a) Experience Installing Nachos (Platform Details)

I installed Nachos 3.4 on the FIU Ocelot server, which runs a Linux environment. After connecting via SSH, I unpacked the tar archive using tar -xzf and then confirmed that the necessary tools—particularly make and gcc—were installed. I also ensured that my LD\_LIBRARY\_PATH environment variable included /usr/local/lib to avoid library-related issues. Compiling Nachos in the nachos-3.4/code directory produced the nachos binary with no errors. The directory structure is quite clear once you identify important subdirectories: threads (for kernel-level threading), userprog (for running user-mode programs under Nachos), and test (which contains sample programs like halt).

b) Experience with the Tests

I ran ./nachos in the threads subdirectory and observed multiple loop messages, confirming thread switching was correct. Then, I tested user programs by executing ./nachos -x test/halt, which displayed “Machine halting!” and printed system stats. Using the -s (single-step) flag provided detailed instruction-level debugging. All tests worked as expected with no runtime errors.

c) Description of the Thread Class in Nachos

The Thread class in Nachos encapsulates a kernel-level thread, maintaining its stack, CPU register state, and status (which can be JUST\_CREATED, RUNNING, READY, or BLOCKED). It provides core methods like Fork, which sets up and runs a specified function in a new thread; Yield, which voluntarily gives up the CPU to let other ready threads run; Sleep, which blocks the thread and lets the scheduler pick a different one; and Finish, signaling that the thread has completed execution. Internally, each Thread tracks its machineState (for saving registers during context switches) and allocates its own stack space to store local variables and function call frames. If running in user mode, the thread also has a separate set of userRegisters and points to an AddrSpace that represents the program’s virtual memory. This structure allows multiple threads to run concurrently under Nachos, managing both kernel-only threads and user-level processes.