COMPUTER ENGINEERING WORKSHOP

S.E. (CIS) OEL REPORT

Project Group ID:

AREEBA KHAN	**CS-110**
ZARA AKRAM	**CS-104**
JAVERIA RIZWAN	**CS-117**

BATCH: 2023

CONTENTS

Contents

PROBLEM DESCRIPTION

Weather Monitoring System Using API Integration

This project focuses on creating an automated weather monitoring system using C programming and Linux tools to track temperature and humidity. Monitoring these factors is crucial for industries like agriculture, healthcare, and disaster management because changes in these factors can affect crop health, public safety, and climate preparedness. Many weather systems today rely on manual data collection, which slows down responses and reduces efficiency. This system aims to automate the process by using APIs to fetch real-time data, process it, and send alerts when necessary.

OBJECTIVES

Collect Real-Time Data: Use the OpenWeather API to fetch weather data and log the results for reference.
Process the Data: Convert the raw data into a readable format, focusing on temperature and humidity.
Generate Alerts: Set up alerts for critical conditions based on specific thresholds.
Automate the Workflow: Use Linux automation tools to keep the system running continuously.
The system offers a reliable foundation for addressing challenges in environmental monitoring.

METHODOLOGY

The system employs a modular design for efficiency and scalability. The methodology includes:

1. **Data Fetching**:
    - Used the **OpenWeather** API with **libcurl** for HTTP GET requests.
    - Stored raw JSON responses in **weather_data.txt** for validation.

2. **Data Processing**:
    - Parsed JSON data using **cJSON** to extract key metrics:
        - **Temperature** (converted to Celsius).
        - **Humidity** (percentage).
    - Saved processed data and analyzed recent readings to compute averages in **processed_sensor_data.txt** with timestamps for analysis.

3. **Critical Alerts**:
    - Monitored parameters using thresholds (e.g., >30°C or >80% humidity).
    - Logged alerts in **logs/alerts.log** and displayed real-time notifications.

4. **Automation**:
    - Configured Linux to run the system and fetch data after specified time.
    - Used shell scripts for seamless compilation and execution.

RESULTS

1. **Raw Data Collection**:
    - Captured weather information in **weather_data.txt**, including API responses.
2. **Processed Data**:
    - Extracted key metrics and calculated averages for recent readings stored them in **processed_sensor_data.txt.**
    - Example entry:

Temperature: 6.00°C, Humidity: 65.00%

3. **Critical Alerts**:
    - Logged critical events with timestamps in **alerts.log.**
    - Example alert:

Critical Alert! Check environment immediately.

4. **Automation**:
    - Successfully executed automated workflows ensuring consistent operation.

CONCLUSION AND FUTURE WORK:

This project demonstrates the effective use of programming and automation to develop a real-time weather monitoring solution. Key achievements include:

- Efficient data collection and processing.
- Reliable detection of critical conditions.
- Automated system operation for consistent monitoring.

FUTURE ENHANCEMENTS:

1. Allow dynamic configuration of thresholds.
2. Expand metrics to include wind speed and air pressure.
3. Integrate data visualization for trend analysis.
4. Enable multi-location monitoring capabilities.

This project lays the groundwork for further innovation in environmental monitoring systems.

DEPARTMENT OF COMPUTER & INFORMATION SYSTEMS ENGINEERING

BACHELORS IN COMPUTER SYSTEMS ENGINEERING

Course Code: CS-219

Course Title: Computer Engineering Workshop

Open Ended Lab

SE Batch 2023, Fall Semester 2024

Grading Rubric

TERM PROJECT Group

Members:

Student No.	Name	Roll No.
S1	**Javeria Rizwan**	**CS23117**
S2	**Areeba Khan**	**CS23110**
S3	**Zara Akram**	**CS23104**

CRITERIA AND SCALES				**Marks Obtained**		
				S1	**S2**	**S3**
Criterion1: Has the student implemented an efficient and scalable solution for data retrieval, processing, and reporting?						
0	1	2	3			
The student has not even implemented a basic solution that meets the project's requirements.	The student has implemented a basic solution that meets the project's requirements but may lack optimization in certain aspects.	The student has implemented a proficient and well-optimized solution.	The student has implemented an exceptionally efficient and scalable solution.			
Criterion 2: Has student demonstrated a strong understanding of C programming fundamentals?						
0	1	2	3			
The student doesn't have basic understanding of C programming fundamentals.	The student exhibits a basic understanding of C programming fundamentals.	The student demonstrates a strong understanding of C programming fundamentals.	The student demonstrates an exceptional understanding of C programming fundamentals.			
Criterion 3: How well written is the report?						
0	1	2	3			
The submitted report is unfit to be graded.	The report is partially acceptable.	The report is complete and concise.	The report is exceptionally written.			
			Total Marks:			