

以下は、提示コード（DocumentAsk.aspx.cs 相当）の動作に対するテストケース（概要）です。想定対象は ASP.NET WebForms 画面、OpenAI API（chat/completions, embeddings）、SQL Server(DocumentEmbeddings)です。各ケースは観点・前提・入力/操作・期待結果の要点のみを列挙しています。

#### 前提・準備

- AppSettings/ConnectionStrings
  - OpenAI\_API\_Key（有効/無効/未設定の各パターン）
  - OpenAI\_ChatModel（例: gpt-5, gpt-4o）
  - OpenAI\_EmbeddingModel（例: text-embedding-3-small）
  - OpenAI\_Temperature（数値/不正値/未設定）
  - ConnectionStrings["ConnStr"]
- DB テストデータ（DocumentEmbeddings）
  - 正常行: Title, ChunkIndex, Chunk, Embedding(JSON 配列) 正常
  - ChunkIndex = NULL, 重複 ChunkIndex(同一 Title で 1 が複数など)
  - Embedding JSON 不正、空配列、長さ不一致など
  - Title に拡張子含むもの: A.cs, B.sql, Guide.docx など
  - レコード件数: 0 件、少量、数千件
- HTTP スタブ/モック
  - chat/completions: 200/429/500/タイムアウト/200 だが空応答
  - embeddings: 同上
- 画面要素

- txtQuestion, chkUseRAG, btnAsk, litAnswer, lblMessage, pnlAnswer, btnExportWord, btnExportPDF

#### A. 画面初期化/Page\_Init

- A-1: 正常初期化

- 前提: 有効な OpenAI\_API\_Key

- 操作: 初回アクセス

- 期待: HttpClient Authorization に Bearer 設定済、DefaultConnectionLimit=100、例外なし

- A-2: API キー未設定

- 期待: 初期化は通るが、以降の API 呼び出しで 401 が発生し、エラー表示・ログ出力

#### B. 問い合わせ実行 (btnAsk\_Click) 基本

- B-1: 入力未指定

- 入力: txtQuestion=""

- 期待: 「質問内容を入力してください。」表示、以降処理なし

- B-2: UseRAG=false の基本成功

- 入力: シンプルな質問、chat API 200/正常応答

- 期待: litAnswer が HTML エンコード表示、pnlAnswer 可視、Session に保存、Word/PDF ボタン有効

- B-3: UseRAG=true の基本成功

- 入力: シンプルな質問、BuildRagContextAsync 正常、chat API 200/正常応答

- 期待: RAG プロンプトで呼ばれ、B-2 同様に表示/ボタン有効

- B-4: chat API 429 → リトライ後成功

- 期待: 指数バックオフ後成功し、B-2 同様

- B-5: chat API 5xx → 最大リトライ後も失敗

- 期待: 例外→「エラー: Chat API エラー: ...」表示、ログ出力、ボタンは無効のまま

- B-6: chat API 200 だが choices[0].message.content 空

- 期待: 「ChatGPT の応答が空でした。」としてエラー表示・ログ出力

- B-7: JSON パース失敗

- 期待: 例外ハンドリング→エラー表示・ログ出力

- B-8: 長文質問（数万文字相当）

- 期待: タイムアウトしない（Timeout=2 分内）、成功または適切なエラー表示、フリーズしない

## C. RAG 文脈生成 BuildRagContextAsync

- C-1: Embedding API 正常 → 文脈抽出正常

- 期待: 上位 topN 抽出、neighbor=1 で前後チャンクが結合され context 生成

- C-2: Embedding API 429 → リトライ後成功

- 期待: 正常に context 生成

- C-3: Embedding API 5xx → リトライ枯渇

- 期待: 例外→btnAsk 側でエラー表示・ログ出力

- C-4: DB 0 件

- 期待: rows.Count==0 で空文字返却、最終プロンプトは抜粋空のまま作成、chat は成功

する

- C-5: Title フィルタ命中（質問内に「A.cs」）

- 入力: 「A.cs のエラーについて」

- 期待: WHERE Title LIKE '%A.cs%' が使われ、関連チャンクのみから抽出

- C-6: Title フィルタ不発（ファイル名なし）

- 期待: 全件対象

- C-7: ChunkIndex = NULL を含む

- 期待: タイトルごとに連番が振られ、近傍連結が正しく動作

- C-8: 同一 Title で ChunkIndex 重複を含む

- 期待: 重複検知・補正ロジックにより辞書化時にキー重複例外が出ない。近傍連結も期待通り

- C-9: Embedding JSON 不正（パース失敗）

- 期待: 例外→btnAsk 側でエラー表示・ログ出力

- C-10: ベクトル長が混在

- 期待: Cos 計算は短い方に合わせた min 長で実施され、例外なし

#### D. 類似検索補助関数

- D-1: GetEmbedding 正常/異常（429/5xx/タイムアウト）

- 期待: リトライポリシー通り、失敗時は例外送出・ログ

- D-2: FindSimilarChunks 正常

- データ: 数十件、Embedding 正常

- 期待: 類似度降順で topN 件返却、Similarity が降順
- D-3: FindSimilarChunks Embedding 空/不正を含む
  - 期待: 不正行はスキップ、残りで計算、全不正なら空リスト
- D-4: CosineSimilarity
  - ケース: 正規直交(=0)、同一ベクトル(=1)、長さ不一致、ゼロベクトル含む
  - 期待: それぞれ 0/1/正しく計算/0 を返却（例外なし）

## E. HTTP リトライ・タイムアウト

- E-1: 429 を 2 回返し 3 回目 200
  - 期待: 3 回目で成功、合計待機時間は  $0.5s \cdot 1^2 + 0.5s \cdot 2^2 = 2.5$  秒程度
- E-2: OperationCanceledException/TaskCanceledException を複数回返す
  - 期待: 最大リトライまで再試行し、最終結果に準拠
- E-3: 4xx (400/401/403) 返却
  - 期待: リトライしない。エラーとして上位に返却

## F. 表示/エンコード/セキュリティ

- F-1: 応答に HTML タグやスクリプトを含む
  - 期待: litAnswer は HtmlEncode 済で表示。XSS 不可
- F-2: 質問入力に特殊文字 (<>&" など)
  - 期待: 処理時は非エンコードだが、画面表示はエンコード。例外なし
- F-3: 超長文応答 (数万字)

- 期待: 画面表示可能、ボタン有効、レスポンス時間が許容範囲

## G. 書き出し (Word/PDF)

### - G-1: Word 出力 正常

- 前提: Session に回答有

- 操作: btnExportWord

- 期待: .docx ダウンロード、ヘッダ「ChatGPT の回答」+ 本文行単位、CompleteRequest で応答終了

### - G-2: Word 出力 回答なし

- 期待: 「出力する回答が見つかりません。」表示

### - G-3: PDF 出力 正常 (Windows/フォントあり)

- 期待: 日本語表示可、行単位で出力、ダウンロード成功

### - G-4: PDF 出力 フォント未存在 (非 Windows など)

- 期待: 例外→「PDF 出力エラー: ...」表示・ログ出力

### - G-5: 大きな回答 (複数ページ)

- 期待: 正しく複数ページ化、ファイル破損なし

### - G-6: ダウンロードヘッダ

- 期待: Content-Type/Content-Disposition が正しい拡張子で設定

## H. 設定/モデル別動作

### - H-1: ChatModel 未設定時のデフォルト

- 期待: "gpt-5" が使用され Temperature 未設定で送信

- H-2: ChatModel=gpt-4 系 + Temperature 設定あり

- 期待: Temperature がリクエストに乗る

- H-3: Temperature に不正値（文字列）

- 期待: 解析不可→null 扱いで未設定送信

- H-4: EmbeddingModel 未設定時のデフォルト

- 期待: text-embedding-3-small 使用

- H-5: ConnectionStrings["ConnStr"] 未設定/不正

- 期待: DB 接続時に例外→エラー表示・ログ

## I. パフォーマンス/大規模

- I-1: DocumentEmbeddings が数万行

- 期待: BuildRagContextAsync がタイムアウトせず完了、メモリ過剰使用なし

- I-2: 同時アクセス（10～50 ユーザー相当）

- 期待: リクエストが捌ける（DefaultConnectionLimit=100 の範囲内）、著しいエラー増加なし

- I-3: 連続クリック

- 操作: btnAsk を連打

- 期待: 二重送信による例外や競合が発生しない（必要があれば UI で制御）

## J. ログ/例外

- J-1: 例外発生時の DBLogger.LogError 呼び出し

- 期待: 各 catch でロギングされる (btnAsk\_Click, GetEmbedding, FindSimilarChunks, CosineSimilarity, Export 系)

- J-2: 例外メッセージのユーザー表示

- 期待: lblMessage に要点が表示され、スタックトレースは表示されない

## K. 文字種/ローカライズ

- K-1: 日本語/英語/記号/絵文字を含む質問・回答

- 期待: 正しく処理・表示・出力

- K-2: 改行コード差異 (CRLF/LF)

- 期待: 画面表示と Word/PDF の改行が崩れない

## テストデータ例 (概要)

- DB: DocumentEmbeddings

- 例 1: Title="A.cs", ChunkIndex=1,2,3, Embedding=正規長ベクトル、Chunk=短文

- 例 2: Title="B.sql", ChunkIndex=NULL, Embedding=正規、Chunk=短文 (NULL を連番補正確認)

- 例 3: Title="A.cs", ChunkIndex=2 (重複)、Embedding=正規、Chunk=別内容 (重複補正確認)

- 例 4: Title="Guide.docx", ChunkIndex=1, Embedding=不正 JSON (例外確認)

- 例 5: Title="Large.txt", 多数行 (パフォーマンス確認)

## 補足

- 本コードは UI 表示時に `HtmlEncode` しており XSS 対策は基本 OK。入力側は未エンコードで API 送信するため、API レスポンスと DB 保存 (`Session` のみに保存) に文字化けがないか留意。
- `PostWithRetryAsync` は `HttpContent` をファクトリで毎回生成しているため、再送時のボディ欠落は起きない想定。ステータスによりリトライ可否が異なる点を必ず網羅。
- PDF のフォント依存は環境差が大きいため、Windows/非 Windows での動作差を確認。

以上が網羅観点に基づくテストケース（概要）です。実装状況に応じて、HTTP スタブの用意、DB テストデータのシナリオ別投入、Selenium 等での E2E 試験を組み合わせ実施してください。