (Common Platform Enumeration) information in a structured format.**

**Features:**

**Pagination:**

**Backend (Server2.js) supports pagination for fetching specific subsets of data based on requested page and limit parameters.**

**Frontend (CVEList.js) manages pagination state and displays data in manageable pages with navigation controls.**

**Duplicate Data Handling:**

**Duplicate check (storeCVEs()) prevents insertion of duplicate CVE records into the database by using the id.**

**Ensures data integrity by avoiding redundant storage of CVE information.**

**Backend Overview:**

**Server 1 (Server1.js):**

**Connects to MongoDB: This server handles data insertion into the MongoDB database.**

**connectToDB() Function: Establishes a connection to MongoDB using Mongoose.**

**storeCVEs() Function:**

**Fetches CVE data from an external API (NIST API) in batches defined by a start index and end index appended to the API URL.**

**Checks if the fetched CVE data already exists in the database to avoid duplicates.**

**Converts date formats from the API into a readable string format.**

**Periodically synchronizes data by running the storeCVEs() function every 24 hours.**

**Server 2 (Server2.js):**

**Retrieves Data from Database: This server handles data retrieval from the MongoDB database.**

**API Endpoint (/api/get-cves):**

**Accepts query parameters (page and limit) to implement pagination.**

**Utilizes Mongoose's paginate function to retrieve the specified number of CVEs based on pagination criteria.**

**Sends paginated CVE data back to the frontend for display.**

**Js (Model.js):**

**To access the backend along with server**