# OPSYS TOOLKIT

A MINI-PROJECT REPORT
Submitted by

**RENUGA DEVI K 231901041**
**PRIYANKA E 231901038**

**in partial fulfilment of the award of the degree of**

**BACHELOR OF ENGINEERING**
**IN**
**COMPUTER SCIENCE AND ENGINEERING**
**(CYBER SECURITY)**

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**An Autonomous Institute**

**CHENNAI**

**MAY 2025**

## BONAFIDE CERTIFICATE

Certified that this project **“OPSYS TOOLKIT”** is the bonafide work of **“ RENUGA DEVI K & PRIYANKA E”** who carried out the project work under my supervision.

**SIGNATURE**
**Mrs. V. JANANEE**
**ASSISTANT PROFESSOR**

Dept. of Computer Science and Engg,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

__________

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

## ABSTRACT:

**OPSYS TOOLKIT: Visualize, Learn, Simulate** is an innovative mobile application designed to revolutionize how Operating System concepts are understood and practiced. Built using Flutter, this offline-friendly app transforms core OS topics—CPU Scheduling, Memory Allocation, Deadlock Detection, and Page Replacement Algorithms—into interactive playground for learners. Unlike traditional learning tools, our app empowers users to simulate real-time scenarios, visualize complex processes, and grasp abstract algorithms through dynamic Gantt charts, memory bar graphs, resource allocation graphs, and page frame visualizations. Users can experiment with various algorithms including FCFS, SJF, Round Robin, Banker's Algorithm, and page replacement strategies like FIFO, LRU, and Optimal, observing their behaviour and outcomes instantly. Whether you're a Student preparing for exams or a curious mind exploring system-level operations, OS Toolkit offers a hands-on, visual-first approach that makes learning not just easier—but exciting. This project bridges the gap between theory and application, helping users move from memorization to mastery.

# ACKNOWLEDGEMENT

We express our sincere thanks to our honourable chairman **Mr. S. MEGANATHAN** and the chairperson **Dr. M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honourable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head of the Department **Mr. BENEDICT JAYAPRAKASH NICHOLAS M.E Ph.D.,** for being ever supporting force during our project work.

We also extend our sincere and hearty thanks to our internal guide **Mrs**. **V JANANEE,** for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of Computer Science and Engineering.

**1. RENUGA DEVI K**

**2. PRIYANKA E**

**TABLE OF CONTENTS**


| CHAPTER NO | TITLE | PAGE NO |
|---|---|---|
| **1** | **INTRODUCTION** | **6** |
| **1.1** | Introduction | **6** |
| **1.2** | Scope of the Work | **6** |
| **1.3** | Problem Statement | **7** |
| **1.4** | Aim and Objectives of the Project | **7** |
| **2** | **SYSTEM DESIGN AND FEATURES** | **8** |
| **2.1** | Cpu scheduling | **8** |
| **2.2** | Memory Allocation | **8** |
| **2.3** | Deadlock Detection | **8** |
| 2.4 | Page Replacement | **8** |
| **3** | **ARCHITECTURE** | **9** |
| **4** | **FLOW CHART** | **10** |
| **5** | **CODE IMPLEMENTATION** | **11** |
| **6** | **SCREENSHOTS** | **12** |
| **7** | **CONCLUSION** | **14** |
| **8** | **FUTURE WORK** | **14** |
| **9** | **REFERENCES** | **15** |

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION

OPSYS Toolkit is a Flutter-based mobile application that provides interactive simulations and visualizations for key Operating System concepts. It focuses on enhancing learning and practical understanding of CPU Scheduling, Memory Allocation, Deadlock Detection, and Page Replacement algorithms. The app supports offline usage and provides visual representations through Gantt charts, memory bar graphs, resource allocation graphs, and page frame layouts.

## 1.2 SCOPE OF THE WORK

The scope of OPSYS Toolkit lies in offering a visual-first, hands-on simulation platform for learners and educators. It allows experimentation with multiple scheduling and memory strategies in real-time, catering to academic, experimental, and research needs.

## 1.3 PROBLEM STATEMENT

Operating System algorithms are conceptually rich but often difficult to visualize. Many learners struggle with understanding how abstract algorithms work behind the scenes. OPSYS Toolkit addresses this issue by offering a mobile-friendly, graphical simulation environment that bridges the gap between theory and practical understanding.

## 1.4 AIM AND OBJECTIVES OF THE PROJECT

The aim is to design a self-contained mobile app that enables interactive learning and analysis of Operating System concepts. The objectives include:

- Implementing multiple CPU scheduling algorithms (FCFS, SJF, RR)
- Visualizing memory allocation techniques
- Demonstrating deadlock detection using Banker's Algorithm
- Simulating page replacement strategies like FIFO, LRU, and Optimal
- Presenting results in user-friendly visual formats

# CHAPTER 2

# SYSTEM DESIGN AND FEATURES

## 2.1 CPU SCHEDULING

Supports FCFS, SJF, and Round Robin algorithms. Users can input arrival and burst times, view Gantt chart representations, and analyze average turnaround and waiting times.

## 2.2 MEMORY ALLOCATION

Visualizes memory usage and partitioning using bar graphs. Helps in understanding fragmentation and memory fit strategies.

## 2.3 DEADLOCK DETECTION

Implements Banker's Algorithm to simulate safe state checks. Uses resource allocation graphs to visually identify safe and unsafe states.

## 2.4 PAGE REPLACEMENT

Simulates FIFO, LRU, and Optimal algorithms with step-by-step frame changes and hit/miss count to understand paging mechanisms.

# CHAPTER 3

# ARCHITECTURE

The architecture of OPSYS Toolkit consists of:

- Input Interface: Flutter UI for accepting user inputs (processes, memory size, priorities, etc.)

- Logic Engine: Dart logic to simulate algorithms (scheduling, memory, deadlock, paging)

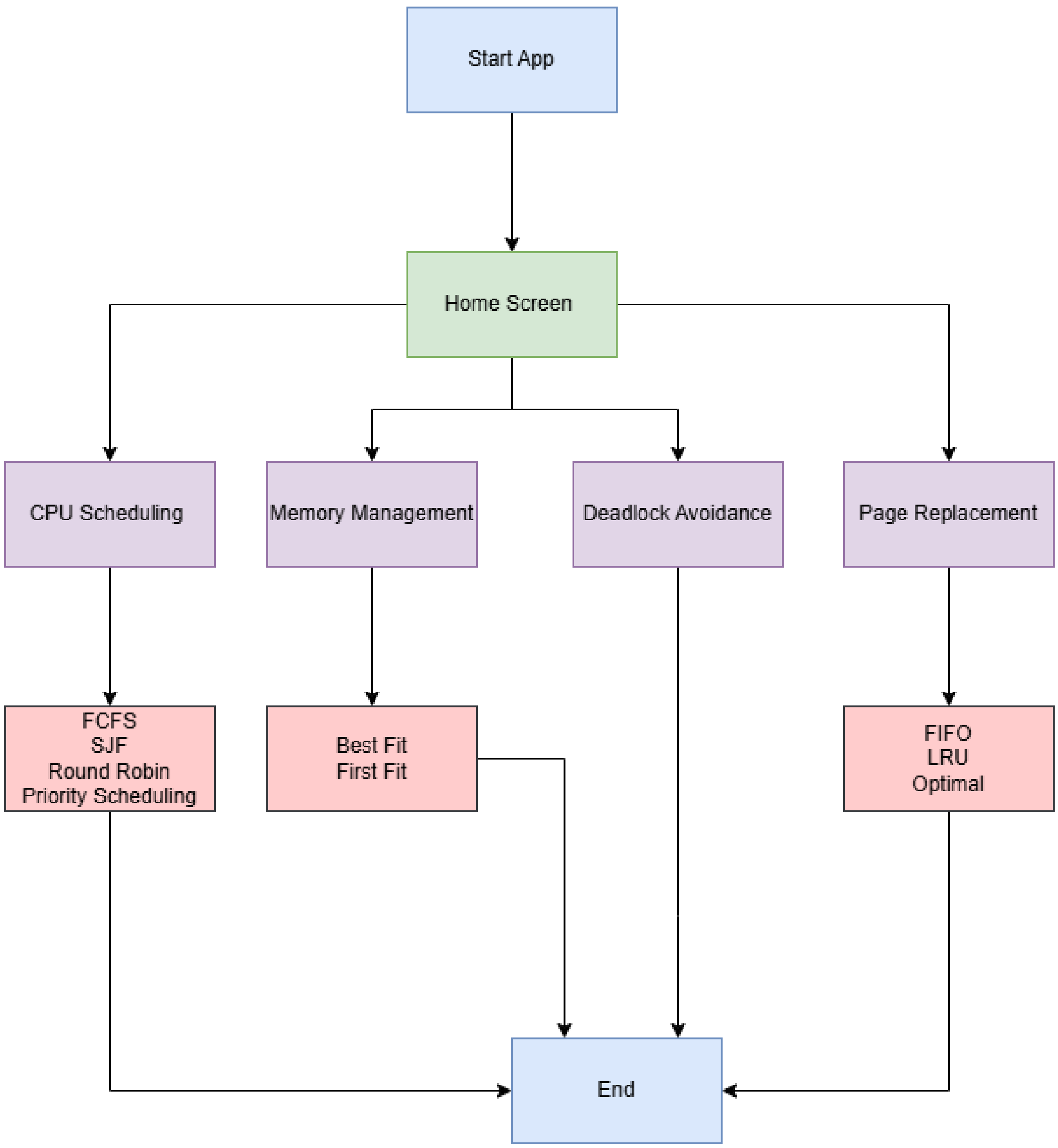
- Visualization Layers: Charts and graphs (Gantt chart, bar graphs, resource allocation graphs) for each module

**Output Summary**: Shows turnaround time, waiting time, page faults, and memory stats

All modules are encapsulated within a mobile app architecture supporting offline use with responsive performance.

# CHAPTER 4

# FLOWCHART



# CHAPTER 5
# CODE IMPLEMENTATION

The OPSYS Toolkit is built using Flutter (Dart) and follows modular implementation:

CPU Scheduling: Separate functions for FCFS, SJF, RR using list sorting and time tracking

Memory Allocation: Uses dynamic list structures to simulate block allocation and visualize fragmentation

Deadlock Detection: Implements Banker's Algorithm with matrix input and safe state detection

Page Replacement: Array and queue-based logic for FIFO, LRU, Optimal page simulation

Visualization is done using Flutter charts, and all results are displayed in-app in an intuitive format.

# CHAPTER 6

# SCREENSHOTS

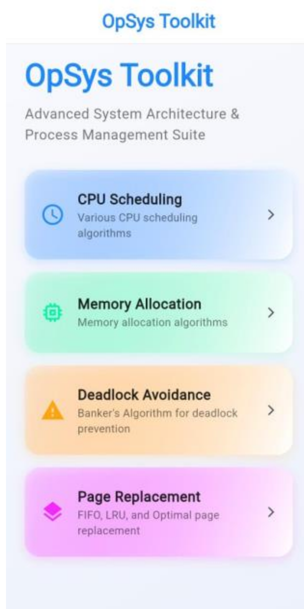



Fig 1. App Introduction Page

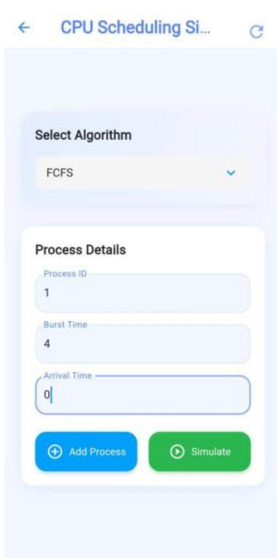



Fig 2. Input Section for CPU Scheduling

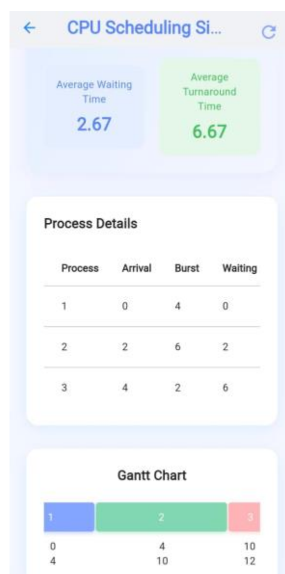



Fig 3. Gantt Chart Output for FCFS

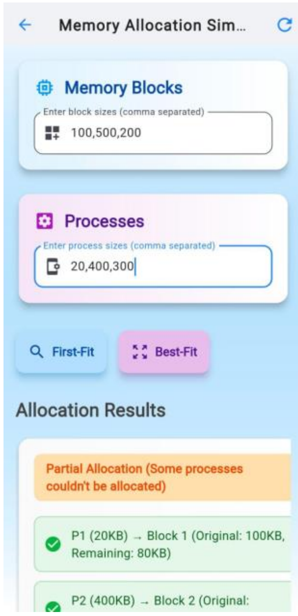



Fig 4. Memory Allocation Visualization


Fig 5. Deadlock Resource Graph


Fig 6. Page Replacement Frame Changes

# CHAPTER 7

## CONCLUSION

OPSYS Toolkit successfully bridges the gap between theoretical learning and practical application of Operating System concepts. It makes abstract algorithms interactive and visual, helping learners grasp critical ideas through simulation. The offline, mobile-friendly design adds accessibility and convenience.

# CHAPTER 8

## FUTURE WORK

- Add Multilevel Queue and Feedback Scheduling
- Real-time process monitoring through OS APIs
- Export results to PDF or Excel
- Add quiz or challenge modes for gamified learning
- Introduce multiprocessor simulation and visual comparison
- Support dark mode and accessibility features

# CHAPTER 9

# REFERENCES


1. Hassan, R., & Patel, A. (2020, November). "Enhancing CPU Scheduling Algorithms for Real-Time Systems." In 2020 IEEE International Conference on Operating Systems and Performance Optimization (pp. 124-128). IEEE.

2. Khan, M., & Sharma, S. (2019, June). "A Comparative Study of Memory Allocation Strategies in Modern Operating Systems." In 2019 IEEE International Conference on System Architecture and Software Engineering (pp. 56-60). IEEE.

3. Wang, L., & Zhang, Y. (2021, March). "Page Replacement Algorithms for Memory Management in Virtualized Environments." In 2021 IEEE Symposium on Computer and Network Systems (pp. 204-208). IEEE.

4. Mehta, S., & Agarwal, S. (2018, July). "Implementation and Optimization of Banker's Algorithm for Deadlock Avoidance." In 2018 IEEE International Conference on Algorithms and Systems (pp. 99-103). IEEE.

5. Patel, R., & Kumar, R. (2022, May). "A Review of Preemptive and Non-Preemptive CPU Scheduling Algorithms." In 2022 IEEE International Conference on Software Engineering and Systems (pp. 157-161). IEEE.

6. Singh, P., & Gupta, M. (2020). "Design and Implementation of Best Fit Memory Allocation Strategy in Operating Systems." IEEE Access, 8, 4079-4090.

7. Liu, Z., & Zhang, T. (2019, August). "Optimizing Resource Allocation with Round Robin Scheduling in OS." In 2019 IEEE International Conference on Computing and Information Technology (pp. 78-82). IEEE.