# Generative AI .NET Project – Knowledge Base

## Phase 1 – Foundation

- Explored use cases in .NET apps (chatbots, Q&A bots, summarization, content generation, code assist).  
- Tested HuggingFace models for API demo + Azure deployment.  
- Learned Prompt Engineering: zero-shot, few-shot, role prompting.  
- Findings: Few-shot & role prompting gave better results; zero-shot was verbose/unreliable.

## Phase 2.1 – .NET Backend & Frontend Integration

✅ Backend:  
- .NET Core Web API with ChatController endpoints (send, stream, history, sessions, duplicate-session, delete).  
- Services layer with IOpenAiService + IChatHistoryService.  
- Persistence in SQL Server with ChatSessions + ChatMessages.  
  
✅ Frontend:  
- Modularized JS files: chat.js, sessions.js, templates.js, utils.js, main.js.  
- Features: streaming, typing dots, session sidebar, model selector, prompt preview.  
  
✅ Deployment:  
- Azure App Service + SQL Azure.  
- Fixed CORS + connection strings in App Config.

## Phase 2.2 – Prompt Templates + Clean UI

- Templates stored in DB with parameters (tone, length).  
- buildPrompt() inserts parameters into templates.  
- buildPromptPreview() updates preview instantly.  
- Admin panel for template CRUD (planned in Phase 2.3).

## Phase 2.2 – Bug Fixes & Enhancements

- Fixed sessionId undefined bug.  
- Fixed circular imports between chat.js and sessions.js.  
- Streaming shows typing dots before assistant reply.  
- Cloning sessions now copies both user + assistant messages.  
- User messages persisted before OpenAI call.

## Project Structure

/2\_OpenAIChatDemo  
 ├── Controllers (ChatController.cs)  
 ├── Data (ChatDbContext.cs)  
 ├── DTOs (ChatRequestDto, ChatResponseDto, ChatSessionDto, ChatMessageDto)  
 ├── Models (ChatSession, ChatMessage)  
 ├── Services (IOpenAiService, OpenAiService, IChatHistoryService, ChatHistoryService)  
 ├── wwwroot/js (chat.js, sessions.js, templates.js, utils.js, main.js)  
 └── Views/Home (Index.cshtml)

## Next Steps – Phase 2.3

- Admin Panel for Prompt Templates (CRUD UI + API).  
- Template Versioning.  
- Advanced parameterized prompts.  
- Session cloning enhancements (cross-model comparisons).