Producer–Consumer – Layer 3

This project implements the classic Producer-Consumer problem using POSIX threads and semaphores. The main goal is to simulate how multiple producers and consumers share a limited buffer while avoiding race conditions and incorrect access to shared data.

For this implementation, I used the standard three-semaphore solution. The buffer is treated as a bounded circular queue, and access to it is controlled using:

mutex – a binary semaphore that ensures only one thread (producer or consumer) can access the buffer at a time.

empty – a counting semaphore that tracks how many empty slots remain. Producers must wait here when the buffer is full.

full – a counting semaphore that tracks how many items are available. Consumers must wait here when the buffer is empty.

Compilation Process

I used Ubuntu inside VirtualBox to run and test the program.

To compile the program in the terminal, use:

gcc -pthread producer_consumer.c -o producer_consumer

or if using a Makefile:

make

Execution Instructions

To run the program:

./producer_consumer

While the program runs, you should see output showing:

• Producers adding items into the buffer

• Consumers removing items

• Updated buffer state after each action

Let it run for at least 1–2 minutes to verify:

• The buffer never exceeds its capacity

• Consumers never consume from an empty buffer

• No threads freeze or get stuck

To stop the program, press:

Ctrl + C