

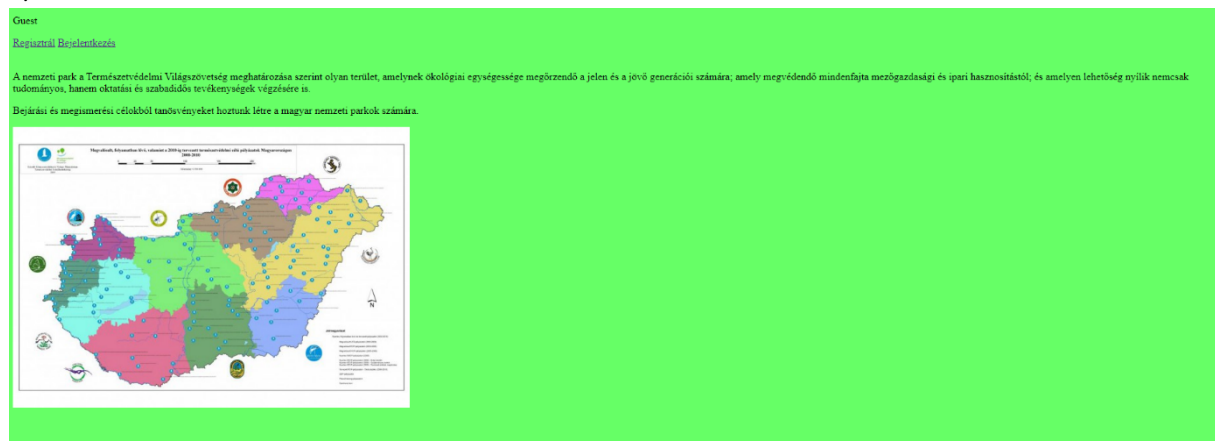
JAVA Gyakorlat beadandó – Munkhárt Levente, Marton Tamás

<https://github.com/TumzunAGozi/java-beadando-feladat>

1. Az első oldalon mutassa be a céget egy látványos weboldalon

Első oldal/Kezdőoldal kód (index.html):

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Guest Home</title>
</head>
<body style="background-color:rgba(0,255,0,0.6)">
<p>Guest</p>
<div th:insert="menu"></div>
<br />
<p>A nemzeti park a Természetvédelmi Világszövetség meghatározása szerint olyan terület, amelynek
ökológiai egységessége megőrzendő a jelen és a jövő generációi számára; amely megvédendő
mindenfajta mezőgazdasági és ipari hasznosítástól; és amelyen lehetőség nyílik nemcsak tudományos,
hanem oktatási és szabadidős tevékenységek végzésére is.<br /></p>
<p>Bejárési és megismerési célokból tanösvényeket hoztunk létre a magyar nemzeti parkok
számára.</p>
<img src=""https://magyarnemzetiparkok.hu/wp-
content/uploads/2014/03/Tv_i_palyazatok_2002_2010-610x431.jpg"/>
</body>
</html>
```



Login gombbal lehet eljutni a „Belépés” oldalhoz.

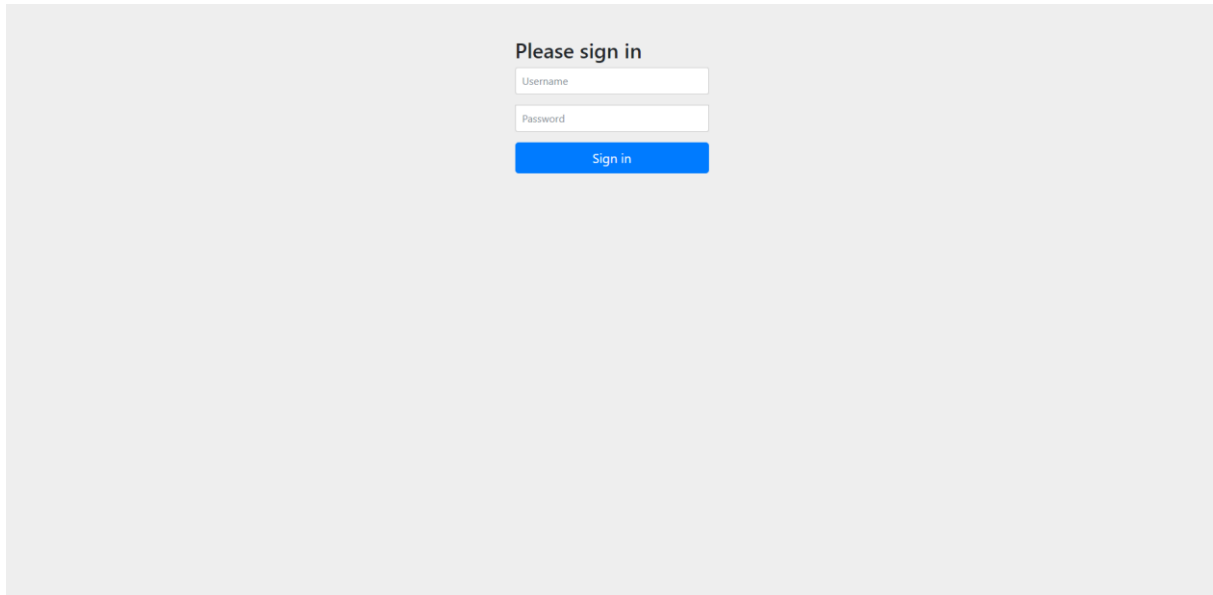
2. Legyen Regisztráció, Bejelentkezési lehetőség

- A „Belépés” menüpont akkor látható, ha nincs bejelentkezve a felhasználó.
- A „Kilépés” menüpont akkor látható, ha be van jelentkezve a felhasználó.

- A „Regisztrál” menüpont a kezdőmenüben mindig látszik.

A rendszer fejlécen jelenítse meg a bejelentkezett felhasználót, ha be van lépve.

A Belépés menüpont:

A screenshot of a login page. At the top, it says "Please sign in". Below this, there are two input fields: "Username" and "Password". Under the "Password" field is a blue button with the text "Sign in". The entire form is centered on a light gray background.

Kód:

```
<div xmlns:th="http://www.thymeleaf.org"
  xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
  <div>
    <span sec:authorize="isAnonymous()">
      <a th:href="@{/login}">Login</a>
    </span>
    <span sec:authorize="isAuthenticated()">
      <a th:href="@{/home}">Home</a>
      <a th:href="@{/logout}">Logout</a>
    </span>
    <span sec:authorize="hasRole('ROLE_ADMIN')">
      <a th:href="@{/admin/home}">Admin</a>
    </span>
  </div>
  <div sec:authorize="isAuthenticated()">
    <h3>Welcome <span sec:authentication="principal.username">User</span></h3>
  </div>
</div>
```

Kilépés menüpont mind a „user” és „admin” oldalon látszik és kilép a kezdőoldalra.

Admin:

[Kezdőlap](#) [Kijelentkezés](#) Admin

Üdv admin@gmail.com

Only Admin can see this page

Kód:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Admin Home</title>
</head>
<body>
  <div th:insert="menu"></div>
  <h1>Only Admin can see this page</h1>
</body>
</html>
```

User:

[Home](#) [Logout](#)

Welcome user@gmail.com

Only logged in users can see this page.

Kód:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org">
<head>
  <title>User Home</title>
</head>
<body>
  <div th:insert="menu"></div>
  <h1>Only logged in users can see this page.</h1>
</body>
</html>
```

3. Legalább 3 felhasználói szerepet különböztessen meg:

Admin, User, Látogató

Admin

email: admin@gmail.com jelszó: jelszo1

User

email: user@gmail.com jelszó: jelszo2

Látogató=Guest

Guest: aaa@gmail.com jelszó: aaa, ezzel nem lehet a loginnál belépni.

9. Használják a GitHub (github.com) verziókövető rendszert.

Branchok: main -> main-edit1 -> main-edit2. Main-edit2 merge-elve lett a main-nel emiatt törölve is lett.

A main a végleges változat.

Végső feladat

Application.properties-nél bemásolva a kért kód.

phpmyAdmin-ról az adatok.sql be van importálva a programba.

A futtat.jar fájl is működik, elindítja a szerveret és a weboldalt.

Egyéb kódok

BeadandoApplication

```
package com.example.beadando;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class BeadandoApplication {
    public BeadandoApplication() {
    }

    public static void main(String[] args) {
        SpringApplication.run(BeadandoApplication.class, args);
    }
}
```

SecurityRoleApplication

```
package com.example.securityrole;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SecurityRoleApplication {

    public static void main(String[] args) {

        SpringApplication.run(SecurityRoleApplication.class, args);
    }
}
```

Foprogram

```
package com.example.securityrole;

public class Foprogram {
    public static void main(String[] args) {
```

```

        SecurityRoleApplication.main(args);
    }
}

```

CustomUserDetailsService

```

package com.example.securityrole;
import java.util.Collection;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.AuthorityUtils;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

@Service
@Transactional
public class CustomUserDetailsService implements UserDetailsService {
    @Autowired
    private UserRepository userRepo;          // Dependency injection
    @Override
    public UserDetails loadUserByUsername(String userName) throws UsernameNotFoundException {
        User user = userRepo.findByEmail(userName)
            .orElseThrow(() -> new UsernameNotFoundException("Email " + userName + " not found"));
        return new org.springframework.security.core.userdetails.User(user.getEmail(),
            user.getPassword(),
            getAuthorities(user));
    }
    private static Collection<? extends GrantedAuthority> getAuthorities(User user) {
        Collection<GrantedAuthority> authorities = AuthorityUtils.createAuthorityList(user.getRole());
        return authorities;
    }
}

```

HomeController

```

package com.example.securityrole;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class HomeController {
    public HomeController() {
    }
}

```

```

@GetMapping("/{")
public String home() {
    return "index";
}

@GetMapping("/{/home"})
public String user() {
    return "user";
}

@GetMapping("/{/admin/home"})
public String admin() {
    return "admin";
}

@GetMapping("/{/guest/home"})
public String guest() {
    return "guest";
}
}

```

User

```

package com.example.securityrole;

import jakarta.persistence.*;

@Entity
@Table(name="users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    private String name;
    private String email;
    private String password;
    private String role;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {

```

```

        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getRole() {
        return role;
    }

    public void setRole(String role) {
        this.role = role;
    }
}

```

UserRepository

```

package com.example.securityrole;

import org.springframework.data.repository.CrudRepository;

import java.util.Optional;

public interface UserRepository extends CrudRepository<User, Integer> {
    Optional<User> findByEmail(String email);
}

```

WebSecurityConfig

```

package com.example.securityrole;

```



```

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.AuthenticationManager;
import
org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfigur
ation;
import
org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;
import org.springframework.security.web.util.matcher.AntPathRequestMatcher;

```

```
@Configuration
```

```
@EnableWebSecurity
```

```
@EnableGlobalMethodSecurity(securedEnabled = true, proxyTargetClass = true)
```

```
public class WebSecurityConfig {
```

```
    @Autowired
```

```
    private UserDetailsService userDetailsService;
```

```
    @Bean
```

```
    public static PasswordEncoder passwordEncoder(){
```

```
        return new BCryptPasswordEncoder();
```

```
    }
```

```
    @Bean
```

```
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
```

```
        http.csrf(csrf -> csrf.disable())
```

```
        .authorizeHttpRequests(
```

```
            auth -> auth
```

```
        .requestMatchers("/resources/**", "/", "/regisztral", "/regisztral_feldolgoz").anonymous()
```

```
        .requestMatchers("/", "/jelszoteszt").anonymous()
```

```
        .requestMatchers("/resources/**", "/", "/home").authenticated()
```

```
        .requestMatchers("/admin/**").hasRole("ADMIN") )
```

```
        .formLogin(
```

```
            form -> form
```

```
        .defaultSuccessUrl("/home").permitAll()
```

```
        ).logout(
```

```
            logout -> logout
```

```
        .logoutRequestMatcher(new AntPathRequestMatcher("/logout"))
```

```
        .logoutSuccessUrl("/")
```

```
        .permitAll()
```

```
        );
```

```
        return http.build();
```

```
    }
```

```

@Bean
public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration)
throws Exception {
    return configuration.getAuthenticationManager();
}
}

```

ApplicationProperties

```

spring.application.name=demo
spring.datasource.url=jdbc:mysql://localhost:3306/adatok
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.format_sql=true

```

Regisztral.html

```

<!DOCTYPE html>
<html xmlns:th="https://www.thymeleaf.org">
<head>
    <title>Bejelentkezés, regisztráció</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<div th:insert="menu"></div>
<h3>Regisztrálja magát, ha még nem felhasználó!</h3>
<form action="#" th:action="@{/regisztral_feldolgoz}" th:object="${reg}" method="post">
    <fieldset>
        <legend>Regisztráció</legend>
        <p><input type="text" th:field="*{name}" placeholder="Név" required/></p>
        <p><input type="text" th:field="*{email}" placeholder="Email" required/></p>
        <p><input type="password" th:field="*{password}" placeholder="Jelszó" required/></p>
        <p><input type="submit" value="Regisztráció" /></p>
    </fieldset>
</form>
</body>
</html>

```

Reghiba.html

```

<!DOCTYPE html>
<html xmlns:th="https://www.thymeleaf.org">

```

```

<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Regisztrációs hiba</title>
</head>
<body>
<div th:insert="menu"></div>
<h1 th:text="{uzenet}" />
<a href="/regisztral">Próbálja újra.</a>
</body>
</html>

```

Regjo.html

```

<!DOCTYPE HTML>
<html xmlns:th="https://www.thymeleaf.org">
<head>
  <title>Sikeres bejelentkezés</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<div th:insert="menu"></div>
<h1 th:utext="'A regisztrációja sikeres. <br/> Azonosítója: ' + ${id}'" />
</body>
</html>

```

Np.java

```
package com.example.securityrole;
```

```
import jakarta.persistence.*;
import java.util.List;
```

```

@Entity
@Table(name="np")
public class Np {
    @Id
    private int id;
    @Column(name = "nev")
    private String nev;

    @OneToMany(mappedBy = "np")
    private List<Telepules> telepules;

    public int getId() {
        return id;
    }
}

```

```

    public void setId(int id) {
        this.id = id;
    }

    public String getNev() {
        return nev;
    }

    public void setNev(String nev) {
        this.nev = nev;
    }

    public List<Telepules> getTelepules() {
        return telepules;
    }

    public void setTelepules(List<Telepules> telepules) {
        this.telepules = telepules;
    }
}

```

NpRepo.java

```

package com.example.securityrole;

import org.springframework.data.repository.CrudRepository;

public interface NpRepo extends CrudRepository<Np, Integer> {

}

```

Telepules.java

```

package com.example.securityrole;

import jakarta.persistence.*;
import java.util.List;

@Entity
@Table(name="telepules")
public class Telepules {
    @Id
    private int id;
    @Column(name="nev")
    private String nev;
    @Column(name="npid")
    private int npid;
}

```

```

@OneToMany(mappedBy = "telepules", cascade = CascadeType.ALL, fetch = FetchType.LAZY)
private List<Ut> ut;

@ManyToOne
@JoinColumn(name = "npid", referencedColumnName = "id", insertable=false, updatable=false)
private Np np;

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getNev() {
    return nev;
}

public void setNev(String nev) {
    this.nev = nev;
}

public int getNpid() {
    return npid;
}

public void setNpid(int npid) {
    this.npid = npid;
}

public List<Ut> getUt() {
    return ut;
}

public void setUt(List<Ut> ut) {
    this.ut = ut;
}

public Np getNp() {
    return np;
}

public void setNp(Np np) {
    this.np = np;
}
}

```

TelepulesRepo.java

```
package com.example.securityrole;

import org.springframework.data.repository.CrudRepository;

public interface TelepulesRepo extends CrudRepository<Telepules, Integer> {

}
```

Ut.java

```
package com.example.securityrole;

import jakarta.persistence.*;
import java.util.List;

@Entity
@Table(name="ut")
public class Ut {
    @Id
    private int azon;
    @Column(name = "nev")
    private String nev;
    @Column(name = "hossz")
    private String hossz;
    @Column(name = "allomas")
    private double allomas;
    @Column(name = "ido")
    private double ido;
    @Column(name = "vezetes")
    private double vezetes;
    @Column(name = "telepulesid")
    private double telepulesid;

    @ManyToOne
    @JoinColumn(name = "telepulesid", referencedColumnName = "id", insertable=false,
updatable=false)
    // @JoinColumn(name = "az", insertable=false, updatable=false)
    private Telepules telepules;

    public int getAzon() {
        return azon;
    }

    public void setAzon(int azon) {
```

```
        this.azon = azon;
    }

    public String getNev() {
        return nev;
    }

    public void setNev(String nev) {
        this.nev = nev;
    }

    public String getHossz() {
        return hossz;
    }

    public void setHossz(String hossz) {
        this.hossz = hossz;
    }

    public double getAllomas() {
        return allomas;
    }

    public void setAllomas(double allomas) {
        this.allomas = allomas;
    }

    public double getIdo() {
        return ido;
    }

    public void setIdo(double ido) {
        this.ido = ido;
    }

    public double getVezetes() {
        return vezetes;
    }

    public void setVezetes(double vezetes) {
        this.vezetes = vezetes;
    }

    public double getTelepulesid() {
        return telepulesid;
    }

    public void setTelepulesid(double telepulesid) {
```

```
        this.telepulesid = telepulesid;
    }

    public Telepules getTelepules() {
        return telepules;
    }

    public void setTelepules(Telepules telepules) {
        this.telepules = telepules;
    }
}
```

UtRepo.java

```
package com.example.securityrole;

import org.springframework.data.repository.CrudRepository;

public interface UtRepo extends CrudRepository<Ut, Integer> {

}
```