Trường Đại học Công nghệ - ĐHQGHN

Khoa Điện tử Viễn thông

BÀI TẬP LỚN

Lập trình DSP

XÂY DỰNG ỨNG DỤNG
CHUYỂN ĐỔI MUSIC SHEET THÀNH NHẠC ĐIỆN TỬ

Nhóm thực hiện:
Lê Trung Kiên - 21021602
Bùi Khương Duy - 21020253

Giảng viên:
Nguyễn Hồng Thịnh

HÀ NỘI 2024

Contents

1

Giới thiệu

1.1 Mô tả bài toán

Chủ đề bài tập lớn ”Xây dựng ứng dụng tạo các bản nhạc điện tử dựa trên music sheet” nhằm phát triển một công cụ cho phép người dùng chuyển đổi các bản nhạc mô tả dưới dạng văn bản (script) thành các bản nhạc điện tử hoàn chỉnh. Các nốt nhạc trong bản nhạc được mã hóa bằng chuỗi ký tự, trong đó mỗi ký tự đại diện cho một nốt nhạc và được lưu trữ trong một file .txt. Người dùng có thể lựa chọn các loại nhạc cụ khác nhau (ví dụ: piano, guitar, violin, v.v.) và ứng dụng sẽ tự động xử lý thông qua thuật toán để tạo ra âm thanh tương ứng cho từng nốt nhạc.

Ứng dụng cung cấp các phím chức năng tiện ích như PLAY, PAUSE, RESET và TEMPO (điều chỉnh nhịp điệu của bản nhạc). Đặc biệt, khi người dùng chọn một nhạc cụ, giao diện sẽ hiển thị hình ảnh của nhạc cụ đó, kèm theo bàn phím đàn ảo hoặc dây đàn ảo. Người dùng có thể tương tác trực tiếp với bàn phím máy tính để tạo ra âm thanh, mang lại trải nghiệm gần gũi và sinh động như đang chơi nhạc cụ thật.

1.2 Lý do chọn chủ đề này

Trong thời đại công nghệ ngày nay, việc sáng tạo âm nhạc trở nên dễ dàng và thuận tiện hơn bao giờ hết nhờ vào những ứng dụng tiên tiến. Bạn không còn phải chi hàng triệu, thậm chí hàng chục triệu đồng để sở hữu một cây đàn piano hay các nhạc cụ đắt tiền khác. Thay vào đó, chỉ với lập trình xử lý âm thanh, bạn có thể thiết kế giao diện nhạc cụ cá nhân hóa, từ đó thỏa sức chơi và sáng tạo những bản nhạc cho riêng mình. Trong bài tập lớn này, chúng tôi giới thiệu một ứng dụng chuyển đổi sheet nhạc thành file âm thanh, cho phép người dùng chỉ cần vài thao tác nhập script và chọn nhạc cụ là có thể tạo ra những bản nhạc

hoàn chỉnh. So với những phần mềm phức tạp, ứng dụng này tiết kiệm đáng kể thời gian và công sức. Nó tự động chuyển đổi nốt nhạc thành âm thanh mà không yêu cầu người dùng phải có kiến thức chuyên sâu, giúp mọi đối tượng, từ người mới bắt đầu đến nhạc sĩ chuyên nghiệp, đều có thể thỏa sức sáng tạo. Hơn nữa, ứng dụng này giúp tiết kiệm chi phí vì bạn không cần phải đầu tư vào phần mềm đắt đỏ hay thuê nhạc sĩ. Với khả năng hỗ trợ đa dạng nhạc cụ và tính linh hoạt cao, đây là công cụ lý tưởng cho những ai hoạt động trong các lĩnh vực sáng tác âm nhạc, sản xuất game, phim ảnh, quảng cáo, cũng như giáo dục âm nhạc.

2

Khái quát cơ bản về ứng dụng

2.1 Giới thiệu chung

Ứng dụng sẽ nhận đầu vào là một **script mô tả bản nhạc** được viết dưới dạng văn bản, trong đó mỗi ký tự đại diện cho một nốt nhạc (ví dụ: “BBCCF AFAGB ...”), với các khoảng trắng phân cách các phần trong bản nhạc. Người dùng có thể dễ dàng chỉnh sửa và nhập script này vào ứng dụng. Bên cạnh đó, người dùng cũng sẽ chọn ít nhất **hai loại nhạc cụ** từ một danh sách có sẵn trong giao diện người dùng, chẳng hạn như piano, guitar, violin, trống, sáo, v.v. Ứng dụng sẽ xử lý các nốt nhạc và tạo ra âm thanh tương ứng cho từng nhạc cụ đã chọn. Sau khi hoàn thành, kết quả sẽ được xuất ra dưới dạng một **file âm thanh .wav**, mà người dùng có thể lưu lại và nghe. Mỗi nốt nhạc trong script sẽ được tạo thành âm thanh từ các nhạc cụ đã chọn, và các âm thanh này sẽ được kết hợp lại trong file âm thanh đầu ra, tạo ra một bản nhạc điện tử đầy đủ và đa dạng.

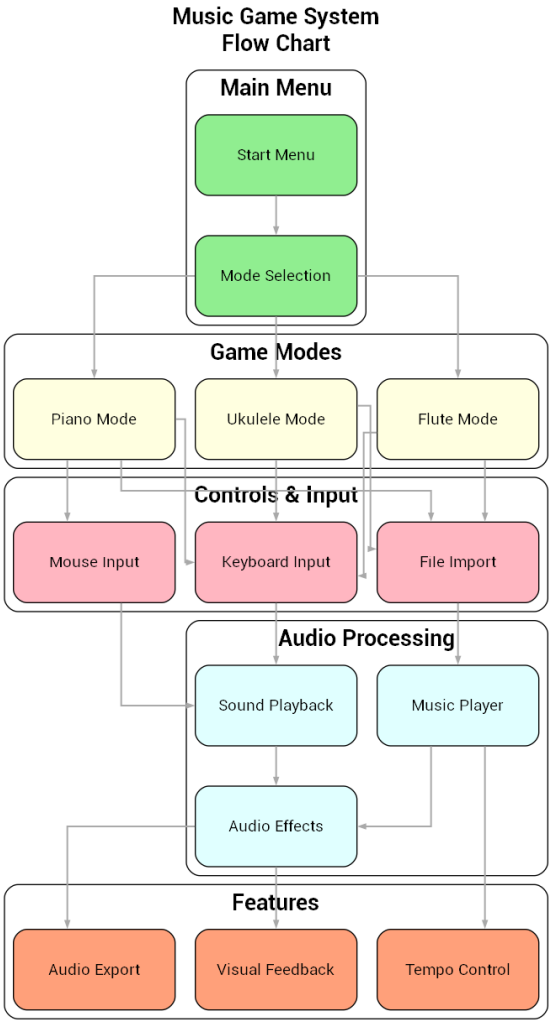

Figure 1: Flow chart of our product

2.2 Quy trình hoạt động

Bước 1. Upload script nhạc:

- Người dùng upload script nhạc dưới dạng văn bản vào ứng dụng. Script này mô tả các nốt nhạc cần phát, ví dụ: “BBCCF AFAGB...”.

Bước 2. Chọn nhạc cụ:

- Người dùng chọn một loại nhạc cụ từ danh sách có sẵn trong giao diện người dùng (ví dụ: piano, ukelele, sáo, kèn,...).

Bước 3. Xử lý bản nhạc:

- Sau khi upload thành công bản nhạc, ứng dụng sẽ phân tích file nhạc, tính toán *duration* (thời gian) của từng nốt, quãng nghỉ và chuyển đổi các ký tự (nốt nhạc) thành các âm thanh tương ứng với từng loại nhạc cụ đã chọn.

Bước 4. Kết hợp âm thanh:

- Sau khi tất cả các nốt nhạc đã được chuyển đổi thành âm thanh, ứng dụng sẽ kết hợp chúng lại thành một bản nhạc hoàn chỉnh, với các âm thanh từ các nhạc cụ đã chọn.

Bước 5. Nghe và điều chỉnh:

- Người dùng có thể nghe thử bản nhạc đã tạo ra và điều chỉnh lại nếu cần thiết (ví dụ: thay đổi nhạc cụ, sửa đổi script nhạc, thay đổi tempo) trước khi xuất lại file âm thanh cuối cùng.

Bước 6. Xuất file âm thanh (tuỳ chọn):

- Cuối cùng, ứng dụng sẽ tạo ra file âm thanh `.wav` và lưu lại cho người dùng. Người dùng có thể phát, lưu hoặc chia sẻ file này.

3

Xây dựng thuật toán và giao diện

Project này sẽ trình bày một phương pháp tạo nhạc với âm thanh mượt mà và chân thực, từ đầu vào là một music sheet với những nốt nhạc khô khan, sẽ được chương trình xuất ra thành giai điệu du dương, như được chơi bởi những nghệ sĩ piano bằng xương bằng thịt - Betthoven D. hay Mozart K.

3.1 Cấu hình bản nhạc

Trong ứng dụng này, việc cấu hình bản nhạc được thực hiện tùy thuộc vào loại nhạc cụ mà người dùng chọn:

1. **Nhạc cụ chơi từng nốt (Piano, Sáo Trúc, ...):**
    - Đối với các nhạc cụ thổi theo nốt, như piano hay sáo trúc, bản nhạc được định nghĩa bằng cách sử dụng các ký tự chữ cái in hoa để đại diện cho các nốt nhạc. Ví dụ:
        - `C` - do, `D` - re, `E` - mi, `F` - fa, `G` - sol, `A` - la, `B` - si, ...
    - Đối với đàn piano, ngoài các nốt bằng tay phải, còn có thêm các nốt tay trái, giúp tạo ra âm thanh đa dạng và hài hòa hơn. Các nốt tay trái được thêm vào bên cạnh nốt tay phải (nốt chính) tương ứng, ngăn cách bởi dấu ”,”.

2. **Nhạc cụ chơi theo hợp âm (Ukelele, ...):**
    - Đối với đàn ukelele, việc chơi nhạc được xác định thông qua các hợp âm. Các hợp âm được định nghĩa theo tên như sau:
        - `C major` - C, `C minor` - Cm, `C7` - C7, ...
    - Các hợp âm này cũng tương tự với các ký hiệu hợp âm như D, E, F, G, A, B.

3. **Quãng Nghỉ:**

   - Dấu phẩy ”,” được sử dụng để đánh dấu các quãng nghỉ trong bản nhạc. Càng nhiều dấu phẩy, quãng nghỉ càng dài. Ví dụ: `C, D,, E,,,` thể hiện các nốt với quãng nghỉ ngày càng dài.

4. **Tệp Âm Thanh:**

   - Các tệp âm thanh được sử dụng trong ứng dụng này đã được thu âm và chỉnh sửa để đảm bảo chất lượng âm thanh tốt nhất.

3.2 Thuật toán cải thiện chất lượng âm thanh

1. **Echo (Tiếng Vang)**: Để tạo hiệu ứng echo [1](tiếng vang), chúng ta tạo ra một bản sao của tín hiệu âm thanh gốc và phát lại bản sao này sau một khoảng thời gian trễ, với độ lớn giảm dần. Quá trình này giúp tạo ra âm thanh giống như tiếng vọng vang lại từ xa.

   Công thức cơ bản để tạo echo như sau:

   $$y(t) = x(t) + \alpha \cdot x(t - T) \tag{1}$$

   Trong đó:

   - $y(t)$ là tín hiệu âm thanh sau khi áp dụng echo.
   - $x(t)$ là tín hiệu âm thanh gốc.
   - α là hệ số giảm dần độ lớn của tín hiệu echo, với giá trị trong khoảng $0 < \alpha < 1$.
   - T là thời gian trễ giữa tín hiệu gốc và echo, tính bằng mili giây.

2. **Fade-In (Tăng Dần Âm Lượng)**: Fade-in [5] là kỹ thuật thay đổi âm lượng của một đoạn âm thanh từ mức rất thấp (hoặc không có âm thanh) đến mức âm lượng bình thường hoặc tối đa trong một khoảng thời gian nhất định. Điều này giúp tạo ra một hiệu ứng mượt mà, dễ chịu khi âm thanh bắt đầu phát.

   Công thức fade-in cơ bản là:

   $$y(t) = x(t) \cdot \left(1 - e^{-\frac{t}{\tau}}\right) \tag{2}$$

Trong đó:

- $y(t)$ là tín hiệu âm thanh sau khi áp dụng fade-in.
- $x(t)$ là tín hiệu âm thanh gốc.
- τ là thời gian fade-in, chỉ ra tốc độ tăng âm lượng.
- e là cơ số tự nhiên, khoảng 2.718.

3. **Fade-Out (Giảm Dần Âm Lượng)**: Fade-out [5] là kỹ thuật giảm âm lượng của một đoạn âm thanh từ mức bình thường hoặc tối đa xuống mức rất thấp, dần dần cho đến khi âm thanh hoàn toàn tắt. Hiệu ứng này thường được dùng để kết thúc một bản nhạc hoặc một đoạn âm thanh mượt mà.

   Công thức fade-out cơ bản là:

$$y(t) = x(t) \cdot e^{-\frac{t}{\tau}} \tag{3}$$

   Trong đó:

   - $y(t)$ là tín hiệu âm thanh sau khi áp dụng fade-out.
   - $x(t)$ là tín hiệu âm thanh gốc.
   - τ là thời gian fade-out, chỉ ra tốc độ giảm âm lượng.
   - e là cơ số tự nhiên.

3.3 Thuật toán xử lý music sheet

Thuật toán đọc datasheet dùng trong ứng dụng âm nhạc được thiết kế để xử lý dữ liệu từ file text chứa các nốt nhạc và chuyển đổi thành tín hiệu âm thanh. Hệ thống sử dụng cấu trúc mapping đa tầng để chuyển đổi giữa các định dạng: ký hiệu nốt nhạc $(C4, D4, ...)$, phím bấm tương ứng $(Q, W, E, ...)$, và file âm thanh $(.wav)$.

Kết hợp nốt nhạc

$$\mathit{combined}[n] = \sum_{i=1}^{M} \mathit{signals}_i[n] \tag{4}$$

với điều kiện chuẩn hóa:

$$\mathit{normalized} = \mathit{combined} \cdot \frac{32767}{\max(|\mathit{combined}|)} \tag{5}$$

Quản lý thời gian kéo dài của nốt nhạc và thời gian ngắt quãng

$$note\_spacing = sample\_rate \cdot tempo \tag{6}$$

$$pause\_duration = sample\_rate \cdot 0.5 \tag{7}$$

Xử lý âm thanh đầu ra:

$$S_{final}[n] = \sum_{i=1}^{K} S_i[n - d_i] \cdot A_i \tag{8}$$

Trong đó:

- S_{final}: Tín hiệu cuối cùng
- S_i: Tín hiệu của nốt thứ i
- d_i: Độ trễ của nốt thứ i
- A_i: Amplitude của nốt thứ i
- K: Tổng số nốt

Vị trí của mỗi nốt trong combined signal được tính theo:

$$position_i = i \cdot note\_spacing \tag{9}$$

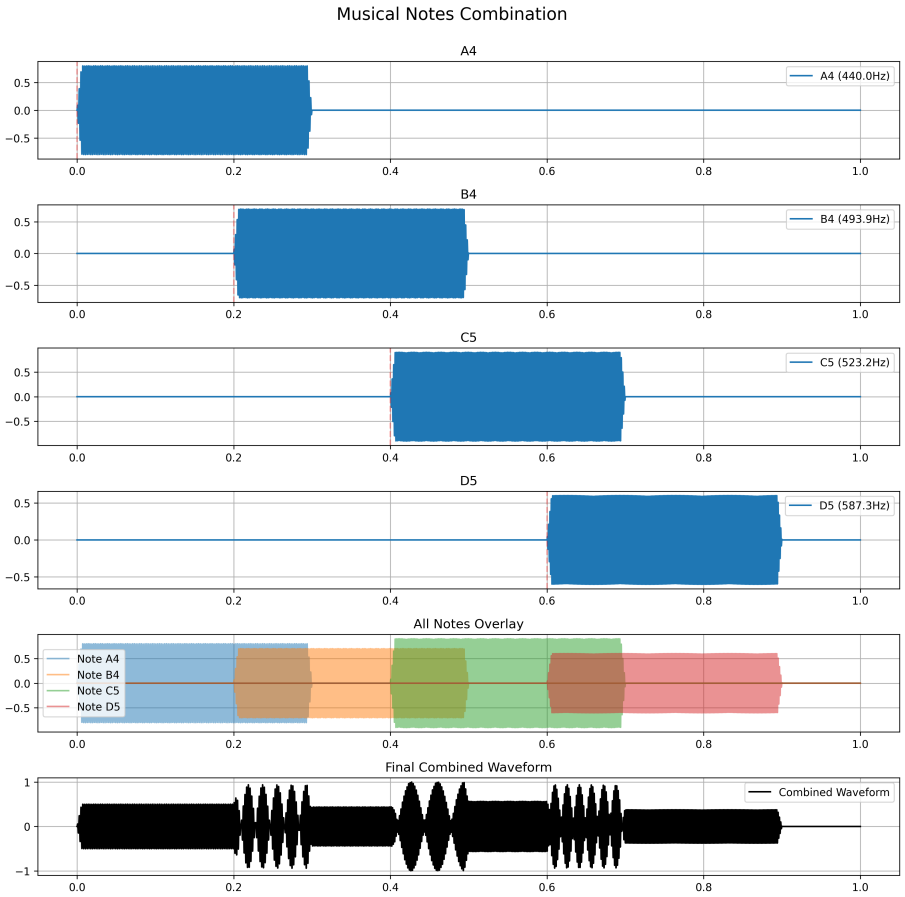

Figure 2: Our combination process

3.4 Threading trong Python

Kỹ thuật **threading** [2] trong GUI giúp thực hiện các tác vụ nặng, như chuyển đổi âm thanh, mà không làm giao diện người dùng bị "đơ". Khi sử dụng threading, các tác vụ này được xử lý trong các thread riêng biệt, giúp người dùng tiếp tục tương tác với giao diện mà không gặp gián đoạn.

Đặc biệt, trong ứng dụng chuyển đổi âm nhạc, threading không chỉ cho phép xử lý các tác vụ chuyển đổi âm thanh, mà còn hỗ trợ phát nhạc trong một luồng riêng biệt. Điều này giúp người dùng có thể nghe thử bài hát trong khi ứng dụng tiếp tục chuyển đổi hoặc xử lý file nhạc, mà không bị ảnh hưởng đến trải nghiệm sử dụng.

Thêm vào đó, việc tách biệt các tác vụ như phát nhạc, xử lý âm thanh, và giao diện giúp tối ưu hóa hiệu suất của ứng dụng và nâng cao sự mượt mà trong suốt quá trình hoạt động.

4

Thực nghiệm

Toàn bộ dự án đã được tải lên Google Drive, có thể truy cập tại [4].

4.1 Cấu trúc Project

Cấu trúc thư mục của dự án như sau:

```
project_folder/
        music_game.py                (file mã nguồn chính)
        piano_lists.py               (file Python chứa danh sách bản nhạc cho piano)
        UTM Bebas.ttf                (font sử dụng trong giao diện)
        Backgrounds/                 (thư mục chứa các hình nền)
            1.jpg
            2.jpg
            3.jpg
            4.jpg
        Tracks/                      (thư mục chứa các bản ghi âm cho nhạc cụ)
            ukulele/                 (thư mục chứa âm thanh cho ukulele)
                C.wav
                D.wav
                E.wav
                ...                  (các bản ghi âm khác cho ukulele)
            piano/                   (thư mục chứa âm thanh cho piano)
                C4.wav
                D4.wav
                E4.wav
                ...                  (các bản ghi âm khác cho piano)
            flute/                   (thư mục chứa âm thanh cho flute)
                C4.wav
```

```
        D4.wav
        E4.wav
        ...                     (các bản ghi âm khác cho flute)
    Songs/                       (thư mục chứa các file bản nhạc dưới dạng .txt)
        fur_elise_flute.txt      (bản nhạc Fur Elise cho flute)
        ukulele_song_1.txt       (bản nhạc ukulele song 1)
        cause_i_love_you.txt    (bản nhạc "Cause I Love You")
        ...                      (các bản nhạc khác)
```

4.2 Hướng dẫn cách thực thi

Trước hết, cần cài đặt các thư viện Python [3] cần thiết bằng cách sử dụng `pip`. Các thư viện cần thiết bao gồm:

- `pygame`: Dùng để tạo giao diện đồ họa và xử lý âm thanh.
- `librosa`: Dùng để xử lý âm thanh, đặc biệt là phân tích các file âm thanh.
- `sounddevice`: Dùng để phát âm thanh từ máy tính.
- `scipy`: Dùng để xử lý dữ liệu âm thanh (ví dụ: đọc/wav file).
- `tkinter`: Dùng để mở hộp thoại chọn file.

Dùng terminal hoặc command prompt và gõ lệnh:

```
pip install pygame librosa sounddevice scipy
```

`tkinter` là thư viện chuẩn trong Python, nên bạn không cần phải cài đặt thêm.

4.3 Chạy Mã Nguồn

Cài đặt các thư viện và chuẩn bị các file cần thiết, sau đó mở terminal hoặc command prompt, di chuyển đến thư mục chứa file Python, sau đó gõ lệnh sau:

```
python music_player.py
```

Lưu ý rằng file Python cần được lưu với phần mở rộng `.py` (ví dụ: `music_player.py`).

4.4 Tương Tác trên Giao Diện

4.4.1 Giao diện khởi động (Màn hình chính)

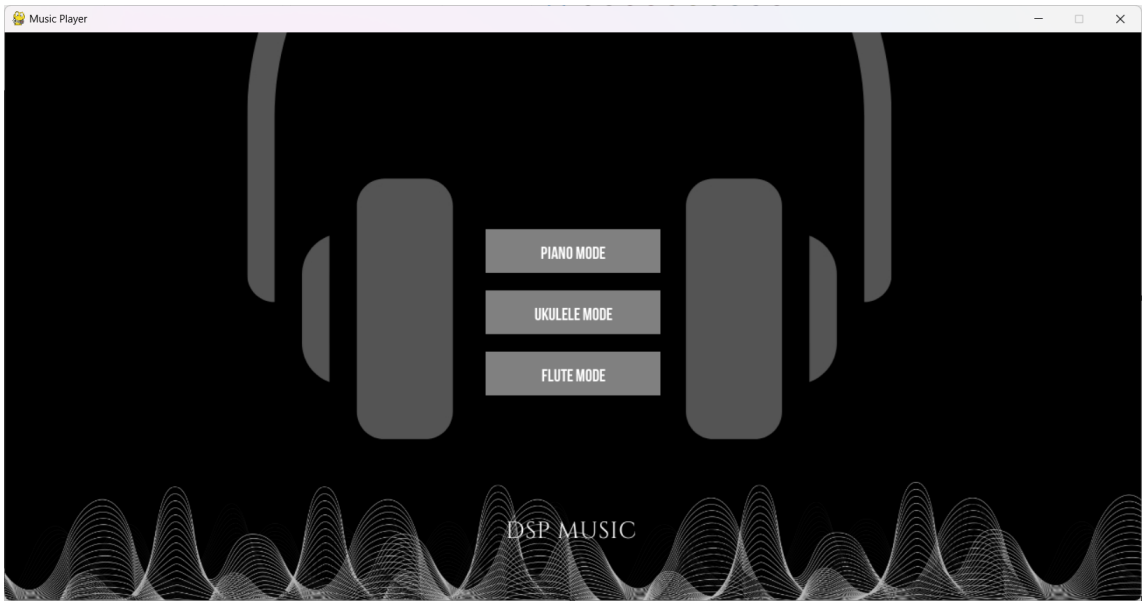

Figure 3: Màn hình chính

Người dùng bấm chọn 1 trong 3 chế độ: Piano, Ukulele hoặc Flute (Sáo trúc)

4.4.2 Giao diện chơi nhạc

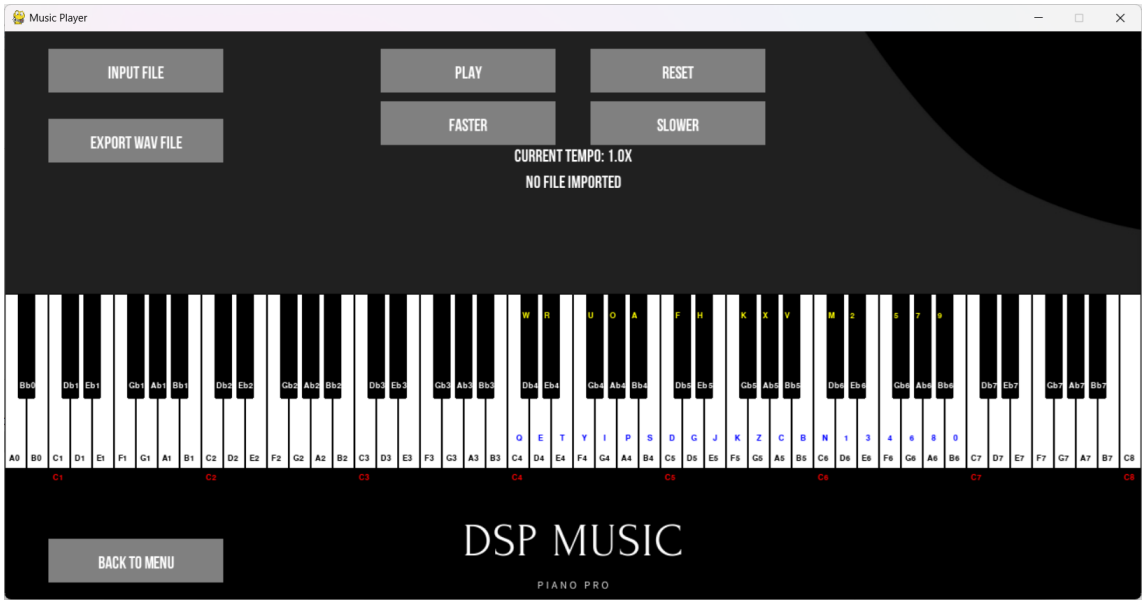

Figure 4: Chế độ chơi piano

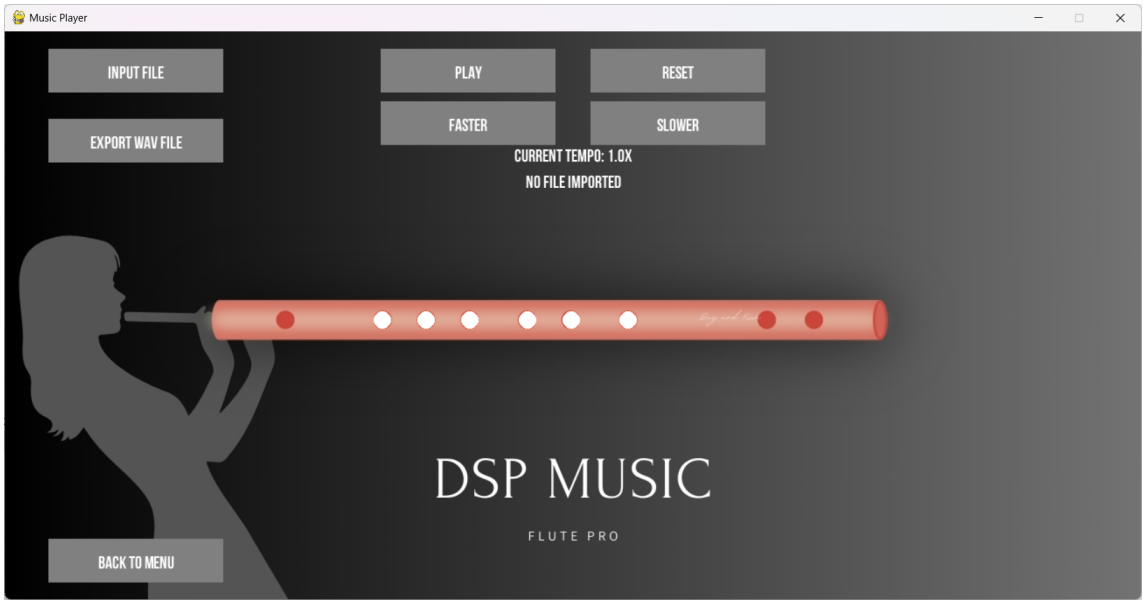

Figure 5: Chế độ chơi sáo trúc

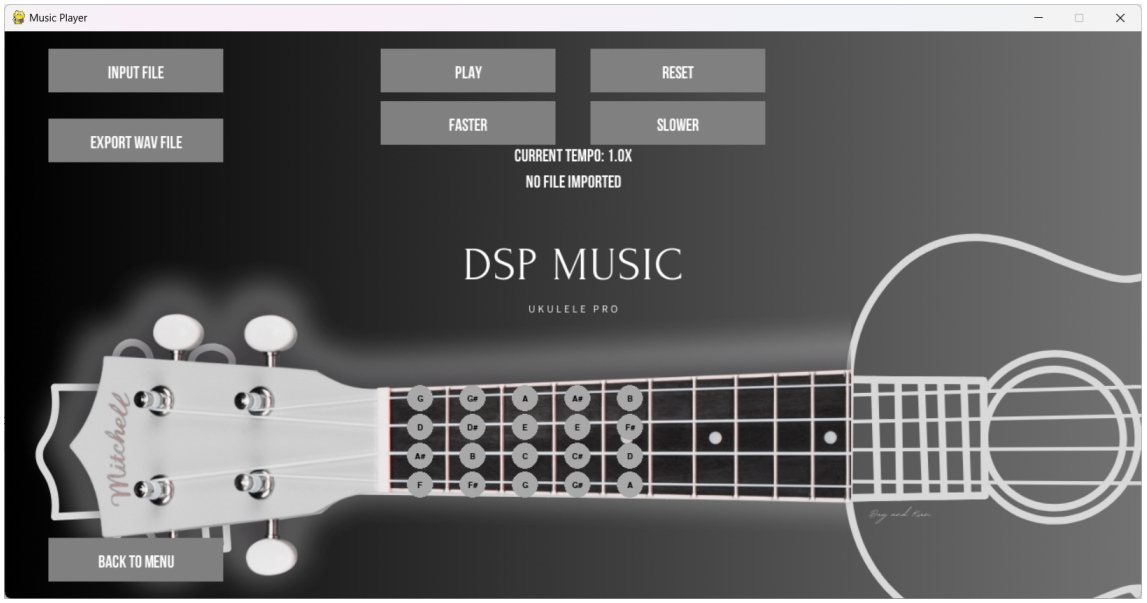

Figure 6: Chế độ chơi ukulele

Tập làm nghệ sĩ: Người dùng trực tiếp sử dụng bàn phím để đánh đàn, từng phím tương ứng với nốt nhạc đã được gợi ý trên phím đàn piano

Tập làm nhạc sĩ: Người dùng click chọn input file để tải file nốt nhạc tự soạn vào chương trình, sau khi import xong, nhấn Play để phát nhạc. Trong quá trình phát nhạc, có thể click tăng/giảm tempo để điều chỉnh tốc độ phát. Để xuất file âm thanh từ bản nhạc đã tạo, click Export.

Đổi chế độ chơi: Người dùng bấm nút back bên dưới giao diện để trở về màn hình chính.

5

Kết luận và Hướng phát triển

Ứng dụng hiện tại đã đạt được những yêu cầu đặt ra trong việc phân tích và tạo ra âm thanh từ các bản nhạc đã soạn sẵn, cũng như cung cấp các công cụ hữu ích cho phép người dùng phát, ghi âm và điều chỉnh nhạc theo ý muốn. Mặc dù có ứng dụng có một số ưu điểm nhất định nhưng vẫn còn một số yếu tố cần cải thiện và mở rộng để nâng cao trải nghiệm người dùng, đồng thời tăng tính linh hoạt của ứng dụng.

Trước hết, một trong những ưu điểm nổi bật của ứng dụng là **giao diện đơn giản và dễ sử dụng**, cho phép người dùng tiếp cận và sử dụng các tính năng mà không cần kiến thức chuyên sâu về âm nhạc. Bên cạnh đó, ứng dụng cũng sở hữu tính linh hoạt cao khi **hỗ trợ nhiều loại nhạc cụ** và cho phép người dùng nhập các **script nhạc tùy chỉnh**, mở ra khả năng sáng tạo vô hạn trong việc tạo ra các bản nhạc đa dạng và độc đáo.

Tuy nhiên, ứng dụng vẫn còn một số hạn chế cần được khắc phục. Một trong những vấn đề lớn là **thuật toán xử lý ngắt nghỉ và thời gian (duration)** của nốt nhạc chưa được tối ưu. Dù ứng dụng có thể tạo ra âm thanh từ các bản nhạc, nhưng việc xử lý thời gian giữa các nốt nhạc vẫn chưa hoàn hảo, dẫn đến hiện tượng gián đoạn hoặc không mượt mà trong các bản nhạc phức tạp. Bên cạnh đó, ứng dụng chưa khai thác triệt để các **kỹ thuật lập trình hướng đối tượng**, đặc biệt là trong việc áp dụng kế thừa lớp **(inheritance)**, khiến mã nguồn chưa thực sự dễ bảo trì và khó mở rộng khi có yêu cầu mới.

Để ứng dụng có thể phát triển mạnh mẽ hơn trong tương lai, một số hướng đi có thể được xem xét. Đầu tiên, việc **mở rộng bộ nhạc cụ** sẽ giúp người dùng có thêm sự lựa chọn phong phú, chẳng hạn như bổ sung các nhạc cụ như trống, violon, kèn, ... Điều này không chỉ tăng tính đa dạng cho bản nhạc mà còn tạo cơ hội sáng tạo mới cho người dùng. Thứ hai, **thêm các hiệu ứng âm thanh nâng cao**, như reverb, chorus, flange, có thể làm phong phú và chiều sâu hơn cho âm thanh, giúp người dùng tạo ra những bản nhạc độc đáo và mang tính cá nhân

cao. Cuối cùng, việc **cải thiện hiệu suất xử lý âm thanh** cũng rất quan trọng, đặc biệt là khi ứng dụng phải xử lý các bản nhạc phức tạp với nhiều nhạc cụ và hiệu ứng. Đảm bảo không có giật lag hay độ trễ sẽ là yếu tố then chốt để mang đến một trải nghiệm âm nhạc mượt mà và chuyên nghiệp hơn.

References

[1] SOUND Effects. https://sound.eti.pg.gda.pl/student/eim/synteza/adamx/eindex.html

[2] Stack Overflow. *Python MP3Play with threading*. 2016. https://stackoverflow.com/questions/41708884/python-mp3play-with-threading.

[3] Python libraries. https://pypi.org/

[4] Our project can be accessed here: https://drive.google.com/drive/folders/1x8yVcNk1jBOifEc2c1SAx8EfaOM7SXBf?usp=sharing

[5] Wikipedia. *Fade (audio engineering)*. https://en.wikipedia.org/wiki/Fade_(audio_engineering).