CSE316 (Term Paper)

B.tech CSE (Lovely Professional University)

Topic: Intelligent CPU Scheduler Simulator

Bachelor Of Technology

Computer Science and Engineering

By:-

Akshaya Kumar Sahoo[12321958]

Vansh Khokhar[12311795]

Arpit[12309787]

Section :- K23DC

Under guidance of

Mrs. Gagandeep Kaur

UID:- 31683

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

Intelligent CPU Scheduler Simulator

---

1. Project Overview

Objective:
Develop an interactive simulator to visualize **CPU scheduling algorithms** (FCFS, SJF, Round Robin, Priority Scheduling) with performance metrics.

Key Features:

- Input process details (Burst Time, Arrival Time, Priority).
- Simulate scheduling algorithms in real-time.
- Generate **Gantt charts** and **performance metrics** (Avg. Waiting Time, Turnaround Time).
- Compare algorithm efficiency.

Scope:

- No OS-level implementation (pure simulation).
- Supports **4 scheduling algorithms**.
- Desktop application with GUI.

---

2. Module-Wise Breakdown

Module	**Description**
Input Module	Collects process details and algorithm selection via GUI.
Scheduling Module	Implements FCFS, SJF, Round Robin, and Priority Scheduling logic.
Visualization Module	Displays Gantt charts and performance metrics using Matplotlib.

---

3. Functionalities

Input Module

- Process entry form (PID, Burst Time, Arrival Time, Priority).
- Algorithm selection dropdown.
- Time quantum input (for Round Robin).

Scheduling Module

- **FCFS:** Executes processes in arrival order.
- **SJF:** Picks the shortest job first.
- **Round Robin:** Uses time slicing for fairness.
- **Priority:** Executes higher-priority processes first.

Visualization Module

- **Gantt Chart:** Timeline of process execution.
- **Metrics Table:** Avg. Waiting Time, Turnaround Time, Throughput.

---

4. Technology Used

Programming Languages:

- **Python** (Primary language for logic and GUI).

Libraries and Tools:

- **Tkinter** (GUI development).
- **Matplotlib** (Gantt chart visualization).
- **Pandas** (Optional for data handling).

Other Tools:

- **GitHub** (Version control).
- **VS Code** (IDE).

---

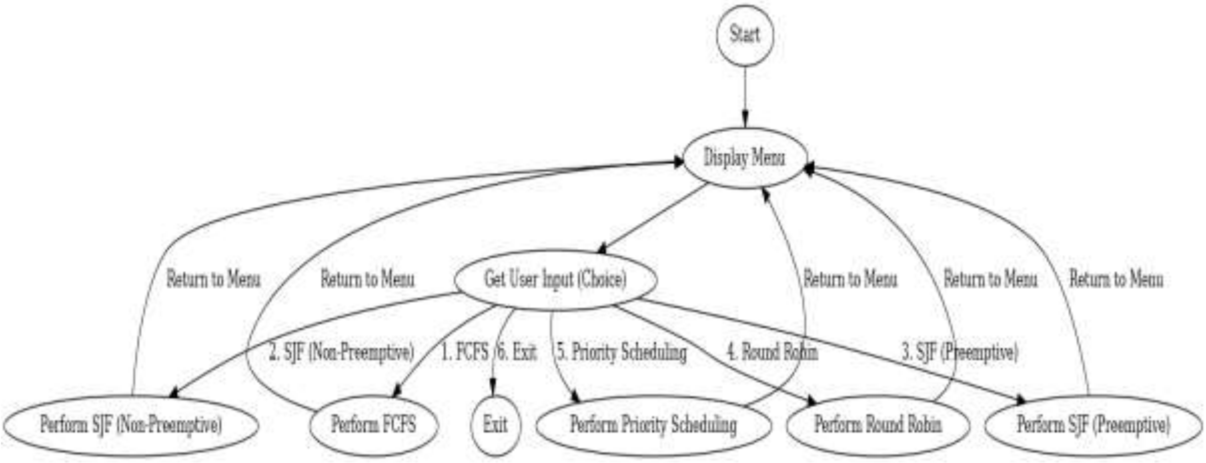
5. Flow Diagram

mermaid

Copy

```
graph TD
    A[Start] --> B[Input Process Details]
    B --> C[Select Algorithm]
    C --> D[Run Simulation]
    D --> E[Generate Gantt Chart]
    E --> F[Display Metrics]
```

F --> G[End]



---

6. Revision Tracking on GitHub

- **Repository Name:** cpu-scheduler-simulator
- **GitHub Link: https://github.com/akshayasahoo1/cpu-scheduler-simulator**

---

7. Conclusion and Future Scope

Conclusion:

- Successfully implemented a **CPU scheduling simulator** with visualization.
- Demonstrated how different algorithms impact performance metrics.

Future Scope:

- Add **preemptive SJF (SRTF)**.
- Integrate **ML-based adaptive scheduling**.
- Deploy as a **web app** using Flask/Django.

---

8. References Appendix

Core Algorithm References

A. Books

1. **Operating System Concepts (10th Ed.)**
   - ***Silberschatz, Galvin, Gagne***

  - Key Chapters: 5 (CPU Scheduling), 6 (Synchronization)
  - Publisher: Wiley
  - ISBN: 978-1119800361

2. Modern Operating Systems (4th Ed.)
   - *Andrew S. Tanenbaum*
   - Key Sections: 2.4 (Scheduling Algorithms)
   - Publisher: Pearson
   - ISBN: 978-0133591620

B. Academic Papers

- A Comparative Study of CPU Scheduling Algorithms
  - *P. Singh et al.*, International Journal of Computer Applications (2017)
  - DOI:10.5120/ijca2017913080

---

2. Python Implementation References

A. Official Documentation

1. Python tkinter (GUI)
   - Python Docs: tkinter
2. Matplotlib (Visualization)
   - Matplotlib Gantt Charts

B. Tutorials

- CPU Scheduling Simulator in Python
  - GeeksforGeeks Tutorial
  - Link

---

3. GitHub & Version Control

- GitHub Guides
  - Hello World GitHub Guide
  - Markdown Cheatsheet (for README)

- **Git Best Practices**
    - GitHub Flow

---

4. Additional Resources

A. Visualization Tools

- **Plotly (Interactive Charts)**
    - Plotly Python Docs

B. Sample Projects

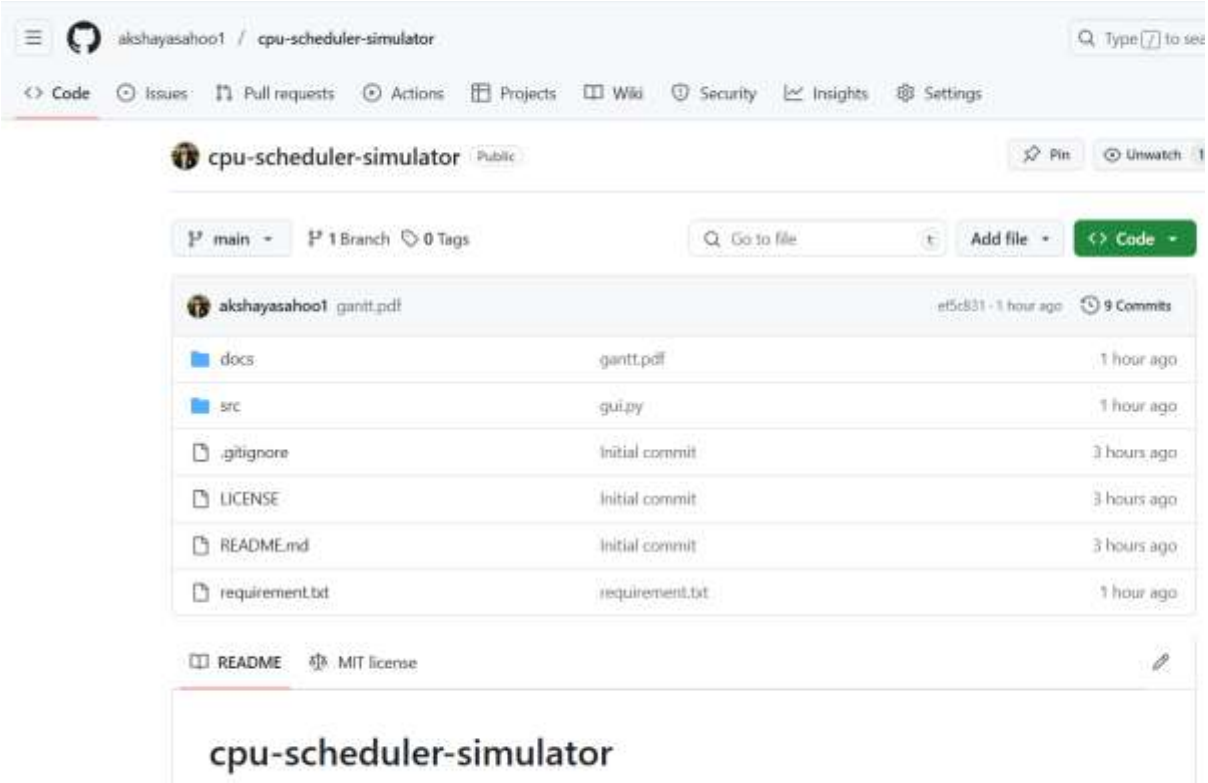
1. **OS-Simulator (GitHub)**
    - Link
    - **Includes FCFS, SJF, Round Robin implementations.**
2. **CPU-Scheduling-Algorithms**
    - Link

C. Solution/Code

python

Copy

```
import tkinter as tk
from tkinter import ttk
import matplotlib.pyplot as plt


# Sample FCFS Implementation
def fcfs(processes):
    processes.sort(key=lambda x: x['Arrival'])
    current_time = 0
    for p in processes:
        p['Completion'] = current_time + p['Burst']
        p['Turnaround'] = p['Completion'] - p['Arrival']
        p['Waiting'] = p['Turnaround'] - p['Burst']

        current_time = p['Completion']

    return processes


# GUI Code (Tkinter)

root = tk.Tk()

root.title("CPU Scheduler Simulator")

# Add input fields, buttons, and visualization logic here

root.mainloop()
```



docs	gantt.pdf	1 hour ago
src	gui.py	1 hour ago
.gitignore	Initial commit	3 hours ago
LICENSE	Initial commit	3 hours ago
README.md	Initial commit	3 hours ago
requirement.txt	requirement.txt	1 hour ago

cpu-scheduler-simulator