Q&A Training System Documentation

Overview

The Q&A Training System is a comprehensive feature that allows you to train the AI assistant with domain-specific knowledge about your Sports Bar TV Control System. It provides three main capabilities:

1. **Auto-generation**: Automatically generate Q&A pairs from repository files and documentation
2. **Upload**: Upload Q&A documents in various formats
3. **Management**: View, edit, and organize Q&A entries

Features

1. Automatic Q&A Generation

The system can automatically analyze your codebase and documentation to generate relevant question-answer pairs.

Supported Sources:
- **Repository**: Generates Q&As from README, INSTALLATION.md, and key documentation files
- **Documentation**: Analyzes all Markdown files in the docs/ folder
- **Codebase**: Examines TypeScript/JavaScript files to create Q&As about APIs and system architecture

How it works:
1. Click “Generate from Repository” or “Generate from Docs” button
2. The system creates a background job to process files
3. For each file, it uses the local AI model (Ollama) to generate 3-5 relevant Q&A pairs
4. Generated Q&As are automatically categorized and stored in the database
5. Progress is shown in real-time with file count and generated Q&A count

Categories:
- `system`: System architecture and design
- `api`: API endpoints and usage
- `features`: System features and capabilities
- `configuration`: Setup and configuration
- `troubleshooting`: Common issues and solutions
- `general`: General information

2. Q&A Document Upload

Upload Q&A documents in multiple formats to quickly populate the training database.

Supported Formats:

Q:/A: Format

```
Q: What is the Sports Bar TV Controller?
A: It's an AI-powered management system for sports bars...

Q: How do I configure a TV?
A: To configure a TV: 1) Go to Matrix Configuration...
```

Question:/Answer: Format

```
Question: What is the Sports Bar TV Controller?
Answer: It's an AI-powered management system for sports bars...

Question: How do I configure a TV?
Answer: To configure a TV: 1) Go to Matrix Configuration...
```

JSON Format

```
[
  {
    "question": "What is the Sports Bar TV Controller?",
    "answer": "It's an AI-powered management system...",
    "category": "system",
    "tags": ["overview", "introduction"]
  }
]
```

Markdown Format

```
## Question
What is the Sports Bar TV Controller?

## Answer
It's an AI-powered management system for sports bars...

## Question
How do I configure a TV?

## Answer
To configure a TV: 1) Go to Matrix Configuration...
```

How to upload:

1. Prepare your Q&A document in one of the supported formats
2. Click “Upload Q&A File” button
3. Select your file (.txt, .json, or .md)
4. The system will parse and validate the content
5. Successfully parsed Q&As are saved to the database
6. You’ll see a summary of how many Q&As were uploaded

3. Q&A Management

View, filter, edit, and delete Q&A entries through the management interface.

Features:

- **Filtering**: Filter by category and source type

- **Search**: Search across questions, answers, and tags
- **Edit**: Modify questions, answers, and categories
- **Delete**: Remove outdated or incorrect Q&As
- **Statistics**: View usage statistics and popular Q&As

Statistics Dashboard:
- Total Q&A count
- Active Q&A count
- Breakdown by category
- Breakdown by source type
- Most frequently used Q&As

Integration with AI Assistant

The Q&A training system is integrated with the existing AI knowledge base:

1. **Enhanced Context**: When you ask the AI assistant a question, it searches both the documentation and Q&A entries
2. **Relevance Ranking**: Q&As are ranked by relevance to your query
3. **Usage Tracking**: The system tracks which Q&As are used most often
4. **Priority**: Q&A entries are shown before general documentation (they’re usually more direct)

API Endpoints

Generate Q&As

```
POST /api/ai/qa-generate
Body: {
  "sourceType": "repository" | "documentation" | "codebase",
  "model": "llama3.2:3b" (optional)
}
Response: { "jobId": "...", "status": "started" }
```

Check Generation Status

```
GET /api/ai/qa-generate?jobId=xxx
Response: {
  "id": "...",
  "status": "running" | "completed" | "failed",
  "totalFiles": 10,
  "processedFiles": 5,
  "generatedQAs": 25
}
```

Upload Q&A File

```
POST /api/ai/qa-upload
Body: FormData with 'file' field
Response: {
  "success": true,
  "saved": 15,
  "total": 15,
  "errors": []
}
```

List Q&A Entries

```
GET /api/ai/qa-entries
Query params:
  - category: Filter by category
  - sourceType: Filter by source type
  - query: Search query
  - stats: Set to 'true' for statistics
Response: Array of Q&A entries or statistics object
```

Create Q&A Entry

```
POST /api/ai/qa-entries
Body: {
  "question": "...",
  "answer": "...",
  "category": "general",
  "tags": ["tag1", "tag2"]
}
Response: Created Q&A entry
```

Update Q&A Entry

```
PUT /api/ai/qa-entries
Body: {
  "id": "...",
  "question": "...",
  "answer": "...",
  "category": "...",
  "isActive": true
}
Response: Updated Q&A entry
```

Delete Q&A Entry

```
DELETE /api/ai/qa-entries?id=xxx
Response: { "success": true }
```

Database Schema

QAEntry Table

- `id`: Unique identifier

- `question`: The question text
- `answer`: The answer text
- `category`: Category (system, api, features, configuration, troubleshooting, general)
- `tags`: JSON array of tags
- `sourceType`: Source (manual, auto-generated, uploaded)
- `sourceFile`: Original file path (for auto-generated/uploaded)
- `confidence`: Confidence score (0.0-1.0) for auto-generated Q&As
- `isActive`: Whether the Q&A is active
- `usageCount`: Number of times used
- `lastUsed`: Last usage timestamp
- `createdAt`: Creation timestamp
- `updatedAt`: Last update timestamp

QAGenerationJob Table

- `id`: Unique identifier
- `status`: Job status (pending, running, completed, failed)
- `sourceType`: Source type for generation
- `sourcePath`: Path to source files
- `totalFiles`: Total files to process
- `processedFiles`: Files processed so far
- `generatedQAs`: Number of Q&As generated
- `errorMessage`: Error message if failed
- `startedAt`: Job start time
- `completedAt`: Job completion time
- `createdAt`: Creation timestamp
- `updatedAt`: Last update timestamp

Best Practices

1. **Start with Documentation**: Generate Q&As from documentation first, as it's usually well-structured
2. **Review Auto-generated Q&As**: Always review and edit auto-generated Q&As for accuracy
3. **Use Categories**: Properly categorize Q&As to make them easier to find and manage
4. **Add Tags**: Use tags to add additional context and improve searchability
5. **Regular Updates**: Regenerate Q&As when documentation or code changes significantly
6. **Monitor Usage**: Check usage statistics to see which Q&As are most helpful
7. **Deactivate Outdated**: Instead of deleting, deactivate outdated Q&As to preserve history

Troubleshooting

Generation Not Working

- Ensure Ollama is running (check OLLAMA_BASE_URL environment variable)
- Verify the model is available (default: llama3.2:3b)
- Check system logs for specific errors
- Ensure sufficient disk space for processing

Upload Parsing Errors

- Verify file format matches one of the supported formats
- Check for special characters or encoding issues
- Ensure questions and answers are properly paired
- Try a smaller file first to test the format

Q&As Not Appearing in AI Responses

- Verify Q&As are marked as active (isActive = true)
- Check that the category is appropriate
- Ensure the question/answer content is relevant to queries
- The AI uses relevance scoring - very generic Q&As may not rank highly

Sample Data

Sample Q&A files are provided in the repository:
- `sample-qa-data.txt` : Q:/A: format example
- `sample-qa-data.json` : JSON format example

Use these as templates for creating your own Q&A documents.

Future Enhancements

Potential future improvements:
- Bulk import from CSV files
- Export Q&As to various formats
- Q&A versioning and history
- Collaborative editing with approval workflow
- Integration with external knowledge bases
- Automatic Q&A validation and quality scoring
- Multi-language support