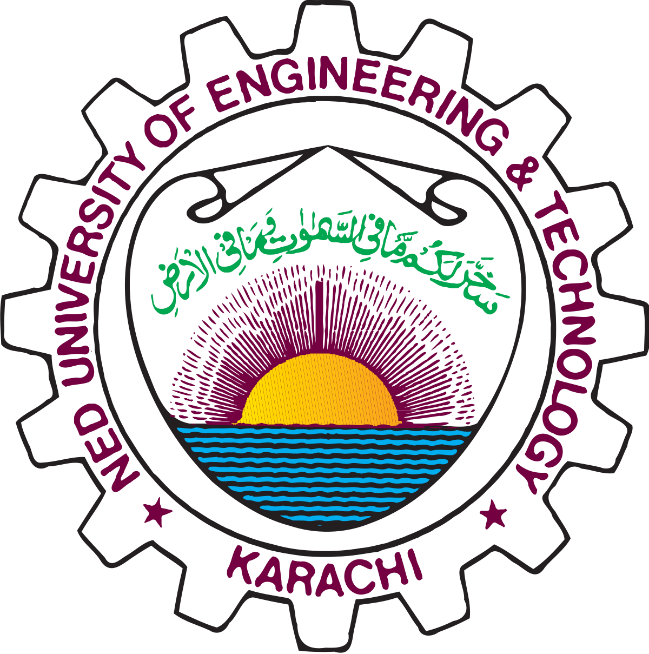
# COMPUTER ENGINEERING WORKSHOP

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

|  |  |
| --- | --- |
| NAME OF MEMBER #1 : SHAYAN HUSSAIN | CS-22100 |
| NAME OF MEMBER #2 : ALI HYDER | CS-22093 |
|  |  |

**BATCH:** 2022

**Jan** 2024



**Department of Computer and Information Systems Engineering**

**NED University of Engg. & Tech.,**

**Karachi-75270**

**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, analysis, and reporting.

**Problem Outline:**

**Data Retrieval:**

* Interact with a free API to retrieve real-time environmental data (e.g., temperature, humidity).

**Data Processing and Analysis:**

* Process the retrieved data using C programming concepts.
* Implement algorithms to analyze environmental data for anomalies or trends.

**Data Storage:**

* Store both raw and processed environmental data in files.

**Reporting System:**

* Develop a reporting system that generates reports based on the analyzed environmental data.

**Automation and Integration:**

* Create shell scripts to automate tasks such as data retrieval, processing, and report generation.
* Integrate shell scripts with the C program to enhance automation.

**Optimization and Efficiency:**

* Utilize pointers and dynamic memory allocation in the C program to optimize data manipulation and enhance efficiency.

**Real-time Alerts:**

* Implement real-time alerts using Linux system calls to notify relevant personnel of critical environmental readings.

**Documentation and Code Organization:**

* Develop comprehensive documentation for the monitoring system.
* Use header files to modularize the C code and enhance code readability.

**CHAPTER 2**

**METHODOLOGY**

Following is the methodology of the C code of the weather forecast generating system. The provided C code is designed to retrieve weather data from a specified API, process the data, generate an environmental report, and send the results via email. Here's a summary of the key components and functionalities:

**“Main.c”**

1. **Libraries:**

* Standard C libraries (stdio.h, stdlib.h, string.h, time.h) for I/O, memory allocation, string manipulation, and time functions.
* libcurl (curl/curl.h) for making HTTP requests.

1. **Variables:**

* Arrays and variables to store weather-related data.
* File pointers for input and output files.

1. **CURL Setup and API Request:**

* Initializes libcurl, sets up URL and headers for the API request.
* Performs the HTTP request and saves the API response to a file.

1. **File Handling:**

* Opens files for reading and writing, handles errors during file operations.
* Copies content between files, closes files.

1. **Data Processing:**

* Reads API response from the file.
* Calls **processData** function to extract and analyze weather data, detecting anomalies (e.g., high temperature, precipitation).

1. **Report Generation:**

* Generates an environmental report with a timestamp, basic environmental data, and detected anomalies.
* Writes the report to a file.

1. **Email Sending:**

* Calls **send\_email\_with\_attachment** function to send an email.
* Uses libcurl to attach the processed data file and sends the email.

1. **Cleanup:**

* Frees allocated memory.
* Closes files.

**“report\_generator.c”**

1. **Libraries:**

* Standard C library (stdio.h).

1. **Function Definitions:**

* generateReport: Creates an environmental report based on input parameters, including timestamp, environmental data, and anomalies.

**“email\_sender.c”**

1. **Libraries:**

* Standard C library (**stdio.h**).
* libcurl (**curl/curl.h**).

1. **Function Definitions:**

* **read\_callback:** Callback function for libcurl to read file content.
* **send\_email\_with\_attachment:** Sends an email with an attached file using libcurl.

**Memory Allocation:**

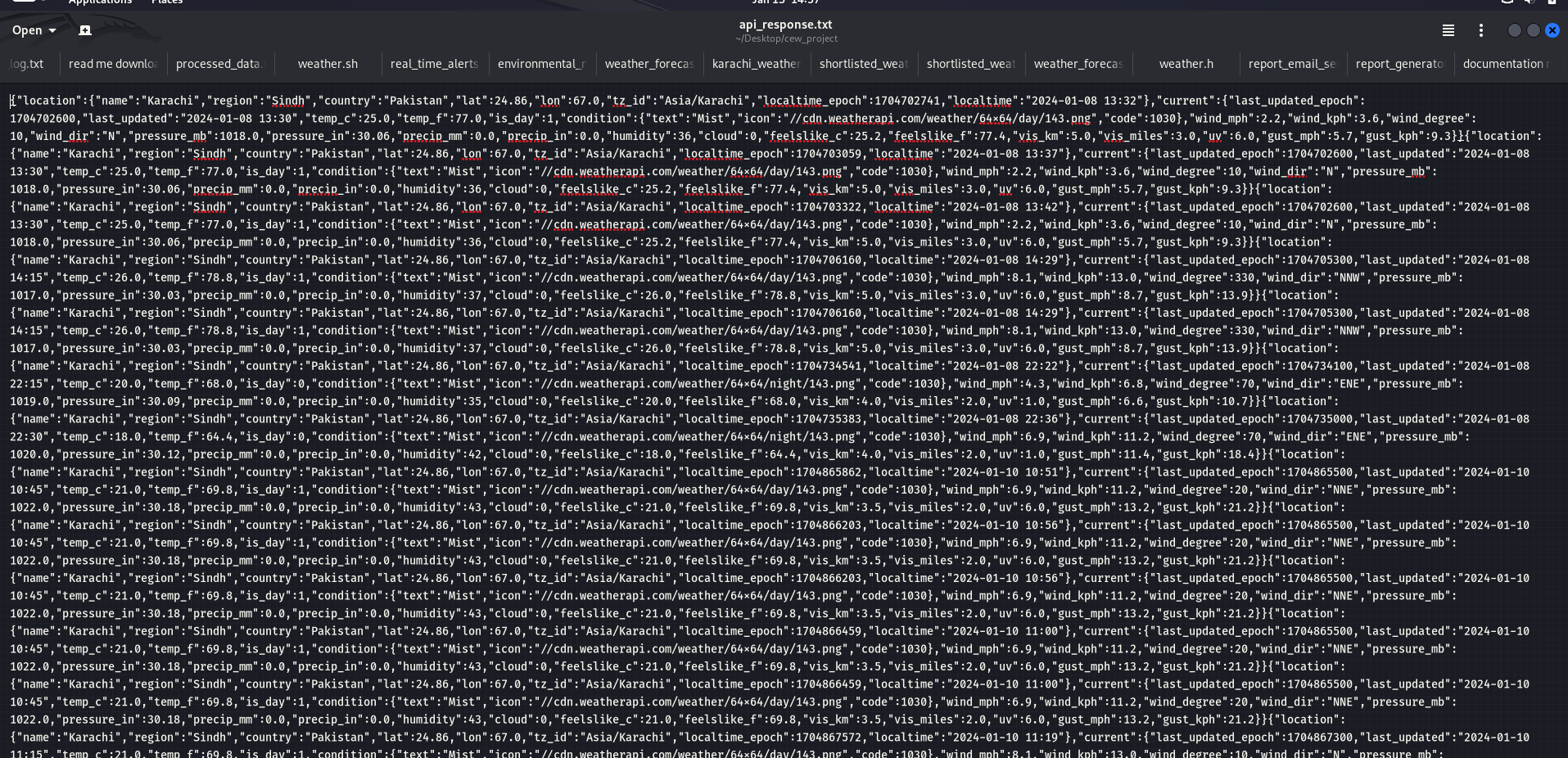
* Uses **malloc** for dynamic memory allocation to store the API response content.

**Overall Flow:**

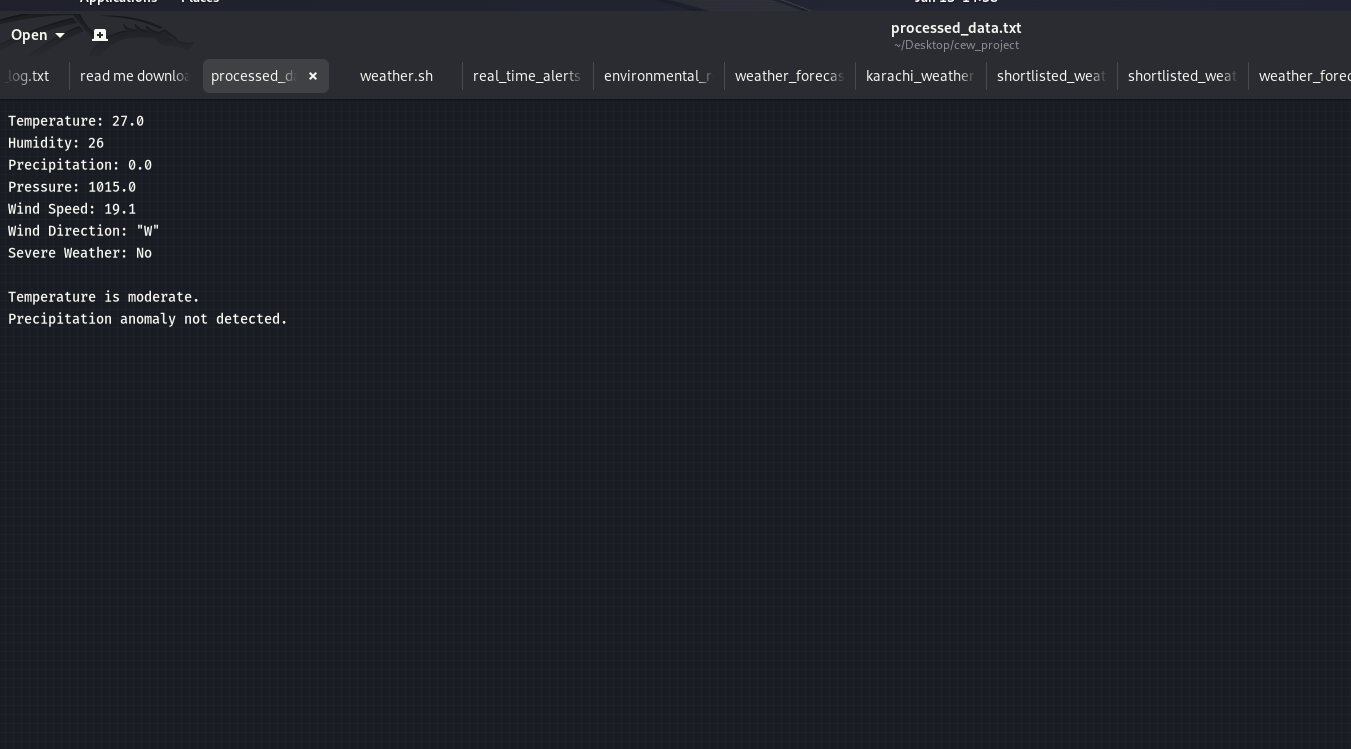
* Makes an API request to retrieve weather data.
* Saves the API response to a file (output\_log.txt).
* Processes the data, detecting anomalies.
* Generates an environmental report (environmental\_report.txt).
* Sends an email with the processed data attached.
* Cleans up by freeing allocated memory and closing files.

**Test Runs:**

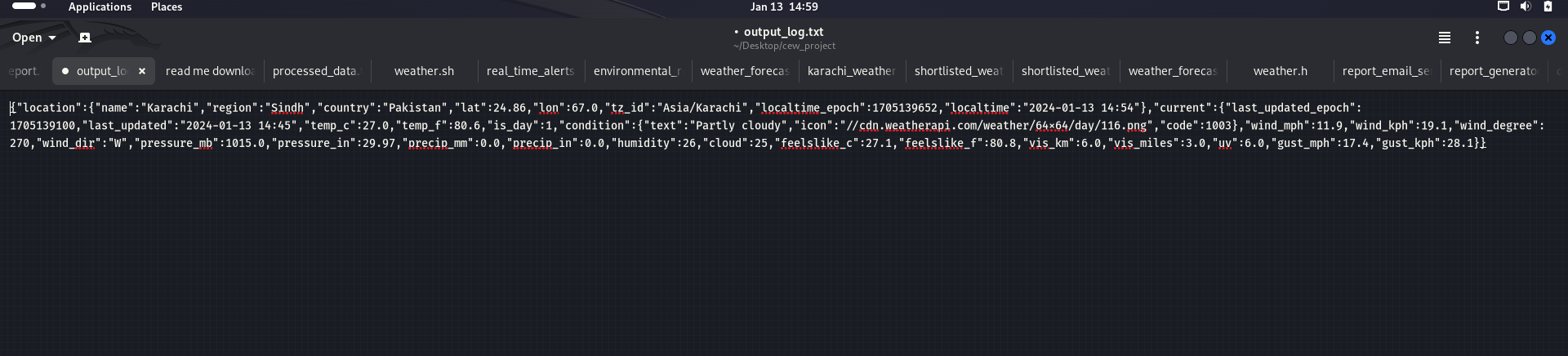
**API RESPONSE RAW DATA:**



**PROCESSED DATA:**



**RAW OUTPUT:**



**EMAIL SENDING:**

