PROJECT REPORT

CPU SCHEDULING SIMULATOR

Presented By:

Pradeep Singh

B.TECH (C.S.E) – 3rd year

(Roll No.: 2219247)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

GRAPHIC ERA UNIVERSITY, DEHRADUN

GRAPHIC ERA HILL UNVERSITY, DEHRADUN

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

<u>CERTIFICATE</u>

It is certified that PRADEEP SINGH (Roll No.- 2219247) has developed mini project on "CPU Scheduling Simulator" for the CS V Semester Mini Project Lab in Graphic Era Hill University, Dehradun.

The research work was carried out by him under my (Dr. Vikrant Sharma) supervision in the academic year 2024-2025.

DATE- 11/01/2025

ACKNOWLEDGMENT

I would like to thank particularly my project Co-ordinator and my Project Guide Dr. Vikrant Sharma(CC SECTION B2) for his patience, support and encouragement throughout the completion of this project and having faith in us.
I would like to express my gratitude to my parents for their continuing support and encouragement. I also wish to thank them for providing me with the opportunity to reach this far in our studies.
At last, I am greatly indebted to all other the people who directly or indirectly helped me during this work.

Pradeep Singh
Class Roll No-44
UNIV Roll No.- 2219247
CSE-B2-V-Sem
Session: 2024-2025
GEHU, Dehradun

TABLE OF CONTENTS

1. INTRODUCTION:
- **1.1 Background**
- **1.2 Objective**
- **1.3 Scope**

2. LITERATURE SURVEY
- **2.1 Overview of CPU Scheduling**
- **2.2 Related Works**
- **2.3 Key Insights**

3. METHODOLOGY
- **3.1 System Design**
- **3.2 Features**
- **3.3 Implementation**
- **3.4 Tools and Libraries**

4. RESULTS AND DISCUSSION
- **4.1 Simulation Results**
- **4.2 User Experience**
- **4.3 Challenges**

5. CONCLUSION AND FUTURE WORK
- **5.1 Conclusion**
- **5.2References**

Introduction

1.1 Background

CPU scheduling is a critical aspect of modern operating systems. It involves allocating CPU time to various processes in a manner that optimizes performance metrics such as turnaround time, waiting time, and response time. With the increasing complexity of applications, selecting an efficient scheduling algorithm becomes essential to ensure optimal utilization of CPU resources.

1.3 Scope

The simulator supports the following algorithms:

1. First-Come, First-Served (FCFS)
2. Shortest Job First (SJF)
3. Shortest Remaining Time First (SRTF)
4. Priority Scheduling
5. Round Robin (RR)

Objective:

This project aims to develop a CPU Scheduling Simulator using Python and Tkinter. The simulator allows users to experiment with different scheduling algorithms and visualize their performance. The key objectives are:

•To understand the workings of common CPU scheduling algorithms.

•To provide an interactive and educational platform for simulating scheduling techniques.

Literature Survey:

2.1 Overview of CPU Scheduling

CPU scheduling has been widely studied in the field of operating systems. Early approaches, such as the FCFS algorithm, prioritize simplicity but may lead to inefficiencies like the convoy effect.

2.2 Related Works

Several studies and tools have explored CPU scheduling simulations:

1. **Educational Simulators**: Tools like Gantt chart generators provide visualization for scheduling algorithms but lack interactivity.

2. **Algorithm Analysis**: Research papers have analyzed the trade-offs between algorithms, focusing on metrics like throughput, waiting time, and fairness.

2.3 Key Insights

The literature highlights the importance of interactive tools in teaching scheduling algorithms. Existing simulators often lack customizability and real-time feedback, which this project aims to address.

Methodology

3.1 System Design

•**Frontend**: Built using Tkinter for GUI, enabling user interaction and data input.

•**Backend**: Algorithms implemented in Python, handling the scheduling logic.

3.2 Features

1.**User Input**: Dropdown menus for algorithm selection and process count.

2.**Dynamic Widgets**: Entry fields for arrival time, burst time, priority, and time quantum.

3.**Visualization**: Real-time display of scheduling processes and metrics

3.3 Implementation

•**Algorithm Modules**: Separate Python files for each algorithm (‘fcfs.py’, ‘sjf.py’, etc.).

•**Error Handling**: Validation to ensure correct and complete input data.

•**Interactivity**: Dynamically generated input fields based on user selections

.

3.4 Tools and Libraries

•**Python 3.x**

•**Tkinter** for GUI development

•**Pillow** for image handling

Results and Discussion

4.1 Simulation Results

The simulator effectively executes and visualizes all five scheduling algorithms. Key metrics such as average waiting time, turnaround time, and Gantt chart representation are displayed.

Example Output:

- **FCFS**:
    - Average Waiting Time: 10 ms
    - Average Turnaround Time: 20 ms
- **SJF**:
    - Average Waiting Time: 8 ms
    - Average Turnaround Time: 18 ms

4.2 User Experience

Feedback from users indicates that the application is intuitive and educational. The dynamic input fields and real-time simulation enhance learning outcomes.

4.3 Challenges

1. Handling erroneous input data.
2. Optimizing the GUI for varying screen resolutions.

Conclusion and Future Work

5.1 Conclusion

The CPU Scheduling Simulator successfully provides an interactive platform to learn and experiment with scheduling algorithms. It meets its objectives by offering a user-friendly interface and accurate simulation results.

5.2 Future Work

1. **Advanced Algorithms**: Adding support for multi-level queue scheduling and multi-core systems.
2. **Performance Metrics**: Detailed comparison charts for various algorithms.
3. **Web-Based Platform**: Deploying the simulator as a web application for wider accessibility.
4. **Visualization Enhancements**: Including animated Gantt charts.

References

1.Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts*. Wiley

2.Tanenbaum, A. S., & Bos, H. (2014). *Modern Operating Systems*. Pearson.

3.Research papers on CPU scheduling algorithms and their educational tools.

4.Youtube Videos