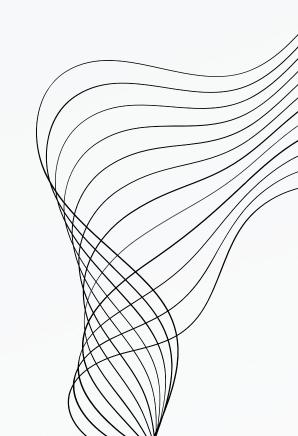


TO DO APPLICATION TECHCAREER.NET

FULL-STACK DEVELOPER BOOTCAMP



icerik

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 Todoltem.java ×
src > main > java > com > kayainc > springboottodoapplication > models > J Todoltem.java > ♦ Todoltem > ♦ description
      @Entity
      @Table(name="todo_item")
      public class TodoItem {
           @Id
           @GeneratedValue(strategy = GenerationType.AUTO)
           @Getter
           @Setter
          private Long id;
           @Getter
           @Setter
 25
           @NotBlank(message = "Description is required")
           private String description;
           @Getter
           @Setter
           private boolean complete;
           @Getter
           @Setter
           private Instant createdDate;
           @Getter
           @Setter
           private Instant modifiedDate;
          public TodoItem() {}
           public TodoItem(String description) {
               this.description = description;
               this.complete = false;
               this.createdDate = Instant.now();
               this.modifiedDate = Instant.now();
```



REPOSITORY

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

J TodoltemRepository.java ×

SERVICE

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

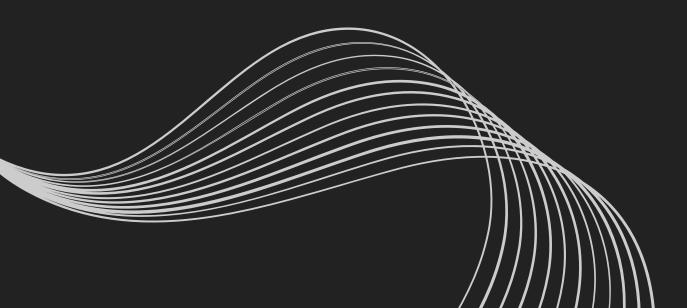
package com.kayainc.springboottodoapplication.business.abstracts;

import java.util.List;
import java.util.ArrayList;

public interface IEntityService<Entity> {

public List<Entity> getAll();
public List<Entity> getAllDone();
public Entity getById(Long id);
public Entity add(Entity entity);
public Entity update(Long id, Entity entity);
public Entity delete(Long id);
public Entity delete(Long id);
public ArrayList<Entity> deleteAll(boolean willUncompletedTasksBeDeleted);

public ArrayList<Entity> deleteAll(boolean willUncompletedTasksBeDeleted);
```



```
J IEntityService.java

                      J TotoltemManager.java 6 X
 src > main > java > com > kayainc > springboottodoapplication > business > concretes > 🔳 TotoItemManager.java >
        // Lombok
        @RequiredArgsConstructor // Injection
        @Service
0
        public class TotoItemManager implements IEntityService<TodoItem> {
            private final TodoItemRepository todoItemRepository;
            private final ModelMapperBean modelMapperBean;
            @Override
            public List<TodoItem> getAll() {
                Iterable<TodoItem> todoItemEntities = todoItemRepository.findAll();
                List<TodoItem> todoItemArrayList = new ArrayList<>();
                for(TodoItem todoItemEntity : todoItemEntities) {
                     todoItemArrayList.add(todoItemEntity);
                return todoItemArrayList;
```



91

93

95

96

100

101

103

API ile uygulamamıza dış servislerle iletişim yeteneği kazandırmaktayız.

```
// http://localhost:8080/api/v1/delete/id
@Override
@Override
@PostMapping(value = "/delete/{id}")
public ResponseEntity<?> delete(@PathVariable(name = "id") Long id) {
    return ResponseEntity.ok(entityService.delete(id));
}

// http://localhost:8080/api/v1/deleteall/willUncompletedTasksBeDeleted
@Override
@PostMapping(value = "/deleteall/{willUncompletedTasksBeDeleted}")
public ResponseEntity
// http://localhost:8080/api/v1/deleteall/willUncompletedTasksBeDeleted
@Override
@PostMapping(value = "/deleteall/{willUncompletedTasksBeDeleted}")
public ResponseEntity<?> deleteAll(@PathVariable(name = "willUncompletedTasksBeDeleted") boolean willUncompletedTasksBeDeleted) {
    return ResponseEntity.ok(entityService.deleteAll(willUncompletedTasksBeDeleted));
}
```

```
J TodoltemApi.java 9+ 

X

src > main > java > com > kayainc > springboottodoapplication > controllers > api > 🤳 TodoItemApi.java
      @RequiredArgsConstructor // Injection
      // API yazarken yazmalıyız
      @RestController
       @RequestMapping("/api/v1")
      @CrossOrigin(origins = FrontendUrl.REACT_URL)
      public class TodoItemApi implements IEntityApi<TodoItem> {
          // Injection
          private final IEntityService entityService;
          private ApiResult apiResult;
          @PostConstruct
          public void todoItemPostConstuct() {
               apiResult = new ApiResult();
          // http://localhost:8080/swagger-ui/index.html
          @GetMapping(value = "/getall")
          public ResponseEntity<List<TodoItem>> getAll() {
               return ResponseEntity.ok(entityService.getAll());
```

FRONTEND


```
frontend > src >  TodoltemRouter.jsx > ...
      class TodoItemRouter extends Component {
        static displayName = "TodoItem_Router";
        constructor(props) {
          super(props);
          this.state = {};
        // componentDidMount() {}
        render() {
          return (
            <div>
              <Header bars="fa-solid fa-bars" />
              <div className="container">
                <Routes>
                  <Route path="/" element={<TodoItemList />} />
                  <Route path="/todoItem/list" element={<TodoItemList />} />
                  <Route path="/todoItem/list/done" element={<TodoItemListDone />} />
                  <Route path="/todoItem/list/todo" element={<TodoItemListTodo />} />
                  <Route path="/todoItem/create" element={<TodoItemCreate />} />
                  <Route path="/todoItem/update/:id" element={<TodoItemUpdate />} />
                  <Route path="/todoItem/detail/:id" element={<TodoItemDetail />} />
                  <Route path="*" element={<Navigate to="/" />} />
                </Routes>
              </div>
              <Footer copyright="fa-regular fa-copyright" />
            </div>
          ):
```

```
# index.css 3 X
frontend > src > # index.css > 4 .create-body .new-todo-button
       .create-body {
        padding: 15px;
        border: 1px solid □rgba(128, 128, 128, 0.25);
        border-radius: 3px;
        max-width: 700px;
        margin 0 auto
       .create-body .input-block {
        position relative
       .create-body .input-block .new-todo-icon {
        color: ■white;
        background-color: ■#00a5ba;
        padding: 10px;
        position absolute
        border-radius: 3px;
```

TEŞEKKÜRLER

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

Hazırlayan: Ersin Kaya

