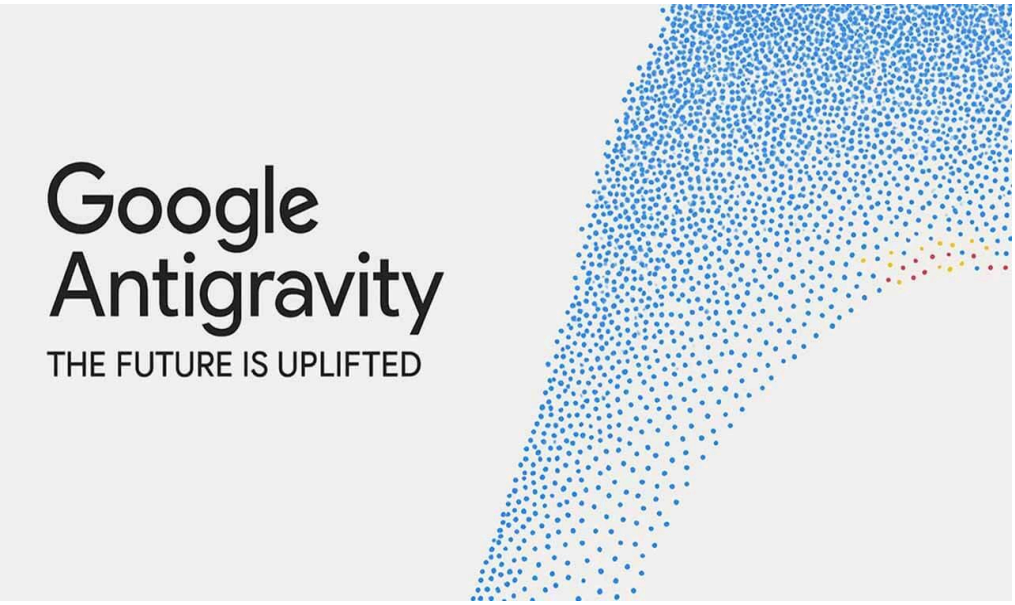
Kasadara

Google Antigravity – Agentic Development Platform & Setup Guide

Installation, Configuration & Usage



Kasadara

Table of Content

Kasadara

1. Executive Summary

- **Google Antigravity**: Agent-based development platform that enables AI agents to plan, execute, and validate development tasks within a familiar IDE.
- Mission Control approach to manage multiple agents in parallel while keeping developers in control.
- Reduces repetitive effort and helps developers focus on decisions, design, and code quality.

2. Key Purposes

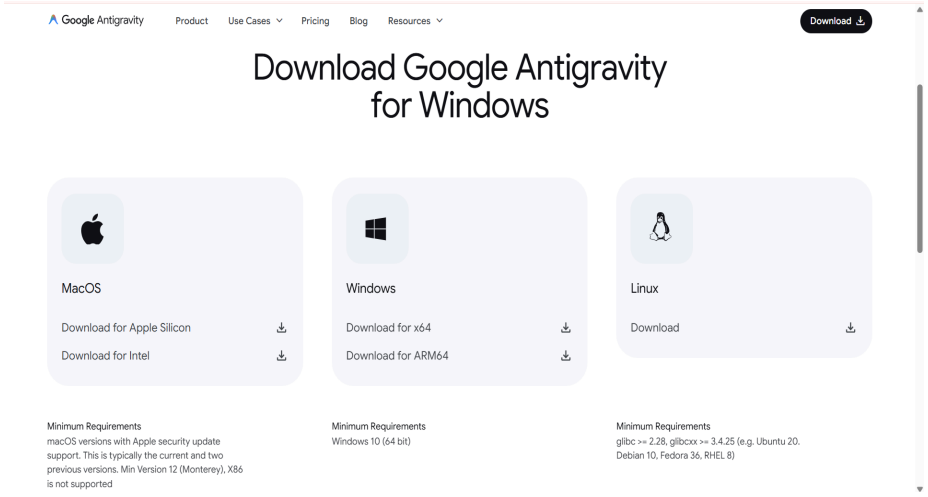
- **Autonomous Agents:** Empowers AI agents to handle full feature requests, plan, code, and verify, reducing developer micromanagement.
- **Mission Control:** A central interface to oversee and interact with these agents, creating workspaces and managing workflows.
- **Parallel Execution:** Enables developers to start new tasks without waiting for others, improving efficiency.
- **Contextual Awareness:** Agents remember context, allowing for seamless handoffs between different tasks
- **Browser Control:** Agents use browsers to gather information and perform visual feedback-driven frontend work
- **Knowledge Base:** Allows developers to build custom knowledge bases from code references.
- **Terminal Support:** Offers control over terminal execution for agents, with configurable permission levels.

3. Installation & Configuration

3.1 System Requirements & Download

Go to the downloads page and click on the appropriate Operating System version that is applicable to your case:

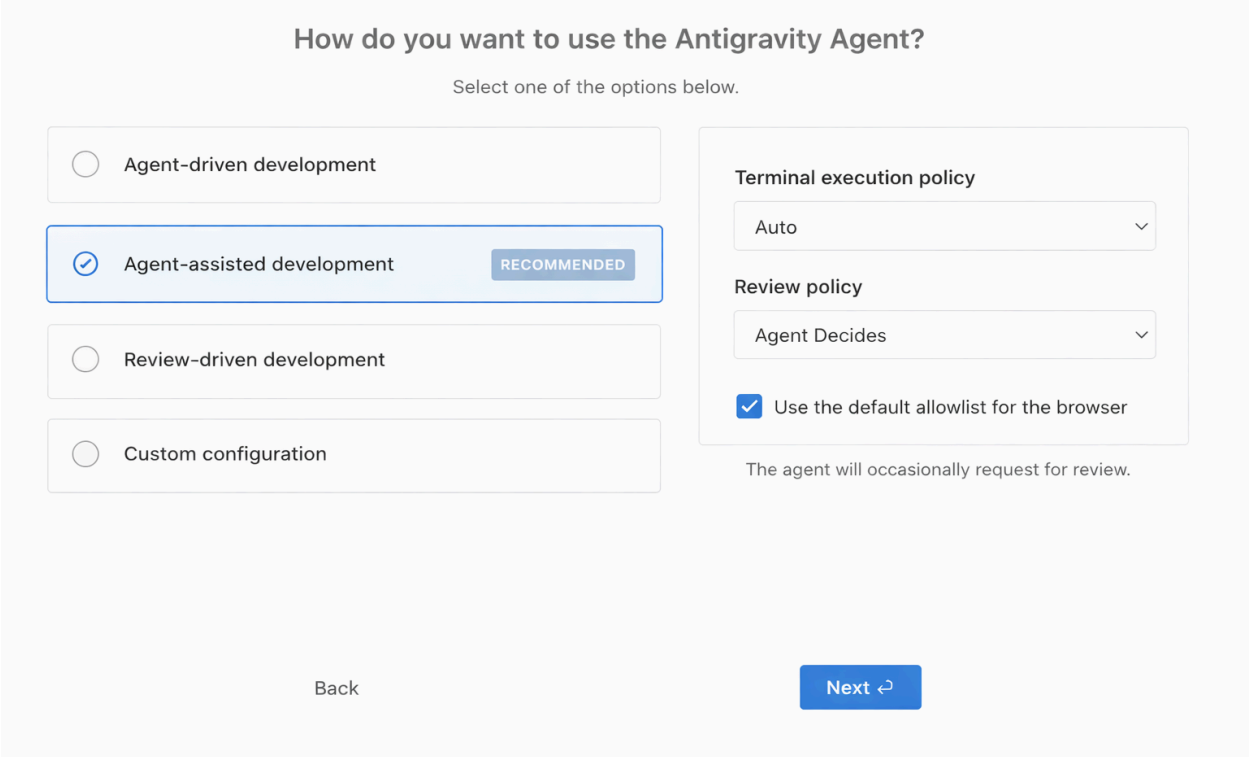
- **Source:** Download the official installer from https://antigravity.google/download
- **OS Support**: Available for macOS (Apple Silicon/Intel), Windows, and Linux.



With the installation completed, Google Antigravity is now ready to be launched on the local system.

3.2 Setup & Configuration

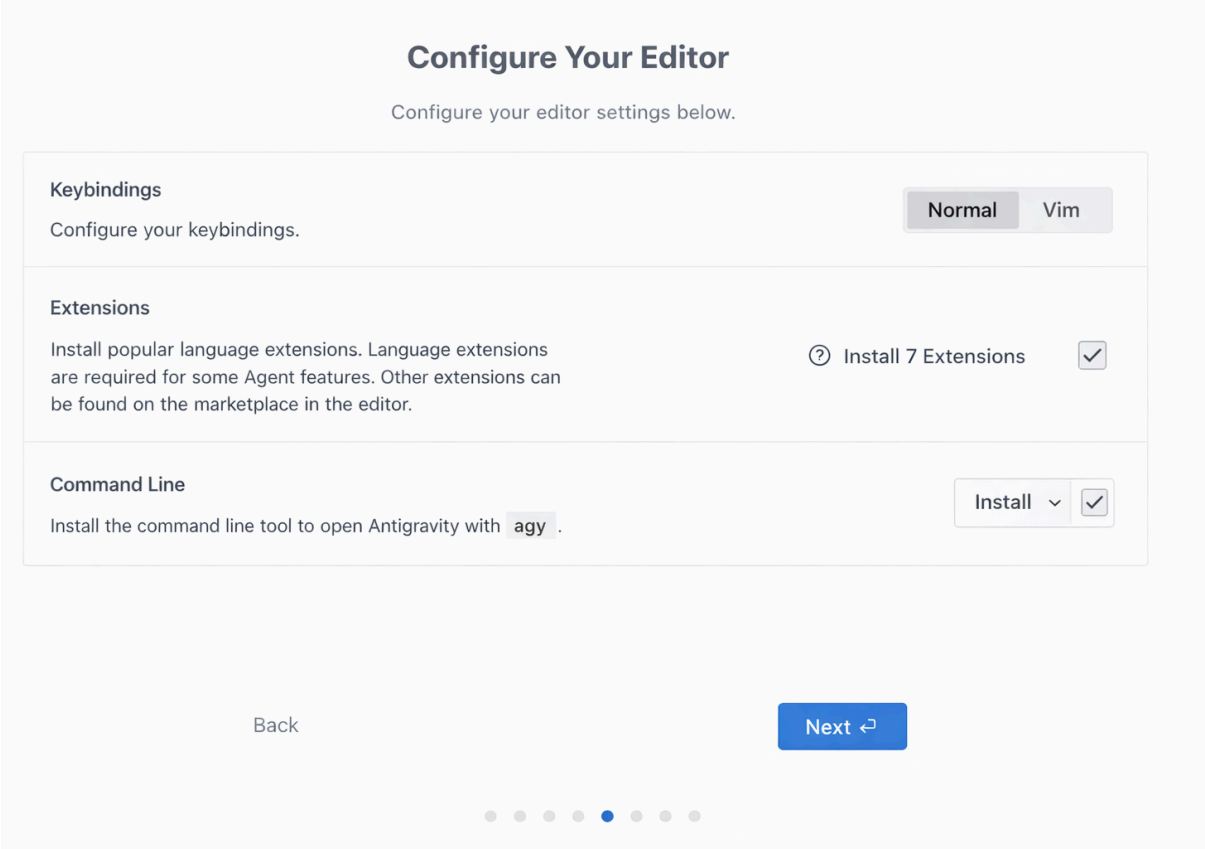
After installation, this screen allows users to configure how the agent is expected to behave.



- **Agent-driven development:** The AI agent takes the lead in the development process
- **Agent-assisted development:** The agent works alongside the developer, providing assistance and occasionally requesting a review.

- **Review-driven development:** The agent performs tasks but relies heavily on the user to review and approve actions.
- **Custom configuration:** Allows the user to manually define specific rules and workflows for the agent's behavior

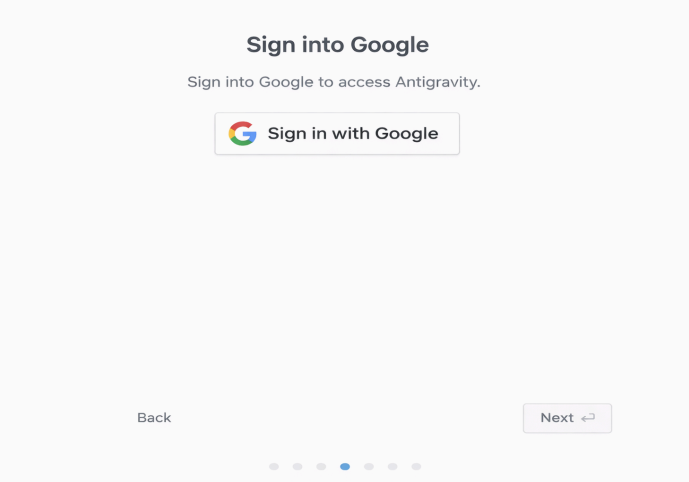
The next step is to configure **the Editor**. Choose your preferences.



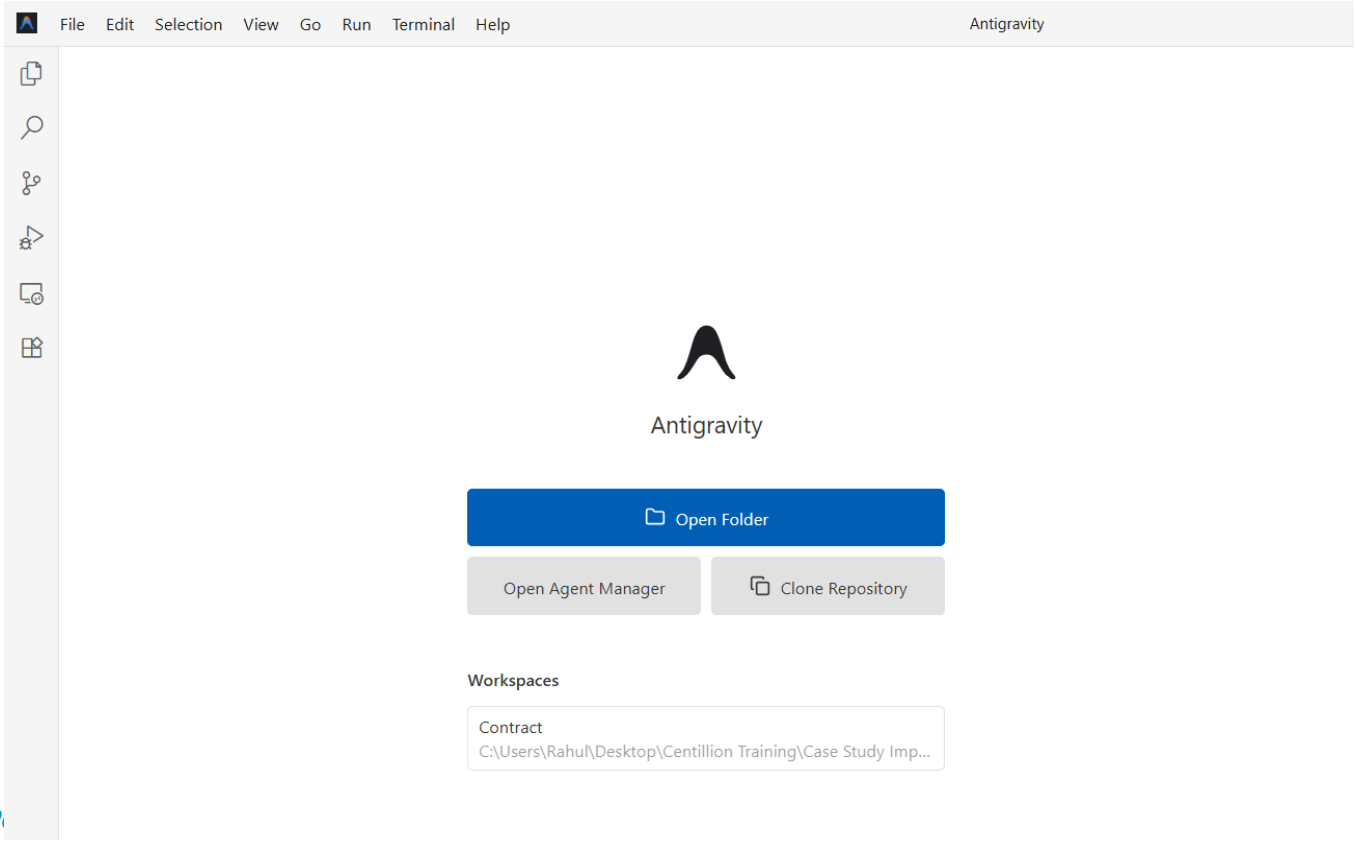
- **Keybindings**: Choose between standard or Vim-style keybindings.

- **Extensions**: Installs essential language extensions required for agent features.
- **Command-Line Tool**: Enables the **agy** CLI for launching Antigravity from the terminal.

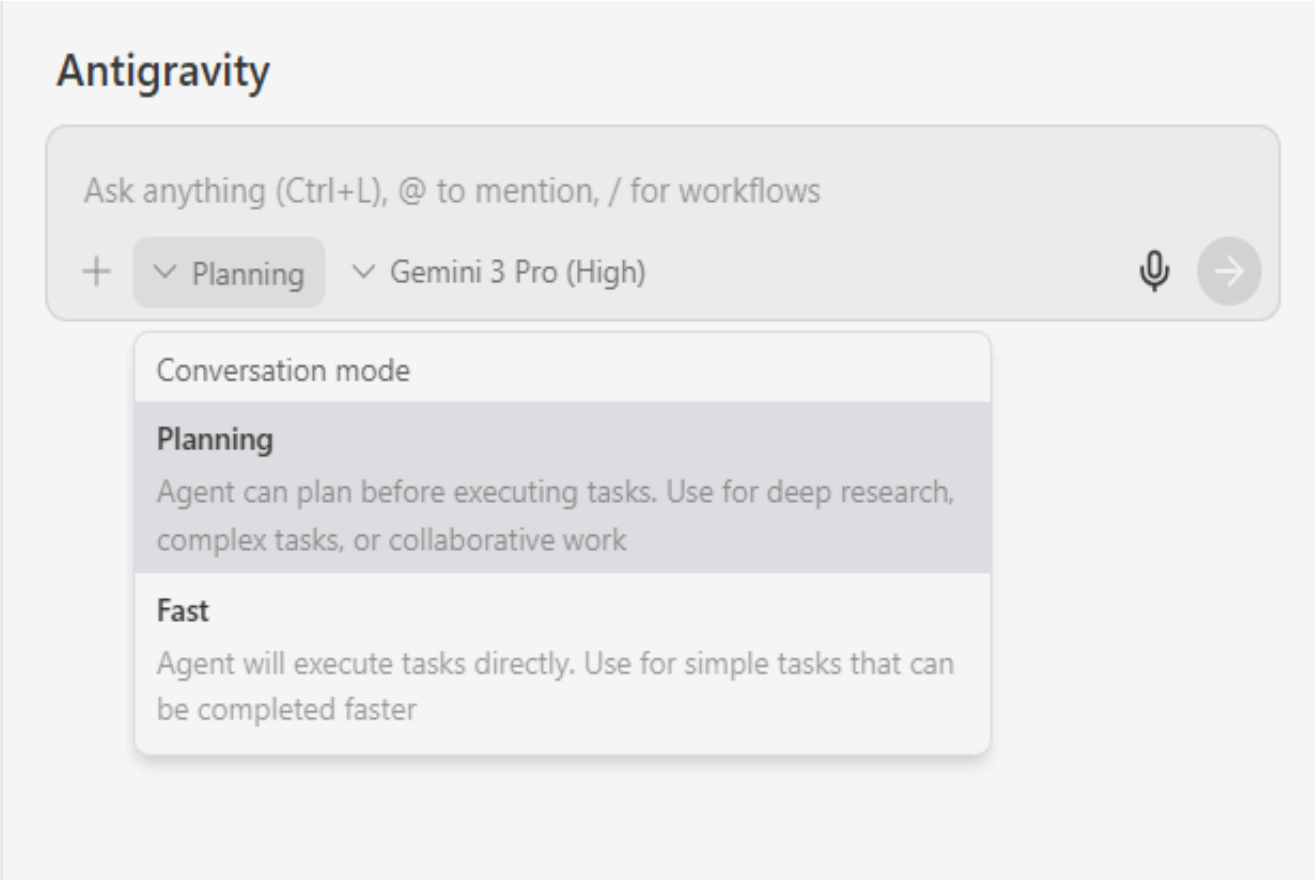
The next step requires signing in with a Google account.



After signing in with Google, Antigravity asks you to open a folder or to clone.



After that, the agent window opens automatically. It can also be opened manually using **Ctrl + Alt + B** or by clicking the toggle icon in the top-right corner.



- **Planning Mode**

  The agent plans before execution.

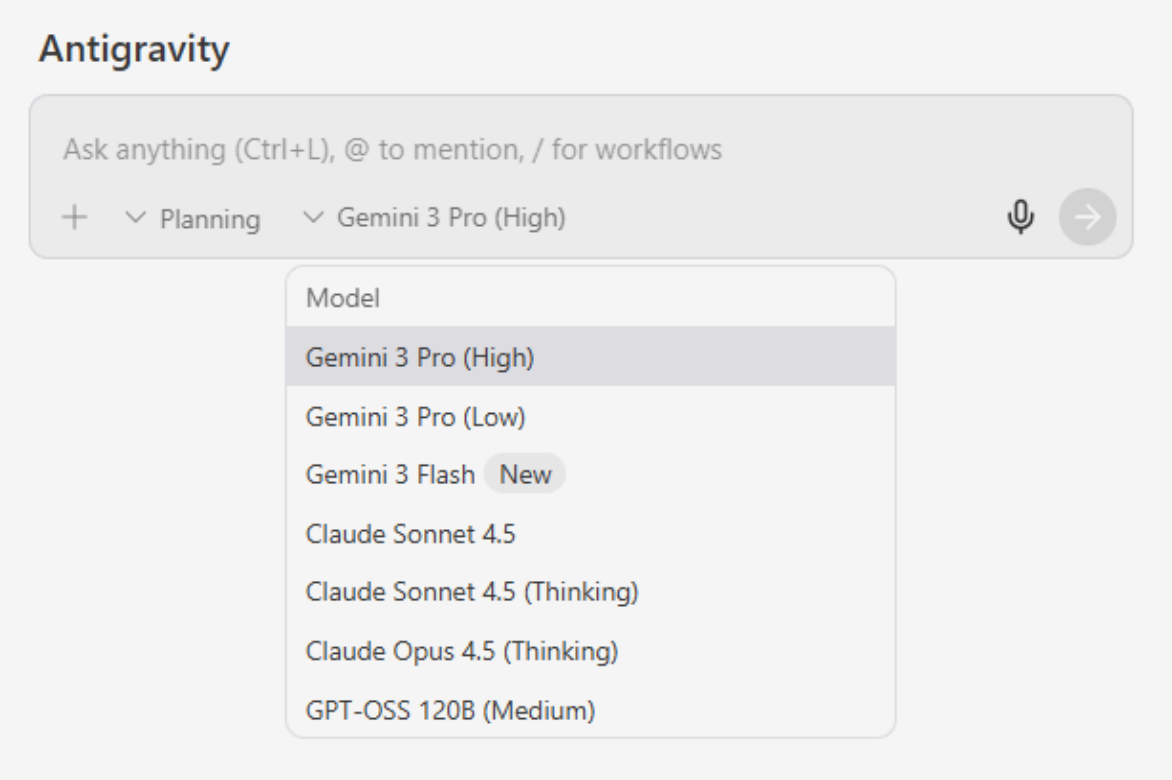
  Best for: Complex or research-heavy tasks.

- **Fast Mode**

  The agent executes immediately.

  Best for: Quick, simple tasks.

Antigravity allows users to choose different AI models based on task complexity,



- **Gemini 3 Pro (High / Low):** Advanced reasoning with selectable performance levels
- **Gemini 3 Flash :** Lightweight model optimized for speed and efficiency
- **Claude Sonnet 4.5 (Thinking):** Balanced reasoning with structured responses
- **Claude Opus 4.5 (Thinking):** High-end model for deep and complex reasoning
- **GPT-OSS 120B (Medium):** Large open-source model for flexible use

4. Autonomous Agent

The agent window provides two distinct conversation modes.

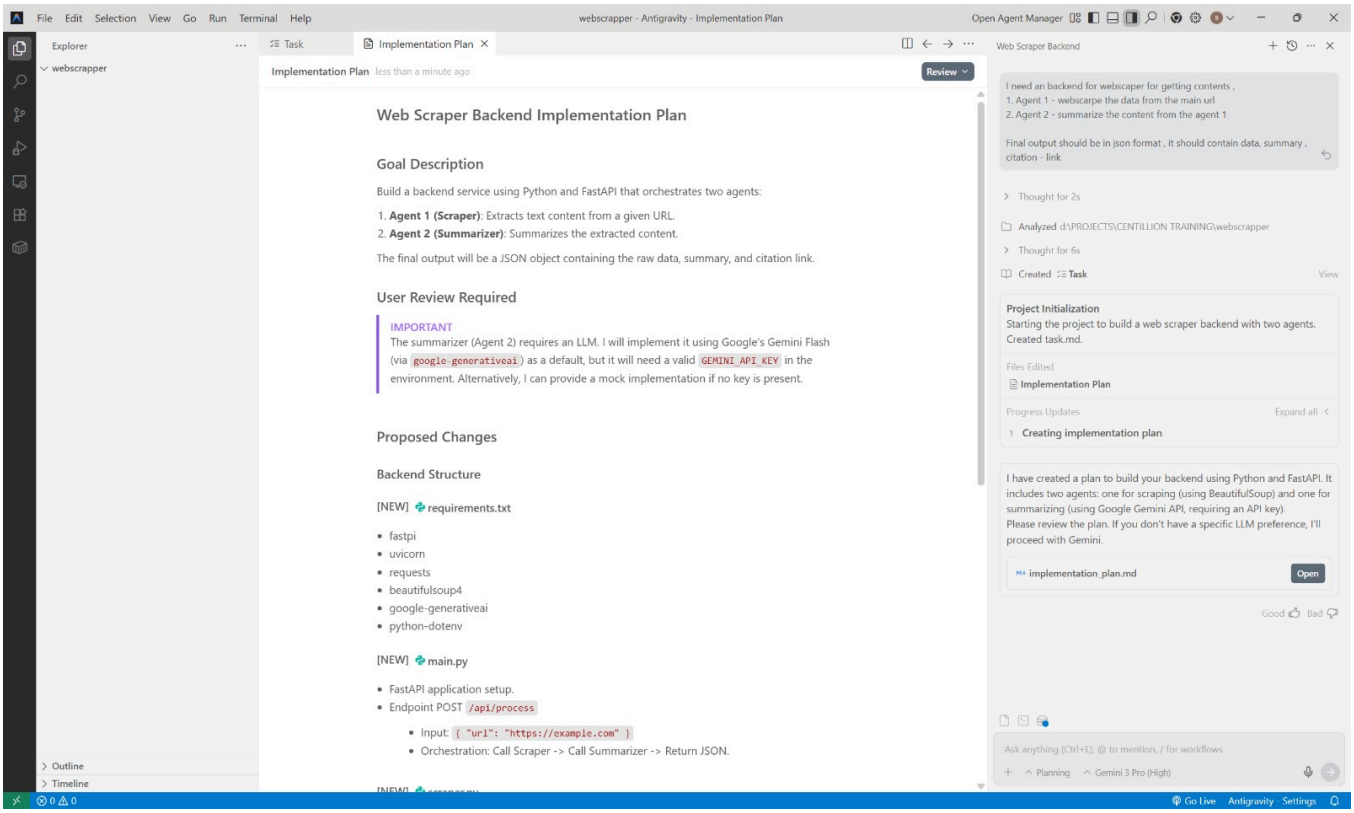
- Planning Mode
- Fast Mode

The agent presents assigned tasks along with a structured implementation plan.

Tasks

- [ ] Initialize Project Structure
  - [ ] Create virtual environment setup (instructions)
  - [ ] Create `requirements.txt`
- [ ] Implement Agent 1 (Scraper)
  - [ ] Create scraper module
  - [ ] Implement `extract_data` function
- [ ] Implement Agent 2 (Summarizer)
  - [ ] Create summarizer module
  - [ ] Integrate with LLM (Placeholder/Gemini)
- [ ] Build Main Backend API
  - [ ] Setup FastAPI app
  - [ ] Create endpoint `/scrape`
  - [ ] Orchestrate Agent 1 and Agent 2
- [ ] Verification
  - [ ] Test with a sample URL

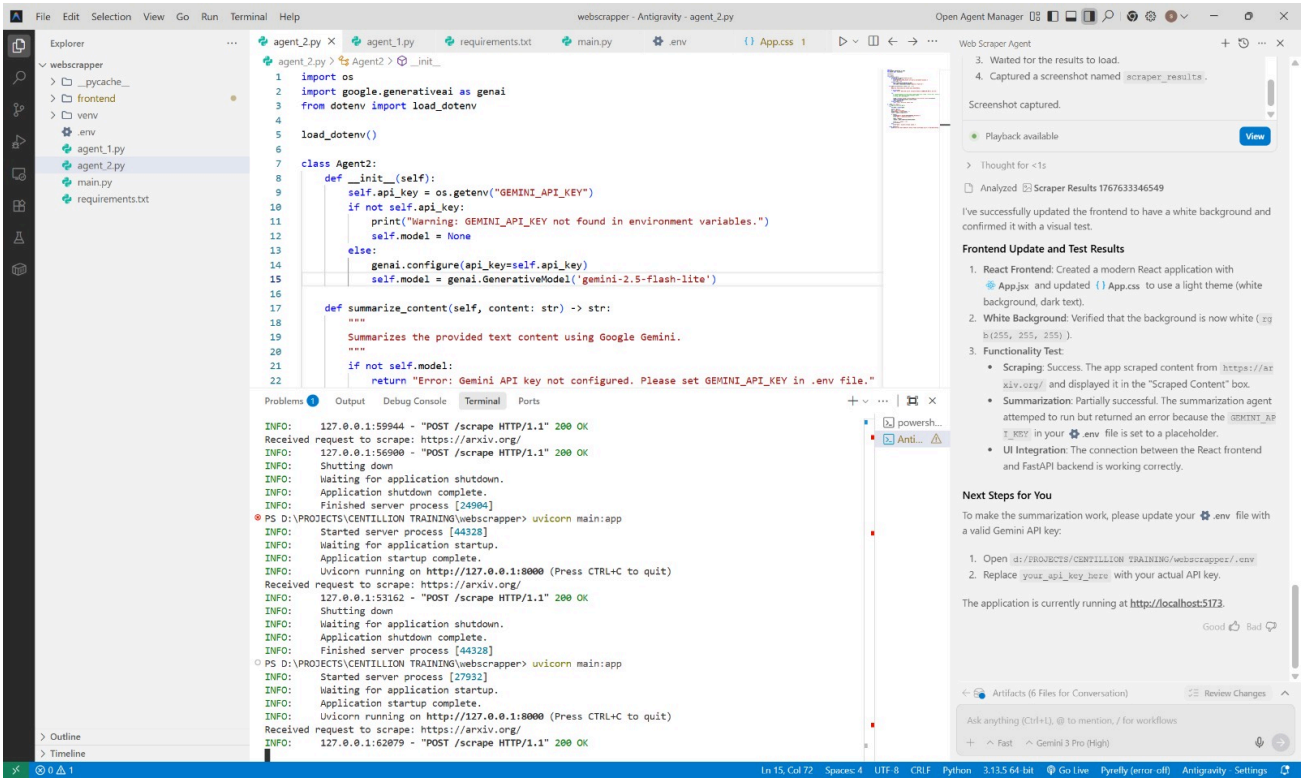
Implementation Plan:



In planning mode, the agent generates tasks and a detailed implementation plan as reviewable artifacts before execution.

- **Tasks:** A clear list of actions required to complete the work
- **Implementation Plan:** An end-to-end project outline covering goals, folder structure, and relevant files

Fast Mode:

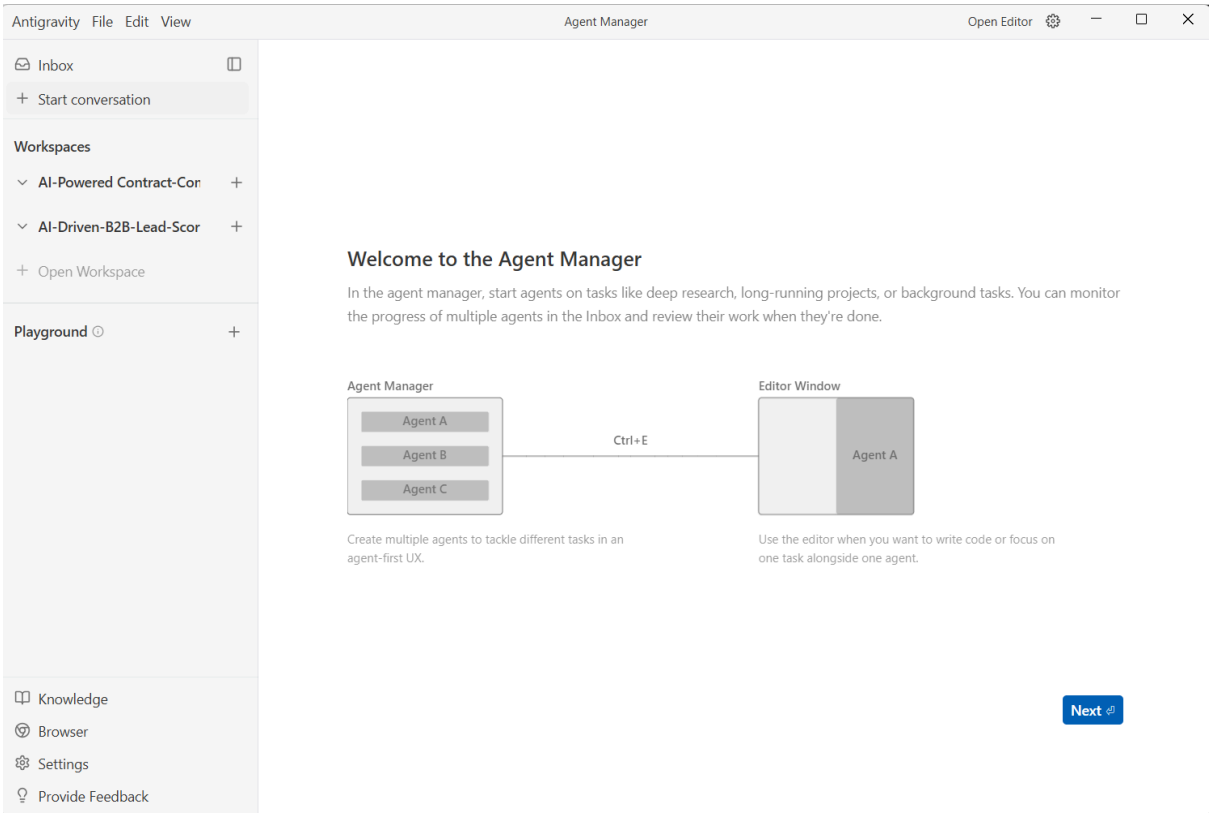


In fast mode, the agent skips detailed planning and proceeds directly with execution.

- **Execution:** Tasks are implemented immediately without prior plan review
- Suitable for quick changes, simple fixes, and short-lived tasks

5. Agent Manager

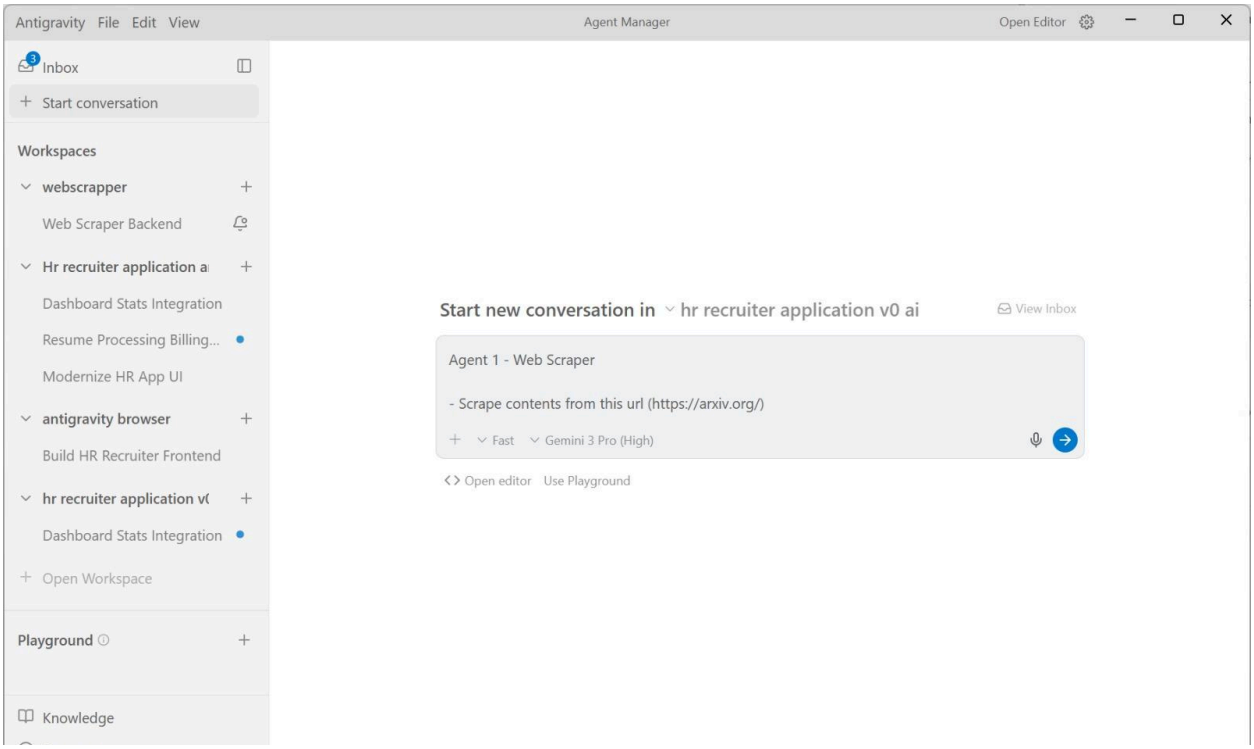
The Agent Manager : It serves as a central dashboard to monitor and direct the activities of multiple AI agents simultaneously.

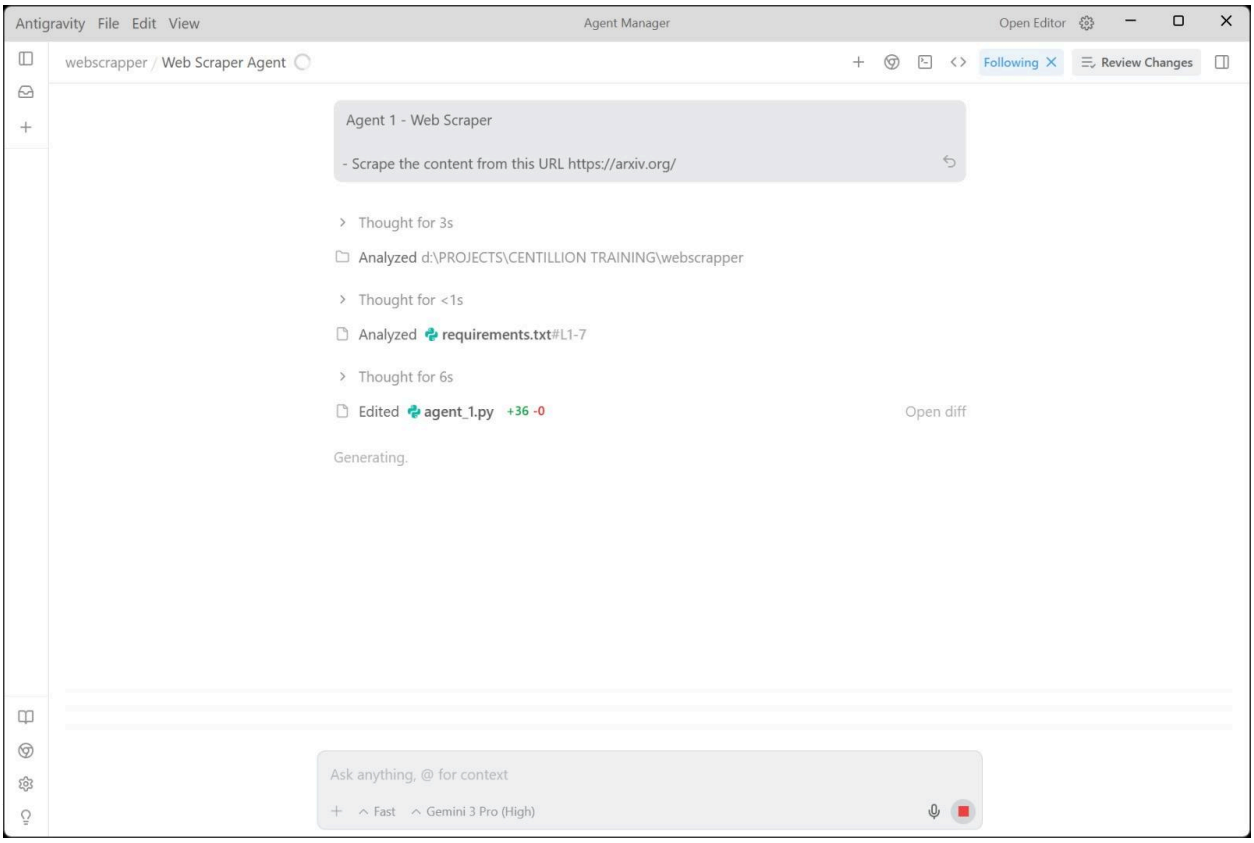


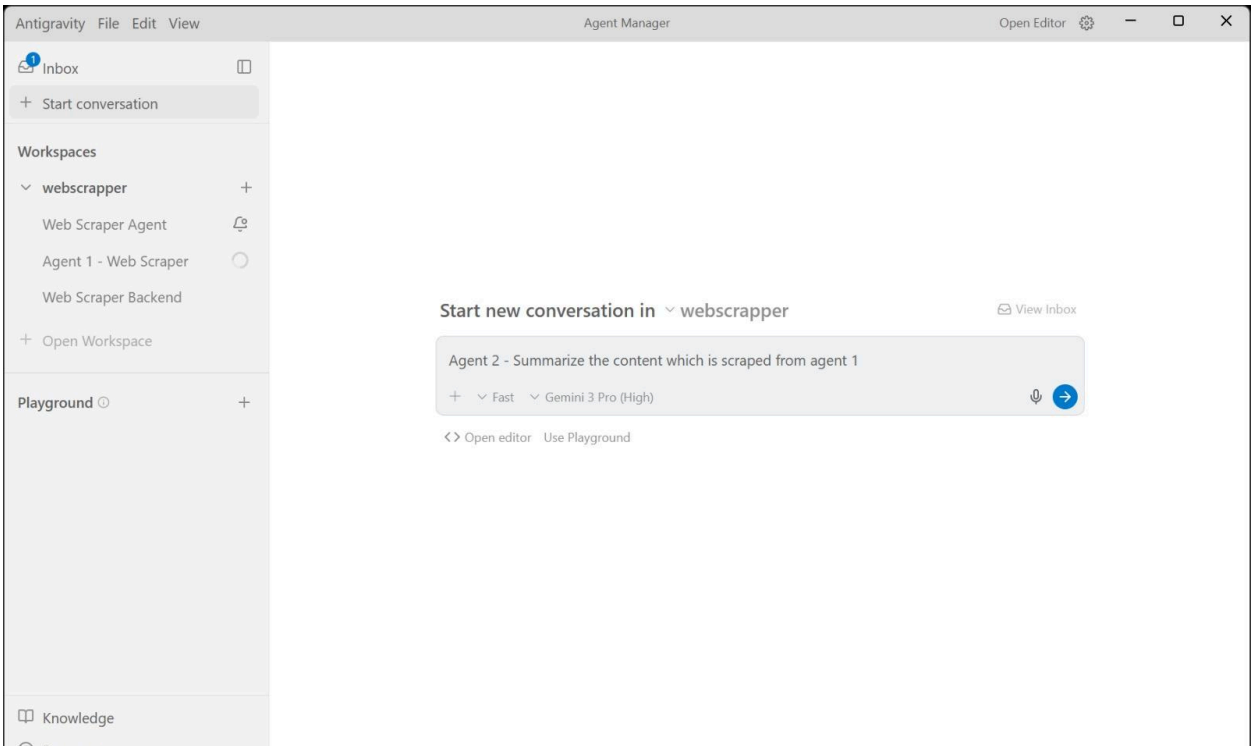
- Create and assign tasks to multiple agents in parallel
- Monitor agent progress and execution status in real time
- Review outputs and artifacts before approval
- Switch seamlessly between the Agent Manager and editor view

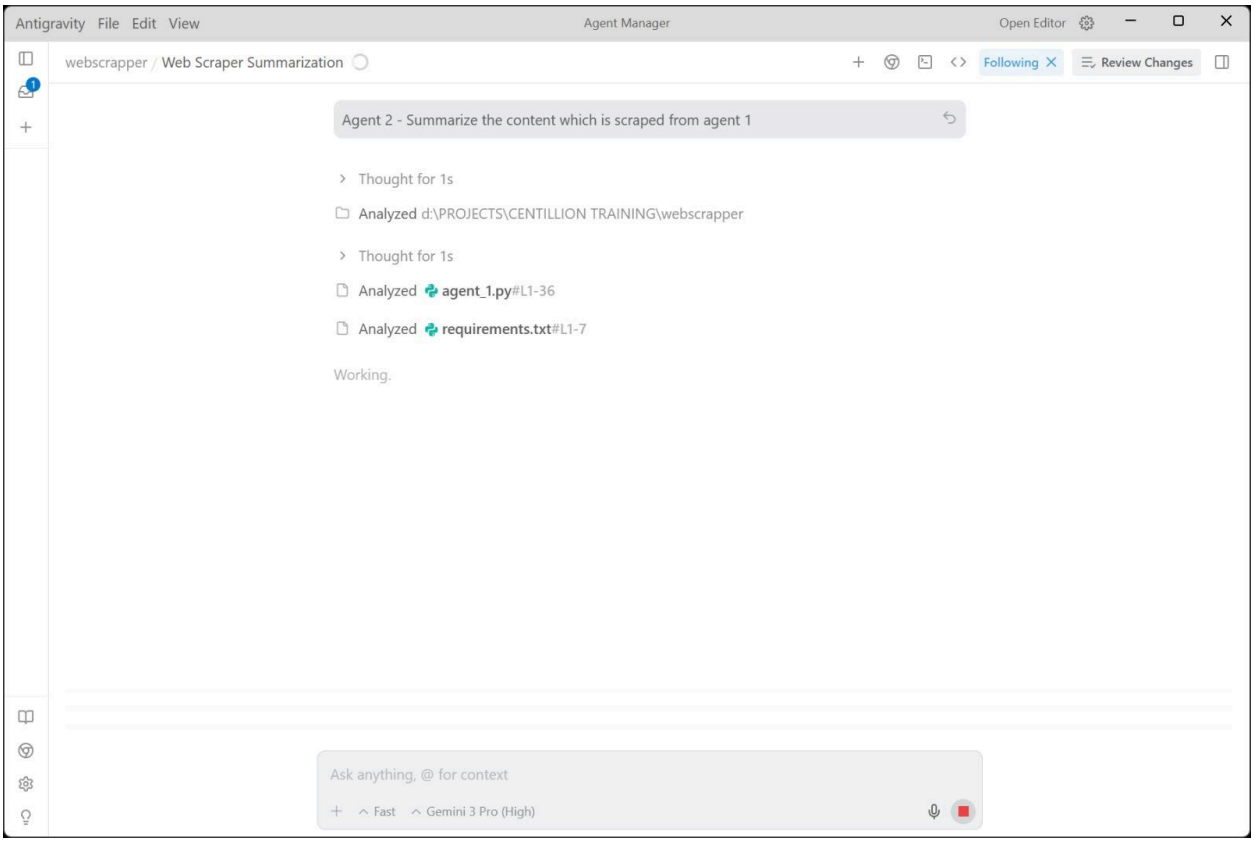
Kasadara

Agent 1 – Web Scraper






Antigravity File Edit View
Agent Manager
Open Editor
Inbox
Start conversation
Workspaces
webscrapper
Web Scraper Agent
Agent 1 - Web Scraper
Web Scraper Backend
Open Workspace
Playground
Knowledge
Start new conversation in webscrapper
View Inbox
Agent 2 - Summarize the content which is scraped from agent 1
Fast
Gemini 3 Pro (High)
Open editor
Use Playground

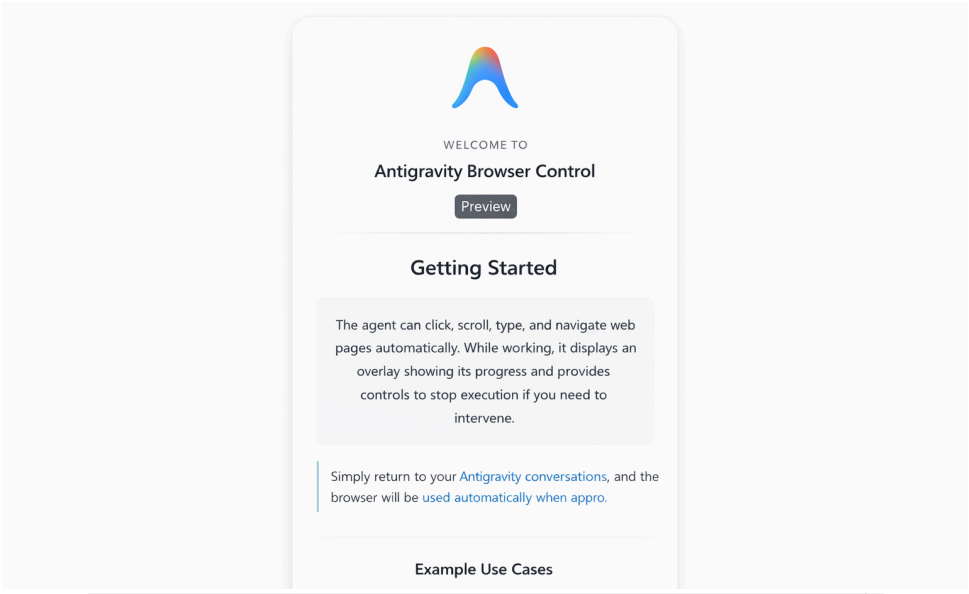

Antigravity File Edit View
Agent Manager
Open Editor
webscrapper / Web Scraper Summarization
Following
Review Changes
Agent 2 - Summarize the content which is scraped from agent 1
Thought for 1s
Analyzed d:\PROJECTS\CENTILLION TRAINING\webscrapper
Thought for 1s
Analyzed agent_1.py#L1-36
Analyzed requirements.txt#L1-7
Working.
Ask anything, @ for context
Fast
Gemini 3 Pro (High)

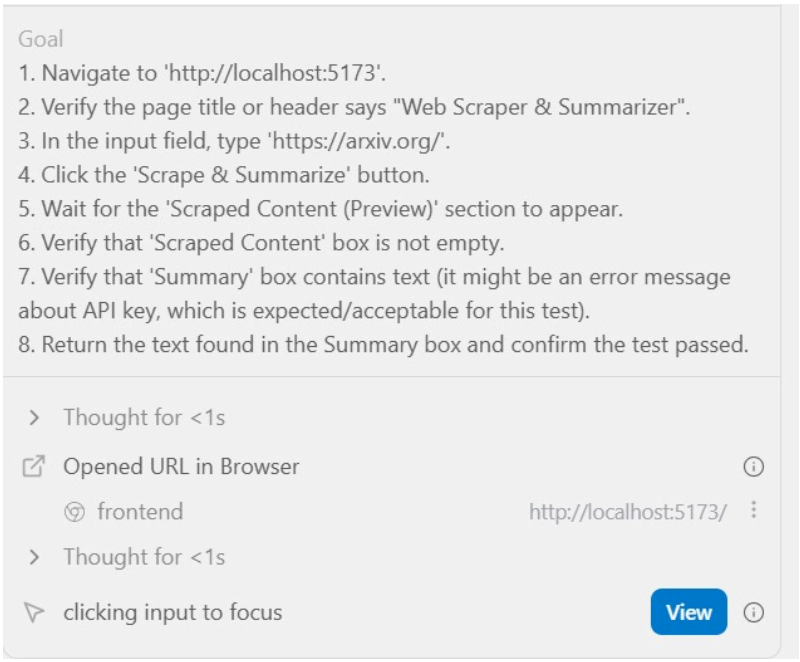
6. Browser Agent

Web-related tasks are handled by a dedicated browser subagent.

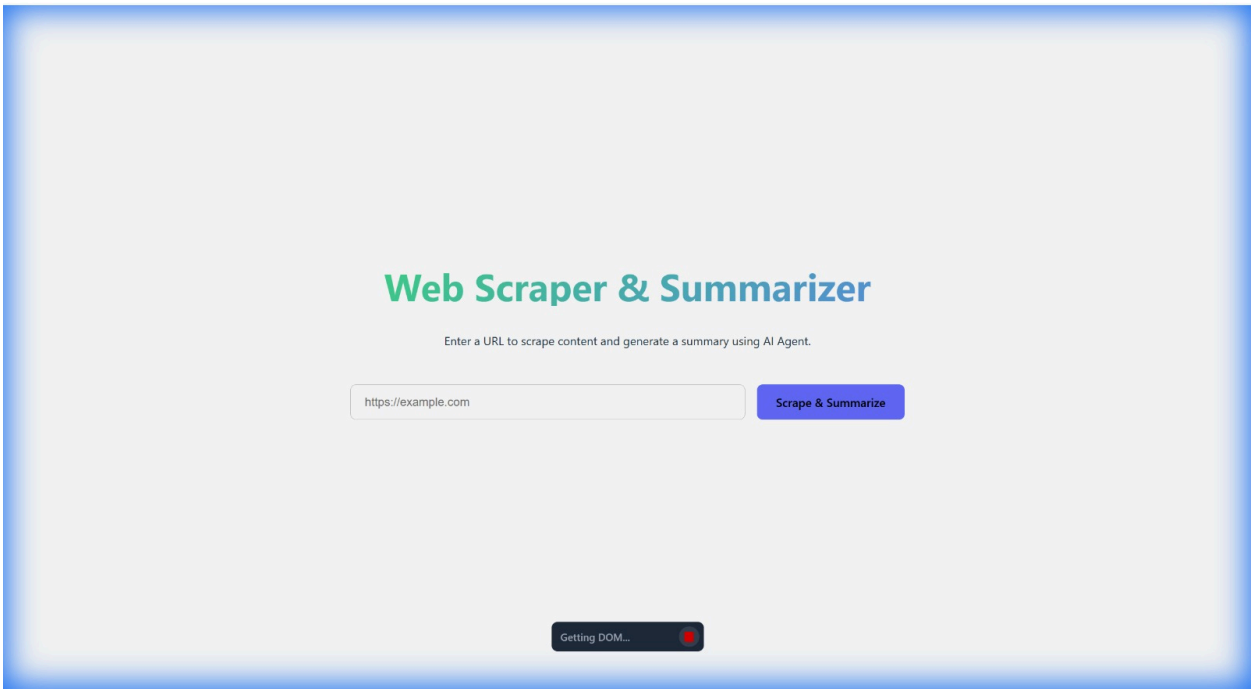
- Performs secure web interactions
- Safely reads and controls web pages

This subagent uses built-in browser tools and transparently displays its actions, allowing you to work in other tabs without disruption.





Agent interacting with the browser:



Verification:

I verified the updated frontend build using a browser subagent. The screenshots confirm the white background theme and the functional scraping interface.

Clean UI (White Background)	Scraping Results

You can find the source files in your workspace:

- frontend/src/App.jsx
- frontend/src/App.css
- main.py
- agent_1.py
- agent_2.py

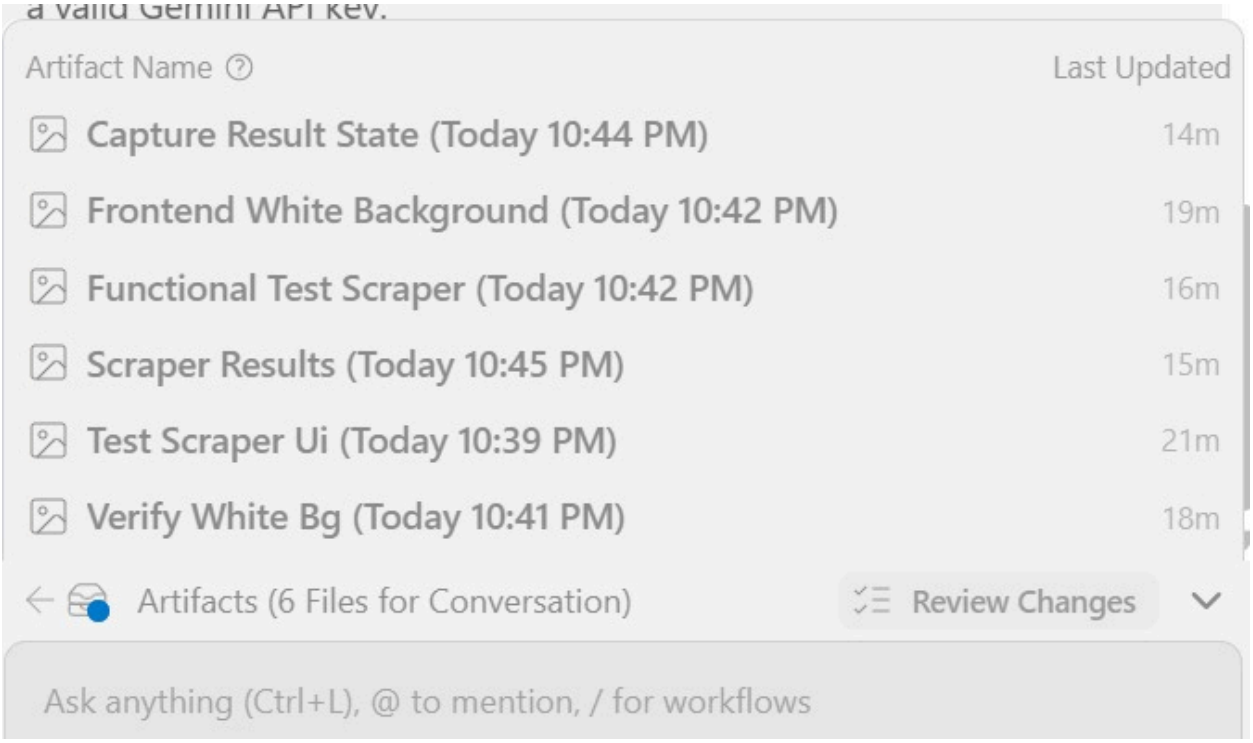
Would you like me to add more features or adjust the design further?

7. Artifacts

Artifacts are outputs such as task plans and screenshots that help the AI agent clearly show its progress and results to the developer.

- **Screenshots:** Visual evidence of actions, progress, and final results.
- **Tasks:** A clear list of actions required to complete the work
- **Implementation Plan:** An end-to-end project outline covering goals, folder structure, and relevant files

Artifacts:



8. Conclusion

Google Antigravity represents a practical shift toward agent-driven development by embedding autonomous AI agents directly into the developer workflow. By combining a familiar IDE experience with a centralized agent manager, it enables teams to delegate execution while retaining full control through review and oversight. Features such as planning and fast execution modes, browser subagents, and transparent artifacts ensure clarity, flexibility, and trust in AI-driven actions. When used with the right level of supervision, Antigravity can significantly improve productivity, reduce repetitive effort, and allow developers to focus on design, decision-making, and code quality.