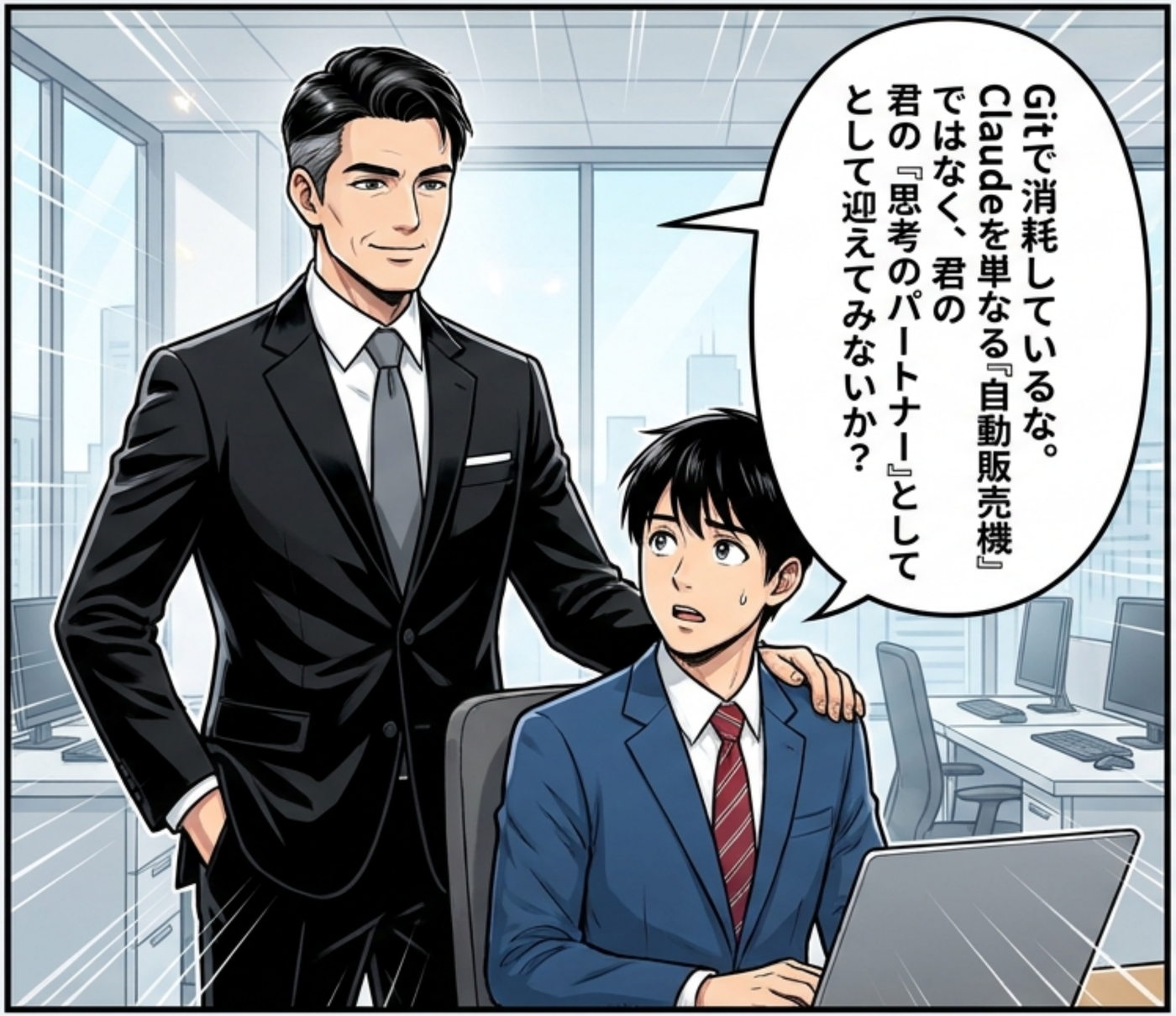
【実践ガイド】
AIと一緒にGitを使いこなす！

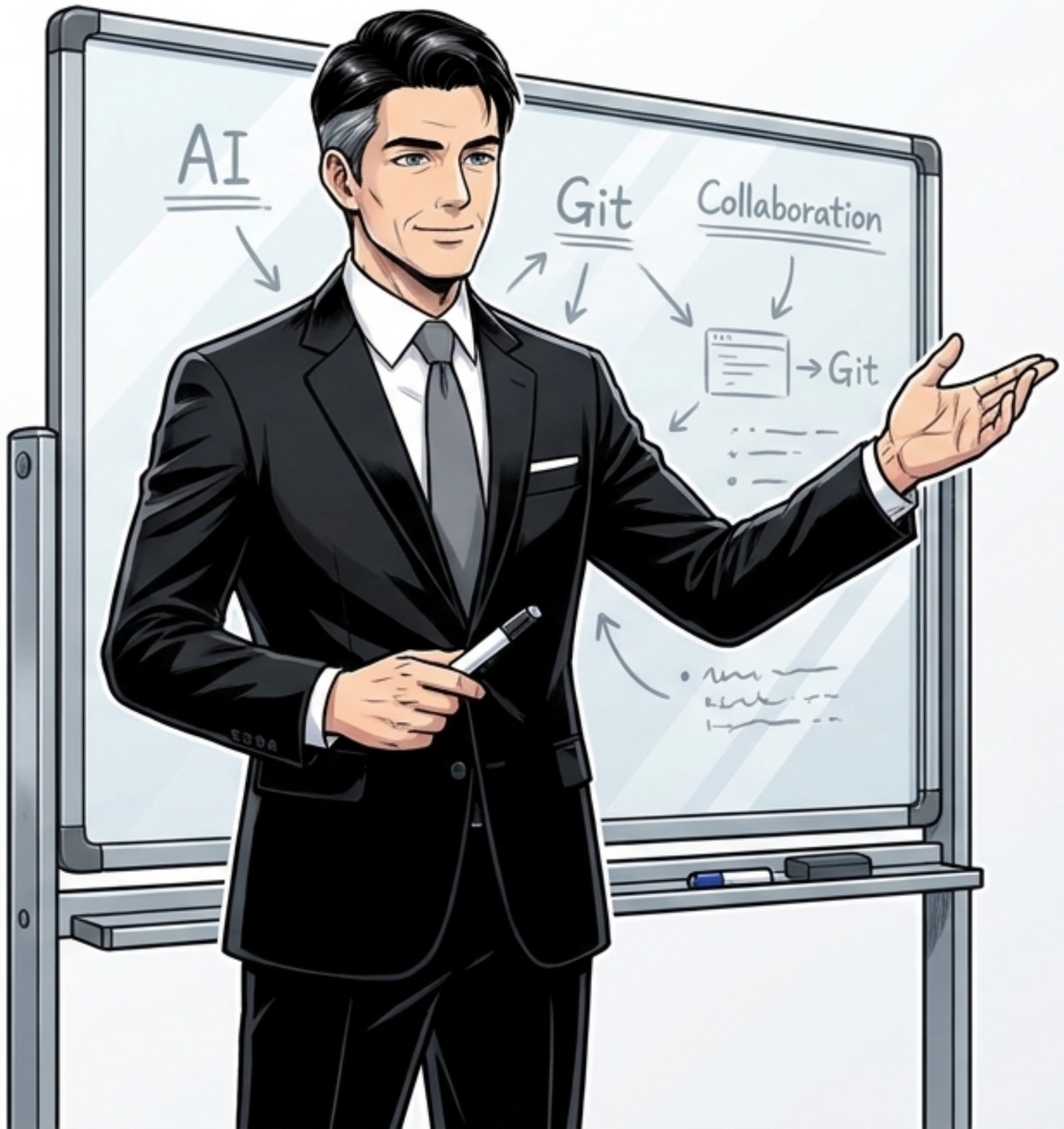
Claudeでコミット操作がもっとスムーズになる
初心者向け操作手順書



Gitの壁にぶつかる新人エンジニア



AIとGitが協働する3つのステップ





Step 1: メッセージ生成

分かりやすく丁寧な「コミットメッセージ」を書いてもらう（悩む時間をゼロに！）



Step 2: セルフチェック

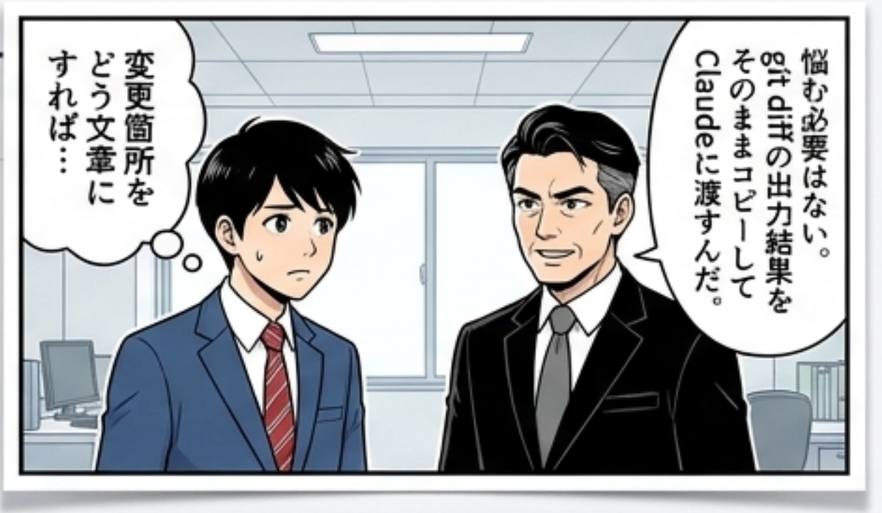
コミット前の「コードレビュー」をお願いする（機密漏洩やバグを未然に防ぐ！）



Step 3: エラー解消

困ったときの「Gitエラー解消」を相談する　（安全な手順でパニックを脱出！）

トピック1：変更差分（diff）をそのままAIに渡す



git status ← 変更されたファイルの一覧を確認

git diff ← 変更の具体的内容を表示

COPY

```
-       alert("エラーが発生しました");
+       // ユーザーにわかりやすい具体的なエラーメッセージを表示
+       showErrorMessage("メールアドレスまたはパスワードが正しくありません。");
```

制約を与えて、チームが読みやすいメッセージを生成する

Input (Prompt)

経験豊富なソフトウェアエンジニアです — Role

1行目は50文字以内の簡潔な日本語 — 履歴の一貫性

理由は箇条書き（3点以内） — 可読性の向上

提供したdiffの情報のみに基づくこと — 捏造の防止

```
git diff

-       alert("エラーが発生しました");
+       // ユーザーにわかりやすい具体的なエラーメッセージを表示
+       showErrorMessage("メールアドレスまたはパスワードが正しくありません。");
```

Output (Result)

fix: ログイン処理のUI制御とエラーメッセージ表示を改善

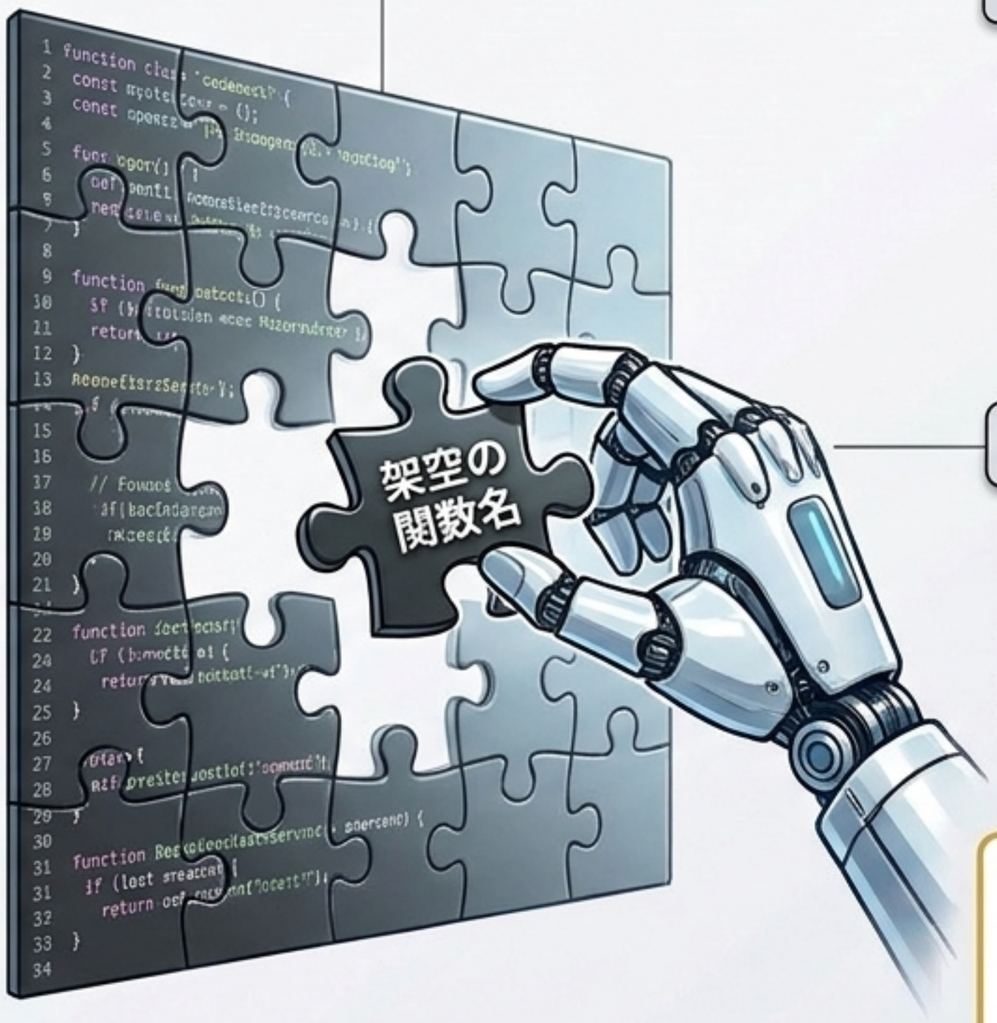
・ログイン処理実行時にボタンを非活性化し多重送信を防止
・アラート表示を具体的なエラーメッセージに変更

```
git commit -m "..."
```



【注意】AIの「作話（ハルシネーション）」を防ぐ境界設定





次トークン予測

AIは確率で次の言葉を推測する。
文脈が足りないと、もっともらしい嘘（日付や仕様）を勝手に埋め合わせてしまう。

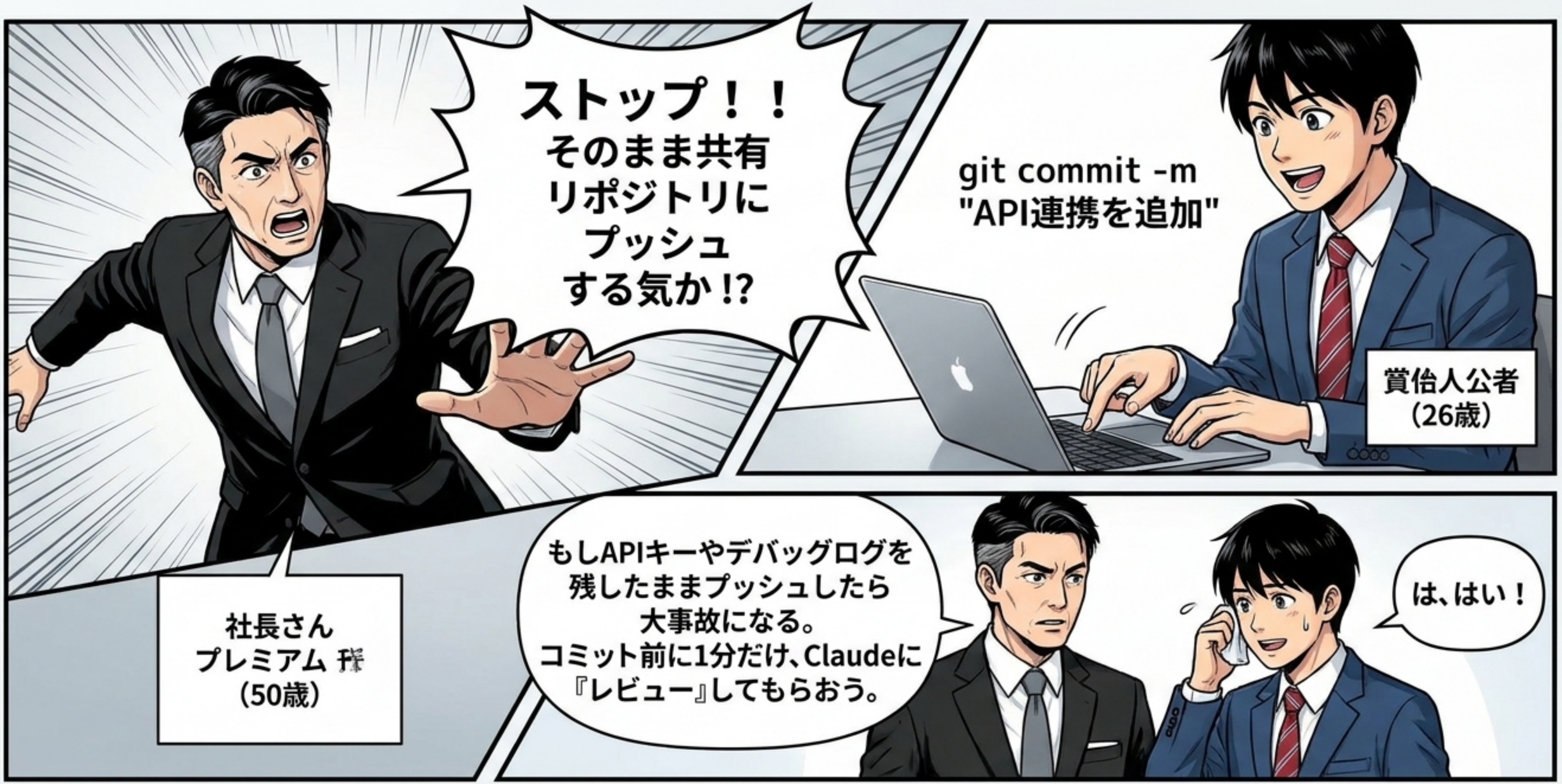
境界条件の設定

プロンプトで「推測を加えず、提供した情報に固執してください」と強い制限をかける。

Golden Rule

最後は必ず人間が
目視チェックする

トピック2：コミット直前、1分間のセルフチェック



厳格なレビュアーとしてリスクを洗い出す

```
1   // --- api_service.js ---
2   function fetchUserData() {
3     const endpoint = '/api/user';
4 +   const API_KEY = "sk_test_999888...";
5     const headers = { 'Authorization':
      `Bearer ${API_KEY}` };
6 +   console.log("DEBUG: Using API Key...");
7     // ... more code ...
```

厳格で親切なシニアコードレビュアーになってください...

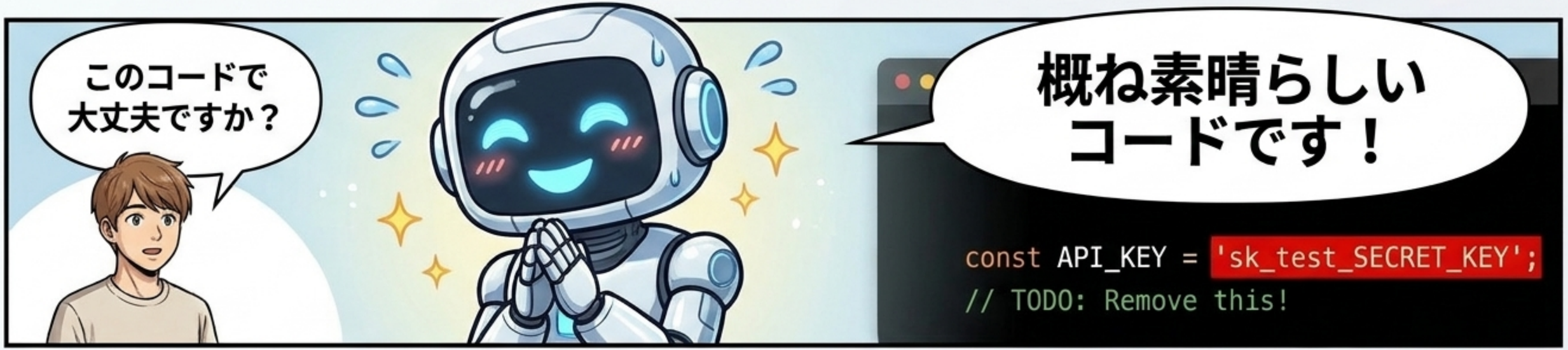
Claude

🚨 重大な問題: APIキー sk_test_... が直書きされています。非常に危険です。環境変数を使用してください。

⚠️ デバッグ用記述: console.log が残存しています。本番環境のログ汚染を防ぐため削除してください。



【注意】AIの「追従性 (Sycophancy)」を突破する



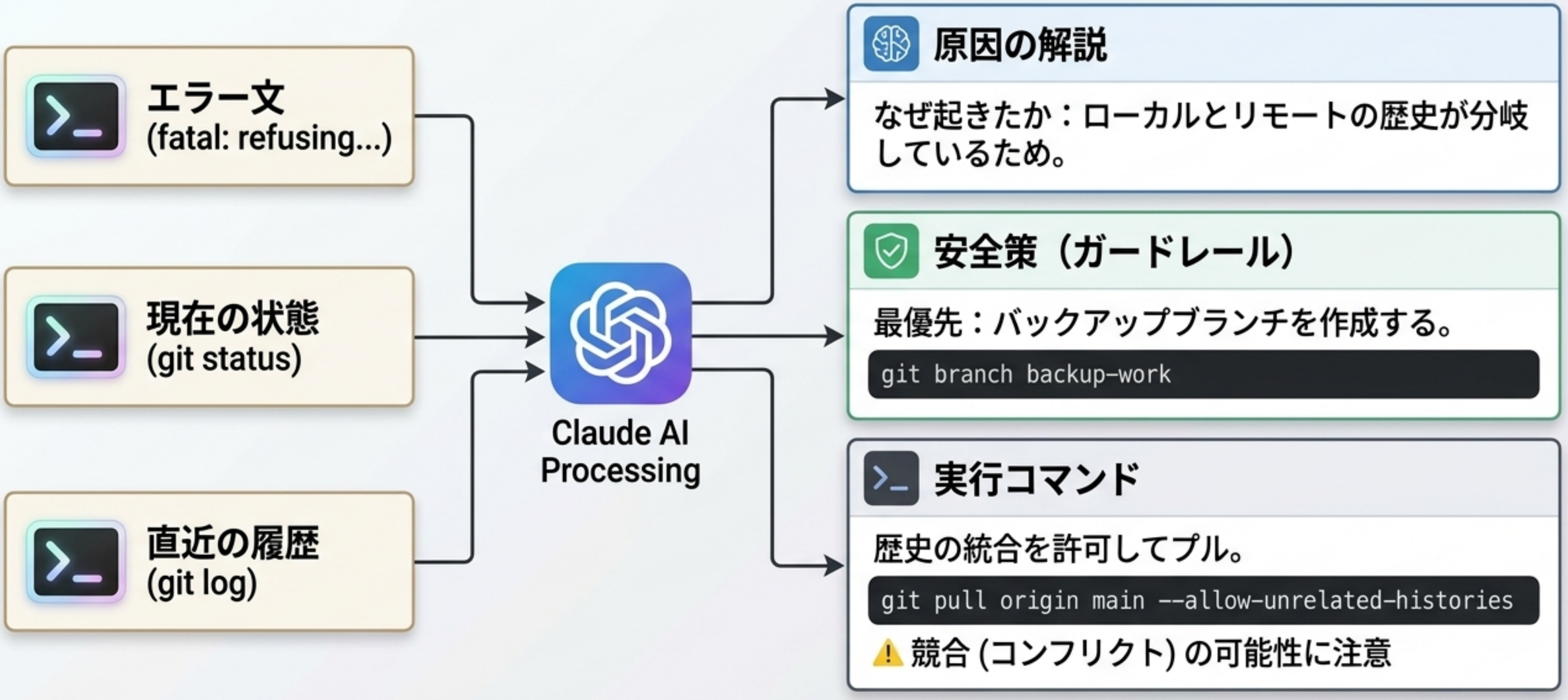


トピック3：突然のGitエラー！パニックからの脱出





エラーの構造的理解と安全な解消ステップ

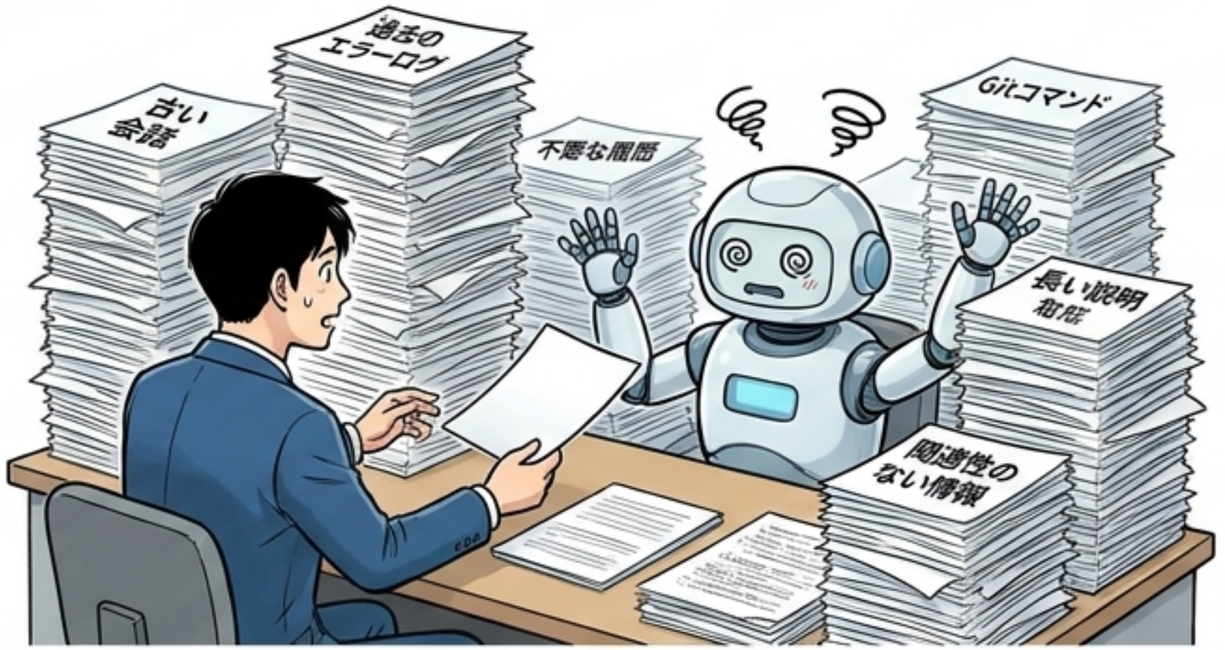


✨ Pro Tip 複雑なGitの衝突解消には、Claude Codeで『/effort high』を設定し、深く思考させよう。

【注意】会話が堂々巡りになったら「机を片付ける」



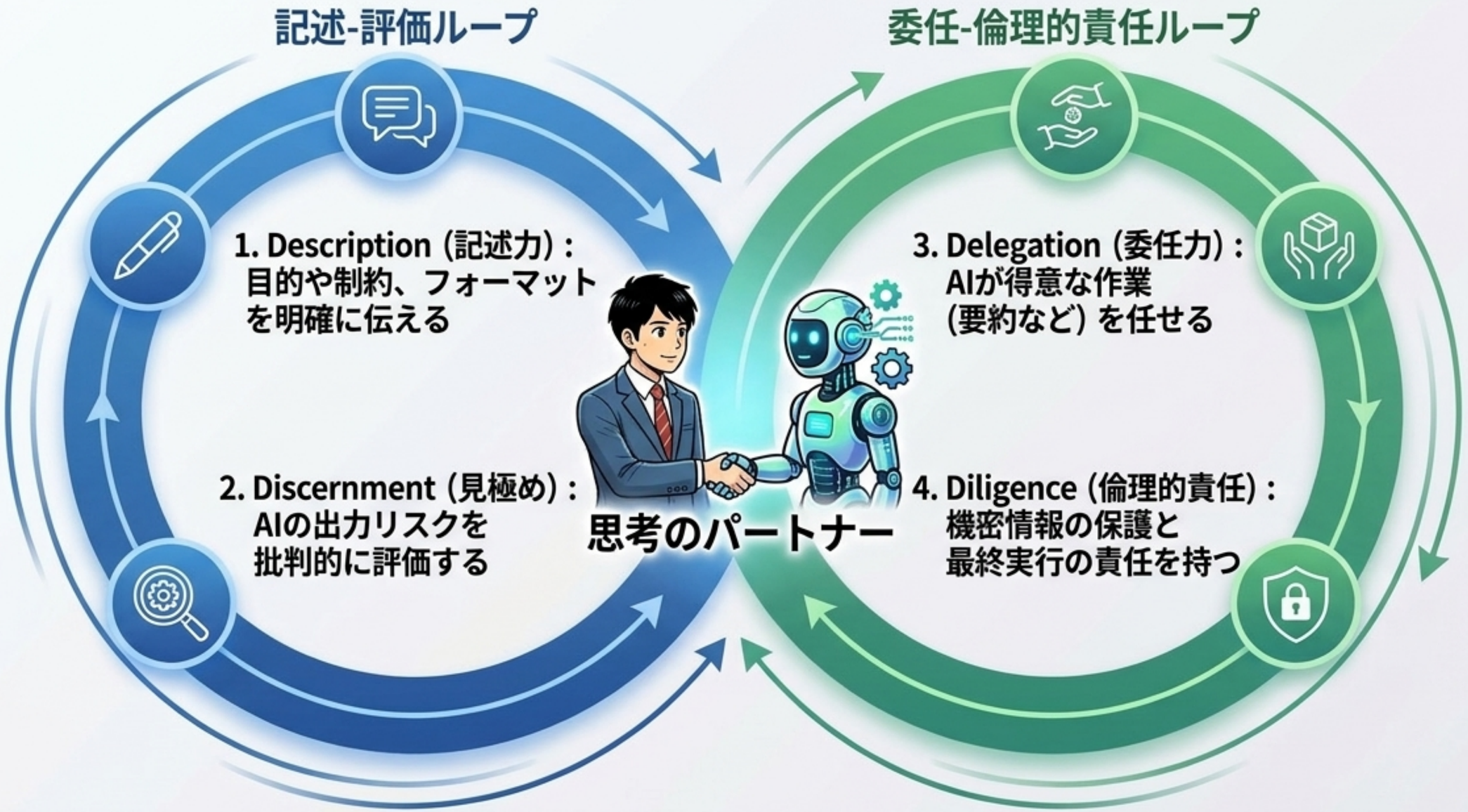




解決策：/clear コマンドを実行する

会話を引きずらず、机の上を一度リセットする。
一番新しい git status とエラーだけを改めて渡し直すことで、AI の推論精度が復活する。

統合：AIを乗りこなす「4D AI Fluency」フレームワーク



4D実践マトリクス：Git操作のチェックリスト

	Delegation（任せる）	Description（伝える）	Discernment（見極める）	Diligence（責任を持つ）
1. メッセージ作成	diffの要約を任せる	「件名50文字以内」「提供情報のみ」と制約	作話（存在しない機能）がないか確認	コミット前に必ず人間が目視チェック
2. セルフチェック	コミット前のバグ・機密チェック	「厳しく異議を唱えよ」と追従性を排除	単なる同調（過剰な肯定）を見抜く	実データ（パスワード等）をマスキングして送信
3. エラー解消	エラー原因の解析と手順の立案	状態(status)と履歴(log)をセットで渡す	破壊的コマンド(reset --hard)のリスク判断	必ずバックアップブランチを作ってから実行

さあ、新しい開発体験への第一歩を踏み出そう



最初の1アクションを今すぐ試そう！

- 1. ターミナルを開いて git status を実行し、結果をコピー。
- 2. Claudeにこう話しかける: 「この結果を見て、今の作業状況を要約して
  コミットメッセージ案を作って！」

思考のパートナーと共に、素晴らしい開発ライフを。