NovaOS Reset Pack

This document contains everything needed to fully reset and redeploy NovaOS with all core and C-Suite agents, plus supporting R&D teams, using the existing folder structure and tools (Redis, mCP, Docker, n8n, LangGraph, GitHub, Render, Baserow). Keep this file handy to restore or share the complete environment.

---

1. Prerequisites

- **Host directory**: `~/Desktop/NovaOS`
- **Installed**: Docker, Docker Compose, Python 3, Redis CLI (on host), n8n CLI, Git, Render CLI (if using render deploy)
- **Environment vars**:
- `OPENAI_API_KEY` (for Codex/Copilot/Aider)
- `REDIS_HOST=redis`
- `BASEROW_API_KEY` (for Baserow integration)

---

2. Folder structure

```
NovaOS/
├─ agents/
│  ├─ NOVA-CORE/           ← orchestrator agent
│  ├─ CEO-VISION/          ← CEO strategist
│  ├─ CTO-AUTO/            ← CTO systems engineer
│  ├─ CMO-AUTO/            ← CMO marketing
│  ├─ CFO-AUTO/            ← CFO finance
│  ├─ CPO-AUTO/            ← CPO offers
│  ├─ CLO-AUTO/            ← CLO legal
│  ├─ CCO-AUTO/            ← CCO customer experience
│  ├─ RESEARCH-ANALYST/    ← market research
│  ├─ CLARITY-COACH/       ← clarity coach
│  ├─ CHIEF-STAFF/         ← operations lead
│  ├─ ENERGY-GUARDIAN/     ← energy management
│  ├─ PROMPT-ENGINEER/     ← prompt design
│  └─ (other tool-specialist agents)
├─ mcp/
│  └─ config.yaml          ← Redis bus config
├─ Dockerfile
├─ docker-compose.yml
├─ run.sh                  ← startup & R&D launcher
├─ langgraph.yml          ← LangGraph workflow definitions
└─ .github/                ← CI/CD definitions
```

3. mCP config (`mcp/config.yaml`)

```
backend:
  type: redis
  host: ${REDIS_HOST}
  port: 6379
channels:
  tasks:
    key: novaos:tasks
```

4. `docker-compose.yml` (full)

```
version: "3.8"
services:
  redis:
    image: redis:alpine
    container_name: novaos_redis

  nova-core:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: nova-core
    volumes:
      - ./:/app
      - ./mcp/:/app/mcp/
    working_dir: /app
    depends_on:
      redis: { condition: service_started }
    environment:
      - REDIS_HOST=redis
      - PYTHONUNBUFFERED=1
    command: ["python", "-u", "agents/NOVA-CORE/main.py"]

  CEO-VISION:
    <<: *nova-core-base
    container_name: novaos_ceo_vision
    command: ["python", "-u", "agents/CEO-VISION/main.py"]

  CTO-AUTO:
    <<: *nova-core-base
    container_name: novaos_cto_auto
    command: ["python", "-u", "agents/CTO-AUTO/main.py"]

  CMO-AUTO:
    <<: *nova-core-base
    container_name: novaos_cmo_auto
    command: ["python", "-u", "agents/CMO-AUTO/main.py"]

  CFO-AUTO:
    <<: *nova-core-base
    container_name: novaos_cfo_auto
    command: ["python", "-u", "agents/CFO-AUTO/main.py"]

  CPO-AUTO:
    <<: *nova-core-base
    container_name: novaos_cpo_auto
    command: ["python", "-u", "agents/CPO-AUTO/main.py"]

  CLO-AUTO:
    <<: *nova-core-base
    container_name: novaos_clo_auto
    command: ["python", "-u", "agents/CLO-AUTO/main.py"]

  CCO-AUTO:
    <<: *nova-core-base
    container_name: novaos_cco_auto
    command: ["python", "-u", "agents/CCO-AUTO/main.py"]

  RESEARCH-ANALYST:
    <<: *nova-core-base
    container_name: novaos_research_analyst
    command: ["python", "-u", "agents/RESEARCH-ANALYST/main.py"]

  CLARITY-COACH:
    <<: *nova-core-base
    container_name: novaos_clarity_coach
    command: ["python", "-u", "agents/CLARITY-COACH/main.py"]

  CHIEF-STAFF:
    <<: *nova-core-base
    container_name: novaos_chief_staff
    command: ["python", "-u", "agents/CHIEF-STAFF/main.py"]

  ENERGY-GUARDIAN:
    <<: *nova-core-base
    container_name: novaos_energy_guardian
    command: ["python", "-u", "agents/ENERGY-GUARDIAN/main.py"]

  PROMPT-ENGINEER:
    <<: *nova-core-base
    container_name: novaos_prompt_engineer
    command: ["python", "-u", "agents/PROMPT-ENGINEER/main.py"]

  # Tool-specialists (examples):
  n8n-flow-builder:
    <<: *nova-core-base
    container_name: novaos_n8n_flow_builder
    command: ["python", "-u", "agents/N8N-FLOW-BUILDER/main.py"]

  langgraph-router:
    <<: *nova-core-base
    container_name: novaos_langgraph_router
    command: ["python", "-u", "agents/LANGGRAPH-ROUTER/main.py"]

  github-deployer:
    <<: *nova-core-base
    container_name: novaos_github_deployer
    command: ["python", "-u", "agents/GITHUB-DEPLOYER/main.py"]

  render-manager:
    <<: *nova-core-base
    container_name: novaos_render_manager
    command: ["python", "-u", "agents/RENDER-MANAGER/main.py"]
```

---

5. Agent Scripts

Each `agents/<FOLDER>/main.py` uses this template:

```
import os, time, redis

r = redis.Redis(host=os.getenv('REDIS_HOST','redis'), port=6379,
decode_responses=True)

def process_task(task: str) -> str:
    # Custom logic per agent
    return None

while True:
    item = r.brpop('novaos:tasks', timeout=5)
    if item:
        _, t = item
        print(f"Received task: {t}")
        resp = process_task(t)
        if resp:
            r.lpush('novaos:tasks', resp)
            print(f"Published response: {resp}")
    time.sleep(1)
```

Adjust `process_task` for each role.

---

6. LangGraph Workflow (`langgraph.yml`)

Provide your pipeline definitions here for end-to-end agent chaining.

---

7. R&D Launcher (`run.sh`)

```
#!/usr/bin/env bash
docker-compose up --build -d

function deploy_team() {
  idea="$1"
  echo "Deploying R&D team for idea: $idea"
  docker exec nova-core \
    sh -c "python -u agents/NOVA-CORE/main.py <<< 'New Idea R&D: $idea'"
}

echo "NovaOS up. Use: deploy_team 'Idea'"
```

Reset Workflow:

1. Overwrite `docker-compose.yml` with Section 4 above.
2. Create/update each `main.py` in `agents/` per Section 5.
3. (Optional) Add your `langgraph.yml` definitions.
4. Run:

   ```
   chmod +x run.sh
   ./run.sh
   ```

5. Test:

```
docker-compose exec redis redis-cli LPUSH novaos:tasks "New POD capsule: Sunset Vibes"
docker-compose logs -f novaos_cto_auto
```

NovaOS is now reset using Baserow, n8n, Docker, LangGraph, GitHub, and Render. Ready for mission-critical operations!