Korisnicko ime Unesite korisnicko ime

Sifra Unesite sifru

Zapamti me:

Prijava

Nemate nalog? Registrujte se

1.

Ime:

Prezime:

Korisnicko ime:

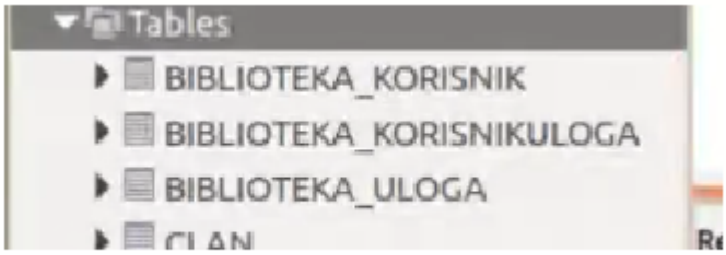
Sifra:

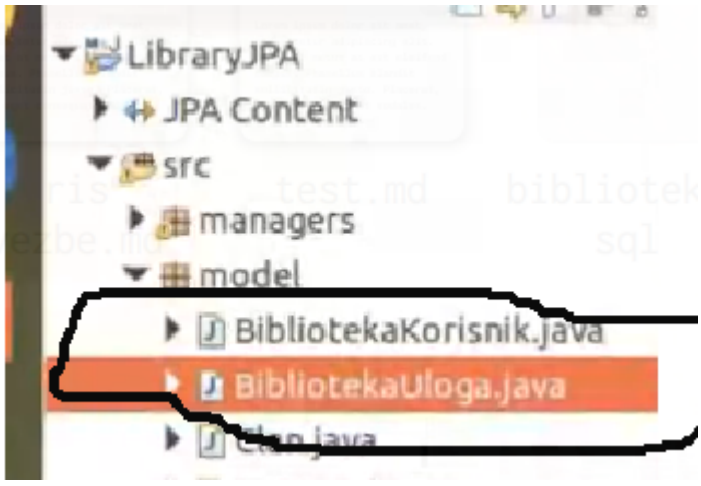
Uloga ADMIN

Sacuvaj

2.

3. 

4. 

5. 

6. JPA Content > pesistance.xml - jel postoji

```
        <class>model.BibliotekaKorisnik</class>
        <class>model.BibliotekaUloga</class>
```

7.
```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-taglibs</artifactId>
</dependency>
```

---

8. UBACIVANJE CSS-a

```
<link rel="stylesheet" type="text/css"
href="${pageContext.request.contextPath}/styles/style.css">
```

---

9. index.jsp

```
...molimo prijavite se klikom na
<a href="/LibraryWeb/auth/loginPage">Link</a>
```

10. LoginControlor.java

```
@Controller
@ControllerAdvice
@RequestMapping(value = "auth")
public class LoginControlor {
    @Autowired
    BibliotekaUlogaRepository r;
    @Autowired
    BibliotekaKorisnikRepository ur;

    @Autowired
    KnjigaRepository kr;

    @ModelAttribute
```

```
    public void getRoles(Model model) {
        List<BibliotekaUloga> roles=r.findAll();
        // 🔽 korisno za combo box
        model.addAttribute("roles", roles);

    }
```

11. isti fajl

```
@RequestMapping(value = "/loginPage", method = RequestMethod.GET)
public String loginPage() {
    System.out.println("CALLED ME!!!!!!!!!!!");
    return "login";

}
```

12. BibliotekaKorisnikRepository.java

```
@Repository
@Transactional
public interface BibliotekaKorisnikRepository extends
JpaRepository<BibliotekaKorisnik, Integer>{
    BibliotekaKorisnik findByKorisnickoIme(String korisnickoIme);
}
```

13. BibliotekaUlogaRepository.java

```
@Repository
@Transactional
public interface BibliotekaUlogaRepository extends
JpaRepository<BibliotekaUloga, Integer>{

}
```

14. LoginControlor.java

```
@RequestMapping(value = "/login", method = RequestMethod.POST)
        public String loginHello() {
        System.out.println("YOU CALLED ME!!!!!!!!!!!");
            return "hello";

        }

        @RequestMapping(value = "/access_denied", method =
```

```
RequestMethod.GET)
        public String deniedPage() {
            return "access_denied";

        }
        @RequestMapping(value = "registerUser", method =
RequestMethod.GET)
        public String newUser(Model model) {
            BibliotekaKorisnik u = new BibliotekaKorisnik();
            model.addAttribute("user", u);
            return "register";
        }

    @RequestMapping(value = "register", method = RequestMethod.POST)
        public String saveUser(@ModelAttribute("user") BibliotekaKorisnik
u) {
            BCryptPasswordEncoder passwordEncoder = new
BCryptPasswordEncoder();
            u.setSifra(passwordEncoder.encode(u.getSifra()));

            for (BibliotekaUloga r : u.getUlogas()) {
                r.addKorisnik(u);

            }
            ur.save(u);
            return "login";

        }

        @RequestMapping(value="/logout", method = RequestMethod.GET)
        public String logoutPage (HttpServletRequest request,
HttpServletResponse response){
            Authentication auth =
SecurityContextHolder.getContext().getAuthentication();
            if (auth != null){

SecurityContextHolder.getContext().setAuthentication(null);
            }
            return "redirect:/auth/loginPage";
        }

    @RequestMapping(value="/pocetna", method = RequestMethod.GET)
    public String getPocetna (){

        return "pocetna";
    }

    @RequestMapping(value="/getSve", method=RequestMethod.GET)
    public String getSve(HttpServletRequest request) {
        List<Knjiga> knjige = kr.findAll();
        request.getSession().setAttribute("knjige", knjige);
        return "sveKnjige";
    }
```

15. webapp > login.jsp

```
...
<div class="main">
    <img src="${pageContext.request.contextPath}/img/knjiga.jpg"
class="center"/>
    <div class="center" style="font-size:25px; text-align:center;">

    <c:url var="loginUrl" value="/login" />
    <c:if test="${not empty param.error}">
        <div class="alert alert-danger">
            <p>Pogresni podaci.</p>
        </div>
    </c:if>
    <form action="${loginUrl}" method="post">
        <table>
            <tr>
                <td>Korisnicko ime</td>
                <td><input type="text" name="username"
                    placeholder="Unesite korisnicko ime" required></td>
            </tr>
            <tr>
                <td>Sifra</td>
                <td><input type="password" name="password"
                    placeholder="Unesite sifru" required></td>
            </tr>
            <tr>
                <td>Zapamti me:</td>
                <td><input type="checkbox" name="remember-me" /></td>
            </tr>
            <tr>
                <td><input type="hidden" name="${_csrf.parameterName}"
                    value="${_csrf.token}" /></td>
                <td><input type="submit" value="Prijava"></td>
            </tr>
        </table><br/><br/>
        Nemate nalog? <a href="/LibraryWeb/auth/registerUser">Registrujte
se</a>
    </form>
    </div>
</div>

</body>
</html>
```

16. security>GeneratePassword.java

```
public class GeneratePassword {
    public static void main(String[] args) {
        String password = "123456";
```

```
        BCryptPasswordEncoder passwordEncoder = new
BCryptPasswordEncoder();
        System.out.println(passwordEncoder.encode(password));
    }
}
```

17.
```
@Service("customUserDetailsService")
public class CustomUserDetailsService implements UserDetailsService{


    @Autowired
    private BibliotekaKorisnikRepository korisnikRepository;
    @Override
    public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
        BibliotekaKorisnik user =
korisnikRepository.findByKorisnickoIme(username);
        UserDetailsImpl userDetails =new UserDetailsImpl();
        userDetails.setUsername(user.getKorisnickoIme());
        userDetails.setPassword(user.getSifra());
        userDetails.setRoles(user.getUlogas());
        return userDetails;

    }
```

18.
```
public class UserDetailsImpl implements UserDetails {


    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private String username;
    private String password;
    private Set<BibliotekaUloga> roles;

    @Override
    public Collection<SimpleGrantedAuthority> getAuthorities() {
        Collection<SimpleGrantedAuthority> authorities = new
ArrayList<SimpleGrantedAuthority>();
        for(BibliotekaUloga r:roles) {
            authorities.add(new
SimpleGrantedAuthority("ROLE_"+r.getNaziv()));
        }
        return authorities;
    }

    @Override
    public String getPassword() {
```

```
        // TODO Auto-generated method stub
        return password;
    }

    @Override
    public String getUsername() {
        // TODO Auto-generated method stub
        return username;
    }

    @Override
    public boolean isAccountNonExpired() {
        // TODO Auto-generated method stub
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
        // TODO Auto-generated method stub
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        // TODO Auto-generated method stub
        return true;
    }

    @Override
    public boolean isEnabled() {
        // TODO Auto-generated method stub
        return true;
    }


    public void setUsername(String username) {
        this.username = username;
    }

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

    public Set<BibliotekaUloga> getRoles() {
        return roles;
    }

    public void setRoles(Set<BibliotekaUloga> roles) {
        this.roles = roles;
    }

}
```

19. CustomUserDetailsService.java opet se gleda

20. SecurityConfig.java

```
@Configuration
@EnableWebSecurity
public class SecurityConfig  extends WebSecurityConfigurerAdapter{

    /*
    * @Override protected void configure(AuthenticationManagerBuilder
auth) throws
    * Exception { auth.inMemoryAuthentication(). withUser("danijela").
    * password("{noop}123456"). roles("ADMIN"); }
    */

    @Autowired
        @Qualifier("customUserDetailsService")
        UserDetailsService userDetailsService;

            @Override
        protected void configure(AuthenticationManagerBuilder auth) throws
Exception {

auth.userDetailsService(userDetailsService).passwordEncoder(new
BCryptPasswordEncoder());

        }


            @Override
            protected void configure(HttpSecurity http) throws Exception {
                http.authorizeRequests()
                .antMatchers("/auth/**") //svaka putanja koja pocinje sa
auth je dostupna svima
                .permitAll()    //dostupno svima
                .antMatchers("/clanovi/**", "/knjige/**", "/reports/**",
"/unos/**")
                .hasAnyRole("BIBLIOTEKAR","ADMIN")
                .and()
                .formLogin()
                .loginPage("/auth/loginPage")
                .loginProcessingUrl("/login")
                .usernameParameter("username")
                .passwordParameter("password")
                .defaultSuccessUrl("/auth/pocetna");
        //      .failureForwardUrl("/index.html")
        //      .and().exceptionHandling()
        //      .accessDeniedPage("/access_denied");
//              .and()
//              .logout()
//              .logoutSuccessUrl("/")
//              .and()
//              .rememberMe().key("uniqueAndSecret");
```

```
            }
    }
```

21. idnex.jsp

```
 <security:authorize access="isAuthenticated()">

    <a href="/LibraryWeb/unos/UnosKnjige.jsp">Unos nove knjige</a>
    <a href="/LibraryWeb/clanovi/unosClana">Unos novog clana</a>
    <a href="/LibraryWeb/reports/SviClanovi.pdf">Svi clanovi - izvestaj</a>
    <a href="/LibraryWeb/ClanoviUPeriodu.jsp">Clanovi u periodu - izvestaj</a>
    <a href="/LibraryWeb/auth/logout">Odjava</a>
</security:authorize>
```