BookKeeper

Technická specifikace

Tento dokument popisuje technické aspekty projektu BookKeeper. BookKeeper je projekt implementující daňovou evidenci (dříve jednoduché účetnictví). Zadání je mírně zjednodušené s ohledem na možnosti a znalosti studentů a rozsah projektu (cca 1 měsíc).

Výchozí stav projektu je na GitHubu:

<https://github.com/sssvt-foltyn/bookkeeper-go>

Použité technologie:

Databáze: MySQL – zdrojový kód v SQL

BE: ASP.NET Core API (webová aplikace) – zdrojový kód v C#

FE: Angular (webová aplikace) – zdrojový kód v HTML, CSS, JavaScript, TypeScript

# Obsah

1. Databáze
   1. Tabulky
2. Datový model
3. Back-endová část
   1. BookKeeperBECommon
   2. BookKeeperBEMain
   3. BookKeeperBERest
4. Front-endová část
   1. book-keeper-fe-angular

# Databáze

Projekt BookKeeper používá následující tabulky. Jejich podrobný popis včetně všech sloupců je v dokumentu **BookKeeper\_DatabaseDesign\_v....docx**.

## Tabulky

**BK\_USER**

Uživatelé oprávnění k přihlášení do aplikace.

**BK\_CONTACT**

Kontakty, které váš byznys používá pro finanční styk – odchozí platby, příchozí platby. Zákazníci, dodavatelé, úřady, organizace apod.

**BK\_INVOICE**

Faktury (hlavičky faktur). Vydané zákazníkům za to, že jsme jim prodali naše výrobky nebo služby. Přijaté od dodavatelů za nákup jejich výrobků nebo služeb pro náš byznys.

**BK\_INVOICE\_ITEM**

Položky faktury.

**BK\_STATEMENT**

Bankovní výpisy (hlavičky výpisů) z účtu, který souvisí s naším byznysem. Slouží pro přehled peněžních prostředků v bance.

**BK\_STATEMENT\_ITEM**

Položky bankovního výpisu.

**BK\_RECEIPT**

Hotovostní doklady (hlavičky dokladů). Týkají se pohybů hotovosti v naší fyzické pokladně. Hotovostní doklad *přijatý* (účtenka z nějakého obchodu) = výdej peněz z fyzické pokladny. Hotovostní doklad *vydaný* (vydáváme za naši firmu, stvrzenka, že nám někdo něco zaplatil) = příjem peněz do fyzické pokladny.

**BK\_RECEIPT\_ITEM**

Položky hotovostního dokladu.

**BK\_JOURNAL**

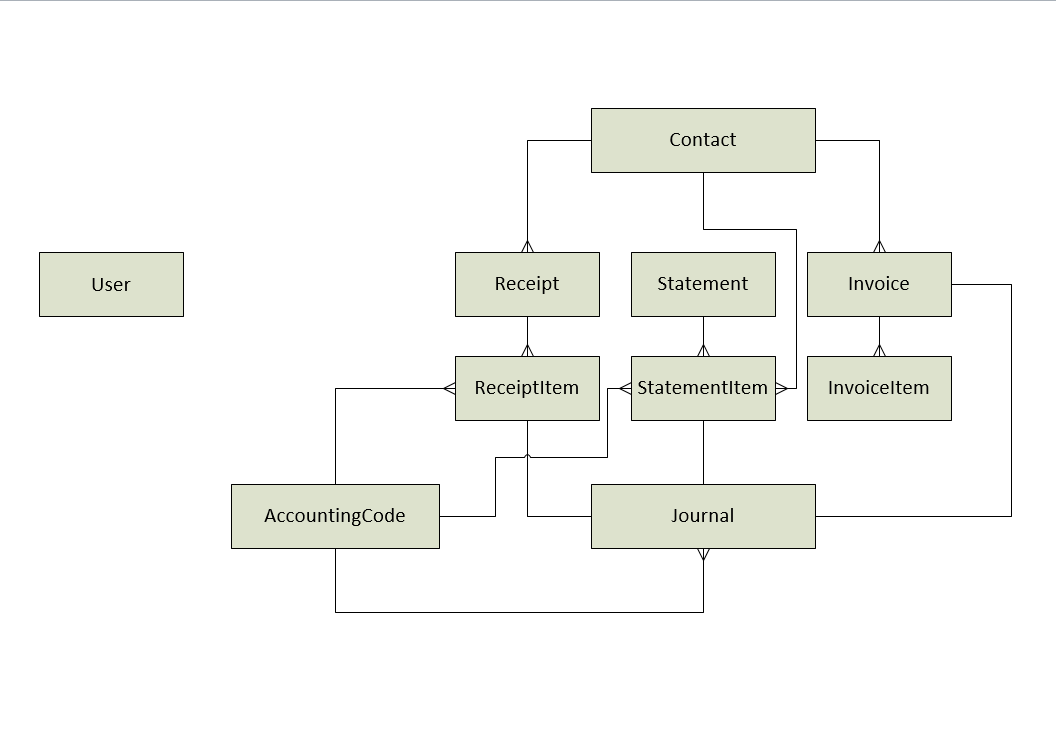
Účetní deník. Slouží k celkovému přehledu, jak si naše firma vede. Každý řádek v deníku reprezentuje jednu účetní operaci. Může to být příjem peněz do fyzické pokladny, výdej peněz z pokladny, příjem peněz na bankovní účet nebo výdej peněz z bankovního účtu. Zároveň se eviduje, zda se daná operace (příjem nebo výdaj) zahrnuje do základu daně z příjmů.

**BK\_ACCOUNTING\_CODE**

Číselník účetních operací (kódová tabulka), jejich kategorizace. U každého kódu je určeno, zda se daná účetní operace (příjem nebo výdaj) zahrnuje nebo nezahrnuje do základu daně z příjmů.

# Datový model

Vztahy mezi business objekty definuje následující E-R diagram.



# Back-endová část

Back-end se skládá ze 3 projektů ve Visual Studiu.

* **BookKeeperBECommon** je knihovna tříd (class library) obsahující business objekty, datovou vrstvu a aplikační (střední) vrstvu. Je targetovaná na .NET 5.0. Používá Entity Framework Core.
* **BookKeeperBEMain** je konzolová aplikace pro otestování datové a aplikační vrstvy back-endu. Je rovněž psaná pro .NET 5.0.
* **BookKeeperBERest** je webový projekt v technologii ASP.NET Core API. Implementuje aplikační programové rozhraní (API) pro REST.

## BookKeeperBECommon

Knihovna tříd s business objekty, datovou a aplikační vrstvou.

**Business objekty**

Business objekty odpovídají jedna k jedné tabulkám v databázi. Jejich názvy a vztahy mezi nimi popisuje E-R diagram (viz část Datový model).

**Datová vrstva**

Obsahuje tzv. repository třídy (např. UserRepoMysql). Tyto třídy komunikují přes Entity Framework přímo s databází.

**Aplikační vrstva**

Obsahuje tzv. servisní třídy (např. UserService). Tyto třídy používají funkcionalitu z repository tříd v datové vrstvě, případně mohou volat metody jiných servisních tříd.

## BookKeeperBEMain

Testovací konzolový projekt.

**Testy**

Třída AdodbMysqlTest otestuje připojení z prostředí .NET k databázi MySQL (MariaDB).

Třída EFMysqlTest otestuje objektově relační mapování (ORM) mezi business objekty v C# a databází MySQL. Použité ORM je Entity Framework Core.

## BookKeeperBERest

Hlavní webový projekt celého back-endu.

**Vrstva REST API**

Konfigurace webové aplikace je ve třídě Startup v metodách CongigureServices a Configure.

Třída UserController realizuje RESTové API pro business objekt User.

Další informace o RESTovém API jsou v dokumentu **REST\_HowDoesItWork\_v....docx**.

# Front-endová část

Front-end projektu BookKeeper je webová aplikace založená na technologii Angular. Z jazykových prostředků používá hlavně HTML, CSS, JavaScript a TypeScript.

## book-keeper-fe-angular

Angularový projekt má dvě vrstvy – komponenty a servisní třídy.

**Komponenty**

Vrstva komponent (např. soubory users.component....) definuje vzhled aplikace. Je to vlastně prezentační vrstva.

Každá komponenta se skládá ze 3 částí:

* Podkladová třída komponenty – např. users.component.ts – definuje datový model a chování.
* Šablona komponenty – např. users.component.html – definuje rozvržení vizuálních prvků na stránce.
* Styly komponenty – např. users.component.css – definuje privátní kaskádové styly (fonty, barvy, odsazení, vizuální reakce na pohyby myši apod.).

**Servisní třídy**

Aby komponenty mohly zobrazovat data, používají vrstvu servisních tříd (např. user.service.ts). Je to vlastně obdoba aplikační vrstvy z back-endu. Servisní třídy komunikují přes HTTP protokol (pomocí angularové třídy HttpClient) s RESTovým API na back-endu.

Celá angularová aplikace používá několik základních souborů:

* **app.module.ts** – deklaruje komponenty, importuje moduly dostupné v celé aplikaci, definuje vizuální shell aplikace (typicky AppComponent)
* **app.component....** – vizuální shell aplikace; v této komponentě skládáme celou stránku (aplikace v Angularu je tzv. SPA – single-page application – tj. v prohlížeči není potřeba přecházet na jiné HTML stránky, celá appka je v jediné obří stránce, která díky routování vždy zobrazí to, co je potřeba)
* **app-routing.module.ts** – definuje routování uvnitř angularové aplikace; přestože Angular generuje SPA appky, je možné se v rámci jedné části aplikace odkázat hyperlinkem na jinou

Další informace k angularovým projektům jsou např. v tutoriálu Tour of Heroes:

<https://angular.io/tutorial>