# IT5403

# OPERATING SYSTEMS

A CASE STUDY

*Submitted by Team Dynamic:*

Gowtham Rajasekaran 2022506084

Aadhira 2022506087

Ragul 2022506054

Vicky 2022506049

Arvinth 2022506316

B. Tech (4/8)

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**MADRAS INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY- CHENNAI**

**CHENNAI - 600 044**

March / April 2024

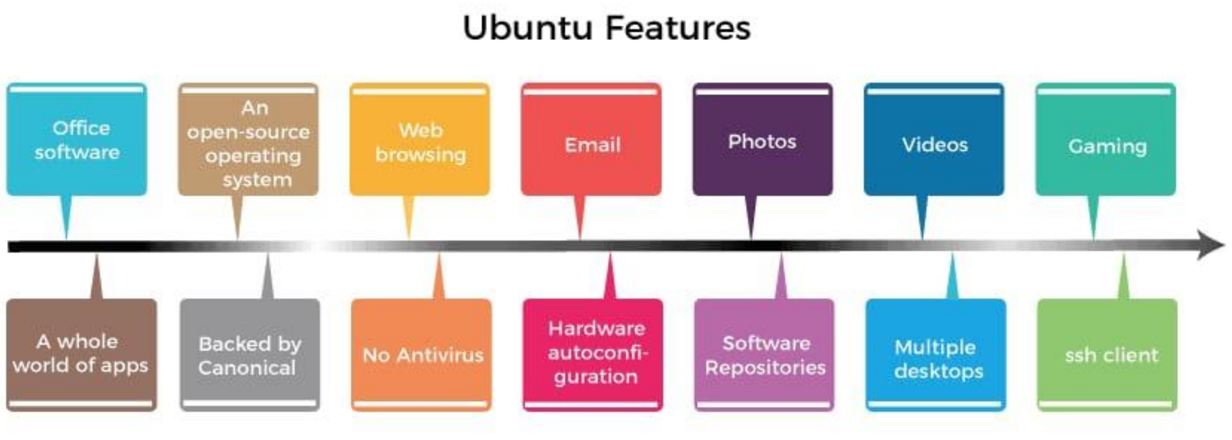
# CONTENTS

# INTRODUCTION

## What is Ubuntu?

Ubuntu is a popular Linux distribution that is widely used for desktop, server, and cloud computing. It is known for its ease of use, regular updates, and strong community support. Developed by Canonical Ltd., Ubuntu is based on the Ubuntu architecture and is released with a new version every six months, with long-term support (LTS) versions released every two years. Ubuntu comes with a variety of pre-installed software and offers a user-friendly interface, making it accessible to both beginners and experienced users alike.

## Key Features



1. **User-Friendly Interface**: Ubuntu offers a sleek and intuitive desktop environment, such as GNOME or Unity, making it easy for users to navigate and operate their systems.

2. **Open Source**: Ubuntu is built on open-source principles, meaning its source code is freely available for anyone to use, modify, and distribute.

3. **Regular Updates**: Canonical releases new versions of Ubuntu every six months, ensuring users have access to the latest features, security patches, and software updates.

4. **Long-Term Support (LTS)**: LTS releases are supported for five years on the desktop and server, providing stability and reliability for users who prefer not to upgrade frequently.

5. **Software Center**: Ubuntu Software Center provides a centralized location for users to discover, install, and manage applications, making it easy to find and install software.

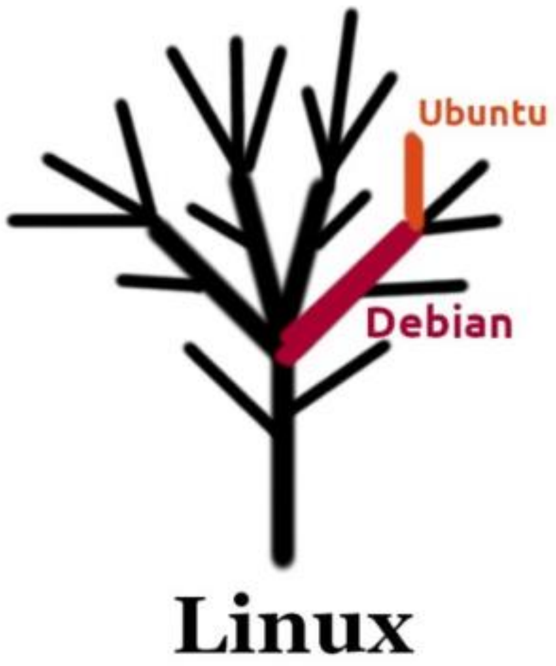
6. **Community Support**: Ubuntu has a large and active community of users and developers who provide support, troubleshooting, and guidance through forums, wikis, and other online resources.

7. **Security**: Ubuntu prioritizes security and includes built-in security features such as AppArmor, which helps protect applications from security threats.

8. **Versatility**: Ubuntu is versatile and can be used for various purposes, including desktop computing, server hosting, cloud deployments, and IoT (Internet of Things) devices.

9. **Accessibility**: Ubuntu aims to be accessible to users with disabilities by providing features such as screen readers, magnification tools, and keyboard navigation options.

10. **Customization**: Users can customize Ubuntu to suit their preferences by choosing from a variety of desktop environments, themes, and extensions available in the software repositories.



## Beyond the Usual: Unique Applications of Ubuntu

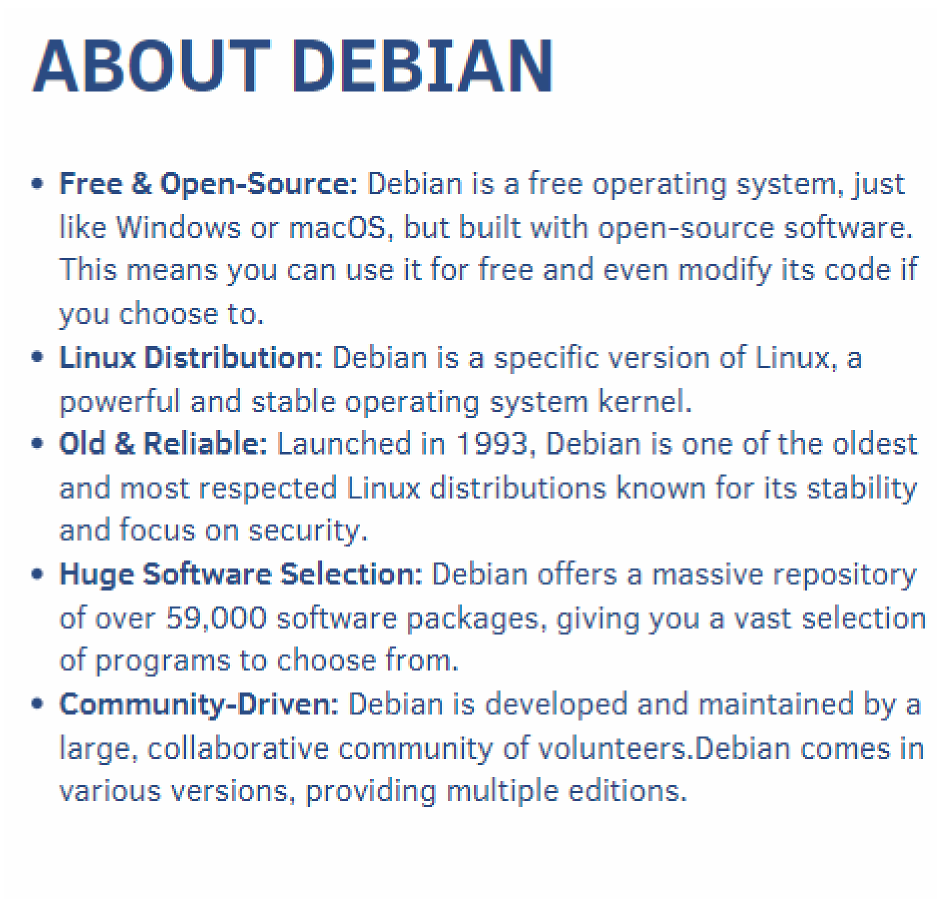
While web servers and desktops are common uses, Ubuntu's power extends to some fascinating real-world applications:

- **Space Exploration:** NASA uses Ubuntu on some of its ground control systems and even experimented with it on the International Space Station.
- **Hollywood Blockbusters:** The visual effects for movies like Avatar and Gravity relied heavily on render farms powered by Ubuntu and Ubuntu. The ability to efficiently manage large compute clusters makes them ideal for such tasks.
- **Supercomputers:** Many of the world's fastest supercomputers run on customized versions of Linux, often Ubuntu-based, due to their scalability and performance optimization potential.
- **Robotics:** From self-driving cars to industrial robots, Ubuntu and Ubuntu provide a stable and customizable foundation for the software that powers them.
- **Digital Preservation:** Libraries and archives use these platforms to store and manage vast amounts of digital data due to their long-term stability and open-source nature, ensuring future access.

## BRIEF EXPLANATION OF OS

DEBIAN

- Debian is a free and open-source Unix-like operating system and Linux distribution.
- It is known for its stability, reliability, and commitment to free software principles.
- It is developed by a global community of volunteers.

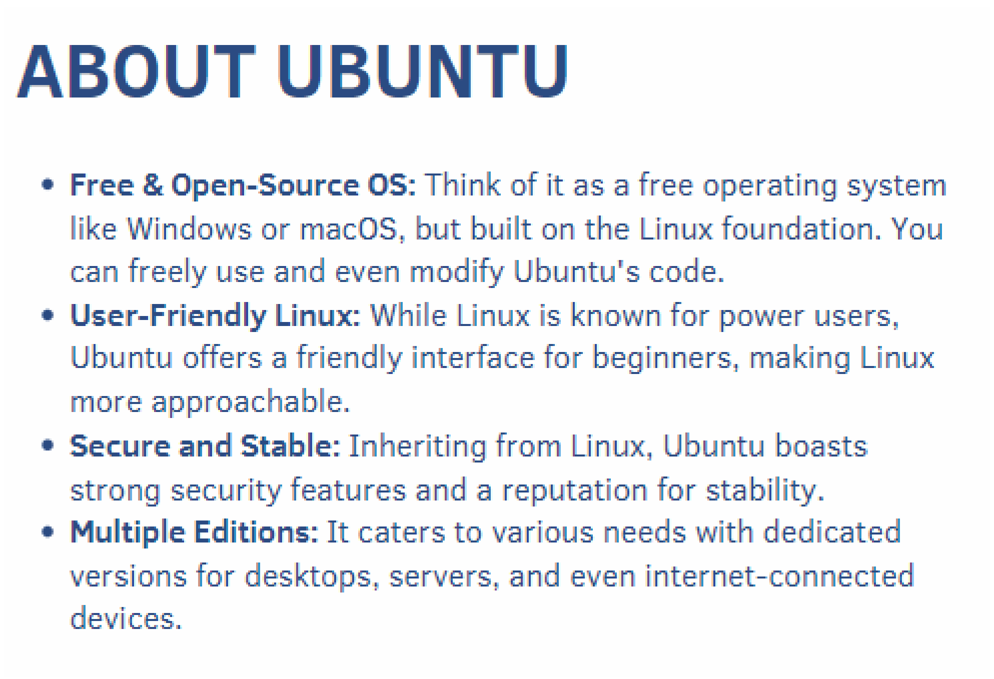


## ABOUT DEBIAN

- **Free & Open-Source:** Debian is a free operating system, just like Windows or macOS, but built with open-source software. This means you can use it for free and even modify its code if you choose to.
- **Linux Distribution:** Debian is a specific version of Linux, a powerful and stable operating system kernel.
- **Old & Reliable:** Launched in 1993, Debian is one of the oldest and most respected Linux distributions known for its stability and focus on security.
- **Huge Software Selection:** Debian offers a massive repository of over 59,000 software packages, giving you a vast selection of programs to choose from.
- **Community-Driven:** Debian is developed and maintained by a large, collaborative community of volunteers.Debian comes in various versions, providing multiple editions.

## UBUNTU

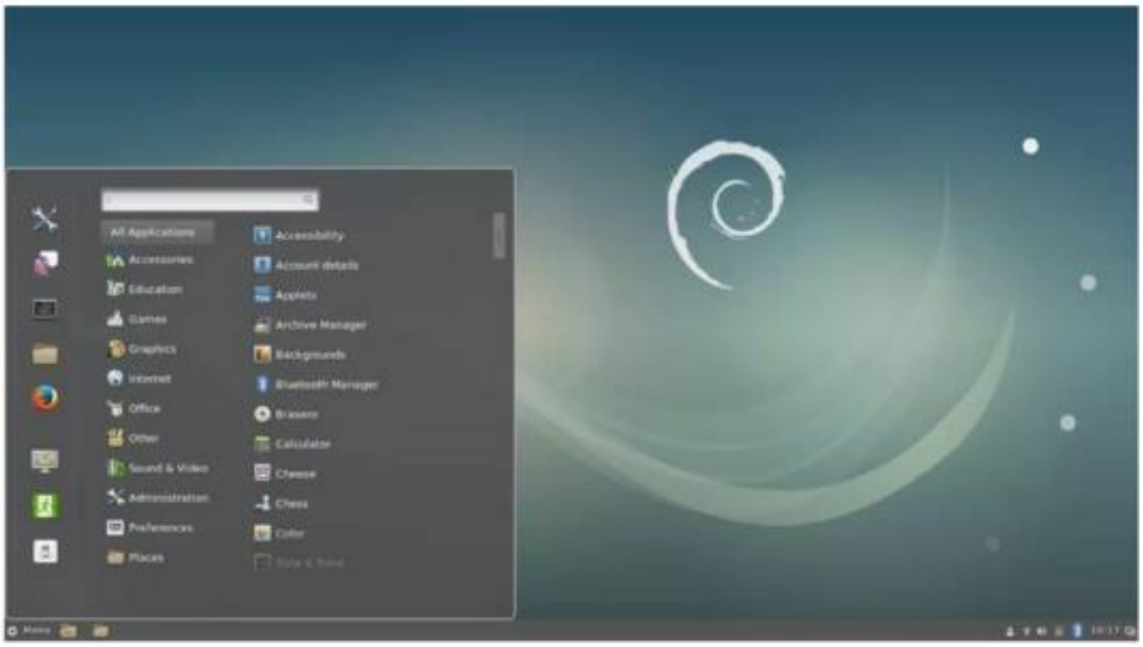
- Ubuntu is a popular Linux distribution based on Debian.
- It is known for its ease of use, extensive software library, regular release cycle, and strong community support.
- Ubuntu is widely used for desktop, server, cloud, and IoT (Internet of Things) deployments.



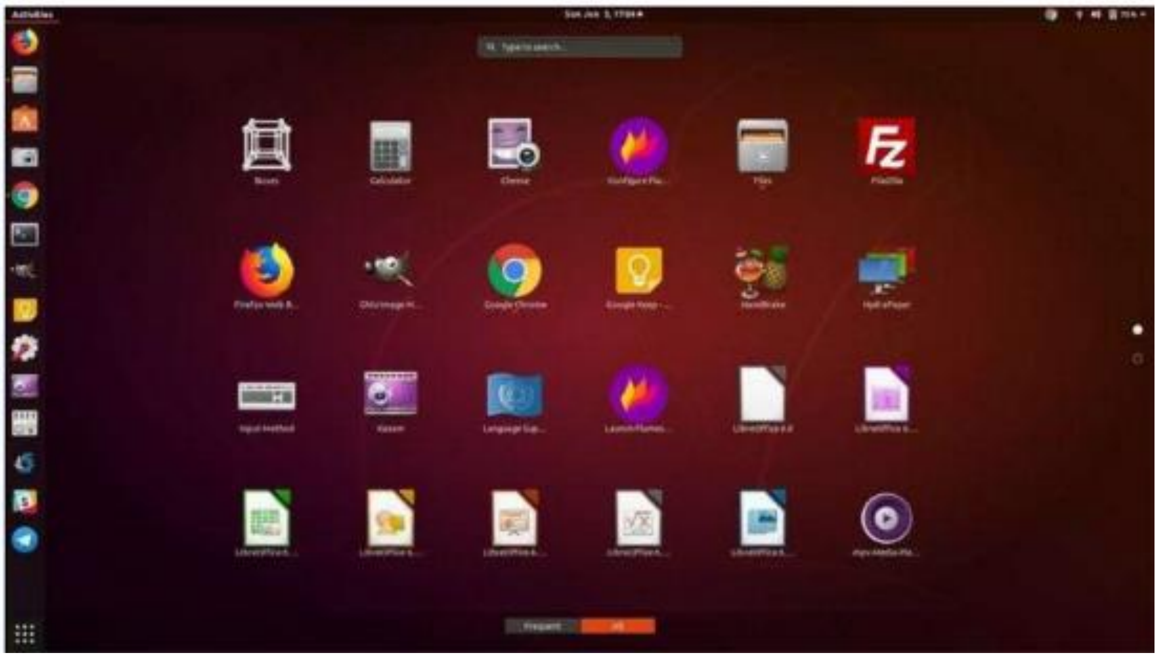
## ABOUT UBUNTU

- **Free & Open-Source OS:** Think of it as a free operating system like Windows or macOS, but built on the Linux foundation. You can freely use and even modify Ubuntu's code.
- **User-Friendly Linux:** While Linux is known for power users, Ubuntu offers a friendly interface for beginners, making Linux more approachable.
- **Secure and Stable:** Inheriting from Linux, Ubuntu boasts strong security features and a reputation for stability.
- **Multiple Editions:** It caters to various needs with dedicated versions for desktops, servers, and even internet-connected devices.

## DEBIAN DESKTOP

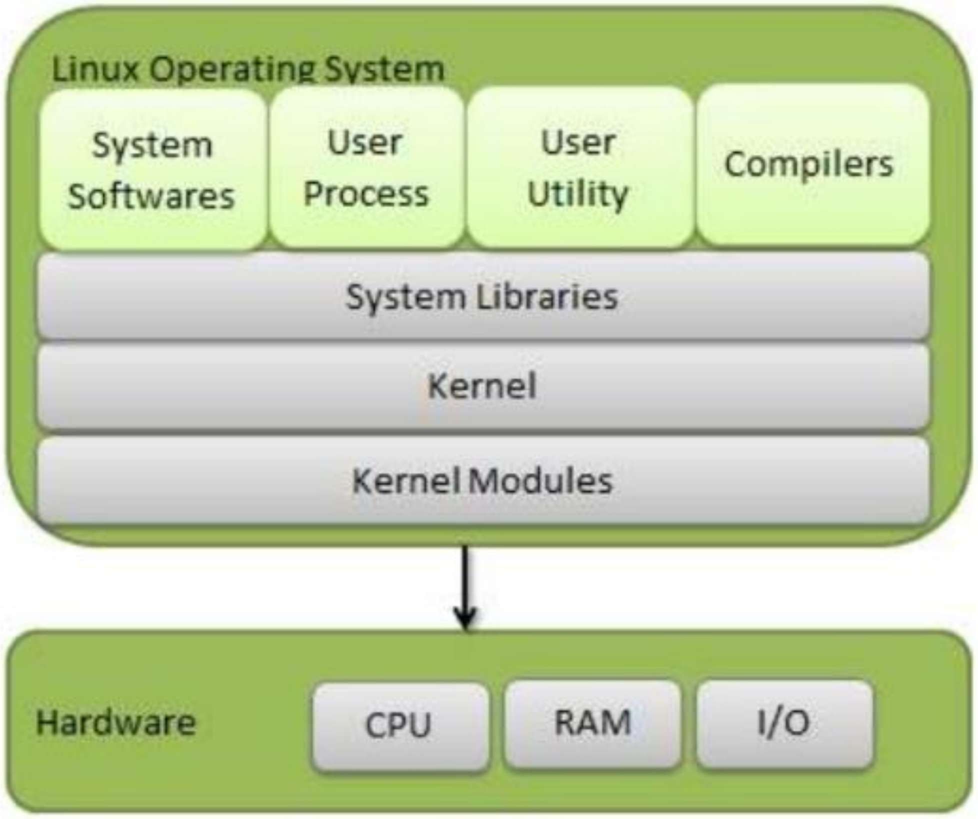


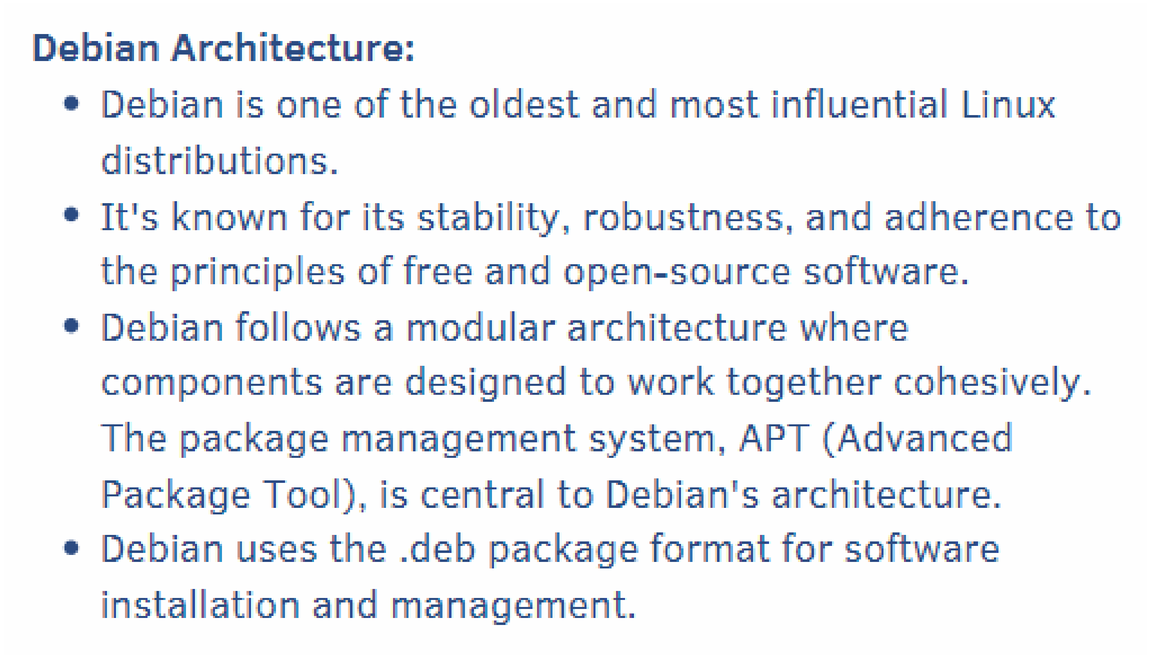
## UBUNTU DESKTOP



## KERNEL ARCHITECTURE

**Ubuntu builds on the Debian architecture and infrastructure and collaborates widely with Debian developers, but there are important differences. Ubuntu has a distinctive user interface, a separate developer community.**





### Debian Architecture:

- Debian is one of the oldest and most influential Linux distributions.
- It's known for its stability, robustness, and adherence to the principles of free and open-source software.
- Debian follows a modular architecture where components are designed to work together cohesively. The package management system, APT (Advanced Package Tool), is central to Debian's architecture.
- Debian uses the .deb package format for software installation and management.

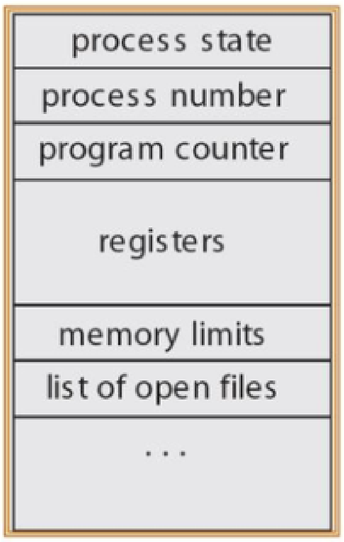
**Ubuntu Architecture:**

- Ubuntu is based on Debian and shares many architectural principles with it. However, Ubuntu aims to provide a more user-friendly and polished experience out of the box.
- Like Debian, Ubuntu utilizes the APT package management system and .deb package format.
- Ubuntu has its own software repositories, which are based on Debian's but often include newer versions of software packages
- Ubuntu has its own desktop environment called Unity and now GNOME, though it supports various other desktop environments.

# PROCESS CONTROL MECHANISMS

## PROCESS CONTROL BLOCK

1. Process ID (PID): Unique identifier for the process.
2. Process State: Current state of the process (e.g., running, ready, blocked).
3. Program Counter (PC): Memory address of the next instruction to be executed.
4. CPU Registers: Values of CPU registers associated with the process.
5. Memory Management Information: Details about the process's memory usage.
6. File Descriptors: Tracks open files, sockets, and other I/O resources.
7. Process Priority: Scheduling priority of the process.
8. Parent Process ID (PPID): Identifier of the parent process.



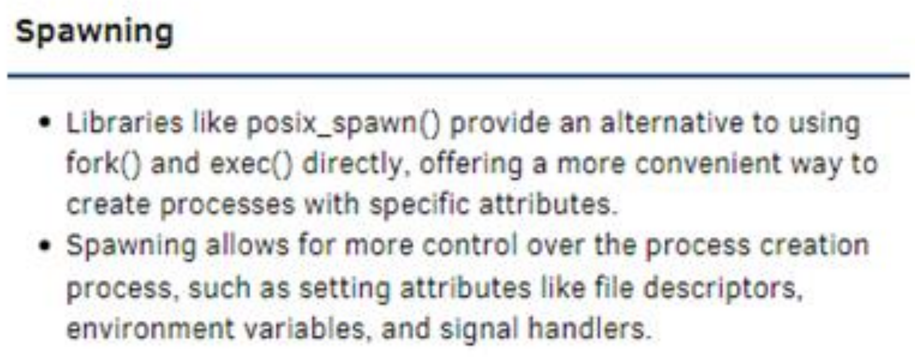
Process Control Block

## Process creation

In Ubuntu, as in any Unix-like operating system, processes can be created using several methods, with forking and spawning being the most common.

### Forking

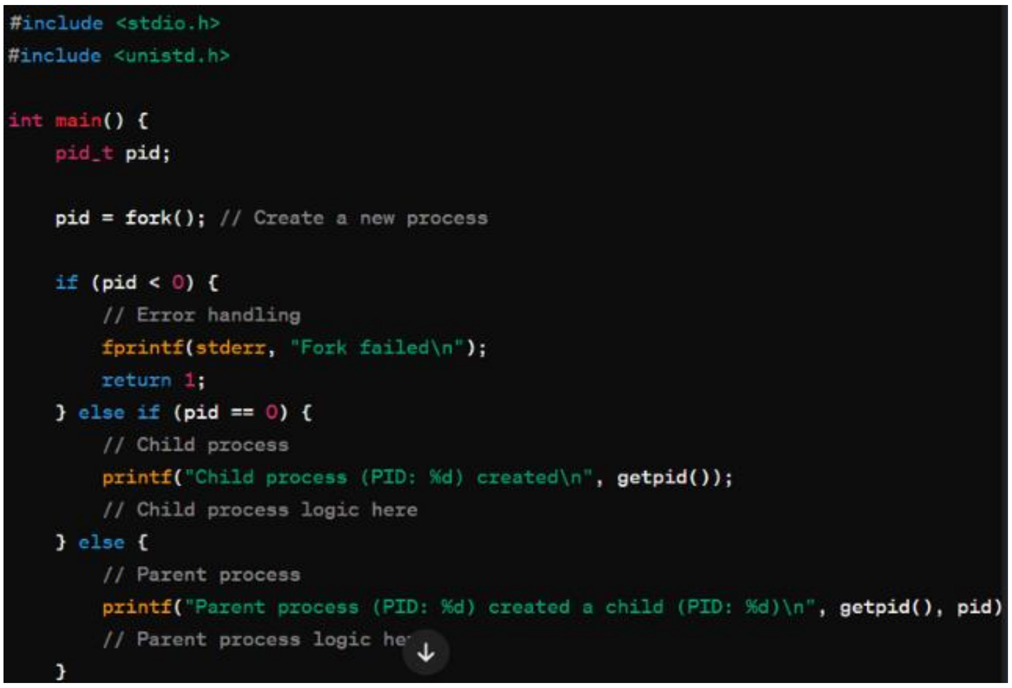
- Forking is a fundamental mechanism for process creation in Unix-like systems.
- The fork() system call is used, which creates a new process (child process) as a copy of the calling process (parent process).



**Spawning**

- Libraries like posix_spawn() provide an alternative to using fork() and exec() directly, offering a more convenient way to create processes with specific attributes.
- Spawning allows for more control over the process creation process, such as setting attributes like file descriptors, environment variables, and signal handlers.

## Process Creation and Termination

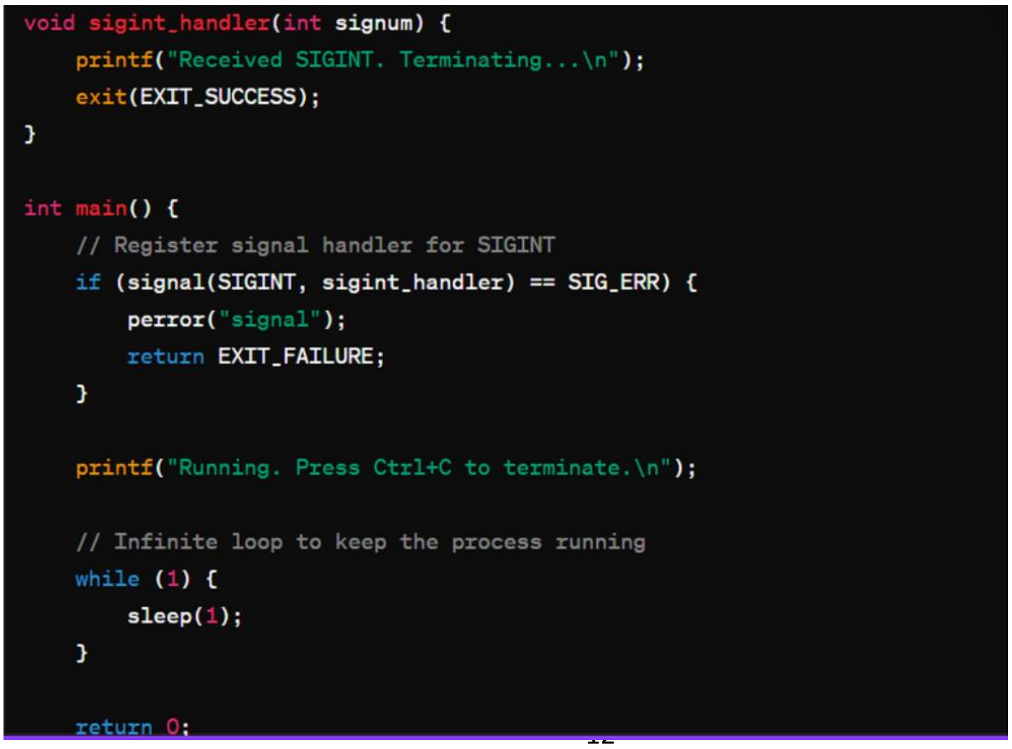


```
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid;

    pid = fork(); // Create a new process

    if (pid < 0) {
        // Error handling
        fprintf(stderr, "Fork failed\n");
        return 1;
    } else if (pid == 0) {
        // Child process
        printf("Child process (PID: %d) created\n", getpid());
        // Child process logic here
    } else {
        // Parent process
        printf("Parent process (PID: %d) created a child (PID: %d)\n", getpid(), pid)
        // Parent process logic he
    }
```



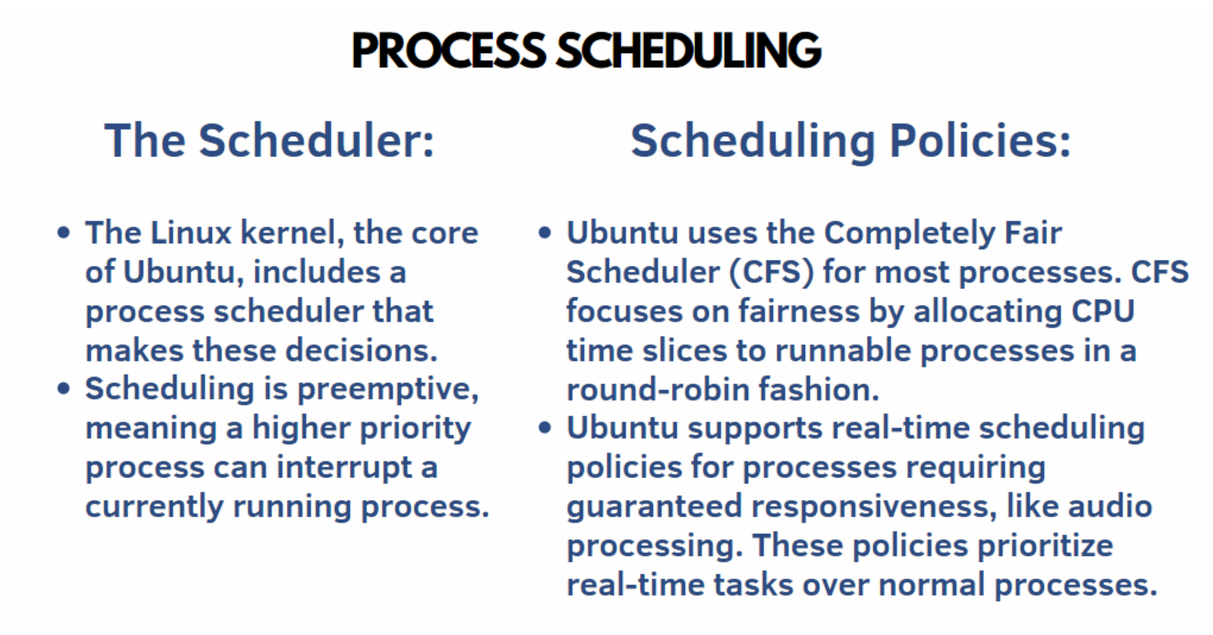
```
void sigint_handler(int signum) {
    printf("Received SIGINT. Terminating...\n");
    exit(EXIT_SUCCESS);
}

int main() {
    // Register signal handler for SIGINT
    if (signal(SIGINT, sigint_handler) == SIG_ERR) {
        perror("signal");
        return EXIT_FAILURE;
    }

    printf("Running. Press Ctrl+C to terminate.\n");

    // Infinite loop to keep the process running
    while (1) {
        sleep(1);
    }

    return 0;
```



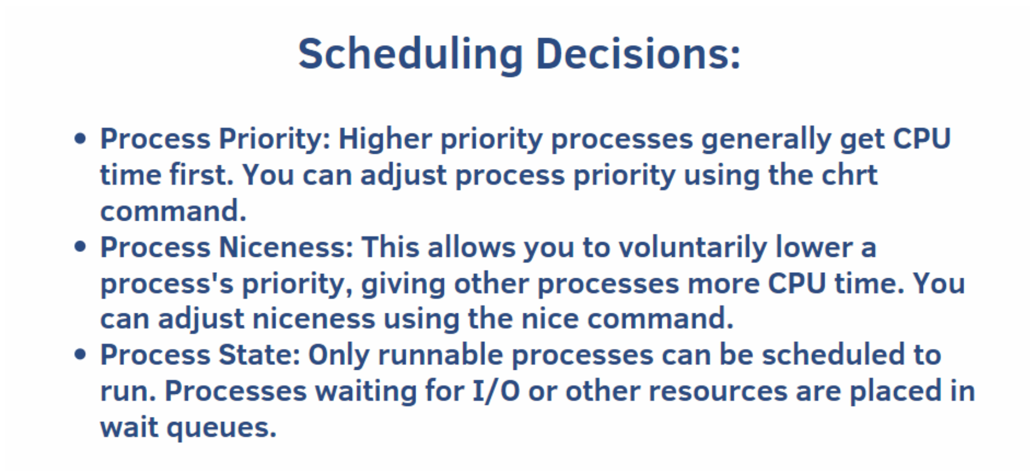
## PROCESS SCHEDULING

### The Scheduler:

- The Linux kernel, the core of Ubuntu, includes a process scheduler that makes these decisions.
- Scheduling is preemptive, meaning a higher priority process can interrupt a currently running process.

### Scheduling Policies:

- Ubuntu uses the Completely Fair Scheduler (CFS) for most processes. CFS focuses on fairness by allocating CPU time slices to runnable processes in a round-robin fashion.
- Ubuntu supports real-time scheduling policies for processes requiring guaranteed responsiveness, like audio processing. These policies prioritize real-time tasks over normal processes.



### Scheduling Decisions:

- Process Priority: Higher priority processes generally get CPU time first. You can adjust process priority using the chrt command.
- Process Niceness: This allows you to voluntarily lower a process's priority, giving other processes more CPU time. You can adjust niceness using the nice command.
- Process State: Only runnable processes can be scheduled to run. Processes waiting for I/O or other resources are placed in wait queues.

## Concurrency control

Concurrency control in Debian/Ubuntu systems primarily revolves around managing multiple processes or threads accessing shared resources concurrently. This is crucial for ensuring system stability, preventing data corruption, and maximizing system utilization. Here are some common methods and tools used for concurrency control in Debian/Ubuntu systems:

1. **Mutexes and Semaphores**: These are low-level synchronization primitives used by developers when writing software to control access to shared resources. Libraries such as pthreads provide functions for creating and managing mutexes and semaphores.
2. **File Locking**: File locking mechanisms such as **flock** or **fcntl** can be used to prevent multiple processes from simultaneously accessing the same file. This helps in avoiding data corruption or race conditions when multiple processes need to read from or write to the same file.
3. **Transactional Databases**: Databases like MySQL, PostgreSQL, or SQLite provide built-in support for concurrency control through transactions. They use mechanisms like

locking, isolation levels, and transaction logs to ensure data consistency and integrity in multi-user environments.

4. **Job Control**: Tools like **cron** or **systemd** allow administrators to schedule tasks or services to run at specific times or intervals, preventing conflicts between concurrently executing jobs.
5. **Resource Limits**: Linux kernel provides facilities for setting resource limits on processes using tools like **ulimit** or **cgroups**. This helps in preventing runaway processes from consuming excessive system resources and causing performance degradation or system instability.
6. **Thread and Process Management**: Utilities like **ps**, **top**, or **htop** provide insights into the currently running processes and their resource utilization, allowing administrators to monitor and manage concurrency effectively.
7. **Message Queues and IPC**: Inter-process communication mechanisms like message queues, shared memory, or sockets enable processes to communicate and synchronize their activities, facilitating concurrency control in distributed systems.
8. **Software Transactional Memory (STM)**: Some programming languages and libraries provide STM implementations that automatically handle concurrency control by managing transactions at the language level, abstracting away the complexities of traditional locking mechanisms.

These are just some of the methods and tools used for concurrency control in Debian/Ubuntu systems. The choice of method depends on the specific requirements of the application or system being developed/deployed.

## Inter-process Communication

Inter-process communication (IPC) in Debian/Ubuntu systems involves mechanisms that allow processes to communicate and synchronize with each other, either within the same system or across different systems. Here are some common IPC mechanisms available in Debian/Ubuntu:

1. **Pipes**: Pipes are one of the simplest forms of IPC and are commonly used for communication between two related processes. In Linux, pipes can be created using the **pipe()** system call or through shell commands using the pipe symbol **|**. They allow one process to write data to the pipe, which can then be read by another process.
2. **Named Pipes (FIFOs)**: Named pipes, also known as FIFOs (first in, first out), provide a mechanism for inter-process communication between unrelated processes. They exist as special files in the file system and can be created using the **mkfifo** command. Processes can write data to a named pipe, which can then be read by other processes.
3. **Message Queues**: Message queues provide a way for processes to exchange messages in a POSIX-compliant manner. Processes can send messages to a message queue using the **msgsnd()** system call and receive messages using the **msgrcv()**

system call. Message queues support both one-to-one and one-to-many communication paradigms.

4. **Shared Memory**: Shared memory allows multiple processes to share a region of memory, enabling them to communicate by reading and writing to the same memory space. In Linux, shared memory segments can be created using the **shmget()** system call and accessed using **shmat()** and **shmdt()**. Shared memory is typically faster than other IPC mechanisms but requires careful synchronization to avoid data corruption.
5. **Sockets**: Sockets provide a network-based IPC mechanism for communication between processes running on the same system or different systems. They can be either stream-oriented (TCP) or message-oriented (UDP). Sockets are widely used for client-server communication and inter-process communication on the same system.
6. **Signals**: Signals are asynchronous notifications sent by the kernel to processes to indicate events such as hardware interrupts or errors. Processes can also send signals to each other using the **kill()** system call. While signals are primarily used for process management and handling, they can also be used for simple forms of IPC.
7. **System V IPC**: System V IPC provides a set of inter-process communication mechanisms, including message queues, shared memory, and semaphores. These mechanisms have been available in Unix-like operating systems for a long time and are still supported in modern Linux distributions like Debian/Ubuntu.

These IPC mechanisms provide developers with flexible options for designing communication between processes in Debian/Ubuntu systems, depending on factors such as performance requirements, complexity, and security considerations.

# Scheduling algorithms

"SELECTING A SCHEDULING ALGORITHM IS LIKE CRAFTING A RECIPE; IT'S ABOUT FINDING THE PERFECT BLEND OF INGREDIENTS TO ENSURE SMOOTH AND EFFICIENT OPERATION."

Both Ubuntu and Debian, being Linux-based operating systems, typically use the same or very similar scheduling algorithms. This is because both Ubuntu and Debian distributions use the Linux kernel as their core operating system. The Linux kernel provides various scheduling algorithms, and the default scheduler, such as the Completely Fair Scheduler (CFS), is commonly used across different Linux distributions, including Ubuntu and Debian.

## Here's a list of commonly used scheduling algorithms in Debian/Ubuntu:

### Completely Fair Scheduler (CFS):

- Default scheduler in the Linux kernel
- Dynamically assigns CPU time slices to processes based on their priority to achieve fairness.

### Round Robin (RR):

- Each process is allocated a fixed time slice (quantum) to execute before being preempted.
- Helps in providing equitable CPU allocation, especially in time-sharing environments.

### Earliest Deadline First (EDF):

- Real-time scheduling algorithm where tasks are scheduled based on their deadlines.
- Guarantees that the task with the earliest deadline will be executed first.

### Shortest Job First (SJF):

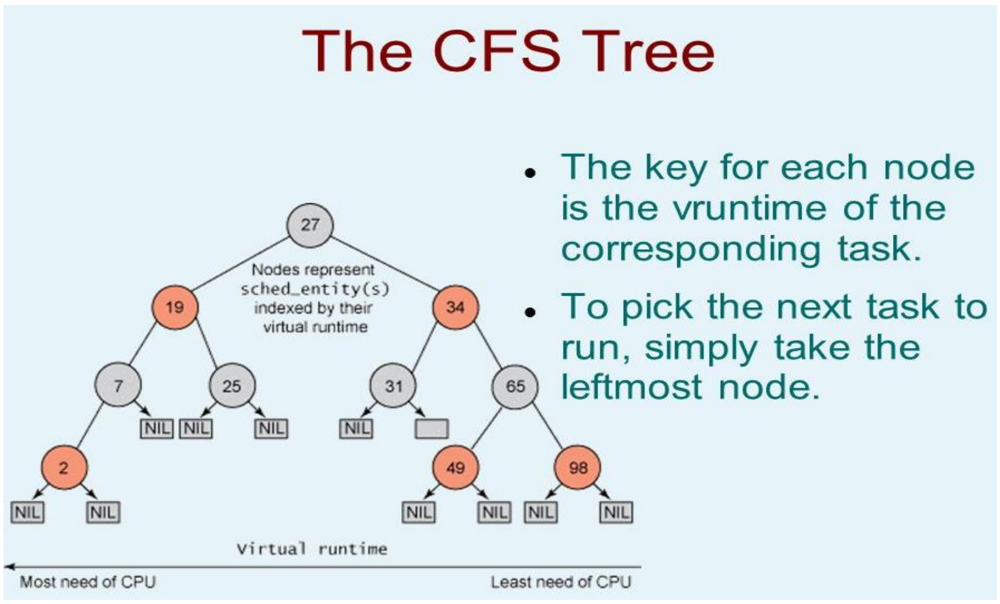
- Selects the process with the shortest expected processing time for execution next.
- Typically used in non-preemptive scenarios and requires knowledge of process burst times.

### Priority Scheduling:

- Each process is assigned a priority, and the scheduler selects the process with the highest priority for execution.
- Can be preemptive or non-preemptive.

### Multi-level Queue Scheduling:

- Divides processes into multiple queues with different priority levels.
- Each queue may have its own scheduling algorithm.



The CFS Tree

- The key for each node is the vruntime of the corresponding task.
- To pick the next task to run, simply take the leftmost node.

## CASE STUDY : UBUNTU FOR WEB SERVERS

## OVERVIEW OF UBUNTU

- **Stability You Can Trust:**Ubuntu prioritizes stability over bleeding-edge features. With extended release cycles and rigorous testing, Ubuntu ensures a rock-solid foundation for your web server. Fewer updates mean less chance of introducing bugs or compatibility issues, keeping your website running smoothly.
- **A Treasure Trove of Web Server Software:** The Ubuntu package repository offers a vast selection of web server software, including Apache, Nginx, and LiteSpeed. This variety allows you to choose the server that best suits your website's needs and preferences.
- **Effortless Configuration and Management:**Ubuntu's package management system, with tools like dpkg and apt, makes installing, configuring, and updating web server software a breeze. These tools handle dependencies and ensure everything works together seamlessly.
- **Long-Term Support for Peace of Mind:**Ubuntu's stable releases are actively supported for several years, ensuring access to critical security updates and bug fixes. This long-term support allows you to focus on your website's content and functionality, knowing the underlying server remains secure.

In essence, Ubuntu provides a stable, secure, and well-maintained platform for your website. The vast software repository, coupled with the ease of management through dpkg and apt, makes it an ideal choice for web hosting, from personal websites to complex enterprise applications.

## HOW AND WHY TO USE UBUNTU FOR WEB SERVERS?

## A SCENARIO BASED EXAMPLE

### Background:

ABC Solutions is a small IT consulting firm with 50 employees, providing services such as web development, software solutions, and IT support for clients. The company currently operates with a mix of Windows and macOS systems, and they are experiencing challenges with software licensing costs, system stability, and security vulnerabilities. To address these issues, they decide to migrate their IT infrastructure to Ubuntu Linux.

### Objectives:

Reduce software licensing costs by transitioning to open-source solutions.

Improve system stability and security by adopting Ubuntu/Ubuntu Linux.

Streamline IT operations and enhance employee productivity.

Minimize downtime and disruption during the migration process.

### Implementation Steps:

1. **Assessment and Planning:**
    - Conduct a comprehensive assessment of current IT infrastructure, including hardware, software, and network configurations.
    - Identify critical applications and services that need to be migrated or replaced with Linux-compatible alternatives.
    - Evaluate employee skill levels and training requirements for using Ubuntu/Ubuntu Linux.
2. **Infrastructure Setup:**
    - Procure hardware compatible with Ubuntu/Ubuntu Linux, ensuring adequate resources for performance and scalability.
    - Install Ubuntu/Ubuntu Server on the company's main server and deploy Ubuntu Desktop on employee workstations.
    - Configure network settings, including DHCP, DNS, and firewall rules, to ensure seamless connectivity.

3. **Software Migration and Integration:**
    - Identify open-source alternatives for proprietary software used by the company, such as LibreOffice for office productivity suites and GIMP for graphic design.
    - Migrate existing data and applications to the Ubuntu/Ubuntu environment, ensuring compatibility and data integrity.
    - Implement collaborative tools such as Nextcloud for file sharing and project management to enhance team collaboration.

4. **Security Implementation:**
    - Harden the Ubuntu/Ubuntu servers and workstations by implementing best practices for system security, including regular software updates, firewall configuration, and user privilege management.
    - Install and configure antivirus software and intrusion detection systems to protect against malware and cyber threats.
    - Implement data encryption for sensitive information stored on company systems, including disk encryption and encrypted communication protocols.

5. **Training and Support:**
    - Provide training sessions and documentation for employees to familiarize them with Ubuntu/Ubuntu Linux and open-source software applications.
    - Establish a helpdesk or support system to address any issues or questions that arise during the transition period and beyond.
    - Encourage knowledge sharing and collaboration among employees to leverage the full potential of the new Linux-based infrastructure.

## Outcome:

The migration to Ubuntu/Ubuntu Linux significantly reduces software licensing costs for ABC Solutions, resulting in cost savings that can be allocated to other areas of the business.

The Ubuntu/Ubuntu Linux environment proves to be stable, secure, and reliable, addressing the company's previous concerns with system stability and security vulnerabilities.

Employee productivity improves as they become more proficient in using Ubuntu/Ubuntu Linux and open-source software tools, leading to smoother workflows and collaboration.

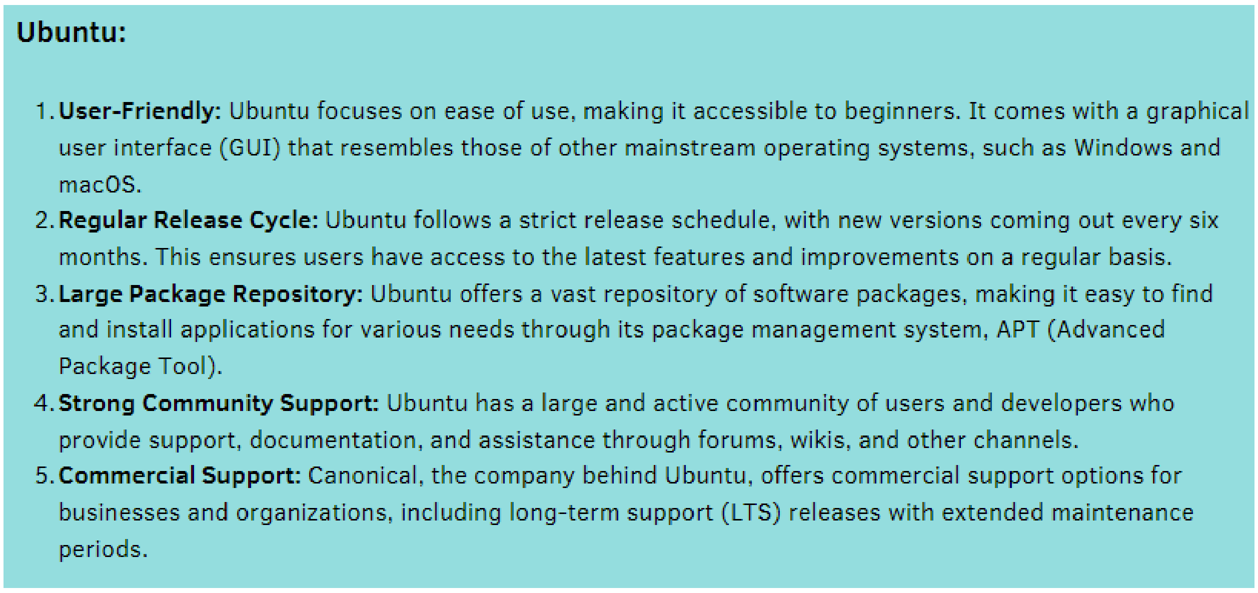
Minimal downtime and disruption occur during the migration process, thanks to careful planning and implementation, ensuring business continuity for ABC Solutions.

## Result:

By successfully implementing Debian/Ubuntu Linux in their IT infrastructure, ABC Solutions achieves their objectives of reducing costs, improving stability and security, and enhancing productivity. The transition to open-source solutions enables the company to leverage the benefits of Linux-based systems while fostering a culture of innovation and collaboration among employees.

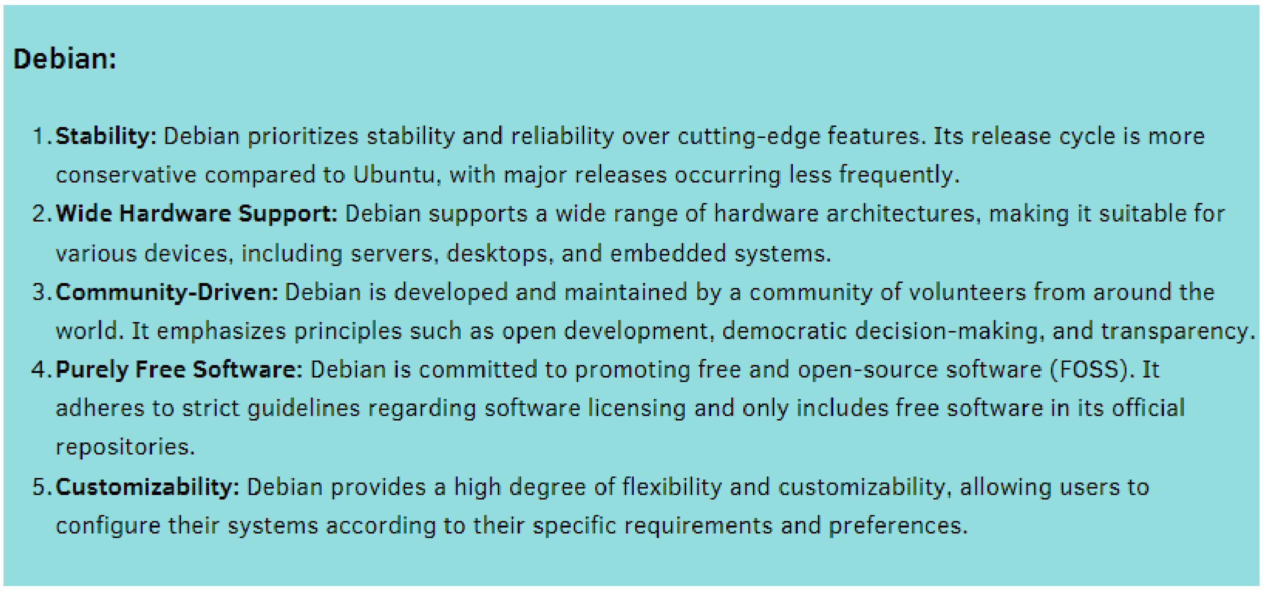
## SALIENT FEATURES:

Ubuntu and Debian are two popular Linux distributions, each with its own set of features and characteristics. Here's a comparison of their salient features and how they contrast with other operating systems:



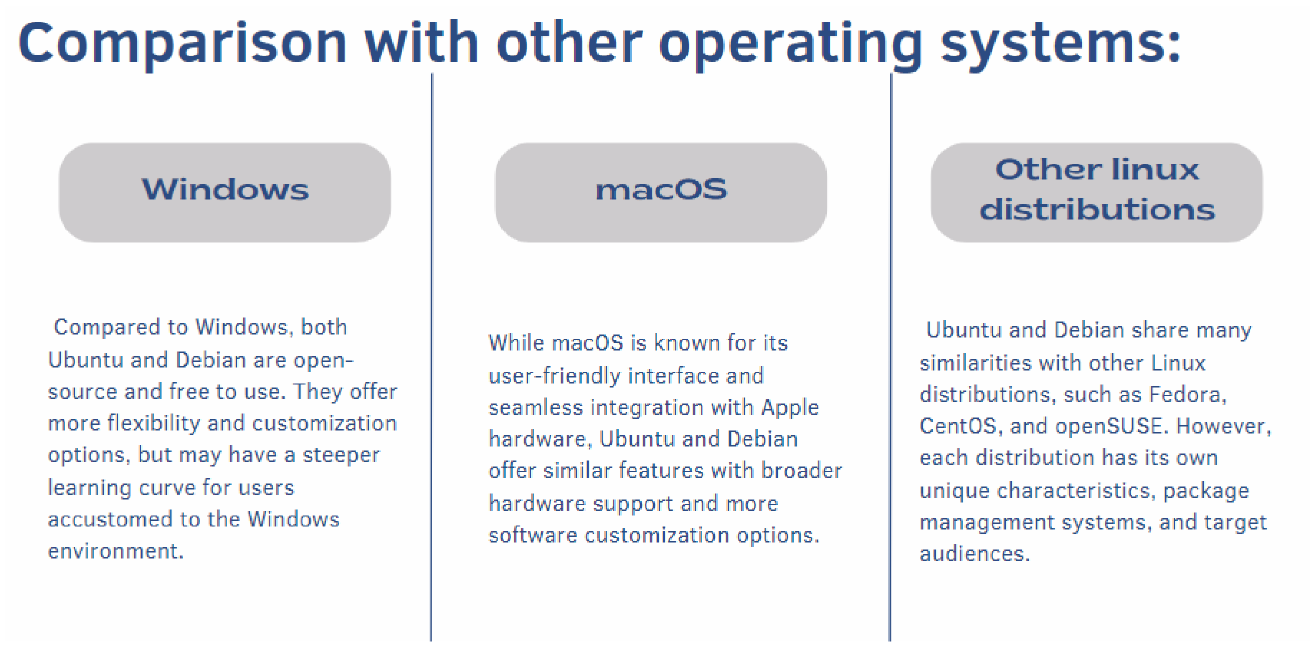
**Ubuntu:**

1. **User-Friendly:** Ubuntu focuses on ease of use, making it accessible to beginners. It comes with a graphical user interface (GUI) that resembles those of other mainstream operating systems, such as Windows and macOS.
2. **Regular Release Cycle:** Ubuntu follows a strict release schedule, with new versions coming out every six months. This ensures users have access to the latest features and improvements on a regular basis.
3. **Large Package Repository:** Ubuntu offers a vast repository of software packages, making it easy to find and install applications for various needs through its package management system, APT (Advanced Package Tool).
4. **Strong Community Support:** Ubuntu has a large and active community of users and developers who provide support, documentation, and assistance through forums, wikis, and other channels.
5. **Commercial Support:** Canonical, the company behind Ubuntu, offers commercial support options for businesses and organizations, including long-term support (LTS) releases with extended maintenance periods.



**Debian:**

1. **Stability:** Debian prioritizes stability and reliability over cutting-edge features. Its release cycle is more conservative compared to Ubuntu, with major releases occurring less frequently.
2. **Wide Hardware Support:** Debian supports a wide range of hardware architectures, making it suitable for various devices, including servers, desktops, and embedded systems.
3. **Community-Driven:** Debian is developed and maintained by a community of volunteers from around the world. It emphasizes principles such as open development, democratic decision-making, and transparency.
4. **Purely Free Software:** Debian is committed to promoting free and open-source software (FOSS). It adheres to strict guidelines regarding software licensing and only includes free software in its official repositories.
5. **Customizability:** Debian provides a high degree of flexibility and customizability, allowing users to configure their systems according to their specific requirements and preferences.



## Comparison with other operating systems:

**Windows**

Compared to Windows, both Ubuntu and Debian are open-source and free to use. They offer more flexibility and customization options, but may have a steeper learning curve for users accustomed to the Windows environment.

**macOS**

While macOS is known for its user-friendly interface and seamless integration with Apple hardware, Ubuntu and Debian offer similar features with broader hardware support and more software customization options.

**Other linux distributions**

Ubuntu and Debian share many similarities with other Linux distributions, such as Fedora, CentOS, and openSUSE. However, each distribution has its own unique characteristics, package management systems, and target audiences.

When comparing Ubuntu, Windows, and macOS for web server usage, several factors come into play. Here's a comparison based on various aspects:

1. **Cost**:
   - Ubuntu: Ubuntu is open-source and free to download and use. There are no licensing fees associated with it.
   - Windows Server: Windows Server comes with licensing fees, which can vary based on the edition and the number of users/devices.
   - macOS Server: macOS Server has a one-time cost, but it's relatively lower compared to Windows Server. However, Apple has shifted its focus away from macOS Server in recent years.

2. **Ease of Use**:
   - Ubuntu: Ubuntu can be more command-line oriented, especially when configuring server components. However, it offers various server management tools and has extensive documentation available.
   - Windows Server: Windows Server typically offers a more user-friendly graphical interface through tools like Windows Admin Center. It may be more intuitive for those familiar with the Windows ecosystem.
   - macOS Server: Historically, macOS Server has been known for its simplicity and user-friendly interface, but Apple has been scaling back its server offerings, and macOS Server is no longer as actively developed.

3. **Software Compatibility**:
   - Ubuntu: Being based on Linux, Ubuntu has strong compatibility with a wide range of server software and tools. Many popular web server software such as Apache, Nginx, and MySQL are well-supported on Ubuntu.

- Windows Server: Windows Server supports a wide range of Microsoft-specific server software and applications. However, some open-source and Linux-based software may require additional configuration or may not be directly compatible.
- macOS Server: While macOS Server supports various services, its compatibility with certain enterprise-grade or industry-standard server software may be limited compared to Ubuntu and Windows Server.

4. **Performance and Stability**:
- Ubuntu: Ubuntu Server is known for its stability and performance, especially in the context of web servers. It's widely used in production environments and benefits from strong community support.
- Windows Server: Windows Server also offers good performance and stability, but it may require more resources compared to Ubuntu for similar tasks. It's often preferred in environments where integration with other Microsoft products is crucial.
- macOS Server: While macOS is stable for typical usage, macOS Server may not be as robust in handling high loads or diverse server tasks compared to Ubuntu and Windows Server.

5. **Security**:
- Ubuntu: Ubuntu Server is known for its robust security features and frequent security updates. Its open-source nature allows for quick identification and patching of security vulnerabilities.
- Windows Server: Windows Server has matured in terms of security features over the years and offers various security enhancements. However, it has historically been a more frequent target for malware and exploits compared to Linux-based systems.
- macOS Server: macOS Server has security features typical of macOS but may not be as extensively hardened or scrutinized in server environments compared to Ubuntu and Windows Server.

## CONCLUSION

- While all three operating systems can serve as a platform for web servers, the choice depends on factors such as cost, familiarity, compatibility, performance, and security requirements of the specific use case
- Ubuntu is often preferred for its cost-effectiveness, robustness, and broad compatibility with server software
- Windows Server is chosen for its integration with the wider Microsoft ecosystem and ease of use for Windows-centric environments.
- macOS Server, while suitable for certain scenarios, may be less commonly used due to its limited feature set and declining support from Apple.