# COMPUTER ENGINEERING WORKSHOP

**S.E. (CIS) OEL REPORT**

**Project Group ID:**


| | |
|---|---|
| Rida Noor Qasmi | CS-04 |
| Arfa Tariq | CS-13 |
| Hafiz Ashan Baig | CS-43 |


**BATCH:** 2023


**Department of Computer and Information Systems Engineering**

**NED University of Engg. & Tech.,**
**Karachi-75270**

## CONTENTS

**CHAPTER 1**

# PROBLEM DESCRIPTION

Construct an integrated environmental monitoring system in C, covering a range of fundamental concepts and practical applications. The project involves interacting with a free API that provides real-time environmental data. The system's core functionalities include data retrieval, processing, and reporting.

- Interact with a free API to retrieve real-time environmental data (e.g., temperature, humidity).
- Store raw and processed data in files.
- Create shell scripts to automate tasks such as data retrieval and processing.
- Utilize the C program's pointers and dynamic memory allocation to optimize data manipulation and enhance efficiency.
- Implement real-time alerts using Linux system calls to notify relevant personnel of critical environmental readings.
- Use header files to modularize the C code and enhance code readability.

**CHAPTER 2**

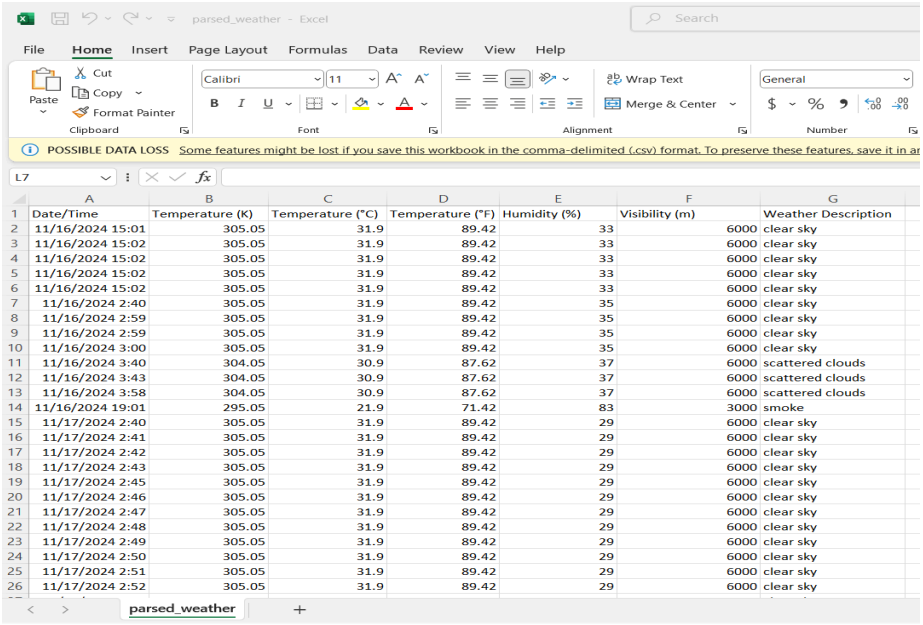
# METHODOLOGY

**1. Fetching Weather Data**:

The process begins with a **Bash script** that calls the `weather_fetch.sh` script. This script fetches weather data from the **OpenWeather API** using a specified API key. The fetched data is saved in a JSON file named **weather_data.json**. The API request retrieves weather details for Karachi, including temperature, humidity, and visibility.


**2. Parsing JSON to CSV**:

After the weather data is fetched, the script calls the **weather_monitor** C program, which parses the `weather_data.json` file. The C program extracts relevant weather parameters—**temperature**, **humidity**, and **visibility**—and saves them along with **timestamps** into a CSV file named **parsed_weather.csv**.



parsed_weather - Excel

Search

File Home Insert Page Layout Formulas Data Review View Help

Cut Copy Format Painter Paste Clipboard

Calibri 11 Font

Wrap Text Merge & Center Alignment

General Number

POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an

L7

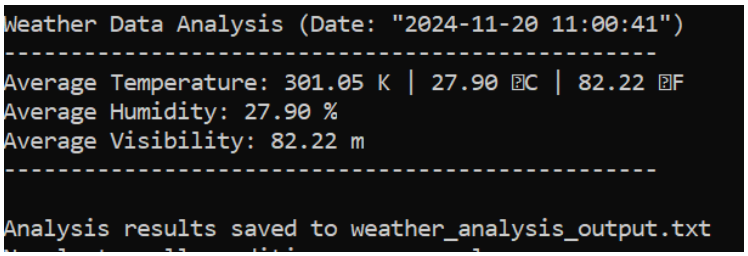
| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | Date/Time | Temperature (K) | Temperature (°C) | Temperature (°F) | Humidity (%) | Visibility (m) | Weather Description |
| 2 | 11/16/2024 15:01 | 305.05 | 31.9 | 89.42 | 33 | 6000 | clear sky |
| 3 | 11/16/2024 15:02 | 305.05 | 31.9 | 89.42 | 33 | 6000 | clear sky |
| 4 | 11/16/2024 15:02 | 305.05 | 31.9 | 89.42 | 33 | 6000 | clear sky |
| 5 | 11/16/2024 15:02 | 305.05 | 31.9 | 89.42 | 33 | 6000 | clear sky |
| 6 | 11/16/2024 15:02 | 305.05 | 31.9 | 89.42 | 33 | 6000 | clear sky |
| 7 | 11/16/2024 2:40 | 305.05 | 31.9 | 89.42 | 35 | 6000 | clear sky |
| 8 | 11/16/2024 2:59 | 305.05 | 31.9 | 89.42 | 35 | 6000 | clear sky |
| 9 | 11/16/2024 2:59 | 305.05 | 31.9 | 89.42 | 35 | 6000 | clear sky |
| 10 | 11/16/2024 3:00 | 305.05 | 31.9 | 89.42 | 35 | 6000 | clear sky |
| 11 | 11/16/2024 3:40 | 304.05 | 30.9 | 87.62 | 37 | 6000 | scattered clouds |
| 12 | 11/16/2024 3:43 | 304.05 | 30.9 | 87.62 | 37 | 6000 | scattered clouds |
| 13 | 11/16/2024 3:58 | 304.05 | 30.9 | 87.62 | 37 | 6000 | scattered clouds |
| 14 | 11/16/2024 19:01 | 295.05 | 21.9 | 71.42 | 83 | 3000 | smoke |
| 15 | 11/17/2024 2:40 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 16 | 11/17/2024 2:41 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 17 | 11/17/2024 2:42 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 18 | 11/17/2024 2:43 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 19 | 11/17/2024 2:45 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 20 | 11/17/2024 2:46 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 21 | 11/17/2024 2:47 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 22 | 11/17/2024 2:48 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 23 | 11/17/2024 2:49 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 24 | 11/17/2024 2:50 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 25 | 11/17/2024 2:51 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |
| 26 | 11/17/2024 2:52 | 305.05 | 31.9 | 89.42 | 29 | 6000 | clear sky |

parsed_weather

**parsed_weather.csv**

### 3. Calculating Averages:

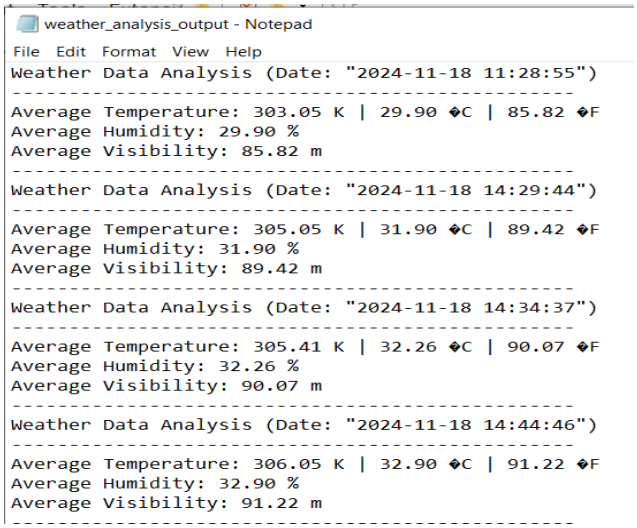
Once the data is parsed, the **weather_analysis** C program is executed through the `calc_avg.sh` script. This program calculates the average temperature, humidity, and visibility for the current day's weather data. The averages are then displayed on the terminal and saved in a text file called **weather_analysis_output.txt**.



```
Weather Data Analysis (Date: "2024-11-20 11:00:41")
-------------------------------------------------
Average Temperature: 301.05 K | 27.90 ?C | 82.22 ?F
Average Humidity: 27.90 %
Average Visibility: 82.22 m
-------------------------------------------------

Analysis results saved to weather_analysis_output.txt
```

**Terminal Output**




**weather_analysis_output.txt**

### 4. Weather Alerts and Automation:

The `calc_avg.sh` script also includes an alert mechanism, which checks if any weather parameters exceed predefined thresholds (e.g., temperature > 30°C, humidity > 70%, visibility < 25 meters). If any condition is met, the system sends an email alert to the designated recipient. The script uses **msmtp** to send the email and ensures the alerts are dispatched automatically.


**Email Alert**

### 5. Scheduling

The scheduling of the weather data collection and processing is handled by the `scheduler2.sh` script. It orchestrates the execution of the weather fetching, parsing, and average calculation processes. It runs the `fetch_and_parse.sh` script to collect and parse weather data, then calls the `calc_avg.sh` script to compute averages.

The entire process is executed in a scheduled manner with a fixed number of iterations, and the results are logged to track the execution status and any errors. This setup ensures that the weather data is fetched, processed, and analyzed at specified intervals, automating the entire workflow.

**CHAPTER 3**

# RESULT

The weather alert system successfully automated the process of fetching, parsing, analyzing, and alerting based on weather data.

1. **Weather Data Fetching**:
   The `weather_fetch.sh` script fetched weather data from the OpenWeather API for Karachi, saving the data in `weather_data.json`.
2. **Data Parsing**:
   The `weather_monitor` C program correctly parsed the data, extracting temperature, humidity, and visibility, and storing it in `parsed_weather.csv`.
3. **Average Calculation**:
   The `weather_analysis` program calculated daily averages for temperature, humidity, and visibility, saving the results to `weather_analysis_output.txt`.
4. **Weather Alerts**:
   The `calc_avg.sh` script detected conditions exceeding set thresholds (e.g., temperature > 30°C, humidity > 70%) and successfully sent email alerts via **msmtp**.
5. **Scheduling and Automation**:
   The `scheduler2.sh` script scheduled and automated the entire process, running the fetch, parse, and analysis steps 13 times, with logs capturing execution details.

The program operated as expected, providing timely weather alerts based on real-time data and automation according to the Karachi/Asia forecast.

Here's the final output from the main bash file (`scheduler2.sh`) that provides an average analysis of the current day's weather and execution status of email alerts.