**1. Create a feature name**

Use **snake\_case** for filesystem names, e.g.

nginx

CopyEdit

new\_feature

and a **CamelCase** for any Python class names (e.g. NewFeatureRequest, NewFeatureResponse).

**2. Create the feature folder**

bash

CopyEdit

mkdir -p app/features/new\_feature

Inside that folder, create these four files:

bash

CopyEdit

app/features/new\_feature/

├── \_\_init\_\_.py

├── schema.py

├── service.py

└── endpoint.py

You can do this in one shot on Unix:

bash

CopyEdit

for f in \_\_init\_\_.py schema.py service.py endpoint.py; do \

touch app/features/new\_feature/$f; \

done

**3. Define your Pydantic models (schema.py)**

In app/features/new\_feature/schema.py, sketch out your request/response contracts, e.g.:

python

CopyEdit

from pydantic import BaseModel

class NewFeatureRequest(BaseModel):

param1: str

param2: int

class NewFeatureResponse(BaseModel):

result: float

detail: str

**4. Implement business logic (service.py)**

In app/features/new\_feature/service.py, write a function that does the heavy lifting:

python

CopyEdit

def run\_new\_feature(data: NewFeatureRequest) -> NewFeatureResponse:

# TODO: implement your algorithm here

output = data.param2 \* 42 # placeholder

return NewFeatureResponse(result=output, detail="ok")

Import your schemas here so mypy/IDE picks up types.

**5. Wire up your HTTP endpoints (endpoint.py)**

In app/features/new\_feature/endpoint.py:

python

CopyEdit

from fastapi import APIRouter, HTTPException

from app.features.new\_feature.schema import NewFeatureRequest, NewFeatureResponse

from app.features.new\_feature.service import run\_new\_feature

router = APIRouter(

prefix="/new\_feature",

tags=["New Feature"]

)

@router.post("/run", response\_model=NewFeatureResponse)

async def run\_feature(payload: NewFeatureRequest):

try:

return run\_new\_feature(payload)

except Exception as e:

raise HTTPException(status\_code=500, detail=str(e))

**6. Register the router**

Edit your central router in app/api/router.py:

python

CopyEdit

from fastapi import APIRouter

# … existing imports …

from app.features.new\_feature.endpoint import router as new\_feature\_router

router = APIRouter(prefix="/api/v1")

# … your other feature routers …

router.include\_router(new\_feature\_router)

No other changes needed here—FastAPI will auto-discover.

**7. Add a smoke-test (tests/test\_new\_feature.py)**

In tests/, create test\_new\_feature.py:

python

CopyEdit

from fastapi.testclient import TestClient

from app.main import app

client = TestClient(app)

def test\_run\_new\_feature\_success():

resp = client.post("/api/v1/new\_feature/run", json={"param1":"foo","param2":2})

assert resp.status\_code == 200

data = resp.json()

assert "result" in data and data["detail"] == "ok"

**8. Install any new dependencies**

If your feature needs extra libraries, add them to requirements.txt and rebuild your Docker image:

bash

CopyEdit

echo "some-library" >> requirements.txt

docker-compose build fastapi

**9. Run your tests**

bash

CopyEdit

pytest -q

**10. Commit & push**

bash

CopyEdit

git add app/features/new\_feature tests/test\_new\_feature.py api/router.py requirements.txt

git commit -m "feat(new\_feature): add New Feature atom with endpoint, schema, service, and tests"

git push