



CDW Documentation

Azure Open AI Chatbot Deployment Checklist

Azure Open AI Chatbot Deployment Checklist

☐ Azure OpenAI Chatbot Deployment Checklist

1. Local Build and Testing

- Bot code tested locally with Bot Framework Emulator.
- Environment variables for `AZURE_OPENAI_ENDPOINT`, `DEPLOYMENT_ID`, and `API_KEY` correctly referenced.
- Docker container runs successfully on `localhost:3978`.

2. Azure Resources Setup

- Resource Group created or selected.
- Azure Container Registry (ACR) created and admin access enabled (if not using managed identity).
- Azure OpenAI Service created with deployed model (e.g., `gpt-35-turbo`).

3. Docker Image Build and Push

- Docker image built with `docker build -t <image> ..`
- Docker image tagged with ACR address.
- Docker image pushed to ACR.

4. Deployment Parameters

- Parameter values prepared, either inline or in `containerapp-params.json`.
- Verified API Key, OpenAI endpoint, and deployment ID.
- Verified ACR server, username, and password (if required).

5. Bicep Deployment

- Bicep template validated with `az bicep build`.
- Deployment executed with either inline parameters or parameter file.
- Verified Container App is running in Azure Portal.

6. Testing After Deployment

- Verified public URL responds (default port 3978).
- Connected Bot Framework Emulator to Container App URL.
- Tested conversation flow using OpenAI backend.

7. Security and Scaling

- Confirmed ACR access is secured.
- Configured auto-scaling (if needed).
- Applied secure ingress settings (HTTPS, auth if needed).

8. Observability

- Log Analytics Workspace attached.
- Application logging enabled and verified in Azure Monitor.

9. Log Analytics and Monitoring

- **Log Analytics Workspace** created and linked to Container App Environment.
- **App Logs** verified in **Log Analytics > Logs** with queries like:

```
AppContainerAppConsoleLogs_CL \\  
| where ContainerAppName_s == "<your-container-app-name>"\  
| sort by TimeGenerated desc
```

- **Metrics monitoring** verified (CPU, Memory, Request Count).

10. Alerts Setup

- **Alert Rule for CPU Usage**
 - Example: Alert if CPU exceeds 80% for 5 minutes.
- **Alert Rule for Memory Usage**
 - Example: Alert if Memory exceeds 75% for 5 minutes.
- **Alert Rule for HTTP 500 Errors**
 - Example: Query for failed requests in Log Analytics.

Example KQL for HTTP 500 Errors:

```
AppContainerAppConsoleLogs_CL  
| where Log_s contains "500"  
| summarize Count = count() by bin(TimeGenerated, 5m)
```

Example Alert Setup Steps:

1. Go to **Azure Monitor > Alerts**.
2. Click + **New Alert Rule**.
3. **Select Resource**: Your Container App or Log Analytics Workspace.
4. **Condition**: Add a metric or log query condition.
5. **Action Group**: Create or select action groups (email, Teams, webhook, etc.).

6. Alert Details: Name and severity.

• Notification Channels Configured

- Email
- Teams
- Webhook (optional)

Bicep Definitions To Create Alerts

```
param containerAppResourceId string
param logAnalyticsWorkspaceId string
param actionGroupId string

@description('Create CPU High Usage Alert')
resource cpuAlert 'Microsoft.Insights/metricAlerts@2018-03-01' = {
  name: 'cpu-high-usage'
  location: 'global'
  properties: {
    description: 'Alert when CPU usage exceeds 80%'
    severity: 3
    enabled: true
    scopes: [containerAppResourceId]
    evaluationFrequency: 'PT1M'
    windowSize: 'PT5M'
    criteria: {
      allOf: [
        {
          metricName: 'cpuUsagePercentage'
          metricNamespace: 'Microsoft.App/containerApps'
          operator: 'GreaterThan'
          threshold: 80
          timeAggregation: 'Average'
        }
      ]
    }
    actions: [
      {
        actionGroupId: actionGroupId
      }
    ]
  }
}

@description('Create Memory High Usage Alert')
resource memoryAlert 'Microsoft.Insights/metricAlerts@2018-03-01' = {
  name: 'memory-high-usage'
  location: 'global'
  properties: {
    description: 'Alert when memory usage exceeds 75%'
    severity: 3
```

```
enabled: true
scopes: [containerAppResourceId]
evaluationFrequency: 'PT1M'
windowSize: 'PT5M'
criteria: {
  allOf: [
    {
      metricName: 'memoryWorkingSetBytes'
      metricNamespace: 'Microsoft.App/containerApps'
      operator: 'GreaterThan'
      threshold: 75
      timeAggregation: 'Average'
    }
  ]
}
actions: [
  {
    actionGroupId: actionGroupId
  }
]
}
```