



NHS Chatbot Generation Service Manual

COMP0016 2022/2023 Team 12

NHS Auto-chatbot

NHS Chatbot Generation Service Manual

Index

<i>NHS Chatbot Generation Service Manual</i>	2
Index	2
Our websites.....	3
<i>Overview</i>	4
Why use our service?	4
How to use our service?	4
Limitations of our service	4
<i>How to Generate a Chatbot</i>	5
Prerequisites.....	5
Steps for Generating the Chatbot.....	6
Steps for Embedding the Chatbot to Your Website	7
<i>Improve Your Chatbot Conversations</i>	8
Monitor Conversations.....	8
Edit Conversations	9
<i>Deploy the Chatbot Generation Service</i>	10
Deploy Locally.....	10
Deploy on Server.....	11
Environment Variables	12
Django Backend (env.py)	12
React.js Frontend (env.js)	14
Azure functions	17
Cost Estimation	18
<i>Documentation for API calls</i>	19
/url_list.....	19
/generate_json_e.....	20
/register_url.....	20
/get_json	21
/del_json	21

Our websites

Generation Service Demo: <http://nhschatbotgenerator.uksouth.cloudapp.azure.com:3000>

Portfolio: <https://students.cs.ucl.ac.uk/2022/group12/index.html>

Frontend Source Code: <https://github.com/Zhiyu-Wang-2021/chatbot-view>

Backend Source Code: <https://github.com/Zhiyu-Wang-2021/chatbot-generator>

Azure Functions Source Code: <https://github.com/Zhiyu-Wang-2021/BingAccessForWatson>

Overview

Why use our service?

NHS trusts and clinics often have the essentials in their services mapped online. Some small trusts and clinics even do not have many IT people in their organisations. If they want to create a chatbot for each trust and clinic from scratch, they need to hire IT people that have advanced skills to develop and maintain it. Moreover, the cost of creating the chatbot can be very expensive as it needs more people and a lot of time to develop. In addition, most people nowadays prefer consulting medical questions from their phones rather than going to the hospital or clinic. Therefore, some NHS trusts and clinics need to have their chatbot.

Our goal for this project is to support all NHS trusts and clinics to create a low-cost (approximately 140-200£/month) chatbot easily without hiring many IT people. To help them reduce the cost and time of developing a chatbot from scratch, we build a chatbot generation service that can create a chatbot for any NHS trust's websites, the so-called NHSBot. There are multiple layers of technology stack behind this service so that the user can get the best possible answer. The predefined questions will get the answer from our web scraping tool, and other general questions will retrieve from Bing API. Our generation service is quick, adaptable, user-friendly, and maintainable. Every trust and clinic only needs 20-30 minutes to create a chatbot with 3 simple steps, and they can implement it into their website. In the future, they can also improve and maintain it by modifying the chatbot dialogue.

How to use our service?

You can try this [demo](#) NHS Chatbot Generation Service or deploy your own NHS Chatbot Generation Service by following the instructions in the "Deploy This Service" section. After opening the service, follow the instructions in the "How to Generate a Chatbot" section to generate your bot.

Limitations of our service

All the information your chatbot knows comes from your website, so **it cannot answer information that is not on your website**. For example, if your website does not have information on opening hours, your chatbot cannot answer the corresponding questions. **But you can easily edit the dialogue manually to include more information.** (See Improve Your Chatbot Conversations).

Rather than generating a chatbot that can be immediately deployed to a website, we generate a JSON file that can create an IBM Watson Assistant chatbot. Therefore, **you will need to manually upload the Dialog JSON file to your IBM Watson Assistant instance after generating it in our Chatbot Generation Service.** However, the uploading process should only take you less than 10 minutes.

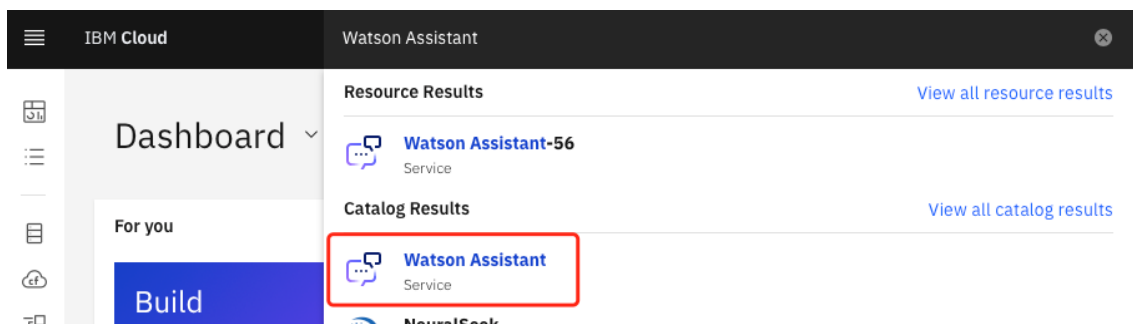
How to Generate a Chatbot

Prerequisites

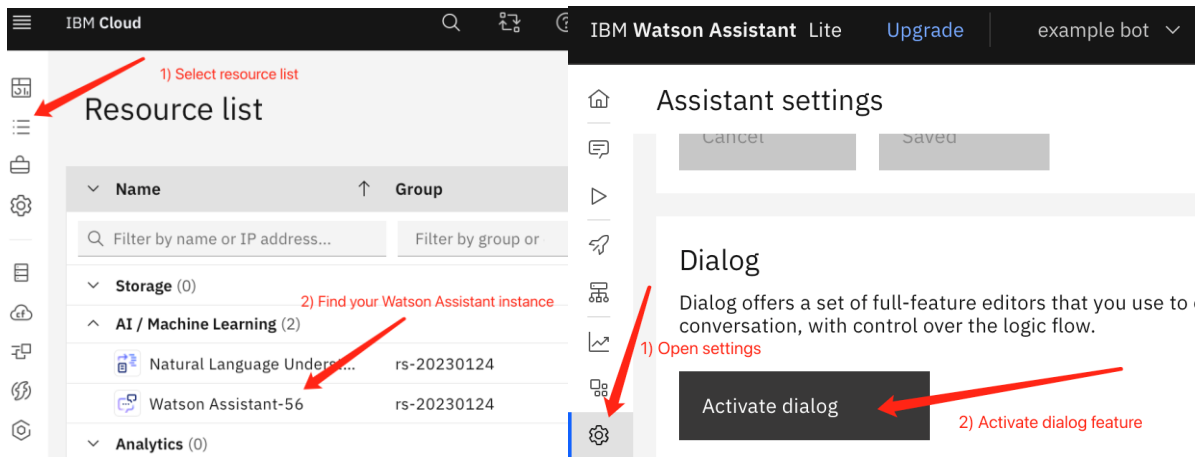
Estimated time: 10 min. (These prerequisites are only required for deploying your chatbot on your website. You will not need them for trying our chatbot generation service demo.)

Since our chatbot generation service is based on IBM Watson Assistant, you will need to have 1) an **IBM Cloud Account**, and 2) an **IBM Watson Assistant instance**.

You can create an account here: <https://www.ibm.com/cloud>. After logging in, you can search for the Watson Assistant resource and create your instance.



If you have created your Watson Assistant successfully, you can find it in your resource list. Launch it once you find it. Lastly, please activate the Dialog feature of your Watson Assistant.



Steps for Generating the Chatbot

Estimated time: 20 min.

Step 1: Upload the link to your trust's official website to this service, giving you a unique reference code corresponding to your chatbot.

URL to your website homepage

https:// www.lawsonpractice.nhs.uk

GENERATE MY CHATBOT

Step 2: Wait for our information extraction system to retrieve key information from your website.

Chatbot Reference Code

641040a969a3d662aa059a42

GET MY CHATBOT JSON FILE

Step 3: Once a success message is displayed, you can use the unique reference code you have just obtained to download your chatbot's Dialog JSON file. (The reference code can also be found in the Generation History.) Before embedding your chatbot to your website, you can try your chatbot directly on this page by clicking the "PREVIEW" button.

Chatbot JSON

```
{
  "intents": [
    {
      "intent": "appointment",
      "examples": [
        {
          "text": "appointment",
          "text": "book appointment",
          "text": "Can I book an appointment?",
          "text": "do you know how to make an appointment",
          "text": "How do I book an appointment?",
          "text": "How to cancel an appointment?",
          "text": "how to get an appointment",
          "text": "how to see a doctor",
          "text": "I want to book an appointment",
          "text": "I want to cancel my appointment",
          "text": "I want to schedule an appointment",
          "text": "I want to see a doctor",
          "text": "make an appointment",
          "text": "tell me how to make an appointment"
        ],
        "description": "",
        "intent": "ask_more_info",
        "examples": [
          {
            "text": "are there any more things to share",
            "text": "can you describe more",
            "text": "can you direct me to some links on your"
          }
        ]
      ]
    }
  ]
}
```

Your Chatbot JSON will appear here after inputting your reference code

DOWNLOAD PREVIEW YOUR CHATBOT

Preview

Hi! I'm NHSBot. How can I help you?

How to get an appointment?

The Lawson Practice » Register With Us » Request an appointment Take Control With eConsult and the NHS App You can request for an appointment, prescriptions, GP letter and much more using our online service. This can save you time waiting to speak to our GP or for a face-to-face appointment.

Generation History

REFRESH

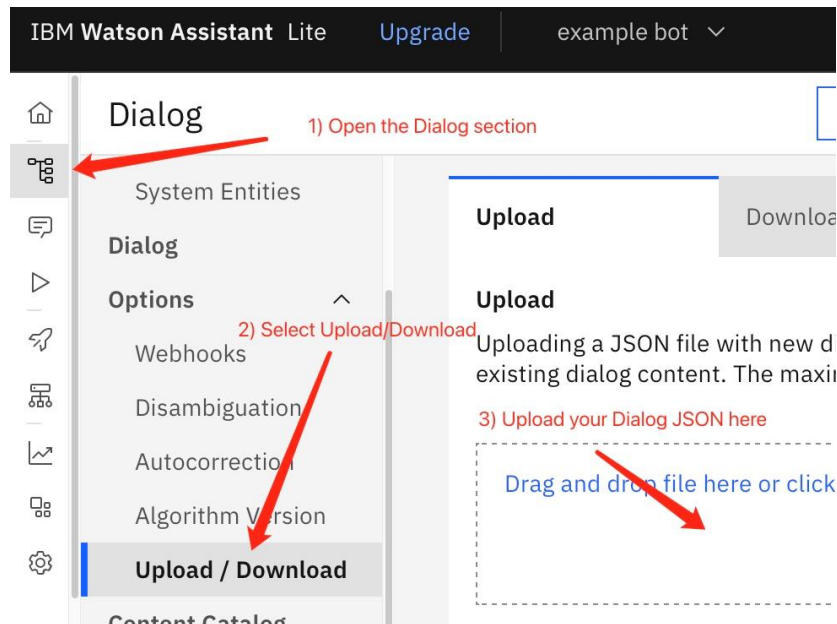
Website	Reference Code
www.lawsonpractice.nhs.uk	641040a969a3d662aa059a42
www.princeofwalesgrouppractice.nhs.uk	64103e7b69a3d662aa059a42
www.sobellmedicalcentre.nhs.uk	64103cc069a3d662aa059a42

Write your message here

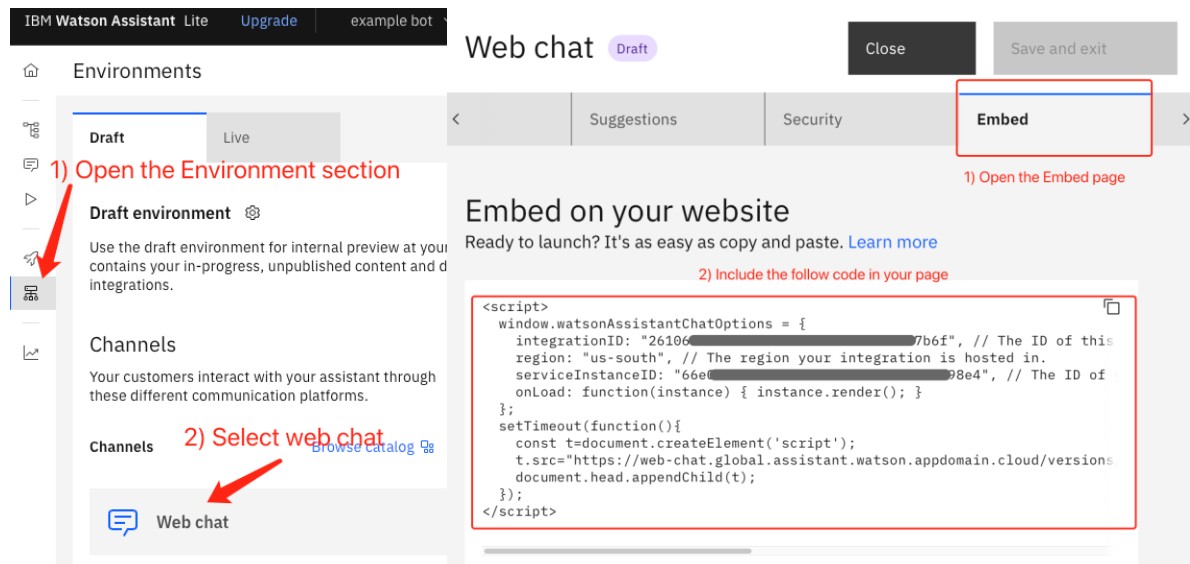
Steps for Embedding the Chatbot to Your Website

Estimated time: 10 min.

Step 1: You can upload the Dialog JSON file you get from our service to your IBM Watson Assistant as shown in the image. When the Dialog JSON file is uploaded, you can try it by clicking the play button, which is in the Preview section.



Step 2: To embed your IBM Watson Assistant to your webpage, you only need to find this prewritten code in your Watson Assistant management page and paste this code to your website.



Improve Your Chatbot Conversations

Because the information gathered by your chatbot comes from your trust's website, there are bound to be questions from patients that cannot be answered effectively. But with Watson Assistant, you can efficiently monitor conversations and improve them. (No programming required.)

Monitor Conversations

You can monitor conversations by following the steps illustrated below. Here you can see the logs of each conversation.

IBM Watson Assistant Trial | 10 days left | [Extend trial](#) | NHS Chatbot Tes... ▾

Analyze

Overview
Action completion
Recognition
Conversations
Filter by topics
Filter by keyword or session ID

Environment: Draft ▾ | Select date range: Past 2 weeks ▾ | Custom date range: 27 Feb 23 to 11 Mar 23

Filter by topics: Topics ▾ | Filter by keyword or session ID: Search

Conversations	Topics	Requests
11 Mar at 12:07 PM c1509535-297a-43b7...	#greetings	"Hi"
	#location_info	"can you tell me the address?"
11 Mar at 12:01 PM 6a50f112-fd5f-4875...	#location_info	"What is the address"
	! Unrecognized	"What is aspirin"
	! Unrecognized	"I have a fever"

1) Open Analyze page

2) Choose this page

And by clicking one of the conversations, you can see the full conversation history that includes your chatbot's responses. The 'Unrecognized' sign means that your chatbot will call a webhook to search

over your website immediately and reply with a reasonable answer.

The full conversation history

Environment: Draft | Select date range: Past 2 weeks | Custom date range: 27 Feb 23 to: ...

Filter by topics: Topics | Filter by keyword or phrase: Search

Conversations	Topics	Requests
11 Mar at 12:01 PM 6a50f172-fd5f-4875...	#location_info	"What is ..."
	! Unrecognized	"What is ..."
	! Unrecognized	"I have a ..."
	#phone_number	"How shc ..."

Click here

11 March at 12:01:08 PM

Duration	Requests	Recognized requests	Searches
3 m 29 secs	5	3	0

Hello! I can answer you questions based on the information provided on this website. How can I help you?

What is the address

#location_info started

We are at this address: Great Ormond Street Hospital Great Ormond Street London WC1N 3JH

What is aspirin

#greetings started

Aspirin is a medicine commonly used to relieve pain, reduce swelling and reduce a high temperature. It also makes the blood less sticky so it is less likely to form a clot. In adults, aspirin is often given to people with heart problems because of this action.

I have a fever

#greetings started

Edit Conversations

If your chatbot answers a question incorrectly or fails to answer it, you can add or delete the generated conversation rules here.

IBM Watson Assistant Trial | 10 days left | Extend trial | NHS Chatbot Tes... |

Dialog

- Intents
- Entities
 - My Entities
 - System Entities
- Dialog**
- Options
 - Webhooks
 - Disambiguation
 - Autocorrection
 - Algorithm Version
 - Upload / Download
- Content Catalog

Add node | Add child node | Add folder

You can add a new node

You can also edit the existing ones

- Welcome
welcome
1 Responses / 0 Context Set / Does not return
- greetings
#greetings
1 Responses / 0 Context Set / Does not return
- No condition set
0 Responses / 0 Context Set / Does not return

Deploy the Chatbot Generation Service

Recommend configuration: Python 3.8.10, PIP 20.0.2, Node 18.13.0, NPM 8.19.3

Deploy Locally

Django Backend

Step 1: Clone the backend code from GitHub.

```
git clone git@github.com:Zhiyu-Wang-2021/chatbot-generator.git
```

Step 2: Create a Python virtual environment.

```
python3 -m venv venv
```

Step 3: Activate your new virtual environment.

Mac/Linux:

```
source ./venv/bin/activate
```

Windows:

```
source ./venv/Scripts/activate
```

Step 4: Install dependencies in this virtual environment.

```
pip install -r requirements.txt
```

Step 5: Run the backend.

```
python3 ./chatbotSite/manage.py runserver --nothreading --noreload
```

React.js Frontend

Step 1: Clone the frontend code from GitHub.

```
git clone git@github.com:Zhiyu-Wang-2021/chatbot-view.git
```

Step 2: Install dependencies.

```
npm install
```

Step 3: Run the frontend service.

```
npm start
```

Deploy on Server

Recommend configuration: Linux (Ubuntu 20.04), 1 CPU, 2 GB memory.

Django Backend

Steps 1-4: The same as the Step 1-4 of local deployment.

Step 5: Run the backend.

```
nohup python3 ./chatbotSite/manage.py runserver 0.0.0.0:8000 --nothreading --noreload &
```

To stop running the backend service, use the following command to find the <process_id> of the backend service process and stop the process.

```
sudo kill $(lsof -t -i:8000)
```

*If the above command failed, add the '-9' parameter for force stop.

```
sudo kill -9 $(lsof -t -i:8000)
```

To check the logs via:

```
tail nohup.out
```

React.js Frontend

Steps 1-2: The same as Steps 1-2 of local deployment.

Step 3: Install the 'pm2' package globally using NPM. (No need for this step if 'pm2' is already installed.)

```
npm install pm2 -g
```

Step 4: Run the frontend service.

```
pm2 start npm -- start
```

To stop running the frontend service, use the following command to find the <process_id> first.

```
pm2 ps
```

Then terminate this process.

```
pm2 stop <process_id>
```

Environment Variables

Django Backend (env.py)

```
MONGODB_URL =
"mongodb://nhschatbotdb:fr3tHFqVSCFauKgWvMiigj3nNYi67yfXKvahPKekoUdXVT88HgP
vlnqL4FY2875XLn6TsiJMyXqgACDbeaftmQ==@nhschatbotdb.mongo.cosmos.azure.com:1
0255/?ssl=true&retrywrites=false&replicaSet=globaldb&maxIdleTimeMS=120000&a
ppName=@nhschatbotdb@"

BING_AZURE_FUNC_URL =
"https://bingaccessforwatson.azurewebsites.net/api/useQnA2ImproveAnswer?cod
e=41mEZaj2kjJphYWmK6eHPFzQwQHliVyW61QpEvrpGUcbAzFuAwDC3w=="

BING_API_KEY = "ef99b91a0209431cb66dd4d32a0b20c6"

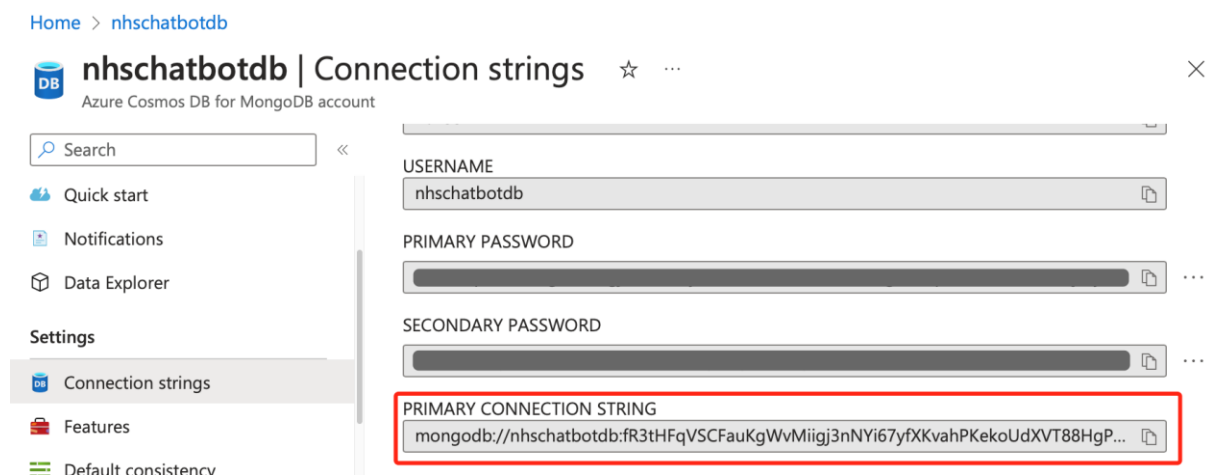
IBM_NLU_URL = 'https://api.us-south.natural-language-
understanding.watson.cloud.ibm.com/instances/2019387f-a518-46b0-aa72-
ccc03cdf67c8'

IBM_NLU_API_KEY = 'Iig5IogwUhp2dIo1FqAUEjkka7OoC42qXbWHWucIU9oU'
```

This is the sample env.py file from our demo. You can change them to use your own IBM Cloud, MS Azure cloud, and MongoDB service. The following is the explanation for each of the variables:

MONGODB_URL

The URL of your MongoDB database.



The screenshot shows the Azure Cosmos DB portal interface. At the top, it says 'Home > nhschatbotdb'. Below that, the account name 'nhschatbotdb | Connection strings' is displayed with a star icon and a close button. Underneath, it says 'Azure Cosmos DB for MongoDB account'. On the left, there is a navigation menu with options like 'Quick start', 'Notifications', 'Data Explorer', 'Settings', 'Connection strings', 'Features', and 'Default consistency'. The 'Connection strings' option is selected. The main area shows a search bar and a list of connection strings. The 'PRIMARY CONNECTION STRING' is highlighted with a red box and contains the value: 'mongodb://nhschatbotdb:fr3tHFqVSCFauKgWvMiigj3nNYi67yfXKvahPKekoUdXVT88HgP...'

BING_AZURE_FUNC_URL

The URL of your “useQnA2ImproveAnswer” Azure Function. To learn more about the Azure functions we use, see the “Azure Functions” section.

BING_API_KEY

The API key of your Bing API.

Search

Regenerate Key1 Regenerate Key2

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

RESOURCE MANAGEMENT

- Quick start
- Keys and Endpoint**
- Pricing tier
- Billing By Subscription
- Properties

These keys are used to access your Bing resource. Do not share your keys. Store them securely-- for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.

Show Keys

Key 1	
Key 2	
Endpoint	https://api.bing.microsoft.com/	
Location	global	



IBM_NLU_URL

The URL of your IBM Natural Language Understanding service.

IBM_NLU_API_KEY

The API key of your IBM Natural Language Understanding service.

Resource list / Natural Language Understanding-12 Active Add tags Details Actions...

Manage

- Getting started
- Service credentials
- Plan
- Connections


Start by viewing the tutorial


[Getting started tutorial](#) [API reference](#)

Plan
Lite
[Upgrade](#)

Credentials

[Download](#) [Show credentials](#)

API key:
..... 

URL:
https://api.us-south.natural-language-understa... 

React.js Frontend (env.js)

```
export const iamKey = "fK1ry7X3D0HFVHC_kJXK1LA6900sxeHH6Yc0MKtEdGHm"
export const instanceId = "d049a552-ee69-4a04-aa9b-1fa4e7994ece"
export const backendBaseUrl = http://127.0.0.1:8000/

export const bingApiConnections = {

  getBearerToken:
  'https://bingaccessforwatson.azurewebsites.net/api/getIAMBearerToken?code=G5Im90xAR8gYcw8MXCMslqQPQXrIfvQnrXWYR3H1152TAzFuJ6xz9g%3D%3D',

  sendMsgToWatsonAssistant:
  'https://bingaccessforwatson.azurewebsites.net/api/sendMsgToWtsnAssistant?code=bbQ1-v-eerh6tNjMAQ_uVzIYi6Kp4fst5pt-Z4SSv4BdAzFu5162rQ%3D%3D',

  getWorkspaceList:
  'https://bingaccessforwatson.azurewebsites.net/api/workspaceMgmt4WtsnAssistant?code=XEs8yTiSgUNT3B0pmHSuEasXN9Gp7MXHSHp2s9wLo80AzFuWfIZpg%3D%3D',

  newWorkspace:
  'https://bingaccessforwatson.azurewebsites.net/api/newWorksapce4WA?code=64JQYzH544yWpZis1C7oA-vWBjYdk43qJCq4DnUEzOijAzFukK8QFw%3D%3D',

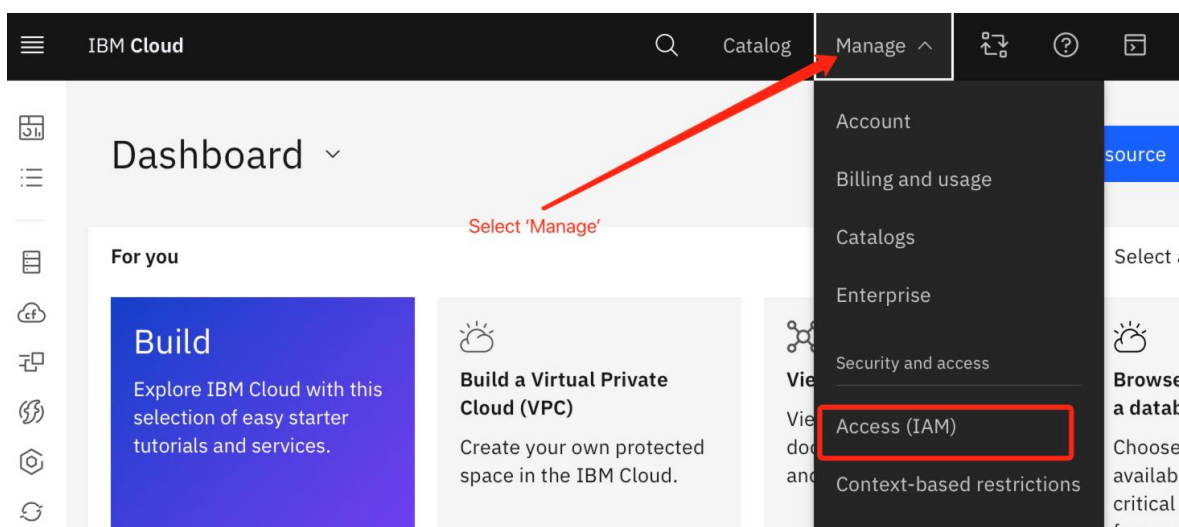
  delWorkspaceById:
  'https://bingaccessforwatson.azurewebsites.net/api/delWorkspace4WAbYID?code=EWv9dpPsZAMxh0ATFHdjYBBChES7mSL73fq81x0sRlalAzFutnXGxw%3D%3D',

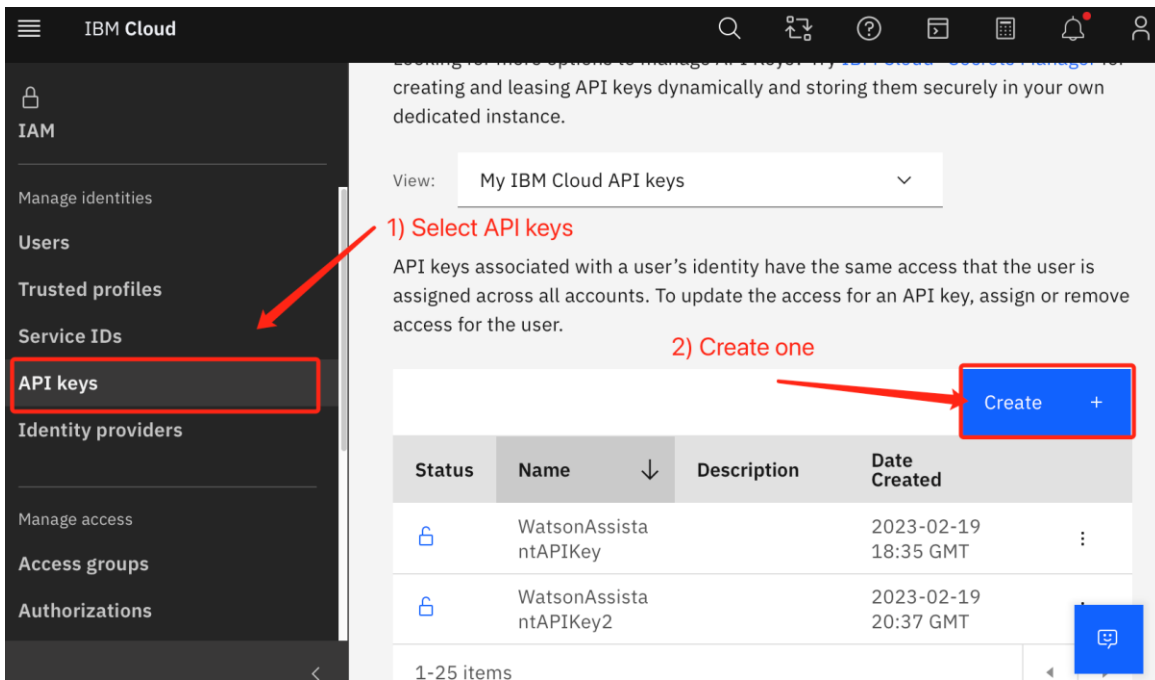
}
```

This is the sample env.js file from our demo. You can change them to use your own Watson Assistant instance and Django backend service. The following is the explanation for each of the variables:

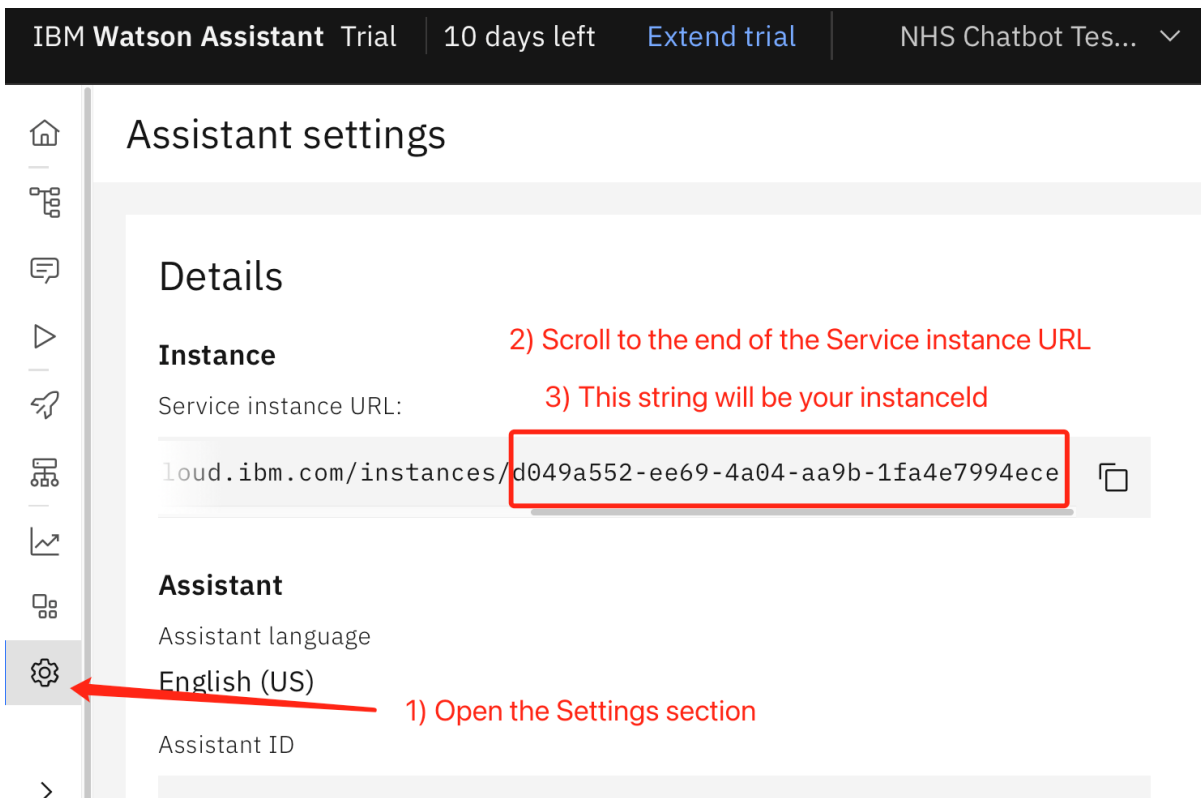
iamKey

This is the key to accessing IBM Cloud API. You can generate this key in IBM Cloud by these steps:





instanceld



backendBaseUrl

This is the base URL to your Django backend; it depends on where you deploy your backend service. It should be "http://127.0.0.1:8000/" by default for a locally run backend.

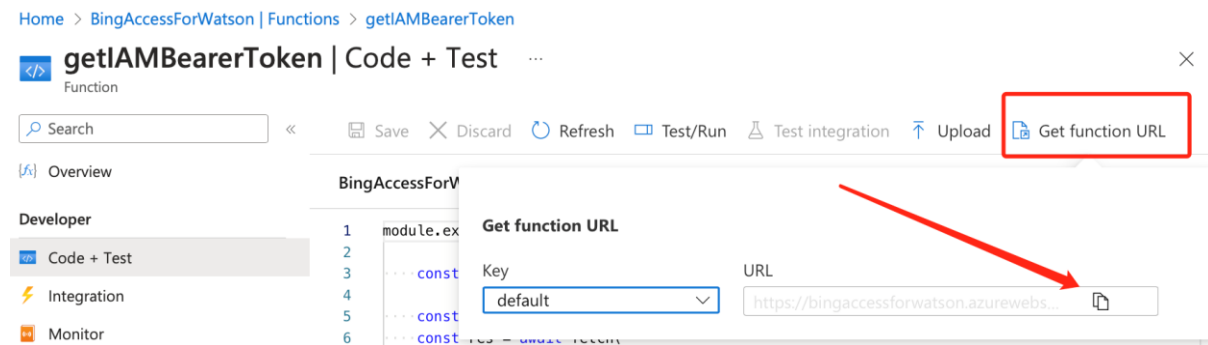
bingApiConnections

This is an object that stores the URLs to corresponding Azure Functions. To learn more about the Azure Functions we use, see the Azure Functions section and this GitHub [page](#) for the source codes. To learn more about the Azure functions we use, see the “Azure Functions” section.

Azure functions

We are using Azure functions to allow our chatbot generation service to communicate with Watson Assistant API and allow Watson Assistant to call Bing API. You can deploy all the Azure Function on your Azure platform and replace the Azure Function URLs in the environment variables. Here is the [link](#) to the GitHub repository of all the Azure functions. They are all designed to be called internally in the chatbot generation service and you should avoid calling them outside of the service.

You can get the URL to an Azure Function as shown in the image below:



Here is the explanation of the Azure Functions we have used in our service.

getIAMBearerToken:

This is for getting the Bearer token by your IBM Cloud IAM key. The Bearer token is used for communicating with Watson Assistant API.

sendMsgToWtsnAssistant:

This is for sending your message to Watson Assistant and getting a response.

workspaceMgmt4WtsnAssistant:

This is for getting the Watson Assistant workspace list.

newWorkspace4WA:

This is for creating a new Watson Assistant workspace.

delWorkspace4WAbbyID:

This is for deleting all the Watson Assistant workspaces.

useQnA2ImproveAnswer:

This is the webhook that Watson Assistant will call when it cannot answer the question based on our web scraping information. This webhook will call Bing API and Azure Language Service to provide the user with a reasonable response.

bingAccessWebHook:

This is the webhook for accessing Bing API without using Azure Language Service to improve the answer. It is solely based on the Bing search engine. This webhook will be automatically called by the "useQnA2ImproveAnswer" function.

Cost Estimation

The following table shows a sample cost estimation based on Great Ormond Street Hospital's data. GOSH receives 280,000 patients annually, and we can assume that 1 out of 20 patients will use a chatbot on GOSH's website. In such a case, there should roughly be 1,200 monthly active users and 4,000 questions being asked monthly.

Service Name	Functionality	Service Tier	Cost (£/month)
IBM Watson Assistant	Deploy chatbot on the web	Plus	168
IBM Natural Language Understanding	Information retrieving (only used during the chatbot generation phase)	Free	0
Bing Search API	Answer general questions	S2	12
Azure Language Service	Improve answers from Bing API	S	5
Azure Functions	Connect our service, Bing API, and Watson Assistant	Consumption (Total Executions)	0
Total			185

Sources:

[IBM Watson Assistant - Pricing](#)

[IBM Natural Language Understanding - Pricing](#)

[Bing Search API Pricing | Microsoft Bing](#)

[Pricing - Language Service | Microsoft Azure](#)

[Pricing - Functions | Microsoft Azure](#)

Documentation for API calls

By looking at this guide you can see how to use our chatbot generation only by the Django API and without using the react front-end part. Typically, you will call the API in this sequence to generate a JSON file for your chatbot: `"/register_url" -> "/generate_json_e" -> "/get_json"`.

[/url_list](#)

Description: List all the URLs and corresponding dialogue reference codes. The dialogue reference code is the Object ID generated by MongoDB automatically.

Type: GET

Sample Response: application/json

```
[
  {
    "_id": {
      "$oid": <String of this object's Object ID>
    },
    "dialog_ref": {
      "$oid": <String of the dialogue object's Object ID>
    },
    "url": <String of your website URL>
  }, ...
]
```

[/generate_json_e](#)

Description: Generate a dialogue JSON from an existing dialogue JSON template. You need to call `/register_url/` first to register your URL in the database and get the dialogue reference code. After getting the "success" response, you can get your chatbot from the database by the dialogue reference code.

Type: POST

Body: application/json

```
{  
  
  "ref": <String of the dialogue object's Object ID>,  
  
  "url": <String of your website URL>  
  
}
```

Sample Response: text/html; charset=utf-8

```
"success"
```

[/register_url](#)

Description: Register your URL in the database and get the dialogue reference code.

Type: POST

Body: application/json

```
{  
  
  "url": <String of your website URL>  
  
}
```

Sample Response: text/html; charset=utf-8

```
<String of the dialogue object's Object ID>
```

[/get_json](#)

Description: Get a dialogue JSON file by its Object ID. You can upload the dialogue JSON to Watson Assistant to deploy your chatbot.

Type: GET

URL Params:

```
"id": <String of the dialogue object's Object ID>
```

Sample Response: application/json

```
<JSON of this dialogue>
```

[/del_json](#)

Description: Delete all the dialogue JSONs and registered URLs in the database.

Type: DELETE

Sample Response: text/html; charset=utf-8

```
"deleted <the total amount of deleted JSONs> JSONs."
```