

# 最新のAI駆動開発と 開発者に必要なスキルセット

# 本日のアジェンダ

1. あなたの『脳』は大丈夫ですか？
2. 生成AIの劇的な進化
3. バイブコーディングによるプログラム生成
  1. プロンプトエンジニアリング
  2. 最近のAI開発エージェント
4. エージェントイックAIツールによるAI駆動開発へ
  1. MCPで多様な連携を促進
  2. プロジェクトの長時間自律走行
  3. 数人によるアジャイル開発を置き換え
5. これからのために リスキングの暫定的な勘所
  1. 批判的思考力と適切なAI出力評価
  2. オーケストレーション能力
  3. コンテキストエンジニアリング

# 1. あなたの『脳』は大丈夫ですか？

## The Memory Paradox: Why Our Brains Need Knowledge in an Age of AI

Barbara Oakley,<sup>1</sup> Michael Johnston,<sup>2</sup> Ken-Zen Chen,<sup>3</sup> Eulho Jung,<sup>4</sup> Terrence Sejnowski<sup>5</sup>

**Abstract:** In an era of generative AI and ubiquitous digital tools, human memory faces a paradox: the more we offload knowledge to external aids, the less we exercise and develop our own cognitive capacities. This chapter offers the first neuroscience-based explanation for the observed reversal of the Flynn Effect—the recent decline in IQ scores in developed countries—linking this downturn to shifts in educational practices and the rise of cognitive offloading via AI and digital tools. Drawing on insights from neuroscience, cognitive psychology, and learning theory, we explain how underuse of the brain’s declarative and procedural memory systems undermines reasoning, impedes learning, and diminishes productivity. We critique contemporary pedagogical models that downplay memorization and basic knowledge, showing how these trends erode long-term fluency and mental flexibility. Finally, we outline policy implications for education, workforce development, and the responsible integration of AI, advocating strategies that harness technology as a complement to – rather than a replacement for – robust human knowledge.

**Keywords:** cognitive offloading, memory, neuroscience of learning, declarative memory, procedural memory, generative AI, Flynn Effect, education reform, schemata, digital tools, cognitive load, cognitive architecture, reinforcement learning, basal ganglia, working memory, retrieval practice, schema theory

### 1. Introduction

Educators in recent decades have often championed “learning how to learn” and critical thinking

バーバラオークレー博士の最新の論文  
<https://arxiv.org/abs/2506.11015>

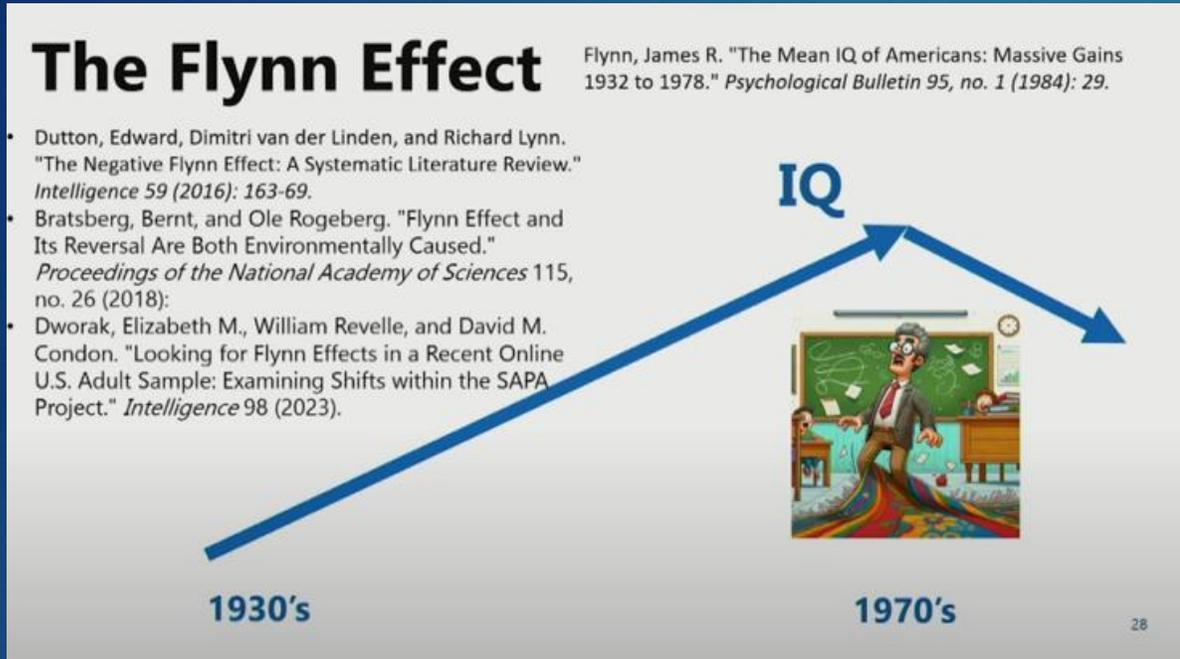


去年8月の講義

<https://www.youtube.com/watch?v=WUFAaqvc3us&t=1919s>

$$5 \times 10 = 500$$

あなたはこれを見て、どう感じますか？



- ▶ 人間のIQは教育制度が整備され、学習することによって年々向上してきました
- ▶ しかし、1970年代あたりから先進国ではIQが下がり始めていました

## ▶ 原因は何でしょうか？

- ▶ 先進各国が『暗記重視』から『考えることを重視』する教育方針にしたからと言われています
- ▶ 加えて電卓が登場し…計算に関して言えば、暗記も思考も不要になってしまったこともあります

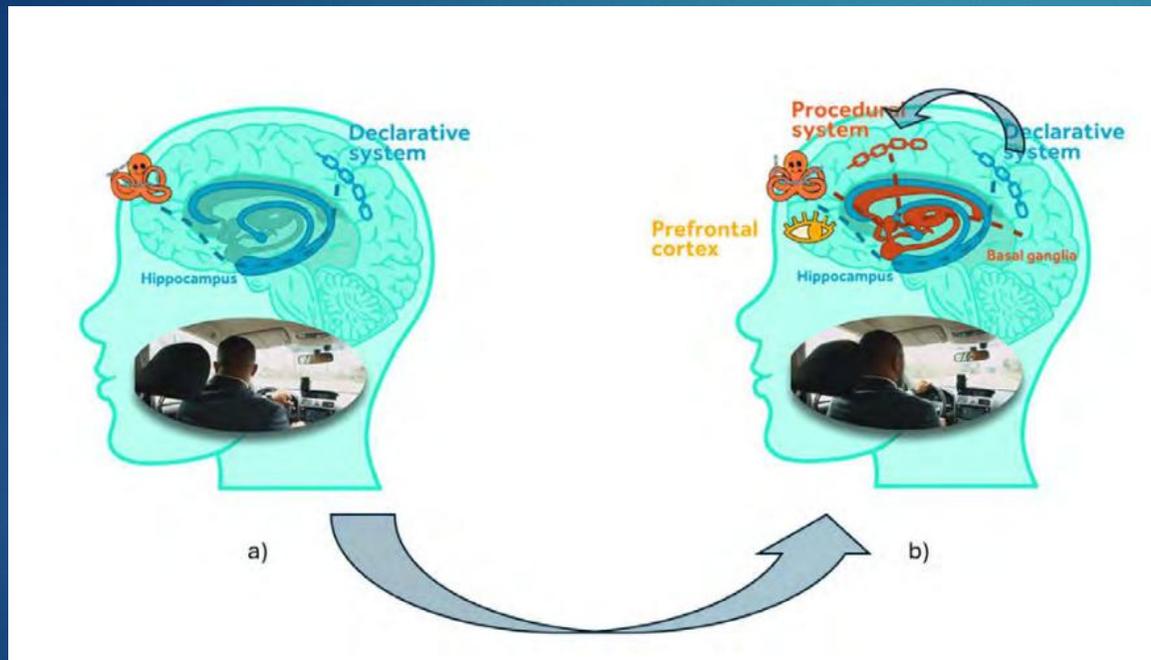
▶ 『Google効果』ってご存じですか？

- ▶ インターネット検索で容易に得られた情報に対しては、記憶せずに忘れてしまう傾向のこと
- ▶ 必要な時にいつでも検索できるために、脳が記憶する必要がないと認識してしまいます
- ▶ 別名：デジタル健忘症
  - ▶ 調べものをした内容をすぐに忘れてしまいがちです
  - ▶ 記憶しようとする意欲が低下していきます

▶ 生成AIでも似たようなことが起こると思いませんか？

- ▶ そうならないようにするためには、どうしたら良いか考えてみましょう

- ▶ 生成AIのお話の前に、脳科学/認知科学での学習と記憶の話です
  - ▶ 会社への車通勤のルートをどう学習し記憶するのかを例にしてみます。

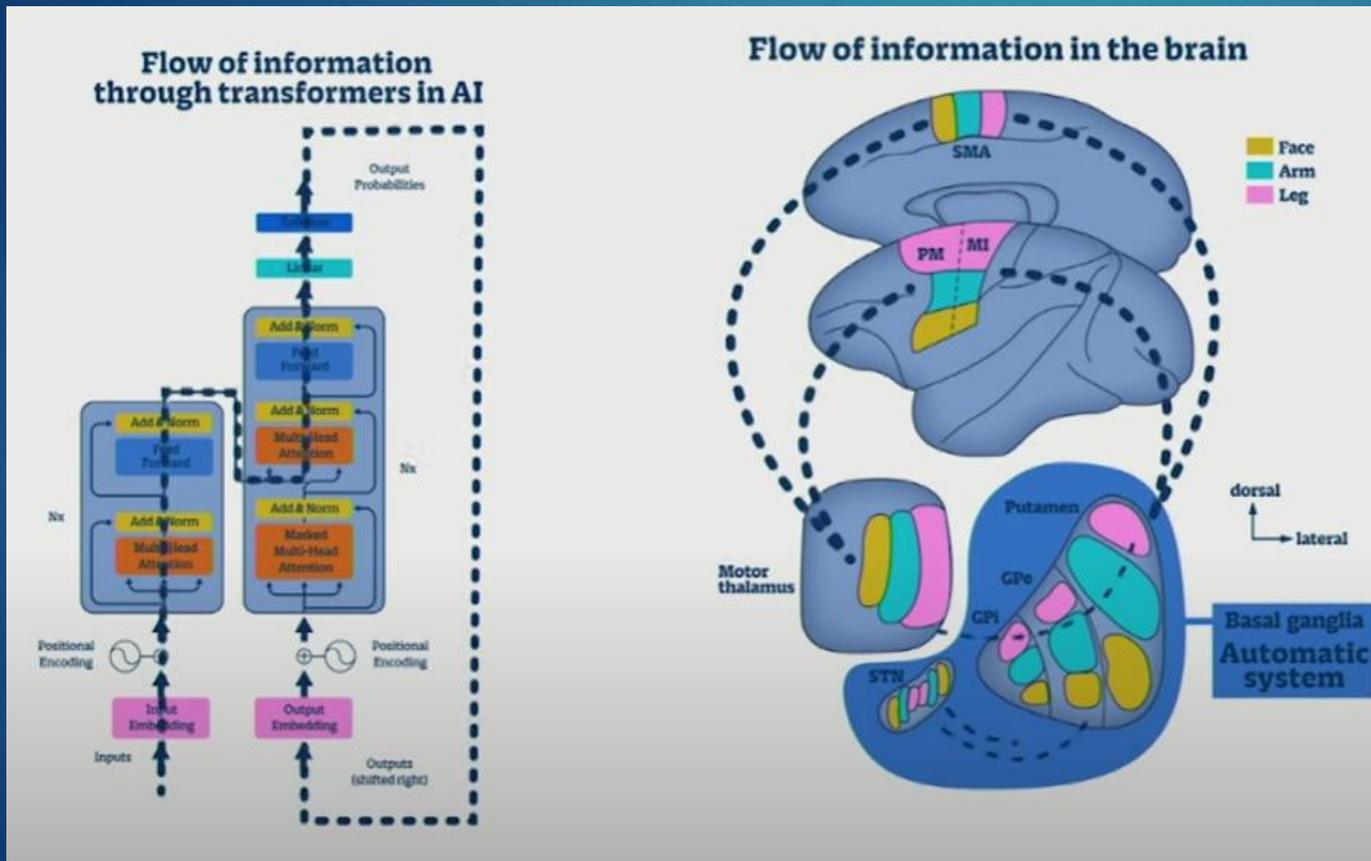


a) 初めてのルートを使う

b) 同じルートを繰り返し使う

- ▶ 宣言的回路(ゆっくり思い出す：知っている)
  - ▶ 初めてのルートを使うとき、前頭前野から海馬を経由して長期記憶領域に記録します
- ▶ 手続的回路(瞬時に思い出す：やれる)
  - ▶ 繰り返しルートを使うと、前頭前野の監視機構が海馬経由の記憶を大脳基底核を経由して長期記憶に移します
- ▶ 反復学習は正確で素早い判断につながります

- ▶ 生成AIのLLMは、脳内の手続的回路と似ています
- ▶ Transformerは、手続的回路から記憶を蘇らせる流れと似ています



- ▶ 『認知オフローディング』という言葉をご存じですか？
  - ▶ 『脳の外部委託』とも言われています
    - ▶ 脳内の作業を外部の道具などに委ねることを指します
      - ▶ メモ帳を使う ⇒ 電卓を使う ⇒ スマホのアプリを使う…
      - ▶ そして生成AIを使う…
  - ▶ 脳内作業をより高度で複雑な課題や判断に充てることが可能になります
  - ▶ しかし … 便利になり … 楽になり … そして怠け者になる可能性もあります
- 
- ▶ 普段から生成AIを壁打ちのパートナーにして自分の脳を鍛えましょう
  - ▶ パートナーは複数いたほうが心強いです

## 2. 生成AIの劇的な進化

## ▶ 『AIの個人利用が転換点を迎えた』 (by MENLO VENTURES)



### AI's Consumer Tipping Point Has Arrived

More than half of American adults (**61%**) have used AI in the past six months, and nearly **one in five** rely on it every day. Scaled globally, that translates to **1.7–1.8 billion people** who have used AI tools, with **500–600 million** engaging daily.<sup>1</sup> This is no longer experimentation; it's habit formation at an unprecedented scale.

Beneath the hype, real usage patterns are more nuanced than expected. They expose hidden gaps and huge opportunities for founders building the next generation of consumer AI products. In this report, based on a survey of over **5,000 U.S. adults**,<sup>2</sup> we unpack what's fueling this adoption, where value is leaking, and how the smartest companies can capture it—revealing the driving forces behind the consumer AI revolution.

#### AI Adoption Among U.S. Adults

**61%**

of U.S. adults have used AI in the past six months

**19%**

of U.S. adults interact with AI daily

### Five Categories of Daily Life

Category	Activities
<b>Routine Tasks</b> 	Personal research <ul style="list-style-type: none"> <li>• Researching purchases</li> <li>• Researching real estate</li> <li>• Personal finance</li> </ul>
	Day-to-day planning and productivity <ul style="list-style-type: none"> <li>• Writing emails</li> <li>• Managing to-do lists</li> <li>• Meal planning</li> <li>• Travel planning</li> </ul>
	Home management <ul style="list-style-type: none"> <li>• Home repairs + maintenance</li> <li>• Managing expenses</li> <li>• Managing childcare</li> </ul>
<b>Learning + Development</b> 	Personal improvement and learning <ul style="list-style-type: none"> <li>• Learning new skills</li> <li>• Researching topics of interest</li> <li>• Practicing a new language</li> <li>• Practicing a new physical activity</li> <li>• Summarizing reading materials</li> <li>• Taking and organizing notes</li> <li>• Personal coding projects</li> </ul>
	Work and school <ul style="list-style-type: none"> <li>• Research for school or work</li> <li>• Getting help on assignments</li> <li>• Coding projects for work or school</li> </ul>

Category	Activities
<b>Physical + Mental Health</b> 	Research and planning <ul style="list-style-type: none"> <li>• Researching health questions</li> <li>• Planning workouts</li> </ul>
	Health management <ul style="list-style-type: none"> <li>• Tracking health and fitness data</li> <li>• Tracking nutrition</li> <li>• Navigating healthcare</li> </ul>
	Mental health <ul style="list-style-type: none"> <li>• Mental health and wellness support</li> </ul>
<b>Connection</b> 	Virtual connection <ul style="list-style-type: none"> <li>• Making new virtual connections</li> <li>• Staying in touch</li> </ul>
	Real-life connection <ul style="list-style-type: none"> <li>• Making new in-person connections</li> <li>• Dating or social coaching</li> </ul>
	Self-connection <ul style="list-style-type: none"> <li>• Exploring spirituality</li> </ul>
<b>Creative Expression</b> 	Creation <ul style="list-style-type: none"> <li>• Creating music or audio</li> <li>• Making videos</li> <li>• Creating images</li> <li>• Creating presentations</li> </ul>
	Editing <ul style="list-style-type: none"> <li>• Editing images and video</li> <li>• Writing support</li> </ul>

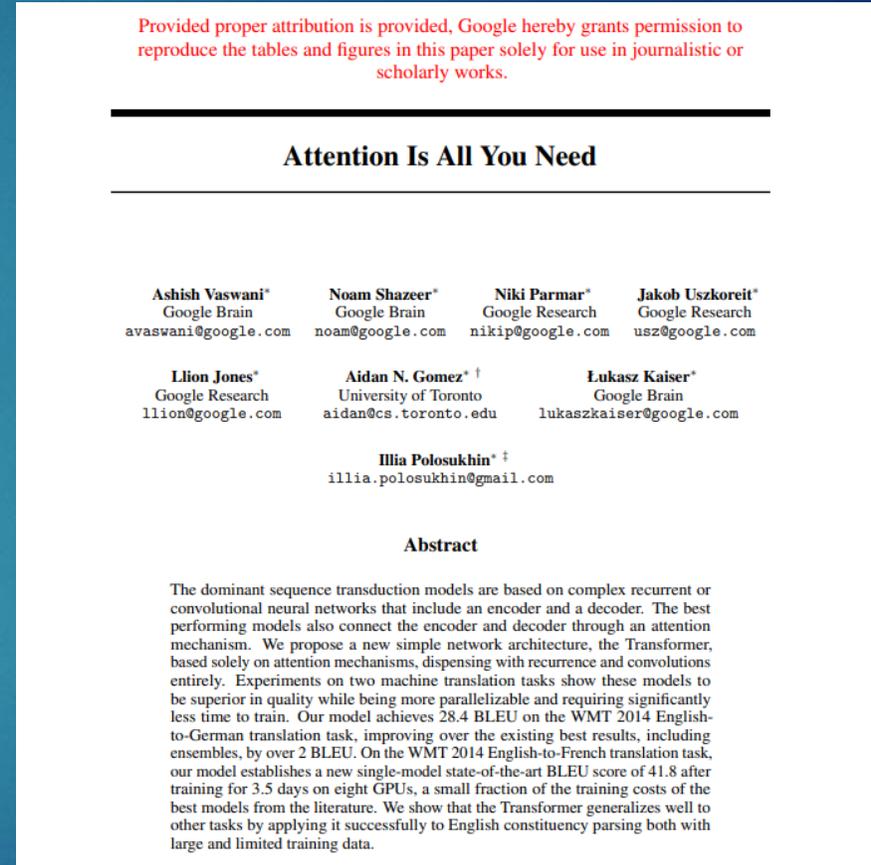
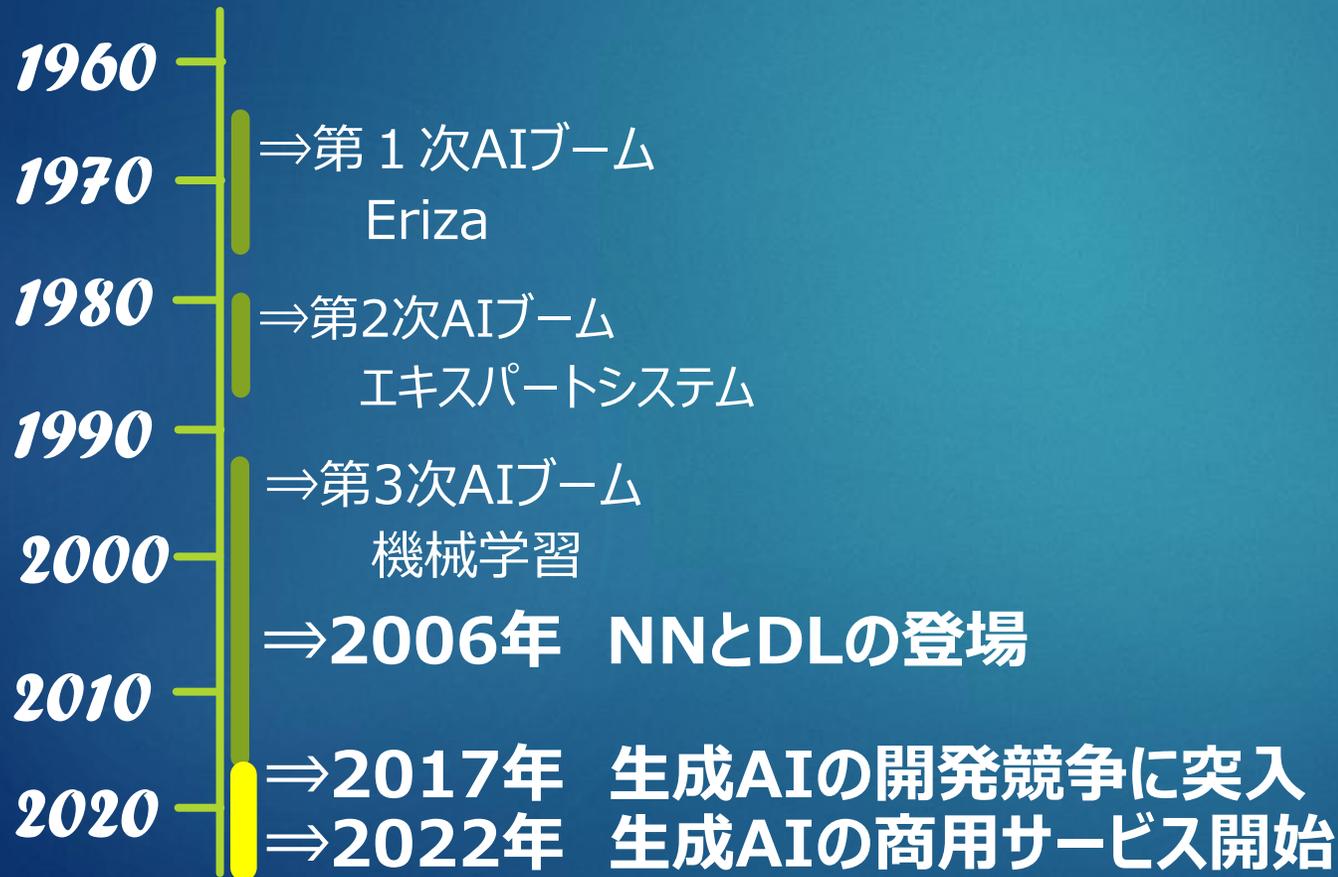
© 2025 Menlo Ventures

<https://menlovc.com/perspective/2025-the-state-of-consumer-ai/>

- ▶ 米国での最新の調査で、成人の61%がAIを使ったことがあり、そのうちの5人に1人が毎日使っているとのことでした
- ▶ 日常生活を構成する5つの活動カテゴリーで使われています
  - ▶ ルーチン業務 (Routine Tasks)
    - ▶ スケジュール調整、買い物調査、書類/メール作成、家計管理/支払い管理、育児、家のメンテナンス…
    - ▶ 一声でワークフロー完遂か、根こそぎ自動化の方向に進んでいます…
  - ▶ 学習と自己成長 (Learning & Development)
    - ▶ 新スキルの習得、調査と要約、コーディングの学習…
    - ▶ 何らかの学習をする人は77%ですが、そのうちAIを使うのは18%にとどまります…
    - ▶ AI駆動開発ツールは今年に入って急増中、ツールも進化中…
    - ▶ AI出力を教材で使うための教師向けツールなどが登場予定…

- ▶ 日常生活を構成する5つの活動カテゴリー… その2
  - ▶ 身体的・精神的健康 (Physical & Mental Health)
    - ▶ ワークアウト作成、進捗提案、栄養提案…
    - ▶ スマートデバイス等による睡眠管理やバイタル管理とアラートなど…
    - ▶ 健康情報を調べる人は71%ですが、AIを使う人は20%です
    - ▶ 保険や医師との連携によるハイブリッドAIケアは現時点では白紙です
  - ▶ つながり (Connection)
    - ▶ 友人や家族と交流する人は76%ですが、AIを使う人は14%のみです…
    - ▶ Snapchat の My AIが利用者1.5億人を突破しました
    - ▶ マッチング、イベント企画など『人間関係を増幅させるAI』はまだ白紙です
  - ▶ 創造的表現 (Creative Expression)
    - ▶ 生画像・動画・音楽・文章など「創る楽しみ」をAIが民主化しました
    - ▶ ツール各社は透明性確保とパートナーシップ強化を急いでいます

▶ Googleの論文『Attention is All You Need』が全ての始まり



<https://arxiv.org/abs/1706.03762>

## ▶ NN(ニューラルネットワーク)がRNNとCNNに派生した背景

### 視点

入力データの構造

### 背景・課題

汎用の全結合 NN (MLP) は入力を一次元ベクトルとしてしか扱えず、空間的・時間的な構造を無視していた

パラメータ効率

画像  $256 \times 256 \times 3$  を全結合層で扱おうと数千万~数億パラメータが必要

不変性の獲得

画像分類では 平行移動、時系列では 時間シフト といった同形操作に頑健であることが重要

勾配伝播の困難さ

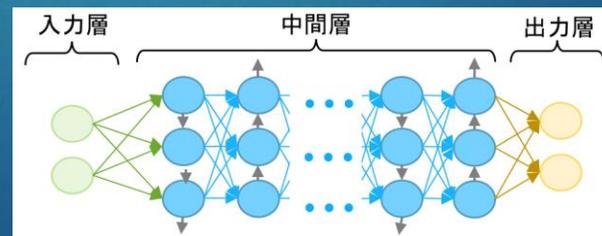
深い MLP は勾配消失/爆発が深刻化しやすい

### RNN が採用したアプローチ

系列データを時刻  $t$  の状態と次時刻  $t+1$  の状態へ“再帰”させることで時間構造を保持

RNN (特に LSTM/GRU) は時系列中のタイミングのずれに頑健

RNN はゲート機構 (LSTM/GRU) で長距離依存の勾配を保持



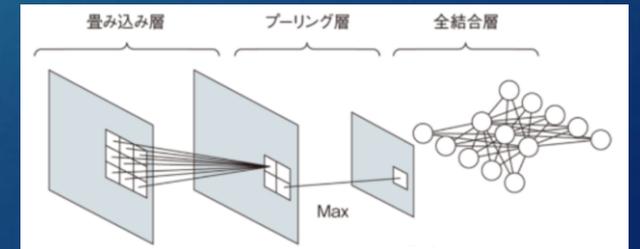
### CNNが採用したアプローチ

画像の 2D 格子構造をそのまま活かすため、局所受容野と重み共有 (畳み込み) を導入

CNN の 重み共有 (同じフィルタを全領域でスライド) により、桁違いに少ないパラメータで高表現力を維持

CNN のプーリングやストライドは位置ずれに対するロバスト性を獲得

CNN は層あたりのパラメータが少なく勾配が安定しやすい



## ▶ Transformerが生まれた背景：CNN/RNNの欠点の解決

解決対象	従来モデルの課題	Transformerのアプローチ	どう改善したか
RNN 系列モデル	① 時間順にしか計算できず並列化が困難 ② 長距離依存の学習が苦手 ③ 勾配消失／爆発	■ 再帰を捨て Self-Attention のみで系列全体を一括処理 ■ 各トークンがシーケンス内の全トークンと直接“注意”可能 ■ Attention 経路は加算・正規化中心で安定	入力トークン同士の計算を同時に行えるため、学習・推論を大幅に高速化 数千～数万トークンまで安定に依存関係を捉えられる
CNN 画像モデル	④ 受容野が局所に限定 ⑤ カーネル・プーリング	■ Global Self-Attention により初層から全パッチ間の関係をモデリング ■ 同一メカニズムでサイズ・スケールに柔軟パラメータはヘッド数や次元設定だけ	深層でも勾配が一定幅で流れ、学習が崩壊しにくい 画像全体の大域的文脈を浅い層で把握、物体間関係を捉えやすい
共通課題	⑥ タスクごとに別アーキテクチャが必要	■ Attention + 位置埋め込み という汎用構成	画像でもテキストでも同一アルゴリズムを再利用でき、実装保守が容易 テキストや画像などを単一フレームワークで扱える“統一プラットフォーム”を実現

## ▶ 生成AIの進化

- ▶ スケーリング競争
- ▶ マルチモーダル化
- ▶ AIEージェントの登場
- ▶ 汎用AIと特化型AI
- ▶ ベンチマークの進化
- ▶ リーズニングモデル/シンキングモデル登場 (LLM/LRM)
- ▶ MCP登場と普及
- ▶ エージェント化/自律化

## ▶ 課題もたくさん

- ▶ AIアライメントとコーパス選定
- ▶ 効率的な学習方法
- ▶ 蒸留化やファインチューニング
- ▶ メモリ効率とコンテキスト長拡大とロングラン化

### 3. バイブコーディングによるプログラム生成

- ▶ バイブコーディングとは、生成AIに対して日常使う言葉(自然言語)で作ってほしい機能を伝えると、プログラムコードを生成してくれることです
    - ▶ 今年の初めにアンドレア カルパシー氏が、これからのAI駆動開発でコード生成する部分をたどって使った言葉です
    - ▶ 瞬く間に広がって…中には『素人でもプログラムを作れる』とか、『プログラム開発を生成AIに丸投げできるようになった』とか、『ソフトウェア開発者は失業する』とか… (笑)
- ▼
- ▶ もともと生成AIは、記憶したものを返すのは得意でしたが、数学の計算やアルゴリズムとかの論理的思考は苦手でした
  - ▶ 現在はぐんぐん賢くなっていて、経験1～3年のプログラマレベルの作業は任せてよいレベルになっています
  - ▶ 開発担当者は生成AIと協働作業(壁打ち)して、完成に持ち込む必要があります

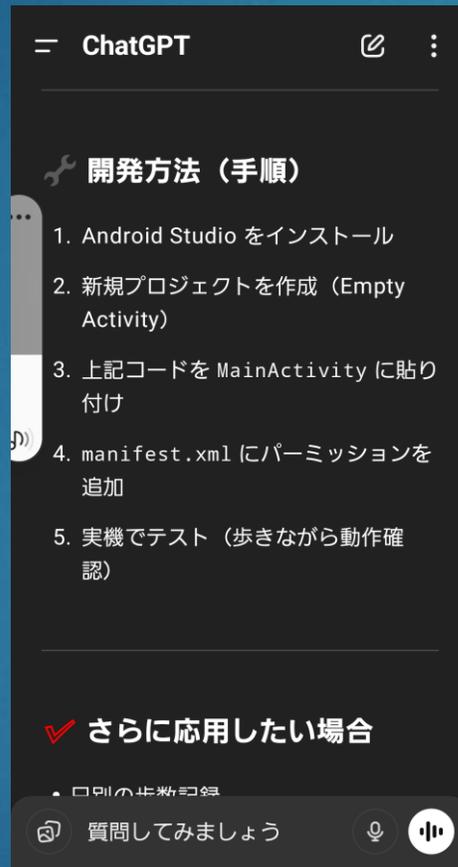
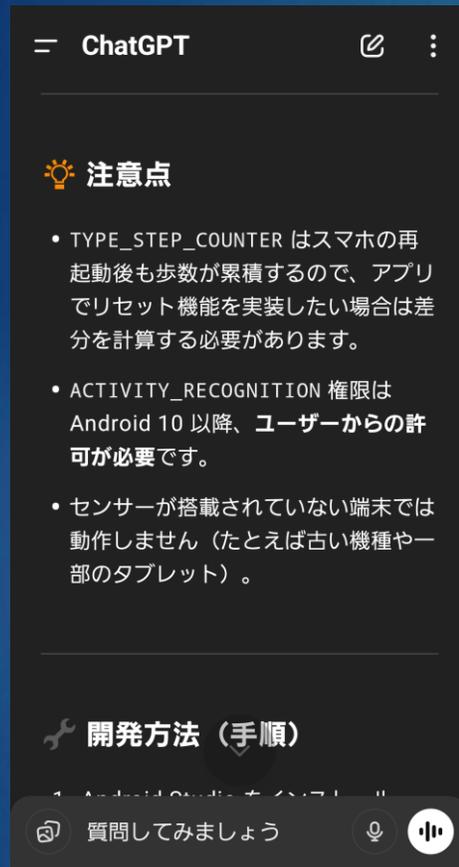
### 3. バイブコーディングによるプログラム生成 プロンプト エンジニアリング

- ▶ プロンプトに込める期待は人によって異なります
- ▶ 受け取ったプロンプトから期待出力を推測するのはLLMによって違います

## ■ ChatGPT o3 でのバイブコーディングの例 その1



## ■ ChatGPT o3 でのバイブコーディングの例 その2



# ■ Gemini 2.5 Pro でのバイブコーディングの例

スマホ歩数計作成方法の解説 2.5 Pro 和明

自分のスマホで動く歩数計を作ってください。

思考プロセスを表示

はい、承知いたしました。ご自身のスマートフォンで動く歩数計を作成する方法について、いくつかのレベルに分けてご説明します。ご自身の目的やプログラミング経験に合わせて、最適な方法を選んでみてください。

**レベル1：一番簡単な方法（アプリや既存機能の利用）**

実は、ほとんどのスマートフォンには、特

Gemini に相談

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

スマホ歩数計作成方法の解説 2.5 Pro 和明

**レベル2：少し挑戦したい方向け（ノーコード/ローコード開発）**

「既存のアプリではなく、自分で作ったもので歩数を測りたい！」でも「本格的なプログラミングは難しそう...」という方には、ブロックを組み合わせてアプリを作れるツールがおすすめです。

おすすめのツール：MIT App Inventor  
マサチューセッツ工科大学（MIT）が提供している、無料で使える教育用のアプリ開発環境です。Webブラウザ上で、パズルのようにブロックを組み合わせるだけで、Androidアプリを作成できます。

歩数計の基本的な仕組み：  
スマートフォンに内蔵されている「加速度

Gemini に相談

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

スマホ歩数計作成方法の解説 2.5 Pro 和明

**レベル3：本格的に開発したい方向け（プログラミング）**

ご自身でコードを書いて、App StoreやGoogle Playで公開するような本格的なアプリを作りたい場合は、以下の方法になります。

1. Androidアプリ開発（使用言語：KotlinまたはJava）  
Android Studioという開発環境を使います。スマートフォンのハードウェア機能にアクセスして歩数を取得します。

- 使用する主なAPI：
  - SensorManager: デバイスのセンサーを管理します。
  - Sensor.TYPE\_STEP\_COUNTER:

Gemini に相談

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

スマホ歩数計作成方法の解説 2.5 Pro 和明

Kotlinでの簡単なコード例：

```
kotlin
// センサーマネージャーの取得
private lateinit var
sensorManager: SensorManager
private var stepCounterSensor:
Sensor? = null

// ... onCreate内などで初期化 ...
sensorManager =
getSystemService(Context.SENSOR_SERVICE) as SensorManager
stepCounterSensor =
sensorManager.getDefaultSensor(
Sensor.TYPE_STEP_COUNTER)

// センサーのリスナー登録
override fun onResume() {
```

Gemini に相談

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

スマホ歩数計作成方法の解説 2.5 Pro 和明

**まとめ**

どの方法で「自分の歩数計」を作りたいか、イメージは湧きましたでしょうか？

- 手軽に日々の歩数を知りたい → レベル1
- プログラミングの初歩を学びながら、自分だけのアプリを作りたい → レベル2
- 本格的なアプリ開発に挑戦し、スキルを身につけたい → レベル3

ぜひ、ご自身の目的に合った方法で、歩数計作りを楽しんでみてください。

Gemini に相談

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

# ■ Claude Sonnet 4 でのバイブコーディングの例 その1

← Claude Sonnet 4

Kazuaki 1 Artifacts

自分のスマホで動く歩数計を作ってください。

お疲れ様です！スマートフォンで動く歩数計アプリを作成しますね。

具体的な要件について少しお聞きします：

1. どのような機能が必要ですか？（基本的な歩数カウント、目標設定、履歴表示など）
2. デザインの好みはありますか？（シンプル、カラフル、ダークモードなど）
3. 歩数以外に表示したい情報はありますか？（距離、カロリー、時間など）

これらを教えていただければ、より使いやすい歩数計を作ります！

Reply to Claude...

+ Research

← Claude Sonnet 4

Kazuaki 1 Artifacts

全てお任せします

スマートフォンで動く歩数計アプリを作成しますね！加速度センサーを使用して歩数をカウントし、目標設定や履歴表示機能も含めた本格的なアプリにします。

スマートフォン歩数計アプリ  
Interactive artifact

スマートフォン用の歩数計アプリを作成しました！以下の機能を搭載しています：

主要機能：

- 加速度センサーを使用した本格的な歩数カウント
- 目標設定機能 (1,000～

Reply to Claude...

+ Research

← Claude Sonnet 4

主要機能： 1 Artifacts

- 加速度センサーを使用した本格的な歩数カウント
- 目標設定機能 (1,000～50,000歩)
- リアルタイム進捗表示（円形プログレスバー）
- 距離・カロリー計算（歩数から自動算出）
- 達成マイルストーン（1,000歩、5,000歩、10,000歩など）
- 履歴機能（過去10日分の記録）

使い方：

1. 「開始」ボタンを押して計測開始
2. スマホを持って歩くと自動的に歩数がカウントされま
3. 目標値は自由に変更可能

Reply to Claude...

+ Research

← Claude Sonnet 4

使い方： 1 Artifacts

1. 「開始」ボタンを押して計測開始
2. スマホを持って歩くと自動的に歩数がカウントされます
3. 目標値は自由に変更可能
4. 履歴は自動保存されます

技術的特徴：

- iOS/Androidの加速度センサーに対応
- センサー非対応デバイスではシミュレーションモード
- レスポンシブデザインで様々な画面サイズに対応
- モダンなグラスモーフィズムデザイン

実際のスマートフォンでご利用いただくと、歩行時の加速度変化を検知して正確に歩数をカウントします！

Reply to Claude...

+ Research

# ■ Claude Sonnet 4 でのバイブコーディングの例 その2

```

X 歩数計アプリ - ソースコード解説
JavaScript コア機能
1. 初期化とセンサー許可

let steps = 0;
let goal = 10000;
let isTracking = false;
let lastAcceleration = { x: 0
let stepThreshold = 1.2;
let stepCooldown = false;

// センサーの許可を確認 (iOS 13+対
async function checkPermissio
  if ('DeviceMotionEvent' i
    if (typeof DeviceMoti
      const permission
      if (permission !=
        showStatus('て
  }
    
```

```

X 歩数計アプリ - ソースコード解説
2. 歩数検出アルゴリズム

// デバイスモーション処理 (歩数検出
function handleMotion(event)
  if (!isTracking) return;

  const acceleration = even
  if (!acceleration) return

  const currentAcceleration
    x: acceleration.x ||
    y: acceleration.y ||
    z: acceleration.z ||
  };

  // 加速度の変化量を計算 (3軸合
  const deltaX = Math.abs(c
  const deltaY = Math.abs(c
    
```

```

X 歩数計アプリ - ソースコード解説
3. 計測開始・停止

function startTracking() {
  if (!isTracking) {
    isTracking = true;
    startTime = new Date(

    // ボタン状態変更
    document.getElementById('
    document.getElementById('

    if (window.DeviceMoti
      // 実際のセンサー使用
      window.addEventListener(
        showStatus('歩数計
    } else {
      // シミュレーションモ
      startSimulation()
    }
  }
    
```

```

X 歩数計アプリ - ソースコード解説
4. 表示更新とプログレス計算

function updateDisplay() {
  // 歩数表示
  document.getElementById('

  // 進捗率計算 (円形プログレス用
  const progress = Math.min
  document.getElementById('
  document.getElementById('

  // 距離計算 (平均歩幅0.7m)
  const distance = (steps *
  document.getElementById('

  // カロリー計算 (1歩あたり0.04
  const calories = Math.rou
  document.getElementById('
    
```

# ■ Claude Sonnet 4 でのバイブコーディングの例 その3

× 歩数計アプリ - ソースコード解説

### 5. 達成チェックとマイルストーン

```
function checkAchievements()
  const milestones = [1000,
  milestones.forEach(milesto
    if (steps === milesto
      showAchievement(`
    }
  });

  if (steps === goal) {
    showAchievement(`🏆 🎉
  }
}

function showAchievement(mess
  const achievementDiv = do
```

Reply to Claude...

+ 🗨️ 🔍 Research 🎤 🔊

× 歩数計アプリ - ソースコード解説

### 6. 履歴機能 (LocalStorage使用)

```
// 履歴保存
function saveToHistory() {
  if (steps > 0) {
    const today = new Dat
    const existingIndex =

    if (existingIndex >=
      // 同日の記録は最大値
      history[existingI
    ] else {
      history.unshift({
    }

    // 最新10件のみ保持
    history = history.sli
    localStorage.setItem(
```

Reply to Claude...

+ 🗨️ 🔍 Research 🎤 🔊

× 歩数計アプリ - ソースコード解説

### 技術ポイント

#### 歩数検出アルゴリズム

- 3軸加速度センサーを使用
- 前回との差分計算で歩行を検出
- **\*\*閾値 (1.2) \*\***とクールダウン (300ms) で誤検出を防止
- ベクトル合成で方向に依存しない検出

#### レスポンスデザイン

- CSS GridとFlexboxでレイアウト
- グラスモーフィズム効果でモダンなUI
- CSS Custom Propertiesでプログレス表示

#### センサー対応

- **\*\*iOS 13+\*\***のpermission対応
- センサー非対応時のシミュレーション

Reply to Claude...

+ 🗨️ 🔍 Research 🎤 🔊

× スマートフォン歩数計アプリ

2025年6月30日 月曜日

0 歩

Console Messages

Uncaught Error: Uncaught  
TypeError: Cannot read

Reply to Claude...

+ 🗨️ 🔍 Research 🎤 🔊

- ▶ 非常に単純なプロンプト『自分のスマホで動く歩数計を作ってください』で、皆さんはどの生成AIが、バイブコーディング向いていると感じましたか？
  - ▶ 一般的なプロンプトエンジニアリングの要点はいろいろとされていますが…
    - ▶ 段階的な思考を促す
    - ▶ ロールを明確にする
    - ▶ 明確で具体的な指示をする
    - ▶ コンテキストを含める
- 
- ▶ AI開発エージェントは、その開発プロジェクトの理解力や、支援したり自律的に実行可能な開発プロセス開発フローの深度とカバー範囲が異なります
  - ▶ 最新のバイブコーディングやAI駆動開発でのプロンプトの書き方は、それぞれのAIに聞いてください

## ■ Claude Sonnet 4

- ▶ プロンプト『Cursor + Claude Sonnet 4 でバイブコーディングする際のプロンプトに含める内容の要点を教えてください。』

Cursor と Claude Sonnet 4 のペアプログラミング ~

Cursor + Claude Sonnet 4 でバイブコーディングする際のプロンプトに含める内容の要点を教えてください。

Cursor + Claude Sonnet 4でのペアプログラミングを効果的に行うためのプロンプトの要点をお伝えします。

### コンテキスト設定

- プロジェクトの概要：何を作っているか、技術スタック、アーキテクチャの説明
- 現在のファイル構造：関連するファイルの場所と役割
- 開発環境：使用しているフレームワーク、ライブラリ、設定

### 具体的な要求

- 明確な目標：「○○機能を実装したい」「△△のバグを修正したい」
- 制約条件：パフォーマンス要件、既存コードとの互換性、コーディング規約
- 期待する出力形式：完全なコード、差分のみ、説明付きなど

### コード品質の指針

- 命名規則：変数名、関数名、クラス名の規則
- エラーハンドリング：例外処理の方針
- テスト：単体テストの有無、テストフレームワーク
- ドキュメント：コメントの書き方、JSDoc等の使用

### 協働のための情報

- 段階的な実装：「まず基本機能から」「次にエラーハンドリング」など
- レビューポイント：特に注意してほしい箇所
- 学習目的：理解したい概念があれば明記

Claudeに返信...

Claude Sonnet 4

### 効果的なプロンプト例

```
# プロジェクト: React + TypeScriptのTODOアプリ
# 目標: タスクの優先度設定機能を追加
```

現在のTask型にpriorityフィールドを追加し、TaskList コンポーネントで優先度による並び替えを実装してください。

制約:

- 既存のAPIとの互換性を保つ
- 優先度は1-5の数値 (1が最高優先度)
- TypeScriptの型安全性を維持

期待する出力:

1. 型定義の更新
2. コンポーネントの修正
3. 簡単な説明とテストの提案

このような構造化されたプロンプトを使うことで、Claudeとより効率的にペアプログラミングができるようになります。

Claudeは間違えることがあります。回答内容を必ず確認ください。

Claudeに返信...

Claude Sonnet 4

## ■ Claude Opus 4

- ▶ プロンプト『ClaudeCode でAI駆動開発する際のプロンプトに含める内容の要点を教えてください。』

AI駆動開発のプロンプト戦略

ClaudeCodeでAI駆動開発する際のプロンプトに含める内容の要点を教えてください。

Claude Codeで効率的にAI駆動開発を行うためのプロンプトには、以下の要点を含めることが重要です：

### プロジェクトの文脈と目的

- 明確な目標定義: 作成したいアプリケーションやツールの具体的な目的と期待される成果
- 技術スタック: 使用する言語、フレームワーク、ライブラリの指定
- 制約条件: パフォーマンス要件、セキュリティ要件、互換性の考慮事項

### 詳細な仕様と要件

- 機能要件: 実装すべき具体的な機能のリスト（優先順位付き）
- ユーザーインターフェース: UIの構造、デザイン要件、ユーザーフローの説明
- データ構造: 扱うデータの形式、スキーマ、関係性の定義

### 実装の指針

- コーディング規約: 命名規則、コメントスタイル、フォーマットの好み
- アーキテクチャパターン: MVC、マイクロサービス、モジュラー設計など
- エラーハンドリング: 例外処理やバリデーションの方針

### 段階的な開発アプローチ

- イテレーション計画: 最小動作プロダクト（MVP）から始めて段階的に機能を追加
- テスト戦略: ユニットテスト、統合テストの実装方針
- デバッグ情報: ログ出力やデバッグモードの実装

### 具体例とサンプル

Claudeに返信...

+

🔍 リサーチ

Claude Opus 4

### 具体例とサンプル

- 入出力例: 期待される動作の具体的な例
- エッジケース: 特殊な状況での挙動の定義
- 参考実装: 類似プロジェクトやコードスニペットへの言及

これらの要点を含めることで、Claude Codeはより正確で実用的なコードを生成し、手戻りを最小限に抑えながら効率的な開発が可能になります。特に重要なのは、曖昧さを排除し、具体的に測定可能な要件を提示することです。

🌟

再試行

Claudeは間違えることがあります。回答内容を必ずご確認ください。

Claudeに返信...

+

🔍 リサーチ

Claude Opus 4

## ■ Gemini 2.5 Pro

- ▶ プロンプト『Gemini CLI でAI駆動開発する際のプロンプトに含める内容の要点を教えてください。』

Gemini CLIを使ってAI駆動開発を効率的に進めるためには、プロンプトに以下の要点を盛り込むことが重要です。

**1. 明確な役割と目的の指定**

AIにどのような役割を期待しているのか、そして何を達成したいのかを具体的に伝えましょう。これにより、AIはより文脈に沿った的確な回答を生成しやすくなります。

- 悪い例:

```
このコードをレビューして。
```

- 良い例:

```
あなたは経験豊富なGoエンジニアです。以下のGoコードのパフォーマンスと可読性の観点からレビューし、改善点を指摘してください。
```

**2. 具体的な\*\*コンテキスト (文脈)\*\*の提供**

AIはあなたが提供する情報でしか判断できません。関連するコード、エラーメッセージ、ファイル構造、使用しているフレームワークやライブラリのバージョンなど、できるだけ多くのコンテキストを提供しましょう。

- ファイルの内容を含める:

Gemini に相談

+ 動画 Deep Research Canvas

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

**3. 出力形式の明確な指示**

どのような形式で回答が欲しいのかを具体的に指定することで、手戻りを減らし、そのまま利用できる回答を得やすくなります。

- コード生成の例:

```
gemini "Go言語で、標準入力から受け取った文字列をSHA256でハッシュ化し、16進数文字列として出力するコードを生成してください。"
```

- リスト形式での説明を求める例:

```
gemini "Goのgoroutineとchannelの主な違いを、箇条書きで簡潔に説明してください。"
```

- 特定のフォーマットを要求する例:

```
gemini "以下の関数のドキュメントを、GoDoc形式で生成してください。"
```

```
'''go
func CalculateTax(price int) int {
    // ...
}'''
```

Gemini に相談

+ 動画 Deep Research Canvas

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

```
go
func CalculateTax(price int) int {
    // ...
}
```

**4. 段階的な対話とフィードバック**

一度のプロンプトで完璧な答えを求めようとせず、AIと対話しながら段階的に深掘りしていくのが効果的です。最初の回答を基に、追加の質問や修正指示を与えて、望む結果に近づけていきましょう。

**1. 最初のプロンプト:**

```
gemini "ユーザー情報を管理するREST APIの基本的なエンドポイントをGoで実装してください。フレームワークは任意です。"
```

**2. フィードバックと追加要求:**

```
gemini "ありがとうございます。先ほどのコードに、PostgreSQLデータベースとの接続機能を追加してください。"
```

これらの要点を意識することで、Gemini CLIを強力な開発パートナーとして活用し、コーディング、デバッグ、ドキュメント作成などのタスクを大幅に効率化できます。

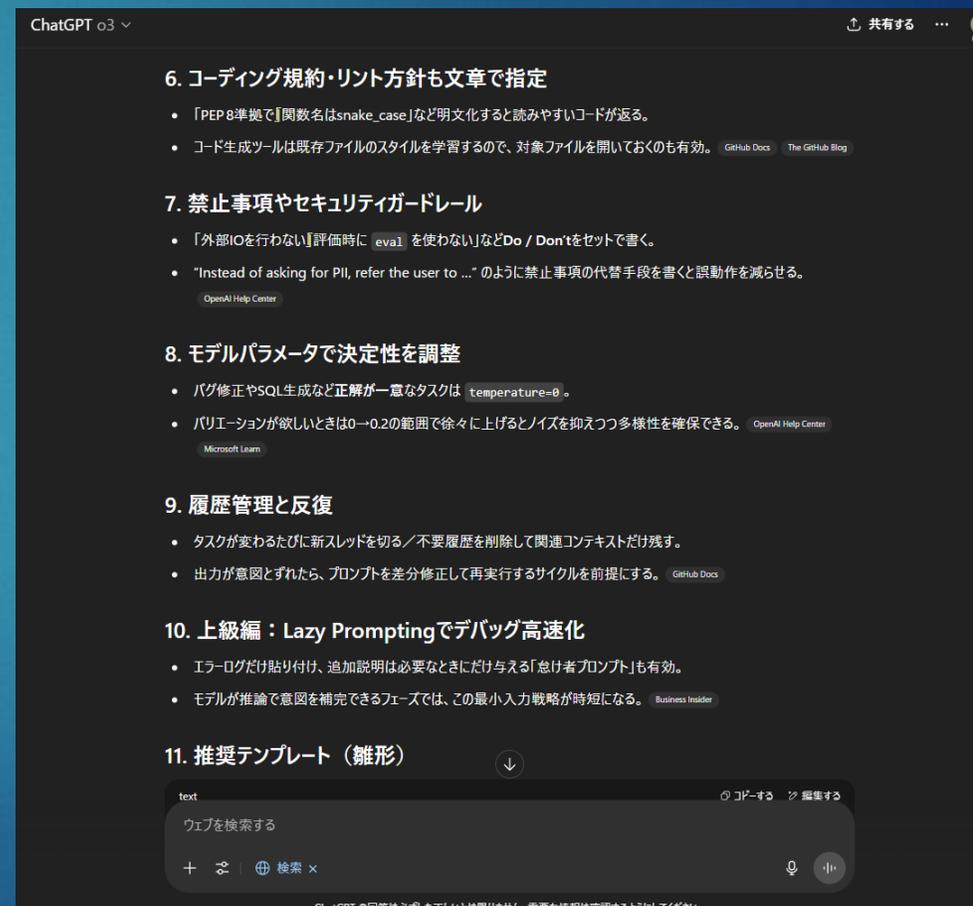
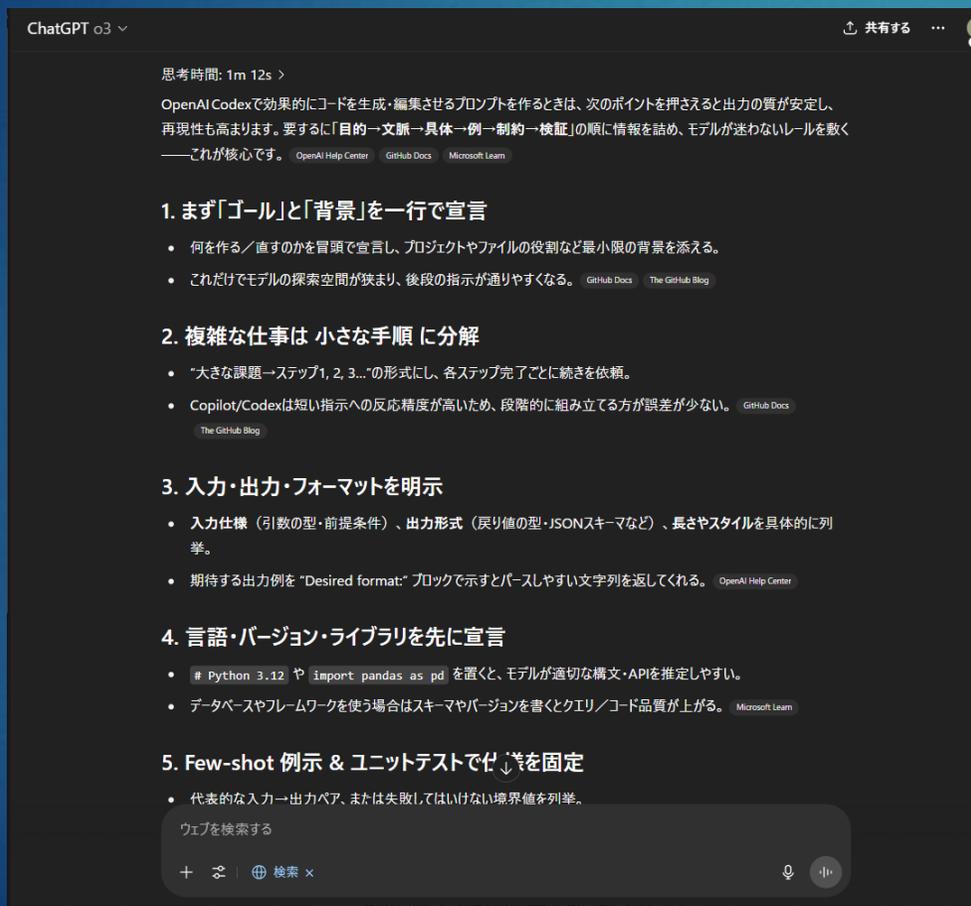
Gemini に相談

+ 動画 Deep Research Canvas

Gemini は不正確な情報を表示することがあるため、生成された回答を再確認するようにしてください。

## ■ ChatGPT o3

### ▶ プロンプト『OpenAI Codex でAI駆動開発する際のプロンプトに含める内容の要点を教えてください。』



### 3. バイブコーディングによるプログラム生成 最新のAIコーディングエージェント

▶ AIコーディングエージェントの進化は他のジャンルより早いです

AI開発ツール	種類	選択可能LLM	制限
Bolt	ブラウザ	独自	JSベースWeb
Cursor	IDE型	Claude, Gemini, OpenAI/GPT など	自由度が高い
Windsurf	IDE型	独自, DeepSeek, Claude, Gemini など	自由度が高い
Devin	対話型 自律型	選択不可でおそらく内製モデル主体	自律動作
ClaudeCode	CLI/自律型	Claude Opus 4, Claude Sonnet 4/3.7	自律動作
Gemini CLI	CLI/自律型	Jemini 2.5 Pro	自律動作
OpenAI Codex CLI	CLI/自律型	o4-mini, codex-mini など	自律動作

▶ 代表的なAIコーディングエージェント その1

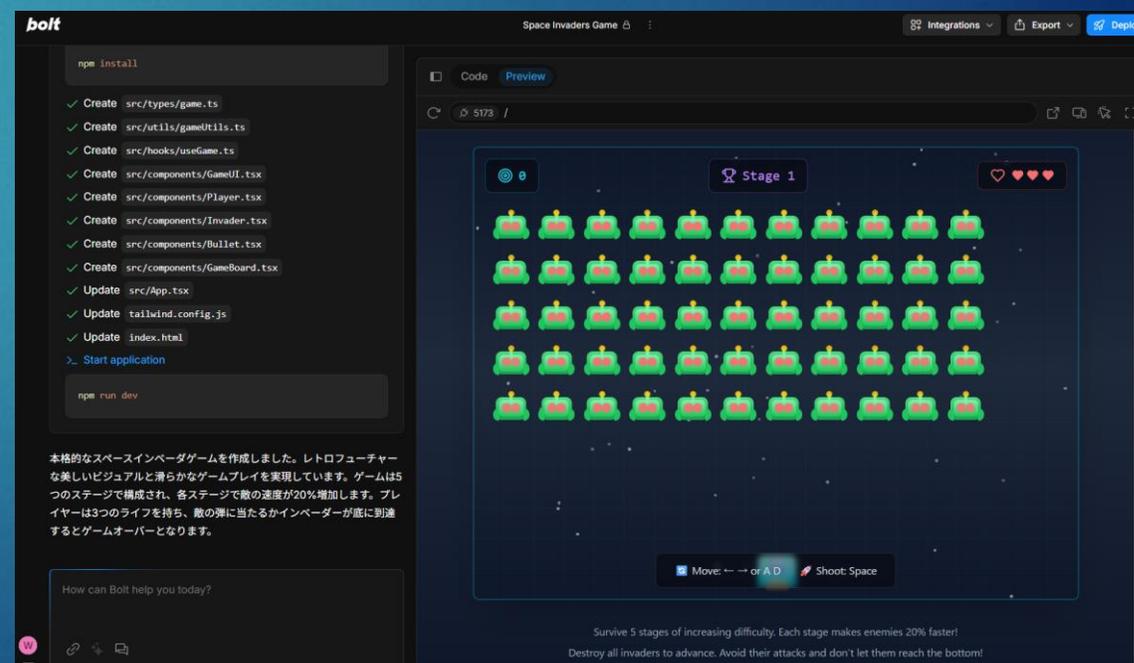
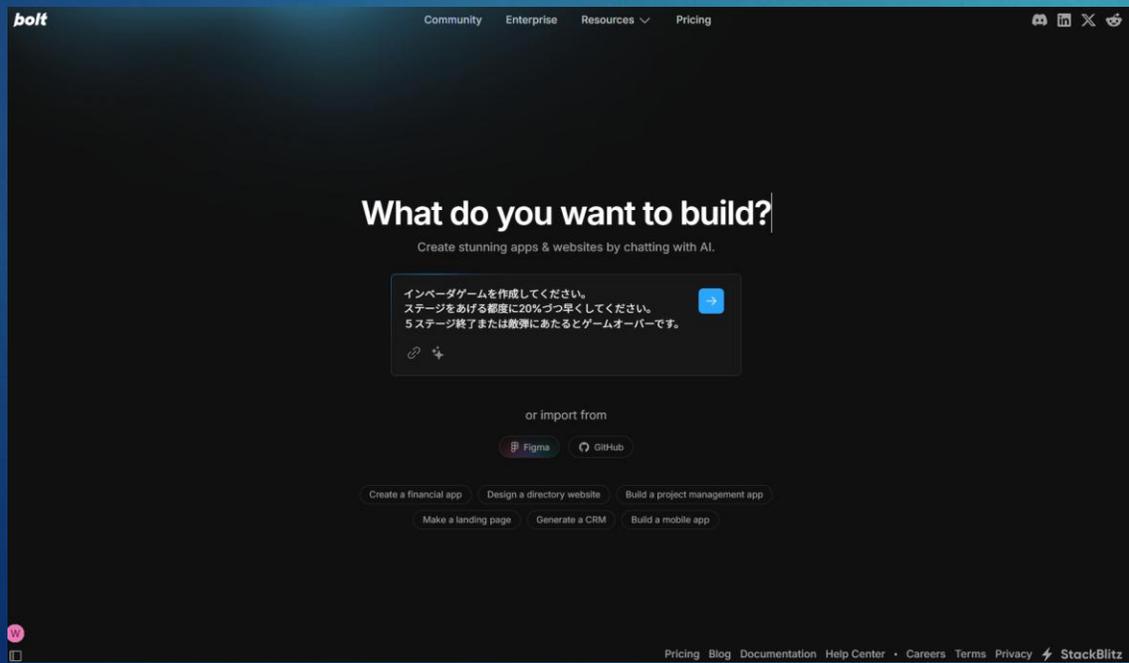
▶ Bolt : ブラウザ完結型Web開発エージェント

1プロンプトでJSアプリを自動生成してWebコンテナで動作確認できます

Figma / Supabase / Stripe など JS 系 SaaS との公式統合が豊富です

Netlify等へワンクリックデプロイできます

非Web系アプリは作成できません



## ▶ 代表的なAIコーディングエージェント その2

### ▶ Cursor : AIネイティブIDE型コーディングエージェント

VS Code派生のAIコーディングエージェントで、VS Codeの拡張機能をまるごと使えます

VS Codeからの移管がスムーズで、環境をまるごと引き継いでAI活用を開始できます

豊富なLLMモデルをリーズナブルに利用可能です (Claude 3.7 Sonnetで人気に)

開発プロジェクト全域を支援していますが、コーディング以外のドキュメント生成も可能

エラー原因を特定して何度でも自動修復してくれます

リファクタリング、CI/CD対応で、パイプラインフローを生成可能です

TDD対応可能で、明示すればテスト設計とレッドのテストコードを生成します

MCP連携がワンクリックで可能です

最新のバージョンアップで以下が追加されました :

Background Agents : ClaudeCode同様のバックグラウンド処理機能

Bug bot : チームメンバーのコーディングのバグを検出

## ▶ 代表的なAIコーディングエージェント その3

### ▶ Windsurf : AIネイティブIDE型開発エージェント

独立IDE型(自称『エージェントネイティブIDE』)のAIコーディングエージェントです  
開発プロジェクト全域を支援しますが、コーディング以外の自動設定範囲は限られています  
IDE機能範囲内で自律的に動作し、エラー原因を特定して何度でも自動修復してくれます  
リファクタリング、CI/CD対応で、複数ファイルにまたがる自動修正が可能です  
TDD対応可能で、明示すればテスト設計とレッドのテストコードを生成します  
70以上の開発言語に対応しています

▶ OpenAI社は、この5月にWindsuff社を買収すると発表しています。

## ▶ 代表的なAIコーディングエージェント その4

### ▶ Devin : 完全自律型AIコーディングエージェント

開発プロジェクト全域をカバーして、デプロイ/公開まで完全自律で動作可能です  
大規模で複雑な開発プロジェクトで自動化したい場合に有効かもしれません  
初期プロンプトから初期計画策定(Key Steps)や未解決問題を生成してきます  
人はチャットで Devin に要不要の判断や質問への回答に答えていきます  
Slack と連携して、メンションのみで依頼ができます、  
高機能なだけに、月額\$500と高価です

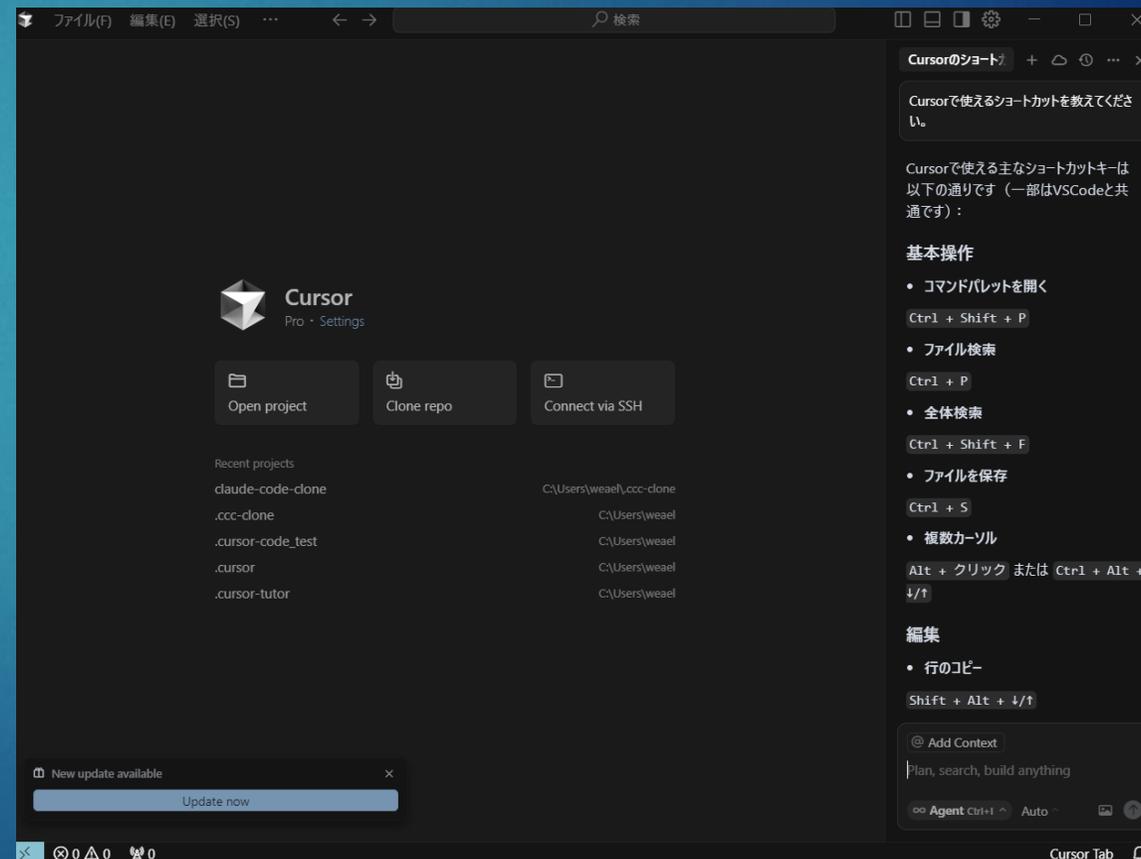
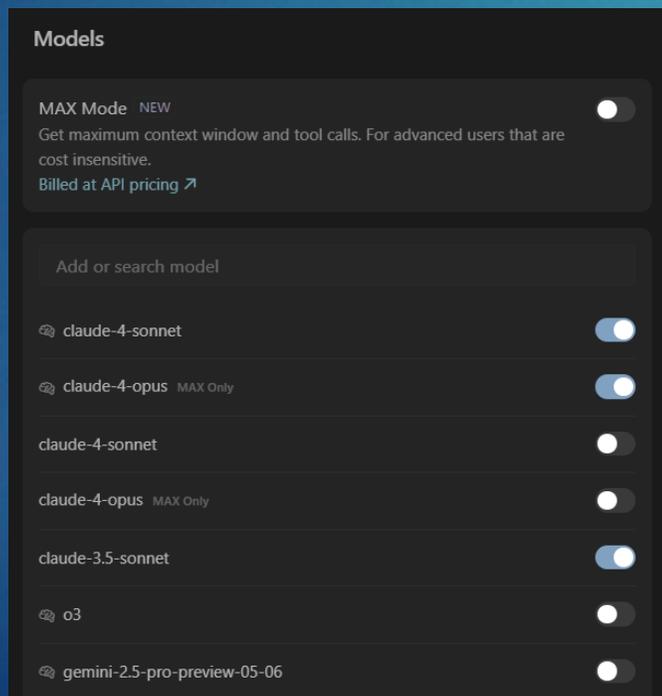
▶ IDE型AIコーディングエージェントのお勧めは Cursor です

▶ Cursor のアカウント登録はGitHubのアカウントで進めるのがお勧めです

▶ 覚えるべきショートカットは、AIチャット機能on/off (Command+Shift+L, Ctrl+L)

▶あとは AI に教わりましょう

▶AIモデルは Claude がお勧めです



▶ Cursorを本格的に使い始める前に、rules を設定しましょう

▶ プロジェクト固有のパターンや制約などをエージェントに伝える仕組みが rules です

The screenshot shows the Cursor IDE interface. On the left, the file explorer shows the project structure with a file named 'coding-rules.mdc' under the '.cursor/rules' directory. The main panel displays the 'Rules' settings for the user 'weasel@attglobal.sakura.n...'. It includes sections for 'Memories', 'User Rules', and 'Project Rules'. The 'User Rules' section shows a rule named 'Always respond in 日本語'. The 'Project Rules' section has an option to 'Include .cursorsrules file'. On the right, a code editor shows the content of 'coding-rules.mdc' with the following text:

```

Rule Type
Always This rule is attached to every chat and command+k request

まず、ユーザーから受け取った指示を確認します：
<指示>
{{instructions}}
<!-- このテンプレート変数はユーザーの入力プロンプトに自動置換されます -->
</指示>

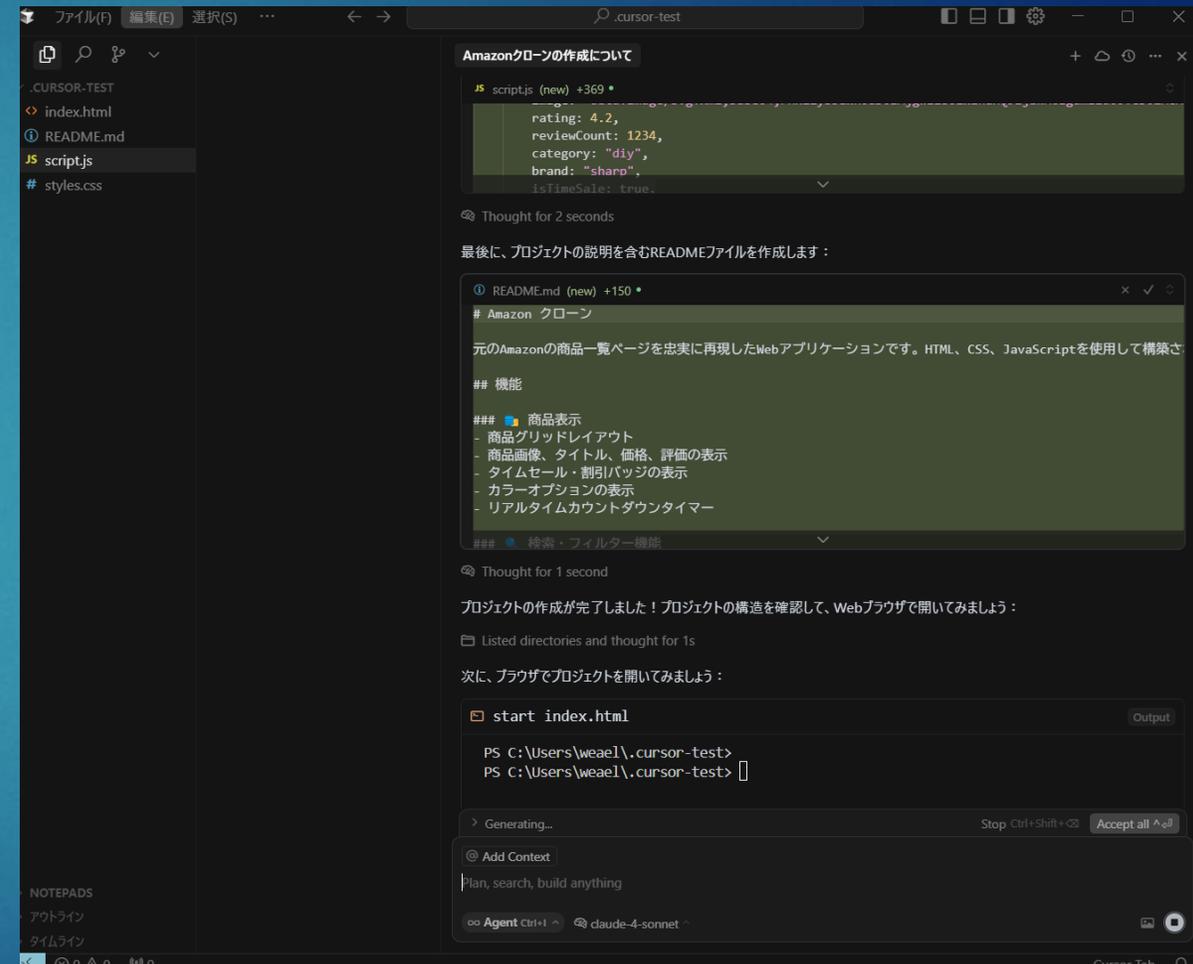
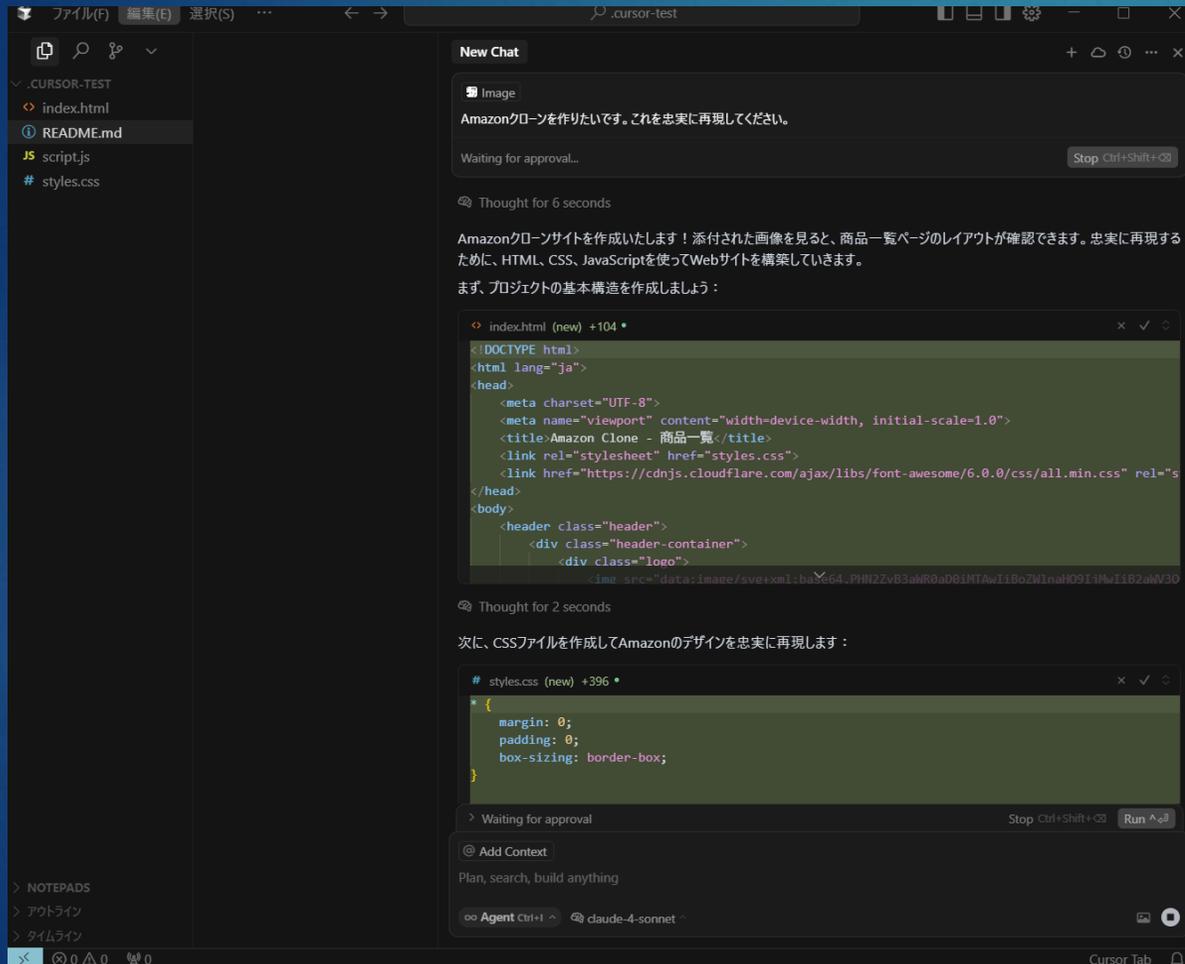
この指示を元に、以下のプロセスに従って作業を進めてください：
---

1. 指示の分析と計画
<タスク分析>
- 主要なタスクを簡潔に要約してください。
- 記載された**守るべきルールディレクトリ/ファイル**を必ずチェックしてください。
- 重要な要件と制約を特定してください。
- 潜在的な課題をリストアップしてください。
- タスク実行のための具体的なステップを詳細に列挙してください。
- それらのステップの最適な実行順序を決定してください。

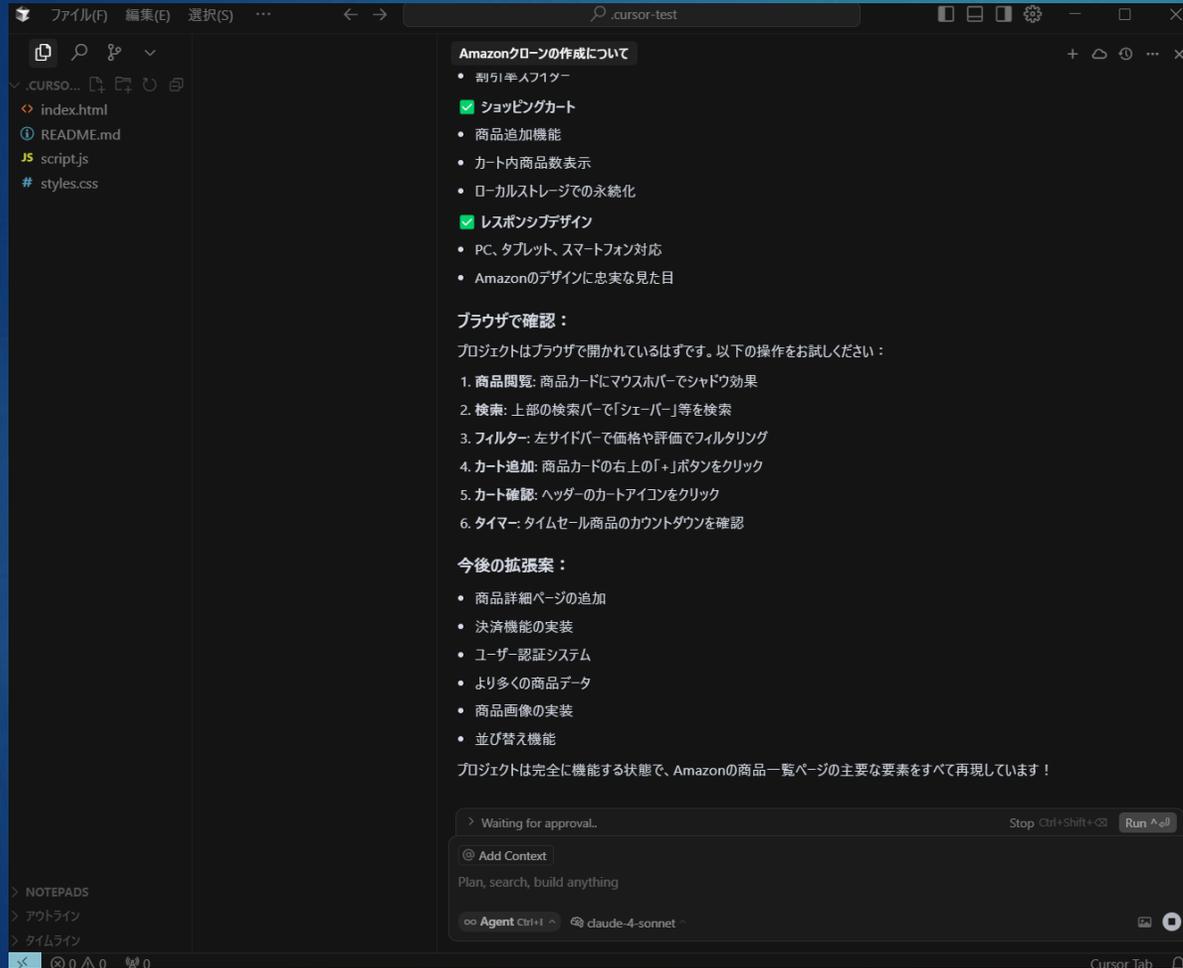
### 重複実装の防止
実装前に以下の確認を行ってください：
- 既存の類似機能の有無
- 同名または類似名の関数やコンポーネント
- 重複するAPIエンドポイント
- 共通化可能な処理の特定

このセクションは、後続のプロセス全体を導くものなので、時間をかけてでも、十分に詳細かつ包括的な分析を行ってください。
</タスク分析>
    
```

# ■ CursorでのAI駆動開発の例 その1

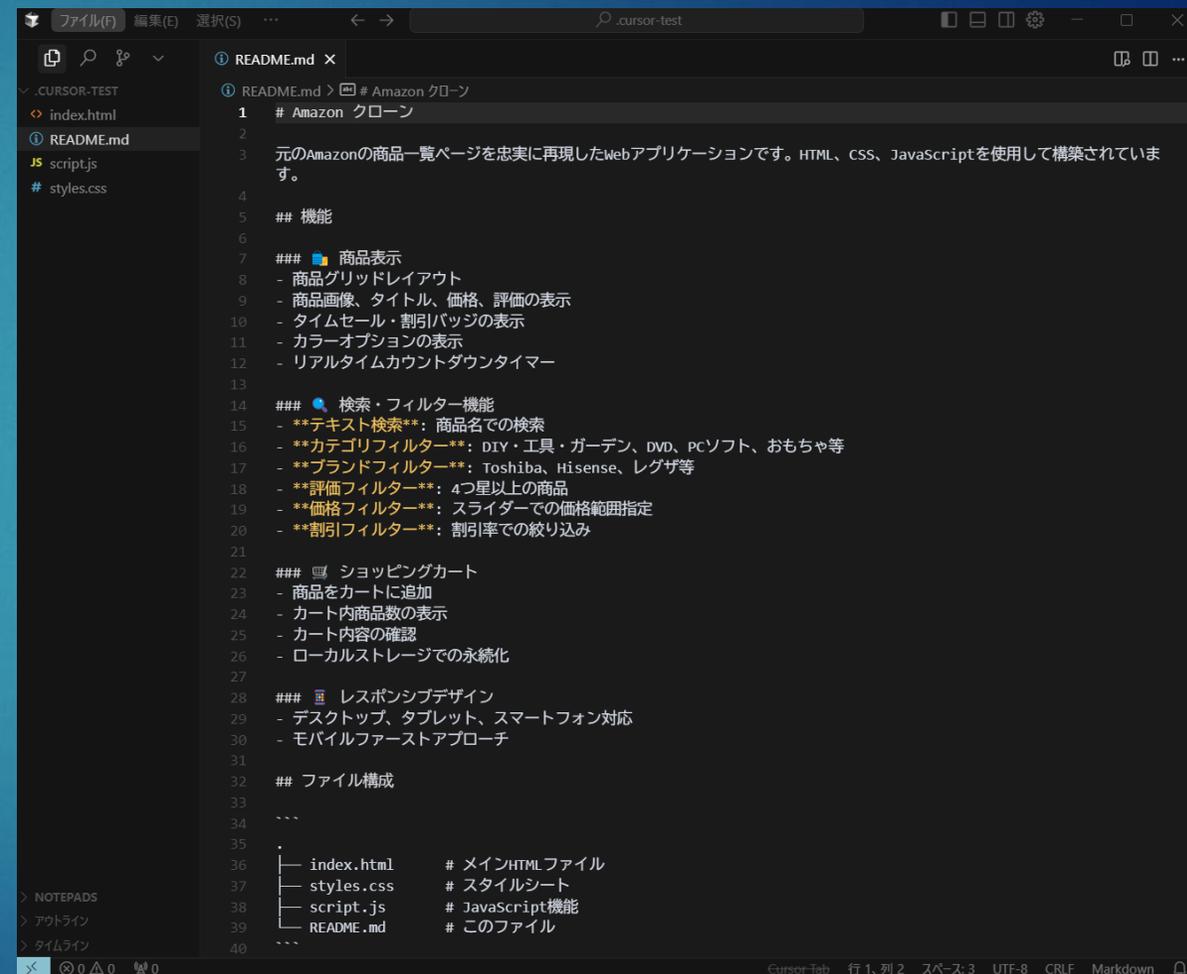
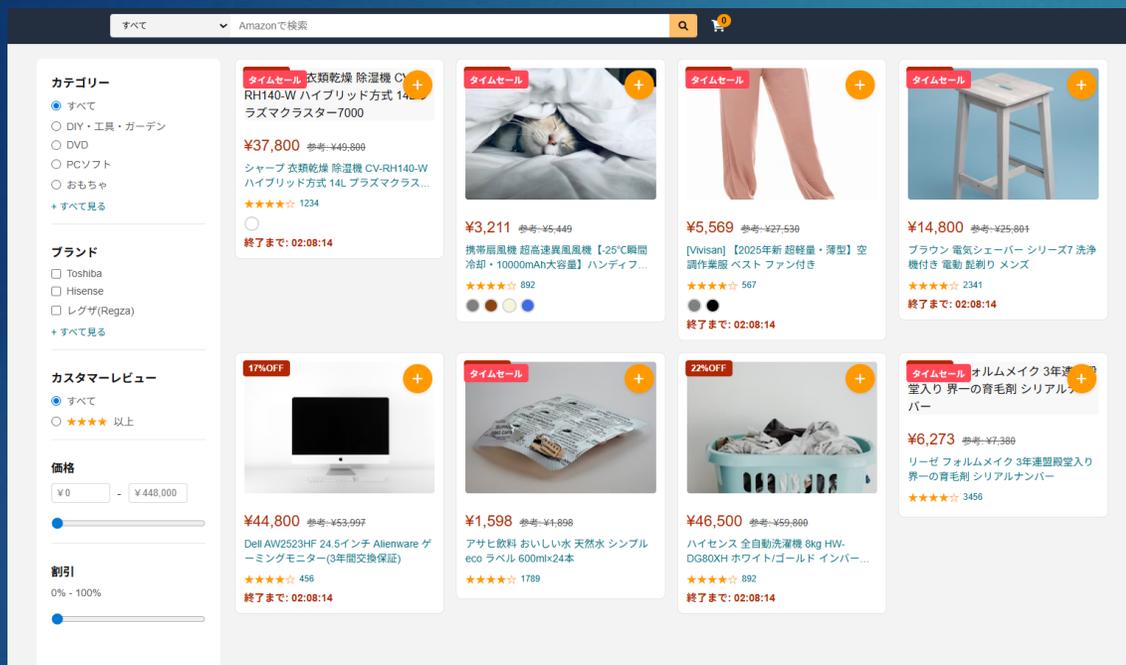


## ■ CursorでのAI駆動開発の例 その2





# CursorでのAI駆動開発の例 その4



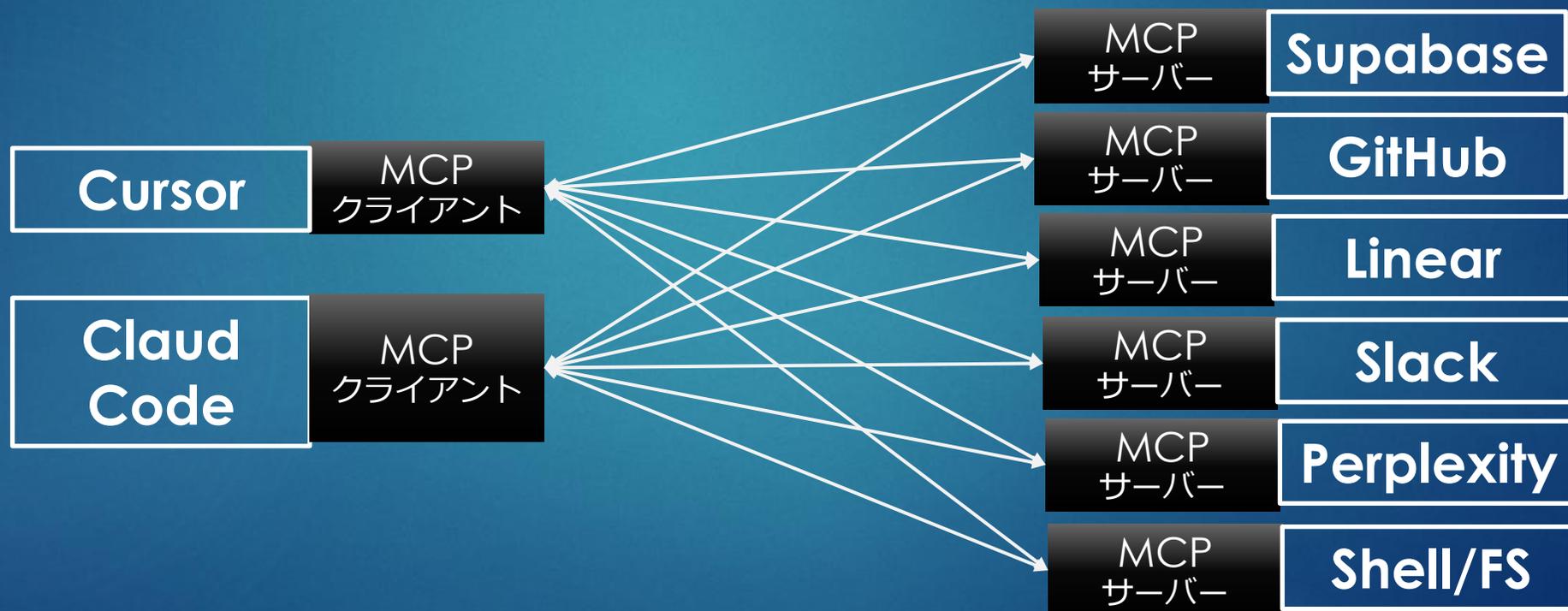
## 4. エージェントAIツールによるAI駆動開発

- ▶ Anthropic社の MCP と ClaudeCode の登場で、AIEージェントの開発が活発になりました
  - ▶ MCP (Model Context Protocol) の登場で、AIEージェントの開発が楽になるとともに、AI のエージェント化が進んでいます
  - ▶ 外部ツールを中心にMCP対応が増えており、MCPがデファクト標準になりつつあります
    - ▶ 外部ツールは、MCPを介して頭脳に該当するご主人のために力を発揮できるようになりました
    - ▶ AIEージェントは、MCPを介して手足や目を獲得し、外部記憶や目を利用できるようになりました
  - ▶ CLI型のエージェントAIツールの ClaudeCode の登場で、開発プロセス全域での自律的な動作が可能となり、人の関与をさらに減らすことになりました

## 4. エージェントAIツールによるAI駆動開発 MCPで多様な連携を促進

## ▶ MCP (Model Context Protocol) とは何か？

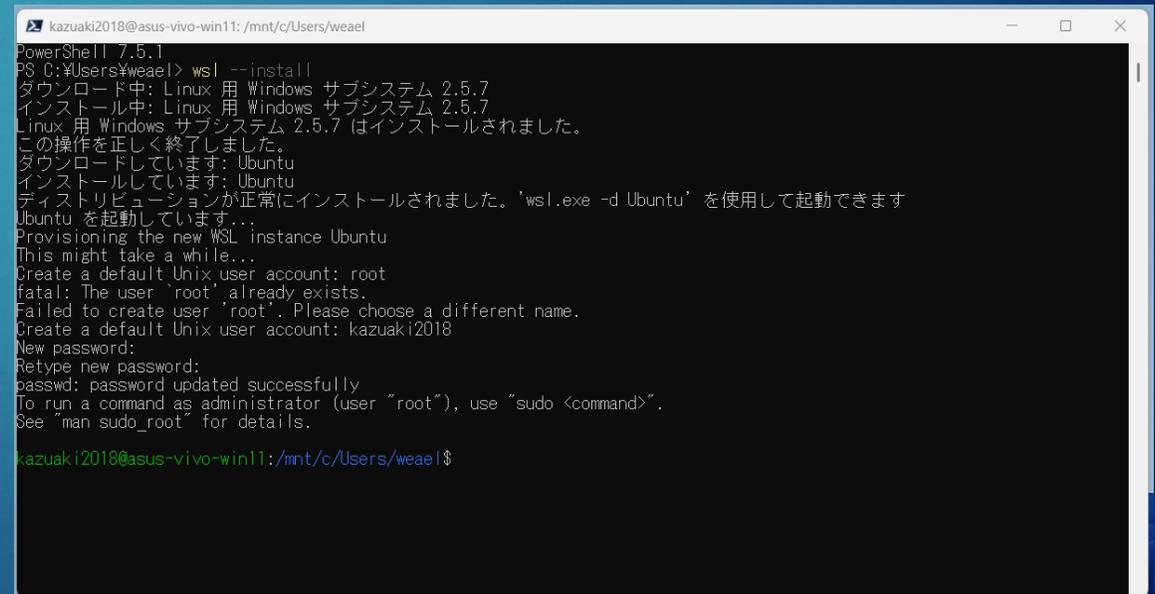
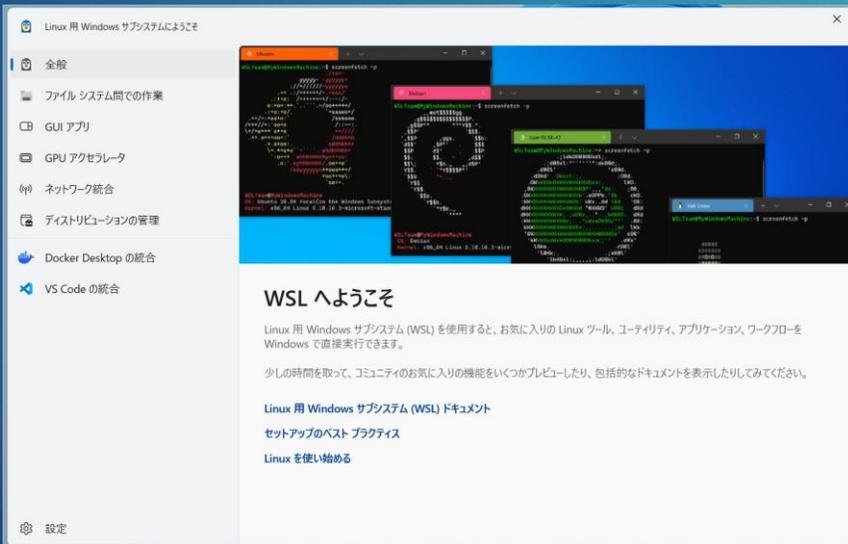
- ▶ Anthropic社が発表した、AIEージェントと外部ツールとのやり取りを標準化したプロトコルです。AIEエージェントの開発を促進するための標準とも言えます。
- ▶ クライアント サーバー型アーキテクチャです



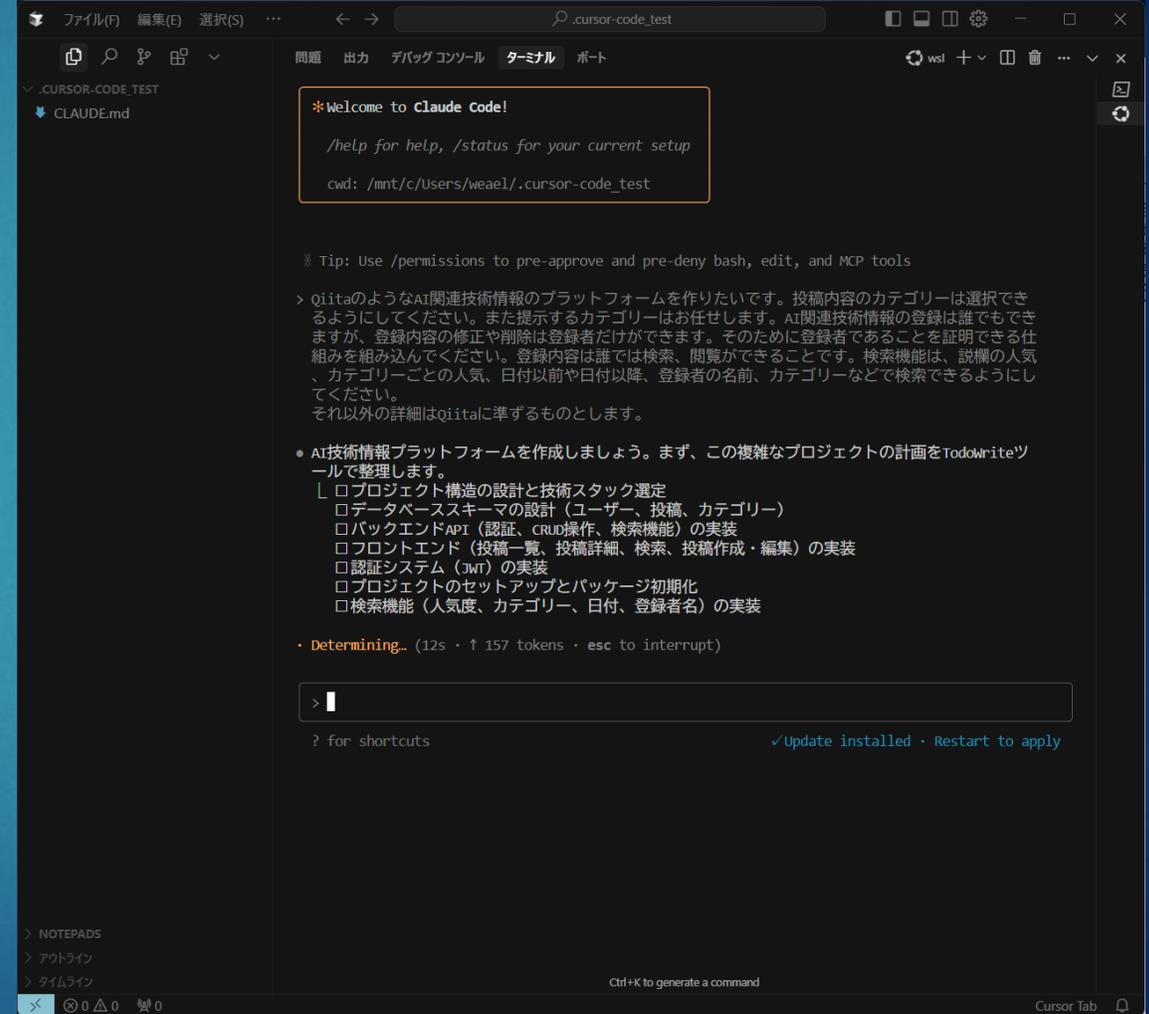
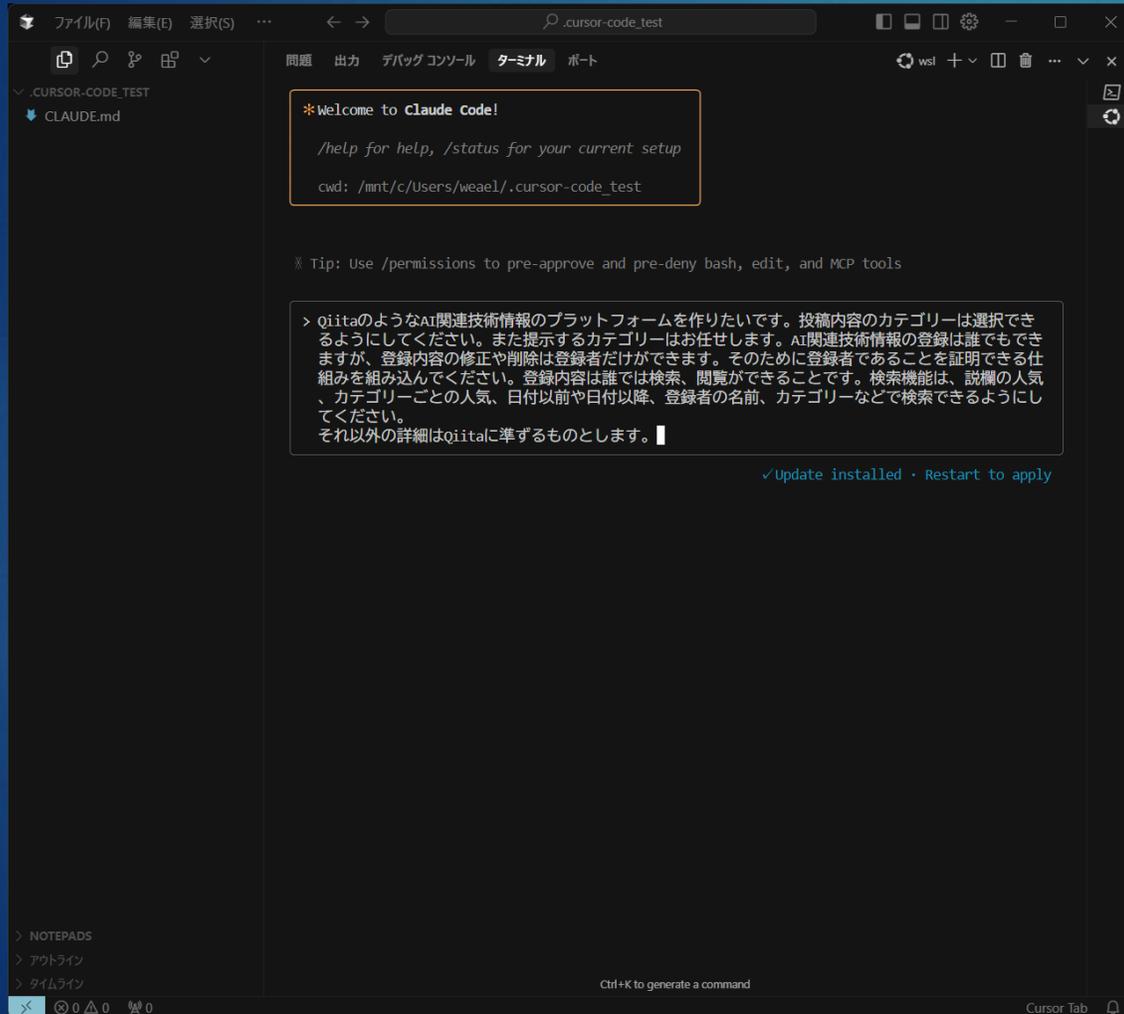
## 4. エージェントAIツールによるAI駆動開発 プロジェクトの長時間自律走行

- ▶ Anthropic社のClaudeCode登場で、マルチエージェントによる自律的な協働作業の道が拓かれました
- ▶ その特徴は
  - ▶ プロジェクト全体の理解力があり、自律的にワークフローと開発タスクを遂行できます
  - ▶ CLIベースでAI処理をバックグラウンド化でき、AI処理中でも指示の変更や追加ができます
  - ▶ GitHub Actionsで連携したりtmux (ターミナルマルチプレクサ)で動かせば複数セッションで同時にコード生成させたり指示に基づいたオーケストレーション動作が期待できます
  - ▶ プロンプトのみならず、開発言語や出力ドキュメントも多言語対応です
  - ▶ 長時間の自動実行が可能です (最長で7時間の実績あり)
  - ▶ MCP連携で動作可能です
    - ▶ CursorなどのMCPクライアントから呼び出し利用可能です
    - ▶ GitHubのissuesやPRを自動管理し、コードレビューやトラッキングなどを共有できます

- ▶ ClaudeCodeで使えるのはAnthropic社のLLMとなります。
- ▶ ClaudeCode はテストや動作確認のための環境構築も自律的に実行しますので、MacOS 10.15以降、または Linux の Ubuntu 20.04 以降や Debian 10 以降が必要です。
- ▶ Windowsで動かすためには WSL (Windows Subsystem for Linux)を使って上記 Linux ディストリビューションを導入する必要があります。



## ■ ClaudeCodeでのAI駆動開発の例 その1



## ■ ClaudeCodeでのAI駆動開発の例 その2

問題を出力 デバッグコンソール ターミナル ポート

組みを組み込んでください。登録内容は誰でも検索、閲覧が可能なことです。検索機能は、説欄の人気、カテゴリごとの人気、日付前や日付以降、登録者の名前、カテゴリなどで検索できるようにしてください。それ以外の詳細はqiitaに準ずるものとします。

- AI技術情報プラットフォームを作成しましょう。まず、この複雑なプロジェクトの計画をTodoWriteツールで整理します。
  - プロジェクト構造の設計と技術スタック選定
  - データベーススキーマの設計 (ユーザー、投稿、カテゴリ)
    - バックエンドAPI (認証、CRUD操作、検索機能) の実装
    - フロントエンド (投稿一覧、投稿詳細、検索、投稿作成・編集) の実装
    - 認証システム (JWT) の実装
    - プロジェクトのセットアップとパッケージ初期化
    - 検索機能 (人気度、カテゴリ、日付、登録者名) の実装
- プロジェクトのセットアップから始めます。Node.js + Express + MongoDB + Reactでモダンなwebアプリケーションを構築します。
  - プロジェクト構造の設計と技術スタック選定
    - データベーススキーマの設計 (ユーザー、投稿、カテゴリ)
    - バックエンドAPI (認証、CRUD操作、検索機能) の実装
    - フロントエンド (投稿一覧、投稿詳細、検索、投稿作成・編集) の実装
    - 認証システム (JWT) の実装
    - プロジェクトのセットアップとパッケージ初期化
    - 検索機能 (人気度、カテゴリ、日付、登録者名) の実装
- まず、プロジェクト構造を作成します :
 

```
Bash(mkdir -p backend/{src/{controllers,models,routes,middleware,config},tests} frontend/{src/{components,pages,services,contexts,styles},public} && ls -la)
```

**Bash command**

```
mkdir -p backend/{src/{controllers,models,routes,middleware,config},tests} frontend/{src/{components,pages,services,contexts,styles},public} && ls -la
```

Create project structure directories

Do you want to proceed?

- Yes
- Yes, and don't ask again for mkdir and ls commands in /mnt/c/Users/weael/.cursor-code\_test
- No, and tell Claude what to do differently (esc)

問題を出力 デバッグコンソール ターミナル ポート

- バックエンドAPI (認証、CRUD操作、検索機能) の実装
- フロントエンド (投稿一覧、投稿詳細、検索、投稿作成・編集) の実装
- 認証システム (JWT) の実装
- プロジェクトのセットアップとパッケージ初期化
- 検索機能 (人気度、カテゴリ、日付、登録者名) の実装

- プロジェクトのセットアップから始めます。Node.js + Express + MongoDB + Reactでモダンなwebアプリケーションを構築します。
  - プロジェクト構造の設計と技術スタック選定
    - データベーススキーマの設計 (ユーザー、投稿、カテゴリ)
    - バックエンドAPI (認証、CRUD操作、検索機能) の実装
    - フロントエンド (投稿一覧、投稿詳細、検索、投稿作成・編集) の実装
    - 認証システム (JWT) の実装
    - プロジェクトのセットアップとパッケージ初期化
    - 検索機能 (人気度、カテゴリ、日付、登録者名) の実装
- まず、プロジェクト構造を作成します :
 

```
Bash(mkdir -p backend/{src/{controllers,models,routes,middleware,config},tests} frontend/{src/{components,pages,services,contexts,styles},public} && ls -la)
```
- バックエンドの初期化を行います :
 

```
Bash(cd backend && npm init -y)
```

**Bash command**

```
cd backend && npm init -y
```

Initialize backend package.json

Do you want to proceed?

- Yes
- Yes, and don't ask again for npm init commands in /mnt/c/Users/weael/.cursor-code\_test
- No, and tell Claude what to do differently (esc)

## ■ ClaudeCodeでのAI駆動開発の例 その3

問題 出力 デバッグコンソール ターミナル ポート

- フロントエンド (投稿一覧、投稿詳細、検索、投稿作成・編集) の実装
- 認証システム (JWT) の実装
- プロジェクトのセットアップとパッケージ初期化
- 検索機能 (人気度、カテゴリー、日付、登録者名) の実装

- まず、プロジェクト構造を作成します：

```

● Bash(mkdir -p backend/{src/{controllers,models,routes,middleware,config},tests}
  frontend/{src/{components,pages,services,contexts,styles},public} && ls -la)
├── total 4
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 .
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun  9 17:24 ..
├── -rwxrwxrwx 1 kazuaki2018 kazuaki2018 882 Jun  9 17:35 CLAUDE.md
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 backend
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 frontend

```

- バックエンドの初期化を行います：

```

● Bash(cd backend && npm init -y)
├── Wrote to /mnt/c/Users/weael/.cursor-code_test/backend/package.json:
{
  "name": "backend",
  ... +13 lines (ctrl+r to expand)

```

- バックエンドの依存関係をインストールします：

```

Bash(cd backend && npm install express mongoose jsonwebtoken bcryptjs cors dotenv
  body-parser && npm install -D nodemon)
├── Running...

```

**Bash command**

```
cd backend && npm install express mongoose jsonwebtoken bcryptjs cors dotenv body-parser &&
npm install -D nodemon
Install backend dependencies
```

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for npm install commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)

NOTEPADS  
アウトライン  
タイムライン

Ctrl+K to generate a command

問題 出力 デバッグコンソール ターミナル ポート

- Bash(mkdir -p backend/{src/{controllers,models,routes,middleware,config},tests}
 frontend/{src/{components,pages,services,contexts,styles},public} && ls -la)
├── total 4
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 .
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 9 17:24 ..
├── -rwxrwxrwx 1 kazuaki2018 kazuaki2018 882 Jun 9 17:35 CLAUDE.md
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 backend
├── drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 frontend
- バックエンドの初期化を行います：

```

● Bash(cd backend && npm init -y)
├── Wrote to /mnt/c/Users/weael/.cursor-code_test/backend/package.json:
{
  "name": "backend",
  ... +13 lines (ctrl+r to expand)

```

- バックエンドの依存関係をインストールします：

```

● Bash(cd backend && npm install express mongoose jsonwebtoken bcryptjs cors dotenv
  body-parser && npm install -D nodemon)
├── Error: /bin/bash: line 1: cd: backend: No such file or directory

```

- List(..)
├── Listed 2 paths (ctrl+r to expand)

```

Bash(npm install express mongoose jsonwebtoken bcryptjs cors dotenv body-parser && npm
  install -D nodemon)
├── Running...

```

**Bash command**

```
npm install express mongoose jsonwebtoken bcryptjs cors dotenv body-parser && npm install
-D nodemon
Install backend dependencies with absolute path
```

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for npm install commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)

NOTEPADS  
アウトライン  
タイムライン

Ctrl+K to generate a command

# ClaudeCodeでのAI駆動開発の例 その4

The screenshot shows the ClaudeCode interface with a terminal window. The terminal output is as follows:

```

問題 出力 デバッグコンソール ターミナル ポート
• バックエンドの初期化を行います：
• Bash(cd backend && npm init -y)
  ↳ wrote to /mnt/c/Users/weael/.cursor-code_test/backend/package.json:
  {
    "name": "backend",
    ... +13 lines (ctrl+r to expand)
• バックエンドの依存関係をインストールします：
• Bash(cd backend && npm install express mongoose jsonwebtoken bcryptjs cors dotenv
  body-parser && npm install -D nodemon)
  ↳ Error: /bin/bash: line 1: cd: backend: No such file or directory
• List(..)
  ↳ Listed 2 paths (ctrl+r to expand)
• Bash(npm install express mongoose jsonwebtoken bcryptjs cors dotenv body-parser && npm
  install -D nodemon)
  ↳ added 101 packages, and audited 102 packages in 8s

  16 packages are looking for funding
  run `npm fund` for details
  ... +9 lines (ctrl+r to expand)
• フロントエンドプロジェクトを作成します：
• Bash(npx create-react-app frontend --template typescript)
  ↳ Running...
  
```

A callout box titled "Bash command" shows the command: `npx create-react-app frontend --template typescript` with the instruction "Create React frontend with TypeScript". Below it, a prompt asks "Do you want to proceed?" with three options: 1. Yes, 2. Yes, and don't ask again for npx create-react-app commands in /mnt/c/Users/weael/.cursor-code\_test, 3. No, and tell Claude what to do differently (esc).

The screenshot shows the ClaudeCode interface with a terminal window. The terminal output is as follows:

```

install -D nodemon)
↳ added 101 packages, and audited 102 packages in 8s

16 packages are looking for funding
run `npm fund` for details
... +9 lines (ctrl+r to expand)
• フロントエンドプロジェクトを作成します：
• Bash(npx create-react-app frontend --template typescript)
  ↳ Error: Command timed out after 2m 0.0s npm warn exec The following package was not found
  and will be installed: create-react-app@5.1.0
  npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
  npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not
  use it. Check out lru-cache if you want a good and tested way to coalesce async requests
  by a key value, which is much more comprehensive and powerful.
  npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
  npm warn deprecated fstream@1.0.12: This package is no longer supported.
  npm warn deprecated uid-number@0.0.6: This package is no longer supported.
  npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
  npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not
  receive security updates. Please upgrade asap.

  create-react-app is deprecated.
  ... +16 lines (ctrl+r to see all)
• 既存のfrontendディレクトリがあるため、手動でReactプロジェクトを作成します：
• Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
  ↳ Running...
  
```

A callout box titled "Bash command" shows the command: `rm -rf frontend && npx create-react-app frontend --template typescript` with the instruction "Remove existing frontend and create new React project". Below it, a prompt asks "Do you want to proceed?" with three options: 1. Yes, 2. Yes, and don't ask again for rm and npx create-react-app commands in /mnt/c/Users/weael/.cursor-code\_test, 3. No, and tell Claude what to do differently (esc).

## ■ ClaudeCodeでのAI駆動開発の例 その5

```

• Bash(npx create-react-app frontend --template typescript)
└─ Error: Command timed out after 2m 0.0s npm warn exec The following package was not found
and will be installed: create-react-app@5.1.0
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not
use it. Check out lru-cache if you want a good and tested way to coalesce async requests
by a key value, which is much more comprehensive and powerful.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not
receive security updates. Please upgrade asap.

create-react-app is deprecated.
... +16 lines (ctrl+r to see all)

• 既存のfrontendディレクトリがあるため、手動でReactプロジェクトを作成します :

• Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
└─ Creating a new React app in /mnt/c/Users/weael/.cursor-code_test/backend/frontend.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template-typescript...
... +80 lines (ctrl+r to expand)

• フロントエンドが作成されましたが、間違った場所にあります。移動させます :

Bash(mv backend/frontend . && ls -la)
└─ Running..

Bash command
mv backend/frontend . && ls -la
Move frontend to correct location

Do you want to proceed?
> 1. Yes
2. Yes, and don't ask again for mv and ls commands in /mnt/c/Users/weael/.cursor-code_test
3. No, and tell Claude what to do differently (esc)
    
```

```

npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not
use it. Check out lru-cache if you want a good and tested way to coalesce async requests
by a key value, which is much more comprehensive and powerful.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not
receive security updates. Please upgrade asap.

create-react-app is deprecated.
... +16 lines (ctrl+r to see all)

• 既存のfrontendディレクトリがあるため、手動でReactプロジェクトを作成します :

• Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
└─ Creating a new React app in /mnt/c/Users/weael/.cursor-code_test/backend/frontend.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template-typescript...
... +80 lines (ctrl+r to expand)

• フロントエンドが作成されましたが、間違った場所にあります。移動させます :

• Bash(mv backend/frontend . && ls -la)
└─ Error: mv: cannot stat 'backend/frontend': No such file or directory

• List(.)
└─ Listed 35 paths (ctrl+r to expand)

• Bash(cd backend && mv frontend .. && cd .. && ls -la)
└─ Running..

Bash command
cd backend && mv frontend .. && cd .. && ls -la
Move frontend to root directory

Do you want to proceed?
> 1. Yes
2. No, and tell Claude what to do differently (esc)
    
```

# ■ ClaudeCodeでのAI駆動開発の例 その6

```

npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not
receive security updates. Please upgrade asap.

create-react-app is deprecated.
... +16 lines (ctrl+r to see all)

• 既存のfrontendディレクトリがあるため、手動でReactプロジェクトを作成します：

• Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
  └─ Creating a new React app in /mnt/c/Users/weael/.cursor-code_test/backend/frontend.

      Installing packages. This might take a couple of minutes.
      Installing react, react-dom, and react-scripts with cra-template-typescript...
      ... +80 lines (ctrl+r to expand)

• フロントエンドが作成されましたが、間違った場所にあります。移動させます：

• Bash(mv backend/frontend . && ls -la)
  └─ Error: mv: cannot stat 'backend/frontend': No such file or directory

• List(.)
  └─ Listed 35 paths (ctrl+r to expand)

• Bash(cd backend && mv frontend .. && cd .. && ls -la)
  └─ Error: /bin/bash: line 1: cd: backend: No such file or directory

• Bash(mv backend/frontend . && ls -la)
  └─ Running..
    
```

**Bash command**

```
mv backend/frontend . && ls -la
```

Move frontend directory using full path

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for mv and ls commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)

```

... +16 lines (ctrl+r to see all)

• 既存のfrontendディレクトリがあるため、手動でReactプロジェクトを作成します：

• Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
  └─ Creating a new React app in /mnt/c/Users/weael/.cursor-code_test/backend/frontend.

      Installing packages. This might take a couple of minutes.
      Installing react, react-dom, and react-scripts with cra-template-typescript...
      ... +80 lines (ctrl+r to expand)

• フロントエンドが作成されましたが、間違った場所にあります。移動させます：

• Bash(mv backend/frontend . && ls -la)
  └─ Error: mv: cannot stat 'backend/frontend': No such file or directory

• List(.)
  └─ Listed 35 paths (ctrl+r to expand)

• Bash(cd backend && mv frontend .. && cd .. && ls -la)
  └─ Error: /bin/bash: line 1: cd: backend: No such file or directory

• Bash(mv backend/frontend . && ls -la)
  └─ Error: mv: cannot stat 'backend/frontend': No such file or directory

• フルパスで移動します：

• Bash(mv /mnt/c/Users/weael/.cursor-code_test/backend/frontend
  /mnt/c/Users/weael/.cursor-code_test/ && ls -la)
  └─ Running..
    
```

**Bash command**

```
mv /mnt/c/Users/weael/.cursor-code_test/backend/frontend
/mnt/c/Users/weael/.cursor-code_test/ && ls -la
```

Move frontend with absolute paths

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for mv and ls commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)



# ■ ClaudeCodeでのAI駆動開発の例 その8

The terminal window shows the following sequence of events:

- Initial state: `Listed 35 paths (ctrl+r to expand)`
- Command 1: `Bash(cd backend && mv frontend .. && cd .. && ls -la)`
  - Error: `Error: /bin/bash: line 1: cd: backend: No such file or directory`
- Command 2: `Bash(mv backend/frontend . && ls -la)`
  - Error: `Error: mv: cannot stat 'backend/frontend': No such file or directory`
- Instruction: フルパスで移動します :
- Command 3: `Bash(mv /mnt/c/Users/weael/.cursor-code_test/backend/frontend /mnt/c/Users/weael/.cursor-code_test/ && ls -la)`
  - Error: `Error: mv: cannot overwrite '/mnt/c/Users/weael/.cursor-code_test/frontend': Directory not empty`
- Command 4: `Bash(rm -rf frontend && mv /mnt/c/Users/weael/.cursor-code_test/backend/frontend /mnt/c/Users/weael/.cursor-code_test/ && ls -la)`
  - Error: `Error: mv: cannot stat '/mnt/c/Users/weael/.cursor-code_test/backend/frontend': No such file or directory`
- Instruction: 現在のディレクトリを確認します :
- Command 5: `Bash(pwd && ls -la)`
  - Output: `/mnt/c/Users/weael/.cursor-code_test/backend`
  - Output: `total 56`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:31 .`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 ..`
  - Output: `... +5 lines (ctrl+r to expand)`
- Command 6: `Bash(cd .. && ls -la)`
  - Output: `Running...`

A confirmation dialog is shown at the bottom:

```

Bash command
cd .. && ls -la
Go to root and check structure

Do you want to proceed?
> 1. Yes
2. Yes, and don't ask again for ls commands in /mnt/c/Users/weael/.cursor-code_test
3. No, and tell Claude what to do differently (esc)
    
```

The terminal window shows the following sequence of events:

- Command 7: `Bash(pwd && ls -la)`
  - Output: `/mnt/c/Users/weael/.cursor-code_test/backend`
  - Output: `total 56`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:31 .`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 ..`
  - Output: `... +5 lines (ctrl+r to expand)`
- Command 8: `Bash(cd .. && ls -la)`
  - Output: `total 4`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 .`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 9 17:24 ..`
  - Output: `-rwxrwxrwx 1 kazuaki2018 kazuaki2018 882 Jun 9 17:35 CLAUDE.md`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:31 backend`
  - Output: `drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 frontend`
- Instruction: 現在ルートディレクトリにいます。frontendディレクトリは空なので、バックエンドのpackage.jsonを設定します :
- Command 9: `Read(backend/package.json)`
  - Output: `Read 28 lines (ctrl+r to expand)`
- Command 10: `Update(backend/package.json)`

An "Edit file" dialog is shown for `backend/package.json`:

```

Edit file
backend/package.json
6   "test": "tests"
7   },
8   "scripts": {
9     "start": "node src/server.js",
10    "dev": "nodemon src/server.js",
11  },
12  "test": "echo \"Error: no test specified\" && exit 1"
13  },
14  "keywords": [],
15  }
    
```

A confirmation dialog is shown at the bottom:

```

Do you want to make this edit to package.json?
> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
    
```

# ■ ClaudeCodeでのAI駆動開発の例 その9

Terminal output:

```

● Bash(cd .. && ls -la)
├─ total 4
│  drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 .
│  drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun  9 17:24 ..
│  -rwxrwxrwx 1 kazuaki2018 kazuaki2018  882 Jun  9 17:35 CLAUDE.md
│  drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:31 backend
│  drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:19 frontend
└─

● 現在ルートディレクトリにいます。frontendディレクトリは空なので、バックエンドのpackage.jsonを設定します：

● Read(backend/package.json)
├─ Read 28 lines (ctrl+r to expand)
└─

● Update(backend/package.json)
├─ Updated backend/package.json with 2 additions
│  6   "test": "tests"
│  7   },
│  8   "scripts": {
│  9     "start": "node src/server.js",
│ 10    "dev": "nodemon src/server.js",
│ 11    "test": "echo \"Error: no test specified\" && exit 1"
│ 12  },
│ 13  "keywords": [],
└─

● フロントエンドプロジェクトを再作成します：

● Bash(rm -rf frontend && npx create-react-app frontend --template typescript)
├─ Running...
└─
    
```

**Bash command**

```

rm -rf frontend && npx create-react-app frontend --template typescript
Recreate React frontend project

Do you want to proceed?
> 1. Yes
2. Yes, and don't ask again for rm and npx create-react-app commands in /mnt/c/Users/weael/.cursor-code_test
3. No, and tell Claude what to do differently (esc)
    
```

```

bio: {
  type: String,
  maxLength: 500,
  default: ''
},
isActive: {
  type: Boolean,
  default: true
}
}, {
  timestamps: true
});

userSchema.pre('save', async function(next) {
  if (!this.isModified('password')) return next();

  try {
    const salt = await bcrypt.genSalt(10);
    this.password = await bcrypt.hash(this.password, salt);
  } catch (error) {
    next(error);
  }
});

userSchema.methods.comparePassword = async function(password) {
  return await bcrypt.compare(password, this.password);
};

userSchema.methods.toJSON = function() {
  const user = this.toObject();
  delete user.password;
  return user;
};

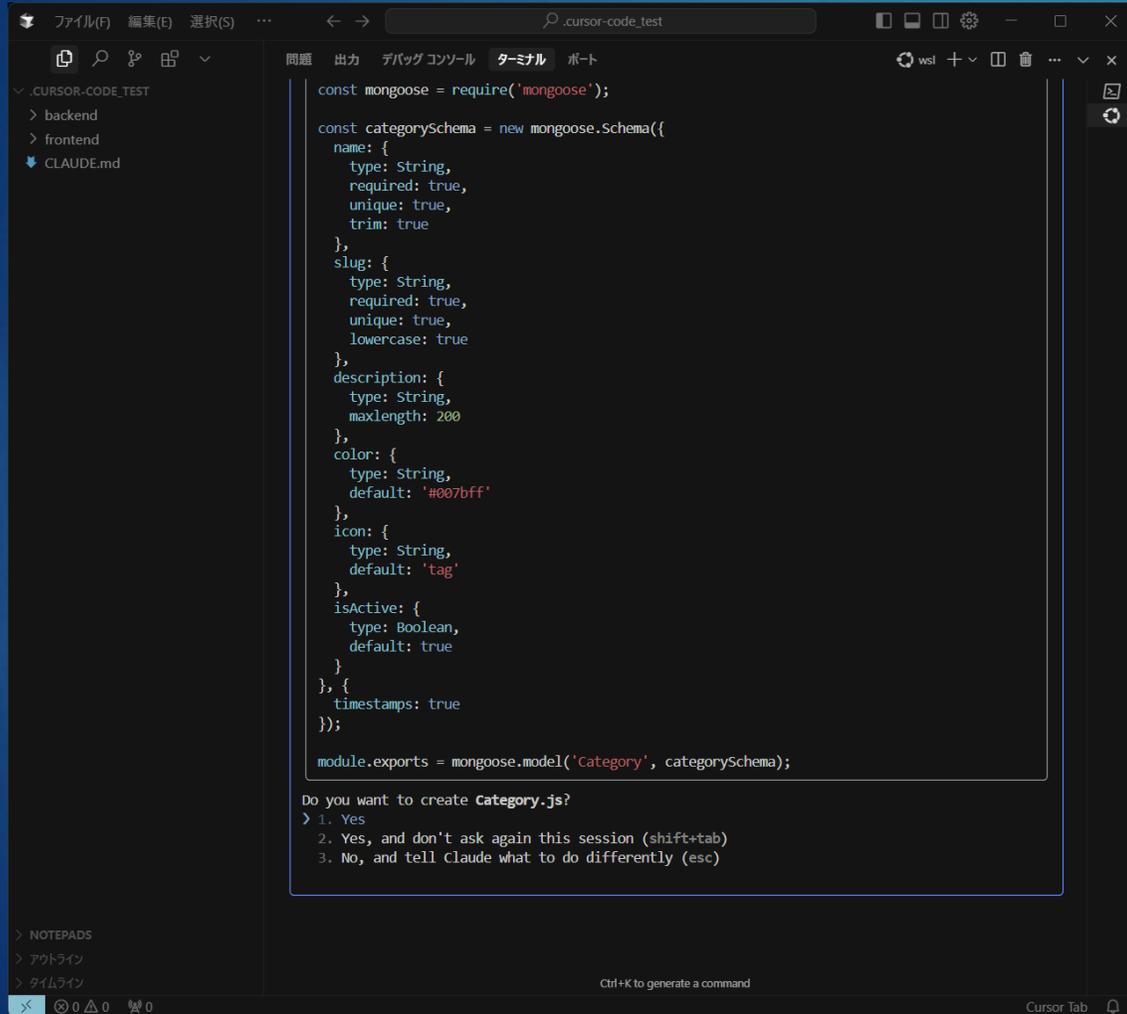
module.exports = mongoose.model('User', userSchema);
    
```

**Do you want to create User.js?**

```

> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
    
```

# ■ ClaudeCodeでのAI駆動開発の例 その10



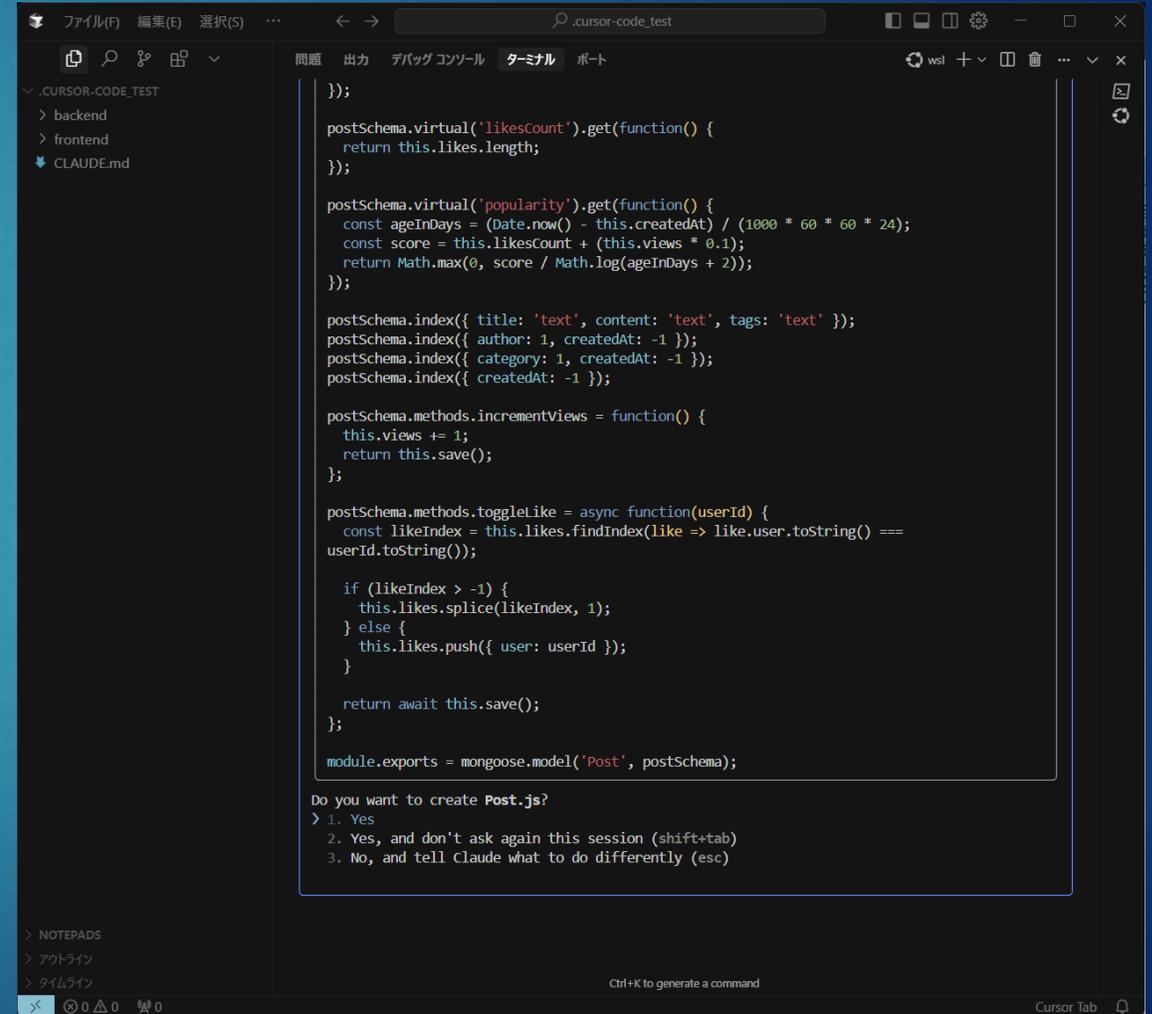
```
const mongoose = require('mongoose');

const categorySchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
    unique: true,
    trim: true
  },
  slug: {
    type: String,
    required: true,
    unique: true,
    lowercase: true
  },
  description: {
    type: String,
    maxLength: 200
  },
  color: {
    type: String,
    default: '#007bff'
  },
  icon: {
    type: String,
    default: 'tag'
  },
  isActive: {
    type: Boolean,
    default: true
  }
}, {
  timestamps: true
});

module.exports = mongoose.model('Category', categorySchema);
```

Do you want to create **Category.js**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)



```
});

postSchema.virtual('likesCount').get(function() {
  return this.likes.length;
});

postSchema.virtual('popularity').get(function() {
  const ageInDays = (Date.now() - this.createdAt) / (1000 * 60 * 60 * 24);
  const score = this.likesCount + (this.views * 0.1);
  return Math.max(0, score / Math.log(ageInDays + 2));
});

postSchema.index({ title: 'text', content: 'text', tags: 'text' });
postSchema.index({ author: 1, createdAt: -1 });
postSchema.index({ category: 1, createdAt: -1 });
postSchema.index({ createdAt: -1 });

postSchema.methods.incrementViews = function() {
  this.views += 1;
  return this.save();
};

postSchema.methods.toggleLike = async function(userId) {
  const likeIndex = this.likes.findIndex(like => like.user.toString() ===
  userId.toString());

  if (likeIndex > -1) {
    this.likes.splice(likeIndex, 1);
  } else {
    this.likes.push({ user: userId });
  }

  return await this.save();
};

module.exports = mongoose.model('Post', postSchema);
```

Do you want to create **Post.js**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

## ■ ClaudeCodeでのAI駆動開発の例 その11

```

'機械学習に関する技術情報', color: '#e74c3c', icon: 'brain' },
  { name: 'Deep Learning', slug: 'deep-learning', description:
'ディープラーニングに関する技術情報', color: '#9b59b6', icon: 'network-wired' },
  { name: 'Natural Language Processing', slug: 'nlp', description:
'自然言語処理に関する技術情報', color: '#3498db', icon: 'comments' },
  { name: 'Computer Vision', slug: 'computer-vision', description:
'コンピュータビジョンに関する技術情報', color: '#2ecc71', icon: 'eye' },
  { name: 'Data Science', slug: 'data-science', description:
'データサイエンスに関する技術情報', color: '#f39c12', icon: 'chart-bar' },
  { name: 'Reinforcement Learning', slug: 'reinforcement-learning', description:
'強化学習に関する技術情報', color: '#e67e22', icon: 'gamepad' },
  { name: 'AI Tools & Frameworks', slug: 'ai-tools', description:
'AI開発ツールとフレームワーク', color: '#1abc9c', icon: 'tools' },
  { name: 'Research Papers', slug: 'research-papers', description: 'AI研究論文の解説',
color: '#34495e', icon: 'file-alt' },
  { name: 'Business Applications', slug: 'business-ai', description:
'ビジネスでのAI活用事例', color: '#95a5a6', icon: 'briefcase' },
  { name: 'Ethics & Society', slug: 'ai-ethics', description: 'AIの倫理と社会への影響',
color: '#8e44ad', icon: 'balance-scale' }
];

try {
  for (const category of categories) {
    await Category.findOneAndUpdate(
      { slug: category.slug },
      category,
      { upsert: true, new: true }
    );
  }
  console.log('Categories seeded successfully');
} catch (error) {
  console.error('Error seeding categories:', error);
}
};

module.exports = connectDB;

Do you want to create database.js?
> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
    
```

```

required: true,
trim: true,
maxlength: 200
},
content: {
... +84 lines (ctrl+r to expand)

```

- データベース設定ファイルを作成します:
- Write(backend/src/config/database.js)
  - Wrote 49 lines to backend/src/config/database.js
- 環境設定ファイルを作成します:
- Write(backend/.env)

```

Create file

backend/.env

NODE_ENV=development
PORT=5000
MONGODB_URI=mongodb://localhost:27017/ai-tech-platform
JWT_SECRET=your-super-secret-jwt-key-change-in-production
JWT_EXPIRE=7d

Do you want to create .env?
> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
    
```

## ■ ClaudeCodeでのAI駆動開発の例 その12

```

    if (!token) {
      return res.status(401).json({ message: 'アクセスが拒否されました。トークンがありません。' });
    }

    const decoded = jwt.verify(token, process.env.JWT_SECRET);
    const user = await User.findById(decoded.id);

    if (!user) {
      return res.status(401).json({ message: 'トークンが無効です。' });
    }

    req.user = user;
    next();
  } catch (error) {
    res.status(401).json({ message: 'トークンが無効です。' });
  }
};

const optionalAuth = async (req, res, next) => {
  try {
    const token = req.header('Authorization')?.replace('Bearer ', '');

    if (token) {
      const decoded = jwt.verify(token, process.env.JWT_SECRET);
      const user = await User.findById(decoded.id);
      req.user = user;
    }

    next();
  } catch (error) {
    next();
  }
};

module.exports = { auth, optionalAuth };
  
```

Do you want to create `auth.js`?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

```

    return res.status(401).json({
      message: 'このユーザー名は既に使用されています。'
    });
  }

  const user = await User.findByIdAndUpdate(
    userId,
    { username, bio, avatar },
    { new: true, runValidators: true }
  );

  res.json({
    success: true,
    user: {
      id: user._id,
      username: user.username,
      email: user.email,
      avatar: user.avatar,
      bio: user.bio
    }
  });
} catch (error) {
  res.status(400).json({
    message: 'プロフィール更新に失敗しました。',
    error: error.message
  });
}

module.exports = {
  register,
  login,
  getProfile,
  updateProfile
};
  
```

Do you want to create `authController.js`?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

# ■ ClaudeCodeでのAI駆動開発の例 その13

```

const toggleLike = async (req, res) => {
  try {
    const postId = req.params.id;
    const userId = req.user._id;

    const post = await Post.findById(postId);

    if (!post) {
      return res.status(404).json({
        message: '投稿が見つかりません。'
      });
    }

    await post.toggleLike(userId);

    res.json({
      success: true,
      likesCount: post.likes.length,
      isLiked: post.likes.some(like => like.user.toString() === userId.toString())
    });
  } catch (error) {
    res.status(500).json({
      message: 'いいねの処理に失敗しました。',
      error: error.message
    });
  }
};

module.exports = {
  getPosts,
  getPost,
  createPost,
  updatePost,
  deletePost,
  toggleLike
};
    
```

Do you want to create **postController.js**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

- カテゴリーコントローラーを作成します:
- **Write**(backend/src/controllers/categoryController.js)
  - ↳ Wrote 145 lines to backend/src/controllers/categoryController.js
- ルートファイルを作成します:
- **Write**(backend/src/routes/auth.js)

**Create file**

backend/src/routes/auth.js

```

const express = require('express');
const { register, login, getProfile, updateProfile } =
require('../controllers/authController');
const { auth } = require('../middleware/auth');

const router = express.Router();

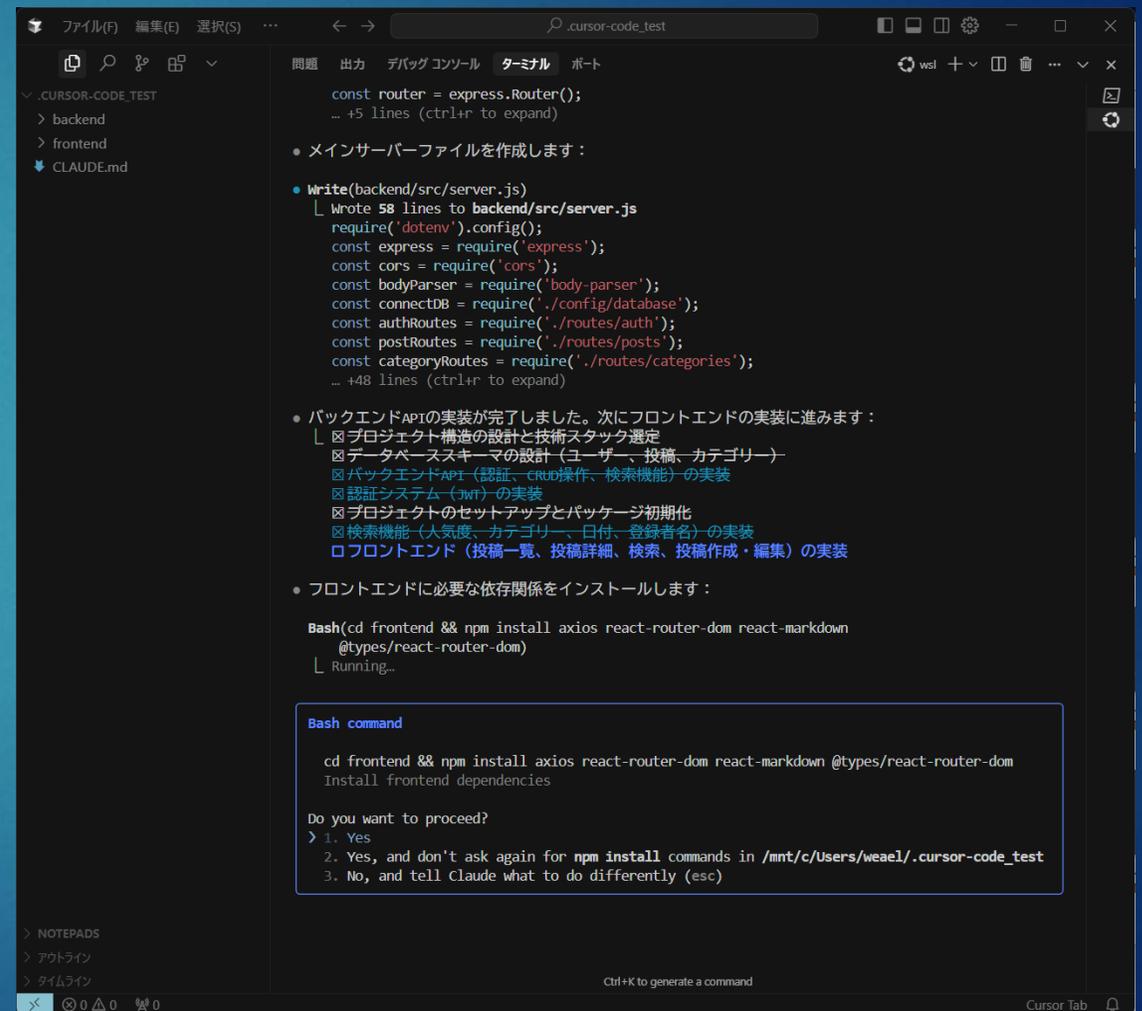
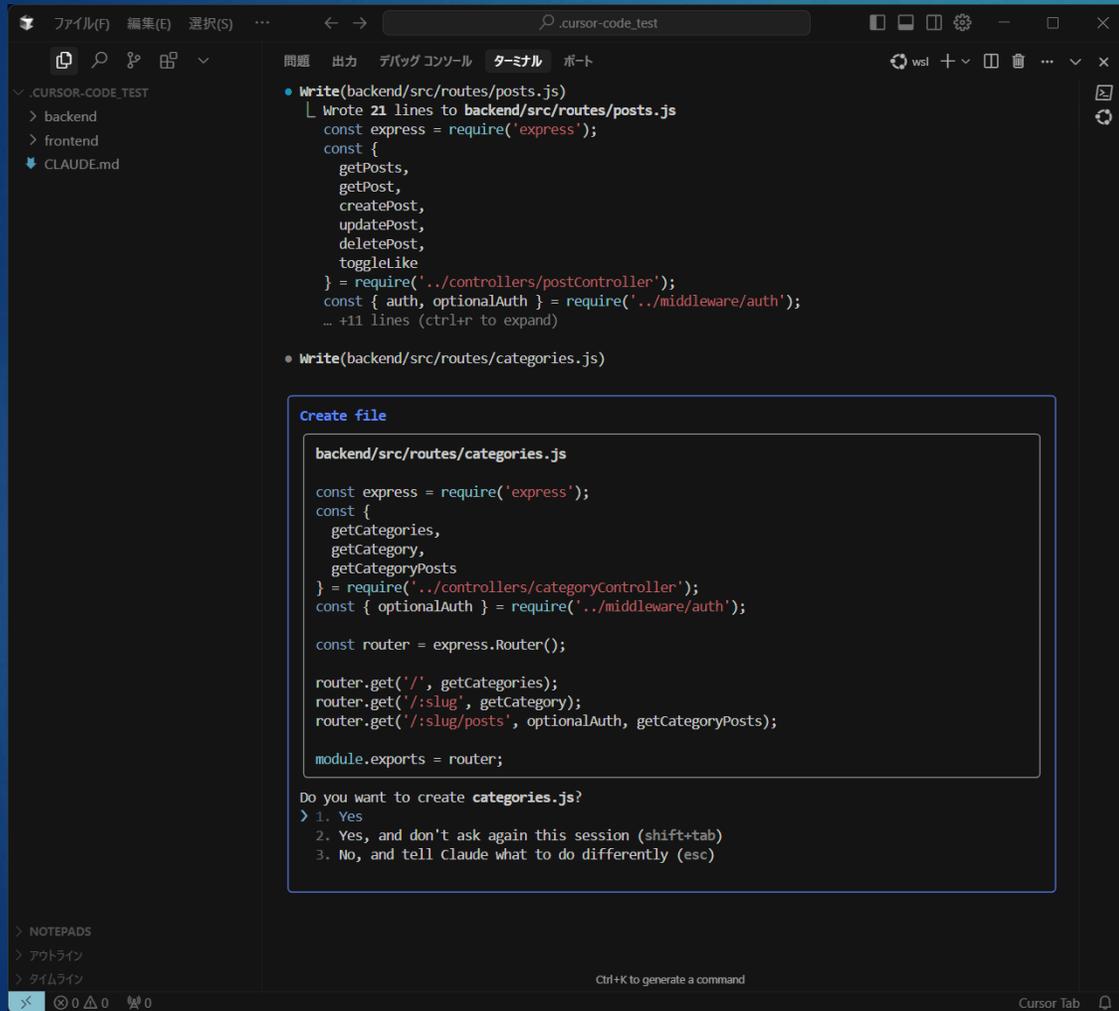
router.post('/register', register);
router.post('/login', login);
router.get('/profile', auth, getProfile);
router.put('/profile', auth, updateProfile);

module.exports = router;
    
```

Do you want to create **auth.js**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

# ■ ClaudeCodeでのAI駆動開発の例 その14



## ■ ClaudeCodeでのAI駆動開発の例 その15

```

setUser(userData);
localStorage.setItem('token', newToken);
axios.defaults.headers.common['Authorization'] = `Bearer ${newToken}`;
} catch (error: any) {
  throw new Error(error.response?.data?.message || '登録に失敗しました');
}
});

const logout = () => {
  setUser(null);
  setToken(null);
  localStorage.removeItem('token');
  delete axios.defaults.headers.common['Authorization'];
};

return (
  <AuthContext.Provider value={{
    user,
    token,
    login,
    register,
    logout,
    loading
  }}>
    {children}
  </AuthContext.Provider>
);
});

export const useAuth = () => {
  const context = useContext(AuthContext);
  if (context === undefined) {
    throw new Error('useAuth must be used within an AuthProvider');
  }
  return context;
};
    
```

Do you want to create AuthContext.tsx?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

```

const response = await axios.put(`/posts/${id}`, postData);
return response.data;
},

async deletePost(id: string): Promise<{ success: boolean; message: string }> {
  const response = await axios.delete(`/posts/${id}`);
  return response.data;
},

async toggleLike(id: string): Promise<{ success: boolean; likesCount: number; isLiked: boolean }> {
  const response = await axios.post(`/posts/${id}/like`);
  return response.data;
}
});

export const categoriesApi = {
  async getCategories(): Promise<CategoriesResponse> {
    const response = await axios.get('/categories');
    return response.data;
  },

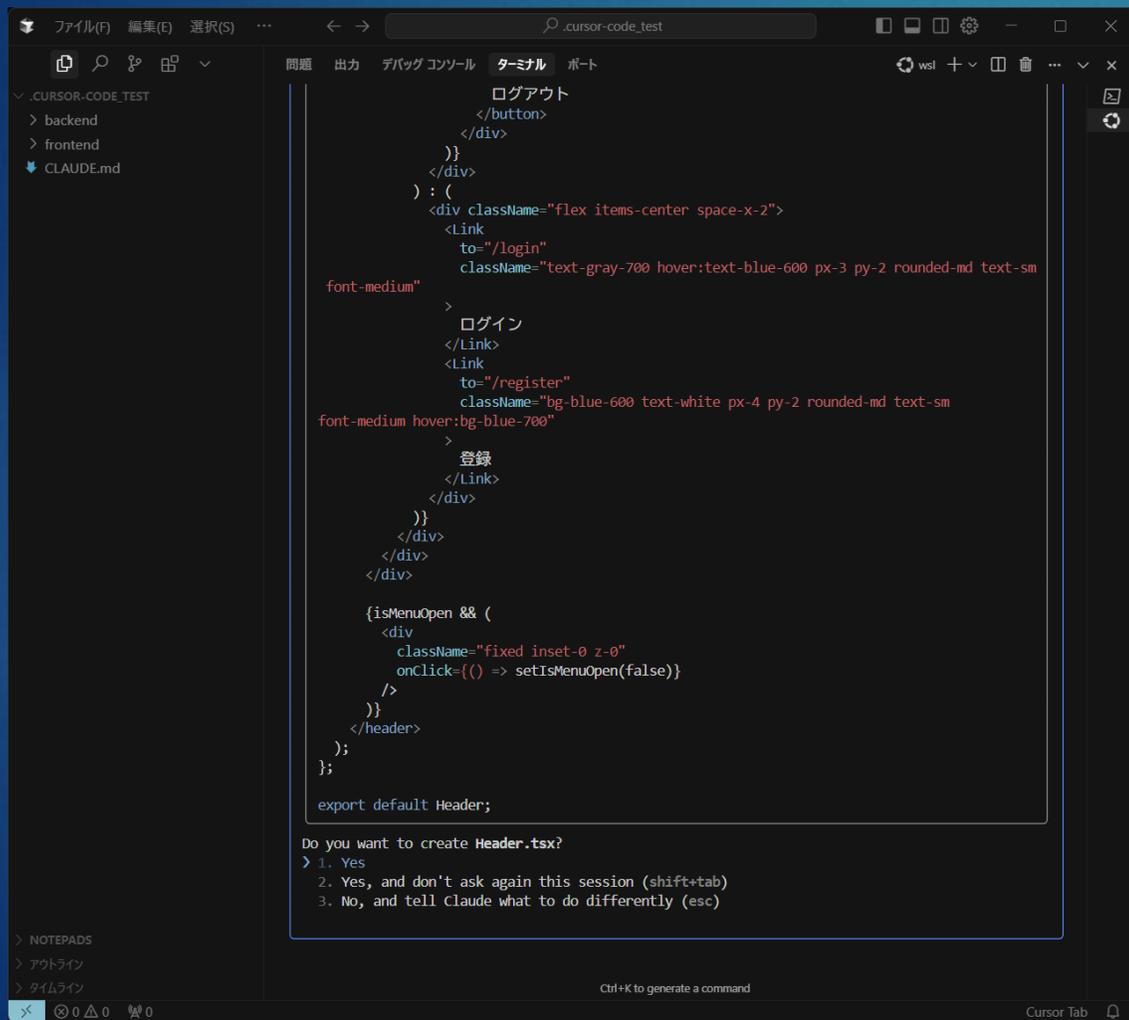
  async getCategory(slug: string): Promise<{ success: boolean; category: Category }> {
    const response = await axios.get(`/categories/${slug}`);
    return response.data;
  },

  async getCategoryPosts(slug: string, params: {
    page?: number;
    limit?: number;
    sort?: 'newest' | 'oldest' | 'popular' | 'views';
  }): Promise<PostsResponse & { category: Category }> {
    const response = await axios.get(`/categories/${slug}/posts`, { params });
    return response.data;
  }
};
    
```

Do you want to create api.ts?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

# ClaudeCodeでのAI駆動開発の例 その16



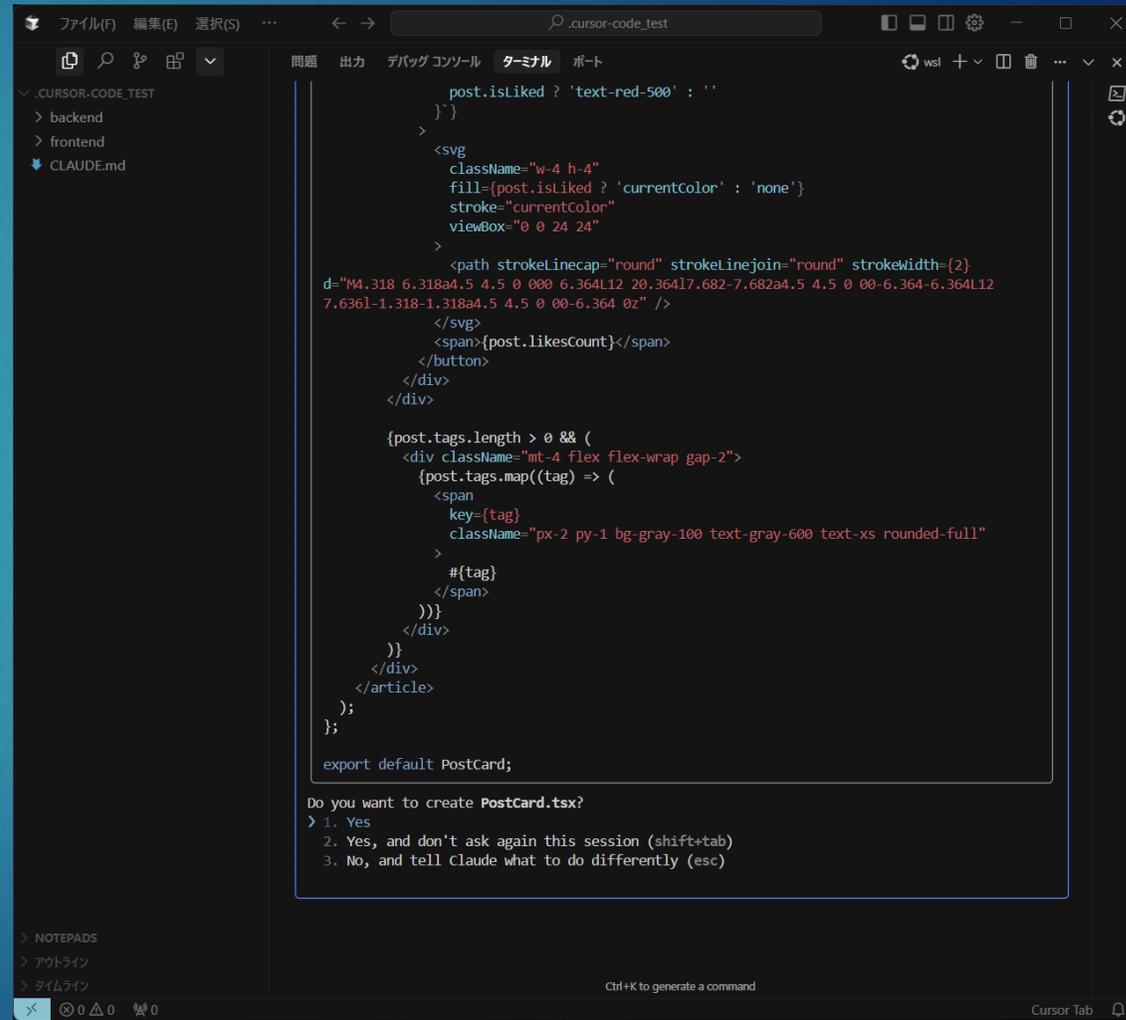
```

    </button>
  </div>
)}
</div>
): (
  <div className="flex items-center space-x-2">
    <Link
      to="/login"
      className="text-gray-700 hover:text-blue-600 px-3 py-2 rounded-md text-sm font-medium"
    >
      ログイン
    </Link>
    <Link
      to="/register"
      className="bg-blue-600 text-white px-4 py-2 rounded-md text-sm font-medium hover:bg-blue-700"
    >
      登録
    </Link>
  </div>
)}
</div>
</div>
{isMenuOpen && (
  <div
    className="fixed inset-0 z-0"
    onClick={() => setIsMenuOpen(false)}
  />
)}
</header>
);
};
export default Header;

```

Do you want to create Header.tsx?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)



```

    post.isLiked ? 'text-red-500' : ''
  }}
  <svg
    className="w-4 h-4"
    fill={post.isLiked ? 'currentColor' : 'none'}
    stroke="currentColor"
    viewBox="0 0 24 24"
  >
    <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2}
      d="M4.318 6.318a4.5 4.5 0 00 6.364 11.207 4.5 4.5 0 00 6.364-11.207z" />
  </svg>
  <span>{post.likescount}</span>
</button>
</div>
);
};
export default PostCard;

```

Do you want to create PostCard.tsx?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

## ■ ClaudeCodeでのAI駆動開発の例 その17

```

const page = i + 1;
return (
  <button
    key={page}
    onClick={() => handlePageChange(page)}
    className={`px-4 py-2 border rounded-lg ${
      currentPage === page
        ? 'bg-blue-600 text-white border-blue-600'
        : 'border-gray-300 hover:bg-gray-50'
    }`}
  >
    {page}
  </button>
);
))}

<button
  onClick={() => handlePageChange(currentPage + 1)}
  disabled={currentPage >= pagination.pages}
  className="px-4 py-2 border border-gray-300 rounded-lg
disabled:opacity-50 disabled:cursor-not-allowed hover:bg-gray-50"
  >
  次へ
</button>
</div>
))}
</main>
</div>
</div>
);
};

export default Home;
    
```

Do you want to create **Home.tsx**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

```

text-gray-700">
  パスワード
</label>
<input
  id="password"
  name="password"
  type="password"
  autoComplete="current-password"
  required
  className="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-lg
shadow-sm focus:outline-none focus:ring-blue-500 focus:border-blue-500"
  value={password}
  onChange={(e) => setPassword(e.target.value)}
  />
</div>
</div>

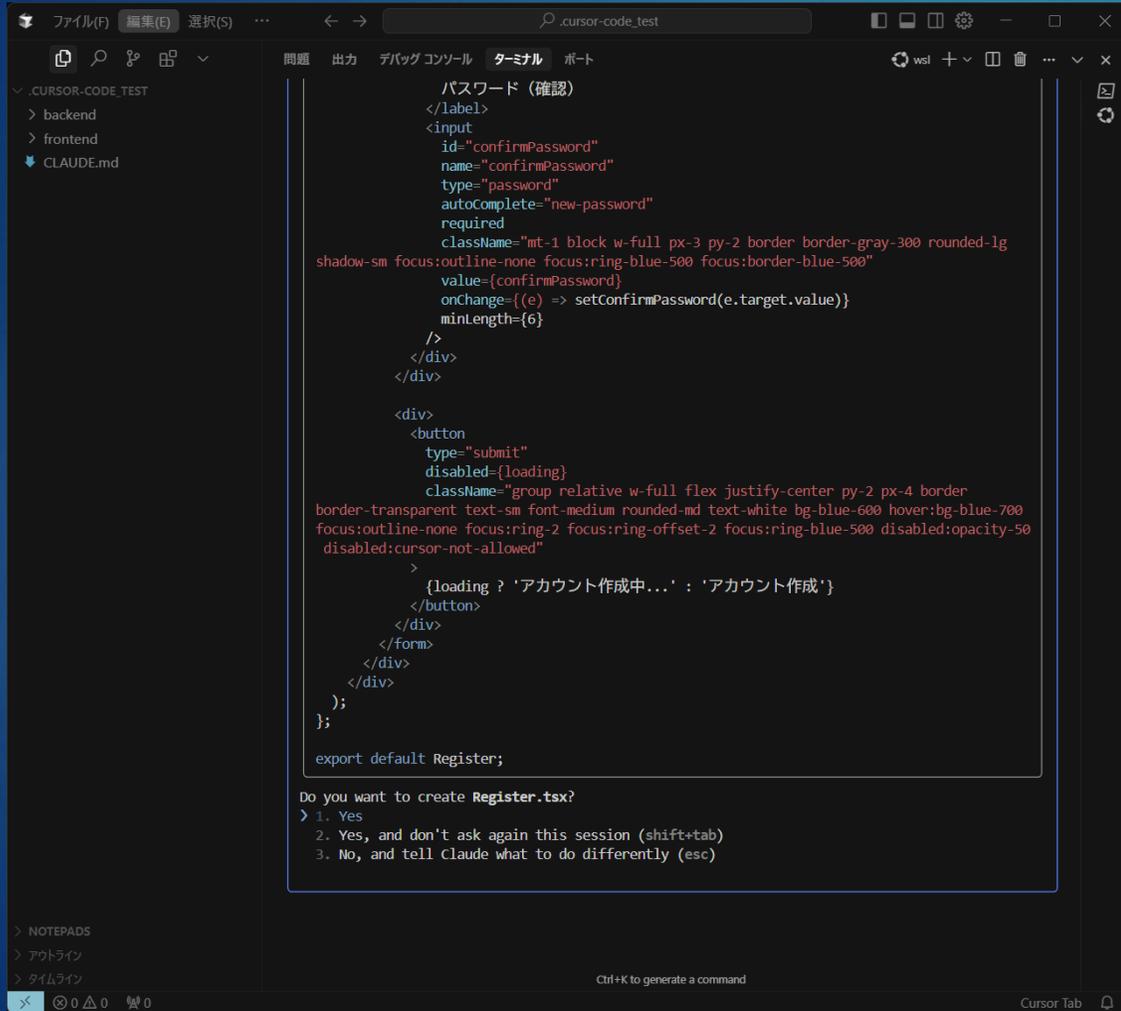
<div>
  <button
    type="submit"
    disabled={loading}
    className="group relative w-full flex justify-center py-2 px-4 border
border-transparent text-sm font-medium rounded-md text-white bg-blue-600 hover:bg-blue-700
focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-blue-500 disabled:opacity-50
disabled:cursor-not-allowed"
  >
    {loading ? 'ログイン中...' : 'ログイン'}
  </button>
</div>
</form>
</div>
);
};

export default Login;
    
```

Do you want to create **Login.tsx**?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

# ■ ClaudeCodeでのAI駆動開発の例 その18



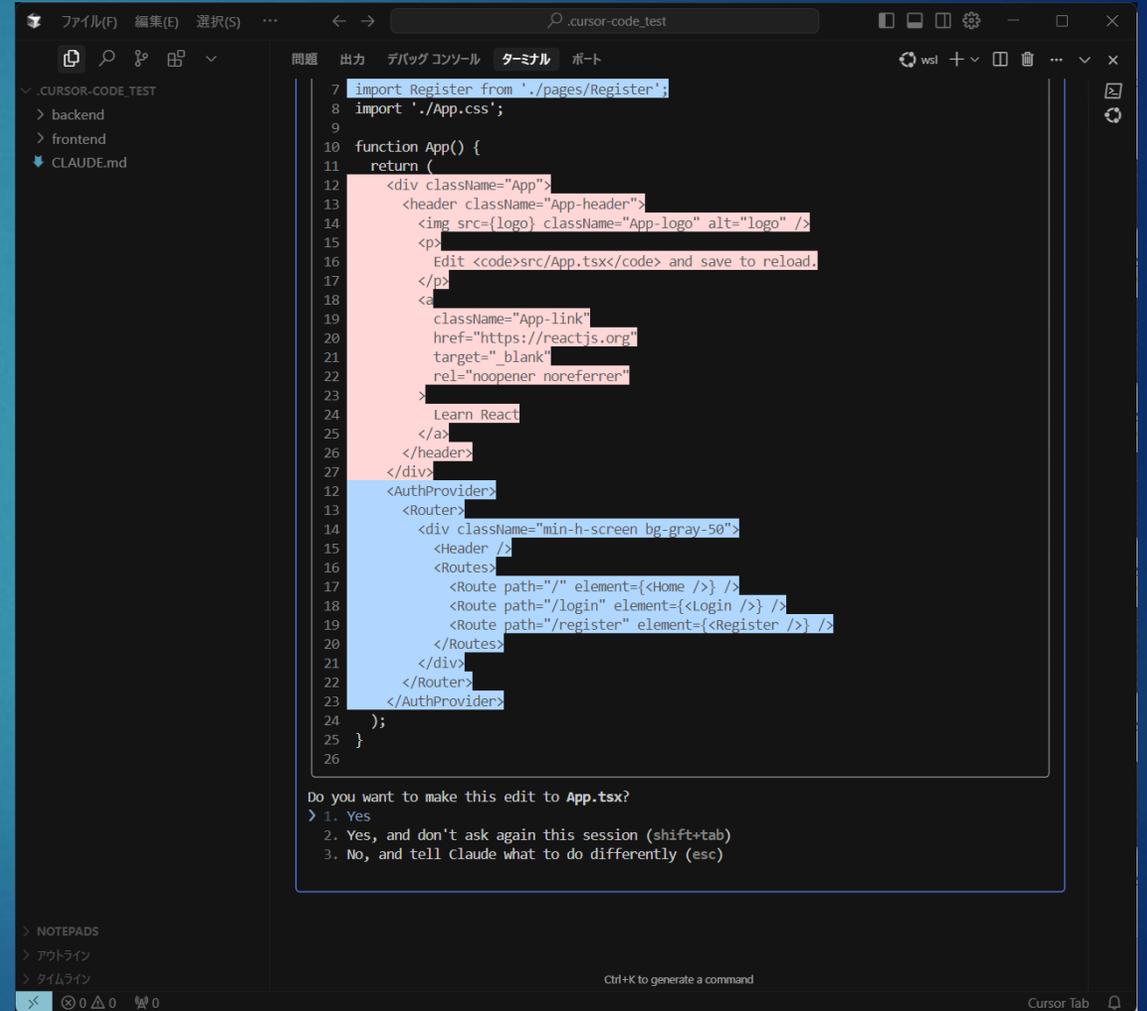
```

        /パスワード (確認)
        </label>
        <input
          id="confirmPassword"
          name="confirmPassword"
          type="password"
          autoComplete="new-password"
          required
          className="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-lg
          shadow-sm focus:outline-none focus:ring-blue-500 focus:border-blue-500"
          value={confirmPassword}
          onChange={(e) => setConfirmPassword(e.target.value)}
          minLength={6}
        />
      </div>
    </div>
    <div>
      <button
        type="submit"
        disabled={loading}
        className="group relative w-full flex justify-center py-2 px-4 border
        border-transparent text-sm font-medium rounded-md text-white bg-blue-600 hover:bg-blue-700
        focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-blue-500 disabled:opacity-50
        disabled:cursor-not-allowed"
      >
        {loading ? 'アカウント作成中...' : 'アカウント作成'}
      </button>
    </div>
  </form>
</div>
</div>
);
};

export default Register;
    
```

Do you want to create Register.tsx?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)



```

import Register from './pages/Register';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.tsx</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
    <AuthProvider>
      <Router>
        <div className="min-h-screen bg-gray-50">
          <Header />
          <Routes>
            <Route path="/" element={<Home />} />
            <Route path="/login" element={<Login />} />
            <Route path="/register" element={<Register />} />
          </Routes>
        </div>
      </Router>
    </AuthProvider>
  );
}
    
```

Do you want to make this edit to App.tsx?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

# ■ ClaudeCodeでのAI駆動開発の例 その19

```

18 <a
19   className="App-link"
20   href="https://reactjs.org"
21   target="_blank"
22   rel="noopener noreferrer"
23 >
24   Learn React
25 </a>
26 </header>
27 </div>
12 <AuthProvider>
13 <Router>
14 <div className="min-h-screen bg-gray-50">
15 <Header />
16 <Routes>
17 <Route path="/" element={<Home />} />
18 <Route path="/login" element={<Login />} />
19 <Route path="/register" element={<Register />} />
20 </Routes>
21 </div>
22 </Router>
23 </AuthProvider>
24 );
25 }
26

```

• Tailwind CSSを追加します :

```

Bash(cd frontend && npm install -D tailwindcss postcss autoprefixer && npx tailwindcss init -p)
└─ Running...

```

**Bash command**

```

cd frontend && npm install -D tailwindcss postcss autoprefixer && npx tailwindcss init -p
Install and initialize Tailwind CSS

```

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for similar commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)

```

23 >
24   Learn React
25 </a>
26 </header>
27 </div>
12 <AuthProvider>
13 <Router>
14 <div className="min-h-screen bg-gray-50">
15 <Header />
16 <Routes>
17 <Route path="/" element={<Home />} />
18 <Route path="/login" element={<Login />} />
19 <Route path="/register" element={<Register />} />
20 </Routes>
21 </div>
22 </Router>
23 </AuthProvider>
24 );
25 }
26

```

• Tailwind CSSを追加します :

```

Bash(cd frontend && npm install -D tailwindcss postcss autoprefixer && npx tailwindcss init -p)
└─ Error: /bin/bash: line 1: cd: frontend: No such file or directory

```

• 現在のディレクトリを確認してからTailwindをインストールします :

```

Bash(pwd && ls -la)
└─ Running...

```

**Bash command**

```

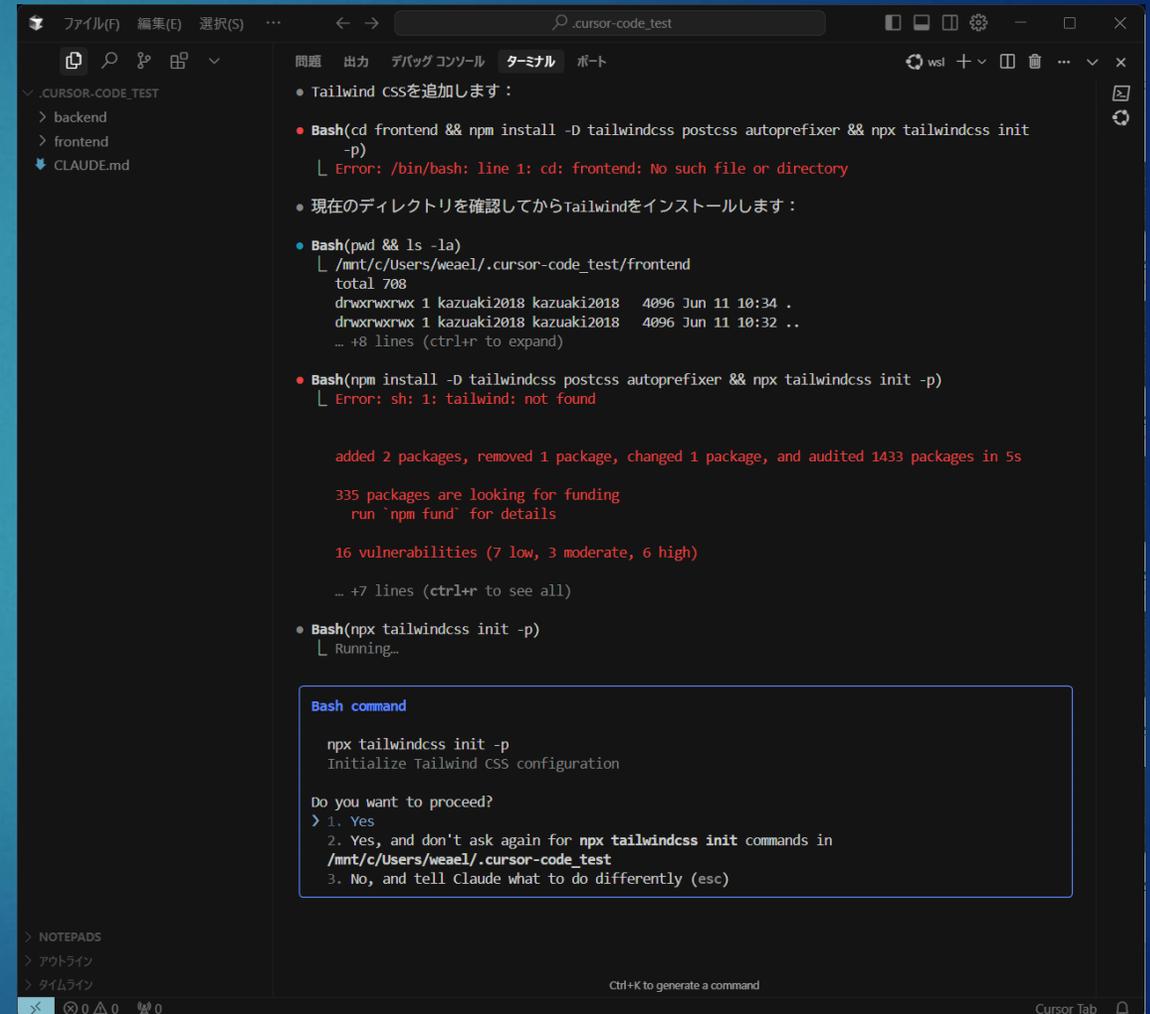
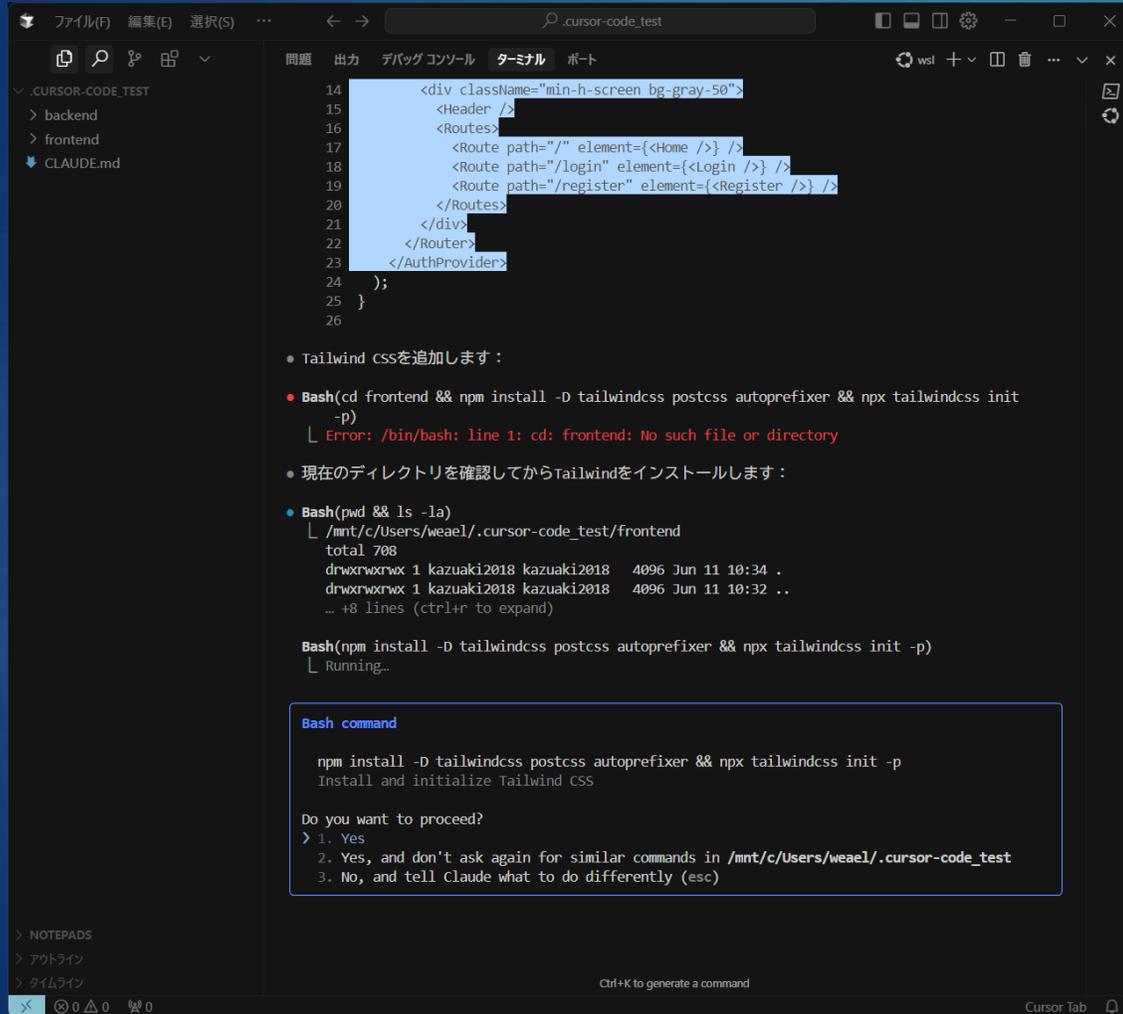
pwd && ls -la
Check current location

```

Do you want to proceed?

1. Yes
2. Yes, and don't ask again for ls commands in /mnt/c/Users/weael/.cursor-code\_test
3. No, and tell Claude what to do differently (esc)

# ■ ClaudeCodeでのAI駆動開発の例 その20



# ■ ClaudeCodeでのAI駆動開発の例 その21

```

    ファイル(F) 編集(E) 選択(S) ...  ← →  🔍 .cursor-code_test  🖱️ 🗑️ 🔄
    問題 出力 デバッグコンソール ターミナル ポート
    〻 wsl + 🗑️ 🗑️ ... 🔄
    > _CURSOR-CODE_TEST
    > backend
    > frontend
    📄 CLAUDE.md

    -p)
    | /bin/bash: line 1: cd: frontend: No such file or directory

    ● 現在のディレクトリを確認してからTailwindをインストールします：

    ● Bash(pwd && ls -la)
    | /mnt/c/Users/weael/.cursor-code_test/frontend
    | total 708
    | drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:34 .
    | drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:32 ..
    | ... +8 lines (ctrl+r to expand)

    ● Bash(npm install -D tailwindcss postcss autoprefixer && npx tailwindcss init -p)
    | Error: sh: 1: tailwind: not found

    added 2 packages, removed 1 package, changed 1 package, and audited 1433 packages in 5s

    335 packages are looking for funding
    run `npm fund` for details

    16 vulnerabilities (7 low, 3 moderate, 6 high)
    ... +7 lines (ctrl+r to see all)

    ● Bash(npx tailwindcss init -p)
    | Error: sh: 1: tailwind: not found

    ● Bash(./node_modules/.bin/tailwindcss init -p)
    | Running...

    Bash command
    ./node_modules/.bin/tailwindcss init -p
    Initialize Tailwind CSS using local binary

    Do you want to proceed?
    > 1. Yes
    2. Yes, and don't ask again for ./node_modules/.bin/tailwindcss commands in
    /mnt/c/Users/weael/.cursor-code_test
    3. No, and tell Claude what to do differently (esc)

    > NOTEPADS
    > アウトライン
    > タイムライン
    Ctrl+K to generate a command
    Cursor Tab
  
```

```

    ファイル(F) 編集(E) 選択(S) ...  ← →  🔍 .cursor-code_test  🖱️ 🗑️ 🔄
    問題 出力 デバッグコンソール ターミナル ポート
    〻 wsl + 🗑️ 🗑️ ... 🔄
    > _CURSOR-CODE_TEST
    > backend
    > frontend
    📄 CLAUDE.md

    -p)
    | /bin/bash: line 1: cd: frontend: No such file or directory

    ● 現在のディレクトリを確認してからTailwindをインストールします：

    ● Bash(pwd && ls -la)
    | /mnt/c/Users/weael/.cursor-code_test/frontend
    | total 708
    | drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:34 .
    | drwxrwxrwx 1 kazuaki2018 kazuaki2018 4096 Jun 11 10:32 ..
    | ... +8 lines (ctrl+r to expand)

    ● Bash(npm install -D tailwindcss postcss autoprefixer && npx tailwindcss init -p)
    | Error: sh: 1: tailwind: not found

    added 2 packages, removed 1 package, changed 1 package, and audited 1433 packages in 5s

    335 packages are looking for funding
    run `npm fund` for details

    16 vulnerabilities (7 low, 3 moderate, 6 high)
    ... +7 lines (ctrl+r to see all)

    ● Bash(npx tailwindcss init -p)
    | Error: sh: 1: tailwind: not found

    ● Bash(./node_modules/.bin/tailwindcss init -p)
    | Running...

    Bash command
    ./node_modules/.bin/tailwindcss init -p
    Initialize Tailwind CSS using local binary

    Do you want to proceed?
    > 1. Yes
    2. Yes, and don't ask again for ./node_modules/.bin/tailwindcss commands in
    /mnt/c/Users/weael/.cursor-code_test
    3. No, and tell Claude what to do differently (esc)

    > NOTEPADS
    > アウトライン
    > タイムライン
    Ctrl+K to generate a command
    Cursor Tab
  
```

## ■ ClaudeCodeでのAI駆動開発の例 その22

Terminal output:

```

... +7 lines (ctrl+r to see all)

• Bash(npx tailwindcss init -p)
  └─ Error: sh: 1: tailwind: not found

• Bash(./node_modules/.bin/tailwindcss init -p)
  └─ Error: /bin/bash: line 1: ./node_modules/.bin/tailwindcss: No such file or directory

• パッケージが正しくインストールされているか確認し、設定ファイルを手動で作成します：

• Write(tailwind.config.js)
  └─ Wrote 10 lines to tailwind.config.js
    └─ /** @type {import('tailwindcss').Config} */
        module.exports = {
          content: [
            "./src/**/*.{js,jsx,ts,tsx}",
          ],
          theme: {
            extend: {},
          },
          plugins: [],
        }

• Write(postcss.config.js)
  
```

Create file prompt:

```

Create file

postcss.config.js

module.exports = {
  plugins: {
    tailwindcss: {},
    autoprefixer: {},
  },
}

Do you want to create postcss.config.js?
> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
  
```

Terminal output:

```

tailwindcss: {},
autoprefixer: {},
},
}

• CSS ファイルを更新してTailwindディレクティブを追加します：

• Read(src/index.css)
  └─ Read 14 lines (ctrl+r to expand)

• Update(src/index.css)
  
```

Edit file prompt:

```

Edit file

src/index.css

1 @tailwind base;
2 @tailwind components;
3 @tailwind utilities;
4
5 body {
6   margin: 0;
7   font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
...
11 font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
12   monospace;
13 }
14
15 .line-clamp-3 {
16   display: -webkit-box;
17   -webkit-line-clamp: 3;
18   -webkit-box-orient: vertical;
19   overflow: hidden;
20 }

Do you want to make this edit to index.css?
> 1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)
  
```

# ClaudeCodeでのAI駆動開発の例 その23

```

74 - GET `/api/auth/profile` - Get user profile
75 - PUT `/api/auth/profile` - Update user profile
76
77 ### Posts
78 - GET `/api/posts` - Get posts with filtering/search
79 - GET `/api/posts/:id` - Get single post
80 - POST `/api/posts` - Create post (auth required)
81 - PUT `/api/posts/:id` - Update post (auth required)
82 - DELETE `/api/posts/:id` - Delete post (auth required)
83 - POST `/api/posts/:id/like` - Toggle like (auth required)
84
85 ### Categories
86 - GET `/api/categories` - Get all categories
87 - GET `/api/categories/:slug` - Get category details
88 - GET `/api/categories/:slug/posts` - Get posts by category
89
90 ## Environment Setup
91
92 Create `backend/.env`:
93 ```
94 NODE_ENV=development
95 PORT=5000
96 MONGODB_URI=mongodb://localhost:27017/ai-tech-platform
97 JWT_SECRET=your-super-secret-jwt-key-change-in-production
98 JWT_EXPIRE=7d
99 ```
100
101 ## Database
102
103 MongoDB collections:
104 - `users` - User accounts and profiles
105 - `posts` - AI technology articles
106 - `categories` - Predefined categories with AI focus
107
108 Categories are automatically seeded on database connection.
109 \ No newline at end of file
    
```

Do you want to make this edit to CLAUDE.md?

1. Yes
2. Yes, and don't ask again this session (shift+tab)
3. No, and tell Claude what to do differently (esc)

• AI技術情報プラットフォームが完成しました！

**実装内容**

バックエンド (Node.js + Express + MongoDB)

- 認証システム: JWTトークンベースの認証
- データベースモデル: User, Post, Category
- API: 完全なCRUD操作と検索機能
- カテゴリー: AI関連の10個のカテゴリー (機械学習、ディープラーニング等)

フロントエンド (React + TypeScript + Tailwind CSS)

- レスポンシブデザイン: モバイル対応
- 認証機能: ログイン・登録・ログアウト
- 投稿機能: 記事の閲覧・作成・編集・削除 (作成者のみ)
- 検索・フィルタ: カテゴリー、人気度、日付、作成者別

**主要機能**

- ユーザー登録・ログイン (JWT認証)
- 投稿のCRUD操作 (作成者のみ編集・削除可能)
- カテゴリー選択システム
- 高度な検索機能 (人気度、カテゴリー、日付、作成者名)
- いいね機能
- 閲覧数カウント
- レスポンシブデザイン

**起動方法**

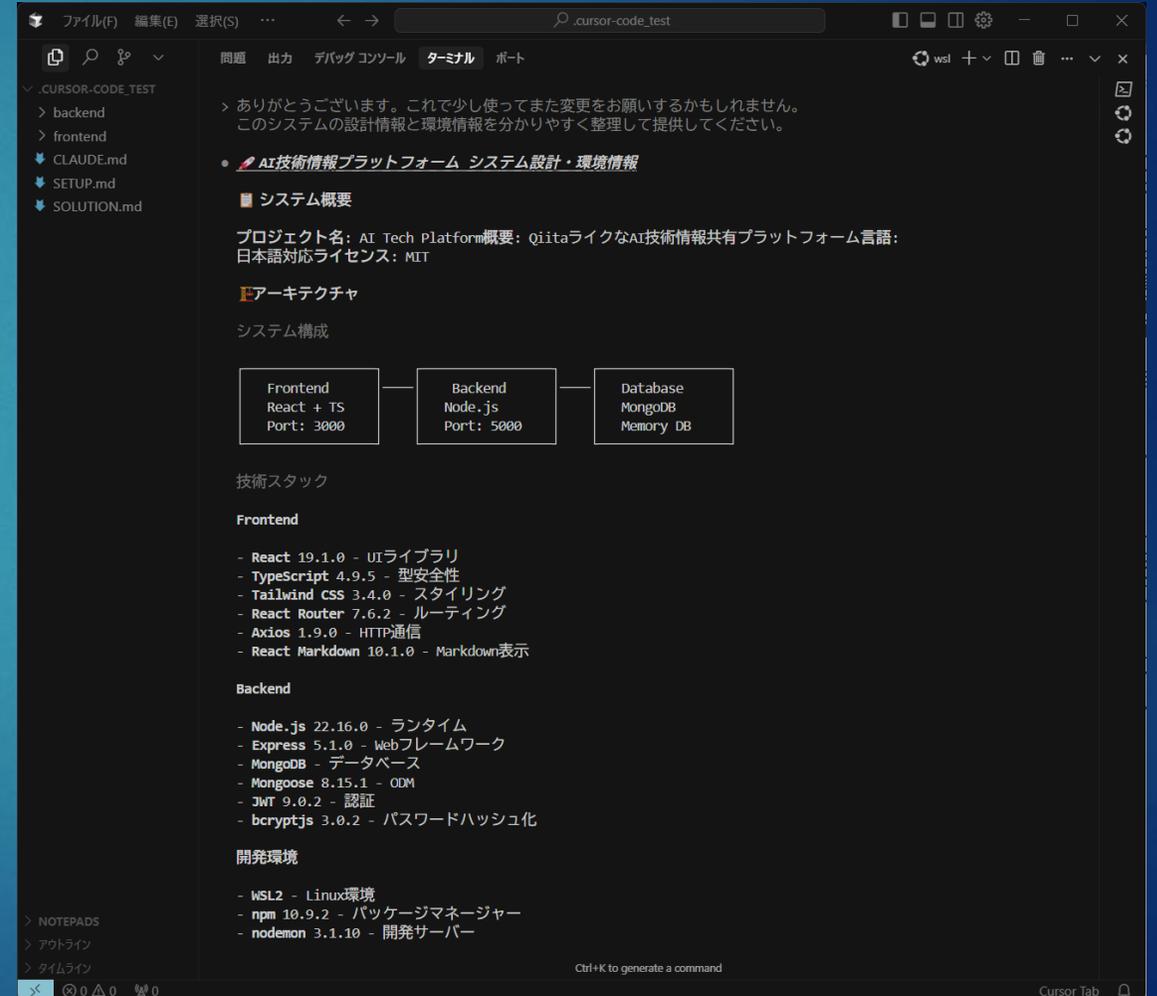
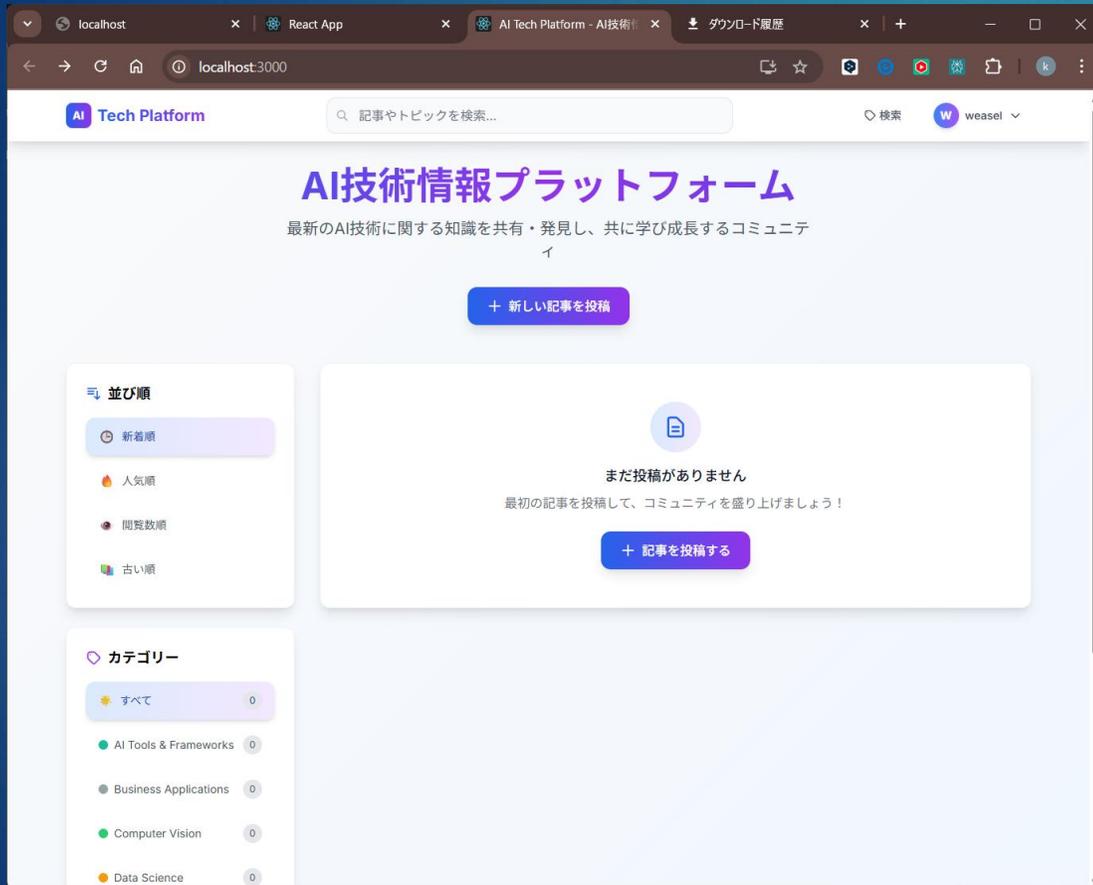
1. MongoDBを起動
2. バックエンド: `cd backend && npm run dev`
3. フロントエンド: `cd frontend && npm start`

Quitaライクな機能を完備したAI技術情報プラットフォームが完成しました。

>

? for shortcuts ✓ Update installed · Restart to apply

# ■ ClaudeCodeでAI駆動開発した成果物 その1



## ■ ClaudeCodeでAI駆動開発した成果物 その2

開発環境

- WSL2 - Linux環境
- npm 10.9.2 - パッケージマネージャー
- nodemon 3.1.10 - 開発サーバー

プロジェクト構造

```

/mnt/c/Users/weael/.cursor-code_test/
├── backend/ # バックエンドAPI
│   ├── src/
│   │   ├── models/ # データモデル
│   │   │   ├── User.js # ユーザーモデル
│   │   │   ├── Post.js # 投稿モデル
│   │   │   └── Category.js # カテゴリーモデル
│   │   ├── controllers/ # ビジネスロジック
│   │   │   ├── authController.js
│   │   │   ├── postController.js
│   │   │   └── categoryController.js
│   │   ├── routes/ # APIルート
│   │   │   ├── auth.js
│   │   │   ├── posts.js
│   │   │   └── categories.js
│   │   ├── middleware/ # ミドルウェア
│   │   │   └── auth.js
│   │   ├── config/ # 設定
│   │   │   └── database.js # DB設定
│   │   ├── server.js # メインサーバー
│   │   ├── package.json
│   │   └── .env # 環境変数
│   └── frontend/ # フロントエンドSPA
│       ├── public/
│       │   └── index.html # HTMLテンプレート
│       └── src/
│           ├── pages/ # ページコンポーネント
│           │   ├── Home.tsx # ホーム画面
│           │   ├── Login.tsx # ログイン
│           │   ├── Register.tsx # ユーザー登録
│           │   ├── PostDetail.tsx # 投稿詳細
│           │   ├── CreatePost.tsx # 投稿作成
│           │   ├── EditPost.tsx # 投稿編集
│           │   └── Search.tsx # 検索画面
│           └── components/ # 再利用コンポーネント
│               ├── Header.tsx # ヘッダー
│               └── PostCard.tsx # 投稿カード

```

```

├── components/ # 再利用コンポーネント
│   ├── Header.tsx # ヘッダー
│   └── PostCard.tsx # 投稿カード
├── contexts/ # React Context
│   └── AuthContext.tsx # 認証状態管理
├── services/ # API通信
│   ├── api.ts # API関数・型定義
│   ├── App.tsx # メインアプリ
│   ├── index.tsx # エントリーポイント
│   └── index.css # グローバルスタイル
├── package.json
├── tailwind.config.js # Tailwind設定
├── postcss.config.js # PostCSS設定
├── CLAUDE.md # プロジェクト説明
├── SETUP.md # セットアップガイド
└── SOLUTION.md # 解決済み問題

データベース設計

User (ユーザー)

{
  _id: ObjectId,
  username: String (unique, 3-20文字),
  email: String (unique, lowercase),
  password: String (bcrypt暗号化),
  avatar: String (URL, optional),
  bio: String (最大500文字),
  isActive: Boolean (default: true),
  createdAt: Date,
  updatedAt: Date
}

Post (投稿)

{
  _id: ObjectId,
  title: String (最大200文字),
  content: String (Markdown),
  excerpt: String (最大500文字, 自動生成),
  author: ObjectId (User参照),
  category: ObjectId (Category参照),
  tags: [String],
  likes: [{
    user: ObjectId (User参照),

```

## ■ ClaudeCodeでAI駆動開発した成果物 その3

category (カテゴリー)

```
{
  _id: ObjectId,
  name: String (unique),
  slug: String (unique, lowercase),
  description: String (最大200文字),
  color: String (カラーコード),
  icon: String,
  isActive: Boolean (default: true),
  createdAt: Date,
  updatedAt: Date
}
```

API エンドポイント

認証 (/api/auth)

メソッド	エンドポイント	説明	認証
POST	/register	ユーザー登録	×
POST	/login	ログイン	×
GET	/profile	プロフィール取得	✓
PUT	/profile	プロフィール更新	✓

投稿 (/api/posts)

メソッド	エンドポイント	説明	認証
GET	/	投稿一覧 (検索・フィルタ対応)	×
GET	/:id	投稿詳細	×
POST	/	投稿作成	✓
PUT	/:id	投稿更新 (作成者のみ)	✓
DELETE	/:id	投稿削除 (作成者のみ)	✓
POST	/:id/like	いいね切り替え	✓

カテゴリー (/api/categories)

メソッド	エンドポイント	説明	認証
GET	/	カテゴリー一覧	×
GET	/:slug	カテゴリー詳細	×
GET	/:slug/posts	カテゴリー別投稿一覧	×

主要機能

主要機能

- ✓ 実装済み機能

- ユーザー認証**
  - ユーザー登録・ログイン・ログアウト
  - JWT トークン認証
  - プロフィール管理
- 投稿管理**
  - 記事作成・編集・削除 (権限制御)
  - Markdown対応
  - 下書き機能
  - タグ付け
- カテゴリーシステム**
  - 10種類のAI専門カテゴリー
  - カテゴリー別フィルタリング
  - カラーコード付きバッジ
- 検索・フィルタリング**
  - キーワード検索 (タイトル・内容・タグ)
  - カテゴリーフィルター
  - 投稿者フィルター
  - 日付範囲フィルター
  - 並び順 (新着・人気・閲覧数・古い順)
- ソーシャル機能**
  - いいね機能
  - 閲覧数カウント
  - 人気度計算
- UI/UX**
  - レスポンシブデザイン
  - モダンなインターフェース
  - アニメーション効果
  - ダークモード対応準備

AI関連カテゴリー

- Machine Learning - 機械学習
- Deep Learning - ディープラーニング
- Natural Language Processing - 自然言語処理
- Computer Vision - コンピュータビジョン
- Data Science - データサイエンス
- Reinforcement Learning - 強化学習
- AI Tools & Frameworks - AIツール・フレームワーク
- Research Papers - 研究論文
- Business Applications - ビジネス応用
- Ethics & Society - 倫理・社会

## ■ ClaudeCodeでAI駆動開発した成果物 その4

```

# WSL2内
Backend: http://localhost:5000
Frontend: http://localhost:3000

# Windows側 (WSL2 IP: 172.29.237.226)
Backend: http://172.29.237.226:5000
Frontend: http://172.29.237.226:3000

環境変数 (backend/.env)

NODE_ENV=development
PORT=5000
MONGODB_URI=mongodb://localhost:27017/ai-tech-platform
JWT_SECRET=your-super-secret-jwt-key-change-in-production
JWT_EXPIRE=7d

🔗 拡張可能性

将来の機能拡張

- ✨ コメント機能
- ✨ フォロー機能
- ✨ 通知システム
- ✨ 画像アップロード
- ✨ タグ管理システム
- ✨ 管理者機能
- ✨ API レート制限
- ✨ 全文検索 (Elasticsearch)

スケーラビリティ

- 📦 Redis キャッシュ
- 📦 CDN 対応
- 📦 ロードバランサー
- 📦 マイクロサービス化

このシステムは完全に動作し、本格的なAI技術情報共有プラットフォームとして利用可能です！

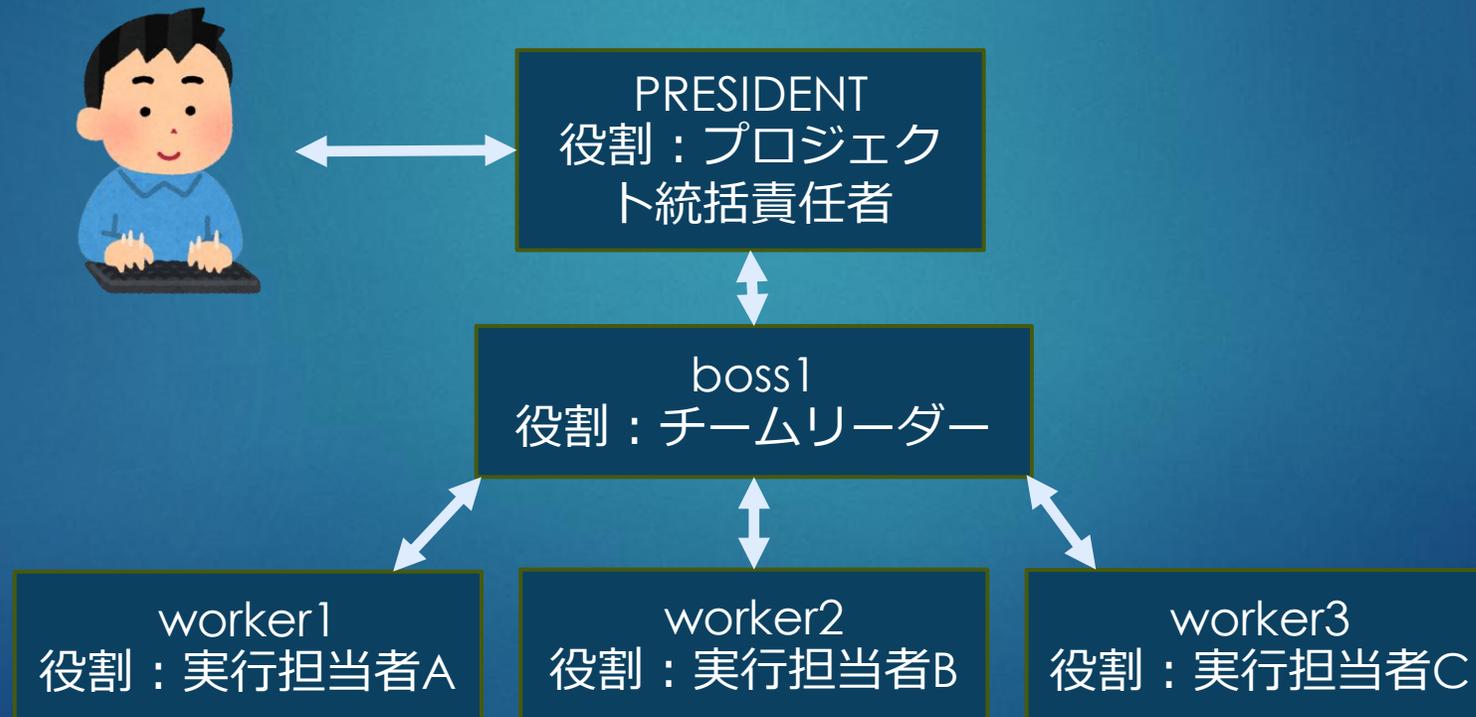
> |
    
```

? for shortcuts      context left until auto-compact: 27%

Cursor Tab

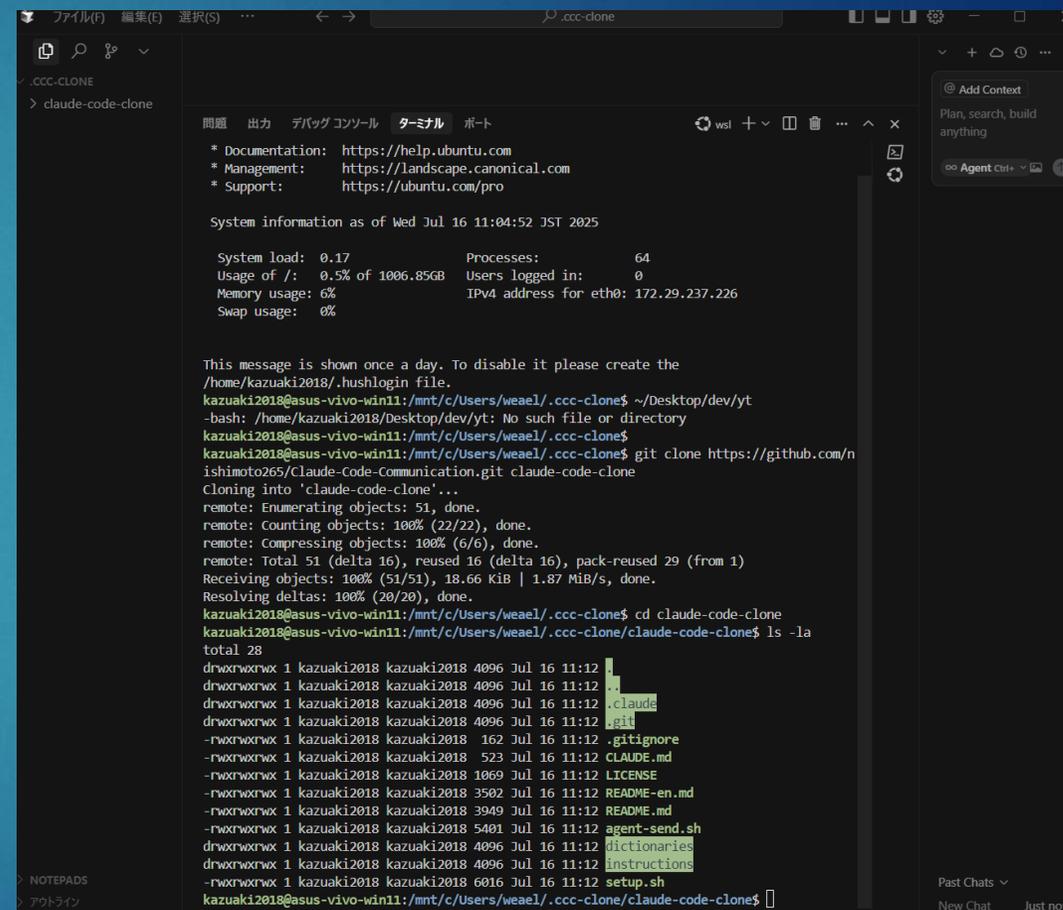
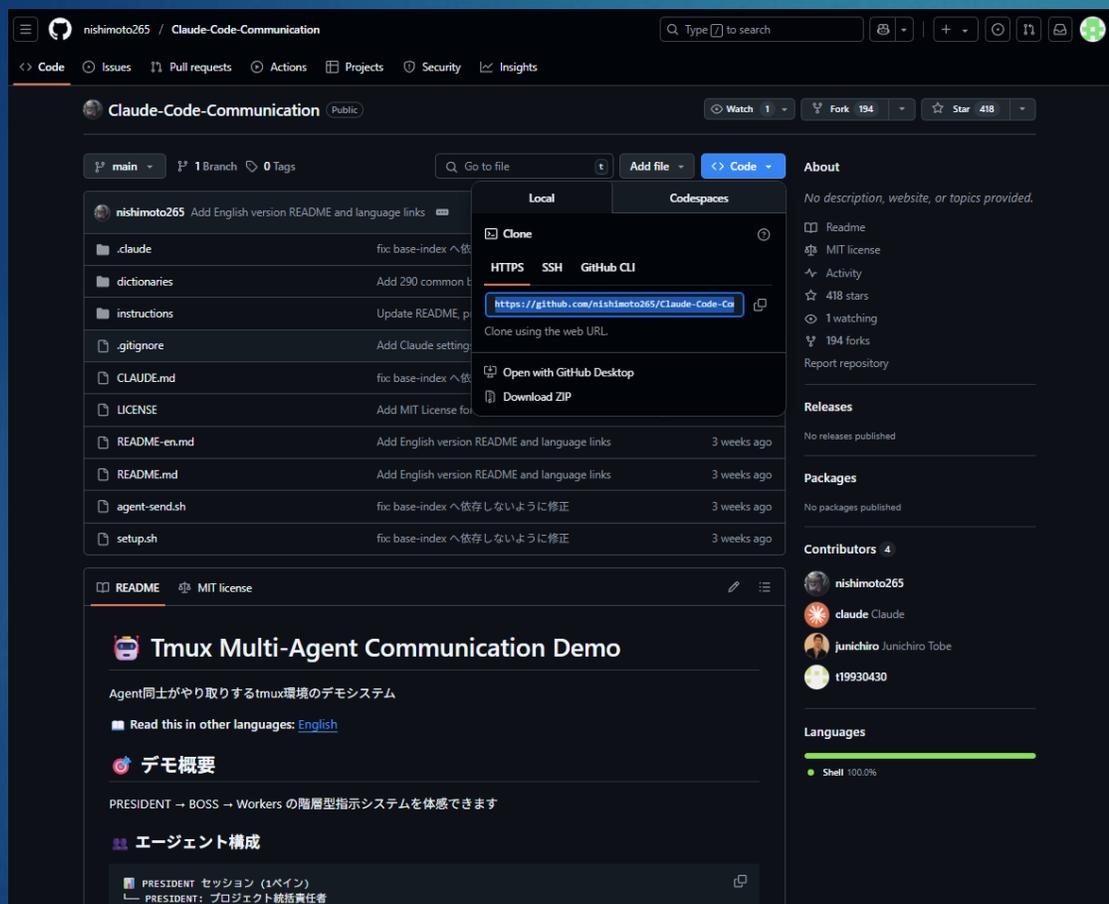
## 4. エージェントAIツールによるAI駆動開発 数人によるアジャイル開発を置き換え

- ▶ 複数の ClaudeCode でチーム開発をAIに代行させることが可能になりつつあります
  - ▶ tmuxでClaudeCodeを複数動かしてチーム開発を代行するいろいろな試みが公開され始めています



▶ 西本氏がGitHub上でtmuxを使ったデモを公開しています

▶ "nishimoto265/Claude-Code-Communication"をクローンしてください



<https://github.com/nishimoto265/Claude-Code-Communication>

- `git clone https://github.com/nishimoto265/Claude-Code-Communication` を実行します

- ▶ tmux 環境を構築します
- ▶ multiagent のセッションをアタッチします

```

kazuaki12018@asus-vivo-win11:/mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ ./setup.sh
=====
[INFO] 🧹 既存セッションクリーンアップ開始...
[INFO] multiagentセッションは存在していませんでした
[INFO] presidentセッションは存在していませんでした
[INFO] 既存の完了ファイルをクリア
[SUCCESS] ✅ クリーンアップ完了

[INFO] 🏗️ multiagentセッション作成開始 (4ペイン)...
[INFO] セッション作成中...
[INFO] セッション作成成功
[INFO] グリッド作成中...
[INFO] 水平分割実行中...
[INFO] 左側垂直分割実行中...
[INFO] 右側垂直分割実行中...
[INFO] ペイン配置確認中...
[INFO] 作成されたペイン数: 4
[INFO] ペイン番号取得中...
[INFO] 検出されたペイン: %0 %1 %2 %3
[INFO] ペインタイトル設定中...
[INFO] 設定中: boss1 (%0)
[INFO] 設定中: worker1 (%1)
[INFO] 設定中: worker2 (%2)
[INFO] 設定中: worker3 (%3)
[SUCCESS] ✅ multiagentセッション作成完了

[INFO] 👑 presidentセッション作成開始...
[SUCCESS] ✅ presidentセッション作成完了

[INFO] 🔍 環境確認中...

📄 セットアップ結果:
=====
📄 Tmux Sessions:
multiagent: 1 windows (created Sun Jul 20 15:18:41 2025)
president: 1 windows (created Sun Jul 20 15:18:41 2025)

📄 ペイン構成:
multiagentセッション (4ペイン) :
    Pane %0: boss1
    Pane %2: worker2
    Pane %1: worker1
    Pane %3: worker3

presidentセッション (1ペイン) :
    Pane: PRESIDENT (プロジェクト統括)

[SUCCESS] 🎉 Demo環境セットアップ完了!

$
    
```

- ./setup.sh を実行します

```

📄 セットアップ結果:
=====
📄 Tmux Sessions:
multiagent: 1 windows (created Sun Jul 20 15:18:41 2025)
president: 1 windows (created Sun Jul 20 15:18:41 2025)

📄 ペイン構成:
multiagentセッション (4ペイン) :
    Pane %0: boss1
    Pane %2: worker2
    Pane %1: worker1
    Pane %3: worker3

presidentセッション (1ペイン) :
    Pane: PRESIDENT (プロジェクト統括)

[SUCCESS] 🎉 Demo環境セットアップ完了!

📄 次のステップ:
1. 🔗 セッションアタッチ:
   tmux attach-session -t multiagent # マルチエージェント確認
   tmux attach-session -t president # プレジデント確認

2. 🖥️ Claude Code起動:
   # 手順1: President認証
   tmux send-keys -t president 'claude' C-m
   # 手順2: 認証後、multiagent一括起動
   # 各ペインのIDを使用してclaudeを起動
   tmux list-panes -t multiagent:agents -F '#{pane_id}' | while read pane; do
     tmux send-keys -t "$pane" 'claude' C-m
   done

3. 📄 指示書確認:
   PRESIDENT: instructions/president.md
   boss1: instructions/boss.md
   worker1,2,3: instructions/worker.md
   システム構造: CLAUDE.md

4. 🎬 デモ実行: PRESIDENTに「あなたはpresidentです。指示書に従って」と入力
kazuaki12018@asus-vivo-win11:/mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ tmux attach-session -t multiagent
    
```

- tmux attach-session -t multiagent を実行します

▶ 4つの仮想セッションがアタッチされました

▶ president のセッションをアタッチし、ClaudeCode を完全自律モードで起動します

```

kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ export PS1='(\[\033[1;31m\]\boss1\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
(boss1) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=== boss1 エージェント ==='
=== boss1 エージェント ===
(boss1) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$

kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ export PS1='(\[\033[1;34m\]\worker1\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
(worker1) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=== worker1 エージェント ==='
=== worker1 エージェント ===
(worker1) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$

kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ export PS1='(\[\033[1;34m\]\worker2\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
(worker2) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=== worker2 エージェント ==='
=== worker2 エージェント ===
(worker2) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$

kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ export PS1='(\[\033[1;34m\]\worker3\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
(worker3) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=== worker3 エージェント ==='
=== worker3 エージェント ===
(worker3) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$
    
```

```

cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
export PS1='(\[\033[1;35m\]PRESIDENT\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
echo '=== PRESIDENT セッション ==='
echo 'プロジェクト統括責任者'
echo '=====
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ cd /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ export PS1='(\[\033[1;35m\]PRESIDENT\[\033[0m\]) \[\033[1;32m\]\w\[\033[0m\]$ '
(PRESIDENT) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=== PRESIDENT セッション ==='
=== PRESIDENT セッション ===
(PRESIDENT) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo 'プロジェクト 統括責任者'
プロジェクト統括責任者
(PRESIDENT) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ echo '=====
(PRESIDENT) /mnt/c/Users/weael/.ccc-c-clone/claude-code-clone$ tmux send-keys -t president 'claude --dangerously-skip-permissions' C-m
    
```

- tmux attach-session -t president を実行します
- tmux send-keys -t president 'claude --dangerously-skip-permissions' C-m を実行します

- ▶ president から multiagent セッションの ClaudeCode を一括起動します
- ▶ ClaudeCode が4つ起動されているのを確認します

```

export PS1='\[\033[1;35m\]PRESIDENT\[\033[0m\] \[\033[1;32m\]\w\[\033[0m\]$ '
echo '=== PRESIDENT セッション ==='
echo 'プロジェクト統括責任者'
echo '=====
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ cd /mnt/c/Users/weae1/.ccc-clone/claude-code-clone
kazuaki2018@asus-vivo-win11:/mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ export PS1='\[\033[1;35m\]PRESIDENT\[\033[0m\] \[\033[1;32m\]\w\[\033[0m\]$ '
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ echo '=== PRESIDENT セッション ==='
=== PRESIDENT セッション ===
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ echo 'プロジェクト 統括責任者'
プロジェクト統括責任者
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ echo '=====
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ tmux send-keys -t president 'claude --dangerously-skip-permissions' C-m
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ claude --dangerously-skip-permissions
(PRESIDENT) /mnt/c/Users/weae1/.ccc-clone/claude-code-clone$ claude --dangerously-skip-permissions

*Welcome to Claude Code!
/help for help, /status for your current setup
cwd: /mnt/c/Users/weae1/.ccc-clone/claude-code-clone

Tip: Run claude --continue or claude --resume to resume a conversation
You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.

> for i in {0..3}; do tmux send-keys -t multiagent:0.$i 'claude --dangerously-skip-permissions' C-m; done
    
```

```

Tip: Create custom slash commands by adding .md files to .claude/commands/ in your project or ~/.claude/commands/ for commands that work in any project
You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.

> try "how do I log an error?"
? for shortcuts Bypassing Permissions

files to .claude/commands/ in your project or ~/.claude/commands/ for commands that work in any project
You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.

> try "how does CLAUDE.md work?"
? for shortcuts Bypassing Permissions

files to .claude/commands/ in your project or ~/.claude/commands/ for commands that work in any project
You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.

> try "edit agent-send.sh to..."
? for shortcuts Bypassing Permissions

files to .claude/commands/ in your project or ~/.claude/commands/ for commands that work in any project
You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.

> try "how does README.md work?"
? for shortcuts Bypassing Permissions
    
```

- for i in {0..3}; do tmux send-keys -t multiagent:0.\$i 'claude --dangerously-skip-permission' C-m; done を実行します

- ▶ president にプロンプトを投入して各セッションが動き出すのを確認します
- ▶ 完了したら、各セッションの処理内容を確認して評価します

The screenshot shows the Claude Code interface in a terminal window. The left sidebar displays a file tree with folders like 'settings.local.json', 'dictionaries', 'instructions', and files like 'boss.md', 'president.md', 'worker.md'. The main terminal area shows the following text:

```
(PRESIDENT) /mnt/c/Users/weael/.ccc-clone/claude-code-clone$ tmux send-keys -t president 'claude --dangerously-skip-permissions' C-m
(PRESIDENT) /mnt/c/Users/weael/.ccc-clone/claude-code-clone$ claude --dangerously-skip-permissions
```

A yellow box highlights the welcome message:

```
*Welcome to Claude Code!
/help for help, /status for your current setup
cwd: /mnt/c/Users/weael/.ccc-clone/claude-code-clone
```

Below this, there is a tip and instructions on how to use Claude Code. At the bottom, a prompt is entered: > あなたはpresidentです。指示書に従って

The screenshot shows the Claude Code interface with multiple agents working. The terminal output includes:

```

• Bash(/agent-send.sh worker3
  "あなたはworker3です。Hello World
  作業開始")
  ↳ 送信中: multiagent:agents.3 ←
    あなたはworker3
    'Hello World 作業開始'
    ✓ 送信完了: worker3 に
    'あなたはworker3です。Hello Wor
    ... +1 line (ctrl+r to expand)

* Accomplishing.. (26s · ↑ 413 tokens ·
  esc to interrupt)

> |

? for shortcuts Bypassing Permissions

• I'll execute the Hello World task as worker1
  according to the instructions.

• Update Todos
  ↳ Display Hello World message
  ↳ Create worker1 completion file
  ↳ Check if all workers completed and
  report if last

* Envisioning.. (7s · ↑ 117 tokens ·
  esc to interrupt)

> |

? for shortcuts Bypassing Permissions

```

On the right side, there are two panels showing the state of different agents:

- Top panel: Shows 'auda' and 'Code! https://claude.ai/upgrade then run /login.' Below it, it says '> あなたはworker2です。Hello World 作業開始' and 'I'll execute the Hello World task as worker2 according to the instructions.' It also shows 'Puzzling..' status.
- Bottom panel: Shows 'You can now use a Claude Pro subscription with Claude Code! https://claude.ai/upgrade then run /login.' Below it, it says '> あなたはworker3です。Hello World 作業開始' and '\* Synthesizing..' status.

• プロンプト『あなたはpresidenです。指示書に従って』と入力します

• Bossから指示が出て作業開始していることを確認します

- ▶ 数人によるアジャイル(スクラム)チームを、1名+エージェントAI開発ツール群で置き換える試みが多数行われています
- ▶ 期待される生産性向上
  - ▶ 単一の開発プロジェクトを置き換えると、生産性は3~5倍と言われています
  - ▶ 5人のスクラムチームを置き換えると、最大4人分のジョブをAIが自律実行してくれるので…その生産性は従来比で：

**12~20倍**

- ▶ 期待される次の進化は…
  - ▶ 開発プロジェクトそのものの上流設計を実施して提案できるようになるか…
    - ▶ プロジェクト要素に分解、レイヤ化と並行化の最適化、ルールと優先順位の最適化を実施した上で、タスク構造を提案し、オーケストレーションの半自動化を目指す想定しています

## 5. これからのために リスキングの暫定的な勘所

▶ 経験1～2年のジュニアな開発者は不要になるのか？…

- ▶ 未経験者や経験が少ない者は入口にすら立てない、というジレンマを感じませんか？
- ▶ コーディングはAIが代行してくれますが、それが正しく実行されたか確認する必要があり、AIが自律的にバグを是正できない場合は、あなたが肩代わりする必要があります
- ▶ そのためには、従来の何倍ものスピードで学ぶ（学び直す）必要があります

▶ 今までの学習法

- ▶ 知識取得重視
- ▶ 人に教わる、自習する
- ▶ 自ら手を動かすOJT



▶ AI時代の学習法

- ▶ 論理的判断を鍛えるためにAIを使う
- ▶ AIを評価し指示できる力を身につける
- ▶ AIとの壁打ちを納得できるまで回して学習
- ▶ AIを同時に複数使い比較する

## 5. これからのために リスキングの暫定的な勘所 批判的思考力と適切なAI出力評価

## ▶ 生成AIの使用で、あなたの思考力と責任感が試されます

### ▶ ハルシネーションの内包や発現

- ▶ LLMの出力に誤情報や幻覚や解釈間違いが含まれていませんか？
- ▶ バイアス形成や誤情報が拡散する可能性はありませんか？

### ▶ リスク管理

- ▶ コンプライアンス違反やセキュリティ違反になりませんか？
- ▶ 誰かが被害を受ける可能性はありませんか？

## ▶ 生成AIの使用で、あなたの意思決定力が試されます

### ▶ 自動化ツールとしてのAIエージェントですか…

- ▶ それを使い続けて、あなた自身の意思決定の質はどれくらい高まりましたか？
- ▶ それを使い続けて、あなた自身の意思決定のスピードはどれくらい早くなりましたか？

▶ あなたの思考力や意思決定力をどのように高めますか？



- ▶ 開発技術者必須の基礎学習は、AIとの壁打ちで効果的に学習できます
  - ▶ 論理学、統計学、認知バイアス
  - ▶ 開発プロセス（開発プロンプト、コーディング、ライブラリ利用、API利用、SQL、機能設計/データ設計/非機能設計、リファクタリング、CI/CD、Gitなどなど）
- ▶ 基本的な作業（AIとの協働作業）
  - ▶ 的確なプロンプトの作成して投入し、生成結果の複数ソースを検証(レビュー)します
  - ▶ AIコーディングエージェントの誤謬を分析して解決を促すようにするか、自ら解決します
- ▶ 開発初心者はAI開発ツールに様々な課題でコーディングしてもらい、それを見て学び、実行して検証方法を学ぶ…を納得するまで回すのが効果的です

## 5. これからのために リスキングの暫定的な勘所 オーケストレーション能力

- ▶ 従来、一定の規模や複雑さを持ったプロジェクトでは、オーケストレーション能力はPMOやアジャイルリーダーに必須でした
- ▶ 現時点のAI開発ツールは、すべて自律的にやれるわけではないので、開発技術者による設計、割り当て、調整などの作業が必要になります
  - ▶ 以下について、その目的や役割や作業手順などをAIを利用して学習できます
    - ▶ 最適なタスク分割と開発フローのケーススタディ
    - ▶ CI/CDパイプラインとリファクタリングのケーススタディ
  - ▶ AI開発ツールで指示⇒生成⇒評価を回して以下を進めてください
    - ▶ ルール(あるいはシステムプロンプト)の改善やレイヤ化
    - ▶ 複数のAI開発ツールやAPI/アプリとの連携の可能性を検討
      - ▶ 新しい連携ツールの発見や、新たに AIエージェントを開発する契機になります

## 5. これからのために リスキリングの暫定的な勘所 コンテキストエンジニアリング

## ■ Claude Opus 4

- ▶ プロンプト『最近なぜコンテキストエンジニアリングが重要と言われるようになったのですか？』

コンテキストエンジニアリングの重要性 ↓

共有

KM 最近なぜコンテキストエンジニアリングが重要と言われるようになったのですか？

コンテキストエンジニアリング (Context Engineering) が重要視されるようになった背景には、主に以下の要因があります：

**大規模言語モデル (LLM) の普及と限界の認識** GPTやClaude、Geminiなどの高性能なLLMが広く利用されるようになり、同時にこれらのモデルが持つ制約も明らかになってきました。モデル自体の再学習やファインチューニングは高コストで時間がかかるため、プロンプトやコンテキストの工夫によって性能を最大化する手法が注目されるようになりました。

**RAG (Retrieval-Augmented Generation) の実用化** 外部のデータベースやドキュメントから関連情報を検索し、それをコンテキストとして与えることで、LLMの知識を動的に拡張する手法が実用段階に入りました。これにより、最新情報や専門知識をモデルに効率的に提供できるようになりました。

**コンテキストウィンドウの拡大** 最新のLLMでは、数十万から百万トークン規模のコンテキストを扱えるようになり、より複雑で長大な情報を入力として与えることが可能になりました。これにより、コンテキストの設計と最適化がより重要な技術要素となっています。

**企業での実用化ニーズ** 多くの企業がLLMを業務に組み込む中で、自社特有の知識やルール、最新データを効果的に活用する必要が生じています。コンテキストエンジニアリングは、モデルの再学習なしにこれを実現する実践的な手法として注目されています。

**精度向上とハルシネーション対策** 適切なコンテキスト設計により、LLMの回答精度を向上させ、誤った情報の生成 (ハルシネーション) を減らすことができることが実証されています。特に、具体的な事実や数値を扱う場合に効果的です。

これらの要因により、コンテキストエンジニアリングは単なるプロンプトの工夫を超えて、LLMを実用的に活用するための重要な技術分野として確立されつつあります。



再試行 ↓

Claudeは間違えることがあります。回答内容を必ずご確認ください。

## ■ Gemini 2.5 Pro

### ▶ プロンプト『最近は何でコンテキストエンジニアリングが重要と言われるようになったのですか？』

Gemini 2.5 Pro

最近は何でコンテキストエンジニアリングが重要と言われるようになったのですか？

思考プロセスを表示

最近、コンテキストエンジニアリングが重要視されるようになったのは、**大規模言語モデル (LLM) の能力を最大限に引き出し、より高度で実用的なAIアプリケーションを開発するために不可欠な技術だからです。**

従来の「プロンプトエンジニアリング」がAIへの「指示文」そのものを工夫する技術だったのに対し、コンテキストエンジニアリングは、その指示の背景にある**情報環境全体を設計する、より包括的なアプローチ**です。

---

#### なぜ今、コンテキストエンジニアリングが重要なのか？

##### 1. プロンプトエンジニアリングの限界

単純な質疑応答であれば、優れたプロンプト（指示文）だけで対応できました。しかし、AIを実際の業務で活用しようとする、以下のような課題に直面します。

- 過去のやり取りを覚えていない: 会話の流れを理解できず、一貫性のない応答をしてしまう。
- 自社の専門知識がない: 社内文書や最新の業界動向など、外部の情報を参照できない。
- 状況判断ができない: ユーザーの立場や目的を汲み取った、最適な回答ができない。

これらの課題は、指示文の工夫だけでは解決が難しく、「AIが参照すべき情報（コンテキスト）は何か」を設計する必要が出てきました。

Gemini 2.5 Pro

#### 2. LLMの性能を最大限に引き出すため

LLMは、適切な「材料」がなければその能力を発揮できません。コンテキストエンジニアリングは、LLMにとっての「最高の材料」を準備する役割を担います。

- 精度の向上:** 関連性の高い情報だけをAIに与えることで、ノイズを減らし、回答の精度を飛躍的に高めます。
- パーソナライズ:** ユーザーの過去の履歴や好みをコンテキストとして与えることで、一人ひとりに最適化された応答が可能になります。
- コスト削減:** 不要な情報を削ぎ落とし、必要最低限のデータ（トークン）でAIを動作させることで、API利用料などのコストを削減できます。

#### 3. 動的で複雑なタスクへの対応

コンテキストエンジニアリングは、単に静的な情報を与えるだけではありません。状況に応じて必要な情報を動的に**選択・生成し、AIに提供**します。

例えば、AIアシスタントに「明日の会議の資料を準備して」と依頼した場合、コンテキストエンジニアリングは以下のような働きをします。

- カレンダー連携:** 「明日の会議」がどの会議かを特定する。
- 議事録検索 (RAG):** 過去の関連会議の議事録を検索・要約する。
- ファイルアクセス:** 共有フォルダから関連資料を探し出す。
- 役割設定:** 「あなたは優秀なアシスタントです」といった役割を与え、適切な形式で資料をまとめるよう指示する。

このように、複数の情報源やツールを連携させ、タスクを解決するための最適な情報環境を構築するのがコンテキストエンジニアリングの真髄です。

## ■ Gemini 2.5 Pro

▶ プロンプト『最近なぜコンテキストエンジニアリングが重要と言われるようになったのですか？』

The screenshot shows the Gemini 2.5 Pro interface with a comparison table between Prompt Engineering and Context Engineering. Below the table is a summary paragraph and a search bar.

	プロンプトエンジニアリング	コンテキストエンジニアリング
焦点	AIへの「指示文（プロンプト）」そのもの	AIを取り巻く「情報環境」全体
アプローチ	静的・単発の指示	動的・継続的な情報提供
目的	1回のやり取りで最高の出力を得る	一貫性のある、より高度なタスクを遂行する
例	「以下の文章を要約してください」	会話履歴、外部DB、ツールを連携させ、文脈に合った要約を生成するシステム

Google スプレッドシートにエクスポート

簡単に言えば、**プロンプトエンジニアリングは「点」のアプローチ、コンテキストエンジニアリングは「線」や「面」でAIの能力を捉えるアプローチ**と言えます。AI開発の焦点が、モデル自体の性能から「いかに賢く使うか」に移る中で、コンテキストエンジニアリングはAI活用の成否を分ける鍵として、その重要性を増しているのです。

このトピックの詳細なリサーチ レポートを作成するよう Gemini に リクエストできます [今すぐ試す](#) ×

## ■ ChatGPT o3

### ▶ プロンプト『最近なぜコンテキストエンジニアリングが重要と言われるようになったのですか？』

ChatGPT o3

思考時間: 29 秒

近年、大規模言語モデル（LLM）の入力コンテキストが数十万～100万トークン規模に拡大し、外部データやツール呼び出しを組み合わせた高度なワークフローが普及したことで、「どの情報を・どんな構造で・どのタイミングでモデルに渡すか」を設計する **コンテキストエンジニアリング** がソフトウェア開発とAI運用のボトルネックになりつつあります。単なる「良いプロンプト」を書く段階（プロンプトエンジニアリング）を超え、**コンテキスト全体の設計**—長大な履歴、検索結果、社内ドキュメント、ユーザープロフィール、メモリーブロックなどを統合的に扱う技術と実務が不可欠になったことが、重要視される最大の理由です。以下で背景と要因を整理します。

### コンテキストエンジニアリングとは？

#### 定義とプロンプトエンジニアリングとの違い

- 定義：LLM に与えるあらゆる入力（プロンプト本文・履歴・外部文書・ツール出力・メタデータなど）を取捨選択し構造化する技術領域。 [The New Stack](#) [Analytics Vidhya](#) [GIGAZINE](#)
- 違い：プロンプトエンジニアリングが「単一の指示文の改善」に焦点を当てるのに対し、コンテキストエンジニアリングは **情報源の統合・スロット化・トークン制限管理・更新頻度・セキュリティ境界** まで設計対象に含みます。 [llamaindex.ai](#) [Medium](#)

### なぜ最近注目されるのか

#### 1. コンテキストウィンドウの急拡大

- Google Gemini 1.5 Pro や Anthropic Claude 3 などが 200K～1M token の入力を扱えるようになり、「入れられる情報量 > 必要情報の選択」が課題へと転換。 [フォーブス](#) [Anthropic](#)
- 大容量ゆえに冗長入力では推論コストが跳ね上がるため、重要度に基づくサンプリングや階層化が必須。 [The New Stack](#)

ChatGPT o3

#### 2. RAG とベクターデータベースの普及

- Retrieval-Augmented Generation（RAG）が標準化し、外部検索→要約→挿入のパイプライン設計が主要タスクに。 [ワイキベディア](#) [Google Cloud](#) [ウォール・ストリート・ジャーナル](#)
- ベクターストア選定・類似度閾値・chunk サイズといったハイパーパラメータがモデル性能を左右し、「検索精度 + コンテキスト配置」の両立が求められる。 [llamaindex.ai](#)

#### 3. エンタープライズ導入とコンプライアンス要求

- 社内ドキュメントや個人情報を安全に取り込みつつ権限制御を保つ「コンテキスト境界」設計が急務になり、ガバナンス担当者を巻き込む工程が増大。 [Qlita](#) [ウォール・ストリート・ジャーナル](#)
- ツール側でも「メモリーブロック『アクセススコープ付き埋め込み』」などエンタープライズ機能が整備され、実装知識が新たな専門領域に。 [llamaindex.ai](#)

#### 4. マルチエージェント／ツール呼び出しの高度化

- タスク分割やコード生成エージェントが増え、エージェント間で共有・分配する文脈を設計する必要が生じた。 [Medium](#)
- 典型的なフロー：①ユーザー要求→②RAGで補完→③ツール実行→④結果を再挿入→⑤回答という **多段ラウンド** を安定化させるアーキテクチャが求められる。 [The New Stack](#)

#### 5. 永続メモリとパーソナライズの課題

- GPT-4 系の「メモリー機能」や LangChain/LlamaIndex の履歴ストアは便利だが、ノイズ混入・優先度破綻による誤応答が報告され、**メモリーフィルタリング戦略**が研究対象に。 [OpenAI Community](#) [llamaindex.ai](#)
- 企業や個人で **長期プロジェクト履歴** を扱う際、何を「固定コンテキスト」とし、何を都度検索するかのポリシー策定が不可欠。 [Qlita](#) [GIGAZINE](#)

## ■ ChatGPT o3

▶ プロンプト『最近なぜコンテキストエンジニアリングが重要と言われるようになったのですか？』

ChatGPT o3 共有する ... WE PLUS

### 実務での応用例

ユースケース	典型的コンテキスト構成	成果
APIドキュメント自動回答	①固定：仕様YAML ②動的：最新バージョン差分	開発効率30%向上 <span style="font-size: small;">Qlita</span>
社内ポリシーチャットボット	①権限制御付き PDF ②Role メタデータ	HR 問い合わせの75%削減 <span style="font-size: small;">ウォール・ストリート・ジャーナル</span>
コーディングエージェント	①プロジェクトツリー要約 ②Issue 履歴 ③RAGでAPI例	PR作成時間40%短縮 <span style="font-size: small;">Medium</span>

### 今後の展望と学習リソース

- ツールチェーンの統合：LangGraph、LlamaIndexMemoryBlocks など「コンテキストOS」的レイヤが進化。 llamaindex.ai
- 新しい指標：トークンコスト／精度だけでなく、Context Quality Score（情報鮮度・冗長率・権限整合性）を測るメトリクス研究が進行中。 The New Stack
- キャリア機会：テックライター×MLOps×情報設計のハイブリッド職種として「コンテキストエンジニア」が求人に登場し始めている。 Analytics Vidhya

**まとめ**

大規模化したコンテキストウィンドウと RAG の標準装備化により、「どの情報を、どう整理し、どう守るか」がモデル性能・コスト・信頼性を左右する最大のチューニングポイントになりました。プロンプト単体では解決できない「情報編成工学」としての、↓コンテキストエンジニアリングが、今後の AI 活用の成否を決めるキーコンピタンスになるでしょう。

『AI は人間を置き換えれないが…

AI を使う人間が、AI を使わない人間を置き換える』

ありがとうございました