

**Szoftverfejlesztő és tesztelő technikus**

Készítette: Nyitribusz Roland, Bíró Patrik, Kovács Gergő

2025

**Tartalomjegyzék**

A cég adatai: 14

A cég tagjai: 14

A cég leírása: 14

Hogyan kerültünk kapcsolatba a céggel? 14

A cég kívánságai: 14

Online Foglalási és Regisztrációs Rendszer 14

Feladatok: 14

Lőtér Kezelő Rendszer (Adminisztrációs) 15

Feladatok: 15

Lövészeti Tréningek és Oktatások 15

Feladatok: 15

A program felépítése, műszaki megvalósítása: 15

Bevezetés: 15

A program bemenetei: 15

Ezek az alábbiak: 15

1. Felhasználói bemenetek: 15

1.1. Felhasználó foglalási adatai: 15

1.2. Foglalási információk: 15

1.3. Eszközkérések: 16

A program kimenetei: 16

1. Felhasználói kimenetek: 16

1.1. Foglalási visszaigazolások: 16

Időpont megerősítése: 16

Az oldal visszajelzést küld ha a foglalás sikeres 16

Adminisztrátori kimenetek: 16

Az adminisztrátorok és dolgozók számára a program az alábbi kimeneteket biztosítja: 16

1.2. Foglalások és események kezelése: 16

Foglalások listája: 16

Hibalehetőségek és kiküszöbölése: 17

1.1. Felhasználói adatok helytelen megadása: 17

Hibalehetőség: 17

Kiküszöbölés: 17

Input validáció: 17

Felhasználói visszajelzések: 17

1.2. Foglalási ütközések: 17

Hibalehetőség: 17

Kiküszöbölés: 17

Foglalási rendszer valós idejű frissítése: 17

18

Az oldal bemutatása 18

A program működésének bemutatása 19

Frontend program bemutatása: 19

Index.html 19

Kapcsolat.hmtl 22

Technikai dokumentáció jellegű leírás: 22

jelentkezes html 22

Indopont.html 23

Admin.html: 25

Technikai dokumentáció jellegű leírás: Élménycsomag 28

1. Használt technológiák: 28

HTML5: 28

CSS3: 28

Reszponzív design: 28

Képek és animációk: 28

2. Oldalszerkezet és fájlok 28

Fő fájl: 28

További fájlok (kapcsolódó menüpontok alapján): 28

3. HTML felépítés 29

A weboldal fő HTML tagjei: 29

4. Stílus (CSS) 29

Színvilág: 29

Háttér: narancs-barna színátmenet 29

Csomagnevek: orangered 29

Menülinkek: narancs → piros hover-effekt 29

Vonal animáció: fehér → narancs színű animált vízszintes vonal 29

Elrendezés 29

Nagvigáció 29

Animáció 29

5. Csomagok tartalma 30

6. Felhasználói élmény 30

Oldal célja: 30

Technikai dokumentáció jellegű leírás: Időpont 31

1. Felépítés HTML struktúra 31

Alap HTML felépítés: 31

Főbb szakaszok: 31

A form mezői: 31

2. Megjelenés - CSS stílusok 32

Alap színek: 32

Elrendezés: 32

Menü: 32

3. Működés - JavaScript használata (merged.js) 32

Fő feladatai: 32

4. Fájlstruktúra 33

Backend 33

Frontend 34

Oldal célja 35

Technikai dokumentáció jellegű leírás: Jelentkezés Oktatónak – Webes űrlapfelület 35

1. Főbb technológiák 35

HTML Részletezés 35

1. Navigációs sáv (<nav>) 35

Mezők: 35

Gomb: 36

CSS Részletezés 36

1. Layout 36

2. Színezés 36

3. Reszponzív elemek 36

4. Hover effektek 37

Oldal célja 37

Technikai dokumentáció jellegű leírás: Kapcsolat 37

1. Főbb technológiák 37

HTML5-Tartalomszervezés és oldalstruktúra 37

CSS3-Megjelenés, animációk, layout 37

Flexbox-Rugalmas elrendezés az oldalon 37

Inline CSS-A kapcsolat.css fájl tartalma a HTML végébe lett ágyazva 37

HTML Részletezés 38

1. Fejléc (<header> + <nav>) 38

2. Fő tartalom 38

3. Lábléc (<footer>) 38

CSS Részletezés 39

Alapbeállítások 39

Navigáció 39

Elválasztó vonal animáció 39

Oldal célja 40

1. Technológiai alapok 40

2. Felépítés 40

Fejléc (<header>) 40

Fő tartalom (<main>) 41

Lábjegyzet (<footer>) 41

3. Funkciók 41

4. Design és stílus 42

5. JavaScript működés – GYIK 42

Technikai dokumentáció jellegű leírás: 42

1. Technológiai alapok 42

2. Fő funkciók 43

Csomagok és Oktatók betöltése 43

Példa: 43

Dropdown mezők feltöltése 43

Adatok beküldése az űrlapról 43

Példa JSON, amit a szerver felé küld: 44

API Végpontok 44

Hibakezelés 45

Kódfolyamat áttekintés 45

Oldal célja 46

Technikai dokumentáció jellegű leírás: 46

Alapvető felépítés 46

Oldalszerkezet részletezése 47

1. Fejléc és Navigáció 47

2. Főtartalom: Nyitvatartás 47

3. Lábléc 47

Stílus (CSS) részletezése 48

1. Háttér és elrendezés 48

2. Táblázat stílus 48

3. Navigációs menü 49

Oldal célja 50

Technikai dokumentáció jellegű leírás: 50

Alkalmazás célja 50

Szerkezeti felépítés 50

1. HTML (Felhasználói felület) 50

2. CSS (Megjelenés) 50

3. JavaScript (Funkcionalitás) 51

Az űrlap mezői 51

Interaktív elemek 52

Design jellemzők 52

Biztonság és validáció 52

Oldal célja 52

Technikai dokumentáció jellegű leírás: 52

Technológiák 52

Fő funkciók áttekintése 53

Strukturális Felépítés 53

Főbb komponensek 54

API Kapcsolatok 54

Formában gyűjtött adatok 54

Elérhetőség ellenőrzés logikája 55

Beküldési folyamat 55

Hibakezelés 56

Interfész viselkedés 56

Oldal célja 56

Backend program bemutatása: 56

AppointmentController.cs – Részletes Dokumentáció 56

Áttekintés 57

Namespace és Osztály 57

Függőségek 57

Konstruktor 57

Metódusok részletesen 57

1. GET: api/Appointment 57

2. GET: api/Appointment/{id} 58

3. POST: api/Appointment 58

4. PUT: api/Appointment/{id} 58

5. DELETE: api/Appointment/{id} 59

Segédfüggvény 59

Adatbázis kontextus (VizsgaremekContext) 60

Összefoglalás 60

InstructorsController.cs – Részletes dokumentáció 61

Alapbeállítások 62

Konstruktor és kontextus 62

Metódusok – CRUD részletesen 63

1. GET: api/instructors 63

4. PUT: api/instructors/{id} 64

5. DELETE: api/instructors/{id} 64

Segédfüggvény 65

Összegzés 65

PackagesController.cs – Részletes dokumentáció 66

Áttekintés 66

CRUD Műveletek 67

GET: api/packages – Összes csomag lekérése 67

GET: api/packages/{id} – Egy csomag lekérése ID alapján 67

POST: api/packages – Új csomag létrehozása 67

PUT: api/packages/{id} – Csomag módosítása 68

DELETE: api/packages/{id} – Csomag törlése 68

Összefoglaló táblázat 69

RegistrationsController.cs – Részletes dokumentáció 70

Áttekintés 70

Konstrukció 70

Konstruktor 70

CRUD műveletek 71

GET: api/registrations – Összes regisztráció lekérdezése 71

GET: api/registrations/{id} – Egy regisztráció lekérdezése 71

POST: api/registrations – Új regisztráció létrehozása 71

PUT:Regisztráció frissítése 72

DELETE:Regisztráció törlése 72

Segédfüggvény: Regisztráció létezik-e 72

Submint.cs – Részletes dokumentáció 73

Fő funkciója: 73

Főbb részei: 73

1. Attribútumok és útvonal 73

2. Konstruktor – Adatbázis kontextus injektálása 73

3. POST metódus – Foglalás mentése 73

4. Érvényesítés és adatmentés 74

5. Hibakezelés 75

Összegezve 75

Program.cs – Részletes dokumentáció 75

A program célja: 75

Feladata: 75

API végpontok: 76

Alkalmazás indítása 76

Az adatbázis adatbázismodell-diagramja 76

Adatbázismodell összefoglaló 76

Webes felület 77

Fejlesztési lehetőségek 77

A program használatának rövid bemutatása 77

Időpontfoglalás: 77

Oktatói jelentkezés: 77

Admin bejelentkezés: 77

A program melyik részével találkozik a tulajdonos, és melyik részével a felhasználó. 77

Tulaj (Admin) részei: 77

Felhasználó részei: 77

Használati útmutató mindkettőnek 78

Felhasználók számára: 78

Leendő oktatók számára: 78

Adminisztrátorok számára: 79

Tesztelés 80

Tesztelési szempontok 80

1. Használhatóság és navigáció 80

2. Teljesítmény és sebesség 80

A tesztekhez végzett kód 81

Teszteredmények dokumentációja 81

A fejlesztés során használt projektszervezési eszközök 84

Melléklet 84

A program forráskódja 84

Források/linkek 85

# 

ROPAGE - A Lőtér

# A cég adatai:

*A cég neve:* ROPAGE

*A cég telephelye:* 8100, Várpalota, Pipacs utca 9.

Weboldal: <https://lotervizsgaremek.com>

# A cég tagjai:

* Nyitribusz Roland – Cégtulajdonos
* Kovács Gergő – Alkalmazott/Társtulajdonos
* Bíró Patrik – Alkalmazott

# A cég leírása:

A cégünk magas szinten foglalkozik adatbáziskezeléssel, weboldal készítéssel,

mobil illetve asztali alkalmazások fejlesztésével.

# Hogyan kerültünk kapcsolatba a céggel?

Csapatunkat megkereste egy nemrég indult lőtér tulajdonosa megbízott minket, hogy készítsünk egy weboldalt illetve egy adatbázist, ami tárolja a felhasználók adatait.

# 

# A cég kívánságai:

# Online Foglalási és Regisztrációs Rendszer

Feladatok:

Weboldal integráció, foglalási és regisztrációs funkciók fejlesztése

Felhasználói élmény optimalizálása

# Lőtér Kezelő Rendszer (Adminisztrációs)

Feladatok:

Ügyfél-, és oktatóadatbázis kezelése, kérdések listája

# 

# 

# Lövészeti Tréningek és Oktatások

Feladatok:

Olyan oldal fejlesztése, amely lehetővé teszi a tréningek időpontjainak online keresését és foglalását.

# A program felépítése, műszaki megvalósítása:

# Bevezetés:

A weboldal célja, hogy biztosítja a lőtér működéséhez szükséges feladatok ellátását, beleértve a látogatók kezelését, valamint a felhasználói élmény javítását. A program feladatai közé tartozik a látogatók számára biztosított időpontok és helyek kezelése.

# 

# A program bemenetei:

A lőtér program bemenetei azok az adatok, amelyeket a felhasználók, dolgozók és rendszer többi komponensei biztosítanak a program számára.

Ezek az alábbiak:

1. Felhasználói bemenetek:
   1. Felhasználó foglalási adatai:

A felhasználók számára szükséges az email cím, név és egyéb személyes adatok megadása

* 1. Foglalási információk:

A felhasználóknak meg kell adniuk a kívánt időpontot, a foglalt csomagot és a kívánt

oktatót

* 1. Eszközkérések:

A felhasználó által választott fegyverek , melyek a foglalás részeként kerülnek rögzítésre.

A program kimenetei:

A lőtér program kimenetei azok az információk, amelyek a rendszerből kiáramlanak, és a felhasználók, adminisztrátorok vagy más rendszerek számára hasznosak. A program kimenetei az adatbázisban tárolt információk, illetve külső rendszereknek küldött adatok.

1. Felhasználói kimenetek:

A felhasználói kimenetek azok az információk, amelyeket a rendszer a felhasználók számára biztosít az alkalmazás felületén, emailben vagy egyéb módon.

* 1. Foglalási visszaigazolások:

Időpont megerősítése:

Az oldal visszajelzést küld ha a foglalás sikeres

Adminisztrátori kimenetek:

Az adminisztrátorok és dolgozók számára a program az alábbi kimeneteket biztosítja:

* 1. Foglalások és események kezelése:

Foglalások listája:

Az adminisztrátorok számára elérhető egy részletes lista az összes felhasználói foglalásról, amely tartalmazza az időpontokat, és csomagokat. Az adminisztrátorok ezen a listán keresztül módosíthatják vagy törölhetik a foglalásokat.

## 

# Hibalehetőségek és kiküszöbölése:

A weboldal működtetése során különféle hibalehetőségek merülhetnek fel, amelyek hatással lehetnek a rendszer stabilitására, a felhasználói élményre és a biztonságra:

* 1. Felhasználói adatok helytelen megadása:

Hibalehetőség:

A felhasználók helytelen adatokat adhatnak meg a regisztráció során (pl. érvénytelen email cím), ami problémát okozhat a belépésnél és az adatkezelésnél.

Kiküszöbölés:

Input validáció:

Az összes felhasználói adatot validálni kell a frontend és backend oldalon is. Például, az email cím formátumát és az egyéb kötelező mezőket ellenőrizni kell.

Felhasználói visszajelzések:

Ha a felhasználó hibás adatokat ad meg, akkor hibaüzenetet kap, amely tájékoztatja őt a problémáról (pl. „A megadott email cím érvénytelen!”).

* 1. Foglalási ütközések:

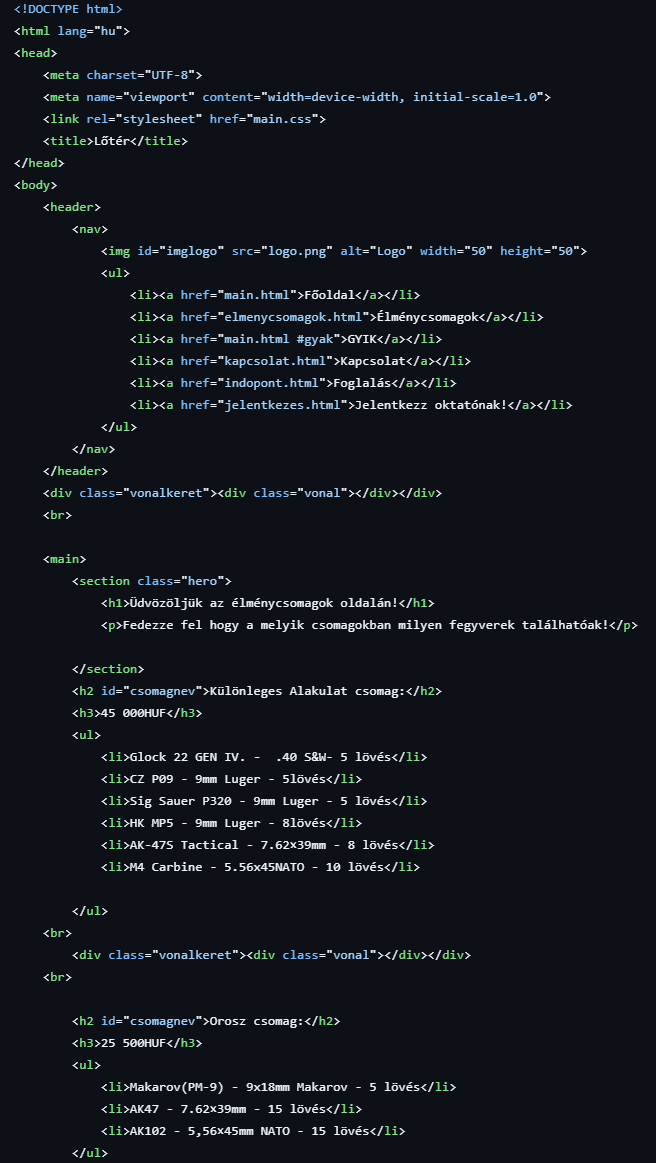
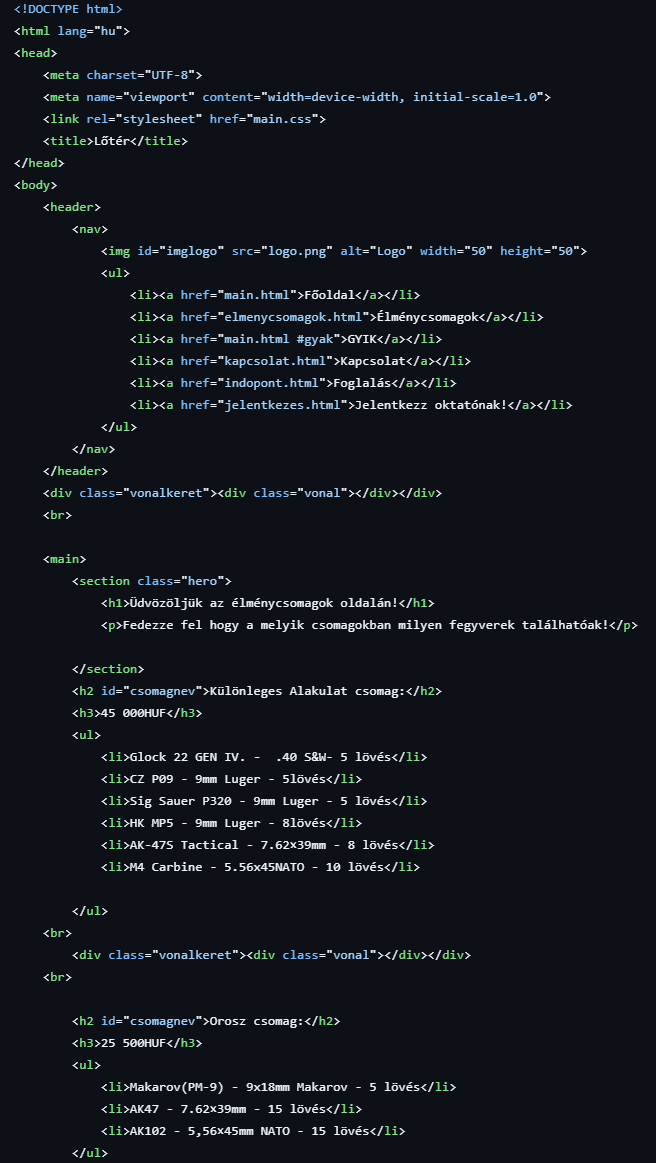
Hibalehetőség:

Ha 7-nél több felhasználó próbálja ugyanazt az időpontot lefoglalni ugyanahhoz az oktatóhoz, miközben az oktató már foglalt, ütközés léphet fel, amely rossz felhasználói élményt eredményezhet.

Kiküszöbölés:

Foglalási rendszer valós idejű frissítése:

A foglalási rendszer folyamatosan frissítse a foglalások számát. Ha egy időpont már betelt, azt azonnal jelezze a felhasználónak.



# Az oldal bemutatása

Az élménylövészet programunk lehetőséget nyújt bárki számára, hogy biztonságos és ellenőrzött környezetben próbálhasson ki különböző típusú fegyvereket. A program során valódi, ismert kézi- és automata lőfegyverekkel ismerkedhetnek meg a résztvevők, tapasztalt oktatók felügyelete mellett.

A csomagok úgy lettek kialakítva, hogy mindenki megtalálja a számára izgalmas élményt – legyen szó első alkalommal érkezőkről, fegyverrajongókról vagy céges csapatépítőkről. Minden résztvevő rövid biztonsági oktatásban részesül a lövészet előtt, így semmilyen előképzettségre vagy fegyvertartási engedélyre nincs szükség.

Weboldalunkon a felhasználók tájékoztatót találnak a csomagokban szereplő fegyverekről és kalibereikről, sőt online foglalásra is van lehetőség. A helyszínen egyedi csomag is összeállítható, amelyhez helyben árajánlatot biztosítunk.

# 

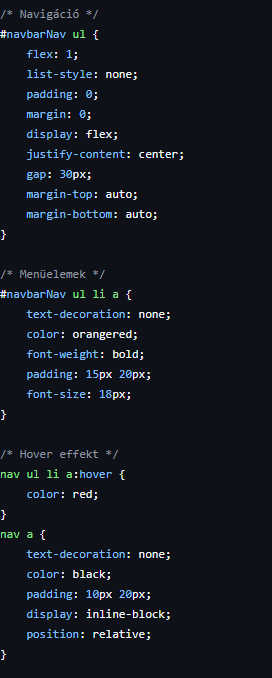
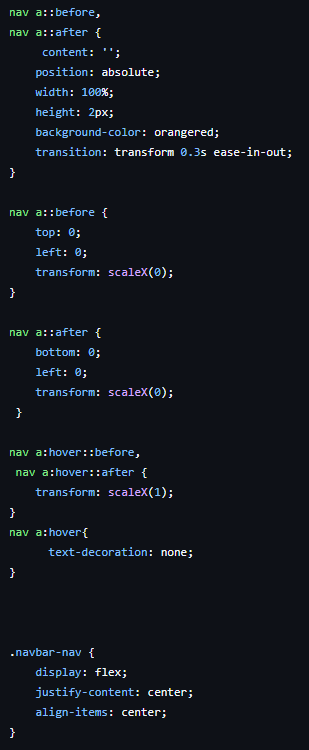
# A program működésének bemutatása

Frontend program bemutatása:

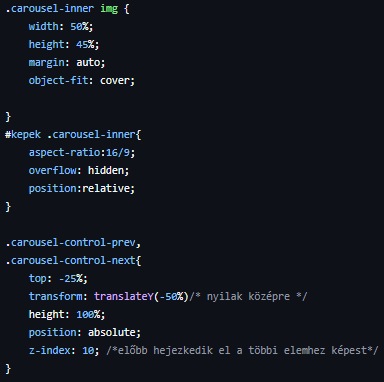
# 

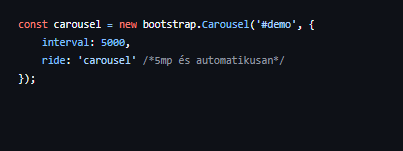
# Index.html

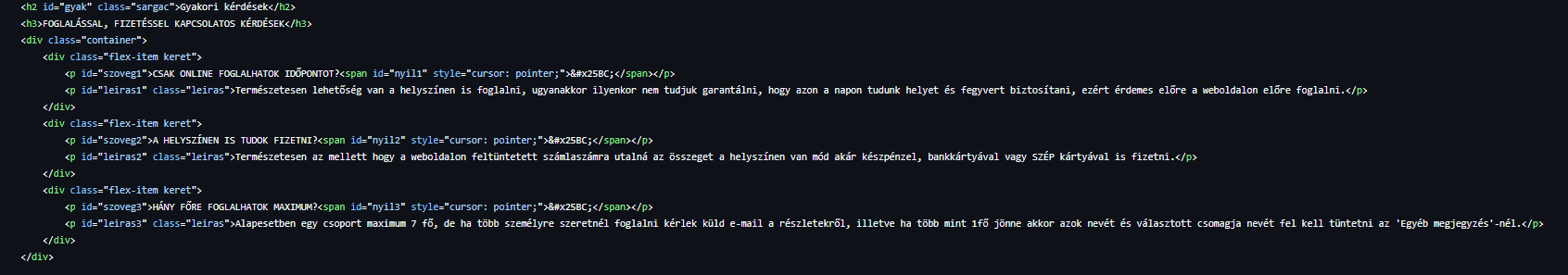
Nav-bár menü HTML és CSS kódja:

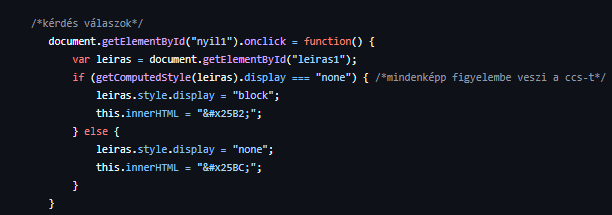


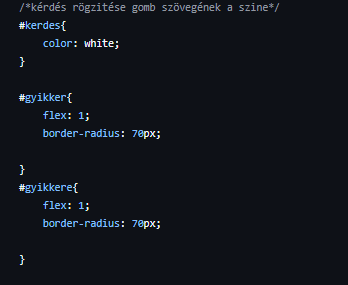
A képek automatikus váltakozása:





Gyakran Ismételt kérdések:





Kapcsolat.

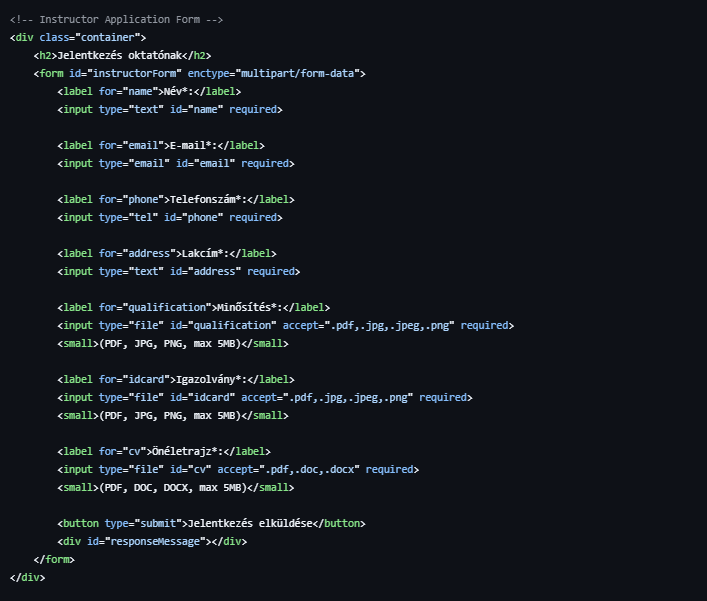
# Kapcsolat.hmtl

# Technikai dokumentáció jellegű leírás:

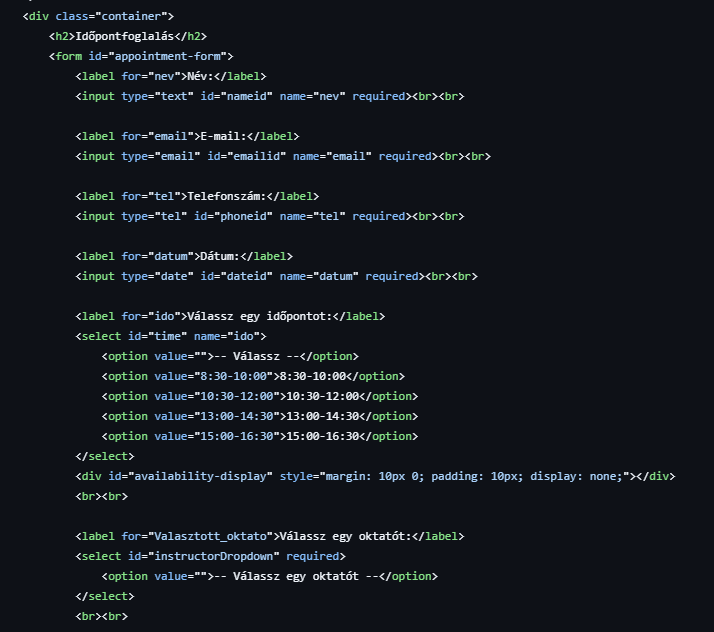
A kapcsolat menüpontnál megtalálható egy térkép a lőtér címével.

# jelentkezes html

Lehet jelentkezni oktatónak is amihez az alábbi adatok kellenek:



# Indopont.html

Itt lehet időpontot foglalni

Ki lehet választani az oktatók és a csomagot

Admin.html:

Itt látható az időpontok táblázata ezeket az adatok kell kitölteni





Akik el lettek fogadva, azok bekerülnek a felvett oktatók közé



Feltett kérdéseket tartalmazó táblázat



Ha az oktató jelszót szeretne cserélni, akkor azt itt tudja megtenni.

# Technikai dokumentáció jellegű leírás: Élménycsomag

# 1. Használt technológiák:

HTML5:

a weboldal alapstruktúrájának létrehozásához (fejléc, menüpontok, tartalmi szekciók, lábléc).

CSS3:

a weboldal megjelenésének (színek, elrendezés, tipográfia, animációk) kialakításához.

Reszponzív design:

a viewport meta tag segítségével biztosítja a mobilbarát megjelenés

Képek és animációk:

A vizuális élmény fokozására.

# 2. Oldalszerkezet és fájlok

Fő fájl:

index.html – Főoldal

További fájlok (kapcsolódó menüpontok alapján):

elmenycsomagok.html – Az élménycsomagokat bemutató oldal, amely a <main> szakaszban listázza a csomagokat.

kapcsolat.html – Kapcsolatfelvételi oldal

indopont.html – Időpontfoglalás

jelentkezes.html – Oktatói jelentkezés

logo.png – A navigációs menüben megjelenő logó

style.css – A külső stíluslap (alternatív vagy kiegészítő)

# 

# 3. HTML felépítés

A weboldal fő HTML tagjei:

<header>: Navigációs menü, logóval.

<main>: Az oldal fő tartalma – csomagok listázása.

<ul>: Az egyes fegyverek felsorolása csomagonként.

<footer>: Lábléc információ (szerzői jog).

Egyedi elemek: div.vonalkeret > div.vonal – animált szétválasztó vonalak csomagok között.

# 4. Stílus (CSS)

Színvilág:

Háttér: narancs-barna színátmenet

Csomagnevek: orangered

Menülinkek: narancs → piros hover-effekt

Vonal animáció: fehér → narancs színű animált vízszintes vonal

Elrendezés

display: flex + flex-direction: column használat a teljes oldal struktúrájában

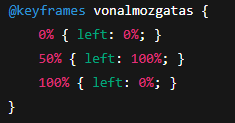
min-height: 100vh biztosítja, hogy a lábléc mindig az oldal alján legyen

Nagvigáció

Dinamikus vonal animáció a menü linkjei alatt (hover effekt)

Középre igazított, sorban elhelyezett menüpontok

Animáció



Ez a kulcsfontosságú animáció biztosítja, hogy a vonal folyamatosan mozgó hatást keltsen a csomagok között

# 

# 5. Csomagok tartalma

Minden csomag felsorolásszerűen bemutatja az elérhető fegyvereket, típusukat, kaliberüket és a lövések számát. Összesen **7 élménycsomag** szerepel, különféle kombinációkkal:

Különleges Alakulat csomag

Orosz csomag

Magyar csomag

9mm-es pisztolycsomag

9mm-es géppisztolycsomag

5.56 NATO & .300 Blackout csomag

7.62-es fegyverek csomag

A leírás alatt extra információk is szerepelnek, pl. lehetőség egyedi csomag összeállítására a helyszínen.

# 6. Felhasználói élmény

Csomagok jól elkülönülnek vizuálisan, így könnyen áttekinthetők

Menüsáv egyszerű, gyors navigációt biztosít

A vonal animáció figyelemfelkeltő, de nem zavaró

# Oldal célja:

A felhasználó egy **modern dizájnnal ellátott weboldalon** keresztül böngészhet a csomagok között, illetve hozzáférhet más fontos oldalakhoz (főoldal, kapcsolat, foglalás, oktatói jelentkezés stb.). Az oldal HTML5 és CSS3 technológiákkal készült, animációval és interaktív effektekkel kiegészítve.

# 

# Technikai dokumentáció jellegű leírás: Időpont

# 1. Felépítés HTML struktúra

# Alap HTML felépítés:

<!DOCTYPE html>

<html lang="hu">

<head> ... </head>

<body> ... </body>

</html>

# Főbb szakaszok:

**<header>** – A navigációs menü és a logó.

**<div class="container">** – Az űrlapot tartalmazó fő tartalmi blokk.

**<form id="appointment-form">** – Maga a foglaló űrlap.

**<footer>** – Lábléc szerzői jogi információval.

# A form mezői:

**Név** (<input type="text">)

**E-mail** (<input type="email">)

**Telefonszám** (<input type="tel">)

**Dátum** (<input type="date">)

**Idősáv** (<select> listából választható)

**Oktató választása** (<select id="instructorDropdown">)

**Csomag választása** (<select id="packageDropdown">)

**Mentés gomb** (<button>)

# 

# 2. Megjelenés - CSS stílusok

A dizájn modern, vizuálisan figyelemfelkeltő, ugyanakkor letisztult:

Alap színek:

**Háttér:** Narancs-barna színátmenetes háttér

**Gomb:** élénk orangered, hover-re piros

**Lábléc/fejléc:** fekete háttér, fehér szöveg

Elrendezés:

**Flexbox** használata a középre igazításhoz.

**.Container** – központi doboz lekerekített sarkokkal, árnyékolással.

**Responsiveness:** az oldal reszponzív alapstruktúrával rendelkezik, de mobiloptimalizálás még bővíthető.

Menü:

**<nav>** menü animált vonalas hover effektel.

**Szélesség, betűméret és padding** beállítva az esztétikus megjelenésért.

# 3. Működés - JavaScript használata (merged.js)

Fő feladatai:

**Oktatók listájának betöltése** az instructorDropdown mezőbe.

**Csomagok dinamikus betöltése** a packageDropdown mezőbe.

**Űrlap validálása** (ellenőrzés, hogy minden mező ki van-e töltve).

**Mentés gombra eseménykezelő** (pl. beküldés naplózása vagy feldolgozása).

**Visszajelzés megjelenítése** a felhasználónak (pl. sikeres foglalás jelzése egy div vagy alert segítségével).

# 

# 4. Fájlstruktúra

# Backend

# ├── Controllers

# │ ├── AppointmentsController.cs

# │ ├── AuthController.cs

# │ ├── InstructorsController.cs

# │ ├── PackagesController.cs

# │ ├── SaveTextController.cs

# │ └── Submit.cs

# ├── Models

# │ ├── Appointment.cs

# │ ├── Instructor.cs

# │ ├── InstructorStatus.cs

# │ ├── Package.cs

# │ ├── Questions.cs

# │ └── VizsgaremeKContext.cs

# ├── Properties

# │ └── launchSettings.json

# ├── Services

# │ ├── EmailService.cs

# │ └── FileUploadService.cs

# ├── bin/Debug/net6.0

# ├── obj

# ├── wwwroot/uploads

# │ ├── cvs

# │ ├── idcards

# │ └── qualifications

# ├── Program.cs

# ├── appsettings.Development.json

# ├── appsettings.json

# ├── loter-HunFuzeMain-patch-2.sin

# └── tempbackend.csproj

# └── tempbackend.csproj.user

# Frontend

# ├── en

# │ ├── eimenycsomagoken.html

# │ ├── indoponten.html

# │ ├── kapcsolaten.html

# │ ├── mainen.html

# │ └── nyrivalartasen.html

# ├── images

# │ ├── angol.jpg

# │ ├── favicon.ico

# │ ├── kep1.jpg

# │ ├── kep2.jpg

# │ ├── kep3.jpg

# │ ├── logo.png

# │ └── magyar.jpg

# ├── js

# │ ├── gyk.js

# │ ├── instructor.js

# │ └── mergeal.js

# ├── styles

# │ └── style.css

# ├── admin.html

# ├── bejelentkezes.html

# ├── bootstrap.min.css

# ├── bootstrap.min.js

# ├── eimenycsomagok.html

# ├── index.html

# ├── indopont.html

# ├── jelentkezes.html

# ├── jquery.min.js

# ├── kapcsolat.html

# └── nyrivalartas.html

# Oldal célja

Ez a program a lövészeti élményeket kínáló szolgáltatáshoz készült időpontfoglaló rendszer frontendje. A cél egy felhasználóbarát, vizuálisan igényes felület létrehozása volt, ahol az érdeklődők könnyedén foglalhatnak időpontot különböző csomagokra és választhatnak oktatót is.

# Technikai dokumentáció jellegű leírás: Jelentkezés Oktatónak – Webes űrlapfelület

# 1. Főbb technológiák

**HTML5** – szerkezeti elemek létrehozása

**CSS3** – formázás, dizájn

**Reszponzív dizájn** – minden eszközön jól működik (desktop/mobil)

# 

# HTML Részletezés

# 1. Navigációs sáv (<nav>)

Tartalmaz 6 menüpontot

Logó bal oldalon (#imglogo)

Hover effekt és animáció CSS-sel megvalósítva (::before, ::after)

# 

Mezők:

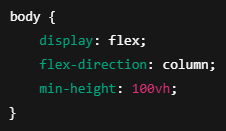
| ID | Típus | Kötelező | Megjegyzés |
| --- | --- | --- | --- |
| nev | text |  | Oktató neve |
| email | email |  | Validált e-mail cím |
| tel | tel |  | Telefonszám |
| lakcim | text |  | Oktató állandó lakcíme |
| minosites | text |  | Oktatói minősítés megnevezése |
| minositesideje | date |  | Minősítés megszerzésének dátuma |

Gomb:



# CSS Részletezés

# 1. Layout



# 2. Színezés

Háttér: linear-gradient(to right, #87574a, #b46324)

Form mezők háttere: sötét szürke doboz (rgb(75, 75, 75))

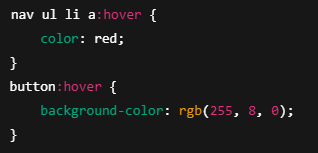
Gomb: orangered, hover-rel sötétedő effektus

# 3. Reszponzív elemek

Navigációs menü: flex, középre igazítva, szakadó távolságokkal

Container (űrlapdoboz) szélessége: 50%, középre zárva, box-shadow-val

# 4. Hover effektek



# Oldal célja

Ez az oldal egy webalapú űrlapot biztosít olyan személyek számára, akik oktatóként szeretnének csatlakozni egy szervezethez (pl. lőtér). A jelentkezési adatok továbbításra kerülnek a backendhez, ahol további feldolgozás történhet (pl. adatbázisba mentés, visszaigazoló e-mail küldése).

# Technikai dokumentáció jellegű leírás: Kapcsolat

# 1. Főbb technológiák

HTML5-Tartalomszervezés és oldalstruktúra

CSS3-Megjelenés, animációk, layout

Flexbox-Rugalmas elrendezés az oldalon

Inline CSS-A kapcsolat.css fájl tartalma a HTML végébe lett ágyazva

# 

# HTML Részletezés

# 1. Fejléc (<header> + <nav>)

-Weboldal navigációs sávja

-6 fő menüpont:

Főoldal

Élménycsomagok

GYIK

Kapcsolat (ez az oldal)

Foglalás

Jelentkezés oktatónak

-Logó képpel (logo.png)

# 2. Fő tartalom

-Cím: **"Üdvözöljük a kapcsolat oldalon!"**

-Három személy elérhetősége:

**Név, telefonszám, email**

-Céges információ:

Telephely címe

Telefonszám

E-mail cím

-Elválasztók: minden blokkot animált vonal tagol (.vonalkeret és .vonal)

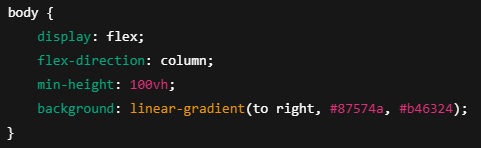
# 3. Lábléc (<footer>)

Egyszerű szöveges lábléc, fekete háttérrel

Tartalma: „© 2025 Lőtér. Minden jog fenntartva.”

# CSS Részletezés

Alapbeállítások



Flexbox segítségével az oldal teljes képernyőt kitölti

Narancs-barna színátmenetes háttér

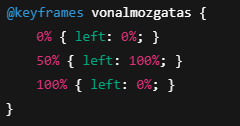
Navigáció

Flexbox elrendezés

Narancssárga menüpontok, hover állapotban pirosra váltanak

:before és ::after pszeudo-elemekkel animált alsó és felső vonalak a menüpontoknál

Elválasztó vonal animáció



A .vonal osztály segítségével egy jobbra-balra mozgó animáció jelenik meg minden szekció között

# Oldal célja

A *Kapcsolat oldal* célja, hogy a látogatók egyszerűen és átláthatóan megtalálják a weboldalt működtető személyek és a cég hivatalos elérhetőségeit. Az oldal részeként bemutatásra kerül három kapcsolattartó (névvel, telefonszámmal, e-mail címmel és képpel), valamint a lőtér hivatalos telephelyének adatai.

A felhasználói élményt tovább növeli a letisztult dizájn, az animált elválasztó vonalak és az egységes megjelenés. Az oldal célja az átláthatóság, bizalomépítés és könnyű kommunikáció biztosítása az érdeklődők számára.

# 1. Technológiai alapok

**HTML5** – A weboldal vázának kialakításához.

**CSS3** – A vizuális megjelenésért, stílusért.

**JavaScript (vanilla)** – A gyakori kérdések interaktív kezeléséhez.

**Bootstrap 4** – A reszponzív dizájn és carousel elemek támogatásához.

# 2. Felépítés

# Fejléc (<header>)

Tartalmaz egy **navigációs menüt** és egy **logót**.

Menü elemei: Főoldal, Élménycsomagok, GYIK, Kapcsolat, Nyitvatartás, Foglalás, Jelentkezés oktatónak.

# Fő tartalom (<main>)

**Hero szekció:** figyelemfelkeltő címsor és egy „Foglaljon most!” gomb.

**Rólunk szekció:** bemutatja a lőtér filozófiáját és hozzáállását a biztonságos lövészethez.

**Szolgáltatások:** felsorolás formájában, szűk, jól olvasható információk.

**GYIK szekció:** interaktív kérdés-válasz blokkok JavaScript segítségével.

**Kapcsolat:** e-mail, telefonszám, cím, valamint beágyazott Google Maps térkép.

# Lábjegyzet (<footer>)

Egyszerű szöveg, copyright információval.

# 

# 3. Funkciók

| Funkció | Leírás |
| --- | --- |
| Képnézegető carousel | Bootstrap alapú, reszponzív képváltás |
| GYIK szekció | JavaScript-tel bővíthető és összecsukható válaszok |
| Foglalás gomb | Linkel a foglalási oldalra |
| Kérdés rögzítése | Felhasználó megadhat saját kérdést (még nem tárolja) |
| Oldal tetejére gomb | Scrollozás után megjelenik, az oldal tetejére visz |
| Mobilbarát dizájn | Bootstrapnek köszönhetően |

# 4. Design és stílus

Háttér: **lineáris színátmenet** barna árnyalatokkal.

Fő színek: narancs, barna, fehér – katonai, outdoor stílushoz illeszkedik.

Gombok, kiemelések: narancssárga és piros árnyalatok.

Egyedi animáció: **vonalmozgás a szekciók között**, vizuálisan feldobja az élményt.

# 5. JavaScript működés – GYIK

Minden kérdés-címhez tartozik egy nyíl (▼ vagy ▲).

Rákattintásra kibontja vagy elrejti a leírást.

Minden elem saját id-t használ a szkriptben.

Az oldal alján lévő „felfelé” gomb szintén JS-sel jelenik meg scroll alapján.

# Technikai dokumentáció jellegű leírás:

# 1. Technológiai alapok

**Frontend**: HTML + Vanilla JavaScript

**Backend API**: Swagger által dokumentált REST API-k:

GET /api/Packages – csomagok lekérése

GET /api/Instructors – oktatók lekérése

POST /api/Submit – időpont adatainak elküldése

# 2. Fő funkciók

# Csomagok és Oktatók betöltése

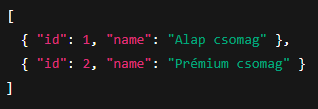
A DOM betöltődése után automatikusan meghívódik két aszinkron függvény:

loadPackages() és loadInstructors()

Ezek fetch() segítségével lekérik az adatokat az API-ból

A válasz JSON formátumban érkezik, majd kitöltik a dropdown mezőket (select elemeket) a kapott adatokkal.

Példa:



# Dropdown mezők feltöltése

A populateDropdown() függvény általánosan használható bármely select mező kitöltésére. Az option elemek értékei az egyes objektumok id tulajdonságai, a szöveg pedig a name mezőből származik.

# Adatok beküldése az űrlapról

Amikor a felhasználó kitölti az űrlapot és rákattint a „Küldés” gombra:

A handleFormSubmit függvény letiltja az alapértelmezett űrlap-küldést (event.preventDefault()).

Ellenőrzi, hogy minden mező ki van-e töltve.

Az időintervallumból ("8:30-10:00") kiválasztja a kezdési időt, és összeállít egy DatumIdo értéket ISO 8601 formátumban pl.: 2025-04-30T08:30:00.

Példa JSON, amit a szerver felé küld:

{

"Nev": "Kiss Péter",

"Email": "peter@gmail.com",

"DatumIdo": "2025-04-30T08:30:00",

"Tel": "+36201234567",

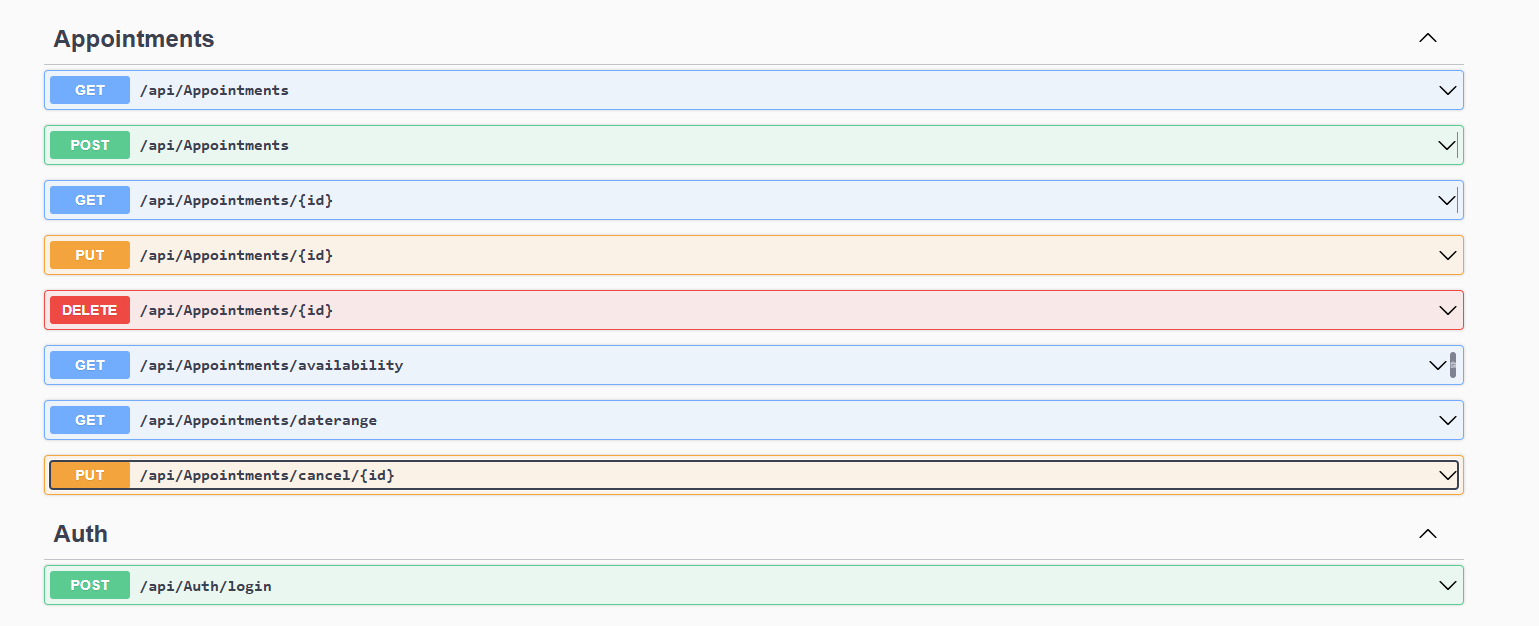
"ValasztottOktatoId": 2,

"ValasztottCsomagId": 1

}

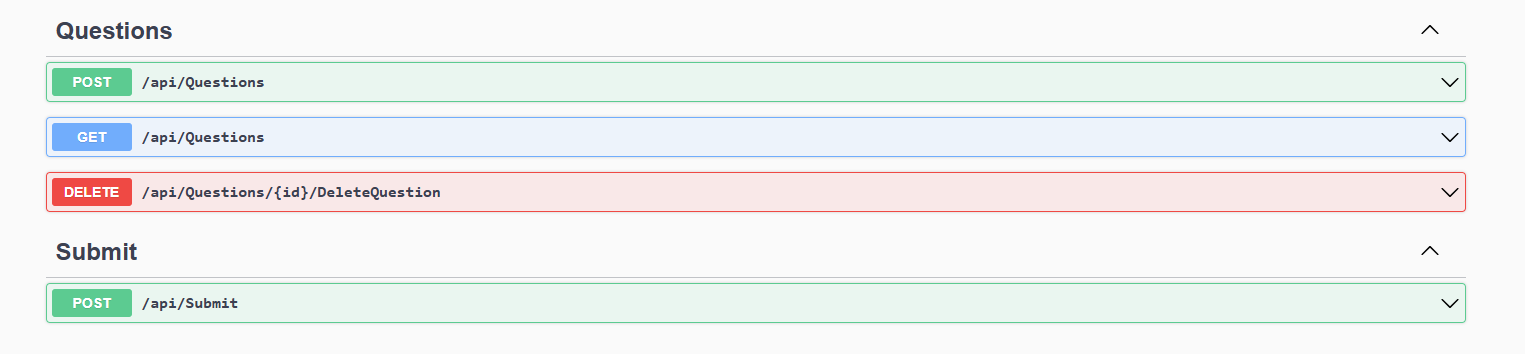
A sendDataToServer() funkció POST kérést küld, és hibakezelést is tartalmaz. Siker esetén megjelenik egy alert: Sikeres időpontfoglalás!, hiba esetén pedig figyelmeztető üzenet.

# API Végpontok



Foglalások (Appointments) és az admin oldal belépésének (Auth) ellenőrzés



Oktatók és csomagok kezelése

# 

# Hibakezelés

A fetch hívások és az adatkiküldés során minden hibát logol a konzolra

# 

# Kódfolyamat áttekintés

1. A DOM betöltése után lefut a fő DOMContentLoaded esemény.

2. Lekéri a csomagokat és oktatókat az API-ból.

3. Kitölti a dropdown mezőket.

4. A felhasználó kitölti az űrlapot.

5. Az adatokat a program validálja és beküldi a szerverre.

6. A válasz alapján értesíti a felhasználót.

# Oldal célja

Ez a webes alkalmazás egy egyszerű, de hatékony **időpontfoglaló rendszer**, amely lehetővé teszi az ügyfelek számára, hogy online foglaljanak időpontot különféle szolgáltatási csomagokra és oktatókhoz. Az alkalmazás három fő komponensből áll:

**Szolgáltatási csomagok lekérdezése**

**Oktatók listázása**

**Időpont foglalása**

A felhasználó egy űrlap segítségével adhatja meg nevét, elérhetőségeit, választhat egy csomagot és egy oktatót, majd megadhatja a kívánt időpontot. Az adatokat a rendszer egy backend API-n keresztül továbbítja, ahol azok feldolgozásra kerülnek.

# Technikai dokumentáció jellegű leírás:

# Alapvető felépítés

A weboldal HTML5 szabványra épül, és a következő fő szekciókból áll:

**Fejléc (<header> és <nav>)** – navigációs menü logóval.

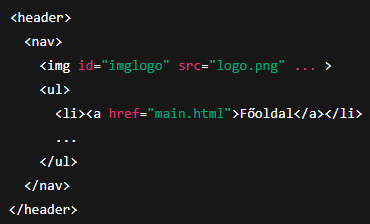
**Főtartalom (<h2>, <h3>, <table>)** – a nyitvatartási információk.

**Lábléc (<footer>)** – jogi nyilatkozat.

**Beágyazott CSS (<style>)** – az oldal stílusának meghatározása.

# Oldalszerkezet részletezése

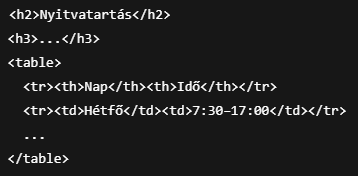
# 1. Fejléc és Navigáció



A navigációban több aloldal is elérhető: pl. „Főoldal”, „Kapcsolat”, „Foglalás”, „Jelentkezés oktatónak”, stb.

A logó (logo.png) megjelenik a menü elején.

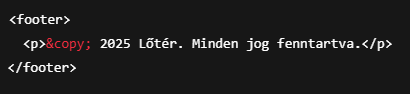
# 2. Főtartalom: Nyitvatartás



A nyitvatartási időket egy táblázatban jeleníti meg.

Minden napra ugyanaz az idősáv van beállítva (7:30–17:00), de ez később igény szerint módosítható.

# 3. Lábléc

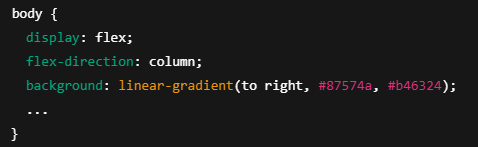


Egyszerű, középre igazított szöveg, amely a szerzői jogokat tünteti fel.

# Stílus (CSS) részletezése

A stílus közvetlenül a HTML-dokumentumba van beágyazva <style> tagben.

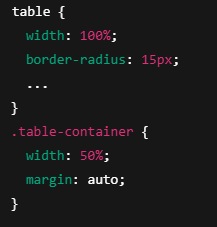
# 1. Háttér és elrendezés



A flexbox segítségével függőlegesen rendezi az oldalt.

Szép színátmenetes háttér teszi vizuálisan vonzóvá az oldalt.

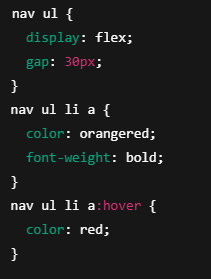
# 2. Táblázat stílus



Kerekített sarkú, árnyékkal ellátott táblázat.

Középre igazított, reszponzív konténerben.

# 3. Navigációs menü



A menüpontok vízszintesen jelennek meg, narancssárga betűszínnel.

Hover-effekt piros színnel.

# Oldal célja

A weboldal ezen aloldala egy **modern, letisztult megjelenésű „Nyitvatartás” oldal**, amely tájékoztatja a látogatókat a lőtér nyitvatartási idejéről. Az oldal célja, hogy a felhasználók gyorsan és egyértelműen információt kapjanak arról, hogy mely napokon és időpontokban érhető el személyesen vagy telefonon a szolgáltató.

A letisztult, mobilbarát megjelenés mellett az oldal tartalmaz:

**egy navigációs menüt** a weboldal többi oldalához,

**egy jól strukturált táblázatot** a heti nyitvatartási időkkel,

**stíluselemeket**, amelyek fokozzák a vizuális élményt és felhasználóbarát megjelenést biztosítanak.

Ez az oldal ideális információs felület minden olyan látogató számára, aki szeretné tudni, mikor látogathatja meg a helyszínt, vagy mikor hívhatja az üzemeltetőt.

# Technikai dokumentáció jellegű leírás:

# Alkalmazás célja

A program célja egy olyan **webalapú űrlap létrehozása**, amely:

lehetővé teszi időpontfoglalás lebonyolítását,

listázza az elérhető oktatókat és csomagokat,

modern, átlátható formában fogadja be a látogatók adatait.

# Szerkezeti felépítés

# 1. HTML (Felhasználói felület)

A struktúra főbb részei:

<header> Navigációs menü és logó

<form> Időpontfoglaló űrlap (Név, E-mail, Telefonszám, Dátum, Idő, Oktató, Csomag)

<footer> Lábléc, jogi információk

# 2. CSS (Megjelenés)

A stíluslap beágyazva található, és biztosítja:

reszponzív elrendezést,

színátmenetes hátteret,

kerekített sarkokat és árnyékot a container körül,

hover effekteket a menükön és gombokon.

# 3. JavaScript (Funkcionalitás)

A JavaScript külső fájlból (js/merged.js) van betöltve, és felelős:

az oktatók és csomagok lekéréséért API-ból,

az űrlapadatok validálásáért,

az adatok beküldéséért a szerver felé (POST).

# Az űrlap mezői

| Mező | Típus | Kötelező? | Megjegyzés |
| --- | --- | --- | --- |
| Név | Szöveg (text) |  | A foglaló neve |
| E-mail | E-mail (email) |  | Kapcsolattartási cím |
| Telefonszám | Telefonszám (tel) |  | Elérhetőség |
| Dátum | Dátum (date) |  | A foglalás napja |
| Időpont | Legördülő lista (select) |  | Előre megadott intervallumok |
| Oktató | Legördülő lista (select) |  | API-ból töltődik be |
| Élménycsomag | Legördülő lista (select) |  | API-ból töltődik be |

# Interaktív elemek

**Dinamikus dropdown-ok:** Az instructorDropdown és packageDropdown mezők automatikusan töltődnek be, mikor az oldal betöltődik.

**Mentés gomb:** A submit-btn gomb küldi el az adatokat az API-n keresztül.

**Validálás:** A JavaScript ellenőrzi, hogy minden mező ki van-e töltve, mielőtt adatot küldene.

# Design jellemzők

**Színvilág:** narancs-barna árnyalatok

**Betűtípus:** Arial, sans-serif – letisztult és jól olvasható.

**Reszponzív felépítés:** A layout alkalmazkodik a képernyőmérethez, így mobilbarát.

**Animációk:** Hover effektek a menüpontokon és a gombokon.

# 

# Biztonság és validáció

HTML szinten a mezők required attribútumot kaptak.

A JavaScript (a merged.js fájlban) ellenőrzi az adatokat, mielőtt azokat beküldené.

A kommunikáció JSON formátumban történik a szerver felé.

# Oldal célja

Ez az oldal a lőtér időpontfoglalási rendszerének webes felülete, amely felhasználóbarát módon teszi lehetővé az érdeklődők számára, hogy **egyszerűen foglaljanak időpontot** kiválasztott oktatóhoz és élménycsomaghoz. A foglalási folyamat során a felhasználónak mindössze néhány adatot kell megadnia (név, e-mail, telefonszám, dátum és idő), majd egyetlen gombnyomással elmentheti a jelentkezését.

A weboldal modern, reszponzív kialakítású, vizuálisan letisztult stílusával pedig bármilyen eszközön (mobil, tablet, asztali gép) könnyen kezelhető.

# Technikai dokumentáció jellegű leírás:

# Technológiák

**JavaScript (ES6+)**

**Fetch API**

**DOM manipuláció**

**REST API hívások**

Backend kapcsolódási pont:https://loter-production.up.railway.app/swagger

# Fő funkciók áttekintése

| Funkció | Leírás |
| --- | --- |
| Dinamikus dropdown betöltés | Oktatók és csomagok lekérdezése az API-ból |
| Időpont-ellenőrzés | Elérhető helyek lekérdezése adott oktató, dátum és idő szerint |
| Form validálás | Kötelező mezők ellenőrzése |
| Adatküldés | Foglalási adatok POST kéréssel való beküldése |
| Felhasználói visszajelzések | Siker / hibaüzenetek alert formájában |

# Strukturális Felépítés

A program egyetlen DOMContentLoaded eseménykezelőben működik, így biztosítva, hogy minden DOM elem elérhető legyen a JavaScript betöltése után.

# Főbb komponensek

API\_BASE – API elérési út alapja.

MAX\_PARTICIPANTS – Maximális létszám egy időpontra.

elements – Az összes szükséges DOM elem struktúrában elmentve.

init() – Inicializáló függvény, elindítja az alkalmazást.

loadDropdowns() – Oktatók és csomagok betöltése.

checkAvailability() – Elérhetőség ellenőrzése valós időben.

handleFormSubmit() – Form elküldése (POST).

getFormData() – Mezők összegyűjtése és validálása.

submitAppointment() – POST kérés küldése az API-ra.

showSuccess() / showError() – Visszajelzések megjelenítése.

# API Kapcsolatok

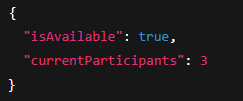
| Cél | URL | HTTP Módszer | Megjegyzés |
| --- | --- | --- | --- |
| Oktatók lekérdezése | /api/Instructors | GET | Dropdown feltöltése |
| Csomagok lekérdezése | /api/Packages | GET | Dropdown feltöltése |
| Elérhetőség ellenőrzése | /api/Appointments/availability | GET + param | Valós idejű lekérdezés |
| Foglalás mentése | /api/Appointments | POST | JSON formátumú foglalási adat |

# Formában gyűjtött adatok

| Mező | Kódnév | Kötelező? |
| --- | --- | --- |
| Oktató ID | instructorId |  |
| Csomag ID | packageId |  |
| Dátum | date |  |
| Időintervallum | timeRange |  |
| Név | clientName |  |
| E-mail | email |  |
| Telefonszám | phone |  |

# Elérhetőség ellenőrzés logikája

Az Appointments/availability végpont visszaadja:



A program ez alapján jeleníti meg:

** X hely maradt**

** Foglalt (7/7)**

# Beküldési folyamat

A felhasználó kitölti az űrlapot.

A submit-btn gomb aktiválja a handleFormSubmit() függvényt.

Az adatok validálásra kerülnek (getFormData()).

A submitAppointment() függvény POST-ot küld az API-nak.

Sikeres mentés esetén alert: **„Sikeres időpontfoglalás!”**

Hiba esetén alert: **„Hiba történt a foglalás során”**

# Hibakezelés

Nem töltött mezők → hibaüzenet

API válaszhiba → throw new Error() üzenettel

Váratlan hibák → console + alert

# Interfész viselkedés

A foglalás gomb szövege „Mentés” → „Küldés…” válik mentés közben.

Spinner ikon jelenik meg a gombban.

Gomb letiltás történik az elküldés során (disabled = true).

A sikeres elküldés után az űrlap újra kiürül.

# Oldal célja

Ez a JavaScript program egy modern, dinamikus **időpontfoglalási rendszer vezérlője**, amely egy webes felületen keresztül biztosítja a látogatók számára a foglalás lehetőségét. A program automatikusan betölti az oktatók és élménycsomagok adatait egy API-ból, majd a felhasználó által kiválasztott időpontra valós időben ellenőrzi a helyek elérhetőségét.

A sikeres kitöltés után az űrlap adatai JSON formátumban továbbításra kerülnek egy backend API-hoz. A rendszer célja egy egyszerű, gyors, és hibatűrő foglalási élmény nyújtása a felhasználók számára.

Backend program bemutatása:

# AppointmentController.cs – Részletes Dokumentáció

Az ASP.NET Core Web API egyik REST-alapú kontrollere, és a **VizsgaremekApp** nevű alkalmazásban a **lőtéri időpontfoglalásokat** kezeli.

# Áttekintés

A AppointmentController egy REST API vezérlő, amely a VizsgaremekContext entitáskontextust használja, hogy CRUD (Create, Read, Update, Delete) műveleteket hajtson végre az Appointments entitáson.

# Namespace és Osztály



A vezérlő a ShootingRangeAPI.Controllers névtér alatt helyezkedik el.

Az [ApiController] attribútum biztosítja a modellezés és automatikus érvényesítés lehetőségeit.

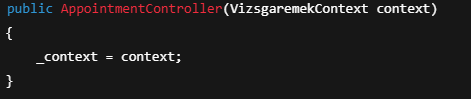
[Route("api/[controller]")] -> az útvonal: api/Appointment

# Függőségek



**VizsgaremekContext**: Az Entity Framework Core DbContext osztályából származik. Ezen keresztül érhetjük el az Appointments táblát.

# Konstruktor



Konstruktor-injektálással kapja meg a kontroller a DbContext példányát, így minden metódus ezen keresztül kommunikálhat az adatbázissal.

# Metódusok részletesen

# 1. GET: api/Appointment



**Cél**: Az összes időpont lekérése.

**Visszatérés**: List<Appointment>

**HTTP válasz**: 200 OK + a foglalások listája

# 2. GET: api/Appointment/{id}



**Cél**: Egy adott időpont lekérése az azonosító alapján.

**Paraméter**: id – int típusú, URL-ből jön.

**Visszatérés**:

200 OK + az időpont objektum

404 Not Found, ha nem létezik

# 3. POST: api/Appointment



**Cél**: Új időpont létrehozása.

**Bemenet**: Appointment típusú JSON a kérés törzsében (request body).

**Folyamat**:

Hozzáadás a kontextushoz

Mentés az adatbázisba

**Visszatérés**: 201 Created + az új időpont, valamint Location header a lekérdezési URL-lel.

# 4. PUT: api/Appointment/{id}



**Cél**: Egy létező időpont frissítése.

**Validáció**: Az URL-ben megadott id és a body Id mezőjének egyeznie kell.

**Folyamat**:

Módosított állapot beállítása

Mentés

Kivételkezelés ütközés esetén (DbUpdateConcurrencyException)

**Visszatérés**:

204 No Content – sikeres frissítés

400 Bad Request – id ütközés

404 Not Found – nem létező időpont

# 5. DELETE: api/Appointment/{id}



**Cél**: Egy időpont törlése azonosító alapján.

**Folyamat**:

Megkeresi az időpontot

Ha megtalálja, eltávolítja az adatbázisból

**Visszatérés**:

204 No Content – sikeres törlés

404 Not Found – ha nem található

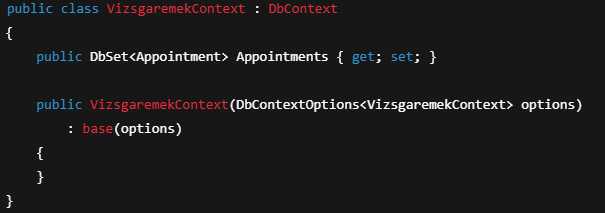
# Segédfüggvény



Ellenőrzi, hogy létezik-e időpont az adott id-val.

Belső használatra készült (pl. PUT esetén, amikor ellenőrizni kell létezik-e az erőforrás).

# Adatbázis kontextus (VizsgaremekContext)



# Összefoglalás

| Művelet | Metódus | URL | Leírás |
| --- | --- | --- | --- |
| Listázás | GET | /api/appointment | Összes időpont lekérése |
| Lekérdezés | GET | /api/appointment/{id} | Időpont lekérése ID alapján |
| Létrehozás | POST | /api/appointment | Új időpont létrehozása |
| Módosítás | PUT | /api/appointment/{id} | Létező időpont frissítése |
| Törlés | DELETE | /api/appointment/{id} | Időpont törlése |

# InstructorsController.cs – Részletes dokumentáció

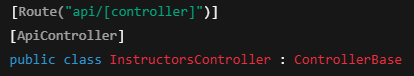
Az ASP.NET Core Web API alkalmazás része, és az oktatók (instructorok) kezelését végzi a rendszerben.

***Áttekintés***

Az InstructorsController az ASP.NET Core Web API egyik REST vezérlője, amely a VizsgaremekContext adatbáziskontextust használja az **oktatók (instructors)** kezelésére.

Ez a kontroller támogatja a **CRUD (Create, Read, Update, Delete)** műveleteket.

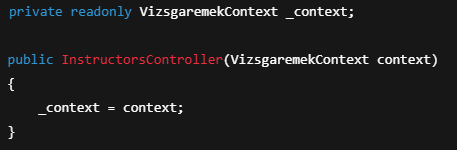
# Alapbeállítások



[ApiController] attribútum segíti az automatikus modell-ellenőrzést és hibakezelést.

[Route("api/[controller]")]: A kontroller URL-je: api/instructors lesz.

# Konstruktor és kontextus

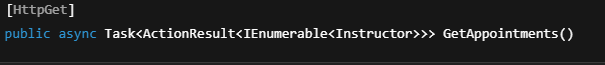


**\_context**: Az adatbázis-kapcsolatot biztosítja (Entity Framework Core DbContext).

Konstruktoron keresztül kapja meg az InstructorsController a kontextust.

# Metódusok – CRUD részletesen

# 1. GET: api/instructors



**Cél**: Az összes oktató lekérése az adatbázisból.

**Visszatérési érték**: List<Instructor>

**HTTP válasz**: 200 OK + oktatók listája

***2. GET: api/instructors/{id}***



**Cél**: Egy adott oktató lekérdezése ID alapján.

**Bemeneti paraméter**: id – az oktató azonosítója

**Kimenet**:

200 OK + Instructor objektum

404 Not Found, ha az oktató nem létezik

***3. POST: api/instructors***



**Cél**: Új oktató rögzítése.

**Bemenet**: Instructor objektum (JSON formátumban)

**Művelet**:

Hozzáadja az új oktatót az adatbázishoz.

Mentés az adatbázisba (\_context.SaveChangesAsync)

**Visszatérés**:

201 Created + az új oktató

Location header: URL az új rekordhoz

# 4. PUT: api/instructors/{id}



**Cél**: Létező oktató frissítése.

**Bemenet**:

id: URL-ből

Instructor objektum: JSON-ben

**Ellenőrzés**:

Ha az id nem egyezik az objektum Id mezőjével → 400 Bad Request

**Mentés**:

Állapot módosítása (EntityState.Modified)

Mentés

**Kivételek**:

DbUpdateConcurrencyException: ütközés esetén

**Visszatérés**:

204 No Content: sikeres

404 Not Found: ha nem létezik a rekord

# 5. DELETE: api/instructors/{id}



**Cél**: Oktató törlése ID alapján.

**Művelet**:

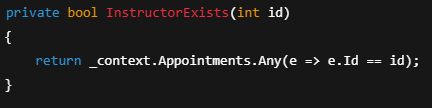
Lekérdezi az oktatót

Ha létezik, törli és elmenti

**Kimenet**:

204 No Content: sikeres törlés -404 Not Found: nem található

# Segédfüggvény



**Cél**: Ellenőrzi, hogy létezik-e az oktató az adott azonosítóval.

# Összegzés

| Művelet | Metódus | URL | Leírás |
| --- | --- | --- | --- |
| Listázás | GET | /api/instructors | Összes oktató lekérése |
| Lekérdezés | GET | /api/instructors/{id} | Egy oktató ID alapján |
| Létrehozás | POST | /api/instructors | Új oktató rögzítése |
| Módosítás | PUT | /api/instructors/{id} | Oktató frissítése |
| Törlés | DELETE | /api/instructors/{id} | Oktató törlése |

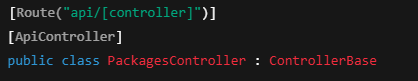
# PackagesController.cs – Részletes dokumentáció

# Áttekintés

A PackagesController célja az **élménycsomagok (packages)** elérhetővé tétele, kezelése és adminisztrációja egy API-n keresztül.

Ez a vezérlő a **CRUD (Create, Read, Update, Delete)** műveleteket támogatja a Package entitáson keresztül, az Entity Framework Core adatbázis-kontextust (VizsgaremekContext) használva.

***Alapbeállítások***

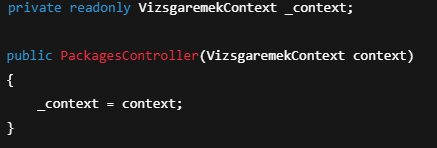


[ApiController]: Automatikus modell érvényesítés, hibakezelés.

[Route("api/[controller]")]: A kontroller URL-je: api/packages lesz.

ControllerBase: Alaposztály, amely nem tartalmaz nézetkezelést (csak API-khoz ideális).

***Konstruktor***



A VizsgaremekContext az EF Core adatbáziskapcsolatot biztosítja.

Dependency Injection segítségével kapja meg a vezérlő.

# CRUD Műveletek

# GET: api/packages – Összes csomag lekérése



Lekéri az összes Package rekordot az adatbázisból.

**Visszatérési érték**: 200 OK + élménycsomagok listája (List<Package>)

# GET: api/packages/{id} – Egy csomag lekérése ID alapján



Megkeresi a megadott ID-jú élménycsomagot.

**Visszatérés**:

200 OK + Package objektum

404 Not Found, ha nincs ilyen csomag

# POST: api/packages – Új csomag létrehozása



Új élménycsomagot ad az adatbázishoz.

**Mentés**: EF Core SaveChangesAsync()

**Visszatérés**:

201 Created + új Package

Location header-ben az új erőforrás elérhetősége

# PUT: api/packages/{id} – Csomag módosítása



Ellenőrzi, hogy az ID egyezik-e az objektum ID-jával.

**Mentés**: módosított állapot beállítása és mentése

**Visszatérés**:

204 No Content – sikeres frissítés

400 Bad Request – ID nem egyezik

404 Not Found – ha nincs ilyen csomag

# DELETE: api/packages/{id} – Csomag törlése



ID alapján megkeresi, majd eltávolítja a csomagot.

**Visszatérés**:

204 No Content – sikeres törlés

404 Not Found – ha nincs ilyen ID-jú csomag

# Összefoglaló táblázat

| Művelet | Metódus | URL | Leírás |
| --- | --- | --- | --- |
| Lekérdezés | GET | /api/packages | Összes csomag lekérése |
| Lekérdezés | GET | /api/packages/{id} | Egy csomag lekérése ID szerint |
| Létrehozás | POST | /api/packages | Új csomag létrehozása |
| Módosítás | PUT | /api/packages/{id} | Csomag frissítése |
| Törlés | DELETE | /api/packages/{id} | Csomag törlése |

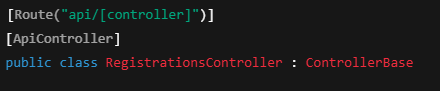
# RegistrationsController.cs – Részletes dokumentáció

# Áttekintés

A RegistrationsController osztály az **API végpontjait** kezeli a felhasználói **regisztrációk** (Registration entitás) CRUD (Create, Read, Update, Delete) műveleteihez egy lőtér időpontfoglaló rendszerben.

A vezérlő a VizsgaremekContext EF Core adatbáziskontextust használja a Registrations táblával való kommunikációhoz.

# Konstrukció

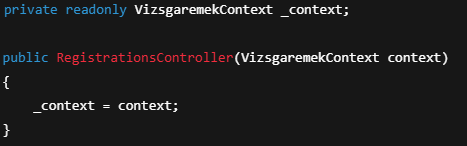


[Route("api/[controller]")]: A vezérlő URL-je api/registrations lesz.

[ApiController]: Bekapcsolja az automatikus modell validációt és hibakezelést.

ControllerBase: Alaposztály Web API-k számára (nem tartalmaz nézetkezelést).

# Konstruktor



A VizsgaremekContext példányt dependency injection használatával kapja meg.

Ez biztosítja az adatbázis-hozzáférést a vezérlő metódusaihoz.

# CRUD műveletek

# GET: api/registrations – Összes regisztráció lekérdezése



Lekéri az összes regisztrációt az adatbázisból.

**Visszatérés**: 200 OK + List<Registration>

# GET: api/registrations/{id} – Egy regisztráció lekérdezése



Megkeresi a megadott ID-jú regisztrációt.

**Visszatérés**:

200 OK – sikeres lekérdezés

404 Not Found – ha nincs ilyen ID

# POST: api/registrations – Új regisztráció létrehozása



Új Registration objektum mentése.

EF Core segítségével adatbázisba kerül.

**Visszatérés**:

201 Created

A Location fejléc tartalmazza az új regisztráció elérési útját.

# 

# PUT:Regisztráció frissítése



Frissíti a meglévő regisztrációt az adott ID alapján.

Validálja az ID egyezését, majd frissíti a rekordot.

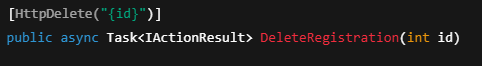
**Visszatérés**:

204 No Content – sikeres mentés

400 Bad Request – ha az ID nem egyezik

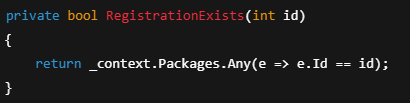
404 Not Found – ha nem létezik a regisztráció

# DELETE:Regisztráció törlése



Törli az adott ID-jú regisztrációt.

# Segédfüggvény: Regisztráció létezik-e



# Submint.cs – Részletes dokumentáció

# Fő funkciója:

Ez az objektum egy felhasználói időpontfoglalás adatait tartalmazza. A foglalást elmenti az adatbázisba, ha minden rendben van.

# Főbb részei:

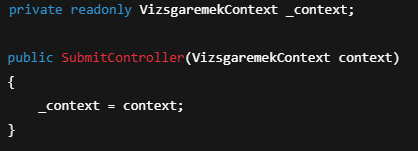
# 1. Attribútumok és útvonal



[ApiController]: Ez egy beépített attribútum, amely automatikusan kezeli a modellérvényesítési hibákat (pl. ha kötelező mező hiányzik).

[Route("api/[controller]")]: A vezérlő elérhetőségi útvonala api/submit lesz (a vezérlő neve SubmitController, a [controller] értéke tehát submit).

# 2. Konstruktor – Adatbázis kontextus injektálása



A konstruktor paraméterként megkapja az Entity Framework Core által használt VizsgaremekContext példányát, így lehetőség nyílik az adatbázis elérésére.

# 3. POST metódus – Foglalás mentése

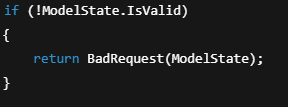


Ez a metódus a **POST** kérésekre reagál.

A kérés törzsében (body) egy Appointment típusú JSON objektumot vár.

A data objektum a következő mezőket tartalmazhatja: Nev, Email, DatumIdo, Tel, ValasztottOktatoId, ValasztottCsomagId.

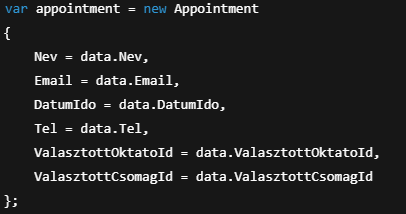
# 4. Érvényesítés és adatmentés



Ha a kapott adatok nem felelnek meg a modell érvényességi szabályainak, 400 Bad Request választ küld vissza.



Hibakeresés céljából kiírja a fogadott adatokat a konzolra.

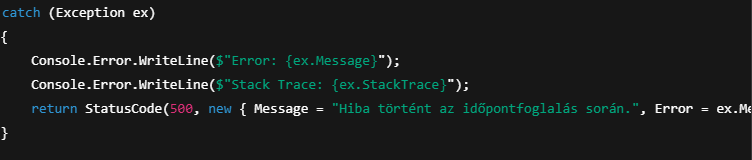


A data objektum alapján létrehoz egy új Appointment példányt.

Hozzáadja az adatbázishoz, majd menti az adatokat.

Siker esetén 200 OK választ küld egy pozitív visszajelzéssel.

# 5. Hibakezelés



Ha bármilyen kivétel történik (pl. adatbázis hiba), azt naplózza, és 500 Internal Server Error választ küld vissza.

# Összegezve

## Ez az API-végpont:

Lehetővé teszi, hogy egy külső kliens (pl. frontend) időpontot foglaljon.

Érvényesíti a foglalási adatokat.

Elmenti azokat egy adatbázisba (EF Core segítségével).

Visszajelzést küld a sikeres vagy sikertelen foglalásról.

# Program.cs – Részletes dokumentáció

# A program célja:

Ez a fájl egy **lövészethez kapcsolódó időpontfoglalási rendszer** backendjének belépési pontja.

Feladata:

A webalkalmazás konfigurálása

Adatbáziskapcsolat beállítása

API végpontok elérhetővé tétele

Swagger dokumentáció engedélyezése

CORS szabályok definiálása

Kezdeti adatok feltöltése az adatbázisba (instructors, packages)

# API végpontok:

POST /api/submit – Új időpontfoglalás létrehozása

GET /api/instructors – Oktatók lekérése (ha van implementálva)

GET /api/packages – Csomagok lekérése (ha van implementálva)

# Alkalmazás indítása



**Swagger csak fejlesztői környezetben** fut.

**MapControllers()** engedélyezi a kontroller alapú útvonalakat.

**UseAuthorization()** előkészíti az alkalmazást jogosultságkezelésre,

# Az adatbázis adatbázismodell-diagramja

# Adatbázismodell összefoglaló

## Instructor

Id (PK)

Name

## Package

Id (PK)

Name

Price

## Appointment (Foglalás)

Id (PK)

Nev, Email, Tel, DatumIdo

ValasztottOktatoId (FK → Instructor.Id)

ValasztottCsomagId (FK → Package.Id)

## Registration (Regisztráció)

Id (PK)

Nev, Email, Tel, Lakcim, Minosites, MinositesIdeje

# 

# Webes felület

https://lotervizsgaremek.com

# Fejlesztési lehetőségek

asztali program és mobil alkalmazás

# A program használatának rövid bemutatása

Időpontfoglalás:

A vendégek kiválasztják az időpontot, az oktatót és az élménycsomagot, majd megadják adataikat.

Oktatói jelentkezés:

Az oktatók online űrlapon keresztül adhatják be jelentkezésüket, mellékelve önéletrajzukat, igazolványukat és minősítéseiket.

Admin bejelentkezés:

Az adminisztrátorok beléphetnek az admin felületre, ahol kezelhetik az időpontokat, jelentkezőket, oktatókat és csomagokat.

# A program melyik részével találkozik a tulajdonos, és melyik részével a felhasználó.

Tulaj (Admin) részei:

**Admin bejelentkezés** → Foglalások, oktatók, csomagok kezelése.

**Admin felület** → Minden háttérmunka innen megy.

Felhasználó részei:

**Főoldal** → Üdvözlés, lőtér bemutatása.

**Élménycsomagok** → Milyen csomagok közül lehet választani.

**GYIK** → Kérdésekre válaszok.

**Kapcsolat** → Hogyan tud velünk kapcsolatba lépni.

**Nyitvatartás** → Mikor vagyunk nyitva.

**Foglalás** → Időpont foglalása lövészetre.

**Jelentkezés oktatónak** → Oktatóknak szóló jelentkezési űrlap.

**Nyelvváltás** → Magyar ↔ Angol átváltás.

# Használati útmutató mindkettőnek

Felhasználók számára:

1. Nyisd meg a főoldalt.
2. A menüből válaszd ki a kívánt részt (pl. Foglalás).
3. A foglalás során:  
   1. Add meg neved, e-mail címed, telefonszámod.
   2. Válassz időpontot, oktatót és csomagot.
   3. (Megjegyzés opcionális.)
   4. Küldd el a foglalást a "Mentés" gombbal.
4. A rendszer értesít, ha a foglalás sikeres.

Leendő oktatók számára:

1. Navigálj a "Jelentkezz oktatónak!" menüpontra.
2. Töltsd ki a jelentkezési űrlapot:  
   1. Név, e-mail, telefonszám, lakcím, minősítés.
   2. Igazolvány, önéletrajz és minősítések feltöltése kötelező.
3. Küldd el a jelentkezést. Az admin jóváhagyása után kapsz visszajelzést.

Adminisztrátorok számára:

1. Lépj be az "Admin bejelentkezés" menüpontban.
2. A belépés után:  
   1. Ellenőrizheted a foglalásokat, jelentkezéseket.
   2. Elfogadhatod vagy elutasíthatod az oktatói jelentkezéseket.
   3. Új csomagokat és oktatókat adhatsz hozzá.

# Tesztelés

# Tesztelési szempontok

# 1. Használhatóság és navigáció

**Menüstruktúra**: Ellenőriztem, hogy a menüpontok logikusan és könnyen navigálhatóak-e.

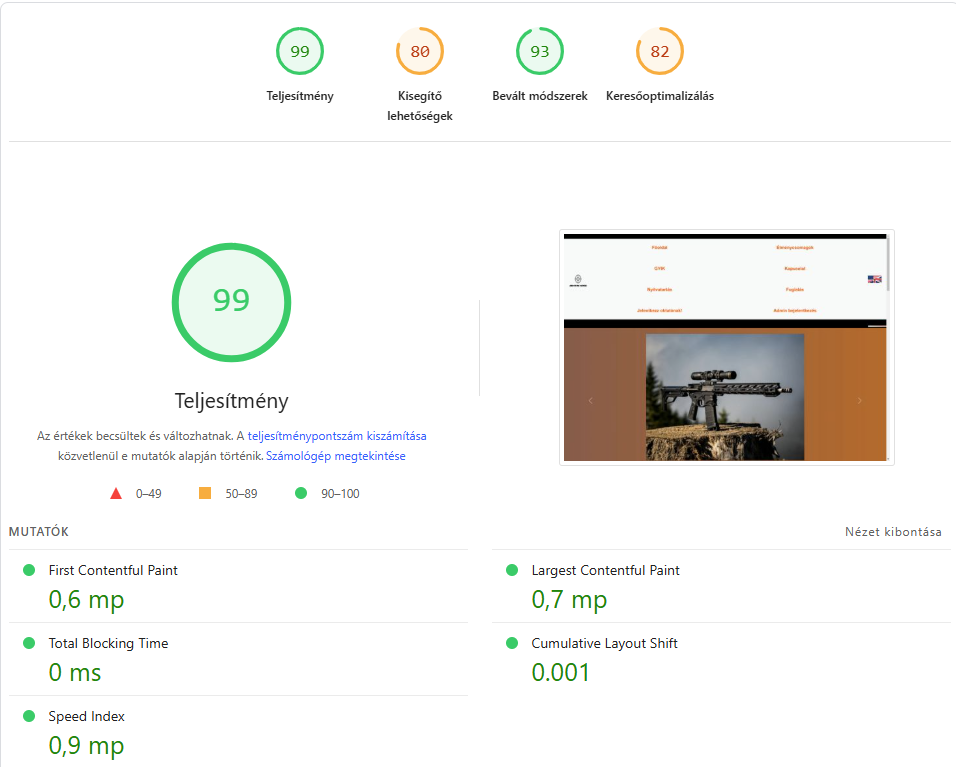
**Szövegek olvashatósága**: Megyőződtem róla, hogy a betűtípusok és színek megfelelő kontrasztot biztosítanak.

**Űrlapok működése**: Teszteltem a kapcsolatfelvételi és foglalási űrlapok működését, beleértve a hibakezelést és visszaigazolásokat.

# 2. Teljesítmény és sebesség

**Oldalbeöltési idő**: [GTmetrix](https://gtmetrix.com/) oldalán mértem meg a sebességét.



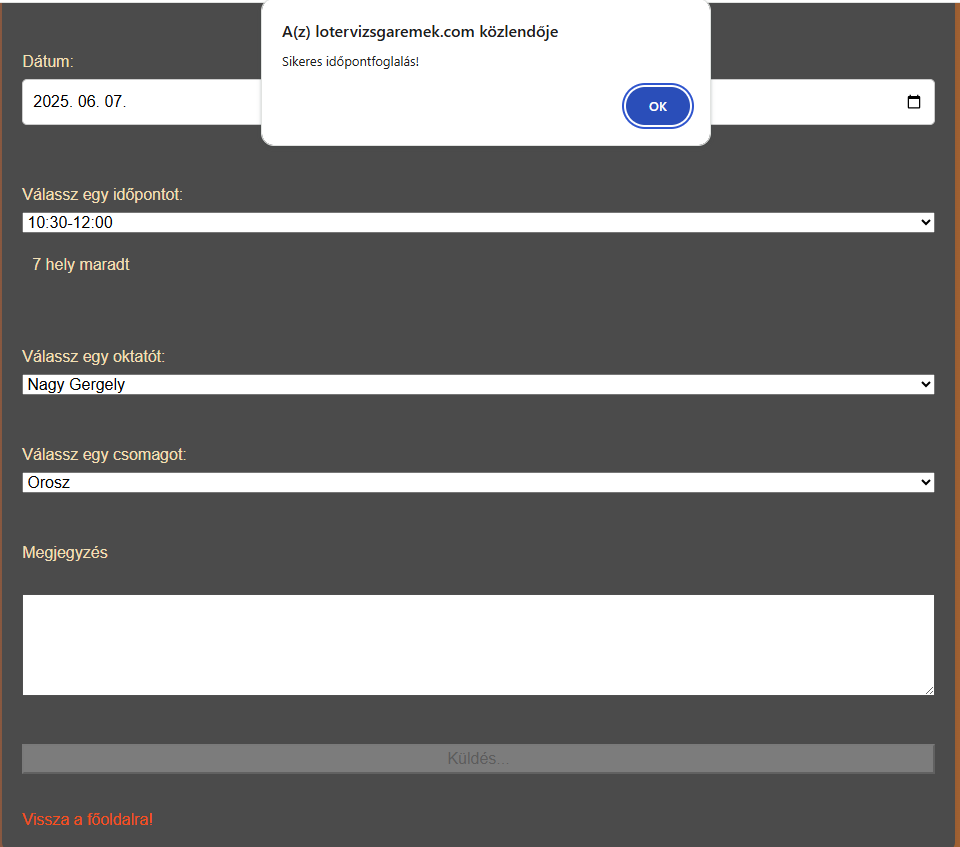


Az oldal teljesítményét a <https://pagespeed.web.dev/> oldalán teszteltük.

**Mobilbarát megjelenés**: Ellenőriztem, hogy az oldal reszponzív-e, és megfelelően jelenik meg különböző eszközökön.

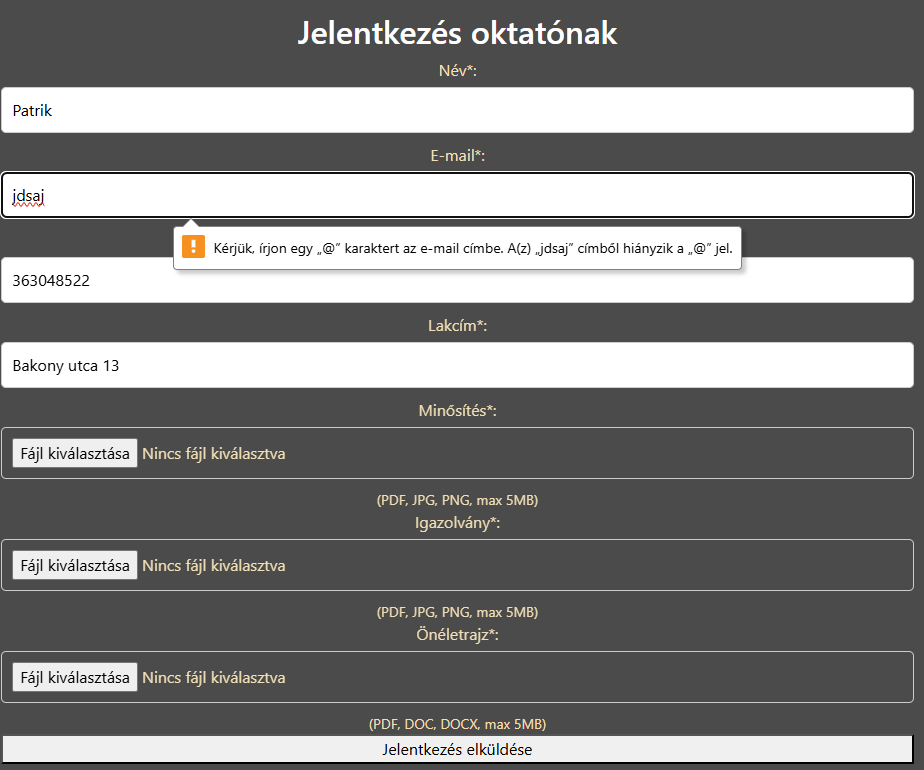
# A tesztekhez végzett kód

# Teszteredmények dokumentációja

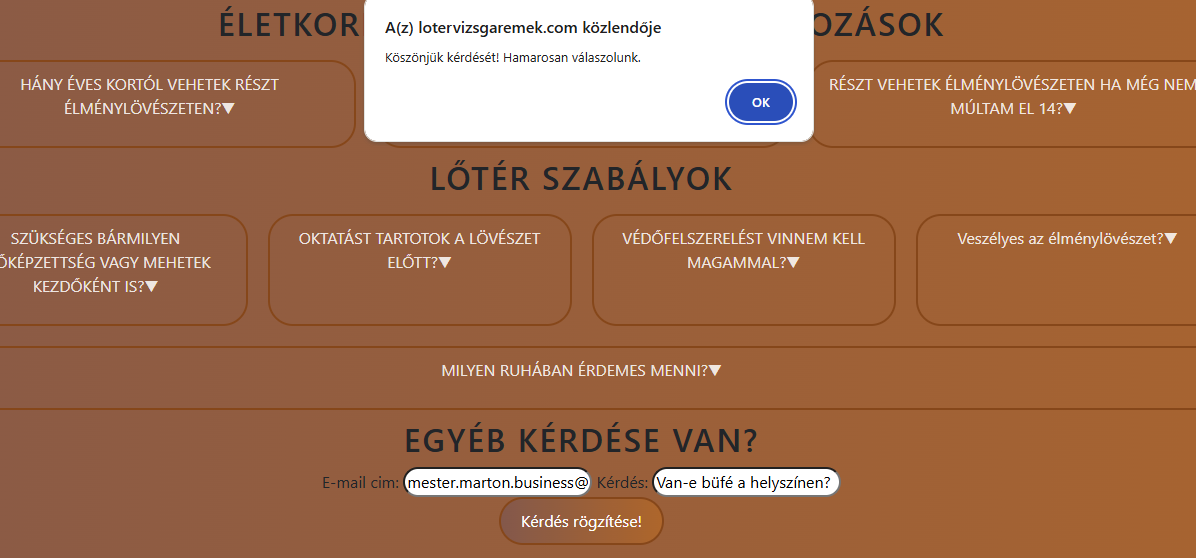


Amennyiben minden mező megfelelően van kitöltve abban az esetben az időpontfoglalás mentésre kerül az admin oldalon illetve az adatbázisba és kapsz egy válasz üzenetet az oldal tetején,hogy sikeres időpontfoglalás történt.



Az E-mail cím mező kéri az érvényes formátumot

Ugyan ez működik a jelentkezés oldalon is

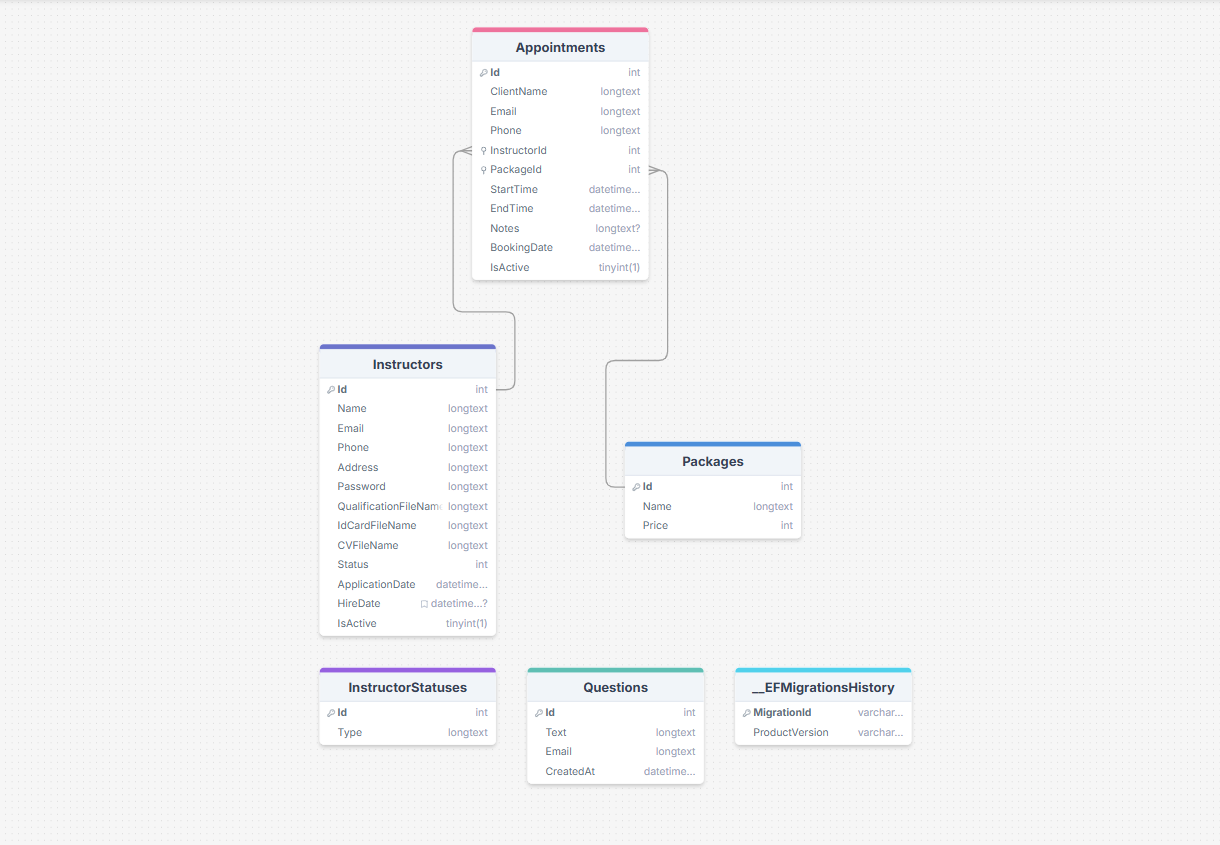


Ha van kérdés akkor az rögzül az admin oldalon illetve rögzül az adatbázisban.



Amennyiben a mezők nincsenek kitöltve úgy küld egy üzenetet

## Adatbázis modell:



# A fejlesztés során használt projektszervezési eszközök

Github

Discord

# Melléklet

<https://github.com/HunFuzeMain/loter>

# A program forráskódja

<https://github.com/HunFuzeMain/loter>

# Források/linkek

* <https://lotervizsgaremek.com/> - weboldal linkje
* <https://www.design.com> – logó tervezés
* <https://unsplash.com> – képek
* <https://pagespeed.web.dev/> - tesztelés
* <https://railway.com/> - backend és adatbázis tárolás
* korábbi iskolai feladatok felhasználása
* https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css - bootstrap
* <https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js> - bootstrap
* <https://drawsql.app/> - adatbázis modell