What you want is:

> CLAWBOLT = immutable control system
> Workspace projects = editable, disposable, safe

Below is the correct, professional approach to do this, step by step.

🔒 Goal (Restated Clearly)

- CLAWBOLT is finished
- Its code must not be modified accidentally
- User can freely:
  - Code
  - Edit
  - Break
  - Experiment

    inside project workspaces
- CLAWBOLT keeps running unchanged

This is how real production systems are protected.

✅ Core Strategy (One Sentence)

> Separate CLAWBOLT (immutable core) from Workspaces (mutable user projects), enforced by filesystem permissions + process boundaries.

1️⃣ Directory Separation (Non-Negotiable)

Final Layout

/opt/clawbolt/ ← IMMUTABLE (system)
/opt/clawbolt-core/ ← (optional split)
/home/user/workspace/ ← MUTABLE (projects)

Example

/opt/clawbolt/
├── core/
├── telegram/
├── agent/
├── security/
└── systemd/

/home/jayson/workspace/
├── projectA/
├── projectB/
└── experiments/

🚫 Never mix workspace inside CLAWBOLT dir

2️⃣ Make CLAWBOLT Read-Only (Critical Step)

Once “final”, lock it.

```
sudo chown -R root:root /opt/clawbolt
sudo chmod -R 755 /opt/clawbolt
sudo chmod -R a-w /opt/clawbolt
```

Result:

- CLAWBOLT runs
- No one can edit it (even you, without sudo)
- Accidental edits = impossible

3️⃣ Run CLAWBOLT as a Dedicated System User

Create a service user:

sudo useradd -r -s /usr/sbin/nologin clawbolt

Systemd service:

User=clawbolt
Group=clawbolt

Why:

- CLAWBOLT cannot touch your workspace unless explicitly allowed
- Compartmentalization

4️⃣ Controlled Access to Workspace (Explicit Only)

Give CLAWBOLT read-only access to workspace:

sudo setfacl -R -m u:clawbolt:rx /home/jayson/workspace

If needed:

- Write access only to temp folders
- Never to CLAWBOLT core

5️⃣ Policy-Based File Access (Inside CLAWBOLT)

Add a hard rule in code:

ALLOWED_PATHS = ["/home/jayson/workspace"]

```
def is_allowed_path(path):
    return any(path.startswith(p) for p in ALLOWED_PATHS)
```

Any request outside:

- Auto-deny
- Requires Telegram approval

6️⃣ Immutable Mode (Advanced, Optional)

For maximum protection:

```
sudo chattr -R +i /opt/clawbolt
```

⚠️ This makes files literally unchangeable, even by root, until unlocked.

Unlock only when upgrading:

```
sudo chattr -R -i /opt/clawbolt
```

7️⃣ Separate Virtual Environments

CLAWBOLT should never share dependencies with workspace projects.

```
/opt/clawbolt/venv/
/home/jayson/workspace/projectA/venv/
```

This avoids:

- pip breaking CLAWBOLT
- version conflicts

8️⃣ Read-Only Runtime Checks (Self-Defense)

CLAWBOLT can self-check:

```
import os

if os.access("/opt/clawbolt", os.W_OK):
    alert("⚠️ CLAWBOLT core is writable — unsafe state")
```

Send alert to Telegram.

9️⃣ Workspace Safety Philosophy

Workspace rules:

- Anything can break → OK
- Delete → OK
- Rewrite → OK
- Test → OK

CLAWBOLT rules:

- No edits
- No pip install
- No hotfix
- No live modification

Upgrades happen only by:

- Explicit maintenance mode

- Manual intervention

🔐 Mental Model (Very Important)

Think of CLAWBOLT like:

- systemd
- docker
- ssh
- network-manager

You use it.

You don’t edit it daily.

🧠 Final Recommendation (Best Practice)

Do ALL of these:

✔️ Separate directories

✔️ Read-only permissions

✔️ Dedicated system user

✔️ Workspace ACL

✔️ Self-check alerts

Optional but powerful:

✔️ Immutable flag

✔️ Maintenance mode toggle

🧩 One-Line Answer to Your Question

Freeze CLAWBOLT as an immutable system service, isolate it from editable workspaces, and enforce boundaries using filesystem permissions and a dedicated service user.