

Java alkalmazások – Gyakorlat - Beadandó feladat -Dokumentáció

Beadandót készítette:

Borbély Bettina -mx4hyq
Slemmer Kevin - lzjiaq

Feladatmegosztás:

Bettina: téma, kapcsolat menü, üzenetek menü, diagram menü, crud menü
Kevin: Autentikáció, főoldal menü, adatbázis menü, admin menü

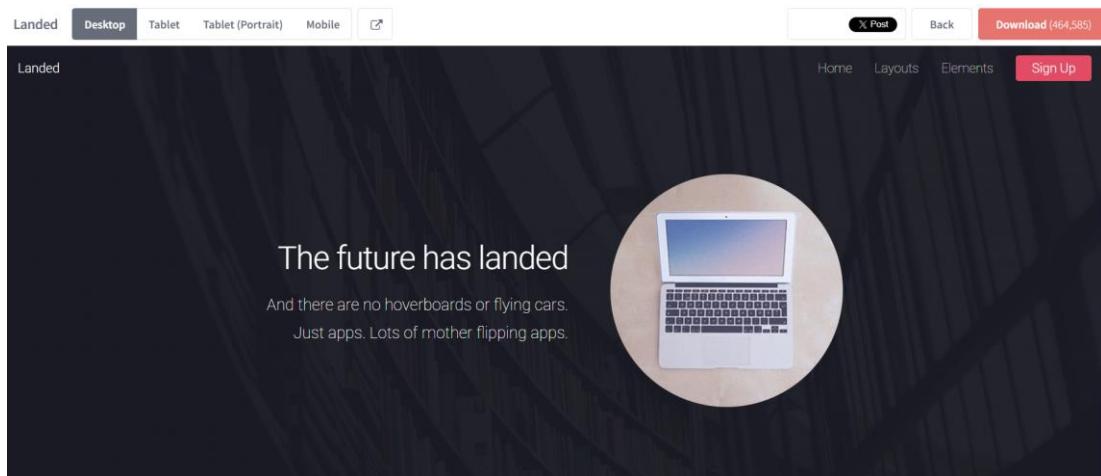
Adatbázis: Városok

Weboldal: <http://rivendell.nje.hu:9443/MX4HYQ-LZJIAQ-03/>

1. Téma:

A weboldalak elkészítéséhez egy ingyenesen elérhető, modern és teljes mértékben reszponzív sablont választottunk. A feladathoz az HTML5 UP - Landed nevű témáját használtuk.

A téma elérhetősége: <https://html5up.net/landed>



2. Autentikáció:

Az alkalmazásban teljes autentikációs rendszer működik, amely a regisztrációt, bejelentkezést és kijelentkezést is tartalmazza. A felhasználók szerepkör alapján különböző jogosultságokat kapnak.

com.example.demo/security/SecurityConfig

```
17 @Configuration
18 public class SecurityConfig {
19
20     private final UserRepository userRepository;  2 usages
21
22     public SecurityConfig(UserRepository userRepository) { this.userRepository = userRepository; }
23
24     @Bean
25     public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
26         http
27             .authorizeHttpRequests( AuthorizationManagerRequestMatcher auth -> auth
28                 // Nyilvános oldalak
29                 .requestMatchers( "/index", "/home", "/about").permitAll()
30                 .requestMatchers( "/contact").permitAll() // csak a kapcsolat form nyilvános
31                 .requestMatchers( "/css/**", "/js/**", "/images/**", "/fonts/**").permitAll()
32                 .requestMatchers( "/register", "/login").permitAll()
33
34             // CSAK bejelentkezett felhasználó
35             .requestMatchers( "/contact/all").hasAnyRole( ...roles: "REGISTERED", "ADMIN")
36             .requestMatchers( "/uzenetek/**").hasAnyRole( ...roles: "REGISTERED", "ADMIN")
37
38             // CSAK admin
39             .requestMatchers( "/admin/**").hasRole("ADMIN")
40
41             // minden más: szabadon marad
42             .anyRequest().permitAll()
43     }
44
45     .formLogin( FormLoginConfigurer<HttpSecurity> form -> form
46         .LoginPage( "/login")
47         .defaultSuccessUrl( defaultSuccessUrl: "/", alwaysUse: true)
48         .permitAll()
49     )
50
51     .logout( LogoutConfigurer<HttpSecurity> logout -> logout
52         .logoutUrl( "/logout")
53         .logoutSuccessUrl( "/logout")
54         .invalidateHttpSession(true)
55         .deleteCookies( ...cookieNamesToClear: "JSESSIONID")
56         .permitAll()
57     )
58
59     .exceptionHandling( ExceptionHandlingConfigurer<HttpSecurity> ex -> ex.accessDeniedHandler(accessDeniedHandler()));
60 }
```

```

59         return http.build();
60     }
61
62
63     @Bean
64     public AccessDeniedHandler accessDeniedHandler() {
65         return ( HttpServletRequest request, HttpServletResponse response, AccessDeniedException accessDeniedException ) ->
66             response.sendRedirect( s: "/" );
67     }
68
69     @Bean
70     public UserDetailsService userDetailsService() {
71         return email -> userRepository.findByEmail(email).Optional<User>
72             .map( User user -> {
73                 String role = user.getRole().name().toUpperCase(); // REGISTERED vagy ADMIN
74                 return new org.springframework.security.core.userdetails.User(
75                     user.getEmail(),
76                     user.getPassword(),
77                     Collections.singletonList(new SimpleGrantedAuthority( role: "ROLE_" + role ))
78                 );
79             }) Optional<User>
80             .orElseThrow(() -> new UsernameNotFoundException("Felhasználó nem található: " + email));
81     }
82
83     @Bean
84     public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }
85     @Bean
86     public org.thymeleaf.extras.springsecurity6.dialect.SpringSecurityDialect springSecurityDialect() {
87         return new org.thymeleaf.extras.springsecurity6.dialect.SpringSecurityDialect();
88     }
89
90 }
91
92 }
93

```

dom.example.demo/repository/UserRepository

```

2 package com.example.demo.repository;
3
4 import com.example.demo.entity.User;
5 import org.springframework.data.jpa.repository.JpaRepository;
6 import java.util.Optional;
7
8     public interface UserRepository extends JpaRepository<User, Long> { 7 usages
9         Optional<User> findByEmail(String email); 2 usages
10    }

```

src/main/java/com/example/demo/entity/User.java

```

1  @Entity
2  @Table(name = "users")
3  @Data
4  public class User {
5
6
7      @Id
8      @GeneratedValue(strategy = GenerationType.IDENTITY)
9      private Long id;
10
11      @NotBlank(message = "Név megadása kötelező")
12      private String name;
13
14      @NotBlank(message = "Email megadása kötelező")
15      @Email(message = "Érvénytelen email cím")
16      @Column(unique = true)
17      private String email;
18
19      @NotBlank(message = "Jelszó megadása kötelező")
20      @Size(min = 6, message = "Jelszó legalább 6 karakter")
21      private String password;
22
23      @Enumerated(EnumType.STRING)
24      private Role role = Role.registered;
25
26      private LocalDateTime emailVerifiedAt;
27      private String rememberToken;
28
29      @Column(name = "created_at")
30      private LocalDateTime createdAt;
31
32      @Column(name = "updated_at")
33      private LocalDateTime updatedAt;
34
35      public enum Role { 3 usages
36          registered, admin
37      }

```

A User entitás egy Role enumot tartalmaz, amely meghatározza a felhasználók jogosultságait:

- registered - regisztrált látogató
- admin - adminisztrátor

A nem bejelentkezett felhasználók automatikusan a „látogató” szerepkörbe tartoznak.

A UserRepository az adatbázisból tölti be a felhasználókat e-mail cím alapján.

A jelszavak BCrypt titkosítással kerülnek mentésre.

A SecurityConfig egy saját UserDetailsService-t használ, amely az adatbázisban tárolt e-mail, jelszó és szerepkör alapján hozza létre a Spring Security által használt felhasználói objektumot.

A biztonsági konfiguráció URL-alapú védelemmel látja el az alkalmazást:

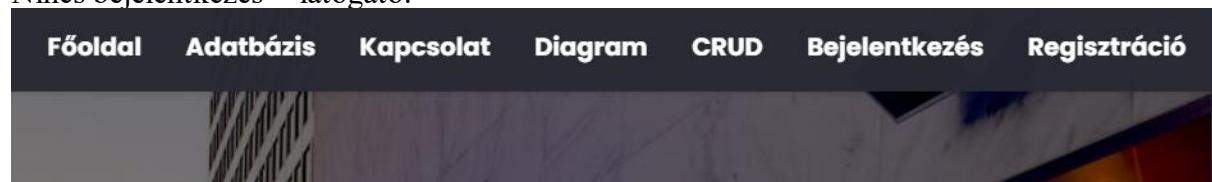
- **Nyilvános oldalak:** /, /home, /about, /contact, /login, /register
- **Regisztrált látogató + admin:** /contact/all, /uzenetek/**
- **Csak admin:** /admin/**

A rendszer egyedi bejelentkezési oldalt használ (**/login**).

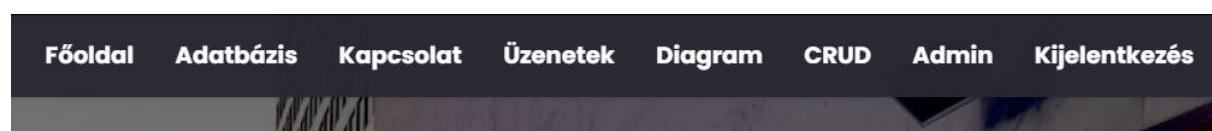
Sikeressé válik a bejelentkezés után a főoldalra irányítja a felhasználót.

A kijelentkezés a **/logout** végponton történik, amely törli a sessiont és a cookie-kat.

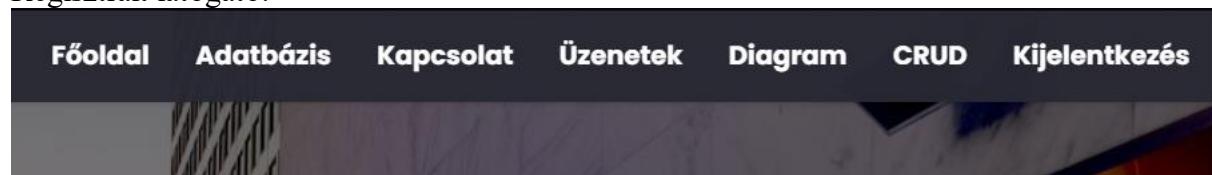
Nincs bejelentkezés – látogató:



Admin:



Regisztrált látogató:



src/main/java/com/example/demo/controller/AuthController.java

```
14  @Controller
15  public class AuthController {
16
17      @Autowired
18      private UserRepository userRepo;
19
20      @Autowired
21      private PasswordEncoder passwordEncoder;
22
23      @GetMapping("/login")
24      public String login() { return "auth/login"; }
25
26
27      @GetMapping("/register")
28      public String showRegisterForm(Model model) {
29          model.addAttribute("user", new User());
30          return "auth/register";
31      }
32
33
34      @PostMapping("/register")
35      public String register(@Valid @ModelAttribute("user") User user,
36                             BindingResult result,
37                             Model model) {
38
39          if (result.hasErrors()) {
40              return "auth/register";
41          }
42
43          if (userRepo.findByEmail(user.getEmail()).isPresent()) {
44              model.addAttribute("error", attributeValue: "Ez az e-mail cím már foglalt!");
45              return "auth/register";
46          }
47
48          user.setPassword(passwordEncoder.encode(user.getPassword()));
49          user.setRole(User.Role.registered);
50
51          userRepo.save(user); // mentés adatbázisba
52
53          return "redirect:/login?registered";
54      }
55  }
```

Az **AuthController** kezeli a felhasználói hitelesítéssel kapcsolatos funkciókat, mint a bejelentkezés és regisztráció. A **/login** végpont visszaadja a bejelentkezási oldalt. A **/register** GET végpont egy új regisztrációs űrlapot jelenít meg. A **/register POST** végpont ellenőrzi a felhasználó adatait, lekezeli a hibákat, ellenőrzi, hogy az e-mail cím már foglalt-e, és ha minden rendben van, a jelszót **titkosítja (PasswordEncoder)** és elmenti az adatbázisba. A sikeres regisztráció után a felhasználó a bejelentkezási oldalra kerül átirányításra.

Bejelentkezási felület:

The screenshot shows a dark-themed login form. At the top, the title 'Bejelentkezés' is displayed. Below it is a horizontal input field labeled 'Email cím'. Underneath is another horizontal input field labeled 'Jelszó'. To the right of the 'Jelszó' field is a white rectangular button with the text 'Belépés' (Login). At the bottom left of the form, there is a link 'Nincs fiókod? [Regisztrálj!](#)' (No account? [Register!](#)).

Regisztrációs felület:

The screenshot shows a registration form with the following fields and buttons:

- Név:** Teljes név
- Email:** Email
- Jelszó:** Jelszó
- Buttons:** Regisztráció, Már van fiókod? Jelentkezz be!

3. Főoldal menü

A főoldal a HorizonTours cég bemutatkozó oldala, látványos hero szekcióval, szlogen és akciógombok megjelenítésével.
A legnépszerűbb úti célok kártyákon, képekkel, rövid leírással és árakkal láthatók.
A menü dinamikusan változik a felhasználó szerepkörétől függően: látogatók, regisztrált felhasználók és adminok különböző menüpontokat látnak.

src/main/java/com/example/demo/controller/HomeController.java

```
8  @Controller
9  public class HomeController {
10
11     @GetMapping("/")
12     public String home(Model model) {
13         model.addAttribute("companyName", "HorizonTours");
14         model.addAttribute("slogan", "Fedezd fel a világ legszebb látnivalóját velünk!");
15         return "fooldal/index"; // → templates/index.html
16     }
17 }
```

src/main/resources/templates/index.html

```
2  <!DOCTYPE html>
3  <html xmlns:th="http://www.thymeleaf.org"
4      xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
5
6  <head th:replace="fragments/head :: head('HorizonTours - Fedezd fel a világot')"></head>
7
8  <body>
9  <!-- Header fragment --&gt;
10 &lt;header th:replace="fragments/header :: header"&gt;&lt;/header&gt;
11
12 &lt;main&gt;
13
14     &lt;!-- Hero szekció --&gt;
15     &lt;section class="hero"&gt;
16         &lt;div class="hero-content"&gt;
17             &lt;h1&gt;HorizonTours&lt;/h1&gt;
18             &lt;p&gt;Fedezd fel a világ legszebb látnivalót magyar idegenvezetővel, kényelmesen és biztonságban&lt;/p&gt;
19             &lt;div class="hero-buttons"&gt;
20                 &lt;a href="#" class="btn primary"&gt;Foglalj most&lt;/a&gt;
21                 &lt;a href="#" class="btn secondary"&gt;Ajánlatok böngészése&lt;/a&gt;
22             &lt;/div&gt;
23         &lt;/div&gt;
24     &lt;/section&gt;
25
26     &lt;!-- Legnépszerűbb úti célok --&gt;
27     &lt;section class="section" id="destinations"&gt;
28         &lt;h2&gt;Legnépszerűbb Úti céljaink 2025-ben&lt;/h2&gt;
29         &lt;div class="destinations-container"&gt;
30
31             &lt;div class="card"&gt;
32                 &lt;img src="https://images.pexels.com/photos/1285625/pexels-photo-1285625.jpeg?auto=compress&amp;cs=tinysrgb&amp;w=1600" alt="Santorini"&gt;
33                 &lt;div class="card-body"&gt;
34                     &lt;h3&gt;Santorini naplemente túra&lt;/h3&gt;
35                     &lt;p&gt;Kék-fehér házak, vulkanikus tengerpart, bortúra és a világ legszebb naplementéje.&lt;/p&gt;
36                     &lt;span class="price"&gt;599 000 Ft/fő - 8 nap/7 éj&lt;/span&gt;
37                     &lt;a href="#" class="btn card-btn"&gt;Érdekel a program&lt;/a&gt;
38                 &lt;/div&gt;
39             &lt;/div&gt;
40
41             &lt;div class="card"&gt;
42                 &lt;img src="https://images.pexels.com/photos/2166559/pexels-photo-2166559.jpeg?auto=compress&amp;cs=tinysrgb&amp;w=1600" alt="Japán"&gt;
43                 &lt;div class="card-body"&gt;
44                     &lt;h3&gt;Cserecsnyevirágzás Japánban&lt;/h3&gt;
45                     &lt;p&gt;Tokió, Kiotó, Hiroshima, szusi workshop és a varázslatos hanami.&lt;/p&gt;
46                     &lt;span class="price"&gt;1 290 000 Ft/fő - 12 nap/11 éj&lt;/span&gt;
47                     &lt;a href="#" class="btn card-btn"&gt;Érdekel a program&lt;/a&gt;
48                 &lt;/div&gt;
49             &lt;/div&gt;
50
51             &lt;div class="card"&gt;
52                 &lt;img src="https://images.pexels.com/photos/753626/pexels-photo-753626.jpeg?auto=compress&amp;cs=tinysrgb&amp;w=1600" alt="Párizs"&gt;
53                 &lt;div class="card-body"&gt;
54                     &lt;h3&gt;Párizsi romantikus hétvége&lt;/h3&gt;
55                     &lt;p&gt;Eiffel-torony, Louvre, hangulatos kávézók és romantikus séták a Szajnánál.&lt;/p&gt;
56                     &lt;span class="price"&gt;450 000 Ft/fő - 4 nap/3 éj&lt;/span&gt;
57                     &lt;a href="#" class="btn card-btn"&gt;Érdekel a program&lt;/a&gt;
58                 &lt;/div&gt;
59             &lt;/div&gt;
60
61         &lt;/div&gt;
62     &lt;/section&gt;
63
64     &lt;/main&gt;
65
66     &lt;!-- Footer fragment --&gt;
67     &lt;footer th:replace="fragments/footer :: footer"&gt;&lt;/footer&gt;</pre>
```

69. sortól stílusok

Ez a HomeController kezeli a főoldal megjelenítését a webalkalmazásban.

A / URL-re érkező kérések esetén a home() metódus fut le.

A metódus a modelhez hozzáadja a cégnév (companyName) és a szlogen (slogan) adatait.

Ezután visszaadja a fooldal/index sablont, amely a főoldal HTML-jét jeleníti meg.

Főoldal Adatbázis Kapcsolat Diagram CRUD Bejelentkezés Regisztráció

HorizonTours

Fedezd fel a világ legszebb látnivalóit magyar idegenvezetővel, kényelmesen és biztonságban

[Foglalj most](#) [Ajánlatok böngészése](#)

Legnépszerűbb úti céljaink 2025-ben



Santorini naplemente túra

Kék-fehér hőszak, vulkanikus tengerpart, bortúra és a világ legszebb naplementéje.

599 000 Ft/fő – 8 nap/7 éj

[Érdekel a program](#)



Cseresznyevirágzás Japánban

Tokió, Kiotó, Hiroshima, szusi workshop és a varázslatos hanami.

1290 000 Ft/fő – 12 nap/11 éj

[Érdekel a program](#)



Párizsi romantikus hétvége

Eiffel-torony, Louvre, hangulatos kártezők és romantikus séták a Szajnánál.

450 000 Ft/fő – 4 nap/3 éj

[Érdekel a program](#)



© Untitled. All rights reserved. | Design: HTML5 UP

4. Adatbázis menü

src/main/java/com/example/demo/controller/TableController.java

```
18  @Controller
19  public class TableController {
20
21  @Autowired
22  private TableService tableService;
23
24  @GetMapping("adatbazis")
25  public String showTablePage(Model model) {
26      model.addAttribute("tables", List.of("varos", "megye", "lelekszam"));
27      return "adatbazis/adatbazis";
28  }
29
30  @PostMapping("adatbazis")
31  public String getTableData(@RequestParam("table") String tableName, Model model) {
32      List<Map<String, Object>> data = tableService.getTableData(tableName);
33
34      List<String> fieldNames = data.isEmpty()
35          ? List.of()
36          : new ArrayList<>(data.get(0).keySet()); // + csak a saját kulcsok!
37
38      List<List<Object>> rowValues = data.stream()
39          .map(Map::entrySet)
40          .map(Map::values)
41          .collect(Collectors.toList());
42
43      model.addAttribute("tables", List.of("varos", "megye", "lelekszam"));
44      model.addAttribute("selectedTable", tableName);
45      model.addAttribute("fieldNames", fieldNames);
46      model.addAttribute("rowValues", rowValues);
47
48      return "adatbazis/adatbazis";
49  }
```

// src/main/java/com/example/demo/service/TableService.java

```
15  @Service 2 usages
16  public class TableService {
17
18  @Autowired
19  private VarosRepository varosRepo;
20
21  @Autowired
22  private MegyeRepository megyeRepo;
23
24  @Autowired
25  private LelekszamRepository lelekszamRepo;
26
27  @V  public List<Map<String, Object>> getTableData(String tableName) { 1 usage
28      List<Map<String, Object>> data = switch (tableName) {
29          case "varos" -> varosRepo.findAll();
30          case "megye" -> megyeRepo.findAll();
31          case "lelekszam" -> lelekszamRepo.findAll();
32          default -> throw new IllegalArgumentException("Ervénytelen tábla név");
33      };
34
35      List<Map<String, Object>> result = new ArrayList<>();
36
37      for (Object obj : data) {
38          Map<String, Object> map = new LinkedHashMap<>();
39          for (Field field : obj.getClass().getDeclaredFields()) {
40              field.setAccessible(true);
41              try {
42                  map.put(field.getName(), field.get(obj));
43              } catch (IllegalAccessException e) {
44                  map.put(field.getName(), null);
45              }
46          }
47          result.add(map);
48      }
49
50      return result;
51  }
52 }
```

A kód egy dinamikus adatbázis-megjelenítő felületet valósít meg, ahol a felhasználó kiválaszthatja, melyik tábla (varos, megye, lelekszam) adatait szeretné látni. A **TableController** kezeli az oldal megjelenítését és a kiválasztott tábla adatainak lekérését. A **TableService** a választott tábla összes rekordját lekéri az adatbázisból.

Válassz táblát

varos

Listáz

id	nev	megyeid	megyeszekhely	megyeijogu	created_at	updated_at
1	Mindszent	8	0	0		
2	Komádi	16	0	0		
3	Tiszacsege	16	0	0		
4	Nyíradony	16	0	0		
5	Rákóczifalva	15	0	0		
6	Mezőkeresztes	3	0	0		

5. Kapcsolat menü:

Lehetővé teszi a felhasználók számára üzenetek küldését az oldal tulajdonosának. Az űrlap adatainak **szerver oldali validációja** biztosítja a helyes és biztonságos adatbevitelét, míg az **adatbázisba mentés** lehetővé teszi az üzenetek nyomon követését és későbbi feldolgozását.

src/main/java/com/example/demo/controller/ContactController.java

```
11  @Controller
12  @RequestMapping("/*"/contact")
13  public class ContactController {
14
15      private final MessageService messageService;  3 usages
16
17      public ContactController(MessageService messageService) {
18      this.messageService = messageService;
19  }
20
21  // Kapcsolat űrlap megjelenítése
22  @GetMapping/*
23  public String showForm(Model model) {
24      model.addAttribute( attributeName: "message", new Message());
25      return "contact/form";
26  }
27
28  // Űrlap elküldése
29  @PostMapping/*
30  public String submitForm(@ModelAttribute Message message, Model model) {
31      messageService.save(message);
32      model.addAttribute( attributeName: "success", attributeValue: "Üzenet sikeresen elküldve!");
33      model.addAttribute( attributeName: "message", new Message()); // Üritjük az űrlapot
34      return "contact/form";
35  }
36
37  @GetMapping("/*"/all")
38  public String listMessages(Model model) {
39      List<Message> messages = messageService.findAllMessagesOrdered();
40      model.addAttribute( attributeName: "messages", messages);
41      return "contact/list";
42  }
```

src/main/java/com/example/demo/service /MessageService.java

```
9   @Service 3 usages
10  public class MessageService {
11
12      private final MessageRepository messageRepository;  4 usages
13
14      public MessageService(MessageRepository messageRepository) {
15      this.messageRepository = messageRepository;
16  }
17
18      public Message save(Message message) {
19      return messageRepository.save(message);
20  }
21
22      public List<Message> findAll() { no usages
23      return messageRepository.findAll();
24  }
25      public List<Message> findAllMessagesOrdered() { 1 usage
26      List<Message> messages = messageRepository.findAll();
27      messages.sort(( Message m1, Message m2) -> m2.getCreatedAt().compareTo(m1.getCreatedAt()));
28      return messages;
29  }
30  }
```

src/main/java/com/example/demo/repository /MessageRepository.java

```
public interface MessageRepository extends JpaRepository<Message, Long> { 3 usages
    List<Message> findAllByOrderByCreatedAtDesc();  no usages
}
```

src/main/java/com/example/demo/model /Message.java

```
6     @Entity
7     @Table(name = "uzenetek")
8     public class Message {
9
10    @Id
11    @GeneratedValue(strategy = GenerationType.IDENTITY)
12    private Long id;
13
14    private String nev;  2 usages
15
16    private String email;  2 usages
17
18    private String varos;  2 usages
19
20    private String kor;  2 usages
21
22    private String uzenet;  2 usages
23
24    private LocalDateTime createdAt;  3 usages
25
26    private LocalDateTime updatedAt;  2 usages
27
28    // Konstruktor
29    public Message() {
30        this.createdAt = LocalDateTime.now();
31    }
32
33
34    // Getterek és setterek
35    public Long getId() { return id; }
36    public void setId(Long id) { this.id = id; }
37
38    public String getNev() { return nev; }
39    public void setNev(String nev) { this.nev = nev; }
40
41    public String getEmail() { return email; }
42    public void setEmail(String email) { this.email = email; }
43
44    public String getVaros() { return varos; }  no usages
45    public void setVaros(String varos) { this.varos = varos; }  no usages
46
47    public String getKor() { return kor; }
48    public void setKor(String kor) { this.kor = kor; }
49
50    public String getUzenet() { return uzenet; }  no usages
51    public void setUzenet(String uzenet) { this.uzenet = uzenet; }  no usages
52
53    public LocalDateTime getCreatedAt() { return createdAt; }  2 usages
54    public void setCreatedAt(LocalDateTime createdAt) { this.createdAt = createdAt; }  no usages
55
56    public LocalDateTime getUpdatedAt() { return updatedAt; }  no usages
57    public void setUpdatedAt(LocalDateTime updatedAt) { this.updatedAt = updatedAt; }  no usages
58
59 }
```

src/main/resources/templates/contact/form.html

```
1  <!DOCTYPE HTML>
2  <html xmlns:th="http://www.thymeleaf.org"
3  	xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
4
5  <head th:replace="~{fragments/head :: head('Kapcsolatfelvétel')}">
6  	<style>
7
8  	</style>
9  </head>
10
11
12  <body class="is-preload landing">
13  <div id="page-wrapper">
14
15
16      <!-- HEADER (menü + login/logout) -->
17      <th:block th:replace="~{fragments/header :: header}"></th:block>
18
19      <!-- FŐ TARTALOM -->
20      <section id="one" class="wrapper style1 special">
21          <div class="container">
22              <header class="major">
23                  <h2>Kapcsolatfelvétel</h2>
24                  <p>Írd meg nekünk, mit szeretnél!</p>
25              </header>
26
27
28              <!-- SIKERES ÜZENET -->
29              <div th:if="${success}" class="success">
30                  <p th:text="${success}"></p>
31              </div>
32
33
34              <!-- ÚRLAP -->
35              <div class="box">
36                  <form th:action="@{/contact}" th:object="${message}" method="post">
37                      <div class="row gtr-uniform">
38
39                          <div class="col-6 col-12-xsmall">
40                              <label for="nev">Név *</label>
41                              <input type="text" th:field="*{nev}" id="nev" required />
42                          </div>
43
44                          <div class="col-6 col-12-xsmall">
45                              <label for="email">Email *</label>
46                              <input type="email" th:field="*{email}" id="email" required />
47                          </div>
48
49                          <div class="col-6 col-12-xsmall">
50                              <label for="varos">Város *</label>
51                              <input type="text" th:field="*{varos}" id="varos" required />
52                          </div>
53
54                          <div class="col-6 col-12-xsmall">
55                              <label for="kor">Kor *</label>
56                              <input type="text" th:field="*{kor}" id="kor" placeholder="pl. 25" required />
57                          </div>
58
59                          <div class="col-12">
60                              <label for="uzenet">Üzenet *</label>
61                              <textarea th:field="*{uzenet}" id="uzenet" rows="6" required></textarea>
62                          </div>
63
64                          <div class="col-12">
65                              <button type="submit" class="button primary fit">Küldés</button>
66                          </div>
67
68                      </div>
69                  </form>
70              </div>
71          </section>
72
73
74      <!-- FOOTER -->
75      <th:block th:replace="~{fragments/footer :: footer}"></th:block>
```

A **ContactController** kezeli az űrlap megjelenítését, az üzenetek elküldését és az összes üzenet listázását. Az üzeneteket a **Message** entitás tárolja az adatbázisban, beleértve a felhasználó nevét, email címét, városát, korát és az üzenet tartalmát. A **MessageService** és a **MessageRepository** biztosítják az üzenetek mentését és lekérdezését, valamint az időrendi sorrend szerinti megjelenést. A szerver oldali validáció bevezetésével az űrlapadatok biztonságosan ellenőrizhetők és menthetők az adatbázisba.

The screenshot shows a contact form titled "Kapcsolatfelvétel". The header includes the text "Írd meg nekünk, mit szeretnél!" and a red horizontal bar. The form fields are arranged in two columns: "Név *" and "Email *", both with input boxes. Below them are "Város *" and "Kor *", with "Város *" having a dropdown menu containing "pl. 25". The bottom section is a large "Üzenet *" area with a scroll bar. At the bottom is a red "Küldés" button.

6. Üzenetek menü

A felhasználók táblázatban tekinthetik meg az adatbázisba mentett üzeneteket, fordított időrendben, a legfrissebb üzenet elől. minden üzenet mellett megjelenik a küldés időpontja is. A menü csak bejelentkezett felhasználók számára érhető el, így az üzenetek biztonságosan kezelhetők.

src/main/java/com/example/demo/controller/ContactController.java

```
37     @GetMapping("/{all}")
38     public String listMessages(Model model) {
39         List<Message> messages = messageService.findAllMessagesOrdered();
40         model.addAttribute("attributeName: "messages", messages);
41         return "contact/list";
42     }
```

src/main/java/com/example/demo/repository/MessageRepository.java

```
8  public interface MessageRepository extends JpaRepository<Message, Long> { 3 usages
9      List<Message> findAllByOrderByCreatedAtDesc(); no usages
10 }
```

src/main/resources/templates/contact/list.html

```
1      <!DOCTYPE HTML>
2      <html xmlns:th="http://www.thymeleaf.org"
3          xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
4
5      <head th:replace="~{fragments/head :: head('Beérkezett Üzenetek')}">
6          <style>
7
8          </style>
9      </head>
10
11     <body class="is-preload landing">
12         <div id="page-wrapper">
13
14             <th:block th:replace="~{fragments/header :: header}"></th:block>
15
16             <div class="card">
17                 <div class="card-header">
18                     <h1>Beérkezett Üzenetek</h1>
19                 </div>
20
21                 <div th:if="${messages == null or messages.isEmpty()}" class="empty">
22                     Még nincs beérkezett üzenet
23                 </div>
24
25                 <div style="..." th:if="${messages != null and !messages.isEmpty()}">
26                     <table>
27                         <thead>
28                             <tr>
29                                 <th>Név</th>
30                                 <th>Email</th>
31                                 <th>Város</th>
32                                 <th>Kor</th>
33                                 <th>Üzenet</th>
34                                 <th class="timestamp">Elküldve</th>
35                             </tr>
36                         </thead>
37                         <tbody>
38                             <tr th:each="m : ${messages}">
39                                 <td th:text="${m.nev}">Név</td>
40                                 <td th:text="${m.email}">email@peelda.hu</td>
41                                 <td th:text="${m.varos}">Város</td>
42                                 <td th:text="${m.kor}">18</td>
43                                 <td th:text="${m.uzenet}">Üzenet...</td>
44                                 <td class="timestamp" th:text="#{temporals.format(m.createdAt, 'yyyy.MM.dd HH:mm')}">
45                                     2025.12.08 14:20
46                                 </td>
47                             </tr>
48                         </tbody>
49                     </table>
50
51                 </div>
52             </div>
53
54             <th:block th:replace="~{fragments/footer :: footer}"></th:block>
55
56         </div>
57
58     </body>
59
60 
```

A ContactController GET /contact/all útvonalán lekéri az összes üzenetet az adatbázisból a MessageService segítségével, időrend szerint (legfrissebb elől), és a contact/list nézetnek továbbítja. A Message entitás tárolja a felhasználó nevét, email címét, városát, korát, az üzenet tartalmát, valamint a létrehozás és frissítés idejét. Egy táblázatban jeleníti meg az üzeneteket, minden sorhoz kiírva az üzenet küldésének időpontját. Ha még nincs üzenet, a felhasználó egy „Még nincs beérkezett üzenet” üzenetet lát.

Név	Email	Város	Kor	Üzenet	Elküldve
Teszt	tesztelek@gmail.com	Iregszemcse	21	Teszt betti felé	2025.12.08 01:49

7. Diagram menü

A diagram a magyarországi megyékhez tartozó városok számát ábrázolja.

src/main/java/com/example/demo/controller/ChartController.java

```

14  public class ChartController {
15
16     private final CountyRepository countyRepository; 2 usages
17     private final CityRepository cityRepository; 2 usages
18
19     public ChartController(CountyRepository countyRepository, CityRepository cityRepository) {
20         this.countyRepository = countyRepository;
21         this.cityRepository = cityRepository;
22     }
23
24     @GetMapping("/{chart}")
25     public String chart(Model model) {
26
27         List<County> counties = countyRepository.findAll();
28
29         List<CountyData> countyDataList = counties.stream() Stream<County>
30             .map(County c -> new CountyData(c.getNev(), cityRepository.findAllByMegye_Id(c.getId()).size())) Stream<CountyData>
31             .sorted((CountyData a, CountyData b) -> Integer.compare(b.getCount(), a.getCount())) // csökkenő sorrend
32             .toList();
33
34         List<String> labels = countyDataList.stream() Stream<CountyData>
35             .map(CountyData::getName) Stream<String>
36             .toList();
37
38         List<Integer> data = countyDataList.stream() Stream<CountyData>
39             .map(CountyData::getCount) Stream<Integer>
40             .toList();
41
42         model.addAttribute("labels", labels);
43         model.addAttribute("data", data);
44
45         return "diagram/cities";
46     }
47
48     // Segédosztály a párosításra
49     public static class CountyData { 4 usages
50         private final String name; 2 usages
51         private final int count; 2 usages
52
53         public CountyData(String name, int count) { 1 usage
54             this.name = name;
55             this.count = count;
56         }
57
58         public String getName() {
59             return name;
60         }
61
62         public int getCount() { 3 usages
63             return count;
64         }
65     }
66 }

```

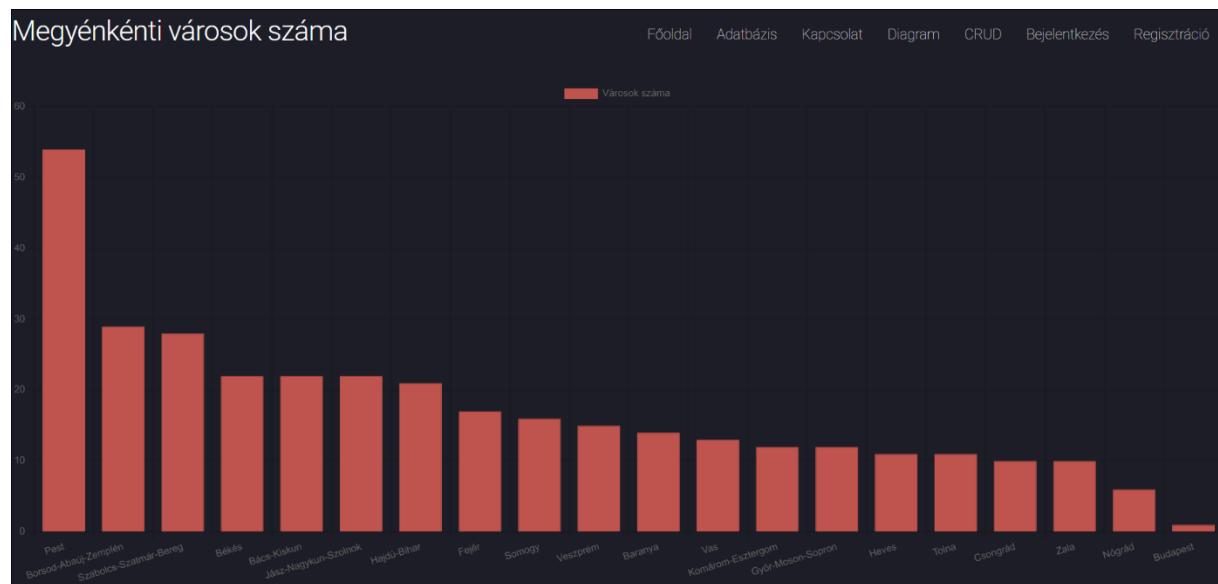
```
// src/main/java/com/example/demo/repository/CountyRepository.java
```

```
3 > import ...  
5  
6 Q public interface CountyRepository extends JpaRepository<County, Long> { 7 usages  
7     Long id(Long id);  
8 }
```

```
// src/main/java/com/example/demo/repository/CityRepository.java
```

```
3 > import ...  
8  
9 @Repository 7 usages  
10 Q public interface CityRepository extends JpaRepository<City, Long> {  
11     List<City> findAllByMegye_Id(int megyeId); 1 usage  
12 }
```

A vezérlő lekéri az összes megyét, majd minden megyéhez meghatározza, hány város tartozik hozzá. Az adatokat egy segédosztályba rendezi és a városok száma szerint csökkenő sorrendbe rakja. Ezután külön listába gyűjti a megyék neveit és a hozzájuk tartozó városszámokat, hogy diagramhoz felhasználhatók legyenek. A lekészített adatokat hozzáadja a modellhez, majd visszatér a „**diagram/cities**” nézethez, ahol a grafikon megjelenik. A **CountyRepository** egy JPA adattároló, a **CityRepository** pedig a megyékhez tartozó városok lekérdezését végzi.



8. CRUD menü

Az oldalon egy teljes CRUD funkciót valósítottunk meg az adatbázis város táblájához, amely lehetővé teszi az adatok megjelenítését, új rekordok felvitelét, valamint a meglévő rekordok módosítását és törlését.

src/main/java/com/example/demo/controller/CityCrudController.java

```
14  @Controller
15  @RequestMapping(@"/crud")
16  public class CityCrudController {
17
18      private final CityService cityService; 7 usages
19      private final CountyRepository countyRepository; 6 usages
20
21  public CityCrudController(CityService cityService, CountyRepository countyRepository) {
22      this.cityService = cityService;
23      this.countyRepository = countyRepository;
24  }
25
26  // --- listázás (Read) ---
27  @GetMapping(@"/")
28  public String listCities(Model model) {
29      List<City> cities = cityService.findAll();
30      model.addAttribute( attributeName: "cities", cities);
31      return "crud/list";
32  }
33
34  // --- Új létrehozása (Create) ---
35  @GetMapping(@"/new")
36  public String newCityForm(Model model) {
37      model.addAttribute( attributeName: "city", new City());
38      model.addAttribute( attributeName: "counties", countyRepository.findAll());
39      return "crud/new";
40  }
41
42  @PostMapping(@")
43  public String createCity(@Valid @ModelAttribute City city,
44      BindingResult bindingResult,
45      @RequestParam Long megye,
46      Model model) {
47
48      if (bindingResult.hasErrors()) {
49          model.addAttribute( attributeName: "counties", countyRepository.findAll());
50          return "crud/new";
51      }
52
53      // ID automatikusan generálódik az adatbázisban
54      city.setVersion(null);
55
56      city.setMegeye(countyRepository.findById(megye)
57          .orElseThrow(() -> new IllegalArgumentException("Invalid county ID: " + megye)));
58
59      cityService.save(city);
60      return "redirect:/crud";
61  }
62
63  // --- Szerkesztés (Update) ---
64  @GetMapping(@"/edit/{id}")
65  public String editCityForm(@PathVariable Long id, Model model) {
66      City city = cityService.findById(id);
67      if (city == null) {
68          throw new IllegalArgumentException("Invalid city Id: " + id);
69      }
70      model.addAttribute( attributeName: "city", city);
71      model.addAttribute( attributeName: "counties", countyRepository.findAll());
72      return "crud/edit";
73  }
74
75  @PostMapping(@"/update/{id}")
76  public String updateCity(@PathVariable Long id,
77      @ModelAttribute City cityForm,
78      @RequestParam Long megye) {
79
80      City city = cityService.findById(id);
81      if (city == null) {
82          throw new IllegalArgumentException("Invalid city Id: " + id);
83      }
84
85      // Fisszítjük csak a szükséges mezőket
86      city.setNevicityForm.getNev());
87      city.setMegyeszekhely(cityForm.getMegyeszekhely());
88      city.setMegyejogu(cityForm.getMegyejogu());
89      city.setMegeye(countyRepository.findById(megye)
90          .orElseThrow(() -> new IllegalArgumentException("Invalid county ID: " + megye)));
91
92      cityService.save(city);
93      return "redirect:/crud";
94  }
95
96  // --- Törlesz (Delete) ---
97  @GetMapping(@"/delete/{id}")
98  public String deleteCity(@PathVariable Long id) {
99      cityService.deleteById(id);
100     return "redirect:/crud";
101 }
```

src/main/java/com/example/demo/service/CityService.java

```
9  public interface CityService { 4 usages 1 implementation
10     List<City> findAll(); 1 usage 1 implementation
11     City findById(Long id); 2 usages 1 implementation
12     City save(City city); 1 implementation
13     void deleteById(Long id); 1 usage 1 implementation
14     boolean existsById(Long id); no usages 1 implementation
15     Map<Long, Long> getCountByCounty(); no usages 1 implementation
16 }
```

src/main/java/com/example/demo/service/CityServiceImpl.java

```
13 @Service
14 public class CityServiceImpl implements CityService {
15
16     private final CityRepository cityRepository; 7 usages
17
18     public CityServiceImpl(CityRepository cityRepository) { this.cityRepository = cityRepository; }
19
20     @Override 1 usage
21     public List<City> findAll() { return cityRepository.findAll(); }
22
23     @Override 2 usages
24     public City findById(Long id) { return cityRepository.findById(id).orElse(null); }
25
26     @Override
27     public City save(City city) { return cityRepository.save(city); }
28
29     @Override 1 usage
30     public void deleteById(Long id) { cityRepository.deleteById(id); }
31
32     @Override no usages
33     public boolean existsById(Long id) { return cityRepository.existsById(id); }
34
35     @Override no usages
36     public Map<Long, Long> getCountByCounty() {
37         return cityRepository.findAll().stream()
38             .collect(Collectors.groupingBy(
39                 City::getCounty,
40                 Collectors.counting()
41             ));
42     }
43 }
```

A **CityCrudController** kezeli a városokhoz tartozó összes **CRUD** műveletet, vagyis a városok listázását, létrehozását, szerkesztését és törlését. A **CityService** és annak implementációja biztosítja az adatbázissal való kommunikációt, például a lekérdezést, mentést és törlést. A vezérlő a megyeadatokat is betölti a megfelelő űrlapokhoz, így a városok megyéhez rendelhetők

ID	Név	Mező	Megyesékhely	Megyei jogú	Műveletek
1	Mindszent	Csongrád	Nem	Nem	<button>Szerkesztés</button> <button>Törlés</button>
2	Komlódi	Hajdú-Bihar	Nem	Nem	<button>Szerkesztés</button> <button>Törlés</button>
3	Tiszacsege	Hajdú-Bihar	Nem	Nem	<button>Szerkesztés</button> <button>Törlés</button>

Város szerkesztése

Mindszent

ID	Város neve
1	Mindszent

Megye

Csongrád

Megyeszékhely

Nem

Megyei jogú város

Nem

Mentés

Vissza

Új város hozzáadása

Város neve

Megye

-- Válassz megyét --

Megyeszékhely

Nem

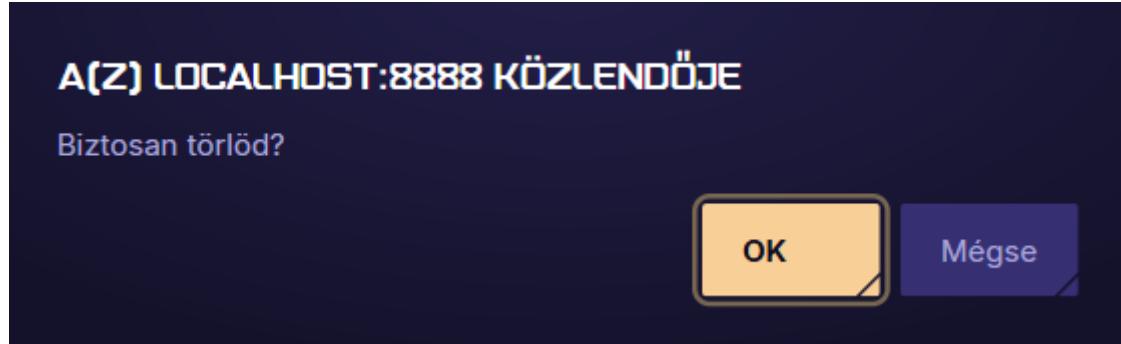
Megyei jogú város

Nem

Mentés

Mégse

Város törlése:



9. RESTful menü

-

10. Admin menü:

Az Admin menü egy olyan külön felület, amely csak bejelentkezett adminisztrátor számára érhető el, így a normál felhasználók nem látják

src/main/java/com/example/demo/controller/AdminController.java

```
4   > import ...  
13  
14     @Controller  
15     @RequestMapping(@"/admin")  
16     public class AdminController {  
17  
18         @Autowired  
19         private UserRepository userRepository;  
20  
21         @GetMapping({@"/", @"/", @"/users"})  
22         public String showUsers(Model model) {  
23             List<User> users = userRepository.findAll();  
24             model.addAttribute( attributeName: "users", users);  
25             model.addAttribute( attributeName: "pageTitle", attributeValue: "Admin - Felhasználók kezelése");  
26             return "admin/users";  
27         }  
28     }
```

Az **AdminController** egy vezérlő, amely az /admin útvonalon elérhető admin felületet kezeli. A **showUsers** metódus lekéri az összes felhasználót a **UserRepository** segítségével.

A lekért felhasználókat és az oldal címét hozzáadja a **Model**-hez.

Visszatér az **admin/users** nézethez, ahol a felhasználók listája megjelenik az **admin** felületen. A menü és a funkciók csak az adminok számára láthatók.

11	Registered	registered@gmail.com	registered
12	admin	Admin@gmail.com	admin