Round Robin CPU Scheduler
User Guide

Operating System - DA Case Study - I

Student Details:
Name: Vaidant Sharma
Roll No: 23BKT0016
Course: Operating System
DA Case Study - I
Course Code: BCSE303L
Class Number: VL2025260101320

Links:
GitHub: https://github.com/VaidantSharma/RoundRobin_Visualizer
Live Demo: https://roundrobinvisualizer-woad.vercel.app/

November 2, 2025

1 Setup Instructions

1.1 Prerequisites

- **Node.js**: Version 18.0.0 or higher
- **npm**: Version 9.0.0 or higher
- **Modern Web Browser**: Chrome 120+, Firefox 121+, Edge 120+, or Safari 17.0+

1.2 Installation

1. Clone the repository: `git clone https://github.com/VaidantSharma/RoundRobin_Visualizer.git`
2. Navigate to project directory: `cd RoundRobin_Visualizer`
3. Install dependencies: `npm install`
4. Start development server: `npm run dev`
5. Open browser: `http://localhost:3000`

Alternative: Access the live demo at roundrobinvisualizer-woad.vercel.app without installation.

2 User Interface Guide

2.1 Process Configuration

- **Add Process**: Click "Add Process" to create new processes (default: burst=5, arrival=0)
- **Burst Time**: Total CPU time required (minimum: 1)
- **Arrival Time**: Time unit when process arrives (minimum: 0)
- **Remove**: Click trash icon to delete a process
- **Time Quantum**: Maximum CPU time per process before preemption (default: 4)
- **Speed**: Animation speed slider (1x to 100x)

2.2 Controls

- **Start/Pause**: Toggle simulation execution
- **Reset**: Restore initial state and clear all data
- **Real-time Info**: Displays current time, context switches, executing process, quantum left, and remaining time

2.3 Visualization Areas

1. **Process Configuration Panel**: Add/configure processes and system parameters
2. **Control Panel**: Simulation controls and real-time statistics
3. **Process Flow Animation**: Visual representation of process lifecycle
4. **Gantt Chart**: Timeline view of process execution sequence
5. **Performance Metrics**: Detailed statistics table with averages

3 Animation Features

3.1 Color Coding

Each process has a unique color: Blue (P1), Green (P2), Orange (P3), Red (P4), Purple (P5), Pink (P6), Cyan (P7), Lime (P8). Colors are consistent across all visualizations.

3.2 Visualization Flow

The application uses a top-to-bottom flow:

1. **New Arrival** (Purple theme): Processes appear at their arrival time
2. **Ready Queue** (Blue theme): FIFO queue of processes waiting for CPU
3. **CPU Execution** (Yellow theme): Currently executing process with pulsing animation
4. **Completed** (Green theme): Finished processes with checkmark symbol
5. **Return to Queue** (Orange): Preempted processes returning to ready queue

3.3 Animation Transitions

- **Slide In Top**: New processes arriving (purple border)
- **Move Down Row**: Moving to queue/CPU/completed sections
- **Pulse**: Continuous animation for executing process
- **Move Up To Queue**: Preempted process returning to queue (orange border)

3.4 Performance Metrics

The application calculates:

- **Turnaround Time** = Completion Time - Arrival Time
- **Waiting Time** = Turnaround Time - Burst Time
- **Average Turnaround Time** and **Average Waiting Time**
- **Total Context Switches**

Metrics are displayed in a table with per-process details and aggregate summary cards.

4 Browser Requirements

4.1 Supported Browsers

Browser	Minimum Version
Google Chrome	120+
Mozilla Firefox	121+
Microsoft Edge	120+
Safari (macOS/iOS)	17.0+
Opera	106+

4.2 Required Features

- JavaScript enabled
- CSS Grid, Flexbox, and Animations support
- ES6+ JavaScript features
- React 19 compatibility

4.3 Compatibility Notes

- Internet Explorer is **not supported**
- Mobile browsers supported on modern devices
- Responsive design supports screens from 320px width
- Update browser if experiencing issues

For more information, visit: GitHub Repository