

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP LỚN
HỌC PHẦN: LẬP TRÌNH VỚI PYTHON
Đề tài: Phát triển ứng dụng Music Player

Nhóm lớp học: 01

Nhóm bài tập: 05

Thành viên nhóm:

- 1. Nguyễn Đức Linh - B20DCAT109**
- 2. Nguyễn Đăng Hạnh - B20DCAT053**
- 3. Đỗ Ngọc Huế - B20DCAT073**
- 4. Nguyễn Việt Anh - B20DCAT010**

Giảng viên hướng dẫn: Vũ Minh Mạnh

1. Giới thiệu đề tài

Nghe nhạc trên máy tính/laptop là một trong nhu cầu giải trí phổ biến của người dùng, nó giúp thư giãn, phục vụ nhu cầu giải trí và giúp cho quá trình học tập hay làm việc hiệu quả hơn. Ngày nay các trình nghe nhạc trực tuyến trên các website hay các ứng dụng nghe nhạc offline khá phổ biến đối với người dùng máy tính. Người dùng nếu như không thể truy cập internet thường xuyên, có thể tải các bản nhạc yêu thích về máy tính của mình để nghe bất cứ khi nào dù cho không có kết nối mạng. Việc xây dựng ứng dụng phát nhạc cá nhân giúp người dùng có thể cá nhân hóa, trải nghiệm có thể thú vị hơn mỗi khi nghe nhạc. Chẳng hạn như có thể tùy biến giao diện máy phát nhạc theo ý thích của mình, hiển thị trên giao diện các thông tin về bài hát họ yêu thích, chọn lựa bài hát cần nghe một cách dễ dàng, nhanh chóng...

Xuất phát từ nhu cầu trên, với mong muốn sáng tạo theo ý muốn, nhóm đã quyết định chọn chủ đề bài tập lớn là phát triển ứng dụng trình phát nhạc offline Music Player. Ứng dụng có đầy đủ những chức năng cơ bản của một chương trình nghe nhạc offline đáp ứng đủ nhu cầu của người dùng.

2. Lựa chọn ngôn ngữ lập trình và mô hình phát triển

Mô hình MVC

Hệ thống được thiết kế theo mô hình MVC. MVC – Model View Controller là một cấu trúc bao gồm Model có tác dụng hỗ trợ lưu trữ các logic dữ liệu, View là chế độ xem của GUI và cuối cùng Controller là bộ não của ứng dụng giúp kiểm soát cách các dữ liệu được hiển thị. Cụ thể, Controller được sử dụng để liên kết giữa View và Model. Việc phát triển theo mô hình MVC giúp đơn giản hóa quá trình quản lý và cho phép các nhà phát triển có thể thực hiện nhiều công việc trên ứng dụng.

Ngôn ngữ Python

Python là ngôn ngữ mã nguồn mở, được phát triển và duy trì bởi một cộng đồng lớn các lập trình viên trên toàn thế giới. Python có thư viện và framework rất phong phú, giúp cho việc lập trình trở nên nhanh chóng và dễ dàng hơn. Ngoài ra, Python còn có cú pháp đơn giản, dễ hiểu, giúp các lập trình viên mới bắt đầu học lập trình dễ dàng tiếp cận.

Python là ngôn ngữ lập trình đa năng, có thể được sử dụng để xây dựng nhiều loại ứng dụng khác nhau, từ các ứng dụng desktop đơn giản đến các ứng dụng web phức tạp, các trò chơi, máy học, trí tuệ nhân tạo, xử lý ngôn ngữ tự nhiên, phân tích dữ liệu và nhiều ứng dụng khác.

Với bài tập lớn này nhóm sử dụng những thư viện sau phục vụ cho phát triển ứng dụng:

Pygame

Pygame là một thư viện của ngôn ngữ Python được sử dụng để phát triển trò chơi 2D và là một nền tảng nơi bạn có thể sử dụng một tập hợp các modul Python để phát triển trò chơi. Thư viện này chứa đầy đủ các công cụ hỗ trợ lập trình game như đồ hoạt, hoạt hình, âm thanh, và sự kiện điều khiển.

Tkinter

Tkinter là một thư viện giao diện đồ họa (GUI) phổ biến được sử dụng trong Python. Tkinter cung cấp các widget đồ họa để tạo ra giao diện cho ứng dụng, bao gồm các nút, hộp văn bản, menu, hộp chọn và nhiều hơn nữa. Tkinter cũng hỗ trợ định dạng, phông chữ, màu sắc, kích thước và định dạng hình ảnh. Tkinter cho phép tạo ra các ứng dụng đồ họa đơn giản và trực quan trên nền tảng máy tính.

PIL

PIL (Python Imaging Library) là một thư viện Python sử dụng để xử lý hình ảnh. Thư viện này cho phép các thao tác xử lý ảnh phổ biến như cắt, chỉnh sửa kích thước, chuyển đổi định dạng, đóng dấu, tạo hiệu ứng, v.v. Tuy nhiên, thư viện này đã ngừng phát triển kể từ năm 2011 và được thay thế bằng thư viện Pillow, một bản fork của PIL với các cải tiến và bổ sung mới.

Ttkthemes

Ttkthemes là một thư viện Python cung cấp giao diện người dùng được thiết kế bằng thư viện Widget Tkinter với nhiều chủ đề khác nhau. Nó cho phép người dùng tùy chỉnh các thành phần giao diện người dùng như button, textbox, progressbar,... theo các chủ đề khác nhau để tạo ra các ứng dụng có giao diện đẹp và chuyên nghiệp hơn. Ttkthemes cung cấp nhiều chủ đề đẹp và phù hợp với nhiều phong cách thiết kế khác nhau, từ phong cách truyền thống đến phong cách hiện đại.

Mixer

Thư viện mixer trong pygame cung cấp các tính năng liên quan đến xử lý âm thanh, bao gồm phát và kiểm soát âm lượng các tệp âm thanh, cho phép các lập trình viên tạo ra các ứng dụng liên quan đến xử lý âm thanh như trình phát nhạc, trò chơi âm thanh, ứng dụng ghi âm và xử lý âm thanh real-time.

Pyodbc

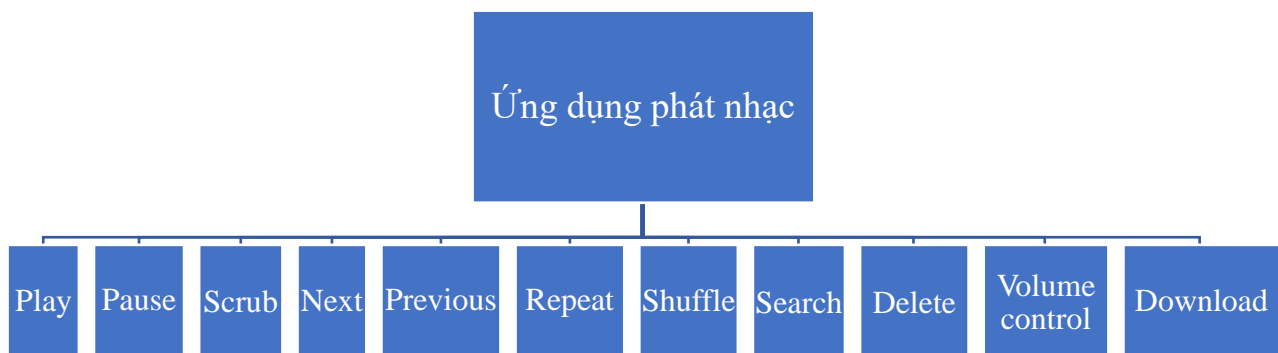
Pyodbc là một thư viện mã nguồn mở của Python giúp truy cập cơ sở dữ liệu ODBC trở nên đơn giản. Pyodbc triển khai đặc tả DB API 2.0. Để cài đặt thư viện này chúng ta sử dụng công cụ quản lý package của Python là PIP.

Os

Thư viện os của Python cung cấp các hàm để tương tác với hệ điều hành. Thư viện này cho phép bạn thực hiện các thao tác như tạo thư mục mới, xóa thư mục, đổi tên file/thư mục, lấy danh sách các file/thư mục trong một thư mục cụ thể. Để sử dụng thư viện này bạn không cần phải cài đặt gì thêm.

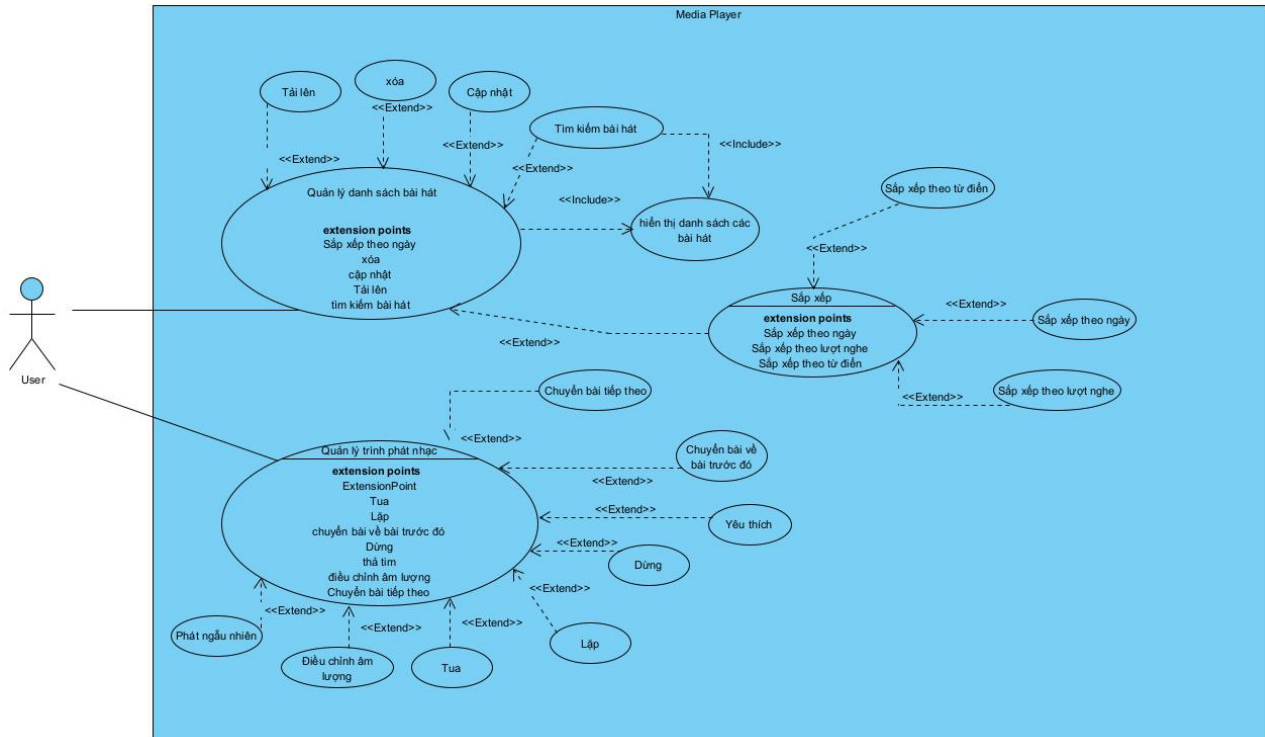
3. Thiết kế hệ thống

3.1 Sơ đồ các chức năng cơ bản chính



- Chức năng Play: cho phép người dùng phát bài hát được chọn
- Chức năng Pause: cho phép người dùng tạm dừng bài hát đang phát
- Chức năng Scrub: cho phép người dùng tua bài hát đến đoạn bất kỳ của bài hát đang phát
- Chức năng Next: cho phép người dùng phát bài hát ngay sau bài hát đang phát có trong danh sách.
- Chức năng Previous: cho phép người dùng quay lại bài hát ngay trước bài hát đang phát có trong danh sách.
- Chức năng Repeat: cho phép người dùng lặp đi lặp lại bài hát đang phát.
- Chức năng Shuffle: cho phép người dùng phát bài hát ngẫu nhiên có trong danh sách.
- Chức năng Search: cho phép người dùng tìm kiếm bài hát có trong danh sách.
- Chức năng Delete: cho phép người dùng xóa bài hát được chọn
- Chức năng Volume control: cho phép người dùng bật/ tắt hay tăng/giảm âm lượng.
- Chức năng Download: cho phép người dùng tải bài hát và thêm vào danh sách của ứng dụng.

3.2 Mô hình tổng quan hệ thống ứng dụng Music player



3.3 Cơ sở dữ liệu của ứng dụng

Việc kết nối ứng dụng nghe nhạc với database có thể giúp cải thiện trải nghiệm nghe nhạc của người dùng bằng cách cho phép họ lưu trữ và quản lý các bài hát, danh sách phát và thông tin bài hát trên một cơ sở dữ liệu. Điều này giúp cho việc tìm kiếm và phát nhạc trở nên dễ dàng hơn, đồng thời cho phép người dùng lưu trữ thông tin về các bài hát yêu thích của họ. Ngoài ra, việc kết nối với database còn cho phép ứng dụng nghe nhạc cung cấp các tính năng như tìm kiếm, lọc và sắp xếp bài hát theo các tiêu chí nhất định.

Column Name	Data Type	Allow Nulls
id_song	nvarchar(200)	<input type="checkbox"/>
name_song	nvarchar(200)	<input checked="" type="checkbox"/>
singer	nvarchar(200)	<input checked="" type="checkbox"/>
date_download	date	<input checked="" type="checkbox"/>
love	int	<input checked="" type="checkbox"/>
listen	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

4. Các modul chính

4.1 Lớp View

```

view.py x Model.py Controller.py
Project > source > view.py > View > play_and_show_song
22
23 class View(Frame):
24
25     name_song = '...' #Tên bài đang phát
26     singer = '.....' #Tên ca sĩ đang phát
27     check = False # Kiểm tra pause
28     song_now = '' #Link mp3
29     current_list = list() #Danh sách hiện tại
30     love = False #Trạng thái bài hát yêu thích?
31     repeat = True #Phát lại
32     dem = 0
33     dem_shuffle = 0
34     check_lovelist = True #Kiểm tra lovelist đã được bật?
35     frames = [] #Lưu ảnh gif
36     check_gui2 = False #Kiểm tra Gui2 đã được bật?
37     check_scale = True
38     current_time = 0
39     check_shuffle_song = False
40     check_header_playlist = ''
41     a = [1,1,1,1,1]
42
43     #Khởi tạo
44     def __init__(self,parent,controller) -> None:
45         Frame.__init__(self,parent)
46         self.controller = controller
47         # self.parent = parent
48
49         #Lưu ảnh gif
50         self.gif = Image.open("../Project/images/gif.gif")
51         for frame in range(0, self.gif.n_frames):
52             self.gif.seek(frame)
53             View.frames.append(ImageTk.PhotoImage(self.gif))
54
55         self.init()
56
57     # Playlist frame
58     def init_frame_main(self):
59         # Frame playlist

```

❖ Chi tiết các nút/chức năng của ứng dụng

Chức năng Play/Pause

```

# Play/Pause button
self.play_button = Button(self.Header,
                           bd=0,
                           cursor='hand2',
                           bg="#222222",
                           activebackground='#222222',
                           command=self.pause_song)

if View.check == False: # Nếu nhạc dừng
    self.play_icon=PhotoImage(file='../Project/images/pause1.png')
    self.play_button.config(image=self.play_icon)
else:
    self.play_icon=PhotoImage(file='../Project/images/pause.png')
    self.play_button.config(image=self.play_icon)
self.play_button.place(x=525,y=125,height=40,width=40)

```

```

# Pause and play song
def pause_song(self):
    if len(View.song_now)>0:
        if View.check == True:
            self.after_cancel(self.updater)
            mixer.music.pause()
            self.play_icon=PhotoImage(file='../Project/images/pause1.png')
            self.play_button.config(image=self.play_icon)
            View.check = False
        else:
            self.play_icon=PhotoImage(file='../Project/images/pause.png')
            self.play_button.config(image=self.play_icon)
            mixer.music.unpause()
            self.scale_update()
            View.check = True

```

Chức năng Scub

```

# Thanh phát nhạc
self.style.configure("greyish.Horizontal.TProgressbar", troughcolor="white")
self.progress_scale = ttk.Scale(self.Header,
                                orient="horizontal",
                                style='TScale',
                                from_=0,
                                value=1,
                                length=455,
                                cursor='hand2')
self.progress_scale.place(x=280,y=90,height=5)

self.progress_scale.bind('<ButtonRelease>',self.progress_scale_moved)

self.time_elapsed_label = tk.Label(self.Header,
                                    fg="white",
                                    background="#222222",
                                    activebackground="#222222")
self.music_duration_label = tk.Label(self.Header,
                                     fg="white",
                                     background="#222222",
                                     activebackground="#222222")

self.progress_scale['value'] = View.current_time
self.time_elapsed_label.config(text=time.strftime('%M:%S', time.gmtime(self.progress_scale.get())))
if View.check_scale==True:
    self.music_duration_label.config(text="00:00")
    View.check_scale = False
else:
    if len(View.song_now)>0:
        self.music_duration_label.config(text=time.strftime('%M:%S', time.gmtime(self.music_length)))
        self.progress_scale.config(to=self.music_length)
    else:
        self.music_duration_label.config(text="00:00")
self.time_elapsed_label.place(x=240,y=83)
self.music_duration_label.place(x=740,y=83)

```

```

# Tua thời gian chạy bài hát
def progress_scale_moved(self,event):
    self.after_cancel(self.updater)
    View.scale_at=self.progress_scale.get()
    mixer.music.load(View.song_now)
    mixer.music.play(0,View.scale_at)
    self.scale_update()

```

Hiện thị thời gian chạy của bài hát trên thanh phát nhạc bài hát đó

```

# Update thời gian phát bài hát
def scale_update(self):
    global auto_play
    if self.progress_scale['value'] < self.music_length:
        self.progress_scale['value'] += 1
        View.current_time = self.progress_scale.get()
        self.time_elapsed_label['text'] = time.strftime('%M:%S', time.gmtime(View.current_time))
        self.updater = self.after(1000, self.scale_update)
    elif View.repeat==False:
        a = self.controller.current_song(View.name_song)
        self.play_and_show_song(a)
    elif auto_play == True :
        self.next_song()
    elif self.progress_scale['value'] > self.music_length:
        self.progress_scale['value'] = 0
        self.time_elapsed_label['text'] = "00:00"
        self.updater = self.after(1000, self.scale_update)

```

Chức năng Next

```

# Next button
self.next_icon = PhotoImage(file='./Project/images/next.png')
self.next_button = Button(self.Header,
                           image=self.next_icon,
                           bd=0,
                           cursor='hand2',
                           bg="#222222",
                           activebackground='#222222',
                           command=self.next_song)
self.next_button.place(x=615,y=125,height=40,width=40)

```

```

# Next song
def next_song(self):
    a = list()
    kt = True
    if len(View.song_now)>0:
        for i in range(len(View.current_list)):
            if View.song_now==View.current_list[i][0] :
                if i != len(View.current_list) - 1 :
                    a = View.current_list[i+1]
                else:
                    a = View.current_list[0]
            kt = False
            break
        if kt:
            View.current_list = list(self.controller.update_song())
            for i in range(len(View.current_list)):
                if View.song_now==View.current_list[i][0] :
                    if i != len(View.current_list) - 1 :
                        a = View.current_list[i+1]
                    else:
                        a = View.current_list[0]
                break
            self.play_and_show_song(a)

```

Chức năng Previous

```

# Previous button
self.previous_icon = PhotoImage(file='./Project/images/previous.png')
self.previous_button = Button(self.Header,
                              image=self.previous_icon,
                              bd=0,cursor='hand2',
                              bg="#222222",
                              activebackground='#222222',
                              command=self.previous_song)
self.previous_button.place(x=435,y=125,height=40,width=40)

```



```

# Previous song
def previous_song(self):
    a = list()
    kt = True
    if len(View.song_now)>0:
        for i in range(len(View.current_list)):
            if View.song_now==View.current_list[i][0] :
                if i != 0 :
                    a = View.current_list[i-1]
                else:
                    a = View.current_list[len(View.current_list)-1]
                kt = False
                break
        if kt:
            View.current_list = list(self.controller.update_song())
            for i in range(len(View.current_list)):
                if View.song_now==View.current_list[i][0] :
                    if i != 0 :
                        a = View.current_list[i-1]
                    else:
                        a = View.current_list[len(View.current_list)-1]
                    break
    self.play_and_show_song(a)

```

Chức năng Repeat

```

# Repeat button
self.again_button = Button(self.Header,
                             bd=0,
                             cursor='hand2',
                             bg="#222222",
                             activebackground='#222222',
                             command=self.repeat_song)
if View.repeat==True:
    self.again_icon = PhotoImage(file='./Project/images/stt.png')
    self.again_button.config(image=self.again_icon)
else:
    self.again_icon = PhotoImage(file='./Project/images/again.png')
    self.again_button.config(image=self.again_icon)
self.again_button.place(x=345,y=125,height=40,width=40)

```

```

# Repeat
def repeat_song(self):
    if View.repeat:
        self.again_icon = PhotoImage(file='./Project/images/again.png')
        self.again_button.config(image=self.again_icon)
        View.repeat=False
    else:
        self.again_icon = PhotoImage(file='./Project/images/stt.png')
        self.again_button.config(image=self.again_icon)
        View.repeat= True

```

Chức năng Shuffle

```

# Shuffle button
self.fix_button = Button(self.Header,
                           bd=0,
                           cursor='hand2',
                           bg="#222222",
                           activebackground='#222222',
                           command=self.shuffle_song)
if View.check_shuffle_song == False:
    self.fix_icon = PhotoImage(file='./Project/images/fix.png')
    self.fix_button.config(image=self.fix_icon)
else:
    self.fix_icon = PhotoImage(file='./Project/images/fix1.png')
    self.fix_button.config(image=self.fix_icon)
self.fix_button.place(x=705,y=125,height=40,width=40)

```

```

# Shuffle
def shuffle_song(self):
    global auto_play
    if View.check_shuffle_song == False :
        if View.dem != 0 and View.dem_shuffle != 0:
            self.after_cancel(self.updater)
            random.shuffle(View.current_list)
            a = View.current_list[0]
            self.play_and_show_song(a)
            View.dem_shuffle = 1
            self.fix_icon = PhotoImage(file='./Project/images/fix1.png')
            self.fix_button.config(image=self.fix_icon)
            View.check_shuffle_song = True
        else :
            self.fix_icon = PhotoImage(file='./Project/images/fix.png')
            self.fix_button.config(image=self.fix_icon)
            View.check_shuffle_song = False

```

Chức năng Volume control

```

# Volume button => turn up/down
self.volume_icon = PhotoImage(file="./Project/images/volume.png")
self.Volumn_button = Button(self.background_left,
                              image=self.volume_icon,
                              cursor='hand2',
                              bg="#222222",
                              bd=0,
                              font=self.f1,
                              anchor=CENTER,
                              foreground='white',
                              activebackground='#222222',
                              command=self.volumn_song)
self.Volumn_button.place(x=5,y=350,height=40,width=40)

self.style.configure("myStyle.Horizontal.TScale")
self.volu_song = ttk.Scale(self.background_left,
                             from_=0,
                             to=1,
                             value=1,
                             orient=HORIZONTAL,
                             cursor="hand2",
                             command=self.vol,
                             style="myStyle.Horizontal.TScale")
self.volu_song.place(x=45,y=370,height=4,width=100)

```

```
# Turn on/off volume
def vol(self,*args):
    global volume_start
    mixer.music.set_volume(self.volu_song.get())
    if self.volu_song.get()==0:
        self.volume_icon = PhotoImage(file="./Project/images/volume0.png")
        self.Volumn_button.config(image=self.volume_icon)
        volume_start = False
    else:
        self.volume_icon = PhotoImage(file="./Project/images/volume.png")
        self.Volumn_button.config(image=self.volume_icon)
        volume_start = True
```

```
# Turn up/down volume
def volumn_song(self):
    global volume_start
    if volume_start == True:
        self.volume_icon = PhotoImage(file="./Project/images/volume0.png")
        self.Volumn_button.config(image=self.volume_icon)
        mixer.music.set_volume(self.volu_song.set(0))
        volume_start = False
    else :
        self.volume_icon = PhotoImage(file="./Project/images/volume.png")
        self.Volumn_button.config(image=self.volume_icon)
        mixer.music.set_volume(self.volu_song.set(1))
        volume_start = True
```

Chức năng Download

```
# Download
self.download_icon = PhotoImage(file="./Project/images/download.png")
self.download = Button(self.main,
                        image=self.download_icon,
                        bd=0,
                        cursor='hand2',
                        bg='black',
                        fg='white',
                        activebackground='black',
                        command=self.download_song,
                        anchor=CENTER)
self.download.place(x=400,y=0,height=30,width=70)
```

```
def download_song(self):
    webbrowser.open("https://www.nhaccuatui.com/")
```

Chức năng Search

```

# Search button
self.search_icon = PhotoImage(file='./Project/images/search.png')
self.search_button = Button(self.search_image_label,
                             image=self.search_icon,
                             bd=0,
                             cursor='hand2',
                             bg='white',
                             highlightthickness=2,
                             command=self.search_song)
self.search_button.place(x=370,y=4,height=25,width=25)
self.search_image_label.place(x=190,y=230,height=35,width=415)

```

```

# Search song
def search(self,event): # Auto search
    if View.check_lovelist == False:
        View.current_list = self.controller.search_lovelist(self.text_var.get())
    else:
        View.current_list = self.controller.search(self.text_var.get())
    self.play_list.delete(*self.play_list.get_children())
    for i in View.current_list:
        value = [i[1],i[2],i[3],i[5]]
        self.play_list.insert('',END,values=value)
def search_song(self): # Use search button
    if View.check_lovelist == False:
        View.current_list = self.controller.search_lovelist(self.text_var.get())
    else:
        View.current_list = self.controller.search(self.text_var.get())
    self.play_list.delete(*self.play_list.get_children())
    for i in View.current_list:
        value = [i[1],i[2],i[3],i[5]]
        self.play_list.insert('',END,values=value)

```

Chức năng Delete

```

# Delete button
self.delete_icon = PhotoImage(file='./Project/images/cancel.png')
self.delete_button = Button(self.main,
                             image=self.delete_icon,
                             bd=0,
                             cursor='hand2',
                             bg='black',
                             fg='white',
                             activebackground='black',
                             command=self.delete_song)
self.delete_button.place(x=600,y=0,width=70,height=30)

```

```

# Delete a song which selected
def delete_song(self): # use delete button
    mixer.quit()
    self.controller.delete_song(View.song_now)
    self.update_playlist()
def delete(self,event): # use keyboard
    mixer.quit()
    self.controller.delete_song(View.song_now)
    self.update_playlist()

```

Chức năng Reset (reset list)

```
# Reset button => reset/update playlist
self.reset_icon = PhotoImage(file='./Project/images/reset.png')
self.reset_button = Button(self.main,
                             image=self.reset_icon,
                             bd=0,
                             cursor='hand2',
                             bg='black',
                             fg='white',
                             activebackground='black',
                             command=self.update_playlist,
                             anchor=CENTER)
self.reset_button.place(x=500,y=0,height=30,width=70)
```

```
# Update playlist
def update_playlist(self):
    View.current_list = list(self.controller.update_song())
    self.play_list.delete(*self.play_list.get_children())
    for i in View.current_list:
        value = [i[1],i[2],i[3],i[5]]
        self.play_list.insert('',END,values=value)
    if not View.check_lovelist:
        View.check_lovelist = True
```

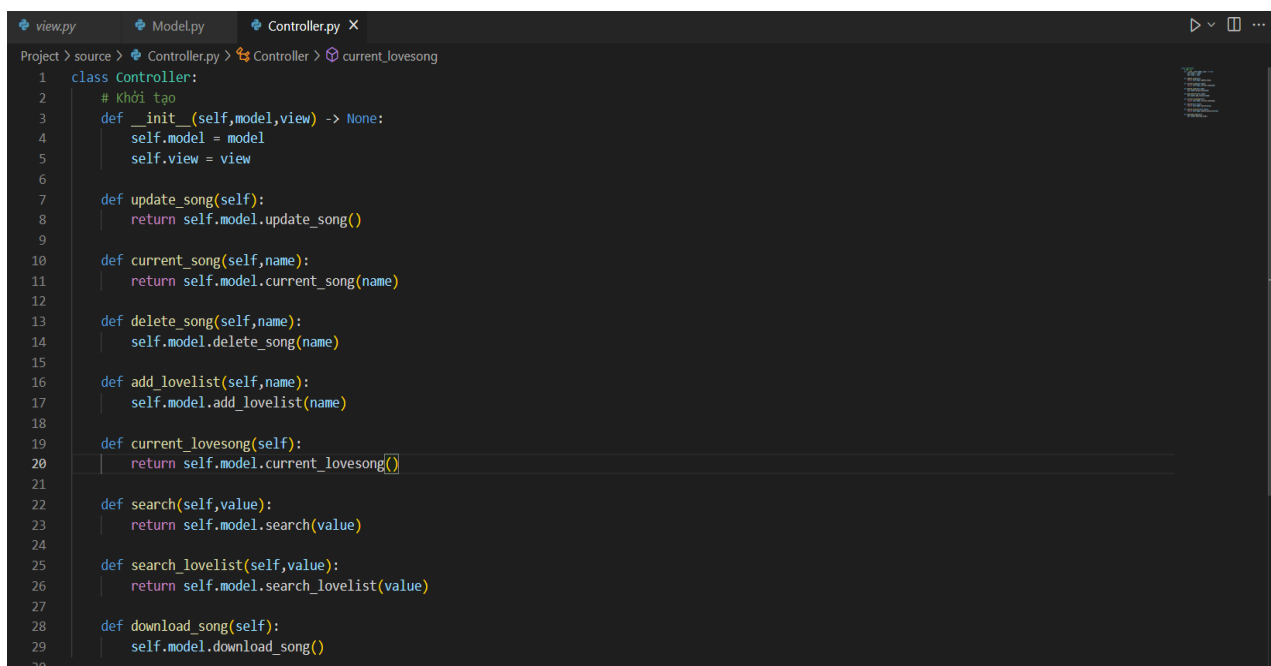
Thêm vào danh sách bài hát yêu thích và hiển thị

```
# Add lovelist
def add_lovelist(self):
    if View.love == False:
        self.love_icon = PhotoImage(file='./Project/images/loved.png")
        self.love_button.config(image=self.love_icon)
        View.love = True
    else:
        self.love_icon = PhotoImage(file='./Project/images/love.png")
        self.love_button.config(image=self.love_icon)
        View.love = False
    if len(View.song_now)>0:
        self.controller.add_lovelist(View.song_now)

# Show lovelist
def lovelist(self):
    if View.check_gui2:
        self.init_frame_main()
        self.button_play.place_forget()
        View.check_gui2 = False
    if View.check_lovelist == True:
        View.current_list = list(self.controller.current_lovesong())
        self.play_list.delete(*self.play_list.get_children())
        for i in View.current_list:
            value = [i[1],i[2],i[3],i[5]]
            self.play_list.insert('',END,values=value)
        View.check_lovelist = False
    else:
        self.update_playlist()
        View.check_lovelist = True
```

4.2 Lớp Controller

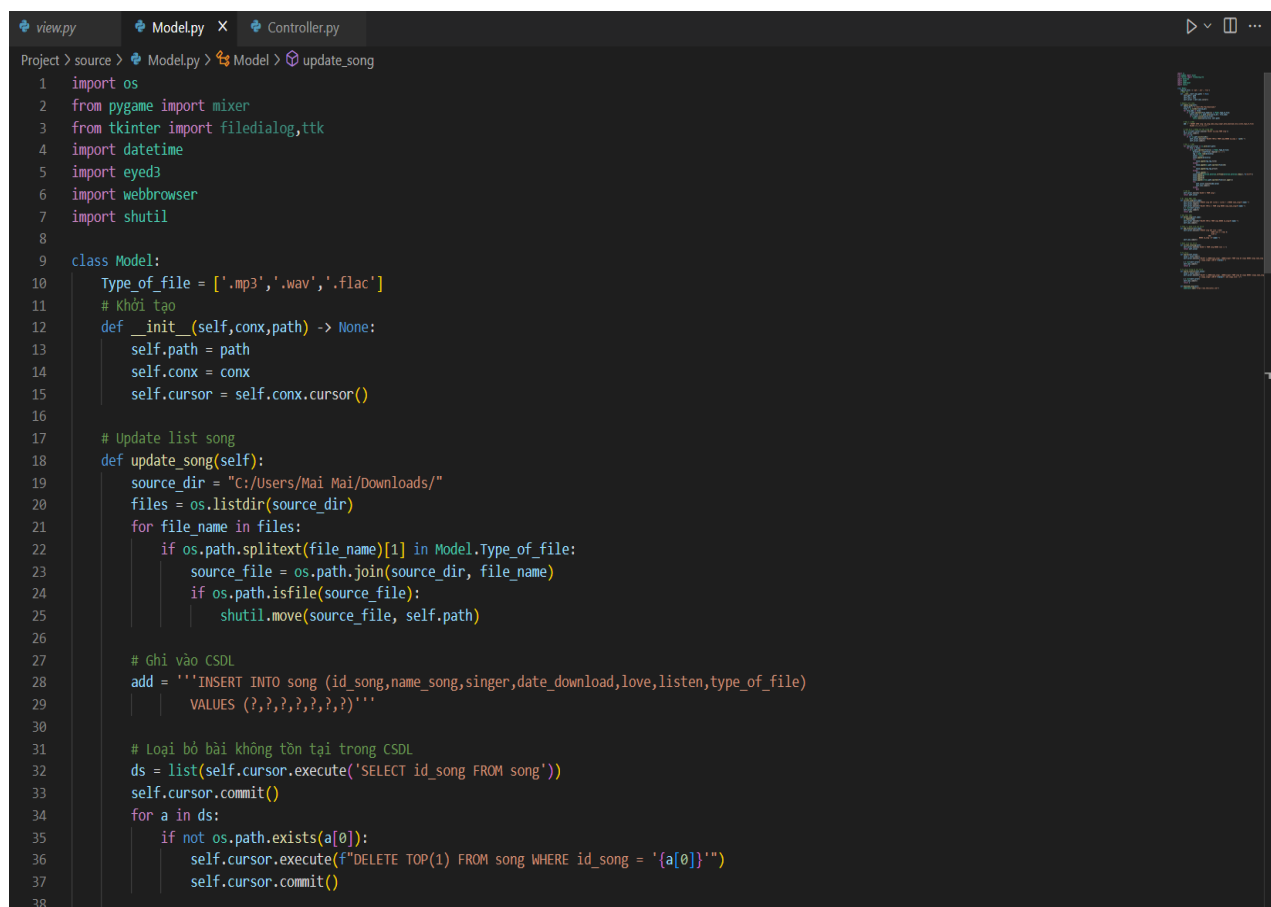
Controller là bộ não của ứng dụng giúp kiểm soát cách các dữ liệu được hiển thị. Cụ thể, Controller được sử dụng để liên kết giữa View và Model.



```
1 class Controller:
2     # Khởi tạo
3     def __init__(self,model,view) -> None:
4         self.model = model
5         self.view = view
6
7     def update_song(self):
8         return self.model.update_song()
9
10    def current_song(self,name):
11        return self.model.current_song(name)
12
13    def delete_song(self,name):
14        self.model.delete_song(name)
15
16    def add_lovelist(self,name):
17        self.model.add_lovelist(name)
18
19    def current_lovesong(self):
20        return self.model.current_lovesong()
21
22    def search(self,value):
23        return self.model.search(value)
24
25    def search_lovelist(self,value):
26        return self.model.search_lovelist(value)
27
28    def download_song(self):
29        self.model.download_song()
30
```

4.3 Lớp Model

Model hỗ trợ lưu trữ các logic dữ liệu, tương tác với hệ thống cơ sở dữ liệu



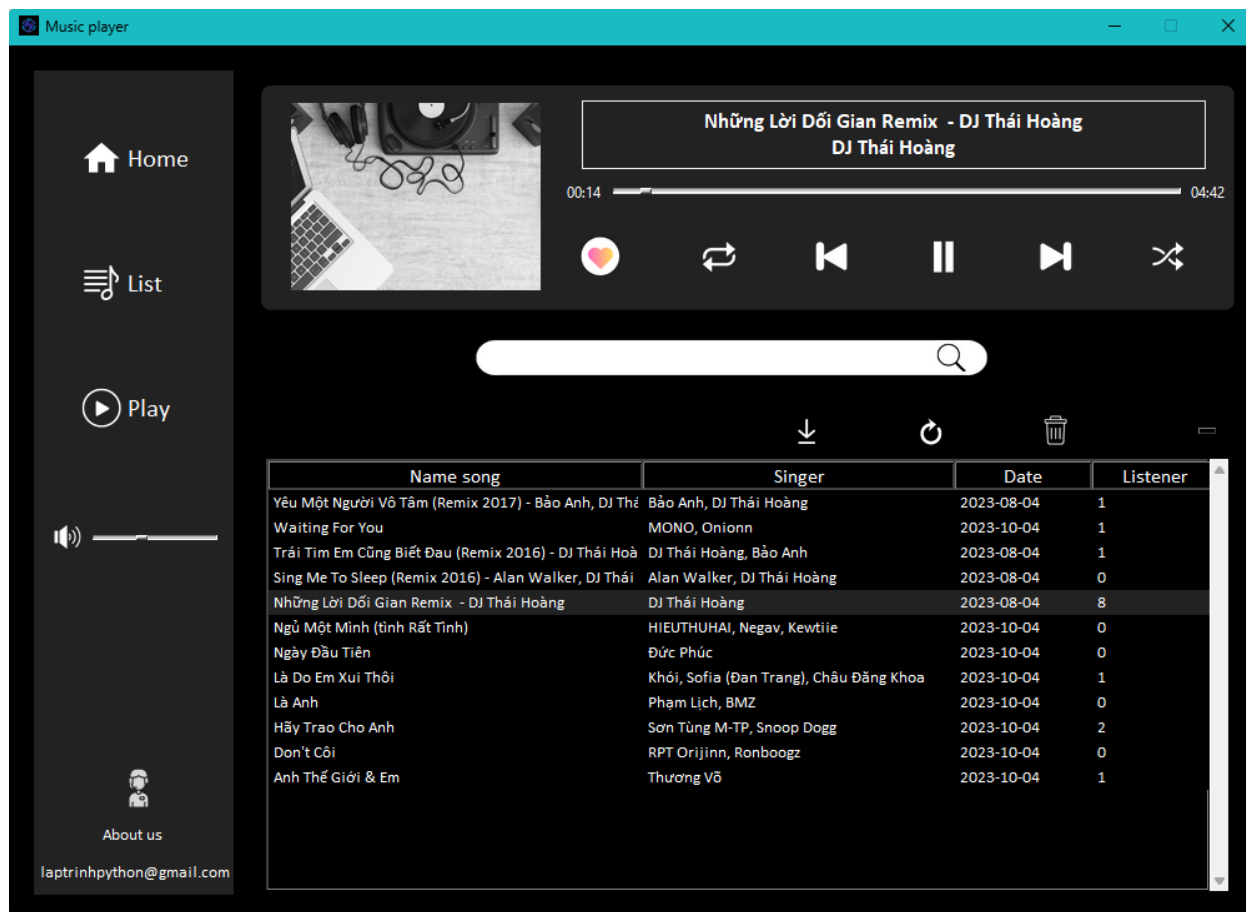
```
1 import os
2 from pygame import mixer
3 from tkinter import filedialog,ttk
4 import datetime
5 import eyed3
6 import webbrowser
7 import shutil
8
9 class Model:
10     Type_of_file = ['.mp3','.wav','.flac']
11     # Khởi tạo
12     def __init__(self,conx,path) -> None:
13         self.path = path
14         self.conx = conx
15         self.cursor = self.conx.cursor()
16
17     # Update list song
18     def update_song(self):
19         source_dir = "C:/Users/Mai Mai/Downloads/"
20         files = os.listdir(source_dir)
21         for file_name in files:
22             if os.path.splitext(file_name)[1] in Model.Type_of_file:
23                 source_file = os.path.join(source_dir, file_name)
24                 if os.path.isfile(source_file):
25                     shutil.move(source_file, self.path)
26
27     # Ghi vào CSDL
28     add = '''INSERT INTO song (id_song,name_song,singer,date_download,love,listen,type_of_file)
29         VALUES (?,?,,?,,?)'''
30
31     # Loại bỏ bài không tồn tại trong CSDL
32     ds = list(self.cursor.execute('SELECT id_song FROM song'))
33     self.cursor.commit()
34     for a in ds:
35         if not os.path.exists(a[0]):
36             self.cursor.execute(f"DELETE TOP(1) FROM song WHERE id_song = '{a[0]}'"')
37             self.cursor.commit()
38
```

5. Kết quả triển khai của ứng dụng

Mục home

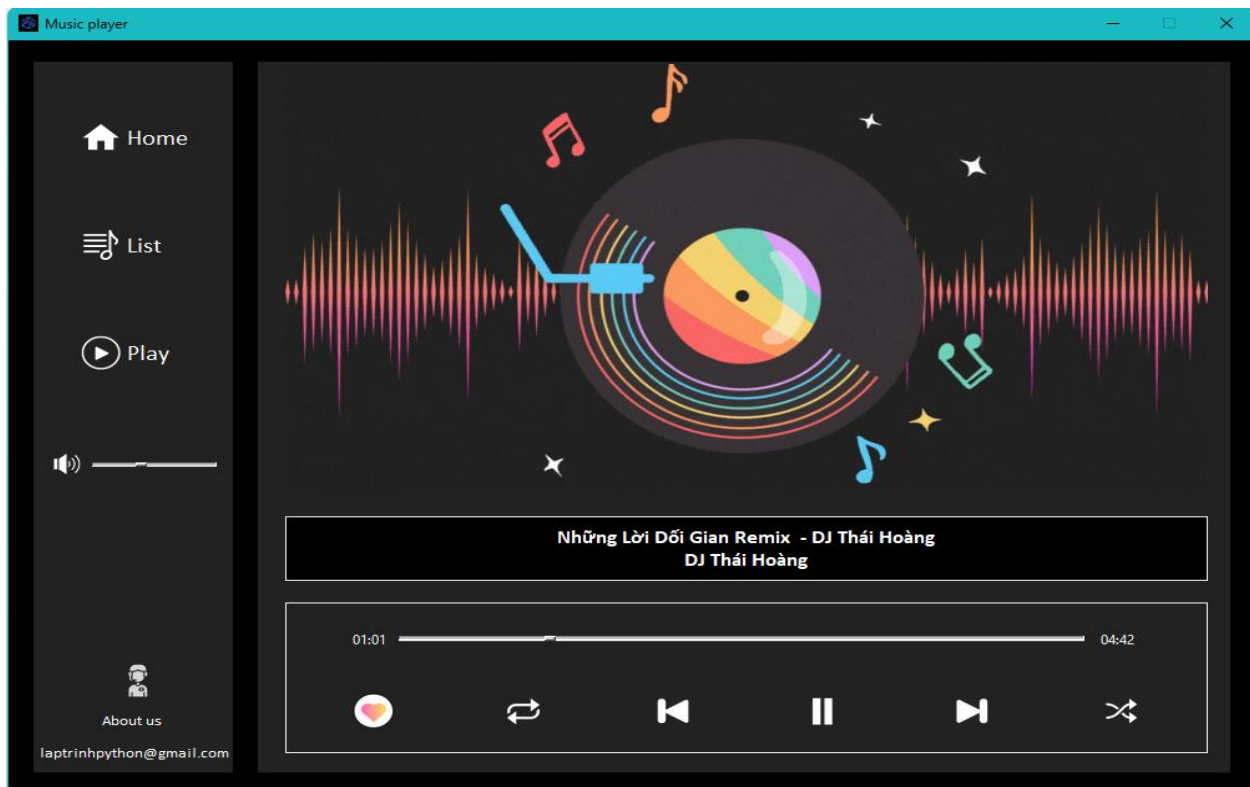
Đây là giao diện tổng quát (giao diện chính) của ứng dụng. Ở đây hiển thị chi tiết và đầy đủ các chức năng chính như pause, repeat, reset, ...

Tại giao diện này, có chứa mục danh sách yêu thích, nơi mà chỉ hiển thị danh sách các bài hát người dùng yêu thích thay vì hiển thị tất cả các bài hát có trong cơ sở dữ liệu của ứng dụng. Để thêm bài hát vào danh sách bài hát yêu thích, người dùng chỉ cần thao tác đơn giản click chuột vào icon trái tim dưới bài hát đang phát, sau khi click hình trái tim chuyển sang màu khác, ngay lập tức ứng dụng sẽ cập nhật bài hát đó vào danh sách các bài hát yêu thích.



Mục Play

Tại đây giao diện sẽ chỉ hiển thị trạng thái của bài hát đang phát sẽ bao gồm đầy đủ các chức năng cơ bản để thay đổi trạng thái bài hát đó.



6. Đánh giá và định hướng phát triển thêm trong tương lai

Ưu điểm nổi bật của ứng dụng:

- Giao diện đẹp, thân thiện phù hợp với đối tượng người dùng.
- Dễ dàng tùy chỉnh, phát triển và thay đổi các modul.
- Đáp ứng đủ nhu cầu nghe nhạc của một ứng dụng nghe nhạc thông thường.
- Ứng dụng có thể đọc được đa dạng các loại file âm thanh khác nhau như: mp3, wav, flac, ...

Những điểm hạn chế của ứng dụng:

- Khâu phát triển chưa được chuyên nghiệp dẫn tới khó khăn trong việc sửa lỗi và nâng cấp hệ thống.
- Số lượng các nghiệp vụ được ứng dụng hỗ trợ cho người dùng còn hạn chế.
- Tốc độ xử lý các nghiệp vụ của người dùng còn chậm. Do đó ứng dụng chưa thể cạnh tranh với các ứng dụng có sẵn và phổ biến hiện nay.

Định hướng phát triển thêm trong tương lai.

- Trong tương lai, nhóm sẽ tích hợp AI cho ứng dụng trở nên thông minh và tốt hơn. Với việc tích hợp AI cho ứng dụng, người dùng sẽ có được trải nghiệm tốt hơn khi sử dụng ứng dụng có thể kể đến như:
 - + Tìm kiếm bài hát bằng giọng nói.
 - + Nhận biết và tìm kiếm một bài hát thông qua đoạn nhạc đề xuất

- + Hiện lyric của bài hát có thể thêm chức năng karaoke.
- + Phát triển thêm tính năng khi có kết nối internet.

BẢNG PHÂN CÔNG NHIỆM VỤ

STT	Tên thành viên Mã sinh viên	Nhiệm vụ	Tỷ lệ đóng góp
1	Nguyễn Đức Linh B20DCAT109	<ul style="list-style-type: none">- Lập trình giao diện chính Home- Lập trình lớp Model truy vấn cơ sở dữ liệu- Lập trình lớp Controller- Làm chức năng tìm kiếm bài hát trong list nhạc- Làm chức năng sắp xếp theo các tiêu chí- Tham gia làm báo cáo	38%
2	Nguyễn Đăng Hạnh B20DCAT053	<ul style="list-style-type: none">- Lập trình xử lý chức năng cho tất cả các button trong giao diện- Bắt các sự kiện thay đổi trong giao diện- Xử lý thanh thời gian phát nhạc, âm lượng,...- Tham gia làm báo cáo	32%
3	Đỗ Ngọc Huế B20DCAT073	<ul style="list-style-type: none">- Lập trình giao diện phát nhạc Play- Lên ý tưởng và thiết kế đồ họa cho ứng dụng- Tham gia làm báo cáo	20%
4	Nguyễn Việt Anh B20DCAT010	<ul style="list-style-type: none">- Làm chức năng sắp xếp khi click vào column header trong list nhạc- Làm báo cáo, slide- Thuyết trình	10%

Tài Liệu Tham Khảo

1. Python Crash Course, 2nd Edition. Copyright © 2019 by Eric Matthes.
2. Các video bài giảng trên youtube.com như:
 - <https://www.youtube.com/watch?v=SCos1o368iE>
 - <https://www.youtube.com/watch?v=zMrr2E3heDM>
 - <https://www.youtube.com/watch?v=OmkQpaXdxs>
3. Tham khảo các thư viện python trên: <https://www.pythontutorial.net/>
4. <https://www.geeksforgeeks.org/pygame-tutorial/>