FLUX WIKI
STARTER KIT — SETUP (AGENT STEPS)

AGENT-EXECUTABLE SETUP — PYTHON, AWS CLI + LEAST-PRIVILEGE IAM, S3, CLOUDFRONT, .ENV, AND THE LOCAL PORTAL, EACH WITH VERIFICATION.

flux.dantesisofo.com/wiki/starter-kit-setup/

FLUX_WIKI_v2.0

JUNE 2026

FLUX DOCUMENTATION SYSTEM Layer 11 — STARTER KIT | starter-kit-setup
flux.dantesisofo.com/wiki/starter-kit-setup/

STARTER KIT — SETUP (AGENT STEPS)

Steps an AI agent runs (or instructs the operator to run), each with a verification check. All values are placeholders — read real ones from `.env` (CONFIGURATION).

STEP 1 — PYTHON + DEPS

```
python3 --version          # expect 3.12.x
python3 -m pip install -r requirements.txt
```

`requirements.txt` (pinned): `boto3 botocore Flask Werkzeug pillow reportlab qrcode piexif python-dotenv anthropic`. No Node, no NAS, no `requests` (use `urllib`). **Verify:** `python3 -c "import boto3,PIL,flask,reportlab,qrcode,piexif,dotenv,anthropic"`.

STEP 2 — AWS CLI + IAM (LEAST PRIVILEGE)

Create a dedicated IAM user (never root). Attach a policy scoped to the operator's bucket + distribution:

```
{ "Version":"2012-10-17","Statement":[
  {"Effect":"Allow","Action":["s3:ListBucket"],"Resource":"arn:aws:s3:::YOUR_BUCKET"},
  {"Effect":"Allow","Action":["s3:GetObject","s3:PutObject"],"Resource":"arn:aws:s3:::YOUR_BUCKET/*"},
  {"Effect":"Allow","Action":["cloudfront:CreateInvalidation"],"Resource":"arn:aws:cloudfront::YOUR_ACCOUNT_ID:distribu
]}
```

`aws configure` (key, secret, region). **Verify (read-only):** `aws sts get-caller-identity` → shows the IAM user ARN.

> Note: `~/.aws/credentials` lives in `$HOME`, never inside the project folder.

STEP 3 — S3 BUCKET

Create `YOUR_BUCKET`. Recommended: **private bucket + CloudFront OAC** (only CloudFront reads it). Alternative: S3 static-website (public-read). **Verify:** `aws s3 ls s3://YOUR_BUCKET` succeeds (empty is fine).

STEP 4 — CLOUDFRONT

Distribution with `YOUR_BUCKET` as origin; default root object `index.html`. Record the **Distribution ID** → `FLUX_CF_DIST`, and the domain → `FLUX_BASE_URL`. **Verify (read-only):** `python3 -c "import boto3;print(boto3.client('cloudfront').get_distribution(Id='YOUR_DIST_ID')['Distribution']['Status'])"` → `Deployed`.

STEP 5 — .env

```
cp .env.example .env   # then fill in FLUX_BUCKET, FLUX_CF_DIST, FLUX_REGION,
                       # FLUX_BASE_URL, FLUX_AUTHOR, COLLECTION, CORRIDOR_TARGET, PORTAL_USER/PASS
```

Verify: `python3 -c "import config"` prints no error and the resolved bucket. `.env` is gitignored (SECURITY).

STEP 6 — SEED CATALOG

Create `data/catalog.json`:

```
{ "schema_version":"1.1","last_updated":null,"reserved_slugs":[],"walks":[] }
```

Verify: `build_hub(catalog)` renders an empty-state hub without error.

STEP 7 — LOCAL PORTAL (OPTIONAL)

```
cd portal && python3 app.py     # serves on 127.0.0.1:5050
```

Verify: `curl -s -o /dev/null -w '%{http_code}' http://127.0.0.1:5050/studio/` → `401` unauth; `200` with `-u $PORTAL_USER:$PORTAL_PASS`.

> **Operational rules (encode as warnings):** - After editing publisher/worker code, **restart the portal** (a long-running process holds stale code in memory and will publish stale output). - Don't run the portal on two machines at once (concurrent publishes race on `catalog.json`).

STEP 8 — FIRST-RUN GATE (READ-ONLY BEFORE ANY WRITE)

```
aws sts get-caller-identity
aws s3 ls s3://YOUR_BUCKET
```

Then publish the **sample walk to a throwaway bucket** to prove the happy path (PROJECT WORKFLOW) before pointing at the real bucket.

SETUP ACCEPTANCE CHECKLIST

- [ ] deps import cleanly
- [ ] `aws sts get-caller-identity` shows the dedicated IAM user
- [ ] `aws s3 ls s3://YOUR_BUCKET` works
- [ ] CloudFront distribution status `Deployed`
- [ ] `import config` resolves all required vars
- [ ] empty hub renders from the seed catalog
- [ ] portal returns 401/200

Next: DEPLOYMENT.

FLUX_WIKI_v2.0 — flux.dantesisofo.com/wiki/starter-kit-setup/