**A SYNOPSIS ON**



First Come First Served



**Submitted in partial fulfilment of the requirement for the award of the degree of**

**BACHELOR OF COMPUTER APPLICATION**

**Submitted by:**

**Student Name 1 - Pranjal Nath Goswami** **University Roll No. 2371289**

**Student Name 2 - Diya Bisht** **University Roll No.2371101**

**Student Name 3 - Karan Singh**  **University Roll No.2371171**

**Student Name4 – Pradeep Singh bora University Roll No.2371283**

***Under the Guidance of***

***Supervisor Name***

***Designation***

**Project Team ID: ID No.**



# School of Computing

**Graphic Era Hill University, Bhimtal, Uttarakhand**

**March-2025**



**CANDIDATE’S DECLARATION**

I/We hereby certify that the work which is being presented in the Synopsis entitled **“Title of the project”** in partial fulfilment of the requirements for the award of the Degree of Bachelor Computer Application of the Graphic Era Hill University, Bhimtal shall be carried out by the undersigned under the supervision of **Guide Name, Designation**, School of Computing, Graphic Era Hill University, Bhimtal.

Name1-Pranjal Nath Goswami University Roll no1-2371289 signature

Name2- Diya Bisht University Roll no2-2371101 signature

Name3-Karan Singh University Rollno3-2371171 signature

Name4- Pradeep Singh bora University Roll no4-2371283 signature

The above mentioned students shall be working under the supervision of the undersigned on the **“Title of the project”**

Signature Signature

**Supervisor** **Head of the Department**

**Internal Evaluation (By DPRC Committee)**

**Status of the Synopsis:** Accepted / Rejected

**Any Comments:**

**Name of the Committee Members: Signature with Date**

# Table of Contents

|  |  |  |
| --- | --- | --- |
| Chapter no. | Description | PageNo. |
|  |  |  |
| 1. | Project Overview, Technology Stack (Features) | 1. page |
|  |  |  |
| 2. | System Architecture | 2.page |
|  |  |  |
| 3. | Key Calculation (CT,TAT,WT) | 3. page |
|  |  |  |
| 4. | Technical Implementation  (backend architecture,frontend component,testing) | 4.page |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**First Come First Serve (FCFS) CPU Scheduling Algorithm Implementation**

## Project Overview

A sophisticated implementation of the First Come First Serve (FCFS) CPU scheduling algorithm with a modern web interface. This project demonstrates efficient process management and real-time scheduling visualization.

## Technology Stack

* **Backend**: C++ with Crow Framework
* **Frontend**: HTML5, CSS3, JavaScript
* **Build System**: CMake
* **Version Control**: Git
* **Documentation**: Markdown

## Features

* **Interactive Process Management** o Dynamic process addition and removal o Real-time scheduling visualization o Intuitive user interface
* **Advanced Scheduling Metrics** o Completion Time calculation o Turnaround Time analysis o Waiting Time optimization
* **Real-time Visualization** o Gantt chart representation o Process timeline display o Performance metrics dashboard
* **Error Handling & Validation** o Input validation o Process limit enforcement o Error message display

## o System Architecture

┌─────────────────┐ ┌─────────────────┐ ┌─────────────────┐

│ Frontend │ │ Backend │ │ FCFS │

│ (HTML/CSS/JS) │◄───►│ (C++/Crow) │◄───►│ Algorithm │

└─────────────────┘ └─────────────────┘ └─────────────────┘

**Getting Started**

## Prerequisites

* C++ compiler (C++17 or later)
* CMake (version 3.10 or later)
* Git
* Modern web browser

## Installation

1. Clone the repository:

git clone https://github.com/PranjalNG/OS-project.git cd OS-project 2. Build the project:

mkdir build cd build cmake .. cmake --build .

1. Run the application:

./fcfs\_scheduler

1. Access the web interface:

o Open your browser o Navigate to http://localhost:5000 **Adding Processes**

1. Enter process details:

o Process ID (unique identifier) o Arrival Time (when process enters the system) o Burst Time (CPU time required)

1. Click "Add Process" to include in scheduling queue

## Viewing Results

* **Gantt Chart**: Visual representation of process execution
* **Timeline**: Detailed process execution sequence
* **Metrics**: Performance analysis including:

o Completion Time o Turnaround Time o Waiting Time

**Performance Metrics**

## Key Calculations

1. **Completion Time (CT)** o Time when process finishes execution o Formula: CT = Previous Process CT + Current Process BT
2. **Turnaround Time (TAT)** o Total time from arrival to completion o Formula: TAT = CT - AT
3. **Waiting Time (WT)** o Time process waits in ready queue o Formula: WT = TAT - BT

## Average Metrics

* Average Turnaround Time = Σ(TAT) / n
* Average Waiting Time = Σ(WT) / n

## Technical Implementation 1.Backend Architecture struct Process {

int pid; int arrival\_time; int burst\_time; int completion\_time; int turnaround\_time; int waiting\_time;

};

## 2.Frontend Components

* Process input form
* Real-time visualization
* Results display panel
* Error handling interface

## 2.Testing

* Unit tests for FCFS algorithm
* Integration tests for API endpoints
* UI/UX testing for web interface
* Performance benchmarking

## Future Enhancements

**1.Algorithm Extensions** o Priority scheduling o Round Robin implementation o Multi-level queue scheduling

**2.Un Improvements**

Dark mode support Responsive design optimizatio