Created by Dipti Sharabh Chaudhari on 19-09-2021

Text Translation API Documentation

Prerequisites

The reader needs to have:

- Node.js installed on your computer.
- A basic understanding of Node.js, javascript, and mysql.
- A code editor. I will use Visual studio code.
- A browser to test the application routes.

Initialize the application

We will initialize the application by running the following command:

```
>npm init
```

Installing the required dependencies

Next, we will install the required dependencies.

We need to install the `google-translate-api` API dependency for translating text supplied.

Run the following command:

```
>npm install google-translate –save
```

`Import the dependencies`

In the root folder of the application, create a new file named `index.js` and add the following:

const translate = require("translate-google");// import google translator

const Translations = require("../models/index");//import model for the database

const sequelize = require("../config/Dbconnections");// sequelize the database means the table is created automatically as per model into the database

Created by Dipti Sharabh Chaudhari on 19-09-2021

Get webpage Route:

This route gets the `index`

```
const express = require("express");

const route = express.Router();
const translateController = require("./controllers/index");

route.post("/translate", translateController.getTranslationResponse);

module.exports = route;
```

Translating the fetched text:

If the text is found in database :

```
sequelize
   .query(
     "CALL getTranslatedResponse(:fromLanguage ,:toLanguage, :text) ",
     {
       replacements: { fromLanguage: from, toLanguage: to, text: text },
     }
   )
```

here, getTranslatedResponse is a MSsql stored procedure fetches the result of the translated text.

If the text is new then store it into the database:(using Mssql stored procedure)

```
DBconnections.query(
        "CALL addTranslatedResponse(:fromLanguage, :toLanguage, :text )",
        {
          replacements: {
            fromLanguage: from,
            toLanguage: to,
            text: text,
            translatedText: data,
          },
        }
      )
```

here, addTranslatedResponse MSSql stored procedure is used to store the data.

Created by Dipti Sharabh Chaudhari on 19-09-2021

The procedure scripts used in this project:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `getTranslatedResponse`(
IN fromLanguage char(200),
IN toLanguage char(200),
IN text TEXT
)
BEGIN
SELECT *
FROM translation.translations
WHERE fromLanguage = fromLanguage
AND toLanguage = toLanguage
AND textContent = textContent;
END
--------------------------------------------------------------------------
CREATE DEFINER=`root`@`localhost` PROCEDURE `addTranslatedResponse`(
IN fromLanguage int,
 IN toLanguage varchar(20),
 IN text Varchar(20),
 IN translatedText Varchar(20)
)
BEGIN
        insert into Translators(fromLanguage, toLanguage, text, translatedText) values
(fromLanguage,toLanguage, text,translatedText);
END
-------------------------------------------------------------
```

Running Project:

To run this backend use following command:

>npm run start:dev

Test Cases

Sr No.	Test Case	Expected Result
1	At GUI level check the input text is not empty.	For empty field display error message.
2	At GUI level check the input text contains valid character	For spacial character like(&#)display error message that filed contains invalid characters.
3	Input text is present in database.	Fetch translated text from database and display to user.
4	Input text is not present in database	Fetch and save translated text using translate-google api in database with all languages and display translated text .

Created by Dipti Sharabh Chaudhari on 19-09-2021