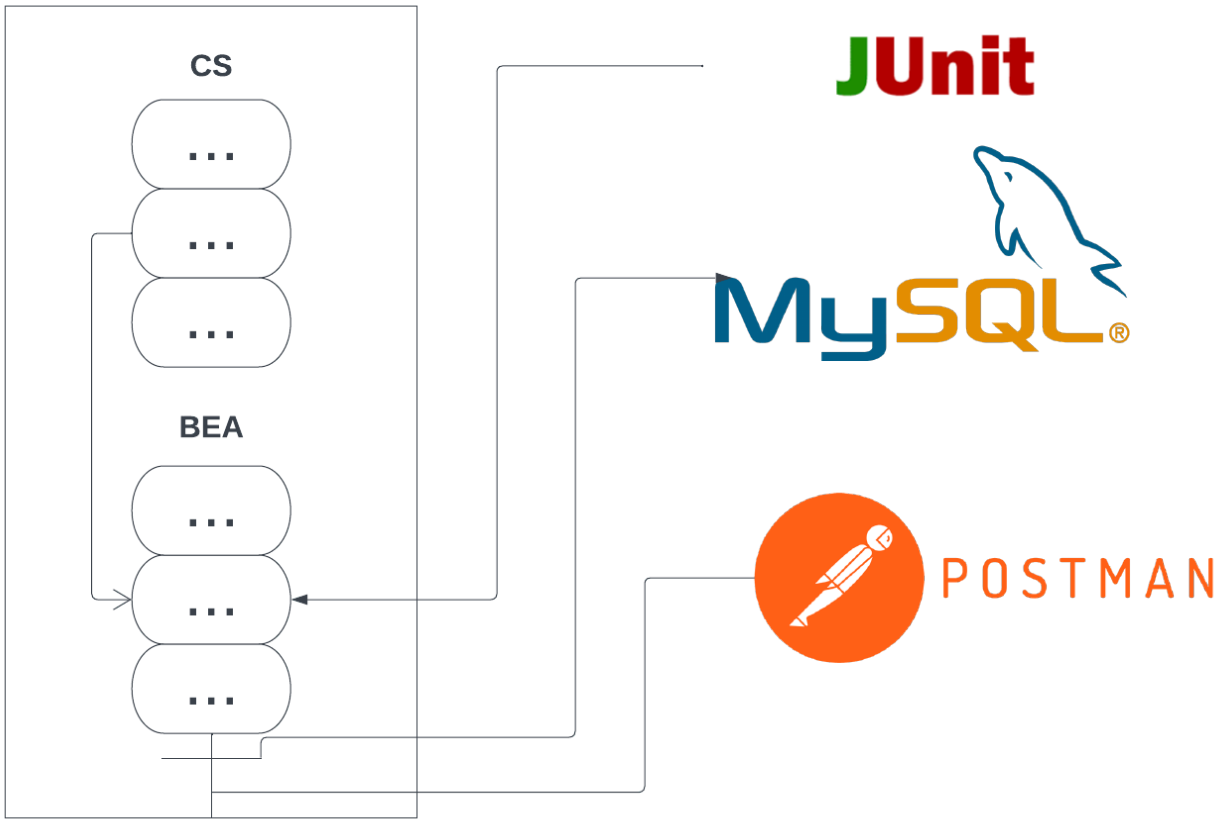
SYSTEMTESTDOKUMENT

1.Systemüberblick:



Der Benutzer sendet eine Anfrage und der Server empfängt und verarbeitet sie. Der Server erstellt und verwaltet Testfälle, automatisiert den Test und ruft Ergebnisse ab und verarbeitet sie. Die Ergebnisse werden dann gemeldet und dem Benutzer präsentiert. Dieser Entwurf bietet ein fortschrittlicheres Testsystem, indem er die Verwaltung und Automatisierung von Testfällen hinzufügt. Testfälle können verschiedene Anwendungsfälle, Eingabewerte, Erwartungen und andere Testparameter umfassen. Die Testautomatisierung ermöglicht die automatische Ausführung von Testfällen und den automatischen Abruf von Ergebnissen. Darüber hinaus kann das Anwendungstestsystem häufig Fehlermanagement, Sicherheitstests, Leistungstests, Datenbankverwaltung und andere Testkomponenten umfassen. Dieser Entwurf soll ein grundlegendes Verständnis eines allgemeinen erweiterten Testsystems vermitteln und kann an unsere Bedürfnisse angepasst werden.

2.Systemfälle:

2.1.1.Pre-Conditions:

Sie müssen über die Webservice-URL und den API-Schlüssel einer Mietwagenanwendung verfügen. Der Mietwagen-Webservice muss zugänglich und betriebsbereit sein.

2.1.2.Test-steps:

Name des Testfalls: Autovermietungsprozess
Beschreibung des Testfalls: In diesem Szenario testen wir den Autovermietungsprozess per Webservice.
Schritte des Testfalls:

Erstellen Sie gültige Benutzerinformationen, Fahrzeugoptionen und Mietdaten.

```
const userData = {
  firstName: "John",
  lastName: "Doe",
  email: "john.doe@example.com",
  phone: "555-1234567"
};

const carOptions = {
  brand: "Toyota",
  model: "Corolla",
  fuelType: "Benzin",
  transmission: "Otomatik"
};

const rentalDates = {
  startDate: "2023-06-10",
  endDate: "2023-06-15"
};
```

Senden Sie eine HTTP-POST-Anfrage mit der Webservice-URL und dem API-Schlüssel.

```
pm.sendRequest({
  url: "https://api.example.com/rental",
  method: "POST",
  headers: {
    "Content-Type": "application/json",
    "Authorization": "Bearer YOUR_API_KEY"
  },
  body: {
    mode: "raw",
    raw: JSON.stringify({
      userData,
      carOptions,
      rentalDates
    })
  }
}, function (err, response) {
  if (err) {
    console.error(err);
  } else {
    console.log(response.json());
  }
});
```

Holen Sie sich die Antwort des Webservice und überprüfen Sie die zurückgegebenen Daten.

```
pm.test("Check response status code", function () {
  pm.response.to.have.status(200);
});

pm.test("Check response body", function () {
  const responseBody = pm.response.json();
  pm.expect(responseBody).to.have.property("transactionId");
  pm.expect(responseBody).to.have.property("rentalDetails");
});
```

Vergleichen Sie erwartete Ergebnisse und zurückgegebene Daten.

```
pm.test("Compare expected and actual results", function () {
const responseBody = pm.response.json();
pm.expect(responseBody.transactionId).to.exist;
pm.expect(responseBody.rentalDetails).to.be.an("object"); });
```

Speichern Sie das Testergebnis.

2.1.3.Post-Conditions:

Die Webservice-Anfrage wird erfolgreich abgeschlossen und die Autovermietung wird bearbeitet.

2.1.4.Test Data:

Nutzerinformation:
Name: John
Nachname: Doe
E-Mail: john.doe@example.com
Telefon: 555-1234567
Fahrzeugoptionen:
Marke: Toyota
Modell: Corolla
Kraftstofftyp: Benzin
Getriebetyp: Automatik
Mietdaten:
Startdatum: 10.06.2023
Enddatum: 15.06.2023

2.1.5.Expected Result:

Nach erfolgreichem Abschluss der Anfrage werden vom Webservice eine Transaktionsnummer und Mietinformationen zurückgegeben.

2.1.6.Actual Result:

Vom Webservice als Ergebnis der Anfrage zurückgegebene Daten.

2.1.7.Verdict:

Das Ergebnis des Testszenarios wird bewertet (Pass, Fail).

2.2.Database connection

Eine Autovermietungsanwendung muss über eine Datenbankverbindung und Tabellen verfügen.

Name des Testfalls: Autovermietungsprozess

In diesem Szenario testen wir den Autovermietungsprozess über die Datenbank.

Erstellen Sie gültige Benutzerinformationen, Fahrzeugoptionen und Mietdaten.

```
INSERT INTO users (first_name, last_name, email, phone)
VALUES ('John', 'Doe', 'john.doe@example.com', '555-1234567');

INSERT INTO cars (brand, model, fuel_type, transmission)
VALUES ('Toyota', 'Corolla', 'Benzin', 'Otomatik');
```

Wir können eine praktische Tabellenstruktur für Mietdaten verwenden oder Start- und Enddaten in einer einzigen Tabelle speichern.

Führen Sie den Mietvorgang durch.

Tabelle: Vermietungen

```
INSERT INTO rentals (user_id, car_id, start_date, end_date)
VALUES (1, 1, '2023-06-10', '2023-06-15');
```

Fragen Sie die Transaktion ab und überprüfen Sie die zurückgegebenen Daten.

```
SELECT r.transaction_id, u.first_name, u.last_name, c.brand, c.model
FROM rentals r
JOIN users u ON r.user_id = u.user_id
JOIN cars c ON r.car_id = c.car_id
WHERE r.transaction_id = 'XYZ123';
```

Vergleichen Sie erwartete Ergebnisse und zurückgegebene Daten.

Wir können die mit der SQL-Abfrage zurückgegebenen Ergebnisse mit den erwarteten Ergebnissen vergleichen.

Speichern Sie das Testergebnis.

Der Mietwagenvorgang wurde erfolgreich in der Datenbank durchgeführt.

3.Rückverfolgbarkeit:

Testszenario 1: Neuwagenmiete
Anwendungsfall: Autovermietung
Webservice-Endpunkt: /rental
****

Testszenario 2: Vorhandenes Auto mieten
Anwendungsfall: Autovermietung
Webservice-Endpunkt: /rental
****

Testfall 3: Abfrage einer gültigen Miete
Anwendungsfall: Anfrage für Mietprozess
Webservice-Endpunkt: /rental/{transactionId}
****

Testfall 4: Stornierung einer gültigen Anmietung
Anwendungsfall: Stornierung einer Miete
Webservice-Endpunkt: /rental/{transactionId}/cancel
****

Testszenario 5: Verlängerung der Autovermietungsdauer
Anwendungsfall: Verlängerung der Autovermietungsdauer
Webservice-Endpunkt: /rental/{transactionId}/extend
****

Testszenario 6: Verfügbare Mietwagenoptionen auflisten
Anwendungsfall: Auflistung von Mietwagenoptionen
Webservice-Endpunkt: /options/rental

Da stellt jeder Testfall einen Anwendungsfall dar, und diese Anwendungsfälle sind mit den entsprechenden Webservice-Endpunkten verknüpft. Die Durchführung jedes Testfalls dient dazu, die Anforderungen des jeweiligen Anwendungsfalls zu testen und zu überprüfen, ob die Webservice-Endpunkte wie erwartet funktionieren. Traces (Links) zeigen die Beziehung zwischen den Use Case- und Webservice-Endpunkten jedes Testfalls und helfen uns sicherzustellen, dass der Testprozess den Anforderungen entspricht.

Buğra Balcı
Fırat Ulaş Güneş