📌 Project Title

Real-Time Word Counter and Token Classifier in C

🎯 Objective

To design and implement a real-time monitoring application that:

- Continuously observes a text file for changes
- Identifies and classifies tokens into words, integers, and floating-point numbers
- Supports portable string parsing and modular design principles

📂 Functional Requirements

✅ Core Features

Feature	Description
File Monitoring	Polls a text file (input.txt) for changes and updates token statistics
Token Parsing	Reads tokens using customizable delimiter logic while preserving valid float symbols
Token Classification	Distinguishes between words, integers, and floating-point numbers
Real-Time Feedback	Displays updated statistics in each polling cycle
Stop Condition	Terminates if keyword exit is detected
Logging	Prints details such as polling iteration, file size change, and result summary

📊 Token Types

- **Word**: Any non-numeric token not recognized as integer or float
- **Integer**: A whole number with optional leading '+' or '-'
- **Float**: A decimal number with exactly one '.' and valid digit structure

□ Non-Functional Requirements

- **Portability**: Compatible with POSIX systems (uses unistd.h and sleep)
- **Memory Safety**: Token parsing bounded by MAX_TOKEN_SIZE to avoid overflow
- **Clean Coding**: Modular functions with descriptive naming and MISRA C–friendly design (optional integration)
- **Extendability**: Structured to allow easy addition of new token types (e.g. date, hexadecimal)

⚙️ Technical Specifications

- **Language**: C (C99 or later recommended)
- **Compiler**: GCC / Clang

- **Dependencies**: Standard C libraries (stdio.h, stdlib.h, ctype.h, etc.)
- **Polling Interval**: 2 seconds (POLL_INTERVAL)
- **File Path**: Hardcoded to input.txt

🗁 File Structure

word_counter.c # Main source file input.txt # Input monitored for changes

 Suggested Enhancements

- Dynamic file path using command-line arguments
- Log output to separate summary file
- Integration with static analysis tools like Cppcheck or Clang-Tidy for student projects

🧑🏫 Educational Relevance

This project is designed to:

- Reinforce understanding of file I/O and string handling
- Practice parsing logic and basic classification techniques
- Introduce real-time monitoring without concurrency