CPU Scheduling Algorithm Simulator

platform web tech HTML | CSS | JavaScript | Tailwind

Team Members

This project was collaboratively developed by the following team members:

1. **HAN Raby**
   Student ID: e20220580
2. **CHEA Chanminea**
   Student ID: e20220335
3. **HENG Oulong**
   Student ID: e20221390
4. **LIM Kity**
   Student ID: e20221479

Table of Contents

- CPU Scheduling Algorithm Simulator
  - Team Members
  - Table of Contents
  - Project Overview
  - Project Objectives
  - Key Features
  - Algorithms Implemented
  - Performance Metrics
  - Sample Input Scenario
  - Algorithm Comparison
  - How to Run the Project
  - Screenshots
  - Challenges Faced
  - Conclusion

Project Overview

The CPU Scheduling Algorithm Simulator is a concise, web-based educational application developed to illustrate the operational behaviour of classical and contemporary CPU scheduling strategies. The simulator presents each scheduling decision visually through an interactive Gantt chart and reports quantitative performance metrics for systematic analysis.

Project Objectives

- Demonstrate the scheduling order and temporal execution of processes under multiple algorithms.
- Provide accurate computation of per-process and aggregate performance metrics (Waiting Time and Turnaround Time).
- Offer an accessible, interactive environment for laboratory exercises and coursework demonstrations in an undergraduate Operating Systems curriculum.

Key Features

- **Interactive Input Model:** Accepts Arrival Time, Burst Time and Priority for each process; supports dynamic editing of the process table.
- **Gantt Chart Visualization:** Real-time timeline rendering of process execution sequences to facilitate conceptual understanding.
- **Performance Summary:** Per-process Waiting Time (WT) and Turnaround Time (TAT), and overall averages, presented in a tabular summary.
- **Configurable Parameters:** Selectable algorithm modes and a configurable time quantum for Round Robin and MLFQ queue quanta.
- **Priority Support:** Priority values are used by MLQ and MLFQ modes to determine queue assignment and scheduling order.

Algorithms Implemented

1. **First Come First Serve (FCFS)**

- Non-preemptive scheduling where processes are executed strictly in order of arrival.

2. **Shortest Job First (SJF) — Non-preemptive**

- Selects the process with the smallest burst time from the ready set; once started, a process runs to completion.

3. **Shortest Remaining Time (SRT) — Preemptive**

- At each scheduling decision the process with the least remaining CPU time is selected; preemption occurs when a newly arrived process has a smaller remaining time.

4. **Round Robin (RR)**

- Time-shared preemptive scheduling that rotates processes in the ready queue using a fixed time quantum; suited for equitable CPU distribution.

5. **Multilevel Queue (MLQ)**

- Fixed partitioning into queues according to priority classes; higher priority queues are served preferentially with no dynamic migration between queues.

6. **Multilevel Feedback Queue (MLFQ)**

- Three-level feedback architecture with distinct time quanta per level, promotion/demotion rules, and an aging mechanism to mitigate starvation while adapting to observed CPU behaviour.

Performance Metrics

The simulator reports the following canonical metrics for each process and their averages.

- Waiting Time (WT): total time a process spends waiting in the ready queue prior to its execution completion.

```
WT = TAT - BT
```

- Turnaround Time (TAT): interval from arrival to completion of the process.

```
TAT = CT - AT
```

Where: CT = Completion Time, AT = Arrival Time, BT = Burst Time.

Sample Input Scenario

Process	Arrival Time	Burst Time	Priority
P1	0	5	2
P2	1	3	1
P3	2	8	3
P4	3	6	2

This table is provided as a reproducible test case for classroom demonstration and lab exercises. Use the simulator controls to select different algorithms and observe the resultant Gantt chart and metric table.

Algorithm Comparison

Algorithm	Mode	Preemptive	Typical Behaviour	Suitable For
FCFS	Non-preemptive	No	Simple, deterministic order based on arrival	Introductory examples
SJF	Non-preemptive	No	Minimizes average waiting time when service times are known	Offline scheduling analysis
SRT	Preemptive	Yes	Dynamically favours shorter remaining jobs; can reduce average waiting time	Interactive, dynamic workloads
RR	Preemptive	Yes	Time-slicing for fairness; response time depends on quantum	Time-sharing systems

Algorithm	Mode	Preemptive	Typical Behaviour	Suitable For
MLQ	Non-preemptive (per queue)	Mixed	Fixed-priority segregation; strict service order by queue	Systems with distinct job classes
MLFQ	Preemptive (feedback)	Yes	Adaptive priorities, aging to avoid starvation	Systems requiring responsiveness and adaptability

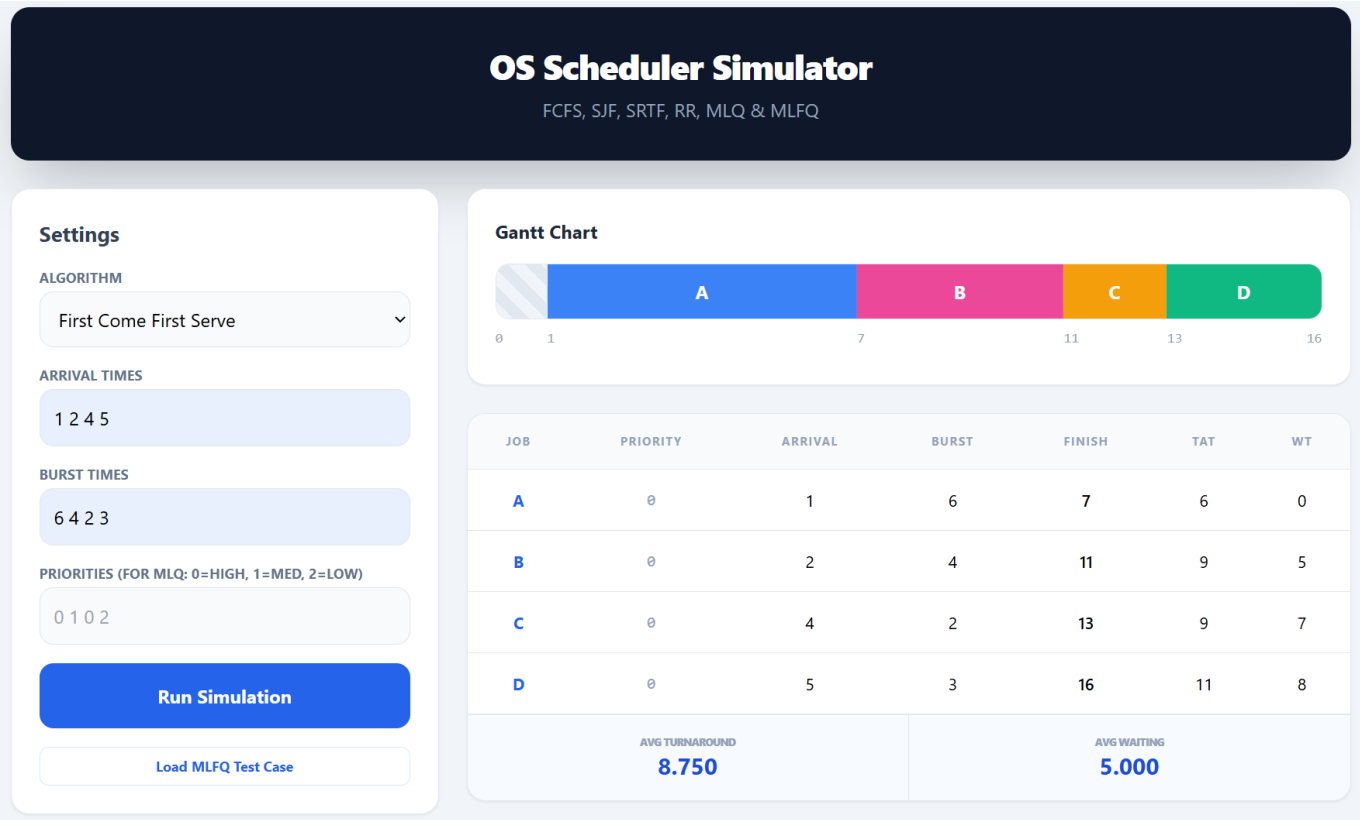
How to Run the Project

Open the application by loading `index.html` in any modern web browser (for example, Chrome, Edge or Firefox). No build or installation steps are required; the application runs entirely in the browser.

Steps:

1. Launch a web browser.
2. Open the project file `index.html` from the project root.
3. Enter process records (Arrival Time, Burst Time, Priority), select an algorithm, and execute the simulation.

Screenshots



OS Scheduler Simulator

FCFS, SJF, SRTF, RR, MLQ & MLFQ

Settings

ALGORITHM

Shortest Job First (Non-Preemptive)

ARRIVAL TIMES

1 2 4 5

BURST TIMES

6 4 2 3

PRIORITIES (FOR MLQ: 0=HIGH, 1=MED, 2=LOW)

0 1 0 2

Run Simulation

Load MLFQ Test Case

Gantt Chart

JOB	PRIORITY	ARRIVAL	BURST	FINISH	TAT	WT
A	0	1	6	7	6	0
B	0	2	4	16	14	10
C	0	4	2	9	5	3
D	0	5	3	12	7	4

AVG TURNAROUND **8.000**

AVG WAITING **4.250**

OS Scheduler Simulator

FCFS, SJF, SRTF, RR, MLQ & MLFQ

Settings

ALGORITHM

Shortest Remaining Time (Preemptive)

ARRIVAL TIMES

1 2 4 5

BURST TIMES

6 4 2 3

PRIORITIES (FOR MLQ: 0=HIGH, 1=MED, 2=LOW)

0 1 0 2

Run Simulation

Load MLFQ Test Case

Gantt Chart

JOB	PRIORITY	ARRIVAL	BURST	FINISH	TAT	WT
A	0	1	6	16	15	9
B	0	2	4	6	4	0
C	0	4	2	8	4	2
D	0	5	3	11	6	3

AVG TURNAROUND **7.250**

AVG WAITING **3.500**

OS Scheduler Simulator

FCFS, SJF, SRTF, RR, MLQ & MLFQ

Settings

ALGORITHM

Round Robin

TIME QUANTUM (RR ONLY)

2

ARRIVAL TIMES

1 2 4 5

BURST TIMES

6 4 2 3

PRIORITIES (FOR MLQ: 0=HIGH, 1=MED, 2=LOW)

0 1 0 2

Run Simulation

Load MLFQ Test Case

Gantt Chart

JOB	PRIORITY	ARRIVAL	BURST	FINISH	TAT	WT
A	0	1	6	15	14	8
B	0	2	4	13	11	7
C	0	4	2	9	5	3
D	0	5	3	16	11	8

AVG TURNAROUND 10.250

AVG WAITING 6.500

OS Scheduler Simulator

FCFS, SJF, SRTF, RR, MLQ & MLFQ

Settings

ALGORITHM

Multilevel Queue (Fixed Priority)

ARRIVAL TIMES

0 1 2 4

BURST TIMES

4 6 8 2

PRIORITIES (FOR MLQ: 0=HIGH, 1=MED, 2=LOW)

0 1 2 0

Run Simulation

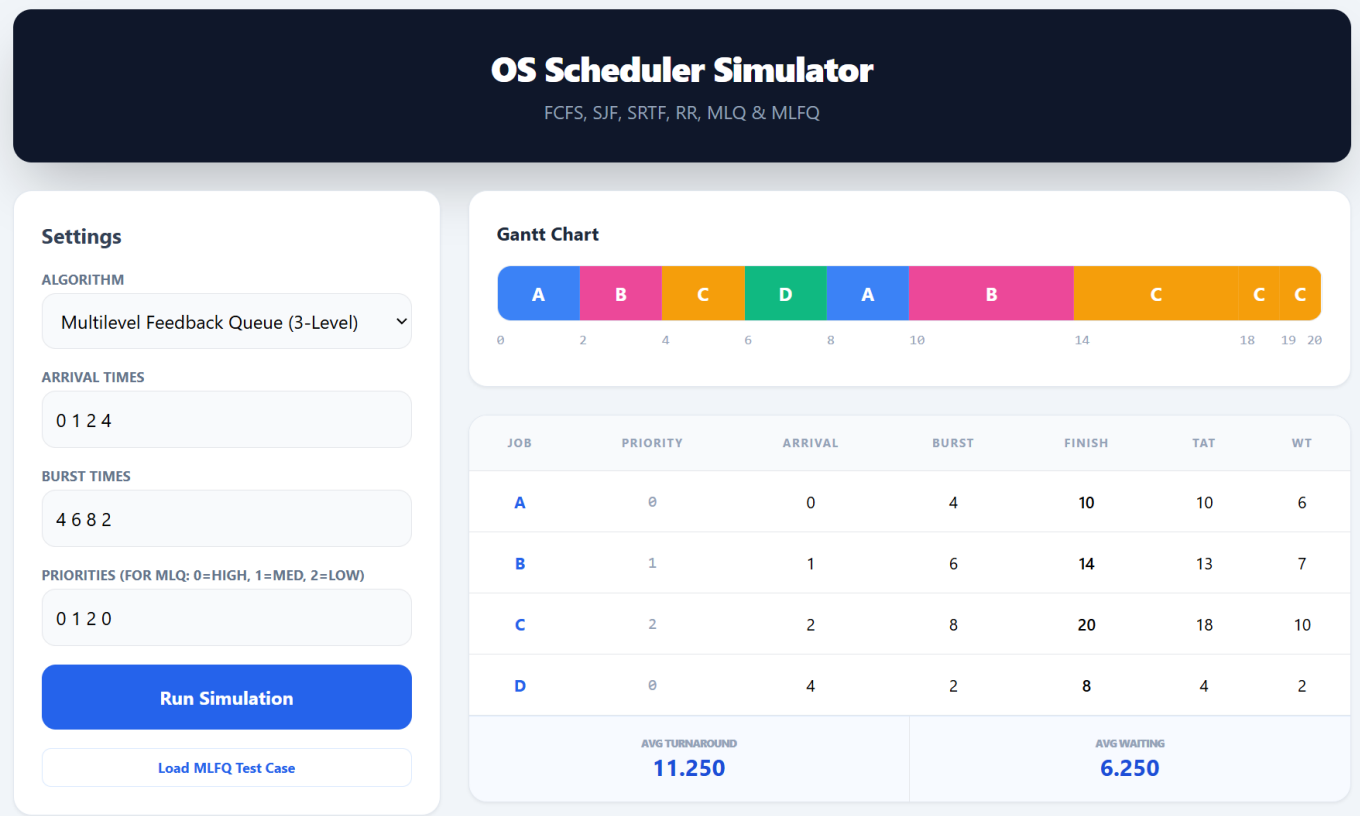
Load MLFQ Test Case

Gantt Chart

JOB	PRIORITY	ARRIVAL	BURST	FINISH	TAT	WT
A	0	0	4	4	4	0
B	1	1	6	12	11	5
C	2	2	8	20	18	10
D	0	4	2	6	2	0

AVG TURNAROUND 8.750

AVG WAITING 3.750



Challenges Faced

- Ensuring numerical precision and consistency in timing calculations across preemptive and non-preemptive modes.
- Designing an intelligible Gantt chart representation that accurately conveys preemption and context switches.
- Balancing clarity of the interactive interface with the need to present comprehensive performance statistics.

Conclusion

This simulator provides a focused, pedagogically oriented environment for the study of CPU scheduling algorithms. Its visual and quantitative presentation supports laboratory exercises, comparative analysis and classroom demonstrations in an undergraduate Operating Systems course.