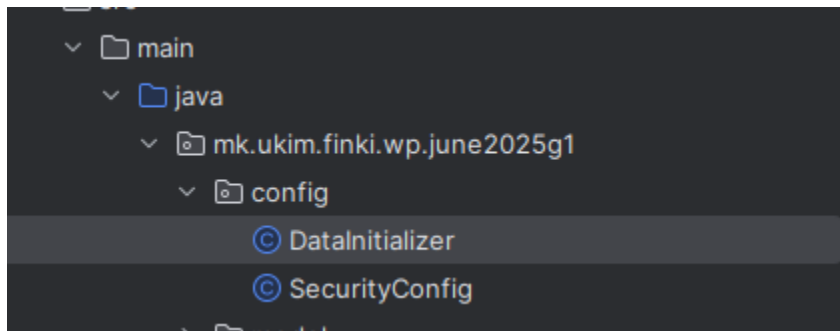


Чекори за решавање задача практично - Веб Програмирање

Чекор 1. Initial	1
Чекор 2: Models	2
Чекор 3: Repository	4
Чекор 4: Services.....	5
Чекор 5: Controller	8
Чекор 6: Thymeleaf	13
Чекор 7: Security	17

Чекор 1. Initial

!!! Препорачливо е да ви биде отворен кодот од аудиториски на еден прозорец од IntelliJ а испитната на друг за копирање код. !!!



Ќе одиме со top-down approach што значи ќе одиме фајл по фајл (освен Security Config тоа оставаме за крај).

```

1 package mk.ukim.finki.wp.june2025g1.config;
2
3 > import ...
4
11
12 @Component
13 public class DataInitializer {
14
15     private final StartupService startupService; 2 usages
16     private final FounderService founderService; 3 usages
17
18     public DataInitializer(StartupService startupService, FounderService founderService) {
19         this.startupService = startupService;
20         this.founderService = founderService;
21     }
22
23
24     private Industry randomizeIndustry(int i) { 1 usage
25         if (i % 4 == 0) return Industry.AI;
26         if (i % 4 == 1) return Industry.BIOTECH;
27         if (i % 4 == 2) return Industry.FINTECH;
28         return Industry.CYBERSECURITY;
29     }
30
31     @PostConstruct
32     public void initData() {
33         for (int i = 1; i <= 3; i++) {
34             this.founderService.create(name: "Founder " + i, email: "founder" + i + "@example.com");
35         }
36
37         List<Founder> founders = this.founderService.listAll();
38
39         for (int i = 1; i <= 10; i++) {
40             String name = "Startup " + i;
41             Double valuation = 500 + (i * 100.0);
42             Integer yearFounded = 2010 + i;
43             Industry period = this.randomizeIndustry(i);
44             Long founderId = founders.get((i - 1) % founders.size()).getId();
45
46             this.startupService.create(name, valuation, yearFounded, period, founderId);
47         }
48     }
49 }

```

Самата класа ја аотираме како @Component, и кај initData функцијата додаваме PostConstruct анотација.

Чекор 2: Models

Продолжуваме со Моделите, ќе го земеме пократкиот модел па ќе отидеме на главниот.

```

1 package mk.ukim.finki.wp.june2025g1.model;
2
3 > import ...
4
5
6
7
8
9
10 @Data
11 @NoArgsConstructor
12 @Entity
13 public class Founder {
14     @Id
15     @GeneratedValue(strategy = GenerationType.IDENTITY)
16     private Long id;
17     private String name;
18     private String email;
19
20     public Founder(String name, String email) { 2 usages
21         this.name = name;
22         this.email = email;
23     }
24 }
25

```

Го аотираме со @Entity, потоа јавниот клуч додаваме дека е @Id и дека истиот е автоматски генериран.

```

1 package mk.ukim.finki.wp.june2025g1.model;
2
3 > import ...
4
5
6
7 @Data
8 @NoArgsConstructor
9 @Entity
10 public class Startup {
11
12     @Id
13     @GeneratedValue(strategy = GenerationType.IDENTITY)
14     private Long id;
15     private String name;
16     private Double valuation;
17     private Integer yearFounded;
18     private boolean active = true;
19     @Enumerated(value = EnumType.STRING)
20     private Industry industry;
21
22     @ManyToOne
23     private Founder founder;
24
25     public Startup(String name, Double valuation, Integer yearFounded, Industry industry, Founder founder) { 2 usages
26         this.name = name;
27         this.valuation = valuation;
28         this.yearFounded = yearFounded;
29         this.industry = industry;
30         this.founder = founder;
31     }
32
33     public Startup(String name, Double valuation, Integer yearFounded, boolean active, Industry industry, Founder founder) { 1 usage
34         this.name = name;
35         this.valuation = valuation;
36         this.yearFounded = yearFounded;
37         this.active = active;
38         this.industry = industry;
39         this.founder = founder;
40     }
41 }
42

```

Главниот модел истото го правиме, само што имаме `@Enumerated(value = EnumType.STRING)` анотација тука value делот ни е битен. Доколку немаме дека вредноста е од тип стринг тогаш ќе ги дава ординално односно со нумеричка вредност 1,2 3 итн. Ова не го сакаме.

```

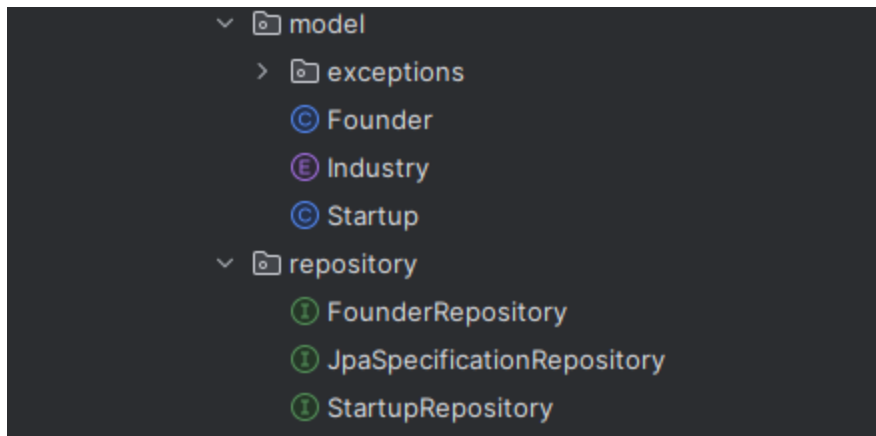
@ManyToOne
private Founder founder;

```

Оваа анотација се гледа според барањата во readme фајлот т.е еден Фоундер може да има повеќе стартапс.

Чекор 3: Repository

Тука предлог мој е одма да си го ископирате фајлот `JpaSpecificationRepository` од аудиториската вежба и да го поставите во `Repository` пакетот.



Откако ќе го направите тоа, започнете со repository од помалиот модел, во овој случај тоа е FounderRepository.

```
1 package mk.ukim.finki.wp.june2025g1.repository;
2
3 import mk.ukim.finki.wp.june2025g1.model.Founder;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface FounderRepository extends JpaRepository<Founder, Long> { 2 usages
7 }
8
```

Тука правиме Јра анотација на класата, се очекува моделот како прв аргумент, а типот на клуч како втор.

```
1 package mk.ukim.finki.wp.june2025g1.repository;
2
3 import mk.ukim.finki.wp.june2025g1.model.Startup;
4
5 public interface StartupRepository extends JpaRepository<Startup, Long> { 2 usages
6 }
7
```

Кај главниот модел тука правиме **наследување од фајлот што го ископиравме!** и истото се очекува, но тука ги имаме сите методи кои ни се потребни за да направиме пагинација и пребарување.

Чекор 4: Services

Тука исто моја препорака е да го ископирате фајлот `FieldFilterSpecification` (services package) и да го поставите во `services` пакетот.

Почнувате со `FounderServiceImpl` дека наследува од интерфејсот што е даден.

```
1 package mk.ukim.finki.wp.june2025g1.service.impl;
2
3 import mk.ukim.finki.wp.june2025g1.model.Founder;
4 import mk.ukim.finki.wp.june2025g1.service.FounderService;
5
6 import java.util.List;
7
8 public class FounderServiceImpl implements FounderService { no usages
9     @Override 2 usages
10     public Founder findById(Long id) {
11         return null;
12     }
13
14     @Override 8 usages
15     public List<Founder> listAll() {
16         return List.of();
17     }
18
19     @Override
20     public Founder create(String name, String email) {
21         return null;
22     }
23 }
24
```

Тука `СЕКОГАШ` се анотира со

```
@Service
@AllArgsConstructor
```

Анотација на самата класа и се прави `Dependency Injection` (не знаеш што е `Dependency Injection`? Го паѓаш предметот автоматски).

```

1 package mk.ukim.finki.wp.june2025g1.service.impl;
2
3 import lombok.AllArgsConstructor;
4 import mk.ukim.finki.wp.june2025g1.model.Founder;
5 import mk.ukim.finki.wp.june2025g1.model.exceptions.InvalidFounderIdException;
6 import mk.ukim.finki.wp.june2025g1.repository.FounderRepository;
7 import mk.ukim.finki.wp.june2025g1.service.FounderService;
8 import org.springframework.stereotype.Service;
9
10 import java.util.List;
11
12 @Service
13 @AllArgsConstructor
14 public class FounderServiceImpl implements FounderService {
15     private final FounderRepository founderRepository;
16     @Override 2 usages
17     public Founder findById(Long id) {
18         return founderRepository.findById(id).orElseThrow(InvalidFounderIdException::new);
19     }
20
21     @Override 8 usages
22     public List<Founder> listAll() {
23         return founderRepository.findAll();
24     }
25
26     @Override
27     public Founder create(String name, String email) {
28         return founderRepository.save(new Founder(name, email));
29     }
30 }
31

```

Откако ќе ги направиш останатите функции и DI одиш на другиот сервис.

TIPS & TRICKS

Секогаш на главната класа треба да имаме DI од самиот repository на сервисот како и помошниот сервис од другата класа.


```

20 @Service
21 @AllArgsConstructor
22 public class StartupServiceImpl implements StartupService {
23     private final StartupRepository startupRepository;
24     private final FounderService founderService;
25     @Override 15 usages
26     public List<Startup> listAll() {
27         return startupRepository.findAll();
28     }

```

Моја препорака е да се реши цел service одеднаш се со цел да не се навраќаме повторно и во интерес на време.

CRUD функционалностите треба да ги знаете и на сон па така откако ќе ги искуцате одите на последните 2 функции кои треба да ги направите (филтрирање и некоја додатна функционалност)

```

60
61 @Override 1 usage
62 public Startup deactivate(Long id) {
63     Startup s = findById(id);
64     s.setActive(false);
65     startupRepository.save(s);
66     return s;
67 }
68
69 @Override 17 usages
70 public Page<Startup> findPage(String name, Double valuation, Integer yearFounded, Industry industry, Long founderId, int pageNum, int pageSize) {
71     Specification<Startup> specification = Specification.allOf(
72         filterContainsText(Startup.class, field("name", name),
73             greaterThan(Startup.class, field("valuation", valuation),
74                 greaterThan(Startup.class, field("yearFounded", yearFounded),
75                     filterEqualsV(Startup.class, field("industry", industry),
76                         filterEqualsV(Startup.class, field("founder.id", founderId)
77                     ));
78     );
79     return startupRepository.findAll(specification, PageRequest.of(pageNum, pageSize));
80 }
81

```

Сите овие методи filterContainsText, greaterThan тие се од FieldFilterSpecification фајлот кој го ископиравте.

Чекор 5: Controller

Тука исто препорака од мене е да се реши цел одеднаш па потоа да се оди на thymeleaf.

Како и во сите други класи што правевме прво нешто и тука е анотацијата и DI.

```
10
11 @Controller
12 @AllArgsConstructor
13 @RequestMapping({"/", "/startups"})
14 public class StartupController {
15     private final StartupService startupService;
16     private final FounderService founderService;
17 }
```

Откако ќе го направите ова моја следна препорака е да си ги анотирате рутите, односно да си ставите GetMapping или PostMapping+ патека на рутата. Секаде каде што враќате некој тип на view тука е GetMapping + патеката. Каде што има праќаме податоци е PostMapping.

```
@GetMapping
public String listAll(@RequestParam(required = false) String name,
                     @RequestParam(required = false) Double valuation,
                     @RequestParam(required = false) Integer yearFounded,
                     @RequestParam(required = false) Industry industry,
                     @RequestParam(required = false) Long founderId,
                     @RequestParam(defaultValue = "1") int pageNum,
                     @RequestParam(defaultValue = "10") int pageSize,
                     Model model)
{
    model.addAttribute("startups", startupService.findPage(name, valuation, yearFounded,
    industry, founderId, pageNum - 1, pageSize));
    model.addAttribute("founders", founderService.listAll());
    model.addAttribute("industries", Industry.values());
    return "list";
}
```

Тука да речеме на главниот фајл каде што ќе ги листаме сите стартапи не додаваме патека бидејќи на горната слика имаме две рути по default "/" и "/startups".

Во коментарот што го гледаме во фајлот ни го кажува ова.

```
/**
 * This method should use the "list.html" template to display all startups.
 * The method should be mapped on paths '/' and '/startups'.
 * The arguments that this method takes are optional and can be 'null'.
 * The filtered startups that are the result of the call
 * findPage method from the StartupService should be displayed.
 * If you want to return a paginated result, you should also pass the page number and the page size as arguments.
 *
 * @param name      Filters startups whose names contain the specified text.
 * @param valuation  Filters startups with a valuation greater than the specified value.
 * @param yearFounded Filters startups founded after the specified year.
 * @param industry   Filters startups by the specified industry.
 * @param founderId  Filters startups by the specified founder ID.
 * @param pageNum    The page number.
 * @param pageSize   The number of items per page.
 * @return The view "list.html"
 */
```

Тука секогаш ни е потребно да ги вратиме сите стартапи, сите основачи и индустриите кои ги имаме (тип енум).

```
35
36 @GetMapping
37 public String listAll(@RequestParam(required = false) String name,
38                      @RequestParam(required = false) Double valuation,
39                      @RequestParam(required = false) Integer yearFounded,
40                      @RequestParam(required = false) Industry industry,
41                      @RequestParam(required = false) Long founderId,
42                      @RequestParam(defaultValue = "1") int pageNum,
43                      @RequestParam(defaultValue = "10") int pageSize,
44                      Model model)
45 {
46     model.addAttribute("startups", startupService.findPage(name, valuation, yearFounded, industry, founderId, pageNum, pageSize));
47     model.addAttribute("founders", founderService.listAll());
48     model.addAttribute("industries", Industry.values());
49     return "list";
50 }
51
```

Па така истото го правиме тука. Исто многу битно во методот каде што праќаме одредени податоци од контролер во view мора да имаме и Model во аргументот на функцијата.

```

52  /**
53   * This method should display the "form.html" template.
54   * The method should be mapped on path '/startups/add'.
55   *
56   * @return The view "form.html".
57   */
58   @GetMapping("/add")
59   public String showAdd(Model model) {
60       model.addAttribute("founders", founderService.listAll());
61       model.addAttribute("industries", Industry.values());
62       return "form";
63   }
64

```

Тука ги додаваме основачите и индустриите се со цел да ги излистаме во формата.

```

65  /**
66   * This method should display the "form.html" template.
67   * However, in this case, all 'input' elements should be filled with the appropriate value for the startup that is updated.
68   * The method should be mapped on path '/startups/edit/{id}'.
69   *
70   * @return The view "form.html".
71   */
72
73   @GetMapping("/edit/{id}")
74   public String showEdit(@PathVariable Long id, Model model) {
75       model.addAttribute("founders", founderService.listAll());
76       model.addAttribute("industries", Industry.values());
77       model.addAttribute("startup", startupService.findById(id));
78       return "form";
79   }

```

Истото го правиме и во едит само што овдека додаваме @PathVariable анотација за ид полето и го праќаме самиот стартап во формата.

```

81  /**
82   * This method should create a new startup given the arguments it takes.
83   * The method should be mapped on path '/startups'.
84   * After the startup is created, all startups should be displayed.
85   *
86   * @param name      The name of the startup
87   * @param valuation  The valuation of the startup
88   * @param yearFounded The year the startup was founded
89   * @param industry   The industry of the startup
90   * @param founderId  The id of the founder of the startup
91   * @return Redirects to the list of startups
92   */
93  @PostMapping
94  public String create(@RequestParam(required = false) String name,
95                      @RequestParam(required = false) Double valuation,
96                      @RequestParam(required = false) Integer yearFounded,
97                      @RequestParam(required = false) String industry,
98                      @RequestParam(required = false) Long founderId) {
99      startupService.create(name, valuation, yearFounded, industry, founderId);
100     return "redirect:/startups";
101 }
102

```

Пост барањата се едно од полесните тука нема некоја философија.

```

103  /**
104   * This method should update an existing startup given the arguments it takes.
105   * The method should be mapped on path '/startups/[id]'.
106   * After the startup is updated, all startups should be displayed.
107   *
108   * @param id      The id of the startup that is being updated
109   * @param name     The name of the startup
110   * @param valuation The valuation of the startup
111   * @param yearFounded The year the startup was founded
112   * @param industry The industry of the startup
113   * @param founderId The id of the founder of the startup
114   * @return Redirects to the list of startups
115   */
116  @PostMapping("/{id}")
117  public String update(@PathVariable Long id,
118                      @RequestParam(required = false) String name,
119                      @RequestParam(required = false) Double valuation,
120                      @RequestParam(required = false) Integer yearFounded,
121                      @RequestParam(required = false) Industry industry,
122                      @RequestParam(required = false) Long founderId) {
123      startupService.update(id, name, valuation, yearFounded, industry, founderId);
124      return "redirect:/startups";
125  }

```

```

127  /**
128   * This method should delete the startup that has the appropriate identifier.
129   * The method should be mapped on path '/startups/delete/[id]'.
130   * After the startup is deleted, all startups should be displayed.
131   *
132   * @param id The ID of the startup to delete
133   * @return Redirects to the list of startups
134   */
135  @PostMapping("/delete/{id}")
136  public String delete(@PathVariable Long id) {
137      startupService.delete(id);
138      return "redirect:/startups";
139  }
140
141  /**
142   * This method should deactivate the startup that has the appropriate identifier.
143   * The method should be mapped on path '/startups/deactivate/[id]'.
144   * After the selected startups is closed, all startups should be displayed.
145   *
146   * @param id The ID of the startup to deactivate
147   * @return Redirects to the list of startups
148   */
149  @PostMapping("/deactivate/{id}")
150  public String deactivate(@PathVariable Long id) {
151      startupService.deactivate(id);
152      return "redirect:/startups";
153  }
154  }

```


Чекор 6: Thymeleaf

Почнувате со list.html потоа се префрлате на form.html ----

Тука идете во ауд кодот и го зимате овој дел доколку не е даден на испит

```
<div xmlns:th="http://www.thymeleaf.org"
xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
```

Без ова нема да имате thymeleaf сугестии во кодот и може да ви отежни.

```
1 <!DOCTYPE html>
2 <html lang="en" xmlns:th="http://www.thymeleaf.org" xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
3 <head>
4     <title>Startup List</title>
5 </head>
6 <body>
7     
8 </div>
```

Тука прв чекор најбитен е да започнете секаде каде што наме полето е празно да додадете. Многу честа грешка е да ви биде погрешно името што сте го напишале па да не знаете каде ви паѓа апликацијата. Имињата треба да бидат исти како што се во FindPage делот во сервисот.

```
68
69 @Override 17 usages
70 public Page<Startup> findPage(String name, Double valuation, Integer yearFounded, Industry industry, Long founderId, int pageNum, int pageSize) {
71     Specification<Startup> specification = Specification.allOf(
72         filterContainsText(Startup.class, field("name", name),
73         greaterThan(Startup.class, field("valuation", valuation),
74         greaterThan(Startup.class, field("yearFounded", yearFounded),
75         filterEqualsV(Startup.class, field("industry", industry),
76         filterEqualsV(Startup.class, field("founder.id", founderId)
77     );
78     return startupRepository.findAll(specification, PageRequest.of(pageNum, pageSize));
79 }
80
81
```

Треба да бидат исти со последниот дел од аргументот, значи не founder.id туку founderId.

Следен чекор е да ги анотирате во формата за филтрирање индустриите и основачите.

```

20 <label for="industry">Industry</label>
21 <select id="industry" name="industry">
22   <option value="">All</option>
23   <!-- For each Industry you should have one <option> like below -->
24   <option th:each="i : ${industries}" th:text="${i.name()}" th:value="${i.name()}">[industry]</option>
25 </select>
26
27 <label for="founderId">Founder</label>
28 <select id="founderId" name="founderId">
29   <option value="">All</option>
30   <!-- For each founder you should have one <option> like below -->
31   <option th:each="f : ${founders}" th:text="${f.name}" th:value="${f.id}">[founder.name]</option>
32 </select>

```

Следно си ги анотирате стартапите

```

52 <!-- For each startup you should have one <tr> like below -->
53 <tr class="item" th:each="s : ${startups}">
54   <td th:text="${s.name}">[startup.name]</td>
55   <td th:text="${s.valuation}">[startup.valuation]</td>
56   <td th:text="${s.yearFounded}">[startup.yearFounded]</td>
57   <td th:text="${s.industry}">[startup.industry]</td>
58   <td th:text="${s.founder.name}">[startup.founder.name]</td>
59   <td th:text="${s.active ? 'ACTIVE' : 'DEACTIVATED'}">[startup.active] -> ACTIVE/DEACTIVATED</td>
60 </tr>

```

Кај додатните копчиња што се формите секогаш се со th:action а каде што имаме <a> таг тука е th:href, ова е исто честа грешка и после ќе се прашувате кај ви паѓа кодот.

```

60 </td>
61 <!-- StartupController.edit -->
62 <a class="edit-item" th:href="@{/startups/edit/{id}} (id=${s.getId()})" >Edit</a>
63
64 <!-- StartupController.delete -->
65 <form th:action="@{/startups/delete/{id}} (id=${s.getId()})" method="post">
66   <button type="submit" class="delete-item">Delete</button>
67 </form>
68
69 <!-- StartupController.close -->
70 <form th:action="@{/startups/deactivate/{id}} (id=${s.getId()})" method="post" th:if="${s.isActive()}">
71   <button type="submit" class="close-item">Deactivate</button>
72 </form>
73
74 </td>

```

Потоа се префлрате на form.html

Тука истото го правите со празните полиња со имиња прво го пополнувате па одите на листање на основачи и индустрии.


```

40 <label for="industry">Industry</label><br/>
41 <select id="industry" name="industry" size="5" required>
42   <!-- For each Industry you should have one <option> like below -->
43   <option th:each="i : ${industries}" th:text="${i.name()}" th:value="${i.name()}" th:selected="${startup != null && startup.industry == i}">[industry]</option>
44 </select>
45 </div>
46
47 <div>
48   <label for="founderId">Founder</label><br/>
49   <select id="founderId" name="founderId" required size="5">
50     <!-- For each founder you should have one <option> like below -->
51     <option th:each="f : ${founders}" th:text="${f.name}" th:value="${f.id}" th:selected="${startup != null && startup.founder == f}">[founder.founder]</option>
52   </select>
53 </div>

```

Откако ќе го направите ова идете на формата скроз горе и ја анотирате

```

<form method="post" th:action="@${startup == null ? '/startups' : '/startups/' +
startup.id}"}>

```

Бидејќи користиме еден фајл за креирање и едитирање проверуваме дали стартап постои (го праќаме во едит функцијата во контролерот). Доколку постои методот ќе биде за едитирање, доколку не е пратен ќе биде за креирање. Истово ова го означуваме и со th:value полињата значи доколку постои нека ја има таа вредност инаку остави празно.

```

<div>
  <label for="name">Startup Name</label>
  <input type="text"
    id="name"
    name="name"
    th:value="${startup?.name}"
    required>
</div>

```

```

<div>
  <label for="valuation">Valuation (in millions)</label>
  <input type="number"
    id="valuation"
    name="valuation"
    th:value="${startup?.valuation}"
    required>
</div>

```

И така натаму.

Со ова имате 80 поени. Останува уште делот за security

Чекор 7: Security

```
34
35 @Configuration
36 @EnableWebSecurity
37 @EnableMethodSecurity
38 public class SecurityConfig {
39
40     /**
41      * Remove the implementation of the following method and replace it with your own code to implement the security requests.
42      * If you do not wish to implement the security requests, leave this code as it is.
43      */
44     @Bean
45     public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
46         http
47             .csrf(AbstractHttpConfigurer::disable)
48             .headers(HeadersConfigurer<HttpSecurity> httpSecurityHeadersConfigurer ->
49                 httpSecurityHeadersConfigurer.frameOptions(HeadersConfigurer.FrameOptionsConfig::disable)
50             )
51             .authorizeHttpRequests((AuthorizationManagerRequestMat... requests) -> requests
52                 .requestMatchers(@"/**")
53                 .permitAll()
54             );
55         return http.build();
56     }
57 }
58 |
```

Вака ви изгледа фајлот. Прво нешто што правите е анотација горе ставате `@AllArgsConstructor` па ќе го закоментирате цел Bean

```

36  @Configuration
37  @EnableWebSecurity
38  @EnableMethodSecurity
39  @AllArgsConstructor
40  public class SecurityConfig {
41
42      /**
43       * Remove the implementation of the following method and replace it with your own code to implement the security requests.
44       * If you do not wish to implement the security requests, leave this code as it is.
45       */
46      /** @Bean
47       public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
48          http
49              .csrf(AbstractHttpConfigurer::disable)
50              .headers(httpSecurityHeadersConfigurer ->
51                  httpSecurityHeadersConfigurer.frameOptions(HeadersConfigurer.FrameOptionsConfig::disable)
52              )
53              .authorizeHttpRequests((requests) -> requests
54                  .requestMatchers("/**")
55                  .permitAll()
56              );
57          return http.build();
58      }*/
59  }

```

Идете на фајлот од аудиториската наречен WebSecurityConfig.java се наоѓа во config пакетот и ги зимате првите два бинови

```

59      .permitAll()
60  );
61  return http.build();
62  }*/
63  @Bean
64  public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
65
66      http
67          .csrf(AbstractHttpConfigurer::disable)
68          .headers(( HeadersConfigurer<HttpSecurity> headers) -> headers
69              .frameOptions(HeadersConfigurer.FrameOptionsConfig::sameOrigin)
70          )
71          .authorizeHttpRequests(( AuthorizationManagerRequestMat... requests) -> requests
72              .requestMatchers(@"/", @"/home", @"/assets/**", @"/register") AuthorizedUrl
73              .permitAll() AuthorizationManagerRequestMat...
74              .requestMatchers(@"/admin/**").hasRole("ADMIN")
75              .anyRequest() AuthorizedUrl
76              .authenticated()
77          )
78          .formLogin(( FormLoginConfigurer<HttpSecurity> form) -> form
79              .loginPage(@"/login")
80              .permitAll()
81              .failureUrl( authenticationFailureUrl: "/login?error=BadCredentials")
82              .defaultSuccessUrl( defaultSuccessUrl: "/products", alwaysUse: true)
83          )
84          .logout(( LogoutConfigurer<HttpSecurity> logout) -> logout
85              .logoutUrl(@"/logout")
86              .clearAuthentication(true)
87              .invalidateHttpSession(true)
88              .deleteCookies( ...cookieNamesToClear: "JSESSIONID")
89              .logoutSuccessUrl(@"/login")
90          )
91          .exceptionHandling(( ExceptionHandlingConfigurer<HttpSecurity> ex) -> ex

```

Тука во requestMatchers ги бришеме последните две рути, нас ни треба само / и /startups.

```

@Bean
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

    http

        .csrf(AbstractHttpConfigurer::disable)
        .headers(( HeadersConfigurer<HttpSecurity> headers) -> headers
            .frameOptions(HeadersConfigurer.FrameOptionsConfig::sameOrigin)
        )
        .authorizeHttpRequests(( AuthorizationManagerRequestMat... requests) -> requests
            .requestMatchers(Ⓢ"/", Ⓢ"/startups") AuthorizedUrl
            .permitAll() AuthorizationManagerRequestMat...
            .anyRequest().hasRole("ADMIN")
        )
        .formLogin(( FormLoginConfigurer<HttpSecurity> form) -> form
            .loginPage("/login")
            .permitAll()*/
            .failureUrl( authenticationFailureUrl: "/login?error=BadCredentials")
            .defaultSuccessUrl( defaultSuccessUrl: "/startups", alwaysUse: true)
        )
        .logout(( LogoutConfigurer<HttpSecurity> logout) -> logout
            .logoutUrl(Ⓢ"/logout")
            .clearAuthentication(true)
            .invalidateHttpSession(true)
            .deleteCookies( ...cookieNamesToClear: "JSESSIONID")
            .logoutSuccessUrl(Ⓢ"/")
        )

        .exceptionHandling(( ExceptionHandlingConfigurer<HttpSecurity> ex) -> ex
            .accessDeniedPage( accessDeniedUrl: "/access_denied")
        );

    return http.build();
}

```

Вториот бин го тргаме коментарот на анотацијата

Ставаме username & password на првите два корисника како и нивниот соодветен ROLE, user 3 и user 4 не ни требаат па слободно може да ги избришеме.

```

95
96 // In Memory Authentication
97 @Bean
98 public UserDetailsService userDetailsService() {
99     UserDetails user1 = User.builder()
100         .username("user")
101         .password(passwordEncoder.encode("user"))
102         .roles("USER")
103         .build();
104     UserDetails user2 = User.builder()
105         .username("admin")
106         .password(passwordEncoder.encode("admin"))
107         .roles("ADMIN")
108         .build();
109
110     return new InMemoryUserDetailsManager(user1, user2);
111 }
112 }
113 |

```

Доколку ни свети црвено passwordEncoder тоа значи не сме го додале го додаваме скроз горе го копираме од истиот фајл од каде што ги земавме двете функции. И дополнително треба да го аотираме во Main фајлот.

```

1 package mk.ukim.finki.wp.june2025g1;
2
3 > import ...
4
5
6
7
8
9 @SpringBootApplication
10 public class June2025G1Application {
11
12 >     public static void main(String[] args) { SpringApplication.run(June2025G1Application.class, args); }
13
14
15
16 |
17 @Bean
18 > PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(strength: 10); }
19
20
21 }
22

```

Откако ќе го направиме ова се враќаме во list.html и додаваме дека рутата е само за корисници со ROLE што е еднаков на админ.

```
<td th:text= "${s.active ? 'ACTIVE' : 'DEACTIVATED'}" >{s.startup.active} </td> </tr> </tbody> </table> </div> <div> <!-- StartupController.edit --> <a class="edit-item" th:href="@{/startups/edit/{id}}" (id=${s.getId()})" sec:authorize="hasRole('ADMIN')">Edit</a> <!-- StartupController.delete --> <form th:action="@{/startups/delete/{id}}" (id=${s.getId()})" method="post" sec:authorize="hasRole('ADMIN')"> <button type="submit" class="delete-item">Delete</button> </form> <!-- StartupController.close --> <form th:action="@{/startups/deactivate/{id}}" (id=${s.getId()})" method="post" th:if="${s.isActive()}" sec:authorize="hasRole('ADMIN')"> <button type="submit" class="close-item">Deactivate</button> </form> </div> <div> <!-- StartupController.showAdd --> <a class="add-item" href="@{add}" sec:authorize="hasRole('ADMIN')">Add new startup</a> </div>
```

Идеме во SeleniumScenarioTest го ставаме нашиот индекс и го пуштаме тестот.

```
✓ SeleniumScenarioTest (mk.ukim.finki. 5 sec 66 ms)
  ✓ test_list_10pt() 1 sec 594 ms
  ✓ test_pagination_10pt() 53 ms
  ✓ test_filter_5pt() 869 ms
  ✓ test_filter_service_5pt() 60 ms
  ✓ test_create_10pt() 705 ms
  ✓ test_create_mvc_10pt() 49 ms
  ✓ test_edit_10pt() 337 ms
  ✓ test_edit_mvc_10pt() 28 ms
  ✓ test_delete_3pt() 302 ms
  ✓ test_delete_mvc_2pt() 20 ms
  ✓ test_security_urls_10pt() 386 ms
  ✓ test_security_buttons_10pt() 365 ms
  ✓ test_additional_functionality_3pt() 281 ms
  ✓ test_additional_functionality_mvc_2pt() 17 ms
  ! Class Configuration
```

Цел код искуцан е во рок од 35 минути и нема потреба од воопшто да го гледате на browser, но за почеток препорачливо е да гледате и доколку ви паѓа да гледате во DevTools што се праќа како и грешката што се јавува во терминал.