**Library Management System**

Projekt Library Management System je program, ktorý uľahčuje prácu zamestnancov alebo správcov knižnice pracovať s knihami a DVD-čkami, taktiež s klientami a pomocou tejto aplikácie mať tak prehľad o všetkom, čo sa v knižnici deje.

Aplikácia je vytvorená pre potreby zamestnancov alebo správcov knižnice, nie pre klientov, ktorý si z knižnice niečo požičiavajú.

Celá aplikácia beží na troch projektoch, ktoré medzi sebou komunikujú pomocou **DependencyInjection.** Konzolová aplikácia stojí na troch projektoch:

* Library.Core – komunikačná logika medzi Infrastructure a UI
* Library.Infrastructure – Entity Framework databáza Code first princíp
* Library.UI – dizajn aplikácie a všetkých stránok

**Library.UI**

Keď spustíme aplikáciu, zobrazí sa nám toto Menu:

Obrázok, na ktorom je text

Automaticky generovaný popis

**Base:** hlavné menu je naprogramované v Library.UI > Pages > MainPage. Každé jedno menu je dedené od vedľajšej triedy MenuPageBase, ktorá sa ešte dedí od PageBase (obidve sa nachádzajú v priečinku v ceste Library.UI > Base)

**Helpers:** v tomto priečinku máme dve triedy ako pomôcky. V InputHelper.cs máme zadefinované funkcie na lepšie konvertovanie dát od užívateľa. OutputHelper obsahuje funckiu na farebnú škálovosť textov.

**Pages:** všetok vizuál je naprogramovaný v triedach a ďalších podpriečinkoch. Vizuál alebo teda dizajn, čo sa týka stránok v aplikácii.

**UIElements**: tu sú taktiež dve pomôcky na zobrazenie menu a možností výberu v rozličných stránkach, ktoré sú už potom samostatne prispôsobené.

Všetky vyššie spomenuté priečinky a ich obsah a následne obsah všetkých projektov sú spojené v triede Application.cs, kde sa zohráva celá logika. Následne trieda Application je spustená pomocou metódy Run(), ktorá sa spustí v Program.cs súbore.

**Library.Infrastructure**

V tomto projekte sa nachádza viac-menej Entity framework core, čo slúži na vytvorenie databázy pomocou Code-first princípu. Projekt obsahuje dva priečinky a to:

**Data:** v ňom sa nachádza priečinok s názvom **Repositories,** kde pre každú jednu triedu je pomocou LibraryContext.cs vytvorená databáza pre jednotlivé veci v knižnii ako sú napríklad: knihy, dvd, užívatelia, objednávky,...

* LibraryContext.cs – komunikácia s databázou s už spomínaným Entity Frameworkom.

**Migrations:** migrácie v EF Core poskytuje spôsob, ako postupne aktualizovať databázovú schému, aby bola synchronizovaná s dátovým modelom aplikácie pri zachovaní existujúcich údajov v databáze.

Ďalej tento projekt obsahuje InfrastructureModel.cs, ktorý funguje na princípe dependenfcy injection v preklade vkladanie závislostí medzi Library.Core.Abstraction.Repositories resp. konkrétnych repo interfacoch v tomto priečinku a Library.Infrastructure.Data.Repositories resp. konrétnych repo triedach v tomto priečinku.

**Library.Core**

**Abstractions:** nachádzajú sa tu dva podpriečinky a to **Repositories** a **Services,** kde sú hlavné interfaci (rozhrania), z ktorých sa dedia metódy do ostatných súboroch, v ktorých sa vykonávajú CRUD (create, read, update, delete) funkcie.

**Base:** abstraktná trieda EntityBase.cs, ktorá obsahuje field s názvom Id, ktoré dedí jedna trieda Title.cs v **Entities** od ktorej zase dedia rôzné entity v danom priečinku

**Entities:** triedy obsahujúce potrebnú implementáciu dát do databázy, napr. čo všetko potrebujeme vedieť o konkrétnej knihe, DVD-čku, užívateľovi, objednávky, atď...

**Enums:** ďalej tu máme priečinok obsahujúci dva enumy –

* eTitleCountUpdate – slúži ako definovanie akcie v repositároch, či už pridávanie alebo mazanie
* eTitleType – slúži na identifikáciu titulu, buď to je dvd alebo kniha

**Events:** obsahuje event TitleReturnedEventArgs na sledovanie, či bol titul (dvd alebo kniha) vrátený

**Services:** obsahuje triedy resp. servicy pre manažovanie správ pre užívateľov, manažovanie poradovníka a manažovanie objednávok

**Settings:** tu sú definové nastavenia pre následné vypočítanie poplatku za nevrátenie titulu a kontrola výpožičnej doby

Projekt Library.Core ešte obsahuje CoreModule.cs, ktorý pomocou Dependency Injection vkladá závislosti alebo teda prepája Library.Core.Abstractions a Library.Core.Services, resp. príslušné triedy a interfacy v týchto dvoch priečinkoch