

ソースコードレビューチェックリスト

このチェックリストは、提供されたPythonソースコードの品質、セキュリティ、パフォーマンス、保守性を評価するためのものです。

1. セキュリティ

- * ****機密情報のハードコーディング:**** APIキー、パスワード、認証情報などがソースコード内に直接記述されていないか？
- * ****認証情報の処理:**** パスワードなどの認証情報は、ハッシュ化やソルト処理など、安全な方法で保存・比較されているか？平文での比較は避けるべき。
- * ****ユーザー入力の検証:**** ファイルパス、データベースクエリ、シェルコマンドなど、ユーザーからの入力が直接使用される箇所で、適切なサニタイズや検証が行われているか？（例: パストラバーサル、SQLインジェクションの脆弱性がないか）
- * ****広範な例外捕捉:**** `except Exception as e:` のように広範な例外を捕捉し、セキュリティ上重要なエラー情報が隠蔽されていないか？

2. パフォーマンス

- * ****非効率なループ処理:**** ループ内で繰り返し同じ計算やリソースを消費する処理が行われていないか？（例: ループ内で毎回ハッシュ計算、データベースクエリなど）
- * ****データ構造の選択:**** 処理対象のデータ量に対して、適切なデータ構造（リスト、セット、辞書など）が選択されているか？
- * ****不必要なリソース消費:**** 不要なファイルI/O、ネットワークリクエスト、メモリ割り当てなどがないか？

3. 可読性と保守性

- * ****命名規則:**** 変数名、関数名、クラス名は、その目的や内容を明確に表しているか？（例: PEP 8に準拠しているか）
- * ****コメントとドキュメンテーション:**** コードの意図、複雑なロジック、重要な設計判断について、適切なコメントやDocstringが記述されているか？
- * ****マジックナンバー/マジックストリング:**** コード中に意味不明な数値や文字列が直接記述されていないか？定数として定義すべき。
- * ****関数の単一責任:**** 各関数やメソッドは、一つの明確な責任のみを持っているか？（単一責任の原則 - SRP）
- * ****コードの重複:**** 同じようなロジックが複数の箇所で繰り返されていないか？DRY (Don't Repeat Yourself) 原則に従っているか？

4. エラーハンドリング

- * ****具体的な例外捕捉:**** 予期されるエラー（ファイルが見つからない、不正な入力など）に対して、具体的な例外（`FileNotFoundError`, `ValueError` など）を捕捉し、適切に処理しているか？
- * ****エラーメッセージ:**** エラー発生時に出力されるメッセージは、ユーザーにとって分かりやすいか、またはデバッグに役立つ情報を含んでいるか？
- * ****リソースの解放:**** ファイルやネットワーク接続などのリソースは、エラー発生時でも確実に閉じられているか？（`with` ステートメントの使用など）

5. テスト容易性

- * ****モジュール性:**** コードは独立した小さな単位に分割されており、それぞれを単体テストしやすいか？
- * ****依存関係:**** 外部サービスやデータベースへの依存が明確で、テスト時にモック

ク化しやすいか？