BusinessContextParserService

September 4, 2026

BusinessContextParserService

Kind: Service

Source: `atloria-monorepo/apps/api/src/ai/services/business-context-parser.service.ts`

Business Context Parser Service

Extracts text from PDF, Markdown, and Word files, then uses AI to summarize and structure the content into actionable business context for documentation.

`BusinessContextParserService` extracts source content from PDF, Markdown, and Word files and converts it into structured `BusinessContext` using AI-assisted parsing. It provides a backend service boundary for turning unstructured business documentation or raw text into context that can be reused by documentation and prompt-generation workflows.

Methods

Method	Signature	Returns	Description
`parseContextFiles`	`parseContextFiles(filePaths: string[])`	`Promise<BusinessContext>`	Parse business context files (PDF, Markdown, Word)
`parseRawText`	`parseRawText(rawText: string)`	`Promise<BusinessContext>`	Parse raw business context text (already extracted from files)
`formatContextForPrompt`	`formatContextForPrompt(context: BusinessContext)`	`string`	Format business context as a readable string for AI prompts

Dependencies

- `AzureClaudeProvider`

Where it refuses work

- `BusinessContextParserService` stops the work with `Error` when `allText.length === 0`.

When something fails

- `BusinessContextParserService` handles failure in 4 places: it lets it reach the caller in 2, logs it and continues in 1, and turns it into a return value in 1.

Diagram

```
sequenceDiagram
  participant Caller as API/Documentation Workflow
  participant Parser as BusinessContextParserService
  participant Extractor as File Text Extractors
  participant AI as AI Provider

  Caller->>Parser: parseContextFiles(files)
  Parser->>Extractor: Extract PDF/Markdown/Word text
  Extractor-->>Parser: Combined raw text
  Parser->>AI: Summarize and structure business context
  AI-->>Parser: BusinessContext
  Parser-->>Caller: BusinessContext

  Caller->>Parser: formatContextForPrompt(context)
  Parser-->>Caller: Prompt-ready context string
```

Usage

```
import { BusinessContextParserService } from './ai/services/business-context-parser.service';

class DocumentationWorkflow {
  constructor(
    private readonly businessContextParser: BusinessContextParserService,
  ) {}

  async buildDocumentationContext(files: Express.Multer.File[]) {
    const context = await this.businessContextParser.parseContextFiles(files);

    const promptContext =
      this.businessContextParser.formatContextForPrompt(context);

    return {
      context,
```

```
      promptContext,
    };
  }

  async parsePastedNotes(notes: string) {
    return this.businessContextParser.parseRawText(notes);
  }
}
```

AI Coding Instructions

- Keep file extraction responsibilities aligned with supported formats: PDF, Markdown, and Word documents; validate unsupported or empty inputs before invoking AI parsing.
- Use `parseContextFiles()` for uploaded documents and `parseRawText()` for pasted or already-extracted content rather than duplicating extraction logic in callers.
- Treat AI-generated `BusinessContext` as structured application data: validate required fields and handle incomplete or malformed model responses safely.
- Use `formatContextForPrompt()` when passing business context into downstream AI prompts to preserve a consistent prompt representation.
- Ensure integrations propagate extraction and AI-provider failures with actionable errors without exposing sensitive document content in logs.

Relationships

- DEPENDS_ON → `AzureClaudeProvider`

Referenced By

- `AIModule` (MODULE_PROVIDES)
- `AIModule` (MODULE_EXPORTS)
- `SyncController` (DEPENDS_ON)