**Проект №10 – Библиотека. Никола Терзийски Ф№72006**

Проектът е базиран на реална библиотека, в която може да се регистрират хора, да се добавят книги, да се извежда информация за дадена книга и др. Идеята му е да симулира една нормално работеща система, коята се използва в библиотеките.

Документацията представлява описание на проекта в работеща среда, функциите които съм използвал и проблемите, с които съм се срещнал.

Както се вижда – основната ми структура в проекта е **Книга(Book)** и **Потребител(User)**. Книгата може да бъде много типове, спрямо жанра и. Използвал съм класове за инициализирането им, като данните на книгите и съответно на потребителите, биват генерирани от отворения файл, който е отворен предварително. Програмата не може да започне преди да бъде отворен файл. Проекта не е лесен, защото изисква знания в почти всяка област. Изключително много използване на файлове и реализация на класове.

В тази документация ще видите (на теория) как съм реализирал моя от началото до края.

-**Стартова точка**

0.Както навсякъде, има наръчник, с който може да разберете как работи дадена команда.

Това става като се напише командата *help*

При невалидна команда излиза съобщение, с което разбирате, че въведената от вас команда е невалидна.

Един потребител може да пази създадените данни по време на работа в същия файл с командата **save** или чрез **saveas <path>** по даден път.

1.Старта на програмата започва с отварянето на даден файл чрез командата *open <path>.* Ако се въведе невалиден път, то програмата изкарва грешка, с която ви казва, че пътят е невалиден.

2. След като въведете правилен път и бъде отворен файла, автоматично се генерира **Сесия(Session),** в която съм вкарал почти цялата си функционалност за проекта. Идеята ми е, да има едно място, което е лесно и достъпно и да върши функционалността. В конструктора на Сесията биват генерирани потребителите и всичките книги, които се намират в дадения файл. Това съответно става, чрез използването на *namespace* ***data,*** който се намира в класа **TextData** ***.*** Това е мястото където се генерират всички потребители. Винаги има един потребител, който е admin.

За моя помощ съм си реализирал собствени **String** и **Vector**

Admin-потребителя е само един. Има доста различни функции от тези на обикновения потребител .

При отварянето на файла, ще бъде генериран на конзолата текст, който текст обяснява как работи програма и къде трябва да бъдете влезли в акаунта си, за да може да правите дадени операции.

**Users**

2.1 При LogIn функцията, може да бъдете логнати, ако сте регистриран в системата или по-точно записан във файла users.txt, който пази всички пароли и потребителски имена.

Когато използвате дадената функция, в сесията бива генериран *User,* който пази човека, който се е логнал в програмата

2.2 Ако искате да бъдете регистриран това може да стане само чрез администратора, който той трябва да влезне и да ви регистрира чрез командата *users add*

2.3 Ако искате да излезнете от системата, трябва да напишете logout командата, която ще ви махне и съответно ще ви изтрие от сесията.

Админът също така може да изтрива хора от системата.

**Books**

Книгите също могат да бъдат махана и добавяни, но не само това. Както и при потребителите, за да бъде добавена една книга или махната, потребителя, който извършва действието трябва да бъде администратор. Ако е обикновен потребител той може да върши останалите функции като: сортиране, търсене спрямо ID, изкарване на всички книги, и преглеждане на дадена книга спрямо критерий.

1. При добавяне на книгите, всичко компоненти на книгата са задължителни, за което са реализирани допълнително проверки – дали полето е празно или съответно за оценката – дали оценката е < 0 или > 10.

За генерирането на ID на дадена книга се използва неймспейса – data, който използва помощен файл id.txt, който всеки път генерира уникално число(текущото id = миналото id + 1).

2.При махането на книгата, тя бива махана спрямо дадено заглавие. Ако на една книга заглавието съвпада с това на книгата, която трябва да бъде махната, тя бива изтрита

3.При използването на другите функции има допълнителни компоненти за входа .

3.1 Функцията sort изисква начин, по който да се сортират данните, като опции са : автор, заглавие, година и рейтинг

3.2 Функцията all просто изкарва всички книги

3.3 Функцията find намира книги по дадени критерии.

-Критериите може да са: автор, заглавие, таг..

3.4 Последната функция books info – дава информация за дадена книга спрямо нейното ID.

-Има проверки за входовете на функциите дали дадените ID, критерии и т.н. са празни входове или невалидни стойности.

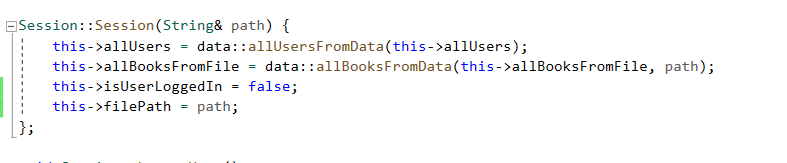
-Съответно съм направил и проверки дали дадените данни за книга или човек съществуват. В клас **Validator** с подаване на данни като първо е критерия или името на потребителя и после вектор от книги или съответно вектор от потребители.

Функциите find, sort, info, all се извършват само върху книгите, които са запазени във файла. Ако преди това сте добавили книга, и искате да я намерите чрез функцията info, тя няма да се появи, а ще изпише, че такава книга не съществува, защото още реално не е създадено във файла. Нейното създаване още не е приключило.

При изтриването е обратно, понеже я трием, тя ще се изтрие от текущите книги, но от файла няма да се изтрие все още, но функциите find, sort, info няма да взимат изтритата

**Реализация на класове**

При създаването на Сесия, генерирането на книгите и хората изглежда:



path – подаваме адреса на подадения път в main.cpp, където е даден адреса на файла с книгите, който да бъде отворен.

Има поле filePath където пази през цялото време на програмата подадения път.

-При създаването на книга тя първоначално има ID = 0. Равно е на 0, защото при проверките за info ние имаме първоначална книга, която е празна и съответно сравняваме подаденото Id на функцията, която търси еквивалентно Id от тези на запазените книги.

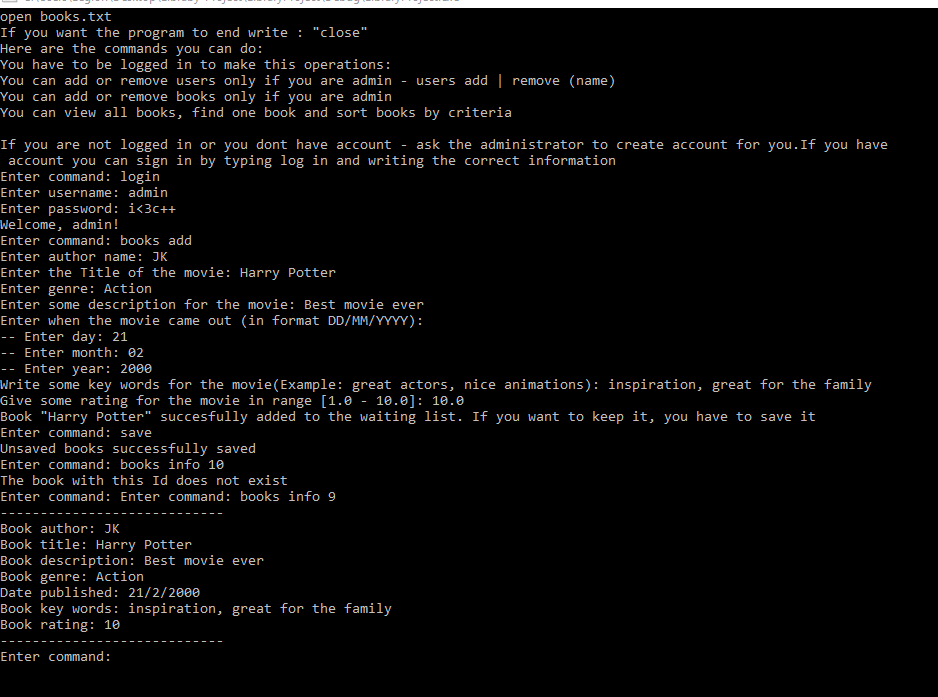
-И сега идва съществената част защо е равна 0. Защото, ако не намери книга със съответсващо Id, проверката дали findBook == 0 ще даде true и ще изкара съобщение, че няма книга с такова Id.

-Имаме и поле, което е isUserLoggedIn за по-бърза проверка. Реално при повечето функции, където изисква влизане в акаунта, просто ако това поле е false, ще изкара грешка, че потребителят не е влязъл в акаунта си.

**Алгоритми**

Използвал съм метода не мехурчето за сортиране за функцията **books sort** <criteria> <ascending,descending>. За да може да бъде реализиран метода, трябва да напиша операторите ‘<’ и ‘>’ за String.

**Примери**



1.Първоначално се отваряме файла.

2.После ни излиза информацията с това как работят командите за отворен файл.

3.Трябва да се логнеме, за да може да добавим книга

4. Пишем командата **books add** и добавяме книга -> Излизат ни полета за въвеждане на данни.

5. Опитваме се да намерим книгата, която добавихме, преди да сме я запазили във файла => грешка, че такава книга не съществува(все още). След това пишем **save** и книгата ни се добавя във файла.

6. Пишем командата отново **books info <id>,** след като сме добавили книгата => излиза ни информация за книгата.

**Излизане**

Програмата може да бъде спряна:

1. Ако сме отволи файл, първо трябва да напишем close, за да излезнем от файла.
2. После трябва да напишем exit, за да спрем изпълниението на програмата.

**Бъдещи цели**

Програма може да се усъвършенства изключително много.

1. Може да се създаде файл, който е от библиотеки и да се създаде още едно Id на книгата , което съответства на ID-то на библиотеката.
2. Може да се направи и цени на книгите и те да биват продавани.
3. Да се направи поле да книгата в библиотеката isTaken и един метод TakeBook(), който взима книги от библиотеката от потребител. Администратора може да отчита дали един човек е върнал книгата в срок, даден му от библиотеката, който може да бъде свързан с часа на взимане на книгата. Потребителят може да бъде глобяван или гонен от библиотеката, ако не е върнал книгата си на време
4. Да се създадат секции – чуждестранни книги и български книги.
5. В библиотеката може да се направят и допълнителни полета за книги(ленти)

Github папка на проекта - <https://github.com/nikolaTerziiski/Libraby-Project/tree/master/LibraryProject>