Trang 3.1: AI Agents - Các đặc vụ chuyên gia và cơ chế định tuyến thông minh (Intelligent Routing).

1. Agent là gì?

Agent là một "nhân cách AI chuyên gia" được cấu hình sẵn với kiến thức sâu, công cụ và hành vi riêng biệt cho một lĩnh vực cụ thể trong phát triển phần mềm.

Hãy hình dung Agent như một chuyên viên tư vấn: thay vì hỏi một người biết tất cả một cách chung chung, bạn đang nói chuyện trực tiếp với **kiến trúc sư Frontend**, **chuyên gia bảo mật**, hay **kỹ sư kiểm thử** tùy theo từng vấn đề.

---

2. Hệ thống Định tuyến thông minh (Intelligent Routing)

Cơ chế hoạt động

Điểm đặc biệt quan trọng nhất: **bạn không cần phải gọi tên Agent thủ công**. Hệ thống sẽ tự động phân tích yêu cầu của bạn và kích hoạt (các) Agent phù hợp nhất.

Yêu cầu của bạn
|

▼

Hệ thống phân tích (Silent Analysis)

|

▼

Phát hiện lĩnh vực (Frontend? Backend? Security?...)

|

▼

Chọn Agent chuyên gia tương ứng

|

▼

Thông báo cho bạn Agent đang được dùng:

□□ Đang áp dụng kiến thức của @[agent-name]...

|

▼

Trả lời ở cấp độ chuyên gia

Ví dụ thực tế

Yêu cầu của bạn	Agent được kích hoạt
"Thêm xác thực JWT vào API"	`@security-auditor` + `@backend-specialist`
"Sửa lỗi nút dark mode bị lệch màu"	`@frontend-specialist`
"Login trả về lỗi 500 ngẫu nhiên"	`@debugger`
"Thiết kế schema cho bảng users và orders"	`@database-architect`
"Viết unit test cho service xác thực"	`@test-engineer`
"Tối ưu tốc độ tải trang"	`@performance-optimizer`
"Lập kế hoạch tính năng thanh toán"	`@project-planner`

Ghi đè thủ công (Override)

Nếu bạn muốn buộc sử dụng một Agent cụ thể, chỉ cần đề cập tên trong yêu cầu:

"Dùng security-auditor để review toàn bộ phần auth của tôi"
"Nhờ debugger phân tích lỗi này theo quy trình 4 bước"

---

3. Checklist bắt buộc trước khi AI viết code

Trước khi thực hiện bất kỳ tác vụ code hoặc thiết kế nào, AI **bắt buộc** phải hoàn thành checklist sau:

Bước	Kiểm tra	Nếu chưa làm
1	Đã xác định đúng Agent cho lĩnh vực này chưa?	→ Dừng lại. Phân tích lại lĩnh vực yêu cầu.
2	Đã đọc file cấu hình của Agent chưa?	→ Dừng lại. Mở và đọc `.agent/agents/{agent}.md`
3	Đã thông báo Agent đang dùng cho người dùng chưa?	→ Dừng lại. Thêm thông báo `◫ Đang áp dụng @[agent]...`
4	Đã tải các Skills cần thiết từ frontmatter chưa?	→ Dừng lại. Kiểm tra trường `skills:` và đọc chúng.

---

4. Danh sách đầy đủ 20 Agents

Nhóm Quản lý & Điều phối

Agent	Chuyên môn	Skills sử dụng
`orchestrator`	Điều phối nhiều Agent làm việc song song cho tác vụ phức tạp	`parallel-agents`, `behavioral-modes`
`project-planner`	Khám phá yêu cầu, lập kế hoạch và phân rã công việc	`brainstorming`, `plan-writing`, `architecture`
`product-manager`	Yêu cầu nghiệp vụ, user stories	`plan-writing`, `brainstorming`
`product-owner`	Chiến lược sản phẩm, quản lý backlog, định nghĩa MVP	`plan-writing`, `brainstorming`

Nhóm Phát triển

Agent	Chuyên môn	Skills sử dụng

`frontend-specialist`	Giao diện Web (React, Next.js, Tailwind CSS)	`react-best-practices`, `frontend-design`, `tailwind-patterns`, `web-design-guidelines`
`backend-specialist`	API, Business Logic, máy chủ	`api-patterns`, `nodejs-best-practices`, `database-design`
`database-architect`	Thiết kế schema, SQL, tối ưu truy vấn	`database-design`, `prisma-expert`
`mobile-developer`	iOS, Android, React Native, Flutter	`mobile-design`
`game-developer`	Logic game, cơ học trò chơi	`game-development`
`devops-engineer`	CI/CD, Docker, hạ tầng cloud	`deployment-procedures`, `docker-expert`

Nhóm Chất lượng & Bảo mật

Agent	Chuyên môn	Skills sử dụng
`security-auditor`	Kiểm tra bảo mật, tuân thủ OWASP	`vulnerability-scanner`, `red-team-tactics`
`penetration-tester`	Bảo mật tấn công (Offensive Security)	`red-team-tactics`
`test-engineer`	Chiến lược kiểm thử toàn diện	`testing-patterns`, `tdd-workflow`, `webapp-testing`
`qa-automation-engineer`	Kiểm thử E2E và CI Pipeline	`webapp-testing`, `testing-patterns`
`debugger`	Phân tích nguyên nhân gốc rễ của lỗi	`systematic-debugging`
`performance-optimizer`	Tối ưu hiệu năng và Core Web Vitals	`performance-profiling`

Nhóm Chuyên biệt

Agent	Chuyên môn	Skills sử dụng
`seo-specialist`	Xếp hạng tìm kiếm, khả năng hiển thị (SEO + GEO)	`seo-fundamentals`, `geo-fundamentals`
`documentation-writer`	Hướng dẫn sử dụng, tài liệu kỹ thuật	`documentation-templates`
`code-archaeologist`	Phân tích và tái cấu trúc code cũ (Legacy code)	`clean-code`, `code-review-checklist`
`explorer-agent`	Khám phá và phân tích codebase hiện có	*(không có skill cố định, dùng ngữ cảnh)*

---

5. Khi nào nên sử dụng Agent nào?

Tình huống	Agent phù hợp

Bắt đầu dự án mới, chưa biết thiết kế thế nào	`project-planner` → sau đó `orchestrator`
Xây dựng trang web, component UI	`frontend-specialist`
Tạo API endpoint, kết nối database	`backend-specialist`
Thiết kế bảng DB, model dữ liệu	`database-architect`
Phát hiện lỗ hổng bảo mật	`security-auditor`
Ứng dụng bị lỗi không rõ nguyên nhân	`debugger`
Muốn viết test cho code hiện có	`test-engineer`
Trang web tải chậm, điểm Lighthouse thấp	`performance-optimizer`
Cần lên kế hoạch triển khai sản phẩm	`project-planner` + `devops-engineer`
Cần phân tích một codebase lạ	`explorer-agent`

---

6. Agents làm việc cùng nhau

Các Agent có thể **phối hợp** để giải quyết các vấn đề phức tạp đa lĩnh vực. Sử dụng Agent `orchestrator` để điều phối:

Ví dụ: Xây dựng hệ thống đăng nhập hoàn chỉnh

```
orchestrator điều phối:
├── database-architect → Thiết kế bảng users
├── backend-specialist → Viết API authentication
├── security-auditor → Review bảo mật, kiểm tra JWT
├── frontend-specialist → Xây dựng form login
└── test-engineer → Viết test cho toàn bộ flow
```

> **Lưu ý thực tế:** AI thực sự xử lý tuần tự (không song song thực sự), nhưng `orchestrator` quản lý ngữ cảnh và đảm bảo mỗi phần được thực hiện đúng bởi "góc nhìn" của Agent chuyên gia tương ứng.

Phiên bản #2
Được tạo 2026-03-04 06:10:24 UTC bởi Nam Đặng
Được cập nhật 2026-03-04 07:15:08 UTC bởi Nam Đặng