**Water Leakage Detection App**

**Udit Sarkar, Jan 2025**

**Project Description**

This is a leak detection web app. This reads data of multiple sensors that are fitted on a pipeline through Lorawan server, evaluates the data and displays whether a leakage is created or not on the UI.

**Admin and User Interfaces**

This app has 2 interfaces, one for admin , another for user.

The following functionalities are for Admin interface:

1. **Add/Delete/Update Sensor**: Admin can add a sensor by entering the sensorId, sensor location, and the sensor type. These are also editable. Admin also can delete a sensor.
2. **List of Sensors**: There is a table listing all the added sensors.
3. **Hard Reset:** Admin can delete all the sensor and their related datas.
4. **Soft Reset:**  This button is to remove all the history datas for a sensor.
5. **Interactable Map:** There is a map where all the sensors are shown, the sensor marker will be green when there is no leakage detection by the sensor and red for leakage detection by the sensor.
6. **Live Leak List:** This is a table that will list all the sensors that are detecting leakage in the pipeline, along with their type, time of leakage detection, distance of the leakage from the sensor. Once the leakage is fixed the sensor from the list will be removed.
7. **History Of Leakage Data:** This is a table that shows the history of leakage detection. There is a toggle button to show and hide it. Also this has a filter functionality to filter the data according to date , and download the data as csv file.
8. **Data Table:** On clicking over the marker of a particular sensor this will display all the history data of that sensor. On top of it SensorId and sensor type will also be displayed. The data in the table contains Time, Fcnt, SF and the encrypted data. Also there is a filtering option according to data and time and a download as csv button.

The following are the functionalities for User Interface:

1. **List of Sensors**
2. **Interactable Map**
3. **Live Leak List**
4. **History Of Leakage Data**
5. **Data Table**Now you can work accordingly.

**Technicalities**

**MongoDB Database**

The structure of the database used is:

sensorDB

sensormetadatas: sensorID,type,location(latitude,longitude) for each sensor

leakhistories: sensorID,type,time,distanceshistory for each leakage detection

sensor1: All datas received from the lorawan server for each data received for sensor1

sensor2

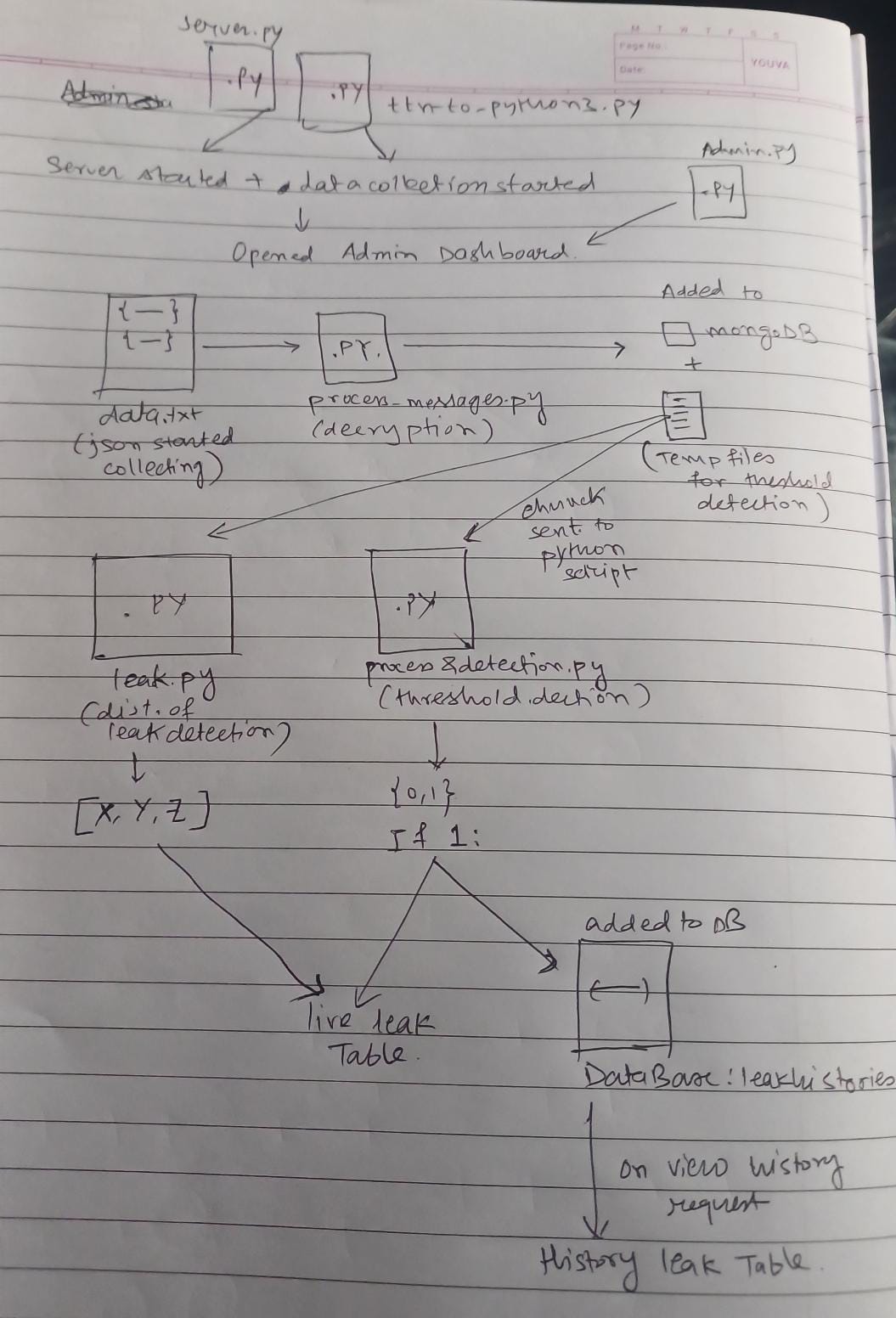
.

.

.

1. **Adding/Updating/Deleting a sensor:** When a sensor is added, a new document gets created in the sensormetadata collection, and a new collection with the sensorID is created.
2. **Hard Reset:** On hard reset, all the sensors collection and all the documents in sensorsmetadata gets deleted.
3. **Soft Reset:** On soft reset, all the documents inside that particular sensor collection gets deleted.
4. **Threshold Detection:** Chunk of data is passed to the threshold detection python script that return 0 or 1 , whether threshold is reached or not.
5. **Distances Determination:** The same chunk on which detection is passed to a distance determination python code that return the distance.
6. **History of leakage Data:** Whenever leakage is detected , a document is added to the leakhistories collection, and the table displays the 6 recent leakages.
7. **Sensor Data Decryption:** The Lorawan server sends encrypted data which is decrypted by a python code.

**Total work flow:**

****

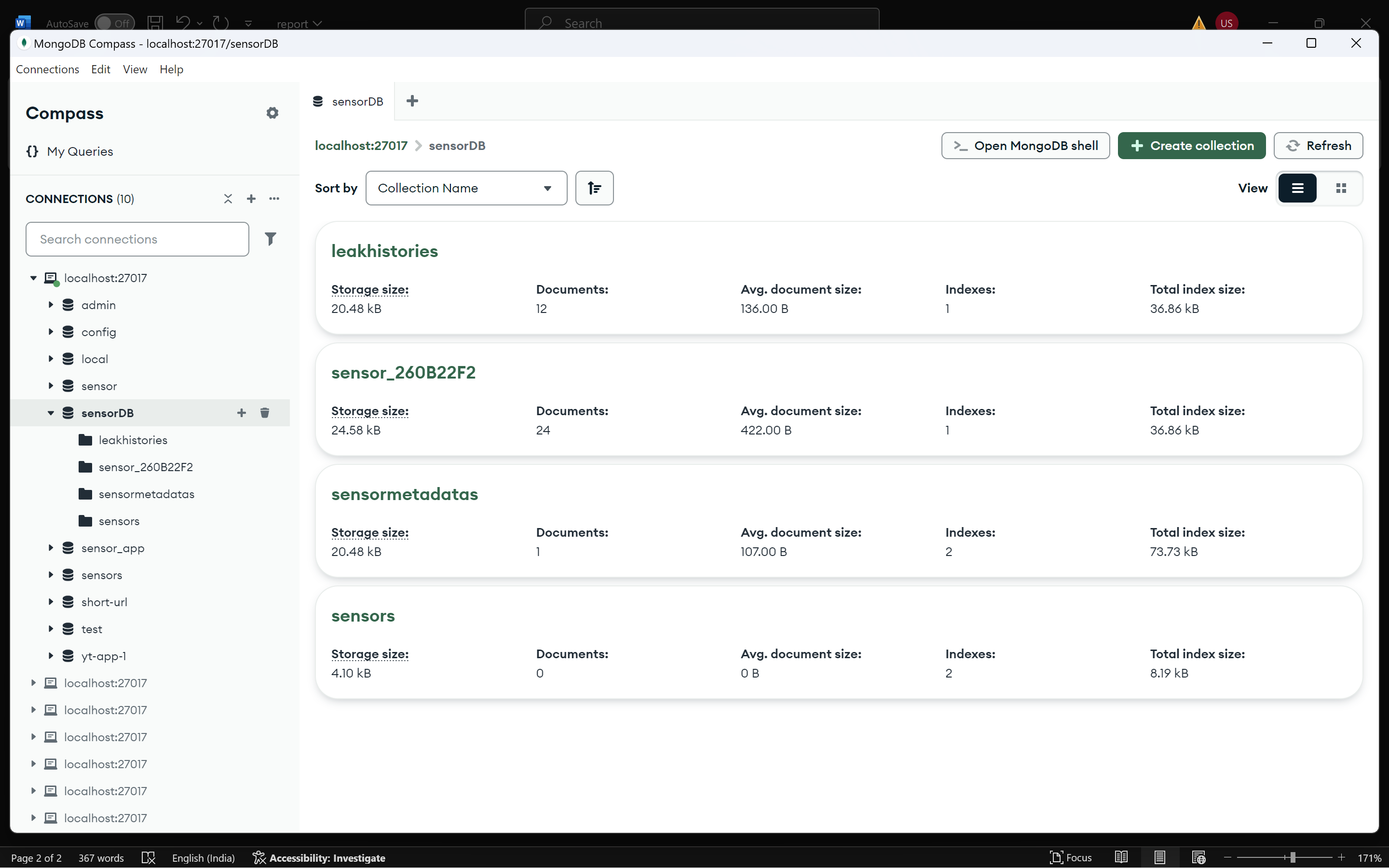
Pre requisites:

1. Install nojejs 10.8.1 - https://www.geeksforgeeks.org/install-node-js-on-windows/
2. Install mongodb 7.0.12 - <https://www.geeksforgeeks.org/install-mongodb-compass-on-windows/>
3. Create a Database in mongodb named ‘sensorDB’ - https://www.geeksforgeeks.org/create-database-using-mongodb-compass/
4. Open mongodb compass app, on the left bar click on connect to localhost:27017 then Create 2 collections in the sensorDB database named ‘leakhistories’ and ‘sensormetadatas’- <https://www.mongodb.com/docs/compass/current/collections/>
5. Install python 3.12.1 - <https://www.python.org/downloads/>
6. Import these packages using pip install command – os,time,pandas,numpy,openpyxl,base64,json,time,subprocess,datetime, paho-mqtt
7. In the directory run the command ‘npm init’
8. Install all the dependencies by running this command – ‘npm install body-parser@1.20.3 canvas@2.11.2 chokidar@4.0.1 express@4.21.2 jsdom@25.0.1 mongodb@6.12.0 mongoose@8.8.4 node-cache@5.1.2 nodemon@3.1.9 path@0.12.7 plotly@1.0.6 [socket.io@4.8.1](mailto:socket.io@4.8.1) ejs’

**Setup**:

1. Double click launch.py (start server+data collection)
2. Double click admin dashboard for admin interface and user dashboard for user interface(to open the web app)

**To start server+data collection + open web app at once – double click admin.py or user.py**

1. To change the chunck size for threshold detection - open app.js, line 213
2. To change the thresholds – open 220,278 line in processing&detection.py & 220,302 lines in leak\_x.py
3. To checkout the database -  click on the mongodb compass app downloaded, then on the left bar, connect to the localhost:27017 , then the sensorDB database will be visible.