



# Java Alkalmazások Gyakorlat Beadandó Dokumentáció


Github link: <https://github.com/HamNorb/Cukraszda2>

## Adatbázis leírása

A kiindulási adatbázisunk 3 táblát tartalmazott, amelyek a következők voltak:

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	id 	int(11)			Nem	Nincs		
<input type="checkbox"/>	2	sutiid 	int(11)			Nem	Nincs		
<input type="checkbox"/>	3	ertek	int(11)			Nem	Nincs		
<input type="checkbox"/>	4	egyseg	varchar(255)	utf8mb4_general_ci		Igen	NULL		

1. ábra

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	id 	int(11)			Nem	Nincs		
<input type="checkbox"/>	2	sutiid 	int(11)			Nem	Nincs		
<input type="checkbox"/>	3	mentes	varchar(255)	utf8mb4_general_ci		Igen	NULL		
<input type="checkbox"/>	4	nev	varchar(255)	utf8mb4_general_ci		Igen	NULL		
<input type="checkbox"/>	5	suti_id	int(11)			Nem	Nincs		

2. ábra

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1	id 	int(11)			Nem	Nincs		AUTO_INCREMENT
<input type="checkbox"/>	2	nev	varchar(255)	utf8mb4_general_ci		Igen	NULL		
<input type="checkbox"/>	3	tipus	varchar(255)	utf8mb4_general_ci		Igen	NULL		
<input type="checkbox"/>	4	dijazott	tinyint(1)			Nem	Nincs		

3. ábra

Ez lett kiegészítve egy users táblával, hogy megvalósítsuk a felhasználók kezelését, illetve egy uzenet táblával, hogy a kapcsolatok oldalt is megtudjuk valósítani.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1 <b>id</b> 🔑	int(11)			Nem	<i>Nincs</i>		AUTO_INCREMENT
<input type="checkbox"/>	2 <b>name</b>	varchar(255)	utf8mb4_general_ci		Igen	<i>NULL</i>		
<input type="checkbox"/>	3 <b>email</b>	varchar(255)	utf8mb4_general_ci		Igen	<i>NULL</i>		
<input type="checkbox"/>	4 <b>password</b>	varchar(255)	utf8mb4_general_ci		Nem	<i>Nincs</i>		
<input type="checkbox"/>	5 <b>role</b>	varchar(255)	utf8mb4_general_ci		Igen	<i>NULL</i>		

4. ábra

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
<input type="checkbox"/>	1 <b>id</b> 🔑	bigint(20)			Nem	<i>Nincs</i>		AUTO_INCREMENT
<input type="checkbox"/>	2 <b>content</b>	varchar(255)	utf8mb4_general_ci		Igen	<i>NULL</i>		

5. ábra

Az adatbázisok tábláit az alábbi modellekkel és hozzájuk tartozó repositorykkal (ezekről csak egy képet mellékelünk az egyszerűség kedvéért) reprezentáltuk a forráskódban:

```
public class Ar {  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private int id;  
  
    public void setId(int id) { this.id = id; }  
  
    public void setSutiid(int sutiid) { this.sutiid = sutiid; }  
  
    public void setErtek(int ertek) { this.ertek = ertek; }  
  
    public void setEgyseg(String egyseg) { this.egyseg = egyseg; }  
  
    public void setSuti(Suti suti) { this.suti = suti; }  
  
    public int getId() { return id; }  
  
    public int getSutiid() { return sutiid; }  
  
    public int getErtek() { return ertek; }  
  
    public String getEgyseg() { return egyseg; }  
  
    public Suti getSuti() { return suti; }  
  
    private int sutiid; 2 usages  
    private int ertek; 2 usages  
    private String egyseg; 2 usages  
    // Relationship Mapping  
    @ManyToOne 2 usages  
    @JoinColumn(name = "sutiid", insertable = false, updatable = false)  
    private Suti suti;
```

6. ábra

```

@Entity 3 usages  luczay
@Table(name = "tartalom")
public class Tartalom {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private int sutiid; 2 usages
    private String mentes; 2 usages
    // Relationship Mapping
    @ManyToOne 2 usages
    @JoinColumn(name = "sutiid", insertable = false, updatable = false)
    private Suti suti;

    public int getId() { return id; }

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

    public int getSutiid() { return sutiid; }

    public void setSutiid(int sutiid) { this.sutiid = sutiid; }

    public String getMentes() { return mentes; }

    public void setMentes(String mentes) { this.mentes = mentes; }

    public Suti getSuti() { return suti; }

    public void setSuti(Suti suti) { this.suti = suti; }
    // Getters and Setters
}

```

7. ábra

```
@Entity 7 usages HamNorb
@Table(name="users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    private String name; 2 usages
    private String email; 2 usages
    private String password; 2 usages
    private String role; 2 usages

    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; }
}
```

8. ábra

```
@Entity 13 usages HamNorb
public class Suti {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY) // AUTO_INCREMENT
    private int id;
    private String nev; 2 usages
    private String tipus; 2 usages

    public boolean isDijazott() { return dijazott; }

    public void setDijazott(boolean dijazott) { this.dijazott = dijazott; }

    public String getTipus() { return tipus; }

    public void setTipus(String tipus) { this.tipus = tipus; }

    public String getNev() { return nev; }

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

    public int getId() { return id; }

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

    private boolean dijazott; 2 usages
}
```

9. ábra

```
@Entity 8 usages HamNorb
public class UzenetOsztaly {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
    private String content; 2 usages

    // Getterek és setterek
    public long getId() { return id; }

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

    public String getContent() { return content; }

    public void setContent(String content) { this.content = content; }
}
```

10. ábra


```
public interface UzenetRepository extends JpaRepository<UzenetOsztaly, Long> { 2 usages HamNorb
}
```

11. ábra

## 1. Feladat

Főoldal
Belépés
Regisztrál
Home
Üzenet
Korábbi üzenetek
Sötét

### Köszöntés



**Nagyi mama**

Fiát, Andrew-t többször átnevelő táborba küldte, legutolsónál akkor, amikor az rátámadott George-ra egy anyjával váltott esők után. Nagyi alkoholproblémája akkor vált szorongatóvá, amikor Lynette gyerekeit felügyelve némi bor elfogyasztása után kidőlt, a gyerekek pedig elszöktek. Nagyi nehezen, de rávette magát, hogy elmenjen az Anonim Alkoholisták összejövetelére, ám továbbra is úgy gondolta, hogy nincs problémája az ivással – az összejövetelekre azért ment el, hogy ezzel segítsen megakadályozni fia önállósodását.

12. ábra

### Honnan származik?

Az általános iskolát Csornán végezte. 1977-ben érettségizett a győri Révai Miklós Gimnáziumban. 1977–78-ban Kalocsán mint előfelvételis teljesített sorkatonai szolgálatot.

1978-ban felvették az Eötvös Loránd Tudományegyetem Állam- és Jogtudományi Karára, ahol 1983-ban diplomázott. Egyetemi éve alatt a ELTE Jogász Társadalomtudományi Szakkollégium (jelenlegi nevén Bibó István Szakkollégium) tagja volt. Az egyetem elvégzése után a Budapest VI. kerületi Tanács lakáscsere osztályán dolgozott előadóként. 1985-től 1990-ig az MTA Szociológiai Kutatóintézetében dolgozott tudományos munkatársként. Kutatási területe az Országgyűlés törvényalkotó munkája volt. Emellett ügyvédi szakvizsgát tett.

2007-ben bejelentette, hogy indulni kíván a 2009-es európai parlamenti választáson. Miután a választáson mandátumot szerzett, lemondott országgyűlési képviselői mandátumáról; ezt követően az Európai Parlamentben politizált, 2012 tavaszáig. .

### Miért szeretjük?

Elnöki ciklusa végén derült ki, hogy volt államfőként egy olyan svábhegyi luxusvillába költözik, amit majdnem egy milliárd forint értékben bővítettek ki, ehhez ráadásul egy külön kormányrendeletet is hoztak, hogy a kerület ingatlanépítési szabályait megkerülhessék. Ellenzéki oldalról fel is szólították, hogy ne fogadja el az ingatlant, főleg azután, hogy adóbevallása szerint három ingatlannal is rendelkezik.

### Honnan szerezhető be?

Nagyi híres szakácsnő lesz, és hihetetlen sikert ér el szakácskönyvével. Férjét, Orsont kiengedték a börtönből, miután Nagyi kérte, hogy vallja be, hogy ő gázolta el Mike Delfinót. Eközben a kis Benjamingt Danielle elviszi, mivel egy ügyvéd elvette feleségül, ezentúl tudja ő is nevelni nem várt gyermekét. Ebbe Nagyi teljesen beleszkad, megint inni kezd. Katherine siet a segítségére: hozzáköltözik egy időre, amíg segít barátnőjének letenni az italt. Később együtt dolgoznak Nagyi vállalkozásában mint társak, ám Nagyi Hodge felülkerekedik barátnőjén, aki csak jelentéktelen szereplő a társas vállalkozásban. Orsont sehol sem foglalkoztatják, tekintettel bűnös előéletére. Nagyi ad neki munkát, nem kevesek rosszallására. Andrew megtalálja álmai hercegét, akinek persze megérkezik az édesanyja, aki elcsalná a fiatal párt egy másik városba lakni, ám Nagyi igazi született feleség: házat vásárol nekik a közvetlen közelben, hogy ne veszítse el fiát.

13. ábra

## 2. és 3. Feladat

A szerepek elkülönítését spring security és a user adatbázis segítségével valósítottuk meg. A spring security segített a szerepek elkülönítésében, aminek megvalósítása az 14. és 15. ábrán található. A user táblával pedig a userok tárolását, illetve nem tárolását oldottuk meg. Előbbi esetben a vendég szerep van kiosztva.



```

<div xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity3"
      class="background-div">
  <div>
    <span sec:authorize="isAnonymous()">
      <a th:href="@{/http://localhost:8080/}" class="custom-button">Főoldal</a>
      <a th:href="@{/login}" class="custom-button">Belépés</a>
      <a th:href="@{/regisztral}" class="custom-button">Regisztrál</a>
      <a th:href="@{/home2}" class="custom-button">Home</a>
      <a th:href="@{/feladat}" class="custom-button">Üzenet</a>
      <a th:href="@{/messages}" class="custom-button">Korábbi üzenetek</a>
      <a th:href="@{/sutik}" class="custom-button">Sütik</a>
    </span>

    <span sec:authorize="isAuthenticated()">
      <a th:href="@{/index}" class="custom-button">Főoldal</a>
      <a th:href="@{/home}" class="custom-button">Home</a>
      <a th:href="@{/logout}" class="custom-button">Logout</a>
      <a th:href="@{/feladat}" class="custom-button">Üzenet</a>
      <a th:href="@{/home2}" class="custom-button">Home2</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>

```

14. ábra

```

public class WebSecurityConfig {
    @Autowired
    private UserDetailsService userDetailsService;
    @Bean
    @HamNorb
    public static PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }

    @Bean
    @HamNorb+1
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
        http.csrf(csrf -> csrf.disable())
            .authorizeHttpRequests(auth -> auth
                .requestMatchers("@{/css/**}", "@{/images/**}", "@{/js/**}").permitAll() // Statikus fájlok engedélyezése
                .requestMatchers("@{/}", "@{/regisztral}", "@{/regisztral/feldolgoz}", "@{/home2}", "@{/fooldal}", "@{/feladat}", "@{/feladat2}", "@{/eredmeny}", "@{/messages}")
                .requestMatchers("@{/admin/**}").hasRole("ADMIN") // Admin oldalak
                .anyRequest().authenticated() // Minden máshoz bejelentkezés szükséges
            )
            .formLogin(form -> form
                .defaultSuccessUrl( defaultSuccessUrl: "/home", alwaysUse: true).permitAll()
            )
            .logout(logout -> logout
                .logoutRequestMatcher(new AntPathRequestMatcher("@{/logout}"))
                .logoutSuccessUrl("/")
                .permitAll()
            );
        return http.build();
    }
}

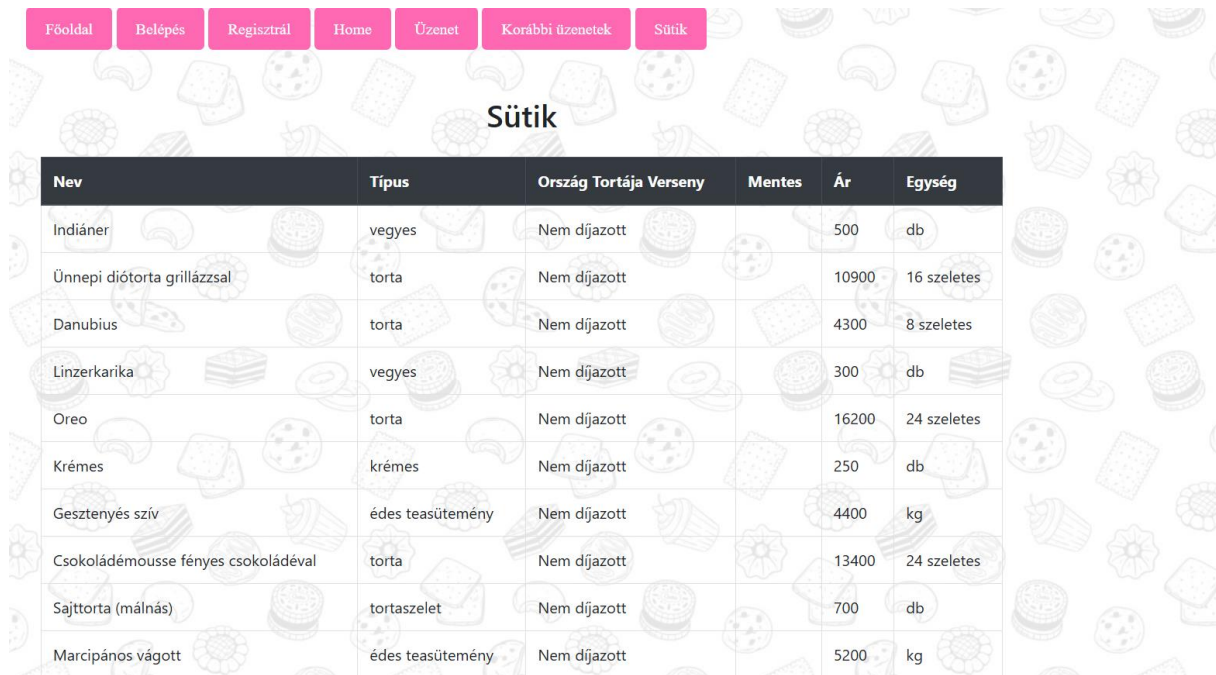
@Bean
@HamNorb
public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception {
    return configuration.getAuthenticationManager();
}

```

15. ábra

## 4. Feladat

Ezt a feladatot egy sütik nevű oldalon valósítottuk meg, ide írtuk ki a 3 tábla adatait összefűzve, ami az 16. ábrán tekinthető meg.



The screenshot shows a web application interface. At the top, there is a navigation bar with buttons: Főoldal, Belépés, Regisztrál, Home, Üzenet, Korábbi üzenetek, and Sütik. Below the navigation bar, the title "Sütek" is displayed. Underneath the title is a table with the following data:

Nev	Típus	Ország Tortája Verseny	Mentes	Ár	Egység
Indiáner	vegyes	Nem díjazott		500	db
Ünnepi diótorta grillázzsal	torta	Nem díjazott		10900	16 szeletes
Danubius	torta	Nem díjazott		4300	8 szeletes
Linzerkarika	vegyes	Nem díjazott		300	db
Oreo	torta	Nem díjazott		16200	24 szeletes
Krémes	krémes	Nem díjazott		250	db
Gesztenyész szív	édes teasütemény	Nem díjazott		4400	kg
Csokoládémousse fényes csokoládéval	torta	Nem díjazott		13400	24 szeletes
Sajttorta (málnás)	tortaszelet	Nem díjazott		700	db
Marcipános vágott	édes teasütemény	Nem díjazott		5200	kg

16. ábra

Szerver oldalon ezt nem egy joint használó sql lekérdezéssel valósítottuk meg, hanem felhasználtuk a java JPA, meg maga a java nyelv nyújtotta előnyöket. Elsőnek lekértük az ár és tartalom adatokat az adatbázisból, majd az árakon végig iteráltunk. Minden egyes iterációnál csináltunk egy SutiTartalomAr objektumot aminek a propety értékeit az Ar objektum segítségével töltöttük fel, kivéve a Mentés propertyt. Annak az értékét a lekért tartalom adatokból szedtük ki. Végül a SutiTartalomAr objektumot hozzáadtuk a sutiTartalomArak listához. Ezt a listát pedig tovább adtuk a thymeleaf html oldalunknak a Model objektum segítségével, amin végig iteráltunk és megjelenítettük az eredményt egy táblában. Ennek a megvalósítása az 17., 18., 19. és 20. ábrákon látható.

```

@GetMapping("/sutik")
public String sutik(Model model) {
    List<Ar> arak = arRepo.findAll();
    List<Tartalom> tartalom = tartalomRepo.findAll();
    List<SutiTartalomAr> sutiTartalomArak = new ArrayList<>();

    for (Ar ar : arak) {
        SutiTartalomAr sutiTartalomAr = new SutiTartalomAr(
            ar.getSuti().getNev(),
            ar.getSuti().getTipus(),
            mentes: "",
            ar.getErtek(),
            ar.getEgyseg(),
            ar.getSuti().isDijazott());

        StringBuilder mentes = new StringBuilder();
        for (Tartalom t : tartalom) {
            if (t.getSuti().getId() == ar.getSuti().getId()) {
                mentes.append(t.getMentes()).append(", ");
            }
        }

        if (mentes.length() > 2) {
            mentes.delete(mentes.length() - 2, mentes.length());
            sutiTartalomAr.setMentes(mentes.toString());
        }

        sutiTartalomArak.add(sutiTartalomAr);
    }

    model.addAttribute("sutiTartalomArak", sutiTartalomArak);
}

```

17. ábra

```

public class SutiTartalomAr { 4 usages ▲ luczay
    private String nev; 3 usages
    private String tipus; 3 usages
    private String mentes; 3 usages
    private int ertek; 3 usages
    private String egyseg; 3 usages
    private boolean dijazott; 3 usages
    private String dijazott_string; 3 usages

    public SutiTartalomAr(String nev, String tipus, String mentes, int ertek, String egyseg, boolean dijazott) { 1 usage ▲ luczay
        this.nev = nev;
        this.tipus = tipus;
        this.mentes = mentes;
        this.ertek = ertek;
        this.egyseg = egyseg;
        this.dijazott = dijazott;

        if (dijazott) {
            this.dijazott_string = "Dijazott";
        } else {
            this.dijazott_string = "Nem dijazott";
        }
    }

    public String getNev() { return nev; }

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

    public String getTipus() { return tipus; }

    public void setTipus(String tipus) { this.tipus = tipus; }

```

18. ábra

```

<body>
<div th:insert="menu"></div>
<div class="container mt-5">
  <h2 class="mb-4 text-center">Süti</h2>
  <table class="table table-bordered table-hover">
    <thead class="thead-dark">
      <tr>
        <th>Név</th>
        <th>Típus</th>
        <th>Ország Tortája Verseny</th>
        <th>Mentes</th>
        <th>Ár</th>
        <th>Egység</th>
      </tr>
    </thead>
    <tbody>
      <tr th:each="item : ${sutiTartalomArak}">
        <td th:text="${item.nev}"></td>
        <td th:text="${item.tipus}"></td>
        <td th:text="${item.dijazott_string}"></td>
        <td th:text="${item.mentes}"></td>
        <td th:text="${item.ertek}"></td>
        <td th:text="${item.egyseg}"></td>
      </tr>
    </tbody>
  </table>
</div>

<!-- Optional: Bootstrap JS (for interactive components, if needed) -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

```

19. ábra

## 5. Feladat

A kapcsolat űrlapot az Üzenet nevű oldalon valósítottuk meg, ez látható a . ábrán. Az űrlapban Id és Message mezők vannak, amiket elküldve a szerver az UrlapController osztály urlapSubmit metódusával kezel le, ami a 20. ábrán látható.

The screenshot shows a web application interface. At the top, there is a horizontal navigation bar with pink buttons labeled 'Főoldal', 'Belépés', 'Regisztrál', 'Home', 'Üzenet', 'Korábbi üzenetek', and 'Süti'. Below the navigation bar, the word 'Form' is displayed in a large, bold, red font. Underneath 'Form', there are two input fields: the first is labeled 'Id:' and contains the value '0'; the second is labeled 'Message:' and is empty. Below these input fields are two buttons: 'Submit' and 'Reset'. The entire form area has a light gray background with a repeating pattern of various pastries, cakes, and cookies.

20. ábra

```

@Controller
public class UrlapController {

    private final UzenetService uzenetService;

    public UrlapController(UzenetService uzenetService) { this.uzenetService = uzenetService; }

    @GetMapping("/feladat")
    public String urlapForm(Model model) {
        model.addAttribute("attr1", new UzenetOsztaly());
        return "urlap";
    }

    @PostMapping("/feladat2")
    public String urlapSubmit(@ModelAttribute UzenetOsztaly uzenetOsztaly, Model model) {
        uzenetService.saveUzenet(uzenetOsztaly);
        model.addAttribute("attr2", uzenetOsztaly);
        return "eredmeny";
    }
}

```

com.example.securityrole.UzenetOsztaly attr2

demo

21. ábra

## 7. Feladat

A restful apit a suti táblára valósítottuk meg. A megfelelő mappingekeket felhasználva a 22. és 23. ábrán látható, hogy a /suti útvonalon érhető el a REST API, és a java JPA könyvtárat használtuk ki a Spring Boot @RestController segítségével a restful tevékenység megvalósításához. Továbbá ehhez a feladathoz a törlés miatt a táblák sql definícióit ki kellett egészíteni ON DELETE CASCADE kóddal a foreign keyeknél, hogy süti törlése esetén törlődjenek az ár és tartalom táblák.

```

@DeleteMapping(Ⓢ"/suti/{id}")  lucay
public void sutiDelete(@PathVariable int id) { sutiRepo.deleteById(id); }
}

@RestController Ⓢ  lucay
public class SutiController {
    @Autowired private SutiRepo sutiRepo;

    @GetMapping(Ⓢ"/suti")  lucay
    public Iterable<Suti> sutiAll() {
        return sutiRepo.findAll();
    }

    @GetMapping(Ⓢ"/suti/{id}")  lucay
    public Optional<Suti> sutiById(@PathVariable int id) {
        return sutiRepo.findById(id);
    }

    @PostMapping(Ⓢ"/suti")  lucay
    public Suti sutiCreate(@RequestBody Suti suti) { return sutiRepo.save(suti); }

    @PutMapping(Ⓢ"/suti/{id}")  lucay
    public Suti sutiUpdate(@RequestBody Suti suti, @PathVari
        return sutiRepo.findById(id)
            .map(s -> {
                s.setNev(suti.getNev());
                s.setTipus(suti.getTipus());
                s.setDijazott(suti.isDijazott());
                return sutiRepo.save(s);
            }).orElseGet(() -> {
                suti.setId(id);
                return sutiRepo.save(suti);
            });
    }

    @DeleteMapping(Ⓢ"/suti/{id}")  lucay

```

22. ábra

```

@DeleteMapping(Ⓢ"/suti/{id}")  lucay
public void sutiDelete(@PathVariable int id) { sutiRepo.deleteById(id); }
}

```

23. ábra