**This also works  
This works**

**git clone** [**https://github\_pat\_11AZXCTAA02Q2klUsgwwnv\_RkuFcWZZxILGhlafewfnqpkWzDfmaieAXEOZWK3bYluTHLLYHCBkd6Vy6cf@github.com/AlokDas1234/ProcureHub.git**](https://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@github.com/AlokDas1234/ProcureHub.git) **ProcureHub Django Channels Deployment Notes**

**1. Droplet Setup**

* Created a DigitalOcean droplet (Ubuntu).
* Updated the system:

sudo apt update && sudo apt upgrade -y

**2. Installed Dependencies**

* Installed Python, pip, and virtual environment support:

sudo apt install python3-pip python3-venv -y

* Installed Redis (Channels layer backend):

sudo apt install redis-server -y

sudo systemctl enable redis-server

sudo systemctl start redis-server

**3. Pulled Code from GitHub**

* Cloned your repository:

cd /opt

sudo git clone https://github.com/AlokDas1234/ProcureHub.git

cd ProcureHub

**4. Python Virtual Environment**

* Created and activated virtualenv:

python3 -m venv venv

source venv/bin/activate

* Installed requirements:

pip install --upgrade pip

pip install -r requirements.txt

**5. Django Configurations**

* Edited settings.py:

python

ALLOWED\_HOSTS = ['139.59.88.65', 'localhost', '127.0.0.1']

*(Temporary: ALLOWED\_HOSTS = ['\*'] for testing)*

* Applied migrations:

python manage.py migrate

* Collected static files (if used):

python manage.py collectstatic --noinput

**6. Configured Daphne with Supervisor**

* Installed Supervisor:

sudo apt install supervisor -y

* Created /etc/supervisor/conf.d/myapp.conf:

ini

[program:myapp]

directory=/opt/ProcureHub

command=/opt/ProcureHub/venv/bin/daphne -b 0.0.0.0 -p 8001 myproject.asgi:application

autostart=true

autorestart=true

stderr\_logfile=/var/log/myapp.err.log

stdout\_logfile=/var/log/myapp.out.log

user=root

environment=PATH="/opt/ProcureHub/venv/bin"

* Activated and started:

sudo supervisorctl reread

sudo supervisorctl update

sudo supervisorctl start myapp

sudo supervisorctl status myapp

**7. Testing**

* Accessed the app at:

http://139.59.88.65:8001/

✅ **Status:** App is running with Django Channels & Daphne via Supervisor.  
⚠️ **Next Step (Recommended):** Use **Nginx reverse proxy** so you can access the site without :8001 and handle SSL + WebSockets properly.

If you want, I can now give you the **Nginx + SSL setup** so your bidding app runs on  
https://yourdomain.com with WebSockets support.  
Would you like me to prepare that?

Ask ChatGPT

**How to deploy with new chages?**

Here’s the step-by-step (assuming your app is running via gunicorn + daphne/uvicorn + supervisor/systemd or similar):

### 1. Go to your project folder

cd /opt/ProcureHub

### 2. Check if git is initialized

If you didn’t clone from GitHub earlier, you’ll need to set the remote. Run:

git remote -v

* If you see your GitHub repo → good ✅
* If not, add it:

git init

git remote add origin https://github.com/AlokDas1234/ProcureHub.git

### 3. Fetch & pull the latest code

git fetch origin

git reset --hard origin/main

⚠️ Replace main with master if your repo branch is named that.

### 4. Install dependencies (if updated)

Activate your venv:

source venv/bin/activate

Install requirements:

pip install -r requirements.txt

### 5. Run migrations (if you changed models)

python manage.py migrate

### 6. Collect static files (if needed for CSS/JS)

python manage.py collectstatic --noinput

### 7. Restart your services

Depending on how you set it up:

**If using supervisor:**

supervisorctl restart all

**If using systemd:**

sudo systemctl restart yourproject.service

sudo systemctl restart daphne # if you run websocket with daphne

**If running manually for testing:**

python manage.py runserver 0.0.0.0:8000

After that → your live server will be on the **latest GitHub code** 🚀

👉 Do you want me to check your repo structure and write you an exact **systemd service file** for your droplet so you can just git pull && systemctl restart every time?

**1.what is the difference between git pull and git fetch origin?**

###### ChatGPT said:

Great question 👍 This is something that confuses a lot of people at first.

### 🔹 git fetch origin

* Downloads **new commits** from the remote (origin) but does **not** change your working directory or current branch.
* It just updates your **remote tracking branches** (like origin/main).
* Your local main branch will still point to the old commit until you explicitly merge or reset.

Example:

git fetch origin

Now your repo knows about new commits on origin/main, but your local main stays where it was.

### 🔹 git pull

* Is basically **two commands combined**:
* git fetch origin
* git merge origin/main
* So it will fetch new commits **and immediately try to merge** them into your current branch.

Example:

git pull

This updates your local main branch with commits from origin/main.

### 🔹 Key Difference

* git fetch = **Look but don’t touch** (see what’s new, but your code stays the same).
* git pull = **Fetch + merge** (your code is updated right away).

👉 In your case (deployment), I recommended:

git reset --hard origin/main

because:

* You don’t want merge conflicts on the server.
* You want the server code to **exactly match GitHub’s main branch** (no local edits).

✅ Quick summary:

* Use **git fetch** if you want to preview or inspect changes.
* Use **git pull** if you want to actually bring those changes into your working branch.
* Use **git reset --hard origin/main** in deployment to force your server to match GitHub 100%.

Do you want me to also explain how to safely use git fetch + git diff to review what changed before pulling (so you can check differences before updating on server)?