COMPUTER ENGINEERING WORKSHOP

(CEW)

S.E. (CIS) PROJECT REPORT

MEMBERS:

MUHAMMAD OWAIS ALAM (CS-22070)

MUHAMMAD ROHAN KHAN (CS-22072)

MUHAMMAD MUDASIR SHAIKH (CS-22135)

SUBMITTED TO:

MS. MAHNOOR MALIK

BATCH 2022

Integrated Environmental Monitoring System

ABSTRACT:

Our project, the "Integrated Environmental Monitoring System," built in C on Linux, addresses the need for real-time environmental data analysis. Using C programming, it interacts with a free API, fetching and processing data to spot anomalies. The system stores data efficiently and generates reports. To boost efficiency, we use techniques like dynamic memory allocation and pointers. Clear documentation and organized code are priorities for readability. This integrated solution offers a strong foundation for environmental monitoring, blending modern computer technologies with practical applications, focusing on core functions like anomaly detection through optimized techniques.

API USED:

We have use a free API to retrieve raw data by making HTTP request.API used is:
OPEN WEATHER MAP.

LIBRARY USED:

curl/curl.h: This library is used for making HTTP requests and interacting with web services. It is employed to fetch weather data from the OpenWeatherMap API.

jansson.h: Jansson is a C library for encoding, decoding, and manipulating JSON data. In this program, it is used to parse the JSON response obtained from the OpenWeatherMap API.

stdio.h, stdlib.h, string.h, time.h: Standard C libraries for input/output, memory allocation, string manipulation, and time-related functions.

FILE DESCRIPTION:

1. codeagain.c
This file contains the main program logic. It includes functions to fetch weather data, parse JSON, save data to files, and send emails.

2. email_sender.c
This file defines the send_email_with_attachment function used for sending emails with attachments. It includes a read callback function for libcurl to read data from a file.

3. email_sender.h
This header file contains the function declaration for send_email_with_attachment. It is included in both codeagain.c and email_sender.c to ensure proper function declaration.

DATA HANDLING:

The program fetches weather data in JSON format from the OpenWeatherMap API.
Jansson library is used to parse the JSON response and extract relevant information.
Processed data is formatted and saved to a text file (processed_weather_report.txt).
Raw data is saved to a text file (raw_data.txt) along with a timestamp.

COMMON FUNCTION USED:

Following are the function we used throughout the program:

1) send_email_with_attachment: Sends an email with an attachment using libcurl.
2) save_processed_data_to_file: Saves processed data to a file with overwriting capability.
3) save_raw_data_to_file_with_timestamp: Saves raw data to a file with a timestamp in append mode.
4) write_callback: Callback function for libcurl to handle received data during an HTTP request
5) get_current_time: Returns the current time as a formatted string (HH:MM:SS).
6) get_formatted_time: Returns the formatted time from a timestamp.
7) print_weather_info: Prints weather information in a formatted way.
8) main (in codeagain.c): Main function orchestrating the program execution, fetching weather data, parsing JSON, saving data, and sending emails.
9) read_callback (in email_sender.c): Callback function for libcurl to read data from a file for email attachment.

EMAIL SENDING FUNCTIONALITY:

The send_email_with_attachment function uses libcurl to send emails.
It takes recipient email, cc email, and file path as parameters.
The email server details, authentication, and recipient information are set using libcurl options.
The file content is read using a callback function and sent as an attachment.

FUTURE EXPANSIONS:

As we look to the future, several potential expansions can further enhance the capabilities of the Integrated Environmental Monitoring System. Integration with additional environmental sensors and data sources can broaden the scope of monitored parameters. Implementing machine learning algorithms for anomaly detection could increase the system's accuracy and predictive capabilities. Enhanced visualization tools for data analysis and reporting can provide more insightful and user-friendly presentations of environmental information. Moreover, extending the system to support real-time alerts and notifications for unusual environmental events would contribute to proactive monitoring. These expansions aim to keep the system adaptable and responsive to evolving environmental monitoring needs.

CONCLUSION:

In conclusion, the "*Integrated Environmental Monitoring System*" successfully addresses the critical need for real-time environmental data analysis and reporting. Leveraging the capabilities of the C programming language on the Linux platform, the system efficiently interacts with a free API, processes real-time and raw data, and identifies anomalies. The incorporation of optimization techniques, such as dynamic memory allocation, structure utilization, and pointer management, enhances the system's overall efficiency. The importance of clear documentation and organized code for readability and maintenance has been emphasized throughout the project. This integrated solution not only showcases the synergy of contemporary technologies in computer engineering but also provides a practical application in the field of environmental monitoring.