🐬 Django Project Setup with Docker-Compose

💡 Overview

This project is fully containerized using **Docker** and **Docker Compose**. This setup ensures that your Python environment, all dependencies (including the troublesome `mysqlclient`), and the MariaDB database run consistently across all operating systems (Windows, macOS, Linux).

This eliminates common local development headaches like missing system libraries and database version conflicts.

✅ Prerequisites

Before you begin, please ensure you have the following installed:

1. Docker Desktop (Includes Docker Engine and Docker Compose).
    - **Windows Users:** For best performance, verify that Docker Desktop is configured to use **WSL 2**.

🚀 Quick Start: First-Time Setup (4 Steps)

Run these commands **in order** from the root directory of your project (where `docker-compose.yml` is located).

Step	Command	Description
1. Build	`docker-compose build`	Creates the container image by installing necessary **Linux system packages** and all Python dependencies from `requirements.txt`.
2. Start	`docker-compose up -d`	**Launches the services** (`db` and `web`) in the background (detached mode).
3. Migrate	`docker-compose run web python manage.py migrate`	Executes the database setup script to create all necessary tables inside the MariaDB container.
4. Superuser	`docker-compose run web python manage.py createsuperuser`	Creates an administrative account for Django. Follow the prompts in your terminal.

Application Access: The Django development server is now running and accessible at http://localhost:8000

⚙️ Daily Development & Management

Use these commands to control the Docker containers during your development workflow.

Command	Purpose	When to Use
`docker-compose up`	**Starts all services.** Runs in the foreground, showing real-time logs.	Daily startup if you want to monitor output.
`docker-compose up -d`	**Starts all services** in the background.	Daily startup for clean terminal use.
`docker-compose stop`	**Stops running containers** quickly without deleting them.	To pause work briefly.
`docker-compose start`	**Restarts stopped containers** (resumes from the state they were stopped).	To quickly resume paused work.
`docker-compose down`	**Stops, removes containers, and removes the network.**	To completely shut down the environment and free resources.
`docker-compose logs -f`	**View live, streaming logs** from all running containers.	Debugging or monitoring activity.

Running Django Management Commands

To execute any standard Django command (like `test`, `shell`, or `makemigrations`), you must execute it *inside* the running `web` container using the `docker-compose run web` prefix.

Django Command	Full Docker Execution Command
`python manage.py makemigrations`	`docker-compose run web python manage.py makemigrations`
`python manage.py shell`	`docker-compose run web python manage.py shell`
`python manage.py test`	`docker-compose run web python manage.py test`