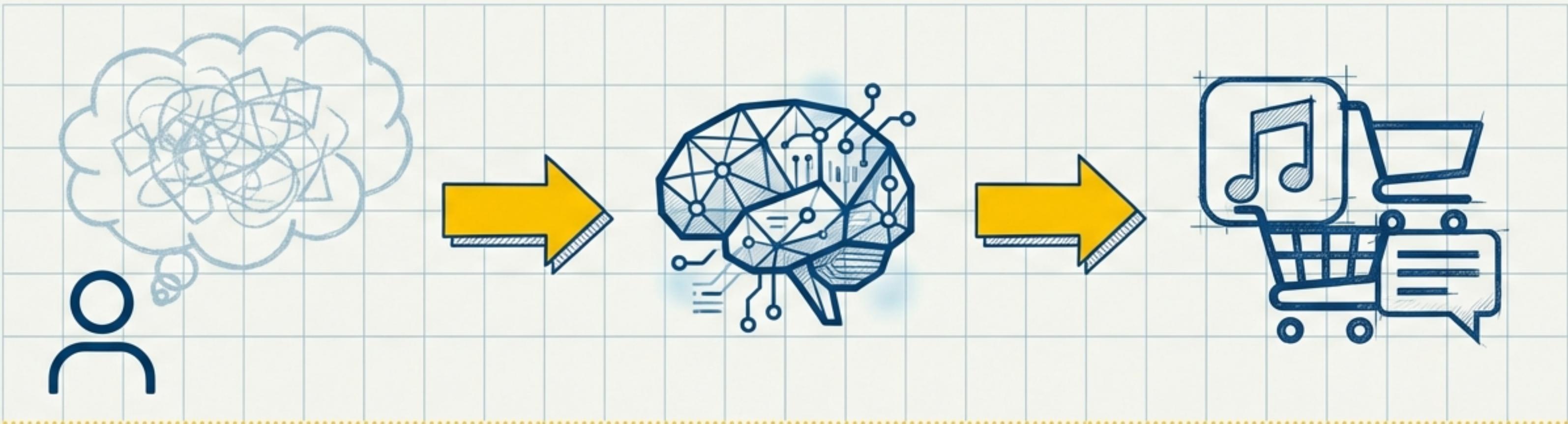




AI開発のためのWebアーキテクチャ設計図

AIを最強の開発パートナーにするための、4つの基本原則

AIに「アプリを作って」と頼む。 でも、本当に伝えたいことは何だろう？

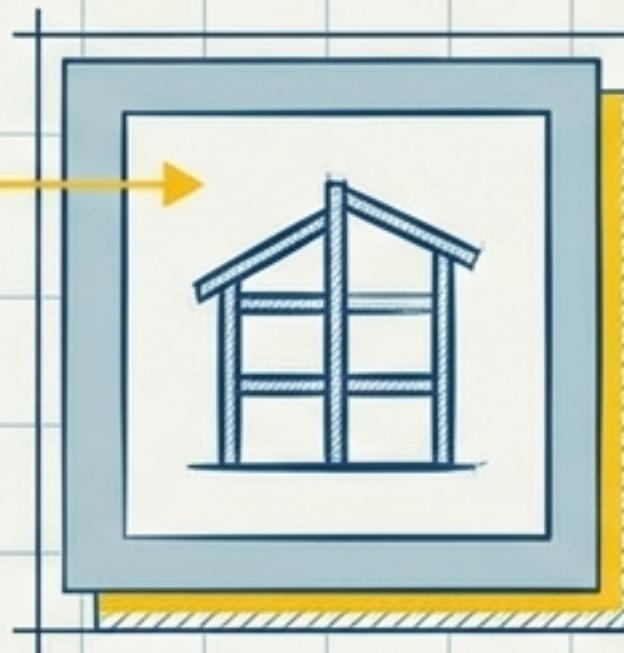


漠然とした指示は、予測不能な結果につながります。
AIと共に優れたものを創るためにには、共通言語、つまり共通の「設計図」が必要です。
このプレゼンテーションが、その設計図を提供します。

AI開発を成功させる4つの柱

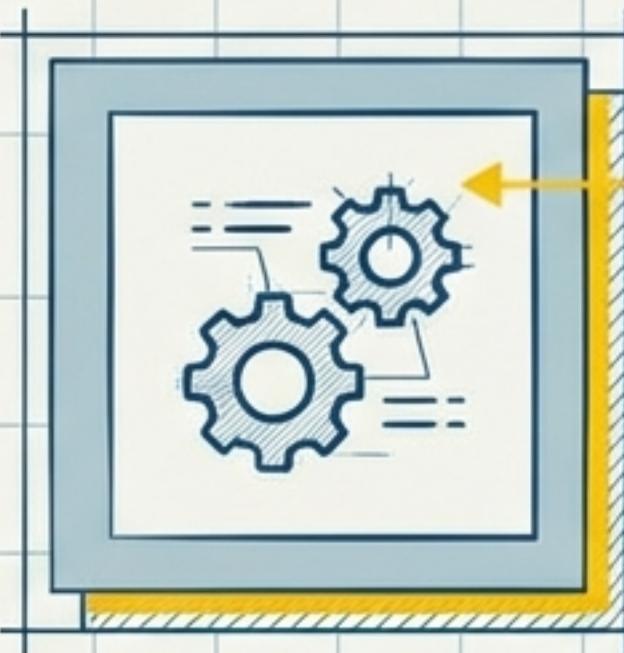
1. 構造 (Structure)

アプリの全体像を理解する



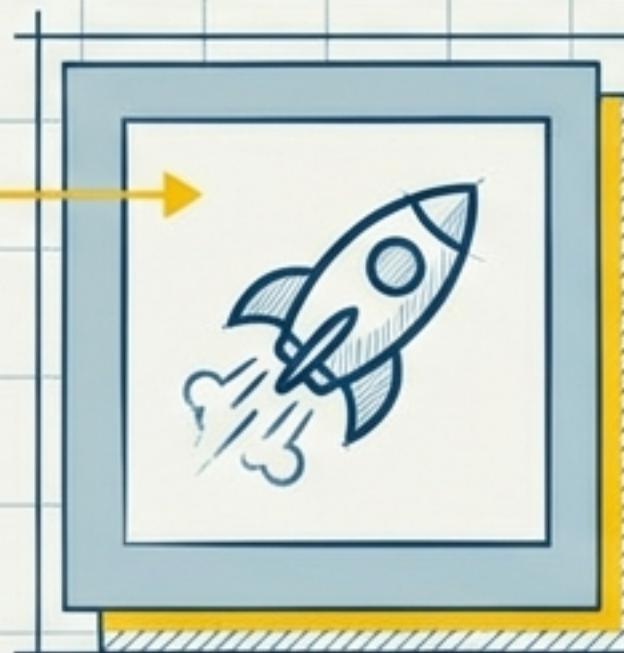
2. 通信 (Communication)

UIとAPIの境界線を知る



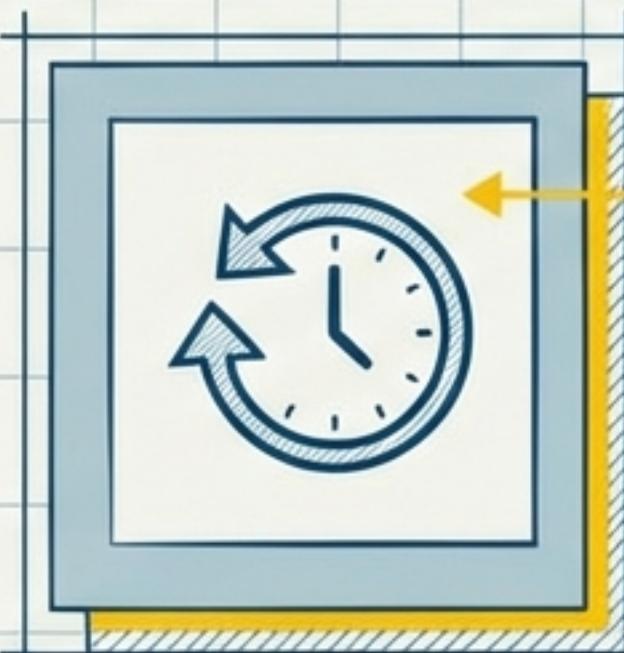
3. 公開 (Launch)

デプロイの本質を掴む



4. 時間 (Time)

Gitで安全に未来を書き換える



01

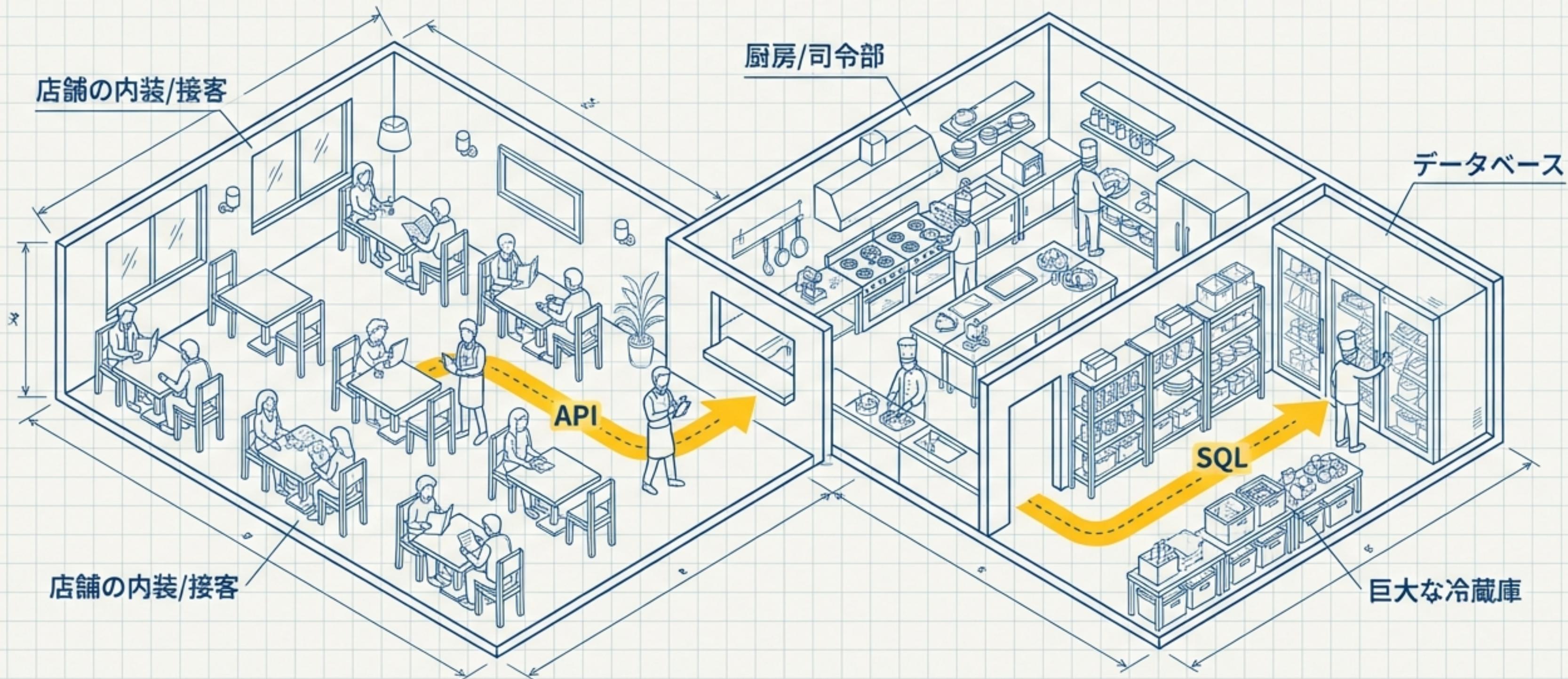


構造 (Structure)

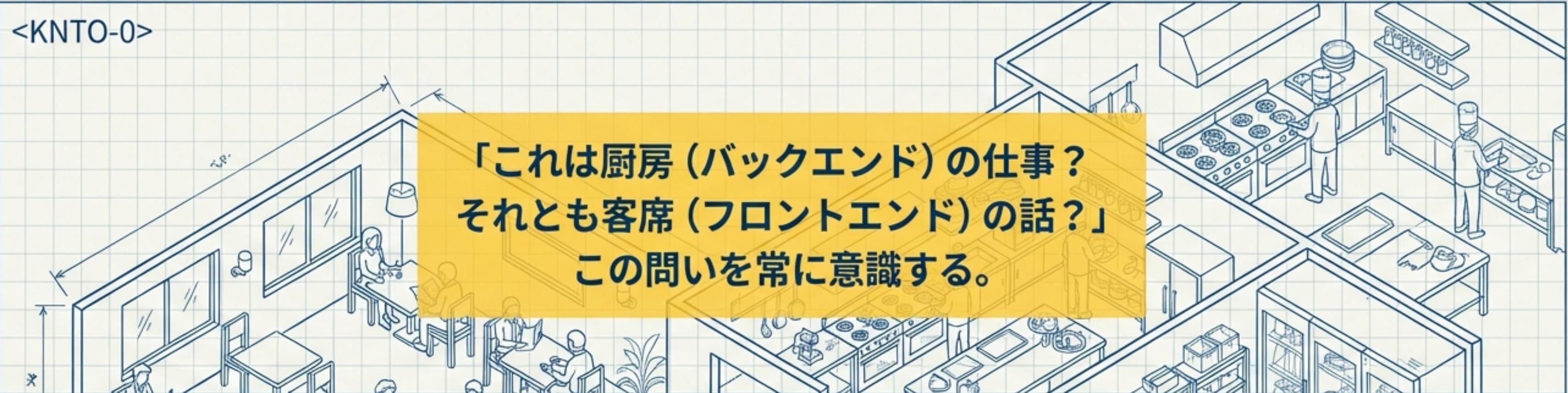
フロントエンド / バックエンド / DBの関係

アプリケーションは、 一つのレストランである

レストランで例えると、フロントエンドは「客席とウェイター」、バックエンドは「厨房のシェフ」、データベースは「食材庫」です。お客様（ユーザー）は厨房（バックエンド）に直接入ることはできません。必ずウェイター（フロントエンド）を通して注文します。



指示の精度が劇的に向上する思考法



フロントエンド (Frontend)

ブラウザで動く部分。
ReactやVue.jsなどで作られます。
ユーザーの目に触れる全てです。



バックエンド (Backend)

サーバーで動く部分。
Node.jsやPythonなどで作られます。
データの加工や保存を担当します。



データベース (DB)

データを永続的に保存する場所。
PostgreSQLやMySQLなどが
使われます。



02

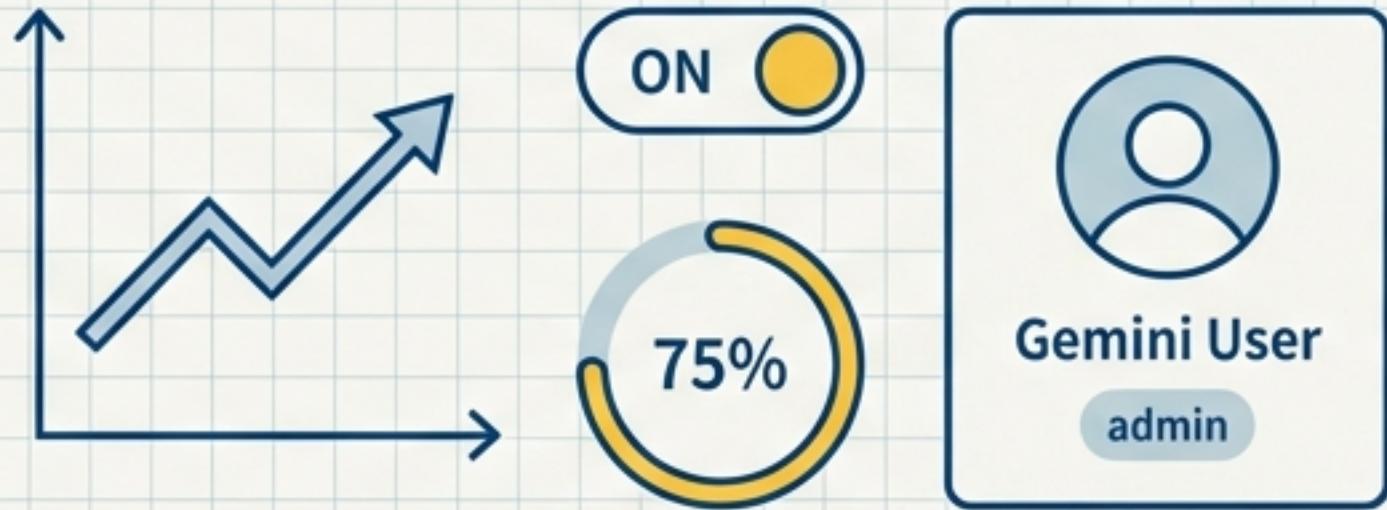


通信(Communication)

UIとAPI、その明確な境界線

「人間との対話」と「機械との対話」

UI (User Interface)



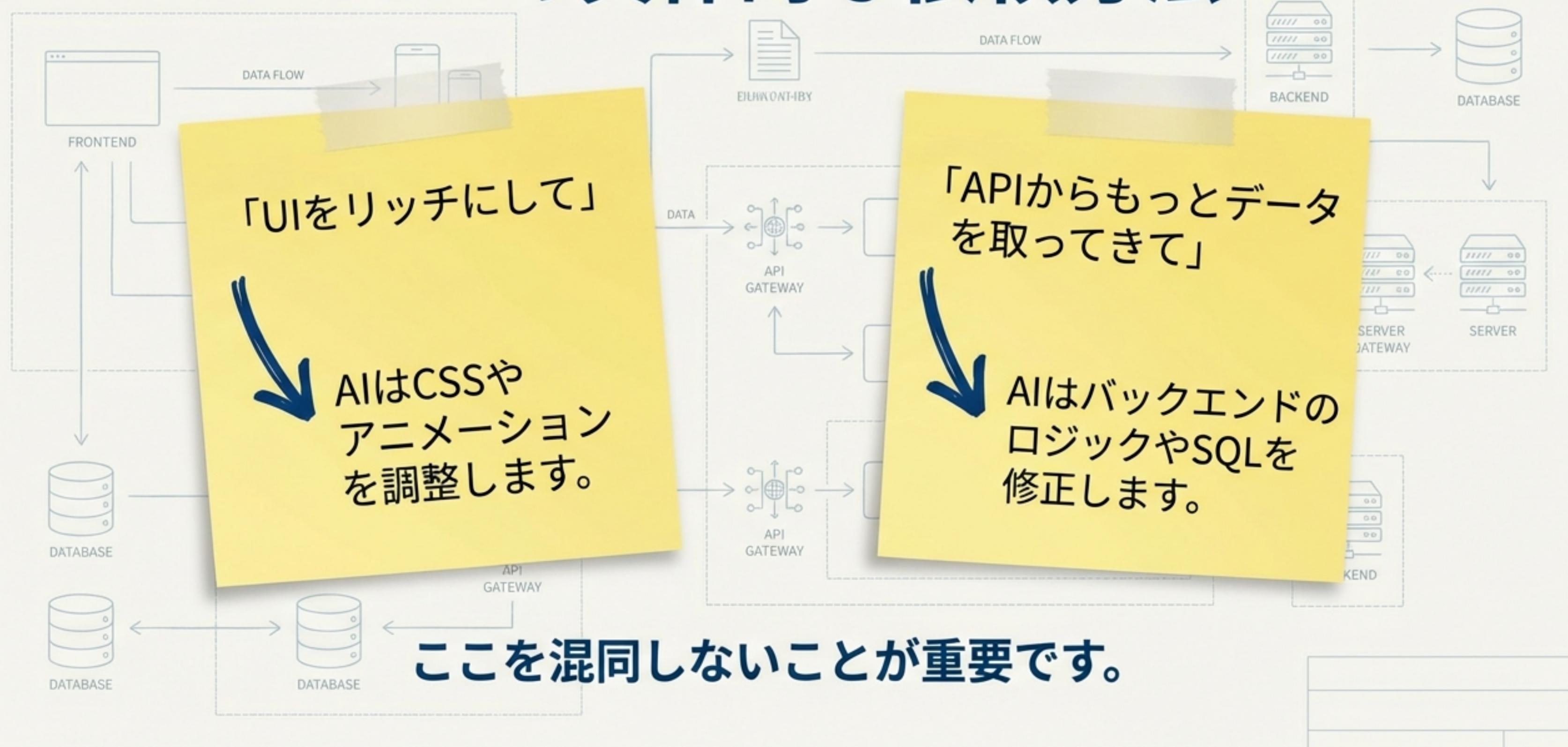
人間とコンピュータの接点です。ボタン、入力フォーム、画面レイアウトなど、「人間が操作するもの」です。

API (Application Programming Interface)

```
{  
  "userId": "u_12345",  
  "name": "Gemini User",  
  "role": "admin",  
  "lastLogin": "2026-01-10T10:00:00Z"  
}
```

フロントエンドとバックエンド（またはシステム同士）が会話するための窓口です。

AIへの具体的な依頼方法



03

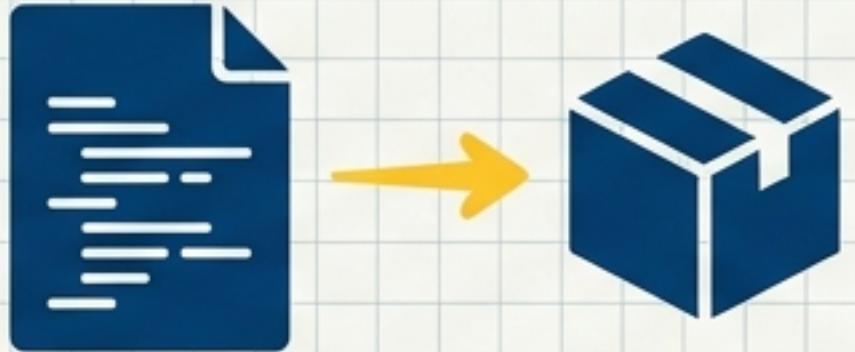


公開 (Launch)

デプロイとは「何が起きている行為」か

ローカルPCから、世界へ

1: ビルド



人間が読みやすいコードを、機械が効率よく実行できる形式に変換・圧縮すること。

2: 転送



サーバー（VercelやRenderなど）にファイルを送ること。

3: 起動



サーバー上でプログラムを走らせ、リクエストを待ち受ける状態にすること。



よくあるエラー

「ローカルでは動いていたのに、デプロイしたら動かない」という現象は、環境変数（APIキーなど）の設定忘れや、ビルドプロセスの違いによって起こります。

04



時間 (Time)

Gitはなぜ必須なのか

Git: 失敗した未来を、なったことにする技術



恐れずに実験するための唯一の手段

AIは時々、大胆な修正を行って既存の機能を壊すことがあります。Gitがあれば、「昨日の状態に戻して」と念る必要はありません。コマンド一つで確実に戻れます。

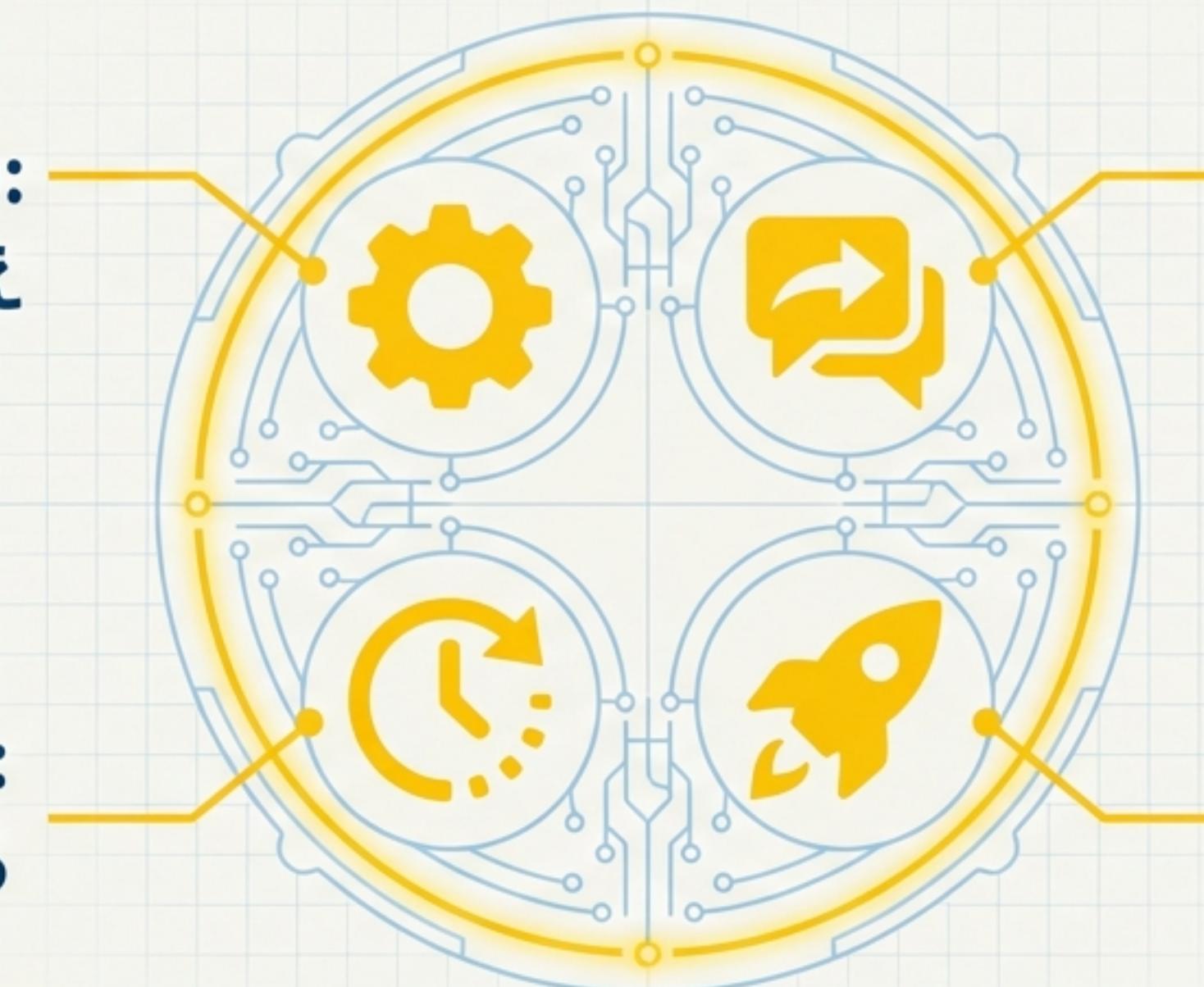
AIは、最強の相棒になる

構造 (Structure):
全体像を捉え

時間 (Time):
安心して任せる

通信 (Communication):
的確に分離し

公開 (Launch):
世界に届け



この「設計図」を手に、AIとの開発を始めよう。