

# Step-by-Step Guide – Azure AI Foundry → PHP Chatbot → Render

SBM Project • Technical Tutorial

This document provides a complete walkthrough on how to create an AI Assistant in Azure AI Foundry, connect it to a PHP-based chatbot, and finally deploy it online using Render.

## Important Warning – Assistant vs. Agent

There is a key difference between an Agent and an Assistant in Azure AI Foundry.

An Agent is a higher-level concept used mainly inside Azure AI Foundry for testing and internal workflows. It cannot be directly integrated or deployed to an external website.

An Assistant, on the other hand, provides the necessary API access (Assistant ID, Deployment, Endpoint, API Key) that allows you to connect it to your own application (for example, a PHP chatbot) and host it online (via Render or any other platform).

Content: 1) Azure (creation & setup), 2) GitHub (PHP chatbot code), 3) Render (deployment to the cloud).

# 1. Azure AI Foundry – Creating & Configuring the Assistant

Azure AI Foundry is the platform where we create and manage our AI Assistant. This is the 'brain' of the chatbot: it hosts the AI model, the knowledge files, and provides the secure API endpoint. Think of it as the server that does all the heavy lifting, while our PHP code is just the interface.

The screenshot shows the Azure AI Foundry dashboard. At the top, there's a navigation bar with icons for Home, Docs, Notifications (4), Settings, Help, and a Language Model selector (AB). Below the header, a message says "Keep building with Azure AI Foundry". On the right, there are buttons for "View all resources", "+ Create new", and "Help". A search bar is also present.

The main area displays a table of resources:

Resource name	Type	Region	Created on
sbmchatbot	Azure OpenAI	eastus	Aug 7, 2025 1:09 PM
chatbott-rag	Azure OpenAI	eastus	
nfarhat-0171	Project (default) nfarhat-0171-resource (AI Foundry)	eastus2	

Below the table, a section titled "Explore models and capabilities" features a large preview window. The preview shows the Azure AI Foundry interface with the title "Create smarter agents with Azure AI Foundry". It includes a "Create an agent" button, a "Model deployment" dropdown set to "GPT-4o mini", a "System message" input field containing "You are a helpful agent...", and a "Knowledge" panel. To the right, there's a "Research-agent" section with a weather forecast for New York, Los Angeles, and Chicago, and a "Describe what you'd like to do or use / to reference files, people, and more..." input field.

Azure AI Foundry – List of resources.

Here you see the resources available. Our main resource is named sbmchatbot. This resource gives us access to Azure OpenAI services in the East US region.

The screenshot shows the Azure OpenAI Resource configuration page. On the left, there's a sidebar with navigation links like Home, Get started, Model catalog, Playgrounds, Chat, Assistants (selected), Video, Audio, Images, Tools, Fine-tuning, Azure OpenAI Evaluation, Stored completions, Batch jobs, Monitoring, Shared resources, Deployments, Quota, and Guardrails. The main content area has a header "Welcome to Azure OpenAI" and a sub-header "Explore the generative AI models and craft unique prompts for your use cases." It features a "Resource configuration" section with fields for Name (sbmchatbot), Subscription (Azure subscription 1), Subscription ID (b843ad5e-c7f3-434f-91ee-adc43fcfb6d2), View access control (IAM), API key (redacted), Resource group (abrahimiSBM), Pricing tier (Standard S0), Azure OpenAI endpoint (https://sbmchatbot.openai.azure.com/), and Location (eastus). Below this is a "Get started" section with "Assistants playground" and "Chat playground" options.

Azure OpenAI – Resource configuration page.

Important fields to note: the API key (acts like a password for apps), the endpoint (URL to send requests), and the subscription details. We will later use the API key and endpoint in our PHP code and Render deployment.

The screenshot shows the Assistant playground setup screen. The sidebar is identical to the previous one. The main area is titled "Assistants playground" and includes a toolbar with New assistant, Select assistant, View code, and Delete buttons. It features a "Setup" section with fields for Assistant id (asst\_WrGbhVrlqfqOqg5g1omyddBB), Assistant name (sbmSupportBot), Deployment (sbm-deployment (version:0125)), and Instructions (You are an IT support assistant for SBM. You must strictly answer only based on the knowledge files provided via File Search. Do not answer from general knowledge or...). There are also sections for Assistant Description and Tools. On the right, there are buttons for Clear chat, Logs, Thread files, JSON response (0 tokens), and a text input field for Type user query here.

Assistant playground – Setup of our Assistant.

This screen shows the Assistant ID (a unique identifier for our bot), the Assistant name, and the Deployment. The Deployment links to a specific AI model version, so you can control exactly which model powers your chatbot.

The screenshot shows the 'Assistants playground' section of the Azure AI Foundry interface. On the left, a sidebar lists various tools and resources under categories like Home, Get started, Model catalog, Playgrounds, Chat, Assistants (selected), Tools, Fine-tuning, and Shared resources. The main area is titled 'Assistants playground' and contains sections for 'Assistant Description', 'Tools', and 'Model settings'. Under 'Tools', 'File search' is enabled, connected to 'AssistantVectorStore\_30891'. There's also a 'Code interpreter' section with a 'File search' toggle, a 'Functions' section with an '+ Add function' button, and a 'Temperature' slider. A large input field at the bottom is labeled 'Type user query here. (Shift + Enter for new line)'. At the top right, there are buttons for 'Clear chat', 'Logs', 'Thread files', and 'JSON response'.

Assistant playground – Tools configuration.

Here we enabled File Search, which connects our Assistant to specific documents (like SBM manuals). This means the chatbot will only answer using company-approved documents, avoiding generic responses.

The screenshot shows the 'Chat playground' section of the Azure AI Foundry interface. The sidebar is identical to the one in the previous screenshot. The main area is titled 'Chat playground' and features a 'Setup' section on the left with a 'Deployment' dropdown set to 'sbm-deployment (version:0125)' and a text input for model instructions. Below this are sections for 'Add your data' and 'Parameters'. To the right is a 'Chat history' section with a message icon and a placeholder 'Start with a sample prompt'. Below it are three cards: 'Historical fiction' (Write a scene set in ancient Rome, focusing on the daily life of a common citizen.), 'Recipe creation' (Invent a recipe for a dish that combines flavors from two different cuisines.), and 'Marketing slogan' (Create a catchy marketing slogan for a new eco-friendly product.). At the bottom, there's a large input field for user queries and a note '11/16384 tokens to be sent'.

Chat playground – Quick model test.

This is a sandbox to try out the AI model before final integration. It helps you confirm the Deployment is working correctly.

Azure AI Foundry | Azure OpenAI / sbmchatbot / Data files

Playgrounds Chat Assistants PREVIEW Video PREVIEW Audio PREVIEW Images Tools Fine-tuning Azure OpenAI PREVIEW Evaluation Stored completions PREVIEW Batch jobs Monitoring Shared resources Deployments Quota Guardrails + Controls Risks + alerts PREVIEW Data files Assistant vector stores PREVIEW

Connect, create, or manage your data

Find all your uploaded datasets (for use in Assistants, batch, fine-tuning, and evaluation) here, as well as results and error files output by batch and fine-tuning. Learn about preparing a dataset for batch or fine-tuning

+ Add data Refresh Delete Download

Name	File ID	Status	Purpose	Size	Created
SBM Website Content - Home Page 1.pdf	assistant-314mY3e4c2oNJLtyg7zEA	Processed	assistants	329.44 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SOLUT...	assistant-PRBiuv9H9BCHDxoisGmM3	Processed	assistants	228.84 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SOLUT...	assistant-Lkkn3q9oEdxFZWPNy1BT8Y	Processed	assistants	213.61 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SOLUT...	assistant-ENrQMHfmcc7H4wdxJxKtGw	Processed	assistants	204.15 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SOLUT...	assistant-SrRrr8StxSwWtCy1K2VyrX	Processed	assistants	226.05 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SOLUT...	assistant-9LKsy6mSWF3EvrrRRXCvyG	Processed	assistants	230.63 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SERVIC...	assistant-1Ppr31JeamX7MxcCbiaQdx	Processed	assistants	221.74 KB	Au
SBM WEBSITE CONTENT – SUBPAGES – SERVIC...	assistant-2TdQE2qV5iQxCUcxnz5uK	Processed	assistants	236.11 KB	Au

< Prev Next > 25/Page

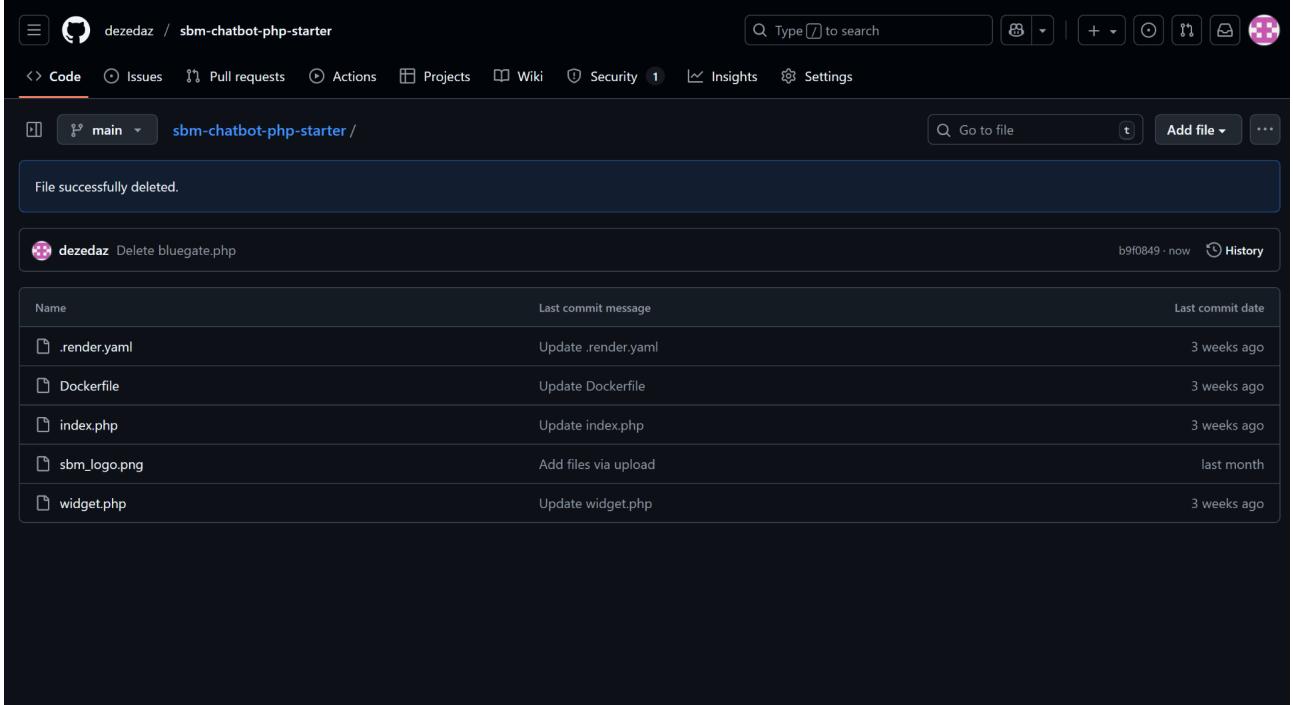
<https://ai.azure.com/resource/datafile?wsid=/subscriptions/b843ad5e-c7f3-434f-91ee-adc43cfb6d2/resourceGroups/abrahimiSBM/providers/Microsoft.CognitiveServices/accounts/sbmchatbot&tid=1b16c903-5d47-48a7-90a4-ff2df8a18982>

Data files – Uploaded SBM documents.

Each document has been processed and indexed. Notice the status 'Processed'. This indexing is what makes File Search possible: the AI can now look up answers in these files.

## 2. GitHub – Hosting the PHP Chatbot Code

GitHub is where we keep the chatbot's source code. Think of it as a shared folder with version control: you can update, track changes, and connect it directly to Render for automatic deployment.



The screenshot shows a GitHub repository page for 'dezedaz / sbm-chatbot-php-starter'. The 'Code' tab is selected. In the main area, there is a message: 'File successfully deleted.' Below this, a commit history is shown:

Name	Last commit message	Last commit date
.render.yaml	Update .render.yaml	3 weeks ago
Dockerfile	Update Dockerfile	3 weeks ago
index.php	Update index.php	3 weeks ago
sbm_logo.png	Add files via upload	last month
widget.php	Update widget.php	3 weeks ago

GitHub repository – Project structure.

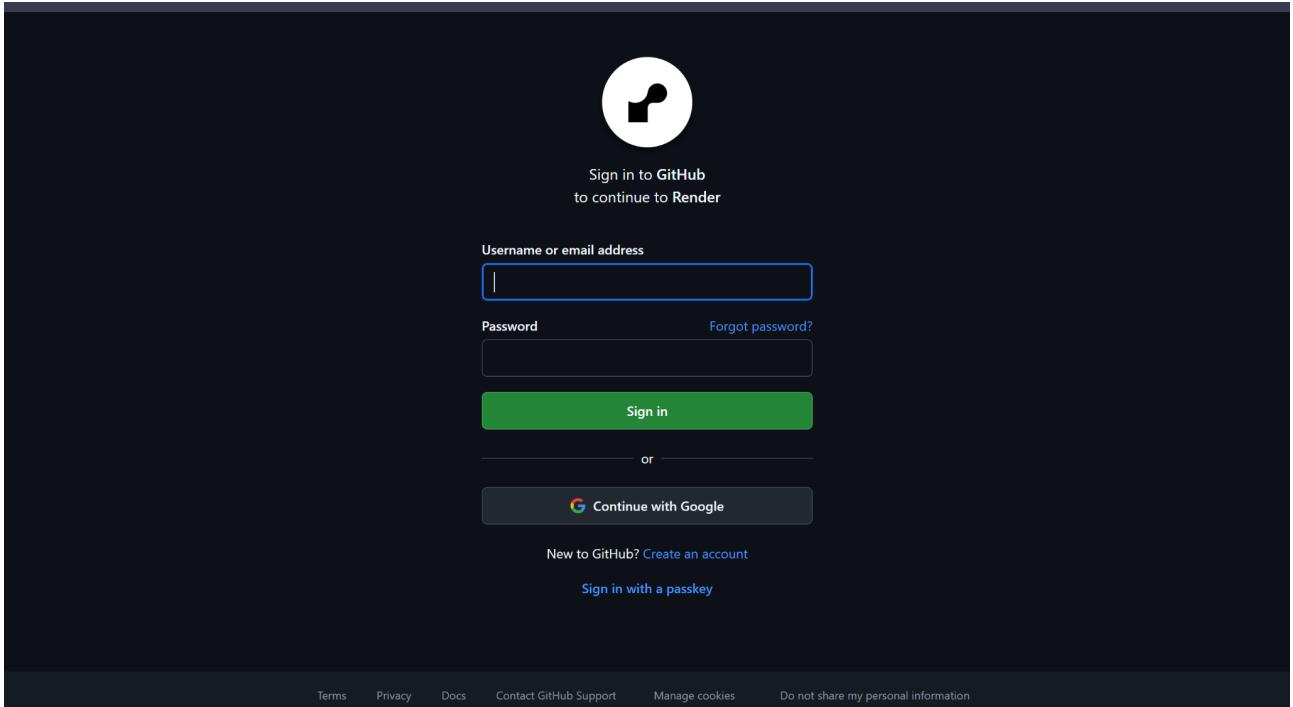
Files included: index.php (main chatbot UI), widget.php (extra widget option), sbm\_logo.png (branding), Dockerfile (instructions to build an image with PHP), and .render.yaml (Render configuration).

Each of these files plays a role:

- index.php: Displays the chatbot interface, handles sessions (chat history), reset, dark mode, etc.
- widget.php: Optional component to embed the chatbot in another site.
- sbm\_logo.png: The company logo.
- Dockerfile: Ensures Render knows how to build a container with PHP +cURL (needed for API calls).
- .render.yaml: Defines how Render should deploy this app (service type, port, environment variables).

### 3. Render – Deploying the Chatbot Online

Render is the hosting service. It builds the app from GitHub, runs it inside a Docker container, and exposes it to the web with a public URL. This is the step that makes your chatbot accessible to anyone online.



Render login – Sign in with GitHub.

Render can connect directly to your GitHub account. This way, each time you push changes to GitHub, you can redeploy easily.

Render dashboard – Active service.

Here you see the deployed service 'sbm-chatbot-php-starter'. Status is green (Deployed). The region and runtime (Docker) are also displayed.

The screenshot shows the Render interface with the service 'sbm-chatbot-php-starter' selected. The left sidebar has sections for MONITOR (Logs, Metrics), MANAGE (Environment, Shell, Scaling, Previews, Disks), and a Render MCP server section. The Environment section is currently active. The main area displays 'Environment Variables' with three entries: ASSISTANT\_ID, AZURE\_API\_KEY, and AZURE\_ENDPOINT, each with a redacted value. Below this is a 'Secret Files' section with instructions for storing plaintext files containing secret data like .env files or private keys, accessible during builds and runtime from /etc/secrets/<filename>.

Render – Environment variables.

Notice the keys: ASSISTANT\_ID, AZURE\_API\_KEY, and AZURE\_ENDPOINT. These values replace sensitive information in code. By using environment variables, we keep the code clean and secure.

Once configured, Render builds the Docker image, launches the service, and provides you with a URL. Testing is simple: open the URL and start chatting. The assistant should answer based only on the indexed SBM documents.

### Troubleshooting & Common Issues

- Authentication errors (401/403): Check your API key and endpoint.
- No response or timeout: Check logs in Render, confirm the Assistant ID.
- Irrelevant answers: Ensure File Search is enabled and documents are linked.
- Build failures: Check the Dockerfile for PHP + cURL installation and versions.

The next step: You will receive the PHP files with detailed inline comments. Those comments explain the code line by line, making it easier for you to understand and extend the project.