Visualizing CPU Scheduling Algorithms:

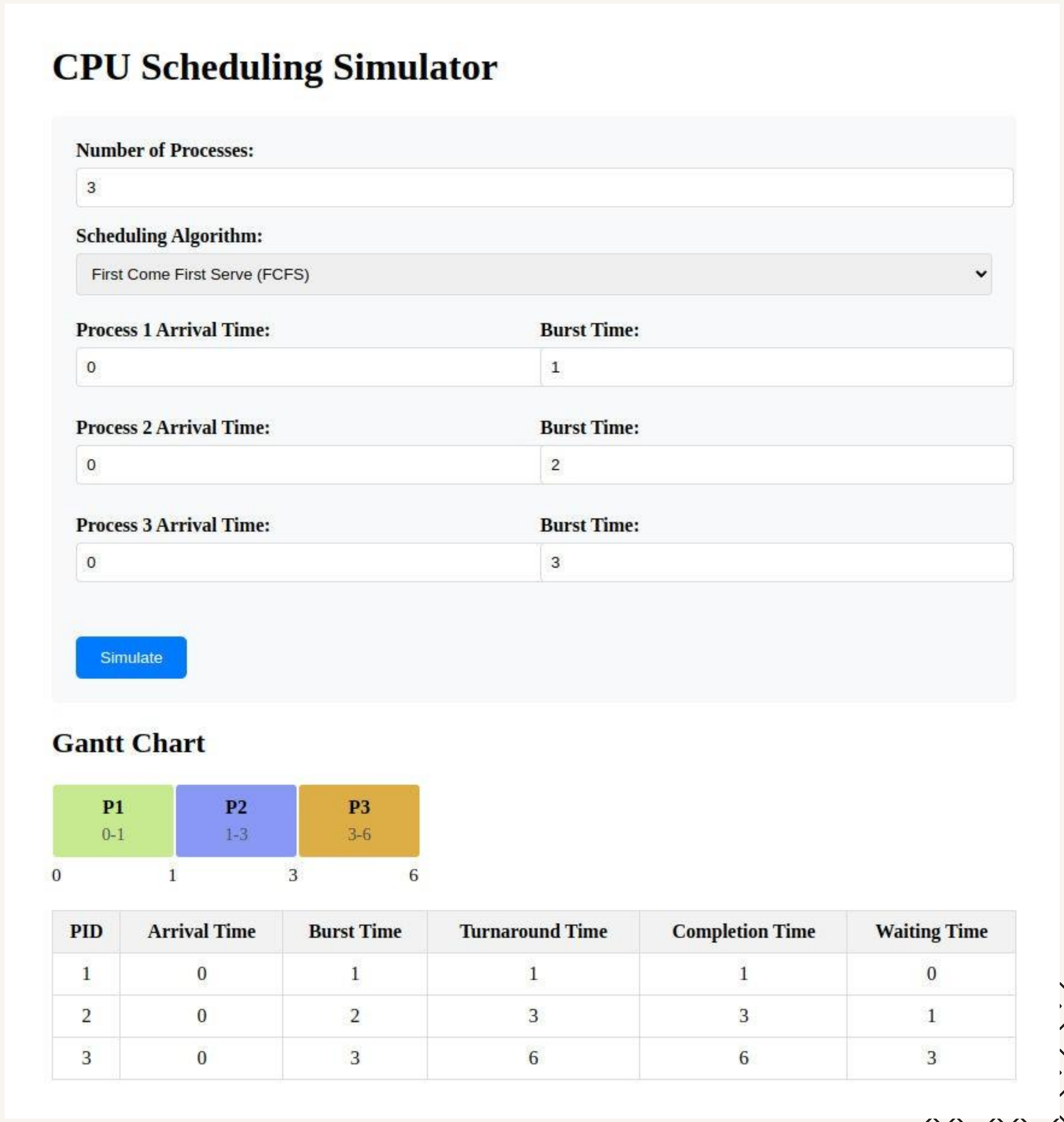
Utsav Shah (2203128)
Nayan Garg (2203315)

Get the project here! - GitHub

Introduction to CPU Scheduling

In this presentation, we will explore **CPU Scheduling Algorithms**, which are essential for managing how processes access the CPU. Understanding these algorithms helps optimize **system performance** and resource utilization, leading to more efficient computing. We will visualize various algorithms and their effectiveness in different scenarios.

CPU Scheduling Algorithms Visualiser



CPU Scheduling Simulator

Number of Processes: 3

Scheduling Algorithm: First Come First Serve (FCFS)

Process 1 Arrival Time: 0	**Burst Time:** 1
Process 2 Arrival Time: 0	**Burst Time:** 2
Process 3 Arrival Time: 0	**Burst Time:** 3

Gantt Chart

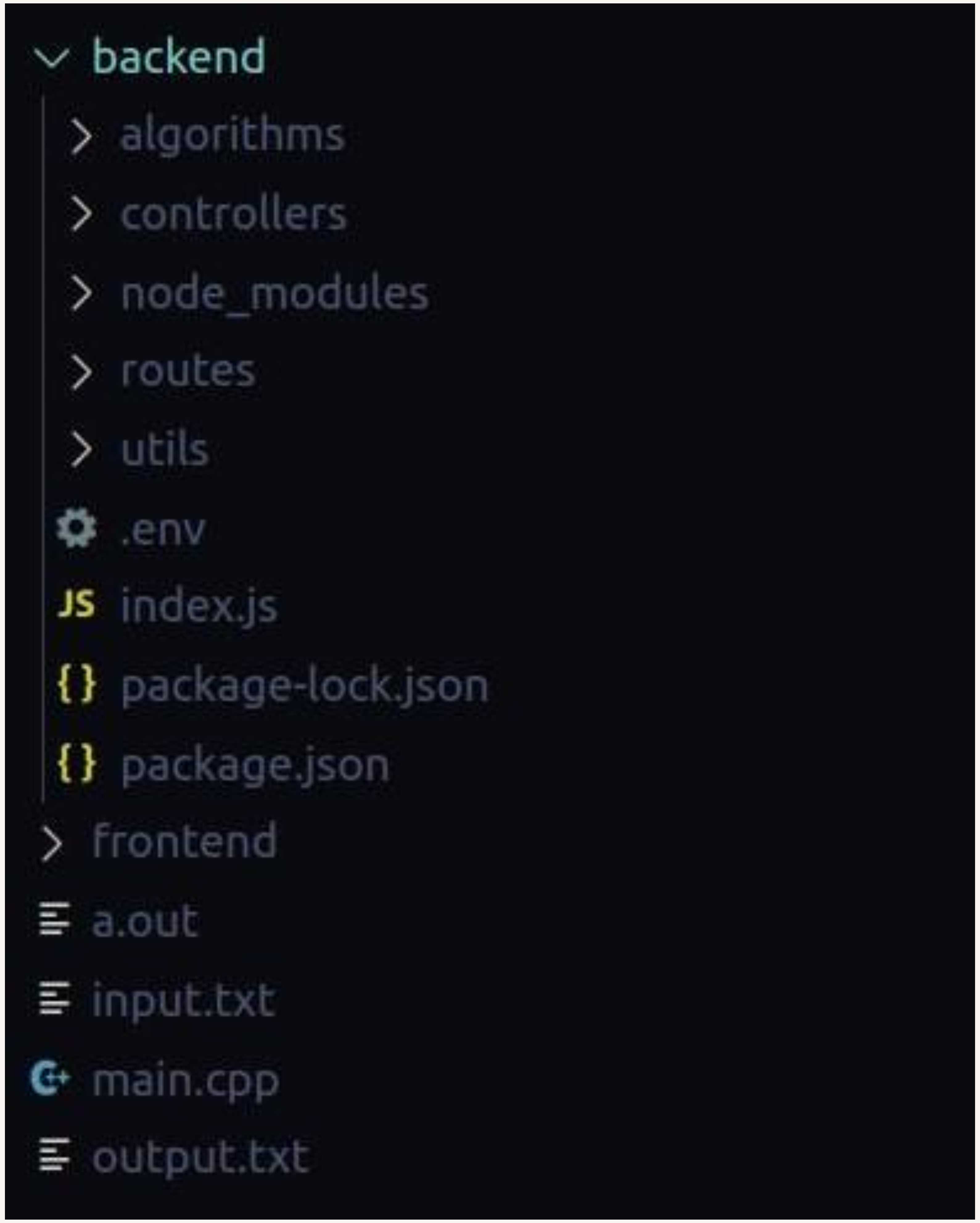
P1	P2	P3
0-1	1-3	3-6

0 1 3 6

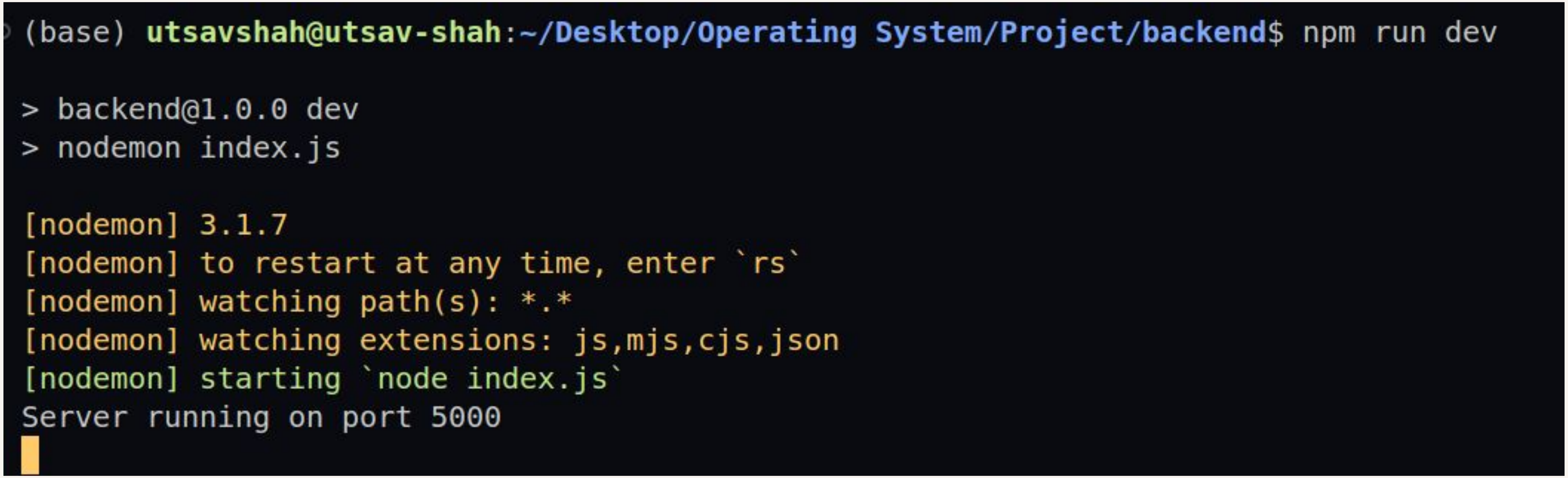
PID	Arrival Time	Burst Time	Turnaround Time	Completion Time	Waiting Time
1	0	1	1	1	0
2	0	2	3	3	1
3	0	3	6	6	3

Installation

- Go into backend folder and do **npm install.**
- Then do **npm run dev :** Server will start running**.**



```
backend
  algorithms
  controllers
  node_modules
  routes
  utils
  .env
  index.js
  package-lock.json
  package.json
frontend
a.out
input.txt
main.cpp
output.txt
```



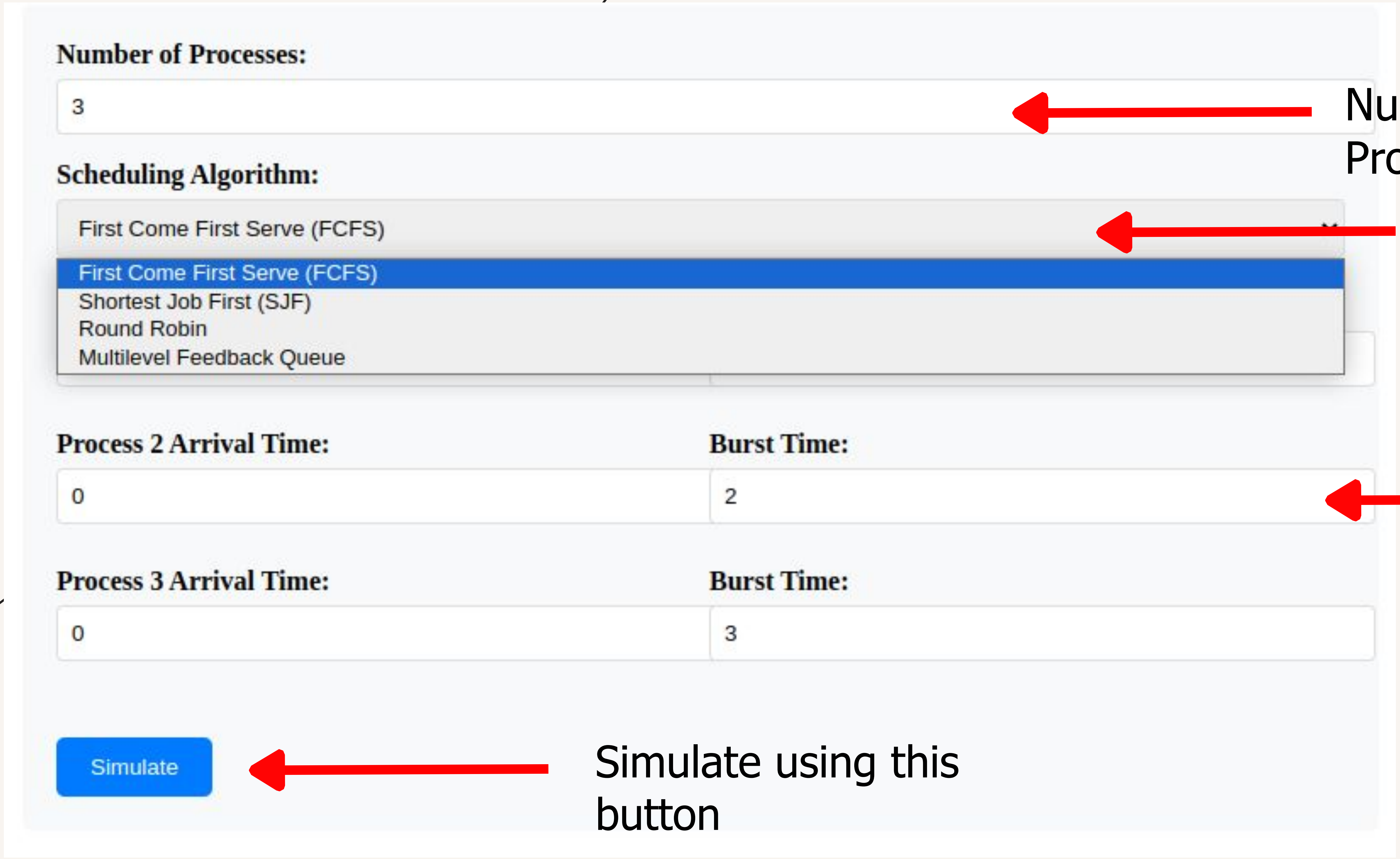
```
(base) utsavshah@utsav-shah:~/Desktop/Operating System/Project/backend$ npm run dev

> backend@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Server running on port 5000
```

Frontend part :

You can directly open HTML file in browser.



Number of Processes:

3

Scheduling Algorithm:

First Come First Serve (FCFS)

Process 2 Arrival Time: 0

Burst Time: 2

Process 3 Arrival Time: 0

Burst Time: 3

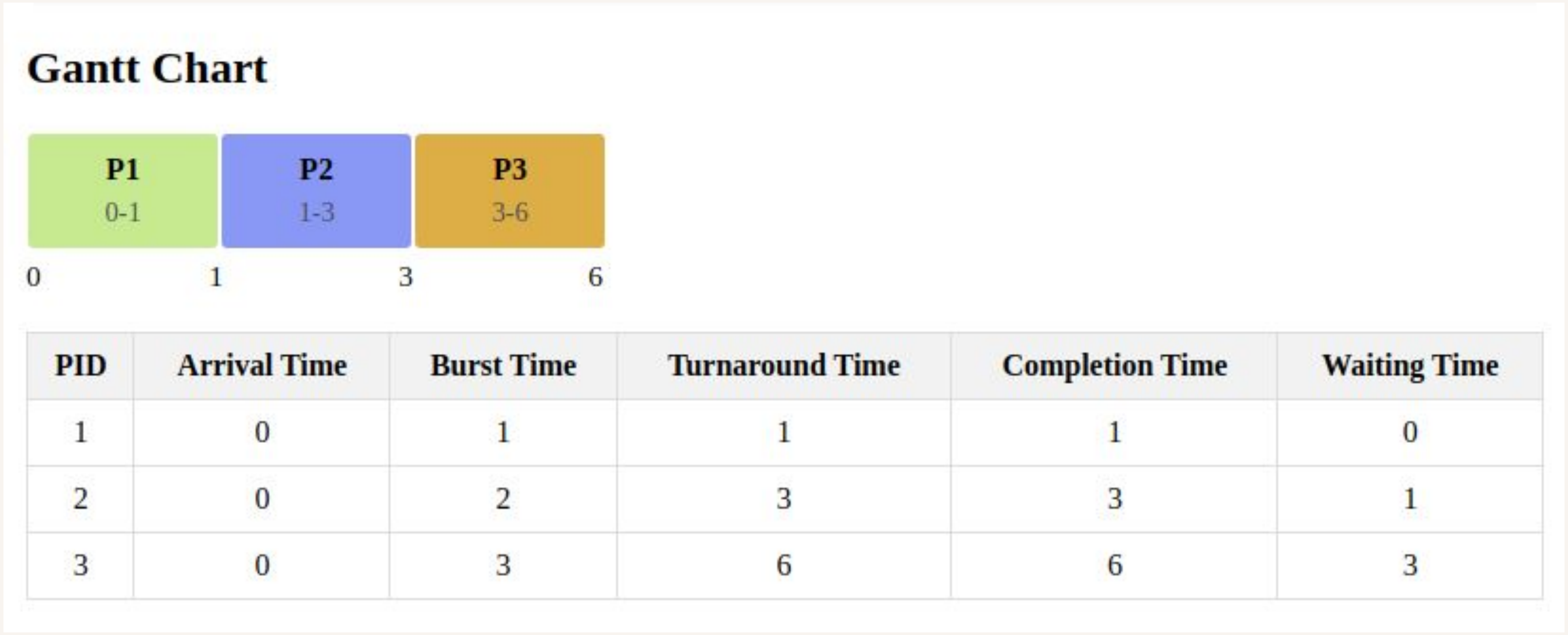
Number of Process

Select algorithm from dropdown

Process Details

Simulate using this button

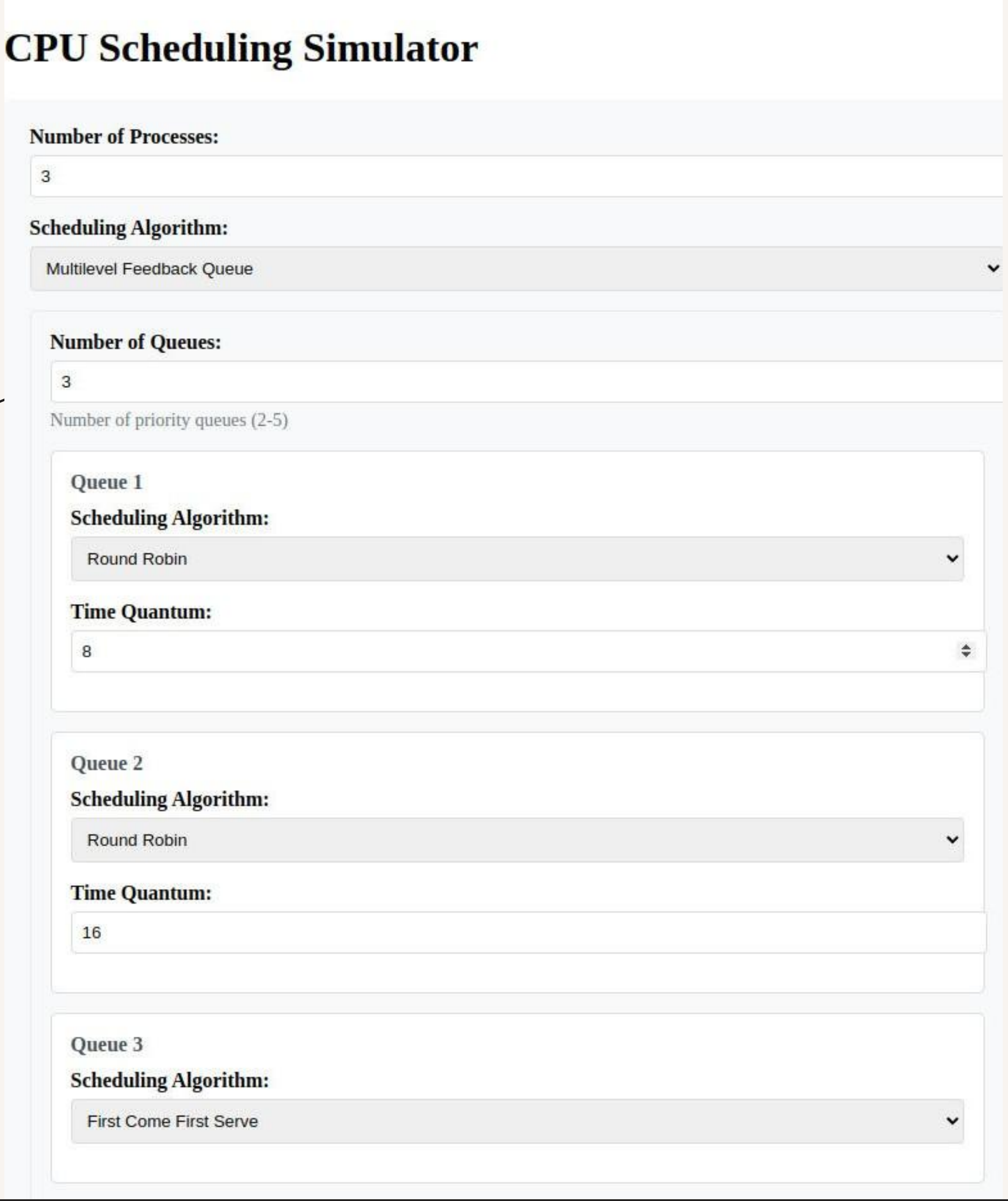
Result



Gantt Chart

PID	Arrival Time	Burst Time	Turnaround Time	Completion Time	Waiting Time
1	0	1	1	1	0
2	0	2	3	3	1
3	0	3	6	6	3

Example MLFQ



CPU Scheduling Simulator

Number of Processes:

3

Scheduling Algorithm:

Multilevel Feedback Queue

Number of Queues:

3

Number of priority queues (2-5)

Queue 1

Scheduling Algorithm:

Round Robin

Time Quantum:

8

Queue 2

Scheduling Algorithm:

Round Robin

Time Quantum:

16

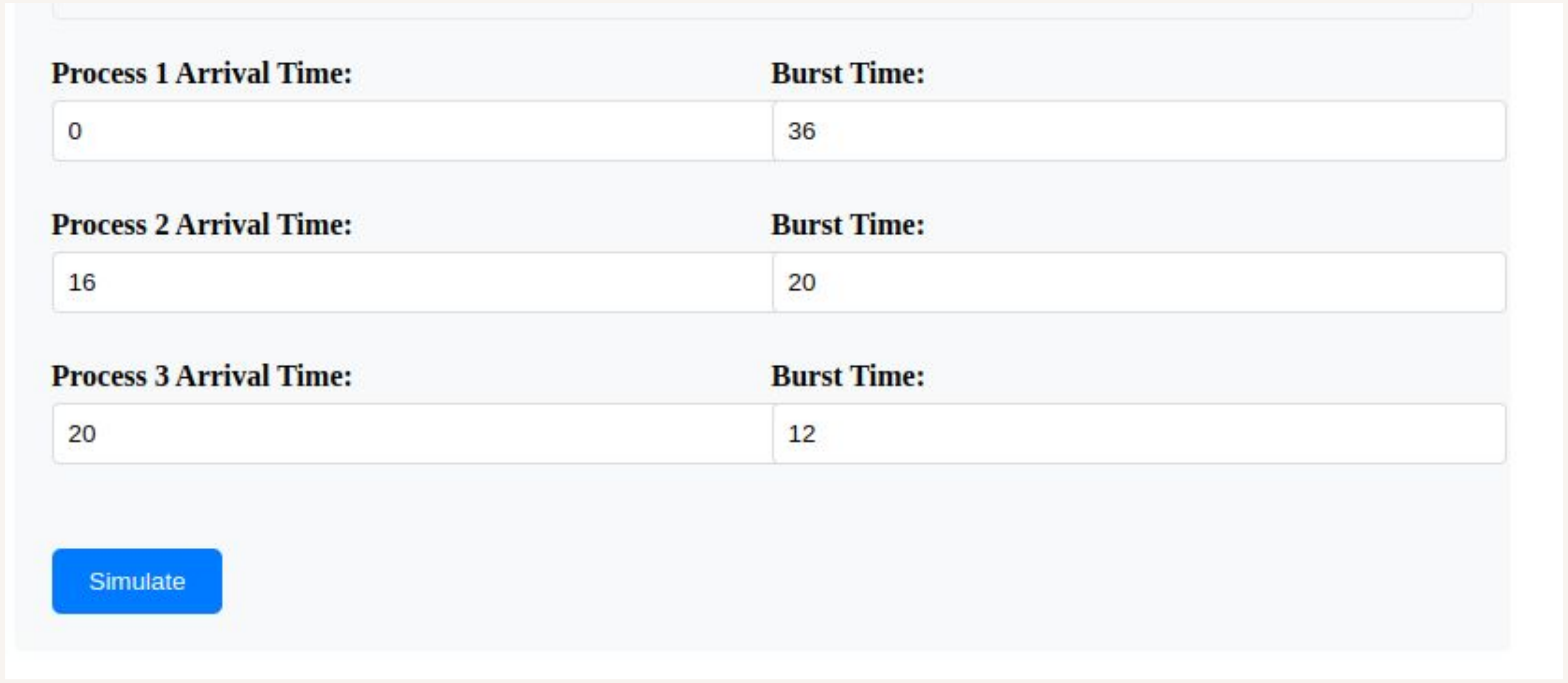
Queue 3

Scheduling Algorithm:

First Come First Serve

By default number of queues will be 3 but this boxes will not appear, so change the number of queues once and then make it 3 to get these boxes.

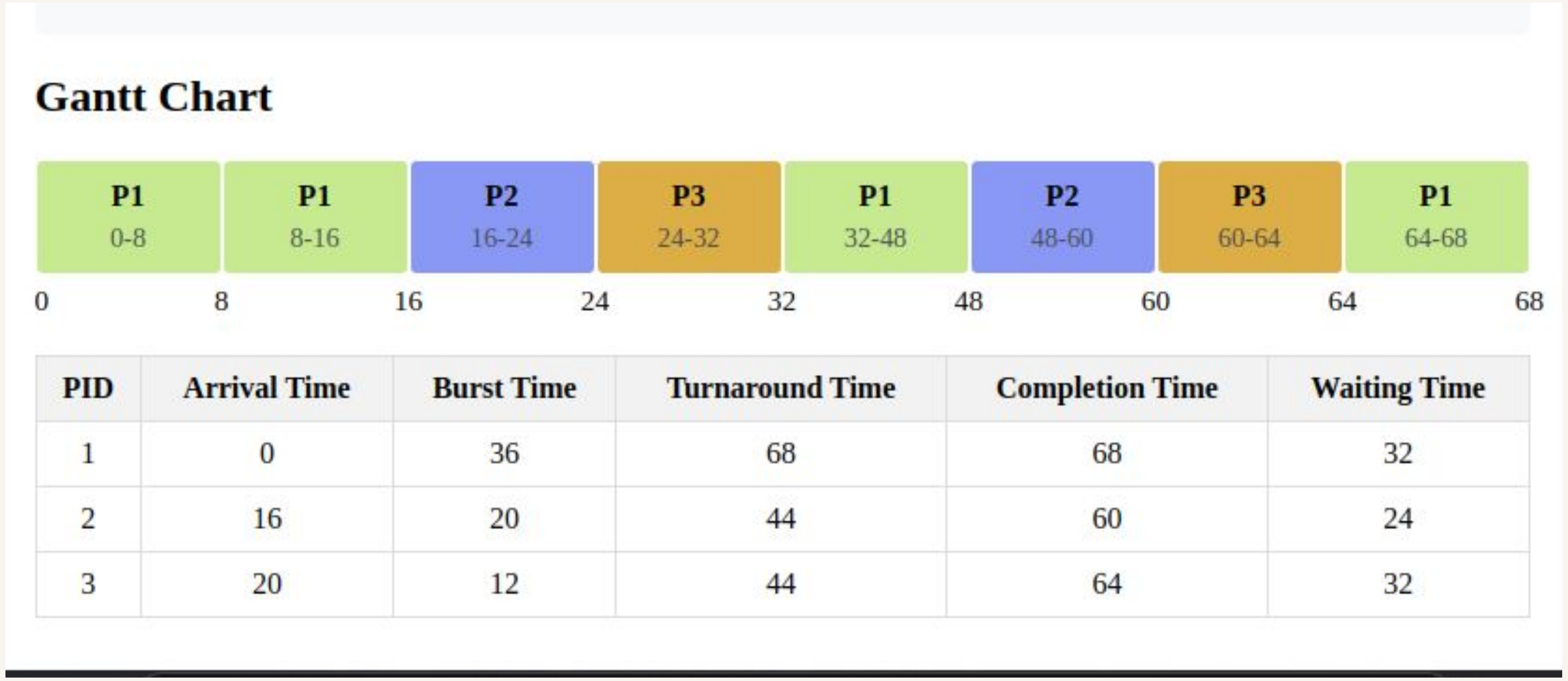
Same for number of processes.



Process 1 Arrival Time: 0 **Burst Time:** 36

Process 2 Arrival Time: 16 **Burst Time:** 20

Process 3 Arrival Time: 20 **Burst Time:** 12



Gantt Chart

P1	P1	P2	P3	P1	P2	P3	P1
0-8	8-16	16-24	24-32	32-48	48-60	60-64	64-68

```
0    8    16    24    32    48    60    64    68
```

PID	Arrival Time	Burst Time	Turnaround Time	Completion Time	Waiting Time
1	0	36	68	68	32
2	16	20	44	60	24
3	20	12	44	64	32

References

1. MLFQ Explanation : Youtube
2. MLFQ : Geeks For Geeks
3. W3 Schools for web development.
4. AI for help in frontend development, GPT and Claude.

Notes:

Files submitted :

1. Algorithms code in JavaScript.
2. MLFQ code in CPP (Coded it and then converted it to JS. After getting better understanding, wrote remaining algorithms directly in JS).
3. HTML File and Script file : frontend part.

Thanks!