Milestone 5 - Präsentation

Parkplatzverwaltung

Ahmet Efe Kutbay - 200503019
Altay Yavuz Elmas - 200503046

Vorstellung des Projekts

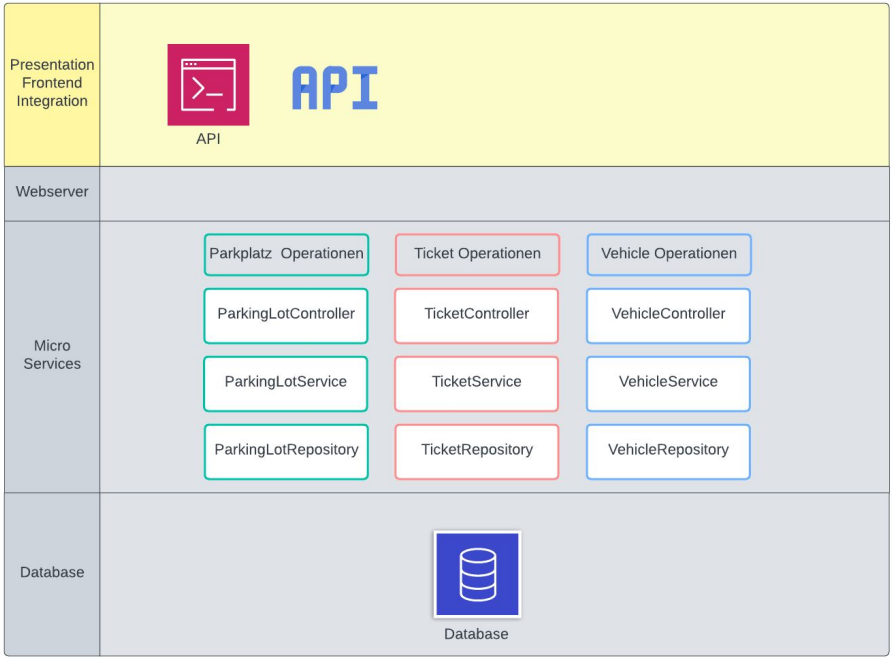
- Parkplatzverwaltungssytem
- Ziele
- Technologien

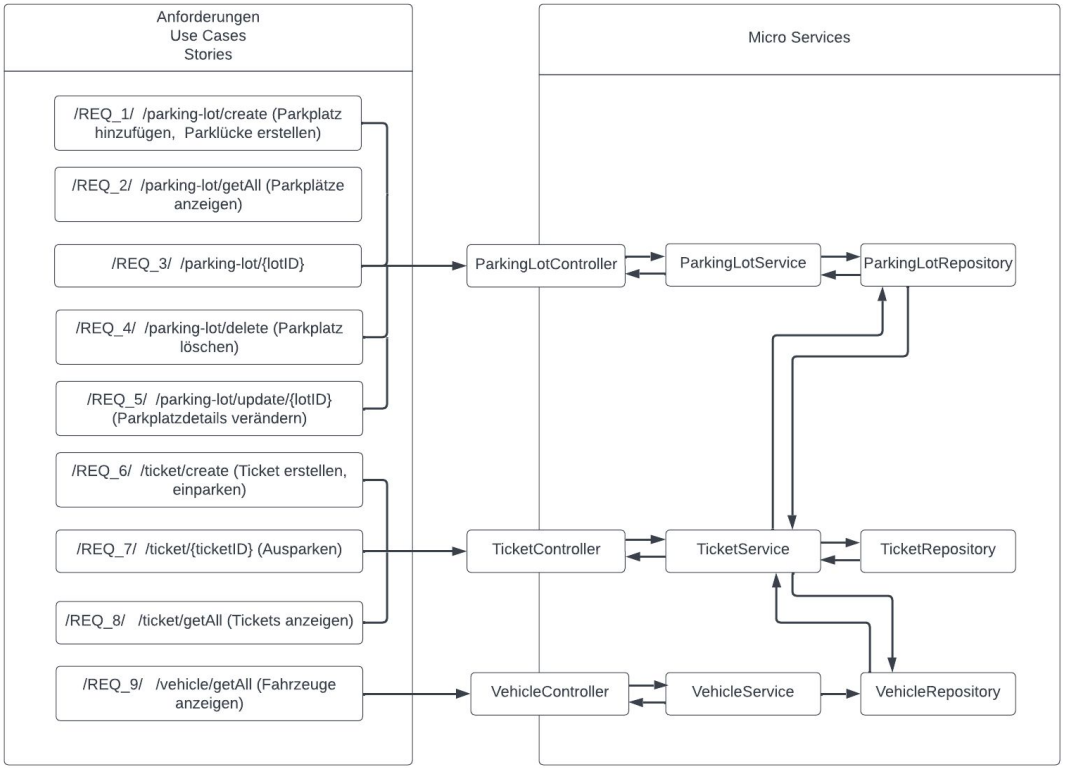
Vorstellung von Use Cases

Von Pau angebotene Dienste/Use Cases:

- Ticket erstellen
- Parkplatz erstellen
- Ausparken
- Parkplatz löschen
- Parkplatzdetails verändern
- Parkplätze anzeigen
- Tickets anzeigen
- Fahrzeuge anzeigen

Vorstellung der Systemarchitektur




Anforderungen
Use Cases
Stories
Micro Services
/REQ_1/ /parking-lot/create (Parkplatz hinzufügen, Parklücke erstellen)
/REQ_2/ /parking-lot/getAll (Parkplätze anzeigen)
/REQ_3/ /parking-lot/{lotID}
/REQ_4/ /parking-lot/delete (Parkplatz löschen)
/REQ_5/ /parking-lot/update/{lotID} (Parkplatzdetails verändern)
/REQ_6/ /ticket/create (Ticket erstellen, einparken)
/REQ_7/ /ticket/{ticketID} (Ausparken)
/REQ_8/ /ticket/getAll (Tickets anzeigen)
/REQ_9/ /vehicle/getAll (Fahrzeuge anzeigen)
ParkingLotController
ParkingLotService
ParkingLotRepository
TicketController
TicketService
TicketRepository
VehicleController
VehicleService
VehicleRepository

Vorstellung von Test Cases

Component Integration Tests sind wie folgt:

- CI_TC_REQ_1: Parkplatz hinzufügen
- CI_TC_REQ_2: Parkplätze anzeigen
- CI_TC_REQ_3: nur ein Parkplatz anzeigen
- CI_TC_REQ_4: Parkplatz löschen
- CI_TC_REQ_5: Parkplatzdetails verändern
- CI_TC_REQ_6: Ticket erstellen
- CI_TC_REQ_7: Ausparken
- CI_TC_REQ_8: Tickets anzeigen
- CI_TC_REQ_9: Fahrzeuge anzeigen

CI_TC_REQ_1

Diese Test Case ist für die createParkingLot Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass parkingLotService gemockt ist.
2. Ein ParkingLotCreateRequest wird hergestellt.
3. Junit Test Case ruft createParkingLot mit request auf.
4. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_2

Diese Test Case ist für die getAll Use Case von ParkingLotController.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass parkingLotService gemockt ist.
2. Ein ParkingLotDtoList wird hergestellt.
3. Junit Test Case ruft getAll auf.
4. Response Data werden analysiert und die erwarteten Daten werden ausgewählt

CI_TC_REQ_3

Diese Test Case ist für getParkingLotById Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass parkingLotService gemockt ist.
2. Ein ParkingLotDto wird hergestellt.
3. Ein ParkingLotResponse wird hergestellt.
4. Junit Test Case ruft getParkingLotById mit lotID = 1 auf.
5. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_4

Diese Test Case ist für die deleteParkingLot Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass parkingLotService gemockt ist.
2. Ein ParkingLotResponse wird hergestellt.
3. Ein ParkingLotDto wird hergestellt.
4. Junit Test Case ruft deleteParkingLot mit lotID = 1 auf.
5. Response Data werden analysiert und die erwarteten Daten werden ausgewählt

CI_TC_REQ_5

Diese Test Case ist für die updateParkingLot Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass parkingLotService gemockt ist.
2. Ein ParkingLotUpdateRequest wird hergestellt.
3. Ein ParkingLotDto wird hergestellt.
4. Ein ParkingLotResponse wird hergestellt.
5. Junit Test Case ruft updateParking mit lotID = 1 und parkingLotUpdateRequest auf.
6. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_6

Diese Test Case ist für die createTicket Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass ticketService gemockt ist.
2. Ein TicketCreateRequest wird hergestellt.
3. Ein TicketDto wird hergestellt
4. Ein TicketResponse wird hergestellt.
5. Junit Test Case ruft createTicket mit TicketCreateRequest auf.
6. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_7

Diese Test Case ist für parkOut Use Case.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass ticketService gemockt ist.
2. Ein TicketDto wird hergestellt
3. Ein TicketResponse wird hergestellt.
4. Junit Test Case ruft parkOut mit ticketID auf.
5. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_8

Diese Test Case ist für die getAll Use Case von TicketController.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass ticketService gemockt ist.
2. Ein TicketDtoList wird hergestellt.
3. Junit Test Case ruft getAll auf.
4. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

CI_TC_REQ_9

Diese Test Case ist für die getAll Use Case von VehicleController.

Test Steps sind wie folgt:

1. Es wird sichergestellt, dass vehicleService gemockt ist.
2. Ein VehicleDtoList wird hergestellt
3. Junit Test Case ruft getAll auf.
4. Response Data werden analysiert und die erwarteten Daten werden ausgewählt.

Vielen Dank für Ihre Aufmerksamkeit