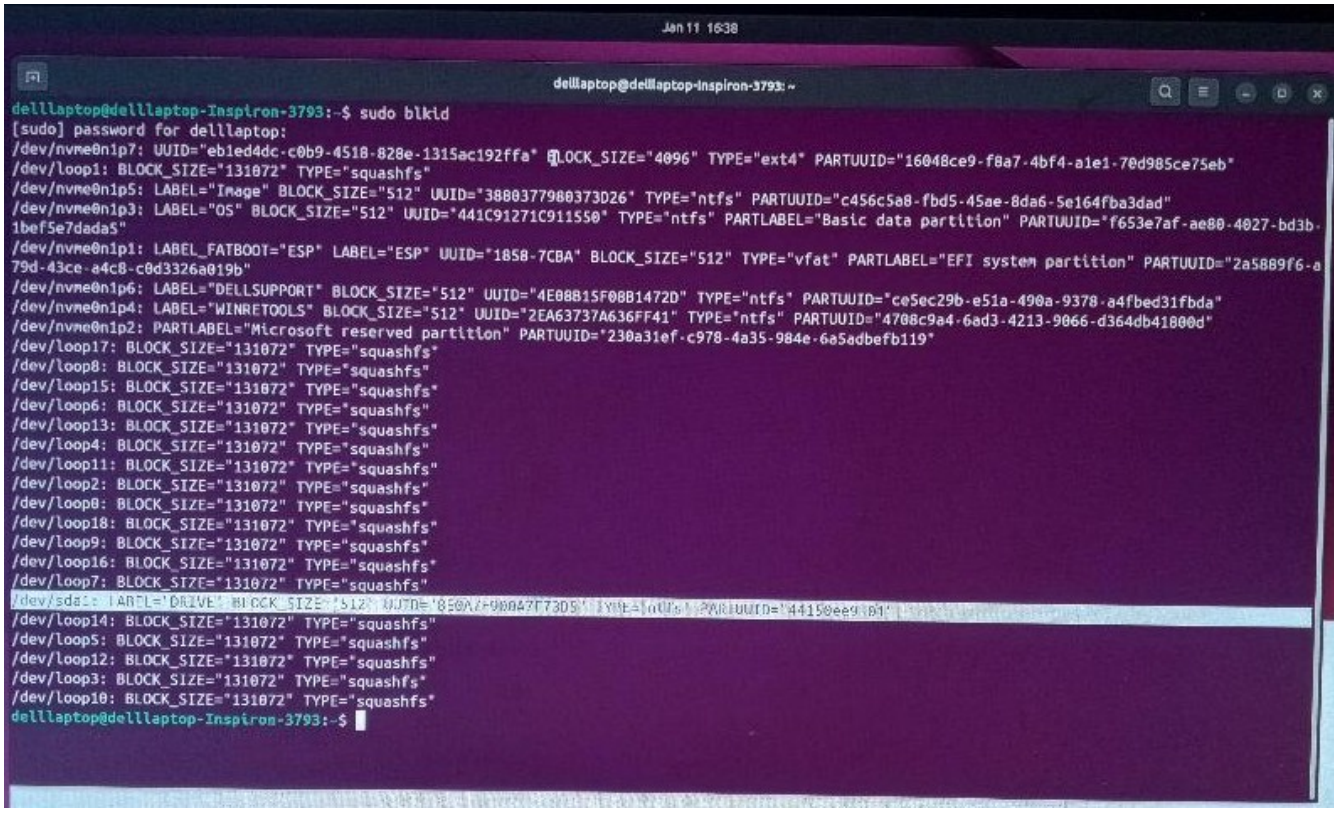
To link an MP4 from an external drive to a webpage on Ubuntu, first **mount the drive** (preferably automatically via `/etc/fstab`), then **create a symbolic link** (symlink) from your web server's root (like `/var/www/html/`) to the external drive's video folder using `ln -s`, ensuring the web server user (`www-data`) has permissions; then, reference the video using the symlinked path in your HTML `<video>` tag.

Step 1: Mount the External Drive (if not already)

1. **Find your drive's UUID:**

```bash
sudo blkid
```

(Note the UUID for your external drive.)
The UUID=”8E0A7F900A7F73D5” and the type is “ntfs”.



2. **Create a mount point:**

```bash
sudo mkdir /mnt/external_media
```

3. **Edit** `/etc/fstab` for auto-mounting (replace `YOUR_UUID` and `ntfs` or `ext4` as needed):

```bash
sudo nano /etc/fstab
```

Add a line like:

`UUID=YOUR_UUID /mnt/external_media <filesystem_type> defaults,nofail 0 2`

```
==========================================================================
#/etc/fstab: static file system information
#
# Use blkid to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name a devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>   <options>   <dump>      <pass>
# / was on /dev/nvme0n1p7 during curtin installation
/dev/disk/by-uuid/eb1ed4dc-c0b9-4518-828e-1315ac192ffa / ext4 defaults 0 1
# /boot/efi was on /dev/nvme0n1p1 during curtin installation
/dev/disk/by-uuid/1858-7CBA /boot/efi vfat defaults 0 1
/swap.img   none  swap  sw    0     0
UUID="2228FF0E28FEDFA9" /var/html/media ntfs-3g defaults 0 0
==========================================================================
```

4. **Mount it:**

```bash
sudo mount -a
```

Step 2: Create a Symbolic Link

1. **Create a symlink** from your web root to the mounted drive's video folder (e.g., `/mnt/external_media/videos/`):

```bash
sudo ln -s /mnt/external_media/videos/ /var/www/html/media
```

(This makes the `/var/www/html/media` folder point to your external drive's video folder).

Step 3: Set Permissions

1. **Grant Apache access** (Ubuntu's default web server user is `www-data`):

```bash
sudo chown -R www-data:www-data /mnt/external_media/videos/
sudo chmod -R 755 /mnt/external_media/videos/
```

This allows Apache to read the files.

Step 4: Link in your HTML

1. **Use the `src` attribute** in your HTML `<video>` tag, referencing the symlink path:

```html
<video width="640" height="360" controls>
  <source src="media/your_video.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

Access the video at `http://localhost/media/your_video.mp4` (or your server's IP/domain).

BASH (Bourne Again SHell) is a widely used **command-line interpreter** and a powerful scripting language for interacting with the operating system in Ubuntu and other Linux distributions. It is the default shell for many Linux systems, serving as the interface where users type commands.

What BASH Is

- **Command Interpreter:** Bash reads commands typed by a user in a terminal and tells the operating system (kernel) what programs to run.
- **Scripting Language:** It allows users to write sequences of commands and logic (known as shell scripts) into a text file to automate repetitive tasks, system administration, and software development processes.
- **Superset of the Bourne Shell (sh):** Developed as a free alternative to the original Unix `sh` shell, Bash is largely compatible with `sh` scripts but includes enhanced features like command-line editing, command history, loops, and conditional statements.

How BASH Relates to Ubuntu and Apache2

- **Ubuntu Default:** Ubuntu typically uses Bash as the default interactive shell when you open a terminal.
- **System Management:** In the context of managing software like the Apache2 web server on Ubuntu, system administrators and developers use Bash commands and scripts for essential tasks:
  - **Installation:** Using the `apt install apache2` command in the Bash terminal.
  - **Service Management:** Starting, stopping, and restarting the Apache service using commands like `sudo systemctl restart apache2` or `sudo service apache2 status`.
  - **Automation:** Writing Bash scripts to automate the creation of Apache virtual hosts, log file management, or to monitor the server's status.

In short, Bash is the primary tool you use to type instructions to your Ubuntu system to install, configure, and manage applications like Apache2.

In Ubuntu with Apache2, **`nano` is a simple, user-friendly command-line text editor used to view and edit configuration files (like Apache's virtual hosts in `/etc/apache2/`) or website content (like `index.html` in `/var/www/`) directly in the terminal**, allowing you to make changes without a graphical interface, using shortcuts like `Ctrl+X` to exit and save.

What it is:

- A basic, easy-to-use text editor for Linux/Unix systems, often pre-installed.

How it relates to Apache2:

- When you install Apache2 on Ubuntu, you often need to edit its settings or your website's files, such as:
  - Creating new virtual host files: `sudo nano /etc/apache2/sites-available/your-site.conf`.
  - Editing your main website file: `sudo nano /var/www/html/index.html`.
- It provides a visual way (within the terminal) to manage text-based configuration files, unlike editing machine code directly.

Key Commands (shown at the bottom of the nano screen):

- `^X` (Ctrl+X): Exit.
- `^O` (Ctrl+O): Write Out (Save).
- `^W` (Ctrl+W): Where Is (Search).

Essentially, `nano` is your tool for editing text files when you're working on a server without a graphical desktop environment.