Nagendra Rishi Raj Chanda

TEXT to Speech – 801309529

Endpoint: <http://104.248.226.24:3000/>

Tech stack used: REACT  
  
Swagger Definition

The Swagger definition is a document that describes the structure of an API. It includes information such as the endpoints, methods, parameters, request and response formats, and security requirements. Here's what's happening in the Swagger definition for the Microsoft Speech Service Text-to-Speech API:

Definitions

The definitions section of the Swagger definition describes the data models used in the API. In this case, there is a single data model called SpeakRequest, which is used to describe the format of the request body for the /cognitiveservices/v1 endpoint. The SpeakRequest model has a single property called speak, which is an object that conforms to the SSML format.

Paths

The paths section of the Swagger definition describes the endpoints and methods used in the API. In this case, there is a single endpoint called /cognitiveservices/v1, which supports the POST method.

Request Body

The request body is defined in the requestBody section of the Swagger definition. The content property specifies the expected media type of the request body, which in this case is application/json. The schema property specifies the data model used for the request body, which is the SpeakRequest model defined earlier.

Responses

The responses section of the Swagger definition describes the possible responses that the API can return. In this case, there is a single response defined for the /cognitiveservices/v1 endpoint, which is a 200 OK response with an MP3 audio file in the audio/mp3 format.

API Request

Now that we've covered the Swagger definition, let's take a closer look at the actual API request that's being made. Here's what's happening in the API request:

Endpoint

The API request is being sent to the /cognitiveservices/v1 endpoint, which is the only endpoint defined in the Swagger definition.

Method

The API request is using the POST method, as defined in the Swagger definition.

Request Headers

The API request includes three headers:

Content-Type: Specifies the media type of the request body, which in this case is application/ssml+xml.

X-Microsoft-OutputFormat: Specifies the desired output format for the synthesized speech, which in this case is audio-16khz-128kbitrate-mono-mp3.

Authorization: Specifies the API key used to authenticate the request.

Request Body

The request body is an SSML document that specifies the language, gender, name, and text to synthesize into speech. The SSML document is formatted as XML, as required by the API.

Response

If the API request is successful, the API will return an MP3 audio file in the audio/mp3 format, as defined in the Swagger definition. The contents of the audio file are binary data that can be played back as synthesized speech.  
  
Explanination of code:

import React, { useState } from "react";

import axios from "axios";

const TextToSpeech = () => {

const [text, setText] = useState("");

const handleInputChange = (event) => {

setText(event.target.value);

};

const getTokenAsync = async (tokenEndpoint) => {

try {

const response = await fetch(tokenEndpoint, {

method: "POST",

headers: {

"Ocp-Apim-Subscription-Key": "0f9cd7ac7a9243868cf99a4a7cc1416c",

},

});

const token = await response.text();

return token;

} catch (error) {

console.error("Error in token request:", error);

return null;

}

};

const playAudioAsync = async (token, speechEndpoint) => {

const headers = {

"Content-Type": "application/ssml+xml",

// "User-Agent": "SI-text-to-speech-api-finalproject",

"X-Microsoft-OutputFormat": "audio-16khz-128kbitrate-mono-mp3",

Authorization: `Bearer ${token}`,

};

const body = `

<speak version='1.0' xml:lang='en-US'><voice xml:lang='en-US' xml:gender='Male'

name='en-US-ChristopherNeural'>

${text}

</voice></speak>

`;

try {

const response = await axios.post(speechEndpoint, body, {

headers,

responseType: "arraybuffer",

});

console.log("Audio Played");

const audioContext = new (window.AudioContext ||

window.webkitAudioContext)();

audioContext.decodeAudioData(response.data, (buffer) => {

const source = audioContext.createBufferSource();

source.buffer = buffer;

source.connect(audioContext.destination);

source.start();

});

} catch (error) {

console.error("Error in textToSpeech:", error);

}

};

const handlePlayAudio = async () => {

const tokenEndpoint =

"https://eastus.api.cognitive.microsoft.com/sts/v1.0/issuetoken";

const speechEndpoint =

"https://eastus.tts.speech.microsoft.com/cognitiveservices/v1";

// Get the token

const token = await getTokenAsync(tokenEndpoint);

if (!token) {

return;

}

console.log(token);

await playAudioAsync(token, speechEndpoint);

};

return (

<div>

<h1>Text-to-Speech</h1>

<textarea

value={text}

onChange={handleInputChange}

rows="5"

cols="50"

placeholder="Enter your text here..."

/>

<br />

<button onClick={handlePlayAudio}>Play Audio</button>

</div>

);

};

export default TextToSpeech;

useState Hook

The useState hook is used to define a state variable called text. This variable stores the text that the user inputs into the text area.

handleInputChange Function

The handleInputChange function is a callback that is called whenever the user inputs text into the text area. It updates the text state variable with the current value of the text area.

getTokenAsync Function

The getTokenAsync function is an asynchronous function that sends a request to the token endpoint to retrieve an access token. The function takes in the token endpoint URL and returns the token string if successful. The access token is required to authorize the text-to-speech API request.

playAudioAsync Function

The playAudioAsync function is an asynchronous function that sends a request to the text-to-speech API to synthesize speech from the input text. The function takes in the access token, speech endpoint URL, and the input text. It uses the axios library to send the API request with the appropriate headers and request body. If the request is successful, the function decodes the response data into an audio buffer and plays it back using the Web Audio API.

handlePlayAudio Function

The handlePlayAudio function is a callback that is called when the user clicks the "Play Audio" button. It retrieves the access token using the getTokenAsync function and calls the playAudioAsync function with the access token and the input text.

Rendering

The TextToSpeech component is a functional component that renders a text area for the user to input text, a button to initiate the text-to-speech synthesis, and an audio player to play back the synthesized speech. The handleInputChange function is called whenever the user inputs text into the text area, and the handlePlayAudio function is called when the user clicks the "Play Audio" button.  
  
References:

Microsoft Speech Service Text-to-Speech API

Web Audio API

React.js

useState hook

useEffect hook

axios library

js-yaml library

swagger-ui-react library

Swagger specification

Fetch API

CSS Styling in React

Below are the SS of the END point





 

 