

SENIOR PROJECT CN4-2023

Guitar Chord Progressions and Fingerings Optimization

Submitted to

School of Information, Computer and Communication Technology
Sirindhorn International Institute of Technology
Thammasat University

December 2023

by

Apinya Sriyota 6322771534

Sasikarn Khodphuwiang 6322772375

Chanawong Karoon-ngampun 6322774025

Advisor: Assoc. Prof. Dr. Cholwich Nattee

Abstract

This project aims to leverage artificial intelligence (AI) techniques to optimize chord progressions and fingerings for guitar playing. The goal is to develop an AI-driven system that can analyze songs, identify challenging chord transitions, and generate alternative chord progressions and fingerings that enhance playability while preserving musical quality. By applying AI algorithms and machine learning, the project aims to revolutionize the way guitarists approach chord selection and fingering techniques, providing personalized and efficient solutions for optimizing their playing experience.

Acknowledgements

We would like to express our deepest gratitude to our advisor, Assoc. Prof. Dr. Cholwich Nattee for his guidance, suggestions, comments, and critiques of this project.

Contents

Abstract	i
Acknowledgments	ii
Contents	iii
Abbreviations	v
List of Figures	vi
List of Tables	vii
1 Project Concept	1
1.1 Summary	1
1.2 Motivation	1
1.3 Users and Benefits	1
1.4 Typical Usage	2
1.5 Main Challenges	2
2 Requirements Specification	3
2.1 System Description	3
2.2 Requirements	5
3 Design Specification	8
3.1 System Architecture	8
3.2 Detailed Design	9
4 Implementation	11
4.1 Song Search	11
4.2 Chord Generation	12
4.3 Feedback Collection	17

5	Conclusion and Future Work	18
5.1	Conclusion	18
5.2	Future Work	18
6	References	19

Abbreviations

The technical terms listed below will be expressed by their abbreviations in this report. Other abbreviations will be defined at their first occurrence.

API Application Programming Interface

List of Figures

2.1	System Perspective	3
2.2	System Functions	3
3.1	System architecture	8
3.2	Framework of Chord Progression and Fingering Generation	9
4.1	User interface of song search page	11
4.2	User interface of chord generation with lyrics	12
4.3	User interface of feedback collection	17

List of Tables

4.1	Chord data of “honey” by boy pablo from chord-and-tabs	13
4.2	Song information of “honey” by boy pablo	13
4.3	Chord information of c:diminished	14
4.4	Music theory for chord substitution	15

Chapter 1

Project Concept

1.1 Summary

The project's core concept revolves around developing a model to provide users with guitar chords and finger positions for their selected songs. Users can input a song name and artist name, and the program will offer customization options for users to select from a wide range of preferences, ensuring that musicians and guitar enthusiasts can access the information they need in a way that suits their skill level and musical style.

1.2 Motivation

Playing the guitar is a popular hobby globally, yet beginners often face challenges in finding accessible chords. This project was inspired by situations where new players wish to play a song but can't locate available chord progressions. Leveraging AI and diverse data, our program offers chords in various styles for all skill levels—from beginners to seasoned guitarists. Our goal is to make guitar learning more accessible and inclusive.

1.3 Users and Benefits

The primary beneficiaries of this program are guitarists, musicians, and individuals passionate about playing the guitar. The benefits of this program include:

- Customization: Users can personalize their experience by selecting their preferred range and style, such as choosing from a wide range of finger positions, chord variations, or even alternative tunings.
- Comprehensive Learning: The program provides not only chords but also finger positions, enabling users to learn songs accurately and comprehensively.
- Accessibility: Guitarists can access chords information for virtually any song by entering a song name and artist name, eliminating the need for manual searches across multiple platforms.

- Accuracy: Chords provided by the program will be sourced from reliable and reputable guitar chord databases, ensuring accuracy and reliability.

1.4 Typical Usage

The typical usage of the program involves the following steps:

- Input Song Information: Users enter the name of the song they want to learn.
- Customize Preferences: Users can fine-tune their experience by defining their preferences, such as range, style, and difficulty level.
- Retrieve Chords and Finger Positions: The program employs AI models and trusted sources to generate and present guitar chords and finger positions that align with the user's defined preferences.
- Practice and Learn: Users can practice the song using the provided chords and finger positions, steadily mastering it and adapting to their preferred playing style.

1.5 Main Challenges

While developing this project, several challenges must be addressed, including:

- Licensing and Copyright: Addressing potential copyright and licensing issues when collecting chords for copyrighted songs.
- Data Integration: Seamlessly integrating data from various sources while maintaining data quality and consistency.
- Customization Complexity: Offering a wide range of customization options to cater to diverse user preferences without overwhelming the user interface.
- AI Model Accuracy: Ensuring that AI models generate accurate guitar chords and finger positions for an extensive song library.
- User Interface Design: Creating an intuitive and user-friendly interface that allows users to customize their preferences without overwhelming complexity.

Chapter 2

Requirements Specification

2.1 System Description

ChordBrew is a web-based platform designed for guitar enthusiasts, utilizing artificial intelligence (AI) to assist in their playing experiences. Users can input a song title and artist name and receive chord progression and fingering suggestions tailored to their preferences. The platform offers an individualized learning experience with options for different musical styles and difficulty levels, facilitating a more personalized learning journey.

2.1.1 Perspective

The relationship between ChordBrew, other systems, and users is shown in Figure 2.1.

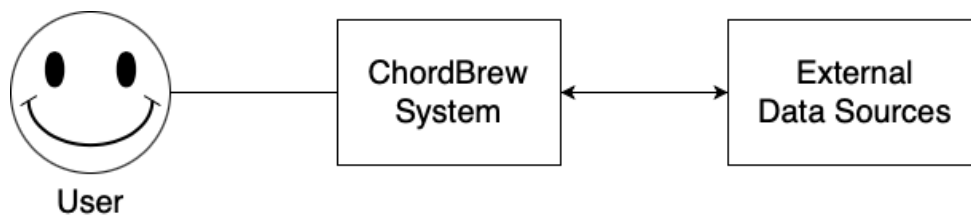


Figure 2.1: System Perspective

ChordBrew is an interactive digital platform designed for guitar enthusiasts. It bridges user inputs with the capabilities of our dataset and external data sources. Users input the song and artist names, which prompt the system to generate a chord arrangement. Through seamless integration of user preferences, our internal model, and external musical information, ChordBrew delivers personalized chord progressions and fingering suggestions, providing a comprehensive and enhanced guitar learning experience.

2.1.2 Functions

The functions of ChordBrew are shown in Figure 2.

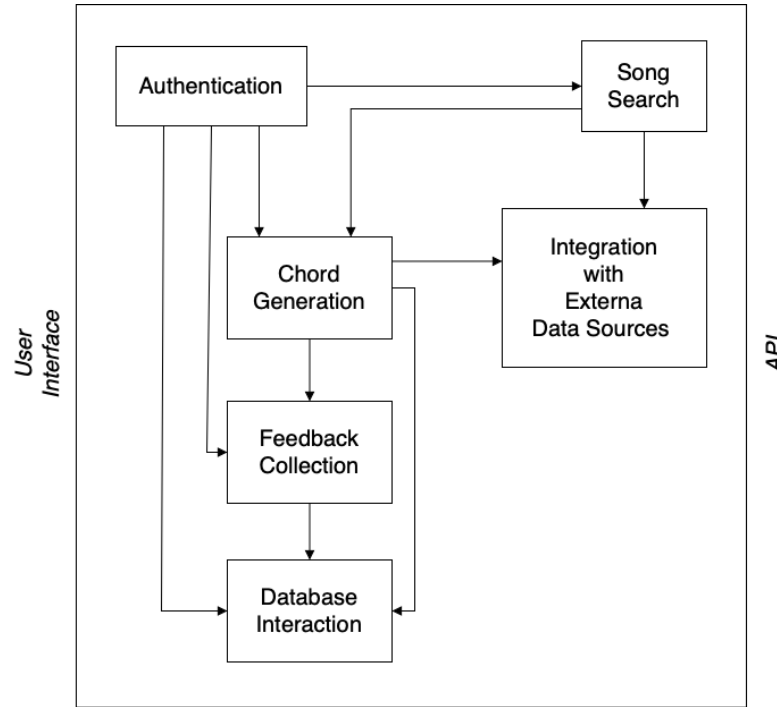


Figure 2.2: System Functions

The functions are summarized as:

- **Authentication:** Manages user registrations, logins, and profiles, ensuring data security and personalized user experiences.
- **Song Search:** Enables users to search for specific songs or artists within the ChordBrew database and external data sources.
- **Chord Generation:** Utilizes the AI model to analyze songs and provide optimized chord progressions and fingerings based on user preferences.
- **Database Interaction:** Stores user data, generated chord progressions, feedback, and caches results from external sources for quick retrieval.
- **Integration with External Data:** Connects with external platforms like YouTube via API connectors to fetch song data or mp3 files for analysis.
- **Feedback Collection:** Allows users to provide feedback on chord suggestions and integrates this feedback for continuous improvement of the AI model.

2.2 Requirements

The requirements of the system are listed in the following subsections.

2.2.1 General

- G1 ChordBrew must adhere to relevant data protection and copyright regulations
- G2 ChordBrew must present chords in a clear, single-page layout for easy reading during live performances.
- G3 ChordBrew should provide an interactive platform for guitarists of all skill levels.
- G4 ChordBrew should support a wide range of song selections spanning various genres.
- G5 ChordBrew should offer an adjustable interface, allowing users to change text size for better readability.
- G6 ChordBrew should provide an auto-scrolling feature for tabs, aiding users in seamless playing.
- G7 ChordBrew should provide theme or background options to ensure optimal visibility in various lighting conditions.
- G8 ChordBrew may offer features to save user preferences and song favorites for future visits.
- G9 ChordBrew may incorporate user suggestions for platform improvements.

2.2.2 Authentication

- A1 The Authentication module must allow users to register with unique credentials.
- A2 The Authentication module must encrypt stored passwords to protect user data.
- A3 The Authentication module should provide a mechanism for users to recover forgotten passwords.
- A4 The Authentication module may incorporate a mechanism for sustained user sessions on subsequent visits.

2.2.3 Song Search

- S1 The Song Search module must enable users to search for songs based on song or artist names.
- S2 The Song Search function must show relevant details such as song name, artist, and an option to generate chords.

- S3 The Song Search module should display real-time results as users type in the search bar.
- S4 The Song Search function may provide filters or sorting options based on genre, popularity, or release date.

2.2.4 Chord Generation

- C1 The Chord Generation function must analyze the selected song to generate chord progressions.
- C2 The Chord Generation function must visually represent finger placements for each chord.
- C3 The Chord Generation function must ensure accuracy in tablature generation for optimal playability.
- C4 The Chord Generation function should let users customize chord suggestions based on skill level or style.
- C5 The Chord Generation function should offer an option to generate guitar tabs alongside chord progressions.
- C6 The Chord Generation function may adapt chord suggestions based on historical user feedback.
- C7 The Chord Generation function may allow users to toggle between chord views and tab views for flexibility.

2.2.5 Database Interaction

- D1 The Database Interaction function must securely store user data, including profiles and preferences.
- D2 The Database Interaction function must schedule regular backups to prevent data loss.
- D3 The Database Interaction function should allow for quick retrieval of stored data.
- D4 The Database Interaction module should ensure data encryption and regular backups to prevent data breaches or losses.
- D5 The Database Interaction function may cache frequently accessed data to optimize performance.
- D6 The Database Interaction module may offer users insights into their search history or most accessed chords.

2.2.6 Integration with External Data

- I1 The Integration function must manage API rate limits to prevent overuse or bans.
- I2 The Integration module should cache frequently accessed external data to minimize API calls and improve response times.
- I3 The Integration module may fetch song data from platforms like YouTube when requested by users.
- I4 The Integration module may expand its sources to include other music platforms or databases in the future.

2.2.7 Feedback Collection

- F1 The Feedback Collection module must allow users to rate and comment on chord suggestions.
- F2 The Feedback Collection module should store feedback to refine and improve chord recommendations.
- F3 The Feedback Collection function may prioritize popular song feedback for model adjustments.

Chapter 3

Design Specification

3.1 System Architecture

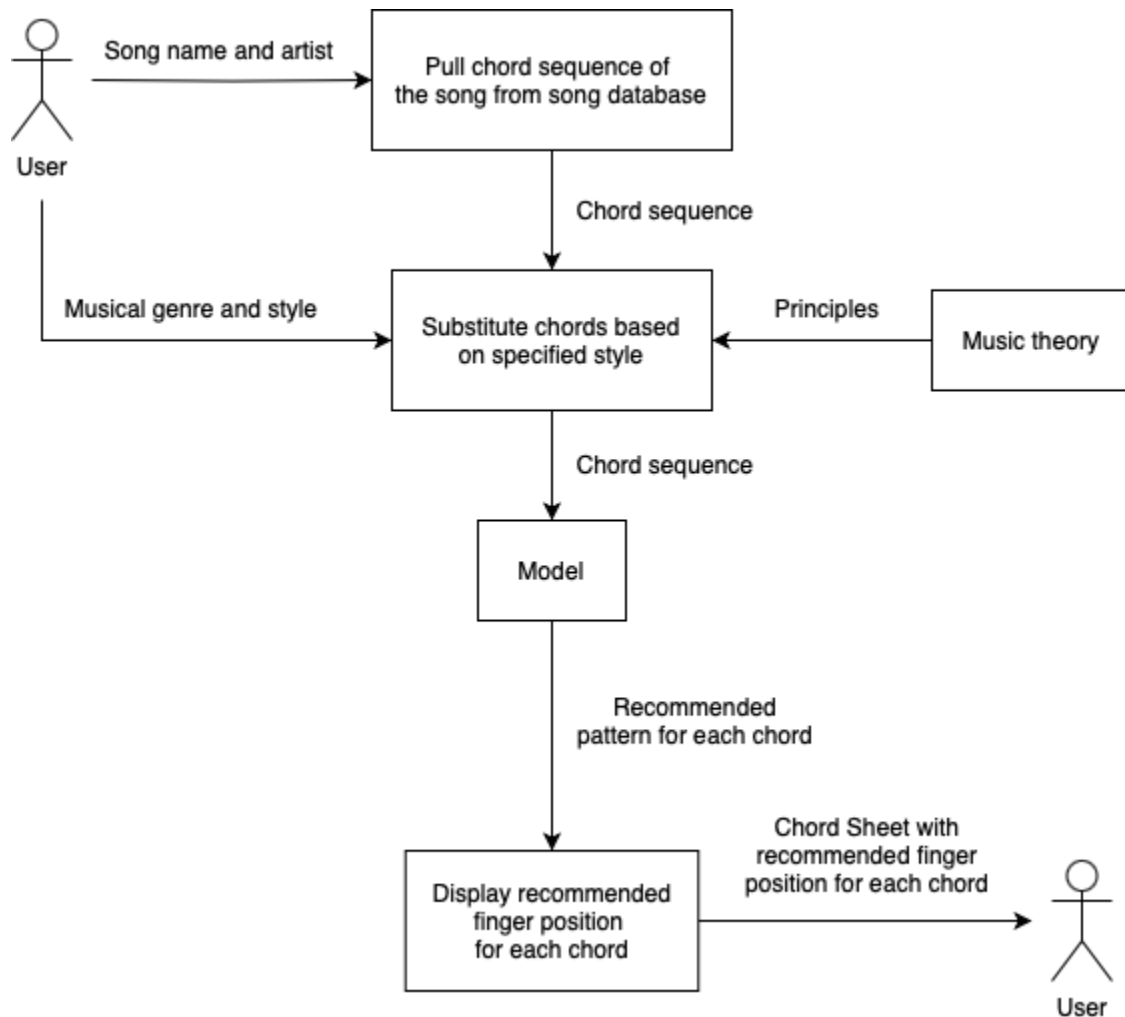


Figure 3.1: System architecture

The primary focus of this study involves the specification and design of ChordBrew, emphasizing its capability to generate optimal finger positions for each chord within a song transitioned by minimizing the distance across all patterns.

3.2 Detailed Design

3.2.1 Framework of Chord Progression and Fingering Generation

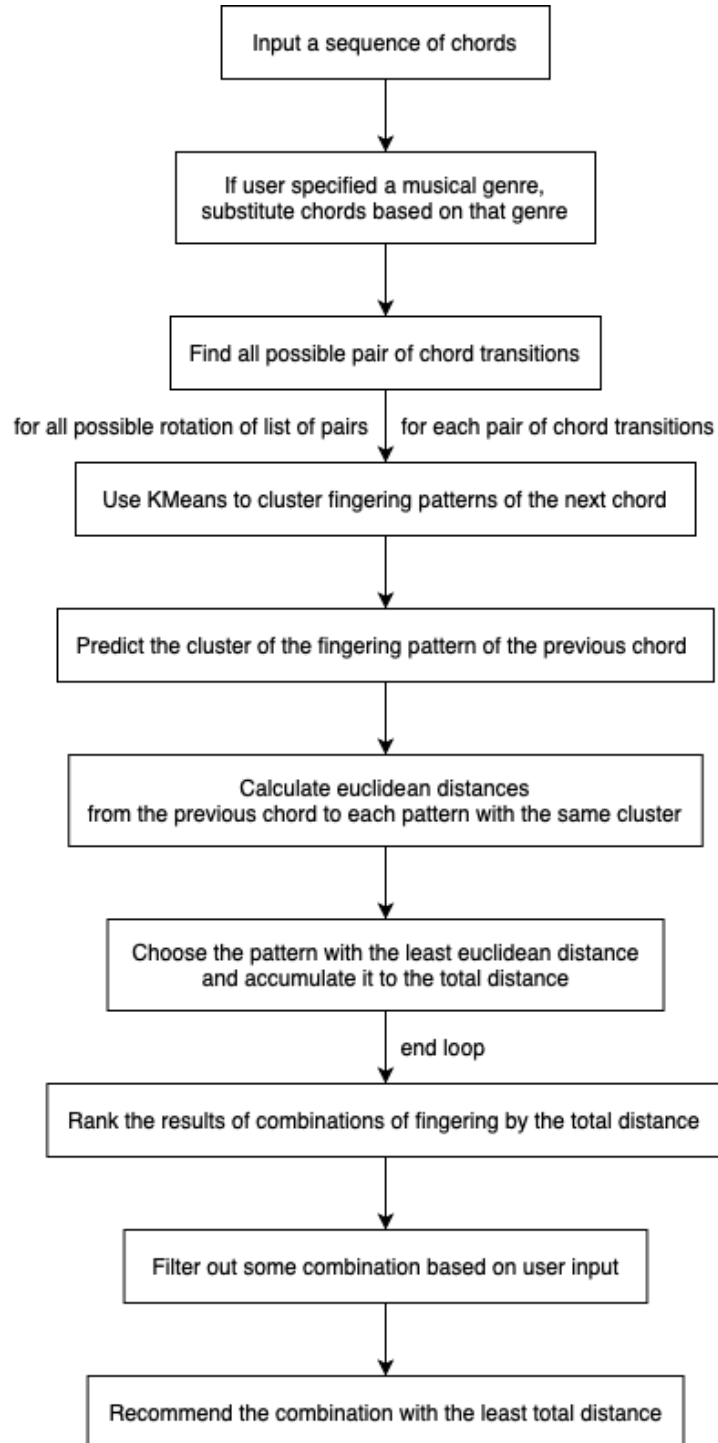


Figure 3.2: Framework of Chord Progression and Fingering Generation

3.2.2 Database

The song database attributes:

- Key: key of this song
- Tune: guitar tuning of chords
- Capo: capo fret (if any)
- Chords: all distinct chords in this song
- Sequence: sequence of chord use in this song

The song information database attributes:

- Name: song name
- Album: album name
- Artist: artist name
- Year: released year
- Genre: genre of the song
- Language: language used in this song
- Country: country that the artist live in
- Lyric: lyrics of the song grouped by part of this song

The chord database attributes:


- Chord: chord name
- Notation: chord notation
- Type: chord type
- Formula: chord progression formula
- Notes: notes in this chord
- Fingering 1 - 10: finger positions used to play this chord, varying from 1 to 10 different patterns of finger position

Chapter 4


Implementation

4.1 Song Search

4.1.1 User Interface

CHORD GENERATOR 

Song : Champagne Problems Artist : Taylor Swift Style : Original Simplified Chords : Default

[Generate](#) 




Figure 4.1: User interface of song search page

4.2.2 Data Collection

We conducted web scraping procedures to gather chord data from 2 platforms i.e. **chord-and-tabs.net**[7] and **e-chord.com**[6]. Similarly for song information, we gathered it from a reliable source, **genius.com**[5]. Additionally, supplementary song information was manually collected from Google.

key	tune	capo	chords	sequence
"G"	"EADGBE"	" "	["F", "D", "C", "Em", "G", "Am", "Dm", "Gmaj7", "Fmaj7", "Cmaj7", "Bm7", "Am7"]	{"verse1": ["" , "Gmaj7", "Fmaj7", "Cmaj7", "Gmaj7", "Fmaj7", "Cmaj7", "Gmaj7", "Fmaj7", "Cmaj7", "Gmaj7", "Fmaj7", "Cmaj7"], ...

Table 4.1: Chord data of “honey” by boy pablo from chord-and-tabs

name	album	artist	year	genre	language	country	lyric
"honey"	"Wachito Rico"	"boy pablo"	"2020"	["Alternative/Indie"]	"en"	"Norway"	{"verse1": ["The flowers", "wither", "\nAnd the", " clouds grow bigger\nUp in the garden", " alone, I'm ", "freezing, but it's ", "fine\nCause I'm ", "thinking of ", "you\nAnd\nall", " of the things we could do\nRunning as", " fast as I can\n, glad ", "I forgot to tie", " my shoes\n"], ...

Table 4.2: Song information of “honey” by boy pablo

We manually conducted a collection of finger positions for common guitar chords from the website www.all-guitar-chords.com[1]. This involved documenting which fingers are used for each chord, specifying the string and fret positions for these fingers in the given chord pattern.

chord	notation	type	formula	notes	fingering1	fingering2
c:diminished	c	diminished	1-b3-b5	C D# F#	{1: {"string": 5, "fret": 3, "note": "C"}}, 2: {"string": 4, "fret": 4, "note": "F#"}}, 3: {"string": 2, "fret": 4, "note": "Eb"}}, 4: {"string": 3, "fret": 5, "note": "C"}}}	{1: {"string": [3, 6], "fret": 8, "note": ["Eb", "C"]}}, 2: {"string": 5, "fret": 9, "note": "F#"}}, 3: {"string": 4, "fret": 10, "note": "C"}}}

Table 4.3: Chord information of c:diminished

4.2.2 Music Theory

In our algorithm, we integrated music theory principles from www.fachords.com[2] to establish rules for chord substitution. This approach ensures that the algorithm accurately reflects established music theory, allowing for effective and harmonious chord substitutions in various musical contexts. This integration serves as a foundational aspect of the algorithm's design, enhancing its capability to simulate and suggest musically coherent and creative chord progressions.

We've incorporated a range of music theory principles for chord substitution, specifically designed to enable users to adapt songs to their preferred style, like jazz. This feature allows for seamless transition from original chord progressions to those commonly used in specific music genres. As a result, users can play any song in the style they enjoy, with the algorithm guiding them through stylistically appropriate chord substitutions that align with the unique characteristics of their chosen genre. Here are all music theory principle we use:

Song type	Reason	How
rock	Give us a different sound than C major	Replacing a major or minor chord with its sus2 or sus4 version (C to C sus2 or C sus4)
["funk", "boogie woogie", "R&B"]	Make it sound more interesting	Adding the 9th to a chord (C to C9)
["older swing", "jazz"]	-	Replacing the V7 with vii dim7 (G7 to B dim7)
jazz	A tritone substitution occurs whenever a dominant 7th chord is being substituted or replaced by another dominant 7th chord with a root a tritone interval away (basically you shift the chord 6 half-steps up)	Using Tritone Substitution
jazz	The subdominant IV and super tonic ii are also used to easily replace each other.	Subdominant and Super Tonic Substitution
["jazz", "classic rock"]	-	Leading Tone Substitution

Table 4.4: Music theory for chord substitution

4.2.4 Algorithm and Model

Our objective is to develop an algorithm that enhances the guitar self-training experience for users. The algorithm is intended to provide fingering positions for each chord, considering different skill levels and musical preferences. The primary objective is to facilitate smooth chord transitions with minimal finger movement. Here are overview of our algorithms:

1. **Fingering Feature Table:** Each fingering pattern of every chord is captured in a unique feature table. In this table, each row corresponds to a specific fingering pattern, and the columns include the following features:
 - a. 'index_finger_string': The string on which the index finger is positioned.
 - b. 'index_finger_fret': The fret at which the index finger is placed.
 - c. 'middle_finger_string': The string on which the middle finger is positioned.
 - d. 'middle_finger_fret': The fret at which the middle finger is placed.
 - e. 'ring_finger_string': The string on which the ring finger is positioned.
 - f. 'ring_finger_fret': The fret at which the ring finger is placed.
 - g. 'pinky_finger_string': The string on which the pinky finger is positioned.

- h. 'pinky_finger_fret': The fret at which the pinky finger is placed.
- i. 'index_middle_dist': The distance between the index and middle fingers.
- j. 'middle_ring_dist': The distance between the middle and ring fingers.
- k. 'ring_pinky_dist': The distance between the ring and pinky fingers.
- l. 'index_ring_dist': The distance between the index and ring fingers.
- m. 'middle_pinky_dist': The distance between the middle and pinky fingers.
- n. 'index_pinky_dist': The distance between the index and pinky fingers.
- o. 'average_dist': The average distance of fingers.
- p. 'string_coverage': The number of strings covered by the chord fingering.
- q. 'fret_coverage': The number of frets covered by the chord fingering.
- r. 'fret_start': The starting fret of the chord fingering.
- s. 'barre_chord': Indicates whether the fingering represents a barre chord (1 for yes, 0 for no).

Each feature provides specific information about the fingering position, allowing for a comprehensive representation of each chord pattern in the context of the feature table.

2. **Input and Option Selection:** Users input a sequence of chords, e.g., ["Bm", "F#m", ...]. Also, users have the flexibility to modify the 'style' of the song and adjust the simplicity of the chords through a dropdown menu. When selecting the 'style,' music theory principles tailored to the chosen style are implemented. Additionally, users can further customize their chord results using options such as 'Avoid Barre,' 'Lower fret only,' and 'Less Fingers,' available as filters for refining the chord output.
3. **Transition Mapping:** The algorithm maps transitions between chords, e.g., {"Bm": "F#m", "F#m": "D", "D": "Bm"}.
4. **Model Construction:** A model is developed to recommend fingering positions for each chord, optimizing for ease of transition.

We implemented clusterization loop as follows:

- a. For all possible rotations of pairs of **Transition Mapping**:
- b. For each pair in **Transition Mapping**:
 1. Utilize KMeans to cluster the fingering patterns of the next chord.
 2. Predict the cluster of the fingering pattern of the previous chord.
 3. Calculate Euclidean distances from the previous chord to each pattern within the same cluster.
 4. Choose the pattern with the least Euclidean distance and accumulate it to the total distance.

After the loop:

1. Rank the results of combinations of fingering by the total distance.
2. Filter out some combinations based on user input.
3. Recommend the combination with the least total distance.

4.3 Feedback Collection

4.3.1 User Interface

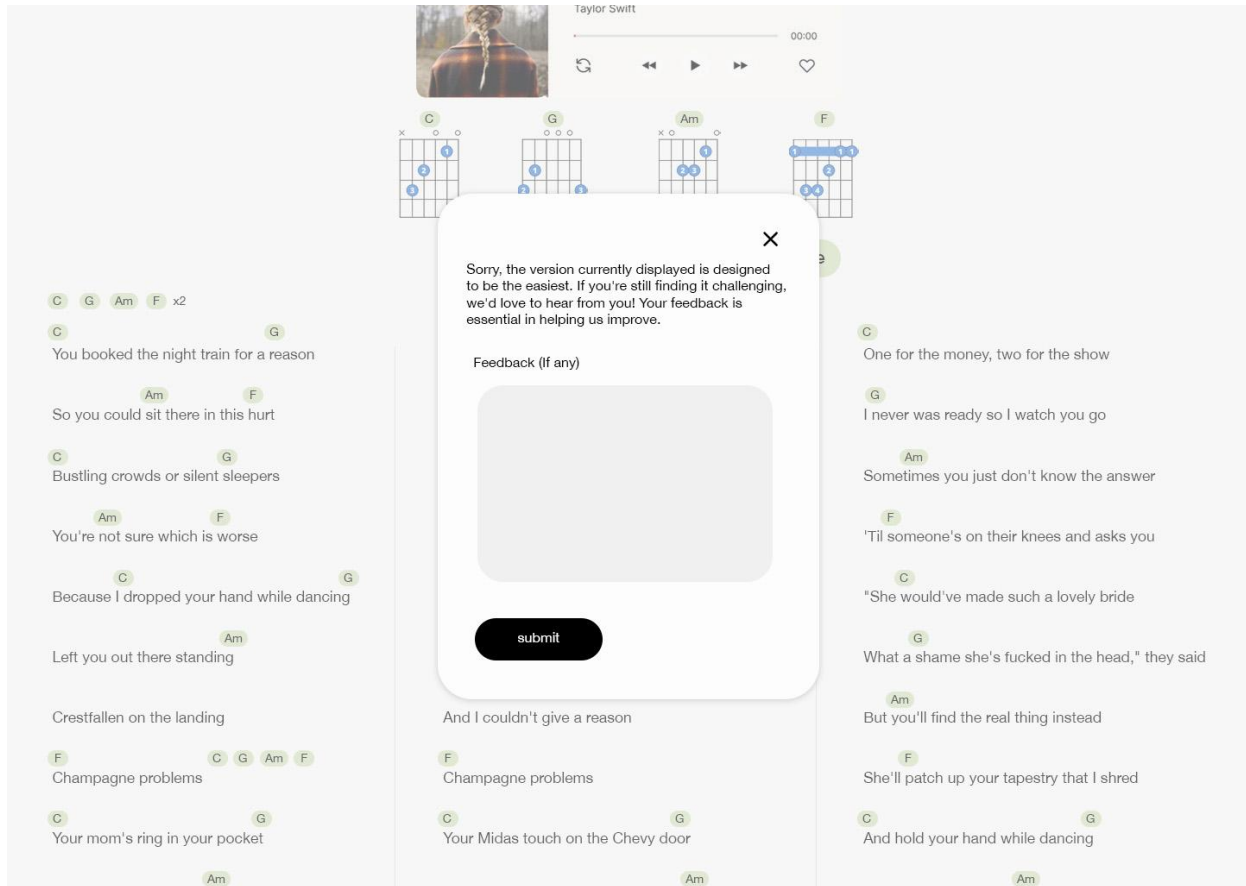


Figure 4.3: User interface of feedback collection

Chapter 5

Conclusion and Future work

5.1 Conclusion

This work introduces a model for generating chord progressions and finger placements. It takes a chord sequence as input and recommends optimal finger positions for each chord to minimize movement. Additionally, we propose an idea and prototype for a chord website system with a user interface, enhancing the overall user experience. ChordBrew is designed to assist beginners or users learning guitar independently. It aims to provide chords aligned with their skill level or match their preferred playing style.

5.2 Future Work

Our plans include developing chord transposition, expanding the chord database to encompass all chord types, and enhancing our algorithm. By gathering feedback, we gain insights into user experiences, enabling us to collect user ratings as labels for future model utilization. In summary, obtaining user feedback stands as a crucial factor for enhancing ChordBrew in the future.

Chapter 6

References

- [1] All Guitar Chords. (n.d.). Retrieved from <https://www.all-guitar-chords.com/>
- [2] Fachords. (n.d.). Chord Substitution.
Retrieved from <https://www.fachords.com/chord-substitution/#p1>
- [3] StringKick. (n.d.). Guitar Music Theory.
Retrieved from <https://www.stringkick.com/blog-lessons/guitar-music-theory/#intro>
- [4] The Acoustic Guitarist. (n.d.). Guitar Chord Theory.
Retrieved from <https://theacousticguitarist.com/guitar-chord-theory/#:~:text=open%20strings%20from%20ringing%20out>
- [5] Genius. (n.d.). Retrieved from <https://genius.com/>
- [6] E-Chords. (n.d.). Retrieved from <https://www.e-chords.com/>
- [7] Chords and Tabs. (n.d.). Retrieved from <https://www.chords-and-tabs.net/>
- [8] Ariga, S., Goto, M., & Yatani, K. (2017).
Strummer: An interactive guitar chord practice system.
<https://ieeexplore.ieee.org/abstract/document/8019338>
- [9] Raboanary, T.H., Randriamahenintsoa, F.H., Raboanary, H.A., Raboanary, T.M., &
Raboanary, J.A. (2017). Finding optimal bass guitar fingerings.
<https://ieeexplore.ieee.org/document/8095457/authors#authors>