

## 単体テスト仕様 (DocumentAsk クラス)

### 1. 対象・前提

#### - 対象クラス/メソッド

- DocumentAsk.Page\_Init
- DocumentAsk.btnAsk\_Click
- DocumentAsk.GetEmbedding
- DocumentAsk.FindSimilarChunks
- DocumentAsk.CosineSimilarity
- DocumentAsk.PostWithRetryAsync
- DocumentAsk.btnExportWord\_Click
- DocumentAsk.btnExportPDF\_Click
- DocumentAsk.CreateJsonContent
- DocumentAsk.BuildRagContextAsync
- DocumentAsk.ExtractTitleLike

#### - 外部依存

- HttpClient (OpenAI API: chat/completions, embeddings)
- SQL Server 接続 (DocumentEmbeddings テーブル)
- ASP.NET WebForms 環境 (HttpContext/Session/Response/ScriptManager/UI コントロール)
- 設定値 (ConfigurationManager.AppSettings, ConnectionStrings)
- ロガー DBLogger

- OpenXML, iTextSharp
- テスト形態
  - 可能な限りユニットテスト（HttpClient は HttpResponseMessage モック、DB はテスト用 DB またはフェイク/スタブ、HttpContext は偽装）
  - SQL を伴う部分（FindSimilarChunks, BuildRagContextAsync の DB I/O）は小規模のテスト用 DB を用いた動作確認（②）を行う。難しい場合は机上確認を併記
- 事前条件（ビルド前提）
  - ソースには破損/重複行が含まれるため、明らかなタイポ/重複/欠落は修正済みであること
    - 例：重複したクラス断片、壊れたコメント、途中で切れたコード、二重定義、変数名の破損など
  - DTO 定義（ChatCompletionRequestDto, ChatMessageDto, ChatCompletionResponseDto, EmbeddingRequestDto, EmbeddingResponseDto 等）は正しく実装済み
  - BasePage 及び UI コントロール（txtQuestion, lblMessage, pnlAnswer, litAnswer, chkUseRAG, btnExportExcel/PDF など）が利用可能

## 2. テスト観点（①～⑨対応の方針）

- ① 正常/異常ルート：各メソッドに対し成功/失敗の代表ケースを網羅
- ② 動作での全ルート確認：リトライ分岐、RAG 近傍連結、エンコードなど、目視ではなく実行で確認（困難箇所は机上確認）
- ③ 外部処理：OpenAI API 呼出、DB、ファイル出力（Word/PDF）、ログ出力の境界をモック/スタブで確認
- ④ テーブル検索：1 件目/中間/最終の抽出・結合・類似度ソートをテストデータで確認

- ⑤ 繰り返し：ループの境界（1件目、中間、最大、最大+1）と **neighbor** 範囲、リトライ回数、文書行分割の繰り返し
- ⑥ 変数上限下限：topN、neighbor、Temperature、ベクトル長、空/極端値の境界
- ⑦ エラーチェック：各種例外/非正ケースのユーザー表示メッセージ、ログ出力、再スロー有無
- ⑧ 演算：コサイン類似のゼロ除算回避、長さ不一致、負値を含むベクトルの取り扱い
- ⑨ その他：XSS エスケープ、レスポンスヘッダ、コンテンツタイプ、タイムアウト・キャンセル時のハンドリング

### 3. テストデータ（代表）

- AppSettings/ConnString
- OpenAI\_API\_Key: 正常文字列／未設定／空文字
- OpenAI\_ChatModel: "gpt-5"／"gpt-4o"／未設定（デフォルト適用）
- OpenAI\_EmbeddingModel: 正常／未設定（デフォルト適用）
- OpenAI\_Temperature: "0.2"／"abc"（不正→null）
- ConnectionStrings["ConnStr"]: 正常／不正（接続失敗）
- DB テーブル DocumentEmbeddings（テスト用）
- 最低限の列: Id, Title, ChunkIndex, Chunk, Embedding(JSON 配列)
- ケース別に以下を用意
  - A: 3 件（Title=T1, ChunkIndex=1.3, ベクトル長=4）
  - B: ChunkIndex NULL 含む（自動連番補正の確認）
  - C: 同一 Title 内で ChunkIndex 重複（辞書化の重複回避）

- D: Embedding JSON が空/不正/長さ不一致/0 ベクトル
- OpenAI API モック応答
- chat/completions: 正常 JSON/choices 空/HTTP 4xx/5xx/429/タイムアウト
- embeddings: 正常 JSON (ベクトル長=4) /空データ/HTTP エラー
- UI 入力
- txtQuestion: 空/通常文/RAG 有効/XSS 文字列
- PDF/Word 出力
- Session["ChatAnswer"] = null/""/複数行文字列 (日本語・改行・長文)
- フォントファイル存在/不存在 (PDF 出力)

#### 4. テストケース (抜粋・要点)

##### 4-1 Page\_Init (①③⑥)

- TC-PI-01 正常 : 有効な API Key 設定時、Authorization が "Bearer <key>" で設定されること、DefaultConnectionLimit=100
- TC-PI-02 異常 : API Key 未設定時、AuthenticationHeaderValue の生成で例外になるか、許容される場合は以降の呼出で 401 相当になること (仕様決定に合わせ期待値定義)。ログ有無

##### 4-2 btnAsk\_Click (①②③⑦⑨)

- TC-BA-01 入力空 : txtQuestion 空→"質問内容を入力してください。"、処理中断、Export ボタン無効、パネル非表示
- TC-BA-02 正常 (RAG 無効) : chat/completions 正常応答→litAnswer は HtmlEncode され

たテキスト、パネル可視、セッション格納、Export ボタン有効

- TC-BA-03 正常（RAG 有効）：BuildRagContextAsync が呼ばれ、コンテキスト付与プロンプトで呼出されること（モックで検証）
- TC-BA-04 Chat API エラー：HTTP 500→ApplicationException を捕捉→"エラー: Chat API エラー: ..." 表示、ログ出力、Export 無効
- TC-BA-05 応答空：choices[0].message.content 空→"エラー: ChatGPT の応答が空でした。" 表示、ログ出力
- TC-BA-06 Temperature 設定：ChatModel="gpt-5" のとき temperature 未指定、"gpt-4o" のとき AppSettings が数値なら設定、非数値は未設定
- TC-BA-07 finally 動作：ScriptManager.RegisterStartupScript("hideOverlay") が呼ばれること（フレームワークモックで確認、難しければ机上確認）

#### 4-3 GetEmbedding (①③⑥⑦)

- TC-GE-01 正常：embeddings 正常応答→float[] へ変換され返却
- TC-GE-02 異常：HTTP 400/500→ApplicationException スロー、ログ出力
- TC-GE-03 異常：data[0].embedding 空→ApplicationException("Embedding が空です。") スロー、ログ出力

#### 4-4 FindSimilarChunks (①②③④⑤⑥⑧)

- TC-FS-01 正常：3 件投入（A データ）→CosineSimilarity で降順に topN 件が返る
- TC-FS-02 topN 境界：topN=1/topN>件数（全件返却）/topN=0（空）
- TC-FS-03 Embedding 不正：JSON 不正や空→該当行スキップ（例外で全体中断しないこと）
- TC-FS-04 ベクトル長不一致：CosineSimilarity は短い方に合わせ計算（0 にならないこ

と)

- TC-FS-05 0 ベクトル :  $n1=0$  または  $n2=0$  のとき類似度 0

#### 4-5 CosineSimilarity (①⑤⑥⑧)

- TC-CS-01 一致ベクトル→1.0 近傍

- TC-CS-02 直交→0

- TC-CS-03 逆向き→-1 近傍

- TC-CS-04 長さ不一致→短い方の長さで計算

- TC-CS-05 全要素 0→0 (ゼロ除算回避)

- TC-CS-06 例外系 (null 引数) →catch で 0、ログ出力

#### 4-6 PostWithRetryAsync (①②③⑤⑥⑦)

- TC-PR-01 成功即時 : 1 回で 200→返却

- TC-PR-02 429→指数バックオフで再試行し成功→戻り値成功、呼出回数=試行回数

- TC-PR-03 5xx→指定回数まで再試行し最終成功→成功返却

- TC-PR-04 非リトライ系 (400, 401, 403 など) →即返却 (リトライしない)

- TC-PR-05 タイムアウト (TaskCanceledException/OperationCanceledException) →最大回数まで再試行、最終成功/失敗の挙動

- TC-PR-06 maxRetry=0/1/大値 : 境界動作

- TC-PR-07 contentFactory 毎 호출されること (Content 再生成の確認。モックで CreateJsonContent 呼出回数検証)

#### 4-7 btnExportWord\_Click (①②③⑤⑦⑨)

- TC-EW-01 セッション無し : Session["ChatAnswer"]=null/"-"出力する回答が見つかりません。"
- TC-EW-02 正常 : 複数行回答→docx がダウンロード可能、ContentType と Content-Disposition、SuppressContent=true、CompleteRequest 呼び出し
- TC-EW-03 例外 : ストリーム書込み失敗など→"Word 出力エラー: ..." 表示、ログ出力
- TC-EW-04 改行/日本語 : 改行が段落化され、日本語が文字化けしない (OpenXML のみで実質 OK)

#### 4-8 btnExportPDF\_Click (①②③⑤⑦⑨)

- TC-EP-01 セッション無し : メッセージ表示
- TC-EP-02 正常 : PDF 出力、ヘッダ/コンテンツ検証、SuppressContent/CompleteRequest
- TC-EP-03 フォント未存在 : BaseFont.CreateFont 失敗→"PDF 出力エラー: ..."、ログ出力
- TC-EP-04 長文/複数行 : 全行追加される

#### 4-9 CreateJsonContent (①⑥⑦)

- TC-CJ-01 正常 : application/json ヘッダ、UTF-8、JSON 文字列が含まれる
- TC-CJ-02 body=null : シリアライズ可能か要確認。不可の場合は例外→ログ/ハンドリング (机上確認可)

#### 4-10 BuildRagContextAsync (①②③④⑤⑥⑦⑨)

- TC-RG-01 正常 (タイトル無) : 全件読み出し→Cosine 類似度で topN 選抜→neighbor=1 で前後連結→"---" 区切りで返る

- TC-RG-02 正常（タイトル有）：ExtractTitleLike に一致→WHERE Title LIKE が効く（パラメータ値確認）
- TC-RG-03 embeddings エラー：ApplicationException スロー
- TC-RG-04 qvec 空：空文字返却（RAG は不足時注記の上流方針に委ねる）
- TC-RG-05 ChunkIndex NULL 混在：titleCounters により連番補正され、重複せず辞書化可能
- TC-RG-06 ChunkIndex 重複：辞書化前の GroupBy→First により重複キー解消される
- TC-RG-07 neighbor 境界：0／1／大値（範囲外は無視され、重複は used で防止）
- TC-RG-08 rows=0：空文字返却
- TC-RG-09 topN 境界：0／1／>rows
- TC-RG-10 ベクトル長不一致：Cos 関数で短い方に合わせ正常計算

#### 4-11 ExtractTitleLike (①⑥⑨)

- TC-ET-01 正常：拡張子 .cs/.sql 等を含む語を抽出
- TC-ET-02 大文字拡張子：大文字小文字無視
- TC-ET-03 区切り文字：日本語句読点/括弧/全角空白で分割される
- TC-ET-04 不一致：null 返却

#### 4-12 セキュリティ/XSS (⑨)

- TC-SC-01 応答に HTML を含む場合、litAnswer には HtmlEncode 済みとなる（<script> がエスケープ）

#### 4-13 設定値の境界 (⑥)

- TC-CF-01 OpenAI\_Temperature=数値→設定反映 (非 gpt-5)
- TC-CF-02 OpenAI\_Temperature=非数値/未設定→null として送信しない
- TC-CF-03 モデル既定値 : ChatModel 未設定時 "gpt-5"、EmbeddingModel 未設定時 既定を使用

#### 5. 外部処理の確認方法 (③)

##### - HttpClient

- HttpResponseMessage をモックし、要求 URL/メソッド/ヘッダ/ボディ (モデル・メッセージ・temperature 有無) を検証
- 429/5xx/タイムアウト時のリトライ回数・間隔 (少なくとも回数を検証。時間は猶予付きアサート)

##### - DB

- ローカルのテスト用 SQL Server (LocalDB など) で DocumentEmbeddings を作成
- 1 件目/中間/最終 (④) を識別できるようデータ投入し、FindSimilarChunks/BuildRagContextAsync の戻りと順序を検証
- 代替 : IDataReader/IDbConnection 抽象化がないため、完全ユニットが難しければ統合寄りの動作確認 (②)

##### - ファイル出力

- HttpResponseMessage のフェイク (MemoryStream) でヘッダ/ボディ /SuppressContent/CompleteRequest 呼出確認
- 生成された docx/pdf の最小検査 (先頭数バイトのマジック/開けること)

## 6. 繰り返し・境界 (⑤)

- `PostWithRetryAsync` の `attempt` ループ : 1回、最大回数、最大回数+1 (超過しないこと)
- `BuildRagContextAsync` の `neighbor` 連結ループ : `hit.ChunkIndex` で 先頭/中間/末尾、`neighbor=0/1/大値`
- Word/PDF 出力の行追加ループ : 1行、複数行、空行含む

## 7. エラーメッセージ確認 (⑦)

- `btnAsk_Click` : 入力空、Chat API エラー、応答空
- `GetEmbedding` : Embedding API エラー、Embedding 空
- `Export` : Word/PDF 出力エラー
- 表示文言は要件通り日本語で一致していること (末尾句読点含む)

## 8. 演算箇所 (⑧)

- `CosineSimilarity` のゼロ除算回避条件 (`n1==0 || n2==0`) を網羅
- 直交/逆向き/正規化済み/長さ不一致の妥当性
- 負値/小数の扱い

## 9. ② で動作確認が難しい箇所と机上確認

- `ScriptManager.RegisterStartupScript` の呼出 (完全な WebForms ランタイムが必要なため、呼出有無のみモックで確認。実ブラウザ連動は結合/手動で確認)
- `iTextSharp` のフォント依存描画の完全一致 (PDF ビジュアルは手動確認も併用)

- SQL の接続障害/トランジェント障害再現（環境制約がある場合は例外シナリオをハンドラでモックし机上確認）

## 10. 想定される修正・注意

- API Key 未設定時の挙動を明確化（初期化で例外にするか、呼出時に検出するか）
- DTO/JSON モデルの互換性チェック（プロパティ名と `_jsonOpt` のポリシー整合）
- 例外時のユーザー向け文言とログ詳細の分離（内部情報を画面へ出さない）
- DB 読み出し時の Embedding JSON パース失敗の扱い（現在は例外で全体中断の可能性、要件に応じスキップ推奨）
- `BuildRagContextAsync` の `rows.Count==0` で空返却の仕様確認（UI 上の文言は `btnAsk_Click` 側で制御）

## 11. 実施環境

- テストフレームワーク： NUnit/xUnit/MSTest のいずれか
- モック： Moq/JustMock/自作 `HttpMessageHandler`
- SQL： LocalDB/コンテナ SQL Server、初期化スクリプトでセットアップとテアダウン
- WebForms フェイク： `HttpContext/HttpResponseBase` のテストダブル

## 12. 受入基準

- ①～⑧の観点に対し、正常/異常・境界・繰返しが網羅され、期待結果が満たされること
- 外部処理の主要分岐（成功/429/5xx/タイムアウト）が動作で確認済みであること
- セキュリティ（XSS エスケープ）が確認できていること

- 重大な未処理例外が UI をクラッシュさせず、ログが記録されること

以上をもって単体テストを設計・実施してください。必要に応じ、結合テストで補完（特に WebForms 実行時のレスポンス挙動と PDF レイアウト）。