5.Soru için Doküman

Müşteriler sisteme kayıt olur ve sistemde işlem yapmak için Json Web Token alır(JWT)

Bu tokeni kullanarak sistemde işlemler yapar.Bunun için aşağıdaki endpointler kullanılır.

```java
@RestController
@RequestMapping("/auth")
@RequiredArgsConstructor
public class AuthController {

    private final AuthService authService;
    no usages
    @PostMapping("/singUp")
    public ResponseEntity<?> singUp(@RequestBody SingUpRequest singUpRequest){
        return ResponseEntity.ok(authService.singUp(singUpRequest));
    }
    no usages
    @PostMapping("/login")
    public ResponseEntity<TokenResponseDto> login(@RequestBody LoginRequest loginRequest){
        return ResponseEntity.ok(authService.login(loginRequest));
    }

}
```

Admin için entpointler

Admin sisteme ürün ekler

```java
no usages
@RestController
@RequestMapping("/admin")
@RequiredArgsConstructor
public class AdminController {

    private final AdminService adminService;
    no usages
    @PostMapping("/create")
    public ResponseEntity<ProductDto> createProduct(@RequestBody CreateProductRequest createProductRequest){
        return ResponseEntity.ok(adminService.createProduct(createProductRequest));
    }
```

Admin sistemdeki tüm ürünleri görüntüler

```java
    no usages
    @GetMapping("/getAllProduct")
    public ResponseEntity<List<ProductDto>> getAllProducts(){
        List<ProductDto> products = adminService.getAllProduct();
        return ResponseEntity.ok(products);
    }
```

Admin sistemden bir ürün siler

```
    no usages
30  @DeleteMapping("/delete/{id}")
31  public ResponseEntity<String> deleteProduct(@PathVariable Long id){
32      adminService.deleteProduct(id);
33      return ResponseEntity.ok().body("Product deleted successfully");
34  }
```

Admin sistemde bir ürünü günceller

```
    no usages
35  @PutMapping("/update/{id}")
36  public ResponseEntity<ProductDto> updateProduct(@PathVariable Long id, @RequestBody UpdateProductRequest updateProductRequest){
37      return ResponseEntity.ok(adminService.updateProduct(id, updateProductRequest));
38  }
```

Admin tüm siparişleri görüntüler

```
    no usages
39  @GetMapping("/orders")
40  public ResponseEntity<List<OrderDto>> GetAllOrdersForCustomer(){
41      List<OrderDto> orderDtoList = adminService.GetAllOrdersForCustomer();
42      return ResponseEntity.ok(orderDtoList);
43  }
44
```

Customer için endpointler

Customer tüm ürünleri görüntüler

```
15  @RestController
16  @RequestMapping("/customer")
17  @RequiredArgsConstructor
18  public class CustomerController {
19
20      private final CustomerService customerService;
21
        no usages
22      @GetMapping("/getAllProduct")
23      public ResponseEntity<List<ProductDto>> getAllProducts() {
24          List<ProductDto> products = customerService.getAllProduct();
25          return ResponseEntity.ok(products);
26      }
```

Customer ürünün ismine göre arama yapabilir

```
    no usages
28  @GetMapping("/getProduct/{name}")
29  public ResponseEntity<List<ProductDto>> getProduct(@PathVariable String name) {
30      List<ProductDto> products = customerService.getProduct(name);
31      return ResponseEntity.ok(products);
32  }
33
```

Customer sepete ürün ekleyebilir

```
no usages
@PostMapping("/cart")
public ResponseEntity<?> addProductToCart(@RequestBody AddProductInCardDto addProductInCardDto) {
    return customerService.addProductToCart(addProductInCardDto);
}
```

Customer sepetlerini sorgulayabilir

```
no usages
@GetMapping("/cart/{customerId}")
public ResponseEntity<OrderDto> getCartByCustomerId(@PathVariable Long customerId) {
    OrderDto orderDto = customerService.getCartByCustomerId(customerId);
    if (orderDto == null) {
        return ResponseEntity.notFound().build();
    }
    return ResponseEntity.ok(orderDto);
}
```

Customer sepete ürün ekleyip çıkartabilir

```
no usages
@GetMapping("/{customerId}/decuct/{productId}")
public ResponseEntity<OrderDto> removeProductToCart(@PathVariable Long customerId, @PathVariable Long productId) {
    OrderDto orderDto = customerService.removeProductFromCart(customerId, productId);
    return ResponseEntity.ok(orderDto);
}

no usages
@GetMapping("/{customerId}/add/{productId}")
public ResponseEntity<OrderDto> addProductCart(@PathVariable Long customerId, @PathVariable Long productId) {
    OrderDto orderDto = customerService.addProductCart(customerId, productId);
    return ResponseEntity.ok(orderDto);
}
```

Customer verdiği siparişin sonucunu görebilir

```
@PostMapping("/placeOrder")
public ResponseEntity<OrderDto> placeOrder(@RequestBody PlaceOrderDto placeOrderDto) {
    OrderDto orderDto = customerService.placeOrder(placeOrderDto);
    if (orderDto == null) {
        return ResponseEntity.status(HttpStatus.BAD_REQUEST).build();
    }
    return ResponseEntity.status(HttpStatus.CREATED).body(orderDto);
}
```

Customer Id'si ile geçmiş siparişlerini sorgulayabilir

```
no usages
@GetMapping("/orders/{customerId}")
public ResponseEntity<List<OrderDto>> getOrderForCode(@PathVariable Long customerId) {
    List<OrderDto> orderDtoList = customerService.getOrderForCode(customerId);
    return ResponseEntity.ok(orderDtoList);
}
```