



Autowasch Termine Architekturspezifikation

Lotto Crew

-Hüseyin Tuğşad Şen
-Mert Ali Hanbay

200503047
200503009



ARCHITECTUREÜBERBLICK

Die Architektur besteht aus zwei Hauptkomponenten: dem Client und dem Server.

Der Client ist für die Benutzeroberfläche und die Interaktion mit dem Benutzer verantwortlich. Die App wird als Desktop App für Windows entwickelt. Die Benutzeroberfläche wird mit dem Flutter-Framework und der Programmiersprache Dart implementiert.

Der Server ist für die Verarbeitung von Anfragen, die Datenhaltung und die Geschäftslogik verantwortlich. Der Server wird als RESTful API mit dem Spring-Framework und der Programmiersprache Java entwickelt. Als Datenbank wird MongoDB verwendet.

Die Kommunikation zwischen dem Client und dem Server erfolgt über das HTTP-Protokoll. Der Client sendet Anfragen an den Server und empfängt die Antworten.

Die Architektur erlaubt eine skalierbare und erweiterbare Implementierung, da die Komponenten klar voneinander getrennt sind und über eine definierte Schnittstelle kommunizieren.



Ein Beispiel für eine GET-Method, mit der Benutzer abgerufen werden.

GET




http://localhost:8080/users


Response


```
{
  "id": "645156c19202872e7a410ecf",
  "createDate": "2023-05-02T18:30:25.497+00:00",
  "name": "Kaya Autowasch"
},
{
  "id": "645156c59202872e7a410ed0",
  "createDate": "2023-05-02T18:30:29.462+00:00",
  "name": "Shen Autowasch"
},
{
  "id": "645156cc9202872e7a410ed1",
  "createDate": "2023-05-02T18:30:36.839+00:00",
  "name": "Imamson Autowasch"
}
```

UI Beispiel von GET Method

 Autowasch Termine

Home Autowäscher Panel


 Autowäschern Liste



Kaya Autowash
Kavacık Beykoz/İstanbul

09.00 - 18.00
0555 555 55 55


150₺



Shen Autowash
Çağlayan Kağıthane/İstanbul

07.00 - 20.00
0555 555 55 55

120₺



Han Autowash
Levent Beşiktaş/İstanbul

06.00 - 17.00
0555 555 55 55

250₺



Ein Beispiel für eine GET-Methode in der AutowaschController-Klasse.

In unserer AutowaschController-Klasse gibt es eine GET-Methode für den Endpunkt /users, die mithilfe des userService eine Liste von Benutzern zurückgibt.

```
@RestController
@RequestMapping("/users")
@AllArgsConstructor
public class AutowaschController {

    private final UserService userService;

    @GetMapping
    public ResponseEntity<List<user>> getUsers(@RequestParam(required = false) String name) {
        return new ResponseEntity<>(userService.getUsers(name), OK);
    }
}
```



Ein Beispiel für eine Delete-Methode in der AutowaschController-Klasse.

Die mit dem users/(id) Endpunkt gesendete ID wird mithilfe des userService gelöscht.

```
@DeleteMapping("/{id}")
public ResponseEntity<Void> deleteUser(@PathVariable String id) {
    userService.deleteUser(id);
    return new ResponseEntity<>(OK);
}
```



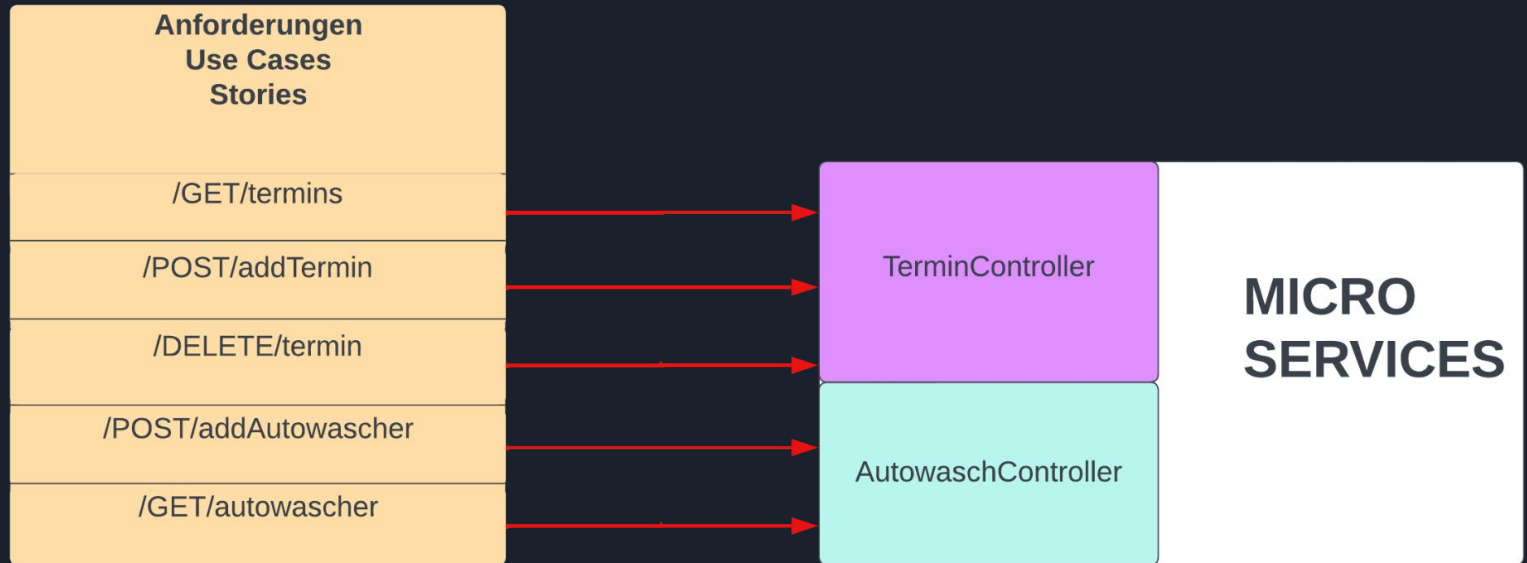
Exception Beispiel

Die von uns geschriebene UserNotFoundException-Klasse erweitert RuntimeException und zeigt bei Auftreten des Fehlers eine Fehlermeldung als String an. In der AutowaschController-Klasse wird bei einem HTTP.status.NOT_FOUND-Statuscode die Fehlermeldung als Antwort zurückgegeben, wenn die Anfrage an den Service gesendet wird.

```
public class UserNotFoundException extends RuntimeException {  
    1 usage  
    public UserNotFoundException(String msg) {  
        super(msg);  
    }  
}
```

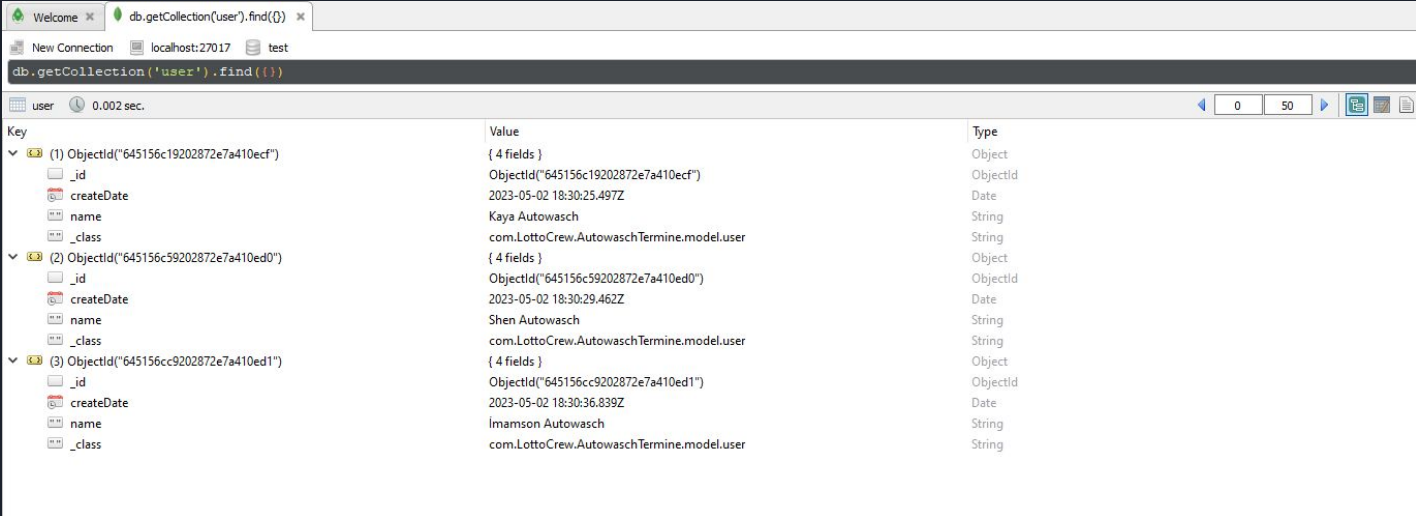
```
@ExceptionHandler(UserNotFoundException.class)  
public ResponseEntity<String> handleUserNotFoundException(UserNotFoundException ex) {  
    return new ResponseEntity<>(ex.getMessage(), NOT_FOUND);  
}
```

Rückverfolgbarkeit der Anforderungen



Beschreibung der DB-Zugriffsschicht

In unserer Datenbank werden die Öffnungszeiten, Namen, Preise, Telefonnummern, Adressen, Passwörter und Bild-IDs der Autowäscher gespeichert. Die Namen, Telefonnummern und Kennzeichen der von den Benutzern erstellten Termine werden ebenfalls in der Datenbank gespeichert.



Key	Value	Type
(1) ObjectId("645156c19202872e7a410ecf")	{ 4 fields }	Object
_id	ObjectId("645156c19202872e7a410ecf")	ObjectId
createDate	2023-05-02 18:30:25.497Z	Date
name	Kaya Autowasch	String
_class	com.LottoCrew.AutowaschTermine.model.user	String
(2) ObjectId("645156c59202872e7a410ed0")	{ 4 fields }	Object
_id	ObjectId("645156c59202872e7a410ed0")	ObjectId
createDate	2023-05-02 18:30:29.462Z	Date
name	Shen Autowasch	String
_class	com.LottoCrew.AutowaschTermine.model.user	String
(3) ObjectId("645156cc9202872e7a410ed1")	{ 4 fields }	Object
_id	ObjectId("645156cc9202872e7a410ed1")	ObjectId
createDate	2023-05-02 18:30:36.839Z	Date
name	Imamson Autowasch	String
_class	com.LottoCrew.AutowaschTermine.model.user	String

(Beispiel Bild von MongoDB)