Amppper
(more features to be added)

Submitted in partial fulfilment of the requirements
for the award of the degree of
Bachelor of Computer Applications

Guide

Submitted by
Ajay Singh Rana

Submitted To

D.S.B. Campus, Kumaun University,Nainital (U.K.)

Acknowledgement

Getting stuck in the process of learning and building something is kind of a common scenario and so did happen to me a lot of times while working on this project which on the submission date too remains under development given my plans to add more features to the project.

While getting struck is frustrating it teaches us to solve problems with grit and so does the **StackOverflow** community where I found answers to most of my problems. The same goes for documentations and **github issues** where we can get help directly from project managers and contributors.

YouTube was the starting point where I learnt some basic GUI development and would like to attribute a channel in particular i.e. **buildwithpython**.

It is also important to mention my **parents** and **teachers** who took faith in me and guided and taught me to achieve.

CERTIFICATE

This is to certify that this project entitled **“ Amppper”** submitted in partial fulfillment of the degree of Bachelor of Computer Applications to the **Dr. Ashish Mehta** through D.S.B.Campus,
done by Mr./Ms. ***Ajay Singh Rana***, Roll No. ***180125330014*** is an is an authentic work carried out by him/her at D.S.B.Campus
under my guidance. The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Signature of the guide

SELF CERTIFICATE

This is to certify that the dissertation/project report entitled **Amppper** done by me is an authentic work carried out for the partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Applications under the guidance of
Dr. Ashish Mehta. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Ajay Singh Rana

Roll No. 180125330014

D.S.B. Campus,
Kumaun University,Nainital(U.K.)

Synopsis

This project is about creating a Desktop application which can be used as a toolset to work on multimedia and PDF files.
As of now the following features for respective media have been successfully added to the project:

- **Audio**
    1. Convert audio file formats
    2. Join audio files
    3. Trim audio files
- **Video**
    1. Join Video Files
    2. Extract Audio from Video Files
    3. Mute a Video File
    4. Trim Video Files
- **PDF**
    1. Split PDF into pages
    2. Extract text from PDF

Following media and features will be added to this project once i'am able to develop those:

- **PDF**
    1. Read a PDF
- **Images**
    1. Rotate
    2. Change format
    3. Add Filters
    4. Create Thumbnails
    5. Resize

The Problem

I would often find myself searching online for tools to work around with audio,video,pdf and images and I saw most of the tools were too heavy and other online tools required an internet connection.

Meanwhile in my quest to find solutions for my problem i came across **ffmpeg** a command line tool used for audio/video/image encoding and decoding I found it interesting but was harder for me to grasp but after a few more searches I found that python had libraries which encapsulated the functions of **ffmpeg** to provide a simpler interface.

Therefore I set out to build a GUI application with some basic functionalities that I often would look up the internet for.

Objective and Scope of the project

When I started out my objective was to create a GUI application which would be able to do the following:

1. Trim an audio file
2. Change the audio file format
3. Join two audio files

But later on the project grew to add some Video and PDF features:

1. Join two Video Files
2. PDF to text file
3. Mute a Video
4. Extract Audio From Video

As of now as the project has been created to solve a personal problem I don't see a wider scope for the project itself.
But as I am keen to add more features to it I hope it can be a lightweight alternative to heavy Softwares that provides tools for single media files and even though these are efficient and have more tools to offer most of these options are either never utilised or under utilized.
I focus on basic functionalities for the different types of media which are frequently used by people.

I plan to add the following features:

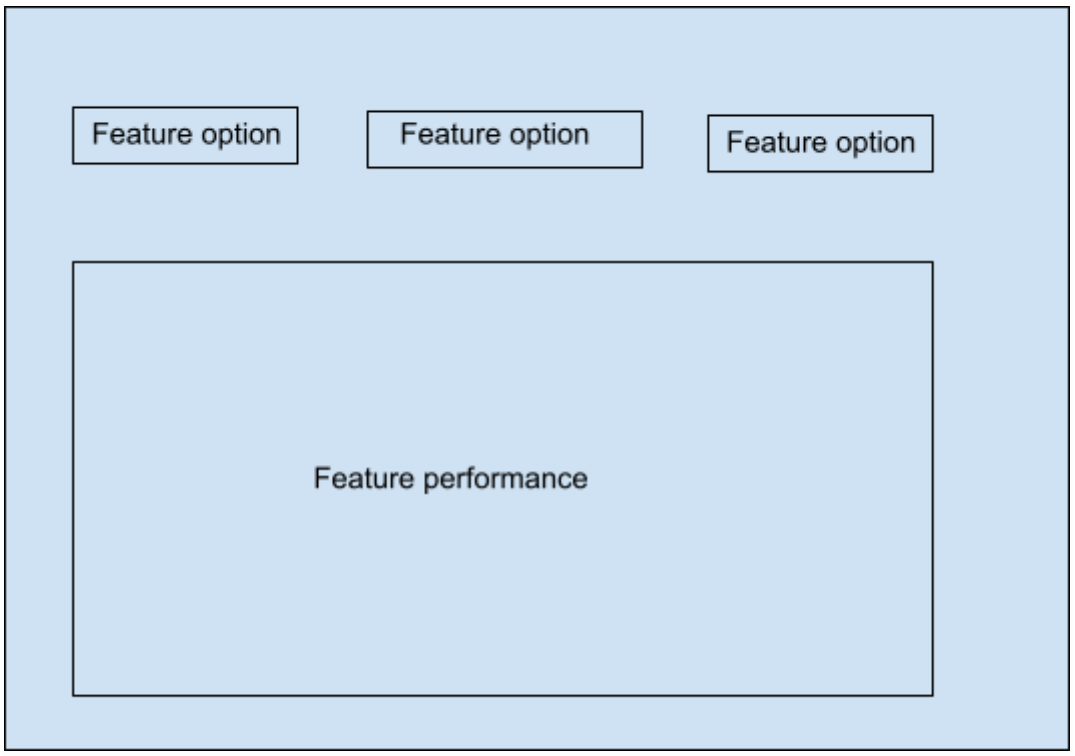
- Add a feature to Read a PDF file
- Audio Recorder
- Feature for creating PDF with command like interface within the GUI
- Basic Image options like rotate,resize,create thumbnailing and filters
- Screenshot and screen recording features
- A simple camera and recorder using openCV

How did the project progress..?

Firstly, I had to give my project an interface and I chose a **GUI** over **CLI.**
Python comes packed with **Tkinter** which is a well documented library and therefore I head on with it.
The next problem was to find libraries to provide the functionalities with and I chose to go ahead with the following:

- **Pydub**
  A python library that ***Manipulates audio with a simple and easy high level interface***
- **MoviePy**
  MoviePy is a Python module for video editing, which can be used for basic operations (like cuts, concatenations, title insertions)
- **PyPDF2**
  To work with pdf's
- **pdfminer**
  PDFMiner is a text extraction tool for PDF documents.

After having chosen the libraries I set out to firstly devise a GUI for the project. And after trying out multiple variants I came up with a single window layout for all features so that it is easier to familiarise with the interface.



Feature interface

Main Window

Code organisation

Organising the code is important too and that's because in a larger project it gets harder and sometimes impossible to maintain the code or even change the code.
At first my code wasn't organised as well and as the project grew I started facing serious problems maintaining my code as it had already grown clumsy. So, I revamped it all and redesigned my code.
Firstly I figured the main issues:

- **Redundant code:**
  I would have to create several Buttons,Labels separately for each Feature I would add which was kind of just increasing the number of lines in the code and wasn't helping at all.Writing a whole new class for adding a new feature with the same old Buttons and labels seemed trivial to me.
- **Tedious Changes:**
  Now each time I wanted to make a change to the look of Buttons I would have to change the code line for each and every button that I implemented. Which seemed frustrating.

How I Changed it All..?

I had learnt about Inheritance, so I created a class titled **CFrame** which would be inherited by all classes implementing a new feature and would contain all GUI widgets commonly shared by a Feature Frame.

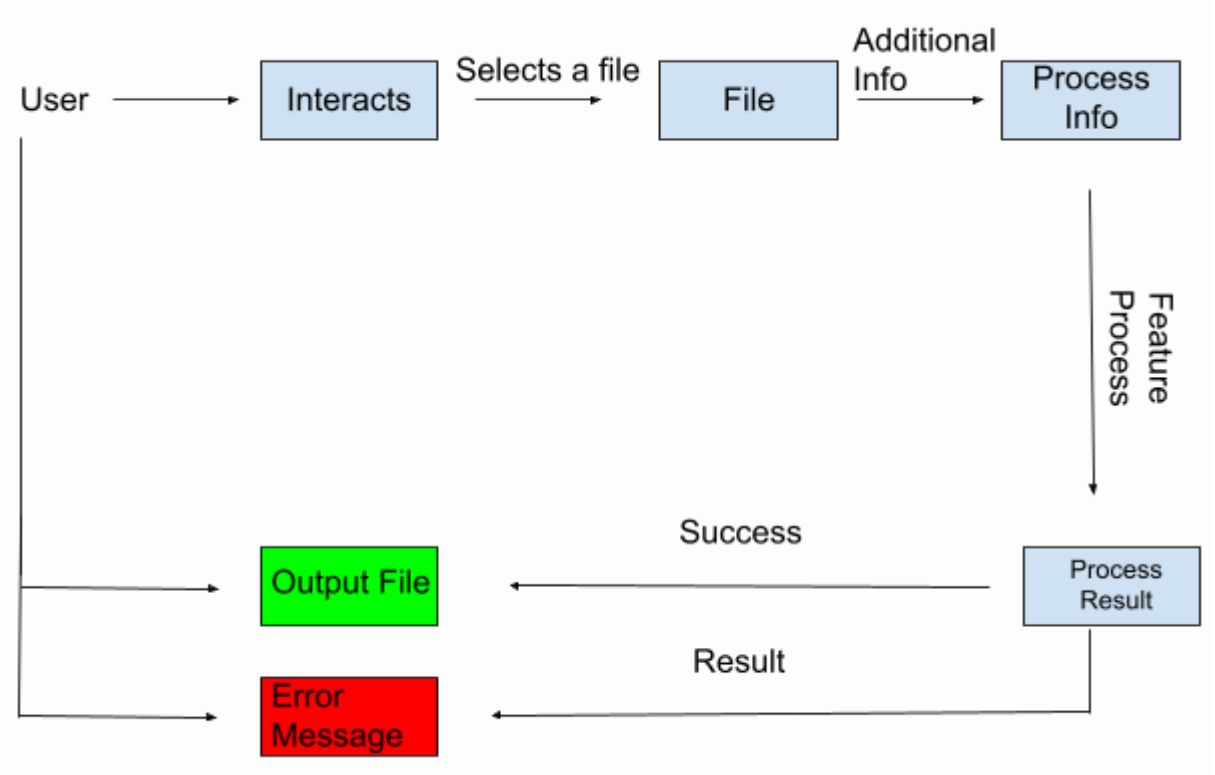
So, I did for the window frames which would list options for different file types.I created a **StandardWindow** class to implement new Frames.

```
class CFrame(tk.Frame):
    """
    This class implements the frame
    that i'll be using in different tool windows
    to display the different functionalities that
    can be used under that tool
    """
    def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
        self.filename = ''
        self.thread = 0
        self.controller = controller
        self.config(bg = '#2AA2BC')

    def browse(self,present = 1):
        self.filename =
filedialog.askopenfilename()
        if(self.filename and present):
            self.label.config(text =
os.path.split(self.filename)[1])

    def run_thread(self,target_func):
        self.thread += 1
        thread = threading.Thread(target =
target_func)
        thread.start()
```

CFrame Class that all other feature classes will inherit from.

How are the features supposed to work..?

Even though it is a GUI program it is not extensive but lightweight and therefore the features would work purely as the command-line tools would do i.e. based on the user input.
Suppose we have to trim an audio file then the user is supposed to select a file for that and would provide the seconds between which the audio is to be trimmed and then the program will work on it in the background and inform the user if the process was successful or not.



Program Workflow

```
def trim(self):
        if(self.filename):
            ext = os.path.splitext(self.filename)[1]
            filename =
filedialog.asksaveasfilename(defaultextension = ext,filetypes
= ((f'{ext} files',f'*{ext}'),))
            if(filename):
                try:
                    format_ =
{'mp3':'mp3','flv':'flv','ogg':'ogg','wav':'wav','wma':'wma',

'avi':'avi','aac':'adts','m4a':'mp4','flac':'flac','opus':'op
us','au':'au',
                                'aiff':'aiff','webm':'webm'}
                    try:
                        start =
int(float(self.from_spin.get()) * 1000)
                        end = int(float(self.to_spin.get()) *
1000)
                        if(start > end):
                            raise

                        self.status.config(text =
'Processing...',bg = '#2a8d12')
                        self.queue.config(text = f'Queued:
{self.thread}',bg = '#2a8d12')
                        self.label.config(text = '---Empty
Selection---')
                        self.to_spin.delete(0,'end')
                        self.from_spin.delete(0,'end')
                        self.filename = ''
                        to_be_saved = self.audio[start : end]
                        to_be_saved.export(filename,format =
format_[ext[1:]])
                        self.status.config(text =
'Success..!',bg = '#2a8d12')
                    except:
                        messagebox.showerror(title =
'Error...',
```

```
                        message = 'Enter the correct values
for the To and From..!')
                except:
                    self.status.config(text = 'Failed..!',bg
= '#ee3456')
                    messagebox.showerror(title =
'Error...',message = 'Incompatible file..!')
            self.label.config(text = '---Empty Selection---')
            self.display_length.config(text = 'Duration:
00:00')
            self.to_spin.delete(0,'end')
            self.from_spin.delete(0,'end')
        else:
                messagebox.showerror(title =
'Error...',message = 'Select a file to Trim..!')

        self.thread -= 1
        self.queue.config(text = f'Queued: {self.thread}',bg
= '#2a8d12')
        if(self.thread == 0):
            time.sleep(1)
            self.queue.config(bg = '#ae34d9')
            self.status.config(text = 'No process..!',bg =
'#ae34d9')
```

Code for a feature process

Dependencies And System Requirements

This program can be run on any of the following systems:

1. Windows
2. Mac OS
3. Linux

Provided the following dependencies are installed on the given system:

- Python (language used for coding the project)
- Tkinter (a built-in Python GUI library)
- Moviepy (a library based on ffmpeg to provide video features)
- Pydub (a library based on ffmpeg to provide audio features
- FFmpeg (for the successful run of the whole project)
- pdfminer (for extracting text from pdf files)
- PyPDF2 (to work with pdf files)

The project has been developed and tested on Ubuntu 20.04 but as the libraries as well as Python itself is cross platform it can be assumed that the program will successfully run on other systems too given the dependencies are installed on the target system.

Conclusion

While working on this project I learnt about various things like the importance of a good community in building and supporting new things. I would often get stuck and I could just put a question on **StackOverflow** and they were there to help.
I had the same experience with **Github Issues**.

Secondly, up until now I would often code either in a **modular programming** way or a **procedural programming** methodology and while learning **Object Oriented Programming** I could never get a hang of it or even understand what importance it actually had.
But while Coding when I saw that OOP could reduce the code in a significant manner and would also help me reuse my previously written code with an additional benefits of easy maintenance I could finally say it is worth it to learn Object Oriented Programming but yeah,for sure it can't be used at each and every problem.

And lastly, the importance of **open source projects.** I would often wonder why people give away and maintain their code for free..? Why would they churn their head to code something they won't be earning from..? But now I know that if it weren't for these people many easily accessible softwares won't be available to common people. Thanks to them, various softwares is accessible freely.
Also, youngsters like me can learn about good coding practices by reading thier code.

Amppper
(more features to be added)

Submitted in partial fulfilment of the requirements
for the award of the degree of
Bachelor of Computer Applications

Guide

Submitted by
Ajay Singh Rana

Submitted To

D.S.B. Campus, Kumaun University,Nainital (U.K.)

Acknowledgement

Getting stuck in the process of learning and building something is kind of a common scenario and so did happen to me a lot of times while working on this project which on the submission date too remains under development given my plans to add more features to the project.

While getting struck is frustrating it teaches us to solve problems with grit and so does the **StackOverflow** community where I found answers to most of my problems. The same goes for documentations and **github issues** where we can get help directly from project managers and contributors.

YouTube was the starting point where I learnt some basic GUI development and would like to attribute a channel in particular i.e. **buildwithpython**.

It is also important to mention my **parents** and **teachers** who took faith in me and guided and taught me to achieve.

CERTIFICATE

This is to certify that this project entitled **“ Amppper”** submitted in partial fulfillment of the degree of Bachelor of Computer Applications to the **Dr. Ashish Mehta** through D.S.B.Campus,
done by Mr./Ms. ***Ajay Singh Rana***, Roll No. ***180125330014*** is an is an authentic work carried out by him/her at D.S.B.Campus
under my guidance. The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Signature of the guide

SELF CERTIFICATE

This is to certify that the dissertation/project report entitled **Amppper** done by me is an authentic work carried out for the partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Applications under the guidance of
Dr. Ashish Mehta. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Ajay Singh Rana

Roll No. 180125330014

D.S.B. Campus,
Kumaun University,Nainital(U.K.)

Synopsis

This project is about creating a Desktop application which can be used as a toolset to work on multimedia and PDF files.
As of now the following features for respective media have been successfully added to the project:

- **Audio**
    1. Convert audio file formats
    2. Join audio files
    3. Trim audio files
- **Video**
    1. Join Video Files
    2. Extract Audio from Video Files
    3. Mute a Video File
    4. Trim Video Files
- **PDF**
    1. Split PDF into pages
    2. Extract text from PDF

Following media and features will be added to this project once i'am able to develop those:

- **PDF**
    1. Read a PDF
- **Images**
    1. Rotate
    2. Change format
    3. Add Filters
    4. Create Thumbnails
    5. Resize

The Problem

I would often find myself searching online for tools to work around with audio,video,pdf and images and I saw most of the tools were too heavy and other online tools required an internet connection.

Meanwhile in my quest to find solutions for my problem i came across **ffmpeg** a command line tool used for audio/video/image encoding and decoding I found it interesting but was harder for me to grasp but after a few more searches I found that python had libraries which encapsulated the functions of **ffmpeg** to provide a simpler interface.

Therefore I set out to build a GUI application with some basic functionalities that I often would look up the internet for.

Objective and Scope of the project

When I started out my objective was to create a GUI application which would be able to do the following:

1. Trim an audio file
2. Change the audio file format
3. Join two audio files

But later on the project grew to add some Video and PDF features:

1. Join two Video Files
2. PDF to text file
3. Mute a Video
4. Extract Audio From Video

As of now as the project has been created to solve a personal problem I don't see a wider scope for the project itself.
But as I am keen to add more features to it I hope it can be a lightweight alternative to heavy Softwares that provides tools for single media files and even though these are efficient and have more tools to offer most of these options are either never utilised or under utilized.
I focus on basic functionalities for the different types of media which are frequently used by people.

I plan to add the following features:

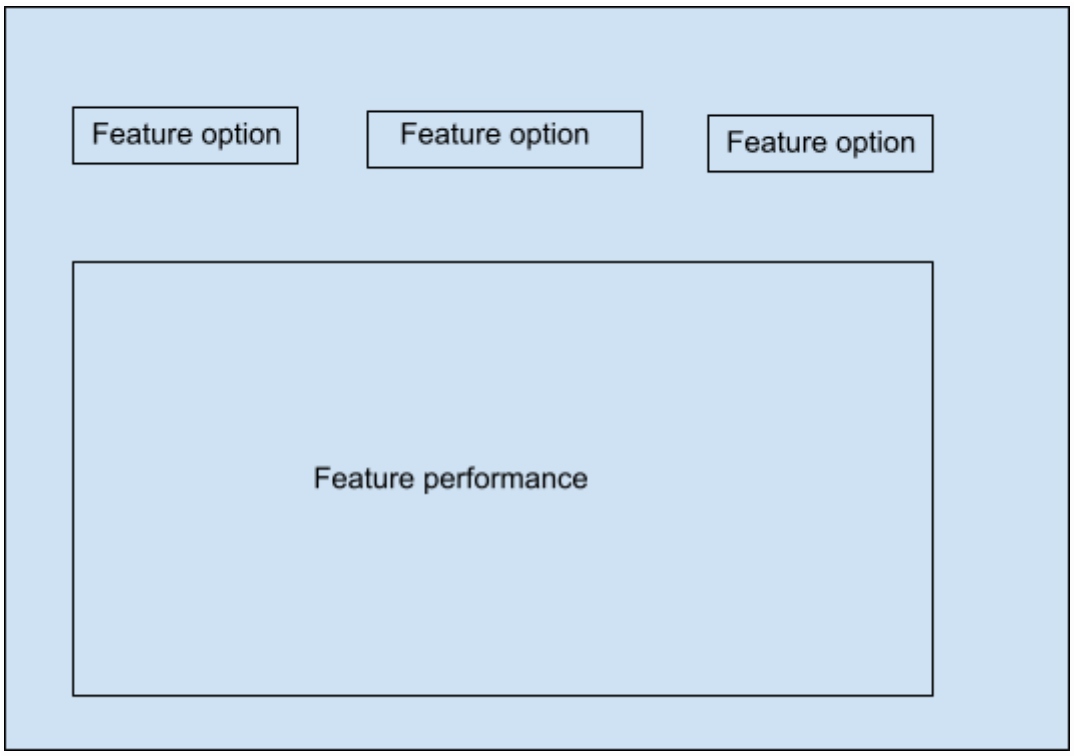
- Add a feature to Read a PDF file
- Audio Recorder
- Feature for creating PDF with command like interface within the GUI
- Basic Image options like rotate,resize,create thumbnailing and filters
- Screenshot and screen recording features
- A simple camera and recorder using openCV

How did the project progress..?

Firstly, I had to give my project an interface and I chose a **GUI** over **CLI.**
Python comes packed with **Tkinter** which is a well documented library and therefore I head on with it.
The next problem was to find libraries to provide the functionalities with and I chose to go ahead with the following:

- **Pydub**
  A python library that ***Manipulates audio with a simple and easy high level interface***
- **MoviePy**
  MoviePy is a Python module for video editing, which can be used for basic operations (like cuts, concatenations, title insertions)
- **PyPDF2**
  To work with pdf's
- **pdfminer**
  PDFMiner is a text extraction tool for PDF documents.

After having chosen the libraries I set out to firstly devise a GUI for the project. And after trying out multiple variants I came up with a single window layout for all features so that it is easier to familiarise with the interface.



Feature interface

Main Window

Code organisation

Organising the code is important too and that's because in a larger project it gets harder and sometimes impossible to maintain the code or even change the code.
At first my code wasn't organised as well and as the project grew I started facing serious problems maintaining my code as it had already grown clumsy. So, I revamped it all and redesigned my code.
Firstly I figured the main issues:

- **Redundant code:**
  I would have to create several Buttons,Labels separately for each Feature I would add which was kind of just increasing the number of lines in the code and wasn't helping at all.Writing a whole new class for adding a new feature with the same old Buttons and labels seemed trivial to me.
- **Tedious Changes:**
  Now each time I wanted to make a change to the look of Buttons I would have to change the code line for each and every button that I implemented. Which seemed frustrating.

How I Changed it All..?

I had learnt about Inheritance, so I created a class titled **CFrame** which would be inherited by all classes implementing a new feature and would contain all GUI widgets commonly shared by a Feature Frame.

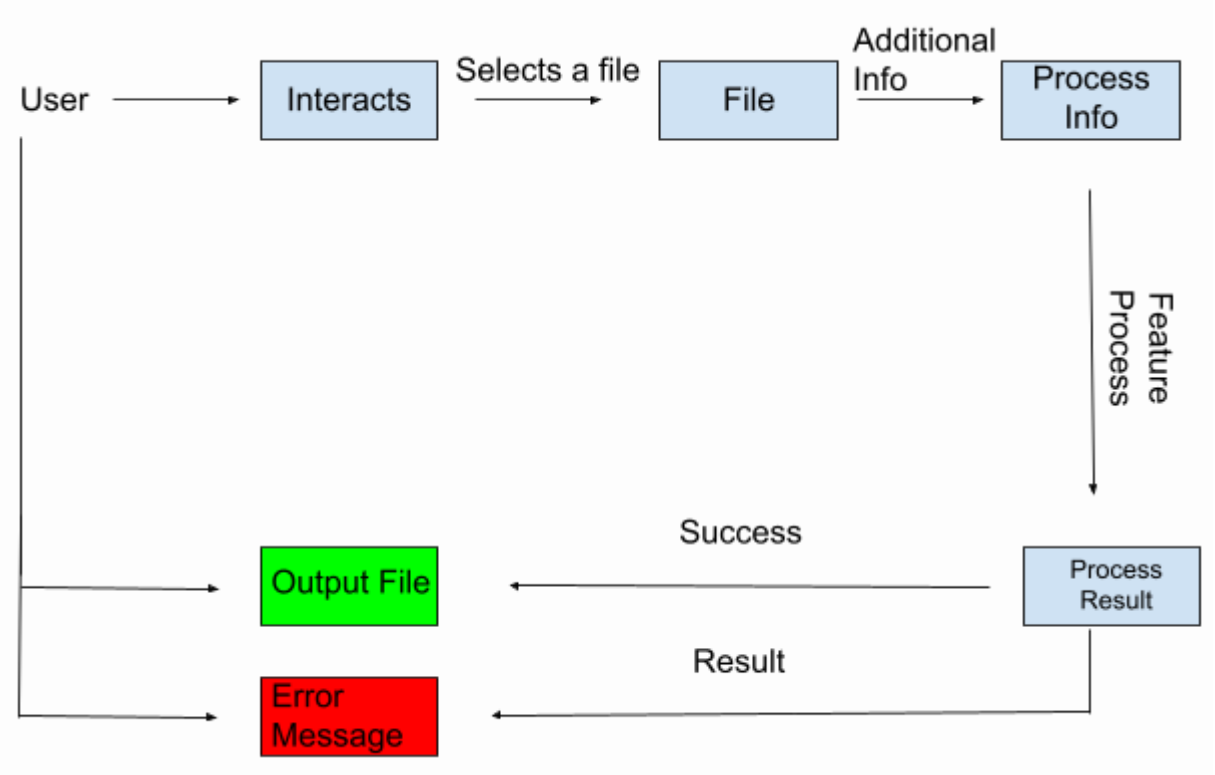
So, I did for the window frames which would list options for different file types.I created a **StandardWindow** class to implement new Frames.

```
class CFrame(tk.Frame):
    """
    This class implements the frame
    that i'll be using in different tool windows
    to display the different functionalities that
    can be used under that tool
    """
    def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
        self.filename = ''
        self.thread = 0
        self.controller = controller
        self.config(bg = '#2AA2BC')

    def browse(self,present = 1):
        self.filename =
filedialog.askopenfilename()
        if(self.filename and present):
            self.label.config(text =
os.path.split(self.filename)[1])

    def run_thread(self,target_func):
        self.thread += 1
        thread = threading.Thread(target =
target_func)
        thread.start()
```

CFrame Class that all other feature classes will inherit from.

How are the features supposed to work..?

Even though it is a GUI program it is not extensive but lightweight and therefore the features would work purely as the command-line tools would do i.e. based on the user input.
Suppose we have to trim an audio file then the user is supposed to select a file for that and would provide the seconds between which the audio is to be trimmed and then the program will work on it in the background and inform the user if the process was successful or not.



Program Workflow

```
def trim(self):
        if(self.filename):
            ext = os.path.splitext(self.filename)[1]
            filename =
filedialog.asksaveasfilename(defaultextension = ext,filetypes
= ((f'{ext} files',f'*{ext}'),))
            if(filename):
                try:
                    format_ =
{'mp3':'mp3','flv':'flv','ogg':'ogg','wav':'wav','wma':'wma',

'avi':'avi','aac':'adts','m4a':'mp4','flac':'flac','opus':'op
us','au':'au',
                                'aiff':'aiff','webm':'webm'}
                    try:
                        start =
int(float(self.from_spin.get()) * 1000)
                        end = int(float(self.to_spin.get()) *
1000)
                        if(start > end):
                            raise

                        self.status.config(text =
'Processing...',bg = '#2a8d12')
                        self.queue.config(text = f'Queued:
{self.thread}',bg = '#2a8d12')
                        self.label.config(text = '---Empty
Selection---')
                        self.to_spin.delete(0,'end')
                        self.from_spin.delete(0,'end')
                        self.filename = ''
                        to_be_saved = self.audio[start : end]
                        to_be_saved.export(filename,format =
format_[ext[1:]])
                        self.status.config(text =
'Success..!',bg = '#2a8d12')
                    except:
                        messagebox.showerror(title =
'Error...',
```

```
                        message = 'Enter the correct values
for the To and From..!')
                except:
                    self.status.config(text = 'Failed..!',bg
= '#ee3456')
                    messagebox.showerror(title =
'Error...',message = 'Incompatible file..!')
            self.label.config(text = '---Empty Selection---')
            self.display_length.config(text = 'Duration:
00:00')
            self.to_spin.delete(0,'end')
            self.from_spin.delete(0,'end')
        else:
                messagebox.showerror(title =
'Error...',message = 'Select a file to Trim..!')

        self.thread -= 1
        self.queue.config(text = f'Queued: {self.thread}',bg
= '#2a8d12')
        if(self.thread == 0):
            time.sleep(1)
            self.queue.config(bg = '#ae34d9')
            self.status.config(text = 'No process..!',bg =
'#ae34d9')
```

Code for a feature process

Dependencies And System Requirements

This program can be run on any of the following systems:

1. Windows
2. Mac OS
3. Linux

Provided the following dependencies are installed on the given system:

- Python (language used for coding the project)
- Tkinter (a built-in Python GUI library)
- Moviepy (a library based on ffmpeg to provide video features)
- Pydub (a library based on ffmpeg to provide audio features
- FFmpeg (for the successful run of the whole project)
- pdfminer (for extracting text from pdf files)
- PyPDF2 (to work with pdf files)

The project has been developed and tested on Ubuntu 20.04 but as the libraries as well as Python itself is cross platform it can be assumed that the program will successfully run on other systems too given the dependencies are installed on the target system.

Conclusion

While working on this project I learnt about various things like the importance of a good community in building and supporting new things. I would often get stuck and I could just put a question on **StackOverflow** and they were there to help.
I had the same experience with **Github Issues**.

Secondly, up until now I would often code either in a **modular programming** way or a **procedural programming** methodology and while learning **Object Oriented Programming** I could never get a hang of it or even understand what importance it actually had.
But while Coding when I saw that OOP could reduce the code in a significant manner and would also help me reuse my previously written code with an additional benefits of easy maintenance I could finally say it is worth it to learn Object Oriented Programming but yeah,for sure it can't be used at each and every problem.

And lastly, the importance of **open source projects.** I would often wonder why people give away and maintain their code for free..? Why would they churn their head to code something they won't be earning from..? But now I know that if it weren't for these people many easily accessible softwares won't be available to common people. Thanks to them, various softwares is accessible freely.
Also, youngsters like me can learn about good coding practices by reading thier code.

Amppper
(more features to be added)

Submitted in partial fulfilment of the requirements
for the award of the degree of
Bachelor of Computer Applications

Guide

Submitted by
Ajay Singh Rana

Submitted To

D.S.B. Campus, Kumaun University,Nainital (U.K.)

Acknowledgement

Getting stuck in the process of learning and building something is kind of a common scenario and so did happen to me a lot of times while working on this project which on the submission date too remains under development given my plans to add more features to the project.

While getting struck is frustrating it teaches us to solve problems with grit and so does the **StackOverflow** community where I found answers to most of my problems. The same goes for documentations and **github issues** where we can get help directly from project managers and contributors.

YouTube was the starting point where I learnt some basic GUI development and would like to attribute a channel in particular i.e. **buildwithpython**.

It is also important to mention my **parents** and **teachers** who took faith in me and guided and taught me to achieve.

CERTIFICATE

This is to certify that this project entitled **“ Amppper”** submitted in partial fulfillment of the degree of Bachelor of Computer Applications to the **Dr. Ashish Mehta** through D.S.B.Campus,
done by Mr./Ms. ***Ajay Singh Rana***, Roll No. ***180125330014*** is an is an authentic work carried out by him/her at D.S.B.Campus
under my guidance. The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Signature of the guide

SELF CERTIFICATE

This is to certify that the dissertation/project report entitled **Amppper** done by me is an authentic work carried out for the partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Applications under the guidance of
Dr. Ashish Mehta. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Signature of the student

Ajay Singh Rana

Roll No. 180125330014

D.S.B. Campus,
Kumaun University,Nainital(U.K.)

Synopsis

This project is about creating a Desktop application which can be used as a toolset to work on multimedia and PDF files.
As of now the following features for respective media have been successfully added to the project:

- **Audio**
    1. Convert audio file formats
    2. Join audio files
    3. Trim audio files
- **Video**
    1. Join Video Files
    2. Extract Audio from Video Files
    3. Mute a Video File
    4. Trim Video Files
- **PDF**
    1. Split PDF into pages
    2. Extract text from PDF

Following media and features will be added to this project once i'am able to develop those:

- **PDF**
    1. Read a PDF
- **Images**
    1. Rotate
    2. Change format
    3. Add Filters
    4. Create Thumbnails
    5. Resize

The Problem

I would often find myself searching online for tools to work around with audio,video,pdf and images and I saw most of the tools were too heavy and other online tools required an internet connection.

Meanwhile in my quest to find solutions for my problem i came across **ffmpeg** a command line tool used for audio/video/image encoding and decoding I found it interesting but was harder for me to grasp but after a few more searches I found that python had libraries which encapsulated the functions of **ffmpeg** to provide a simpler interface.

Therefore I set out to build a GUI application with some basic functionalities that I often would look up the internet for.

Objective and Scope of the project

When I started out my objective was to create a GUI application which would be able to do the following:

1. Trim an audio file
2. Change the audio file format
3. Join two audio files

But later on the project grew to add some Video and PDF features:

1. Join two Video Files
2. PDF to text file
3. Mute a Video
4. Extract Audio From Video

As of now as the project has been created to solve a personal problem I don't see a wider scope for the project itself.
But as I am keen to add more features to it I hope it can be a lightweight alternative to heavy Softwares that provides tools for single media files and even though these are efficient and have more tools to offer most of these options are either never utilised or under utilized.
I focus on basic functionalities for the different types of media which are frequently used by people.

I plan to add the following features:

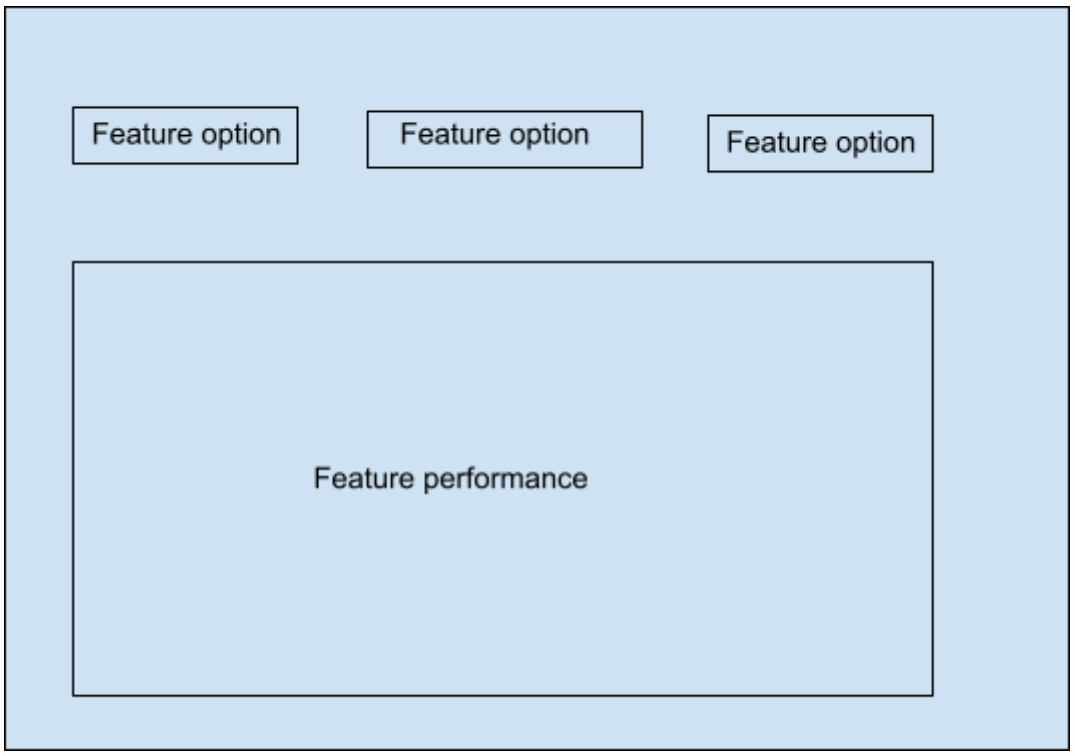
- Add a feature to Read a PDF file
- Audio Recorder
- Feature for creating PDF with command like interface within the GUI
- Basic Image options like rotate,resize,create thumbnailing and filters
- Screenshot and screen recording features
- A simple camera and recorder using openCV

How did the project progress..?

Firstly, I had to give my project an interface and I chose a **GUI** over **CLI.**
Python comes packed with **Tkinter** which is a well documented library and therefore I head on with it.
The next problem was to find libraries to provide the functionalities with and I chose to go ahead with the following:

- **Pydub**
  A python library that ***Manipulates audio with a simple and easy high level interface***
- **MoviePy**
  MoviePy is a Python module for video editing, which can be used for basic operations (like cuts, concatenations, title insertions)
- **PyPDF2**
  To work with pdf's
- **pdfminer**
  PDFMiner is a text extraction tool for PDF documents.

After having chosen the libraries I set out to firstly devise a GUI for the project. And after trying out multiple variants I came up with a single window layout for all features so that it is easier to familiarise with the interface.



Feature interface

Main Window

Code organisation

Organising the code is important too and that's because in a larger project it gets harder and sometimes impossible to maintain the code or even change the code.
At first my code wasn't organised as well and as the project grew I started facing serious problems maintaining my code as it had already grown clumsy. So, I revamped it all and redesigned my code.
Firstly I figured the main issues:

- **Redundant code:**
  I would have to create several Buttons,Labels separately for each Feature I would add which was kind of just increasing the number of lines in the code and wasn't helping at all.Writing a whole new class for adding a new feature with the same old Buttons and labels seemed trivial to me.
- **Tedious Changes:**
  Now each time I wanted to make a change to the look of Buttons I would have to change the code line for each and every button that I implemented. Which seemed frustrating.

How I Changed it All..?

I had learnt about Inheritance, so I created a class titled **CFrame** which would be inherited by all classes implementing a new feature and would contain all GUI widgets commonly shared by a Feature Frame.

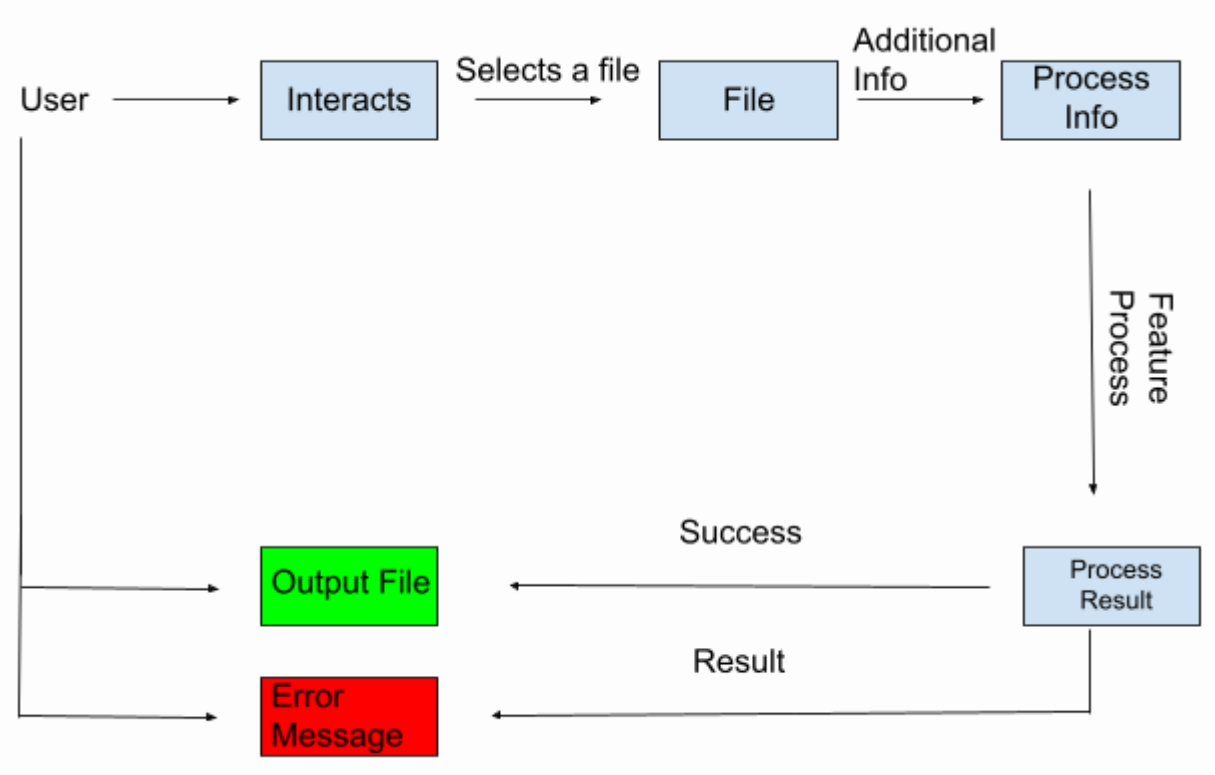
So, I did for the window frames which would list options for different file types.I created a **StandardWindow** class to implement new Frames.

```
class CFrame(tk.Frame):
    """
    This class implements the frame
    that i'll be using in different tool windows
    to display the different functionalities that
    can be used under that tool
    """
    def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
        self.filename = ''
        self.thread = 0
        self.controller = controller
        self.config(bg = '#2AA2BC')

    def browse(self,present = 1):
        self.filename =
filedialog.askopenfilename()
        if(self.filename and present):
            self.label.config(text =
os.path.split(self.filename)[1])

    def run_thread(self,target_func):
        self.thread += 1
        thread = threading.Thread(target =
target_func)
        thread.start()
```

CFrame Class that all other feature classes will inherit from.

How are the features supposed to work..?

Even though it is a GUI program it is not extensive but lightweight and therefore the features would work purely as the command-line tools would do i.e. based on the user input.
Suppose we have to trim an audio file then the user is supposed to select a file for that and would provide the seconds between which the audio is to be trimmed and then the program will work on it in the background and inform the user if the process was successful or not.



Program Workflow

```
def trim(self):
        if(self.filename):
            ext = os.path.splitext(self.filename)[1]
            filename =
filedialog.asksaveasfilename(defaultextension = ext,filetypes
= ((f'{ext} files',f'*{ext}'),))
            if(filename):
                try:
                    format_ =
{'mp3':'mp3','flv':'flv','ogg':'ogg','wav':'wav','wma':'wma',

'avi':'avi','aac':'adts','m4a':'mp4','flac':'flac','opus':'op
us','au':'au',
                                'aiff':'aiff','webm':'webm'}
                    try:
                        start =
int(float(self.from_spin.get()) * 1000)
                        end = int(float(self.to_spin.get()) *
1000)
                        if(start > end):
                            raise

                        self.status.config(text =
'Processing...',bg = '#2a8d12')
                        self.queue.config(text = f'Queued:
{self.thread}',bg = '#2a8d12')
                        self.label.config(text = '---Empty
Selection---')
                        self.to_spin.delete(0,'end')
                        self.from_spin.delete(0,'end')
                        self.filename = ''
                        to_be_saved = self.audio[start : end]
                        to_be_saved.export(filename,format =
format_[ext[1:]])
                        self.status.config(text =
'Success..!',bg = '#2a8d12')
                    except:
                        messagebox.showerror(title =
'Error...',
```

```
                        message = 'Enter the correct values
for the To and From..!')
                except:
                    self.status.config(text = 'Failed..!',bg
= '#ee3456')
                    messagebox.showerror(title =
'Error...',message = 'Incompatible file..!')
            self.label.config(text = '---Empty Selection---')
            self.display_length.config(text = 'Duration:
00:00')
            self.to_spin.delete(0,'end')
            self.from_spin.delete(0,'end')
        else:
                messagebox.showerror(title =
'Error...',message = 'Select a file to Trim..!')

        self.thread -= 1
        self.queue.config(text = f'Queued: {self.thread}',bg
= '#2a8d12')
        if(self.thread == 0):
            time.sleep(1)
            self.queue.config(bg = '#ae34d9')
            self.status.config(text = 'No process..!',bg =
'#ae34d9')
```

Code for a feature process

Dependencies And System Requirements

This program can be run on any of the following systems:

1. Windows
2. Mac OS
3. Linux

Provided the following dependencies are installed on the given system:

- Python (language used for coding the project)
- Tkinter (a built-in Python GUI library)
- Moviepy (a library based on ffmpeg to provide video features)
- Pydub (a library based on ffmpeg to provide audio features
- FFmpeg (for the successful run of the whole project)
- pdfminer (for extracting text from pdf files)
- PyPDF2 (to work with pdf files)

The project has been developed and tested on Ubuntu 20.04 but as the libraries as well as Python itself is cross platform it can be assumed that the program will successfully run on other systems too given the dependencies are installed on the target system.

Conclusion

While working on this project I learnt about various things like the importance of a good community in building and supporting new things. I would often get stuck and I could just put a question on **StackOverflow** and they were there to help.
I had the same experience with **Github Issues**.

Secondly, up until now I would often code either in a **modular programming** way or a **procedural programming** methodology and while learning **Object Oriented Programming** I could never get a hang of it or even understand what importance it actually had.
But while Coding when I saw that OOP could reduce the code in a significant manner and would also help me reuse my previously written code with an additional benefits of easy maintenance I could finally say it is worth it to learn Object Oriented Programming but yeah,for sure it can't be used at each and every problem.

And lastly, the importance of **open source projects.** I would often wonder why people give away and maintain their code for free..? Why would they churn their head to code something they won't be earning from..? But now I know that if it weren't for these people many easily accessible softwares won't be available to common people. Thanks to them, various softwares is accessible freely.
Also, youngsters like me can learn about good coding practices by reading thier code.