BuildClub Deployment Summary (buildclub.org on DigitalOcean)

This is a step-by-step reference for deploying the **BuildClub Django site** to an existing **Ubuntu DigitalOcean droplet** using **Nginx + Gunicorn (systemd socket) + Let's Encrypt**, while keeping other existing sites on the same droplet.

---

0) Assumptions / Known-good setup

- Droplet runs Ubuntu.
- Nginx is already installed and serving other sites.
- Let's Encrypt (certbot) is already used for other domains.
- BuildClub repo is a Django project.
- Deploy user: `django-user`
- Repo path: `/home/django-user/buildclub`
- Gunicorn socket: `/run/buildclub-gunicorn.sock`
- WSGI module: `buildclub_site.wsgi:application`

---

1) DNS / Domain setup (GoDaddy + DigitalOcean)

1.1 Set nameservers in GoDaddy

In GoDaddy → **buildclub.org** → **DNS** → **Nameservers**: - Choose **Custom** - Set: - `ns1.digitalocean.com` - `ns2.digitalocean.com` - `ns3.digitalocean.com`

Verify delegation:

```
dig NS buildclub.org +short
```

During propagation, querying public resolvers can show mixed results:

```
dig NS buildclub.org @8.8.8.8 +short
dig NS buildclub.org @1.1.1.1 +short
```

1.2 Add domain in DigitalOcean DNS

DigitalOcean → **Networking** → **Domains**: - Add domain: `buildclub.org` - Point to droplet IP (e.g. `143.244.183.161`)

Ensure records exist: - `A @ → <droplet_ip>` - `A www → <droplet_ip>`

Verify DNS:

```
dig +short buildclub.org @8.8.8.8
dig +short www.buildclub.org @8.8.8.8
```

---

2) Server prep

2.1 Install baseline packages

```
sudo apt update
sudo apt install -y git nginx python3-venv python3-pip
```

2.2 Create/ensure deploy user

```
sudo adduser --disabled-password --gecos "" django-user || true
sudo usermod -aG www-data django-user
```

2.3 Switch to deploy user

```
sudo -i -u django-user
```

---

3) Clone repo + virtualenv

```
cd ~
git clone https://github.com/titancoder12/BuildClub.git buildclub
cd buildclub

python3 -m venv .venv
source .venv/bin/activate
pip install --upgrade pip
pip install -r requirements.txt
```

Find the correct WSGI module:

```
find . -name wsgi.py
```

Result used: - `./buildclub_site/wsgi.py` → **Gunicorn module:** `buildclub_site.wsgi:application`

---

4) Django database + static + seed

Activate venv:

```
cd ~/buildclub
source .venv/bin/activate
```

Run migrations:

```
python manage.py migrate
```

Collect static:

```
python manage.py collectstatic --noinput
```

Seed initial course content (project already provides this command):

```
python manage.py seed_course_data
```

---

5) Gunicorn via systemd socket (isolated per-site)

5.1 Create socket unit

File: `/etc/systemd/system/buildclub-gunicorn.socket`

```
[Unit]
Description=buildclub gunicorn socket

[Socket]
ListenStream=/run/buildclub-gunicorn.sock
SocketUser=django-user
```

```
SocketGroup=www-data
SocketMode=0660

[Install]
WantedBy=sockets.target
```

5.2 Create service unit

File: `/etc/systemd/system/buildclub-gunicorn.service`

```
[Unit]
Description=buildclub gunicorn daemon
Requires=buildclub-gunicorn.socket
After=network.target

[Service]
User=django-user
Group=www-data
WorkingDirectory=/home/django-user/buildclub
Environment="PATH=/home/django-user/buildclub/.venv/bin"
ExecStart=/home/django-user/buildclub/.venv/bin/gunicorn
  --access-logfile -
  --workers 3
  --timeout 60
  --bind unix:/run/buildclub-gunicorn.sock
  buildclub_site.wsgi:application

[Install]
WantedBy=multi-user.target
```

Note: If Gunicorn fails with `status=203/EXEC`, ensure `gunicorn` is installed in the venv:

```
source /home/django-user/buildclub/.venv/bin/activate
pip install gunicorn
```

5.3 Enable/start

```
sudo systemctl daemon-reload
sudo systemctl enable --now buildclub-gunicorn.socket
sudo systemctl start buildclub-gunicorn.service
sudo systemctl status buildclub-gunicorn.service --no-pager -l
```

Verify socket ownership:

```
ls -l /run/buildclub-gunicorn.sock
```

Expected: - owner `django-user`, group `www-data`, mode `srw-rw----`

---

6) Nginx site config

6.1 Create Nginx site file

File: `/etc/nginx/sites-available/buildclub.org`

Final config (HTTP → HTTPS redirect + HTTPS server):

```
# 1) HTTP -> HTTPS redirect
server {
    listen 80;
    server_name buildclub.org www.buildclub.org;
    return 301 https://$host$request_uri;
}

# 2) HTTPS site
server {
    listen 443 ssl;
    server_name buildclub.org www.buildclub.org;

    ssl_certificate     /etc/letsencrypt/live/buildclub.org/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/buildclub.org/privkey.pem;

    client_max_body_size 25M;

    location /static/ {
        alias /home/django-user/buildclub/staticfiles/;
    }

    location / {
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        proxy_pass http://unix:/run/buildclub-gunicorn.sock;
    }
}
```

6.2 Enable site

Create symlink:

```
sudo ln -s /etc/nginx/sites-available/buildclub.org /etc/nginx/sites-enabled/
buildclub.org
```

If Nginx errors `open() ... failed (2: No such file or directory)`, it means the symlink target file doesn't exist yet (create `/etc/nginx/sites-available/buildclub.org` first).

Test & reload:

```
sudo nginx -t
sudo systemctl reload nginx
```

---

7) Django production host safety

If you see `DisallowedHost` for `buildclub.org`, update `settings.py`.

Project uses env-driven `ALLOWED_HOSTS`:

```
ALLOWED_HOSTS = [
    host.strip()
    for host in os.environ.get("ALLOWED_HOSTS", "").split(",")
    if host.strip()
]
```

Also set `CSRF_TRUSTED_ORIGINS` similarly, and set proxy SSL header:

```
CSRF_TRUSTED_ORIGINS = [
    origin.strip()
    for origin in os.environ.get("CSRF_TRUSTED_ORIGINS", "").split(",")
    if origin.strip()
]

SECURE_PROXY_SSL_HEADER = ("HTTP_X_FORWARDED_PROTO", "https")
```

Restart gunicorn after changes:

```
sudo systemctl restart buildclub-gunicorn.service
```

---

8) Let’s Encrypt certificate (safest method)

To avoid auto-edit surprises, use **certonly**:

```
sudo certbot certonly --nginx -d buildclub.org -d www.buildclub.org
```

Cert paths: - `/etc/letsencrypt/live/buildclub.org/fullchain.pem` - `/etc/letsencrypt/live/buildclub.org/privkey.pem`

Then ensure Nginx HTTPS block references those paths (see section 6).

Test renewal:

```
sudo certbot renew --dry-run
```

---

9) Validation / Troubleshooting

9.1 Check services

```
sudo systemctl status buildclub-gunicorn.service --no-pager -l
sudo systemctl status nginx --no-pager
```

9.2 Check logs

```
sudo journalctl -u buildclub-gunicorn.service -n 200 --no-pager
sudo tail -n 100 /var/log/nginx/error.log
sudo tail -n 100 /var/log/nginx/access.log
```

9.3 Confirm cert served locally by Nginx

```
echo | openssl s_client -servername buildclub.org -connect 127.0.0.1:443 2>/dev/
null | openssl x509 -noout -subject -issuer -dates
```

Expected: - `subject=CN = buildclub.org`

9.4 If browser shows cert mismatch

Usually DNS still points partially to old IPs (GoDaddy parking). Verify:

```
dig +short buildclub.org @8.8.8.8
dig +short www.buildclub.org @8.8.8.8
```

Expected: only your droplet IP.

---

10) Final state checklist

- ✅DNS (GoDaddy → DO nameservers; DO A records to droplet)
- ✅Gunicorn socket/service running (`buildclub-gunicorn.*`)
- ✅Nginx server blocks for buildclub.org enabled
- ✅Let's Encrypt cert issued for buildclub.org
- ✅HTTP redirects to HTTPS
- ✅Django accepts hostnames and CSRF origins
- ✅Site loads cleanly at `https://buildclub.org`

---

11) Deploying code changes (future updates)

This is the **standard production deploy loop** for BuildClub. Use it every time you change code.

11.1 Normal code update (most common)

```
# switch to deploy user
sudo -i -u django-user
cd ~/buildclub
source .venv/bin/activate

# pull latest code
git pull

# update dependencies (safe to always run)
pip install -r requirements.txt

# apply database changes (safe even if no new migrations)
python manage.py migrate

# update static files (safe to always run)
python manage.py collectstatic --noinput
```

```
# restart app server so new code loads
sudo systemctl restart buildclub-gunicorn.service
sudo systemctl status buildclub-gunicorn.service --no-pager -l
```

This does **not** interrupt other sites on the droplet.

11.2 If you only changed Python or templates

Gunicorn must be restarted for code to reload:

```
sudo systemctl restart buildclub-gunicorn.service
```

11.3 If you changed static files (CSS / JS / images)

```
python manage.py collectstatic --noinput
sudo systemctl restart buildclub-gunicorn.service
```

11.4 If you changed environment variables

After editing `/etc/systemd/system/buildclub-gunicorn.service`:

```
sudo systemctl daemon-reload
sudo systemctl restart buildclub-gunicorn.service
```

11.5 If you changed Nginx config

After editing `/etc/nginx/sites-available/buildclub.org`:

```
sudo nginx -t
sudo systemctl reload nginx
```

Reloading Nginx is **zero-downtime**.

11.6 Quick health checks after deploy

```
curl -I https://buildclub.org
sudo journalctl -u buildclub-gunicorn.service -n 50 --no-pager
sudo tail -n 50 /var/log/nginx/error.log
```

11.7 Optional: one-command deploy script

You can wrap the above into a `deploy.sh` script for convenience. This is optional but recommended once changes become frequent.