

User Manual



**Understanding the Process Scheduling Calculator**

**What is this project about?**

This is a **Process Scheduling Calculator** that helps us understand how an operating system decides which task (process) should run next in a CPU.

Imagine you have multiple people waiting in line for a ride at an amusement park. The ride operator has to decide:

* Who goes first? (First Come, First Serve)
* Should people with special passes go before others? (Priority Scheduling)
* What if each person gets a small turn, then moves to the back of the line? (Round Robin Scheduling)

That’s exactly what **CPU Scheduling** does! Our project is like a ride operator for computer tasks!

**How the project works**

**1 The Interface (index.html) – The Frontend**

This is the main webpage where users interact. It allows you to:  
✔️ Select a scheduling algorithm (FCFS, SJF, Priority, Round Robin).  
✔️ Add processes by entering their **Arrival Time, Burst Time (execution time), and Priority**.  
✔️ View added processes in a table.  
✔️ Click "Calculate" to get results.  
✔️ See the results, including **Waiting Time, Turnaround Time**, and a **Gantt Chart** (a visual timeline of process execution).

**2️ The Brain (script.js) – JavaScript Logic**

This file contains all the logic that makes the webpage work. It does three main things:

**1. Adding and Managing Processes**

* When you enter details and click "Add Process," it saves the process in a list.
* You can edit or delete processes anytime before calculation.

**2. Scheduling Algorithms (The Heart of the Calculator)**

When you click "Calculate," it checks which algorithm you selected and applies it:

**First Come First Serve (FCFS)**

* The process that arrives first is executed first.
* Think of a ticket queue at a movie theater – first person in line gets served first!

**Shortest Job First (SJF)**

* The process that needs the least time to complete runs first.
* Imagine you’re in a fast-food line. If someone just needs a coffee (shortest job), they get served before someone ordering a full meal.

**Priority Scheduling**

* The process with the highest priority runs first.
* Think of an emergency room – the most critical patient is treated before others.

**Round Robin (RR)**

* Each process gets a small time slot (Time Quantum) and then moves to the back of the line.
* Imagine kids taking turns on a swing. Everyone gets a fair share!

The JavaScript file performs calculations to determine:  
✔ **Waiting Time** (how long each process waits before execution).  
✔ **Turnaround Time** (total time from arrival to completion).  
✔ The **execution order** of processes.

Finally, it updates the results on the webpage!

**3 The Look & Feel (style.css) – Making It Beautiful**

This file makes everything visually appealing! It:  
✔️ Adds a cool gradient background.  
✔️ Styles buttons with animations.  
✔️ Formats tables nicely.  
✔️ Designs the **Gantt Chart** so it’s easy to read.

**Why is this project useful?**

* Helps students visualize **how CPU scheduling works**.
* Saves time by automating scheduling calculations.
* Gives instant results in a **clean and structured way**.

**Example Scenario**

Imagine you enter these processes:

|  |  |  |  |
| --- | --- | --- | --- |
| **Process ID** | **Arrival Time** | **Burst Time** | **Priority** |
| P1 | 0 | 5 | 3 |
| P2 | 2 | 3 | 1 |
| P3 | 4 | 2 | 2 |

Then you select **Shortest Job First (SJF)** and click "Calculate." The tool will:  
✔️ Determine the best execution order.  
✔️ Show **Waiting Time & Turnaround Time** for each process.  
✔️ Draw a **Gantt Chart** to show the timeline.

This is exactly how real operating systems **schedule tasks inside a computer!**

**Final Thoughts**

This project is like a **mini operating system** for scheduling processes. It takes input from users, processes it with smart logic, and presents results beautifully. It’s a fun and educational way to understand **how computers manage tasks efficiently!**