Fix Summary: Chat Bot and Q&A Training System JSON Parsing Issues

Date

October 8, 2025

Issues Identified

1. Chat Bot JSON Parsing Error

Error Message: `Unexpected token 'd', "data: {"ty"... is not valid JSON`

Root Cause:
- The backend API routes (`/api/chat` and `/api/ai/enhanced-chat`) default to streaming mode (`stream: true`)
- When streaming is enabled, the backend returns Server-Sent Events (SSE) format with `Content-Type: text/event-stream`
- SSE format starts with `data:` prefix followed by JSON, which is NOT valid JSON when parsed directly
- Frontend components were calling these APIs without explicitly setting `stream: false`
- Frontend was attempting to parse SSE responses as JSON using `response.json()`, causing the parsing error

Technical Details:
- SSE format: `data: {"type":"content","content":"Hello"}\n\n`
- Frontend expected: `{"response":"Hello"}`
- The mismatch caused: `Unexpected token 'd', "data: {"ty"...` error

2. Codebase Index Showing 0 Files

Symptoms:
- Total Files: 0
- Total Size: 0 Bytes
- Last Indexed: Never

Root Cause:
- The codebase indexing functionality is working correctly
- The issue was that the sync had never been triggered
- The GET endpoint for stats works properly
- Users need to click "Sync Codebase" button to trigger the initial indexing

Files Modified

1. `/src/app/ai-hub/page.tsx`

Changes:
- Added `stream: false` parameter to the fetch call in `handleSendMessage()`
- Added proper error handling for HTTP errors

- Enhanced error display to show detailed error messages, suggestions, and context
- Added console.error for debugging

Before:

```
body: JSON.stringify({
  message: userMessage,
  useKnowledge: true,
  useCodebase: true
})
```

After:

```
body: JSON.stringify({
  message: userMessage,
  useKnowledge: true,
  useCodebase: true,
  stream: false  // CRITICAL FIX: Explicitly request non-streaming response
})
```

2. `/src/components/TroubleshootingChat.tsx`

Changes:
- Added `stream: false` parameter to the fetch call in `handleSendMessage()`
- Improved error message display
- Added console.error for debugging

Before:

```
body: JSON.stringify({
  message: inputText,
  sessionId: sessionId,
})
```

After:

```
body: JSON.stringify({
  message: inputText,
  sessionId: sessionId,
  stream: false,  // CRITICAL FIX: Explicitly request non-streaming response
})
```

3. `/src/components/AITeachingInterface.tsx`

Changes:
- Added `stream: false` parameter to the fetch call in `handleTestQuestion()`
- Added proper HTTP error handling
- Enhanced error display with detailed messages and suggestions
- Added console.error for debugging

Before:

```
body: JSON.stringify({
  message: testQuestion,
  useKnowledge: true,
  useCodebase: false
})
```

After:

```
body: JSON.stringify({
  message: testQuestion,
  useKnowledge: true,
  useCodebase: false,
  stream: false  // CRITICAL FIX: Explicitly request non-streaming response
})
```

Backend API Routes (No Changes Required)

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

- Already supports both streaming and non-streaming modes
- Defaults to `stream: true` if not specified
- When `stream: false` is passed, returns JSON response
- When `stream: true`, returns SSE format

`/src/app/api/ai/enhanced-chat/route.ts`

- Already supports both streaming and non-streaming modes
- Defaults to `stream: true` if not specified
- Properly handles both response formats

`/src/app/api/ai-assistant/index-codebase/route.ts`

- Working correctly
- GET endpoint returns current stats
- POST endpoint performs indexing
- No changes required

Testing Performed

1. Chat Bot Testing

- ✅ Verified `stream: false` parameter is sent in request
- ✅ Confirmed backend returns JSON format
- ✅ Tested error handling for various scenarios
- ✅ Verified proper error message display

2. Q&A Training Testing

- ✅ Verified test question functionality
- ✅ Confirmed proper error handling
- ✅ Tested with knowledge base enabled

3. Codebase Sync Testing

- ✅ Verified sync button triggers indexing
- ✅ Confirmed stats are displayed correctly after sync
- ✅ Tested file counting and size calculation

Expected Behavior After Fix

Chat Bot

1. User types message and clicks “Send”
2. Frontend sends request with `stream: false`
3. Backend processes request and returns JSON response
4. Frontend parses JSON successfully
5. AI response is displayed in chat

Q&A Training

1. User enters test question
2. Frontend sends request with `stream: false`
3. Backend uses knowledge base to generate response
4. Response is displayed in test area

Codebase Sync

1. User clicks “Sync Codebase” button
2. Backend scans project directory
3. Files are indexed in database
4. Stats are updated and displayed
5. Chat bot can now access codebase for questions

User Instructions

To Fix Chat Bot Issues:

1. Pull the latest changes from the repository
2. Restart the application: `pm2 restart sports-bar-controller`
3. Clear browser cache (Ctrl+Shift+Delete)
4. Refresh the page (Ctrl+F5)
5. Test the chat bot by sending a message

To Fix Codebase Index:

1. Navigate to AI Hub → AI Assistant tab
2. Click the “Sync Codebase” button
3. Wait for indexing to complete (shows progress)
4. Verify stats show correct file count and size
5. Test by asking questions about the codebase

Prevention Measures

For Future Development:

1. **Always specify stream parameter**: When calling chat APIs, explicitly set `stream: true` or `stream: false`
2. **Use TypeScript interfaces**: Define proper request/response types to catch missing parameters
3. **Add request validation**: Backend should validate the stream parameter
4. **Improve error messages**: Frontend should display user-friendly error messages
5. **Add loading states**: Show proper loading indicators during API calls
6. **Test both modes**: Always test both streaming and non-streaming modes

Technical Notes

Server-Sent Events (SSE) Format

```
data: {"type":"status","message":"Generating..."}\n\n
data: {"type":"content","content":"Hello"}\n\n
data: {"type":"done"}\n\n
```

JSON Format

```
{
  "response": "Hello, how can I help you?",
  "sessionId": "abc123"
}
```

Why SSE Was Causing Issues

1. SSE is designed for real-time streaming of data
2. Each SSE message starts with `data:` prefix
3. Frontend's `response.json()` expects pure JSON
4. Parsing `data: {"type":"content"}` as JSON fails because `data:` is not valid JSON
5. The error message shows the first few characters: `Unexpected token 'd', "data: {"ty"...`

Verification Checklist

- [x] Chat bot sends messages without JSON parsing errors
- [x] Q&A training test questions work correctly
- [x] Codebase sync button triggers indexing
- [x] Stats display correctly after sync
- [x] Error messages are user-friendly
- [x] All frontend components use `stream: false`
- [x] Backend APIs support both streaming and non-streaming
- [x] No breaking changes to existing functionality

Related Files

Frontend Components

- `/src/app/ai-hub/page.tsx` - Main AI Hub page with chat interface
- `/src/components/TroubleshootingChat.tsx` - Troubleshooting chat component
- `/src/components/AITeachingInterface.tsx` - Q&A training interface

Backend API Routes

- `/src/app/api/chat/route.ts` - Main chat API with SSE support
- `/src/app/api/ai/enhanced-chat/route.ts` - Enhanced chat with knowledge base
- `/src/app/api/ai-assistant/index-codebase/route.ts` - Codebase indexing

Library Files (No Changes)

- `/src/lib/ai-client.ts` - Direct Ollama client
- `/src/lib/enhanced-ai-client.ts` - Enhanced AI client with features

Conclusion

The root cause of the chat bot errors was a mismatch between the response format (SSE) and the parsing method (JSON). By explicitly setting `stream: false` in all frontend API calls, we ensure the backend returns JSON format that can be properly parsed by the frontend.

The codebase indexing functionality was working correctly but had never been triggered. Users simply need to click the “Sync Codebase” button to initialize the index.

All fixes are backward compatible and do not break existing functionality. The streaming mode is still available for future use when needed.