Make the Server Multithreaded

CMS 330, Spring 2016

Due Friday, May 6, at 11:59 PM

Description

This is the final phase of the `rCache` project. Make sure you've completed Project 6 *before* beginning work on this one.

You may continue to work with a partner on this part of the project.

Your task is to make the server run a pool of worker threads that process hash table operations. The server's main thread will run a loop, waiting for new client connections to appear. When the server receives a connection, it should immediately place the connection in a buffer of waiting connections, then go back to the beginning of the loop and wait for the next arrival.

When a new connection appears in the buffer, a worker thread should retrieve it, read the first character to determine the hash table operations, then call the appropriate hash table function. When the operation completes, the worker thread should close the connection, then go back to waiting for another connection to appear in the buffer.

Note that the main and worker threads are in a producer-consumer relationship with a shared buffer. Therefore, you must use locking to protect the buffer. The main thread must signal a waiting worker thread when it places a new entry in the buffer, and the worker threads must signal the main thread when they free a location.

For this project, please use **semaphores** to implement the producer-consumer setup. The included file `sem-pc.c` has a semaphore-based version of the producer-consumer framework that you can adapt to your own code.

Handing It In

Put your code in a `Project7` directory. Include a `Makefile` and text file indicating the partner you worked with. Note that you should leave your Project 6 code in a separate `Project6` directory.