

# Conversational chatbot for hotel reservations on Azure

07/05/2018 • 6 minutes to read • Contributors      all

## In this article

[Relevant use cases](#)

[Architecture](#)

[Considerations](#)

[Deploy the scenario](#)

[Pricing](#)

[Related resources](#)

This example scenario is applicable to businesses that need to integrate a conversational chatbot into applications. In this scenario, a C# chatbot is used for a hotel chain that allows customers to check availability and book accommodation through a web or mobile application.

Potential uses include providing a way for customers to view hotel availability and book rooms, review a restaurant take-out menu and place a food order, or search for and order prints of photographs. Traditionally, businesses would need to hire and train customer service agents to respond to these customer requests, and customers would have to wait until a representative is available to provide assistance.

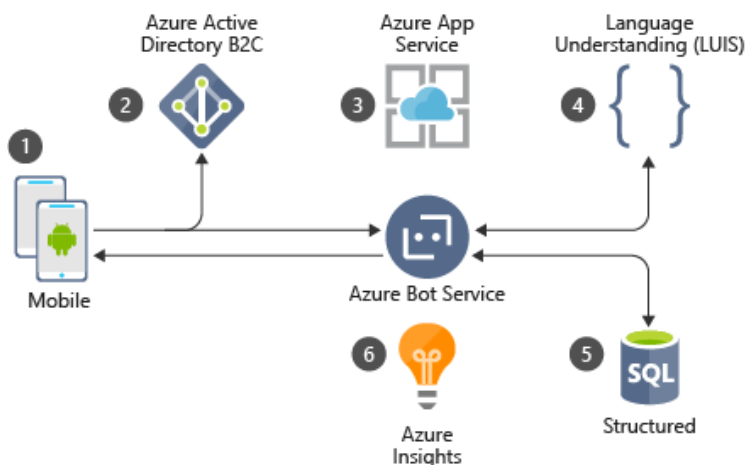
By using Azure services such as the Bot Service and Language Understanding or Speech API services, companies can assist customers and process orders or reservations with automated, scalable bots.

## Relevant use cases

Other relevant use cases include:

- Viewing a restaurant take-out menu and ordering food
- Checking hotel availability and reserving a room
- Searching available photos and ordering prints

## Architecture



This scenario covers a conversational bot that functions as a concierge for a hotel. The data flows through the scenario as follows:

1. The customer accesses the chatbot with a mobile or web app.
2. Using Azure Active Directory (Azure AD) B2C (business-to-consumer), the user is authenticated.
3. Interacting with the Bot Service, the user requests information about hotel availability.
4. Cognitive Services processes the natural language request to understand the customer communication.
5. After the user is happy with the results, the bot adds or updates the customer's reservation in a SQL Database.
6. Application Insights gathers runtime telemetry throughout the process to help the DevOps team with bot performance and usage.

## Components

- [Azure Active Directory \(Azure AD\)](#) is Microsoft's multitenant cloud-based directory and identity management service. Azure AD supports a B2C connector allowing you to identify individuals using external IDs such as Google, Facebook, or a Microsoft Account.
- [App Service](#) enables you to build and host web applications in the programming language of your choice without managing infrastructure.
- [Bot Service](#) provides tools to build, test, deploy, and manage intelligent bots.
- [Cognitive Services](#) lets you use intelligent algorithms to see, hear, speak, understand, and interpret your user needs through natural methods of communication.
- [SQL Database](#) is a fully managed relational cloud database service that provides SQL Server engine compatibility.
- [Application Insights](#) is an extensible Application Performance Management (APM) service that lets you monitor the performance of applications, such as your chatbot.

## Alternatives

- [Microsoft Speech API](#) can be used to change how customers interface with your bot.
- [QnA Maker](#) can be used as to quickly add knowledge to your bot from semi-structured content like an FAQ.
- [Translator Text](#) is a service that you might consider to easily add multi-lingual support to your bot.

## Considerations

### Availability

This scenario uses Azure SQL Database for storing customer reservations. SQL Database includes zone redundant databases, failover groups, and geo-replication. For more information, see [Azure SQL Database availability capabilities](#).

### Scalability

This scenario uses Azure App Service. With App Service, you can automatically scale the number of instances that run your bot. This functionality lets you keep up with customer demand for your web application and chatbot. For more information on autoscale, see [Autoscaling best practices](#) in the Azure Architecture Center.

For other scalability topics, see the [scalability checklist](#) in the Azure Architecture Center.

### Security

This scenario uses Azure Active Directory (Azure AD) B2C, a business-to-consumer identity management service, to authenticate users. With Azure AD B2C, your chatbot doesn't store any sensitive customer account information or credentials. For more information, see the [Azure AD B2C overview](#).

Information stored in Azure SQL Database is encrypted at rest with transparent data encryption (TDE). SQL Database also offers Always Encrypted which encrypts data during querying and processing. For more information on SQL Database security, see [Azure SQL Database security and compliance](#).

For general guidance on designing secure solutions, see the [Azure Security Documentation](#).

## Resiliency

This scenario uses Azure SQL Database for storing customer reservations. SQL Database includes zone redundant databases, failover groups, geo-replication, and automatic backups. These features allow your application to continue running if there is a maintenance event or outage. For more information, see [Azure SQL Database availability capabilities](#).

To monitor the health of your application, this scenario uses Application Insights. With Application Insights, you can generate alerts and respond to performance issues that would impact the customer experience and availability of the chatbot. For more information, see [What is Application Insights?](#)

For other resiliency topics, see [Designing reliable Azure applications](#).

## Deploy the scenario

This scenario is divided into three components for you to explore areas that you are most focused on:

- [Infrastructure components](#). Use an Azure Resource Manager template to deploy the core infrastructure components of an App Service, Web App, Application Insights, Storage account, and SQL Server and database.
- [Web App Chatbot](#). Use the Azure CLI to deploy a bot with the Bot Service and Language Understanding and Intelligent Services (LUIS) app.
- [Sample C# chatbot application](#). Use Visual Studio to review the sample hotel reservation C# application code and deploy to a bot in Azure.

## Prerequisites

You must have an existing Azure account. If you don't have an Azure subscription, create a [free account](#) before you begin.

## Walk-through

To deploy the infrastructure components with a Resource Manager template, perform the following steps.

1. Click the link below to deploy the solution.



2. Wait for the template deployment to open in the Azure portal, then complete the following steps:



- Choose to **Create new** resource group, then provide a name such as *myCommerceChatBotInfrastructure* in the text box.
- Select a region from the **Location** drop-down box.
- Provide a username and secure password for the SQL Server administrator account.
- Review the terms and conditions, then check **I agree to the terms and conditions stated above**.
- Select the **Purchase** button.

It takes a few minutes for the deployment to complete.

## Deploy Web App chatbot

To create the chatbot, use the Azure CLI. The following example installs the CLI extension for Bot Service, creates a resource group, then deploys a bot that uses Application Insights. When prompted, authenticate your Microsoft

account and allow the bot to register itself with the Bot Service and Language Understanding and Intelligent Services (LUIS) app.

Azure CLI	 Copy	 Try It
<pre># Install the Azure CLI extension for the Bot Service az extension add --name botservice --yes  # Create a resource group az group create --name myCommerceChatbot --location eastus  # Create a Web App Chatbot that uses Application Insights az bot create \   --resource-group myCommerceChatbot \   --name commerceChatbot \   --location eastus \   --kind webapp \   --sku S1 \   --insights eastus</pre>		

## Deploy chatbot C# application code

A sample C# application is available on GitHub:

- [Commerce Bot C# sample](#)

The sample application includes the Azure Active Directory authentication components and integration with the Language Understanding and Intelligent Services (LUIS) component of Cognitive Services. The application requires Visual Studio to build and deploy the scenario. Additional information on configuring Azure AD B2C and the LUIS app can be found in the GitHub repo documentation.

## Pricing

To explore the cost of running this scenario, all of the services are pre-configured in the cost calculator. To see how the pricing would change for your particular use case, change the appropriate variables to match your expected traffic.

We have provided three sample cost profiles based on the number of messages you expect your chatbot to process:

- [Small](#): this pricing example correlates to processing < 10,000 messages per month.
- [Medium](#): this pricing example correlates to processing < 500,000 messages per month.
- [Large](#): this pricing example correlates to processing < 10 million messages per month.

## Related resources

For a set of guided tutorials for the Azure Bot Service, see the [tutorial section](#) of the documentation.