
⚙️ PHASE 3 — AGENT COMMAND ROUTER

“AI optional. Control guaranteed.”

This phase turns CLAWBOLT into a real remote operator, not just a messenger.

🎯 PHASE 3 OBJECTIVES (LOCKED)

✅ Slash command parsing

✅ Command isolation (one file = one power)

✅ /screen implemented

✅ /rules implemented

✅ Clean fallback to AI when not a command

✅ Safe expansion path for future commands

🧠 CORE DESIGN DECISION

Rule:

No ambiguity. No AI guessing commands.

📁 UPDATED DIRECTORY (PHASE 3)

```
CLAWBOLT/
│
├── main.py
│
├── telegram/
│   ├── bot.py
│   ├── send_telegram.py
│   └── auth.py
│
├── antigravity/
│   ├── window.py
│   ├── mapper.py
│   └── injector.py
│
├── agent/
│   ├── router.py
│   ├── rules.py
│   └── screen.py
│
├── screen/
│   └── capture.py
│
├── storage/
│   └── screenshots/
│
├── config/
│   └── secrets.env
│
└── logs/
```

1️⃣ COMMAND ROUTER (THE BRAIN)

agent/router.py

```
from agent.rules import show_rules
from agent.screen import do_screen
```

```
COMMANDS = {
    "/rules": show_rules,
    "/screen": do_screen,
}

async def route_command(update, command_text):
    cmd = command_text.split()[0]

    handler = COMMANDS.get(cmd)
    if not handler:

        await update.message.reply_text("❌ Unknown command. Use
/rules")
        return

    await handler(update)
```

This file never touches Antigravity.

This is pure authority.

2️⃣

/rules

COMMAND

agent/rules.py

```
async def show_rules(update):
    rules_text = """
🧠 CLAWBOLT AGENT COMMANDS

/rules   - Show all commands
/screen  - Take screenshot

More coming soon…
"""
```

```
    await update.message.reply_text(rules_text)
```

3️⃣

/screen

COMMAND (INSTANT CONTROL)

agent/screen.py

```
from screen.capture import capture_screen
from telegram.send_telegram import send_image

async def do_screen(update):

    await update.message.reply_text("📸 Capturing screen...")
    path = capture_screen()
    await send_image(update, path)
```

This works:

- With AI
- Without AI
- During crashes
- During UI chaos

That’s the point.