

# AI-Driven Development - 30-Day Challenge

## Task 4 : Connecting Context7 MCP

### Context7 Integration

#### 1. Access Context7

1. Open the official website:  
<https://context7.com/>
2. Sign in to your Context7 account.

#### 2. Add Required Libraries

3. Navigate to the **Libraries** section and include the necessary dependencies for your project.
4. \_\_\_\_\_

#### 3. OpenAI API Reference

5. Review the API documentation here:  
<https://platform.openai.com/docs/apireference/introduction?lang=python>

#### 4. Generate an API Key

1. Open the **OpenAI Dashboard**.
  2. Create a new **API Key** for use in your project.
- 

#### 5. Create Your Project

1. Inside Context7, create a new project—e.g., **Quiz Generator Agent**.
2. In your project workspace, create a folder named **.gemini**.

3. Inside this folder, create a file named **settings.json**.
- 

## 6. Add MCP Configuration

Refer to the setup instructions at:

<https://context7.com/docs/installation#cursor>

Add the following configuration to your **settings.json** file:

```
{  
  "mcpServers": {  
    "context7": {  
      "url": "https://mcp.context7.com/mcp",  
      "headers": {  
        "CONTEXT7_API_KEY": "YOUR_API_KEY"  
      }  
    }  
  }  
}
```

Replace **YOUR\_API\_KEY** with the key you generated earlier.

## 7. Run Gemini and Enable MCP

1. Start Gemini.

2. Execute the command:  
**/mcp**
3. The MCP server will now be added to your project.



```
C:\Windows\system32\cmd.exe

> GEMINI

Tips for getting started:
1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. /help for more information.

> /mcp

Configured MCP servers:

context7 - Ready (2 tools)
Tools:
- get-library-docs
- resolve-library-id

Using: 2 GEMINI.md files | 1 MCP server

> █ Type your message or @path/to/file

~\...\quiz app          no sandbox (see /docs)          auto
```

## PROMPT:

```
C:\Windows\system32\cmd.exe

> /mcp

Configured MCP servers:
[+] context7 - Ready (2 tools)
Tools:
- get-library-docs
- resolve-library-id

> Build my full "Personal Chatbot with Memory" exactly using Streamlit + OpenAI-Agents SDK + Gemini with the file
structure, rules, and details from gemini.md. Generate all required files (tools.py, agent.py, app.py,
pyproject.toml, etc.) with zero extra code.

[+] Request cancelled.

Using: 2 GEMINI.md files | 1 MCP server

> Read the complete project instructions from gemini.md and strictly follow them without adding anything extra.
Build the Streamlit Summarizer & Quiz Generator Agent exactly as specified, using only the required SDK syntax
and tools.

~\...\quiz app          no sandbox (see /docs)          auto
```

## Practical Task :Build the Study Notes

### Summarizer & Quiz Generator Agent

#### GitHub Repository

<https://github.com/irisss3099/Quarter-4---30-days-challenge/tree/main/assignment-4-project/quiz%20app>