TESZTELÉSI DOKUMENTÁCIÓ

Készítették:

Lengyel Bálint

Kelemen Ádám

Juhász Balázs

Konzulens:

Farkas Zoltán

Miskolc

2024.

FELHASZNÁLT TECHNOLÓGIÁK

C#:

A dokumentáció elkészítése során a C# programozási nyelv volt alkalmazva. A C# egy erős típusú, objektumorientált programozási nyelv, amelyet a Microsoft fejlesztett ki a .NET keretrendszer számára. A nyelv támogatja a modern programozási paradigmákat és számos lehetőséget kínál a hatékony és skálázható alkalmazásfejlesztéshez.

NUnit Tesztelési Keretrendszer:

Az NUnit tesztelési keretrendszert választottuk az egységtesztek írásához és futtatásához. Az NUnit egy nyílt forráskódú, keresztplatformos tesztelési keretrendszer, amely lehetővé teszi a fejlesztők számára, hogy egyszerűen írjanak és futtassanak egységteszteket C# és .NET alkalmazásaikhoz. Könnyen kezelhető és rugalmas eszközkészletet biztosít a tesztelési feladatokhoz.

RestRequest:

A RestRequest egy könnyű és intuitív .NET könyvtár, amelyet a HTTP kérések kezelésére használtunk a dokumentációban. Segítségével könnyen hozhatunk létre HTTP kéréseket és kommunikálhatunk RESTful szolgáltatásokkal. A RestRequest lehetővé teszi az egyszerű és hatékony HTTP kérések összeállítását és végrehajtását, és számos hasznos funkciót kínál a webes kommunikációhoz.

Newtonsoft.Json:

A Newtonsoft.Json (vagy röviden Json.NET) a .NET platform legnépszerűbb JSON feldolgozási könyvtára. Ezt a könyvtárat használtuk a JSON adatok feldolgozásához és kezeléséhez a dokumentációban. A Json.NET könnyen integrálható .NET alkalmazásokba, és kiváló teljesítményt és funkcionalitást nyújt a JSON adatok olvasásához és írásához.

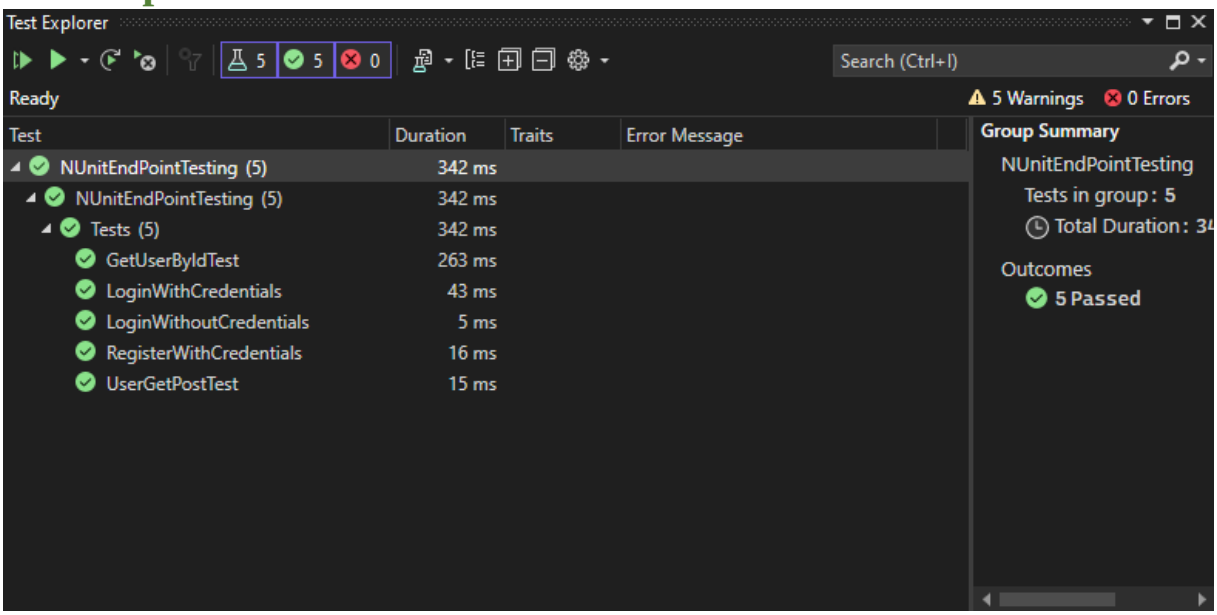
Lighthouse:

A Google Chrome LightHouse egy ingyenes marketing eszköz, amely lehetővé teszi, hogy értékeljük és mérjük az oldalak teljesítményét. Ez az eszköz segít a webfejlesztőknek és webmestereknek megérteni, hogyan működnek az oldalaik és javaslatokat adnak a fejlesztési lehetőségekre.

A Google Chrome LightHouse segítségével könnyedén értékelhetjük az oldalak teljesítményét. Az eszköz mérőpontokat használ, mint például a betöltési sebesség, a felhasználói élmény, az elérhetőség és a bevált gyakorlatok betartása. Ezáltal megkapjuk az oldalunk részletes értékelését és javaslatokat a fejlesztésre.

UNIT TEST

Test Explorer:



LoginWithCredentials: (Bejelentkezés megfelelő adatokkal)

```
[Test]
Tabnine | 0 references
public void LoginWithCredentials()
{
    RestRequest request = new RestRequest("https://localhost:7043/Auth/login", Method.Post);
    request.AddBody(new LoginRequestDto
    {
        UserName = "bazsi",
        Password = "Valami123>"
    }, ContentType.Json);
    RestResponse response = client.Execute(request);
    response.ShouldNotBeNull();
    response.StatusCode.ShouldBe(HttpStatusCode.OK);
    Console.WriteLine("Elvárt eredmény: Json adat");
    Console.WriteLine($"Kapott eredmény: {response.Content}");
    token = response.Content;
}
```

Várt eredmény: bejelentkezés sikeres, válaszban megkapjuk a JTW tokent

A teszt sikeresen lefutott!

LoginWithCredentials: (Bejlentkezés hibás adatokkal (Nem létező felhasználóval)

```
[Test]
Tabnine | ● | 0 references
public void LoginWithoutCredentials() {
    RestRequest request = new RestRequest("https://localhost:7043/Auth/login", Method.Post);
    request.AddBody(new LoginRequestDto
    {
        UserName = "asdasd",
        Password = "asdasd13"
    }, ContentType.Json);
    RestResponse response = client.Execute(request);
    response.ShouldNotBeNull();
    response.StatusCode.ShouldBe(HttpStatusCode.BadRequest);
    Console.WriteLine("Elvárt eredmény: User not found!");
    Console.WriteLine($"Kapott eredmény: {response.Content}");
}
```

Várt eredmény: A bejelentkezés visszautasítja a backend, „User not found!” szöveggel tér vissza! A teszt sikeres!

RegisterWithCredentials: (Regisztráció Tesztelése)

```
[Test]
Tabnine | ● | 0 references
public void RegisterWithCredentials() {

    RestRequest request = new RestRequest($"https://localhost:7043/Auth/Register?validationUrl={GenerateRandomString(7)}", Method.Post);
    request.AddJsonBody(new
    {
        userName = "Sanyika",
        password = "Asdasd123*",
        email = "asdasd@gmail.com"
    });

    RestResponse response = client.Execute(request);
    response.ShouldNotBeNull();
    response.StatusCode.ShouldBeOneOf(HttpStatusCode.OK, HttpStatusCode.BadRequest);
    Console.WriteLine($"Kapott eredmény: {response.Content}");
}
```

A teszt két elvárt eredménnyel térhet vissza. Ha a felhasználó még nem létezik akkor az elvárt eredmény a „User added!” lesz, ha pedig a felhasználó már létező adatokkal (felhasználónév, e-mail cím) szeretne regisztrálni akkor az elvárt eredmény „User existing!” szöveg lesz. A teszt még nem létező adatokkal sikeres! A teszt létező adatokkal sikeres!

UserGetPostTest: (Poszt lekérése a felhasználó id alapján)

```
[Test]
Tabnine | ● | 0 references
public void UserGetPostTest()
{
    int postid = 31;
    RestRequest request = new RestRequest("https://localhost:7043/User/UserGetPost", Method.Get);
    request.AddParameter("id", postid);
    RestResponse response = client.Execute(request);
    response.ShouldNotBeNull();
    response.StatusCode.ShouldBe(HttpStatusCode.OK);
    Console.WriteLine($"Kapott eredmény: {response.Content}");
}
```

Várt eredmény: létező felhasználó id-t megadva a 200-as státuszkóddal tér vissza a backend, és visszaadja a felhazsnáló által posztolt posztokat, vagy ha nem posztolt akkor a „null” értéket. A teszt sikeres!

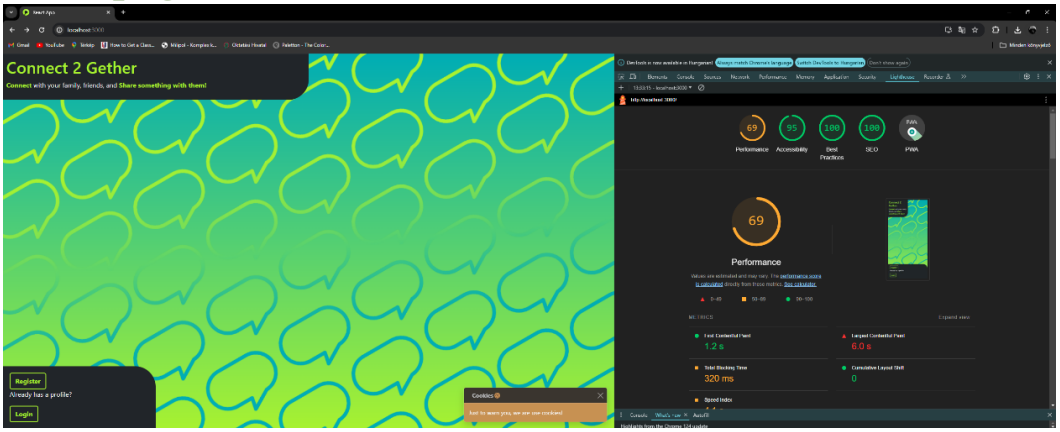
GetUserByIdTest: (Felhasználó lekérdezése ID alapján)

```
[Test]
Tabnine | ● | 0 references
public void GetUserByIdTest()
{
    RestRequest request = new RestRequest("https://localhost:7043/Auth/login", Method.Post);
    request.AddBody(new LoginRequestDto
    {
        UserName = "bazsi",
        Password = "Valami123>"
    }, ContentType.Json);

    RestResponse response = client.Execute(request);

    int userId = 16;
    RestRequest request2 = new RestRequest("https://localhost:7043/User/UserById", Method.Get);
    request.AddHeader("Authorization", "Bearer " + response.Content );
    Console.WriteLine(token);
    request.AddParameter("id", userId);
    RestResponse response2 = client.Execute(request2);
    response.ShouldNotBeNull();
    response.StatusCode.ShouldBe(HttpStatusCode.OK);
    Console.WriteLine($"Kapott eredmény: {response2.Content}");
}
```

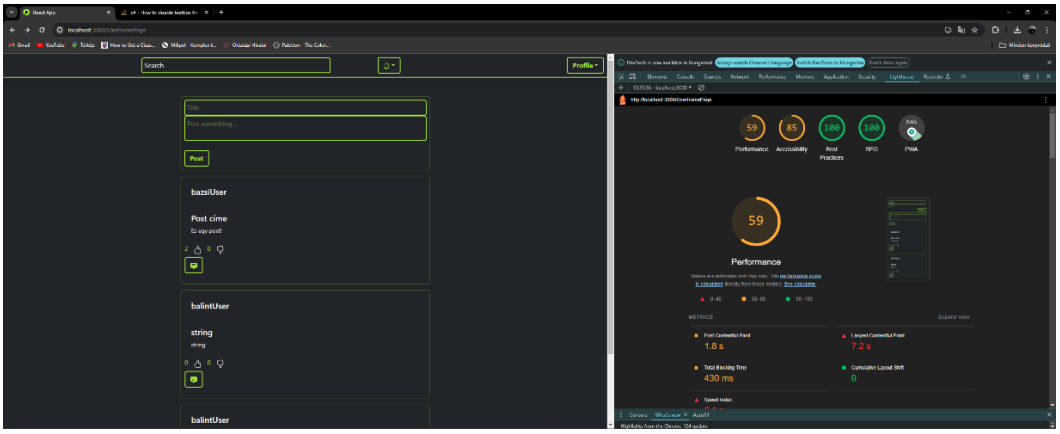
Várt eredmény:létező felhasználó id-t megadva a backend 200-as státuszkóddal tér vissza, viszaadja a felhasználót. A teszt sikeres!

LIGHTHOUSE

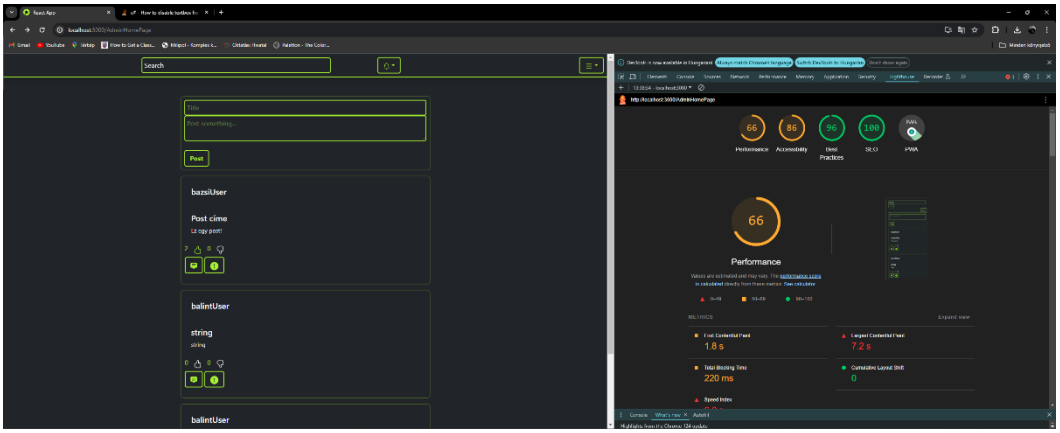
Homepage:



User oldal:



Admin oldal:



Forrásjegyzék

1. Téma: C#, webcím: https://learn.microsoft.com/hu-hu/dotnet/csharp/tour-of-csharp/
2. Téma: NUnit Tesztelési Keretrendszer, webcím: https://learn.microsoft.com/hu-hu/dotnet/core/testing/unit-testing-with-nunit
3. Téma: RestRequest, webcím: https://codegym.cc/hu/groups/posts/hu.294.a-rest-attekintese-2-resz-kommunikacio-kliens-es-szerver-kozott
4. Téma: Newtonsoft.Json, webcím: https://csharptutorial.hu/docs/hellovilag-hellocsharp/12-modern-alkalmazasfejlesztes/json-serialization-newtonsoft/
5. Téma: LightHouse, webcím: https://www.torokbalazs.com/blog/lighthouse-a-weboldalak-teljesitmenymerese