

Fix AI Q&A Generation and Integrate Chatbot Tools

Overview

This PR fixes two critical issues with the AI features:

1. **Q&A Training System generating 0 Q&A pairs** despite processing 126 files
2. **AI Chatbot lacking file system access and code execution capabilities**








Issue 1: Q&A Training Generation Producing 0 Results

Root Cause

The Q&A generation system was processing files correctly but failing to generate any Q&A pairs due to:

- Silent Ollama API failures
- Weak JSON parsing that couldn't handle various response formats
- No response validation before parsing
- Missing error reporting to UI
- No timeout handling

Fixes

-  Added 60-second timeout protection per file
-  Improved JSON parsing with 3 strategies (flexible regex, code blocks, fallback)
-  Enhanced error handling with detailed logging
-  Better prompt engineering for consistent JSON output
-  Comprehensive job tracking with error collection
-  Validates Ollama responses before parsing
-  Filters Q&As by minimum content length

Changes to `src/lib/services/qa-generator.ts`








- Enhanced Ollama API calls with timeout and validation (60s timeout)
- Improved JSON parsing with multiple strategies and retry logic
- Better Q&A validation (10+ char questions, 20+ char answers)
- Comprehensive error tracking and reporting to UI
- Improved prompt with explicit JSON-only instructions

Issue 2: Chatbot Lacking File System Access

Root Cause

The AI chatbot was using the basic `/api/chat` endpoint which only called Ollama directly without any tool integration. The AI tools framework existed but wasn't connected to the main chat interface.

Fixes

-  Integrated AI tools framework into main chat endpoint
-  Added tool execution loop (max 5 iterations)
-  Implemented tool parsing from AI responses
-  Added file system tools (read, write, list, search, info)
-  Added code execution tools (Python, JS, shell, npm, analyze)
-  Enhanced system prompt with tool descriptions
-  Tool results formatted and fed back to AI

Changes to `src/app/api/chat/route.ts`

- Integrated AI tools framework with imports and types
- Added tool discovery and prompting in system message
- Implemented tool execution loop with max 5 iterations
- Added helper functions: `buildToolsPrompt`, `parseToolCalls`, `executeTools`, `formatToolResults`
- Enhanced response with tool metadata (`toolCalls`, `toolResults`, `iterations`)

Available AI Tools

File System Tools

- **`read_file`**: Read contents of a file
- **`write_file`**: Write content to a file
- **`list_directory`**: List files in a directory
- **`search_files`**: Search for files by pattern
- **`get_file_info`**: Get file metadata

Code Execution Tools

- **`execute_python`**: Execute Python code
- **`execute_javascript`**: Execute JavaScript code
- **`execute_shell`**: Execute shell commands
- **`run_npm_command`**: Run npm commands
- **`analyze_code`**: Analyze code structure

Testing

Q&A Generation

1. Start Ollama: `ollama serve`
2. Navigate to AI Hub → Q&A Training
3. Click “Generate from Repository”
4. Watch progress bar and status messages
5. Verify Q&As are created in the list

Chatbot Tools

1. Ask: “Can you read the package.json file?”
2. Ask: “What files are in the `src/lib/ai-tools` directory?”
3. Ask: “Find all TypeScript files related to AI tools”
4. Verify AI uses appropriate tools and returns results

Error Handling

Q&A Generation

- Timeout protection (60 seconds per file)
- Connection error detection
- Invalid response structure detection
- JSON parsing with multiple strategies
- Detailed error messages in job status
- Progress tracking with file counts

Chatbot Tools

- Tool execution error handling
- Invalid tool call detection
- Parameter validation
- Timeout protection (30 seconds per tool)
- Graceful degradation if tools fail

Security

File System Access

- Tools sandboxed to project directory
- Path validation prevents directory traversal
- Read-only operations by default

Code Execution

- Execution timeout limits
- Memory limits enforced
- Sandboxed environment

Files Changed

- `src/lib/services/qa-generator.ts` (180 lines changed)
- `src/app/api/chat/route.ts` (160 lines added)
- `AI_FEATURES_FIX_SUMMARY.md` (comprehensive documentation)

Related Issues

Fixes the issues reported where:

- Q&A Training shows “Generated: 0 Q&As” despite processing 126 files
- Chatbot can’t access file system or execute code despite PR #111 adding the framework

Both issues are now fully resolved with extensive error handling and logging.