Návrh webové aplikace – Internetové bankovnictví

Obsah

[Funkční požadavky 1](#_Toc212306794)

[Entity 1](#_Toc212306795)

[Uživatel 1](#_Toc212306796)

[Účet 2](#_Toc212306797)

[Karty 2](#_Toc212306798)

[Transakce 3](#_Toc212306799)

[Banka 4](#_Toc212306800)

# Funkční požadavky

Aplikace bude podporovat registraci a přihlašování uživatele. Vytváření bankovních účtů, transakcí, bankovních karet uživatelů a zobrazování historie transakcí. Uživatel bude mít možnost zobrazit své bankovní účty, své bankovní karty a svou historii transakcí. Bude mít možnost skrze bankovní účet provádět platby a přijímat platby. Každý bankovní účet bude přiřazen ke specifické bance, tím pádem bude mít uživatel možnost mít účet u různých bank.

Uživatel typu Admin bude mít oprávnění vytvářet nové bankovní účty, nové platební karty a bude mít možnost spravovat uživatele v internetovém bankovnictví včetně jejich bankovních účtů a bankovních karet.

# Entity

Tabulky, které budou v databázi a jejich parametry.

## Uživatel

Uživatel má bankovní účty propojené vazbou 1:N.

public class User

{

public int Id { get; set; }

public string Username { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Tel { get; set; }

public string PasswordHash { get; set; }

public DateTime DateOfBirth { get; set; }

public DateTime CreatedAt { get; set; } = DateTime.UtcNow;

public DateTime? UpdatedAt { get; set; }

public UserType UserType { get; set; } = UserType.USER;

// Relationships

public ICollection <Account> Accounts { get; set; } = new List<Account>();

[NotMapped]

public string FullName => $"{FirstName} {LastName}";

}

## Účet

Obsahuje relaci k uživateli, bankovním kartám a transakcím. Vazba bude typu 1:N, kdy jeden uživatel bude mít N bankovních účtů. Jeden bankovní účet bude mít N bankovních karet a N transakcí.

public class Account

{

public int Id { get; set; }

// Account balance

public decimal Balance { get; set; }

// Account details

public string AccountNumber { get; set; }

public string BankCode { get; set; }

public BankAccountType Type { get; set; }

public DateTime CreatedAt { get; set; } = DateTime.UtcNow;

// Relationships

public int UserId { get; set; }

public User User { get; set; }

public ICollection<Card> Cards { get; set; } = new List<Card>();

public ICollection <Transaction> Transactions { get; set; } = new List<Transaction>();

}

## Karty

Obsahuje kreditní/debetní karty přiřazené k bankovnímu účtu. Vazba bude 1:N, kdy jeden účet může mít více karet.

public class Card

{

public int Id { get; set; }

// Card details

[MaxLength(16)]

public string CardNumber { get; set; }

public DateTime ExpirationDate { get; set; }

[MaxLength(3)]

public string SecurityCode { get; set; }

public CardType Type { get; set; }

public string CardHolderName { get; set; }

public bool IsBlocked { get; set; } = false;

public DateTime CreatedAt { get; set; } = DateTime.UtcNow;

// Relationships

public int AccountId { get; set; }

public Account Account { get; set; }

}

## Transakce

Obsahuje relaci k bankovním účtům. Vazba bude typu 1:N, kdy jeden bankovní účet bude mít N transakcí.

public class Transaction

{

public int Id { get; set; }

// Accounts details

public string FromAccountNumber { get; set; }

public string FromBankCode { get; set; }

public string ToAccountNumber { get; set; }

public string ToBankCode { get; set; }

// Symbols

[MaxLength(10)]

public string ConstantSymbol { get; set; }

[MaxLength(10)]

public string VariableSymbol { get; set; }

// Transaction details

public decimal Amount { get; set; }

public DateTime CreatedAt { get; set; } = DateTime.UtcNow;

public TransactionType TransactionType { get; set; }

public TransactionStatus Status { get; set; }

// Relationship

public int AccountId { get; set; }

public Account Account { get; set; }

}

## Banka

Obsahuje bankovní účty a jsou vázány vazbou typu 1:N, kdy jeden bankovní účet může mít pouze jednu banku, ale banka má pod sebou N bankovních účtů.

public class Bank

{

public int Id { get; set; }

// Bank details

public string Name { get; set; }

public string BankCode { get; set; }

public string Address { get; set; }

public string SwiftCode { get; set; }

// Relationships

public ICollection<Account> Accounts { get; set; } = new List<Account>();

}