Data Querry Resolution System

**Technology Stack:**

* Backend: Python
* Framework: Django
* Frontend: Vue.js
* Database: e.g., PostgreSQL,
* etc

**Functional Requirements:**

**1. Data Issue Tracking:**

* The system should read data issues from CSV files in a specified directory.
* Each CSV file corresponds to a particular hospital and contains data issues. Each hospital has a unique ID e.g 101, 102, 103, etc
* Each row in a CSV file represents record with a variable which has an issue/error.
* Users should be able to view and respond to data issues.
* Users’ response options should include:
  1. **Entry Corrected**
  2. **Data Matches Source Document**
  3. **Data Already Available**
  4. **No Data Needed**
  5. **Pending(***this is the default option for all issues– and it should be greyed in the system to appear like a placeholder***)**

**2. CSV Upload and Frequency:**

* + The system should automatically fetch CSV files from the specified directory on a weekly basis.
  + The frequency of CSV uploads should be configurable.

**3. Configuration Page:**

* + The system should provide a configuration page where administrators can:
    - Add, edit, or delete paths to the directories where CSV files are stored.
    - Configure the schedule for fetching CSV files from the specified directory.
    - Add hospitals manually or upload a csv file with columns; Hospital Name, Hospital Unique ID, Country, Level(referral hospital, District/county, regional hospital, Health Centre)

**4. Issue Status Summarization/analytics**

* + The system should be capable of summarizing the status of data issues based on the responses.
    - Entry Corrected,
    - Data Matches Source Document,
    - Data Already Available,
    - No Data Needed,
    - Pending
  + Summaries should be presented in the form of graphical charts and downloadable reports.
  + Analytics should show trend over time.
  + Users should be able to filter and customize these summaries.

**5. User Registration:**

* + The system should have a user registration page.
  + Administrators should be able to
* create user accounts(names, username, email, password),
* assign roles to system users, and
* assign hospital facility each user is associated with.

*NB: Make hospital list a* ***checkbox*** *because it is possible for user to be assigned to more than one facility especially for data managers.*

* + The system should have functionality to recover the forgotten passwords.

**6. User Access Control:**

* + Users, except for administrators, should have limited access.
  + Users should only be able to access and manage data issues related to the facility they are assigned to.
  + User roles should include "Administrator" and "Regular User."

**7. Notifications:**

* + - The system should notify user(s) when new data issues for the hospital(s) they are affiliated to have been added to the system.
    - Users should receive notifications through their registered email and in-app notifications.

**8. Reminders:**

* + - The system should send reminders to user(s) about pending data issues for their facility.
    - Reminders should be sent at configurable intervals (e.g., daily, weekly).

**9. Logging and Auditing:**

* + - Implement logging and auditing functionalities to track user actions and system events for auditing and troubleshooting.

**10. Deployment:**

* + - I will provide server IP address where we will deploy.