

# Operating System Course Report - First Half of the Semester

B class

October 1, 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Course Overview</b>	<b>4</b>
2.1	Objectives . . . . .	4
2.2	Course Structure . . . . .	4
<b>3</b>	<b>Topics Covered</b>	<b>5</b>
3.1	Basic Concepts and Components of Computer Systems . . . . .	5
3.2	System Performance and Metrics . . . . .	5
3.3	System Architecture of Computer Systems . . . . .	5
3.4	Process Description and Control . . . . .	5
3.5	Scheduling Algorithms . . . . .	6
3.6	Process Creation and Termination . . . . .	6
3.7	Introduction to Threads . . . . .	6
3.8	File Systems . . . . .	6
3.9	Input and Output Management . . . . .	7
3.10	Deadlock Introduction and Prevention . . . . .	7
3.11	User Interface Management . . . . .	7
3.12	Virtualization in Operating Systems . . . . .	7
3.12.1	Pengertian Virtualisasi Sistem Operasi . . . . .	7
3.12.2	Jenis Jenis Virtualisasi pada Sistem Operasi . . . . .	7
3.12.3	Komponen Virtualisasi Sistem Operasi . . . . .	14
3.12.4	Hypervisor dan Virtual Machines . . . . .	15
3.12.5	Contoh Penerapan Virtualisasi pada Sistem Operasi . . . . .	15
3.12.6	Kelebihan dan Kekurangan Virtualisasi pada Sistem Operasi . . . . .	15
3.12.7	Kesimpulan . . . . .	15
<b>4</b>	<b>Assignments and Practical Work</b>	<b>16</b>
4.1	Assignment 1: Process Scheduling . . . . .	16
4.2	Assignment 2: Deadlock Handling . . . . .	16
4.3	Assignment 3: Multithreading and Amdahl's Law . . . . .	17
4.4	Assignment 4: Simple Command-Line Interface (CLI) for User Interface Management . . . . .	17
4.5	Assignment 5: File System Access . . . . .	17



# 1 Introduction

This report summarizes the topics covered during the first half of the Operating System course. It includes theoretical concepts, practical implementations, and assignments. The course focuses on the fundamentals of operating systems, including system architecture, process management, CPU scheduling, and deadlock handling.

## 2 Course Overview

### 2.1 Objectives

The main objectives of this course are:

- To understand the basic components and architecture of a computer system.
- To learn process management, scheduling, and inter-process communication.
- To explore file systems, input/output management, and virtualization.
- To study the prevention and handling of deadlocks in operating systems.

### 2.2 Course Structure

The course is divided into two halves. This report focuses on the first half, which covers:

- Basic Concepts and Components of Computer Systems
- System Performance and Metrics
- System Architecture of Computer Systems
- Process Description and Control
- Scheduling Algorithms
- Process Creation and Termination

- Introduction to Threads
- File Systems
- Input and Output Management
- Deadlock Introduction and Prevention
- User Interface Management
- Virtualization in Operating Systems

## **3 Topics Covered**

### **3.1 Basic Concepts and Components of Computer Systems**

This section explains the fundamental components that make up a computer system, including the CPU, memory, storage, and input/output devices.

### **3.2 System Performance and Metrics**

This section introduces various system performance metrics used to measure the efficiency of a computer system, including throughput, response time, and utilization.

### **3.3 System Architecture of Computer Systems**

Describes the architecture of modern computer systems, focusing on the interaction between hardware and the operating system.

### **3.4 Process Description and Control**

Processes are a central concept in operating systems. This section covers:

- Process states and state transitions
- Process control block (PCB)
- Context switching

### **3.5 Scheduling Algorithms**

This section covers:

- First-Come, First-Served (FCFS)
- Shortest Job Next (SJN)
- Round Robin (RR)

It explains how these algorithms are used to allocate CPU time to processes.

### **3.6 Process Creation and Termination**

Details how processes are created and terminated by the operating system, including:

- Process spawning
- Process termination conditions

### **3.7 Introduction to Threads**

This section introduces the concept of threads and their relation to processes, covering:

- Single-threaded vs. multi-threaded processes
- Benefits of multithreading

### **3.8 File Systems**

File systems provide a way for the operating system to store, retrieve, and manage data. This section explains:

- File system structure
- File access methods
- Directory management

### **3.9 Input and Output Management**

Input and output management is key for handling the interaction between the system and external devices. This section includes:

- Device drivers
- I/O scheduling

### **3.10 Deadlock Introduction and Prevention**

Explores the concept of deadlocks and methods for preventing them:

- Deadlock conditions
- Deadlock prevention techniques

### **3.11 User Interface Management**

This section discusses the role of the operating system in managing the user interface. Topics covered include:

- Graphical User Interface (GUI)
- Command-Line Interface (CLI)
- Interaction between the user and the operating system

### **3.12 Virtualization in Operating Systems**

Virtualisasi memungkinkan beberapa sistem operasi berjalan secara bersamaan pada satu mesin fisik. Bagian ini akan membahasnya:

#### **3.12.1 Pengertian Virtualisasi Sistem Operasi**

#### **3.12.2 Jenis Jenis Virtualisasi pada Sistem Operasi**

##### **1. Full Virtualization**

Virtualisasi Penuh (Full Virtualization), atau yang juga dikenal sebagai Native Virtualization, adalah metode virtualisasi di mana mesin

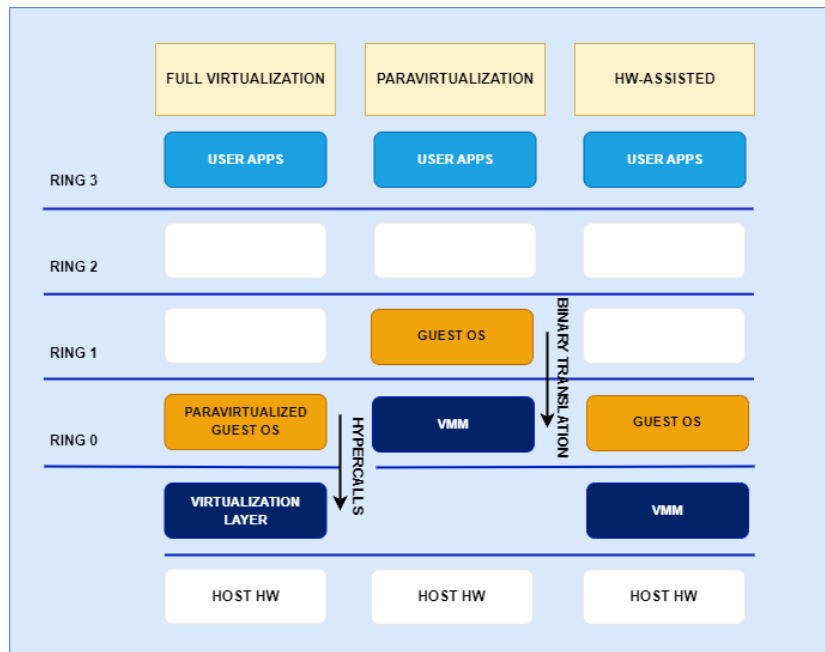


Figure 1: Type of Virtualization

virtual (VM) digunakan untuk menjembatani sistem operasi tamu dengan perangkat keras fisik yang ada di bawahnya. Proses menjembatani ini sangat penting dalam teknik ini karena VM bertindak sebagai perantara antara sistem operasi tamu dan perangkat keras. Beberapa instruksi terprivileg yang sensitif tidak dapat dijalankan langsung oleh sistem operasi tamu dan perlu diperangkap oleh hypervisor, yang kemudian menangani instruksi tersebut, karena perangkat keras diakses melalui hypervisor, bukan secara langsung oleh sistem operasi tamu.

Kinerja Virtualisasi Penuh lebih cepat dibandingkan dengan metode emulasi perangkat keras, namun tetap lebih lambat dibandingkan dengan akses langsung ke perangkat keras fisik. Hal ini terjadi karena hypervisor harus berfungsi sebagai media penghubung yang menyebabkan adanya overhead dalam komunikasi antara sistem operasi tamu dan perangkat keras. Namun, keuntungan terbesar dari virtualisasi penuh adalah tidak memerlukan modifikasi pada sistem operasi tamu. Artinya, sistem operasi yang berjalan di atas virtualisasi penuh dapat berjalan sebagaimana adanya tanpa memerlukan perubahan atau



penyesuaian khusus.

Meski demikian, terdapat batasan dalam virtualisasi penuh, yaitu bahwa sistem operasi tamu harus kompatibel dengan perangkat keras yang ada di bawahnya. Beberapa jenis perangkat keras dapat menimbulkan masalah dalam virtualisasi penuh. Contohnya, beberapa instruksi sensitif tidak selalu bisa ditangkap oleh VM, yang berarti hypervisor harus secara dinamis memindai instruksi tersebut menggunakan mode khusus pemerangkap untuk dapat menangani instruksi-instruksi tersebut. Ini menambah beban kerja hypervisor dan dapat mempengaruhi performa sistem secara keseluruhan.

Beberapa contoh implementasi virtualisasi penuh:

- **Contoh**

- VMware ESX Server: Menggunakan hypervisor tipe 1 yang berjalan langsung di perangkat keras, memungkinkan banyak mesin virtual tanpa modifikasi pada sistem operasi tamu.
- Microsoft Virtual Server: Memungkinkan menjalankan sistem operasi tamu tanpa perubahan, meski kini lebih umum digunakan Hyper-V.

Untuk mengurangi overhead ini, beberapa produsen perangkat keras seperti Intel dan AMD telah memperkenalkan teknologi yang mendukung virtualisasi secara langsung pada level perangkat keras, seperti Intel VT (Virtualization Technology) dan AMD-V. Teknologi ini membantu mengurangi overhead yang disebabkan oleh hypervisor dengan memungkinkan sistem operasi tamu untuk berinteraksi lebih langsung dengan perangkat keras, sehingga meningkatkan efisiensi dan kinerja keseluruhan dari virtualisasi penuh.

Dengan demikian, virtualisasi penuh menjadi salah satu pilihan yang sering digunakan dalam lingkungan server dan data center, karena kelebihanannya dalam mendukung berbagai macam sistem operasi tanpa memerlukan perubahan pada kode sistem operasi tersebut, meskipun tetap ada beberapa batasan terkait performa dan kebutuhan perangkat keras khusus yang harus dipenuhi untuk memastikan virtualisasi berjalan secara optimal.

## 2. **Para-Virtualization**

Paravirtualisasi adalah teknik virtualisasi yang mirip dengan virtual-

isasi penuh, namun dengan pendekatan yang sedikit berbeda. Pada metode ini, hypervisor tetap digunakan untuk berbagi akses ke perangkat keras, tetapi sistem operasi tamu perlu dimodifikasi agar lebih sadar akan proses virtualisasi (lihat Gambar 3). Pendekatan ini menghilangkan kebutuhan untuk melakukan rekompilasi atau pemerangkapan instruksi, karena sistem operasi bekerja sama dengan hypervisor dalam proses virtualisasi.

Salah satu kelemahan utama dari paravirtualisasi adalah bahwa sistem operasi tamu harus dimodifikasi agar kompatibel dengan lingkungan virtualisasi. Hal ini bisa menjadi kerugian bagi sistem operasi yang tidak siap untuk diubah atau dimodifikasi.

Beberapa contoh implementasi paravirtualisasi:

- **Contoh**

- VMware ESXi: Hypervisor tipe 1 yang memanfaatkan Intel VT dan AMD-V untuk efisiensi mesin virtual.
- KVM (Kernel-based Virtual Machine): Hypervisor di Linux yang menggunakan ekstensi virtualisasi prosesor modern untuk kinerja optimal.
- Microsoft Hyper-V: Platform virtualisasi yang memanfaatkan dukungan hardware untuk menjalankan mesin virtual dengan overhead minimal.
- Xen: Meskipun menggunakan paravirtualisasi, Xen juga memanfaatkan ekstensi hardware untuk meningkatkan performa.

Keunggulan utama dari paravirtualisasi adalah kemampuannya untuk memberikan performa yang mendekati sistem non-virtualisasi, karena overhead yang lebih rendah dibandingkan dengan virtualisasi penuh. Mirip dengan virtualisasi penuh, beberapa sistem operasi dapat dijalankan secara bersamaan dalam lingkungan paravirtualisasi. Namun, modifikasi pada sistem operasi menjadi persyaratan yang tidak dapat dihindari dalam metode ini.

Cara kerja paravirtualisasi dimulai dengan modifikasi pada sistem operasi tamu agar dapat berinteraksi secara langsung dengan hypervisor. Dengan adanya Application Programming Interface (API) yang disediakan oleh hypervisor, sistem operasi dapat melakukan panggilan untuk mendapatkan akses ke sumber daya perangkat keras, seperti

CPU dan memori, tanpa melalui proses penjebakan yang kompleks. Proses ini memungkinkan komunikasi yang lebih cepat dan efisien antara sistem operasi dan hypervisor, sehingga mengurangi overhead yang biasanya terdapat dalam virtualisasi penuh. Karena sistem operasi sudah "sadar" akan lingkungan virtualisasi, banyak instruksi sensitif dapat dijalankan dengan lebih langsung, sehingga memberikan kinerja yang lebih baik dibandingkan dengan metode virtualisasi lainnya.

### 3. **Hardware-assisted Virtualization**

Virtualisasi Berbasis Hardware (Hardware-Assisted Virtualization) mirip dengan Virtualisasi Penuh dan Paravirtualisasi, tetapi memiliki perbedaan mendasar karena bergantung pada ekstensi hardware khusus yang dirancang untuk mengurangi overhead dan meningkatkan performa. Teknik ini memanfaatkan fitur yang ada dalam prosesor modern, seperti Intel VT (Vanderpool) dan AMD-V (Pacifica), untuk menangani tugas yang biasanya menyebabkan kemacetan dalam virtualisasi berbasis perangkat lunak, terutama operasi I/O dan instruksi terprivileg pada sistem operasi tamu.

Beberapa contoh implementasi Hardware-Assisted Virtualization:

- **Contoh**

- VMware ESXi: Platform virtualisasi yang populer, menggunakan virtualisasi berbasis hardware untuk meningkatkan performa dan efisiensi.
- KVM (Kernel-based Virtual Machine): Solusi virtualisasi yang dibangun di atas kernel Linux, mendukung banyak mesin virtual dengan overhead yang minimal.
- Microsoft Hyper-V: Platform virtualisasi dari Microsoft yang memanfaatkan fitur virtualisasi berbasis hardware untuk menjalankan berbagai sistem operasi tamu secara efisien.

Keunggulan utama dari virtualisasi berbasis hardware adalah kemampuannya untuk menjalankan sistem operasi tamu yang tidak dimodifikasi. Berbeda dengan Paravirtualisasi, di mana sistem operasi tamu harus dimodifikasi agar bisa bekerja sama dengan hypervisor, virtualisasi berbasis hardware memungkinkan sistem operasi tamu berinteraksi dengan perangkat keras seolah-olah ia berjalan langsung di mesin

fisik. Perangkat keras memberikan dukungan untuk operasi terprivileg dan terlindungi, sehingga meningkatkan efisiensi virtualisasi.

Selain itu, virtualisasi berbasis hardware juga mengurangi kompleksitas dalam menangani instruksi sensitif. Dalam virtualisasi berbasis perangkat lunak, hypervisor harus mencegat dan mengemulasikan instruksi terprivileg, yang tidak efisien. Dengan fitur seperti Intel VT-x dan AMD-V, operasi sensitif ditangani langsung oleh prosesor, yang mempercepat manajemen mesin virtual dan eksekusi sistem operasi tamu.

#### 4. OS-level Virtualization (Containerization)

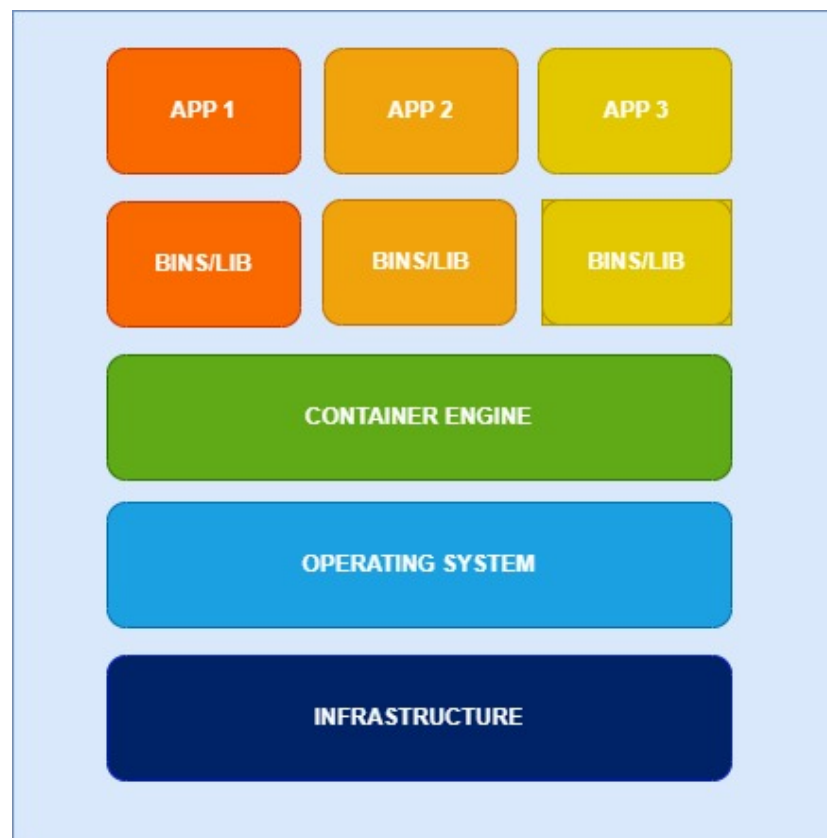


Figure 2: Containerization

OS-Level Virtualization, atau lebih dikenal sebagai Containerization, adalah teknologi yang memungkinkan pengembang untuk membungkus

aplikasi beserta semua dependensinya ke dalam "kontainer" yang dapat dijalankan di berbagai lingkungan yang mendukung teknologi kontainer, seperti Docker atau Kubernetes. Kontainer ini berjalan di atas sistem operasi host yang sama, tetapi memberikan lingkungan runtime yang terisolasi dan konsisten untuk aplikasi. Hal ini memungkinkan pengembang untuk mengembangkan, menguji, dan menyebarluaskan aplikasi dengan lebih efisien dan konsisten.

Keunggulan utama dari teknologi kontainerisasi adalah portabilitas dan efisiensi sumber daya. Kontainer dapat dengan mudah dipindahkan antar lingkungan, seperti dari pengembangan ke produksi, atau antara penyedia cloud yang berbeda. Selain itu, kontainer lebih ringan dan menggunakan lebih sedikit sumber daya dibandingkan dengan mesin virtual tradisional. Namun, kontainer juga memiliki beberapa kerugian, seperti membutuhkan waktu lama untuk belajar dan memerlukan perhatian khusus dalam hal keamanan dan pengelolaan kompleksitas.

Contoh yang paling populer dari teknologi kontainerisasi adalah **Docker**. Dengan Docker, pengembang dapat membuat versi aplikasi mereka dalam wadah yang dapat dijalankan di berbagai lingkungan. Docker juga mendukung alat orkestrasi kontainer seperti Docker Swarm dan Kubernetes, yang membantu mengelola kluster kontainer di beberapa host. Implementasi kontainerisasi biasanya dimulai dengan membuat image kontainer yang mencakup semua dependensi aplikasi, kemudian menjalankan image tersebut menggunakan Docker atau alat lainnya.

Jenis penerapan spesifik virtualisasi yang dibangun di atas dasar virtualisasi sistem operasi:

### 1. Desktop Virtualization

- **Pengertian:** Proses menjalankan sistem operasi desktop dalam lingkungan virtual yang diakses dari jarak jauh.
- **Cara Kerja:** Sistem operasi desktop berjalan di server pusat, diakses melalui klien (thin client atau komputer lain).
- **Keuntungan:** Akses dari mana saja dan manajemen terpusat untuk banyak pengguna.

### 2. Network Virtualization

- **Pengertian:** Pembuatan beberapa jaringan virtual di atas perangkat keras jaringan fisik.
- **Cara Kerja:** Setiap jaringan virtual terpisah seperti jaringan fisik.
- **Keuntungan:** Skalabilitas dan fleksibilitas dalam pengelolaan infrastruktur jaringan.

### 3. Storage Virtualization

- **Pengertian:** Menyatukan beberapa perangkat penyimpanan fisik menjadi satu sumber penyimpanan virtual.
- **Cara Kerja:** Data disimpan di beberapa perangkat tetapi disajikan seolah-olah berasal dari satu perangkat terpusat.
- **Keuntungan:** Meningkatkan fleksibilitas, manajemen, dan penggunaan penyimpanan.

### 4. Application Virtualization

- **Pengertian:** Teknologi yang memungkinkan aplikasi dijalankan di lingkungan virtual terpisah dari sistem operasi dasar.
- **Cara Kerja:** Aplikasi dibungkus dalam lingkungan virtual yang menyimpan semua dependensi dan konfigurasi.
- **Keuntungan:** Pengelolaan terpusat, keamanan yang ditingkatkan, portabilitas, dan efisiensi sumber daya.

#### 3.12.3 Komponen Virtualisasi Sistem Operasi

1. **Host Machine (Host OS):** Sistem operasi fisik yang berjalan langsung di atas perangkat keras dan mengelola sumber daya seperti CPU, RAM, dan penyimpanan. Berfungsi sebagai platform untuk hypervisor.
  - Contoh: Windows, Linux, macOS.
2. **Guest Machine (Guest OS):** Sistem operasi virtual yang berjalan di atas mesin virtual (VM) dan diatur oleh hypervisor.
  - Contoh: Linux, Windows.

3. **Hypervisor:** Perangkat lunak yang memungkinkan virtualisasi dan mengelola VM.
  - Tipe 1 (Bare-metal): Berjalan langsung di perangkat keras. Contoh: VMware ESXi, Hyper-V.
  - Tipe 2 (Hosted): Berjalan di atas sistem operasi host. Contoh: VirtualBox, VMware Workstation.
4. **Virtual Machine (VM):** Lingkungan perangkat lunak yang dibuat oleh hypervisor untuk menjalankan guest OS secara terisolasi dari perangkat keras fisik.

#### 3.12.4 Hypervisor dan Virtual Machines

#### 3.12.5 Contoh Penerapan Virtualisasi pada Sistem Operasi

#### 3.12.6 Kelebihan dan Kekurangan Virtualisasi pada Sistem Operasi

#### 3.12.7 Kesimpulan

Virtualisasi sistem operasi telah menjadi fondasi penting dalam pengelolaan infrastruktur TI modern. Dengan memungkinkan pembuatan dan pengelolaan lingkungan virtual yang meniru perangkat keras fisik, teknologi ini memberikan efisiensi sumber daya yang signifikan, isolasi antara aplikasi, dan kemampuan skalabilitas yang tinggi. Teknik-teknik seperti full virtualization, para-virtualization, dan containerization, serta penggunaan hypervisor tipe 1 dan tipe 2, memberikan fleksibilitas dalam pengelolaan berbagai sistem operasi dan aplikasi. Implementasi luas teknologi ini, yang mencakup solusi seperti VMware, VirtualBox, KVM, dan Docker, menunjukkan kemampuannya dalam mendukung berbagai kebutuhan, mulai dari pengembangan dan pengujian perangkat lunak hingga manajemen infrastruktur TI yang kompleks. Dengan demikian, virtualisasi tidak hanya meningkatkan efisiensi operasional, tetapi juga memberikan kemudahan dalam pengelolaan sumber daya yang beragam, menjadikannya solusi yang sangat diperlukan dalam dunia TI yang terus berkembang.

## References

- [1] Umar, R., & Fakultas Teknologi Industri Universitas Ahmad Dahlan Yogyakarta. (2013). REVIEW TENTANG VIRTUALISASI. *JURNAL INFORMATIKA*, 7(2), 775–777.
- [2] Firmansyah Adiputra. (2015). CONTAINER DAN DOCKER: TEKNIK VIRTUALISASI DALAM PENGELOLAAN BANYAK APLIKASI WEB. *Jurnal Ilmiah SimanteC*, 4(3), 167–169. Jl. Raya Telang, PO BOX 2 Kamal, Bangkalan. Retrieved September 27, 2024.
- [3] Operating System Virtualization – Types, Working, Benefits. (2023, September 8). Retrieved from <https://www.knowledgehut.com/blog/cloud-computing/os-virtualization>
- [4] GeeksforGeeks. (2023, October 31). Types of server virtualization in computer network. Retrieved from <https://www.geeksforgeeks.org/types-of-server-virtualization-in-computer-network/>
- [5] Hashemi-Pour, C., Brush, K., & Kirsch, B. (2024, June 5). virtualization. IT Operations. Retrieved from <https://www.techtarget.com/searchitoperations/definition/virtualization>

## 4 Assignments and Practical Work

### 4.1 Assignment 1: Process Scheduling

Students were tasked with implementing various process scheduling algorithms (e.g., FCFS, SJN, and RR) and comparing their performance under different conditions.

### 4.2 Assignment 2: Deadlock Handling

In this assignment, students were asked to simulate different deadlock scenarios and explore various prevention methods.



### **4.3 Assignment 3: Multithreading and Amdahl's Law**

This assignment involved designing a multithreading scenario to solve a computationally intensive problem. Students then applied **Amdahl's Law** to calculate the theoretical speedup of the program as the number of threads increased.

### **4.4 Assignment 4: Simple Command-Line Interface (CLI) for User Interface Management**

Students were tasked with creating a simple **CLI** for user interface management. The CLI should support basic commands such as file manipulation (creating, listing, and deleting files), process management, and system status reporting.

### **4.5 Assignment 5: File System Access**

In this assignment, students implemented file system access routines, including:

- File creation and deletion
- Reading from and writing to files
- Navigating directories and managing file permissions

## **5 Conclusion**

The first half of the course introduced core operating system concepts, including process management, scheduling, multithreading, and file system access. These topics provided a foundation for more advanced topics to be covered in the second half of the course.