ФМИ проект Музикална колекция

Към курс „Структури от данни и програмиране“

Документация

Мартин Йорданов Пенев ФН: 81990 Група 2

Специалност: Компютърни науки

1) Уводна част:

Проектът представлява реализирана система за музикална библиотека, която поддържа информация за потребители, песни и плейлисти. Програмата съхранява информацията си във файл с формат .***txt***, съответно прочита и записва посредством функции.

Основната цел на проекта е успешно да извлича и преобразува информация от документ и да я запазва в приложението по специфичен начин в структури от данни. Самото конзолно приложение поддържа множество функции, които могат да променят съответните данни и след извикването на определена функция, данните в документа също да се обновяват. Проектът би се харесал на множество потребители и любители на музиката, които чрез приложението ще успяват да създават свои плейлисти с любима музика, да я запазват, да оценяват песни и да се информират каква е любимата музика на техните приятели, които също използват приложението.

Структурата на документацията ще бъде следната:  
(1) Уводна част, в която се описва основната идея на проекта, за какво и по какъв начин биха го използвали хората. Съдържа също така кратко описание на основните функции и кратка информация за начина, по който приложението обработва своите данни и работи с файл. (2) Следва преглед на предметната област, в който се разглеждат подходите и методите при реализацията на съответната система, алгоритми и концепции. В третата част са разгледани конкретните структури от данни и подходи, използвани при създаването на приложението.(3) Накрая на документацията се представят определени тестове и самата реализация. Направено е заключение, в което се коментира възможност за бъдещо развитие на проекта и по какъв начин той може да бъде оптимизиран.

2) Преглед на предметната област:

Основният проблем, който трябва да бъде решен при реализацията на приложението е това да се подберат правилно структурите от данни, в които да се запазва информацията, съответно при затваряне да не се променя състоянието на програмата. При стартиране всички данни трябва да бъдат извлечени и запазени по същия начин, както са били преди затварянето. Проектът коректно се справя със следните функционалности:

* Създава регистрация след подаване на определени данни  
  (sign up)
* Показва всички направени регистрации до момента в системата  
  (show users) – работи и преди влизане в акаунт
* Показва всички песни, добавени до момента в системата  
  (show collection) – работи и преди влизане в акаунт
* Показва списък с всички функции, които се поддържат  
  (help)
* След успешен вход, чрез потребителско име и парола (sign in) е възможно:
  + Корекция на данни като добавяне на жанр в любими (add genre)
  + Премахване на жанр от любими (remove genre)
  + Добавяне на песен в системата (add song)
  + Генериране на плейлист по определени 4 критерия и максимален размер (generate playlist)
  + Плейлистът се записва под определено име при генерирането  
    (generate playlist)
  + Задаване на рейтинг на определена песен, като това е възможно само по веднъж от определен профил за определена песен   
    (rate song)
  + Представяне на информация за съдържанието на определен плейлист по име (show playlist)

Всички методи успешно и коректно обновяват информацията във файла с формат .***txt***. Използвани са следните класове:

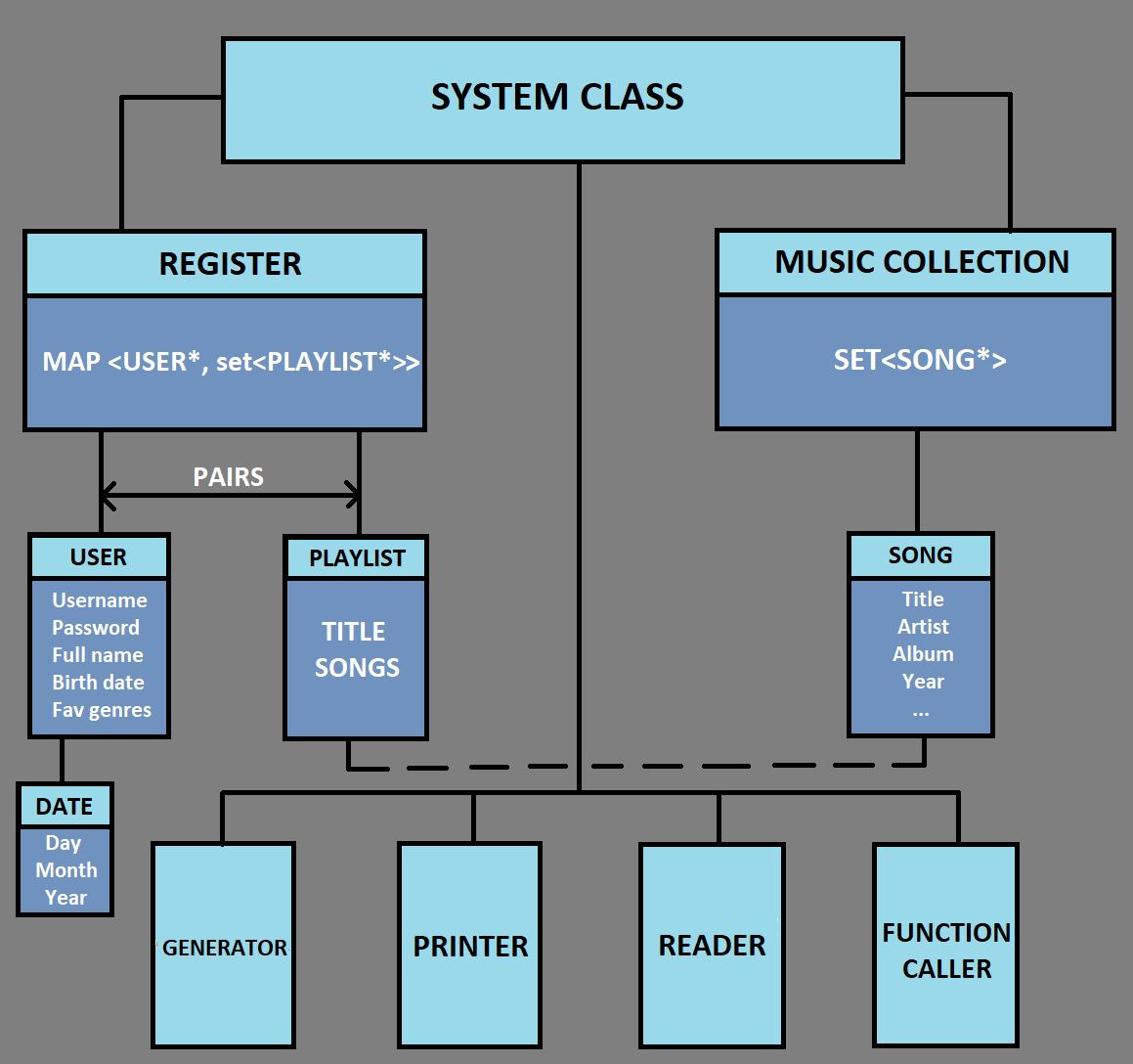
* User – поддържа информацията за потребителите като име, парола, пълно име, рождена дата и опашка от любими жанрове на съответния потребител
* Song – поддържа информацията за песните като име на песен, изпълнител, албум, година на издаване, жанр, рейтинг, списък с хора гласували за съответната песен и число, което е нужно за правилното изчисление на рейтинга след затваряне на програмата
* Playlist – поддържа информацията за плейлистите като име на плейлиста, списък от песни, които той съдържа
* System – основен клас поддържащ цялата информация на приложението и стартиращ програмата, чрез метода run()
* Printer – клас, изпълняващ ролята на конзолен интерфейс. Служи за отпечатването на по-големи текстове
* MusicCollection – един от по-важните класове. Служи за поддържането на списъка от всички песни в система чрез структурата std::set
* Register – клас съдържащ основната структура на приложението, която е std::map, в който се вкарват двойки от потребител и негов списък с плейлисти, съхранен в std::set
* FunctionCaller – класът, поддържащ функционалността на програмата. Служи за извикване на по-сложни функции след стартиране на програмата
* Reader – клас, поддържащ три основни функции, които служат за коректно прехвърляне на информацията от документа в програмата. Прочита съответно потребители, песни и плейлисти
* Generator – клас, който служи за генерирането на плейлист по максимален размер и определени критерии, като критериите, които програмата поддържа са:  
  + По-голям рейтинг от определена стойност, подадена от потребителя
  + Песни само от определен жанр, подаден от потребителя
  + Песни само от любимите жанрове на потребителя
  + Песни издадени преди или след година, подадена от потребителя
* Date – клас, който служи за правилното валидиране и запазване на рождената дата на потребителя

3) Структури от данни и подходи:

Основните структури от данни, които поддържат информацията на програмата са std::map, който служи за запазване на двойки (pair) от тип <User\*, set<Playlist>> и std::set, в който се поддържат данните за всички песни в системата. Структурите се обхождат чрез итератор. На всеки потребител в системата, съответно се съпоставя списък от неговите плейлисти. Тази структура (std::map) осигурява бърз и лесен достъп до потребителите и списъците им и наподобява телефонен указател, но вместо име и номер се запазват всички данни за потребителя в първата част на двойката (first) и всички данни за списъците от плейлисти във втората (second). За списъците от плейлисти е използван std::set. Други структури от данни, които са използвани са std::queue – опашка, в която се запазват любимите жанрове на съответния потребител, като обхождането на опашката се случва без нарушаване на структурата и.

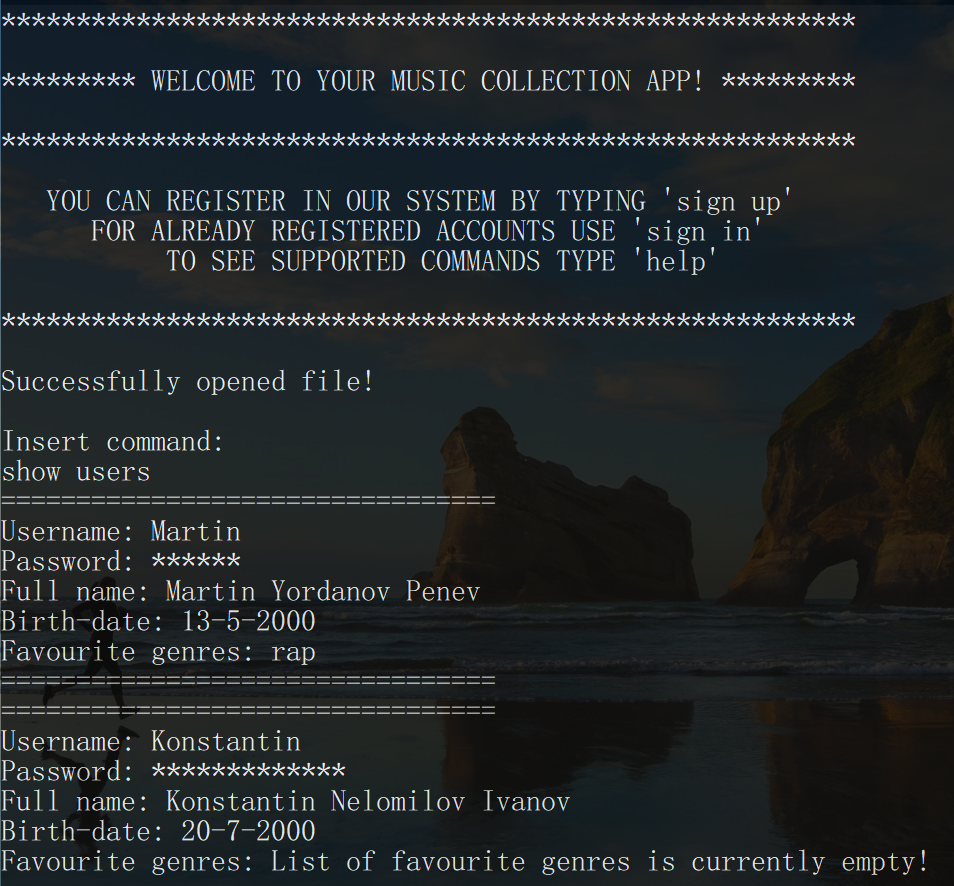
Плейлистите също поддържат информацията за своите песни в опашка, но приоритетна, за да могат да бъдат сортирани и при показване на съдържанието да излизат в съответен ред. Обхождането на опашката от песни в съответен плейлист също се случва без нарушаване на структурата. Използвани са също и вектори (std::vector) за реализацията на командния ред, четенето на думи от файл, съответно в класа Reader.

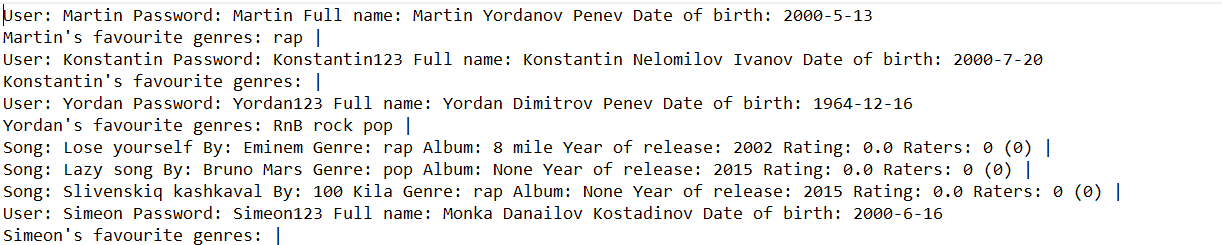
Готовите библиотеки, които са използвани са <iostream>, <vector>, <string>, <map>, <set>, <queue>, <sstream> и <fstream>, като <sstream> и по-точно std::istringstream е използван за реализирането на командния ред и преобразуването на стринг в число. Чрез <fstream> се четат и записват данните в документа. Използвани са също така и някои от функциите на std::string като erase(), find() и insert(), посредством които се обновява главният стринг file\_info, в който се запазва целия текст от файла.   
Методът set\_file\_end() e с голяма важност, тъй като той поставя звезда в края на текста всеки път, когато той бива обновен и по този начин при стартиране на програмата текстът се прочита винаги до звезда. При всяко обновяване на информацията във file\_info имаме съответно и set\_file\_end(), за изместване на края на текста.

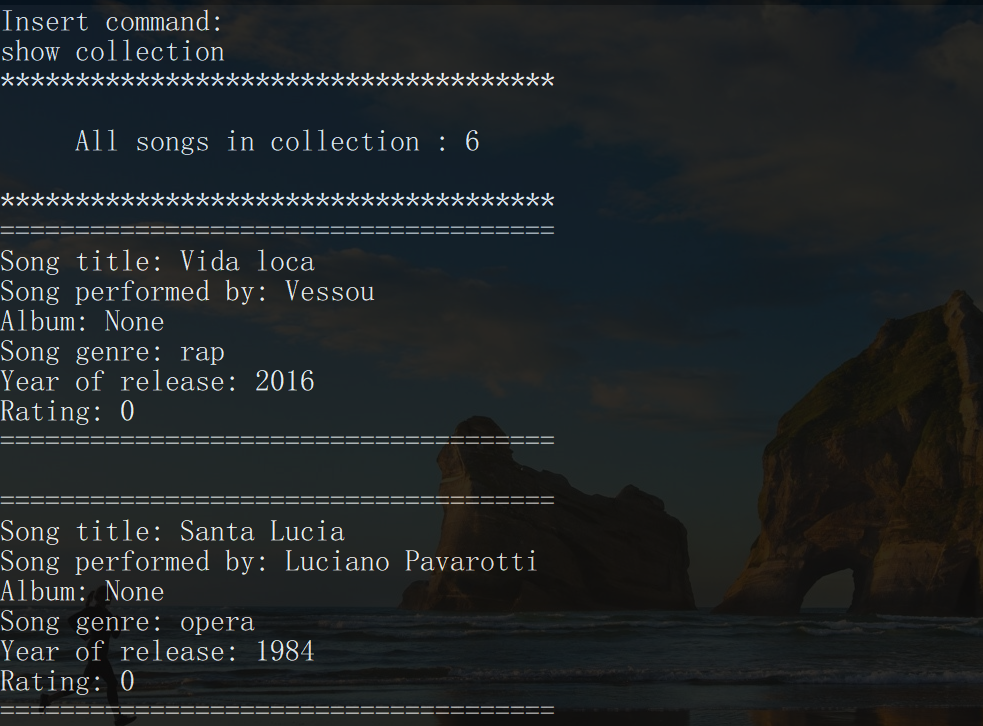
На следната схема е представено свързването на класовете в програмата:

4) Реализация, тестване:

В следният снимков материал е представена кратък тест на програмата и показва как програмата възприема данните от текстовия документ и разполага с тях:

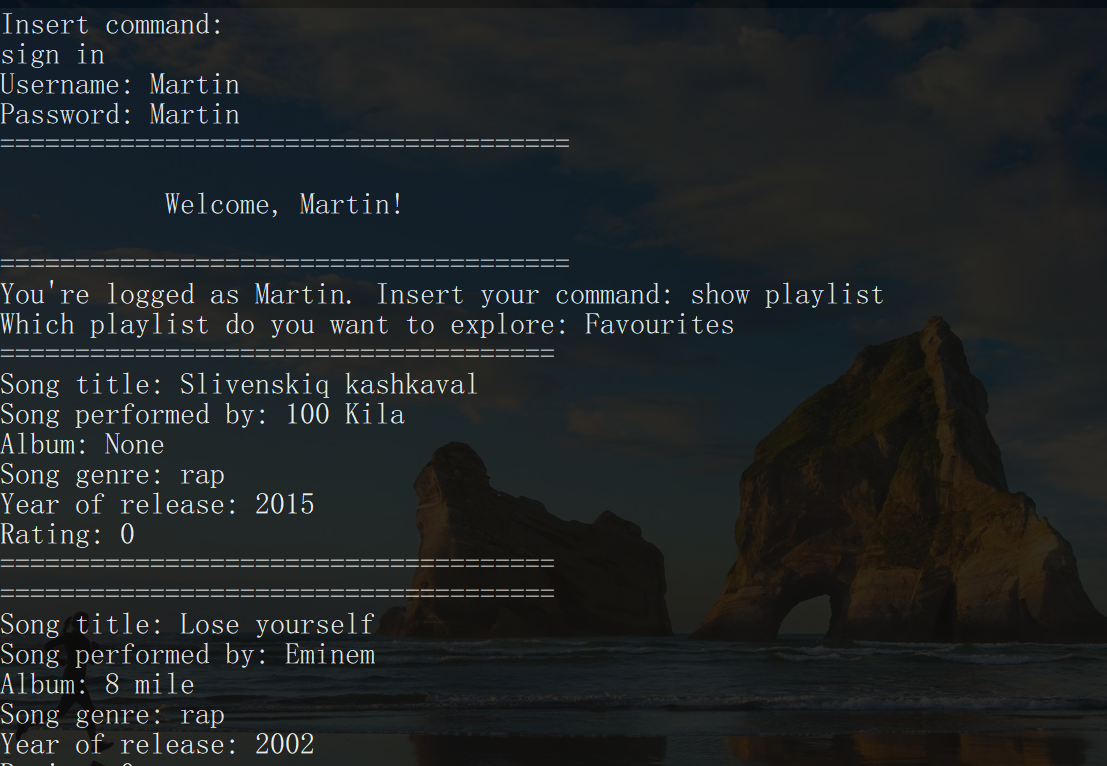
1. Стартиране на програмата. Информацията се прочита автоматично при успешно отваряне на файл. Съответно правим проверка чрез show users:

Всички потребители и информацията за тях е правилно извлечена в програмата, паролите са скрити.  
  
  
  
  
  
  
  
  
…

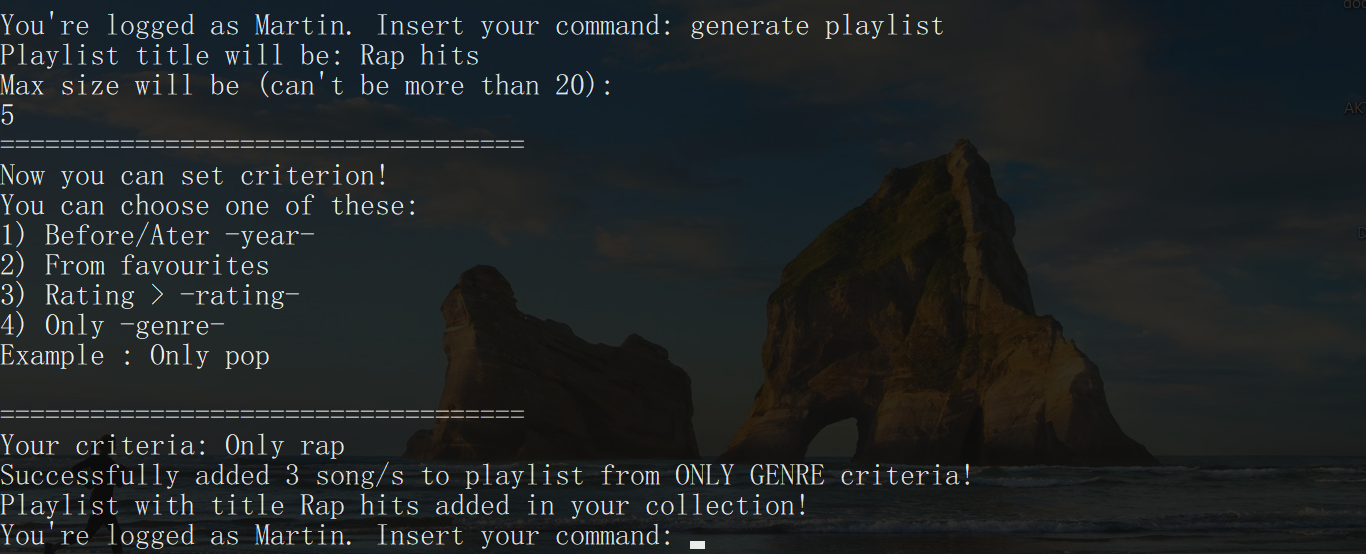
1. Разглеждаме списъкът от песни в програмата: (show collection)

…

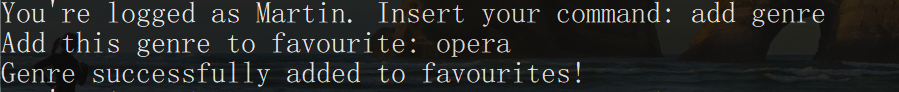
1. Влизаме в профилът на Martin съответно прочитаме неговия плейлист, записан под име Favourites: (sign in) (show playlist)

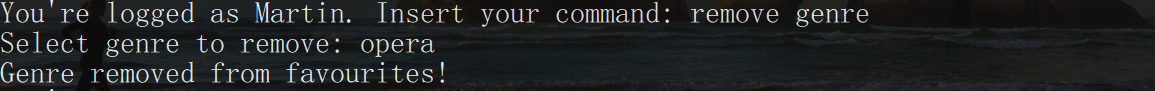


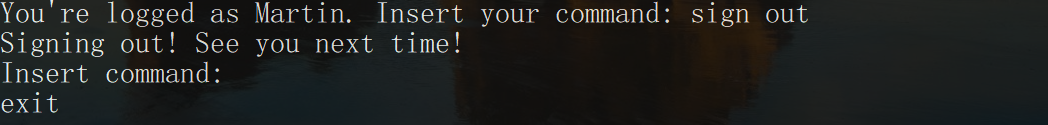
1. Генерираме плейлист по критерий Само рап песни (Only rap): (generate playlist)





Плейлистът се появява в края на файла със съответните си песни и капацитет.  
5. Добавяне и премахване на opera от любими жанрове:



  
6. Излизаме от профила, в който сме влезли и затваряме приложението.  
(sign out) (exit)

Файлът се затваря и програмата също!

ЗАКЛЮЧЕНИЕ: Проектът „Музикална колекция“ представлява конзолно приложение, подходящо за всеки любител на музиката, който желае да запазва своите песни, да създава свои плейлисти и да се информира за най-харесваните и оценявани песни до момента. Реализацията на програмата е съобразена с обектно-ореантираните подходи и подходите при използване на структури от данни. Като предложение за бъдеща реализация може да се посочи метод play\_song(), който да зарежда определена песен по име в паметта и да стартира съответната песен чрез отваряне на линк в Youtube.  
Линк към Гитхъб: <https://github.com/mjpenev/Music-Collection-Project-FINISHED>