Trang 1.2: Lộ trình phát triển và các thành phần chính (Agents, Skills, Workflows).

1. Lịch sử phát triển

Antigravity Kit được phát triển bởi **vudovn** và liên tục được cải tiến dựa trên nhu cầu thực tế từ cộng đồng. Dự án tuân theo chuẩn **Semantic Versioning** (Phiên bản ngữ nghĩa).

Phiên bản 2.0.0 — Ra mắt chính thức

Đây là phiên bản nền tảng, định hình toàn bộ kiến trúc của Antigravity Kit với đầy đủ 3 thành phần cốt lõi:

- Hệ thống **20 AI Agent** chuyên gia theo từng lĩnh vực.
- **37 Skills** — các module kiến thức chuyên sâu có thể tải theo yêu cầu.
- **11 Slash Workflows** — các quy trình làm việc kích hoạt bằng lệnh gạch chéo.
- **CLI Tool** (`ag-kit`) để cài đặt và cập nhật bộ kit một cách dễ dàng.

Phiên bản 2.0.1 — Tháng 01/2026

Tập trung vào chất lượng và chuẩn hóa quy trình:

- Bổ sung

  AGENT_FLOW.md — tài liệu luồng kiến trúc AI Agent toàn diện.
- Tài liệu hóa chính thức **Giao thức Định tuyến Agent** (Agent Routing Checklist) — các bước bắt buộc trước khi AI viết code hoặc thiết kế.
- Tài liệu hóa **Cổng Socratic** (Socratic Gate Protocol) — quy trình hỏi làm rõ yêu cầu trước khi triển khai.
- **Hợp nhất 2 Skills thành 1:** `nextjs-best-practices` + `react-patterns` → `react-best-practices` (gọn hơn, mạnh hơn).

- Thêm Skill mới: `web-design-guidelines` — Chuẩn thiết kế web chuyên nghiệp với hơn 100 quy tắc về UX và Accessibility.
- **Làm rõ thuật ngữ quan trọng:** "Parallel Execution" được đổi thành "Sequential Multi-Domain Execution" để phản ánh chính xác cách AI thực sự hoạt động.

Phiên bản 2.0.2 — Tháng 02/2026

Mở rộng sang hệ sinh thái Rust:

- Thêm Skill mới: **`rust-pro`** — Lập trình hệ thống chuyên sâu với Rust 1.75+ (async, Tokio, axum).
- Cải thiện Workflow `orchestrate.md`.

---

2. Kiến trúc tổng thể

Toàn bộ hệ thống Antigravity Kit hoạt động theo mô hình **"Tải theo yêu cầu" (Load on-demand)** :

```
[ Yêu cầu của người dùng ]
          |
          ▼
[ Hệ thống Định tuyến thông minh ]
  (Đọc GEMINI.md + ARCHITECTURE.md)
          |
          ▼
[ Chọn Agent phù hợp ]
  (frontend-specialist? backend-specialist? debugger?...)
          |
          ▼
[ Agent tải Skills cần thiết ]
  (react-best-practices, api-patterns, systematic-debugging...)
          |
          ▼
[ Trả lời ở cấp độ chuyên gia ]
```

Luồng này đảm bảo rằng:

- AI **không** lãng phí tài nguyên bằng cách tải toàn bộ kiến thức cùng một lúc.
- Mỗi câu trả lời được hỗ trợ bởi **đúng loại kiến thức chuyên sâu** phù hợp với vấn đề.

3. Thành phần 1: Agents (Đặc vụ AI)

3.1. Khái niệm

Mỗi Agent là một **file Markdown** (.md) được lưu trong `.agent/agents/`. File này chứa phần cấu hình YAML và bản mô tả nhân cách/vai trò chuyên môn của AI khi đóng vai trò Agent đó.

Cấu trúc một file Agent:

```
---
name: frontend-specialist
description: Frontend architect expert for web applications
tools: Read, Edit, Write, Bash
skills: react-best-practices, frontend-design, tailwind-patterns
---

# Frontend Specialist

Bạn là một kiến trúc sư Frontend cấp cao...
[Các quy tắc, hành vi và kiến thức chuyên sâu về Frontend]
```

3.2. Danh sách 20 Agents và vai trò

Agent	Chuyên môn
`orchestrator`	Điều phối nhiều Agent phối hợp cho tác vụ phức tạp
`project-planner`	Khám phá yêu cầu và lập kế hoạch chi tiết
`frontend-specialist`	Giao diện Web (React, Next.js, Tailwind CSS)
`backend-specialist`	API, Business Logic, Cơ sở dữ liệu
`database-architect`	Thiết kế Schema và tối ưu SQL
`mobile-developer`	iOS, Android, React Native

Agent	Chuyên môn
`game-developer`	Logic game, cơ học trò chơi
`devops-engineer`	CI/CD, Docker, hạ tầng
`security-auditor`	Kiểm tra bảo mật và tuân thủ
`penetration-tester`	Bảo mật tấn công (Offensive Security)
`test-engineer`	Chiến lược kiểm thử toàn diện
`debugger`	Phân tích nguyên nhân gốc rễ của lỗi
`performance-optimizer`	Tốc độ tải và Web Vitals
`seo-specialist`	Xếp hạng tìm kiếm và khả năng hiển thị
`documentation-writer`	Hướng dẫn sử dụng và tài liệu kỹ thuật
`product-manager`	Yêu cầu nghiệp vụ và câu chuyện người dùng
`product-owner`	Chiến lược sản phẩm và quản lý backlog
`qa-automation-engineer`	Kiểm thử E2E và CI Pipeline
`code-archaeologist`	Phân tích và tái cấu trúc code cũ (Legacy)
`explorer-agent`	Khám phá và phân tích codebase hiện có

3.3. Cách Agent phối hợp với Skills

Mỗi Agent trong phần cấu hình `skills:` liệt kê các Skills mà Agent đó được phép truy cập. Khi Agent nhận được yêu cầu, nó sẽ quyết định xem cần tải Skills nào và đọc chúng để có đủ kiến thức.

Ví dụ:

- `backend-specialist` sử dụng: `api-patterns`, `nodejs-best-practices`, `database-design`
- `security-auditor` sử dụng: `vulnerability-scanner`, `red-team-tactics`
- `debugger` sử dụng: `systematic-debugging`

---

4. Thành phần 2: Skills (Kỹ năng)

4.1. Khái niệm

Skills là **gói kiến thức module** — không phải code cứng nhắc, mà là các **nguyên lý và khung ra quyết định** giúp AI hiểu *tại sao* nên làm theo cách này thay vì chỉ copy một pattern có sẵn.

> **Khác biệt cốt lõi:** Skills giúp AI đưa ra quyết định phù hợp với ngữ cảnh dự án cụ thể của bạn, không phải câu trả lời mẫu chung chung.

4.2. Cấu trúc thư mục một Skill

```
.agent/skills/
└── react-best-practices/
    ├── SKILL.md          # (Bắt buộc) Siêu dữ liệu và hướng dẫn chính
    ├── sections/         # (Tùy chọn) Hướng dẫn chi tiết từng phần
    ├── examples/         # (Tùy chọn) Ví dụ triển khai tham khảo
    ├── scripts/          # (Tùy chọn) Script Python/Bash hỗ trợ
    └── assets/           # (Tùy chọn) Hình ảnh, logo
```

4.3. Danh sách 36 Skills theo nhóm

Frontend & UI (5 Skills)

- `react-best-practices` — Tối ưu React & Next.js (57 quy tắc từ Vercel).
- `web-design-guidelines` — Kiểm toán UI web — 100+ quy tắc Accessibility, UX, Performance.
- `tailwind-patterns` — Tailwind CSS v4 và các tiện ích hiện đại.
- `frontend-design` — Hệ thống thiết kế và các UI/UX pattern.
- `ui-ux-pro-max` — 50 phong cách, 21 bảng màu, 50 font chữ.

Backend & API (4 Skills)

- `api-patterns` — Thiết kế API (REST, GraphQL, tRPC).
- `nestjs-expert` — NestJS modules, Dependency Injection.
- `nodejs-best-practices` — Node.js async pattern và kiến trúc.
- `python-patterns` — Tiêu chuẩn Python, FastAPI, Django.

Database (2 Skills)

- `database-design` — Thiết kế schema và tối ưu hóa.
- `prisma-expert` — Prisma ORM và migrations.

TypeScript/JavaScript (1 Skill)

- `typescript-expert` — Lập trình kiểu nâng cao, hiệu năng.

Cloud & Infrastructure (3 Skills)

- `docker-expert` — Container hóa và Docker Compose.
- `deployment-procedures` — CI/CD và quy trình deploy.
- `server-management` — Quản lý hạ tầng máy chủ.

Testing & Quality (5 Skills)

- `testing-patterns` — Jest, Vitest và các chiến lược kiểm thử.
- `webapp-testing` — Kiểm thử E2E bằng Playwright.
- `tdd-workflow` — Phát triển hướng kiểm thử (TDD).
- `code-review-checklist` — Tiêu chuẩn review code.
- `lint-and-validate` — Linting và validation.

Security (2 Skills)

- `vulnerability-scanner` — Kiểm tra bảo mật, OWASP 2025.
- `red-team-tactics` — Bảo mật tấn công, chiến thuật red team.

Architecture & Planning (4 Skills)

- `app-builder` — Tạo khung ứng dụng full-stack.
- `architecture` — Các mẫu thiết kế hệ thống.
- `plan-writing` — Lập kế hoạch và phân rã công việc.
- `brainstorming` — Đặt câu hỏi Socratic để làm rõ yêu cầu.

Mobile (1 Skill), Game (1 Skill), SEO (2 Skills), Shell/CLI (2 Skills), Other (6 Skills)

- *(Bao gồm: `mobile-design`, `game-development`, `seo-fundamentals`, `geo-fundamentals`, `bash-linux`, `powershell-windows`, `clean-code`, `behavioral-modes`, `parallel-agents`, `mcp-builder`, `documentation-templates`, `i18n-localization`, `performance-profiling`, `systematic-debugging`, `rust-pro`)*

---

5. Thành phần 3: Workflows (Quy trình làm việc)

5.1. Khái niệm

Workflow là các **file Markdown** được lưu trong `.agent/workflows/`. Mỗi file định nghĩa một quy trình từng bước cụ thể để thực hiện một tác vụ lập trình phổ biến.

Cách kích hoạt: Gõ lệnh gạch chéo tương ứng trong cửa sổ chat với AI.

```
/tên-workflow [mô tả yêu cầu]
```

5.2. Tính năng Turbo

Một số bước trong Workflow được đánh dấu `// turbo`. Điều này có nghĩa là AI có thể **tự động chạy lệnh terminal an toàn** ở bước đó mà không cần dừng lại để hỏi bạn, giúp tăng tốc độ thực hiện.

5.3. Tùy chỉnh Workflow

Bạn hoàn toàn có thể tự tạo Workflow riêng bằng cách:

1. Tạo file `.md` mới trong thư mục `.agent/workflows/`.
2. Thêm phần mô tả ở đầu file và các bước thực hiện bên dưới.
3. Gọi Workflow bằng cách gõ `/tên-file` trong chat.

Ví dụ tạo Workflow tùy chỉnh:

```
---
description: Deploy ứng dụng lên môi trường staging
---

# Quy trình Deploy Staging

1. Chạy toàn bộ test suite
2. Build production bundle
3. Deploy lên server staging
4. Chạy smoke test để xác minh
```

6. Thành phần bổ sung: Scripts kiểm tra

Ngoài 3 thành phần chính, Antigravity Kit còn cung cấp **2 script kiểm tra tổng thể** bằng Python trong thư mục `.agent/scripts/`:

Script	Mục đích	Khi nào dùng
`checklist.py`	Kiểm tra ưu tiên (Core checks)	Trong quá trình phát triển, trước khi commit

Script	Mục đích	Khi nào dùng
`verify_all.py`	Kiểm tra toàn diện (Full suite)	Trước khi release lên production

Checklist.py kiểm tra theo thứ tự ưu tiên:

- **P0:** Bảo mật (Lỗ hổng, secrets bị lộ).
- **P1:** Chất lượng code (Lint, Type check).
- **P2:** Kiểm tra Schema cơ sở dữ liệu.
- **P3:** Chạy bộ test tự động.
- **P4:** Kiểm tra UX và Accessibility.
- **P5:** Kiểm tra SEO.
- **P6:** Hiệu năng (Lighthouse — cần cung cấp URL).

> **Lưu ý về phiên bản:** Thông tin trong tài liệu này phản ánh bộ kit **phiên bản 2.0.2**. Để xem lịch sử thay đổi đầy đủ, tham khảo file `CHANGELOG.md` trong thư mục gốc của dự án hoặc trang GitHub của dự án.

Phiên bản #4
Được tạo 2026-03-04 05:00:06 UTC bởi Nam Đặng
Được cập nhật 2026-03-05 04:15:59 UTC bởi Nam Đặng