Test Strategy Document

Rishabh Daga - 2018101015
Outdoor Air Pollution Monitoring in IIIT-H campus
Team-30

Scope

Our client (Mr.Sachin Choudhari,Researcher-Professor-SPCRC Lab at IIIT Hyderabad) and our corresponding TA mentor (Eesha) will review this document and will suggest accordingly for suggestions.
After suggestions being implemented, this will be approved by the same.

Following testing activities are planned:

- First testing would involve the checking of raw scripts and individual software pieces used to generate graphs and other visual information to be rendered on frontend.
- Checking the functionalities of basic UI and play around with all the implementations for ensuring their correct and smooth functioning.
- Third testing would be on the inferences that we draw or the conclusions that we draw out of the existing data.(Mainly done by the client).

Test Approach

Process of testing:

1. Setting up the test for a particular piece of an application (called the system under test)
2. Performing the actual testing (interacting with the system under test)
3. Observing the resulting behavior and checking whether expectations were met.(which is sometimes manual while other times we will be using scripts for observations as well like differences and all)

Testing Levels:

Till the end of R1 we have tested our front end and UI design implementation .Also we have checked our visuals to be rendered and their scripting for test data.Also we have tested on the data cleaning script.

Roles and Responsibilities of each team-mate:

1. Monil - Back-end Developer
2. Anubhav - Front-end Developer and integration of stack
3. Rishabh - UI designer and Front-end developer
4. Shiva - Data management and cleaning

Testing Tools and Environment

- Automation and Test management tools needed for test execution
- Figure out the number of open-source as well as commercial tools required, and determine how many users are supported on it and plan accordingly testing Tool.

Testing tools:

- Postman is a powerful tool for performing integration testing with your API. It allows for repeatable, reliable tests that can be automated and used in a variety of environments and includes useful tools for persisting data and simulating how a user might actually be interacting with the system.
- Self made scripts(.py files) to check for the steep and incorrect data changes.
- Java Script console log for checking the outputs of the js scripts written.
- Sql Alchemy
- Django Framework.

Testing Environment:

1. **Basic Inclusions(basic package)**
   a. Version control system -Gitlab
   b. Text editors-Atom , Vscode
   c. Operating System- Windows/Linux
   d. Web Browser-Firefox
   e. Networking : Basic Local host (port 3000 for frontend, port 5000 for backend)
2. **Network:**
   a. Internet setup
   b. Well functioning server to host with optimal permissions

Use Cases

- Live checking - Checking individual levels of pollutants(live data given by the pre-installed sensors, through a web portal,without any authentication)

- Hourly Tracking - Checking pollutant levels on hourly basis where we will calculate the average of data per time unit.
- Daily Tracking - Checking pollutant levels on a daily basis where we will calculate the average of data per time unit.
- Inter-day Tracking - Individual pollutant mean over an inter-day cycle where the averages are collected over multiple days (a week or month)
- Overall Air Quality - Checking air quality (with existing parameters) and by using certain statistical & mathematical concepts.
- View Heat Map - Viewing the heat map of campus in accordance to the air quality where pollution level increase from blue to red.
- View Line Graph - View live line graph of real time data of pollutant levels.
- View Histogram - Display the data in a more informative way using overlapping histograms showing individual rise and fall of levels on a same place.

Test Cases

- Check if the UI is working properly.
- Check if the data is correct or not.
- Check sample graphs which are plotted.
- Check scripts coded for Line graphs.
- Check scripts coded for Histograms.
- Check the storage of static data stored in SQL databases at the backend.
- Check the functionality of backend.
- Check if the interpolation done is correct or not.
- Check if the conclusions drawn are correct or not.