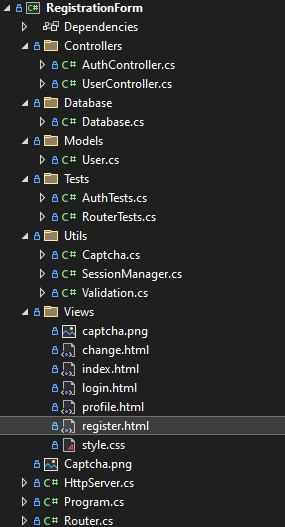
# RegistrationForm – пълна документация на проекта

## 0) Резюме

Самостоятелно конзолно .NET 8 приложение, което играе роля на елементарен HTTP сървър на порт 8080. Предоставя форми за регистрация, вход, промяна на профил и изтриване на потребител. Данните за потребителите се пазят в SQL Server. Има сесийна система в паметта и елементарна CAPTCHA.

## Структура на папките и файловете



## 2) NuGet пакети и системни пространства от имена

### 2.1 Пакети

* NUnit
* System.Data.SqlClient
* System.Drawing.Common – генериране/запис на CAPTCHA изображение.

### 2.2 Използвани using System.\*

* System
* System.Collections.Generic
* System.ComponentModel.DataAnnotations
* System.Data.SqlClient
* System.Drawing
* System.Drawing.Imaging
* System.IO
* System.Linq
* System.Net
* System.Net.Mime
* System.Net.NetworkInformation
* System.Net.Sockets
* System.Reflection
* System.Security.Cryptography
* System.Text
* System.Text.RegularExpressions
* System.Threading.Tasks
* System.Web

## 3) Описание на класовете

### 3.1 Program.cs

Стартира HTTP сървъра на порт 8080.

### 3.2 HttpServer.cs

Реализира уеб сървъра на приложението. Метода Start стартира TcpListener след това отваря един безкраен цикъл, в който взима заявките и ги обработва в стандартен http текст. Добавя и сесия от бисквитките. Различава различните видове заявки (страница, captcha). Накрая предава цялата заявка на клас Router.cs. Метода HandleCaptchaImage има за цел да генерира Captcha изображението, след това изпраща header-a в байтове, след това самото изображение. Метода ExtractSessionId има за цел да върне стойността на бисквитката, ако такава съществува. Така сървърът знае дали има настоящ потребител.

### 3.3 Router.cs

Методът HandleRequest в Router анализира съдържанието на заявката, разбива я на части и определя какви данни да върне спрямо вида на заявката: POST, GET.

### 3.4 Controllers/AuthController.cs

Класът AuthController.cs съдържа методите Register, Login, Logout. В него се извършва част от валидацията на входните данни, извикват се методи от базата данни за обработка на потребителите (Create, Read) и се създават сесии за потребителите където е нужно.

### 3.5 Controllers/UserController.cs

Класът UserController.cs съдържа методите Change, Delete. Както при предният контролер те използват методите от базата данни Edit and Delete.

### 3.6 Database/Database.cs

Класът Database реализира CRUD методите към базата данни. В началото е създадена променливата connectionString, в която се пази връзката към базата данни. Базата данни е правена на MS SQL, съдържа една таблица Users, която е описана по надолу в документацията. За създаването на заявките към базата данни е използвано System.Data.SqlClient.

### 3.7 Models/User.cs

Класът User.cs се използва, за да се създате удобен обект User, който да се използва в другите класове. Той има полета: Id, Email, Name, PasswordHash.

### 3.8 Utils/Validation.cs

В клас Validation има 2 метода:

IsValidEmail.cs, който проверява дали въведения имейл има валиден regex.

HashPassword, който хеш-ва паролите на потребителите чрез алгоритъма SHA256, във формат UTF8. Използван е System.Security.Cryptography.

### 3.9 Utils/SessionManager.cs

Клас SessionManager.cs обработва сесиите. Има подклас session, който създава обекти, които пазят данни за сесията: Email, CreatedAt. Имплементира и методи CreateSession, GetSession, RemoveSession и isAutenticated. Една сесия е с времетраене 15 мин. и сесиите се пазят в Речник. За sessionId е използван имейлът на потребителя, защото по удобно се работи с него, когато трябва да бъде достъпена и така се знае на кого принадлежи тя без допълнителна променлива.

### 3.10 Utils/Captcha.cs

В класът Captcha.cs има подклас, в който се складира captcha-та, като обект и методът GenerateCaptcha, който връща кода, който трябва да се въведе и самата снимка. Използва се System.Drawing, за да се създаде и запази в масив от байтове снимката, а кода се генерира, като случайно четирицифрено число.

### 3.11 Views/

Съдържа html страниците на приложението и css файлът с оформление, който използват всички те. В registration.html има използван JavaScript, за работа с Captcha при създаването на акаунт.

### 3.12 Tests/

Тестовете са правени с NUnit. Ползват модела Arrange, Act, Assert. Има общо 30 теста, като в имената на всеки тест е заложено какво проверява. Разделени са на 2 класа RouterTests и AuthTests, за двата различни класа, над които са направени тестовете.

## 4) Работи ли базата данни?

Кодът е написан, така, че да се свързва с SQL Server ако имате:

1) Инсталирана/достъпна инстанция DEFAULTSQLSERVER на локалната машина.

2) Създадена база REGISTRATIONDB.

3) Създадена таблица Users с колоните Id (INT IDENTITY PK), Email (NVARCHAR UNIQUE), Name (NVARCHAR), PasswordHash (NVARCHAR).

4) Разрешен достъп чрез Windows Authentication за текущия потребител (понеже Trusted\_Connection=True).

## 5) Как работи базата данни и как да я настроите в SQL Server Management Studio (SSMS)

### 5.1 Създаване/проверка на инстанция

* Отворете SQL Server Configuration Manager и уверете се, че имате инстанция с име DEFAULTSQLSERVER. Ако ползвате стандартната MSSQLSERVER или друга (например SQLEXPRESS), променете connection string-а в Database.cs:
  + Пример: Server=.\\SQLEXPRESS;Database=REGISTRATIONDB;Trusted\_Connection=True;

### 5.2 Създаване на база и таблица през SSMS

1. Свържете се към сървъра (Windows Authentication).
2. Десен клик Databases → New Database… → име: REGISTRATIONDB → OK.
3. В нов прозорец New Query изпълнете:

USE REGISTRATIONDB;

CREATE TABLE USERS (  
 Id INT IDENTITY(1,1) PRIMARY KEY,  
 Email NVARCHAR(255) NOT NULL UNIQUE,  
 Name NVARCHAR(255) NOT NULL,  
 PasswordHash NVARCHAR(255) NOT NULL  
 );

### 5.3 Тест на връзката

* Във Visual Studio стартирайте приложението (или dotnet run).
* Изпратете POST към /register през формата register.html. При успех трябва да видите потвърждение, а в SSMS SELECT \* FROM USERS; ще покаже новия запис.

### 5.4 Смяна между Windows и SQL логин

* Ако предпочитате SQL логин: създайте потребител в SQL Server и сменете connection string-а, например:
  + Server=.\\DEFAULTSQLSERVER;Database=REGISTRATIONDB;User Id=appuser;Password=Strong#Pass1;
  + Махнете Trusted\_Connection=True.

## 6) HTTP маршрути и потоци

* GET / - index.html (линкове към login/register)
* GET /register.html - форма за регистрация + CAPTCHA (/captcha.png)
* POST /register - създава запис в БД, валидира имейл и CAPTCHA; хешира паролата
* GET /login.html - форма за вход
* POST /login - създава сесия, показва profile.html с попълнени данни
* GET /profile.html - изисква активна сесия
* GET /change.html - форма за промяна, изисква активна сесия
* POST /change - актуализира име/парола в БД
* POST /delete - изтрива потребителя и връща към index.html
* GET /captcha.png - двоичен PNG отговор с текущата CAPTCHA