ryanjoachim / **mcp-rtfm** Public

Code | Issues 1 | Pull requests 1 | Actions | Projects | Security and quality

Commit `e6f0686`

ryanjoachim committed 2 weeks ago

`https://github.com/ryanjoachim/mcp-rtfm/issues/5`

`expansion`

1 parent `dc98019` commit `e6f0686`

4 files changed +51 -7

Top

Filter files…

- src
  - handlers
    - doc-handlers.ts
    - refresh-handlers.ts
    - search-handlers.ts
  - validation.ts

Search within code

src/handlers/doc-handlers.ts

```
@@ -8,8 +8,8 @@ import { McpError, ErrorCode } from "@modelcontextprotocol/sdk/types.js";
 8  8
 9  9   import { contextManager } from "../project-context.js";
10 10   import { analyzeContent, categorizeContent, updateMetadata, updateSearchIndex } from "../content.js";
11     - import { validateProjectPath } from "../validation.js";
```

```
12      - import { handleToolError, freshTimestamp, getDocsPath } from "../utils.js";
        11 + import { validateProjectPath, validateDocFile } from "../validation.js";
        12 + import { handleToolError, freshTimestamp } from "../utils.js";
13      13   import { logger } from "../logger.js";
14      14   import { ReadDocSchema, UpdateDocSchema } from "../schemas.js";
15      15
         @@ -29,8 +29,9 @@ export const readDoc = async (request: CallToolRequest) =>
         {
29      29       );
30      30     }
31      31
        32 +   const filePath = validateDocFile(docFile, projectPath);
        33 +
32      34     try {
33      -     const filePath = `${getDocsPath(projectPath)}/${docFile}`;
34      35       const content = await fs.readFile(filePath, "utf8");
35      36
36      37       return {
         @@ -59,10 +60,11 @@ export const updateDoc = async (request: CallToolRequest)
         => {
59      60       );
60      61     }
61      62
        63 +   const filePath = validateDocFile(docFile, projectPath);
        64 +
62      65     const ctx = contextManager.getContext(projectPath);
63      66
64      67     try {
65      -     const filePath = `${getDocsPath(projectPath)}/${docFile}`;
66      68
67      69       // Read current file content
68      70       let fileContent = await fs.readFile(filePath, "utf8");
```

src/handlers/refresh-handlers.ts

```
         @@ -17,7 +17,7 @@ import {
17      17     isGitRepository, detectGitChanges, detectFileChanges,
18      18     generateRefreshSuggestions, calculateSummary, applySuggestion
19      19   } from "../changes.js";
20      -  import { validateProjectPath } from "../validation.js";
```

```
      20  + import { validateProjectPath, validateDocFile } from "../validation.js";
21    21    import { logger } from "../logger.js";
22    22    import { RefreshDocumentationSchema } from "../schemas.js";
23    23    import type { DocMetadata } from "../types.js";
          ↓ @@ -60,14 +60,18 @@ export const refreshDocumentation = async (request:
          ↑ CallToolRequest) => {
60    60    async function handleRefreshAnalyzeMode(ctx: ReturnType<typeof
            contextManager.getContext>, projectPath: string, options: { docFile?: string;
            metadata?: Partial<Pick<DocMetadata, "title" | "category" | "tags">> }) {
61    61      const { docFile, metadata } = options;
62    62
      63  +   if (docFile) {
      64  +     validateDocFile(docFile, projectPath);
      65  +   }
      66  +
63    67      const signature = await detectProjectSignature(projectPath);
64    68      const docsPath = getDocsPath(projectPath);
65    69      const docsToUpdate = docFile ? [docFile] : await getActualDocs(docsPath);
66    70
67    71      const results: Array<{ file: string; updated: boolean; message: string }> =
            [];
68    72
69    73      for (const doc of docsToUpdate) {
70        -       const filePath = `${docsPath}/${doc}`;
      74  +       const filePath = validateDocFile(doc, projectPath);
71    75        let content: string;
72    76        try {
73    77          content = await fs.readFile(filePath, "utf8");
          ↓
```

src/handlers/search-handlers.ts

```
          ↑ @@ -7,7 +7,7 @@ import { McpError, ErrorCode } from
          ↑ "@modelcontextprotocol/sdk/types.js";
7     7
8     8     import { contextManager } from "../project-context.js";
9     9     import { searchDocContent, findRelatedDocs } from "../content.js";
10        -  import { validateProjectPath } from "../validation.js";
      10  +  import { validateProjectPath, validateDocFile } from "../validation.js";
11    11    import { handleToolError } from "../utils.js";
12    12    import { SearchDocsSchema, GetRelatedDocsSchema } from "../schemas.js";
```

```
13  13
    @@ -69,6 +69,8 @@ export const getRelatedDocs = async (request:
    CallToolRequest) => {
69  69        );
70  70      }
71  71
        +   validateDocFile(docFile, projectPath);
        +
72  74      const ctx = contextManager.getContext(projectPath);
73  75
74  76      try {
```

src/validation.ts

```
    @@ -4,6 +4,7 @@
4   4
5   5   import * as fs from "fs/promises";
6   6   import path from "path";
        + import { McpError, ErrorCode } from "@modelcontextprotocol/sdk/types.js";
7   8   import { logger } from "./logger.js";
8   9
9   10  // Path traversal pattern
    @@ -51,6 +52,41 @@ export const validateProjectPath = async (projectPath:
    string): Promise<PathVali
51  52    return { isValid: true };
52  53  };
53  54
        + /**
        +  * Validate that a docFile parameter is safe and resolve it to an absolute path
        +  * within the .handoff_docs directory. Prevents path traversal (CWE-22).
        +  *
        +  * @param docFile    The user-supplied doc filename (e.g. "techStack.md")
        +  * @param projectPath The resolved absolute project path
        +  * @returns The safe, resolved absolute path to the doc file
        +  * @throws McpError if docFile contains traversal sequences or escapes the docs
        dir
        +  */
        + export const validateDocFile = (docFile: string, projectPath: string): string =>
        {
        +   if (!docFile || typeof docFile !== "string") {
```

```
      66  +     throw new McpError(ErrorCode.InvalidParams, "docFile must be a non-empty
          string");
      67  +   }
      68  +
      69  +   // Reject path separators — docFile must be a simple filename, not a path
      70  +   if (/[\/\\]/.test(docFile)) {
      71  +     throw new McpError(ErrorCode.InvalidParams, "docFile must not contain path
          separators");
      72  +   }
      73  +
      74  +   // Reject parent-directory traversal sequences
      75  +   if (docFile.includes("..")) {
      76  +     throw new McpError(ErrorCode.InvalidParams, "docFile must not contain path
          traversal sequences");
      77  +   }
      78  +
      79  +   // Resolve and verify the final path stays within .handoff_docs
      80  +   const docsDir = path.resolve(projectPath, ".handoff_docs");
      81  +   const resolvedPath = path.resolve(docsDir, docFile);
      82  +
      83  +   if (!resolvedPath.startsWith(docsDir + path.sep)) {
      84  +     throw new McpError(ErrorCode.InvalidParams, "docFile escapes the
          documentation directory");
      85  +   }
      86  +
      87  +   return resolvedPath;
      88  + };
      89  +
54    90    // ============================================================================
55    91    // Documentation Validation
56    92    // ============================================================================
```

Comments 0

Please sign in to comment.