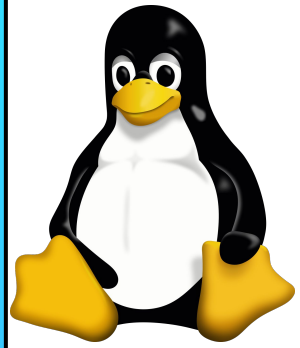


# The Linux Week

Introduction to Linux


# Welcome to Day 8

# Day 8

- Introduction
- What is  a service?
- Examples of a service
- What is a Daemon?
- Examples of Daemons
- Upstart Management
- Systemd Management
- Git Workshop

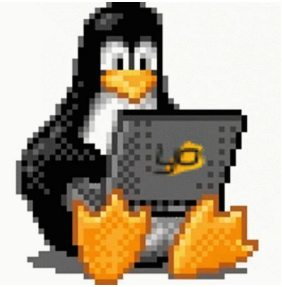

01
Intro
Introduction to Services
and Daemons

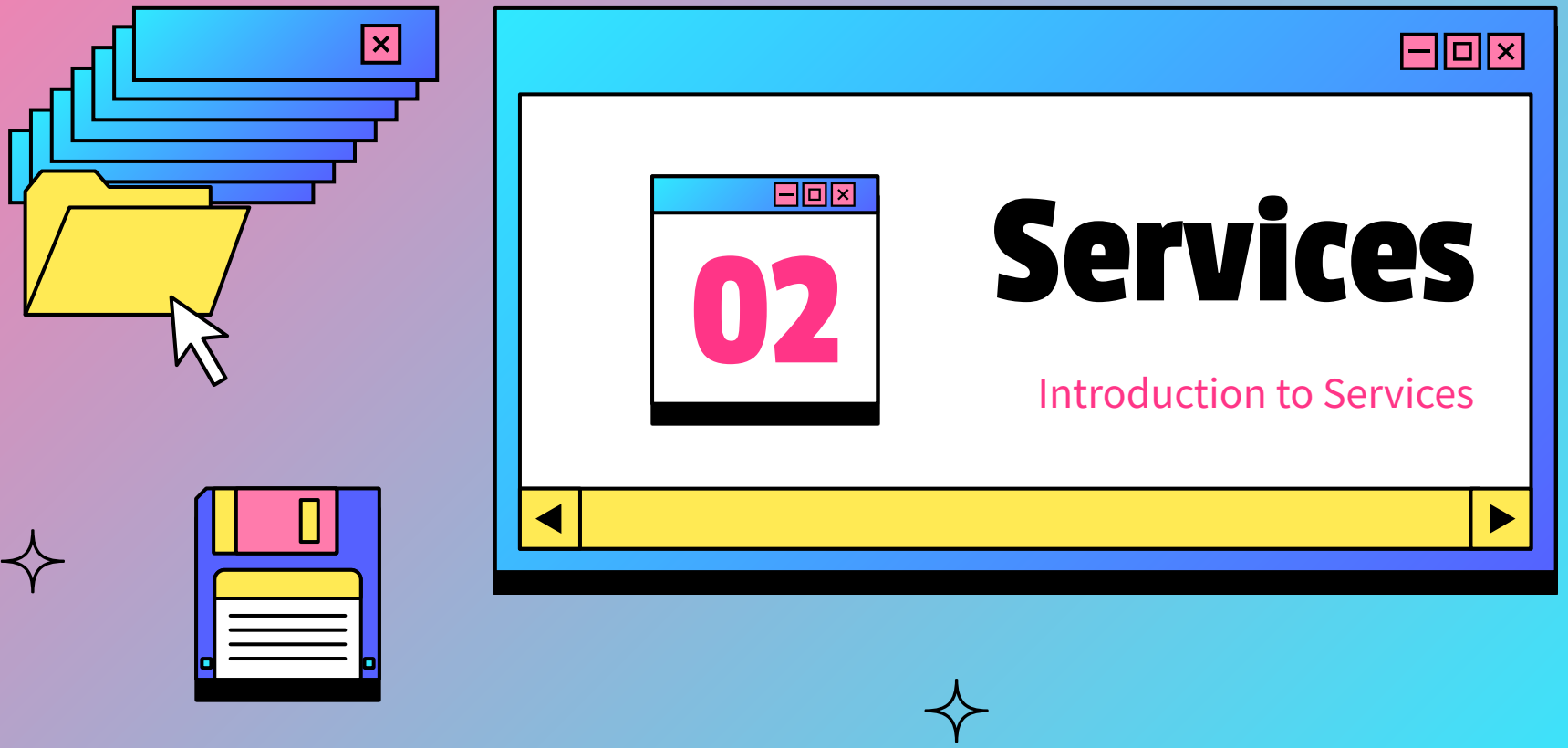
# Introduction

In Linux, services and daemons are crucial components that manage various background processes essential for the smooth operation of the system.

These processes often run in the background and perform tasks such as handling network connections, managing hardware, and providing system logging.

Understanding these components is key to effective system administration and management.




02
Services
Introduction to Services


# Services

A program/process running in the background, providing specific functionality.

# Characteristics

1)

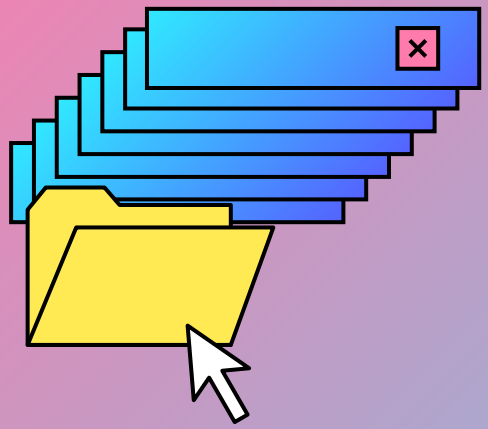
Starts at boot time

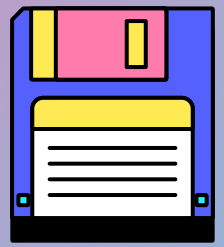
2)

Runs until the system is shut down

3)

Can be managed manually





03

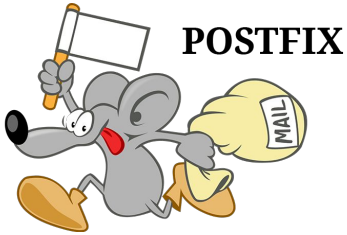
# Examples

Examples of Services

# Examples of Services

## Web Servers

Apache, NGINX




## File Server

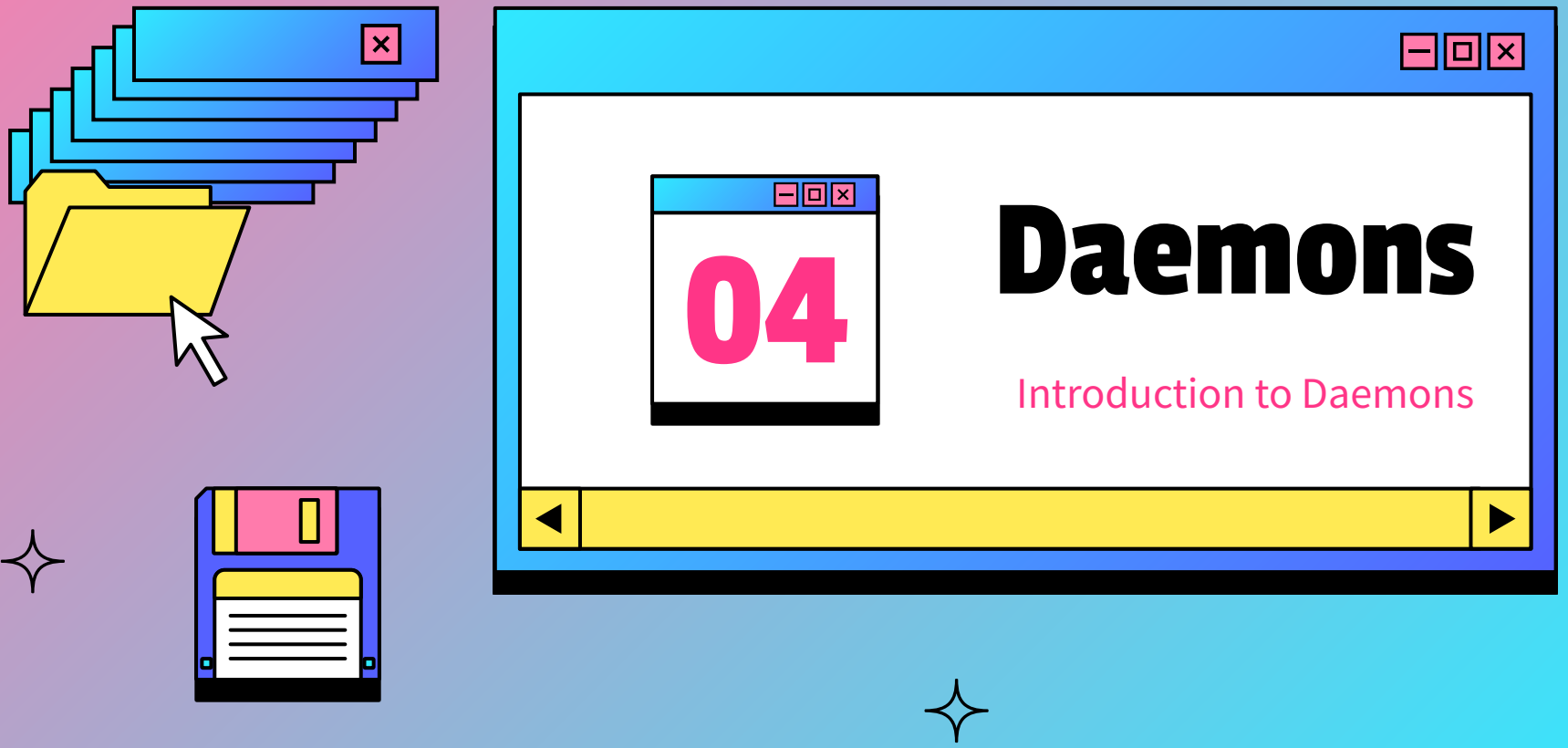
Samba, NFS

## DB Server

MySQL, PostgreSQL

## Mail Server

Postfix, Sendmail


04
Daemons
Introduction to Daemons


# Daemons

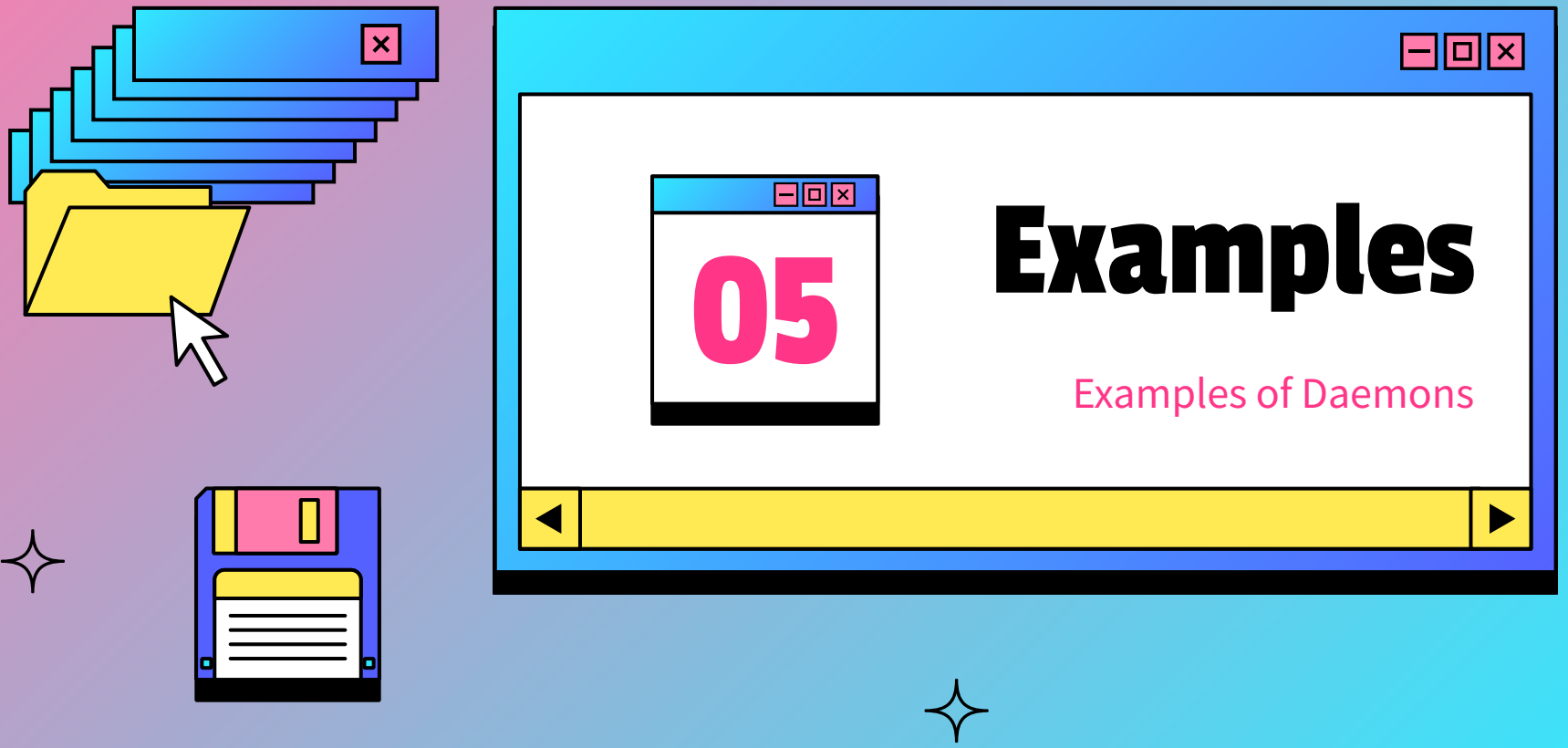
A type of service that runs in the background and performs specific tasks.

# Characteristics

1) Background Execution

2) Long-lived Processes

3) Autonomous operation


05
Examples
Examples of Daemons

# Examples of Daemons

## cron

Schedules and executes periodic tasks

## sshd

Manages SSH connections for remote access

## syslogd

Handles system logging

# Difference?

## Scope

Daemons are a specific type of background process that typically handle system-level tasks, while services can refer to a broader range of background processes, including daemons, that provide specific functionalities.

## User Interaction

Daemons usually do not interact directly with users, whereas services may have user interfaces or interact with users through other means.


06
Upstart
Upstart Management

# Upstart Management

## Start a Service

sudo start service_name

## Stop a Service

sudo stop service_name

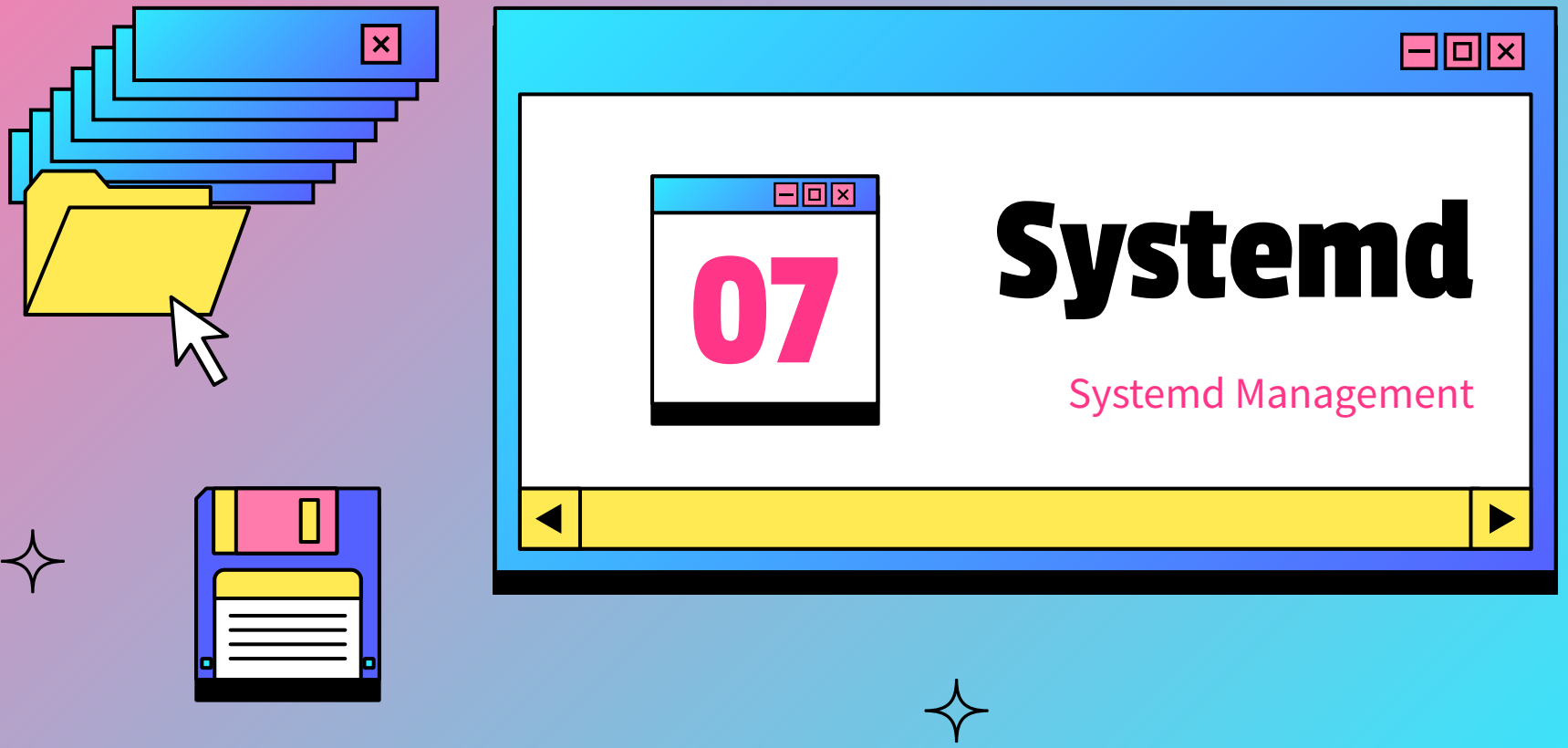
# Upstart Management

## Restart a Service

sudo restart service_name

## Check Status

sudo status service_name


07
Systemd
Systemd Management

# Systemd Management

## Start a Service

sudo systemctl start service_name

## Stop a Service

sudo systemctl stop service_name

## Restart a Service

sudo systemctl restart service_name

# Systemd Management

## Check Status

sudo systemctl status service_name

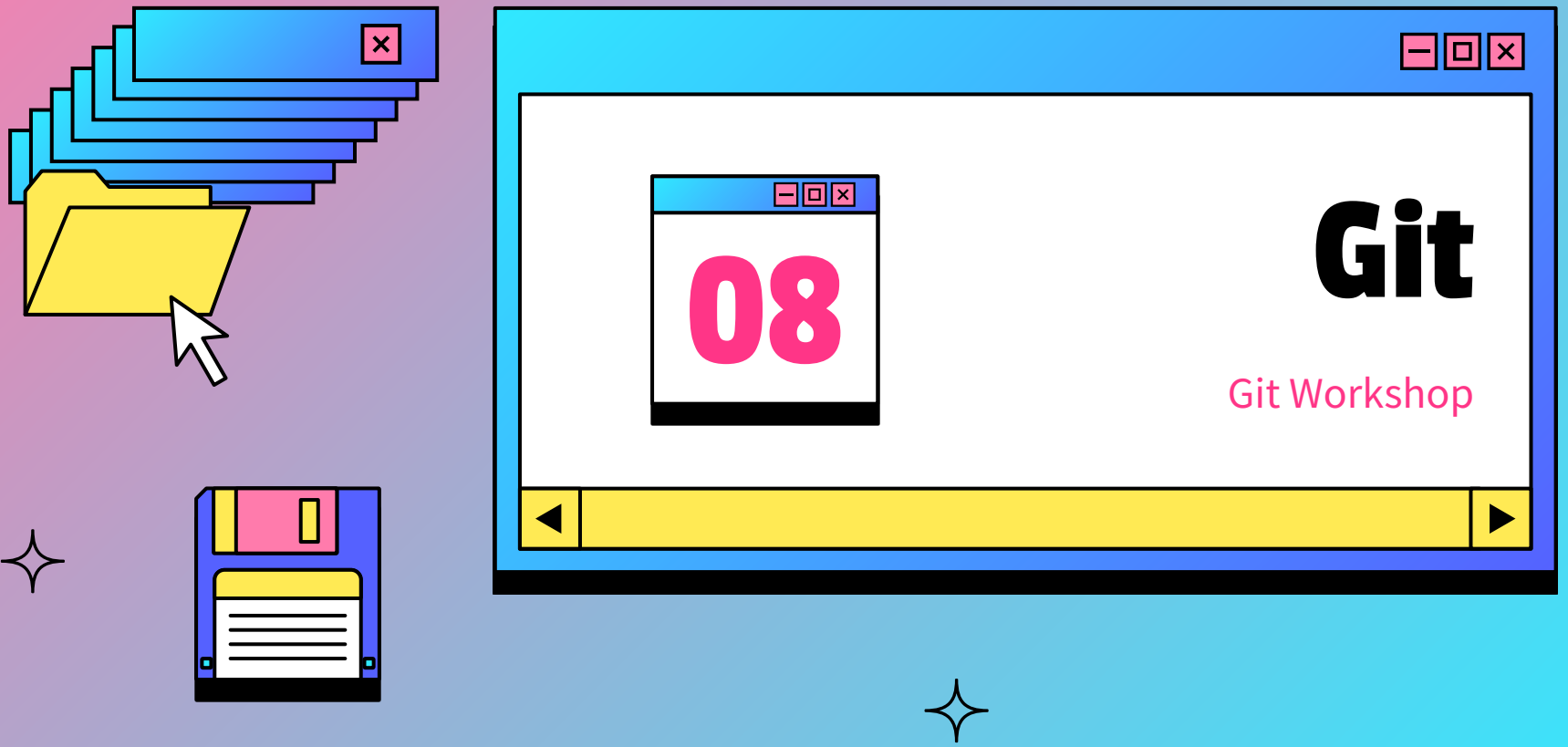
## Disable at Boot

sudo systemctl disable service_name

## Enable at Boot

sudo systemctl enable service_name

# 5 Minute Break


08
Git
Git Workshop

# Git

A distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

# Advantages of Git


## Collaboration

team collaboration with remote repositories (e.g., GitHub, GitLab).

## Track Changes

Tracks history and changes, making it easy to revert to previous states.

## Flexibility

Supports various workflows and branching models

# Basic Commands for Git

**1)**

## Git init

Initialize a new repo

**2)**

## Git clone

Clone existing repo

**3)**

## Git add file

Adding files

**4)**

## Git push

Push changes to remote repo

**5)**

## Git pull

Pull the latest changes from remote repo

**6)**

## Git commit -m

Commit changes with message

# Popular Platforms

1) GitHub

2) GitLab

3) BitBucket


Git Lab
Starting the Git Lab

# New to Git?

Setup your email and username

```
git config --global user.name "John Doe"
git config --global user.email "john.doe@example.com"
```

# New to Git?

Verify your configuration

```
git config --global --get user.name
git config --global --get user.email
```

# Steps to start the Lab - Locally

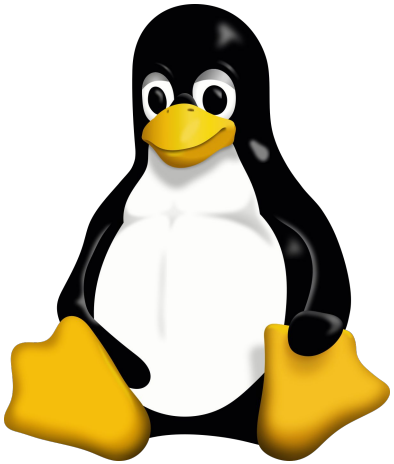
- Create a directory and change to it | mkdir dir - cd dir
- Initialize a git repo | git init
- Create a new file in the directory (README.md) | touch README.md
- Add the README file to the repo | git add README.md
- Commit your changes | git commit -m “read me file added”
- Create a new branch and switch to it | git checkout-b newbranch, git branch new branch
- Make changes to README file | echo “edited line” >> README.md
- Add and commit changes | git add README.md - git commit -m “edited”
- Switch to main or master branch | git checkout master
- Merge the branch | git merge newbranch

# Steps to start the Lab - Remotely

- Add path | git remote add origin https://github.com/your-username/repository-name.git
- Create tokens from your settings -> developer settings
- Add token | git remote set-url origin https://<token>@github.com/your-username/repository-name.git
- Push files | git push -u origin main
- Verify your configuration | git remote -v

# Q/A Session

Thank you !



# End of Day 8!

By Maya Mnaizel