1. **Steps and understanding for creating QA env and production environment for Laravel application:  
     
   Staging environment:**
2. Ensure your ubuntu system is up to date and install required packages like PHP, MySQL, and composer.  
   sudo apt update && sudo apt upgrade -y

sudo apt install apache2 php libapache2-mod-php php-mysql php-xml php-mbstring php-curl unzip git -y

sudo apt install mysql-server -y

1. Create a MySQL Database and user for Laravel.

In a production environment, your Laravel application will handle real user data (e.g., registration info, orders, posts, etc.). A database like MySQL is needed to store and retrieve this data efficiently and securely.  
  
sudo mysql -u root -p

* Use administrative privileges (sudo).
* Start the MySQL client (mysql).
* Log in as the root user (-u root).
* Prompt you to enter the password for that user (-p).

CREATE DATABASE laravel\_staging;

**Create a new database** called laravel\_staging to store your Laravel application's data.

CREATE USER 'laravel\_user'@'localhost' IDENTIFIED BY 'your\_password';

**Make a new user** named laravel\_user who can only connect from your own computer (localhost).

The @'localhost' part specifies that this user can only connect to the MySQL server from the local machine (localhost).

The IDENTIFIED BY 'your\_password' part sets the password for this user. Replace 'your\_password' with a secure password.

GRANT ALL PRIVILEGES ON laravel\_staging.\* TO 'laravel\_user'@'localhost';

This command grants all privileges (permissions) on the laravel\_staging database to the user laravel\_user.

The laravel\_staging.\* part means all tables within the laravel\_staging database. This allows the user to perform any action (SELECT, INSERT, UPDATE, DELETE, etc.) on any table in that database.

FLUSH PRIVILEGES;

It ensures that MySQL recognizes the changes made to user permissions immediately without needing to restart the server.

EXIT;

1. Git clone your Laravel project

cd /var/www/

sudo git clone https://github.com/your-repo/ecommerce-project.git laravel\_staging

This will create a Laravel staging directory inside the var/www and clone the repo

1. Set the necessary permissions.

sudo chown -R www-data:www-data /var/www/laravel\_staging

* **chown**:

This command changes the owner of files or directories.

* **-R**:

Means **recursive**, so it applies the changes to all files and subdirectories inside the specified directory.

* **www-data:www-data**:

Sets both the owner and group to www-data, which is the user that the web server (like Apache or Nginx) uses to manage files.

* **/var/www/laravel\_staging**:

This is the path to the directory you want to change.

sudo chmod -R 755 /var/www/laravel\_staging

This command sets the permissions for the /var/www/laravel\_staging directory (and all its files/subdirectories) to allow

* The owner (www-data) to read, write, and execute files.
* The group (www-data) and others to read and execute files, but not write.
* 7 means the owner (in this case, www-data) can **read, write, and execute** (4+2+1).
* 5 means the group (also www-data) can **read and execute** (4+1) but cannot write.
* 5 means others can also **read and execute** (4+1) but cannot write.

1. Install Laravel dependencies inside your staging directory.

cd /var/www/laravel\_staging

composer install

1. Set Up and edit the environment file in the root directory of your project

The .env file in a Laravel application is crucial as it contains environment-specific configuration options, such as database credentials, application URL, and various settings that determine how your application runs in different environments (local, staging, production, etc.).

cp .env.example .env

nano .env

Change the following: