You Are the Senior Developer

AI as a Force Multiplier, Not a Pilot

Goal: Learn to code 2x faster by becoming the Architect.

Tools: Gemini CLI, Google Antigravity, WPILib VS Code.

The Reality Check: What is an LLM?

- **It IS:** A prediction engine.
    - It predicts the next likely word based on internet data.
- **It is NOT:** A physics engine.
    - It doesn't know our robot has a bent intake.
- **It is NOT:** A logic engine.
    - It doesn't "think"; it mimics syntax patterns.

> ***The Trap:*** *"Auto-pilot coding." If you copy-paste code you don't understand, you create "Technical Debt" that you cannot fix in the queuing line.*

The "Why" (Part 1): The Offline Field

Why can't we just use Cursor/Antigravity for everything?

1. **The Field is Offline:**
   - Possibly no Wi-Fi in the pits. No AI. No StackOverflow.
2. **The Toolchain is Strict:**
   - **WPILib VS Code** contains the specific JDK, GradleRIO, and Vendor Libraries (CTRE, REV) needed to deploy to the RoboRIO.
   - Generic editors will fail the build or the deploy process.

> ***The Rule:*** *Code lives in WPILib VS Code. AI lives in the background.*

The "Why" (Part 2): The Knowledge Gap

You must be the "Senior Dev" who audits the "Junior Dev" (AI).

- **Hallucinations:**
  - AI loves to use old APIs (e.g., `Phoenix 5` instead of `Phoenix 6`).
  - It invents methods like `.setSuperSpeed()` that do not exist.
- **Guiding the Tool:**
  - *Bad Prompt:* "Write code for a shooter." (AI guesses the hardware).
  - *Good Prompt:* "Create a subsystem using `CANSparkMax` on IDs 4 and 5. Use `RevLib`."

> ***Bottom Line:*** *If you don't know the library "Under the Hood," you can't guide the AI.*

Tool 1: Gemini CLI ("The Trench Tool")

Where: Inside your WPILib VS Code Terminal.

When: You have a red build error or forgot syntax.

The Workflow:

Instead of Alt-Tabbing to Chrome, pipe the error directly to AI.

```
> ./gradlew build
# BUILD FAILED: NullPointerException in DriveSubsystem.java:42

> gradlerio build | gemini "Explain this error and fix the syntax"
```

Why: It fixes the immediate "hurt" without breaking your flow.

Tool 2: Antigravity ("Code Archaeology")

Where: A separate window (The "Blueprint Room").

When: You are trying to learn *how* a complex codebase works.

The Workflow:

1. Clone a repo (e.g., Team 254 or our old code) into Antigravity.
2. **Prompt:** *"Scan this folder. Explain how the `AutoSelector` class passes the chosen routine to `RobotContainer`."*
3. **Prompt:** *"I see `new InstantCommand(...)` used here. Explain what that class does in WPILib and why they chose it over `RunCommand`."*

> ***Goal:*** *Use AI to **trace logic** threads that would take hours to read manually.*

Tool 2: Antigravity ("Boilerplate Builder")

When: Scaffolding a new subsystem (The boring typing part).

The "PeaShooter" Prompt:

> *"I need a new subsystem called `PeaShooter`. It must use the Command-Based framework. It has **two flywheels** (SparkMax IDs 5, 6) and **one hood** (Servo PWM 1). Write the boilerplate code including hardware init and empty methods for `setSpeed` and `setHood`."*

The "Senior Dev" Audit (Your Job):

- [ ] Did it import `com.revrobotics` correctly?
- [ ] Did it use the correct Motor Type (Brushed vs Brushless)?
- [ ] **ACTION:** Copy the clean boilerplate to WPILib VS Code. YOU write the logic inside the empty methods.

The Hybrid Workflow

1. **Plan (Antigravity):** "We need a Climber." Generate the Plan/Skeleton.
2. **Verify (The Human):** Read the code. Does it match our wiring?
3. **Transfer (WPILib VS Code):** Copy the file to the real project.
4. **Debug (Gemini CLI):** Fix syntax errors in the terminal.
5. **Deploy:** Test on the Robot.

The Final Challenge: The Vibe Check

The 5-Minute Rule:

If you cannot explain *why* a block of code works in 5 minutes, **delete it.**

Homework Assignment:

1. Pick one file from last year's robot code.
2. Put it in Antigravity.
3. **Prompt:** *"Explain this code to me like I'm a 9th grader."*
4. **Prompt:** *"Are there any potential bugs or 'code smells' in this file?"*

Questions?

"The machine does the typing. You do the thinking."