Проект 10: Библиотека

Даниел Петров, ФН 72091
GitHub хранилище: https://github.com/danielgp1/Project-10---Library

➢ *Описание на проекта*

Проектът „Библиотека“ реализира информационна база данни, имаща за цел да се занимава със съхранението и доставянето на нужната информация за съответните книги и потребители, съдържащи се в системата на дадена библиотека.

➢ *Функционалности*

За да функционира пълноценно, информационната система може да:

- зарежда нужната информация от текстов файл
- извършва редица от команди, след проверка на статуса на потребителя, извършващ съответните промени

команда	изисква ли потребител?	само за администратор?
open	не	не
close	не	не
save	не	не
saveas	не	не
help	не	не
login	не	не
logout	да	не
exit	не	не
books all	да	не
books find	да	не
books sort	да	не
books view	да	не
books add	да	да
books remove	да	да
users add	да	да
users remove	да	да

- обновява своята база данни, презаписвайки вече отвореният текстов файл, или да създаде нов такъв на желаната директория

➢ *Конструиране*

За осъществяването на проекта са използвани следните класове

- Собствено написани помощни класове String и Vector
- Базови класове

class User

```
public:
	User();
	User(String&, String&);

	void setPassword(const String&);   |
	void setName(const String&);       | присвояват въведената стойност
	void setLoggedIn(const bool);      | на съответната член данна
	void setAdmin(const bool);         |


	String getName() const;            |
	String getPassword() const;        | извеждат съответната член данна
	bool getAdmin() const;             |
	bool getLoggedIn() const;          |

	void loadUser(std::ifstream&); - зарежда нужната информация от файл
	void saveUser(std::ostream&) const; - записва информацията на файл

	bool operator==(const User&); - сравнява имената на двама потребители
private:
	String name;
	String password;
	bool isAdmin;
	bool isLoggedIn;
```

class Book

```
public:
	Book();

	void setTitle(const String&);            |
	void setAuthor(const String&);           |
	void setGenre(const String&);            |
	void setSummary(const String&);          | присвояват въведената стойност
	void setYear(const size_t);              | на съответната член данна
	void setTags(const Vector<String>&);     |
	void setRating(const double);            |
	void setId(const size_t);                |

	void printBriefly() const; - принтира кратка информация за книгата
	void printDetailed() const; - принтира цялостна информация за книгата

	void loadBook(std::istream&); - зарежда нужната информация от файл
	void saveBook(std::ostream&) const; - записва информацията на файл

	size_t getID() const;
	const String& getTitle() const;          |
	const String& getAuthor() const;         |
	const Vector<String>& getTags() const;   | извеждат съответната член данна
	const size_t& getYear() const;           |
	const double& getRating() const;         |
```

```
private:
	String title;
	String author;
	String genre;
	String summary;
	Vector<String> tags;
	size_t year;
	double rating;
	size_t id;
```

- Хранилища за потребители и книги

class UserStore

```
public:
	UserStore();

	const size_t getLinesOfFile(std::ifstream&) const; - извежда броят на редовете във файла
	size_t getSize() const; - извежда броя на потребителите в хранилището

	void addUser(const String&,const String&); - добавя нов потребител
	void removeUser(const String&); - премахва избрания потребител
	void clear(); - премахва всички потребители

	User& operator[](const size_t) const; - извежда потребителя на съответния индекс
	int activeUserIndex() const; - извежда индексът на активния потребител

	void loadUsers(std::ifstream& in); - зарежда нужната информация от файл
	void saveUsers(std::ofstream& out); - записва информацията на файл
private:
	Vector<User> users;
	size_t total;
```

class BookStore

```
public:
	enum class option { TITLE, AUTHOR, YEAR, RATING }; - критерии за сортиране
	BookStore();

	size_t getSize() const; - извежда броя на книгите в хранилището

	void swap(Book&, Book&); - разменя мястото на две книги
	void allBooksBrieflyInfo() const; - принтира кратка информация за всички книги
	void bookDetailedInfo(const size_t) const;-принтира подробна информация за всички книги
	void findBook(const String&,const String&) const; - извежда ID на търсената книга
	void sortBooksAscending(const String&); - сортира книгите възходящо по желан критерий
	void sortBooksDescending(const String&); - сортира книгите низходящо по желан критерий

	void addBook(); - добавя нова книга
	void removeBook(const String&); - премахва избраната книга
	void clear(); - премахва всички книги

	void loadBooks(std::ifstream&,size_t); - зарежда нужната информация от файл
	void saveBooks(std::ofstream&); - записва информацията на файл
private:
	Vector<Book> books;
	size_t total;
	size_t id;
```

- Цялостна библиотека

class Library

```
public:
	Library();
	void clear(); - изчиства всички данни

	void open(const String&); - зарежда въведения файл
	void save(); - записва направените промени
	void saveAs(const String&); - записва направените промени в нов файл
	void close(); - затваря отворената база данни

	void logIn(); - задава активен потребител
	void logOut(); - отписва активния потребител
	void exit(); - излиза от библиотеката
	void help() const; - извежда списък с наличните команди

	void usersAdd(const String&, const String&);
	void usersRemove(const String&);

	void booksAll() const;
	void booksInfo(const size_t) const;
	void booksFind(const String&,const String&) const;
	void booksSortAscending(const String&);
	void booksSortDescending(const String&);
	void booksAdd();
	void booksRemove(const String&);
private:
	UserStore users;
	BookStore books;
	bool isOpened;
	bool isSaved;
	bool isChanged;
	String FileName;
```

- Команден интерпретатор

class CommandMenu

```
public:
	CommandMenu();

	void startLibrary(); - изпълнява желаните команди до въвеждане на exit
	void getCommand(); - отделя командата от въведения низ
	void checkCommand(const String&,const String&,const Vector<String>&); - проверява команда
	void checkSecondCommand(const String&); - изчиства ненужни части от командата
	void getParameters(); - отделя параметрите от въведения низ
private:
	Vector<String> allCommands;
	Vector<String> theCommand;
	Vector<String> parameters;
	String command;
```

- ***Идеи за развитие на проекта***

  - добавяне на възможността за взимане на книга
  - поддържане на база от данни за потребители и книгите, които трябва да върнат
  - показване на свободните книги
  - добавяне на наличните копия на дадена книга
  - информация кога даден член на персонала е влизал и оперирал със системата
  - списък с целия персонал