🚀Deploying Your Flask-Based AI Chatbot to Azure with Docker and Custom Domain

This documentation covers the **complete end-to-end process** of deploying your chatbot project (`CCD-AI`) using Docker on Azure App Service and mapping it to a **custom domain** (e.g., `bot.ccd.bhopal.dev`).

Project Structure

Make sure your project is organized like this:

```
CCD-AI/
├── run.py
├── build_db.py
├── requirements.txt
├── Dockerfile
├── .dockerignore
├── chroma_db/ (created later)
├── templates/
├── static/
├── .env
└── ...
```

Step 1: Prepare Docker Environment

🚇1.1 Install Docker Desktop

- Download Docker
- Install and run Docker Desktop (check for the whale 🐋icon in system tray)

1.2 Create Dockerfile

In the root directory, create `Dockerfile`:

```
FROM python:3.11-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .

ARG GOOGLE_API_KEY_1
ENV GOOGLE_API_KEY_1=$GOOGLE_API_KEY_1
RUN python build_db.py
```

```
EXPOSE 8080
CMD ["gunicorn", "--bind", "0.0.0.0:8080", "--timeout", "120", "run:app"]
```

🗼1.3 Create .dockerignore

Create `.dockerignore` to exclude unnecessary files:

```
venv
__pycache__
.git
.gitignore
.env
chroma_db
```

Step 2: Azure Setup

2.1 Install Azure CLI

[Install Azure CLI]

🚚2.2 Login

```
az login
```

2.3 Create Resource Group & ACR

```
az group create --name ccd-chatbot-rg --location "Central India"
az acr create --resource-group ccd-chatbot-rg --name ccdchatbotacr --sku
Basic --admin-enabled true
```

Step 3: Build and Push Docker Image

3.1 Log into Azure Container Registry

```
az acr login --name ccdchatbotacr
```

🏗️3.2 Build the Docker Image

```
docker build `
  --build-arg GOOGLE_API_KEY_1=$((Get-Content .env | Select-String
"GOOGLE_API_KEY_1").Line.Split('=')[1].Trim('"')) `
  -t ccdchatbotacr.azurecr.io/chatbot:latest .
```

🚀3.3 Push Image to ACR

```
docker push ccdchatbotacr.azurecr.io/chatbot:latest
```

Step 4: Deploy to Azure App Service

🏗️4.1 Create App Service Plan

```
az appservice plan create --name ccd-chatbot-plan --resource-group ccd-chatbot-rg --is-linux --sku B1
```

4.2 Deploy the App

```
az webapp create \
  --resource-group ccd-chatbot-rg \
  --plan ccd-chatbot-plan \
  --name ccd-ai-chatbot \
  --deployment-container-image-name ccdchatbotacr.azurecr.io/chatbot:latest \
  --docker-registry-server-password $(az acr credential show -n ccdchatbotacr --query "passwords[0].value" -o tsv) \
  --docker-registry-server-user ccdchatbotacr
```

🚚Step 5: Set Environment Variables

Recommended Way (in PowerShell)

```
$envSettings = Get-Content .env | ConvertFrom-StringData
$envSettings.GetEnumerator() | ForEach-Object {
    $key = $_.Key
    $value = $_.Value
    az webapp config appsettings set `
        --resource-group ccd-chatbot-rg `
        --name ccd-ai-chatbot `
        --settings "$key=$value" | Out-Null
    Write-Host "🦜 Set $key"
}
```

Confirm:

```
az webapp config appsettings list `
  --resource-group ccd-chatbot-rg `
  --name ccd-ai-chatbot `
  --query "[?starts_with(name, 'GOOGLE_API_KEY') || name=='SECRET_KEY'].
{Key:name, Value:value}" -o table
```

🩼Step 6: Map Custom Domain (bot.ccd.bhopal.dev)

🚃Flow Diagram:

```
graph TD
    A[You] -->|Provide TXT + CNAME Info| B[Boss (Domain Owner)]
    B --> C[Adds DNS Records to bhopal.dev]
    C --> D[Azure Verifies TXT]
    D --> E[Traffic routed to your app]
```

Tell Your Boss

Provide him with the following:

TXT Record

Type	Name	Value
TXT	`asuid.bot.ccd.bhopal.dev`	`52eb67d41cc00bc690f3c55c5f4d93075bf68588791e11a126ab1baff02`

CNAME Record

Type	Name	Value
CNAME	`bot.ccd.bhopal.dev`	`ccd-ai-chatbot.azurewebsites.net`

🚛Azure Verification (after records are set)

```
az webapp config hostname add \
  --resource-group ccd-chatbot-rg \
  --webapp-name ccd-ai-chatbot \
  --hostname bot.ccd.bhopal.dev
```

🏩(Optional) Enable Logging

```
az webapp log config \
  --name ccd-ai-chatbot \
  --resource-group ccd-chatbot-rg \
  --web-server-logging filesystem \
  --detailed-error-messages true \
  --failed-request-tracing true

az webapp log tail \
  --resource-group ccd-chatbot-rg \
  --name ccd-ai-chatbot
```

To exit logs:

```
Ctrl + C
```

Final Result

Your chatbot should now be live at:

```
https://bot.ccd.bhopal.dev
```

SSL will be handled automatically by Azure App Service.

Notes

- If using `LangChain` + `Gemini`, verify Google API quotas
- Use `gunicorn` for production deployment in Docker
- Use `Chroma` vector store initialization during build step (already handled)
- If container update is needed:
- Rebuild → push to ACR → restart app

Let me know if you'd like a PDF version or a printable version!