Fake Bank

Tomáš Moravec
Version
Mon Jan 8 2024

Table of Contents

BankWebApp

This is a banking web application developed using C#, JavaScript, and SQL.

Project Structure

The project is structured into several parts:

- `env/Envs.cs` : Contains the connection string for the MSSQL database.
- `database.sql` : Contains the SQL scripts for creating the necessary tables and indices in the database.

Setup

1. Clone the repository.
2. Install the necessary dependencies.
3. Set up the MSSQL database using the provided SQL scripts.
4. Update the connection string in `env/Envs.cs` with your database details.
5. Run the application.

Features

- User registration and login
- Bank account creation
- Transaction processing
- Transaction history
- Admin dashboard
- User dashboard

Custom User Components

- `Navbar` : The navigation bar at the top of the page. made with View Component.
- `Footer` : The footer at the bottom of the page. made with Partial View.

Namespace Index

Package List

Here are the packages with brief descriptions (if available):

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Namespace Documentation

BankWebApp Namespace Reference

Namespaces

- namespace **Controllers**
  HomeController class that inherits from Controller. This class is responsible for handling the requests related to the Home page of the application.
- namespace **Services**
  The DatabaseService class is responsible for managing the database connection. It implements the IDisposable interface to properly close the connection when it's no longer needed.

Classes

class **Program***The Program class is the entry point of the application.*

BankWebApp.Components Namespace Reference

Classes

class **NavbarViewComponent**

BankWebApp.Controllers Namespace Reference

HomeController class that inherits from Controller. This class is responsible for handling the requests related to the Home page of the application.

Classes

class **AccountController***The AccountController class is responsible for handling requests related to the user's bank account. It includes actions for displaying the account index, transferring funds, adding funds (admin only), listing users (admin only), showing user details (admin only), deleting a user (admin only), and viewing transaction history.*

class **HomeController**

Detailed Description

HomeController class that inherits from Controller. This class is responsible for handling the requests related to the Home page of the application.

BankWebApp.env Namespace Reference

Classes

- class **Envs**
  The Envs static class contains environment variables for the application.

BankWebApp.Models Namespace Reference

Classes

- class **AccountHistoryModel**class **AccountIndexModel**
- class **AddFundsViewModel**
- class **AddressModel**
- class **BankAccountModel**
- class **ContactModel**
- class **ErrorViewModel**
- class **ListUsersViewModel**
- class **LoginModel**
- class **RegisterModel**
- class **RolesModel**
- class **TransactionModel**
- class **TransferViewModel**
- class **UserModel**
  Represents a User in the system.

BankWebApp.Services Namespace Reference

The DatabaseService class is responsible for managing the database connection. It implements the IDisposable interface to properly close the connection when it's no longer needed.

Classes

class **DatabaseService***The DatabaseService class is responsible for managing the database operations. It contains methods for getting users, checking if a username exists, registering a user, getting bank accounts by user id, transferring funds, getting roles by user id, getting all bank accounts, adding funds, and getting transactions.*

class **MySignInManager***This class is responsible for managing user sign in and sign out operations.*

class **TransferService***This class provides services for transferring money between bank accounts.*

class **UserService***Service class for managing users.*

Detailed Description

The DatabaseService class is responsible for managing the database connection. It implements the IDisposable interface to properly close the connection when it's no longer needed.

BankWebApp.Tools Namespace Reference

Classes

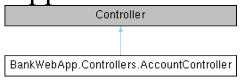
- class **ClaimTools**
- class **PasswordHashes**

Class Documentation

BankWebApp.Controllers.AccountController Class Reference

The AccountController class is responsible for handling requests related to the user's bank account. It includes actions for displaying the account index, transferring funds, adding funds (admin only), listing users (admin only), showing user details (admin only), deleting a user (admin only), and viewing transaction history.

Inheritance diagram for BankWebApp.Controllers.AccountController:



Public Member Functions

- **AccountController** (ILogger< **AccountController** > logger, **UserService** userService, **TransferService** transferService)
  Initializes a new instance of the AccountController class.
- IActionResult **Index** ()
  Displays the account index page.
- IActionResult **Transfer** (bool? success=null, string? reason=null)
  Displays the transfer page.
- IActionResult **Transfer** (**TransferViewModel** model)
  Handles a POST request to transfer funds.
- IActionResult **AddFunds** (bool? success=null, string? reason=null)
  Displays the add funds page (admin only).
- IActionResult **AddFunds** (**AddFundsViewModel** model)
  Handles a POST request to add funds to an account (admin only).
- IActionResult **ListUsers** ()
  Displays the list users page (admin only).
- IActionResult **Show** (string id)
  Displays the details of a user (admin only).
- IActionResult **History** ()
  Displays the transaction history page.

Detailed Description

The AccountController class is responsible for handling requests related to the user's bank account. It includes actions for displaying the account index, transferring funds, adding funds (admin only), listing users (admin only), showing user details (admin only), deleting a user (admin only), and viewing transaction history.

Constructor & Destructor Documentation

BankWebApp.Controllers.AccountController.AccountController (ILogger< AccountController > *logger*, UserService *userService*, TransferService *transferService*)

Initializes a new instance of the AccountController class.

Parameters

logger	The logger used to log information about program execution.
userService	The service used to interact with user data.
transferService	The service used to handle money transfers.

Member Function Documentation

IActionResult BankWebApp.Controllers.AccountController.AddFunds (AddFundsViewModel *model*)

Handles a POST request to add funds to an account (admin only).

Parameters

model	The data from the add funds form.

Returns

A redirect to the add funds page, with success and reason parameters.

IActionResult BankWebApp.Controllers.AccountController.AddFunds (bool? *success* = `null`, string? *reason* = `null`)

Displays the add funds page (admin only).

Parameters

success	Indicates whether the last add funds operation was successful.
reason	The reason for the last add funds operation's failure, if it failed.

Returns

The add funds view.

IActionResult BankWebApp.Controllers.AccountController.History ()

Displays the transaction history page.

Returns

The history view.

IActionResult BankWebApp.Controllers.AccountController.Index ()

Displays the account index page.

Returns

The account index view.

IActionResult BankWebApp.Controllers.AccountController.ListUsers ()

Displays the list users page (admin only).

Returns

The list users view.

IActionResult BankWebApp.Controllers.AccountController.Show (string *id*)

Displays the details of a user (admin only).

Parameters

id	The ID of the user to show.

Returns

The show view.

IActionResult BankWebApp.Controllers.AccountController.Transfer (bool? *success* = `null`, string? *reason* = `null`)

Displays the transfer page.

Parameters

success	Indicates whether the last transfer was successful.
reason	The reason for the last transfer's failure, if it failed.

Returns

The transfer view.

IActionResult BankWebApp.Controllers.AccountController.Transfer (TransferViewModel *model*)

Handles a POST request to transfer funds.

Parameters

model	The data from the transfer form.

Returns

A redirect to the transfer page, with success and reason parameters.

The documentation for this class was generated from the following file:

- Controllers/AccountController.cs

BankWebApp.Models.AccountHistoryModel Class Reference

Properties

- IList< **TransactionModel** > **Transactions** `[get, set]`

The documentation for this class was generated from the following file:

- Models/AccountHistoryModel.cs

BankWebApp.Models.AccountIndexModel Class Reference

Properties

- **UserModel SignedInUser** `[get, set]`
- IList< **BankAccountModel** > **BankAccounts** `[get, set]`

The documentation for this class was generated from the following file:

- Models/AccountIndexModel.cs

BankWebApp.Models.AddFundsViewModel Class Reference

Properties

- IList< **BankAccountModel** > **BankAccounts** `[get, set]`
- decimal **Amount** `[get, set]`
- string **SelectedBankAccountNumber** `[get, set]`
- bool? **Success** `[get, set]`
- string? **Reason** `[get, set]`

The documentation for this class was generated from the following file:

- Models/AddFundsViewModel.cs

BankWebApp.Models.AddressModel Class Reference

Properties

- int **Id** `[get, set]`
- string **Street** `[get, set]`
- string **City** `[get, set]`
- string **PostCode** `[get, set]`
- string **Country** `[get, set]`

The documentation for this class was generated from the following file:

- Models/AddressModel.cs

BankWebApp.Models.BankAccountModel Class Reference

Properties

- int **Id** `[get, set]`
- string **AccountNumber** `[get, set]`
- decimal **Balance** `[get, set]`
- int **UserId** `[get, set]`
- **UserModel User** `[get, set]`

The documentation for this class was generated from the following file:

- Models/BankAccountModel.cs

BankWebApp.Models.ContactModel Class Reference

Properties

- int **Id** `[get, set]`
- string **Email** `[get, set]`
- string **PhoneNumber** `[get, set]`

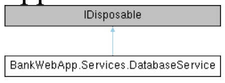
The documentation for this class was generated from the following file:

- Models/ContactModel.cs

BankWebApp.Services.DatabaseService Class Reference

The DatabaseService class is responsible for managing the database operations. It contains methods for getting users, checking if a username exists, registering a user, getting bank accounts by user id, transferring funds, getting roles by user id, getting all bank accounts, adding funds, and getting transactions.

Inheritance diagram for BankWebApp.Services.DatabaseService:



Public Member Functions

- **DatabaseService** ()
  The constructor initializes a new instance of the DatabaseService class. It sets the connection string and opens the connection.

- void **Dispose** ()
  The Dispose method is called when the DatabaseService object is being disposed. It closes the database connection.

- IList< **UserModel** > **GetUsers** ()
  Gets all users from the database.

- bool **UsernameExists** (string _username)
  Gets all users from the database.

- bool **RegisterUser** (**UserModel** user)
  Registers a new user in the database.

- IList< **BankAccountModel** >? **GetBankAccountById** (int UserId)
  Gets a list of bank accounts by user id.

- **BankAccountModel**? **GetBankAccountById** (string Id)
  Gets a bank account by account number.

- **BankAccountModel**? **GetBankAccountByAccountId** (int Id)
  Gets a bank account by account id.

- bool **TransferFunds** (Guid from, Guid To, decimal Amount)
  Transfers funds from one account to another.

- IList< **RolesModel** > **GetRolesById** (int uid)
  Gets a list of roles by user id.

- IList< **BankAccountModel** > **GetAllBankAccounts** ()
  Gets all bank accounts from the database.

- void **AddFunds** (Guid guid, decimal amount)
  Adds funds to a bank account.

- IList< **TransactionModel** > **GetTransactions** ()
  Gets all transactions from the database.

- IList< **TransactionModel** > **GetTransactions** (int uid)
  Gets a list of transactions by user id.

Detailed Description

The DatabaseService class is responsible for managing the database operations. It contains methods for getting users, checking if a username exists, registering a user, getting bank accounts by user id, transferring funds, getting roles by user id, getting all bank accounts, adding funds, and getting transactions.

Member Function Documentation

void BankWebApp.Services.DatabaseService.AddFunds (Guid *guid*, decimal *amount*)

Adds funds to a bank account.

Parameters

guid	The account number to add funds to.
amount	The amount to add.

IList< BankAccountModel > BankWebApp.Services.DatabaseService.GetAllBankAccounts ()

Gets all bank accounts from the database.

Returns

A list of BankAccountModel objects.

BankAccountModel? BankWebApp.Services.DatabaseService.GetBankAccountByAccountId (int *Id*)

Gets a bank account by account id.

Parameters

Id	The account id to get the bank account for.

Returns

A BankAccountModel object.

IList< BankAccountModel >? BankWebApp.Services.DatabaseService.GetBankAccountById (int *UserId*)

Gets a list of bank accounts by user id.

Parameters

UserId	The user id to get the bank accounts for.

Returns

A list of BankAccountModel objects.

BankAccountModel? BankWebApp.Services.DatabaseService.GetBankAccountById (string *Id*)

Gets a bank account by account number.

Parameters

Id	The account number to get the bank account for.

Returns

A BankAccountModel object.

IList< RolesModel > BankWebApp.Services.DatabaseService.GetRolesById (int *uid*)

Gets a list of roles by user id.

Parameters

uid	The user id to get the roles for.

Returns

A list of RolesModel objects.

IList< TransactionModel > BankWebApp.Services.DatabaseService.GetTransactions ()

Gets all transactions from the database.

Returns

A list of TransactionModel objects.

IList< TransactionModel > BankWebApp.Services.DatabaseService.GetTransactions (int *uid*)

Gets a list of transactions by user id.

Parameters

uid	The user id to get the transactions for.

Returns

A list of TransactionModel objects.

IList< UserModel > BankWebApp.Services.DatabaseService.GetUsers ()

Gets all users from the database.

Returns

A list of UserModel objects.

bool BankWebApp.Services.DatabaseService.RegisterUser (UserModel *user*)

Registers a new user in the database.

Parameters

user	The user to register.

Returns

True if the registration was successful, false otherwise.

bool BankWebApp.Services.DatabaseService.TransferFunds (Guid *from*, Guid *To*, decimal *Amount*)

Transfers funds from one account to another.

Parameters

from	The account number to transfer funds from.
To	The account number to transfer funds to.
Amount	The amount to transfer.

Returns

True if the transfer was successful, false otherwise.

bool BankWebApp.Services.DatabaseService.UsernameExists (string *_username*)

Gets all users from the database.

Returns

A list of UserModel objects.

The documentation for this class was generated from the following files:

- Services/DatabaseService.cs
- Services/DatabaseServiceFunctions.cs

BankWebApp.Models.ErrorViewModel Class Reference

Properties

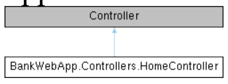
- string? **RequestId** `[get, set]`
- bool **ShowRequestId** `[get]`

The documentation for this class was generated from the following file:

- Models/ErrorViewModel.cs

BankWebApp.Controllers.HomeController Class Reference

Inheritance diagram for BankWebApp.Controllers.HomeController:



Public Member Functions

- **HomeController** (ILogger< **HomeController** > logger, **UserService** userService, **MySignInManager** signInManager)
  HomeController constructor. Initializes a new instance of the HomeController class.
- IActionResult **Index** ()
  Handles the GET request for the Index view.
- IActionResult **Privacy** ()
  Handles the GET request for the Privacy view.
- IActionResult **Login** ()
  Handles the GET request for the Login view.
- IActionResult **Login** (**LoginModel** loginModel)
  Handles the POST request for the Login view.
- IActionResult **Logout** ()
  Handles the request to log out the user.
- IActionResult **Error** ()
  Handles the GET request for the Error view.
- IActionResult **AccessDenied** ()
  Handles the request when the user is denied access.
- IActionResult **Register** ()
  Handles the GET request for the Register view.
- IActionResult **Register** (**RegisterModel** registerModel)
  Handles the POST request for the Register view.

Constructor & Destructor Documentation

BankWebApp.Controllers.HomeController.HomeController (ILogger< HomeController > *logger*, UserService *userService*, MySignInManager *signInManager*)

HomeController constructor. Initializes a new instance of the HomeController class.

Parameters

logger	An instance of ILogger interface to handle logging.
userService	An instance of UserService to handle user related operations.
signInManager	An instance of MySignInManager to handle user sign in operations.

Member Function Documentation

IActionResult BankWebApp.Controllers.HomeController.AccessDenied ()

Handles the request when the user is denied access.

Returns

The Error view along with the request id.

IActionResult BankWebApp.Controllers.HomeController.Error ()

Handles the GET request for the Error view.

Returns

The Error view along with the request id.

IActionResult BankWebApp.Controllers.HomeController.Index ()

Handles the GET request for the Index view.

Returns

The Index view.

IActionResult BankWebApp.Controllers.HomeController.Login ()

Handles the GET request for the Login view.

Returns

The Login view.

IActionResult BankWebApp.Controllers.HomeController.Login (LoginModel *loginModel*)

Handles the POST request for the Login view.

Parameters

loginModel	The login details provided by the user.

Returns

The Login view if the model state is invalid, otherwise redirects to the appropriate view based on the login details.

IActionResult BankWebApp.Controllers.HomeController.Logout ()

Handles the request to log out the user.

Returns

Redirects to the Login view after successfully logging out the user.

IActionResult BankWebApp.Controllers.HomeController.Privacy ()

Handles the GET request for the Privacy view.

Returns

The Privacy view.

IActionResult BankWebApp.Controllers.HomeController.Register ()

Handles the GET request for the Register view.

Returns

The Register view.

IActionResult BankWebApp.Controllers.HomeController.Register (RegisterModel *registerModel*)

Handles the POST request for the Register view.

Parameters

registerModel	The registration details provided by the user.

Returns

The Register view if the model state is invalid, otherwise redirects to the Login view after successful registration.

The documentation for this class was generated from the following file:

- Controllers/HomeController.cs

BankWebApp.Models.ListUsersViewModel Class Reference

Properties

- **UserModel UserModel** `[get, set]`
- IList< **BankAccountModel** > **BankAccounts** `[get, set]`
- IList< **TransactionModel** > **Transactions** `[get, set]`

The documentation for this class was generated from the following file:

- Models/ListUsersViewModel.cs

BankWebApp.Models.LoginModel Class Reference

Properties

- string **Username** `[get, set]`
- string **Password** `[get, set]`
- bool **RememberMe** `[get, set]`

The documentation for this class was generated from the following file:

- Models/LoginModel.cs

BankWebApp.Services.MySignInManager Class Reference

This class is responsible for managing user sign in and sign out operations.

Public Member Functions

- **MySignInManager** (IHttpContextAccessor httpContextAccessor, **UserService** userService)
  Initializes a new instance of the MySignInManager class.

- async Task **SignInAsync** (**UserModel** user, bool isPersistent=false)
  Signs in the specified user.

- async Task **SignOutAsync** ()
  Signs out the current user.

Detailed Description

This class is responsible for managing user sign in and sign out operations.

Constructor & Destructor Documentation

BankWebApp.Services.MySignInManager.MySignInManager (IHttpContextAccessor *httpContextAccessor*, UserService *userService*)

Initializes a new instance of the MySignInManager class.

Parameters

httpContextAccessor	The HTTP context accessor.
userService	The user service.

Member Function Documentation

async Task BankWebApp.Services.MySignInManager.SignInAsync (UserModel *user*, bool *isPersistent* = `false`)

Signs in the specified user.

Parameters

user	The user to sign in.
isPersistent	if set to `true` the sign in is persistent.

Returns

A Task representing the asynchronous operation.

The user model should already be validated before calling this method.

async Task BankWebApp.Services.MySignInManager.SignOutAsync ()

Signs out the current user.

Returns

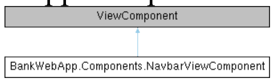
A Task representing the asynchronous operation.

The documentation for this class was generated from the following file:

- Services/MySignInManager.cs

BankWebApp.Components.NavbarViewComponent Class Reference

Inheritance diagram for BankWebApp.Components.NavbarViewComponent:



Public Member Functions

- IViewComponentResult **Invoke** ()

The documentation for this class was generated from the following file:

- Components/NavbarViewComponent.cs

BankWebApp.Program Class Reference

The Program class is the entry point of the application.

Static Public Member Functions

- static void **Main** (string[] args)
  The Main method is responsible for setting up and running the web application.

Detailed Description

The Program class is the entry point of the application.

Member Function Documentation

static void BankWebApp.Program.Main (string[] *args*) `[static]`

The Main method is responsible for setting up and running the web application.

Parameters

args	Command-line arguments passed to the application.

The documentation for this class was generated from the following file:

- Program.cs

BankWebApp.Models.RegisterModel Class Reference

Properties

- string **Username** `[get, set]`
- string **Password** `[get, set]`
- string **ConfirmPassword** `[get, set]`
- string **Email** `[get, set]`
- string **PhoneNumber** `[get, set]`
- string **Street** `[get, set]`
- string **City** `[get, set]`
- string **PostCode** `[get, set]`
- string **Country** `[get, set]`
- bool? **Success** `[get, set]`
- string? **Reason** `[get, set]`

The documentation for this class was generated from the following file:

- Models/RegisterModel.cs

BankWebApp.Models.RolesModel Class Reference

Properties

- int **Id** `[get, set]`
- string **RoleName** `[get, set]`
- int **UserId** `[get, set]`

The documentation for this class was generated from the following file:

- Models/RolesModel.cs

BankWebApp.Models.TransactionModel Class Reference

Properties

- int **Id** `[get, set]`
- int **SenderId** `[get, set]`
- **BankAccountModel** **Sender** `[get, set]`
- int **ReceiverId** `[get, set]`
- **BankAccountModel** **Receiver** `[get, set]`
- decimal **Amount** `[get, set]`
- DateTime **SentAt** `[get, set]`

The documentation for this class was generated from the following file:

- Models/TransactionModel.cs

BankWebApp.Services.TransferService Class Reference

This class provides services for transferring money between bank accounts.

Public Member Functions

- **TransferService** ()
  Initializes a new instance of the TransferService class.

- bool string Reason **TransferMoney** (string FromAcc, string ToAcc, decimal Amount)
- void **PrintMoney** (string AccountNumber, decimal Amount)
  Adds funds to a specified account.

Public Attributes

- bool **Success**
  Transfers money from one account to another.

Detailed Description

This class provides services for transferring money between bank accounts.

Member Function Documentation

void BankWebApp.Services.TransferService.PrintMoney (string *AccountNumber*, decimal *Amount*)

Adds funds to a specified account.

Parameters

AccountNumber	The account number to add funds to.
Amount	The amount of money to add.

Member Data Documentation

bool BankWebApp.Services.TransferService.Success

Transfers money from one account to another.

Parameters

FromAcc	The account number to transfer money from.
ToAcc	The account number to transfer money to.
Amount	The amount of money to transfer.

Returns

A tuple containing a boolean indicating success or failure, and a string containing the reason for failure.

The documentation for this class was generated from the following file:

- Services/TransferService.cs

BankWebApp.Models.TransferViewModel Class Reference

Properties

- IList< **BankAccountModel** > **BankAccounts** `[get, set]`
- string **FromAccountId** `[get, set]`
- string **ToAccountId** `[get, set]`
- decimal **Amount** `[get, set]`
- bool? **Success** `[get, set]`
- string? **Reason** `[get, set]`
- static **TransferViewModel** **Empty** `[get]`

The documentation for this class was generated from the following file:

- Models/TransferViewModel.cs

BankWebApp.Models.UserModel Class Reference

Represents a User in the system.

Properties

- int **Id** `[get, set]`
  Unique identifier for the user.

- string **Username** `[get, set]`
  Username of the user.

- string **PasswordHash** `[get, set]`
  Hashed password of the user. with bcrypt.

- DateTime **CreatedAt** `[get, set]`
  A date when the user was created. (in database)

- **ContactModel Contact** `[get, set]`
  Contact model associated with the user.

- **AddressModel Address** `[get, set]`
  Address model associated with the user.

Detailed Description

Represents a User in the system.

The documentation for this class was generated from the following file:

- Models/UserModel.cs

BankWebApp.Services.UserService Class Reference

Service class for managing users.

Public Member Functions

- **UserService** ()
  Constructor for UserService. Initializes a new instance of the DatabaseService and refreshes the cache.

- IList< **UserModel** > **GetUsers** ()
  Retrieves the list of users. If the cache is null or expired, it refreshes the cache before returning the users.

- **UserModel**? **GetUserById** (int id)
  Retrieves a user by their ID.

- **UserModel**? **GetUserByUsername** (string username)
  Retrieves a user by their username.

- bool string reason **RegisterUser** (**RegisterModel** newUser)
- IList< **BankAccountModel** > **GetBankAccountsById** (int uid)
  Retrieves the bank accounts of a user by their ID.

- **BankAccountModel**? **GetBankAccountsById** (string id)
  Retrieves a bank account by its account number.

- IList< **BankAccountModel** > **GetAllBankAccounts** ()
  Retrieves all bank accounts.

- IList< **RolesModel** > **GetRolesById** (int uid)
  Retrieves the roles of a user by their ID.

- IList< **TransactionModel** > **GetAllTransactions** ()
  Retrieves all transactions.

- IList< **TransactionModel** > **GetTransactionsByAccountId** (int accountId)
  Retrieves the transactions of a bank account by its ID.

Public Attributes

- bool **success**
  Registers a new user.

Detailed Description

Service class for managing users.

Member Function Documentation

IList< BankAccountModel > BankWebApp.Services.UserService.GetAllBankAccounts ()

Retrieves all bank accounts.

Returns

A list of all bank accounts.

IList< TransactionModel > BankWebApp.Services.UserService.GetAllTransactions ()

Retrieves all transactions.

Returns

A list of all transactions.

IList< BankAccountModel > BankWebApp.Services.UserService.GetBankAccountsById (int *uid*)

Retrieves the bank accounts of a user by their ID.

Parameters

uid	The ID of the user.

Returns

A list of bank accounts owned by the user.

BankAccountModel? BankWebApp.Services.UserService.GetBankAccountsById (string *id*)

Retrieves a bank account by its account number.

Parameters

id	The account number of the bank account.

Returns

The bank account with the given account number, or null if no such account exists.

IList< RolesModel > BankWebApp.Services.UserService.GetRolesById (int *uid*)

Retrieves the roles of a user by their ID.

Parameters

uid	The ID of the user.

Returns

A list of roles assigned to the user.

IList< TransactionModel > BankWebApp.Services.UserService.GetTransactionsByAccountId (int *accountId*)

Retrieves the transactions of a bank account by its ID.

Parameters

accountId	The ID of the bank account.

Returns

A list of transactions associated with the bank account.

UserModel? BankWebApp.Services.UserService.GetUserById (int *id*)

Retrieves a user by their ID.

Parameters

id	The ID of the user.

Returns

The user with the given ID, or null if no such user exists.

UserModel? BankWebApp.Services.UserService.GetUserByUsername (string *username*)

Retrieves a user by their username.

Parameters

username	The username of the user.

Returns

The user with the given username, or null if no such user exists.

IList< UserModel > BankWebApp.Services.UserService.GetUsers ()

Retrieves the list of users. If the cache is null or expired, it refreshes the cache before returning the users.

Returns

A list of UserModel instances.

Member Data Documentation

bool BankWebApp.Services.UserService.success

Registers a new user.

Parameters

newUser	The details of the new user.

Returns

A tuple indicating whether the registration was successful and a reason for failure, if applicable.

The documentation for this class was generated from the following file:

- Services/UserService.cs

Index

INDEX