Project Synopsis:

Creating a Demo Using Azure Cognitive Services: Text-to-Speech.

To create a simple application that converts text input into spoken audio using Azure Cognitive Services' Text-to-Speech service.

Architectural Overview

Components:

1. **User Interface (UI)**: A simple web or mobile application interface where users input text .
2. **Azure Cognitive Services - Text-to-Speech**: The core service converts text input into spoken audio.
3. **API Gateway**: Manages and routes requests between the UI and the Text-to-Speech service.
4. **Backend Server**: Handles business logic, communicates with the Text-to-Speech API, and serves the audio to the UI.
5. **Database**: Stores user preferences, logs, and any other necessary data.

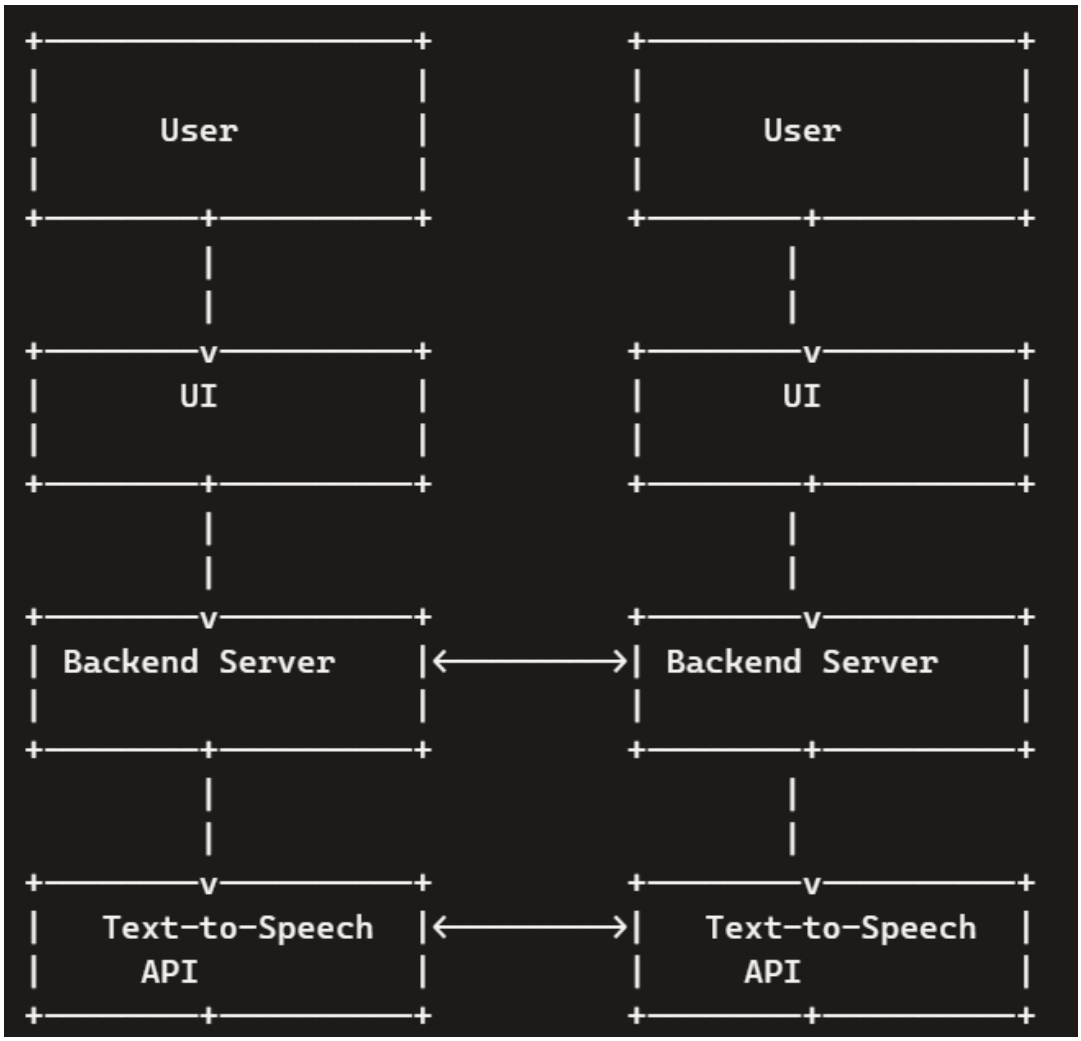
Workflow:

1. **User Input**: The user types of text into the application interface.
2. **API Request**: The application sends the text to the backend server via an API call.
3. **Backend Processing**: The backend server processes the request and sends the text to t he Azure Text-to-Speech API.
4. **Text-to-Speech Conversion**: The Text-to-Speech service converts the text into spoken audio.
5. **Audio Delivery**: The backend server receives the audio file and sends it back to the appl ication interface.

6. **User Playback**: The application plays the audio for the user.

Diagram

Here's a simple diagram to illustrate this architecture:



Steps to Implement:

1. **Set Up UI**: Create a simple web or mobile app interface for text input.
2. **Backend Server**: Develop a backend server to handle API requests and communicate w ith the Text-to-Speech API.
3. **Integrate Text-to-Speech API**: Use the Azure Cognitive Services Text-to-Speech API to convert text to audio.
4. **Deploy and Test**: Deploy the application and test the end-to-end functionality.

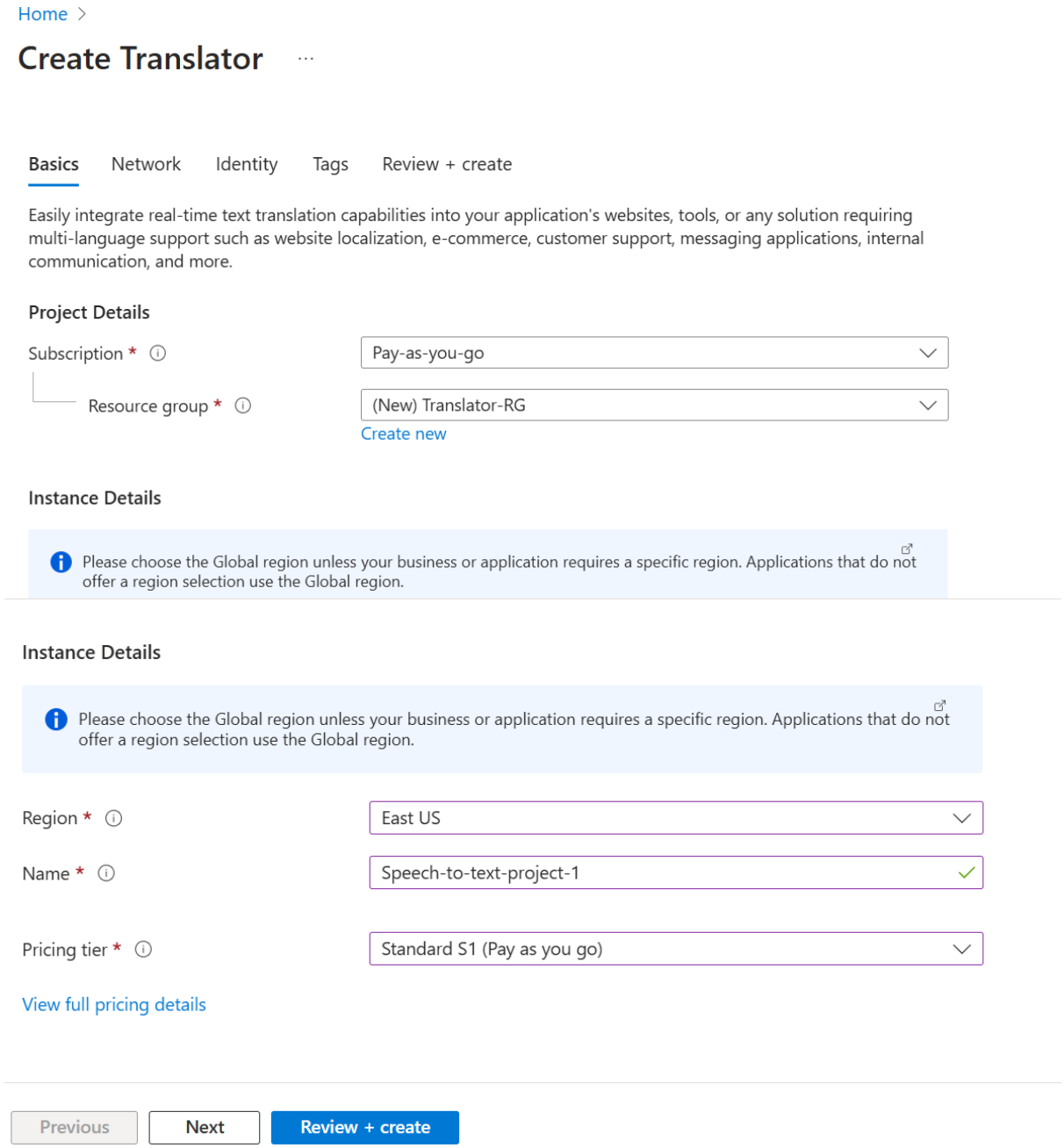
Prerequisites:

An Azure subscription

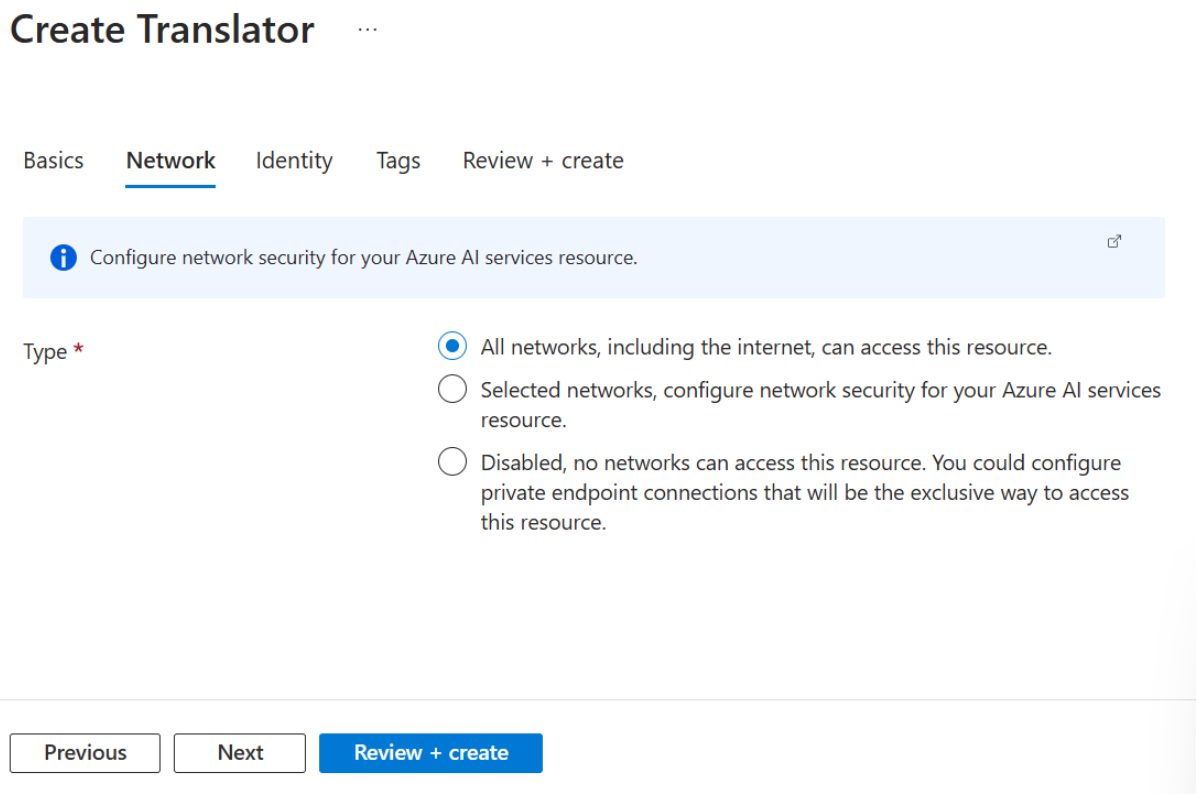
A Text-to-Speech resource created in Azure

Procedure:

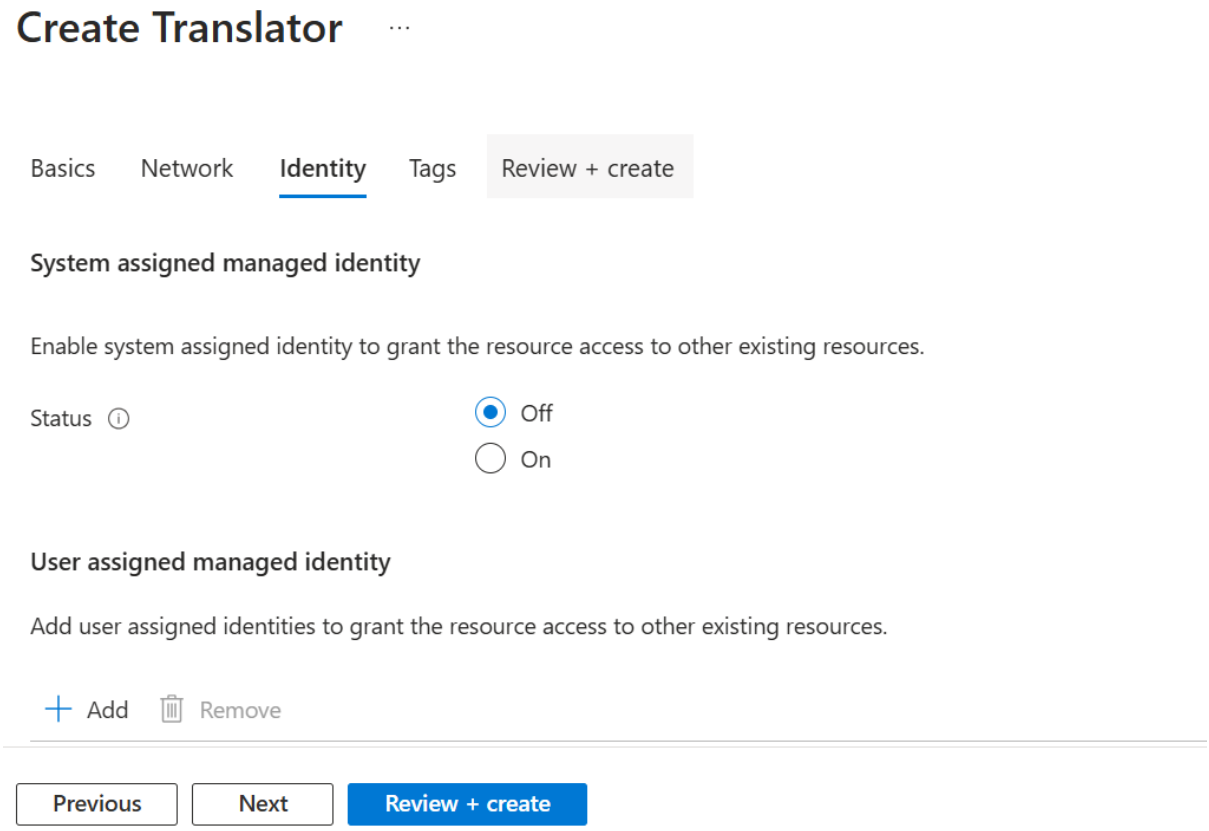
- Login to Azure portal
- Create Translator resource in Azure portal



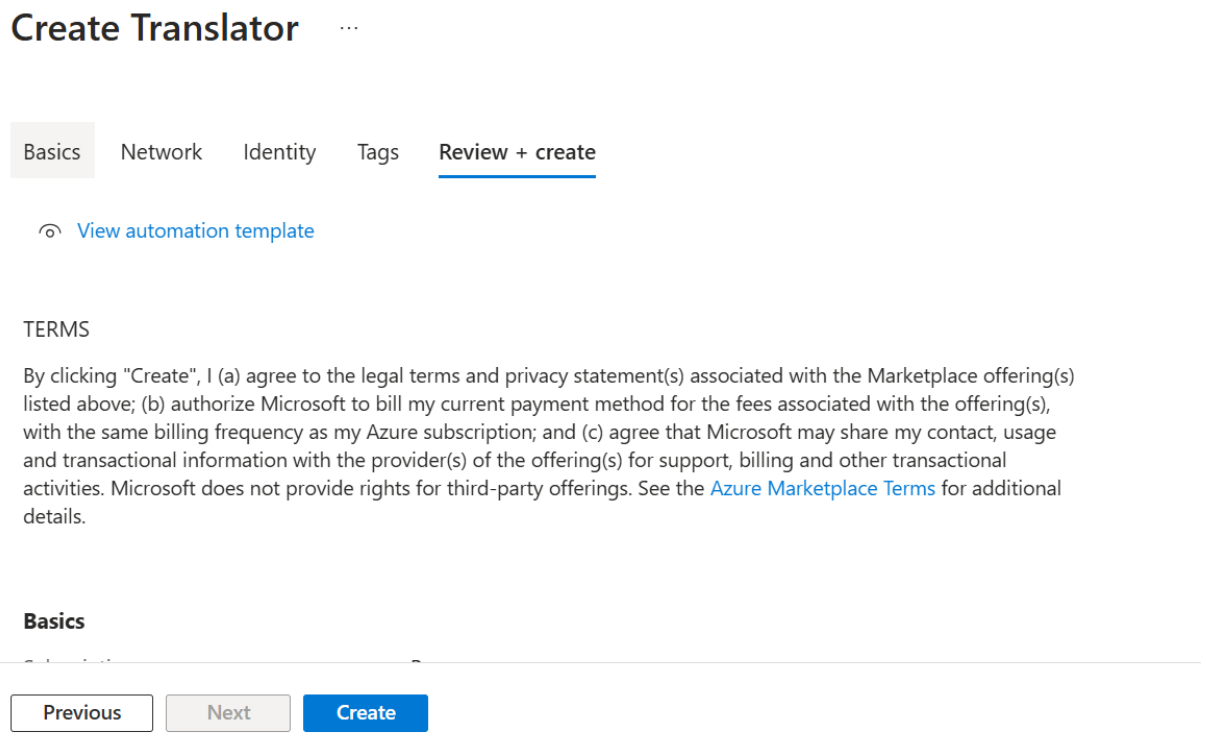
- Next: Network



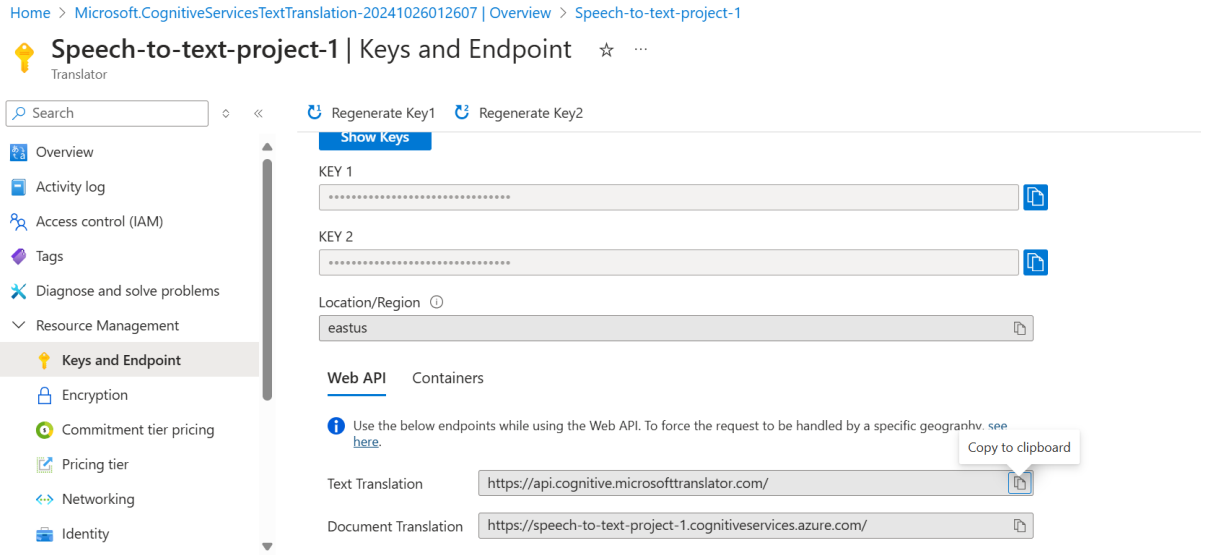
- Next: Identity



- Skip Tags
- Validate, Review and Create



- After your resource deploys, select **Go to** Resource Management-> Keys and Endpoint and retrieve your key and endpoint



- Make sure you have installed Python3 in your local
- sudo apt install python3-pip

```
root@LAPTOP-VJGJ8BEB:~# sudo apt install python3-pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot g++ g++-11 gcc gcc-11 gcc-11-base javascript-common
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libatomic1 libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libgcc-11-dev libgd3
  libgomp1 libisl23 libitm1 libjs-jquery libjs-sphinxdoc libjs-underscore liblsan0 libmpc3 libnsl-dev libpython3-dev
  libpython3.10-dev libquadmath0 libstdc++-11-dev libtirpc-dev libtsan0 libubsan1 libxpm4 linux-libc-dev lto-disabled-list
  make manpages-dev python3-dev python3-wheel python3.10-dev rpcsvc-proto zlib1g-dev
Suggested packages:
  bzip2-doc cpp-doc gcc-11-locales debian-keyring g++-multilib g++-11-multilib gcc-11-doc gcc-multilib autoconf automake
  libtool flex bison gdb gcc-doc gcc-11-multilib apache2 | lighttpd | httpd glibc-doc bzr libgd-tools libstdc++-11-doc
  make-doc
The following NEW packages will be installed:
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot g++ g++-11 gcc gcc-11 gcc-11-base javascript-common
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libatomic1 libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libgcc-11-dev libgd3
  libgomp1 libisl23 libitm1 libjs-jquery libjs-sphinxdoc libjs-underscore liblsan0 libmpc3 libnsl-dev libpython3-dev
  libpython3.10-dev libquadmath0 libstdc++-11-dev libtirpc-dev libtsan0 libubsan1 libxpm4 linux-libc-dev lto-disabled-list
  make manpages-dev python3-dev python3-pip python3-wheel python3.10-dev rpcsvc-proto zlib1g-dev
0 upgraded, 55 newly installed, 0 to remove and 7 not upgraded.
Need to get 69.5 MB of archives.
After this operation, 234 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-dev-bin amd64 2.35-0ubuntu3.8 [20.3 kB]
Ign:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 linux-libc-dev amd64 5.15.0-122.132
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 libcrypt-dev amd64 1:4.4.27-1 [112 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/main amd64 rpcsvc-proto amd64 1.4.2-0ubuntu6 [68.5 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtirpc-dev amd64 1.3.2-2ubuntu0.1 [192 kB]
```

- sudo apt-get update
- Create a new python file **“translator-app.py”** and paste the python code in the file

```
root@LAPTOP-VJGJ8BEB:~# sudo vi translator-app.py
root@LAPTOP-VJGJ8BEB:~# cat translator-app.py
import requests, uuid, json

# Add your key and endpoint
key = "0976244472f9e462b84fce2b7b8a71ea2"
endpoint = "https://api.cognitive.microsofttranslator.com/"

# location, also known as region.
# required if you're using a multi-service or regional (not global) resource. It can be found in the Azure portal on the Key
s and Endpoint page.
location = "eastus"

path = '/translate'
constructed_url = endpoint + path

params = {
    'api-version': '3.0',
    'from': 'en',
    'to': ['fr', 'zu']
}

headers = {
    'Ocp-Apim-Subscription-Key': key,
    # location required if you're using a multi-service or regional (not global) resource.
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}

# You can pass more than one object in body.
```

```
# You can pass more than one object in body.
body = [{
    'text': 'I would really like to drive your car around the block a few times!'
}]

request = requests.post(constructed_url, params=params, headers=headers, json=body)
response = request.json()

print(json.dumps(response, sort_keys=True, ensure_ascii=False, indent=4, separators=(',', ': ')))
root@LAPTOP-VJGJ8BEB:~#
```

- Type the following command in your console:

  sudo python3 translator-app.py

OUTPUT:

Reference:

https://learn.microsoft.com/en-gb/azure/ai-services/translator/quickstart-text-rest-api?tabs=python