**DOKUMENTÁCIÓ**

**Téma:** Elektronikus Napló Webalkalmazás (Java Spring Boot) **Tárgy:** Java alkalmazások – Gyakorlat

Github: https://github.com/Turikrisztian/java-gyak-beadando2.git

Link: http://rivendell.nje.hu:9443/NLI24V-gy/

**Készítették:**


- [Türi Krisztián Jenő] – [NLI24V]
- [Takács Milán] – [KWKABN]


Felhasználók

Admin felhasználó:

Felhasználónév: admin2

Jelszó: admin123

User felhasználó:

Felhasználónév: tanar1

jelszó: tanar123

**1. Bevezetés**

Jelen dokumentáció a „Java alkalmazások” tárgy gyakorlati beadandó feladatának, az Elektronikus Napló webalkalmazásnak a részletes leírását tartalmazza. A projekt célja egy modern, szerver oldali Java technológiákra épülő információs rendszer létrehozása, amely alkalmas egy iskola (a feladatban szereplő Városvégi Gimnázium) tanulóinak, tantárgyainak és osztályzatainak kezelésére.

Az alkalmazás fejlesztése során a Spring Boot keretrendszert használtuk, amely az iparágban elterjedt szabvány a nagyvállalati Java alkalmazások készítésére. A választott téma szorosan illeszkedik a megadott adatforrásokhoz (diak.txt, targy.txt, jegy.txt), melyek egy valósághű iskolai adminisztrációs rendszer alapját képezik.

## 2. Feladatkiírás és Célkitűzések

A feladat egy dinamikus weboldal elkészítése volt, amely megfelel a modern webes követelményeknek (reszponzivitás, biztonság, interaktivitás). A kötelezően megvalósítandó funkciók a következők voltak:

- **Reszponzív dizájn:** Mobilbarát megjelenés Bootstrap segítségével.
- **Autentikáció:** Biztonságos bejelentkezés, regisztráció és jogosultságkezelés (Látogató, Regisztrált felhasználó, Admin).
- **Adatbázis integráció:** Külső forrásfájlok beolvasása és tárolása relációs adatbázisban.
- **Interaktív funkciók:** Kapcsolat űrlap szerver oldali validációval, üzenetek listázása.
- **Vizualizáció:** Adatok grafikus megjelenítése (Chart.js).
- **CRUD műveletek:** Diákok adatainak felvitele, módosítása, törlése és listázása.
- **REST API:** Gépi interfész biztosítása az adatok eléréséhez.

## 3. Felhasznált Technológiák

A fejlesztés során az alábbi szoftveres környezetet és könyvtárakat alkalmaztuk:

- **Nyelv:** Java 17
- **Keretrendszer:** Spring Boot 3.2.3
    - *Spring Web:* MVC architektúra megvalósítása.
    - *Spring Data JPA:* Adatbázis műveletek absztrakciója (Hibernate alapokon).
    - *Spring Security:* Autentikáció és autorizáció.
- **Sablonkezelő:** Thymeleaf (szerver oldali HTML renderelés).
- **Adatbázis:** H2 Database (beágyazott, memóriában futó SQL adatbázis fejlesztéshez). Éles környezetben MySQL-re cserélhető.
- **Frontend:**
    - Bootstrap 5.3 (CSS keretrendszer a reszponzivitásért).

    - Chart.js (JavaScript könyvtár diagramokhoz).
- **Build eszköz:** Maven.
- **Fejlesztőkörnyezet:** IntelliJ IDEA.

## 4. Rendszerterv és Adatbázis

Az alkalmazás szíve a relációs adatbázis, amely a megadott szöveges fájlok alapján épül fel. Az adatmodellt entitás osztályok (Entity) segítségével képeztük le Java kódra.

### 4.1. Adatbázis modell (ER Diagram)

A rendszer négy fő táblát és egy felhasználói táblát használ. A kapcsolatok a következők:

- Egy diáknak (**Diak**) több jegye (**Jegy**) lehet (Egy-a-többhöz kapcsolat).
- Egy tárgyból (**Targy**) több jegy születhet (Egy-a-többhöz kapcsolat).
- A jegy tábla kapcsolótáblaként is funkcionál a diák és a tárgy között, kiegészítve a dátummal, érdemjeggyel és típussal.

### 4.2. Táblák szerkezete

1. **DIÁK (Student):**
    - id: Egyedi azonosító (a forrásfájlból).
    - nev: A tanuló neve.
    - osztaly: Osztályjelzés (pl. 12/A).
    - fiu: Logikai érték (Igaz: fiú, Hamis: lány).
2. **TÁRGY (Subject):**
    - id: Egyedi azonosító.
    - nev: Tantárgy neve (pl. Matematika).
    - kategoria: Tantárgy típusa (pl. Reál, Humán).
3. **JEGY (Grade):**
    - id: Generált azonosító.
    - diakid: Külső kulcs a Diák táblára.
    - targyid: Külső kulcs a Tárgy táblára.
    - ertek: Az osztályzat (1-5).
    - datum: A szerzés ideje.
    - tipus: A jegy típusa (pl. témazáró, felelet).

4. **USERS (Felhasználók):**
    - id, username, password (BCrypt titkosítva), role (jogosultság).
5. **UZENET (Messages):**
    - id, feladoNev, email, szoveg, kuldesIdeje.

## 5. Implementáció Részletei

### 5.1. Projekt szerkezet

A projektet a szabványos Maven struktúra szerint építettük fel, a com.neptun.naplo csomag alatt rétegekre bontva:

- **model:** Az adatbázis entitások (POJO osztályok JPA annotációkkal).
- **repo:** A Spring Data JPA interfészek, amelyek az SQL lekérdezéseket végzik.
- **controller:** A HTTP kéréseket fogadó és a válaszokat összeállító osztályok.
- **config:** A konfigurációs osztályok (Security, DataLoader).

### 5.2. Adatbetöltés (Data Loading)

Az alkalmazás indulásakor a DataLoader osztály automatikusan lefut. Ez a komponens felelős a src/main/resources mappában található diak.txt, targy.txt és jegy.txt fájlok beolvasásáért.

**A beolvasás logikája:**

1. Ellenőrzi, hogy az adatbázis üres-e.
2. Létrehozza az alapértelmezett felhasználókat (Admin, User).
3. Soronként beolvassa a szöveges fájlokat.
4. A tabulátorral (\t) elválasztott adatokat feldolgozza és objektumokká alakítja.
5. A Jegyek beolvasásánál a dátumot a megfelelő formátumra (yyyy.MM.dd) konvertálja.
6. Az objektumokat a Repository-k segítségével perzisztálja az adatbázisba.

### 5.3. Biztonság (Spring Security)

A rendszer védelmét a Spring Security biztosítja. A SecurityConfig osztályban definiáltuk a hozzáférési szabályokat:

.requestMatchers("/messages").hasAnyRole("USER", "ADMIN", "VISITOR")

.requestMatchers("/crud/**", "/admin/**").hasRole("ADMIN")

A jelszavakat nem sima szövegként, hanem **BCrypt** hash formájában tároljuk az adatbázisban, ami megfelel a modern biztonsági előírásoknak. A bejelentkezés form alapú (Form Login).

### 5.4. Üzleti logika és Vezérlők

A WebController osztály kezeli a felhasználói interakciókat. A Thymeleaf sablonmotort használjuk az adatok HTML-be ágyazására. Példa a diagram adatainak előkészítésére:

```
@GetMapping("/chart")
public String chart(Model model) {
   List<Object[]> stats = jegyRepo.findSubjectAverages();
   // ... adatok átalakítása JSON szerű formátumba a Chart.js számára ...
   model.addAttribute("chartLabels", labels.toString());
   model.addAttribute("chartData", data.toString());
   return "chart";
}
```

## 6. Felhasználói Kézikönyv és Funkciók Bemutatása

Ebben a fejezetben képernyőképekkel illusztrálva mutatjuk be az elkészült alkalmazás működését.

### 6.1. Főoldal és Navigáció

Az alkalmazás megnyitásakor (http://localhost:9443) a főoldal fogadja a látogatót. A felső navigációs sávban (Navbar) érhetők el a menüpontok. A dizájn letisztult, a Bootstrap "Jumbotron" stílusát használja.

### 6.2. Regisztráció és Bejelentkezés

A védett tartalmak eléréséhez be kell jelentkezni. A "Regisztráció" menüpont alatt új felhasználót hozhatunk létre, aki alapértelmezetten USER jogot kap.

A bejelentkezés után a rendszer üdvözli a felhasználót a jobb felső sarokban.

### 6.3. Adatbázis megtekintése

Az "Adatbázis" menüpont alatt tekinthetők meg a rendszerbe betöltött adatok. Három fülön (Tab) különítettük el a Diákokat, a Tárgyakat és a Jegyeket. A nagy adatmennyiség miatt a jegyeknél limitáltuk a megjelenített elemek számát.

### 6.4. Kapcsolat és Üzenetek

A "Kapcsolat" menüpont alatt bárki üzenetet küldhet az adminisztrátornak. A rendszer menti a küldő nevét, e-mail címét és az üzenet szövegét, valamint automatikusan hozzárendeli a küldés idejét.

A sikeres küldésről visszajelzést kapunk. A beérkezett üzeneteket az "Üzenetek" menüpont alatt lehet megtekinteni (csak bejelentkezett felhasználóknak). Az üzenetek időrendben fordított sorrendben jelennek meg.

### 6.5. Diagramok (Statisztika)

A "Diagram" menüpont alatt a Chart.js segítségével vizualizáljuk az iskola tanulmányi eredményeit. A diagram a tantárgyak szerinti átlagosztályzatokat mutatja oszlopdiagram formájában.

### 6.6. CRUD Műveletek (Admin felület)

Ez a funkció csak az ADMIN jogosultságú felhasználóknak érhető el (pl. admin/admin belépés után). Itt van lehetőség az adatbázis módosítására:

1. **Create:** Új diák felvétele.
2. **Read:** Diákok listázása szerkesztési lehetőséggel.
3. **Update:** Meglévő diák adatainak szerkesztése.
4. **Delete:** Diák törlése a rendszerből.

## 7. REST API Dokumentáció

A rendszer rendelkezik egy RESTful API végponttal is, amely lehetővé teszi, hogy külső programok JSON formátumban kérjék le az adatokat. Ez megfelel a modern mikroservice architektúrák követelményeinek.

**Végpontok:**

- GET /api/diakok: Az összes diák listája JSON formátumban.
- GET /api/diakok/{id}: Egy konkrét diák adatai ID alapján.

**Tesztelés Postman/cURL segítségével:** Az API működését cURL paranccsal teszteltük: curl http://localhost:9443/api/diakok

## 8. Telepítési és Futtatási Útmutató

Az alkalmazás Linux környezetben történő futtatásához az alábbi lépések szükségesek:

1. **Build:** A forráskódból futtatható JAR fájl készítése a mvn clean package paranccsal.
2. **Feltöltés:** A keletkezett naplo-0.0.1-SNAPSHOT.jar fájl feltöltése a Linux szerverre (pl. a /var/www/naplo mappába).
3. **Futtatás:** Az alkalmazás indítása a háttérben: nohup java -jar naplo-0.0.1-SNAPSHOT.jar &
4. **Elérés:** A szerver 9443-as portján keresztül (http://szerver_ip:9443).

A fejlesztés során a application.properties fájlban a H2 adatbázist használtuk, de éles környezetben a konfiguráció átírható MySQL vagy PostgreSQL elérésre.

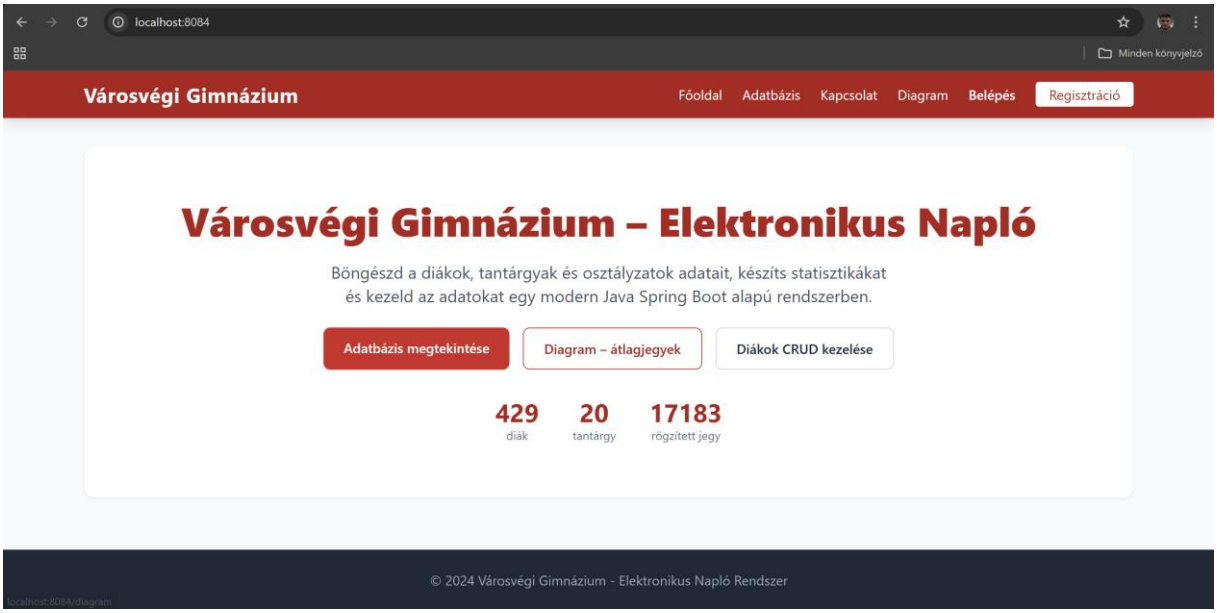
## 9. Projektmunka és Verziókezelés

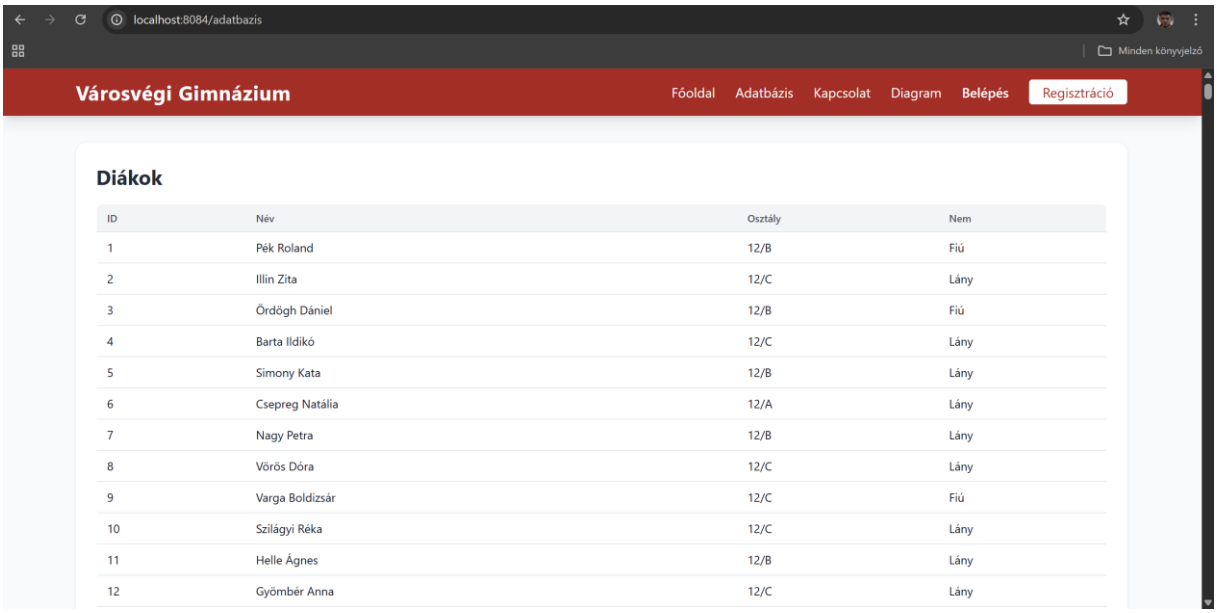
A fejlesztés során a GitHub verziókövető rendszert használtuk a csoportmunka koordinálására.

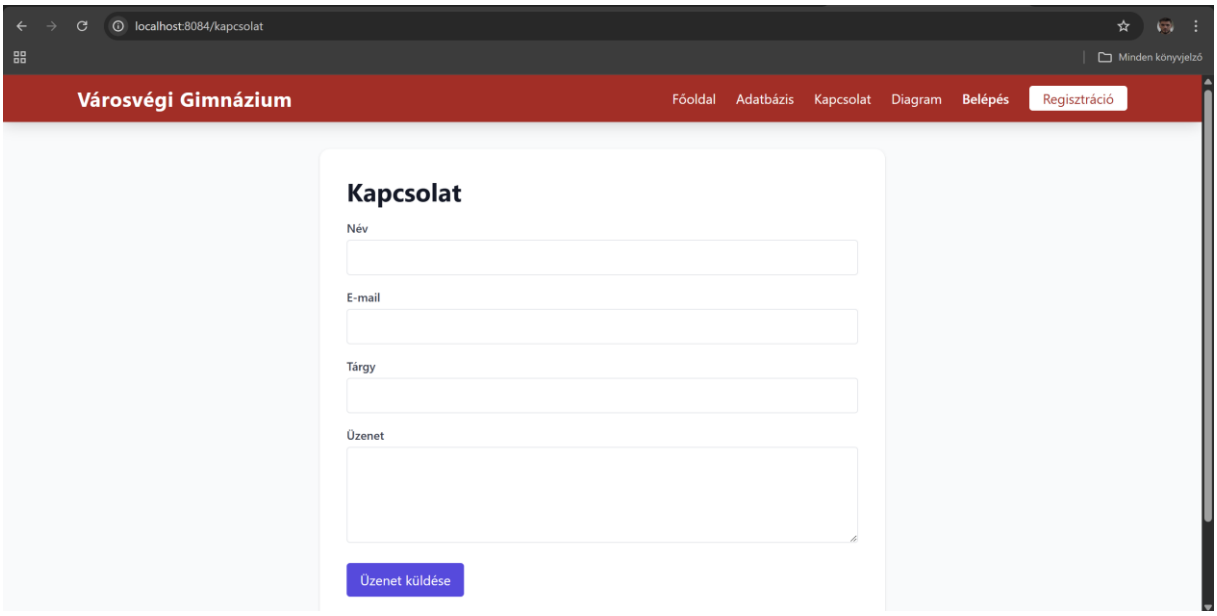
## 10. Összegzés

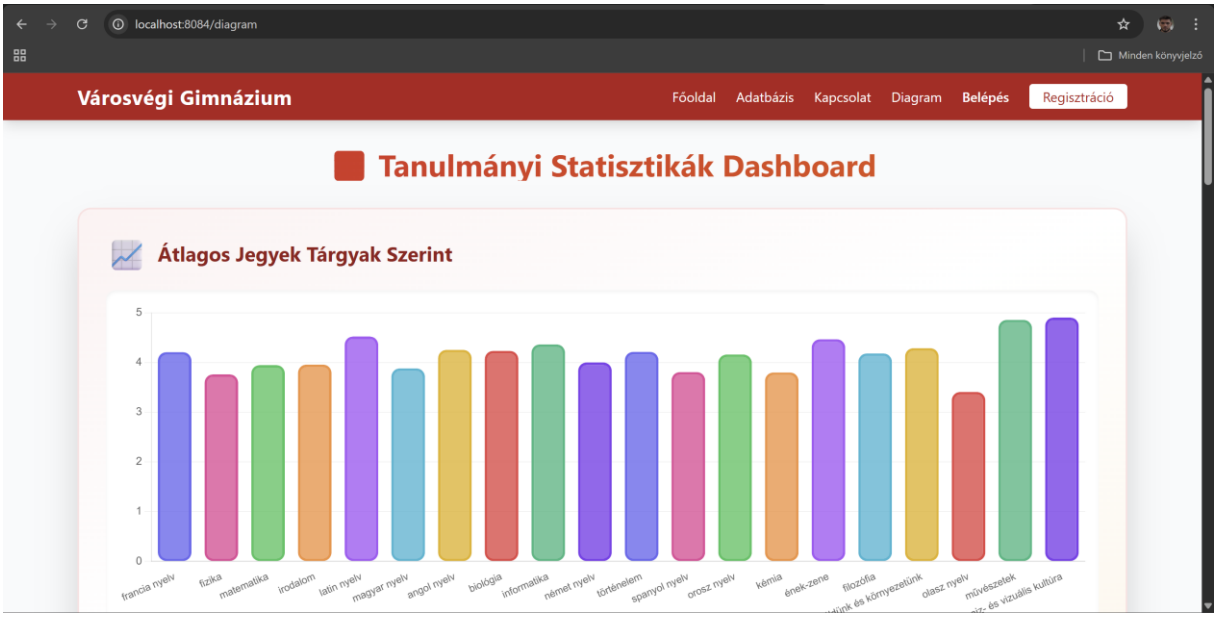
A beadandó feladat során sikeresen megvalósítottunk egy működőképes, többrétegű webalkalmazást Java Spring Boot környezetben. A rendszer teljesíti az összes kötelező és választott követelményt: kezeli a felhasználókat, biztonságos, képes adatokat beolvasni és tárolni, valamint vizuálisan is megjeleníti azokat.

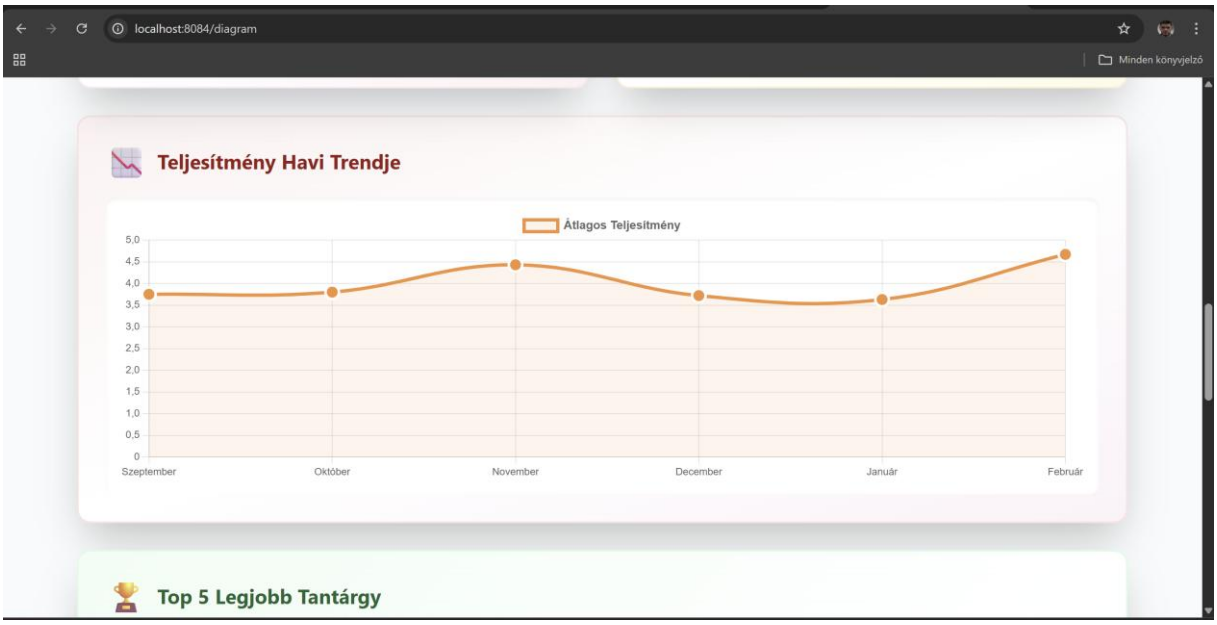
A projekt során elmélyítettük tudásunkat a Spring keretrendszer, a JPA adatbázis-kezelés és a szerver oldali webfejlesztés területén.

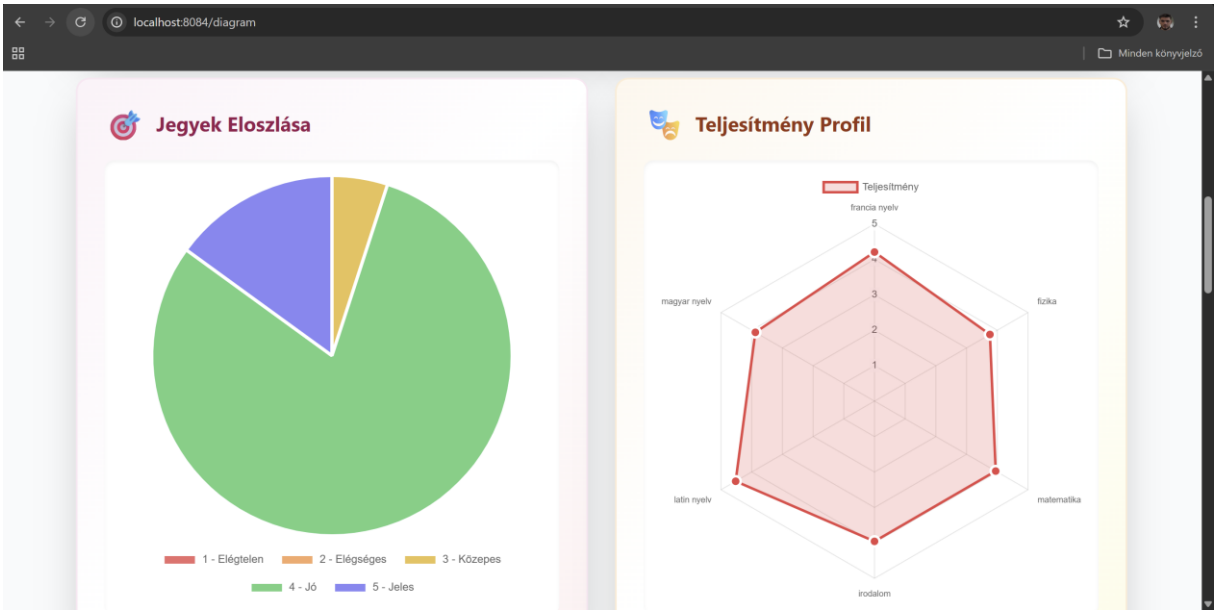


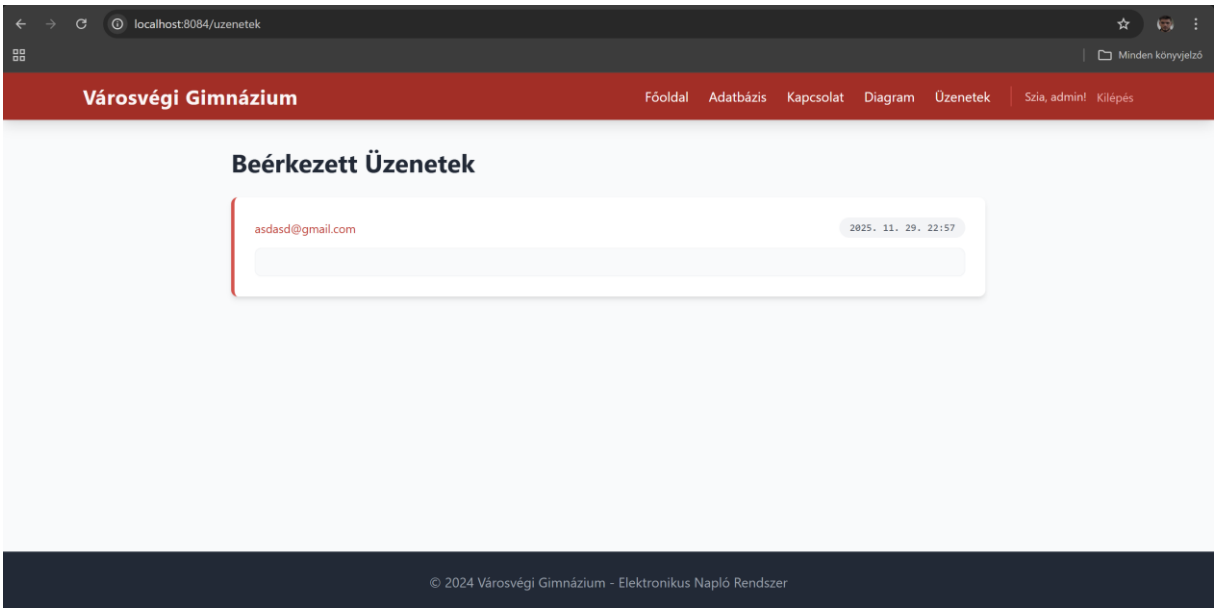



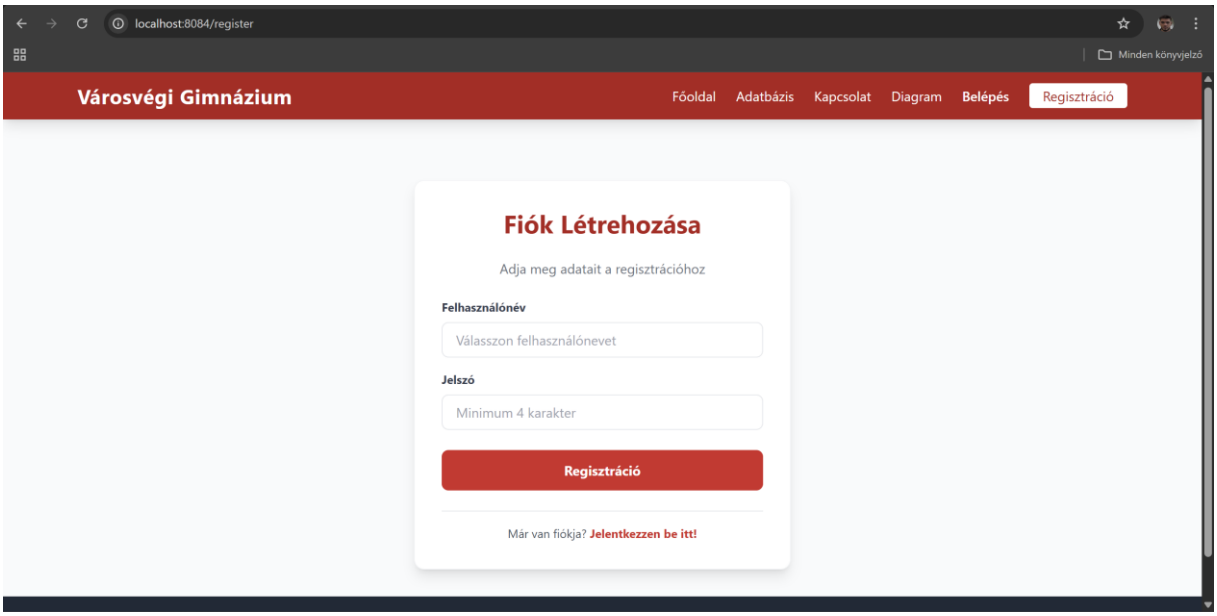

localhost:8084/kapcsolat
Minden könyvjelző
Városvégi Gimnázium
Főoldal
Adatbázis
Kapcsolat
Diagram
Belépés
Regisztráció
Kapcsolat
Név
E-mail
Tárgy
Üzenet
Üzenet küldése


localhost:8084/diagram
Minden könyvjelző
Városvégi Gimnázium
Főoldal
Adatbázis
Kapcsolat
Diagram
Belépés
Regisztráció
Tanulmányi Statisztikák Dashboard
Átlagos Jegyek Tárgyak Szerint
5
4
3
2
1
0
francia nyelv
fizika
matematika
irodalom
latin nyelv
magyar nyelv
angol nyelv
biológia
informatika
német nyelv
történelem
spanyol nyelv
orosz nyelv
kémia
ének-zene
filozófia
olasz nyelv
művészetek


localhost:8084/diagram
Minden könyvjelző
Teljesítmény Havi Trendje
Átlagos Teljesítmény
5,0
4,5
4,0
3,5
3,0
2,5
2,0
1,5
1,0
0,5
0
Szeptember
Október
November
December
Január
Február
Top 5 Legjobb Tantárgy


localhost:8084/diagram
Minden könyvjelző
Jegyek Eloszlása
1 - Elégtelen
2 - Elégséges
3 - Közepes
4 - Jó
5 - Jeles
Teljesítmény Profil
Teljesítmény
francia nyelv
fizika
matematika
irodalom
latin nyelv
magyar nyelv
5
3
2
1


localhost:8084/uzenetek
Minden könyvjelző
Városvégi Gimnázium
Főoldal
Adatbázis
Kapcsolat
Diagram
Üzenetek
Szia, admin!
Kilépés
Beérkezett Üzenetek
asdasd@gmail.com
2025. 11. 29. 22:57
© 2024 Városvégi Gimnázium - Elektronikus Napló Rendszer


localhost:8084/register
Minden könyvjelző
Városvégi Gimnázium
Főoldal
Adatbázis
Kapcsolat
Diagram
Belépés
Regisztráció
Fiók Létrehozása
Adja meg adatait a regisztrációhoz
Felhasználónév
Válasszon felhasználónevet
Jelszó
Minimum 4 karakter
Regisztráció
Már van fiókja? Jelentkezzen be itt!