**Demonstration Guide**

(*LittleMorpheus*)

**Manual for using the LittleMorpheus Applicaiton**

**Prerequisite**

(1) Python 3.7 or higher

(2) Packages installation

(a) playsound – version 1.2.2

(b) SpeechRecognition – version 3.8.1

(c) Get Text to Speech – version 2.2.1

(d) pdfminer – version 20191125

(e) flask – version 1.1.2

(3) HTML5

(4) CSS5

(5) Python IDE – Spyder3 or higher

**Source Code**

***from flask import Flask***

***from flask import request***

***from flask import render\_template***

***import playsound***

***import os***

***import time***

***import speech\_recognition as sr***

***from gtts import gTTS***

***from pdfminer.pdfparser import PDFParser***

***from pdfminer.pdfdocument import PDFDocument***

***from pdfminer.pdfpage import PDFPage***

***from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter***

***from pdfminer.pdfdevice import PDFDevice***

***from pdfminer.pdfpage import PDFTextExtractionNotAllowed***

***from pdfminer.layout import LAParams, LTTextBox, LTTextLine***

***from pdfminer.converter import PDFPageAggregator***

***app = Flask(\_\_name\_\_)***

***@app.route('/')***

***def my\_form():***

***return render\_template("my-form.html")***

***@app.route('/', methods=['POST'])***

***def my\_form\_post():***

***bookName = request.form['bookname']***

***if not bookName :***

***return "<h1>not entered name</h1>"***

***else :***

***base\_path = "/home/natalia/Desktop/Python/PythonWEB"***

***my\_file = os.path.join(base\_path +"/"+ bookName)***

***log\_file = os.path.join(base\_path+"/"+"Book\_Name.txt")***

***password = ""***

***extracted\_text = ""***

***fp =open(my\_file,"rb")***

***parser =PDFParser(fp)***

***document = PDFDocument(parser,password)***

***rsrcmgr = PDFResourceManager()***

***laparams = LAParams()***

***device = PDFPageAggregator(rsrcmgr, laparams=laparams)***

***interpreter = PDFPageInterpreter(rsrcmgr, device)***

***for page in PDFPage.create\_pages(document):***

***interpreter.process\_page(page)***

***layout = device.get\_result()***

***for lt\_obj in layout:***

***if isinstance(lt\_obj, LTTextBox) or isinstance(lt\_obj, LTTextLine):***

***extracted\_text += lt\_obj.get\_text()***

***fp.close()***

***with open(log\_file, "wb") as my\_log:***

***my\_log.write(extracted\_text.encode("utf-8"))***

***str=open('/home/natalia/Desktop/Python/PythonWEB/Book\_Name.txt', 'r').read()***

***tts = gTTS(text=str, lang="en")***

***filename = "Audio.mp3"***

***tts.save(filename)***

***playsound.playsound(filename)***

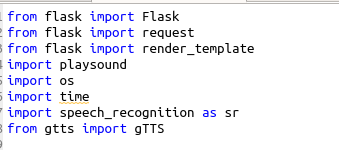
***return str***

***if \_\_name\_\_=='\_\_main\_\_':***

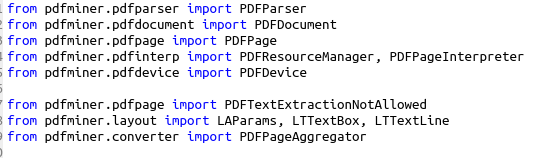
***app.run()***

**Code Explaination**

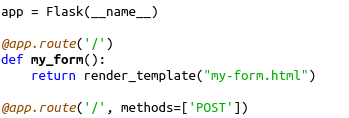
Here the explanation is given regarding how the code works and what it constitutes of? The below mentioned part is used to import the various packages, so that it functionality provided by it will be utilized later in the program.



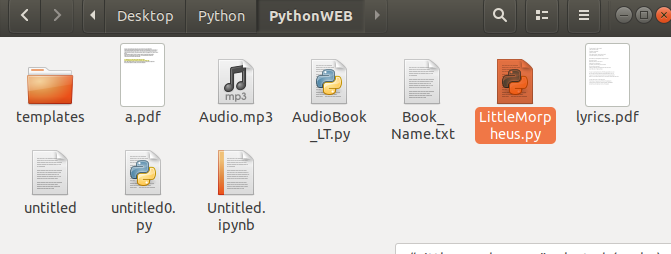
This is the Python package named ***pdfminer****,* which is used for operation over the pdf files. Here we primarily used the Text extraction feature from it.



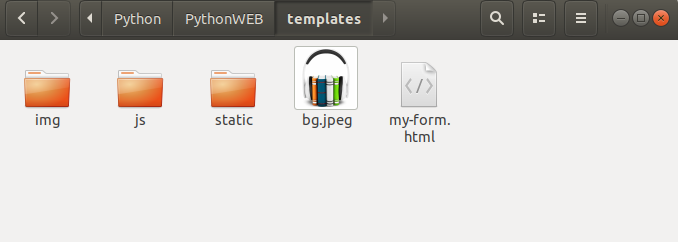
Here the Flask framework is defined under the code,



and in my case the folder was in this path, “***/home/natalia/Desktop/Python/PythonWEB***”. So currently the app route is inside this directory and there is a folder named ***templates****(shown in figure) below:*



Inside the template folders the files related to the web interface part is placed so that the app route will provide the path to the application to operate upon these files (shown below).

What the are the content or code of my-form.html file (which is the main file responsible for the conversion of text to audio) are as follows:

***<!DOCTYPE html>***

***<html lang="en">***

***<head>***

***<meta charset="utf-8">***

***<title>AudioBook\_KCGI@2021</title>***

***</head>***

***<body class="background" background-color="gray">***

***<h1>This will help you to make your AudioBook</h1>***

***<form action="." method="POST">***

***<p>Enter the name of book with extension</p>***

***<input type="text" name="bookname" value="">***

***<button type="submit" name="my-form" onclick="submit()" value="Check !">Click to Submit</button><br>***

***</form>***

***</body>***

***</html>***

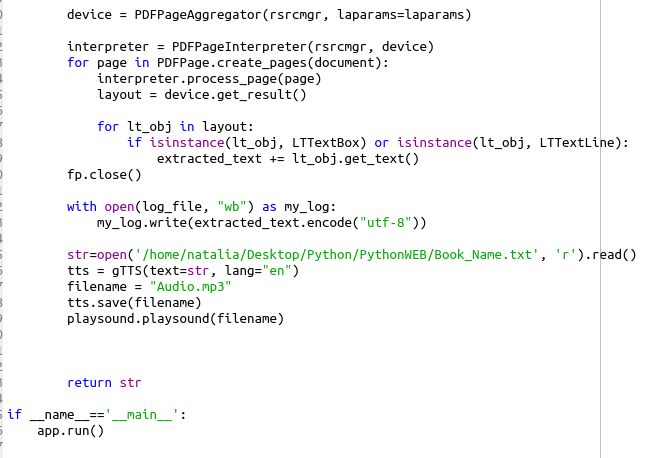
[***http://127.0.0.1:5000/***](http://127.0.0.1:5000/)

***\*Note :*** *we have used post method to get the reply the request from application and the site will be open on a local web-server (*[***http://127.0.0.1:5000/***](http://127.0.0.1:5000/)*) with a port number of 5000.*

***\*\*Limitation:*** *Right now we have tested it only on local host the configuration of this program is according to the local host. But in future if we will host it on internet then we have to make the changes according to that.*

We have defined a method named my\_form which will return the template named my-form.html as the value. Then there is another method named my\_form\_post , it will interact with user to provide the name of the pdf book with full extension.

We have used certain packages functionality to perform operation over the file saved in the directories. Which will save the output file as text file.

After extracting the text from text file then we used the read function and inputted that text to the playsound package to readout that text to us. Also we saved that file into mp3 format as an audio-book.

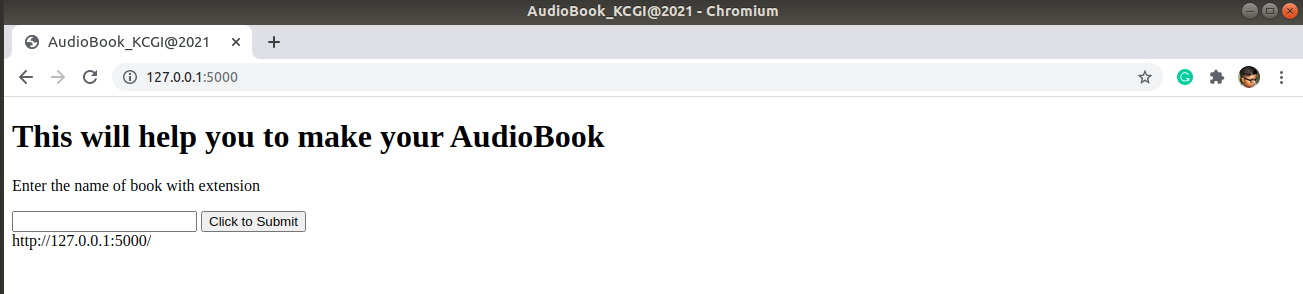
**HOW TO RUN THIS**

I used the spyder ide and pressed F5 to execute the code. The output will be given on the iPython Console (Which is shown as below).



So to run this interface I have to copy the http link (http://127.0.0.1:5000/ ) and paste it on chrome.

The interface will look like this:

and now in this after entering the book name which is test.pdf in this case. We can have our result

(demo video is provided with it). The link is here below:

<https://drive.google.com/file/d/1knNXKraFgc0DPnQLxO8AX9unvlh2uq5L/view?usp=sharing>

We also have another web-page (it was going to be the interface for the user which will access it through internet, thus it is still a work in progress), which is currently shows the text of the pdf on it and it will have an audio player embedded on it so that you can listen or even download (after some changes in coding, future work). The interface looks like this:

