

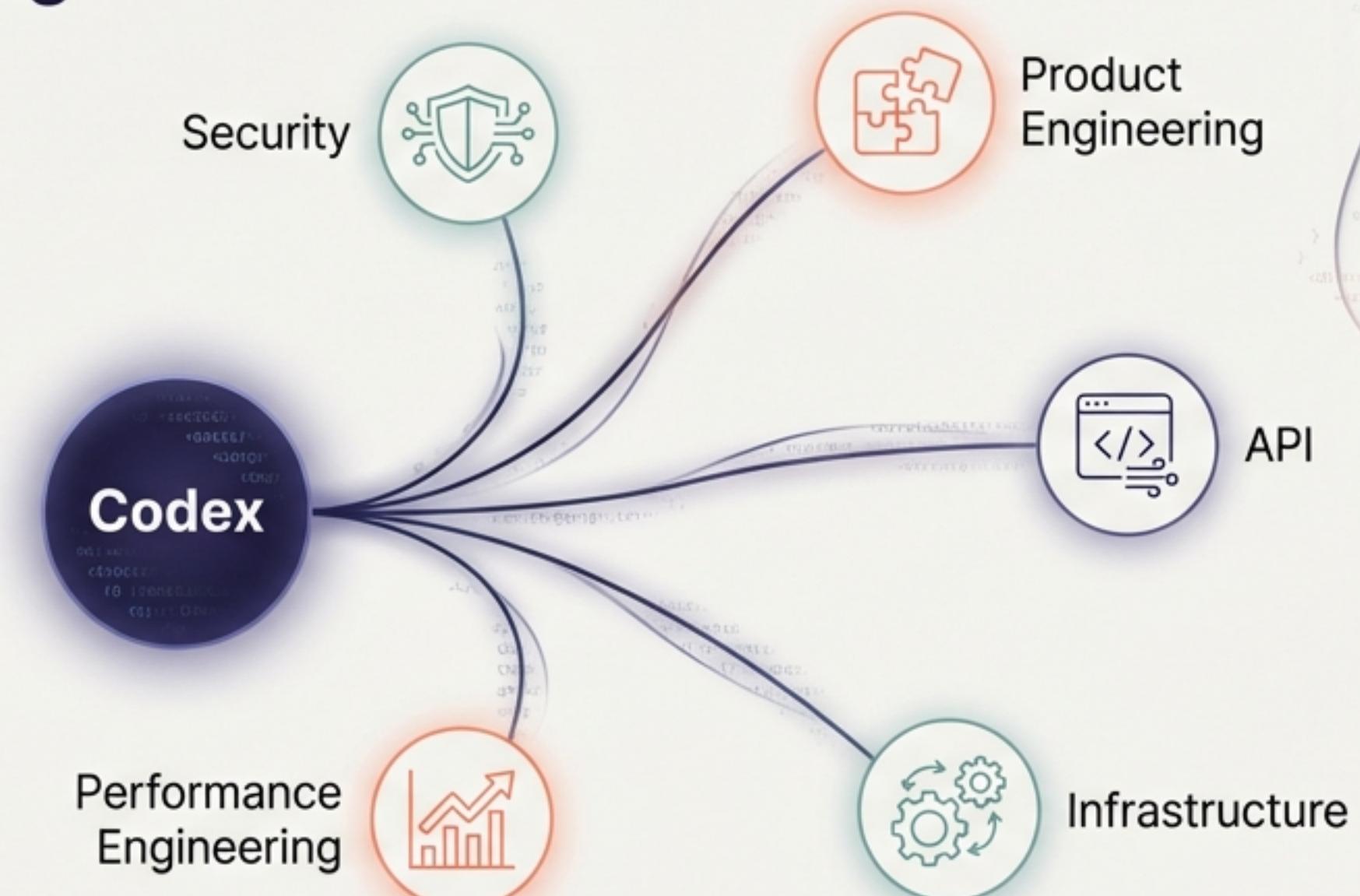
Cách OpenAI Tăng Tốc Kỹ Thuật với Codex

Bí quyết, kinh nghiệm và thực tiễn
tốt nhất từ nội bộ



Hơn cả một công cụ. Một cộng sự trong mọi dòng code.

Codex được sử dụng hàng ngày tại nhiều đội ngũ kỹ thuật của OpenAI như Security, Product Engineering, API, Infrastructure, và Performance Engineering. Nó không chỉ là một tiện ích, mà là một phần không thể thiếu trong quy trình làm việc, giúp chúng tôi:



Tăng tốc (Accelerate): Từ việc xây dựng tính năng mới đến xử lý sự cố.

Cải thiện chất lượng (Improve Quality): Viết code tốt hơn và giảm thiểu nợ kỹ thuật.

Quản lý sự phức tạp (Manage Complexity): Nắm bắt các hệ thống lớn và tái cấu trúc các codebase phức tạp.

Hành trình khám phá sức mạnh của Codex



1. BỐI CẢNH

Tại sao Codex lại quan trọng tại OpenAI.



2. SỰ CHUYỂN HÓA

7 trường hợp sử dụng thay đổi cách chúng tôi làm việc.



3. CẨM NANG

Các bí quyết để bạn khai thác tối đa sức mạnh của Codex.

Làm Chủ Mã Nguồn Phức Tạp

Từ thấu hiểu hệ thống đến tái cấu trúc quy mô lớn, Codex giúp các kỹ sư điều hướng và cải thiện những codebase khó nhất.

Nhanh chóng thấu hiểu và tái cấu trúc hiệu quả

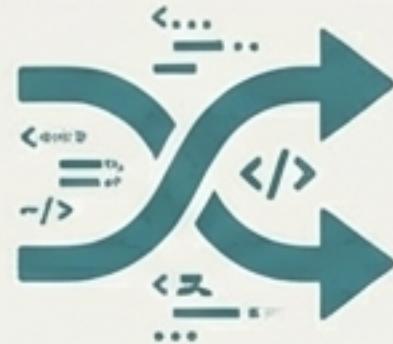


Thấu Hiểu Mã Nguồn

Giúp các đội ngũ nhanh chóng nắm bắt các phần lõi của codebase, truy vết luồng dữ liệu, và điều tra sự cố.

"Khi tôi đang on-call, tôi dán stack trace và hỏi Codex luồng xác thực nằm ở đâu. Nó nhảy thẳng đến đúng file để tôi có thể xử lý sự cố nhanh chóng."

– Site Reliability Engineer, API Platform



Tái Cấu Trúc & Di Chuyển

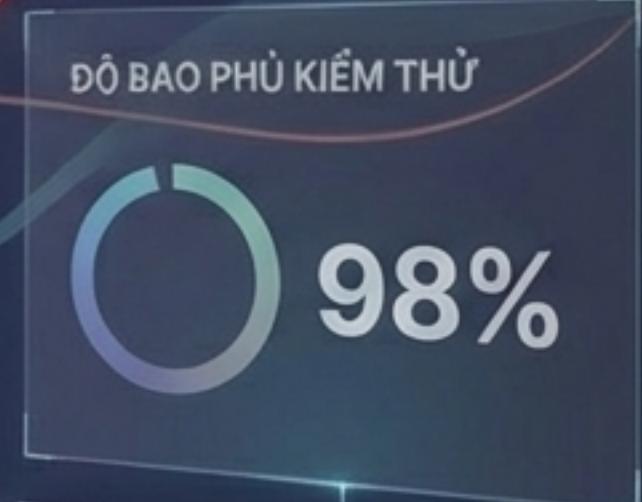
Áp dụng các thay đổi nhất quán trên hàng chục file, dọn dẹp code cũ, và di chuyển sang các dependency mới.

"Codex đã hoán đổi mọi lệnh gọi getUserByll() cũ sang pattern dịch vụ mới của chúng tôi và mở PR. Nó hoàn thành trong vài phút việc đáng lẽ mất hàng giờ."

– Backend Engineer, ChatGPT Web

Xây Dựng Nền Tảng Vững Chắc

Codex không chỉ giúp sửa lỗi mà còn chủ động nâng cao chất lượng code thông qua tối ưu hiệu năng và tăng cường độ bao phủ kiểm thử.



Tối ưu hiệu năng và mở rộng kiểm thử



Tối Ưu Hiệu Năng

Xác định các điểm nghẽn, đề xuất giải pháp thay thế hiệu quả, và chủ động giảm nợ kỹ thuật bằng cách tìm ra các pattern rủi ro.

"Codex rất tuyệt vời trong việc phát hiện các vấn đề về hiệu năng một cách nhanh chóng—tôi tiết kiệm 30 phút làm việc chỉ bằng 5 phút viết prompt."

– Platform Engineer, Model Serving



Cải Thiện Độ Bao Phủ Kiểm Thử

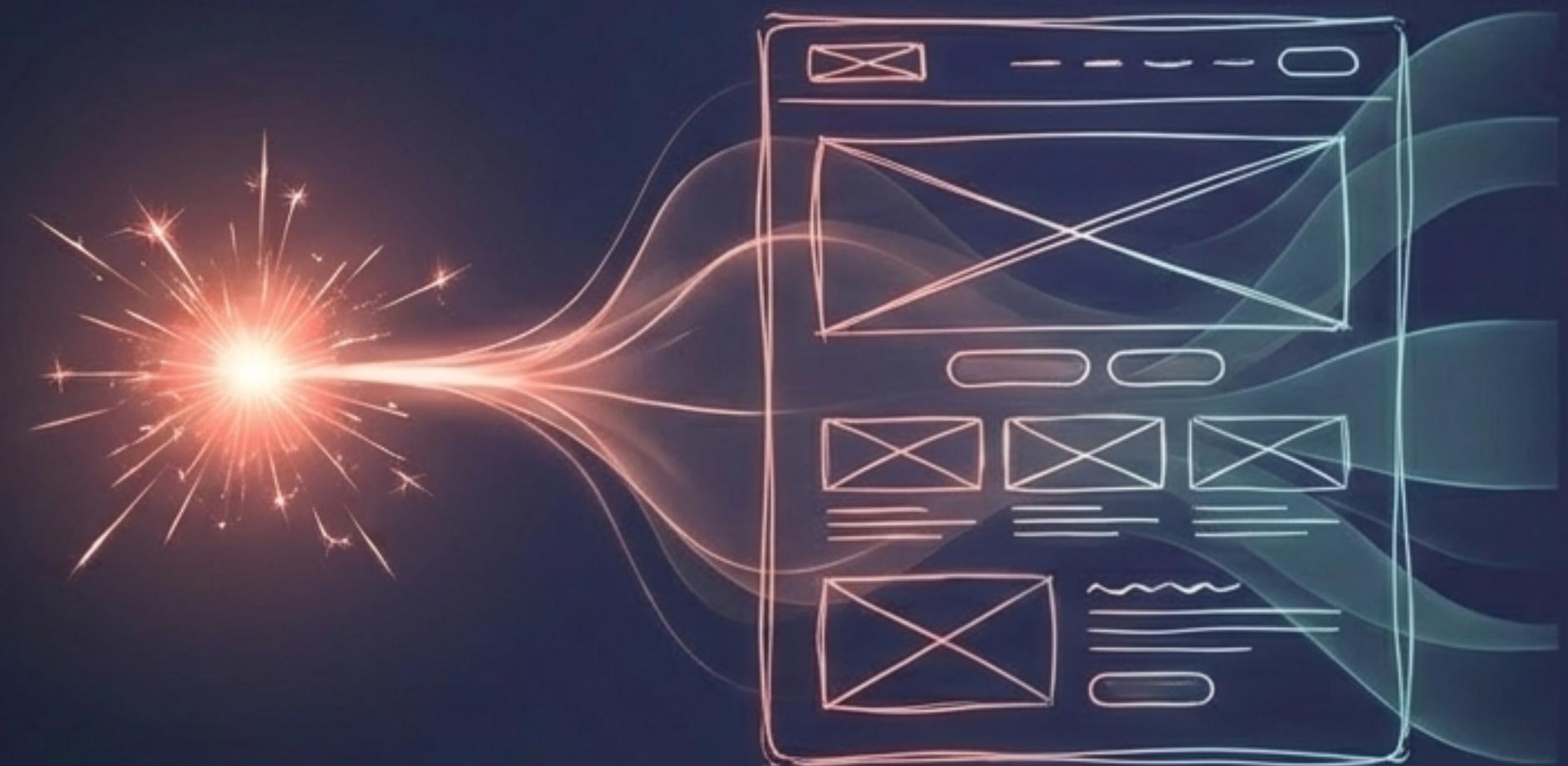
Viết unit test và integration test nhanh hơn, đặc biệt cho các trường hợp biên, đầu vào rỗng, hoặc các trạng thái không hợp lệ thường bị bỏ qua.

"Tôi chỉ định Codex vào các module có độ bao phủ thấp qua đêm và thức dậy với các PR unit-test sẵn sàng để chạy."

– Frontend Engineer, ChatGPT Desktop

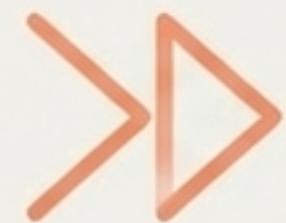
Tăng Tốc Sáng Tạo

Từ ý tưởng ban đầu đến sản phẩm hoàn thiện, Codex giúp các kỹ sư phát triển nhanh hơn, luôn giữ được mạch làm việc và khám phá các giải pháp mới.



```
class Code {  
    @Override  
    public function itemevient) {  
        this.Đeobuitt = heloS;  
    }  
  
    public semat vsson() {  
        return true;  
    }  
  
    public function int eaits() {  
        //return ssete;  
    }  
  
    int main() {  
        System.logText "JetBrains Mono ";  
    }  
}
```

Từ ý tưởng đến sản phẩm mà không lỡ nhịp



Tăng Tốc Độ Phát Triển

Tạo sườn code (scaffolding), API stubs, và xử lý các tác vụ nhỏ nhưng cần thiết ở cuối dự án như tạo script triển khai hoặc file cấu hình.

"Tôi đã họp cả ngày mà vẫn merge được 4 PR vì Codex đã làm việc ở chế độ nền."

– Product Engineer, ChatGPT Enterprise



Giữ Mạch Công Việc

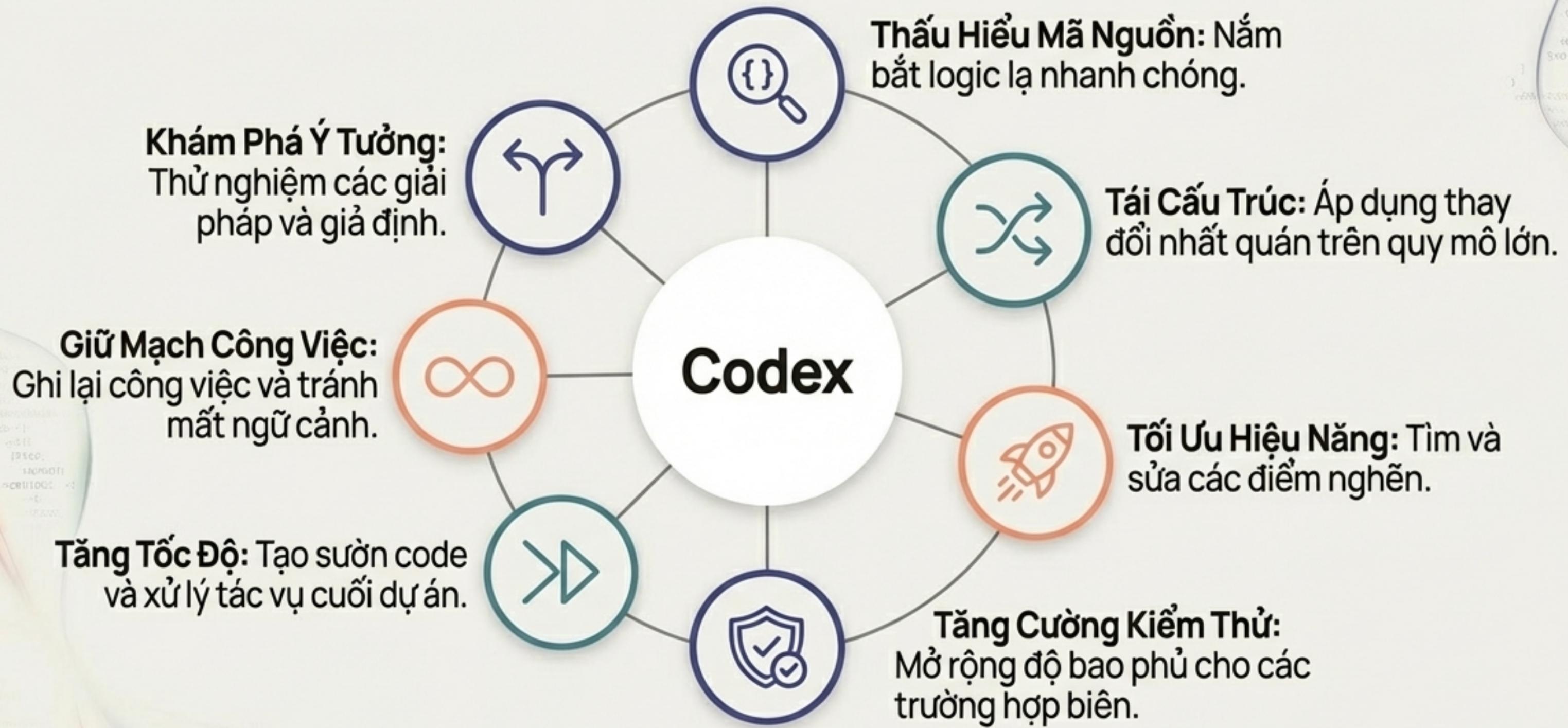
Ghi lại công việc dở dang, chuyển ghi chú hoặc các cuộc thảo luận trên Slack thành prototype mà không làm mất ngữ cảnh khi bị gián đoạn.



Khám Phá & Lên Ý Tưởng

Tìm kiếm các giải pháp thay thế, khám phá các pattern lạ, và kiểm chứng các giả định thiết kế để đưa ra lựa chọn triển khai sắc bén hơn.

Codex: Đối Tác Toàn Diện Trong Vòng Đời Kỹ Thuật



Từ ‘Biết’ đến ‘Làm’: 6 Bí Quyết Để Tối Ưu Hóa Codex

Codex hoạt động hiệu quả nhất khi được cung cấp cấu trúc, ngữ cảnh và không gian để lặp lại. Đây là những thói quen mà các đội ngũ tại OpenAI đang trau dồi để nhận được giá trị nhất quán trong công việc hàng ngày.



Nền tảng cho sự thành công: Ngữ cảnh và Kế hoạch

1.



Bắt đầu với Chế độ Hỏi (Start with Ask Mode)

Với các thay đổi lớn, hãy yêu cầu Codex tạo một kế hoạch triển khai trước. Kế hoạch này sau đó trở thành đầu vào cho các prompt viết code, giúp kết quả đầu ra không bị chêch hướng.

2.



Cấu trúc Prompt như một Github Issue

Cung cấp ngữ cảnh đầy đủ: đường dẫn file, tên component, đoạn diff, và trích dẫn tài liệu. Các prompt như “Triển khai theo cách giống như trong [module X]” sẽ cải thiện kết quả.

3.



Cải thiện Môi trường Liên tục

Thiết lập script khởi động, biến môi trường, và quyền truy cập internet sẽ giảm đáng kể tỷ lệ lỗi. Tinh chỉnh cấu hình môi trường của Codex sau mỗi lần gặp lỗi build.

Tối ưu hóa quy trình làm việc của bạn

4.



Dùng Hàng đợi Tác vụ (Use the Task Queue)

Gửi các tác vụ để ghi lại ý tưởng bất chợt, công việc còn dang dở, hoặc các bản sửa lỗi nhỏ. Hàng đợi này hoạt động như một khu vực chờ để bạn quay lại xử lý khi đã tập trung.

5.



Cung cấp Ngữ cảnh Bền vững với AGENTS.md

Duy trì một file AGENTS.md trong repo để 'dạy' Codex về các quy ước đặt tên, logic nghiệp vụ, hoặc các đặc thù mà nó không thể tự suy ra từ code.

6.



Tận dụng 'Best of N'

Tính năng này cho phép tạo ra nhiều phương án cho cùng một tác vụ. Điều này giúp bạn nhanh chóng khám phá nhiều giải pháp và chọn ra phương án tối ưu nhất.

Đây Mới Chỉ Là Sự Khởi Đầu

Codex vẫn đang trong giai đoạn nghiên cứu, nhưng nó đã tạo ra tác động thực sự đến cách chúng tôi xây dựng sản phẩm—giúp chúng tôi đi nhanh hơn, viết code tốt hơn và giải quyết những công việc mà trước đây có thể đã không bao giờ được ưu tiên.

‘...chúng tôi mong đợi sẽ mở khóa những cách thức phát triển phần mềm thậm chí còn mạnh mẽ hơn.’

Biến Mã Nguồn Thành Đối Tác Sáng Tạo Của Bạn.

