**Library Management System**

**Project Report**

**Project Title:** Library Management System (Reader-Writer Problem)

**Group Members:**

M. Owais Khurram 23k-2041

Faiez Mehboob 23k-2018

Muneeb 23k-2032

**Introduction:**

In this project, (Library Management System) we solved the Reader-Writer problem using multithreading. The problem involves multiple readers and writers accessing a shared resource (library). We used synchronization to ensure that readers can read simultaneously, but writers get exclusive access when writing.

**Objective:**

* Implement reader-writer synchronization using mutex and semaphores.
* Allow multiple readers to read the shared data simultaneously.
* Writers must have exclusive access when updating the data.
* Prevent data corruption by ensuring proper synchronization.

**Synchronization Concepts:**

**Reader-Writer Problem:**

The Reader-Writer problem describes a situation where:

* Multiple readers can simultaneously read a shared resource without interfering with each other.
* Writers, however, require exclusive access to the shared resource.  
  No other readers or writers should access the resource while a writer is writing.

Synchronization Tools Used:

**Mutex:**

* A **mutex** is used to protect access to a shared variable.
* reader\_count keeps track of the number of active readers. And total\_read, total\_writes keeps track of total read and write operations.

A binary semaphore/mutex (read\_block) is used for reader\_count and stats\_mutex is used for total\_read, total\_writes.

* Since multiple threads can modify these global variables at the same time, mutex ensures that only one thread updates it at a time.

**Semaphores:**

* A semaphore (write\_block) is used to manage access to the shared resource.
* It ensures that only one writer or multiple readers access the resource correctly.
* When the first reader arrives, it locks the write\_block (to stop writers).
* When the last reader leaves, it unlocks the write\_block.
* Writers must acquire write\_block before writing.

**Reader Logic:**

* Lock mutex to update reader\_count.
* If it's the first reader, wait on write\_block to block writers.
* Unlock mutex.
* Read the shared resource.
* Lock mutex again to decrement reader\_count.
* If it's the last reader, signal write\_block to allow writers.
* Unlock mutex.

**Writer Logic:**

* Wait on write\_block (blocks until there are no active readers or writers).
* Write/update the shared resource.
* Signal write\_block after finishing.

**Need for Synchronization**

Without synchronization:

* Two writers might overwrite each other’s changes (data inconsistency).
* A reader might read partial updated data while a writer is modifying.
* Multiple writers or readers could corrupt shared variables like reader\_count.

Thus, mutexes and semaphores work together to manage access to shared resources safely, efficiently, and without deadlocks.

**Simple Dry Run:**

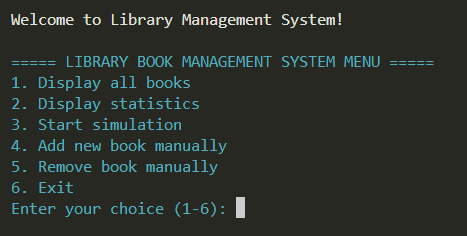
During actual simulation it will be random on how and when readers and writers access the library, but to simply simulate the reader and writer problem, and how we are solving it we can take an example of 2 readers and 1 writer.

**Dry run table:**

|  |  |  |
| --- | --- | --- |
| **Event** | **Reader\_count** | **Action** |
| Reader1 enters | 1 | Blocks writers |
| Reader2 enters | 2 | Continues reading |
| Writer tries to enter | 2 | Writer blocked |
| Reader1 leaves | 1 | Still reading |
| Reader2 leaves | 0 | Writer now allowed to write |
| Writer writes | 0 | Exclusive access |

**Simulation of Project:**

**Menu:**

****

**Menu Error Handling:**

**Wrong input for menu:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Wrong input for readers and writers:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Display books:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Simulation:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Statistics:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Manual book add:**

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Remove book manually:**

**A screenshot of a computer

AI-generated content may be incorrect.**