# Report on

**"Paging Simulation"**

For

**Operating System - 2CS506CC23**

**B. Tech. Semester IV**

## Prepared By :

Gandhi Urva Yogeshkumar (23BCE078)

Kahan Dave (23BTM015)

**Institute of Technology,**

**Nirma University**

**Ahmedabad - 382481**

**March 2025**

# Index

Urva Gandhi - 23BCE078
Kahan Dave - 23BTM015

# Chapter 1. Objectives

## Paging Simulation

**Problem Statement**
Paging is a **memory management** scheme that eliminates the need for contiguous memory allocation by dividing processes into fixed-size pages and mapping them to frames in physical memory. In operating systems, efficient paging mechanisms help in optimal memory utilization, reducing fragmentation, and improving process execution efficiency.

However, challenges like page faults and inefficient replacement strategies persist. This project simulates paging mechanics and implements Page Replacement Algorithms (PRA) to optimize memory usage.

---

**System Overview :**
The Paging Simulation Portal is an interactive web-based tool that:

- Demonstrate the concept of **virtual memory paging** in operating systems.
- Visualizes address translation, page faults, and replacement steps.
- Implements FIFO, LRU, and Optimal PRA techniques.
- Generates statistics (e.g., page fault rate, memory usage).

# Chapter 2. Functionalities

## Core Components:

1. **Configuration Management**
    - **Set frame size (multiples of 1024 bytes).**
    - **Define total virtual pages and physical frames.**
    - **Initialize page-to-frame mappings (e.g., `0:2, 1:1`).**
    - **Real-time validation of inputs and mappings.**
2. **Virtual Memory Table**:
    - Tracks virtual pages and their corresponding frame allocations.
    - Attributes:
        - Virtual Page Number
        - Frame Number (or **'X'** if unallocated).
3. **Physical Memory Table**:
    - Displays frame numbers and their mapped physical addresses.
    - Attributes:
        - Frame Number
        - Physical Address Range.
4. **Page Replacement Algorithms (PRA)**:
    - **FIFO**: Replaces the oldest page in memory.
    - **LRU**: Replaces the least recently used page.
    - **Optimal**: Replaces the page used farthest in the future.

## User Interaction:

1. **Input**: Virtual address (decimal/binary).
2. **Output**:
    - Physical address (if mapped).
    - Page fault notification with PRA selection.
3. **Visualization**:
    - Highlighted memory tables.
    - Step-by-step address translation (binary/decimal).
    - Algorithm-specific explanations (e.g., access history for LRU).

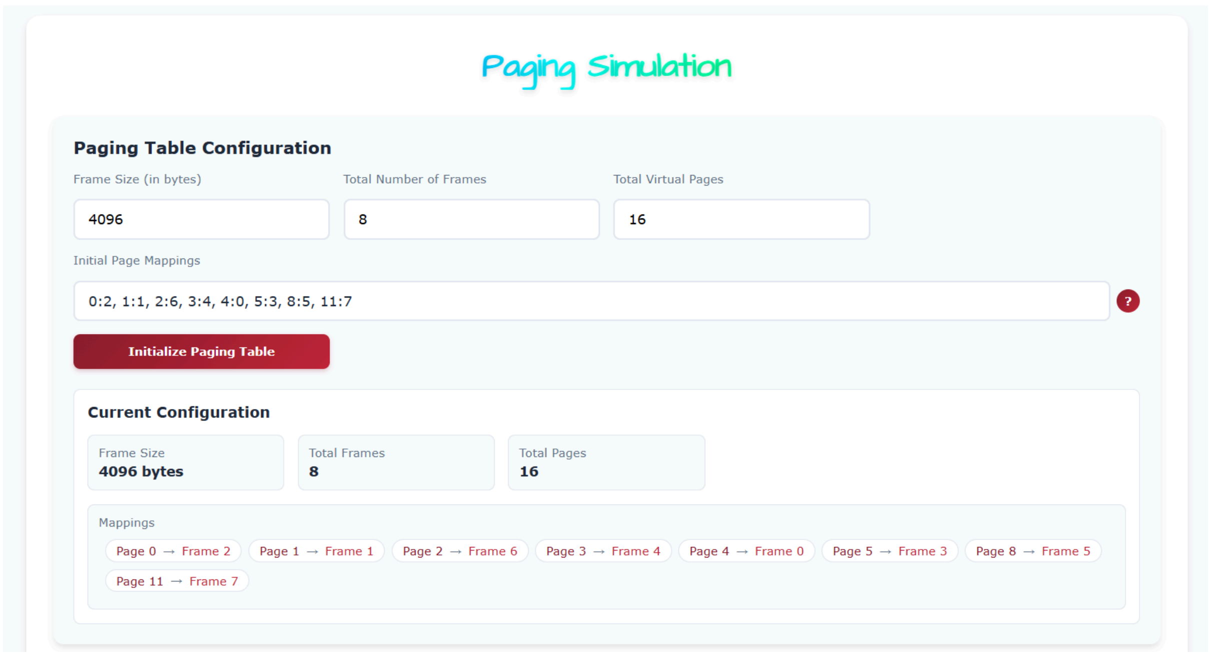
# Chapter 3 . Code

**Technologies Used**: HTML, CSS, JavaScript.

**Key Components**

1. **HTML Structure**
   - Configuration panel for frame/page setup.
   - Input section for addresses and format toggles.
   - Tables for virtual/physical memory display.
   - Dynamic sections for PRA visualization and calculations.
2. **CSS Styling**
   - Themed UI with gradients and shadows.
   - Responsive grids for tables and memory states.
   - Animations for notifications (e.g., page fault alerts).
3. **JavaScript Logic**
   - **`initializePagingTable()`**: Validates inputs and initializes the page table.
   - **`processAddress()`**: Computes page number/offset, checks for page faults.
   - **`executePRA()`**: Implements replacement algorithms and updates the page table.
   - **Dynamic DOM Updates**: Tables, calculations, and visualizations refresh based on user actions.
   - **Event Listeners**: Handle input changes, button clicks, and format toggles.

# Chapter 4 . Output



Paging Simulation

Paging Table Configuration

Frame Size (in bytes): 4096

Total Number of Frames: 8

Total Virtual Pages: 16

Initial Page Mappings: 0:2, 1:1, 2:6, 3:4, 4:0, 5:3, 8:5, 11:7

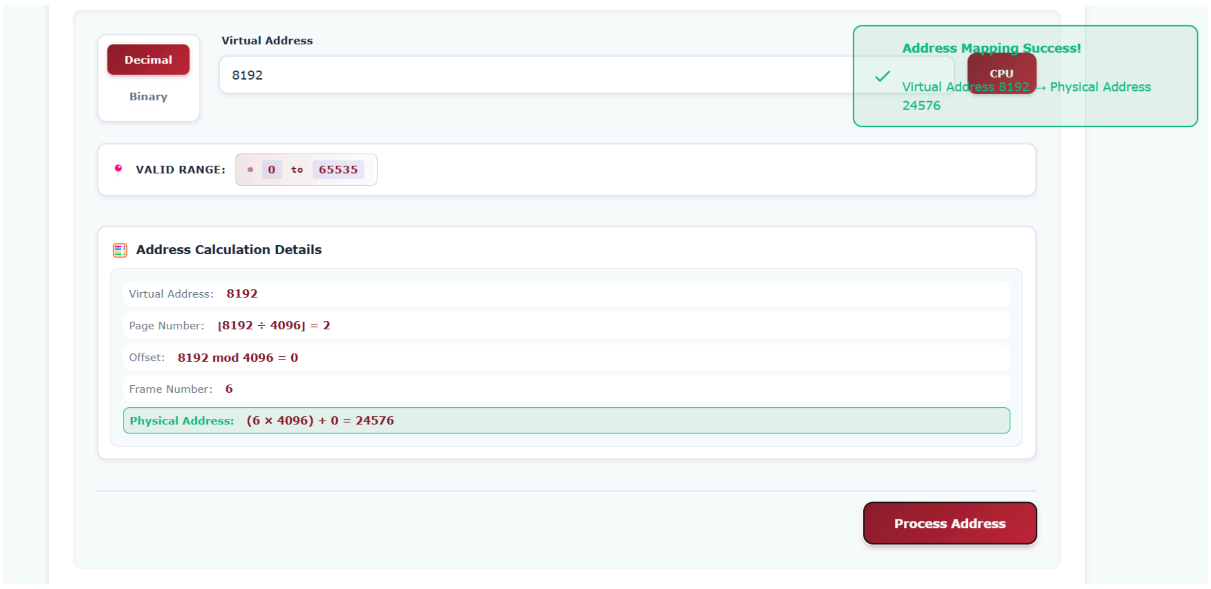
Current Configuration

Frame Size: 4096 bytes

Total Frames: 8

Total Pages: 16

Mappings: Page 0 → Frame 2, Page 1 → Frame 1, Page 2 → Frame 6, Page 3 → Frame 4, Page 4 → Frame 0, Page 5 → Frame 3, Page 8 → Frame 5, Page 11 → Frame 7



Virtual Address: 8192

Address Mapping Success!
✓ Virtual Address 8192 → Physical Address 24576

VALID RANGE: 0 to 65535

Address Calculation Details

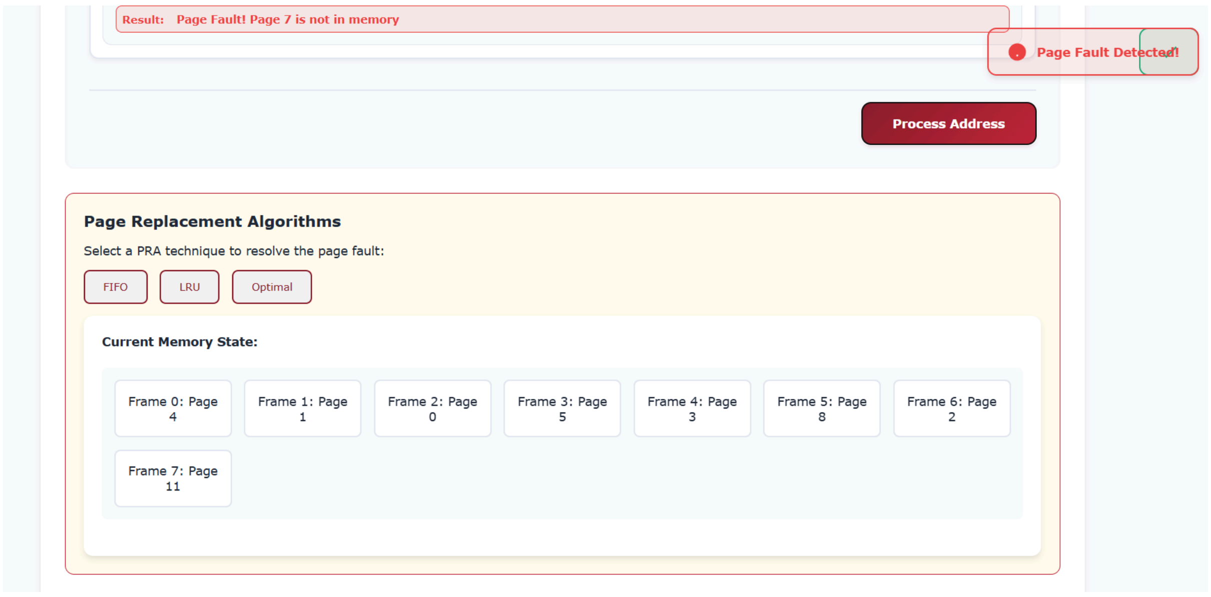
Virtual Address: 8192

Page Number: ⌊8192 ÷ 4096⌋ = 2

Offset: 8192 mod 4096 = 0

Frame Number: 6

Physical Address: (6 × 4096) + 0 = 24576



Result: Page Fault! Page 7 is not in memory

Page Fault Detected!

Process Address

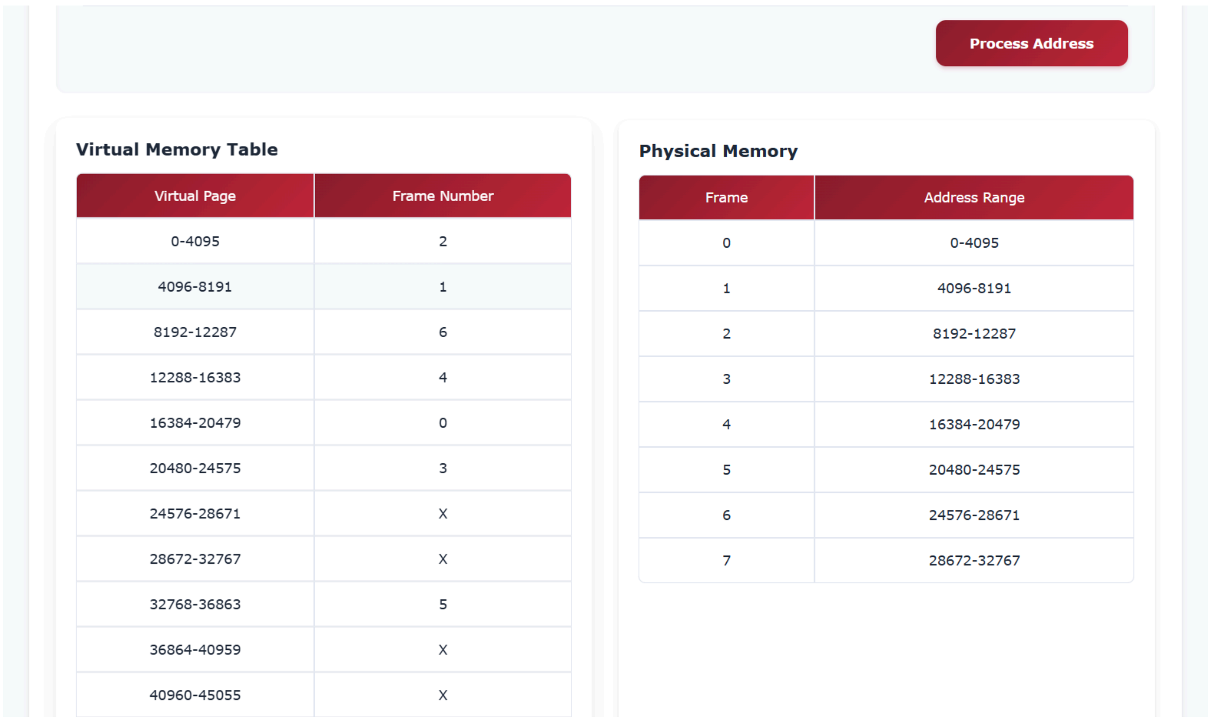
**Page Replacement Algorithms**

Select a PRA technique to resolve the page fault:

FIFO LRU Optimal

**Current Memory State:**

Frame 0: Page 4
Frame 1: Page 1
Frame 2: Page 0
Frame 3: Page 5
Frame 4: Page 3
Frame 5: Page 8
Frame 6: Page 2
Frame 7: Page 11

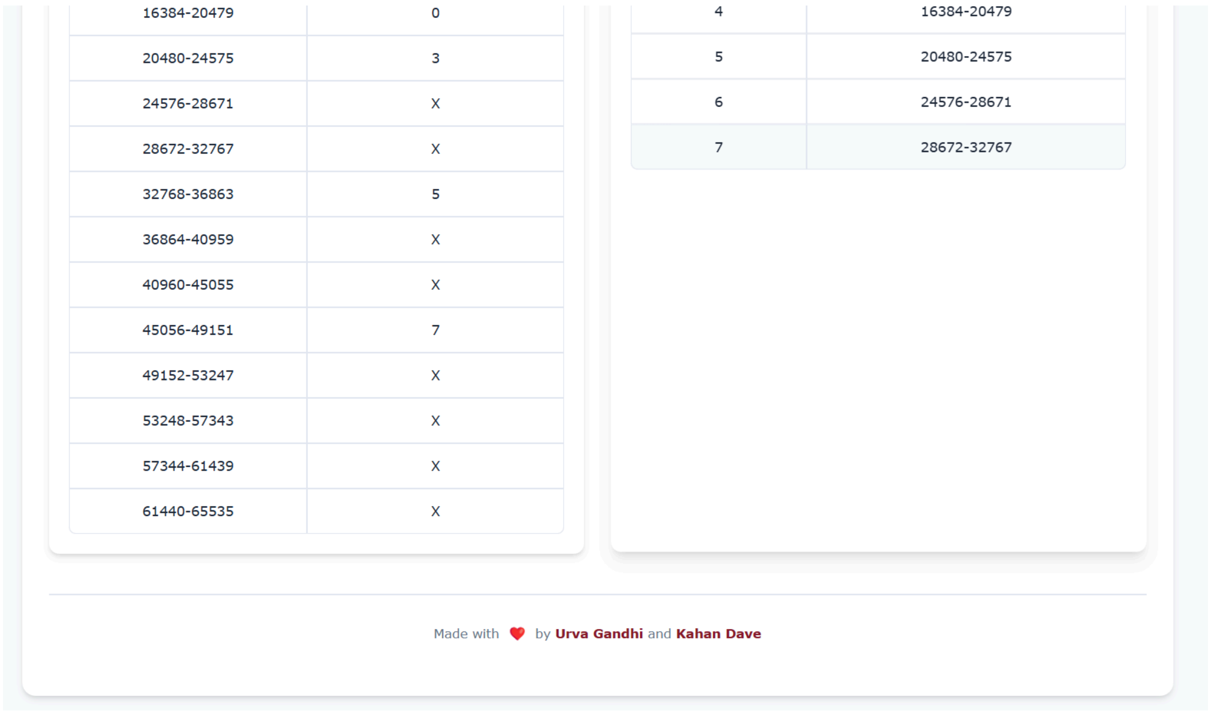


### Virtual Memory Table

| Virtual Page | Frame Number |
|---|---|
| 0-4095 | 2 |
| 4096-8191 | 1 |
| 8192-12287 | 6 |
| 12288-16383 | 4 |
| 16384-20479 | 0 |
| 20480-24575 | 3 |
| 24576-28671 | X |
| 28672-32767 | X |
| 32768-36863 | 5 |
| 36864-40959 | X |
| 40960-45055 | X |

### Physical Memory

| Frame | Address Range |
|---|---|
| 0 | 0-4095 |
| 1 | 4096-8191 |
| 2 | 8192-12287 |
| 3 | 12288-16383 |
| 4 | 16384-20479 |
| 5 | 20480-24575 |
| 6 | 24576-28671 |
| 7 | 28672-32767 |



| | |
|---|---|
| 16384-20479 | 0 |
| 20480-24575 | 3 |
| 24576-28671 | X |
| 28672-32767 | X |
| 32768-36863 | 5 |
| 36864-40959 | X |
| 40960-45055 | X |
| 45056-49151 | 7 |
| 49152-53247 | X |
| 53248-57343 | X |
| 57344-61439 | X |
| 61440-65535 | X |

| | |
|---|---|
| 4 | 16384-20479 |
| 5 | 20480-24575 |
| 6 | 24576-28671 |
| 7 | 28672-32767 |

Made with ❤️ by **Urva Gandhi** and **Kahan Dave**

# Chapter 5. Conclusion

The Paging Simulation successfully demonstrates:

1. **Efficient Address Translation**: Seamless conversion of virtual to physical addresses.
2. **PRA Effectiveness**:
    - **FIFO** is simple but may increase faults.
    - **LRU** balances recency and frequency.
    - **Optimal** (theoretical) minimizes faults but requires future knowledge.
3. **User Engagement**: Interactive visualizations for understanding paging.