**Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации**

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**Московский технический университет связи и информатики**

Кафедра «Информатики»

Практическая работа №2 по дисциплине ВвИТ

«Переводчик»

Выполнил студент группы БИН2003 Беляков А.Н.

Проверил: Егор Аршинов

Москва 2021

**1. Цель работы:**

Создать свой переводчик на Flask

**2. Выполнение работы:**

Основной код app.py:

# Importing the required libraries  
import requests  
import os  
import uuid  
import json  
from dotenv import load\_dotenv  
from flask import Flask, render\_template, request  
  
load\_dotenv()  
app = Flask(\_\_name\_\_)  
  
  
@app.route('/', methods=['GET'])  
def index():  
 return render\_template('index.html')  
  
  
# code  
@app.route('/', methods=['POST'])  
def index\_post():  
 # Read the values from the form  
 original\_text = request.form['text']  
 target\_language = request.form['language']  
  
 # Load the values from .env  
 key = os.environ['KEY']  
 endpoint = os.environ['ENDPOINT']  
 location = os.environ['LOCATION']  
  
 # Indicate that we want to translate and the API  
 # version (3.0) and the target language  
 path = '/translate?api-version=3.0'  
  
 # Add the target language parameter  
 target\_language\_parameter = '&to=' + target\_language  
  
 # Create the full URL  
 constructed\_url = endpoint + path + target\_language\_parameter  
  
 # Set up the header information, which includes our  
 # subscription key  
 headers = {  
 'Ocp-Apim-Subscription-Key': key,  
 'Ocp-Apim-Subscription-Region': location,  
 'Content-type': 'application/json',  
 'X-ClientTraceId': str(uuid.uuid4())  
 }  
  
 # Create the body of the request with the text to be  
 # translated  
 body = [{'text': original\_text}]  
  
 # Make the call using post  
 translator\_request = requests.post(  
 constructed\_url, headers=headers, json=body)  
  
 # Retrieve the JSON response  
 translator\_response = translator\_request.json()  
  
 # Retrieve the translation  
 translated\_text = translator\_response[0]['translations'][0]['text']  
  
 # Call render template, passing the translated text,  
 # original text, and target language to the template  
 return render\_template(  
 'results.html',  
 translated\_text=translated\_text,  
 original\_text=original\_text,  
 target\_language=target\_language  
 )

Также у нас есть код в папке templates, которые являются настройками нашего сайта:

<!DOCTYPE html>

<html lang="en">

<head>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"

  integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">

 <title>Text Translator</title>

</head>

<body>

 <div class="container">

  <h1>Test Translation service</h1>

  <div>Enter the text you wish to translate, choose the language, and click Translate button</div>

  <div>

   <form method="POST">

    <div class="form-group">

     <textarea name="text" cols="20" rows="10"

       class="form-control"></textarea>

    </div>

    <div class="form-group">

     <label for="language">Language:</label>

     <select name="language" class="form-control">

      <option value="en">English</option>

      <option value="it">Italian</option>

      <option value="ja">Japanese</option>

      <option value="ru">Russian</option>

      <option value="de">German</option>

     </select>

    </div>

    <div>

     <button type="submit" class="btn btn-success">Translate</button>

    </div>

   </form>

  </div>

 </div>

</body>

</html>

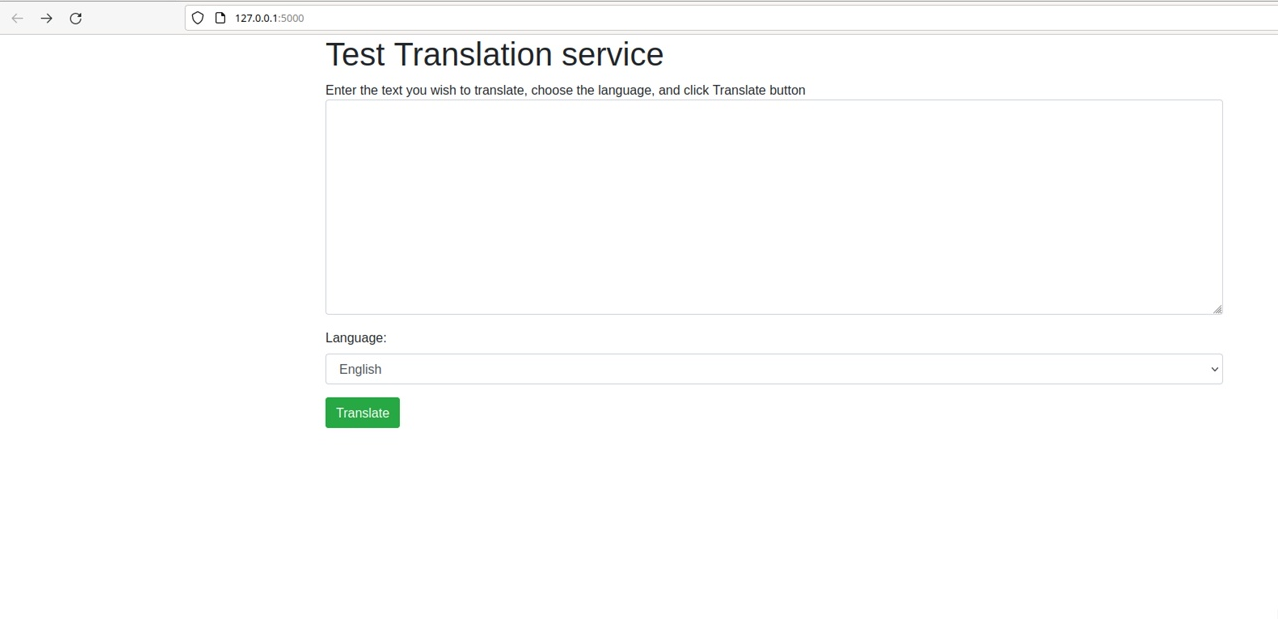
В итоге получаем:

Рисунок 1 – Сайт

**3. Вывод:**

В итоге мы достигли цели, переводчик запускается и способен переводить текст, однако ему нужно постоянное подключение к сервису Азур.