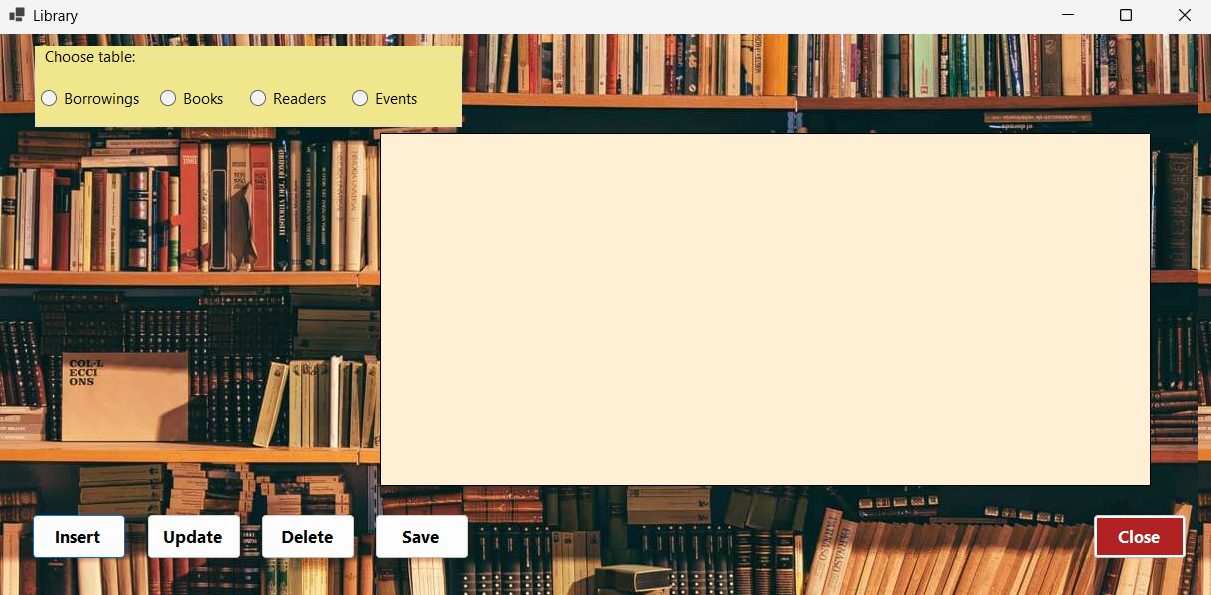


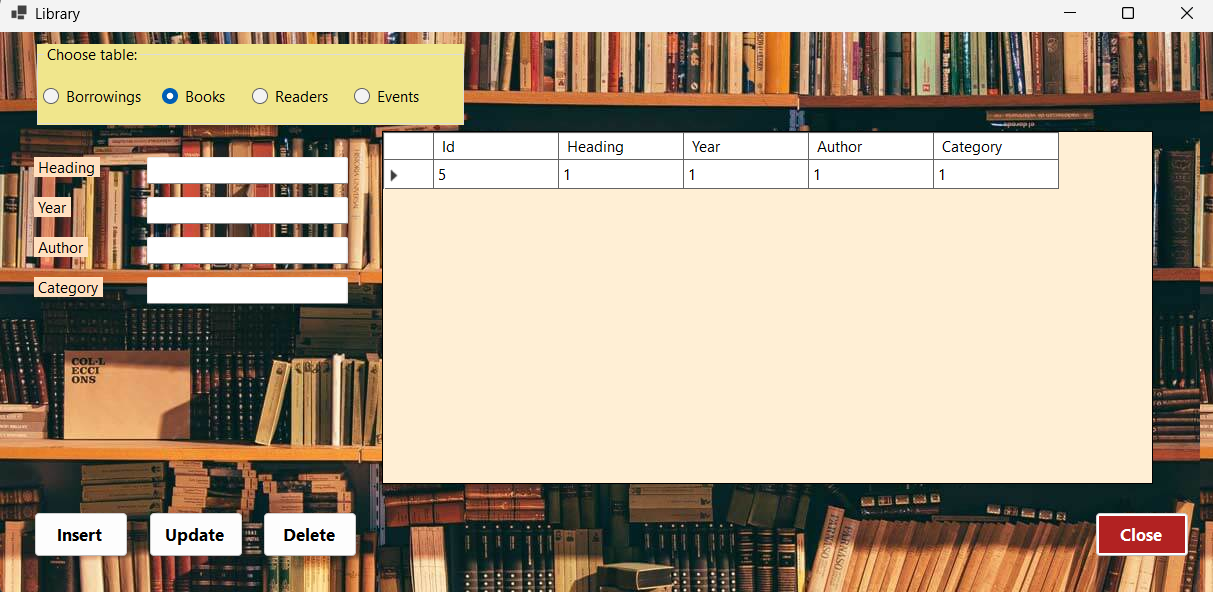
Документация за курсов проект

**Библиотека**

Изготвили: Кристиана Стоянова и Виктор Иванов



При стартиране на приложението се отваря WindowsForms проект(**View**), изграден от четири радиобутона, отговарящи за всяка една от таблиците в проекта. Съдържа Data Grid View, който по-късно ще показва данните от таблиците. Пет бутона, които ще изпълняват фукнциите на нашия проект: Бутонът **Insert** ще добавя написаното в дадена таблица, **Update** ще позволи редактирането на вече съществуващ ред в дадена таблица, **Delete** ще изтрива избран ред на таблицата, а **Save** ще запазва промените след ъпдейта. Бутонът **Close** затваря приложението.

След като се избере таблица, в която да се работи(в случая Books), се показват съответен брой етикети и текстови кутии, съответстващи на колоните в базата данни:  


Нашата база данни има четири таблици: Borrowings, Books, Readers, Events.

Borrowings съдържа Reader ID, Book ID, Date, Date Return, Status.

Books съдържа Heading, Year, Author, Category.

Readers съдържа Name, Phone Number, Email, Date Registration.

Events съдържа Name, Date, Description.

//опиши form1

Всяка таблица в базата данни представлява клас в програмата, чиито параметри са колоните и. Те са описани в проекта Data(**Model**) в папката **Model**. В този проект са и DbContexts, като за всяка таблица има отделен контекст, наследен от DbContext-основният клас в Entity Framework за работа с бази данни. На пример в таблицата Books: Конструкторът BookContext(DbContextOptions<BookContext> options) получава настройки за връзка към базата (например връзков низ) и ги предава към базовия конструктор. DbSet<Book> Books представлява таблица с книги в базата. Чрез това свойство може да се правят заявки (четене, добавяне, изтриване, промяна на записи). Като в data са инсталирани:

<PackageReference Include="Microsoft.EntityFrameworkCore" Version="9.0.4" />

<PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="9.0.4" />

За **controller** е използван проекта **Business.** В него се съдържат четири класа, съдържащи функционалността на бутоните от View. Инсталиран е пакет <PackageReference Include="Microsoft.EntityFrameworkCore" Version="9.0.4" /> и е създадена връзка към проекта Data.

**BookBusiness-**  управлява операции върху таблицата Books в базата данни.

private readonly BookContext bookContext; създава обект bookContext, чрез който работим с базата.

Конструктор: DbContextOptions<BookContext> (настройки за връзка към базата) и създава нов BookContext. – *аналогично за всички таблици*

Методи:

GetBooks() — Връща всички книги от базата като списък.

GetBookById(int id) — Търси книга по ID. Ако не я намери — хвърля грешка "Book not found!".

AddBook(Book book) — Добавя нова книга в базата и записва промените.

DeleteBook(int id) — Намира книга по ID и я изтрива. Ако книгата е заета (примерно свързана с други данни), хвърля специфична грешка "Book is borrowed".

UpdateBook(Book book) — Намира съществуваща книга по ID и обновява нейните данни (автор, категория, година и заглавие).

**BorrowingBusiness**- отговаря за работа с **наеманията (Borrowings)** на книги.

Методите:

GetAll()- Връща списък с всички наемания (Borrowing) от базата.

GetById(int id)- Търси конкретно наемане по ID. Ако не намери — хвърля грешка "Borrowing not found".

Add(Borrowing borrowing)- Добавя нов запис за наемане в базата и записва промените.

Update(Borrowing borrowing)- Намира съществуващо наемане по ID, Обновява всички негови полета: читател (ReaderId), книга (BookId), дата на заемане (Date), дата на връщане (DateReturn) и статус (Status). При грешка при запис (например невалиден Reader или Book ID) — хвърля специфична DbUpdateException с по-добро съобщение.

Delete(int id)- Намира наемането по ID и го изтрива от базата. След това записва промените.

**EventBusiness**- управлява събитията (Events) в базата данни.

Методите:

GetAll()- Връща всички събития от базата като списък (List<Event>).

GetById(int id)- Търси събитие по ID. Ако няма такова — хвърля Exception с текст "Event not found".

Add(Event eventLibrary)- Добавя ново събитие и записва промяната в базата.

Update(Event eventLibrary)- Намира съществуващо събитие по ID. Обновява неговите полета: име (Name), дата (Date) и описание (Description). Записва промените.

Delete(int id)- Намира събитие по ID. Изтрива го от базата и записва промените.

**ReaderBusiness** управлява читателите в базата данни

Методите:

GetReaders()- Връща списък с всички читатели (Reader) от базата.

GetReaderById(int id)- Намира конкретен читател по ID. Ако няма такъв — хвърля грешка "Reader not found".

AddReader(Reader reader)- Добавя нов читател и записва промените в базата.

DeleteReader(int id)- Намира читател по ID. Изтрива читателя от базата. Ако не може да го изтрие (например защото има заети книги), хвърля DbUpdateException с текст "Reader has borrowed books".

UpdateReader(Reader reader)- Намира съществуващ читател по ID. Обновява неговите данни: име (Name), имейл (Email), телефон (PhoneNumber) и дата на регистрация (DateRegistration). Записва промените.