SYMBIOSIS INSTITUTE OF TECHNOLOGY (SIT)
Constituent of Symbiosis International (Deemed University), Pune
(Established under Section 3 of the UGC Act of 1956 vide notification number F-9-12/2001-U-3 of the Government of india)
Re-Accredited by NAAC with 'A' Grade
॥वसुधैव कुटुम्बकम्॥

PROJECT REPORT
OF

AUDIOFY

AUDIOFY: APPLICATION TO CONVERT TEXT TO AUDIO

Bachelor of Technology
Computer Science and Technology

Submitted by:

MEGHANA MARINGANTY
18070122038
2018-2022

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

The purpose of this document is to present all the requirements, functions and applications of the project. All the requirements, right from technologies used to make the project to the limitations of the project, everything will be analyzed. The complete details of all the various functions and their functionalities of our project will be mentioned in detail.Also all the applications that the project offers are analyzed. Then finally the code snippets, output and test cases are mentioned to verify the working of the application.

1.2 Problem Statement

Ever since the lockdown, with almost everyone working from home or having online classes, their screen time is already too much, and if they want to read something to relax, they will have to again look in the phone in case they do not have the required hardcopies. There is an option for them to buy audiobooks, but they are very costly and not affordable to everyone. So this ends up in increasing the screen time of the normal person even more. So the problem statement I am trying to solve is tackling a way to decrease the screen time of a normal person in this pandemic.

1.3 Proposed Solution

The idea is to create a desktop application where the user can upload any pdf document or an image file and the text in that document or file will be extracted and converted into an audio file, which the user can listen to. So it is basically going to be a text to speech converting system. This application will also be supporting documents of different languages so as to cater to a larger audience. It is feasible to create this application as there are a lot of APIs and libraries that are available for free, which can be used for the purpose of extracting text from the uploaded document and then convert the extracted text into an audio file.

1.4 Motivation

Since the pandemic began, with the imposed lockdown everything has shifted from offline to online mode, right from attending regular school and college classes to even workshops, events and seminars being conducted online. This has resulted in an increase in the screentime of not only adults, but also of children by more than 70%. This increased screen time not only physically strains a person, but also has psychological effects on him. So to rewire after a long day if a person wishes to read books, because the deliveries had stopped, there wouldn't be any means for the person to obtain a hardcopy. So they will end up reading on an electronic device itself, increasing the screen time event more. As someone who also faced this problem of not having hardcopies, I tried to think of a solution by which I could not only read but without

increasing my screen time. This is when I thought of trying to develop an application in which if you upload the PDF/Image, it would extract the text, convert it to audio, and the user can listen to it.

1.5 Significance

The significance of this project is that it not only provides a solution to the increased screen time, but at the same time the solution caters to audiences from different regions as the documents of different languages are compatible to be converted to audio. It can also be used by the visually impaired people where someone uploads the document for them, and hence they can also listen to it. This will also help in increasing the curiosity in the people who have no habit of reading, and will make them more inclined to listen to a story, article or a piece of news for that matter.

1.6 Project Scope

This current application extracts the text from the document uploaded and converts that extracted text into audio by using the concepts of Optical Character Recognition and text to speech technologies. It is feasible to create this application as there are a lot of APIs and libraries that are available for free, which can be used for the purpose of extracting text from the uploaded document and then convert the extracted text into an audio file. In the future versions, the following additions can be made:

- More interactive and attractive Graphical User Interface.
- Inclusion of speech recognition, so that the user won't even have to touch the phone, and directly speak the document's name, from which subsequently text will be extracted and converted to audio. This will also be beneficial for the visually impaired.
- Allow a wider variety of document formats to be uploaded and converted like .txt, .docx , .rft, .odt etc.
- Include the option for the user to listen to the audio in male or female voice.

1.7 Beneficiaries

The project is intended to benefit:

- The children who do not read much and have a lot of screentime. The project aims to spark an interest in them to read, through the audiobooks.
- The adults who are very keen to read but do not have the required hardcopies of the same.
- The visually impaired, who with the help of another person who will upload the document for them, can listen to any article or book.
- In all, people from various regions and speaking various languages as it is compatible to process documents of other languages too.

1.8 System Purpose

1.8.1 Users

- The user will be able to upload an Image or a PDF document.
- The user has an option of selecting the language of the document, so that he can listen to documents written in different languages.
- The user can also select the speed at which he wants the audio to be played.
- All the documents converted to audio file will be stored.

1.8.2 Access

It is not necessary for the person to be online all the time, and also once a document is converted to audio, it will be stored like that and hence the user need not convert the document again and again but instead can directly use the audio file that was generated the first time it was created.

1.8.3 Responsibilities

- Allow the user to sign up or login.
- Allow the user to upload an image or PDF document.
- Allow the user to choose the language of the document.
- Allow the user to choose the speed of audio.
- Allow the user to access the converted audio at a time after the audio was first created.
- Allow the user to select from all the audio files he has converted till now.

2. OVERALL DESCRIPTION

2.1 Project Perspective

The project is a desktop application that allows the user to input a document which is an image file or a PDF , through which the text is extracted and this extracted text is converted to an audio file which the user can listen to. One of the main aims of this project is to help decrease the screen time of the user and it will be a better substitute for e-book in case the required hard copy is not present.

2.2 Hardware and Software Requirements

The project is a desktop application that is coded in Python language. For the implementation of this application, there are certain hardware and software requirements that are necessary:

- A laptop with 8GB ram or above.

- Python version 3.6 and above should be installed.
- A python IDE to run the code. I have used Spyder here.
- Tesseract, an Optical Character Recognition software should be installed.
- Notepad to store the user data in byte stream.
- Audio player to play the output audio.

2.3 Libraries, tools, modules and packages

- Tkinter

  It is a package that is used to create a Graphical User Interface in python.

  In this application, all the Graphical User Interface has been made using tkinter, which helps in navigating to different windows and gives a clear view to the user about the different options he can perform as well as any warnings in case invalid data is entered.

- Pickle

  It is a module that is used for serializing and deserializing objects in python. When we pickle an object, we convert it into a byte stream and store it in a file. To retrieve the data of this object, we unpickle the object, where this data is converted from byte stream to its original form.

  In this application, when the user is signing up, after his data is verified to be valid, the user object is converted to byte stream using the pickle.dump() method, which takes the object name and file name as parameters and this data is stored in the file. When a user attempts to login, all the data in the user file which is in the form of byte stream, is retrieved into a list from the file using pickle.load() method which takes filename as parameter. This method converts the byte stream to its original format, and then comparison of all user details takes place to check if the details entered by the person who wants to login are correct or not.

- Pdf2text

  It is a tool that converts a pdf into an image list.

  In this application, in case the user uploads a PDF, each page of the PDF is first converted into an image using the convert_from_path() function which takes the document path as parameter and then stores these images in the folder.

- Pytesseract

  It is an Optical Character Recognition tool for Python. It is a wrapper to Google's Tesseract-OCR engine that is mentioned in the software requirements. It is used to extract text from different images and also can extract text from all image types or formats.

In this application, during the conversion process it iterates through the images to extract text from each image and append it in a string. The way this works is the image path and language of the text (which user has already mentioned) in the image is given as parameters to pytesseract.image_to_string() function, which then uses these two parameters and extracts the text from the image and appends it to the string variable.

- gTTS

  It is a python library which is a tool to interface Google Translate's text-to-speech API. It converts the text that it receives into an audio file. It allows unlimited lengths of text to be read and also allows the user to manipulate the speed at which it text is read in the audio.

  In this application, during the conversion process, after the text is completely extracted it is given to the gTTS() function, which also takes the language input we had given, speed we had mentioned and two functions of preprocessing (which corrects the pronunciation of extracted text) and tokenizer(which checks for different punctuation marks present in the text to handle them accordingly) as parameters. The output of this is the audio file which is stored in the document folder and it starts playing.

2.4 Methodology Used

Software methodology indicates the approach one takes while developing the software. This approach depends on many factors such as requirements, size of the project and also the time we have to develop the project to name a few. The model that I think will be best suited for my project is the Incremental Model. The reasons I will explain below in detail, but to put it in brief, the project can be divided into different modules, such that each module will go through the phases of requirements, design, implementation and testing and every new module will add a functionality to the previous module. This process will continue until the complete system is achieved.

The advantages of using this model is that it is a flexible model and can incorporate changes, the errors can be recognized easily and it will be easier to test and debug. Also, this model is generally used when the software engineering team is not very skilled or trained, and so me being a student who is learning and exploring different things, this is a better fit.

The different modules in this project will be:

1. Conversion/Extraction of text from the PDF

   In the incremental model, the most important requirement is tackled first, and here in this project, the conversion or extraction of text from the PDF is the most crucial, because without this, there will not be anything to convert to audio.

2. Conversion of the extracted text to audio.

In this module, the extracted text is converted to audio and played such that a speech or voice can be heard reading the text. This stage adds a functionality to the previous stage and enables the user to convert and listen to the audio.

3. Graphical User Interface

In this module, the complete GUI will be created which will add the functionality which will help the user to navigate and use the system easily.

From the above reasons and modules we can summarize that as the first module is done, it will be tested and then we can work on the second module, and when that is developed, we test it individually as well as after combining it with the first module as this second stage will be adding functionality to the previous stage. And this will be continued till the final application is developed.

2.5 Limitations

- The desktop application will not work for file formats other than .pdf , .png and .jpg.
- The application will not work on the very older versions of python.
- The user will not have the option of choosing a male or a female voice to read in the audio.
- The application will not work for a multi language document.

2.6 Dependencies

- Python 3.6 and above must be running on the PC.
- All the required libraries must be installed before using the applications.

3. FUNCTIONAL REQUIREMENTS

3.1 User sign up and login

The desktop application allows the user to sign up or login. This module is of medium priority.

1. New user can create a new account by entering his details and signing up.
2. When a new user signs up, a new folder with the user's name will be created, which will have all the files uploaded by that user.
3. A recurring user does not have to sign up but can directly login by entering the required details.
4. The system authenticates the details entered by the user both during sign up as well as login to check whether they are right or wrong and in case the details entered are wrong, he will be given a warning indicating he needs to enter the details again.

5. On successful signing up, the user then has to login, and if that is also successful, the user is directed towards the main menu of the application.

3.2 Uploading the document.

The user will have to upload the document that he wants to convert to audio. This module is of medium priority.

1. It is necessary that the document uploaded must be an image or a PDF document only as other formats of files are not supported in the current version of the application. If any other format of document is uploaded, there will be an error message shown to the user.
2. The user has to choose the language of the document.
3. The user has to choose the speed of the audio.
4. In case an image file is uploaded, a new folder must be created which will store the image file.
5. In case a PDF is uploaded, each page of the PDF will be first converted into image format and a new folder must be created which will store all the individual pages of the PDF as image files.

3.3 Extracting text from the image file.

All the converted image files in the folder will be iterated through to extract the text. This module is of high priority.

1. It is very important that text is properly extracted from each of the image files present in the folder and appended in the right order.
2. The required parameter of language and image location should be set so that the text is properly and efficiently extracted from the image and discrepancies can be avoided.

3.4 Converting the extracted text to audio

All the extracted text has to be converted to an audio file and played for the user. This module is of high priority.

1. It is very important for the extracted text to properly be converted to audio which is a time taking process. The required parameters of language, text and speed should be properly set to avoid discrepancies.
2. Once the text is completely converted to audio, the audio must immediately begin playing.
3. The converted audio must be stored in the folder which has the image files so that the user can access it later in the future and he won't have to go through the process of converting everything again.

4. NON FUNCTIONAL REQUIREMENTS

4.1 Security

The major security is proper login in the account. For this purpose, a proper login mechanism should be used to avoid any form of hacking. Therefore, security is provided where it is necessary for the user to enter a password with the minimum requirements of having at least 8 characters, a capital letter and a number while signing up. Also, since pickles is used to store data, the data is converted into a byte stream which will be more undecipherable and hence more secure.

4.2 Performance

It is necessary that the complete process of uploading, text extraction and text conversion to audio is not a slow process so that user experience can be better. The response time should not be more than 20 seconds, but in case the document uploaded is a very large file, it might take up longer time.

4.3 Usability

It is necessary for the user to be able to easily navigate between the different pages of the application and this is possible through the graphical user interface which allows the user to navigate between different pages of login/sign up and main page. On successful sign up/login, the user is presented to the main page where a button facilitates the uploading of the document by the user. The user will also get to choose the language and speed of the document.Once these are done, the user needs to click the convert button which will start the further process. He can also choose from his library previously converted audio files and click on play button to play the audio.

4.4 Maintainability

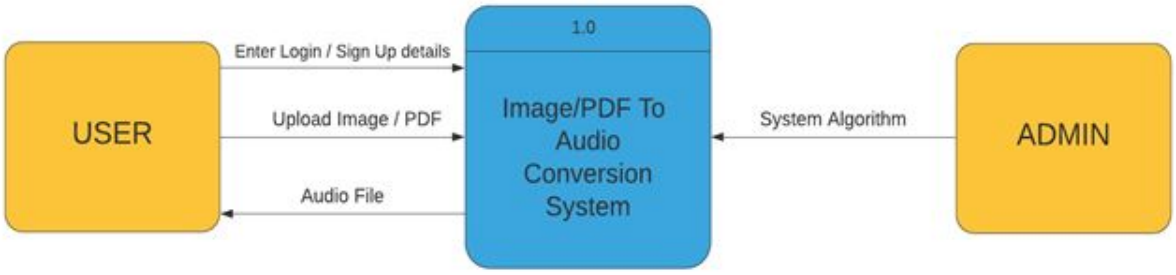
It is important to update the tools used in the software from time to time so that the performance also increases and is more efficient, the coding and technical standards should also be updated.

4.5 Availability

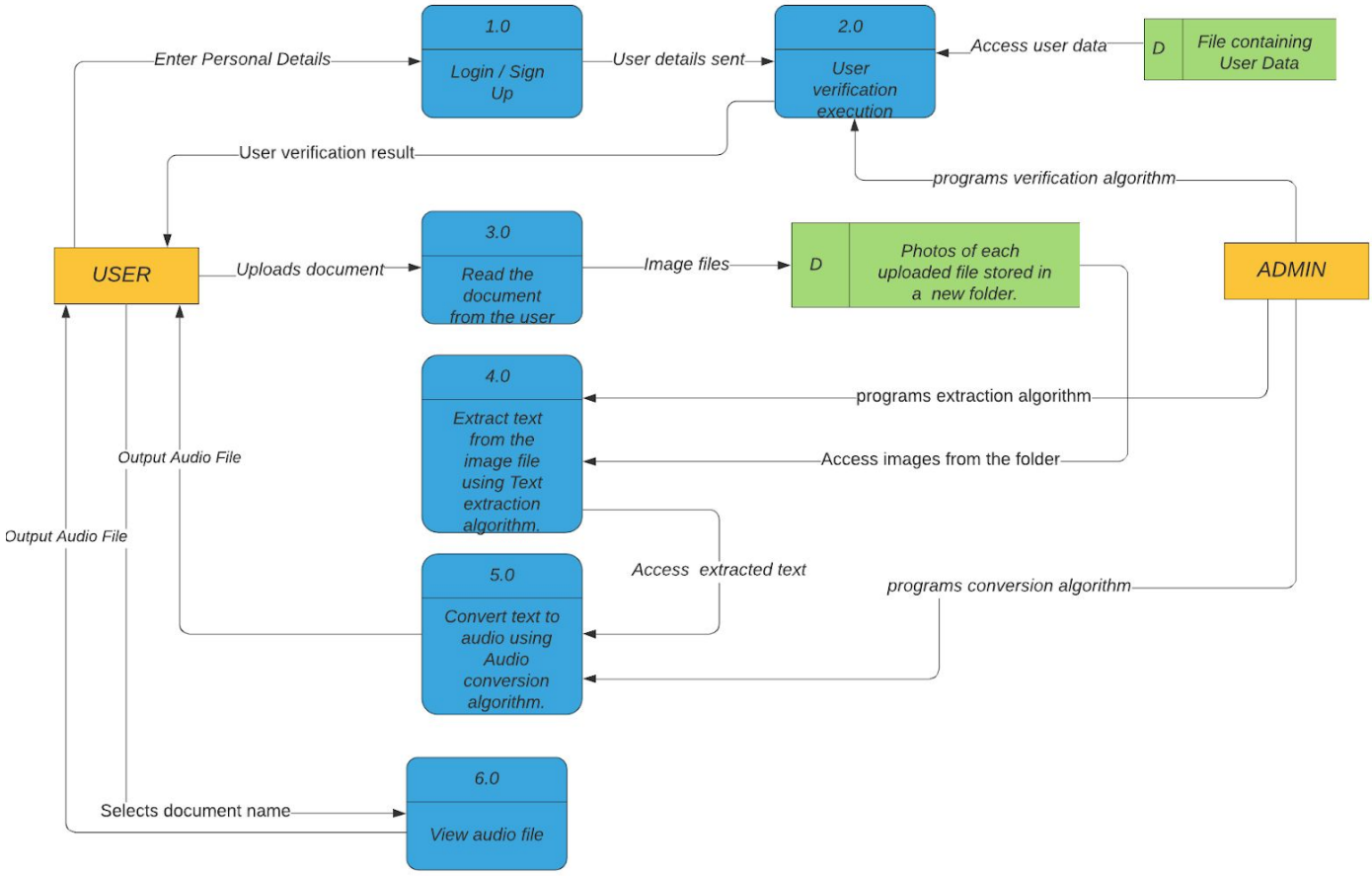
The application can be used at any time and there is no constraint on when and where it can be used.

5. DATA FLOW DIAGRAMS

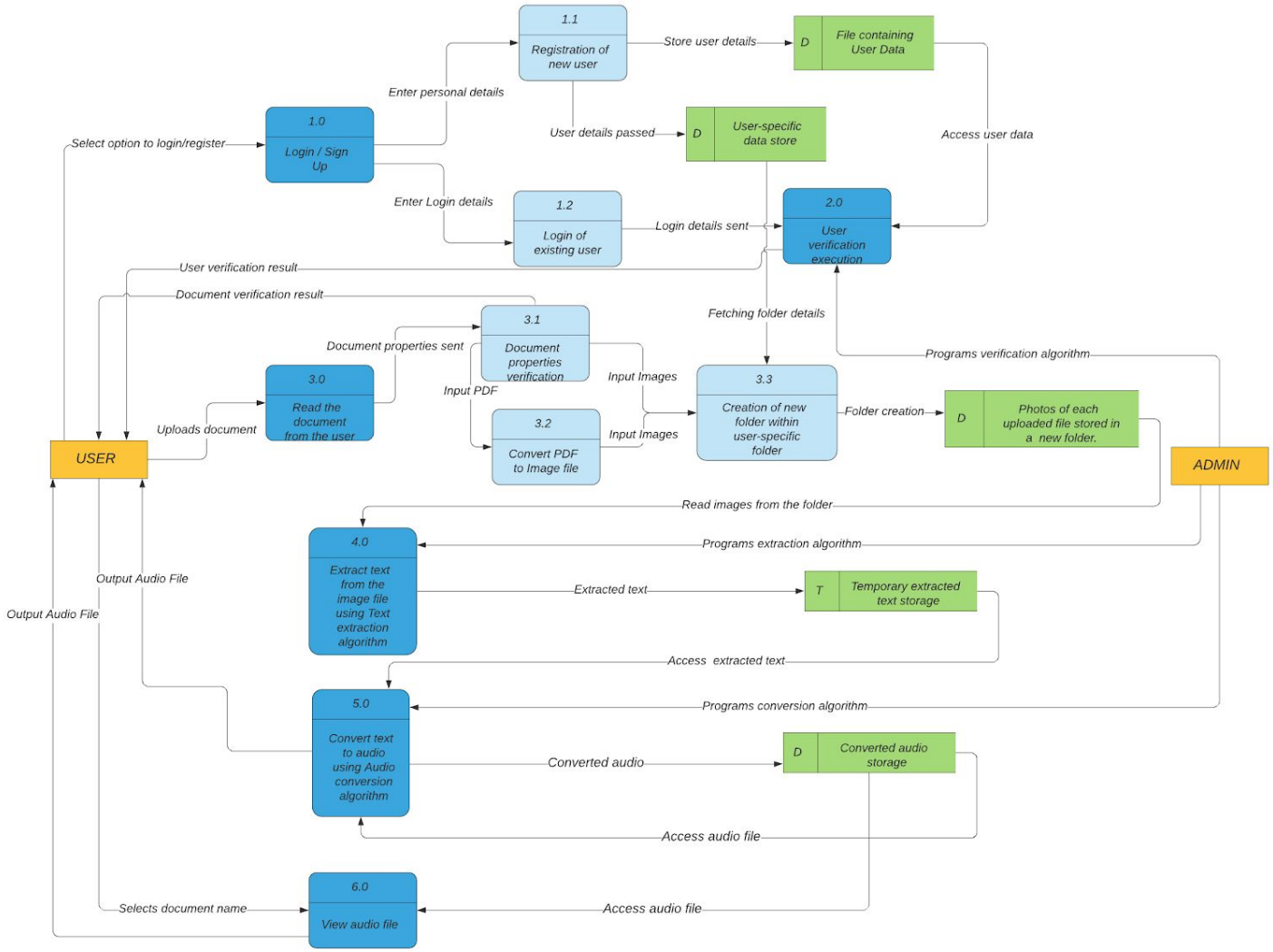
5.1 DFD Level 0



5.2 DFD Level 1

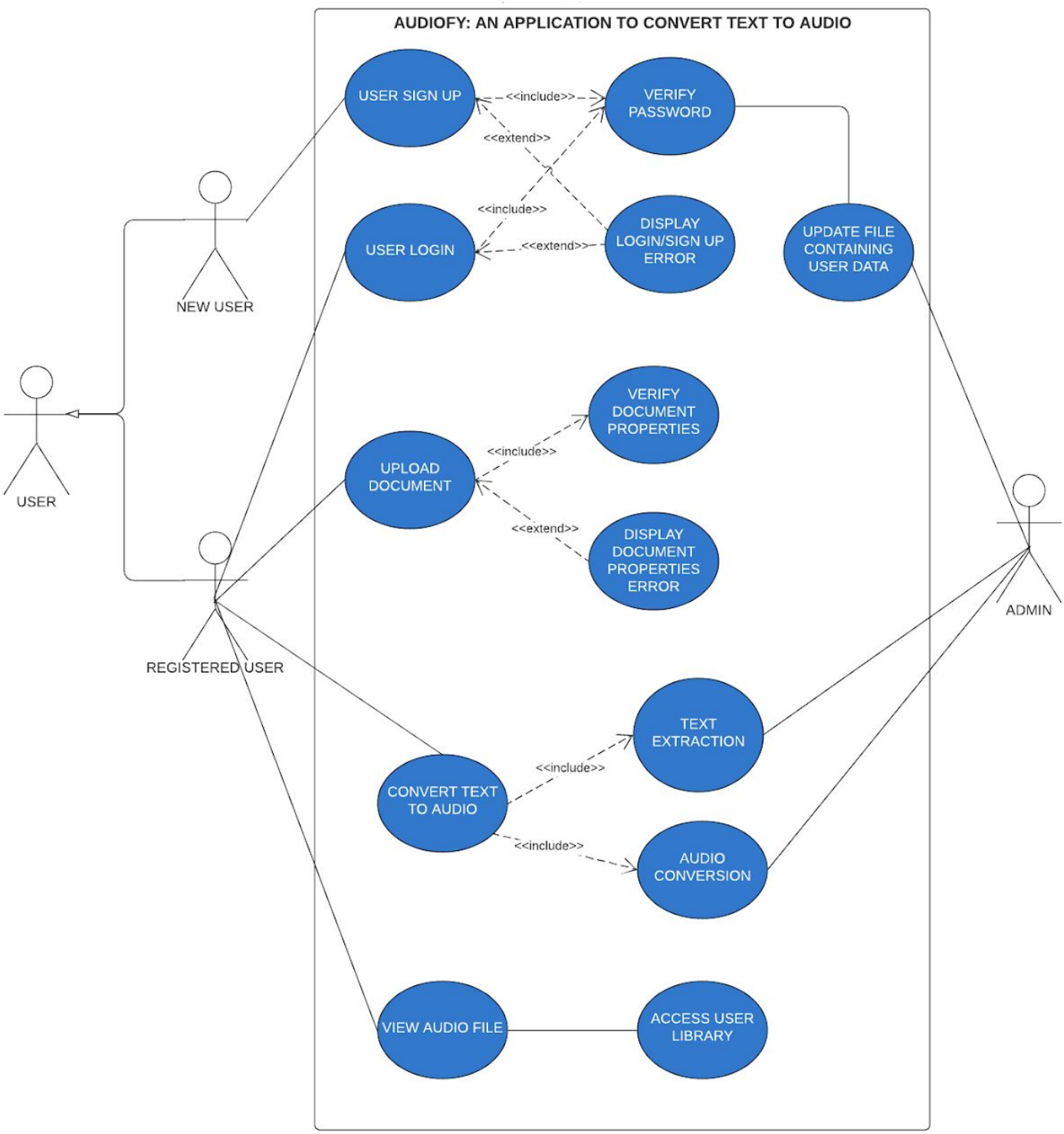


5.3 DFD Level 2



6. USE CASE

6.1 Use Case Diagram



6.2 Use Case Description (for selected cases)

User Sign Up

Use Case Name:	User Sign Up
Summary:	A new user must create an account, so that he can use this application.
Basic Flow:	1. The use case starts when a user indicates that he wants to sign up. 2. The system requests to enter full name, username and password.. 3. The user enters the full name, username and password. 4. User is asked to enter the password again to confirm it. 5. If both passwords match, the system checks if the full name duplicates any existing signed up full names. 6. A new folder is created in the computer with the user's full name. 7. The user account is created and the user database is updated.
Alternative flow:	Step 5: If both the passwords do not match, a dialog box showing an error message is displayed indicating that the user needs to enter the details again. Step 6: If full name already exists in the database, the new folder is created with the full name along with an integer, so as to differentiate from other users of the same name.
Extension points:	None
Preconditions:	None
Postconditions:	A message indicating successful sign up is displayed.
Business rules:	The data entered by the user will be kept confidential.

User Login

Use Case Name:	User Login
Summary:	In order to use the application, the user must login.
Basic Flow:	1. The use case starts when a user indicates that he wants to login. 2. The system requests username and password. 3. The user enters username and password. 4. The system verifies the username and password against all registered users information present in the user database. 5. If it is successful, it will direct him to the main page.

Alternative flow:	Step 4: If the username or password is invalid, a dialog box showing an error message is displayed indicating that the user needs to enter the details again.
Extension points:	none
Preconditions:	The user should have already signed up before.
Postconditions:	A message indicating successful sign up is displayed and the user can now view the main page.
Business rules:	The data entered by the user will be kept confidential.

Upload Document

Use Case Name:	Upload Document
Summary:	For the process to get started, a document must be uploaded by the user.
Basic Flow:	1. The use case starts when a user indicates that he wants to upload a document. 2. The user selects the document and uploads it. 3. It checks if the uploaded document is an image or a PDF. 4. It checks if the uploaded document size is less than 20MB. 5. The user selects the language of the document. 6. The user selects the speed of the audio he desires.
Alternative flow:	Step 3: If the uploaded document is of format other than that of image or PDF, a dialog box showing an error message is displayed indicating that the uploaded document format is not correct and he needs to upload again. Step 4: If the uploaded document has size greater than 20MB, a dialog box showing an error message is displayed indicating that the user needs to upload a document of appropriate size.
Extension points:	None
Preconditions:	The user needs to be logged in.
Postconditions:	The user can now click on convert to convert the text to audio.

Convert text to audio

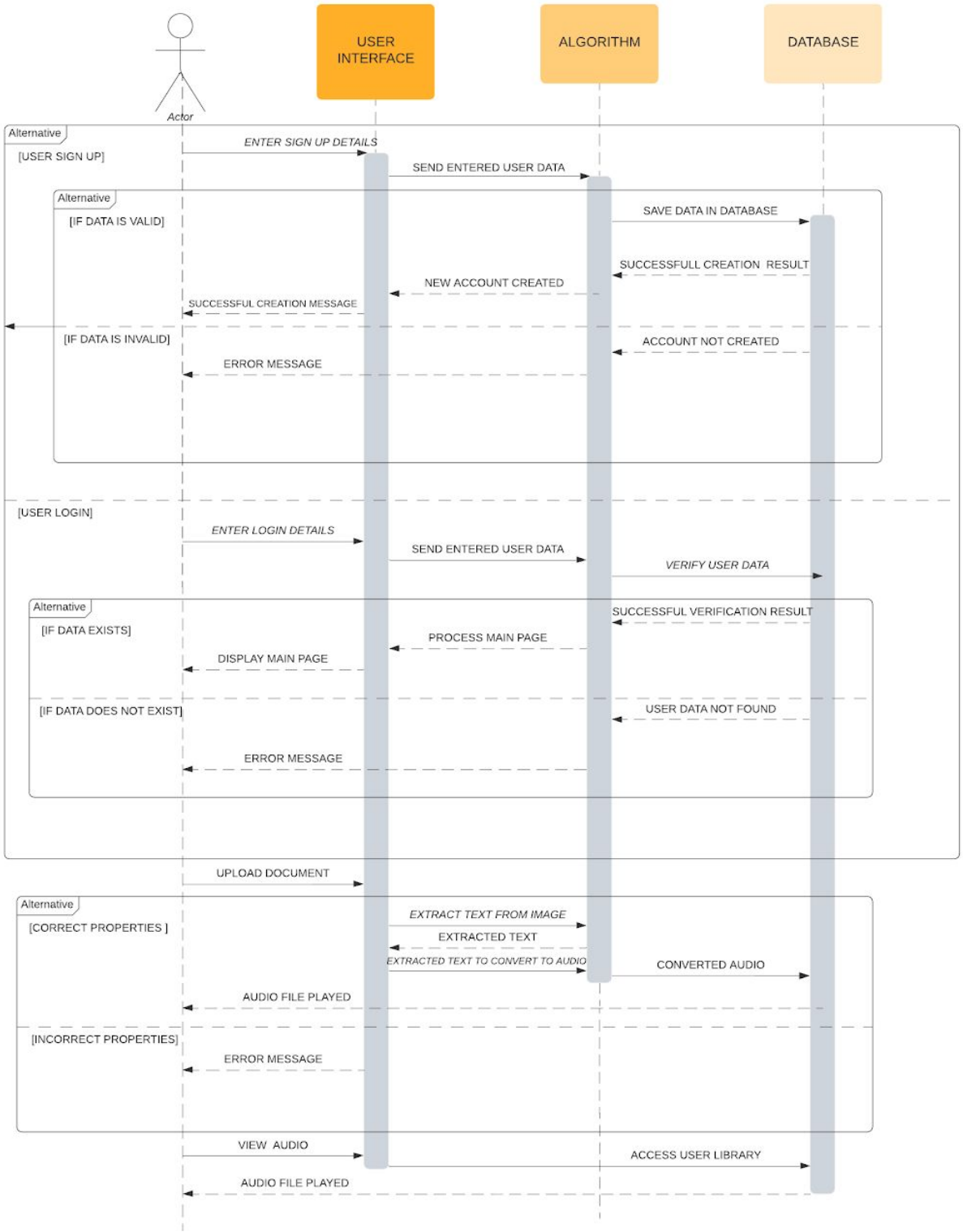
Use Case Name:	Convert text to audio

Summary:	In the conversion process, the text is extracted from the images and then the extracted text is converted to audio.
Basic Flow:	1. The use case starts when the document is verified and uploaded and the user clicks on the convert button. 2. A new folder with the name of the document is created in the current user's folder, and it has the image files of the individual pages in case of a PDF and the image itself in case an image was uploaded. 3. The text is extracted from each image in the folder one by one and appended to a string. 4. The complete extracted text is then converted to audio and saved in the same folder mentioned in point 2..
Alternative flow:	None
Extension points:	None
Preconditions:	The uploaded document has to be verified for size and document type.
Postconditions:	The converted audio begins playing automatically.

View Audio File

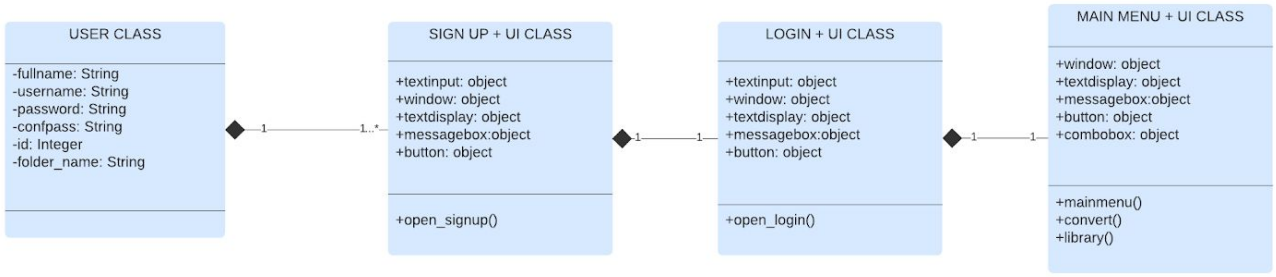
Use Case Name:	View Audio File
Summary:	This represents all the audio files which have been previously converted by the user.
Basic Flow:	1. The use case starts when the user selects an audio from his library of previous documents that he had converted to audio. 2. The user clicks on the play button.
Alternative flow:	None
Extension points:	None
Preconditions:	The user must be logged in.
Postconditions:	The audio of the selected file starts playing.

7. SEQUENCE DIAGRAM

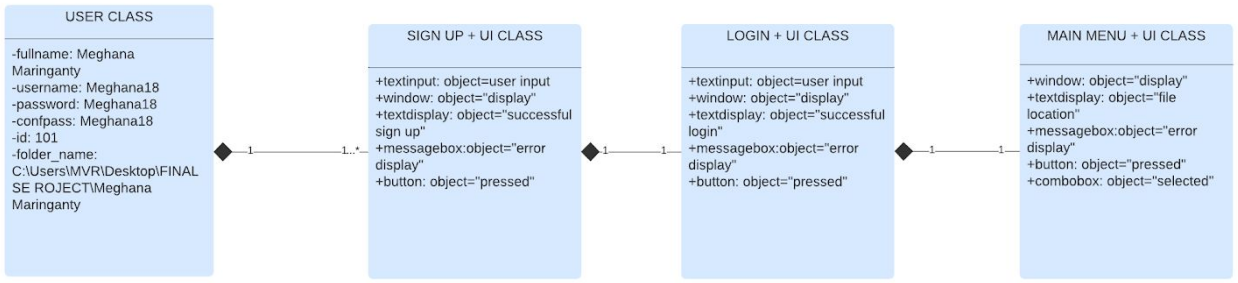


8. CLASS DIAGRAM AND OBJECT DIAGRAM

8.1 Class Diagram



8.2 Object Diagram



9. CODE SNIPPETS.

9.1 Imports

```
from tkinter import *
from tkinter import messagebox
from PIL import ImageTk,Image
from tkinter import ttk
import pickle
from tkinter import filedialog
import pytesseract
from pdf2image import convert_from_path
from pdf2image.exceptions import (
    PDFInfoNotInstalledError,
    PDFPageCountError,
    PDFSyntaxError
)
from gtts import gTTS
from gtts.tokenizer import pre_processors
from gtts.tokenizer import Tokenizer
from gtts.tokenizer import tokenizer_cases
import gtts.tokenizer.symbols
import os
import shutil
from pathlib import Path
import re
```

These are the different libraries, tools, packages and modules that have been imported which will be used in various functions across the code as we will see later.

9.2 Global Variables

```
#********************************************** GLOBAL VARIABLES ***********************************************

global user_list
user_list=[]
global user_count
user_count=0
global newuserlist
newuserlist=[]
global signup_user_list
signup_user_list=[]
global current_user
current_user=0
global fn
fn=""
global selected
selected=""
global media_count
media_count =0
global clicked
clicked=""
global gtts_dict
gtts_dict={"Arabic" : "ar", "Bengali" : "bn","Chinese" : "zh-cn", "Czech" : "cs",
           "Dutch" : "nl", "English" : "en","Filipino" : "tl", "Finnish" : "fi",
           "French" : "fr","German" : "de", "Greek" : "el", "Gujarati" : "gu",
           "Hindi" : "hi","Indonesian" : "id", "Italian" : "it", "Japanese" : "ja",
           "Kannada" : "kn", "Latin" : "la", "Marathi" : "mr","Malayalam" : "ml",
           "Nepali" : "ne", "Polish" : "pl", "Portugese" : "pt", "Romanian" : "ro",
           "Russian" : "ru","Sinhala" : "si", "Spanish" : "es", "Swedish" : "sv",
           "Tamil" : "ta", "Telugu" : "te", "Thai" : "th", "Turkish" : "tr",
           "Urdu" : "ur", "Vietnamese" : "vi"}
global tess_dict
tess_dict={"Arabic" : "ara", "Bengali" : "ben","Chinese" : "chi_sim", "Czech" : "ces",
           "Dutch" : "nld", "English" : "eng","Filipino" : "fil", "Finnish" : "fin",
           "French" : "fra","German" : "deu", "Greek" : "ell", "Gujarati" : "guj",
           "Hindi" : "hin","Indonesian" : "ind", "Italian" : "ita", "Japanese" : "jpn",
           "Kannada" : "kan", "Latin" : "lat", "Marathi" : "mar","Malayalam" : "mal",
           "Nepali" : "nep", "Polish" : "pol", "Portugese" : "por", "Romanian" : "ron",
           "Russian" : "rus","Sinhala" : "sin", "Spanish" : "spa", "Swedish" : "swe",
           "Tamil" : "tam", "Telugu" : "tel", "Thai" : "tha", "Turkish" : "tur",
           "Urdu" : "urd", "Vietnamese" : "vie"}

```

These are all the global variables that have been declared global so that they can be accessed by any function directly and we won't have to declare them again and again.

9.3 User Class

```
class User:

    def __init__(self):
        self.fullname=""
        self.username=""
        self.id=0
        self.password=""
        self.confpass=""
        self.folder_name=""
```

The user class is a class whose objects store all the details of the user whenever a user signs up. It includes instance variables to store user's full name, username, id and folder name of the user.

9.4 Home Page

```
#*********************************************  HOME PAGE ***************************************************************************

root=Tk()
root.configure(bg="#FFFFFF")
root.geometry("1920x1080")
root.title("Home Page")


logo_img=ImageTk.PhotoImage(Image.open("logo.png"))
logo_label=Label(root,image=logo_img,bg="#FFFFFF")
#logo_label.image=logo_img
logo_label.pack(fill=X, pady=(100,0))
login_bt=Button(root, text="Login", bg="#fecd1a", width="50", height="2",font="Montserrat 13 italic", command=open_login)
signup_bt=Button(root, text="Sign Up", bg="#fecd1a", width="50", height="2", font="Montserrat 13 italic", command=open_signup)
login_bt.pack( pady=(30,10), padx="150")
signup_bt.pack( pady="10", padx="150")
root.mainloop()
```

This code snippet is for the Home page, which shows the logo of the application and presents the user with the option of login and sign up.

9.5 Sign Up Page

```
    signup=Toplevel()
    signup.configure(bg="#fddb3a")
    signup.geometry("1920x1080")
    signup.title("Sign Up Page")


    label_title2=Label(signup, text="Sign Up!",bg="#fddb3a",font='Helvetica 40 bold')
    label_title2.grid(row="0", column="0",padx="600", pady=(180,0),columnspan="2")

    e6=Entry(signup, width="40", borderwidth="5")
    label6=Label(signup, text="Full Name:",bg="#fddb3a",font="Oxygen 18" )
    label6.grid(row="1", column="0", padx=(320,0), pady=(50,10))
    e6.grid(row="1", column="1", pady=(50,10), padx=(0,200), ipady="3")

    e3=Entry(signup, width="40", borderwidth="5")
    label3=Label(signup, text="Username:",bg="#fddb3a",font="Oxygen 18" )
    label3.grid(row="2", column="0", padx=(320,0), pady=(0,10))
    e3.grid(row="2", column="1", pady=(0,10), padx=(0,200), ipady="3")

    e4=Entry(signup, width="40", borderwidth="5", show="*")
    label4=Label(signup, text="Password:",bg="#fddb3a",font="Oxygen 18" )
    label4.grid(row="3", column="0", padx=(320,0), pady=(0,10))
    e4.grid(row="3", column="1", padx=(0,200), pady=(0,10),ipady="3")

    e5=Entry(signup, width="40", borderwidth="5", show="*")
    label5=Label(signup, text="Confirm Password:",bg="#fddb3a",font="Oxygen 18" )
    label5.grid(row="4", column="0", padx=(250,0))
    e5.grid(row="4", column="1", padx=(0,200),ipady="3")

    sign_up_btn=Button(signup, text="Sign Up", bg="#0d395f", padx="50", pady="5",font="Montserrat 13", command=confirmation)
    sign_up_btn.grid(row="5",column="0",columnspan="2", pady=(40,10))
```

This is the user interface code for the sign up window, and it has fields to enter the user's full name, username, password , confirm password and signing up by pressing on the sign up button.

```
#********************SIGNUP WINDOW*****************************************************************************************************
def open_signup():


    def confirmation():

        signup_user_list=[]
        g2=open("UserFile.txt","rb")

        while 1:
            try:
                #o = pickle.load(g2)
                #o= Unpickler(g2).load()
                signup_user_list.append(pickle.load(g2))
            except EOFError:
                break
            #newuserlist.append(o)
        g2.close()

        global user_count
        user_count= len(signup_user_list)
        user_count=user_count+1


        if e4.get()==e5.get():
            if len(e4.get()) < 8:
                messagebox.showerror( "Error!" , "Your passwords be atleast 8 characters long, and should have a number and capital letter in it.", parent=signup)
            elif re.search('[0-9]',e4.get()) is None:
                messagebox.showerror( "Error!" , "Your passwords be atleast 8 characters long, and should have a number and capital letter in it.", parent=signup)
            elif re.search('[A-Z]',e4.get()) is None:
                messagebox.showerror( "Error!" , "Your passwords be atleast 8 characters long, and should have a number and capital letter in it.", parent=signup)
            else:
                #global
                x=0
                if any(obj.fullname == e6.get() for obj in user_list):
                    x=x+1
                    print("X======", x)

                if(x>0):
                    user_list[user_count].fullname=e6.get()+str(x)
                else:
                    user_list[user_count].fullname=e6.get()

                user_list[user_count].username=e3.get()
                print(user_list[user_count].username, "aaaaaaa")
                user_list[user_count].password=e4.get()
                user_list[user_count].id=user_count+100
                #user_list[user_count].fullname=e6.get()
                #user_count=user_count+1
                #print(e3.get(), "aaaaaaa")

                #user_count=user_count+1
                conf_signup=Label(signup, text="You have successfully signed up!")
                conf_signup.grid(row="6",column="0", columnspan="2")

                directory_path=r"C:\Users\MVR\Desktop\FINAL SE ROJECT"
                user_path = r"C:\Users\MVR\Desktop\FINAL SE ROJECT" + '\\'  + user_list[user_count].fullname

                user_list[user_count].folder_name = user_path
                g1=open("UserFile.txt","ab")
                pickle.dump(user_list[user_count],g1)
                g1.close()
                print("Creater folder name: ", user_list[user_count].folder_name)

                try:
                    os.mkdir(user_path)
                except OSError:
                    print ("Creation of the directory %s failed" % user_path)
                else:
                    print ("Successfully created the directory %s " % user_path)


                user_count=user_count+1
                #print(user_count)
        else:
            response=messagebox.showerror( "Error!" , "Your passwords do not match! Please enter again.", parent=signup)
```

After the user has entered the details for signing up, all the details of the previously signed up users is retrieved using pickles and stored in a list. This is done so that the current user count and user id can be calculated. Then the text entered in password and confirm password fields is compared, and in case there is any error, it is displayed. Once details are verified, current user's details are stored in the user object created and it is also saved in the user file using pickles. A

new folder with the user's full name is created so that all the documents uploaded by the current user will be in this folder itself. This is done to distinguish between different users.

9.6 Login Page

```
    login=Toplevel()
    login.configure(bg="#fddb3a")
    login.geometry("1920x1080")
    login.title("Login Page")

    label_title=Label(login, text="Login!",bg="#fddb3a",font='Helvetica 40 bold')
    label_title.grid(row="0", column="0",padx="600", pady=(180,0),columnspan="2")

    e1=Entry(login, width="40", borderwidth="5")
    label1=Label(login, text="Username:",bg="#fddb3a",font="Oxygen 18" )
    label1.grid(row="1", column="0", padx=(320,5), pady=(50,10))
    e1.grid(row="1", column="1", pady=(50,10), padx=(0,200),ipady="3")

    e2=Entry(login, width="40", borderwidth="5", show="*")
    label2=Label(login, text="Password:",bg="#fddb3a",font="Oxygen 18" )
    label2.grid(row="2", column="0", padx=(320,5))
    e2.grid(row="2", column="1", padx=(0,200),ipady="3")

    login_btn=Button(login, text="Login", bg="#0d395f", padx="50", pady="5",font="Montserrat 13", command=confirm)
    login_btn.grid(row="3",column="0",columnspan="2", pady=(40,10))
```

This is the user interface code for the login window, and it has fields to enter the user's username, password and logging in by pressing on the login button.

```
# ********************************************** LOGIN WINDOW ********************************************************************

def open_login():

    def confirm():

        global newuserlist
        newuserlist=[]
        g2=open("UserFile.txt","rb")

        while 1:
            try:
                #o = pickle.load(g2)
                #o= Unpickler(g2).load()
                newuserlist.append(pickle.load(g2))
            except EOFError:
                break
            #newuserlist.append(o)
        g2.close()

        #print(type(newuserlist[0]), type(newuserlist[0].username))

        #print(user_count)
        for j in range(0, len(newuserlist)):
            print(len(newuserlist))
            print(newuserlist[j].password)
            if(newuserlist[j].username == e1.get()):
                #print(j, "aaaaaaaaaaaaaaa")
                if(newuserlist[j].password == e2.get()):
                    print(j)
                    print(newuserlist[j].username)
                    print(newuserlist[j].password)
                    print(newuserlist[j].id)
                    print("Folder: ", newuserlist[j].folder_name)
                    global current_user
                    current_user=j
                    login_cnf=Label(login, text="You have successfully logged in!")
                    login_cnf.grid(row="4", column="0",columnspan="2")
                    mainmenu()
                    break
```

```
            else:
                response=messagebox.showerror( "Error!" , "Incorrect Username or Password! Please try again.", parent=login)
                '''unconf_login=Label(login, text="Incorrect details! Please try again.")
                unconf_login.grid(row="4",column="0", columnspan="2")
                e1.delete(0,END)
                e2.delete(0,END)
                open_login()'''
                break
        else:
            continue

```

After the user has entered the details for logging in, all the details of the all the users is retrieved using pickles and stored in a list. Then the text entered in username is compared in the user list and when a match is found, the text entered in the password field is compared with the password of the match found. If they are the same, the login is successful and the main menu is displayed. If it is not successful, a corresponding error dialog box is displayed indicating that the user needs to enter details again to login.

9.7 Main Menu

```
#************************************************** MAIN MENU ***********************************************************

def mainmenu():

    global fn
    main_menu=Toplevel()
    main_menu.configure(bg="#fddb3a")
    main_menu.geometry("1920x1080")
    main_menu.title("Main Menu")

    def selectfile():

        global fn
        filename= filedialog.askopenfilename(parent = main_menu, title="Select a file", initialdir="\C:")
        fn=filename
        print(filename)

        file_size = os.path.getsize(fn)
        print(file_size)

        if(file_size > 20000000):
            response=messagebox.showerror( "Error!" , "File size exceeds 20 MB. Upload a new file.", parent=main_menu)
        elif filename.lower().endswith(('.png', '.jpg', '.jpeg', '.pdf')) != True:
            response=messagebox.showerror( "Error!" , "Upload any pdf or image only.", parent=main_menu)
        else:
            fname_label= Label(main_menu, text = fn, font="Helvetica 10",bg="#fddb3a")
            fname_label.grid(row="1", column="2", columnspan="2", padx="50", pady=(0,50))

```

```
    heading1=Label(main_menu, text="Convert a new file!",font='Helvetica 35 bold',bg="#fddb3a")
    heading1.grid(row="0",column="0",padx=(500,50),pady=(60,50),columnspan="2")

    upload_label= Label(main_menu, text="Upload PDF/Image : ",font='Helvetica 15 bold',bg="#fddb3a")
    upload_label.grid(row="1",column="0",padx=(500,50),pady=(0,50))
    upload_btn= Button(main_menu, text="Upload", bg="#0d395f", padx="50", pady="5",font="Montserrat 13", command=selectfile)
    upload_btn.grid(row="1", column = "1",pady=(0,50) )

    global clicked
    clicked=StringVar()
    clicked.set("Normal")
    speed_label=Label(main_menu, text="Speed: ",font='Helvetica 15 bold',bg="#fddb3a")
    speed_label.grid(row="2",column="0",padx=(500,50),pady=(0,50))
    speed_option=ttk.Combobox(main_menu, width="27", textvariable=clicked)
    speed_option["values"]=("Normal", "Slow")
    speed_option.grid(row="2", column="1",pady=(0,50))

    lang_label= Label(main_menu, text = "Language :", font='Helvetica 15 bold',bg="#fddb3a")
    lang_label.grid(row="3", column="0", padx=(500,50))

    global selected
    selected=StringVar()
    selected.set("English")
    global fn
    print(fn)


    dropdown= ttk.Combobox(main_menu, width="27", textvariable=selected)

    dropdown["values"]= ("Arabic", "Bengali","Chinese", "Czech", "Dutch", "English","Filipino", "Finnish", "French","German", "Greek", "Gujarati", "H
                         "Malayalam","Nepali", "Polish", "Portugese", "Romanian", "Russian", "Oriya", "Sinhala", "Spanish", "Swedish", "Tamil", "Tel
    #dropdown.config(bg="#ec0101")
    dropdown.grid(row="3", column="1")

    convert_btn=Button(main_menu, text="Convert", bg="#0d395f", padx="50", pady="5",font="Montserrat 13", command= lambda: convert(fn))
    convert_btn.grid(row="4", column="0", columnspan="2", pady=(30,10),sticky="n", padx=(600,100))

    heading2=Label(main_menu, text="Choose from Library!",font='Helvetica 35 bold',bg="#fddb3a")
    heading2.grid(row="5",column="0",padx=(500,50),pady=(30,50),columnspan="2")

    library_label=Label(main_menu, text = "Select file :", font='Helvetica 15 bold',bg="#fddb3a")
```

```
    library_label=Label(main_menu, text = "Select file :", font='Helvetica 15 bold',bg="#fddb3a")
    library_label.grid(row="6", column="0", padx=(500,50))

    print("Current user isssssss ", newuserlist[current_user].folder_name)
    folder_list=[f.name for f in os.scandir(newuserlist[current_user].folder_name) if f.is_dir()]

    if not len(folder_list):
        user_lib=Label(main_menu, text="You do not have any saved audio!", font='Helvetica 15 bold',bg="#fddb3a")
        user_lib.grid(row="6", column="1")

    global file_chosen
    file_chosen=StringVar()
    file_chosen.set(folder_list[0])
    folder_list=tuple(folder_list)
    file_option=ttk.Combobox(main_menu, width="27", textvariable=file_chosen)
    file_option["values"]=folder_list
    file_option.grid(row="6", column="1")


    library_btn=Button(main_menu, text="Play", bg="#0d395f", padx="50", pady="5",font="Montserrat 13", command=library)
    library_btn.grid(row="7", column="0", columnspan="2",sticky="n", padx=(600,100),pady=(20,0))
```

This is the user interface for the main menu. When the user successfully logs in, he can do two things:

- He can choose to upload a new document, select the language of the document from the dropdown list of languages and the speed at which he wants the audio to be played. The document properties are verified and he can then click on the Convert button to begin the conversion process.
- He can choose to listen to the audio of a document he had previously converted by selecting the audio he wants to listen to from the list of the converted document audios in his library and click on the Play button to play the audio.

9.8 Document Conversion

```
#********************************************** CONVERSION **********************************************

def convert(fname):


    global selected
    global media_count
    global current_user
    global newuserlist
    global clicked
    print("Clicked: ", clicked)

    if(clicked.get()=="Normal"):
        speed=False
    else:
        speed=True

    print("Speed: ", speed)

    print("Length: ", len(newuserlist))
    print("Current user: ", current_user)
    user_folder=newuserlist[current_user].folder_name
    print("Current user folder: ", user_folder)

    for fol in os.listdir(user_folder):
        if os.path.isdir(fol):
            media_count = media_count+1

    print(media_count)

    #new_folder_path=user_folder+"\\fol"+str(media_count+1)
    #print(new_folder_path)

    new_folder_path=user_folder+"\\"+Path(fname).stem
    print(new_folder_path)

    try:
        os.mkdir(new_folder_path)
    except OSError:
        print ("Creation of the directory %s failed" % new_folder_path)
    else:
        print ("Successfully created the directory %s " % new_folder_path)

    if fname.lower().endswith(('.png', '.jpg', '.jpeg')):

        img = Image.open(fname)

        # save a image using extension
        name = new_folder_path + "\img" + '.jpg'
        print(name)
        #image.save(name, 'JPEG')
        img = img.save(name, 'JPEG')
    else:
        #print(fname)

        images = convert_from_path(fname)

        i = 1
        path_of_images=[]
        for image in images:
            name = new_folder_path + "\img" + str(i) + '.jpg'
            print(name)
            image.save(name, 'JPEG')
            i = i + 1
            #image.save('output' + str(i) + '.jpg', 'JPEG')

    file_text=""

    global selected

    language=selected.get()

    tess_lang= tess_dict[language]
    print(tess_lang)
    gtts_lang= gtts_dict[language]
    print(gtts_lang)

    j=1
    for img in os.listdir(new_folder_path):
        input_path = os.path.join(new_folder_path, img)
        mytext = pytesseract.image_to_string(input_path, lang=tess_lang)
        file_text= file_text + mytext
```

```
        file_text= file_text + mytext
        print(file_text)

    output=gTTS(text=file_text, slow=speed, tld="com", lang=gtts_lang, pre_processor_funcs = [pre_processors.tone_marks,pre_processors.end_of_line,pr
                tokenizer_func=Tokenizer([tokenizer_cases.tone_marks,tokenizer_cases.period_comma,tokenizer_cases.colon,tokenizer_cases.other_punctua
    output.save("testpdf.mp3")
    shutil.move("testpdf.mp3", new_folder_path+"\\"+"testpdf.mp3")
    #output.save("%s.mp3" % os.path.join(new_folder_path+"\\"+testpdf))
    #os.system("start new_folder_path +'\\'+'testpdf.mp3' ")
    os.startfile(new_folder_path+"\\"+"testpdf.mp3")
```

When the user chooses to upload a new document and convert it, a new folder with the document name is created inside the user's own folder. Then it is checked if the uploaded document is an image file, then it is saved as it is in the folder. But if it is a pdf file, each page of the pdf is converted to an image file and stored in the folder. After this is done, we iterate through each image in the folder, extract the text from each image and append the extracted text to a string. Then this extracted text is converted to audio and saved in the document folder.

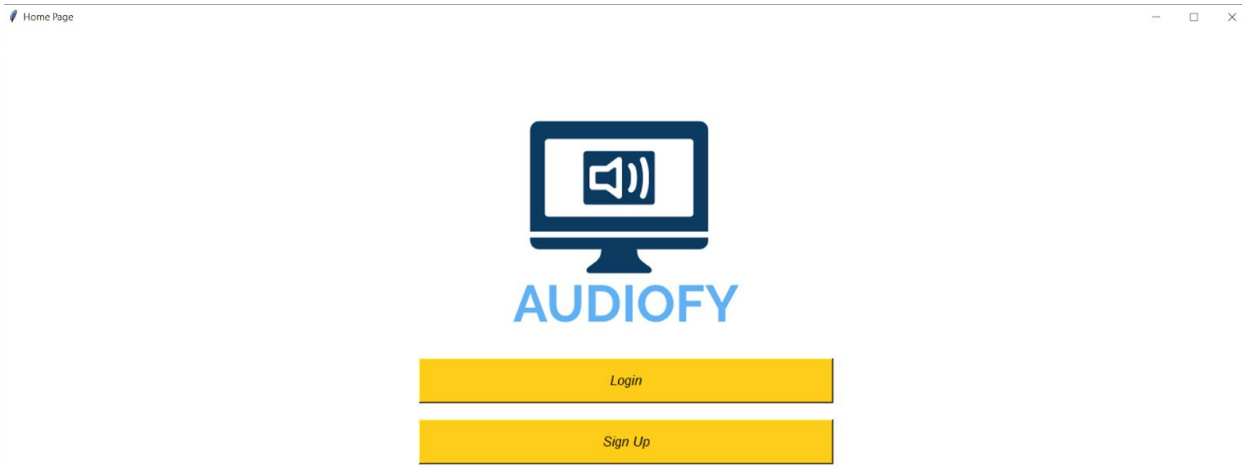
9.9 Library

```
#********************************************** LIBRARY ************************************************************

def library():

    global file_chosen

    chosen_audio_path=newuserlist[current_user].folder_name+"\\"+file_chosen.get()
    os.startfile(chosen_audio_path+"\\"+"testpdf.mp3")
```

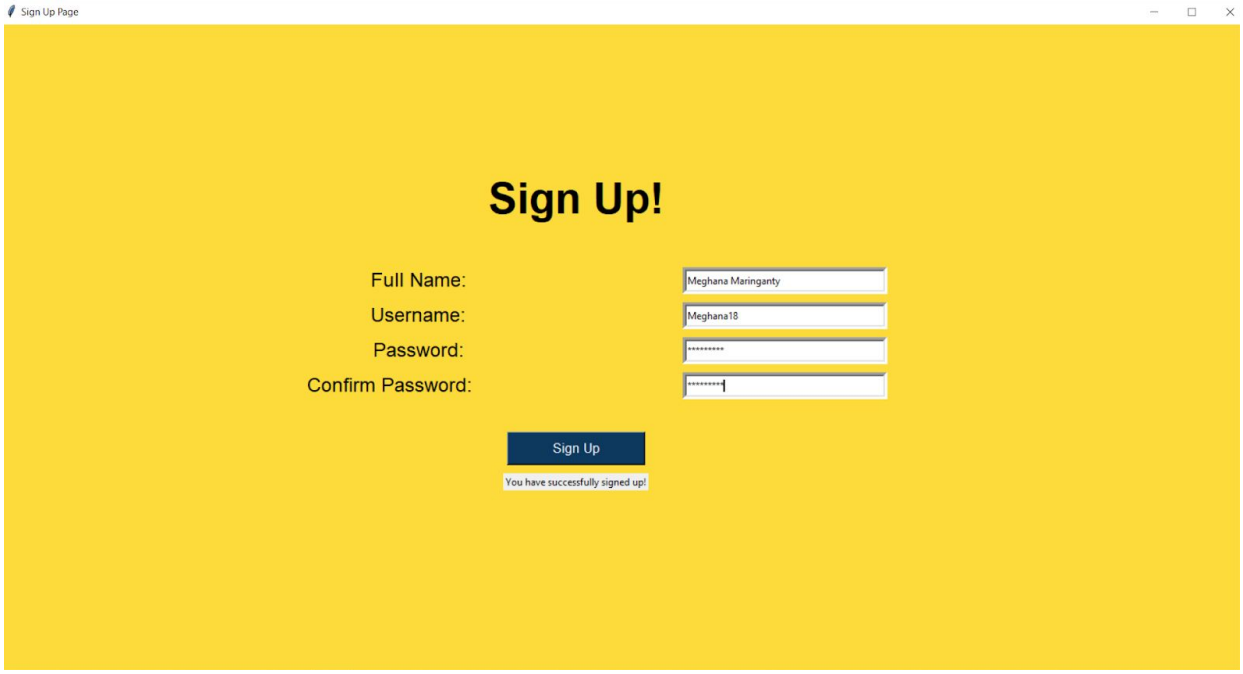
When the user chooses to listen to the audio of a document he had previously converted, the file path of the chosen audio is retrieved and the audio from that folder is fetched and played.

10. OUTPUT

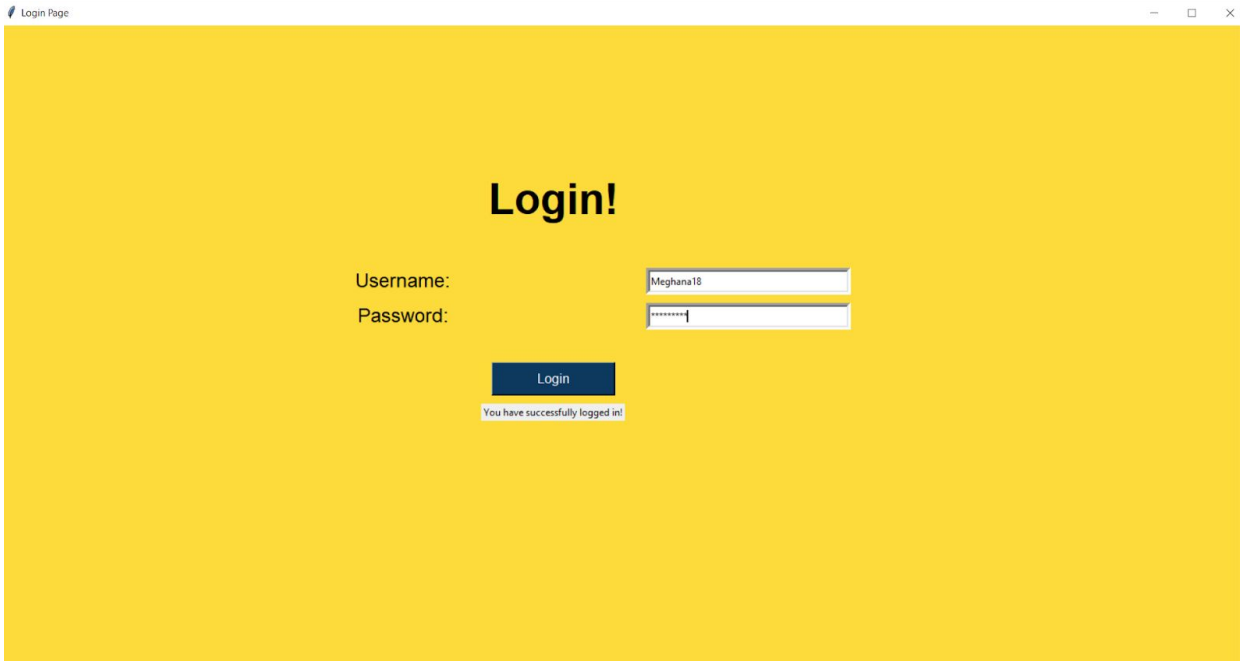
Home Page



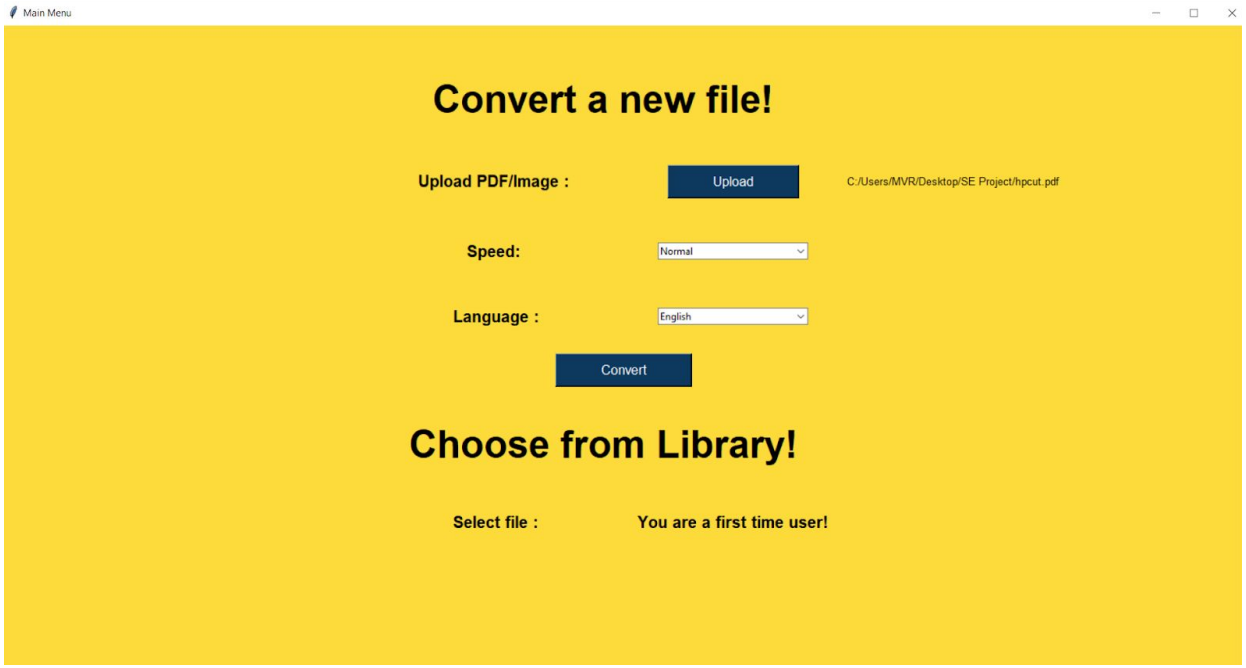
Sign up page



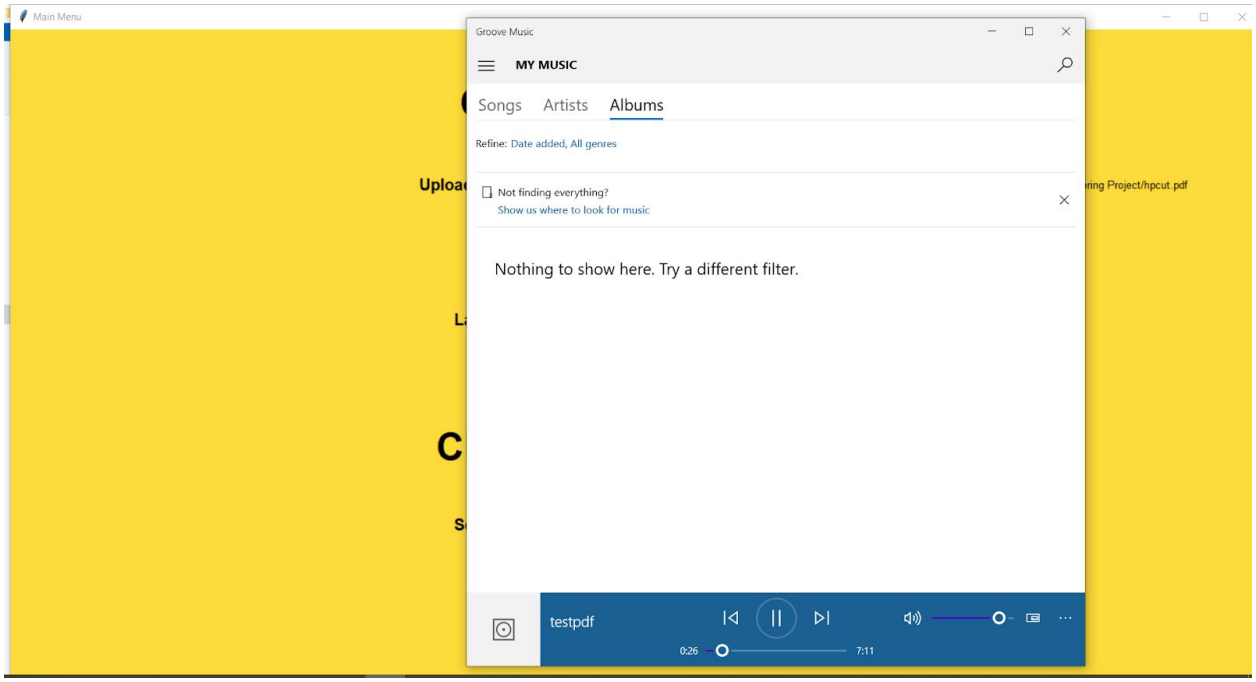
Login page



Main menu (for a first time user)

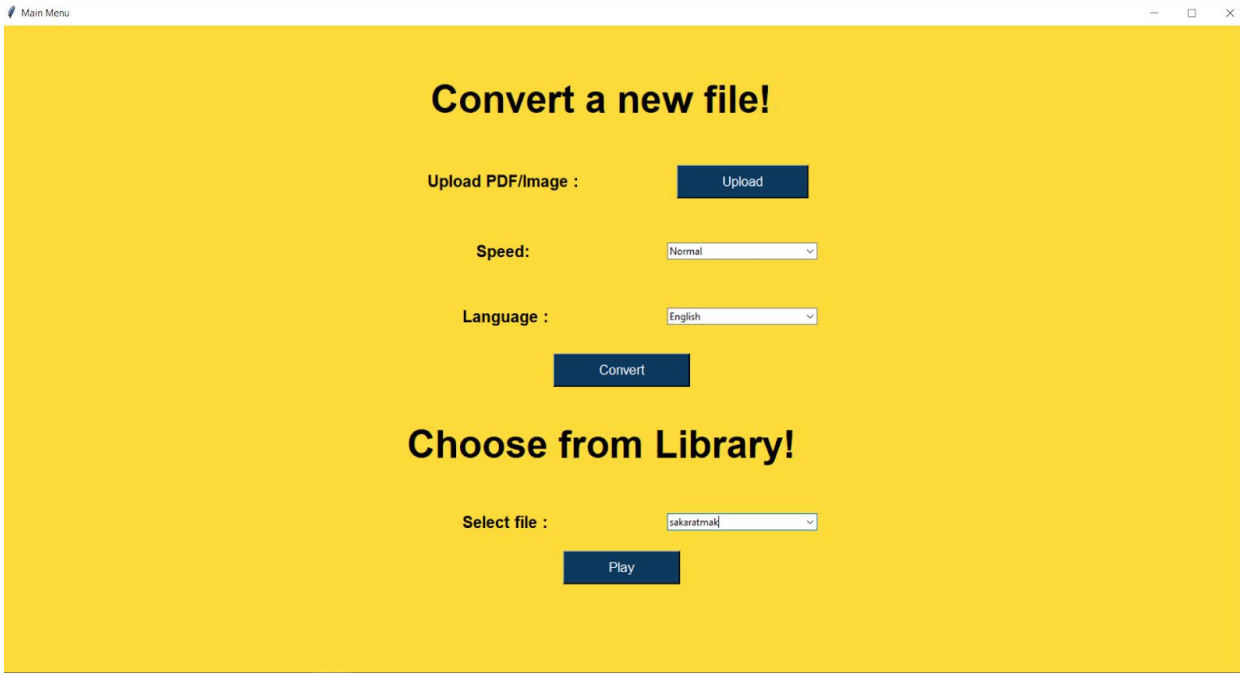


Audio after conversion of document

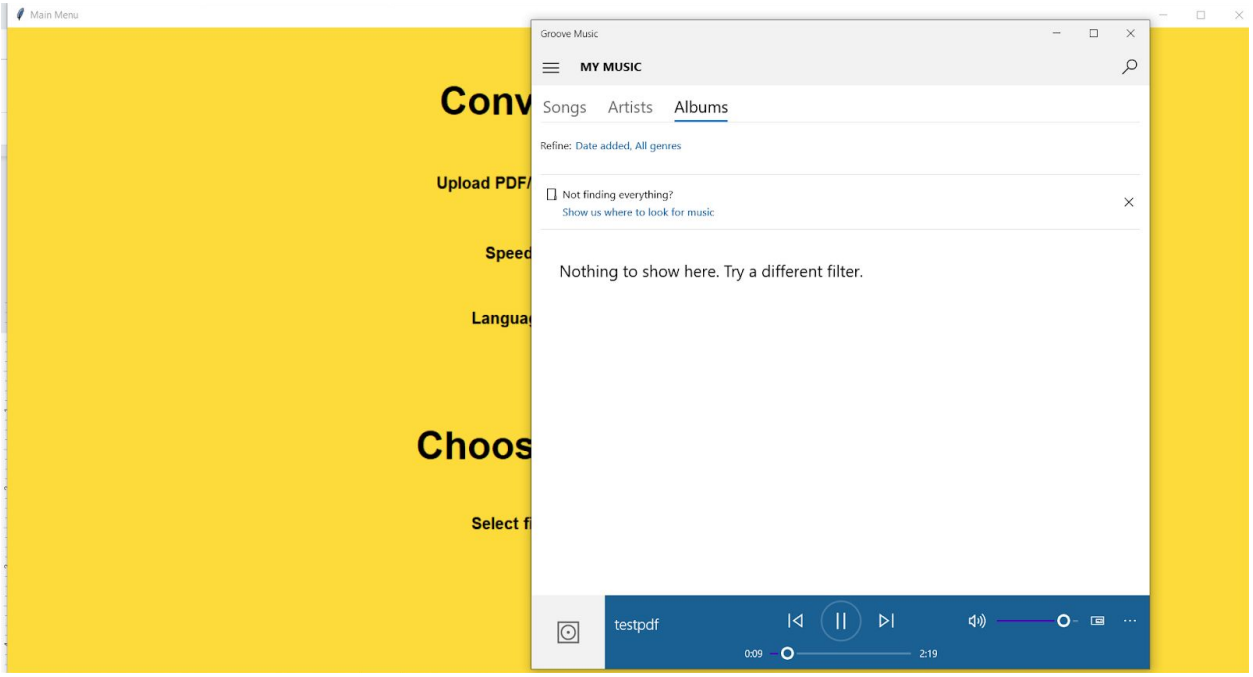


The audio starts playing immediately.

Main menu (for the user who has converted documents before)



Audio of the document chosen from library



The audio starts playing as soon as the play button is clicked.

11. TESTING

11.1 Positive Test Cases

TEST	TEST	TEST STEPS	TEST DATA	EXPECTED	ACTUAL	PASS

CASE ID	SCENARIO			RESULTS	RESULTS	/FAIL
TU01	Check user sign up with valid data.	1. Click on sign up button 2. Enter full name 3. Enter username 4. Enter password 5. Enter confirm password 6. Click Sign Up.	Full Name: Meghana Maringanty Username : Meghana18 Password: Meghana18 Confirm Password: Meghana18	User signed up successfully.	As expected.	Pass.
TU02	Check user login with valid data.	1. Click on login button 2. Enter username 3. Enter password 4. Click Login.	Username : Meghana18 Password: Meghana18	User login successful.	As expected.	Pass.
TU03	Check the uploaded document with a valid extension.	1. Click on the upload button. 2. Select the document. 3. Press Select.	Selected document: C:/harrypotter.pdf	Display selected document name.	As expected.	Pass.
TU04	Check if the uploaded document size is valid.	1. Click on the upload button. 2. Select document to be uploaded. 3. Click on select.	Selected document: C:\Users\MVR\Desktop\SE Project\hpcut.pdf	Display selected document name.	As expected.	Pass.
TU05	Check if the user is able to select the language.	1. Click on the language dropdown box. 2. Choose language	Mouse click.	All the languages available list should be displayed from which	As expected.	Pass.

				user can select.		
TU06	Check if the user is able to select audio speed.	1. Click on the speed dropdown box. 2. Choose speed.	Mouse click.	The speed options of normal and slow should be displayed from which user can select.	As expected.	Pass.
TU07	Check if the user is able to access previously converted documents.	1. Click on the library dropdown box. 2. Choose file.	Mouse click.	Previously converted document names should be displayed from which user can select.	As expected.	Pass.

11.2 Negative Test Cases

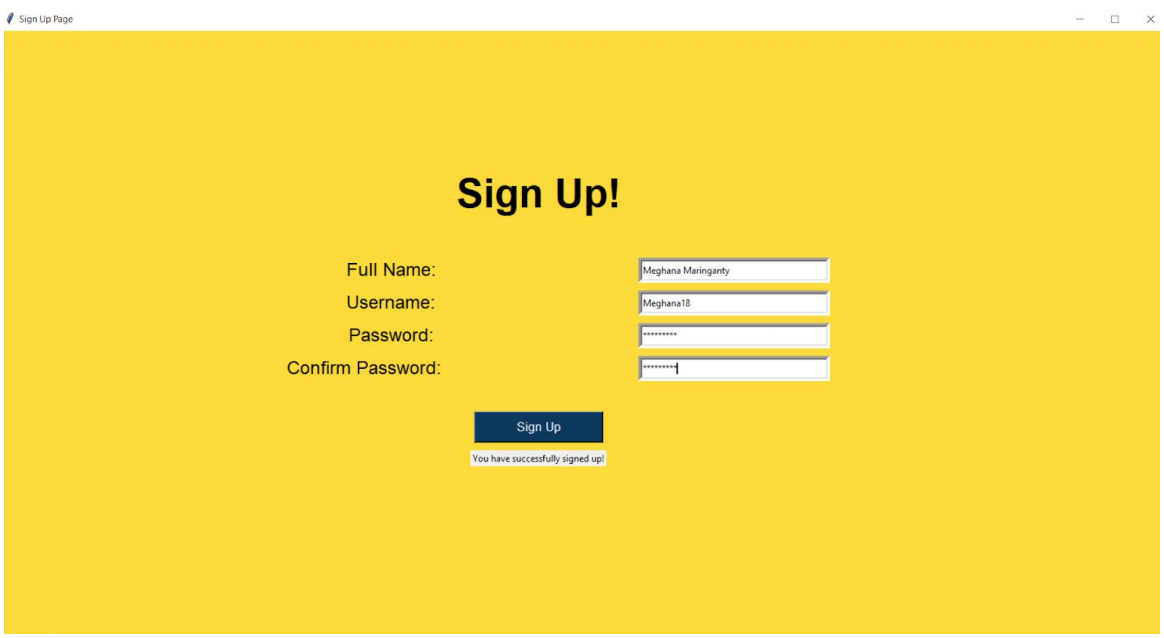
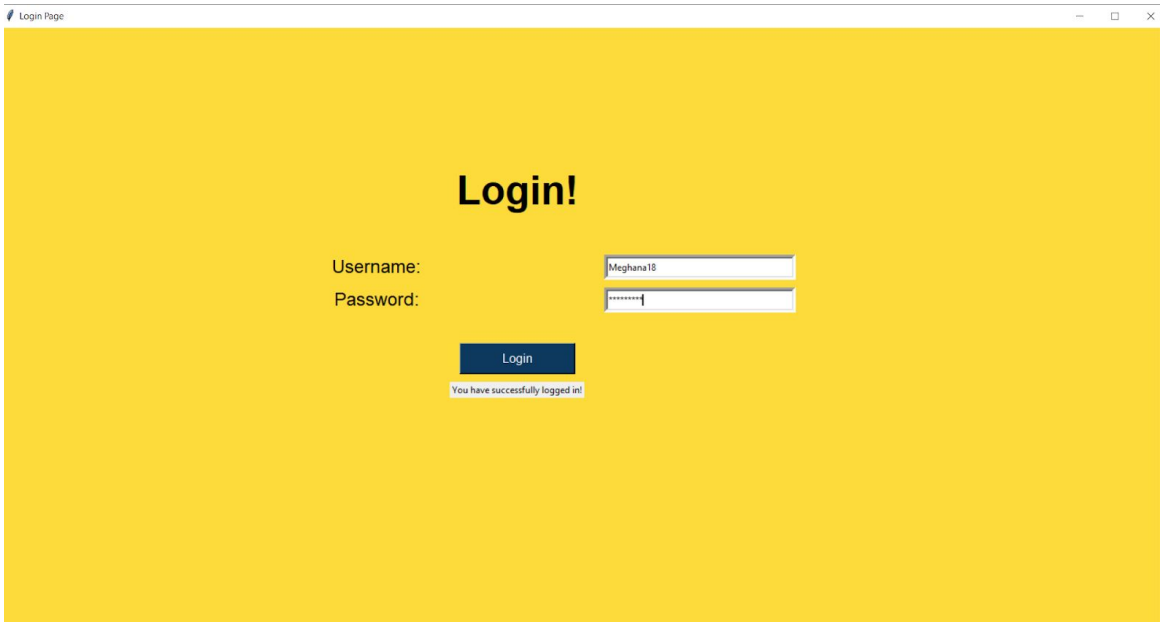
TEST CASE ID	TEST SCENARIO	TEST STEPS	TEST DATA	EXPECTED RESULTS	ACTUAL RESULTS	PASS /FAIL
TU08	Check user sign up with invalid password.	1. Click on sign up button 2. Enter full name 3. Enter username 4. Enter password 5. Enter confirm password 6. Click Sign Up.	Full Name: Rama Devi Username : Rama Password: Ramadevi Confirm Password: Ramadevi	Display error warning dialog box telling password shoul have minimum 8 characte	As expected.	Pass.

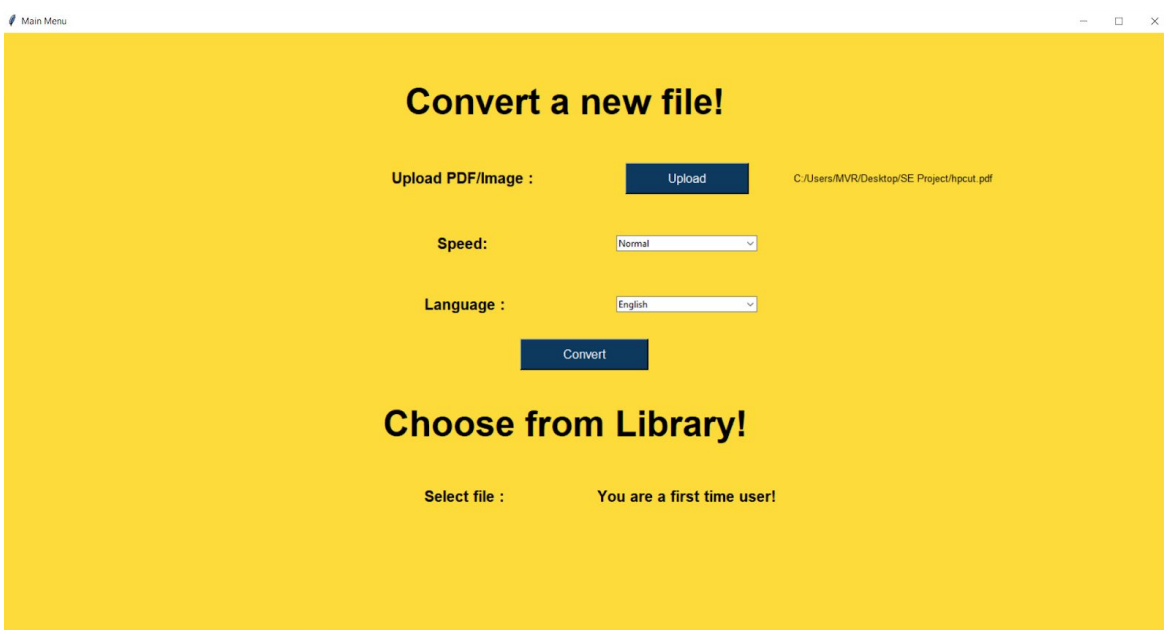
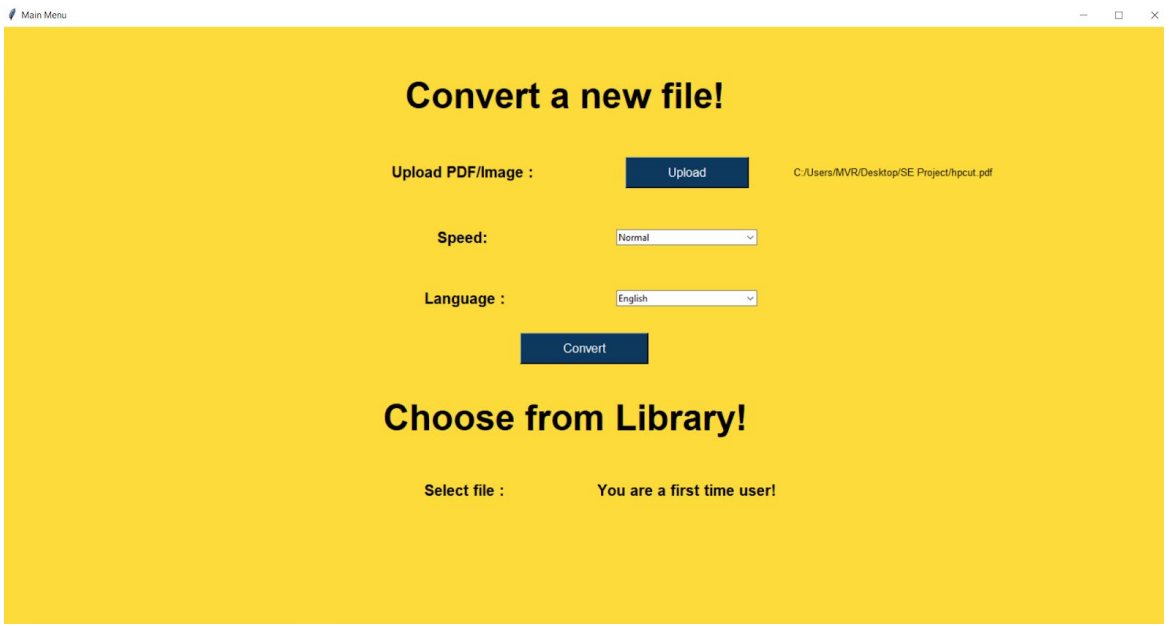
				rs, a number and capital letter.		
TU09	Check user sign up with password mismatch.	1. Click on sign up button 2. Enter full name 3. Enter username 4. Enter password 5. Enter confirm password 6. Click Sign Up.	Full Name: Rama Devi Username : Rama Password: Ramadevi28 Confirm Password: Ramadevi	Display error warning dialog box telling passwords do not match.	As expected.	Pass.
TU10	Check user login with invalid data.	1. Click on login button 2. Enter username 3. Enter password 4. Click Login.	Username : Meghana18 Password: Meghana	User login failed.	As expected.	Pass.
TU11	Check the uploaded document with an invalid extension.	1. Click on the upload button. 2. Select the document. 3. Press Select.	Selected document: C:/harrypotter.dox	Display error warning dialog box telling invalid document type.	As expected.	Pass.
TU12	Check the uploaded document with invalid size.	1. Click on the upload button. 2. Select document to be uploaded. 3. Click on select.	Selected document: C:/harrypotter.pdf	Display error warning dialog box telling invalid document size.	As expected.	Pass.
TU13	Check		Selected	Display	Not as	Fail

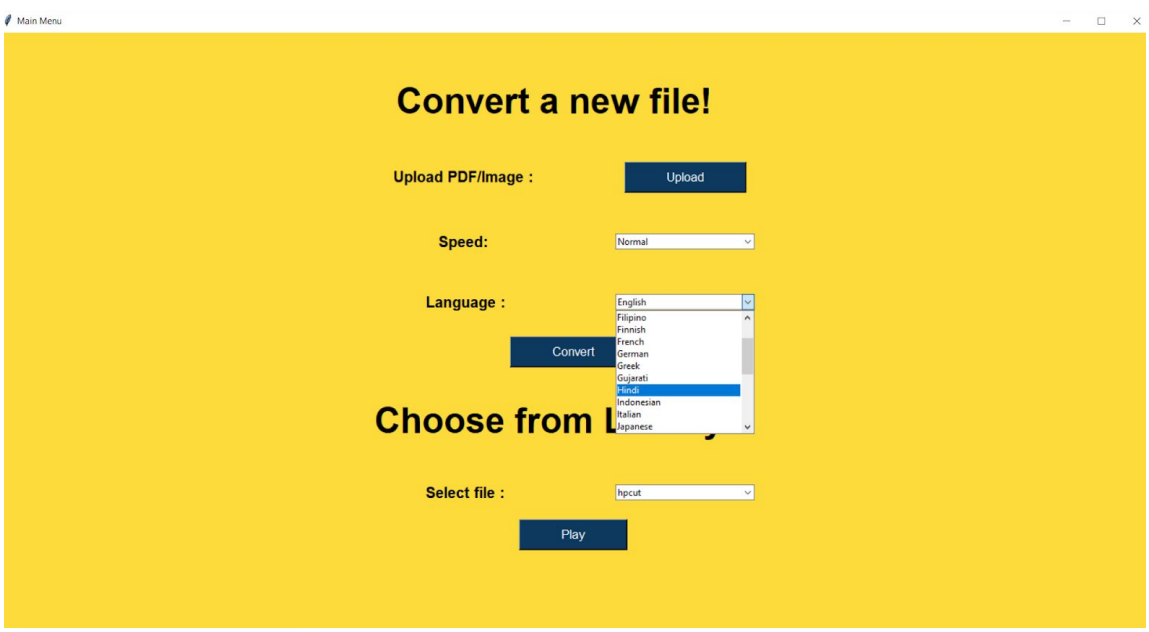
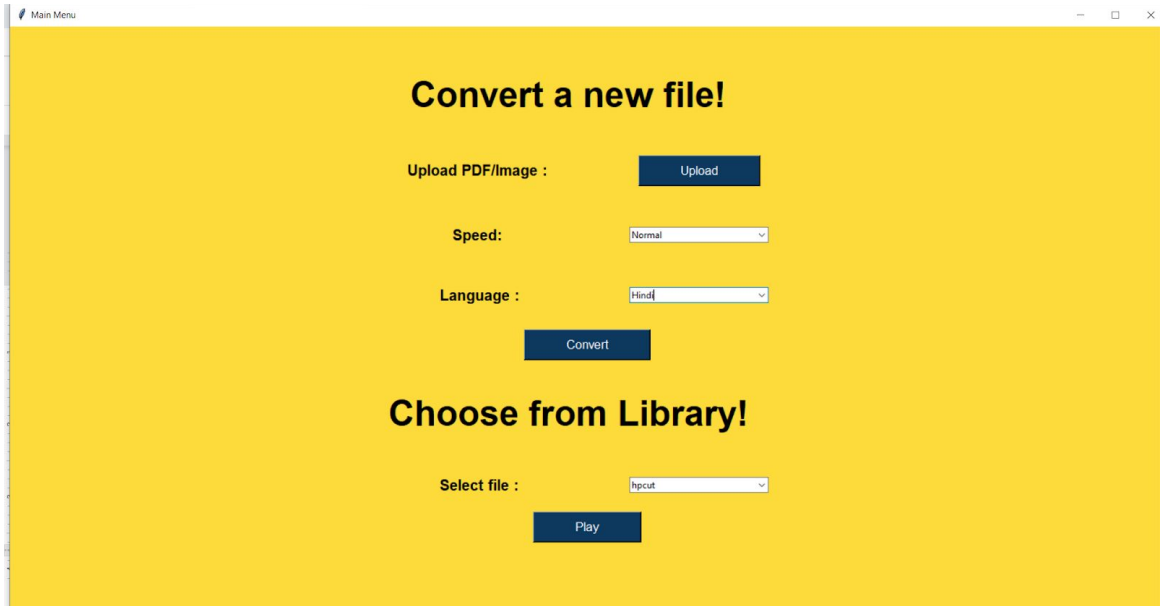
	uploaded document with wrong language selected.	1. Click on the upload button. 2. Select document to be uploaded. 3. Click on select. 4. Choose the language different from what is in the document.	document: C:/harrypotter.pdf	error warning dialog box indicating language selected is wrong.	expected.	
TU14	Check uploaded document with multiple languages.	1. Click on the upload button. 2. Select document to be uploaded. 3. Click on select. 4. Choose the language different from what is in the document.	C:/multilangdoc.pdf	Display error warning box indicating document should have only one language.	Not as expected.	Fail.

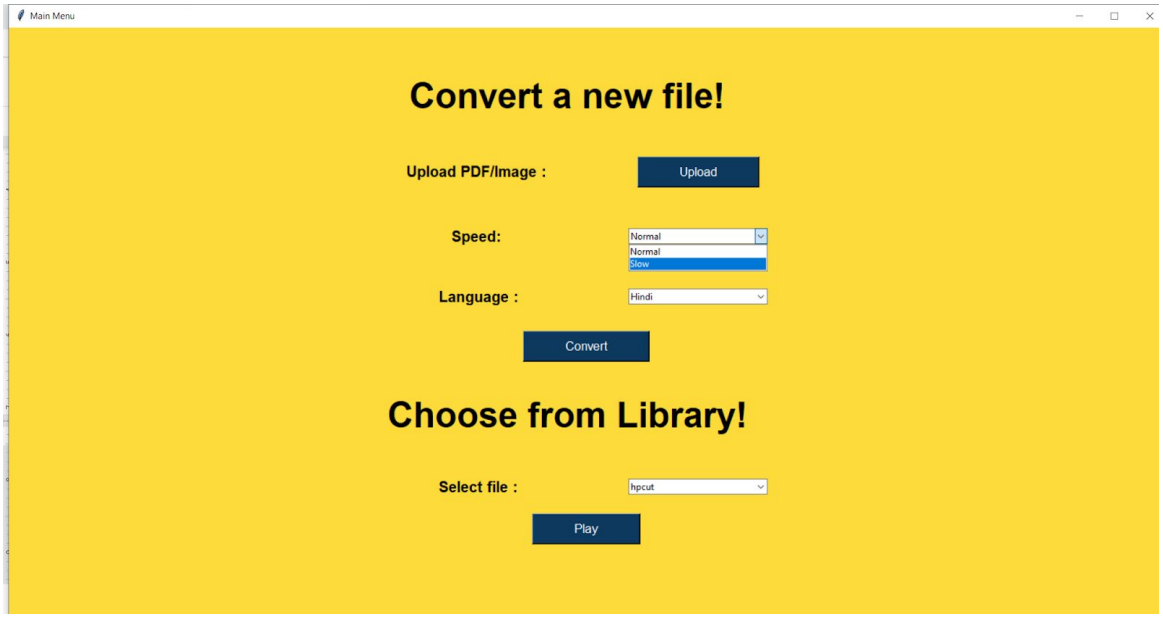
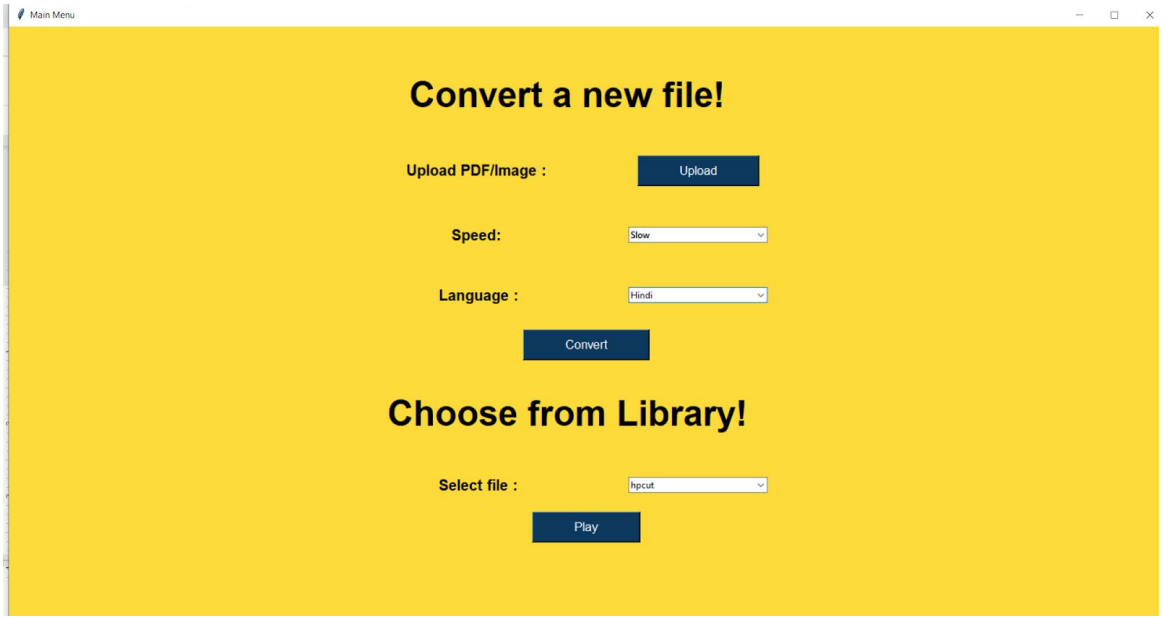
11.3 Test Case screenshots

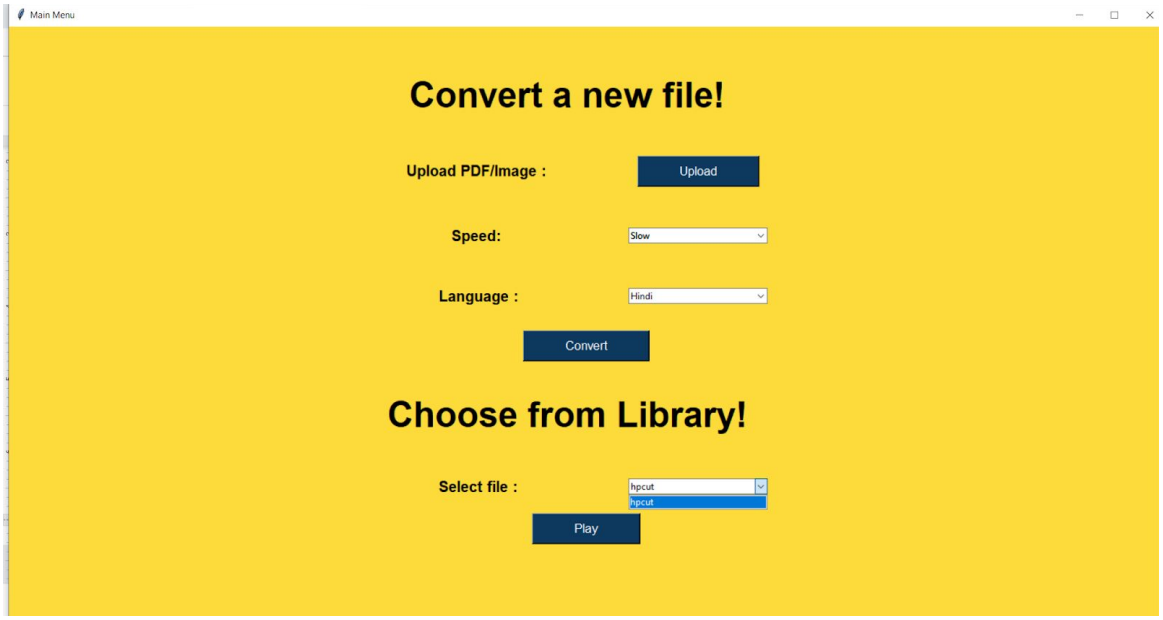
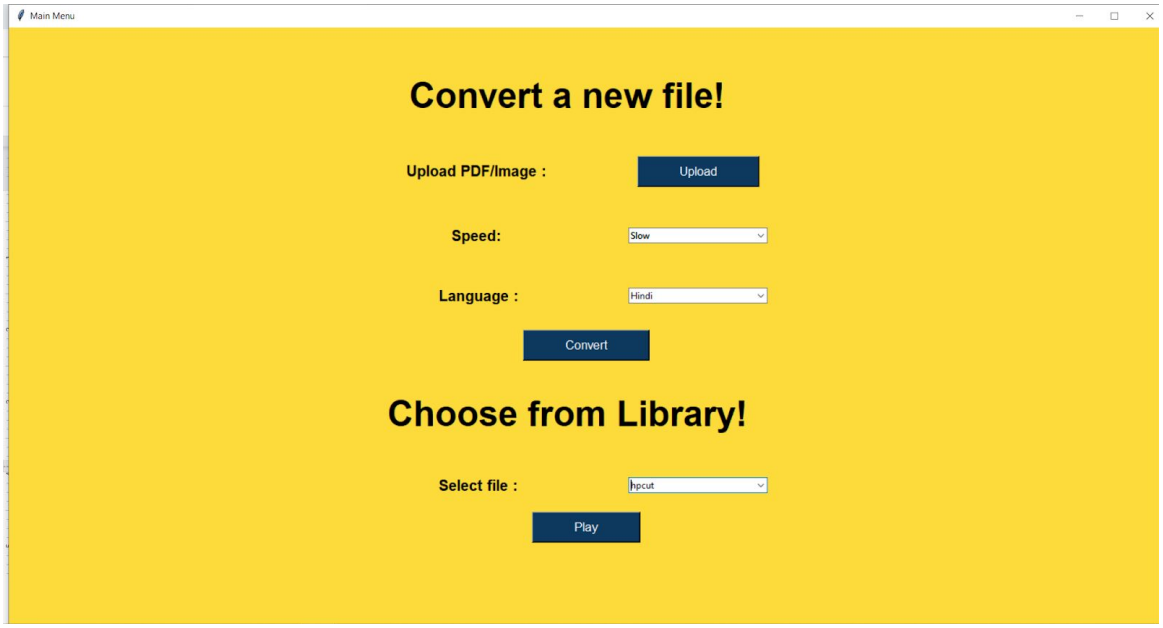
TEST CASE ID	SCREENSHOT

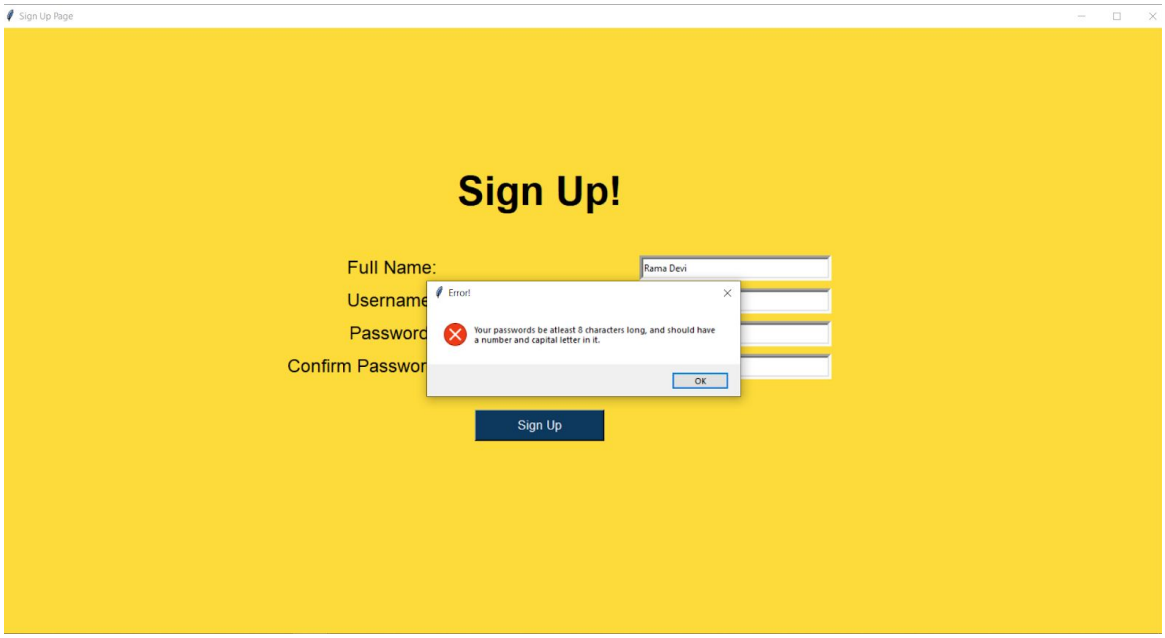
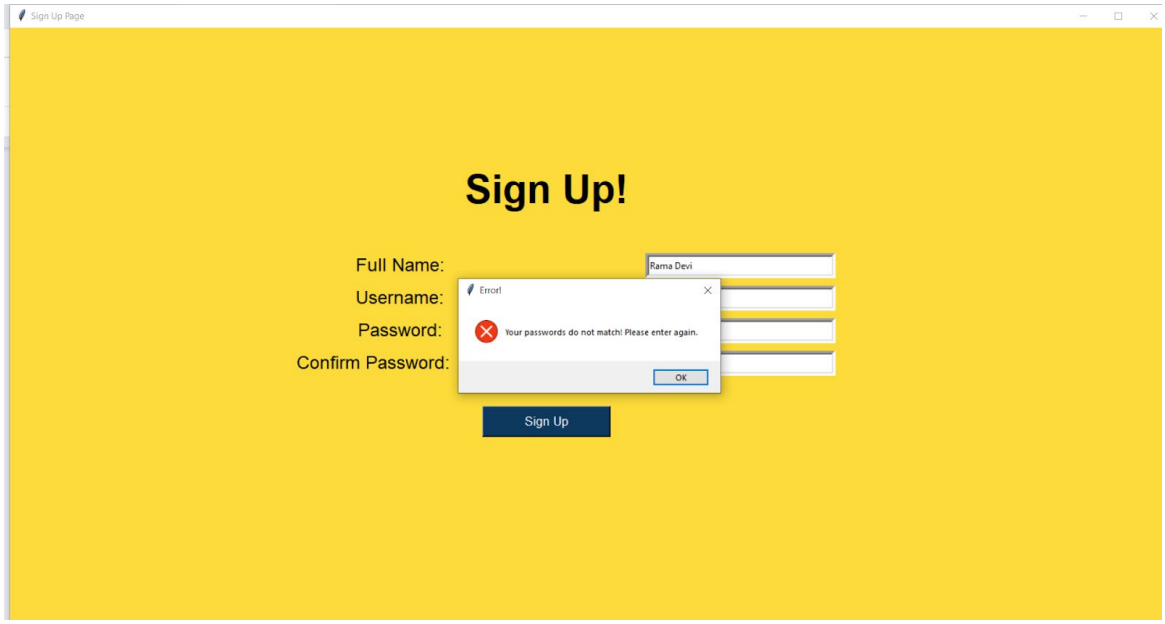
TU01	 Sign Up Page Sign Up! Full Name: Meghana Maringanty Username: Meghana18 Password: Confirm Password: Sign Up You have successfully signed up!
TU02	 Login Page Login! Username: Meghana18 Password: Login You have successfully logged in!

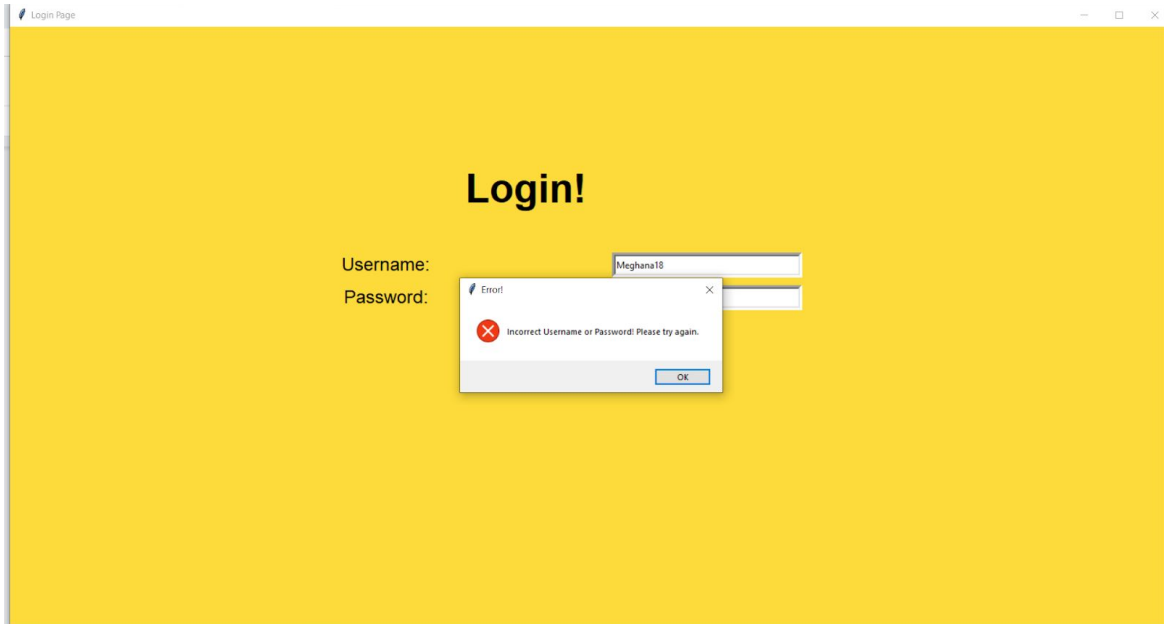
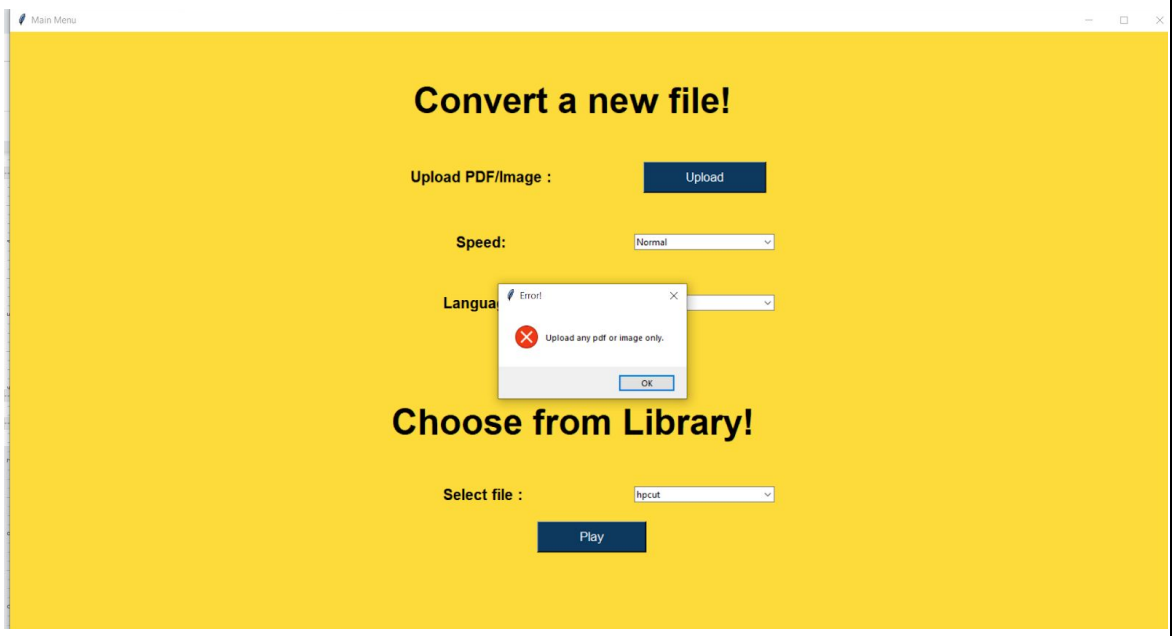
TU03	 Main Menu Convert a new file! Upload PDF/Image : Upload C:/Users/MVR/Desktop/SE Project/hpcut.pdf Speed: Normal Language : English Convert Choose from Library! Select file : You are a first time user!
TU04	 Main Menu Convert a new file! Upload PDF/Image : Upload C:/Users/MVR/Desktop/SE Project/hpcut.pdf Speed: Normal Language : English Convert Choose from Library! Select file : You are a first time user!

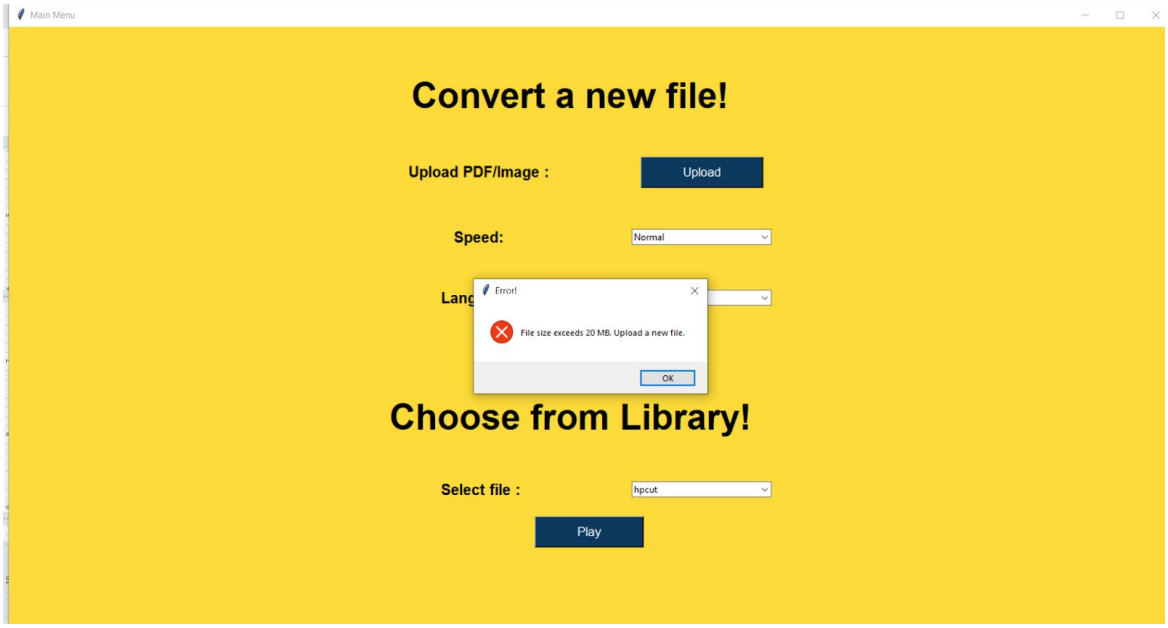
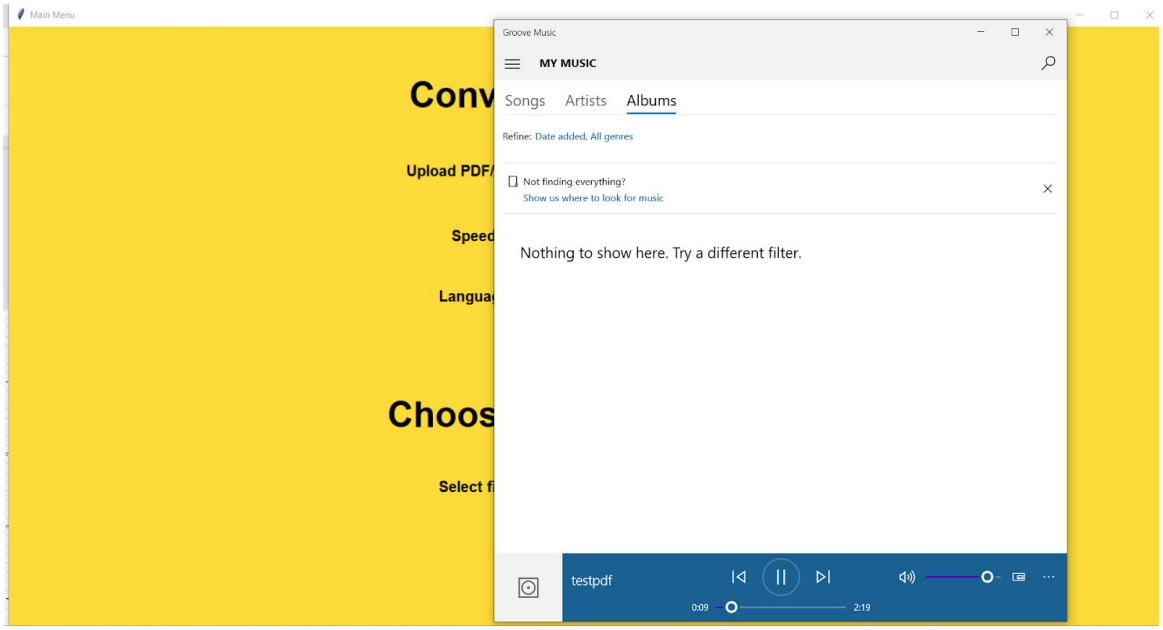
TU05	 After selecting language 

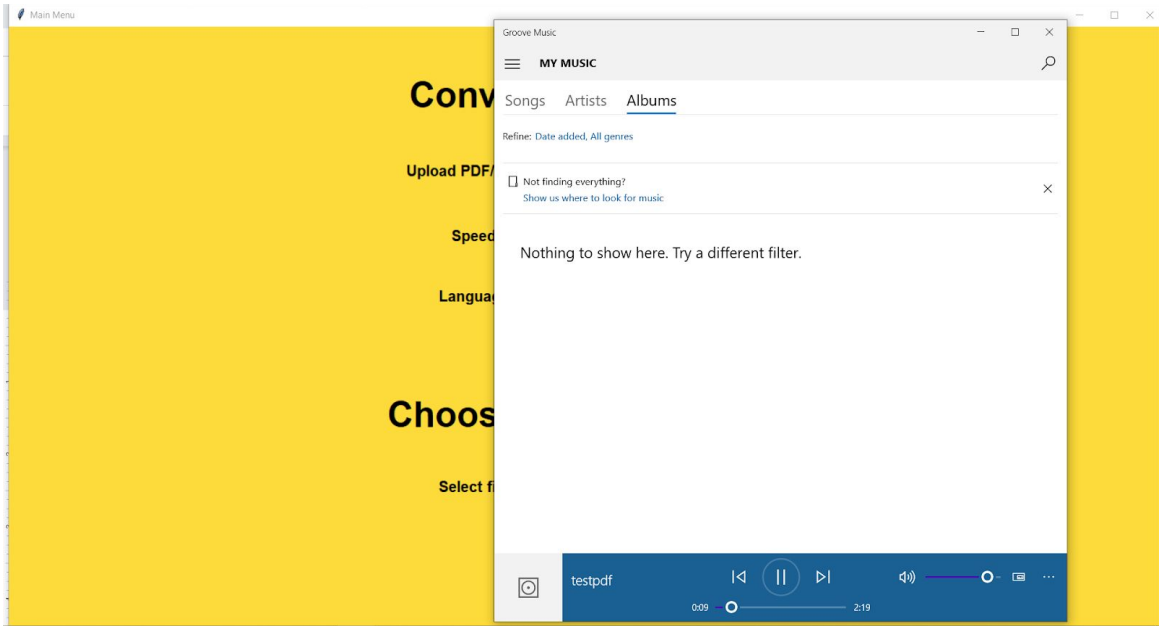
TU06	 After selecting speed, 

TU07	 After selecting the document, 

TU08	 Sign Up Page Sign Up! Full Name: Rama Devi Username Password Confirm Passwor Error! Your passwords be atleast 8 characters long, and should have a number and capital letter in it. OK Sign Up
TU09	 Sign Up Page Sign Up! Full Name: Rama Devi Username: Password: Confirm Password: Error! Your passwords do not match! Please enter again. OK Sign Up

TU10	 Login Page Login! Username: Password: Meghana18 Error! Incorrect Username or Password! Please try again. OK
TU11	 Main Menu Convert a new file! Upload PDF/Image : Upload Speed: Normal Langua Error! Upload any pdf or image only. OK Choose from Library! Select file : hpcut Play

TU12	
TU13	Even though the language selected and language of the document does not match, an audio speaking incoherent speech plays, 
TU14	For a multi language document, since more than one languages cannot be selected from the dropdown, an audio speaking incoherent speech plays,

In case of the last two test cases, the second test case is due to the failure of the first test case. As we need to mention the language for extracting text using tesseract, if wrong language will be mentioned, then the text will not be extracted properly. In case a multi language document is used, since we are given the option of choosing only one language, multiple languages cannot be chosen. This problem can be solved by being able to detect the language of the document uploaded and comparing it with language detected. We can work on this in the future by developing another library or software which detects languages from documents and use it in this application, which will make it more efficient to use and more diverse.

12. VALIDATION AND CONCLUSION

As we conclude all the test cases and verify if all the test cases are passing or not, we have seen that barring two test cases, the results which we expected matched the actual outcomes for the rest, thus indicating that the requirements have been met properly. Since the test cases, which were positive and negative both, tested both functional and non-functional requirements and have passed, we can say that the application is able to handle almost all flow of operations without any discrepancies. Therefore, we can say that the application's validation has been achieved successfully.