# COMPUTER ENGINEERING WORKSHOP

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

**Project Group ID:**

|  |  |
| --- | --- |
| FAIZAN AHMED | CS-052 |
| TAHIR ALI | CS-058 |
|  |  |

**BATCH:** 2023

**Department of Computer and Information Systems Engineering**

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

**Karachi-75270**

**CONTENTS**

**S.No. Page No.**

1. **Problem Description** 1

1. **Methodology**  2
2. **Results**  3

**CHAPTER 1**

# PROBLEM DESCRIPTION

This project involves designing an **Integrated Environmental Monitoring System (IEMS)** in C, which leverages real-time environmental data. The software is designed to help users monitor critical environmental conditions and optimize resource management.

The project's scope includes the following functionalities:

* **Interfacing with APIs** to fetch real-time environmental data such as temperature and humidity.
* **Data Storage:** Raw and processed data are saved in separate files for future analysis.
* **Shell Scripts:** Automate tasks, including data retrieval and preprocessing.
* **Efficiency and Optimization:** Use pointers and dynamic memory allocation to handle large datasets.
* **Alerts:** Utilize Linux system calls to notify stakeholders of significant or critical readings.
* **Code Modularity:** Implement header files for better organization and readability of the C code.

This project aligns with **CLO-1**, focusing on attaining hands-on experience with contemporary computer engineering technologies.

**CHAPTER 2**

**Methodology**

The project was developed following these steps:

1. **API Integration:**
   * A free API providing environmental data (e.g., OpenWeatherMap) was integrated using HTTP requests in the C program.
   * JSON responses from the API were parsed to extract required parameters.
2. **Data Handling:**
   * Raw data was stored in .json files for traceability.
   * Processed data, including daily summaries and averages, was stored in .txt files for simplified reporting.
3. **Shell Scripting:**
   * Bash scripts were created to automate repetitive tasks, such as scheduling data retrieval every hour and summarizing data every 24 hours.
4. **Optimization:**
   * Pointers and dynamic memory allocation ensured memory-efficient processing of data arrays and structures.
5. **Alerts:**
   * Critical conditions (e.g., high temperature or humidity) triggered real-time alerts using kill() and sigaction() Linux system calls.
6. **Modular Programming:**
   * Functions and variables were organized in three files:
     + main.c: Core logic and execution.
     + functions.c: Helper functions.
     + functions.h: Declarations and definitions.

CHAPTER 3

Results

* **Functional Software:** Successfully developed an environmental monitoring system.
* **Automation:** Shell scripts enabled seamless data collection and processing without manual intervention.
* **Optimization:** Memory allocation improved program efficiency, handling up to 1,000 readings in real-time.
* **Alert System:** Linux notifications provided timely alerts to users for critical readings.
* **Modular Code:** The program is easy to maintain and extend due to the use of header files and modular design.

Outputs









