

Syntax Quest

Készítette:

Tóth Viktor

Kiss Gergő

Kutasi Erzsébet

Témaválasztás indoklása

1. Játékmenet és Cél:

- Böngészős rogue-like játék.
- A játékosnak túl kell élnie különböző pályákon.
- Cél: Pontok gyűjtése.

2. Karakterfejlődés:

- Tapasztalatspontok és tárgyak gyűjtése.
- Új képességek és felszerelések megszerzése.

3. Ellenségek és Küzdelmek:

- Különböző ellenségekkel való találkozás.
- Ellenségek legyőzése tapasztalatspontok és tárgyakért.
- Az ellenségek idővel erősebbet sebeznek és gyorsabbak.

4. Taktikai Döntések:

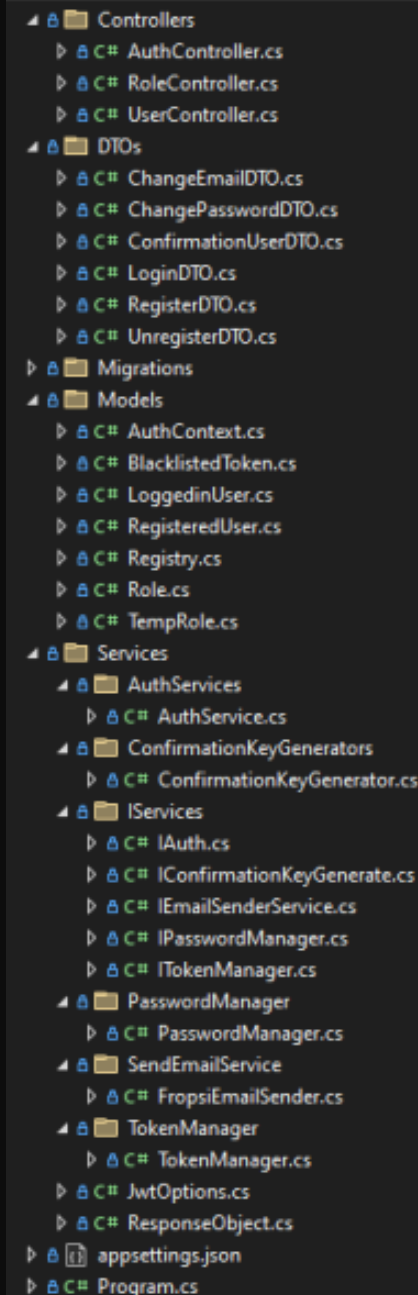
- Támadás vagy menekülés választása.
- Életelixír felvétele, amit az ellenség eldob.

5. Pontgyűjtés:

- A cél a legtöbb pont összegyűjtése.

Backend

- A backend alapját az Asp.Net Core API keretrendszer adja, törekedtünk a lehető legfrissebb sdk használatára, ezért 8.0 lett használva.
-



```
└─ Controllers
   ├── AuthController.cs
   ├── RoleController.cs
   └── UserController.cs
└─ DTOs
   ├── ChangeEmailDTO.cs
   ├── ChangePasswordDTO.cs
   ├── ConfirmationUserDTO.cs
   ├── LoginDTO.cs
   ├── RegisterDTO.cs
   └── UnregisterDTO.cs
└─ Migrations
└─ Models
   ├── AuthContext.cs
   ├── BlacklistedToken.cs
   ├── LoggedinUser.cs
   ├── RegisteredUser.cs
   ├── Registry.cs
   ├── Role.cs
   └── TempRole.cs
└─ Services
   ├── AuthServices
   │   └── AuthService.cs
   ├── ConfirmationKeyGenerators
   │   └── ConfirmationKeyGenerator.cs
   ├── IServices
   │   ├── IAuth.cs
   │   ├── IConfirmationKeyGenerate.cs
   │   ├── IEmailSenderService.cs
   │   ├── IPasswordManager.cs
   │   └── ITokenManager.cs
   ├── PasswordManager
   │   └── PasswordManager.cs
   ├── SendEmailService
   │   └── FropseEmailSender.cs
   ├── TokenManager
   │   └── TokenManager.cs
   ├── JwtOptions.cs
   └── ResponseObject.cs
appsettings.json
Program.cs
```

Swaggerek

Achievement

GET	/Game/achievements	✓	🔒
GET	/Game/achievements/user	✓	🔒
POST	/Game/addAchievement/user	✓	🔒
POST	/Game/createAchievement		
PUT	/Game/updateAchievement		
DELETE	/Game/deleteAchievement		

Auth

POST	/Auth/register	✓	
POST	/Auth/login	✓	
POST	/Auth/confirmAccount	✓	🔒
GET	/Auth/keyValidate	✓	🔒
GET	/Auth/tempusers		
PUT	/Auth/logout		
DELETE	/Auth/unregister		
DELETE	/Auth/deleteUser		

Game

GET	/Game/getStats/user	✓	🔒
GET	/Game/getTopPlayers	✓	🔒
PUT	/Game/resetAccount	✓	🔒
PUT	/Game/updateAccountStats	✓	🔒
PUT	/Game/adminUpdateAccountStats	✓	🔒

Role

GET	/Role/roles	✓	🔒
POST	/Role/createRole	✓	🔒
PUT	/Role/assignrole/user	✓	🔒
PUT	/Role/updateRole	✓	🔒
DELETE	/Role/deleteRole	✓	🔒

User

GET	/User/users	✓	🔒
GET	/User/users/user	✓	🔒
GET	/User/users/status	✓	🔒
PUT	/User/resetPasswordRequest	✓	🔒
PUT	/User/resetPassword	✓	🔒
PUT	/User/changePassword	✓	🔒
PUT	/User/changeEmail	✓	🔒
PUT	/User/updateUser	✓	🔒

```
//Bejelentkezés logika
2 references
public async Task<Object> Login(LoginDTO loginDto)
{
    try
    {
        var token = "";
        var gameContext = new GameContext();

        await using (var context = new AuthContext())
        {
            var user = context.RegisteredUsers.FirstOrDefault(user => user.Username == loginDto.Username);

            if (user == null)
            {
                return ResponseObject.create("Hibás felhasználónév, vagy jelszó!", null!, 400);
            }

            if (user!.Hash == null || !BCrypt.Net.BCrypt.Verify(loginDto.Password, user.Hash))
            {
                return ResponseObject.create("Hibás felhasználónév, vagy jelszó!", null!, 400);
            }

            token = _tokenManager.GenerateToken(user);
            var selectedLoggedInUser = context.LoggedInUsers.FirstOrDefault(u => u.UserId == user.UserId);

            if (selectedLoggedInUser != null)
            {
                var oldToken = selectedLoggedInUser.Token;
                _tokenManager.blackListToken(oldToken);
                selectedLoggedInUser.Token = token;
                context.Update(selectedLoggedInUser);
            }
            else
            {
                user.IsLoggedIn = true;
                await context.AddAsync(new LoggedInUser()
                {
                    UserId = user.UserId,
                    Token = token
                });
            }

            var gameUser = gameContext.Users.FirstOrDefault(u => u.Email == user.Email);

            gameUser!.Lastlogin = DateTime.Now;
            gameContext.Update(gameUser);
            gameContext.SaveChanges();

            await context.SaveChangesAsync();
        }

        return ResponseObject.create("Sikeres bejelentkezés", token, 200);
    }
    catch (Exception ex)
    {
        return ResponseObject.create("Hibás felhasználónév, vagy jelszó!", ex.Message, 400);
    }
}
```

```
//Fiók megerősítés ellenőrzés
2 references
public async Task<Object> IsValidKey(string confirmKey)
{
    try
    {
        var context = new AuthContext();

        var keyCheck = context.Registries.FirstOrDefault(key => key.TempConfirmationKey.Equals(confirmKey));

        if (keyCheck == null)
        {
            return ResponseObject.create("Hibás kulcs, vagy nem létező fiók!", 400);
        }

        return ResponseObject.create("A megadott kulcs helyes!", keyCheck, 200);
    }
    catch (Exception ex)
    {
        return ResponseObject.create(ex.Message, 400);
    }
}
```


Frontend

- React.js
- Axios
- React-router-dom
- Canvas
- Bootstrap

```
const handleLeaderboard = async() => {
  try {
    const response = await axios.get(`https://localhost:7096/Game/getTopPlayers?statName=kills&limit=10`, {
      headers: {
        "Content-Type": "application/json",
        "Authorization": `Bearer ${token}`
      }
    });
    console.log(response.data);
    setLeaderboardData(response.data);
  } catch (error) {
    console.error("Error fetching leaderboard:", error);
  }
  setShowLeaderboard(!showLeaderboard);
  setShowStats(false);
  setShowSettings(false);
}
```

```
▼ vizsga
  ▼ src
    > Assets
    > pages
    ▼ render
      JS canvas.js
      JS DmgPopup.js
      JS HPBar.js
      JS LevelUpUI.js
      JS renderers.js
    ▼ system
      > Hooks
      > PassiveItems
      > Pickups
      > Projectiles
      > Weapons
      JS CButton.js
      JS GameStatCard.js
      JS IntervalSpawner.js
      JS Math.js
      JS Player.js
      JS Slime.js
      JS Spawner.js
      JS StatCard.js
      JS StoneWall.js
      JS Tile.js
      JS App.js
      JS AuthPage.js
      JS index.js
      JS menu.js
```


Bejelentkezés, regisztráció



FELHASZNÁLÓNÉV

JELSZÓ

BEJELENTKEZÉS



FELHASZNÁLÓNÉV

TELJES NÉV

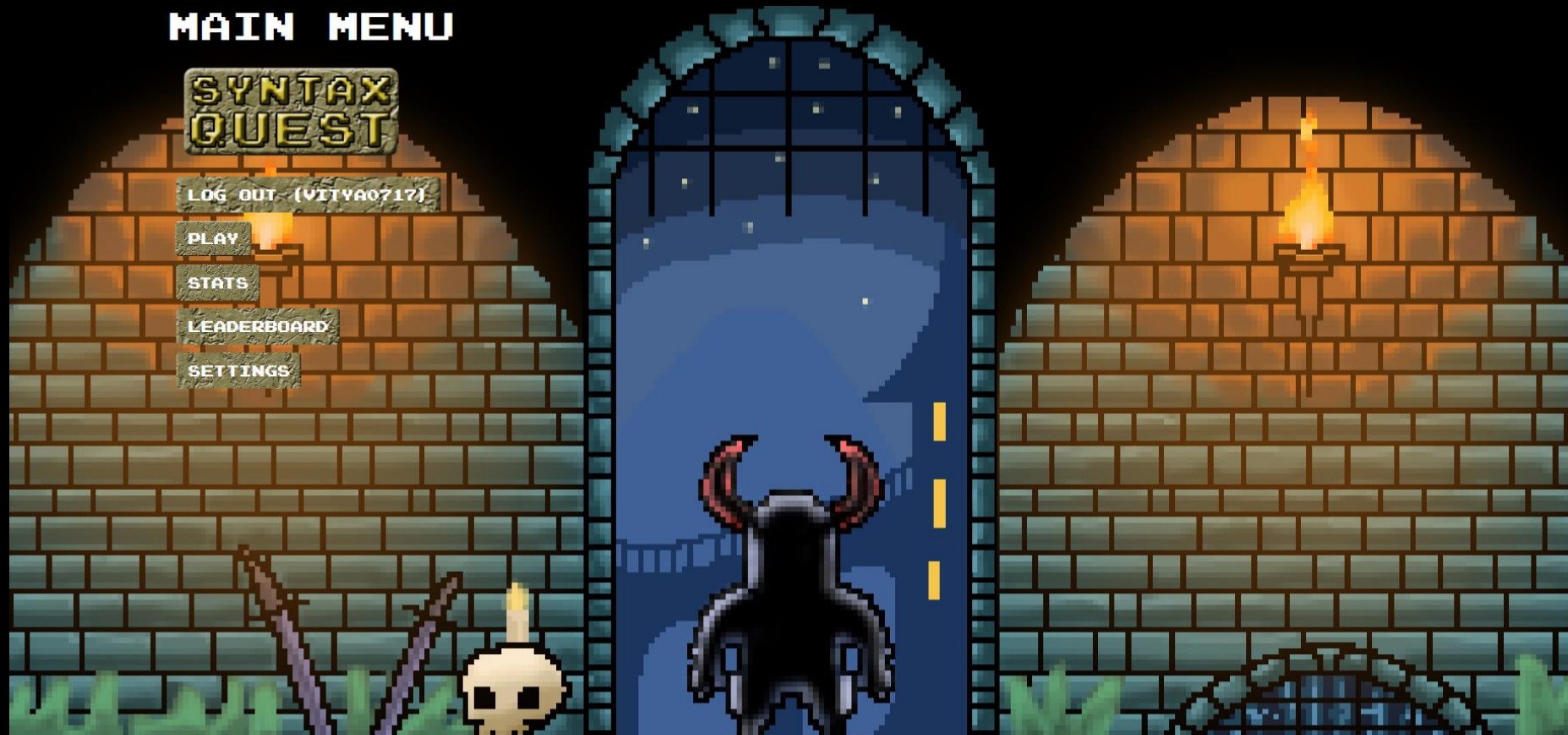
E-MAIL CÍM

JELSZÓ

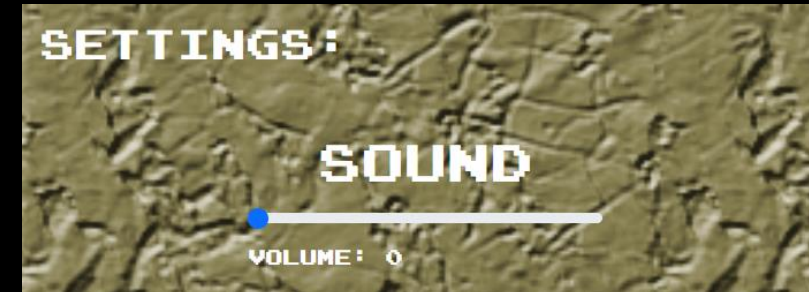
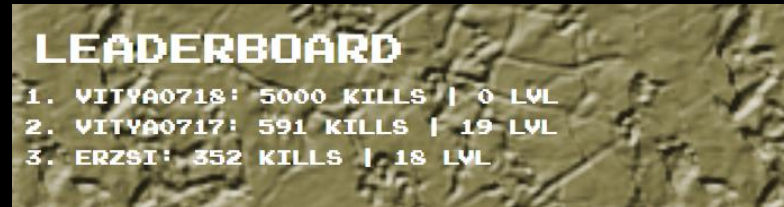
BEJELENTKEZÉS

[MÉG NINCSEK FIÓKOD?](#)

Főmenü



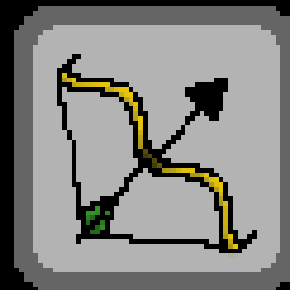
Statisztikák, ranglétra, beállítások



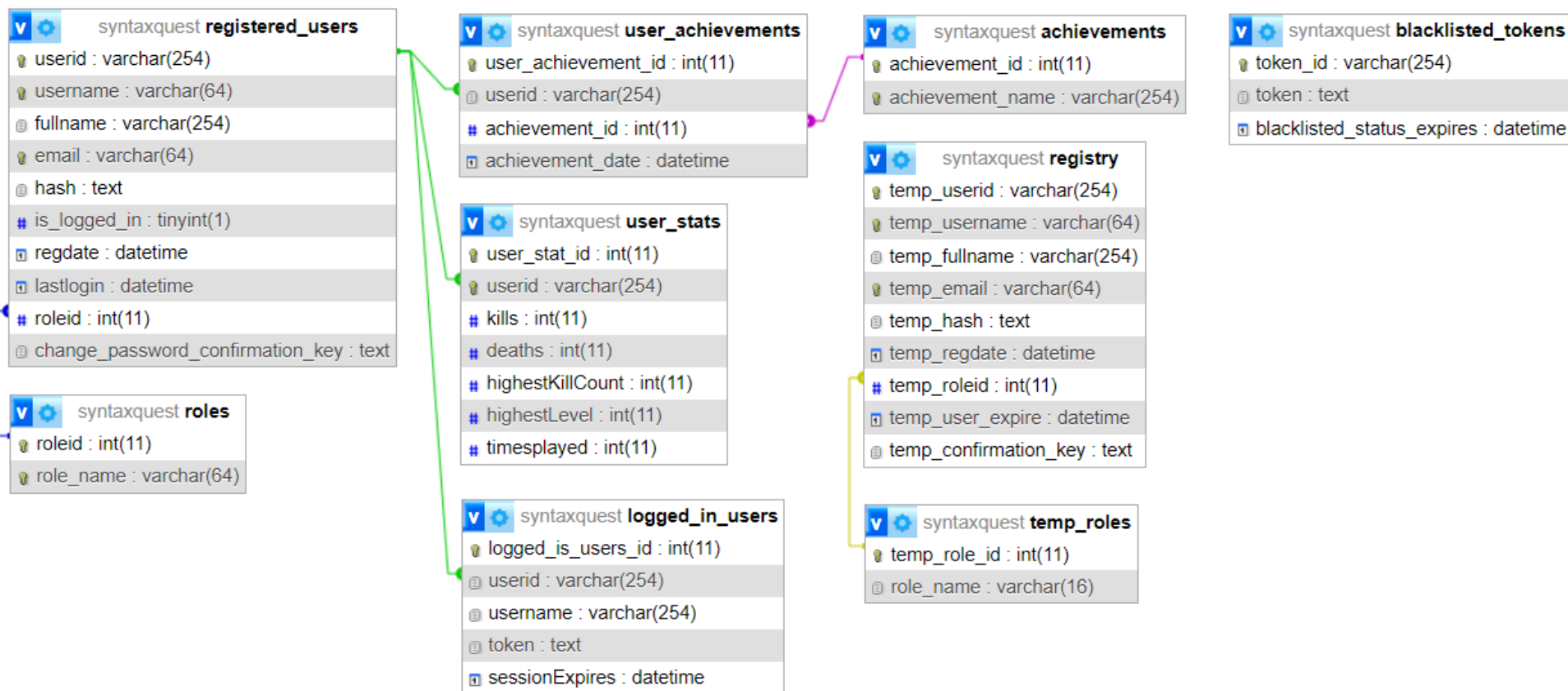
Játék



Játékelemek



Adatbázis



Köszönjük a figyelmet!

A thick, hand-drawn style orange line underlining the text.