Linux Ubuntu 26.04 LTS: Installation & Customize.

; 22.06.2026.20.25.wib [Indonesia]

Hello! Ubuntu 26.04 LTS (Resolute Raccoon) was just officially released on April 23, 2026. As a Long-Term Support (LTS) release, this version brings many major changes, system modernizations, and significant security improvements.

Ubuntu 26.04 LTS (code-named **"Resolute Raccoon"**) is a fantastic release. It brings a sleek new desktop environment experience (GNOME 50) and incredible stability that will be supported until 2031.

Setting it up and making it feel like home is a straightforward process.

Part 1: The Installation Process

Before starting, back up your important data and grab a USB flash drive (at least 12 GB). Your computer needs a 2 GHz dual-core processor, a recommended 8 GB of RAM, and 25 GB of free drive space.

1. Download the ISO

5 mins

Go to the official Ubuntu website and download the Ubuntu 26.04 LTS Desktop ISO file.

2. Flash the USB Drive

10 mins

Download a tool like Rufus (Windows) or balenaEtcher (Cross-platform) from Ubuntu 26.04 LTS. Plug in your USB, select the downloaded Ubuntu ISO, and hit write/flash. Warning: This wipes the USB.

3. Boot from USB

2 mins

Restart your computer and tap your boot menu key (usually F12, F9, Esc, or Del depending on your motherboard). Select your USB drive from the menu.

4. Run the Installer

15-20 mins

When the screen loads, choose Interactive Installation. Follow the prompts to select your language, connect to Wi-Fi, and choose Erase disk and install Ubuntu (for a clean setup) or Manual partitioning if you are dual-booting. Fill out your user account info and let it finish!

Part 2: Customizing Your New Desktop

Once you boot into your fresh installation, it is time to turn it into a personalized powerhouse.

1. Update the System First

Before changing anything visual, update your core files. Open the terminal (Ctrl + Alt + T) and paste this single line to refresh your repositories and apply upgrades:

```
$ sudo apt –y update && sudo apt upgrade
```

2. Tweak the Desktop Interface

Ubuntu 26.04 comes with a built-in Settings app that handles most cosmetic changes. Go to Settings > Appearance to adjust:

Style: Toggle between Light and Dark modes easily. You can also pick a accent color that changes system highlights.

The Dock: Change the dock icon size, hide it automatically when windows overlap (Intellihide), or turn off panel mode so it floats beautifully at the bottom of your screen like a Mac.

3. Install GNOME Tweaks & Extensions

To get real power over your customization, you want to unlock advanced settings. Run this command in your terminal:

```
$ sudo apt -y install gnome-tweaks gnome-shell-extension-manager
```

Open **Extension Manager** from your apps menu. From here, you can browse and install community-made extensions directly without needing a browser plugin.

Recommended Extension	What It Does
Dash to Dock	Replaces the standard Ubuntu dock with a highly customizable floating dock.
Blur my Shell	Adds a beautiful modern blur effect to overview menus and panels.
Just Perfection	Lets you hide or move elements of the UI (like the clock or top panel).

4. Switch Up Your Icons and Themes

If you want to move away from the stock orange-and-grey Yaru theme, you can download custom themes from websites like GNOME-Look.org.

Download your theme files.

Extract them into .themes or .icons folders in your Home directory (press Ctrl + H in the file manager to see hidden folders).

Open the GNOME Tweaks app you just installed to apply them under the Appearance tab.

Part 3: Want to know how to safely dual-boot Ubuntu 26.04 alongside Fedora 44 ??

Dual-booting two Linux distributions is a bit different from dual-booting with Windows. Because both Ubuntu 26.04 and Fedora 44 use GRUB as their bootloader, they can sometimes step on each other's toes during system updates if they aren't configured with care.

To prevent issues, the gold standard is to let one distribution fully control the primary boot menu. Since Ubuntu's GRUB setup is generally more aggressive at keeping its boot menu updated via os-prober, the smoothest approach is to install Fedora first, and then let Ubuntu handle the final boot management.

The Golden Rules for a Safe Dual-Linux Boot

Rule 1: Stick to UEFI Mode. Ensure your motherboard's BIOS is set to UEFI mode, not Legacy/CSM. Both modern distros thrive here.

Rule 2: Share the EFI Partition (But Don't Format It!). Both systems will place their tiny boot files inside your computer's existing EFI system partition. When installing the second OS, never format this partition, or you will erase the first one.

Rule 3: Keep Root Partitions Separate. Fedora 44 defaults to the Btrfs file system, while Ubuntu 26.04 uses Ext4 (or ZFS). Give each OS its own distinct partition block.

Step-by-Step Dual-Boot Walkthrough

Assuming you are starting fresh, or already have Fedora 44 up and running, follow this sequence:

1. Carve Space Inside Fedora

10 mins

Boot into Fedora 44. Open the Disks utility (or GParted). Select your primary drive, shrink your main Fedora partition, and leave the newly

created space as Unallocated / Free Space. You'll want at least 30–40 GB for Ubuntu.

2. Boot the Ubuntu 26.04 Live USB

5 mins

Plug in your Ubuntu flash drive, restart your machine, and access your motherboard's boot menu. Make sure to select the option prefixed with UEFI: [Your USB Name].

3. Navigate to Manual Partitioning

5 mins

Launch the Ubuntu installer. When you reach the "Installation Type" screen, do not click "Erase disk". Instead, select Manual Partitioning (sometimes labeled "Something Else").

4. Assign the Mount Points

10 mins

5. Finish Installation & Update GRUB

15 mins

Proceed with the installation. Once finished, reboot your computer. You will likely boot straight into Ubuntu. Open a terminal (Ctrl + Alt + T) and run:

Ubuntu's GRUB will scan the disk, find Fedora 44, and add it neatly to your startup menu.

What Happens During System Updates?

Every now and then, Fedora will update its Linux kernel and want to rewrite the boot priority so it controls the startup screen. If Fedora "steals" the bootloader screen and you notice Ubuntu disappears from the list, don't panic!

You can easily change it back right from your motherboard BIOS settings:

Tap your BIOS key (F2, F12, or Del) right as the computer turns on.

Go to the Boot Priority or Boot Order tab.

Move ubuntu to the very top of the list above fedora. Save and exit.

Ubuntu's GRUB interface will safely host both options without a hitch.

Part 4: Costumize Ubuntu 26.04 LTS: exfatprogs

While modern versions of Linux (including Ubuntu 26.04 LTS) can natively read and write to exFAT drives right out of the box using the kernel, you need the exfatprogs package if you want the power to maintain, check, label, or format those drives.

⚠ Important Note: In older versions of Ubuntu, people used exfat-utils or exfat-fuse. Those packages are now legacy and deprecated. For Ubuntu 26.04 LTS, exfatprogs is the modern standard toolset developed explicitly for the official Linux kernel driver.

1. Installing exfatprogs

The package lives in Ubuntu's default software repositories. Open your terminal (Ctrl + Alt + T) and run the following commands to update your package list and install the tools:

$ sudo apt update

$ sudo apt –y install exfatprogs

2. Managing Your exFAT Drives (How to Use It)

Once installed, you don't interact with exfatprogs as a single app. Instead, it unlocks several highly useful backend terminal commands for managing your storage devices.

A. Formatting a Drive to exFAT (mkfs.exfat)

If you have a fresh USB drive or SD card and want to format it so it works seamlessly on Windows, Mac, and your Ubuntu machine, use this command.

1. First, find your drive's identifier (e.g., /dev/sdb1) by running:

$ lsblk

2. Once you are 100% sure of the drive name, format it (replacing MyFlashDrive with your preferred name and /dev/sdXn with your actual partition):

$ sudo mkfs.exfat -L "FlashDriveOfme" /dev/sdXn

B. Checking and Repairing an exFAT Drive (fsck.exfat)
If you unplugged your external hard drive without safely ejecting it and it’s acting glitchy or refusing to mount, you can scan and repair its file integrity.

$ sudo fsck.exfat -y /dev/sdXn
(The -y flag automatically answers "yes" to any repair prompts it finds).

C. Changing the Drive Label Without Formatting (exfatlabel)
If you are tired of your USB drive showing up in the file manager as an ugly string of random letters and numbers, you can rename the volume on the fly:

To see the current name:
$ sudo exfatlabel /dev/sdXn

To give it a new name:
$ sudo exfatlabel /dev/sdXn "BackupDrive"

3. Customizing via Graphical Interface (GUI)

If you prefer not to use the terminal to manage your exFAT drives, installing exfatprogs fixes hidden errors in your system's GUI apps.

Now, when you open the pre-installed Disks utility or download GParted from the App Center, the option to format to exFAT or click the gear icon to "Repair Text System" will no longer be greyed out or throw a missing dependency error.

Part 5: Install Server apache2

Setting up an Apache web server (the apache2 package) on Ubuntu 26.04 LTS turns your computer into a local staging environment or a production-ready server capable of hosting websites.

Here is how to install, configure, firewall-protect, and test your Apache installation.

1. Installation

Apache is available in Ubuntu’s default software repositories. Run the following commands to update your package index and install the server:
$ sudo apt update
$ sudo apt –y install apache2

Once the installation finishes, Apache will start automatically. You can verify that the service is running by checking its system status:
Look for a green line that says active (running).

2. Configure the Firewall (UFW)

Before your server can serve web pages to other computers on your network (or the internet), you need to open up the web ports in Ubuntu's built-in Uncomplicated Firewall (UFW).

Apache registers itself with UFW upon installation, providing a few different profiles:

Apache: Opens port 80 (Normal, unencrypted HTTP web traffic).

Apache Full: Opens both port 80 (HTTP) and port 443 (Encrypted HTTPS traffic).

To verify the changes, check the firewall status:
$ sudo ufw status

Apache Secure: Opens only port 443 (HTTPS).

For a fresh setup, it is best to allow Apache Full:
$ sudo ufw allow 'Apache Full'

To verify the changes, check the firewall status:
$ sudo ufw status

3. Test Your Web Server

With the firewall configured, you can verify that Apache is broadcasting correctly.

Find your server's local IP address by typing:
$ hostname -I

Open a web browser on your Ubuntu machine (or any other device on the same local network).

Type your server's IP address into the URL bar (e.g., http://111.111.1.11 or simply http://localhost if you are testing on the same machine).

You should see the default Ubuntu Apache2 Web Server Default Page. This page confirms that the server is installed properly and ready to host files.

4. Where to Put Your Website Files

Now that the server is live, you can start customizing the content it serves.

The Root Directory: By default, Apache looks for website files inside the directory located at /var/www/html/. The default page you saw in the previous step is stored here as index.html.

Adding Files: Because /var/www/html/ is a system directory, you will need root privileges to modify files inside it via the terminal. For example, to create a custom homepage:
$ sudo vim /var/www/html/index.html
Not found
$ sudo apt –y install vim

5. Essential Management Commands

When customizing your server configurations or adding new modules, you will frequently need to manage the underlying Apache service using systemctl:

Stop the server:
$ sudo systemctl stop apache2

Start the server:
$ sudo systemctl start apache2

Restart the server (completely stops and restarts the service, useful for major configuration updates):
$ sudo systemctl restart apache2

Reload the server (applies configuration changes instantly without dropping active website connections):
$ sudo systemctl reload apache2

Enable auto-start at boot:
$ sudo systemctl enable apache2

Part 6: Install PHP

Perfect. Installing PHP alongside Apache turns your web server into a dynamic environment capable of running frameworks like WordPress, Laravel, or custom backend scripts.

Since you are running Ubuntu 26.04 LTS, matching it up with PHP 8.5 is the ideal choice for modern performance and security features.

01. Installing PHP 8.5 and Extensions

Your command is spot on. Let's run the installation to get the core PHP runtime and its most critical extension modules (handling MySQL databases, ZIP compression, image processing, multibyte strings, curl requests, XML parsing, and FastCGI Process Manager):
$ sudo apt update

$ sudo apt install php8.5 php8.5-cli php8.5-common php8.5-mysql php8.5-zip php8.5-gd php8.5-mbstring php8.5-curl php8.5-xml php8.5-fpm -y

To confirm that the PHP command-line interface is installed properly and check your active version, run:
$ php -v

02. Make Apache Prioritize PHP Files

By default, if a directory contains both an index.html and an index.php file, Apache will serve the static HTML file first. To change this behavior so that your dynamic PHP scripts take precedence, you need to edit Apache's configuration.

1. Open the configuration file in the nano text editor:

$ sudo vim /etc/apache2/mods-enabled/dir.conf

2. Look for a block of code that looks like this:

<IfModule mod_dir.c>

    DirectoryIndex index.html index.cgi index.pl index.php index.pias index.shtml index.xhtml index.htm

</IfModule>

3. Move index.php to the very front of the list, right after DirectoryIndex:

<IfModule mod_dir.c>
    DirectoryIndex index.php index.pias index.html index.cgi index.pl index.shtml index.xhtml index.htm
</IfModule>
4. Press ESC+:wq then Enter to save the file, AND exit the editor.

03. Restart Apache to Apply Changes

Whenever you modify Apache configurations or add new modules like PHP, you must restart the web server daemon to hook everything together:

$ sudo systemctl restart apache2

04. Test PHP Integration on Your Server

The best way to verify that Apache is communicating properly with the PHP 8.5 preprocessor is to create a quick info script in your web root directory.

1. Create a file called info.php inside the public folder:
$ sudo vim /var/www/html/info.php

2. Paste the following line of PHP code into the file:
<?php phpinfo(); ?>

3. Save and close the file

4. Open your web browser and navigate to your server's IP address or localhost followed by /info.php:

http://127.0.0.1/info.php

You should see a purple-and-white standard PHP diagnostics web page displaying all information regarding your PHP 8.5 version, loaded extensions, and system paths.

⚠ Security Tip: Once you have verified that everything works perfectly, it is highly recommended to delete this file. It exposes critical configuration details about your server setup to the public internet. Remove it by running:
$ sudo rm /var/www/html/info.php

Part 7: Install Firefox Developer Edition

Your repository setup is almost perfect, but there are two critical things we need to fix right now before you run apt install.

First, there is a small typo in the URL from your echo command compared to your vim command. Second, and most importantly, Ubuntu 26.04 LTS prioritizes its own transitional packages (which force

the Snap version of Firefox) over third-party repositories. If we don't set up APT pinning, Ubuntu will ignore your Mozilla repository completely!

Here is how to fix the repository and successfully install Firefox Developer Edition.

01. Add/Fix the Repository

In your terminal block, your initial echo command missed the https:// prefix, but your vim command correctly included it. Let's make sure /etc/apt/sources.list.d/mozilla.list has the exact, correct line.

Run this to overwrite it cleanly:
$ sudo echo "deb [signed-by=/etc/apt/keyrings/packages.mozilla.org.asc] https://packages.mozilla.org/apt mozilla main" | sudo tee /etc/apt/sources.list.d/mozilla.list > /dev/null

02. Create the APT Pinning Configuration (Crucial Step)

If you run apt install firefox-devedition right now, Ubuntu's default configuration will override it. We need to tell the APT package manager to prioritize Mozilla’s official repository over everything else.

Create and open a new configuration file:
$ sudo vim /etc/apt/preferences.d/mozilla

Paste the following block exactly as it is:

Package: *
Pin: origin packages.mozilla.org
Pin-Priority: 1000

This tells Ubuntu that any package coming from packages.mozilla.org gets a priority score of 1000, forcing the system to prefer it over native Ubuntu packages.

Save and close the file.

03. Update APT and Install Firefox Developer Edition

Now that the repository is fixed and the priority rules are set, tell APT to look at the new packages and install the developer build:

$ sudo apt update
$ sudo apt -y install firefox-devedition

04. Optional: Set Up Language Packs

If you want your browser to use a specific language interface other than standard US English (for example, French or Spanish), you can optionally install your specific language pack using:
$ sudo apt –y install firefox-devedition-l10n-fr

(Simply change fr at the end of the package name to your preferred country/language code).

Or using firefox-devedition-l10n-gb ensures you get the proper British English localization (dictionaries, spelling, and UI layout) right inside the developer build.

Part 8: Install MariaDB

If you want run mariadb-server, run apt -y install maria-server, the system will return a "Package not found" error. Here is the correct command and the essential post-installation steps to lock it down securely.

01. Install MariaDB Server

Since you are running directly as root, paste this exact line:
$ sudo apt -y install mariadb-server

Once the installation completes, verify that the database server daemon is active and running:
$ sudo systemctl status mariadb
$ sudo systemctl start mariadb

02. Secure the Installation (Crucial Step)

By default, a fresh MariaDB installation comes with open settings (like blank root passwords and anonymous user access) that shouldn't be left as-is. Run the built-in security script to close these vulnerabilities:

pias@parlind-desktop:~$ sudo mysql_secure_installation

The script will walk you through a series of security prompts. Type Y (Yes) for all of them to apply standard security practices:

* Enter current password for root: Press Enter (it is blank by default).

* Switch to unix_socket authentication: Type Y. (This allows you to log in instantly using sudo mariadb from your system terminal, removing the need for a separate hardcoded root password).

* Change the root password? Type N (if you enabled unix_socket above, a root password isn't necessary for local access).

* Remove anonymous users? Type Y.

* Disallow root login remotely? Type Y.

* Remove test database and access to it? Type Y.

* Reload privilege tables now? Type Y.

03. Log In and Test Your Database Engine

Because Ubuntu 26.04 uses the secure unix_socket method by default, you do not use passwords to log in locally as root. Instead, simply execute the mariadb command directly from your root shell:

mariadb

You will see the terminal prompt change to the database interface:

MariaDB [(none)]>

To see a list of default system databases and confirm your connection is fully operational, run:

SQL

SHOW DATABASES;

To close the database shell and return to your primary terminal prompt, type:

exit;

Part 9: Costumize Apache2

Now that your core LAMP stack (Apache, PHP 8.5, and MariaDB) is installed, customizing Apache allows you to optimize its performance, cleanly organize multiple websites, and hide server signatures for better security.

Here are the most important configurations to optimize your Apache setup on Ubuntu 26.04 LTS.

01. Create a Custom Virtual Host (Hosting Multiple Sites)

Instead of dropping all your project files directly into the default /var/www/html folder, you should create a dedicated Virtual Host. This allows you to host multiple distinct websites (e.g., siteofMe.local, testblog.dev) on a single server.

Let's set up a custom layout for a site called projectOfMe.local:

Step A: Create the Directory Structure

root@parlind-desktop:~$ sudo mkdir -p /var/www/projectOfMe/public_html

Assign ownership of this directory to your regular system user (replace youruser with your actual Ubuntu username) so you can edit files without typing sudo every time:

$ sudo chown -R youruser:youruser /var/www/projectOfMe/public_html
$ sudo chmod -R 755 /var/www

Step B: Create a Virtual Host Configuration File

Apache stores its site profiles in /etc/apache2/sites-available/. Create a new configuration file for your project:

$ sudo vim /etc/apache2/sites-available/projectOfMe.conf

<VirtualHost *:80>
    ServerAdmin webmaster@projectOfMe.local
    ServerName myproject.local

```
    ServerAlias www.projectOfMe.local
    DocumentRoot /var/www/projectOfMe/public_html


    <Directory /var/www/projectOfMe/public_html>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/projectOfMe_error.log
    CustomLog ${APACHE_LOG_DIR}/projectOfMe_access.log combined
</VirtualHost>
```

Save and close

Step C: Enable Your New Site and Disable the Default Site

Turn on your new configuration using Apache's built-in tools, and disable the out-of-the-box welcome page:

```
$ sudo a2ensite projectOfMe.conf
$ sudo a2dissite 000-default.conf
```

02. Enable Crucial Apache Modules

For modern web development—especially when building custom PHP applications or running frameworks like WordPress—you need to enable standard modules like URL rewriting and security adjustments.

Run these commands to turn on standard modules:

```
$ sudo a2enmod rewrite    # Enables clean URLs, permalinks, and .htaccess routing
$ sudo a2enmod headers    # Allows customization of HTTP response headers
$ sudo a2enmod expires    # Enables browser caching mechanisms for speed
```

03. Harden Server Security (Hide Server Specs)

By default, Apache broadcasts its exact version number and operating system details in HTTP response headers to anyone who pings the

server. This is a security risk because attackers can use it to target known vulnerabilities.

Let's turn this off. Open the primary security configuration file:
$ sudo vim /etc/apache2/conf-available/security.conf

Find these lines and change their values to match the following:

ServerTokens Prod
ServerSignature Off

ServerTokens Prod tells Apache to only return "Apache" as the server name instead of "Apache/2.4.x (Ubuntu)".

ServerSignature Off removes the server version footer line from default 404 or 403 error pages.

Save and close the file.

04. Verify Configuration and Apply Changes

Before you restart the Apache service, always test your syntax to make sure you didn't leave behind any typos or broken brackets.

Run the configuration test:
$ sudo apache2ctl configtest

If you see Syntax OK, your configuration is perfect. You can now safely reload Apache to push all your adjustments live:
$ sudo systemctl restart apache2

05. Customize view of 'index of'

This was an excellent custom tweak. Modifying the <Directory /var/www/html> block in your default file ensures that your root web directory is fully unlocked for modern development.

Here is exactly what you just enabled by making those changes:

Breakdown of Your Customizations

AllowOverride All (The Most Important Change): By default, Ubuntu sets this to None. Changing it to All tells Apache to honor .htaccess files inside your web folders. This is mandatory for enabling clean SEO URLs, custom router redirects, and permalinks in modern PHP frameworks like WordPress, Laravel, or Symfony.

Options Indexes: This allows directory browsing. If a folder doesn't have an index.php or index.html file, Apache will display a visual list of all the files inside that directory instead of throwing a 403 Forbidden error.

Options FollowSymLinks: This permits Apache to follow symbolic links (shortcuts) created within your filesystem, allowing you to easily link assets or folders outside your primary web root without copying files.

Options MultiViews: This enables content negotiation. For example, if a user requests http://localhost/about, Apache will automatically look for a file named about.html or about.php even if the extension isn't typed out in the browser URL bar.

Require all granted: Explicitly tells Apache to let anyone access files from this directory path, preventing unwanted permission errors.

Step 1: Make Sure Your Text Looks Like This

Open your editor if you haven't saved it yet:
$ sudo vim /etc/apache2/sites-available/000-default.conf

Ensure your new block sits neatly inside the main <VirtualHost *:80> wrapper like this:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    <Directory /var/www/html>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

[Apache2]

vim /etc/apache2/mods-available/autoindex.conf

IndexOptions FancyIndexing VersionSort HTMLTable NameWidth=* DescriptionWidth=* Charset=UTF-8

delete HTMLTable

To be

IndexOptions FancyIndexing VersionSort FoldersFirst NameWidth=* DescriptionWidth=* Charset=UTF-8
ESC+:wq
To save file.
Restart Apache2
$ sudo systemctl restart apache2

Step 2: Run a Config Test and Reload

Whenever you touch file parameters inside sites-available, you must verify your syntax rules and let the server daemon parse the update.

01. Test your setup for any missing brackets or syntax slips:

$ sudo apache2ctl configtest

02. If the terminal returns Syntax OK, restart the service to apply your new directory overrides globally:

$ sudo systemctl restart apache2

Part 10: Install phpmyadmin

To manage your MariaDB databases from a clean, web-based graphical interface rather than the command line, installing phpMyAdmin is the perfect next step.

Because Ubuntu 26.04 LTS handles phpMyAdmin a bit differently than older versions (it is best installed from the official repository or directly to ensure compatibility with PHP 8.5), follow these exact steps to get it running and securely linked to Apache.

1. Install phpMyAdmin

Run the installation command from your root terminal:

$ sudo apt -y update
$ sudo apt -y install phpmyadmin

Navigating the Installation Prompts:

During the installation, your terminal will open a couple of blue configuration screens. Use your keyboard to answer them exactly like this:

Web server to configure automatically: Press the Spacebar to put a star next to apache2, then press Tab to select <OK> and hit Enter. (If you don't hit spacebar to check the box, Apache won't load phpMyAdmin!)

Configure database for phpmyadmin with dbconfig-common? Select <Yes> and press Enter.

MySQL application password for phpmyadmin: Press Enter to leave it blank. The installer will automatically generate a secure random password for internal use.

2. Link phpMyAdmin to Apache (If Needed)

The installer usually creates a symlink automatically, but let's make absolutely sure Apache knows where to find phpMyAdmin by manually verifying or linking its configuration configuration block.

Run this command to safely link the configuration file:
$ sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf-available/phpmyadmin.conf

Now, tell Apache to enable this new configuration module:

a2enconf phpmyadmin

systemctl restart apache2

3. Create a Dedicated Database Admin User

By default, MariaDB uses unix_socket authentication for the root user. This means root cannot log into phpMyAdmin via a web browser password interface.

To fix this securely, do not change your root password. Instead, log into MariaDB via terminal and create a dedicated administrative user with full permissions just for phpMyAdmin:

01. Open your database shell:

$ mariadb

02. Create a new user (replace adminuser and YourSecurePassword with your own choices):

CREATE USER 'adminuser'@'localhost' IDENTIFIED BY 'YourSecurePassword';

03. Grant the new user complete administrative privileges:

GRANT ALL PRIVILEGES ON *.* TO 'adminuser'@'localhost' WITH GRANT OPTION;

04. Clear the cached privileges and exit:

FLUSH PRIVILEGES;

EXIT;

4. Access the Interface

Open your web browser and navigate to your server's local address or localhost followed by /phpmyadmin:

http://127.0.0.1/phpmyadmin

You will see the official phpMyAdmin login dashboard. Use the credentials for the adminuser you created in Step 3 to log in.

5. Recommended Optimization for PHP 8.5

Because PHP 8.5 introduces stricter error handling rules, phpMyAdmin might occasionally throw small warning messages at the bottom of your screen regarding temporary folder directories.

To clean this up, create a dedicated temporary storage directory for phpMyAdmin:

```
$ sudo mkdir -p /var/lib/phpmyadmin/tmp
$ sudo chown -R www-data:www-data /var/lib/phpmyadmin/tmp
```

Next, open phpMyAdmin's configuration file:

```
$ sudo vim /etc/phpmyadmin/config.inc.php
```

Find or add this line anywhere near the bottom:

```
$cfg['TempDir'] = '/var/lib/phpmyadmin/tmp';
```

Save and close (Ctrl + O, Enter, Ctrl + X). Your stack is now fully customized, optimized, and ready for development!!!

by: piasDraco [/pias : pias25]