Project SPEC

General Description of the Project
1. **Motivation & Problem Context**
   - Air pollution is a major health concern in Indian cities. Residents often lack access to reliable, localized AQI forecasts. Without predictions, it's difficult to plan preventive measures or raise awareness. My project, ClearSkiesAQI, focuses on forecasting AQI to provide valuable insights into air quality dynamics and can inform future air quality management strategies. The core idea is to use historical and real-time air quality data to predict AQI values. Identify the pollutants that impact AQI the most. Generate daily reports with forecasts and summaries of pollutant contributions.
2. **External Mechanisms & Tools**
   - APIs: Publicly available APIs for air quality data.
   - Python Libraries:
     - pandas → data collection & cleaning
     - scikit-learn → forecasting models
   - Reports can be saved/exported in a format understandable to non-technical users.
3. **Interface & Usability**
   - Basic Version:
     - User specifies city and timeframe.
     - System outputs forecasted AQI and a daily text report.

Task Vignettes
User Task 1: Input a city name and get the AQI forecast for that city

User input (city name):
Step 1:Seema is concerned about the air quality in Pune. She opens the **ClearSkiesAQI app** on her laptop.
- The home screen shows a simple interface:
  - A **search bar** with the placeholder text *"Enter city name..."*
  - A **dropdown menu** for quick access to popular cities (Pune, Delhi, Mumbai, Bangalore, etc.)
  - A large **"Get Forecast"** button below.
Step 2: Seema clicks inside the search bar, types *"Pune"*, and clicks on **Get Forecast**.

Response:
The screen briefly shows a loader.
Then, results are displayed :
- **City**: Pune
- **Date**: Sept 22, 2025
- **Forecasted AQI**: 178 (**Unhealthy**)
The card is color-coded (red for "Unhealthy") so seema can instantly recognize the severity.

Overall Flow of Data with Technical Details
1. User Input:
   - User enters a city name (in CLI or GUI).
   - Input is captured and passed into the data collection module.
2. Data Collection:

- The system queries a public API to fetch historical and/or real-time pollution data.
- Data is returned as JSON/CSV stream.

3. Data Processing & Cleaning:
   - Handle missing values, normalize pollutant units, and filter by city/date.
   - Extract relevant pollutants.
   - Store in dataframe (pandas) for processing.

4. Forecasting Module:
   - Apply a ML model on cleaned data.
   - Output is a forecasted AQI values (dictionary or dataframe).

5. Report Generation:
   - Build a report with forecasted AQI, pollutants, and health recommendations.
   - Output in plain text (CLI) or formatted card/chart (GUI).
   - Optionally export as .txt/.pdf/.csv.

6. Visualization (optional in v1, expanded in v2):
   - Generate AQI trend plots (using matplotlib).
   - Display inline (GUI) or pop up as chart window.

7. User Output:
   - Minimal version: CLI prints forecast + saves optional text file.
   - Ideal version: GUI displays results card + charts + buttons for saving/exporting.