kacchiOS – Project Completion Checklist

Submission-ready checklist summarizing implemented operating system components.

Project Overview

kacchiOS is a minimal 32-bit x86 educational operating system developed to demonstrate core OS concepts including memory management, process management, scheduling, context switching, and inter-process communication.

Feature Checklist

Category	Feature	Status
Boot & Init	Multiboot bootloader, protected mode, kernel entry	Completed
Boot & Init	Serial I/O (COM1) for kernel output/input	Completed
Memory	Heap initialization (meminit)	Completed
Memory	Dynamic allocation (getmem – best fit)	Completed
Memory	Deallocation (freemem)	Completed
Memory	Process stack allocation/deallocation (getstk/freestk)	Completed
Process	Process Control Block (PCB) & process table	Completed
Process	Process creation and termination	Completed
Process	Process states (FREE, READY, RUNNING, BLOCKED)	Completed
Scheduling	Priority-based scheduler	Completed
Scheduling	Round-robin within same priority	Completed
Scheduling	Aging to prevent starvation	Completed
Context Switch	Stack-based context switching	Completed
Context Switch	x86 assembly (save/restore registers & ESP)	Completed
Multitasking	Cooperative multitasking via yield()	Completed
Blocking	Process blocking and wakeup	Completed
IPC	Message passing (send/receive)	Completed
IPC	Blocking receive with automatic wakeup	Completed
Shell	Interactive kernel prompt	Completed

Notes on Design Choices

- Scheduling is cooperative (no timer interrupt), suitable for an educational OS lab.
- Context switching is implemented using x86 assembly and independent process stacks.
- Priority aging is used to ensure fairness and avoid starvation.
- IPC is message-based and integrates with process blocking/wakeup.

Not Implemented (Out of Scope)

Preemptive scheduling, virtual memory, user/kernel mode separation, filesystem, and networking were intentionally excluded as they were not required for this project.

Declaration

All listed components have been implemented, tested on QEMU (x86), and verified through functional and integration testing.