Universitatea Tehnică a Moldovei

PRELUCRAREA ȘI REDAREA FIȘIERELOR AUDIO

PROCESSING AND PLAYING AUDIO FILES

Student: **gr. TI-194,**
Zavorot Daniel

Coordonator: **Cebotari Daria**
asistent universitar

Chişinău, 2023

MINISTERUL EDUCAŢIEI ŞI CERCETĂRII AL REPUBLICII MOLDOVA
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică şi Microelectronică
Departamentul Ingineria Software și Automatică

Admis la susţinere
Şef departament:
FIODOROV Ion dr., conf.univ.

„___" _____ 2023

PRELUCRAREA ȘI REDAREA FIȘIERELOR AUDIO

Proiect de licenţă

Student: _____ **Zavorot Daniel, TI-194**

Coordonator: _____ **Cebotari Daria, asist. univ.**

Consultant: _____ **Cojocaru Svetlana, asist.univ.**

Chişinău, 2023

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

Programul de studii Tehnologia informaţiei

Aprob
Șef departament:
Fiodorov Ion, dr., conf.univ.

_____

„ *06” octombrie 2022*

CAIET DE SARCINI
pentru proiectul de licenţă al studentului

Zavorot Daniel
(numele şi prenumele studentului)

1. Tema proiectului de licenţă *Prelucrarea și redarea fișierelor audio*

confirmată prin hotărârea Consiliului facultăţii nr. *2* **din** „ *06*” *octombrie 2022*

2. Termenul limită de prezentare a proiectului de licență *20.05.2023*

3. Date iniţiale pentru elaborarea proiectului de licență *Sarcina pentru elaborarea proiectului de diplomă.*

4. Conţinutul memoriului explicativ

Introducere
1 Analiza domeniului de studiu
2 Modelarea şi proiectarea sistemului
3 Realizarea sistemului
4 Documentarea produsului realizat
5 Estimarea costurilor proiectului
Concluzii

5. Conţinutul părţii grafice a proiectului de licență

Figura 2.1 - Procesele principale ale utilizatorului

6. Lista consultanţilor:

Consultant	Capitol	Confirmarea realizării activităţii	
		Semnătura consultantului (data)	Semnătura studentului
Cojocaru Svetlana	*Standarde tehnologice, Controlul calităţii, Estimarea costului proiectului*		

7. Data înmânării caietului de sarcini *02.09.2022*

Coordonator *Cebotari Daria* _____

semnătura

Sarcina a fost luată pentru a fi executată de către studentul *Zavorot Daniel*

02.09.2022

semnătura, data

PLAN CALENDARISTIC

Nr. crt.	Denumirea etapelor de proiectare	Termenul de realizare a etapelor	Nota
1	*Elaborarea sarcinii, primirea datelor pentru sarcină*	*02.09.22– 30.09.22*	*10%*
2	*Analiza domeniului de studiu*	*06.10.21– 30.11.21*	*15%*
3	*Proiectarea sistemului*	*01.12.21 – 25.12.21*	*15%*
4	*Realizarea sistemului*	*10.01.22 – 05.03.22*	*35%*
5	*Descrierea sistemului*	*06.03.22– 01.04.22*	*10%*
6	*Estimarea costurilor sistemului*	*02.04.22– 15.04.22*	*15%*
7	*Finisarea proiectului*	*16.04.22– 14.05.22*	*5%*

Student *Zavorot Daniel ( )*

Coordonator de proiect de licență *Cebotari Daria ( )*

UNIVERSITATEA TEHNICĂ A MOLDOVEI

FACULTATEA CALCULATOARE, INFORMATICĂ ŞI MICROELECTRONICĂ
DEPARTAMENTUL INGINERIA SOFTWARE ȘI AUTOMATICĂ
PROGRAMUL DE STUDII TEHNOLOGIA INFORMAŢIEI

AVIZ
la proiectul de licenţă

Titlul: Prelucrarea și redarea fișierelor audio.

Studentul Zavorot Daniel gr. TI-194

1. Actualitatea temei: În prezent, există o multitudine de aplicații desktop de tip music player disponibile pe piață, care oferă utilizatorilor posibilitatea de a asculta și organiza muzica lor preferată pe calculator. Cu toate acestea, popularitatea acestor aplicații a fost în declin în ultimii ani, odată cu creșterea utilizării serviciilor de streaming muzical, precum Spotify, Apple Music etc. Însă există încă oameni care preferă să-și gestioneze propria colecție de muzică și să o asculte de pe propriul lor calculator, ceea ce face ca aplicațiile desktop de tip music player să fie încă relevante și utile pentru acești utilizatori.

2. Caracteristica proiectului de licenţă: Aplicația a fost creată pentru ușurarea de prelucrăării și redării fișierelor audio.

3. Analiza prototipului: Aplicația data este formată pentru a o listă cu muzică personalizată care va fi disponibilă doar pe calculatorul local a utilizatorului. Cu ajutorul acestei liste, utilizatorul va putea asculta sau edita fișierele audio oricând el dorește cu ajutorul unei interfețe grafice.

4. Estimarea rezultatelor obţinute: Acest program este creat pentru prelucrarea și redarea fișierelor audio. Este un program usor in utilizare si intuitiv pentru orce posesor a unui calculator personal.

5. Corectitudinea materialului expus: Materialul expus este prezentat prin referinţe ale unor surse ce au fost scrise de persoane ce deţin experienţa în domeniul Tehnologiilor Informaţionale.

6. Calitatea materialului grafic: Proiectul este prezentat prin: diagrame, tabele, interfeţe ale aplicaţiei.

7. Valoarea practică a proiectului: Este destinat pentru utilizatorilor ce dețin calculatoare personale care rulează sistemul de operare Windows sau Linux.

8. Observaţii şi recomandări: Cerinţele faţă de teza de licenţă au fost îndeplinite în totalitate. Observaţii nu sunt.

9. Caracteristica studentului şi titlul conferit : Studentul Zavorot Daniel a dat dovadă de profesionalism în elaborarea lucrării, a respectat cerinţele impuse şi a elaborat calitativ teza de licență. Din cele relatate, urmează că lucrarea de licenţă poate fi admisă spre susţinere, cu nota 10

Lucrarea în forma electronică corespunde originalului prezentat către susținere publică.

Coordonatorul proiectului de licenţă _____Cebotari Daria, asist.univ.

semnătura, data

REZUMAT

Această lucrare conține cinci capitole care vor oferi o descriere detaliată a sistemului în cauză, sub formă de subcapitole.

Primul capitol, "Analiza domeniului de studiu", se va face o prezentare a domeniului în care se încadrează sistemul, precum și a problemei care a dus la dezvoltarea sa. Vor fi abordate aspecte legate de contextul general în care sistemul va fi utilizat și se va analiza modul în care acesta va aduce beneficii pentru utilizatori și pentru domeniul în cauză. În cadrul comparației cu alte trei sisteme deja existente, se vor evidenția similaritățile și diferențele între acestea. Se va face o analiză a avantajelor și dezavantajelor fiecăruia dintre aceste sisteme. Scopul, obiectivele și cerințele sistemului vor fi definite în baza analizei efectuate în subcapitolul anterior,unde vor fi stabilite criterii de performanță și calitate.

Al doilea capitol, "Modelarea și proiectarea sistemului informatic", subcapitolul referitor la descrierea comportamentală a sistemului, se vor prezenta diagramele de secvențe și diagrama de stări pentru a ilustra modul în care sistemul interacționează cu utilizatorul și cu diferitele componente ale sale. Aceste diagrame vor fi utile pentru a înțelege modul în care sistemul gestionează procesele și evenimentele. În subcapitolul referitor la descrierea structurală a sistemului, se vor prezenta diagramele de clasă și diagrama de componente, care vor arăta structura fizică și logică a sistemului.

Al treilea capitol, "Realizarea sistemului", se va detalia procesul de realizare al sistemului informatic. Se va începe prin prezentarea limbajului de programare utilizat pentru crearea aplicației, împreună cu motivele pentru care a fost ales acesta și avantajele utilizării acestui limbaj. De asemenea, se va descrie framework-ul principal utilizat în cadrul proiectului, precum și alte framework-uri utilizate pentru a ajuta la dezvoltarea aplicației. Se va oferi o descriere detaliată a modului în care aceste framework-uri au fost integrate cu limbajul de programare utilizat. Un alt aspect important care va fi prezentat în acest capitol este tipul de bază de date utilizată pentru a stoca informațiile necesare funcționării aplicației.

Capitolul patru, "Documentarea produsului realizat", se va realiza documentația completă a produsului final. Se vor prezenta cerințele tehnice necesare pentru utilizarea aplicației, cum ar fi specificațiile hardware și software, sistemul de operare recomandat și cerințele de conexiune la internet, dacă este cazul. De asemenea, se vor detalia pașii necesari pentru instalarea și configurarea aplicației, începând de la descărcarea fișierelor până la finalizarea procesului de instalare. În acest sens, se vor oferi instrucțiuni clare și ușor de urmat pentru utilizatorul final. În continuare, se va prezenta o descriere detaliată pentru fiecare funcție a aplicației, inclusiv modalitățile de accesare și utilizare.

Ultimul capitol, "Estimarea costurilor proiectului", se vor analiza în detaliu costurile asociate cu dezvoltarea și implementarea sistemului în cauză. Se vor lua în considerare aspecte precum costurile de cercetare și dezvoltare, costurile cu echipamentele și software-ul necesare, costurile de personal, precum și costurile cu testarea și implementarea sistemului. În ceea ce privește estimarea costurilor, se va stabili un buget aproximativ pentru fiecare etapă a proiectului.

ABSTRACT

This work consists of five chapters that will provide a detailed description of the system in question, in the form of subchapters.

The first chapter, "Analysis of the study domain," will present an overview of the field in which the system falls, as well as the problem that led to its development. Aspects related to the general context in which the system will be used will be addressed, and the way in which it will benefit users and the field in question will be analyzed. In the comparison with three other existing systems, similarities and differences between them will be highlighted. An analysis of the advantages and disadvantages of each of these systems will be made. The system's purpose, objectives, and requirements will be defined based on the analysis conducted in the previous subchapter, where performance and quality criteria will be established.

The second chapter, "Modeling and design of the information system," the subchapter referring to the behavioral description of the system, will present sequence diagrams and state diagrams to illustrate how the system interacts with the user and its various components. These diagrams will be useful in understanding how the system manages processes and events. In the subchapter referring to the structural description of the system, class diagrams and component diagrams will be presented, which will show the physical and logical structure of the system.

The third chapter, "Realization of the system," will detail the process of creating the information system. It will begin by presenting the programming language used to create the application, along with the reasons why it was chosen and the advantages of using this language. Additionally, the main framework used in the project will be described, as well as other frameworks used to assist in the development of the application. A detailed description will be provided on how these frameworks were integrated with the programming language used. Another important aspect that will be presented in this chapter is the type of database used to store the information necessary for the application to function.

Chapter four, "Documentation of the product produced," will create complete documentation of the final product. The technical requirements necessary for using the application, such as hardware and software specifications, recommended operating system, and internet connection requirements, if applicable, will be presented. Additionally, the necessary steps for installing and configuring the application will be detailed, starting from downloading the files to completing the installation process. In this regard, clear and easy-to-follow instructions will be provided for the end-user. Furthermore, a detailed description will be provided for each function of the application, including how to access and use them.

The last chapter, "Estimating project costs," will thoroughly analyze the costs associated with the development and implementation of the system in question. Considerations will include research and development costs, equipment and software costs, personnel costs, as well as testing and implementing the system. Regarding cost estimation, an approximate budget will be established for each stage of the project.

CUPRINS

INTRODUCERE

Playerele de muzică au fost aplicații populare pentru desktop timp de ani de zile și continuă să fie utilizate de milioane de oameni din întreaga lume. În ciuda disponibilității diferitelor servicii de streaming de muzică, mulți oameni preferă să utilizeze aplicațiile de redare a muzicii pentru desktop pentru a asculta muzică. Există mai multe motive pentru care utilizatorii preferă aplicațiile de redare a muzicii pentru desktop. În primul rând, aceste aplicații oferă o experiență de ascultare a muzicii mai personalizată. Utilizatorii pot crea liste de redare personalizate și își pot organiza muzica pe computerul lor în funcție de preferințele lor. În plus, aplicațiile pentru desktop oferă o calitate audio mai bună decât serviciile de streaming online, ceea ce poate fi important pentru utilizatorii care apreciază sunetul de înaltă calitate. De asemenea, utilizatorii pot asculta muzica preferată fără a plăti pentru un serviciu de streaming sau a avea acces la internet.

În acest proiect, vom explora procesul de dezvoltare a unei astfel de aplicații de redare a fișierelor audio. Vom analiza diversele tehnologii și instrumente necesare pentru a crea o astfel de aplicație, vom examina aspectele de design și vom explora cum să asigurăm o experiență plăcută și prietenoasă pentru utilizator. O astfel de aplicație ar trebui să conțină butoanele minime necesare, cum ar fi „redare”, „pauză”, „următorul” și „anteriorul”. Ar trebui, de asemenea, să aibă cel puțin un glisor (slider) pentru volum și să afișeze poziția curentă a fișierului care este redat. Pentru o aplicație relativ modernă, aceasta ar trebui să includă funcții precum modul de amestecare (shuffle), repetare o dată și repetare pentru lista de melodii încărcate, editarea titlului, artistului și / sau copertei albumului pentru melodie și, în cele din urmă, ștergerea fișierelor audio din lista aplicației., editarea titlului, artistului și/sau coperta albumului și, în cele din urmă, ștergerea fișierelor audio din lista aplicației. Sistemul dat oferă aceste funcționalități.

Procesul de dezvoltare a unei aplicații de redare a fișierelor audio implică mai multe etape, inclusiv planificarea, proiectarea, dezvoltarea, testarea și implementarea. Faza de planificare implică definirea obiectivelor proiectului, identificarea cerințelor utilizatorilor și selectarea tehnologiei potrivite. În timpul fazei de proiectare, se creează interfața aplicației și se specifică diferitele caracteristici. Faza de dezvoltare implică implementarea caracteristicilor și crearea codului necesar. În faza de testare, aplicația este testată pentru funcționalitate. În cele din urmă, în faza de implementare, aplicația este lansată pentru utilizatori.

Pentru a crea o aplicație eficientă de redare a fișierelor audio, este esențial să se utilizeze tehnologii adecvate, cum ar fi limbajul de programare Python și biblioteca PyQt5 pentru crearea interfeței grafice și manipularea fișierelor audio. De asemenea, aplicația ar trebui să fie proiectată având în vedere experiența utilizatorului, cum ar fi o interfață clară și intuitivă, un design responsiv și o navigare fără probleme.

Prin urmare o aplicație de redare a fișierelor audio poate fi o aplicație utilă pentru multe persoane care doresc să asculte muzica preferată pe desktopul lor.

1 ANALIZA DOMENIULUI DE STUDIU

În fiecare an tehnologiile informaționale mereu avansează în toate domeniile cum ar fi multimedia, lumea afacerilor etc. Indiferent de domeniul, companiile fie mici, fie mari folosesc cel puțin o programa de tip multimedia.

Progresele rapide în capacitățile tehnologiei de calcul, în special în domeniul comunicațiilor, au definit limitele unei alte domenii de aplicație: sistemele multimedia. Această zonă se referă la aplicații de vizualizare care modelează fenomene complexe în timp real, permit prezentarea și gestionarea documentelor într-o varietate de moduri și sprijină inginerie competitivă etc.. Funcția esențială a sistemelor multimedia se poate spune că este integrarea semnalelor de date, audio și video în aplicații.[1]

Fișierele de sunet redă vorbire, muzică de fundal sau efecte speciale mono sau stereo. Poate fi încorporat în prezentări slide-show, animații, secvențe video, site-uri web, dar poate fi și singur ca o carte audio. Editorii tradiționali și-au extins domeniul de activitate în această direcție, chiar și pe subiecte istorice. Fișierele audio sunt foarte importante pentru cercetarea istoriei orale. Multe instituții, biblioteci și arhive sunt implicate în proiecte de acest gen.[2]

Următoarele sunt componentele comune ale multimedia:

- text - toate producțiile multimedia conțin o anumită cantitate de text. Textul poate avea diferite tipuri de fonturi și dimensiuni;
- grafică - face aplicația multimedia atractivă. În multe cazuri, oamenilor nu le place să citească o cantitate mare de text pe ecran, prin urmare, graficele sunt folosite mai des decât textul pentru a explica un concept, pentru a prezenta informații de fundal etc;
- audio - o aplicație multimedia poate necesita utilizarea vorbirii, a muzicii și a efectelor sonore. Acestea sunt numite elemente audio sau audio ale multimedia;
- video - termenul video se referă la imaginea în mișcare, însoțită de sunet, cum ar fi o imagine la televizor. Elementul video al aplicației multimedia oferă o mulțime de informații într-o perioadă mică de timp;
- animație - este un proces de a face ca o imagine statică să pară în mișcare. O animație este doar o serie continuă de imagini statice care sunt afișate într-o secvență.[3]

După dezvoltarea acestui proiect, o sa obținem un produs de tip aplicație desktop care poate fi accesat de pe orice dispozitiv care acceptă fișiere „exe”. Prin aplicație desktop mă refer la un program care va fi citit și executat pe un sistem de operare care permite executarea unor astfel de programe. Exemple de astfel de tipuri de sisteme de operare sunt: Windows aparținând companiei Microsoft, Linux care este open-source și creat de Linus Torvalds etc.

O aplicație desktop este un program software care poate fi rulat pe un computer independent pentru a efectua o anumită sarcină de către un utilizator final. Unele aplicații desktop, cum ar fi editorul de cuvinte, aplicația de editare foto și playerul media vă permit să efectuați diferite sarcini, în timp ce

altele, cum ar fi aplicațiile de jocuri, sunt dezvoltate exclusiv pentru divertisment. Când achiziționați un computer sau un laptop, există un set de aplicații care sunt deja instalate pe desktop. De asemenea, puteți descărca și instala diferite aplicații desktop direct de pe Internet sau achiziționate de la furnizorii de software. Exemple unora dintre aplicațiile desktop populare sunt aplicații de procesare a textului, cum ar fi Microsoft Word și WPS Office, care sunt concepute pentru a edita conținutul textual, aplicații de jocuri precum Minesweeper și Solitaire care sunt utilizate pentru divertisment, browserele web precum Internet Explorer, Chrome și Firefox care vă ajută să vă conectați la Internet de pe computer, aplicații media player precum iTunes, Windows Media Player și VLC media player care vă permit să ascultați muzică, să vizionați videoclipuri și filme și să creați colecții de conținut media.[4]

1.1 Importanța temei

Aplicațiile multimedia au devenit o parte integrantă a vieții noastre de zi cu zi, iar aplicația Music Player nu face excepție. Misiunea sa este de a oferi utilizatorilor un pachet de caracteristici pentru redarea și gestionarea fișierelor audio pe mașina lor locală. Cu Music Player, utilizatorii își pot organiza, modifica și reda cu ușurință melodiile și albumele preferate cu doar câteva clicuri.

Unul dintre cele mai semnificative avantaje ale acestei aplicații este capacitatea sa de a adăuga, edita și șterge automat fișiere audio. Când un utilizator selectează un fișier audio, Music Player oferă automat un titlu și un nume de artist pe baza informațiilor preluate din fișier. Utilizatorul poate modifica aceste informații sau le poate accepta prin apăsarea butonului „Next”. Pentru cei care au încredere în abilitățile aplicației, butonul „Skip all” poate fi folosit pentru a accepta toate modificările oferite de aplicație pentru toate fișierele audio selectate.

Accesibilitatea Music Player este, de asemenea, un aspect esențial al aplicației. În lumea de astăzi, nu toate aplicațiile existente pot oferi o astfel de funcționalitate gratuit. De cele mai multe ori, interfața cu utilizatorul este foarte simplă sau funcționalitatea este slabă. Cu toate acestea, aplicația Music Player este concepută pentru a atinge un echilibru între caracteristicile de redare și gestionare a fișierelor audio și designul modern oferit utilizatorului.

Pe lângă caracteristicile sale de bază, Music Player are o gamă de opțiuni avansate care permit utilizatorilor să-și ajusteze experiența de ascultare. De exemplu, utilizatorul poate configura o listă de redare sau poate amesteca melodiile aleatoriu, poate ajusta setările egalizatorului. În plus, Music Player acceptă o gamă largă de formate foto pentru pozele de album, dar pentru formate audio doar fișierele cu extensia MP3.

În cele din urmă, aplicația Music Player este, de asemenea, optimizată pentru performanță și stabilitate. Funcționează fără probleme pe majoritatea mașinilor fără a consuma resurse excesive și nu se

blochează sau face crash. Mai mult, Music Player este actualizat regulat cu remedieri de erori și funcții noi pentru a asigura cea mai bună experiență posibilă de utilizator.

Prin urmare aplicația Music Player este un instrument excelent pentru oricine iubește muzica și își dorește un player multimedia puternic, dar ușor de utilizat. Gestionarea automată a fișierelor și designul modern îl fac să iasă în evidență față de alte aplicații similare, iar caracteristicile sale avansate și flexibilitatea îl fac ideal pentru utilizatorii cu nevoi și preferințe diferite. Indiferent dacă ești un ascultător ocazional sau un muzician profesionist, Music Player este o aplicație care merită încercată.

1.2 Sisteme similare cu proiectul realizat

Sunt o mulțime de sisteme existente care oferă posibilități diferite si arată diferit, însă majoritatea au integrat reclamă sau o mulțime de funcționalități care ocupa memoria calculatorului. De asemenea sunt sisteme care sunt cu plata. Mai jos sunt enumerate trei sisteme deja existente care fac parte din categoria proiectului meu:

1. Groove Music
2. Spotify
3. iTunes

În figura 1.1. Aplicația „Groove Music” este reprezentat player-ul integrat a sistemului de operare Windows. Unul dintre avantajele utilizării player-ului integrat Groove Music pe Windows este design-ul și funcționalitatea sa de nivel înalt. Așa cum s-a menționat anterior, însă, ocupă memorie suplimentară pentru toate funcțiile sale pe care utilizatorul nu le poate folosi. În plus, este complet gratuit și nu include nicio reclamă sau funcționalitate similară, spre deosebire de alte sisteme care pot urma.

Un dezavantaj al aplicației Groove Music este incapacitatea sa de a încărca fișiere audio din alte directoare. Utilizatorii pot selecta doar un director specific în care se află toate fișierele audio, directorul implicit fiind "Music", creat automat de sistemul de operare Windows. Această limitare poate fi problematică pentru utilizatorii care au fișiere audio stocate în mai multe locații sau cei care preferă să își organizeze fișierele într-un mod specific.

În comparație, sistemul meu Music Player are avantajul de a putea încărca fișiere audio din orice director de pe computerul utilizatorului. Aceasta oferă utilizatorilor o flexibilitate mai mare și control asupra bibliotecii lor de muzică. În plus, sistemul Music Player permite utilizatorilor să creeze și să administreze ușor liste de redare, să editeze detalii despre cântec, cum ar fi titlu, artist și imagine de copertă, și să șteargă cântece din biblioteca lor, dacă este cazul.

În ansamblu, în timp ce aplicația Groove Music poate avea unele avantaje, sistemul Music Player oferă o flexibilitate și un control mai mare asupra fișierelor audio și organizării acestora, făcându-l o opțiune mai potrivită pentru utilizatorii care necesită mai multe opțiuni de personalizare.

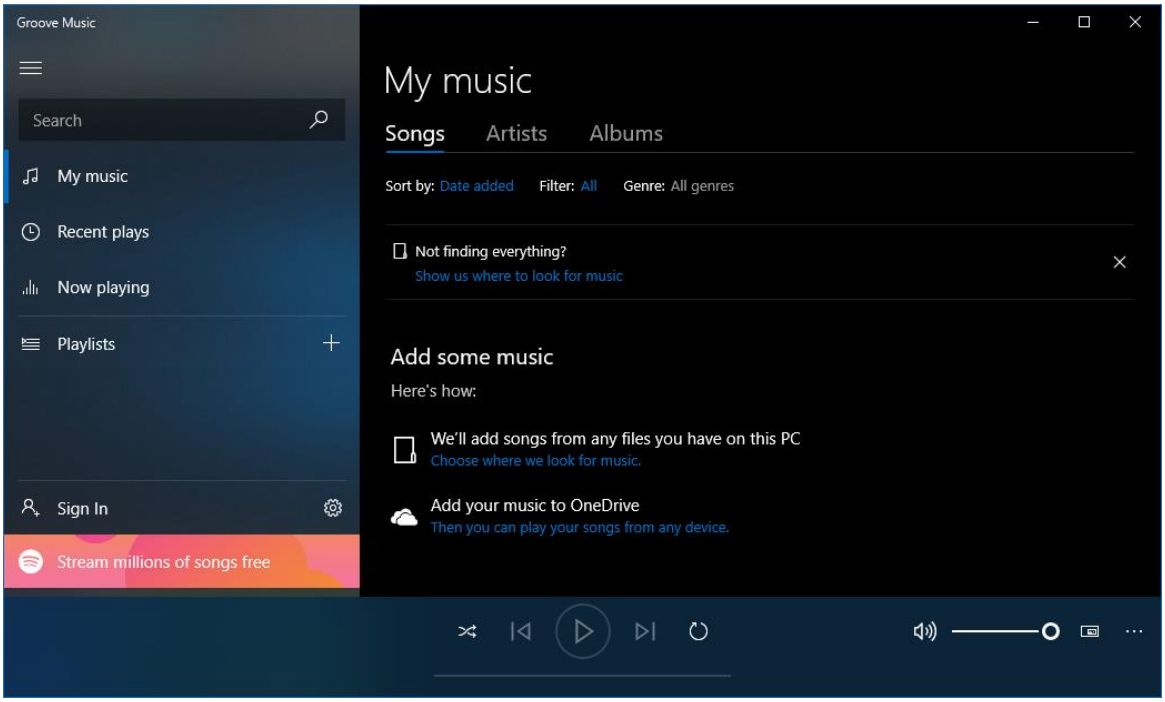

Figura 1.1 - Aplicația „Groove Music”

În figura 1.2. aplicația „Spotify” este reprezentata Aplicația „Spotify” care este una dintre cele mai populare aplicații de streaming de muzică din lume și are o bibliotecă uriașă de melodii. Cu toate acestea, funcționalitățile sale sunt limitate în comparație cu alte aplicații de acest gen. De exemplu, utilizatorii nu pot asculta muzica stocată local și nu pot edita informațiile despre melodiile din biblioteca lor. De asemenea, utilizatorii trebuie să plătească pentru a asculta muzică fără publicitate. În caz contrar, vor fi nevoiți să asculte reclame după fiecare cântec. Deși aplicația Spotify ocupă puțin spațiu de stocare fizic pe dispozitivul utilizatorului, memoria RAM folosită de aceasta este mai mare decât cea a altor aplicații asemănătoare. În ceea ce privește design-ul, aplicația Spotify are un aspect plăcut și este ușor de utilizat. Cu toate acestea, ca și în cazul altor aplicații similare, utilizatorii trebuie să fie conectați la internet pentru a asculta muzica. În general, aplicația Spotify este potrivită pentru utilizatorii care doresc să acceseze o bibliotecă vastă de muzică online și care sunt dispuși să plătească pentru a elimina reclamele. Cu toate acestea, pentru utilizatorii care preferă să-și asculte muzica stocată local și care doresc mai multe opțiuni de personalizare și control asupra bibliotecii lor de muzică, există alternative mai bune.

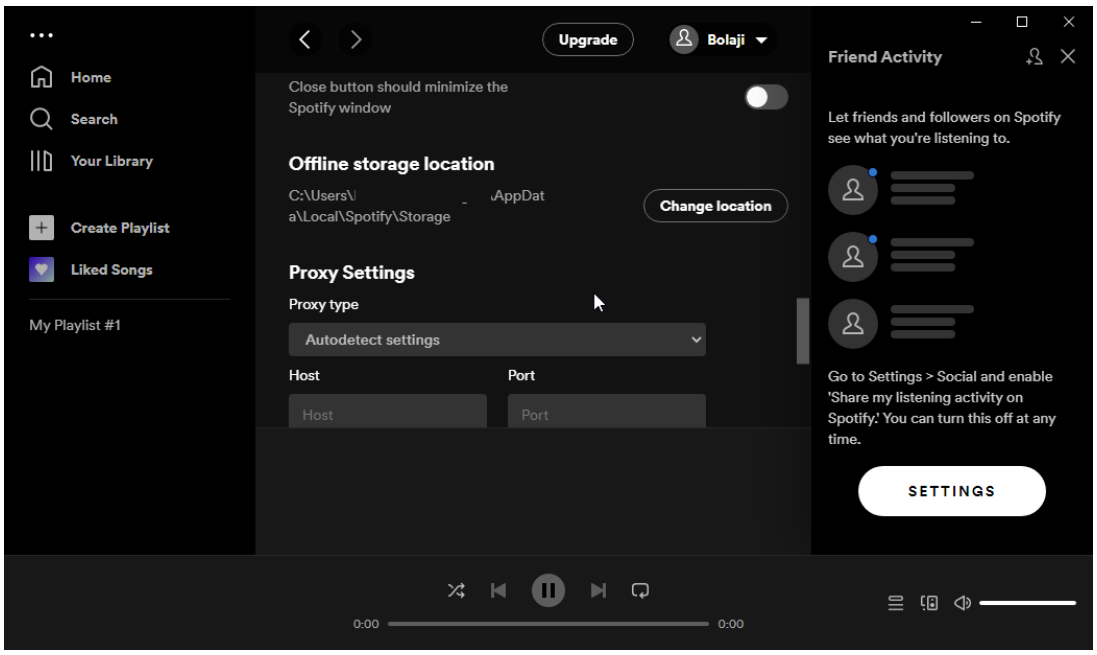

Figura 1.2 - Aplicația „Spotify”

În figura 1.3. aplicația „iTunes” este reprezentată o aplicație făcută de compania Apple inc. care ca si Spotify-ul oferă o biblioteca cu muzica larga pentru o plata lunara. Din experiența proprie pot spune că biblioteca la iTunes este mai mică decât la Spotify, dar plata este la fel. De asemenea design-ul pentru mine este unul deja învechit. La fel ca și sistemul anterior fișierele audio de pe local nu pot fi încărcate. Nu în ultimul rând, sistemele de mai sus la care este posibil încărcarea fișierelor audio de pe mașina locala nu fac copie la file-ul ales de utilizator, adică pe viitor dacă vom șterge directoria unde se află file-ul audio, el va nimeri în coșul de gunoi și nu o să putem să-l accesăm. Aplicația mea a rezolvata aceasta problemă prin copierea file-ului sau a mai multor file-uri deodată in directoria aplicației, prin urmare el va ocupa o memorie mai mare decât sistemele precedente, însă fișierele audio vor fi accesibile până când nu vom șterge aplicația dată de pe sistemul de operare.

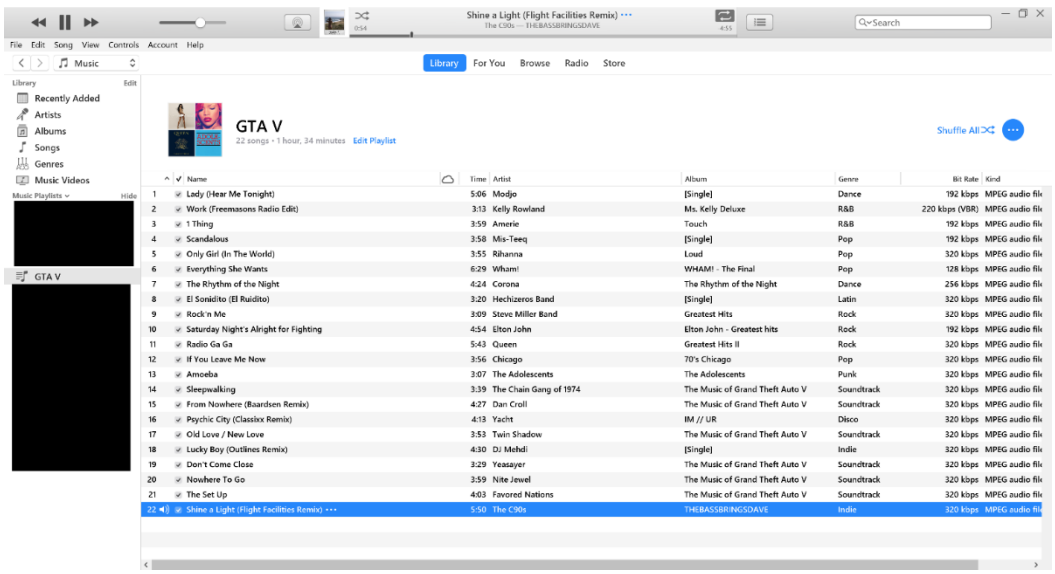

Figura 1.3 - Aplicația „iTunes”

În figura 1.4. meniul principal a aplicației „Groove Music” putem observa aplicației „Groove Music” împreuna cu funcționalul acesteia. Meniul conține o lista cu toate fișierele audio încărcate de către utilizator, de asemenea in partea de jos a aplicație se afla tot funcționalul de baza (Butoanele, Slide-urile, Cover-ul fișierului audio etc.). Interfața aplicației date conține o mulțime de butoane, care dup apararea mea sunt in plus. Aplicația „Music Player” va oferi doar un funcțional de bază pentru citirea, redarea și nu în ultimul rând gestionarea fișierelor audio.

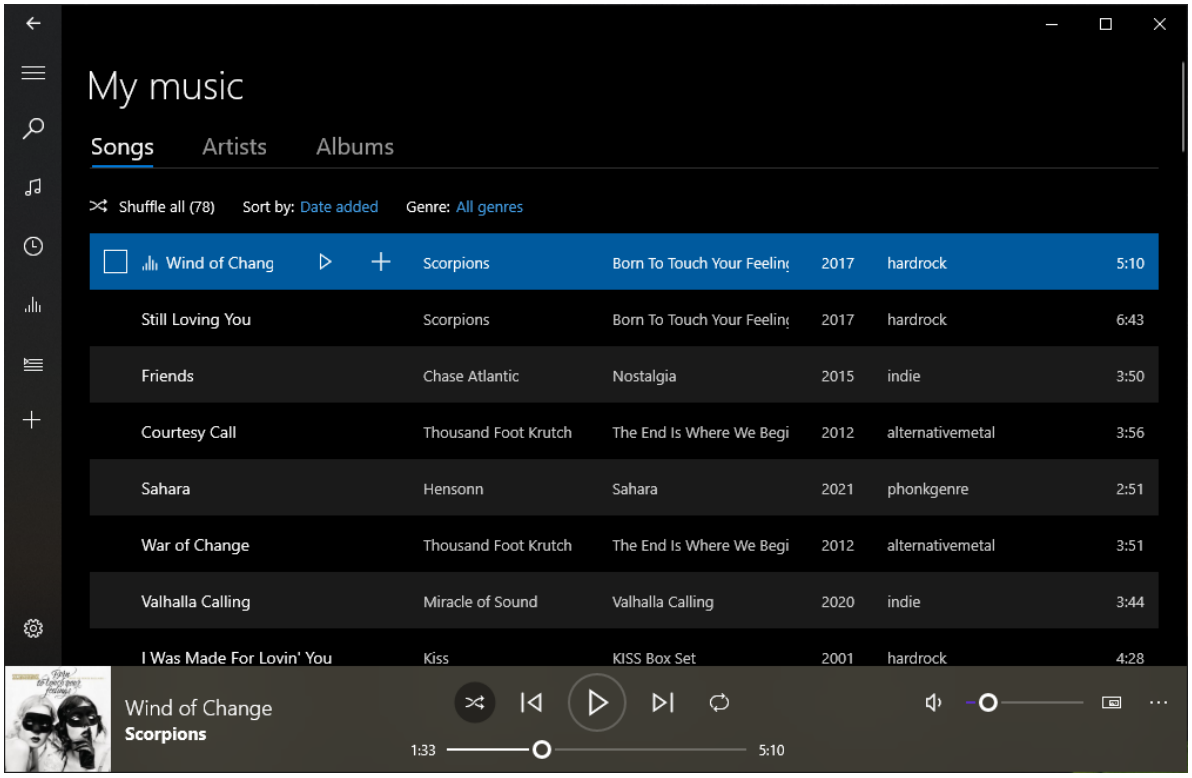

Figura 1.4 - Meniul principal a aplicației „Groove Music”

În figura 1.5. tray meniul SO Windows este reprezentata aplicația „Groove Music”. Cum putem observa, aplicația nu poate funcționa pe fundal. Prin funcționare pe fundal am in vedere ca daca închidem aplicația, ea nu va apărea in tray meniu, prin urmare se va închide total. Aplicația mea va avea funcția ca daca închidem aplicația, ea va funcționa pe fundal pana la momentul când utilizatorul nu va alege opțiunea „Exit” din meniul propus in urma apăsării butonului click-drept a mouse-ului pe icon-ul aplicației ce se afla in tray meniul sistemului de operare.

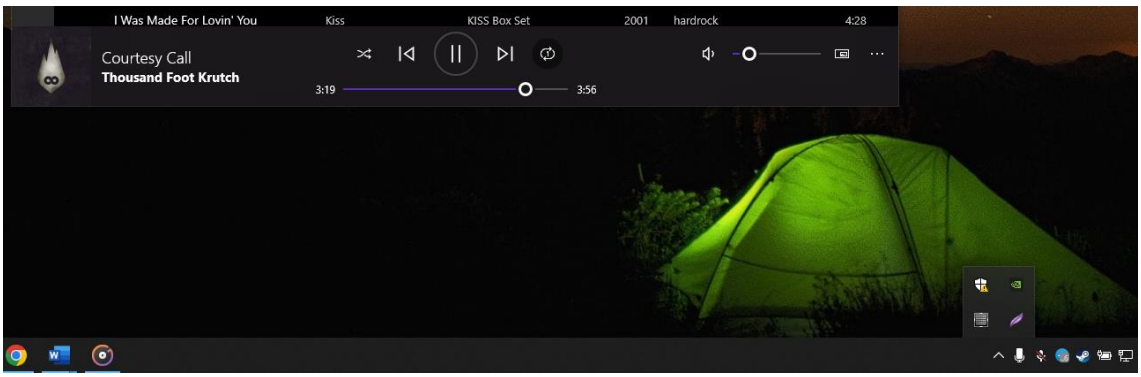

Figura 1.5 - Tray meniul SO Windows

În figura 1.6. funcția de adăugare a fișierelor audio a aplicației „Groove Music” este arătata funcția de adăugare a fișierelor audio in aplicația „Groove Music”. Cum putem vedea, aplicația nu adaugă fișierele propriu-zis, dar directoriul unde ele se afla, prin urmare utilizatorul trebuie manual sa le adauge in directoria aleasa. Aplicația mea va oferi posibilitatea sa alegem una sau mai multe fișiere de-odată din directorii diferite, deoarece după ce alegem fișierele, aplicația automat face cate-o copie pentru fiecare si le adaugă in directoria aplicației, prin urmare daca vom șterge fișierele selectate din directoriile de baza, ele nu se vor șterge din aplicație, deoarece exista copie pentru fiecare fișier.

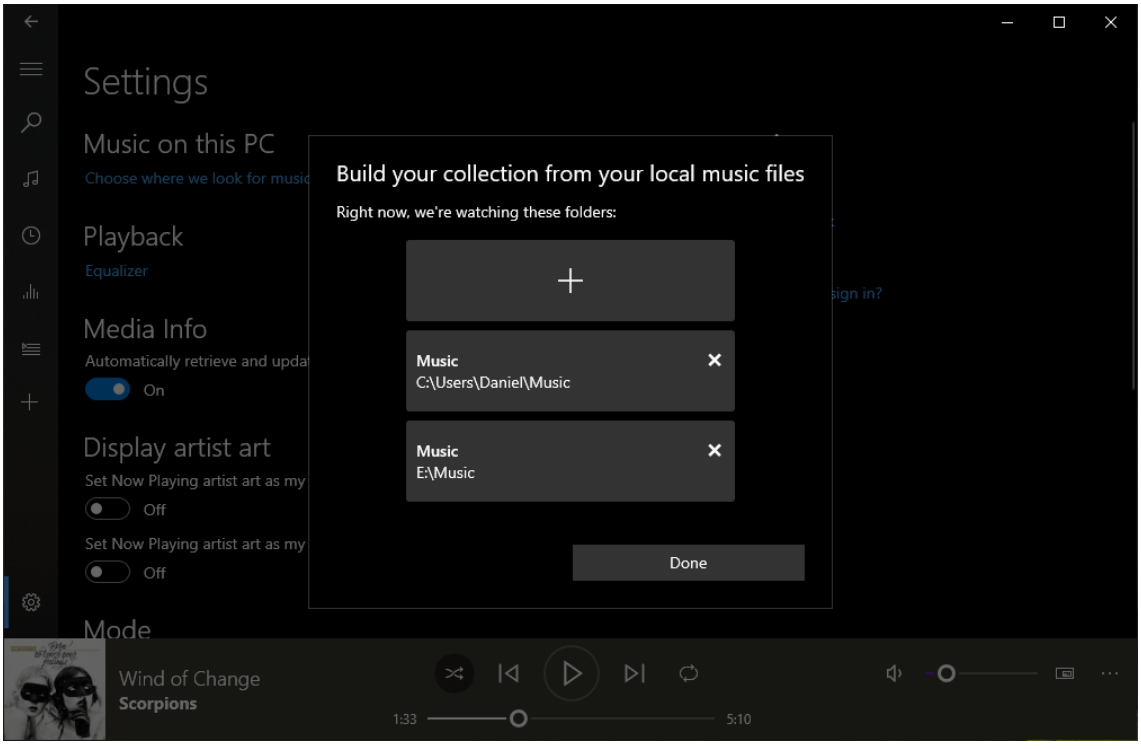

Figura 1.6 - Funcția de adăugare a fișierelor audio a aplicației „Groove Music”

În figura 1.7. funcțiile de baza a aplicației „Groove Music” se afla funcționalul de baza a aplicației „Groove Music”. Aici se afla butoanele ce oferă posibilitatea de gestionare a fișierelor audio, cum ar fi:

butonul „Play”, butonul „Pause”, „Next music”, „Prev. Music”, „Mute volume”. Pe lângă butoanele ce oferă funcțional pentru manipularea fișierelor audio, exista butoane pentru modificarea modului de redare a fișierelor cum ar fi: „Shuffle Mode”, „Repeat Once” si „Reapeat This”. Fiecare din ele schimba modul de redare a fișierelor, de exemplu „Shuffle Mode” reda fișierele încărcate in mod aleatoriu, „Repeat Once” oprește player-ul după ce se termina cântecul curent, iar „Repeat This” pune intr-un „loop” infinit cântecul curent.

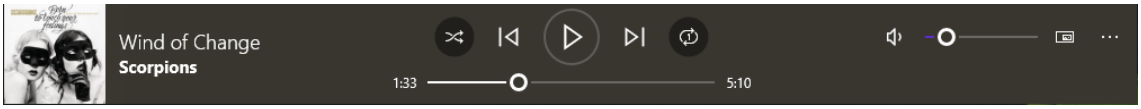

Figura 1.7 - Funcțiile de baza a aplicației „Groove Music”

În figura 1.8. ajustarea volumului in aplicația „Groove Music” este arătat modul de manipulare a volumului canalului audio unde sunt redate fișierele audio din aplicația „Groove Music”. Cum putem vedea, după ce apăsam pe butonul „Mute”, slide-ul volumului nu se duce la 0, este un bug acesta a aplicației sau nu, dar aplicația care va fi proiectata de mine va avea funcția de setare a slide-ului volumului la 0 daca va fi apăsat butonul „Mute”, iar daca vom apasă din nou butonul „Mute” se va seta ultima valoare concomitent cu slide-ul ce reprezintă volumul in aplicație.

Figura 1.8 - Ajustarea volumului in aplicația „Groove Music”

În figura 1.9. funcția de editare a unui fișierului audio in aplicația „Groove Music” putem observa funcția de editare a fișierului audio selectat a aplicației „Groove Music”. Din imagine putem vedea ca oferă posibilitatea la schimbarea denumirii fișierului, schimbarea artistului, schimbarea albumului, genului, anul apariției etc. De asemenea oferă adresa fișierului selectat, in cazul nostru fișierul se afla pe discul „E” in directoria „Music”. Aplicația mea va oferi posibilitatea de asemenea la redactarea fișierului selectat. Va fi posibila schimbarea denumirii fișierului, numele artistului. Pe lângă acest funcțional, va oferi posibilitate la schimbarea cover-ului cântecului ce nu oferă aplicația creata de compania Microsoft „Groove Music”. Prin cuvântul „cover” mă refer la imaginea setata pentru fiecare fișier audio, in aplicația „Groove Music” cover-ul cântecului selectat se afla in stânga-jos a ferestrei aplicației.

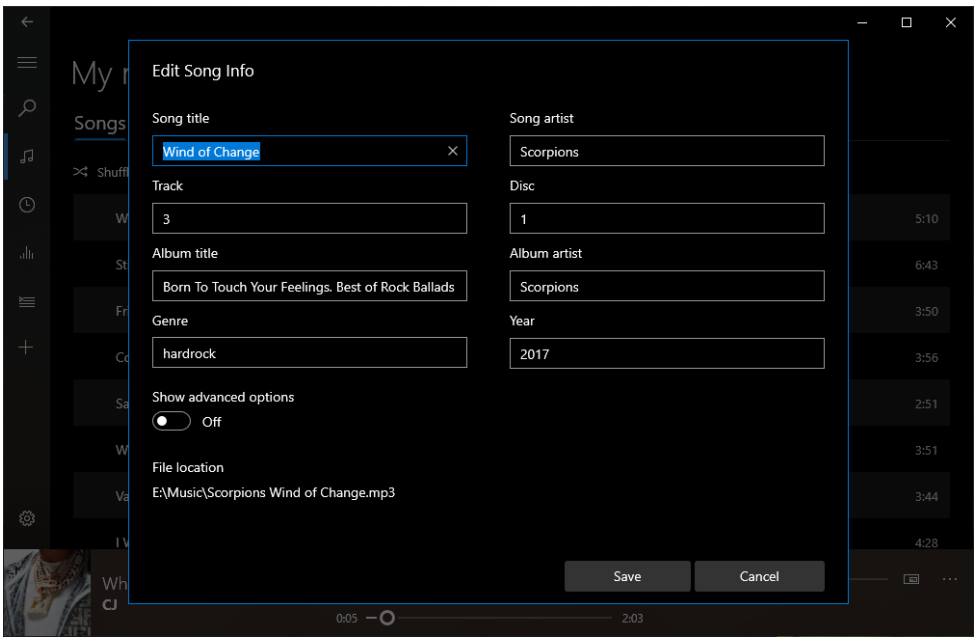

Figura 1.9 - Funcția de editare a unui fișierului audio in aplicația „Groove Music”

În figura 1.10. funcția de ștergere a unui fișierului audio in aplicația „Groove Music” putem vedea funcția de ștergere a aplicației „Groove Music”. Daca alegem opțiunea „Delete” va apărea o fereastra pop-up pentru confirmarea ștergerii a fișierului selectat. După ștergerea lui, el automat se va șterge din directoriul unde se afla, adică va dispărea de pe mașina locala. Aplicația mea va oferă fix așa funcție, doar ca după ștergerea fișierului selectat, aplicația va șterge copia ce a fost creata in directoriul aplicația, iar originalul va rămâne neatins pe mașina locala a utilizatorului.

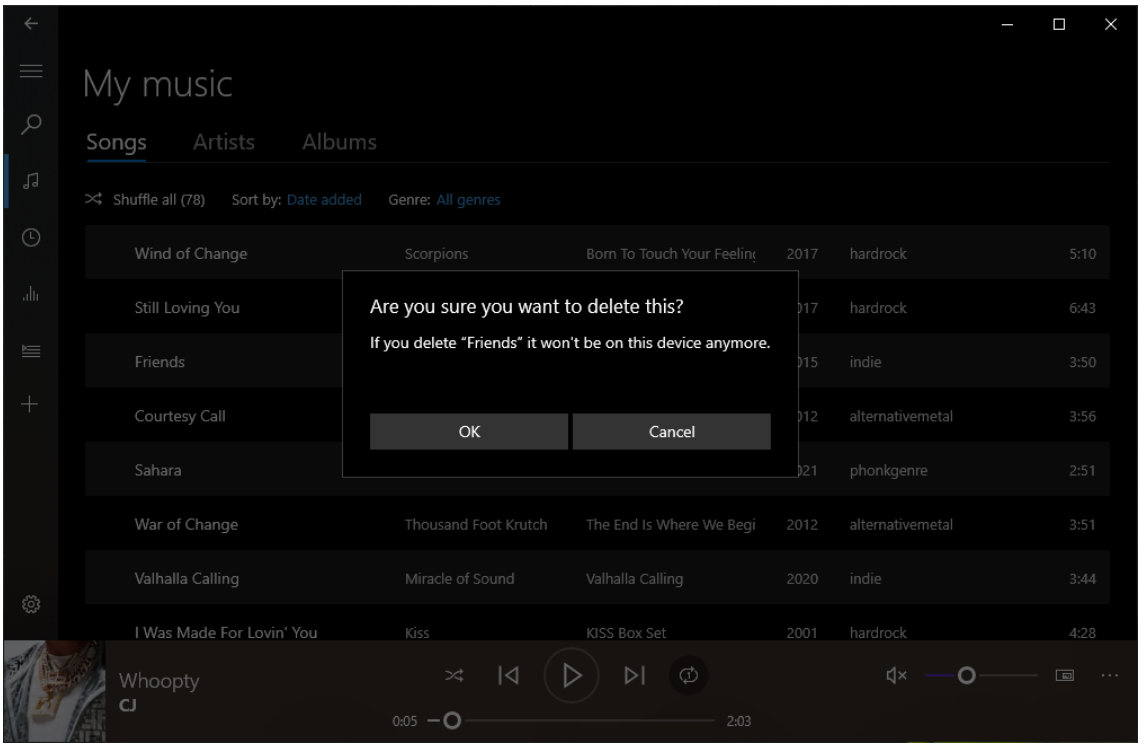

Figura 1.10 - Funcția de ștergere a unui fișierului audio in aplicația „Groove Music”

1.3 Scopul, obiectivele și cerințele sistemului

Sistemul de prelucrare si redare a fișierelor audio are ca scop principal ușurarea gestionarii fișierelor multimedia, in special a celor audio. Acesta oferă o soluție convenabila și ușor de utilizat pentru utilizatorii care doresc să asculte fișierele audio încărcate pe mașina locala cu ajutorul aplicației. Sistemul este construit pe o baza de date nerelaționata, care permite utilizatorilor sa gestioneze cu ușurință datele din fișierele încărcate. Sistemul oferă o varietate de funcții utile pentru a asigura o experiența confortabila pentru utilizator. În plus, sistemul este conceput sa ofere o mare flexibilitate în ceea ce privește gestionarea fișierelor audio. Utilizatorii pot adăuga, edita sau șterge fișiere din baza de date. În ansamblu, scopul acestui sistem este de a oferi utilizatorilor o soluție de gestionare a fișierelor audio care sa fie convenabila, ușor de utilizat și cu o varietate de funcții utile pentru a le oferi o experiența de ascultare confortabila și personalizata.

În urma studierii a mai multor sisteme asemănătoare și analiza neajunsurilor acestora, au fost făcute concluzii care au pus la baza ideea principală a scopului acestei aplicații, și anume de a face un music player ce va oferi utilizatorului o gama larga de funcții pentru gestionarea fișierelor audio.
Pentru a realiza acest scop a fost propus un set de obiective care ar trebui urmărite pentru dezvoltarea acestui proiect:

- Crearea sistemului de „Prelucrarea fișierelor audio”;
- Analiza soluțiilor existente de „Prelucrarea fișierelor audio”;
- Elaborarea caietului de sarcini;
- Argumentarea platformei și soft-ului de realizare;
- Cercetarea si selectarea instrumentelor pentru dezvoltarea platformei;
- Modelarea funcționala a platformei;
- Proiectarea structurii de baza a proiectului;
- Implementarea acțiunilor de citire si redare a fișierelor audio;
- Crearea componentei de vizualizarea a aplicației;
- Proiectarea unei bazei de date de tip NoSQL;
- Adăugarea funcționalului de schimbare a modului de redare;
- Implementarea operațiilor CRUD;
- Testarea sistemului;
- Documentarea sistemului informațional;
- Planificarea proiectului si estimarea costului;
- Concluzii și recomandări.

Cerințele hardware și soft pentru utilizarea aplicației sunt:

- Procesor – Intel Pentium sau AMD cu caracteristici asemănătoare;
- Sistem de operare – Windows / Linux;
- RAM – 2 GB.

2 MODELAREA ȘI PROIECTAREA SISTEMUL INFORMATIC

Proiectarea unui sistem software este un proces complex și necesită multe etape de planificare și dezvoltare. Pentru a putea crea un sistem software funcțional și eficient, este necesar să se folosească o abordare sistematică și unelte de modelare potrivite. În acest sens, limbajul UML (Unified Modeling Language) este una dintre cele mai utilizate tehnologii de modelare a sistemelor software.

UML este un limbaj grafic standardizat care permite proiectarea și documentarea sistemelor software prin intermediul diagramei. Aceasta oferă un set de elemente și reguli de modelare, care permit proiectantului să descrie sistemul din diferite perspective și să dezvolte o viziune comună asupra acestuia.

Odată ce diagramele comportamentale și structurale au fost realizate, se pot folosi acestea pentru a descrie detaliat sistemul. De exemplu, diagrama de clasă poate fi folosită pentru a identifica clasele principale ale sistemului, iar diagrama de secvență poate fi utilizată pentru a arăta modul în care aceste clase interacționează între ele. Aceste diagrame sunt utile nu doar pentru a descrie sistemul, ci și pentru a identifica și a rezolva potențiale probleme care ar putea apărea în timpul dezvoltării sau implementării sistemului. Prin intermediul unei astfel de abordări bazate pe modele, se poate asigura că codul este conform cu specificațiile sistemului și că nu există disjuncții între ceea ce este specificat în diagrame și ceea ce este implementat în cod. În plus, dezvoltatorii pot utiliza diagramele pentru a înțelege mai bine structura și comportamentul sistemului și pentru a face modificări rapide, dacă este necesar. Prin urmare, limbajul UML și diagramele sale sunt instrumente puternice de proiectare a sistemelor software, care permit proiectanților și dezvoltatorilor să creeze sisteme eficiente și ușor de înțeles. Prin intermediul diagramei, se pot descrie detaliat structura și comportamentul sistemului și se poate asigura o implementare corespunzătoare a acestuia. Cu ajutorul acestor instrumente, dezvoltatorii pot crea sisteme software complexe, într-un mod eficient și cu costuri reduse.

Pentru creare aplicației am ales sistemul de operare Windows, dar aplicația a fost creata si pentru sistemul de operare Linux, deoarece Linux nu are aplicație pentru redarea fișierelor audio ca și Windows (Groove Music). De multe ori, în timpul dezvoltării unei aplicații, se pune întrebarea despre compatibilitatea acesteia cu diferite platforme și sisteme de operare. Este important ca aplicația să fie disponibilă pentru cât mai multe platforme posibile, astfel încât utilizatorii să poată să o utilizeze indiferent de sistemul lor de operare preferat.

În cazul meu, am ales să dezvolt aplicația pentru sistemul de operare Windows, deoarece acesta este unul dintre cele mai utilizate sisteme de operare din lume. Windows oferă o varietate de aplicații de redare audio, iar una dintre cele mai populare este Groove Music. Însă, am decis să includ și suportul pentru sistemul de operare Linux, deoarece Linux nu are o aplicație de redare audio la fel de populară ca Groove Music pentru Windows.

În timpul dezvoltării aplicației pentru Linux, am întâmpinat unele dificultăți, deoarece sistemul de operare are unele diferențe semnificative față de Windows. Cu toate acestea, am reușit să găsim soluții și să dezvoltăm o versiune a aplicației care poate fi utilizată pe sistemele de operare Linux. Astfel, utilizatorii Linux pot utiliza aplicația pentru redarea fișierelor audio, chiar dacă sistemul lor de operare nu are o aplicație nativă pentru această funcție.

În general, dezvoltarea aplicațiilor pentru mai multe platforme poate fi un proces complex și costisitor, dar este important să oferi utilizatorilor opțiunea de a utiliza aplicația pe platforma lor preferată. Astfel, m-am asigurat că aplicația mea este disponibilă atât pentru utilizatorii Windows, cât și pentru cei Linux, oferind o experiență de utilizare optimă pentru toți utilizatorii noștri.

2.1 Descrierea comportamentală a sistemului

Sistemul Music Player este o aplicație desktop care permite utilizatorilor să reda și să gestioneze fișierele audio de pe computerul lor. Acesta este un software util pentru cei care își doresc să își organizeze colecția de muzică și să o asculte în mod eficient. Interfața grafică a sistemului Music Player este intuitivă și ușor de utilizat, astfel încât orice utilizator, indiferent de nivelul de cunoștințe în domeniul informatic, poate să își gestioneze fișierele audio într-un mod simplu și eficient.

Descrierea comportamentală a sistemului Music Player:

- Redarea fișierelor audio: Utilizatorii pot selecta fișierele audio pe care doresc să le redă prin intermediul interfeței grafice. Aceste fișiere sunt organizate într-o listă sau într-un arbore de fișiere, pe care utilizatorul le poate accesa și naviga. După selectarea fișierelor, utilizatorul poate da click pe butonul "play" pentru a începe redarea (figura 2.2 și figura 2.6);
- Controlul redării: Utilizatorul poate controla redarea fișierelor audio prin intermediul butoanelor "redare", "pauză", "următorul", " precedentul". Acest lucru îi permite să schimbe melodiile în timp ce ascultă muzică și să oprească sau să reia redarea când dorește (figura 2.3);
- Editarea cântecului: Utilizatorii pot edita titlu, artistul și poza de album al cântecului cu ajutorul unei interfețe grafice. Aceasta funcție este utilă pentru cei care doresc să își personalizeze colecția de muzică (figura 2.5);
- Crearea și gestionarea listelor de redare: Utilizatorii pot crea liste de redare pentru a organiza și gestiona mai bine colecția lor de muzică. Aceste liste de redare pot fi create manual sau prin intermediul unor funcții automate, cum ar fi modul de amestecare (shuffle) sau modul de repetare;
- Ștergerea cântecului: Utilizatorul poate șterge cântecul din lista generală unde sunt afișate toate fișierele audio încărcate de către utilizator. Această funcție este utilă pentru cei care doresc să elimine din colecția de muzică piese care nu mai sunt de interes pentru ei (figura 2.4).

Sistemul Music Player este o aplicație desktop eficientă și ușor de utilizat pentru redarea și gestionarea fișierelor audio. Aceasta oferă utilizatorilor o gamă largă de opțiuni și funcții care le permit să își personalizeze colecția de muzică și să o asculte în mod eficient.

2.1.1 Imaginea generală asupra sistemului

În general, use-case diagramele pot fi un instrument puternic pentru dezvoltatorii de software și pentru echipele de proiectare, ajutându-le să construiască sisteme software mai bune și mai eficiente, care să satisfacă cerințele utilizatorilor și să ofere o experiență de utilizare mai bună.

- Identificarea cazurilor de utilizare: Use-case diagramele permit identificarea tuturor cazurilor de utilizare posibile pentru un sistem. Acestea pot ajuta la definirea cu precizie a comportamentului sistemului și la asigurarea că acesta va satisface nevoile utilizatorilor;
- Gestionarea complexității: Proiectarea și dezvoltarea unui sistem software poate fi o sarcină complexă, cu multiple interacțiuni între diferite componente și module. Use-case diagramele pot ajuta la reducerea complexității prin descrierea interacțiunilor dintre diferitele cazuri de utilizare și componentele sistemului;
- Îmbunătățirea calității software-ului: Use-case diagramele pot fi utilizate pentru a identifica și preveni problemele înainte ca acestea să apară. O diagrame bine construită poate ajuta la evitarea erorilor de proiectare și la asigurarea calității software-ului prin documentarea cerințelor și a comportamentului sistemului;

În figura 2.1 este reprezentat procesele principale a aplicației.

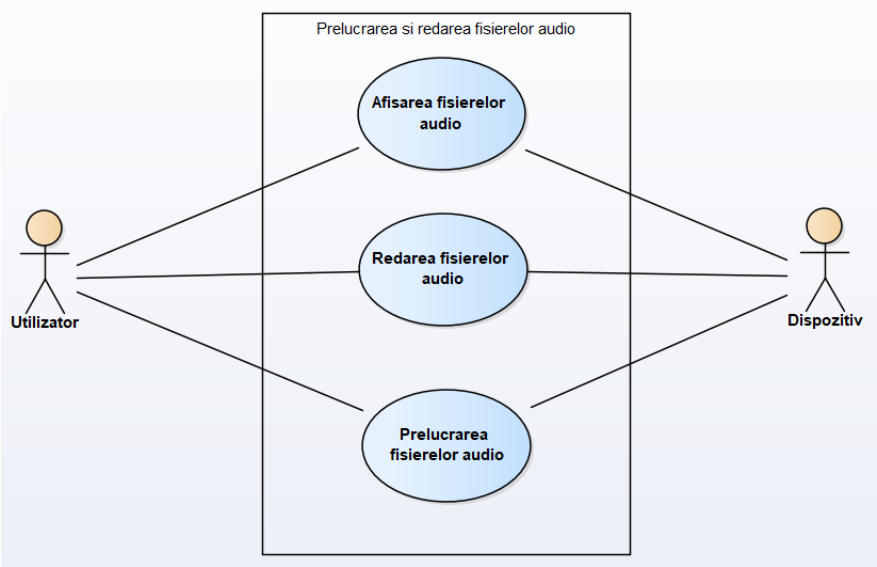

Figura 2.1 - Procesele principale ale utilizatorului

În figura 2.2 este reprezentat de încărcare a unui fișier audio cu ajutorul diagramei use-case.

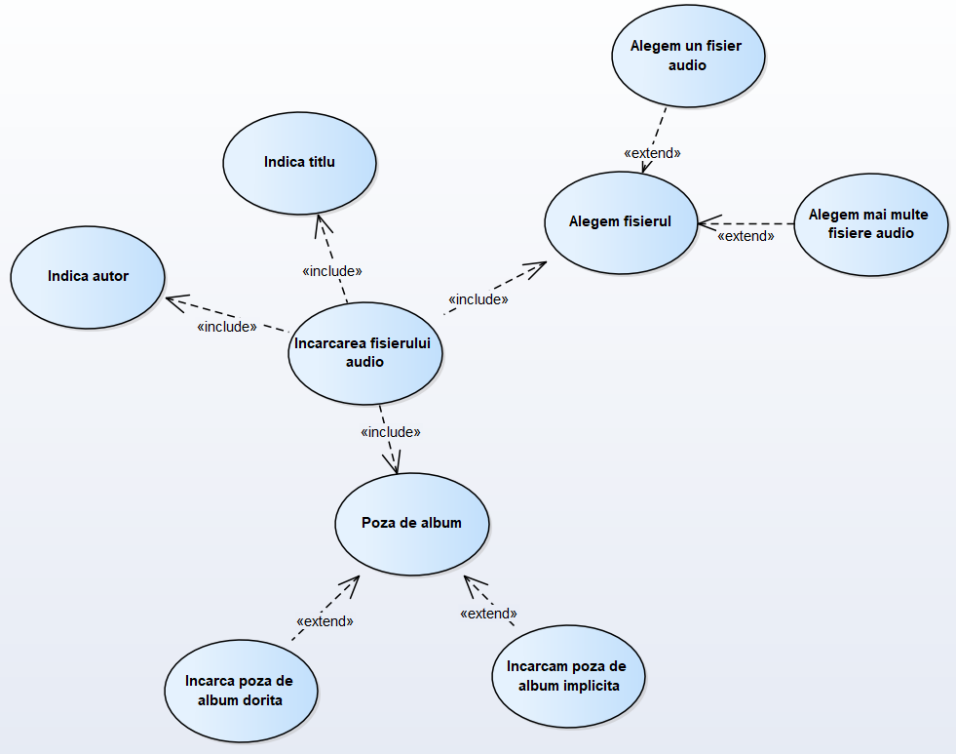

Figura 2.2 - Procesele de încărcare a fișierului audio

2.1.2 Modelarea vizuală a fluxurilor

Diagrama de activitate este utilă pentru a modela comportamentul proceselor sau activităților dintr-un sistem, evidențiind secvența logică a acțiunilor și deciziilor luate.

Această diagramă poate fi folosită pentru a:

- Înțelege și analiza procesele existente;
- Identifica punctele critice ale procesului;
- Îmbunătăți procesele prin optimizarea secvenței de acțiuni;
- Comunica procesele complexe într-un mod clar și simplu;
- Documenta procesele și deciziile luate în timpul acestora;
- Facilita identificarea erorilor și problemele procesului;
- Simplifica dezvoltarea software-ului prin identificarea și definirea clara a acțiunilor și fluxurilor de date necesare.

În figura 2.3 este reprezentat diagrama de activitate a procesului de redare a unui cântec.

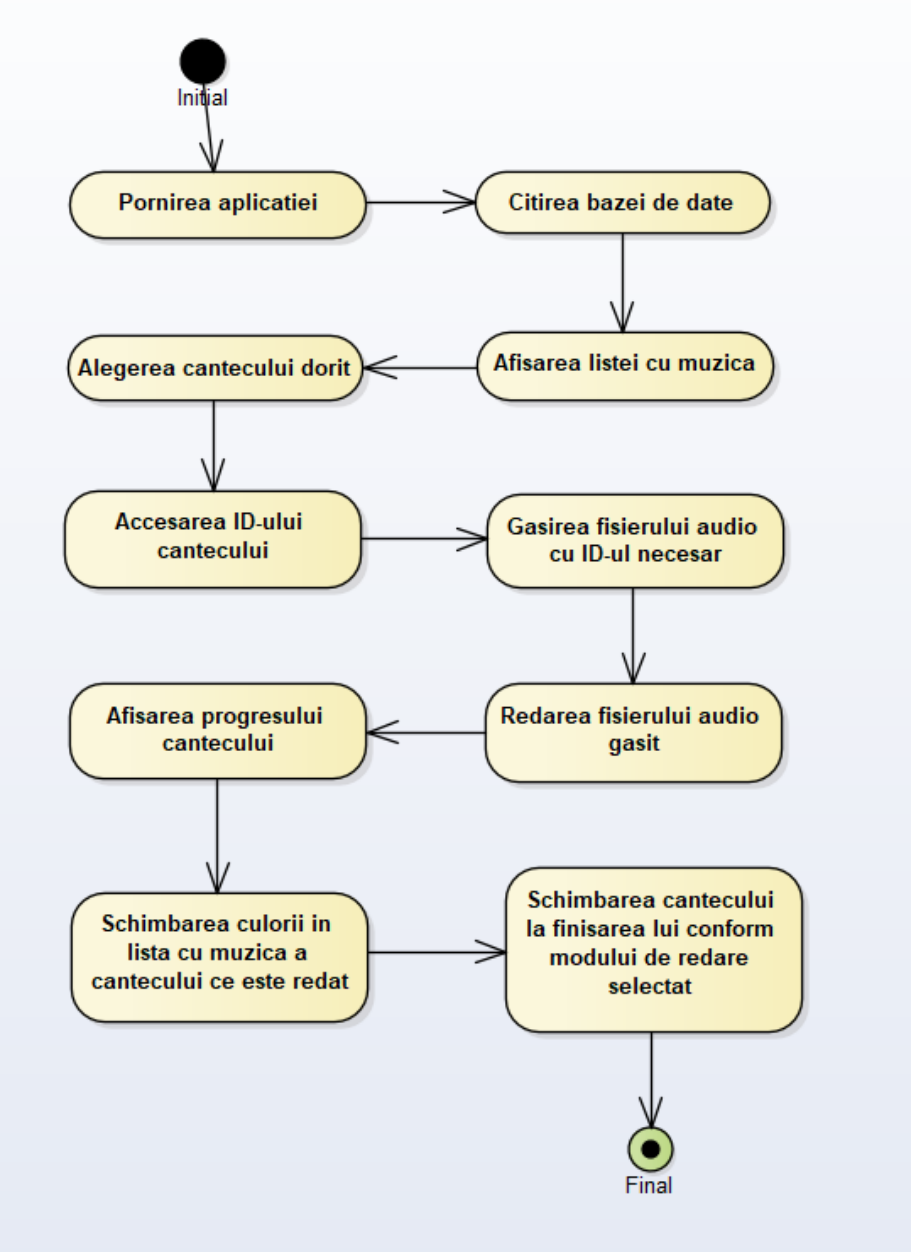

Figura 2.3 - Diagrama de activitate pentru procesul de redare a unui cântec

2.1.3 Stările de tranzacție a sistemului

Diagrama de stare este utilă pentru a modela comportamentul unui obiect sau a unui sistem, evidențiind stările și tranzițiile acestora. Această diagramă poate fi folosită pentru a:

- Înțelege comportamentul obiectelor și a sistemelor în diverse situații;
- Identifica stările critice și tranzițiile între acestea;
- Comunica modul în care un obiect sau un sistem răspunde la evenimente și stimuli externi;
- Documenta comportamentul obiectelor și a sistemelor;
- Simplifica dezvoltarea software-ului prin definirea clară a stărilor și a tranzițiilor între acestea.

În figura 2.4 este reprezentat procesul de ștergere a unui cântec cu ajutorul diagramei de stare.

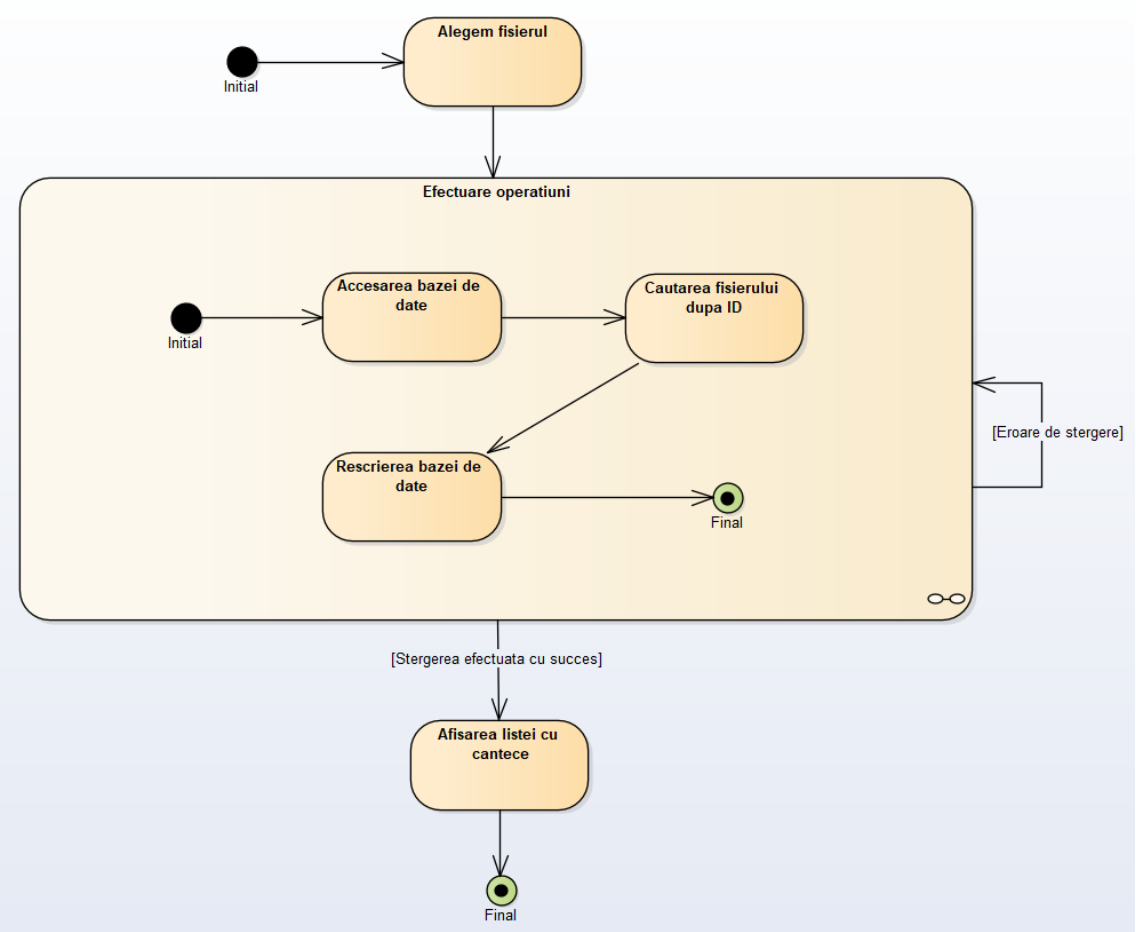

Figura 2.4 - Diagrama de stare a procesului de ștergere a unui cântec

2.1.4 Descrierea scenariilor de utilizare a aplicației

Diagrama de secvență este utilă pentru a modela interacțiunea dintre obiecte sau componente ale unui sistem, evidențiind secvența temporală a mesajelor și a acțiunilor acestora. Această diagramă poate fi folosită pentru a:

- Înțelege și analiza interacțiunile dintre componentele sistemului;
- Identifica ordinea și sincronizarea mesajelor și a acțiunilor între componente;
- Comunica interacțiunile complexe într-un mod clar și simplu;
- Documenta interacțiunile și acțiunile între componente;
- Facilita dezvoltarea software-ului prin definirea clară a interacțiunilor și a secvenței de mesaje și acțiuni între componente.

În figura 2.5 este reprezentat diagrama secvențiala a procesului de editare a unui cântec.

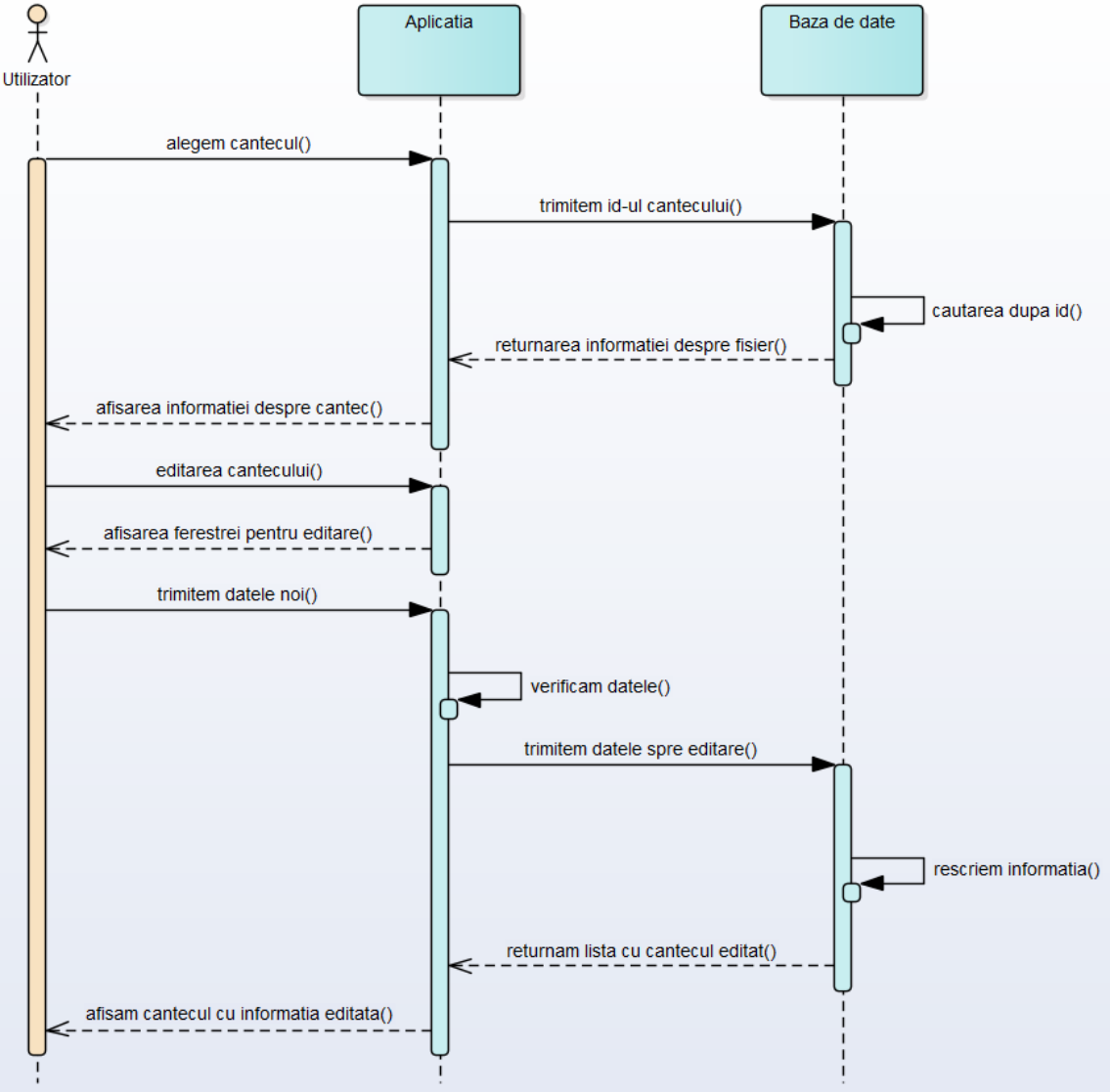

Figura 2.5 - Procesele de editare a unui cântec

2.1.5 Fluxurile de mesaje și legăturile dintre componentele sistemului

Diagrama de colaborare este utilă pentru a modela interacțiunea între obiecte sau componente ale unui sistem, evidențiind structura componentelor și relațiile dintre acestea. Această diagramă poate fi folosită pentru a:

- Înțelege și analiza structura și relațiile dintre componente;
- Identifica dependențele și colaborarea dintre componente;
- Comunica structura și relațiile complexe într-un mod clar și simplu;
- Documenta structura și relațiile dintre componente;
- Facilita dezvoltarea software-ului prin definirea clară a componentelor și a relațiilor dintre acestea.

În figura 2.6 este reprezentat procesele de pornire a aplicației cu ajutorul diagramei de colaborare.

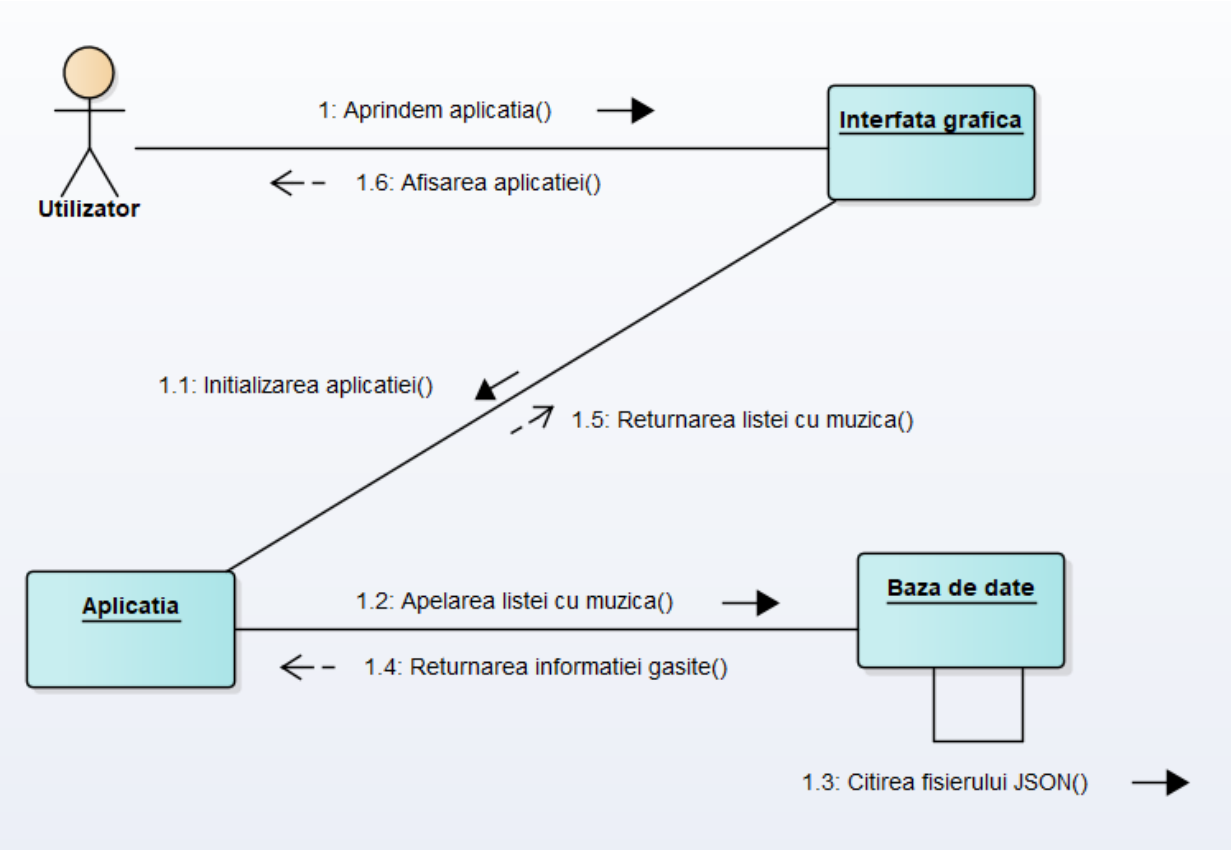

Figura 2.6 - Procesul de pornire a aplicației

2.2 Descrierea structurală a sistemului

Sistemul Music Player pentru Desktop este o aplicație complexă, alcătuită din mai multe componente interconectate, care lucrează împreună pentru a oferi o experiență plăcută și intuitivă de redare a muzicii pentru utilizatorii de desktop.

- Diagrama de clase este o unealtă importantă în dezvoltarea și menținerea acestui sistem, deoarece permite dezvoltatorilor să vizualizeze structura sistemului și modul în care clasele sale interacționează între ele. Această diagramă poate oferi o imagine de ansamblu a arhitecturii sistemului, evidențiind relațiile între obiectele și modulele sale componente și ajutând la identificarea potențialelor probleme de proiectare sau a erorilor de cod (figura 2.7).
- Diagrama de componente este o altă unealtă importantă pentru a înțelege modul în care aplicația gestionează funcționalitatea la interacțiunea cu utilizatorul prin intermediul interfeței grafice. Această diagramă poate oferi o privire detaliată asupra modului în care componentele interacționează între ele pentru a oferi diferite funcționalități utilizatorului. De exemplu, poate fi utilizată pentru a arăta modul în care aplicația manipulează fișierele audio sau modul în care se conectează la o bază de date pentru a gestiona informațiile despre muzică (figura 2.8).

- Diagrama de deployment este o altă unealtă importantă pentru a înțelege modul în care sistemul este reprezentat în sistemul de operare și cum este păstrată aplicația. Această diagramă poate oferi o imagine de ansamblu a arhitecturii hardware și software, evidențiind relațiile între diferitele componente ale sistemului și ajutând la identificarea problemelor de performanță sau a erorilor de configurare (figura 2.9).

În plus, diagramele structurale sunt utile și pentru a asigura o dezvoltare eficientă și o mentenanță facilă a sistemului. Prin utilizarea diagramei de clase, putem identifica ușor clasele care necesită modificări și putem înțelege cum modificările făcute la o clasă pot afecta alte clase din sistem. Aceasta poate reduce timpul și efortul necesar pentru dezvoltare și poate crește calitatea codului. De asemenea, diagrama de componente este utilă pentru împărțirea aplicației în componente mai mici și mai ușor de gestionat, ceea ce poate facilita dezvoltarea în echipe și mentenanța aplicației pe termen lung. Prin identificarea clară a interacțiunilor între componente, putem înțelege mai bine cum să îmbunătățim performanța și scalabilitatea sistemului. Diagrama de deployment poate fi utilă în implementarea și configurarea sistemului în medii diferite, cum ar fi dezvoltarea și testarea într-un mediu de dezvoltare, apoi implementarea într-un mediu de producție. Înțelegerea cum sistemul este reprezentat în mediul de producție poate ajuta la evitarea problemelor legate de incompatibilitatea hardware sau software și la asigurarea faptului că sistemul rulează eficient. Prin urmare, diagramele structurale sunt un instrument esențial în dezvoltarea sistemului Music Player, oferind o înțelegere clară a arhitecturii sistemului și a modului în care componentele sale interacționează între ele.

2.2.1 Descrierea structurii statice a sistemului

Diagrama de clasă este utilă pentru a modela structura statică a unui sistem, evidențiind clasele, atributele, metodele și relațiile dintre acestea. Această diagramă poate fi folosită pentru a:

- Înțelege și analiza structura componentelor și a relațiilor dintre acestea;
- Identifica proprietățile și comportamentul fiecărei clase;
- Comunica structura complexă a sistemului într-un mod clar și simplu;
- Documenta structura componentelor și relațiile dintre acestea;
- Facilita dezvoltarea software-ului prin definirea clară a claselor, a atributelor și a metodelor acestora, precum și a relațiilor dintre clase.

În figura 2.7 este reprezentat diagrama de clasa a aplicației.

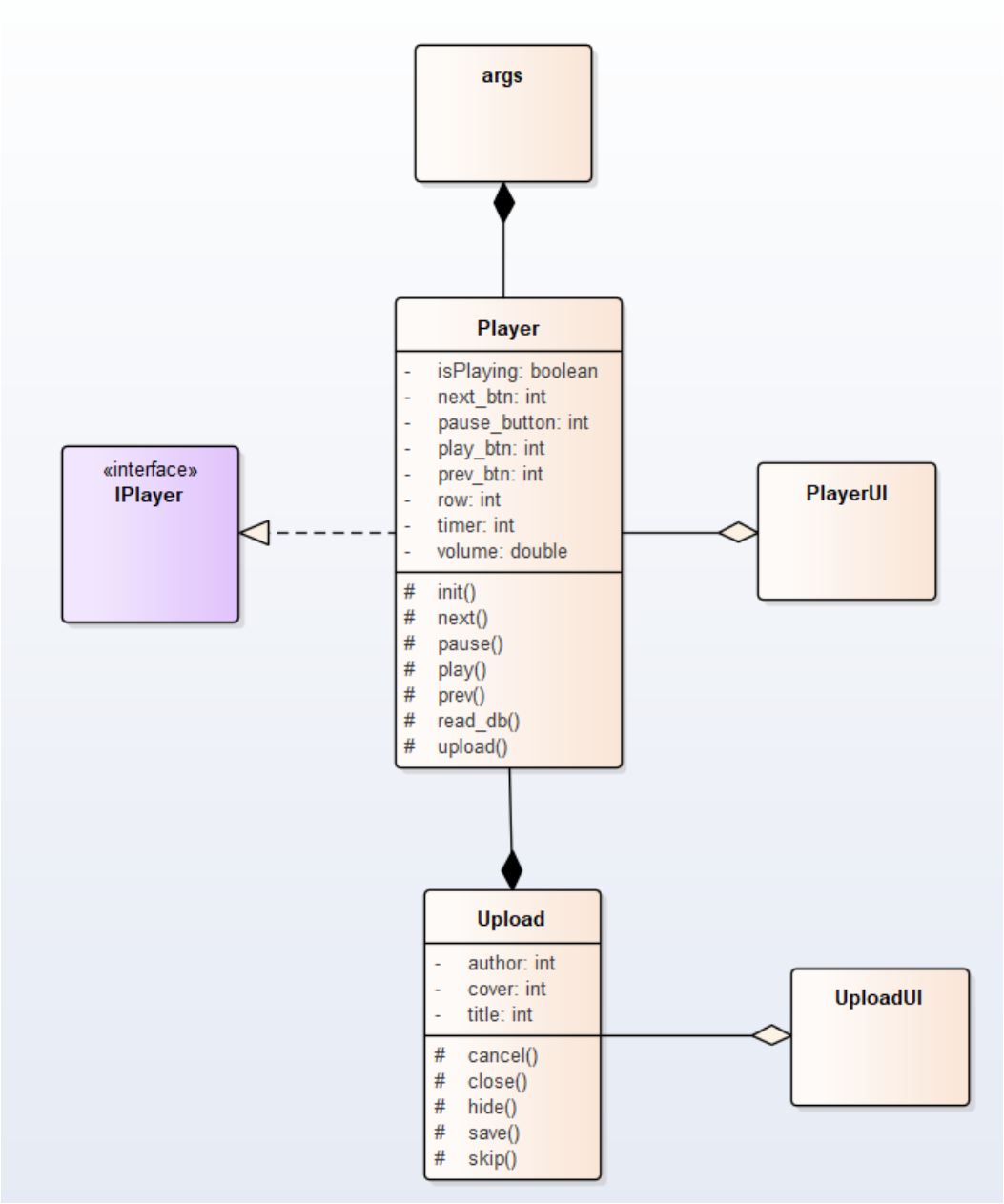

Figura 2.7 - Diagrama de clasa a aplicației

2.2.2 Relațiile de dependență între componentele sistemului

Diagrama de componente este utilă pentru a modela componentele și dependențele acestora într-un sistem software, evidențiind relațiile dintre acestea și mediul de rulare. Această diagramă poate fi folosită pentru a:

- Înțelege și analiza componentele și dependențele dintre acestea într-un sistem software
- Identifica componentele critice și dependențele acestora
- Comunica structura și relațiile complexe între componente într-un mod clar și simplu
- Documenta structura și relațiile dintre componente

- Facilita dezvoltarea software-ului prin definirea clară a componentelor și a dependențelor dintre acestea. De asemenea, poate fi utilă în procesul de testare și implementare a sistemului.

În figura 2.8 este reprezentat componentele principale a aplicației.

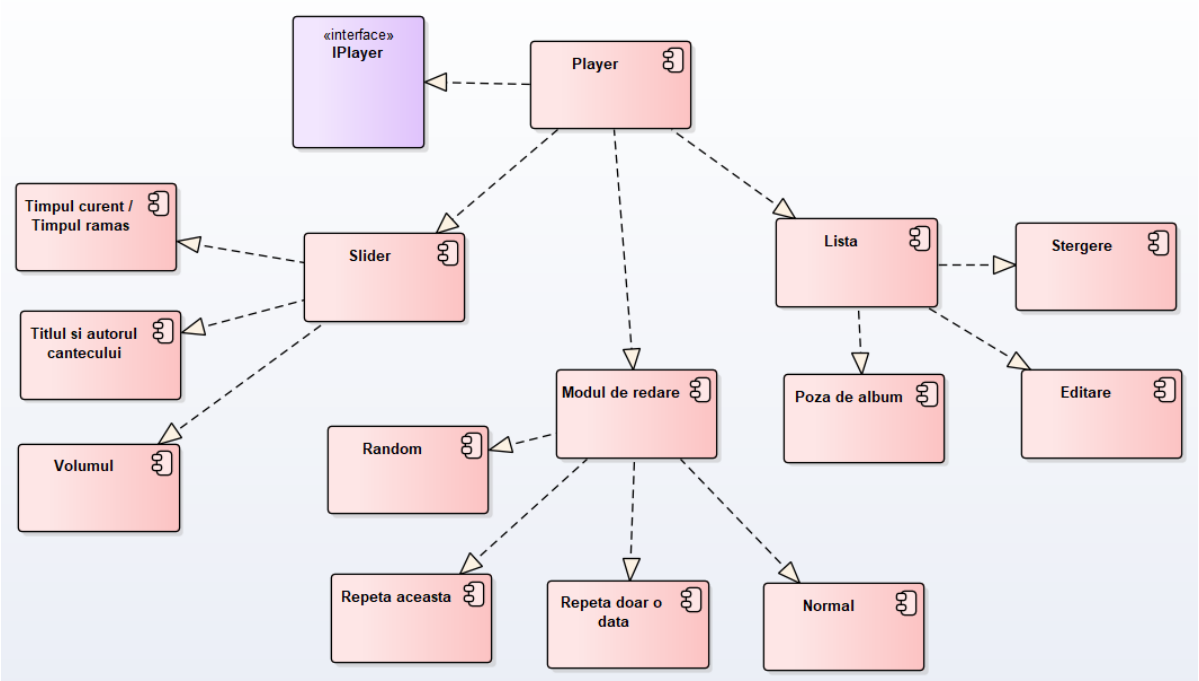

Figura 2.8 – Componentele aplicației

2.2.3 Modelarea echipamentelor mediului de implementare

Diagrama de deployment este utilă pentru a modela arhitectura fizică a unui sistem software, evidențiind distribuția componentelor pe noduri de calcul și rețelele de comunicare dintre acestea. Această diagramă poate fi folosită pentru a:

- Înțelege și analiza modul în care componentele sistemului sunt distribuite și conectate între ele în mediul de rulare
- Identifica resursele hardware necesare pentru implementarea și rularea sistemului
- Comunica arhitectura fizică a sistemului într-un mod clar și simplu
- Documenta arhitectura fizică a sistemului
- Facilita implementarea, testarea și administrarea sistemului prin definirea clară a resurselor hardware și a rețelelor de comunicare între nodurile de calcul.

În figura 2.9 este reprezentat sistemul de operare unde este păstrată aplicația.

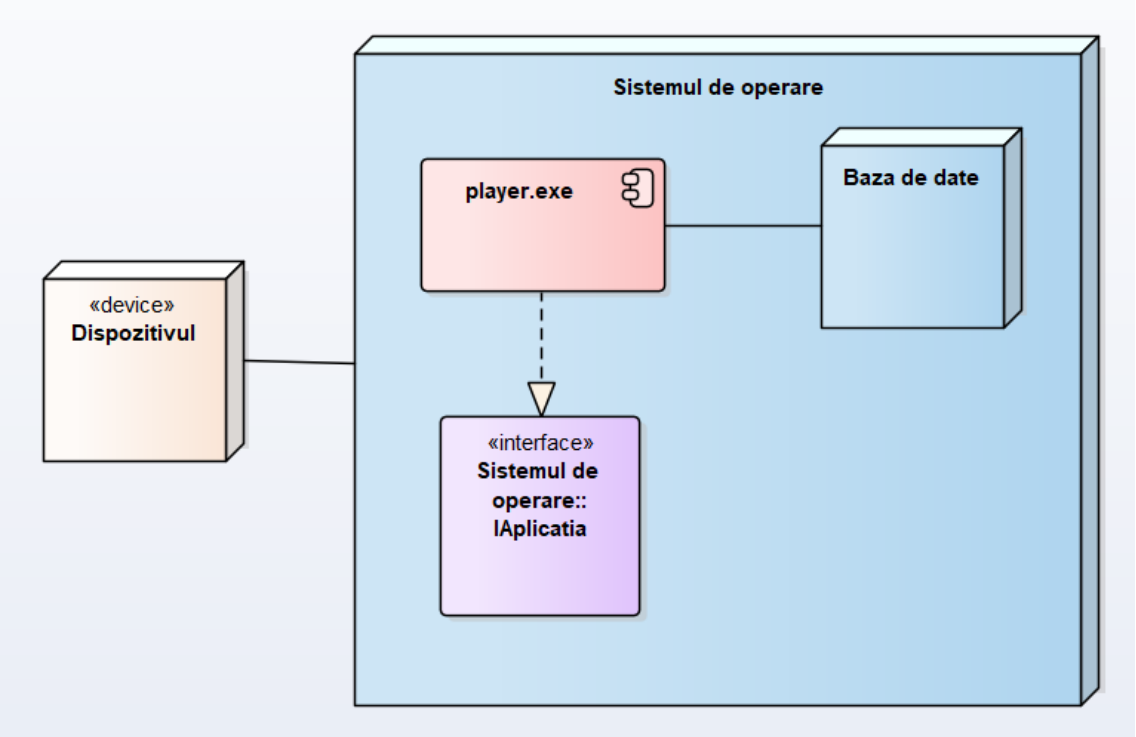

Figura 2.9 - Reprezentarea sistemului de operare unde este păstrată aplicația

3 REALIZAREA SISTEMULUI

Python este un limbaj de programare la nivel înalt, interpretat și dinamic, care este utilizat pe scară largă pentru o gamă largă de scopuri, inclusiv dezvoltarea web, calculul științific, analiza datelor, inteligența artificială și multe altele. Python este cunoscut pentru simplitatea și lizibilitatea sa, ceea ce îl face o alegere populară atât pentru începători, cât și pentru dezvoltatorii experimentați.

PyQt este un set de legături Python pentru cadrul aplicației Qt și rulează pe toate platformele acceptate de Qt, inclusiv Windows, OS X, Linux, iOS și Android. PyQt este o alegere populară pentru dezvoltarea aplicațiilor desktop datorită ușurinței sale de utilizare și comunității mari de utilizatori și dezvoltatori. Oferă un set bogat de componente ale interfeței grafice cu utilizatorul (GUI) și acceptă dezvoltarea rapidă a aplicațiilor (RAD).

JSON (JavaScript Object Notation) este un format ușor de schimb de date care este ușor de citit și scris de oameni și ușor de analizat și generat de mașini. Este adesea folosit ca bază de date pentru aplicații de dimensiuni mici și mijlocii, în special în dezvoltarea web, datorită simplității și ușurinței sale de utilizare. JSON este, de asemenea, folosit ca o modalitate de schimb de date între un client și un server, ceea ce îl face o alegere populară pentru API-uri. În concluzie, Python este un limbaj de programare versatil și popular, care este utilizat pe scară largă pentru o varietate de scopuri. PyQt oferă un set puternic de instrumente pentru dezvoltarea aplicațiilor desktop cu un set bogat de componente GUI. JSON este o alegere populară pentru bazele de date mici și mijlocii, în special în dezvoltarea web, datorită simplității și ușurinței sale de utilizare.

De asemenea pentru unirea acestor toate tehnologii într-un sistem, am folosit PyPi, un instrument ce ajută la descarcarea, updatarea bibliotecilor pentru limbajul de programare Python. PyPI înseamnă Python Package Index, care este depozitul oficial pentru pachetele Python open-source. Este întreținut de comunitatea Python și permite dezvoltatorilor să caute, să descarce și să instaleze cu ușurință pachete Python care pot fi folosite în proiectele lor. PyPI are o colecție vastă de pachete Python, de la biblioteci pentru analiza datelor și învățarea automată până la cadre web și seturi de instrumente GUI. Utilizatorii pot căuta pachete folosind cuvinte cheie sau răsfoind prin categorii. Fiecare pachet are propria sa pagină care include informații precum numele pachetului, versiunea, descrierea, dependențele și documentația. Pentru a utiliza un pachet de la PyPI, dezvoltatorii pot utiliza pur și simplu un manager de pachete precum pip pentru a-l instala. Pip descarcă și instalează automat pachetul și orice dependențe necesare. PyPI a făcut mai ușor pentru dezvoltatori să-și partajeze codul și să colaboreze la proiecte, ceea ce a contribuit la popularitatea limbajului de programare Python.

3.1 Descrierea la nivel de cod pe module

In figura 3.1 este reprezentat structura sistemului.

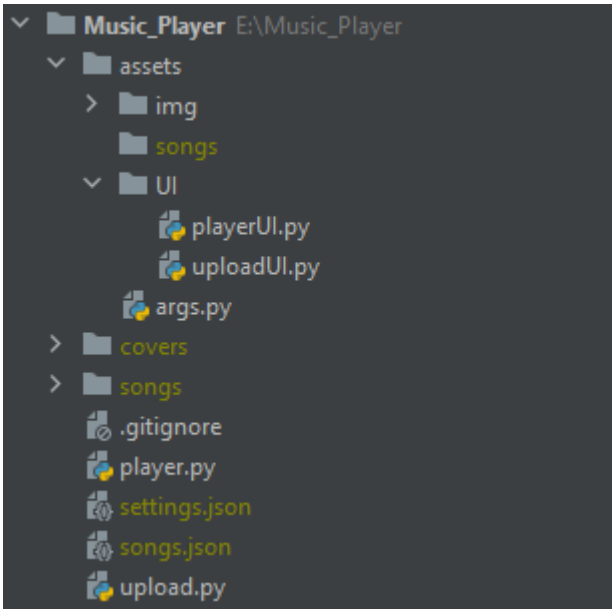

Figura 3.1 - Structura aplicației

În codul de mai jos este reprezentat citirea fișierului de tip JSON unde este stocată informația despre fișierele audio.

```
def read_songs_from_json(self):
      if not os.path.exists('songs'):
          os.makedirs('songs')
      self.player = QMediaPlayer()
      self.playlist = QMediaPlaylist(self.player)
      try:
          with open("songs.json", "r", encoding="utf-8") as file:
              data = json.load(file)
          self.titles.clear()
          self.artists.clear()
          self.covers.clear()
          for i in data["Songs"]:
              self.titles.append(i["title"])
              self.artists.append(i["artist"])
              if i["cover"] == "Undefined":
                  self.covers.append("no_image.jpg")
              else:
                  self.covers.append(i["cover"])
      except Exception as e:
          print(e)
```

```
        self.read_files_songs()
```

În codul de mai jos este reprezentat citirea și rescrierea fișierului „settings” ce include setările player-ului făcute de către utilizator.

```
    def settings_read(self):
        try:
            with open("settings.json", "r", encoding="utf-8") as f:
                data = json.load(f)
            for i in data["Settings"]:
                self.volume = i["Volume"]
                self.lastVolume = self.volume
                self.row = i["Row"]
                self.mode = i["Mode"]
            self.currentIndex = self.row
        except Exception as e:
            print(e)

    def settings_write(self):
        settings_list = {}
        settings_list["Settings"] = []
        settings_list["Settings"].append({
            "Volume": self.volume,
            "Row": self.row,
            "Mode": self.mode
        })
        with open("settings.json", "w", encoding="utf-8") as f:
            json.dump(settings_list, f, indent=4)
```

În secvența de cod de mai jos este reprezentat funcționalul butoanelor „Redare”, „Pauză”, „Următorul” și „Anteriorul”.

```
    def play(self):
        if len(self.titles) > 0:
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.newIndex = self.player.playlist().currentIndex()
                self.checkStyle()
            else:
                self.player.pause()
                self.isPlaying = False
                self.checkStyle()
```

```
    def next(self):
        if len(self.titles) > 0:
            self.playlist.next()
            self.newIndex = self.player.playlist().currentIndex()
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.ui.playButton.setStyleSheet(pause_btn_css)

    def prev(self):
        if len(self.titles) > 0:
            if int(self.now_sec) < 10:
                self.playlist.previous()
                self.newIndex = self.player.playlist().currentIndex()
            else:
                self.player.setPosition(0)
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.ui.playButton.setStyleSheet(pause_btn_css)
```

Mai jos este reprezentat timer-ul aplicației. Această metodă este ca un „loop”, adică este chemată de mai multe ori pe secundă, dar mai concret 60 de ori, deoarece ochiul uman poate să observe maxim 60 de cadre pe secundă, prin urmare informația afișată cu ajutorul interfeței grafice va fi actualizată după acest criteriu.

```
    def time_hit(self):
        self.checkStyle()
        self.checkstyleVolume()
        if self.isPlaying:
            self.ui.musicSlider.setMaximum(self.player.duration())
            if not self.ui.musicSlider.isSliderDown():
                self.ui.musicSlider.setValue(self.player.position())
            self.newIndex = self.player.playlist().currentIndex()
            self.checkList()

            song_min, song_sec = self.convertMillis(int(self.player.duration()))
            if song_sec < 10:
                self.song_duration = "{0}:0{1}".format(int(song_min), int(song_sec))
            else:
                self.song_duration = "{0}:{1}".format(int(song_min), int(song_sec))
```

```
            now_min, self.now_sec = self.convertMillis(int(self.ui.musicSlider.value()))
            if self.now_sec < 10:
                self.now_duration = "{0}:0{1}".format(int(now_min), int(self.now_sec))
            else:
                self.now_duration = "{0}:{1}".format(int(now_min), int(self.now_sec))

            self.ui.durationLabel.setText(str(self.now_duration) + " / " +
str(self.song_duration))

            if self.repeatonce:
                if self.now_duration == self.song_duration:
                    self.isPlaying = False
                    self.ui.playButton.setStyleSheet(play_btn_css)
                    self.player.stop()
        self.settings_write()
```

De asemenea am rescris acțiunea unei aplicații obișnuite. În codul de mai jos putem observa că am creat bara de titlu proprie a aplicației, am rescris event-ul de închidere a aplicației și event-ul de minimizare a ferestrei aplicației. Prin urmare acțiunea de închidere va face ca aplicația să se mute pe fundal, adică nu va fi afișată fereastra aplicației, iar procesul redării fișierelor audio va continua mai departe.

```
    def mousePressEvent(self, event):
        self.start = self.mapToGlobal(event.pos())
        self.pressing = True

    def mouseMoveEvent(self, event):
        if self.pressing and (
                self.ui.titleBarLabel.underMouse() or self.ui.titleBarInfoLabel.underMouse() or
self.ui.titleBarTitle.underMouse()):
            self.end = self.mapToGlobal(event.pos())
            self.movement = self.end - self.start
            self.setGeometry(self.mapToGlobal(self.movement).x(),
                             self.mapToGlobal(self.movement).y(),
                             self.width(),
                             self.height())
            self.start = self.end

    def minimizeButton_clicked(self):
        self.showMinimized()

    def closeButton_clicked(self):
        self.hide()
```

```
    def closeEvent(self, event):
        event.ignore()
        self.hide()
```

3.2 Testarea sistemului

Testarea manuală de regulă se realizează în câțiva pași:

- Instalare: Descărcăm aplicația și o instalăm pe mașina locală. Verificăm dacă procesul de instalare este fără probleme și dacă nu există erori. Ne asigurăm că aplicația este instalată în locația dorită și sunt create comenzi rapide;

- Testare funcțională: Odată instalată, lansăm aplicația și începem să testăm funcționalitățile acesteia. Încercăm să redăm fișierele muzicale și verificăm dacă sunt redate corect. Verificăm dacă putem întrerupe, reda sau să putem pune la pauză muzica. Încercăm să trecem la diferite părți ale muzicii pentru a ne asigura că funcționează corect. Testăm diferite formate de fișiere media pentru a ne asigura că aplicația poate primi doar fișiere cu extensia MP3;

- Testarea interfeței cu utilizatorul: verificăm aspectul general al aplicației. Ne asigurăm că toate butoanele, fișierele, meniurile și link-urile funcționează conform așteptărilor. De asemenea ne asigurăm că textul și imaginile sunt clare și vizibile;

- Gestionarea erorilor: Încercăm să introducem intenționat o intrare nevalidă sau să efectuați o acțiune care nu ar trebui să fie posibilă și verificăm cum gestionează aplicația aceste erori, de exemplu la anularea încărcării unei poze de album, în acest caz poza trebuie să rămână aceiași care a fost înainte de apăsarea butonului de alegere a altei poze de album. Ne asigurăm că aplicația nu se blochează;

- Testare de compatibilitate: testăm aplicația pe diferite sisteme de operare (SO Windows și Linux) și configurăm hardware pentru să ne asigurăm că funcționează conform așteptărilor;

- Testarea performanței: verificăm dacă aplicația funcționează bine și nu consumă prea multe resurse. Încercăm să rulăm aplicația pentru o perioadă lungă de timp pentru a vedea dacă se blochează sau încetinește;

- Testare de securitate: aici nu este ce să testăm, deoarece aplicația nu preia informație personală de la utilizator și nu are ieșire la internet, ea este rulată doar pe mașina locală;

- Testare de regresie: Efectuăm testarea de regresie pentru a ne asigura că noile modificări sau funcții adăugate la aplicație nu încalcă funcționalitatea existentă. Adică când adăugăm funcționalități noi, aplicația trebuie să meargă cu succes împreună cu aceste funcționalități, prin urmare testul a fost efectuat cu succes.

Pe parcurs, în urma testării manuale a sistemului dat au fost depistate o mulțime de bug-uri minori și câteva bug-uri majori care au fost soluționate și fixate, prin urmare testarea manuală a ajutat foarte mult pentru crearea unui sistem sigur și funcțional.

CONCLUZII

În concluzie, Music Player este o aplicație multimedia puternică și ușor de utilizat, concepută pentru iubitorii de muzică care doresc să gestioneze și să reda fișierele lor audio cu ușurință. Gestionarea automată a fișierelor și designul modern îl fac să iasă în evidență față de alte aplicații similare, în timp ce caracteristicile sale avansate și flexibilitatea se adresează utilizatorilor cu nevoi și preferințe diferite. În plus, accesibilitatea și performanța optimizată a aplicației o fac o alegere excelentă atât pentru ascultătorii obișnuiți, cât și pentru muzicienii profesioniști. În general, Music Player este o unealtă excelentă care merită cu siguranță să fie încercată.

De asemenea, cum am menționat în capitolul doi, UML (Unified Modeling Language) este un limbaj grafic de modelare care este frecvent utilizat în ingineria software pentru a vizualiza și proiecta sisteme complexe. Atunci când se creează un proiect, cum ar fi un player de muzică, UML poate fi un instrument util pentru dezvoltatori pentru a gestiona arhitectura, funcționalitatea și interacțiunile sistemului. Pentru a crea un astfel de player de muzică utilizând UML, am început prin crearea unei diagrame de use-case pentru a identifica actorii sistemului și a crea imaginea generală a sistemului. De asemenea am creat un use-case pentru afișsarea funcționalității de încărcare a unui fițier audio pe care sistemul le va efectua la apăsarea butonului de către utilizator. Le fel am creat o diagramă de clasă pentru a modela obiectele din sistem și relațiile lor (de exemplu, clasa Player și clasa Upload). Tot așa am creat o diagramă de secvență pentru a ilustra interacțiunile dintre obiecte în timpul executării anumitor cazuri de utilizare (de exemplu editarea unui cântec), iar pe lângă aceste diagrame mai sunt și altele care oferă o imagine a structurii sistemul dat. În general, UML poate fi un instrument puternic pentru crearea unui proiect de player de muzică bine organizat și bine proiectat. Prin utilizarea UML pentru a planifica arhitectura și funcționalitatea proiectului, ne-am asigurat că produsul final este eficient și ușor de întreținut.

În capitolul trei am analizat cele mai importante instrumente care au fost utilizate pentru crearea acestui sistem. Am aflat că Python este un limbaj de programare versatil și popular, utilizat pentru o gamă largă de scopuri. PyQt5 oferă un set puternic de instrumente pentru dezvoltarea aplicațiilor desktop cu un set bogat de componente GUI. JSON este o alegere populară pentru bazele de date mici și mijlocii, în special în dezvoltarea web sau a aplicațiilor mici cum ar fi Music Player, datorită simplității și ușurinței sale de utilizare. PyPI, indexul de pachete Python, este un instrument esențial pentru descărcarea și actualizarea bibliotecilor și pachetelor Python, ceea ce face mai ușor partajarea codului și menținerea lui. Procesul de testare manuală a ajutat la identificarea și remedierea bug-urilor minore și majore, asigurând că sistemul este sigur și funcțional. În general, aceste tehnologii și instrumente contribuie la popularitatea și succesul limbajului de programare Python, care la rândul lui a oferit o gamă largă de biblioteci care au contribuit la crearea eficientă a sistemului dat.

BIBLIOGRAFIE

1. [Resursa electronica] – Regim de acces: https://skobelevserg.jimdofree.com/
2. Adriana Bogdan - ISTORIA PRIN APLICAŢII MULTIMEDIA; [Resursa electronica] – Regim de acces: http://www.historica-cluj.ro/anuare/AnuarHistorica2014/24.pdf
3. [Resursa electronica] – Regim de acces: https://www.tutorialspoint.com/
4. [Resursa electronica] – Regim de acces: https://v2cloud.com/glossary/what-is-a-desktop-app

ANEXA A

Denumire anexă

MINISTERUL EDUCAŢIEI ŞI CERCETĂRII AL REPUBLICII MOLDOVA

UNIVERSITATEA TEHNICĂ A MOLDOVEI

CAIETUL

STAGIULUI DE PRACTICĂ

PENTRU STUDENȚII STUDIILOR SUPERIOARE DE LICENȚĂ – CICLULUI I

Stagiul de practică *de licenţă*
tipul stagiului de practică

A studentului (ei) *Zavorot Daniel*
numele, prenumele

Facultatea *Calculatoare, Informatică și Microelectronică*

Programul de studii *Tehnologia informaţiei*

Ciclul *I* Anul de studii *IV* Grupa *TI-194*

Locul stagiului de practică Universitatea Tehnica a Moldovei
denumirea unităţii economice

Conducătorul stagiului de practică
de la UTM *asist.univ. Cojocaru Svetlana*
funcţia, numele, prenumele

Conducătorul stagiului de practică
de la unitatea economică
funcţia, numele, prenumele

Chişinău, 2023

I. CAIETUL DE SARCINI

Nr. crt.	Tematica lucrărilor preconizate	Termene planificate	
		început/ sfârşit	**numărul de zile**
1	*Analiza domeniului de studii și Modelarea și proiectarea sistemul informatic*	*30.01.2023-03.02.2023*	*5 zile*
2	*Realizarea sistemului*	*06.02.2023-10.02.2023*	*5 zile*
3	*Realizarea sistemului*	*20.02.2023-24.02.2023*	*5 zile*
4	*Realizarea raportului și prezentării*	*27.02.2023-03.03.2023*	*5 zile*

II. SARCINA INDIVIDUALĂ
(SE INDICĂ TEMA PROIECTULUI DE LICENȚĂ)

Tema: ___Prelucrarea și redarea fișierelor audio_____

Nr. crt.	Conținutul concis al sarcinilor individuale	Mențiune cu privire la realizare
1	*Analiza domeniului de studii*	• *Se descrie domeniul din care face parte proiectul creat (tehnologii informaționale). Ce tip de aplicație (web, mobile...etc). În ce domeniu se va aplica proiectul (învățământ, sfera serviciilor.....etc).* • *Se va argumenta importanța creării unui astfel de proiect.* • *Se va face o descriere a cel puțin 3 sisteme deja existente (sau sisteme care au ceva comun din punct de vedere funcțional cu ce se va face în proiect). Se va face o comparare a sistemelor descrise.* • *În baza comparării se va scri scopul și obiectivele, cerințele sistemului. Determinarea cerințelor funcționale, nefuncționale. Detrminarea funcționalului sistemului și subsistemelor. Se va descrie cît mai detaliat caietul de sarcini a proiectului.*
2	*Modelarea și proiectarea sistemul informatic*	• *Descrierea comportamentală a sistemului (imaginea generală asupra sistemului; modelarea vizuală a fluxurilor; stările de tranzacție a sistemului (Statechart Diagram); descrierea scenariilor de utilizare a aplicaţiei (Sequence Diagram); fluxurile de mesaje şi legăturile dintre componentele sistemului (Collaboration Diagram).* • *Descrierea structurală a sistemului (descrierea structurii statice a sistemului; relatiile de dependență între componentele; modelarea echipamentelor mediului de implementare.*
3	*Realizarea sistemului*	• *Se face o introducere în ce s-a folosit pentru realizarea sarcinilor. Se descrie pe scurt tehnologiile folosite, limbaje de programare utilizate, instrumente, algoritmi, șabloane...etc , totul ce s-a folosit la realizarea proiectului.* • *Se descrie la nivel de cod realizarea funcționalităților* • *Se descrie testarea sistemului*
4	*Realizarea raportului*	• *De structurat după sarcini individuale* • *De redactat confor standardelor de redactare*
5	*Realizarea prezentării*	• *Slide cu denumirea temei* • *Slide cu Analiza domeniului* • *Slide cu Scopul și Obiectivele proiectului* • *Slide cu Imaginea generală a sistemului* • *Slide cu Descrierea structurii statice a sistemului* • *Slide cu Instrumente, tehnologii utilizate pentru realizarea sustemului* • *Slide cu Video- prezentarea aplicației*

Conducătorul _____ / *Cojocaru Svetlana* /
semnătura *numele şi prenumele*

FIŞA DE ACTIVITATE

Săptâmâna I

Sarcini planificate:

1. Descrierea aplicației, din ce domeniul face parte, tipul aplicației.
2. Argumentarea de ce este necesar crearea unui astfel de proiect.
3. Studierea a mai multor sisteme asemănătoare. Compararea sistemelor si alegerea acelor mai bune funcționalități.
4. Descrierea scopului si obiectivelor care vor fi realizate.

1. Descrierea aplicației, din ce domeniul face parte, tipul aplicației.

Aplicația create face parte din domeniul tehnologii informaționale, tipul aplicației fiind multimedia pentru desktop (aplicației GUI pentru manipularea file-urilor cu extensia .mp3, de asemenea lucrarea cu file-urile .png, .jpg, .json si .ico). Proiectul meu se va aplica in sfera serviciilor fiind o aplicație gratis.

2. Argumentarea de ce este necesar crearea unui astfel de proiect.

Importanta creării unui astfel de proiect este ca sistemele de operare au player integrat, însă au o mulțime de funcționalități care eu nu le-am folosit si nu cred ca o sa le folosesc, prin urmare ocupa mai multa memorie RAM. Proiectul meu va avea minimul necesar, de exemplu slide cu volum, slide-ul muzicii care va permite rularea cântecului înapoi sau înainte, adăugarea si ștergerea pieselor, modificarea denumirii unui cântec si modificarea copertei cântecului. De asemenea eu mă folosesc de SO Linux care are doar minimul necesar si prin urmare player integrat lipsește. Proiectul meu va putea fi startat pe ambele sisteme de operare (Windows si Linux).

3. Studierea a mai multor sisteme asemănătoare. Compararea sistemelor si alegerea a celor mai bune funcționalități

Sunt o mulțime de sisteme existente care oferă posibilități diferite si arata diferit, însă majoritatea au integrat reclama sau o mulțime de funcționalități care ocupa memoria calculatorului. De asemenea sunt sisteme care sunt cu plata.

Mai jos sunt enumerate trei sisteme deja existente care fac parte din categoria proiectului meu:

1. Groove Music
2. Spotify
3. iTunes

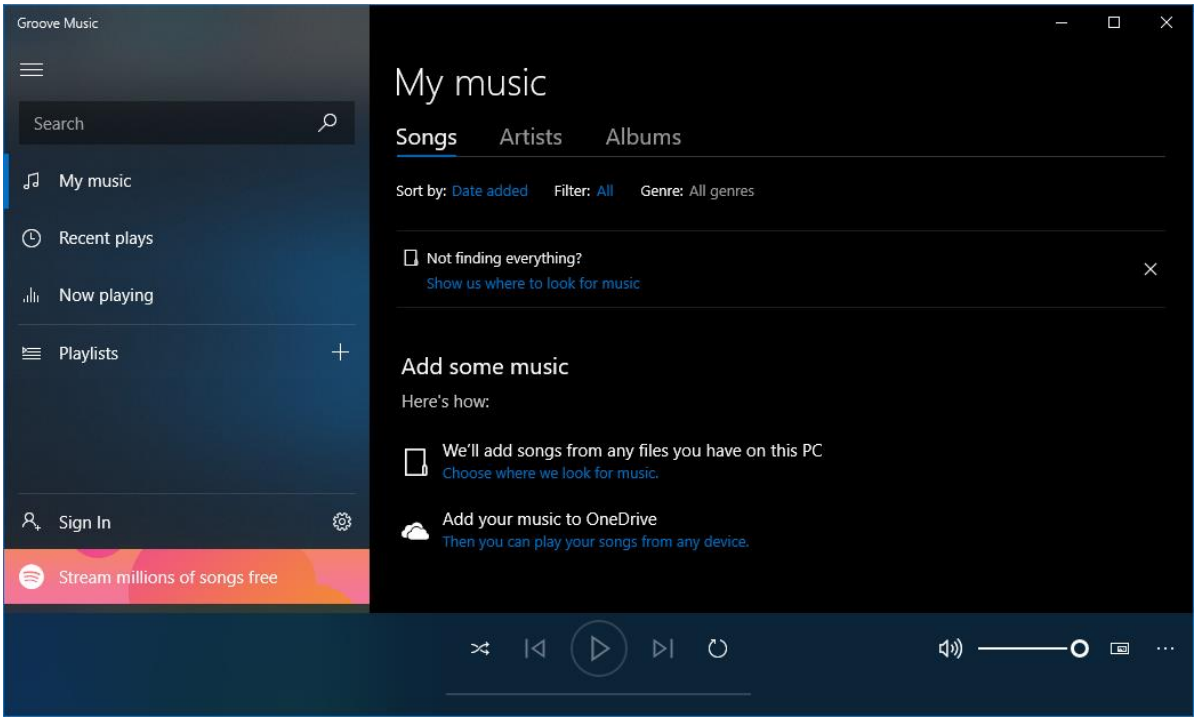

Figura 3.1 - Aplicația „Groove Music”

In Figura 3.1. Aplicația „Groove Music” este reprezentat player-ul integrat a sistemului de operare Windows. Cu el m-am folosit o buna parte din timp si pot spune ca design-ul este la un nivel înalt ca si funcționalitățile, însă cum am spus anterior, ocupa memorie in plus pentru toate funcționalitățile lui care eu nu le folosesc. Încă un punct pozitiv ar fi ca este absolut gratis fără reclama sau ceva de genul in comparație cu sistemele care vor urma. In comparație cu sistemul meu, aplicația Groove Music nu poate încarcă file-urile audio din alte directorii, doar de ales o directorie anumita unde se vor afla toate fișierele

audio, default este directoria „Music” creata automat de către sistemul de operare Windows. Mai jos vom putea observa diferența intre sistemul dat si sistemul care este creat de mine.

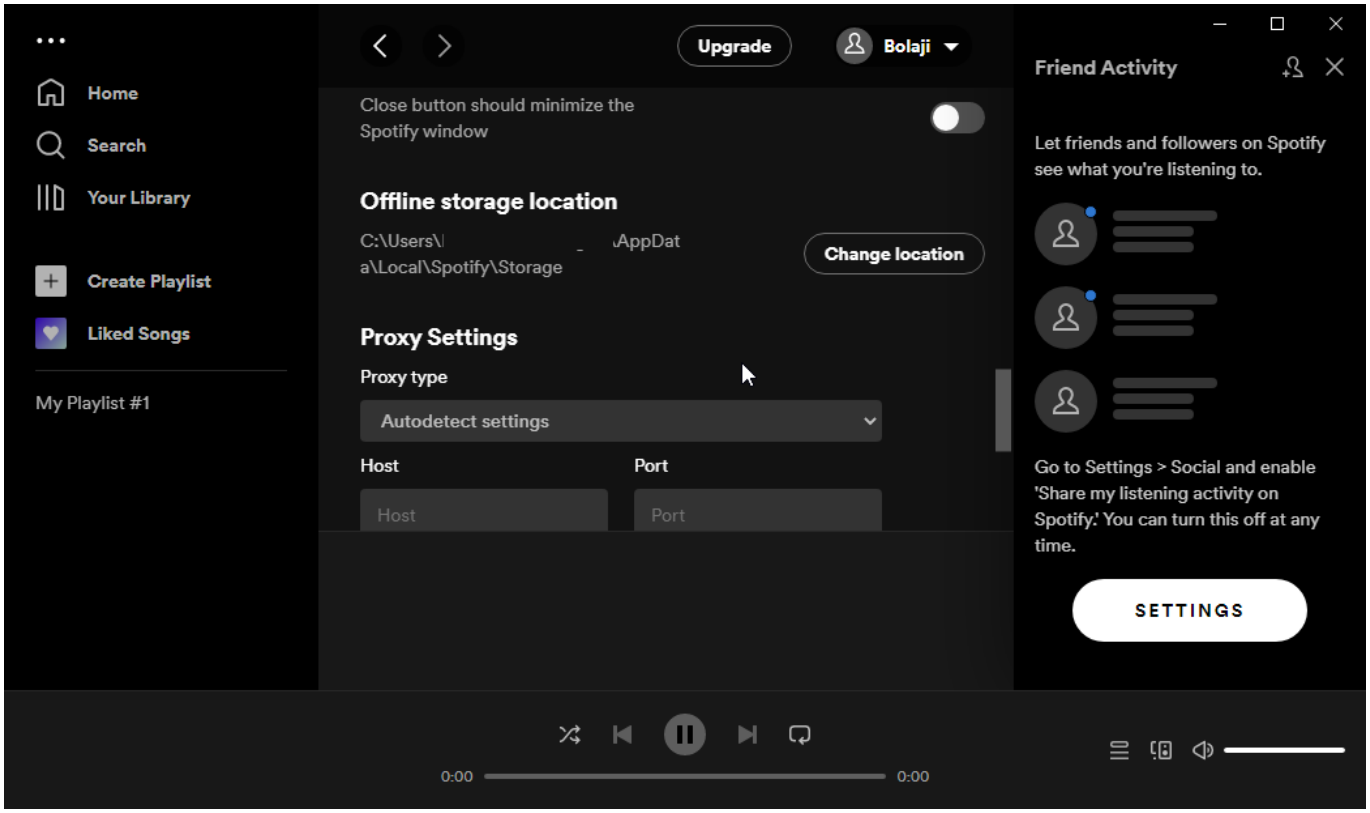

Figura 3.2 - Aplicația „Spotify”

In Figura 3.2. Aplicația „Spotify” este reprezentata o aplicație cu o biblioteca mare de muzica, fiind numărul unul in lume in categoria sa. In funcționalități are minimul necesar, si un design frumos. Ca si Groove Music, o buna parte din timp m-am folosit si de Spotify, si pot spune ca este o aplicație limitata, adică nu poți asculta de pe local muzica ta, nu poți modifica nimic (am in vedere cântecele din biblioteca), fiecare luna trebuie sa plătești ca sa asculți muzica, daca nu atunci va apărea reclama fiecare al doilea cântec, si fiind o aplicație care gestionează cântecele online, ocupa memorie fizic puțin, însă memoria RAM ocupata este mai mare decât sistemul precedent.

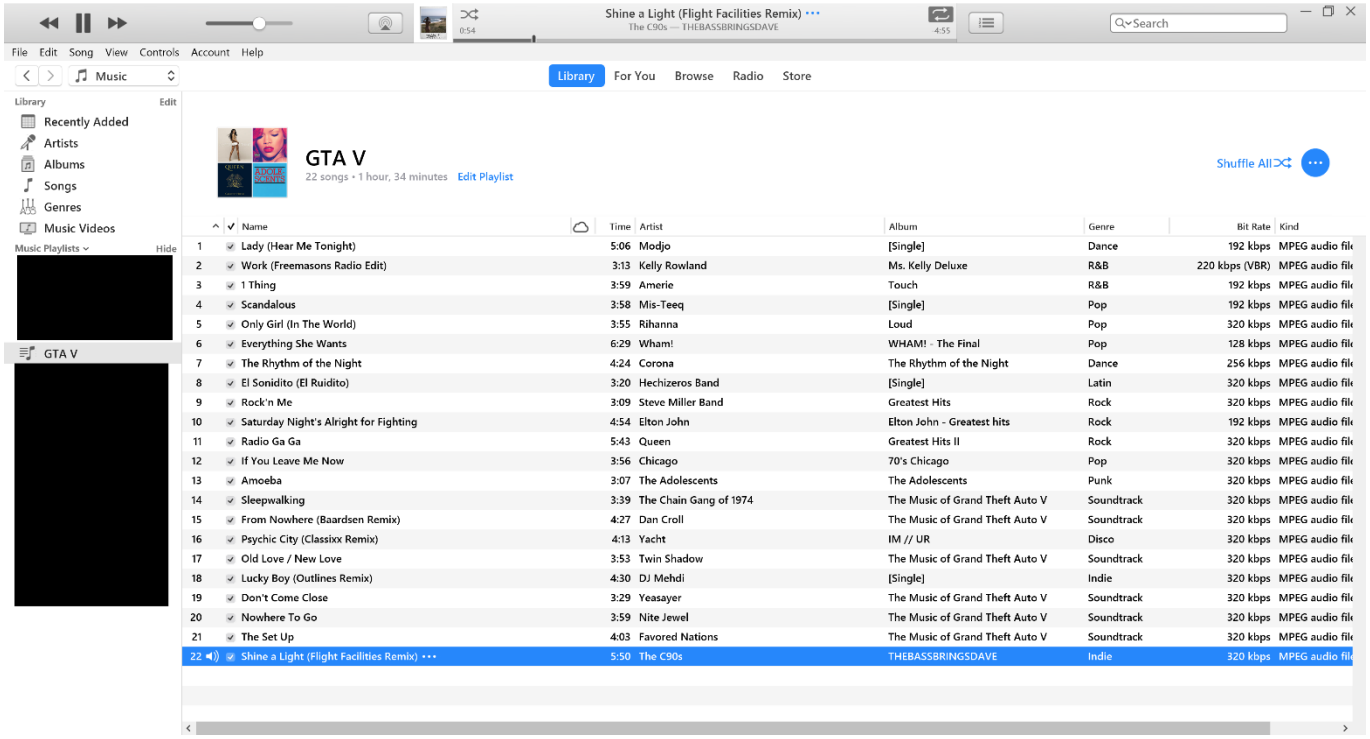

Figura 3.3 - Aplicația „iTunes”

In Figura 3.3. Aplicația „iTunes” este reprezentata o aplicație făcută de compania Apple inc. care ca si Spotify-ul oferă o biblioteca cu muzica larga pentru o plata lunara. Din experiența proprie pot spune ca biblioteca la iTunes este mai mica decât la Spotify, dar plata este la fel. De asemenea design-ul pentru mine este unul deja învechit. La fel ca si sistemul anterior fișierele audio de pe local nu pot fi încărcate.

Nu in ultimul rând, sistemele de mai sus la care este posibil încărcarea fișierelor audio de pe mașina locala nu fac copie la file-ul ales de utilizator, adică pe viitor dacă vom șterge directoria unde se afla file-ul audio, el va nimeri in coșul de gunoi si nu o să putem sa-l accesăm. Aplicația mea a rezolvata aceasta problema prin copierea file-ului sau a mai multor file-uri deodată in directoria aplicației, prin urmare el va ocupa o memoria mai mare decât sistemele precedente, însă fișierele audio vor fi accesibile până când nu vom șterge aplicația dată de pe sistemul de operare.

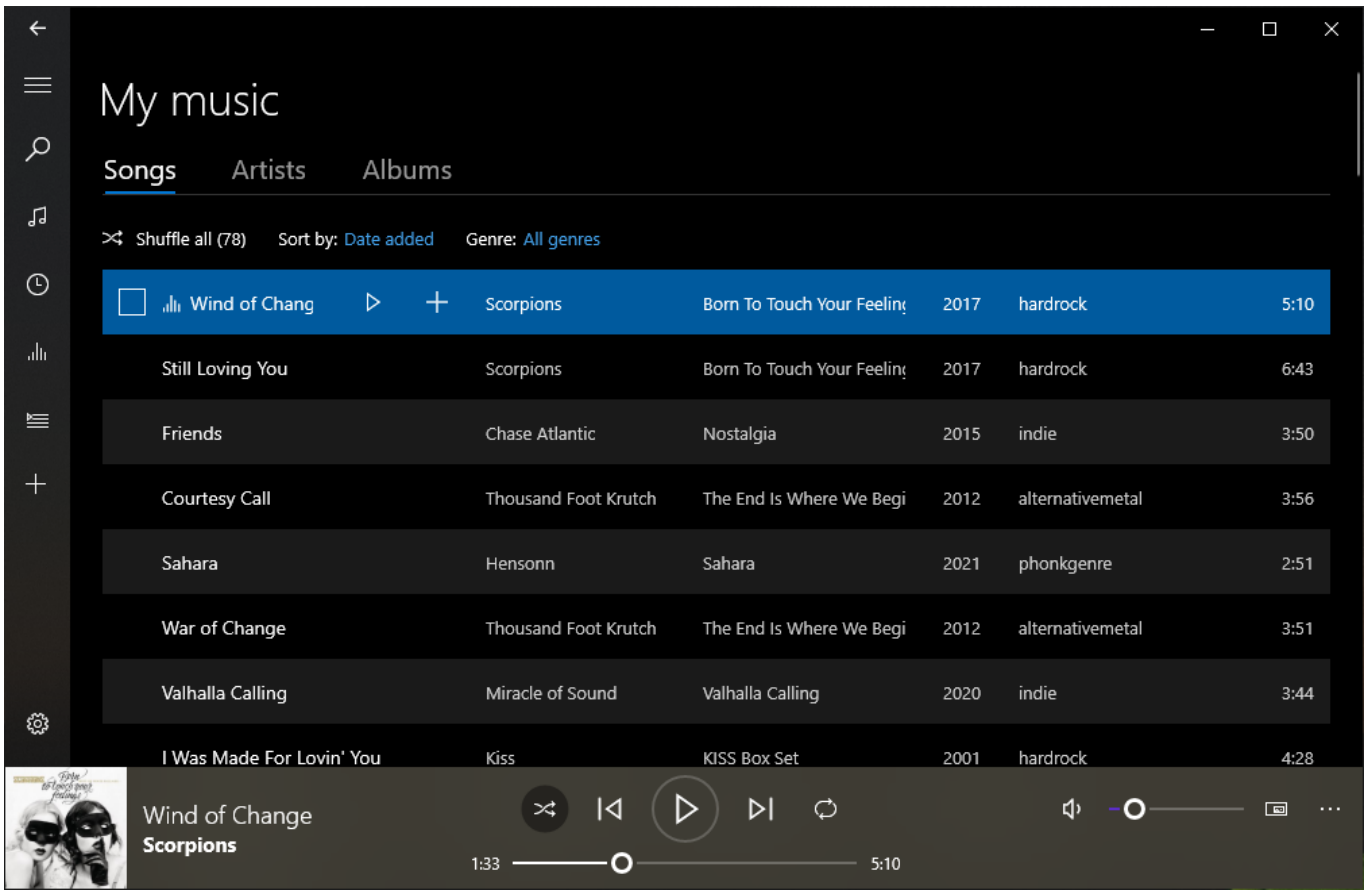

Figura 3.4 - Meniul principal a aplicației „Groove Music”

In Figura 3.4. Meniul principal a aplicației „Groove Music” putem observa aplicației „Groove Music” împreuna cu funcționalul acesteia. Meniul conține o lista cu toate fișierele audio încărcate de către utilizator, de asemenea in partea de jos a aplicație se afla tot funcționalul de baza (Butoanele, Slide-urile, Cover-ul fișierului audio etc.). Interfața aplicației date conține o mulțime de butoane, care dup apararea mea sunt in plus. Aplicația „Music Player” va oferi doar un funcțional de baza pentru citirea, redarea si nu in ultimul rând gestionarea fișierelor audio.

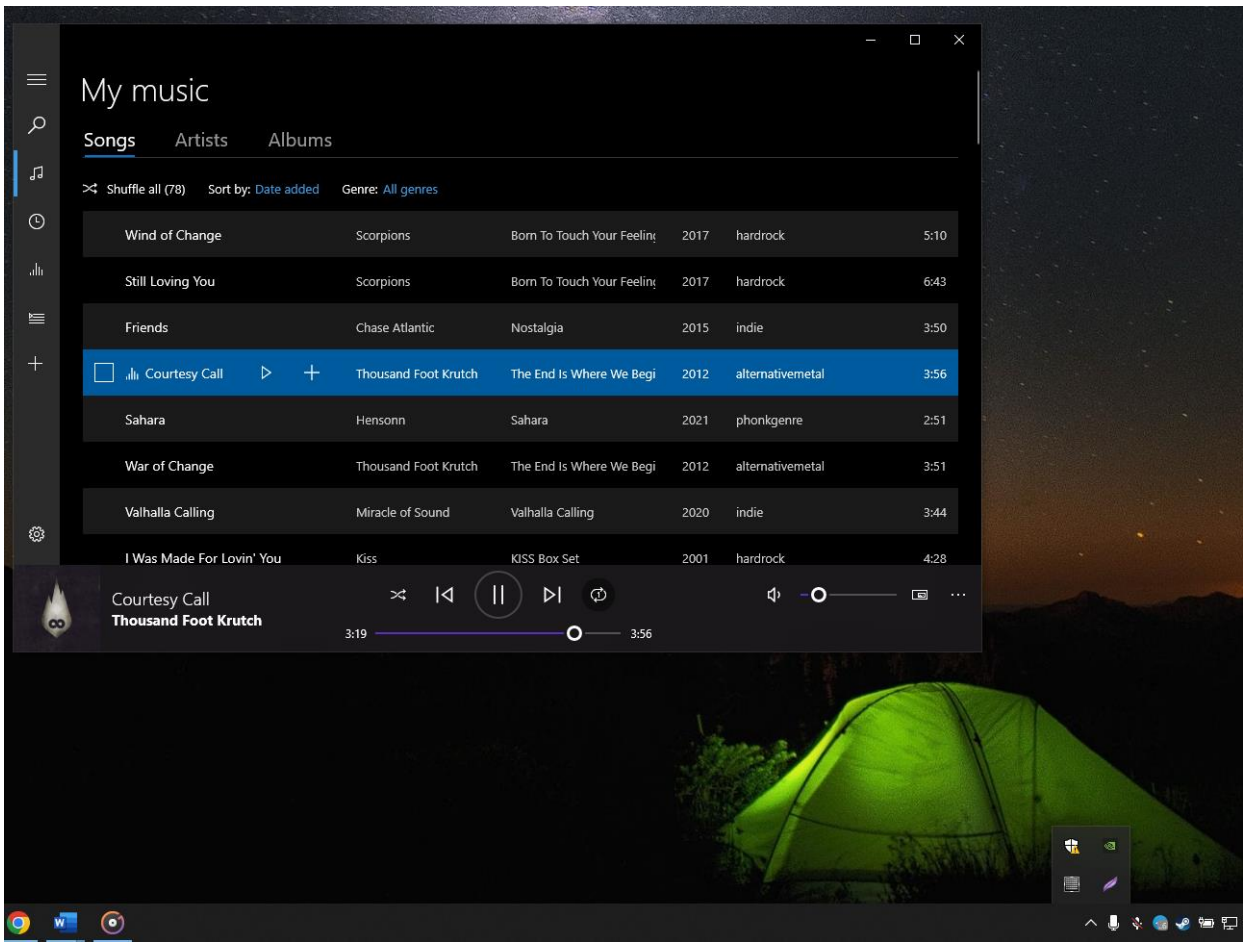

Figura 3.5 - Tray meniul SO Windows

In Figura 3.5. Tray meniul SO Windows este reprezentata aplicația „Groove Music”. Cum putem observa, aplicația nu poate funcționa pe fundal. Prin funcționare pe fundal am in vedere ca daca închidem aplicația, ea nu va apărea in tray meniu, prin urmare se va închide total. Aplicația mea va avea funcția ca daca închidem aplicația, ea va funcționa pe fundal pana la momentul când utilizatorul nu va alege opțiunea „Exit” din meniul propus in urma apăsării butonului click-drept a mouse-ului pe icon-ul aplicației ce se afla in tray meniul sistemului de operare.

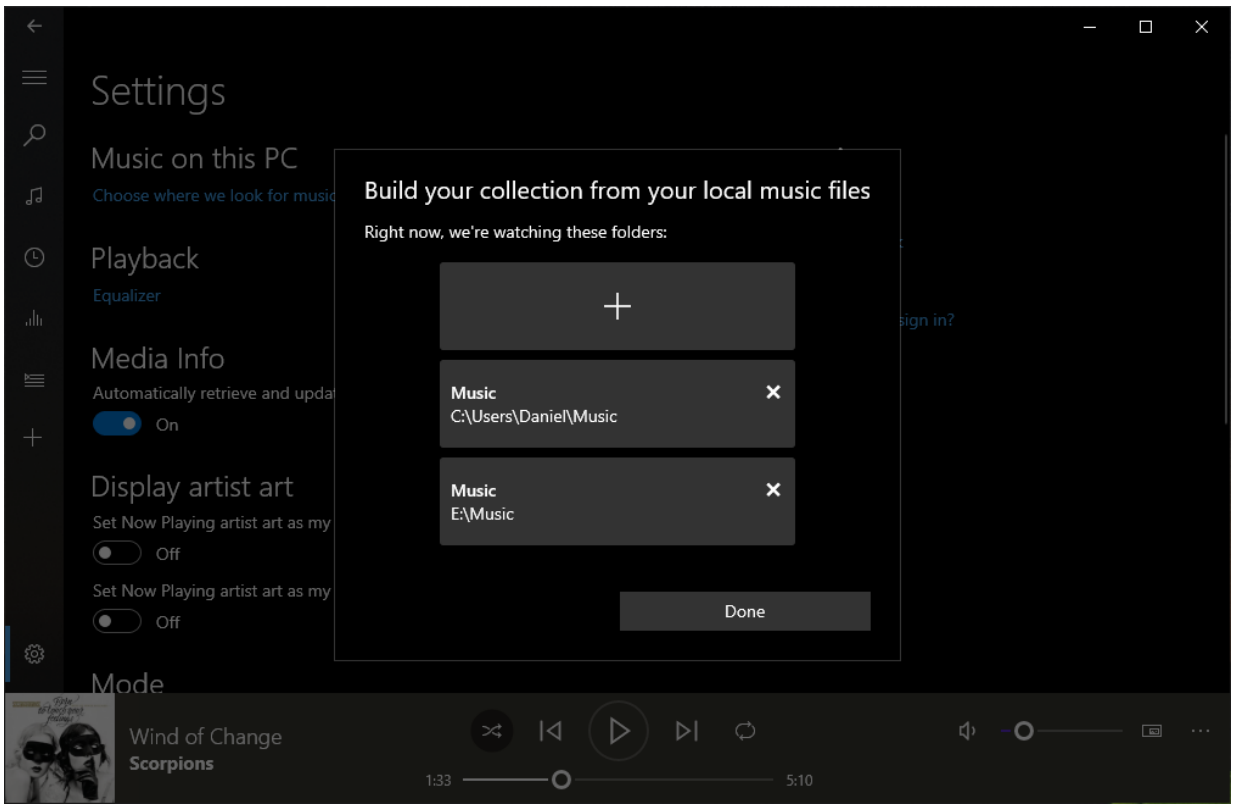

Figura 3.6 - Funcția de adăugare a fișierelor audio a aplicației „Groove Music”

In Figura 3.6. Funcția de adăugare a fișierelor audio a aplicației „Groove Music” este arătata funcția de adăugare a fișierelor audio in aplicația „Groove Music”. Cum putem vedea, aplicația nu adaugă fișierele propriu-zis, dar directoriul unde ele se afla, prin urmare utilizatorul trebuie manual sa le adauge in directoria aleasa. Aplicația mea va oferi posibilitatea sa alegem una sau mai multe fișiere de-odată din directorii diferite, deoarece după ce alegem fișierele, aplicația automat face cate-o copie pentru fiecare si le adaugă in directoria aplicației, prin urmare daca vom șterge fișierele selectate din directoriile de baza, ele nu se vor șterge din aplicație, deoarece exista copie pentru fiecare fișier.

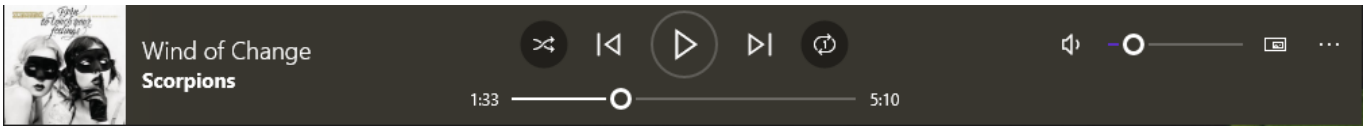

Figura 3.7 - Funcțiile de baza a aplicației „Groove Music”

In Figura 3.7. Funcțiile de baza a aplicației „Groove Music” se afla funcționalul de baza a aplicației „Groove Music”. Aici se afla butoanele ce oferă posibilitatea de gestionare a fișierelor audio, cum ar fi: butonul „Play”, butonul „Pause”, „Next music”, „Prev. Music”, „Mute volume”. Pe lângă butoanele ce oferă funcțional pentru manipularea fișierelor audio, exista butoane pentru modificarea modului de redare a fișierelor cum ar fi: „Shuffle Mode”, „Repeat Once” si „Reapeat This”. Fiecare din ele schimba modul de redare a fișierelor, de exemplu „Shuffle Mode” reda fișierele încărcate in mod

aleatoriu, „Repeat Once” oprește player-ul după ce se termina cântecul curent, iar „Repeat This” pune intr-un „loop” infinit cântecul curent.

Figura 3.8 - Ajustarea volumului in aplicația „Groove Music”

In Figura 3.8. Ajustarea volumului in aplicația „Groove Music” este arătat modul de manipulare a volumului canalului audio unde sunt redate fișierele audio din aplicația „Groove Music”. Cum putem vedea, după ce apăsam pe butonul „Mute”, slide-ul volumului nu se duce la 0, este un bug acesta a aplicației sau nu, dar aplicația care va fi proiectata de mine va avea funcția de setare a slide-ului volumului la 0 daca va fi apăsat butonul „Mute”, iar daca vom apasă din nou butonul „Mute” se va seta ultima valoare concomitent cu slide-ul ce reprezintă volumul in aplicație.

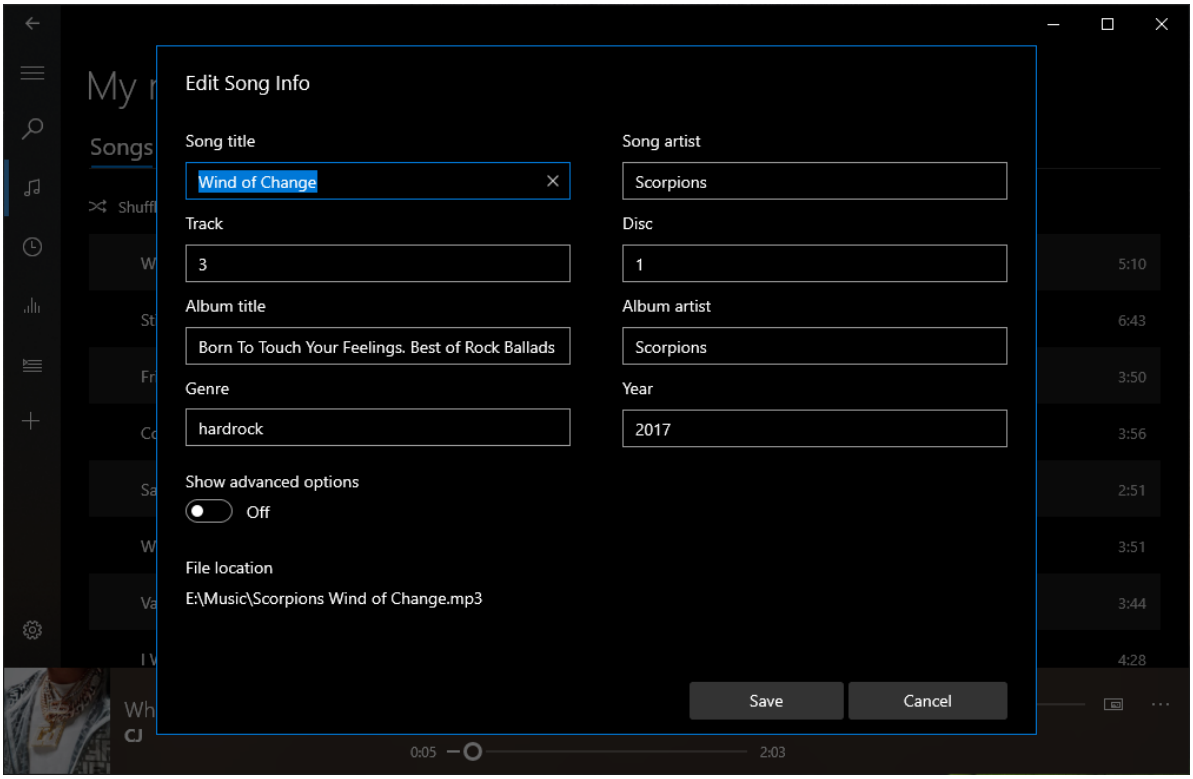

Figura 3.9 - Funcția de editare a unui fișierului audio in aplicația „Groove Music”

In Figura 3.9. Funcția de editare a unui fișierului audio in aplicația „Groove Music” putem observa funcția de editare a fișierului audio selectat a aplicației „Groove Music”. Din imagine putem vedea ca oferă posibilitatea la schimbarea denumirii fișierului, schimbarea artistului, schimbarea albumului, genului, anul apariției etc. De asemenea oferă adresa fișierului selectat, in cazul nostru fișierul se afla pe

discul „E” in directoria „Music”. Aplicația mea va oferi posibilitatea de asemenea la redactarea fișierului selectat. Va fi posibila schimbarea denumirii fișierului, numele artistului. Pe lângă acest funcțional, va oferi posibilitate la schimbarea cover-ului cântecului ce nu oferă aplicația creata de compania Microsoft „Groove Music”. Prin cuvântul „cover” mă refer la imaginea setata pentru fiecare fișier audio, in aplicația „Groove Music” cover-ul cântecului selectat se afla in stânga-jos a ferestrei aplicației.

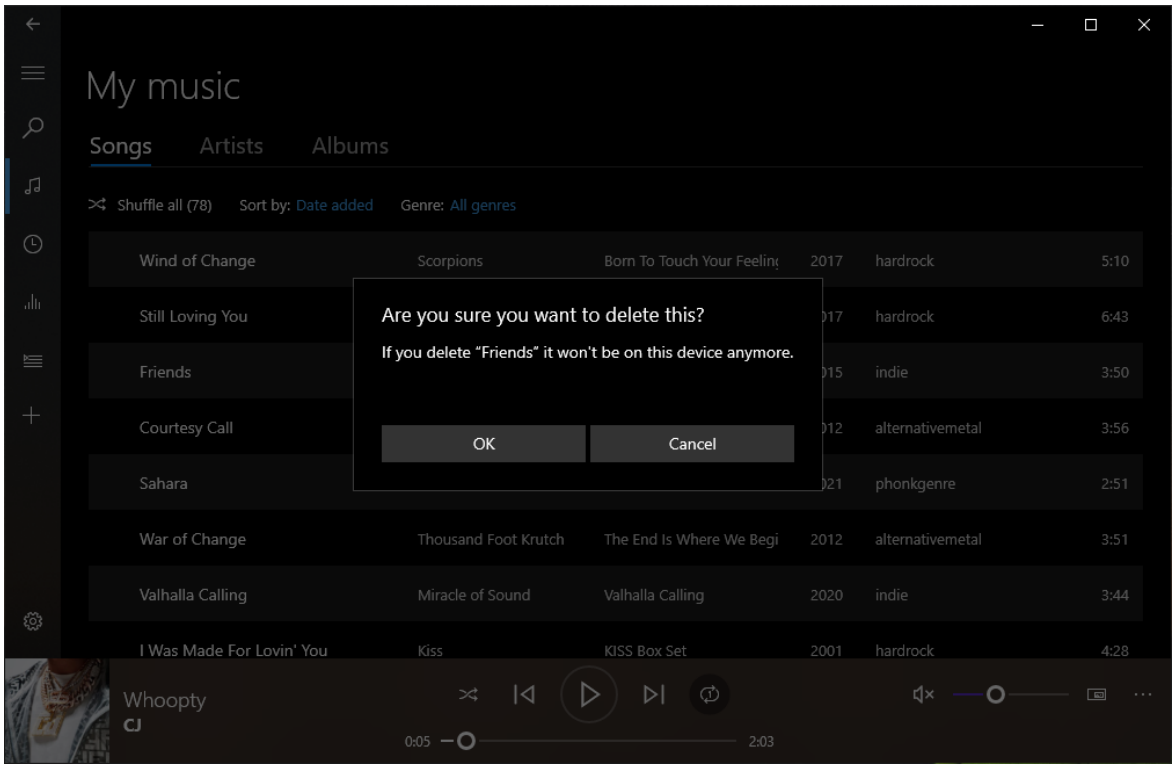

Figura 3.10 - Funcția de ștergere a unui fișierului audio in aplicația „Groove Music”

In Figura 3.10. Funcția de ștergere a unui fișierului audio in aplicația „Groove Music” putem vedea funcția de ștergere a aplicației „Groove Music”. Daca alegem opțiunea „Delete” va apărea o fereastra pop-up pentru confirmarea ștergerii a fișierului selectat. După ștergerea lui, el automat se va șterge din directoriul unde se afla, adică va dispărea de pe mașina locala. Aplicația mea va oferă fix așa funcție, doar ca după ștergerea fișierului selectat, aplicația va șterge copia ce a fost creata in directoriul aplicația, iar originalul va rămâne neatins pe mașina locala a utilizatorului.

4. Descrierea scopului si obiectivelor care vor fi realizate.

1. Crearea sistemului de „Prelucrarea fișierelor audio”;
2. Analiza soluțiilor existente de „Prelucrarea fișierelor audio”;
3. Elaborarea caietului de sarcini;

4. Argumentarea platformei și soft-ului de realizare;
5. Cercetarea si selectarea instrumentelor pentru dezvoltarea platformei;
6. Modelarea funcționala a platformei;
7. Proiectarea structurii de baza a proiectului;
8. Implementarea acțiunilor de citire si redare a fișierelor audio;
9. Crearea componentei de vizualizarea a aplicației;
10. Proiectarea unei bazei de date de tip NoSQL;
11. Adăugarea funcționalului de schimbare a modului de redare;
12. Implementarea operațiilor CRUD;
13. Testarea sistemului;
14. Documentarea sistemului informațional;
15. Planificarea proiectului si estimarea costului;
16. Concluzii și recomandări;

FIŞA DE ACTIVITATE

Săptămâna II

Sarcini planificate:

1. Modelarea si proiectarea sistemului informatic
2. Realizarea sistemului

1. Modelarea si proiectarea sistemului informatic

In figura 1.1 este reprezentat procesele principale a aplicației.

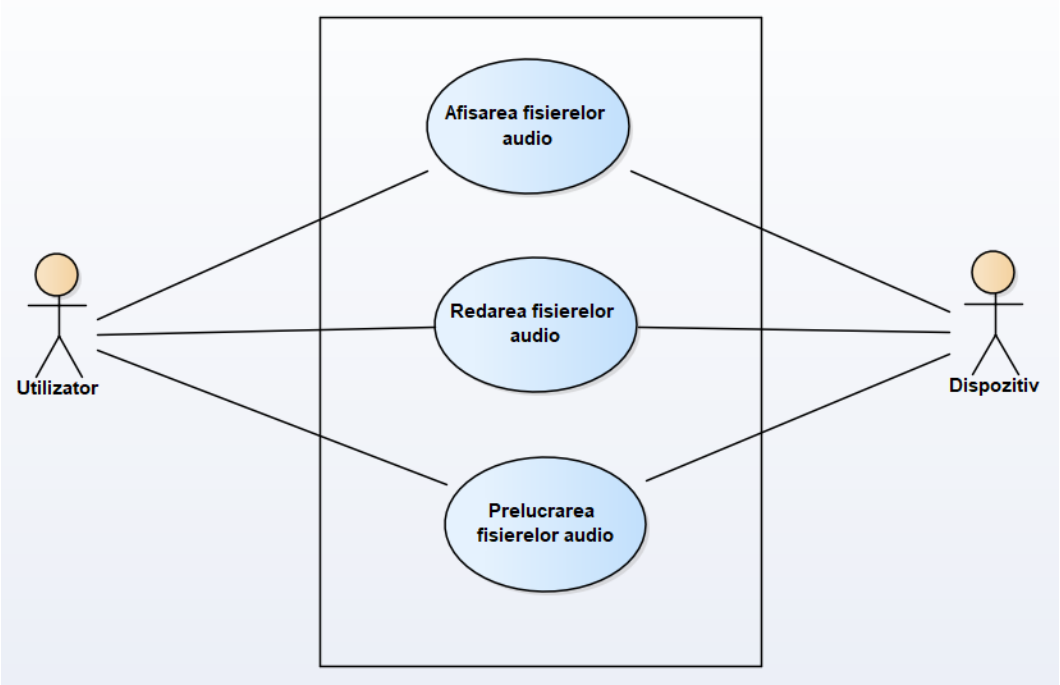

Figura 1.1 - Procesele principale ale utilizatorului

In figura 1.2 este reprezentat de încărcare a unui fișier audio cu ajutorul diagramei use-case.

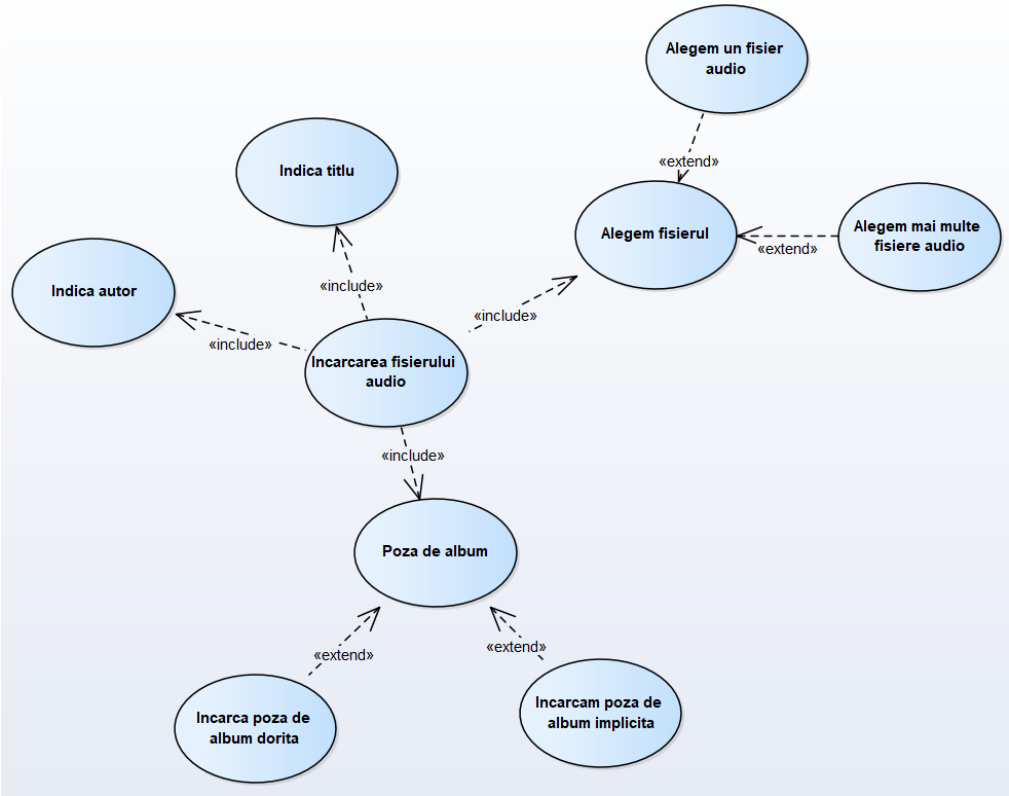

Figura 1.2 - Procesele de încărcare a fișierului audio

In figura 1.3 este reprezentat procesele de pornire a aplicației cu ajutorul diagramei de colaborare.

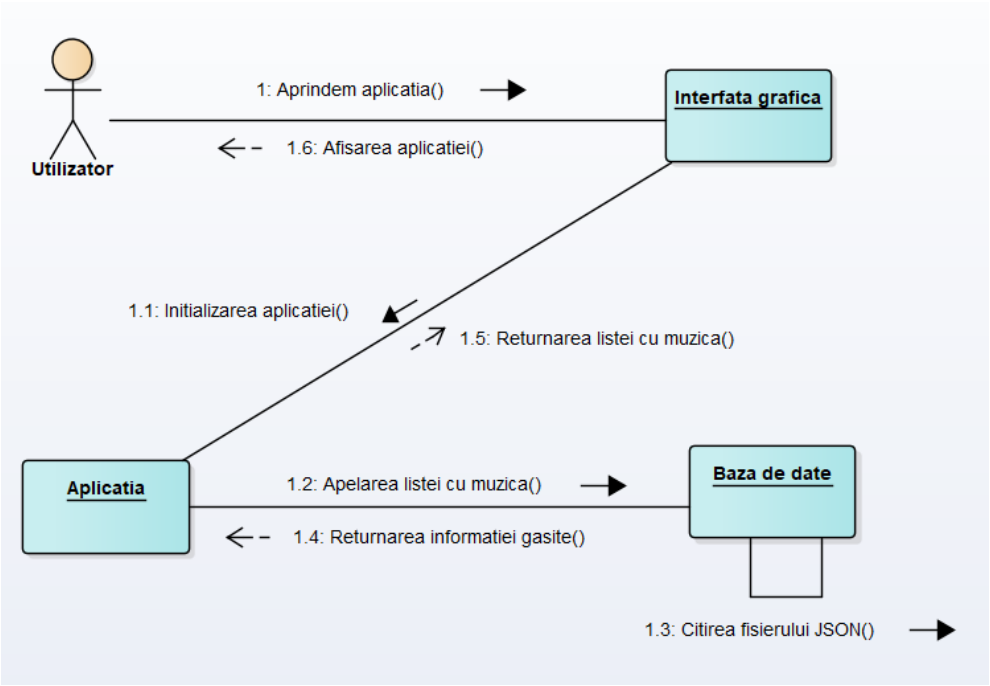

Figura 1.3 - Procesele de pornire a aplicației

In figura 1.4 este reprezentat diagrama secvențiala a procesului de editare a unui cântec.

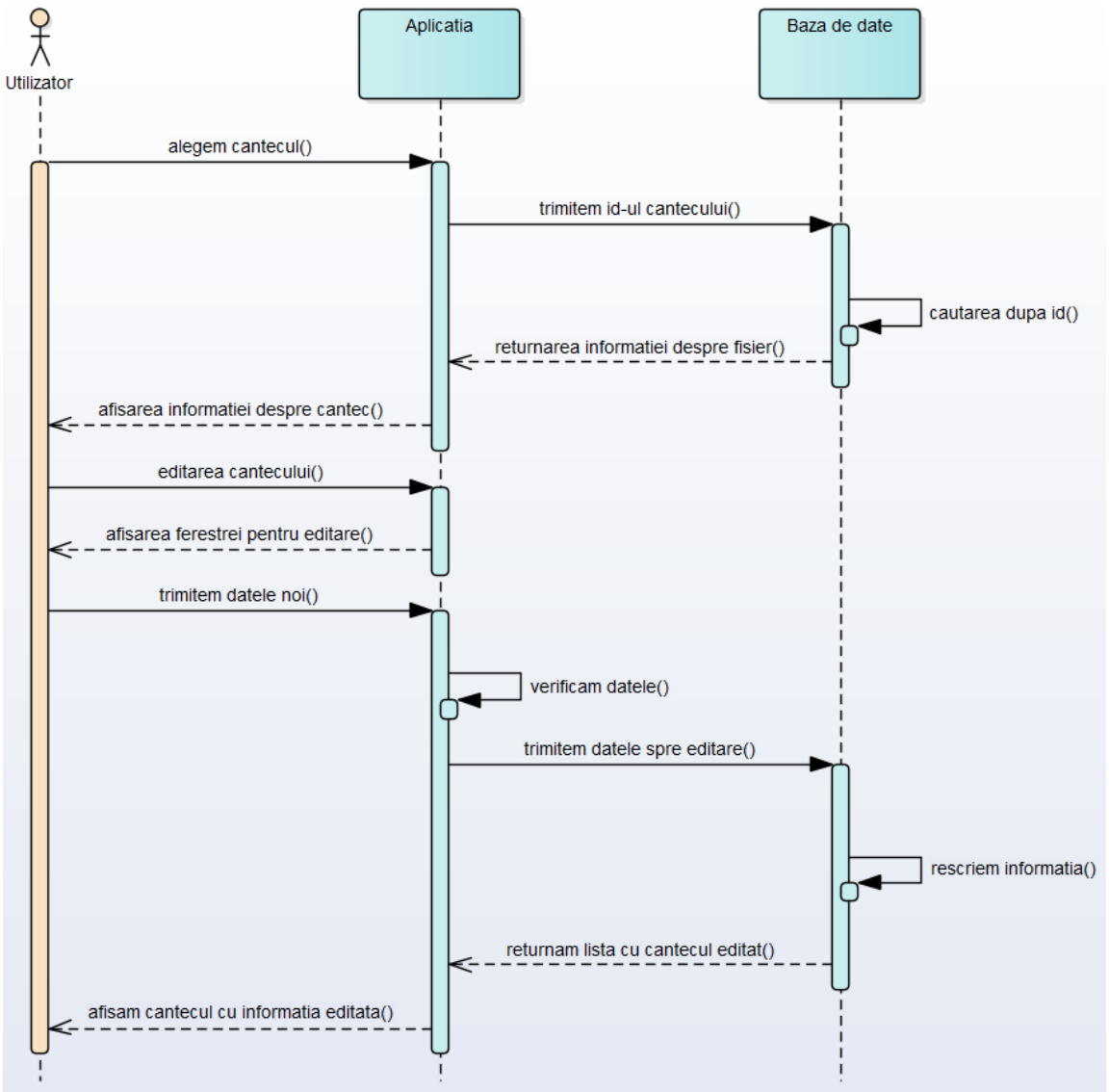

Figura 1.4 - Procesele de editare a unui cântec

In figura 1.5 este reprezentat diagrama de clasa a aplicației.

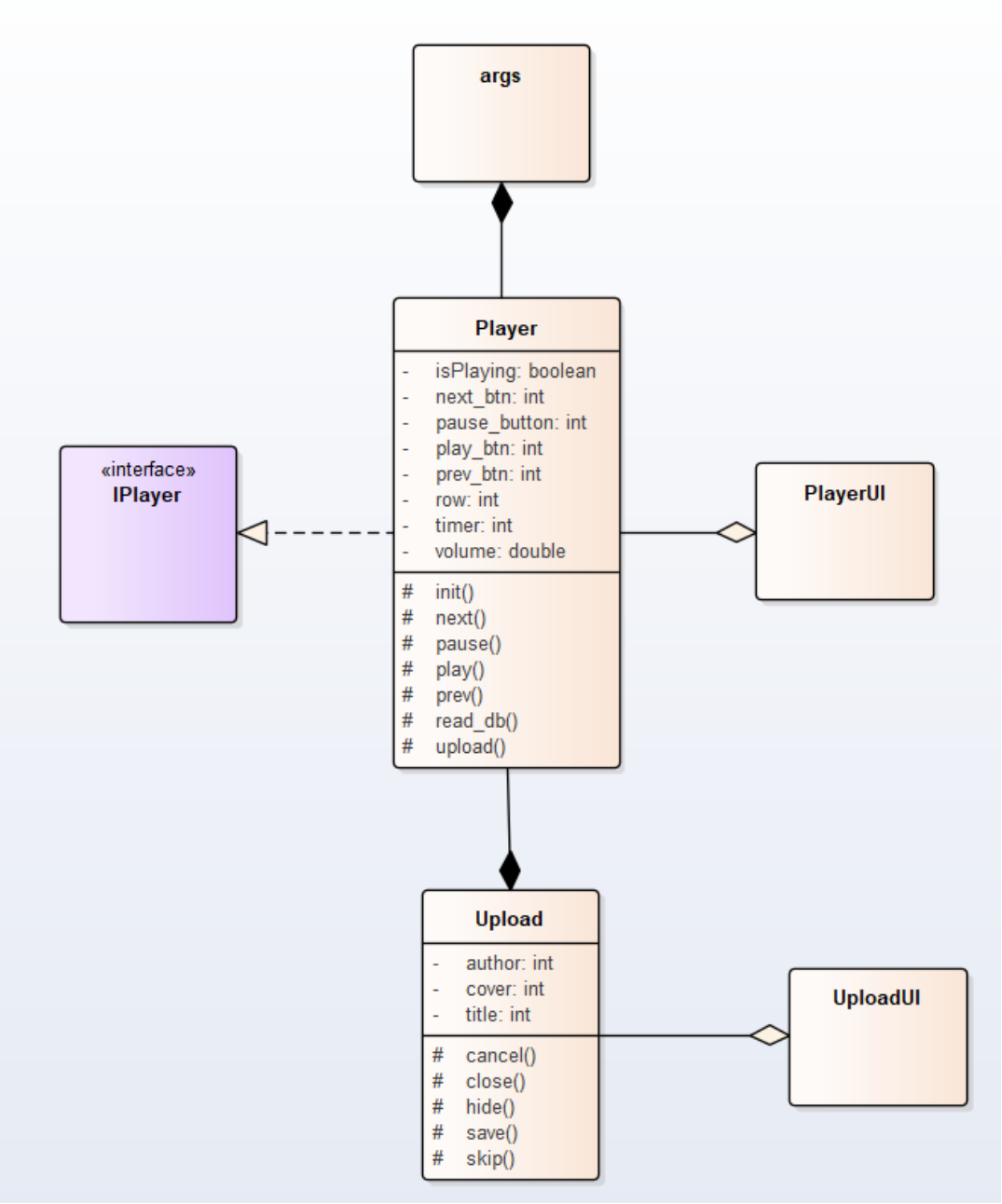

Figura 1.5 - Diagrama de clasa a aplicației

In figura 1.6 este reprezentat diagrama de activitate a procesului de redare a unui cântec.

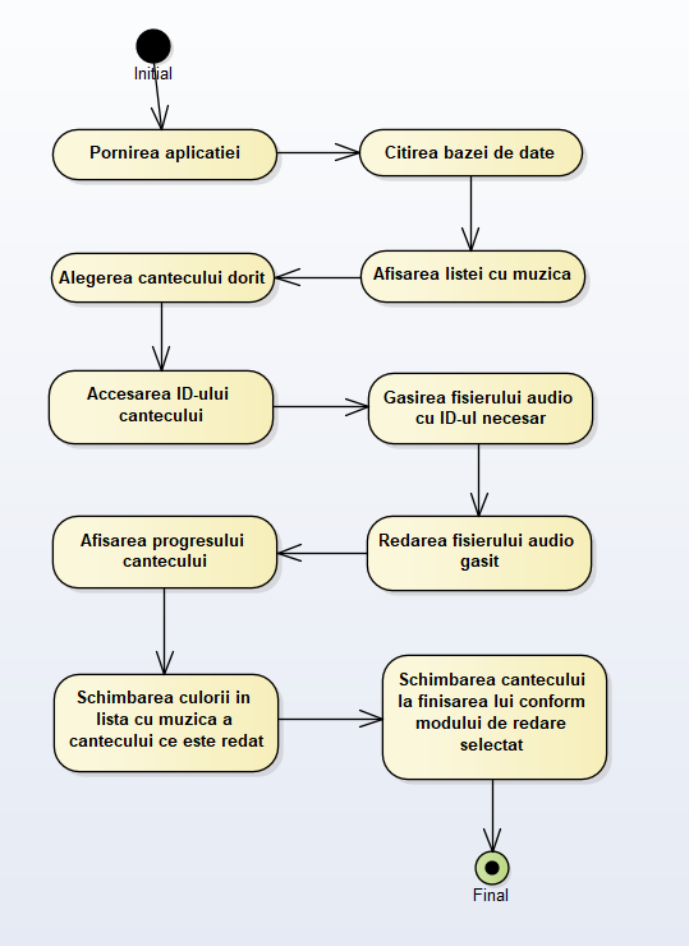

Figura 1.6 - Diagrama de activitate pentru procesul de redare a unui cântec

In figura 1.7 este reprezentat procesul de ștergere a unui cântec cu ajutorul diagramei de stare.

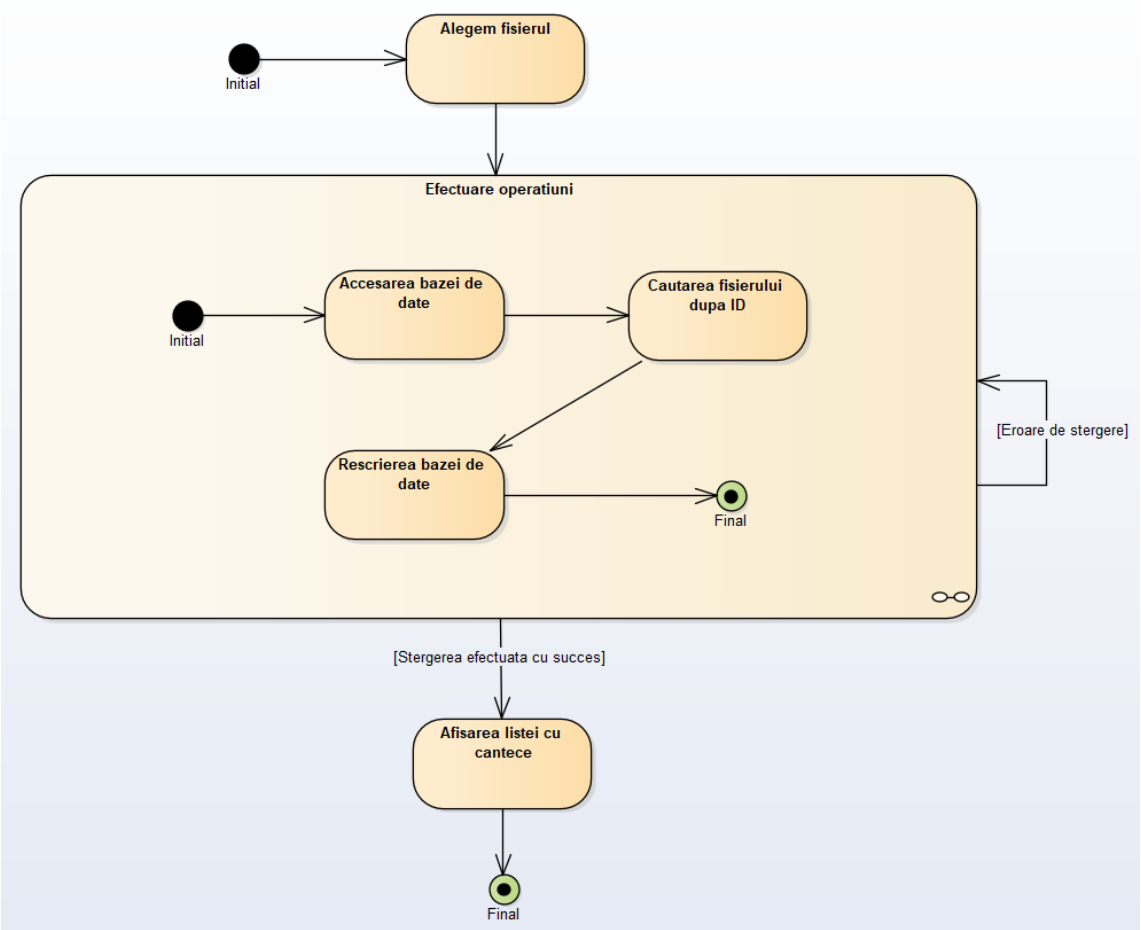

Figura 1.7 - Diagrama de stare a procesului de ștergere a unui cântec

In figura 1.8 este reprezentat componentele principale a aplicației.

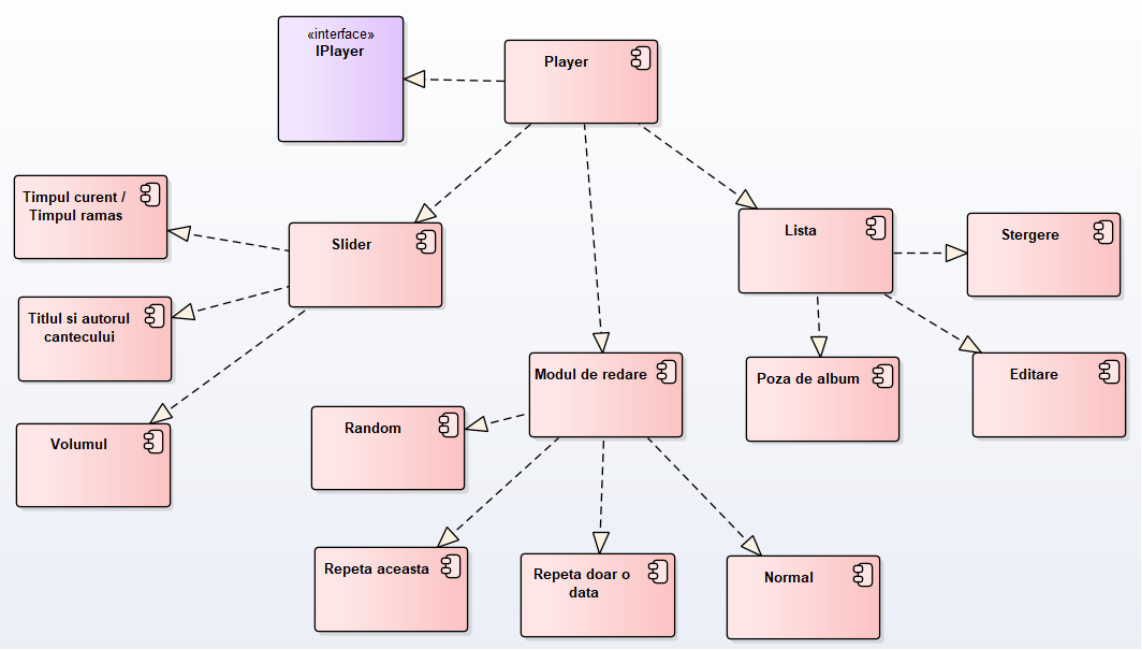

Figura 1.8 - Componentele aplicației

In figura 1.9 este reprezentat sistemul de operare unde este păstrată aplicația.

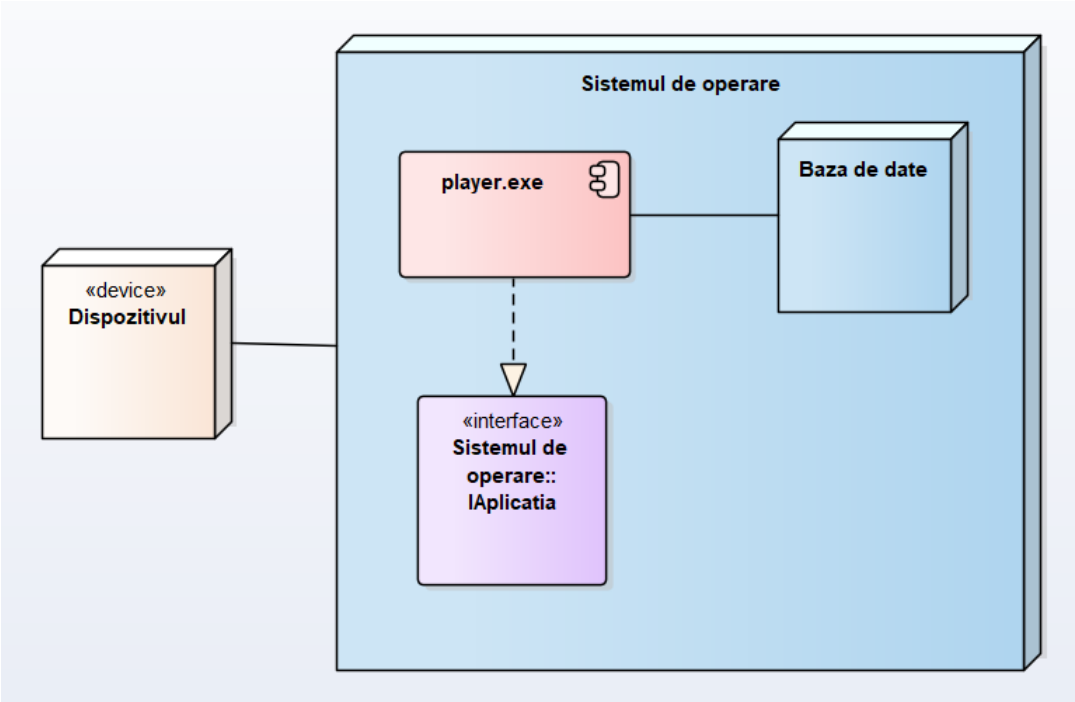

Figura 1.9 - Reprezentarea sistemului de operare unde este păstrată aplicația

2. Realizarea sistemului

Python este un limbaj de programare la nivel înalt, interpretat și dinamic, care este utilizat pe scară largă pentru o gamă largă de scopuri, inclusiv dezvoltarea web, calculul științific, analiza datelor,

inteligența artificială și multe altele. Python este cunoscut pentru simplitatea și lizibilitatea sa, ceea ce îl face o alegere populară atât pentru începători, cât și pentru dezvoltatorii experimentați.

PyQt este un set de legături Python pentru cadrul aplicației Qt și rulează pe toate platformele acceptate de Qt, inclusiv Windows, OS X, Linux, iOS și Android. PyQt este o alegere populară pentru dezvoltarea aplicațiilor desktop datorită ușurinței sale de utilizare și comunității mari de utilizatori și dezvoltatori. Oferă un set bogat de componente ale interfeței grafice cu utilizatorul (GUI) și acceptă dezvoltarea rapidă a aplicațiilor (RAD).

JSON (JavaScript Object Notation) este un format ușor de schimb de date care este ușor de citit și scris de oameni și ușor de analizat și generat de mașini. Este adesea folosit ca bază de date pentru aplicații de dimensiuni mici și mijlocii, în special în dezvoltarea web, datorită simplității și ușurinței sale de utilizare. JSON este, de asemenea, folosit ca o modalitate de schimb de date între un client și un server, ceea ce îl face o alegere populară pentru API-uri.

În concluzie, Python este un limbaj de programare versatil și popular, care este utilizat pe scară largă pentru o varietate de scopuri. PyQt oferă un set puternic de instrumente pentru dezvoltarea aplicațiilor desktop cu un set bogat de componente GUI. JSON este o alegere populară pentru bazele de date mici și mijlocii, în special în dezvoltarea web, datorită simplității și ușurinței sale de utilizare.

In figura 2.1 este reprezentat structura sistemului.

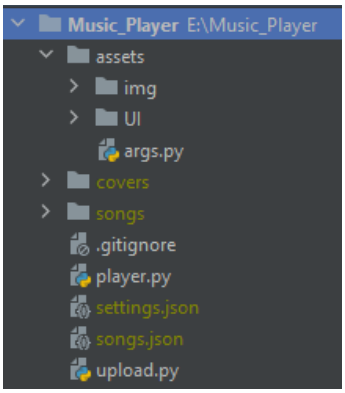

Figura 2.1 - Structura aplicației

In codul de mai jos este reprezentat citirea file-ului de tip JSON unde este stocata informația despre fișierele audio.

```
def read_songs_from_json(self):
    if not os.path.exists('songs'):
```

```
            os.makedirs('songs')
        self.player = QMediaPlayer()
        self.playlist = QMediaPlaylist(self.player)
        try:
            with open("songs.json", "r", encoding="utf-8") as file:
                data = json.load(file)
            self.titles.clear()
            self.artists.clear()
            self.covers.clear()
            for i in data["Songs"]:
                self.titles.append(i["title"])
                self.artists.append(i["artist"])
                if i["cover"] == "Undefined":
                    self.covers.append("no_image.jpg")
                else:
                    self.covers.append(i["cover"])
        except Exception as e:
            print(e)
        self.read_files_songs()
```

In codul de mai jos este reprezentat citirea si rescrierea fișierului „settings” ce indica setările player-ului făcute de către utilizator.

```
    def settings_read(self):
        try:
            with open("settings.json", "r", encoding="utf-8") as f:
                data = json.load(f)
            for i in data["Settings"]:
                self.volume = i["Volume"]
                self.lastVolume = self.volume
                self.row = i["Row"]
                self.mode = i["Mode"]
            self.currentIndex = self.row
        except Exception as e:
            print(e)

    def settings_write(self):
        settings_list = {}
        settings_list["Settings"] = []
        settings_list["Settings"].append({
            "Volume": self.volume,
            "Row": self.row,
            "Mode": self.mode
        })
        with open("settings.json", "w", encoding="utf-8") as f:
```

```
            json.dump(settings_list, f, indent=4)
```

In codul de mai jos este reprezentat funcționalul butoanelor „Play”, „Next” si „Previous”.

```
    def play(self):
        if len(self.titles) > 0:
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.newIndex = self.player.playlist().currentIndex()
                self.checkStyle()
            else:
                self.player.pause()
                self.isPlaying = False
                self.checkStyle()



    def next(self):
        if len(self.titles) > 0:
            self.playlist.next()
            self.newIndex = self.player.playlist().currentIndex()
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.ui.playButton.setStyleSheet(pause_btn_css)

    def prev(self):
        if len(self.titles) > 0:
            if int(self.now_sec) < 10:
                self.playlist.previous()
                self.newIndex = self.player.playlist().currentIndex()
            else:
                self.player.setPosition(0)
            if not self.isPlaying:
                self.player.play()
                self.isPlaying = True
                self.ui.playButton.setStyleSheet(pause_btn_css)
```

In codul de mai jos este reprezentat timer-ul aplicației. Aceasta metoda este ca un „loop”, adică este chemata de mai multe ori pe secunda, dar mai concret 60 de ori pe secunda (60 de cadre pe secunda maxim poate sa observe ochiul uman).

```
    def time_hit(self):
        self.checkStyle()
        self.checkstyleVolume()
        if self.isPlaying:
```

```
            self.ui.musicSlider.setMaximum(self.player.duration())
            if not self.ui.musicSlider.isSliderDown():
                self.ui.musicSlider.setValue(self.player.position())
            self.newIndex = self.player.playlist().currentIndex()
            self.checkList()

            song_min, song_sec = self.convertMillis(int(self.player.duration()))
            if song_sec < 10:
                self.song_duration = "{0}:0{1}".format(int(song_min), int(song_sec))
            else:
                self.song_duration = "{0}:{1}".format(int(song_min), int(song_sec))

            now_min, self.now_sec = self.convertMillis(int(self.ui.musicSlider.value()))
            if self.now_sec < 10:
                self.now_duration = "{0}:0{1}".format(int(now_min), int(self.now_sec))
            else:
                self.now_duration = "{0}:{1}".format(int(now_min), int(self.now_sec))

            self.ui.durationLabel.setText(str(self.now_duration) + " / " +
str(self.song_duration))

            if self.repeatonce:
                if self.now_duration == self.song_duration:
                    self.isPlaying = False
                    self.ui.playButton.setStyleSheet(play_btn_css)
                    self.player.stop()
        self.settings_write()
```

De asemenea am rescris acțiunea unui aplicații obișnuite. In codul de mai jos putem observa ca am creat bara de titlu proprie a aplicației, am rescris eventul de închidere a aplicației si eventul de minimizare a ferestrei aplicației. Prin urmare acțiunea de închidere va face ca aplicația sa se mute in background, prin urmare nu va fi afișata fereastra aplicației, dar procesul redării fișierelor audio va continua mai departe.

```
    def mousePressEvent(self, event):
        self.start = self.mapToGlobal(event.pos())
        self.pressing = True

    def mouseMoveEvent(self, event):
        if self.pressing and (
                self.ui.titleBarLabel.underMouse() or self.ui.titleBarInfoLabel.underMouse() or
self.ui.titleBarTitle.underMouse()):
            self.end = self.mapToGlobal(event.pos())
            self.movement = self.end - self.start
```

```
            self.setGeometry(self.mapToGlobal(self.movement).x(),
                             self.mapToGlobal(self.movement).y(),
                             self.width(),
                             self.height())
            self.start = self.end

    def minimizeButton_clicked(self):
        self.showMinimized()

    def closeButton_clicked(self):
        self.hide()

    def closeEvent(self, event):
        event.ignore()
        self.hide()
```

In figura 2.2 este reprezentat aplicația propriu-zis.

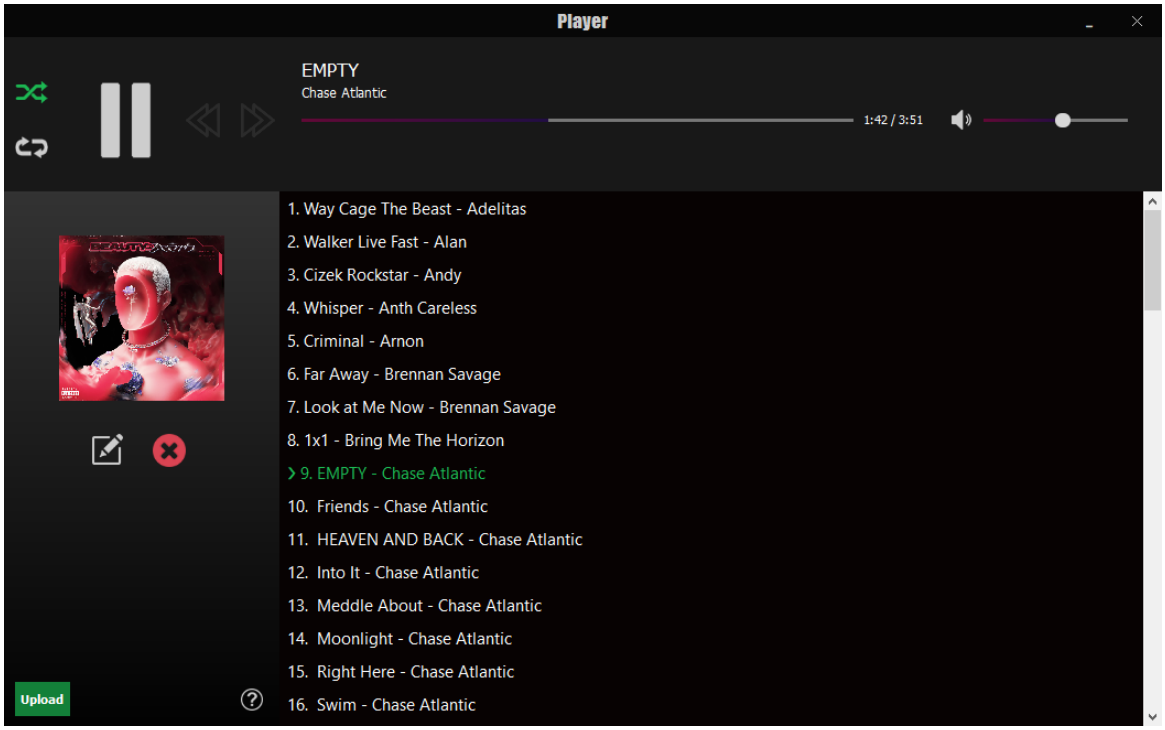

Figura 2.2 - Aplicația

FIŞA DE ACTIVITATE

Săptămâna III

Sarcini planificate:

1. Realizarea sistemului

1. Realizarea sistemului

Săptămâna aceasta am adăugat funcționalități noi pentru aplicația mea, una din funcționalități este butoanele de navigare (play/pause, next si prev) care se afla in toolbar-ul SO Windows cum putem sa vedem in figura 1.1.

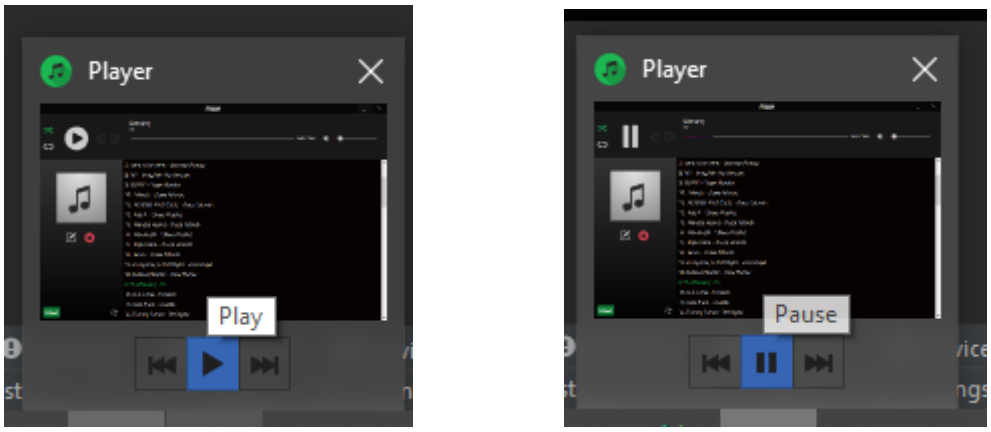

Figura 1.1 - Butoanele de pe toolbar-ul SO Windows.

Pentru realizarea acestei funcționalități, am folosit librăria principala a aplicației „pyqt5”, in care se afla pachetul „QtWinExtras”, unde la rândul lor se afla clasele „QWinThumbnailToolBar” si „QWinThumbnailToolButton”.

În codul de mai jos este arătat cum am implementat acest funcțional:

```
        self.toolBar = QWinThumbnailToolBar(self)


        self.toolBtnPrev = QWinThumbnailToolButton(self.toolBar)
        self.toolBtnPrev.setToolTip('Prev')
        self.toolBtnPrev.setIcon(self.style().standardIcon(QStyle.SP_MediaSkipBackward))
        self.toolBtnPrev.clicked.connect(self.prev)
        self.toolBar.addButton(self.toolBtnPrev)
```

```
        self.toolBtnControl = QWinThumbnailToolButton(self.toolBar)
        self.toolBtnControl.setToolTip('Play')
        self.toolBtnControl.setIcon(self.style().standardIcon(QStyle.SP_MediaPlay))
        self.toolBtnControl.clicked.connect(self.play)
        self.toolBar.addButton(self.toolBtnControl)

        self.toolBtnNext = QWinThumbnailToolButton(self.toolBar)
        self.toolBtnNext.setToolTip('Next')
        self.toolBtnNext.setIcon(self.style().standardIcon(QStyle.SP_MediaSkipForward))
        self.toolBtnNext.clicked.connect(self.next)
        self.toolBar.addButton(self.toolBtnNext)

    def showEvent(self, event):
        super(PlayerWindow, self).showEvent(event)
        if not self.toolBar.window():
            self.toolBar.setWindow(self.windowHandle())
```

Pe lângă adăugarea funcționalităților noi, am testat manual sistemul, in urma căreia s-au depistat 2 bug-uri majori si câteva bug-uri minore care au fost fixate cu succes. Testarea manuala de regula se realizează in câțiva pași:

- Instalare: Descărcați aplicația și instalați-o pe mașina dvs. Verificați dacă procesul de instalare este fără probleme și dacă nu există erori. Asigurați-vă că aplicația este instalată în locația dorită și sunt create comenzi rapide;
- Testare funcțională: Odată instalată, lansați aplicația și începeți să testați funcționalitățile acesteia. Încercați să redați fișierele muzicale și verificați dacă sunt redate corect. Verificați dacă puteți întrerupe, reda sau opri muzica după cum este necesar. Încercați să treceți la diferite părți ale muzicii pentru a vă asigura că funcționează corect. Testați diferite formate de fișiere media pentru a asigura ca aplicația poate primi doar fișiere cu extensia „.mp3”;
- Testarea interfeței cu utilizatorul: verificați aspectul general al aplicației. Asigurați-vă că toate butoanele, filele, meniurile și linkurile funcționează conform așteptărilor. De asemenea asigurați-vă că textul și imaginile sunt clare și vizibile;
- Gestionarea erorilor: Încercați să introduceți intenționat o intrare nevalidă sau să efectuați o acțiune care nu ar trebui să fie posibilă și verificați cum gestionează aplicația aceste erori, de exemplu la anularea încărcării unei poze de album, in acest caz poza trebuie sa rămână aceiași

care a fost înainte de apăsarea butonului de alegere a altei poze de album. Asigurați-vă că aplicația nu se blochează;

- Testare de compatibilitate: testați aplicația pe diferite sisteme de operare (SO Windows 10/11 și Linux) și configurații hardware pentru a vă asigura că funcționează conform așteptărilor;
- Testarea performanței: verificați dacă aplicația funcționează bine și nu consumă prea multe resurse. Încercați să rulați aplicația pentru o perioadă lungă de timp pentru a vedea dacă se blochează sau încetinește;
- Testare de securitate: aici nu este ce sa testez, deoarece aplicația nu preia informație personala de la utilizator si nu are ieșire la internet, ea este rulata doar pe mașina locala;
- Testare de regresie: Efectuați testarea de regresie pentru a vă asigura că noile modificări sau funcții adăugate la aplicație nu încalcă funcționalitatea existentă. Cum am scris anterior, am adăugat funcționalități noi, iar aplicația merge cu succes împreună cu aceste funcționalități, prin urmare testul a fost efectuat cu succes.

FIŞA DE ACTIVITATE

Săptămâna IV

Sarcini planificate:

1. Realizarea prezentării

1. Realizarea prezentării

Slide-ul cu titlu prezentat în figura 1.1:

Figura 1.1 – Slide-ul cu titlu

Slide-urile cu scopul și obiectivele prezentat în figura 1.2 și figura 1.3:

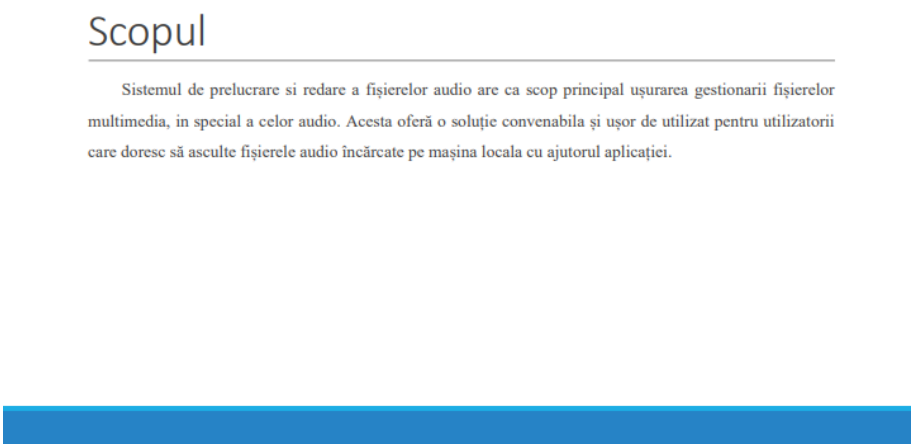

Figura 1.2 – Slide-ul cu scopul

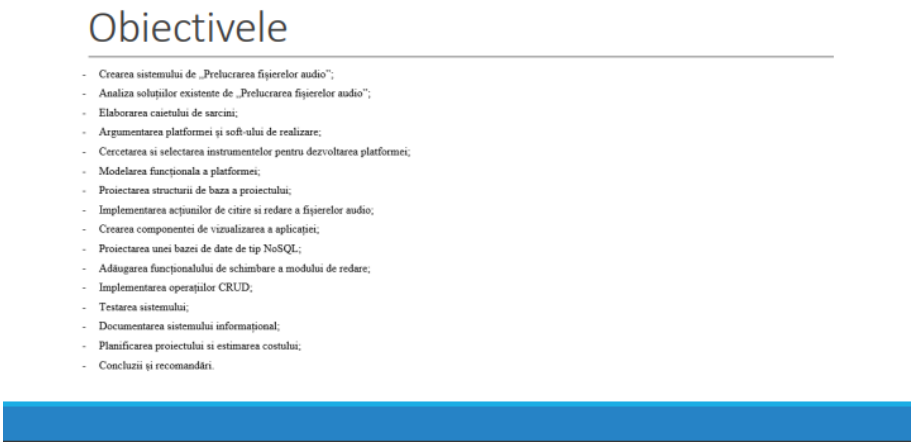

Figura 1.3 – Slide-ul cu obiectivele

Slide-ul cu analiza domeniului prezentat în figura 1.4:

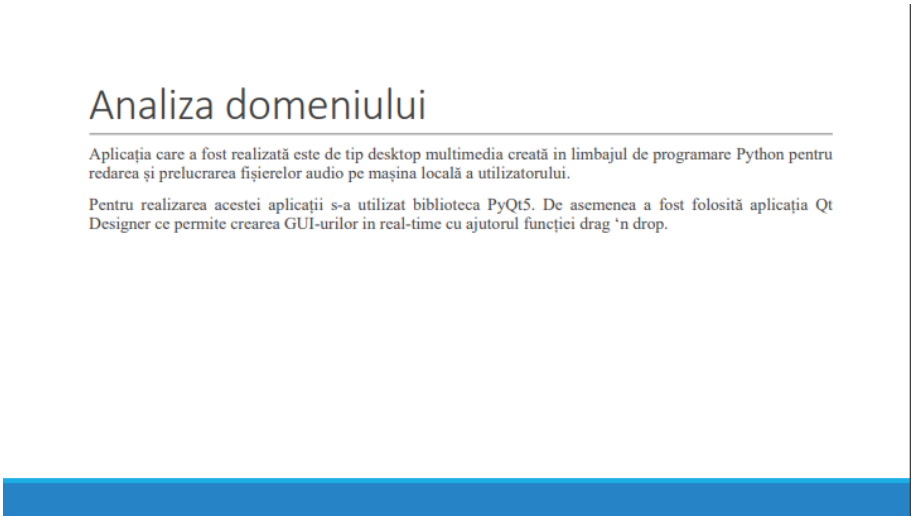

Figura 1.4 – Slide-ul cu analiza domeniului

Slide-ul cu compararea soluțiilor existente prezentat în figura 1.5:

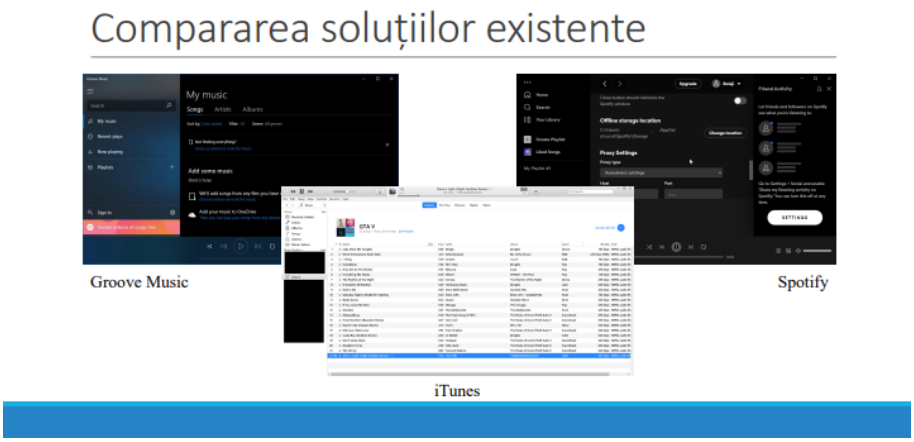

Figura 1.5 – Slide-ul cu compararea soluțiilor existente

Slide-urile cu UML-uri a sistemului prezentat în figura 1.6:

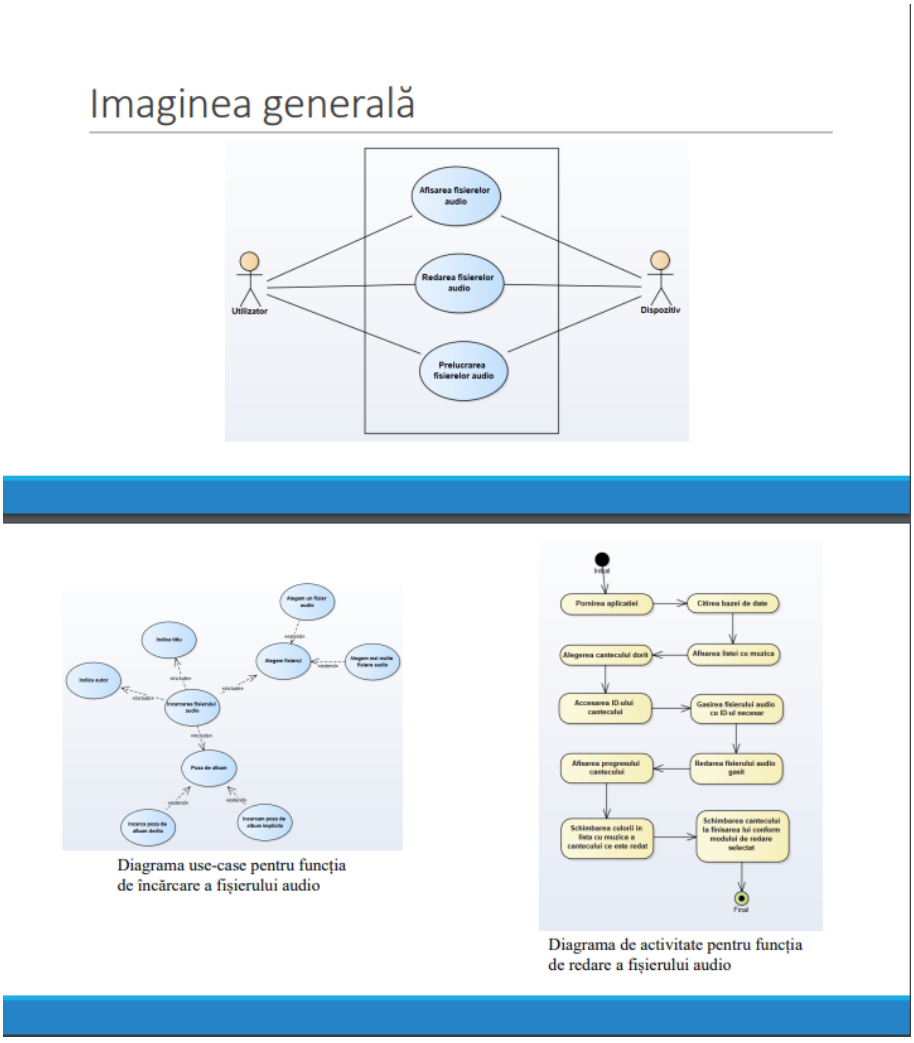

Figura 1.6 – Slide-ul cu imaginea generală s sistemului

Slide-urile cu structura proiectului prezentat în figura 1.7 și figura 1.8:

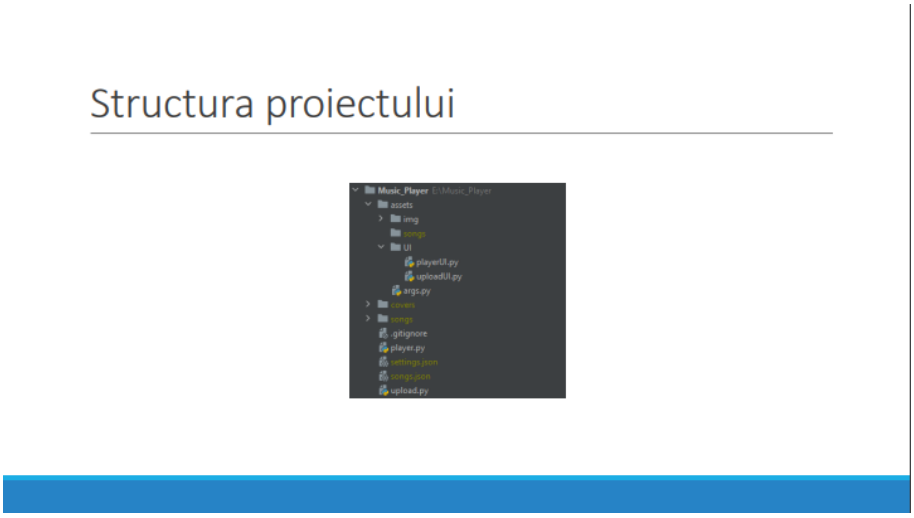

Figura 1.7 – Slide-ul cu structura proietului

Figura 1.8 – Slide-ul cu structura bazei de date

Slide-ul cu instrumente și tehnologii utilizate prezentat în figura 1.9:

Figura 1.1 – Slide-ul cu instrumente și tehnologii utilizate

Iar la final se află slide-ul cu demo prezentarea sistemului realizat.