Project Scope: Electric Emergency and Disturbance Dashboard

1. Project Title: Electric Emergency and Disturbance Dashboard Development

2. Project Goal: To develop a user-friendly, modular, and visually informative dashboard application for the Department of Energy to effectively monitor and analyze electric outages and disturbances across the United States. This dashboard will provide key insights into outage causes, impact areas, and severity, enabling proactive identification of vulnerabilities and improved response efforts.

3. Objectives:

- **Develop a robust backend API:** Create a Node.js powered API capable of reading and serving outage data stored in JSON format according to the defined schema.
- **Implement a Swagger UI:** Integrate Swagger UI for clear API documentation and interactive testing.
- **Design and build a modular frontend:** Develop a slick and modular user interface using Vue.js and Bootstrap for displaying outage data.
- **Visualize key outage metrics:** Implement the following dashboard widgets using Graph.js to provide insightful visualizations:
    - A pie chart illustrating the distribution of the **number of customers affected** and **Demand Loss (MW)** categorized by **Event Type**.
    - A pie chart illustrating the distribution of the **number of customers affected** and **Demand Loss (MW)** categorized by **NERC Region**.
    - A chart displaying the **duration of each event** alongside its corresponding **Event Type**.
- **Display top impacted areas:** Create a list displaying the top 10 **Areas Affected** based on either the **number of customers affected** or **Demand Loss (MW)**.
- **Ensure data integrity:** Accurately process and display data according to the provided JSON schema.

4. Scope of Work:

4.1 Backend Development:

- Develop a Node.js API with endpoints to:
    - Read and retrieve outage data from the specified JSON file.
    - Potentially implement filtering and sorting capabilities based on the defined JSON schema fields (e.g., by date, NERC Region, Event Type).
- Implement Swagger UI for API documentation and testing.
- Ensure the backend API can efficiently handle the expected volume of data.

4.2 Frontend Development:

- Develop the user interface using Vue.js for component-based architecture and reactivity.

- Utilize Bootstrap for responsive layout and styling to create a "slick" interface.
- Integrate Graph.js library to create the specified dashboard widgets:
    - Pie chart for customers affected/demand loss by event type.
    - Pie chart for customers affected/demand loss by NERC region.
    - Chart for event duration by event type.
- Develop a list component to display the top 10 affected areas based on user selection (number of customers or demand loss).
- Implement modular design principles to allow for easy addition or modification of dashboard widgets in the future.
- Ensure the frontend can consume data from the backend API.

4.3 Data Handling:

- The application will read data from a static JSON file adhering to the provided schema. *(Note: Integration with external databases or real-time data feeds is outside the current project scope.)*
- Data processing will be performed to aggregate and format data for display in the dashboard widgets.

5. Deliverables:

- A fully functional backend API developed using Node.js.
- Swagger UI documentation for the backend API.
- A fully functional frontend application developed using Vue.js and Bootstrap.
- Implemented dashboard widgets as specified (two pie charts, one duration chart, and one top 10 list).
- A deployed and accessible instance of the dashboard application (specific deployment environment to be determined separately).
- Project documentation including setup instructions and basic user guide.

6. Out of Scope:

- Integration with external data sources or real-time data feeds.
- User authentication and authorization.
- Advanced data filtering, sorting, or searching capabilities beyond basic retrieval.
- Data persistence beyond the initial JSON file.
- Automated testing frameworks (unit, integration, end-to-end).
- Mobile responsiveness beyond basic Bootstrap capabilities.
- Custom map integrations for visualizing affected areas.
- Alerting or notification systems.
- Detailed performance optimization beyond standard coding practices.
- Support for browsers other than the latest versions of Chrome, Firefox, and Edge.

7. Assumptions:

- The provided JSON schema is accurate and will remain consistent throughout the project.

- The JSON data will be readily available for development and testing.
- The project team has the necessary skills and access to development tools and environments.
- Clear communication channels will be maintained between the development team and the Department of Energy stakeholders.

8. Acceptance Criteria:

- The backend API successfully serves data from the JSON file and is documented via Swagger UI.
- The frontend application is deployed and accessible.
- All specified dashboard widgets are implemented and display accurate data based on the JSON file.
- The dashboard interface is functional, user-friendly, and adheres to basic design principles.
- The top 10 affected areas list accurately reflects the data based on the selected metric.

9. Project Timeline: *(To be determined based on resource allocation and complexity analysis.)*

10. Project Team: *(To be defined based on available resources.)*

11. Reporting and Communication: *(To be defined based on stakeholder requirements.)*

This project scope document outlines the boundaries and objectives for the Electric Emergency and Disturbance Dashboard project. Any changes or additions to this scope will require formal review and approval from all relevant stakeholders.