Real-Time Logging System for Online Shopping !!

# using multi-threading ..

# Introduction:-

* In today’s digital age online shopping has become an integral part of our daily lives . With the increasing number of online transactions, it is crucial to have robust logging system in a place to monitor and record all shopping activities in real-time.
* This project ”Real-Time System Logging for Online Shopping using Multi-Threading”, aims to design and implement a scalable and efficient logging system that can handle multiple events simultaneously, providing valuable insight threats into system activity and security threats.

# Methodologies:-

* Multi-threading: To simulate concurrent shopping events and log messages in real-time.
* Synchronization: Using mutexes and lock guards to ensure thread-safe access to shared resources (log file).
* Modular Programming: Organizing code into reusable functions (logMessage, shoppingEvent, main) for better maintainability.
* Sequential Programming: Using conditional statements (if-else) and loops (for loop) to control program flow.
* Procedural Programming: Using procedures (functions) to perform specific tasks (logging, event simulation).
* Real-time Programming: Logging events as they occur, providing a real-time record of system activity.
* Test-Driven Development (TDD): Implicitly used by testing the logging mechanism with simulated shopping events.
* Iterative Development: Developing the project in stages, adding features and functionality incrementally.

# Code :-

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <thread>

#include <mutex>

#include <chrono>

#include <ctime>

// Mutex for synchronizing access to the log file

std::mutex logMutex;

// Function to write log messages to a file

void logMessage(const std::string& message) {

std::lock\_guard<std::mutex> guard(logMutex);

std::ofstream logFile("shopping\_log.txt", std::ios::app);

if (logFile.is\_open()) {

auto timestamp = std::chrono::system\_clock::now();

std::time\_t timestamp\_c = std::chrono::system\_clock::to\_time\_t(timestamp);

logFile << std::ctime(&timestamp\_c) << message << std::endl;

} else {

std::cerr << "Log file is not open." << std::endl;

}

}

// Real-time thread function to simulate various shopping events

void shoppingEvent(const std::string& event) {

std::this\_thread::sleep\_for(std::chrono::seconds(1)); // Simulate work

logMessage(event);

}

int main() {

// Log user login event

logMessage("Customer logged in");

// Array of products to suggest

std::vector<std::string> products = {"Gua sha tool", "Sheet masks", "Dr.sheth's sunscreen"};

// Display the products as suggestions

std::cout << "Products you might like:" << std::endl;

for (int i = 0; i < 3; ++i) {

std::cout << i + 1 << ". " << products[i] << std::endl;

}

// Handle user input for product selection

int choice;

std::cout << "Choose a product to buy (enter the number): ";

std::cin >> choice;

// Validate user input and process

std::string chosenProduct;

if (choice >= 1 && choice <= 3) {

switch (choice) {

case 3:

chosenProduct = "Dr.sheth's sunscreen";

break;

case 1:

chosenProduct = "Gua sha tool";

break;

case 2:

chosenProduct = "Sheet masks";

break;

}

logMessage("Customer purchased " + chosenProduct);

} else {

std::cout << "Invalid choice. Please choose a number between 1 and 3." << std::endl;

}

// Log user logout event

logMessage("Customer logged out");

std::cout << "Logging of events completed." << std::endl;

return 0;

}

# Output:-



