



TO DO APPLICATION TECHCAREER.NET

FULL-STACK DEVELOPER BOOTCAMP

İÇERİK

01

GENEL BAKIŞ

02

ENTITY

03

REPOSITORY

04

SERVICE

05

API

06

FRONTEND

07

KAPANIŞ

GENEL BAKIŞ

Bu proje bootcamp kapsamında eğitimini aldığım Spring Boot ve React teknolojileri kullanılarak geliştirilmiştir.

Spring Boot ile öncelikle projenin altyapısı oluşturulmuştur. Ardından projenin diğer servislerle iletişime geçebilmesi için API yazılmıştır.

SPRING BOOT

Önyüz tarafında ise React ile projenin komponentleri oluşturulmuştur.

REACT

Son olarak CSS kullanılarak projede istenen tasarım elde edilmeye çalışılmıştır.

CSS

ENTITY

J TodoItem.java ×

src > main > java > com > kayainc > springboottodoapplication > models > J TodoItem.java > TodoItem > description

```
13  @Entity
14  @Table(name="todo_item")
15  public class TodoItem {
16
17      @Id
18      @GeneratedValue(strategy = GenerationType.AUTO)
19      @Getter
20      @Setter
21      private Long id;
22
23      @Getter
24      @Setter
25      @NotBlank(message = "Description is required")
26      private String description;
27
28      @Getter
29      @Setter
30      private boolean complete;
31
32      @Getter
33      @Setter
34      private Instant createdAt;
35
36      @Getter
37      @Setter
38      private Instant modifiedDate;
39
40      public TodoItem() {}
41
42      public TodoItem(String description) {
43          this.description = description;
44          this.complete = false;
45          this.createdAt = Instant.now();
46          this.modifiedDate = Instant.now();
47      }
```

REPOSITORY

CrudRepository bir Spring Data interface'idir. Bu interface implemente edildiğinde en basit haliyle Crud işlemleri elde edilir.

J TodoItemRepository.java ✕

src > main > java > com > kayainc > springboottodoapplication > repositories > J TodoItemRepository.java > Language :

```
1  package com.kayainc.springboottodoapplication.repositories;
2
3  import org.springframework.data.repository.CrudRepository;
4
5  import com.kayainc.springboottodoapplication.models.TODOItem;
6
7  public interface TODOItemRepository extends CrudRepository<TODOItem, Long> {
8      // <TODOItem, Long> -> Long = TODOItem>Id->DataType
9  }
10
```


SERVICE

J IEntityService.java X J TotoItemManager.java 6

src > main > java > com > kayainc > springboottodoapplication > business > abstracts > J IEntityService.java > ...

```
1 package com.kayainc.springboottodoapplication.business.abstracts;
2
3 import java.util.List;
4 import java.util.ArrayList;
5
6 public interface IEntityService<Entity> {
7
8     public List<Entity> getAll();
9     public List<Entity> getAllDone();
10    public List<Entity> getAllTodo();
11    public Entity getById(Long id);
12    public Entity add(Entity entity);
13    public Entity update(Long id, Entity entity);
14    public Entity delete(Long id);
15    public ArrayList<Entity> deleteAll(boolean willUncompletedTasksBeDeleted);
16
17 }
```

J IEntityService.java J TotoItemManager.java 6 X

src > main > java > com > kayainc > springboottodoapplication > business > concretes > J TotoItemManager.java >

```
27 // Lombok
28 @RequiredArgsConstructor // Injection
29
30 @Service
31 public class TotoItemManager implements IEntityService<TodoItem> {
32
33     private final TodoItemRepository todoItemRepository;
34     private final ModelMapperBean modelMapperBean;
35
36     @Override
37     public List<TodoItem> getAll() {
38         Iterable<TodoItem> todoItemEntities = todoItemRepository.findAll();
39         List<TodoItem> todoItemArrayList = new ArrayList<>();
40
41         for(TodoItem todoItemEntity : todoItemEntities) {
42             todoItemArrayList.add(todoItemEntity);
43         }
44         return todoItemArrayList;
45     }
46 }
```

API

API ile uygulamamıza dış servislerle iletişim yeteneği kazandırmaktayız.

```
89
90 // http://localhost:8080/api/v1/delete/id
91 @Override
92 @PostMapping(value = "/delete/{id}")
93 public ResponseEntity<?> delete(@PathVariable(name = "id") Long id) {
94     return ResponseEntity.ok(entityService.delete(id));
95 }
96
97 // http://localhost:8080/api/v1/deleteall/willUncompletedTasksBeDeleted
98 @Override
99 @PostMapping(value = "/deleteall/{willUncompletedTasksBeDeleted}")
100 public ResponseEntity<?> deleteAll(@PathVariable(name = "willUncompletedTasksBeDeleted") boolean willUncompletedTasksBeDeleted) {
101     return ResponseEntity.ok(entityService.deleteAll(willUncompletedTasksBeDeleted));
102 }
103 }
```

```
J TodoItemApi.java 9+ X
src > main > java > com > kayainc > springboottodoapplication > controllers > api > J TodoItemApi.java

25 // Lombok
26 @RequiredArgsConstructor // Injection
27 @Log4j2
28
29 // API yazarken yazmalıyız
30 @RestController
31 @RequestMapping("/api/v1")
32 @CrossOrigin(origins = FrontendUrl.REACT_URL)
33
34 public class TodoItemApi implements IEntityApi<TodoItem> {
35
36     // Injection
37     private final IEntityService entityService;
38     private ApiResult apiResult;
39
40     @PostConstruct
41     public void todoItemPostConstruct() {
42         apiResult = new ApiResult();
43     }
44
45     // Swagger-UI
46     // http://localhost:8080/swagger-ui/index.html
47
48     // http://localhost:8080/api/v1/getall
49     @Override
50     @GetMapping(value = "/getAll")
51     public ResponseEntity<List<TodoItem>> getAll() {
52         return ResponseEntity.ok(entityService.getAll());
53     }
54 }
```

FRONTEND

TodoItemRouter.jsx ✕

frontend > src > TodoItemRouter.jsx > ...

```
18
19 class TodoItemRouter extends Component {
20   static displayName = "TodoItem_Router";
21
22   constructor(props) {
23     super(props);
24     this.state = {};
25   }
26
27   // componentDidMount() {}
28
29   render() {
30     return (
31       <div>
32         <Header bars="fa-solid fa-bars" />
33         <div className="container">
34           <Routes>
35             <Route path="/" element={<TodoItemList />} />
36             <Route path="/todoItem/list" element={<TodoItemList />} />
37             <Route path="/todoItem/list/done" element={<TodoItemListDone />} />
38             <Route path="/todoItem/list/todo" element={<TodoItemListTodo />} />
39             <Route path="/todoItem/create" element={<TodoItemCreate />} />
40             <Route path="/todoItem/update/:id" element={<TodoItemUpdate />} />
41             <Route path="/todoItem/detail/:id" element={<TodoItemDetail />} />
42
43             <Route path="*" element={<Navigate to="/" />} />
44           </Routes>
45         </div>
46
47         <Footer copyright="fa-regular fa-copyright" />
48       </div>
49     );
50   }
51 }
```

index.css 3 ✕

frontend > src > # index.css > .create-body .new-todo-button

```
26
27 .create-body {
28   padding: 15px;
29   border: 1px solid #00a5ba;
30   border-radius: 3px;
31   max-width: 700px;
32   margin: 0 auto;
33 }
34
35 .create-body .input-block {
36   position: relative;
37 }
38
39 .create-body .input-block .new-todo-icon {
40   color: white;
41   background-color: #00a5ba;
42   padding: 10px;
43   position: absolute;
44   border-radius: 3px;
45 }
```


TEŞEKKÜRLER

Başta eğitimcimimiz Hamit MIZRAK ve bu eğitimi organize eden TechCareer.net ekibine olmak üzere sunumumu dinleyen herkese çok teşekkür ederim.

Hazırlayan: Ersin Kaya

