New Topic

Contents

[New Topic 1](#_Toc48928138)

# FLASK

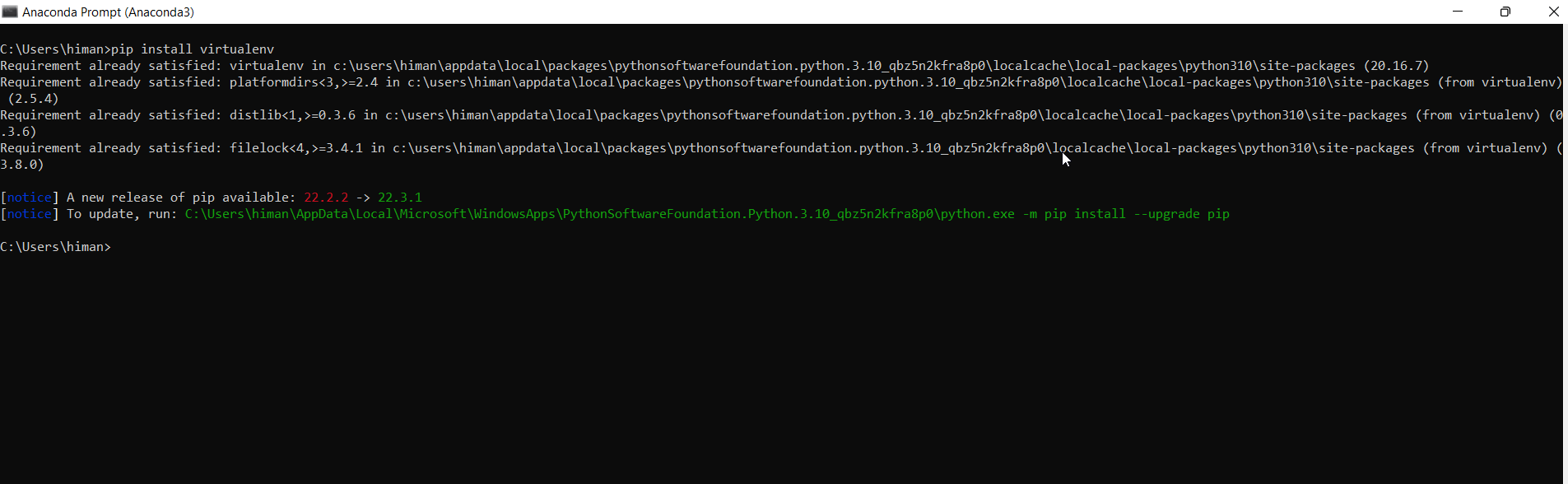
Flask is a web application framework written in Python.

## FLASK ENVIRONMENT

* Install Virtualenv for development environment

##### CODE:

pip install virtualenv



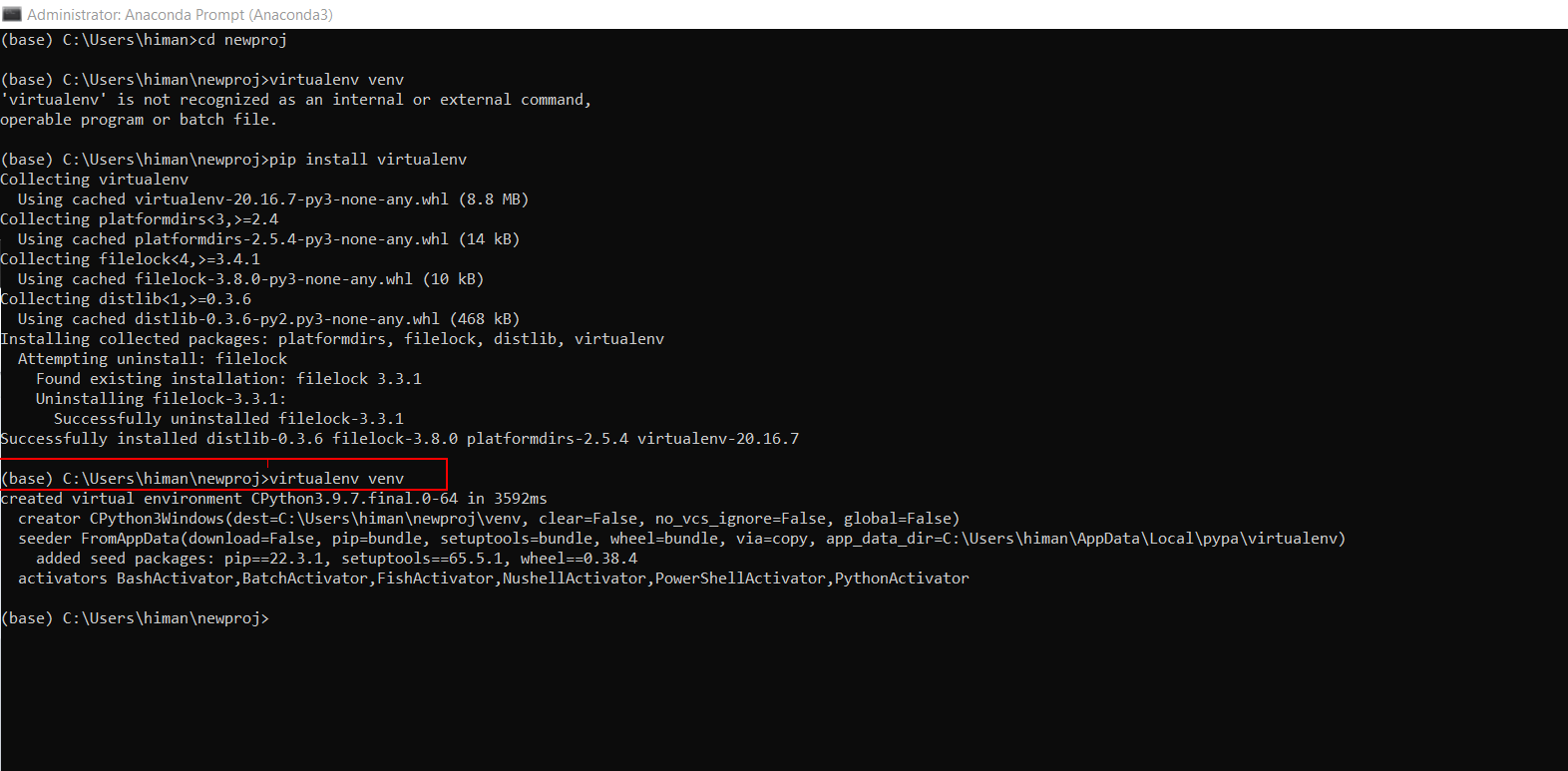
* Once installed, new virtual environment is created in a folder.

##### CODE:

mkdir newproj

cd newproj

virtualenv venv

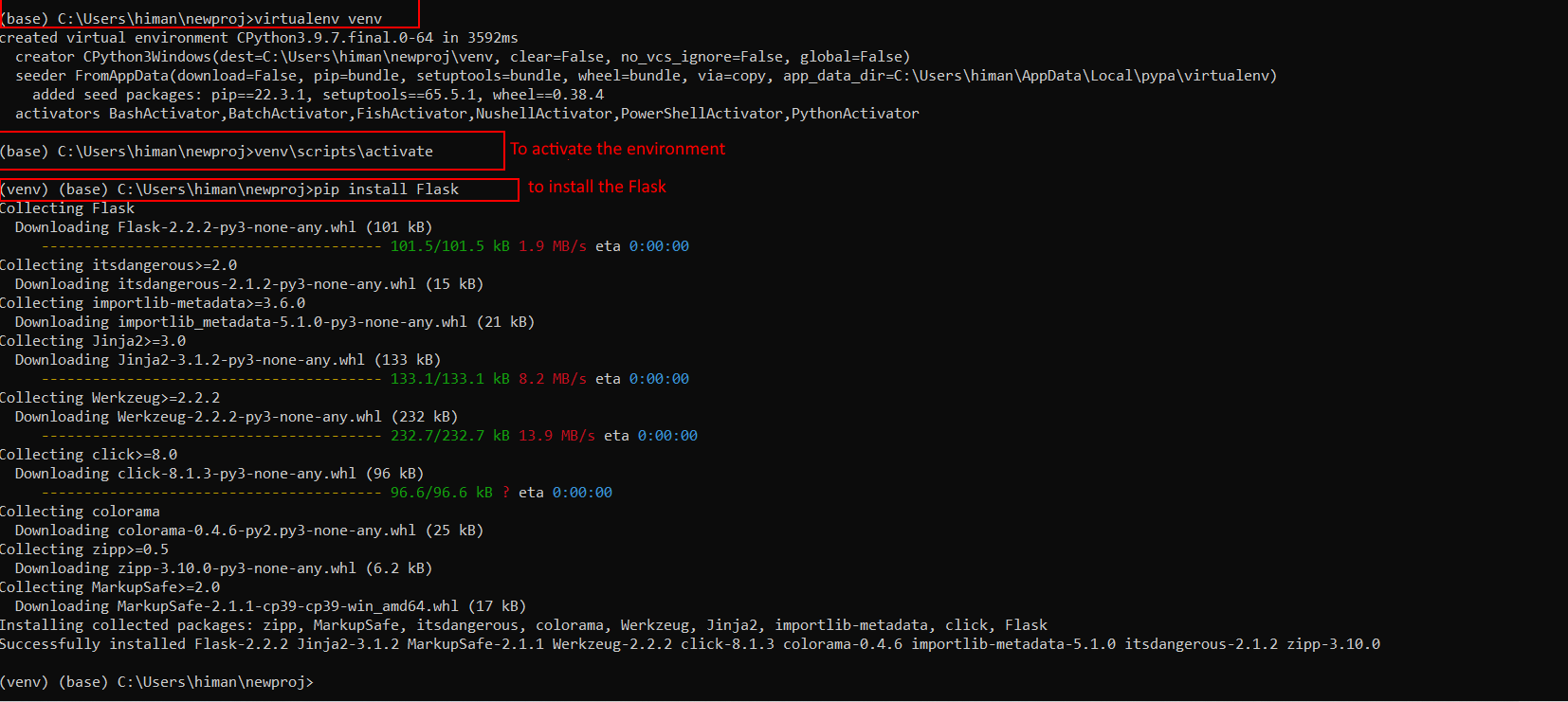


* Activate the environment and install the Flask framework

##### CODE:

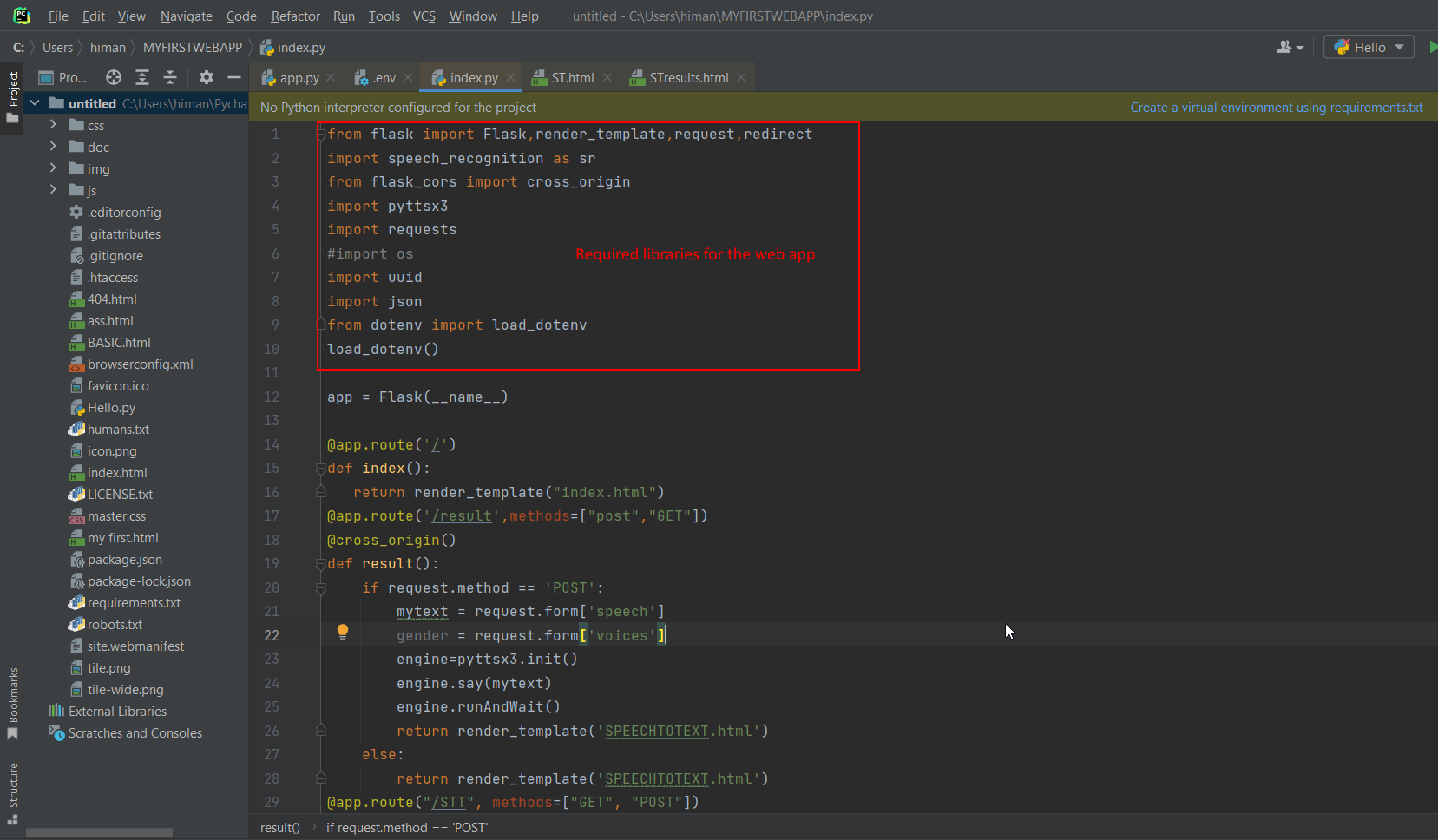
venv\scripts\activate

pip install Flask



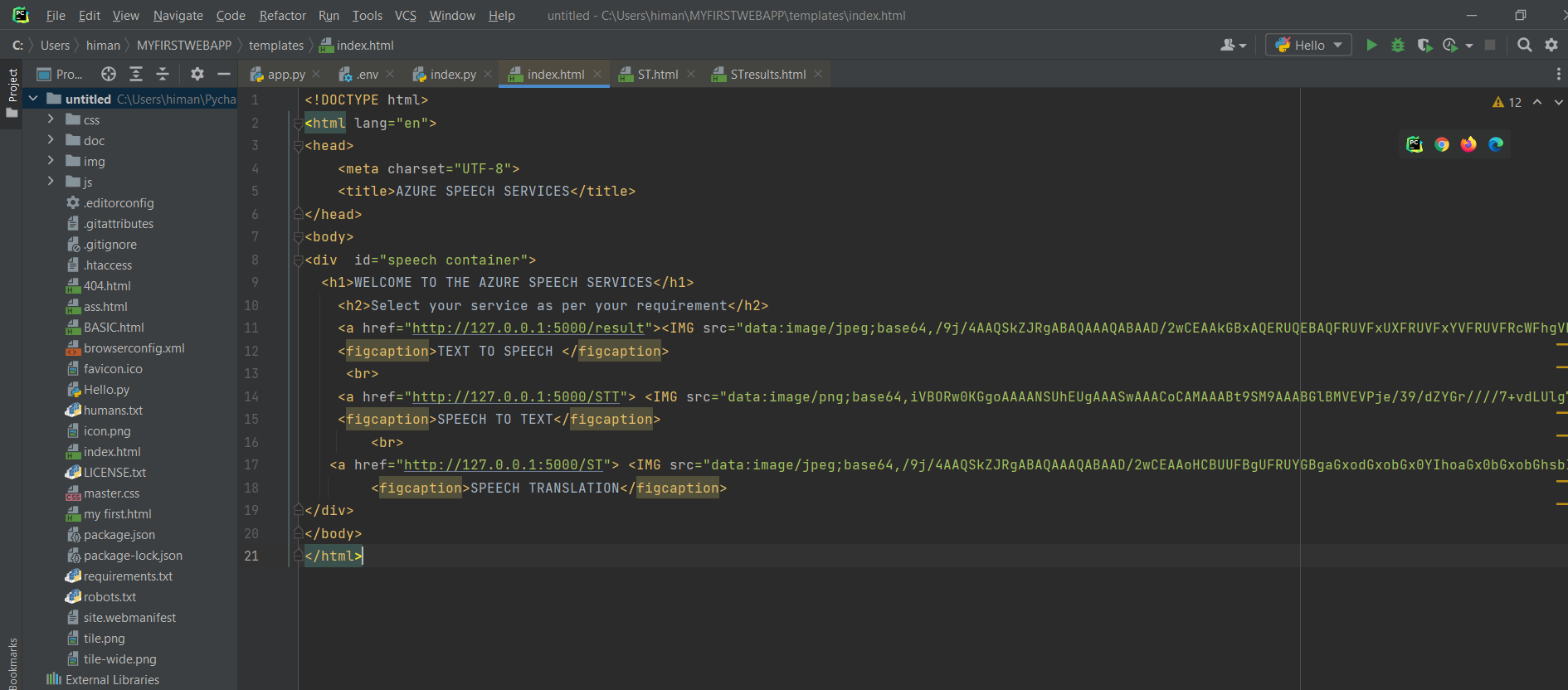
## CREATING PYTHON FILE (INDEX.PY)

from flask import Flask,render\_template,request,redirect  
import speech\_recognition as sr  
from flask\_cors import cross\_origin  
import pyttsx3  
import requests  
#import os  
import uuid  
import json  
from dotenv import load\_dotenv  
load\_dotenv()  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/')  
def index():  
 return render\_template("index.html")  
@app.route('/result',methods=["post","GET"])  
@cross\_origin()  
def result():  
 if request.method == 'POST':  
 mytext = request.form['speech']  
 gender = request.form['voices']  
 engine=pyttsx3.init()  
 engine.say(mytext)  
 engine.runAndWait()  
 return render\_template('SPEECHTOTEXT.html')  
 else:  
 return render\_template('SPEECHTOTEXT.html')  
@app.route("/STT", methods=["GET", "POST"])  
def STT():  
 transcript = ""  
 if request.method == "POST":  
 print("FORM DATA RECEIVED")  
  
 if "file" not in request.files:  
 return redirect(request.url)  
  
 file = request.files["file"]  
 if file.filename == "":  
 return redirect(request.url)  
  
 if file:  
 recognizer = sr.Recognizer()  
 audioFile = sr.AudioFile(file)  
 with audioFile as source:  
 data = recognizer.record(source)  
 transcript = recognizer.recognize\_google(data, key=None)  
  
 return render\_template('STT.html', transcript=transcript)  
  
@app.route('/ST', methods=['GET'])  
def ST():  
 return render\_template('ST.html')  
  
  
# code  
@app.route('/ST', methods=['POST'])  
def ST\_post():  
 # Read the values from the form  
 original\_text = request.form['text']  
 target\_language = request.form['language']  
  
 # Load the values from .env  
 key = "dc56b8aa751646459acd64805ed3a801"  
 endpoint = "https://api.cognitive.microsofttranslator.com/"  
 location = "centralindia"  
  
 # 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(  
 'STresults.html',  
 translated\_text=translated\_text,  
 original\_text=original\_text,  
 target\_language=target\_language  
 )  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug = True)



### CREATING HTML FILE (INDEX.HTML)

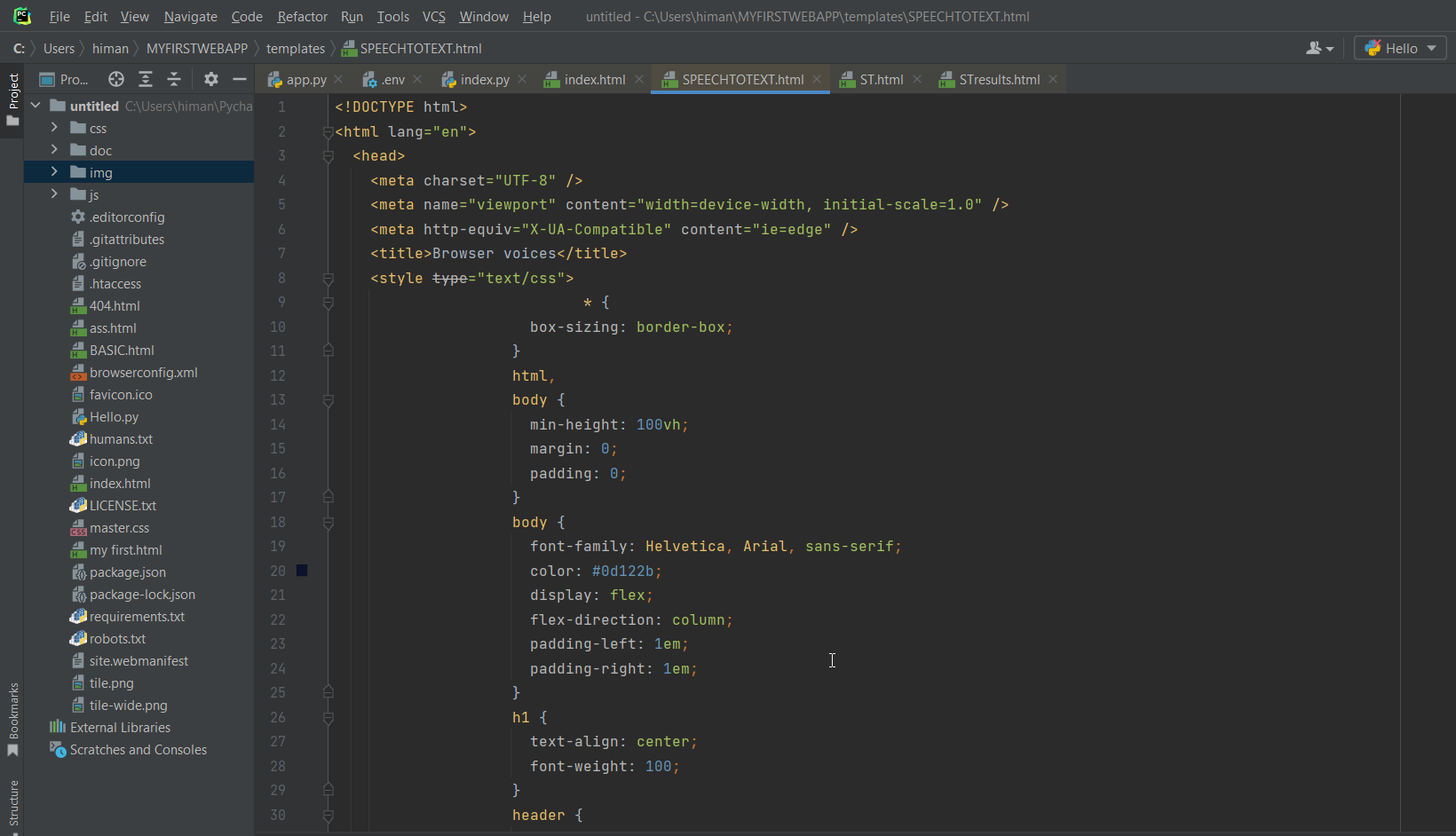
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>AZURE SPEECH SERVICES</title>  
</head>  
<body>  
<div id="speech container">  
 <h1>WELCOME TO THE AZURE SPEECH SERVICES</h1>  
 <h2>Select your service as per your requirement</h2>  
 <a href="http://127.0.0.1:5000/result"><IMG src="data:image/jpeg;base64," WIDTH="100" HEIGHT="100"></a>  
 <figcaption>TEXT TO SPEECH </figcaption>  
 <br>  
 <a href="http://127.0.0.1:5000/STT"> <IMG src="data:image/png;base64," WIDTH="100" HEIGHT="100"></a>  
 <figcaption>SPEECH TO TEXT</figcaption>  
 <br>  
 <a href="http://127.0.0.1:5000/ST"> <IMG src="data:image/jpeg;base64," WIDTH="100" HEIGHT="100"></a>  
 <figcaption>SPEECH TRANSLATION</figcaption>  
</div>  
</body>  
</html>



# CREATING FILES FOR TEXT TO SPEECH SERVICE

## CREATING HTML FILE(TEXT-TO-SPEECH.HTML)

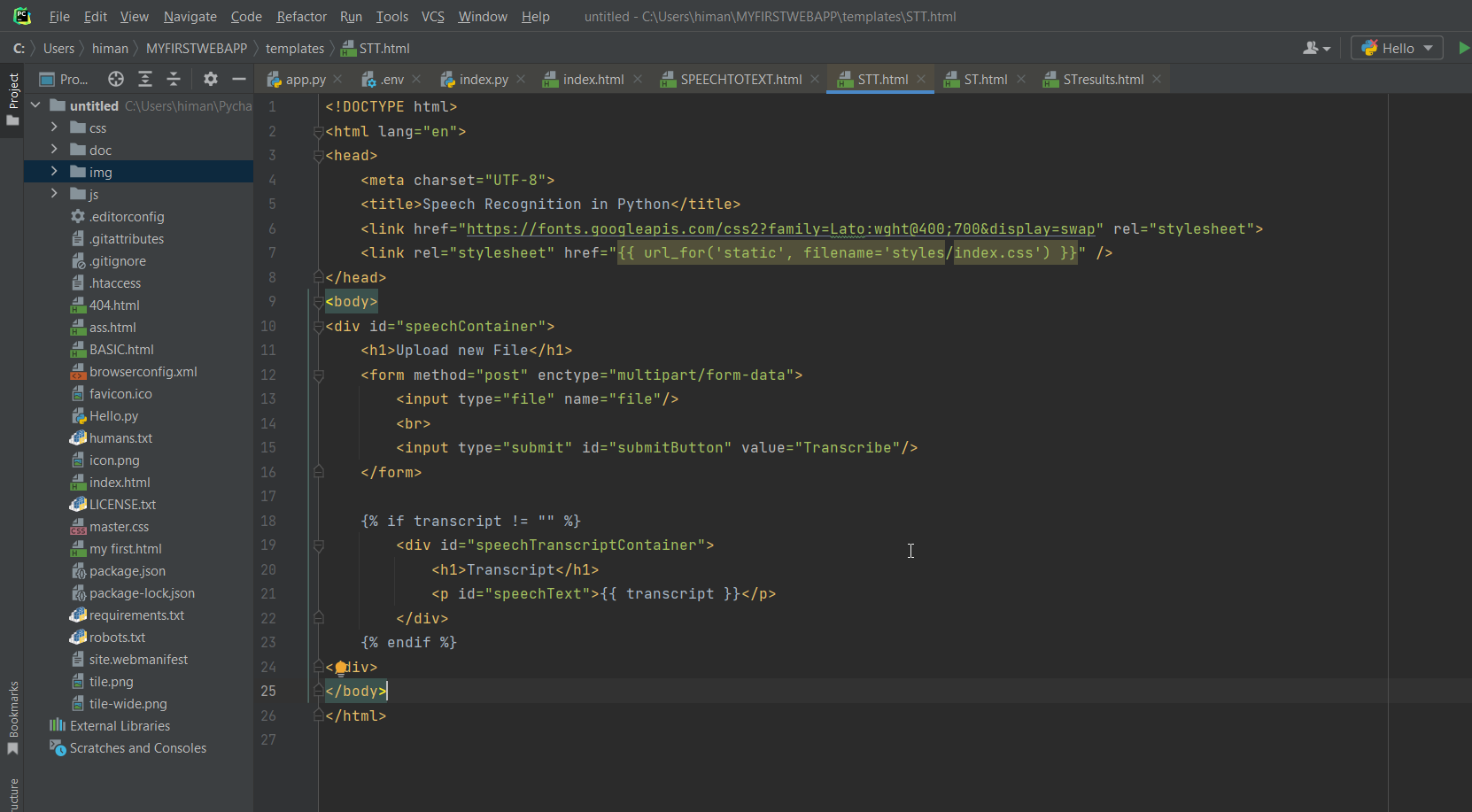
<!DOCTYPE html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8" />  
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
 <meta http-equiv="X-UA-Compatible" content="ie=edge" />  
 <title>Browser voices</title>  
 <style type="text/css">  
 \* {  
 box-sizing: border-box;  
 }  
 html,  
 body {  
 min-height: 100vh;  
 margin: 0;  
 padding: 0;  
 }  
 body {  
 font-family: Helvetica, Arial, sans-serif;  
 color: #0d122b;  
 display: flex;  
 flex-direction: column;  
 padding-left: 1em;  
 padding-right: 1em;  
 }  
 h1 {  
 text-align: center;  
 font-weight: 100;  
 }  
 header {  
 border-bottom: 1px solid #0d122b;  
 margin-bottom: 2em;  
 }  
 main {  
 flex-grow: 2;  
 display: flex;  
 justify-content: space-around;  
 align-items: center;  
 background-color: #fff;  
 border-radius: 12px;  
 margin-bottom: 2em;  
 }  
 @keyframes bg-pulse {  
 0% {  
 background-color: #fff;  
 }  
  
 50% {  
 background-color: #c7ecee;  
 }  
  
 100% {  
 backgrouond-color: #fff;  
 }  
 }  
 main.speaking {  
 animation: bg-pulse 1.5s alternate ease-in-out infinite;  
 }  
 .input {  
 text-align: center;  
 width: 100%;  
 }  
 label {  
 display: block;  
 font-size: 18px;  
 margin-bottom: 1em;  
 }  
 .field {  
 margin-bottom: 2em;  
 }  
 input {  
 font-size: 24px;  
 padding: 0.5em;  
 border: 1px solid rgba(13, 18, 43, 0.25);  
 border-radius: 6px;  
 width: 75%;  
 display: inline-block;  
 transition: border-color 0.25s;  
 text-align: center;  
 }  
 input:focus,  
 select:focus {  
 border-color: rgba(13, 18, 43, 1);  
 }  
 select {  
 width: 75%;  
 font-size: 24px;  
 padding: 0.5em;  
 border: 1px solid rgba(13, 18, 43, 0.25);  
 border-radius: 6px;  
 transition: border-color 0.25s;  
 }  
 button {  
 font-size: 18px;  
 font-weight: 200;  
 padding: 1em;  
 width: 200px;  
 background: transparent;  
 border: 4px solid #f22f46;  
 border-radius: 4px;  
 transition: all 0.4s ease 0s;  
 cursor: pointer;  
 color: #f22f46;  
 margin-bottom: 2em;  
 }  
 button:hover,  
 button:focus {  
 background: #f22f46;  
 color: #fff;  
 }  
  
 a {  
 color: #0d122b;  
 }  
 .error {  
 color: #f22f46;  
 text-align: center;  
 }  
 </style>  
 </head>  
 <body>  
  
 <header>  
 <h1>Text to Speech</h1>  
 </header>  
  
 <main>  
 <form class="input" id="voice-form" method="post">  
 <div class="field">  
 <label for="speech">Type Something</label>  
 <input type="text" name="speech" id="speech" required />  
 </div>  
 <div class="field">  
 <label for="voices">Choose a Voice Assistant</label>  
 <select name="voices" id="voices">  
 <option value="Male">Male</option>  
 </select>  
 </div>  
 <button>  
 Say it!  
 </button>  
 </form>  
 </main>  
 </body>  
</html>



# CREATING FILES FOR SPEECH TO TEXT SERVICE

## CREATING HTML FILE(STT.HTML)

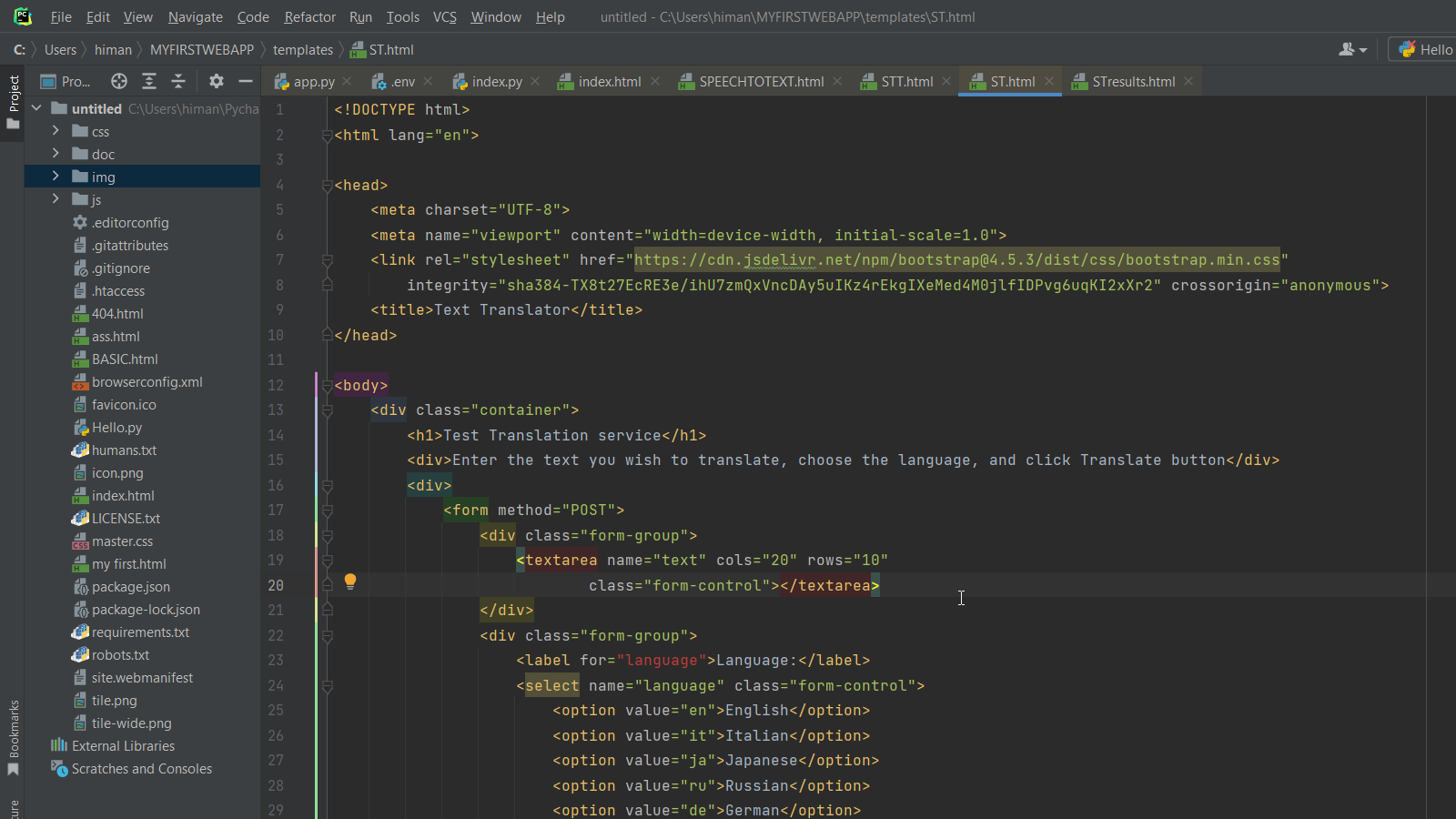
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Speech Recognition in Python</title>  
 <link href="https://fonts.googleapis.com/css2?family=Lato:wght@400;700&display=swap" rel="stylesheet">  
 <link rel="stylesheet" href="{{ url\_for('static', filename='styles/index.css') }}" />  
</head>  
<body>  
<div id="speechContainer">  
 <h1>Upload new File</h1>  
 <form method="post" enctype="multipart/form-data">  
 <input type="file" name="file"/>  
 <br>  
 <input type="submit" id="submitButton" value="Transcribe"/>  
 </form>  
  
 {% if transcript != "" %}  
 <div id="speechTranscriptContainer">  
 <h1>Transcript</h1>  
 <p id="speechText">{{ transcript }}</p>  
 </div>  
 {% endif %}  
</div>  
</body>  
</html>



# CREATING FILES FOR SPEECH TRANSLATION SERVICE

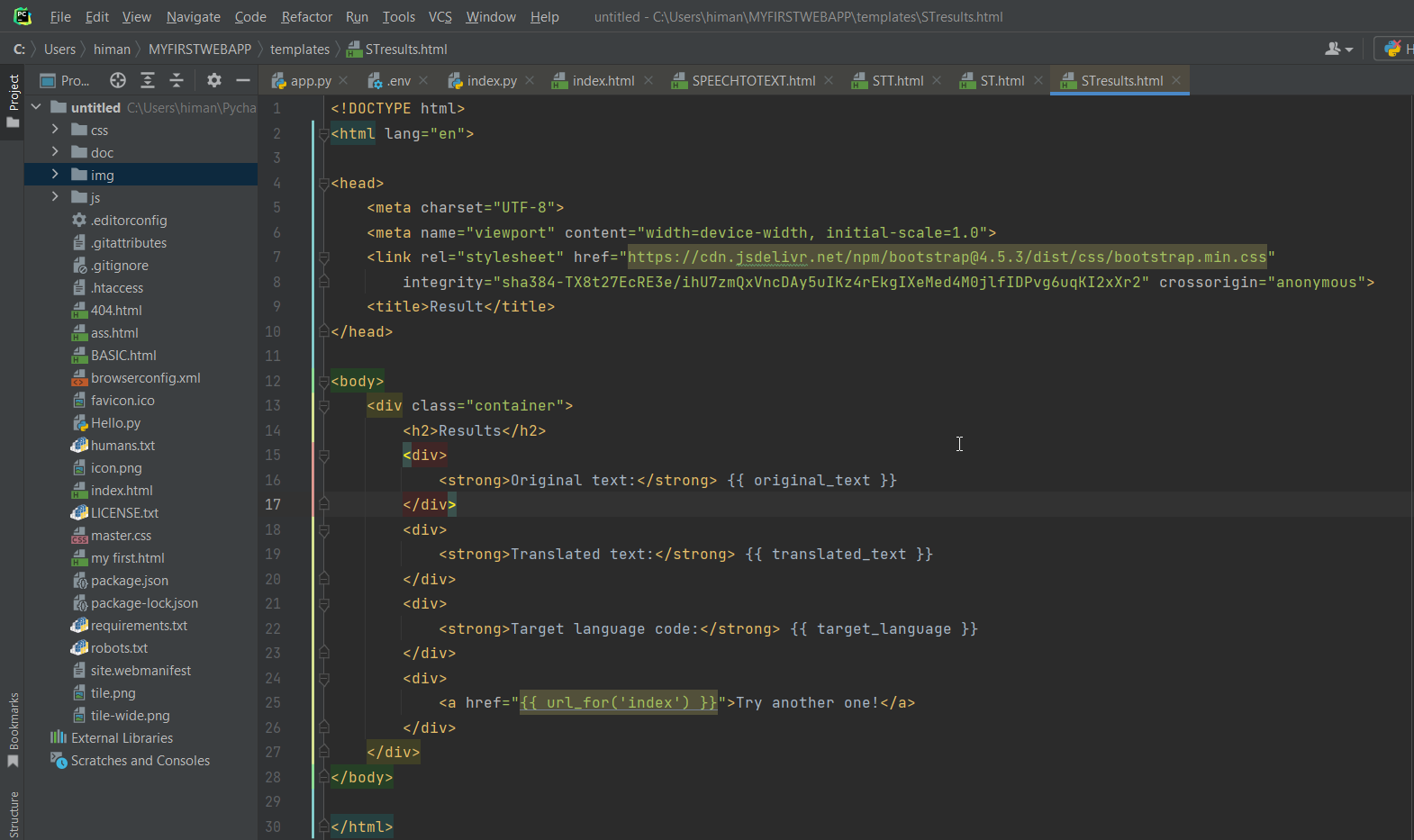
## CREATING HTML FILE(ST.HTML)

<!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>  
 <option value="hi">Hindi</option>  
 <option value="af">Afrikaans</option>  
 <option value="ga">Irish</option>  
 <option value="bn">Bengali</option>  
 <option value="es">Spanish</option>  
 <option value="ta">Tamil</option>  
 <option value="ur">Urdu</option>  
 </select>  
 </div>  
 <div>  
 <button type="submit" class="btn btn-success">Translate</button>  
 </div>  
 </form>  
 </div>  
 </div>  
</body>  
  
</html>



## CREATING HTML FILE(STRESULT.HTML)

<!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>Result</title>  
</head>  
  
<body>  
 <div class="container">  
 <h2>Results</h2>  
 <div>  
 <strong>Original text:</strong> {{ original\_text }}  
 </div>  
 <div>  
 <strong>Translated text:</strong> {{ translated\_text }}  
 </div>  
 <div>  
 <strong>Target language code:</strong> {{ target\_language }}  
 </div>  
 <div>  
 <a href="{{ url\_for('index') }}">Try another one!</a>  
 </div>  
 </div>  
</body>  
  
</html>

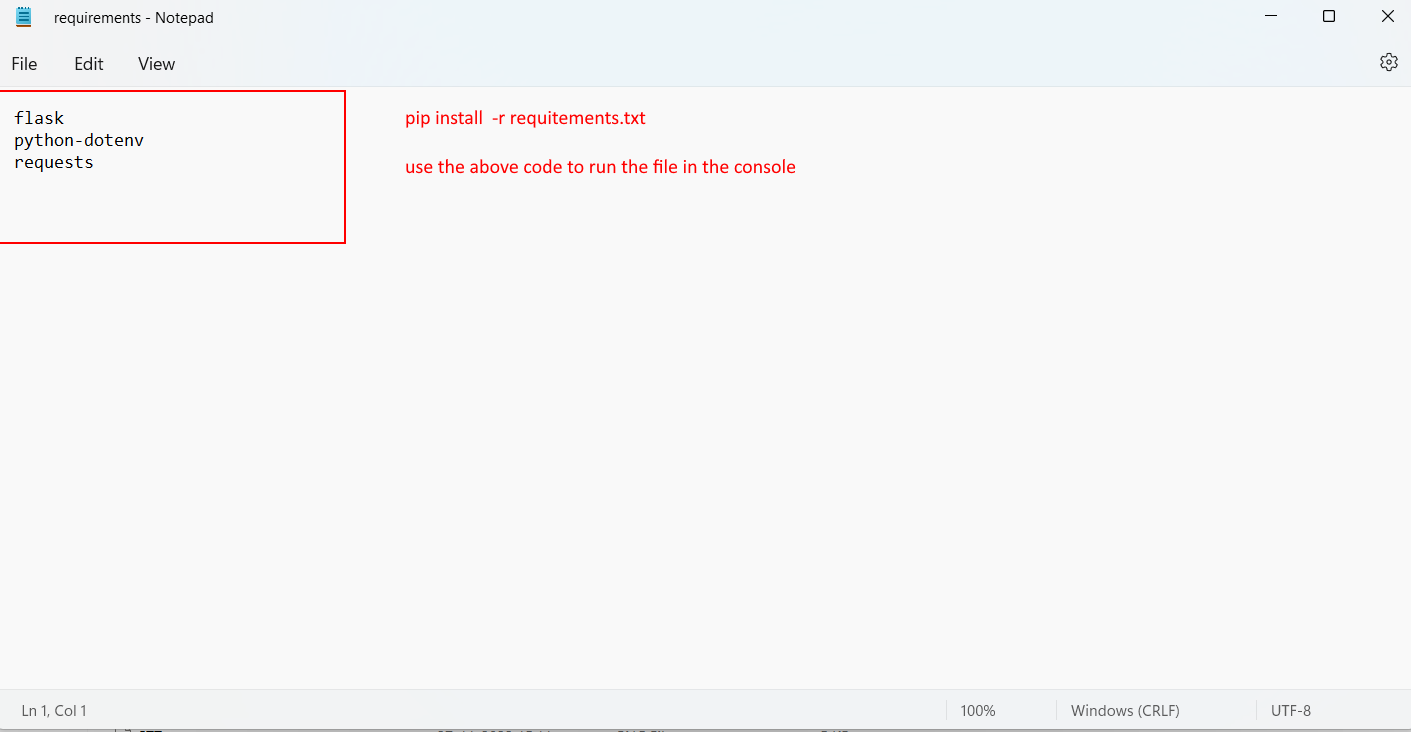


## CREATING TEXT FILE(REQUIRENEMTS.TXT)

Flask

python-dotenv

requests

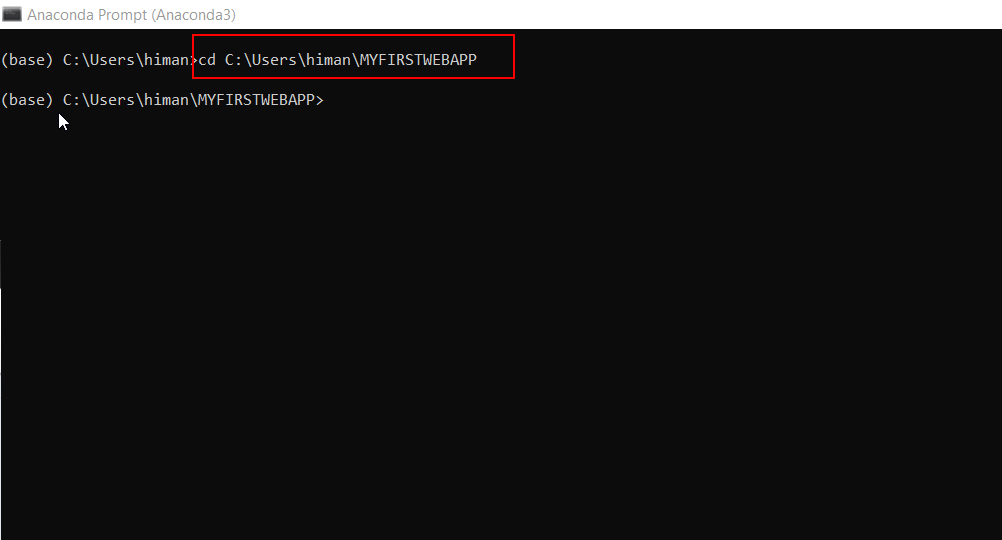


## RUNNING THE WEB APP USING FLASK

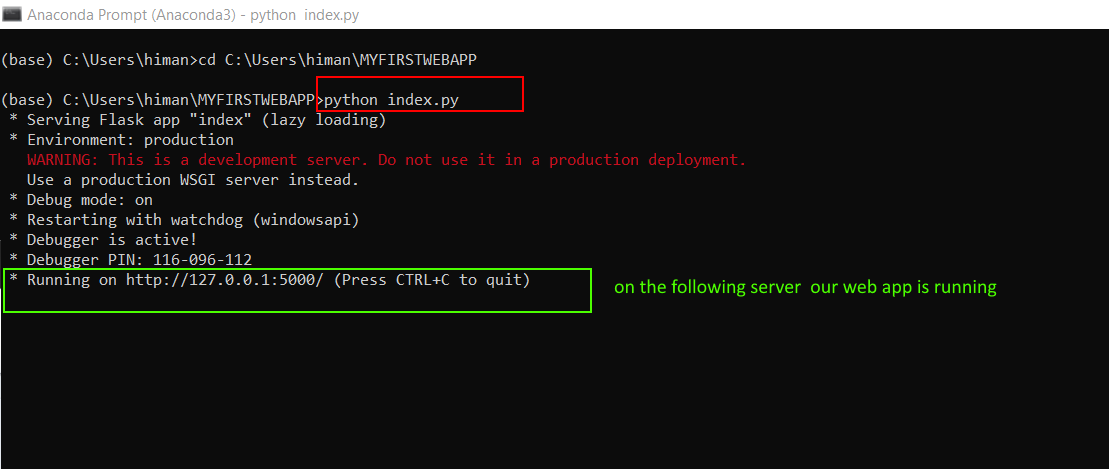
* Prerequisites

All the activity to install flask and setup environment has been done as steps shown above in the documents.

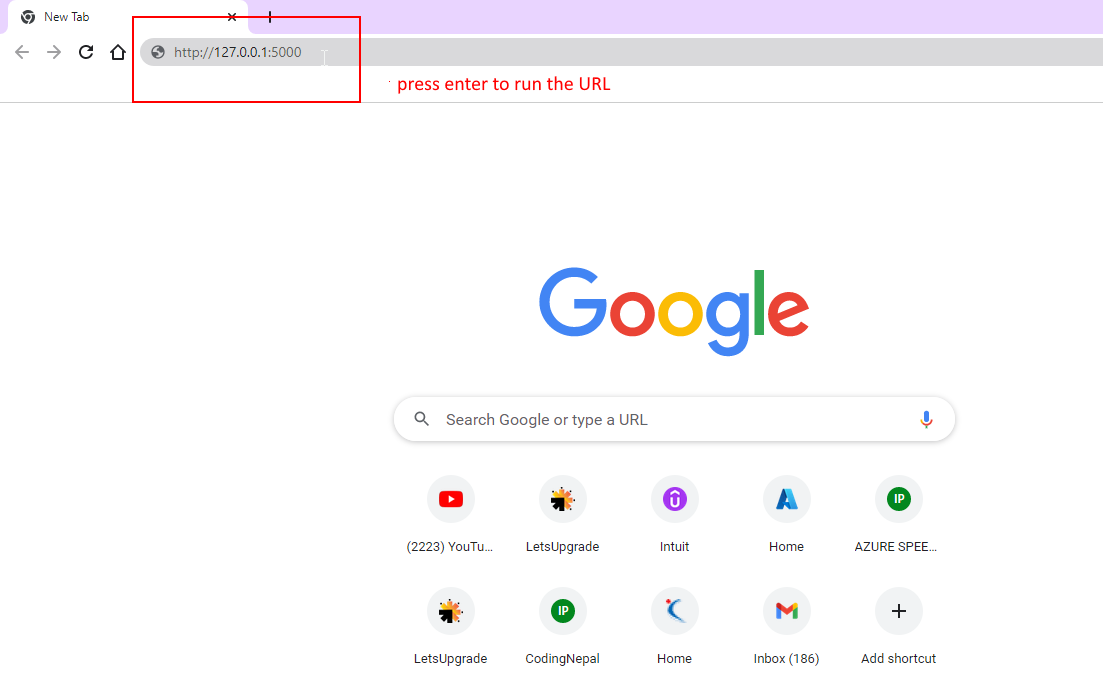
* Change directory of counsole where the file are present.



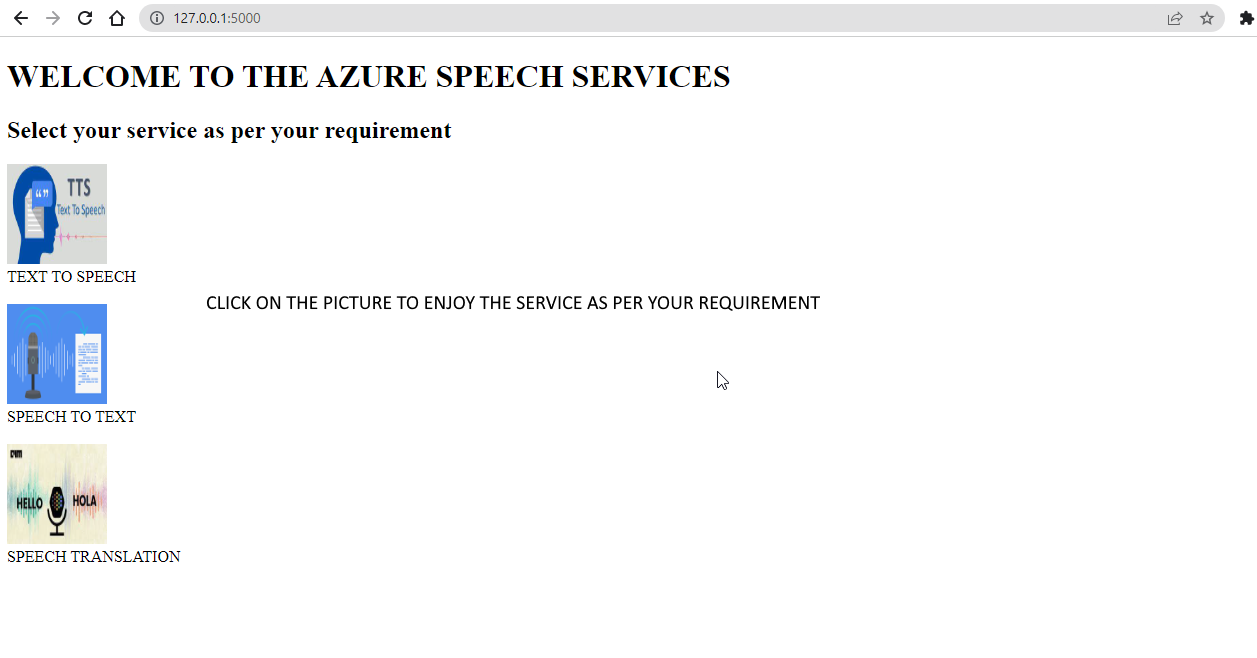
* Code to run the python file (index.py)on local host



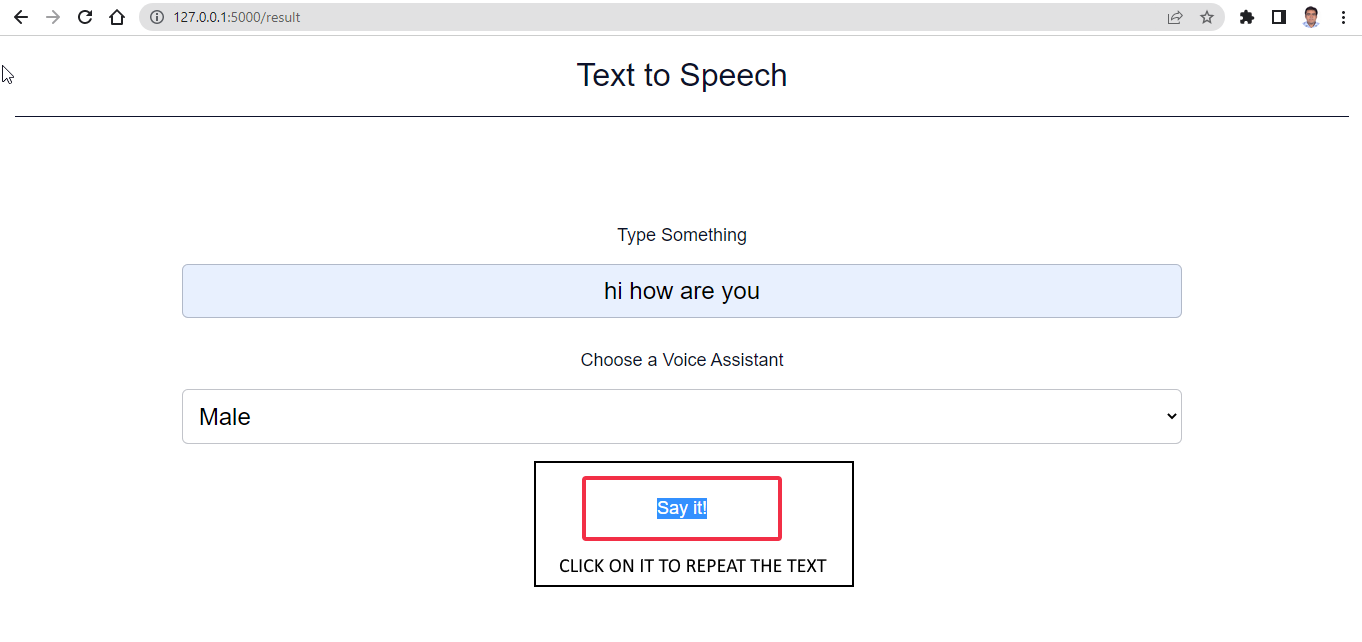
* Copy the following URL and run on your Google Chrome



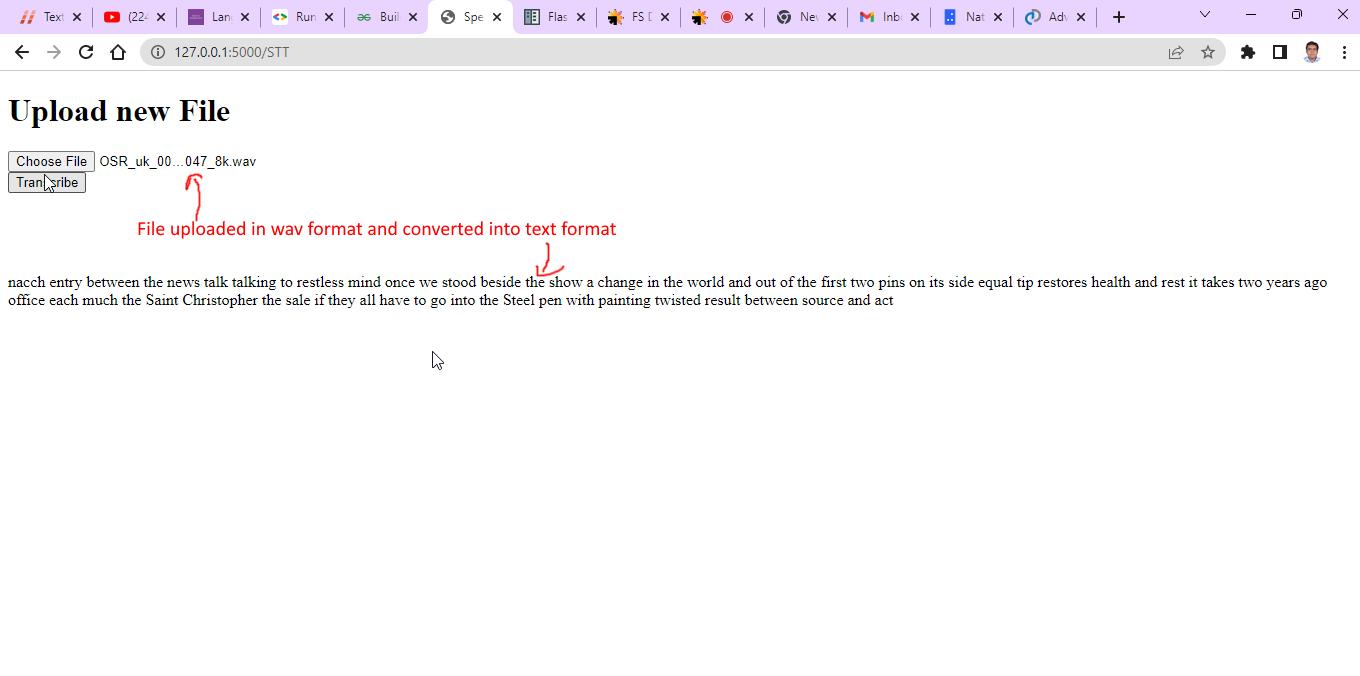
* After running the URL the following screen apperars enjoy the web app.



* After clicking on the Text to Speech pic



* After clicking on the Speech to Text pic



* After clicking on the Text Translation pic

