Capston 2 for Web Application Development by JAVA

Car Rental

Kholud Mohammed Almutairi

Outlines

02

Car rental

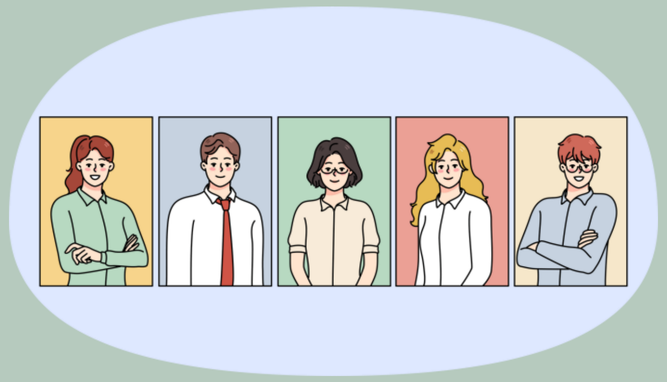
- System for renting cars
- Rentals available for specified periods, usually ranging from days to several days.
- Wide range of car options offered.
- Users can browse available cars and view details like model, price, and availability.

Tables

Car

Customers

Reservations


Employees

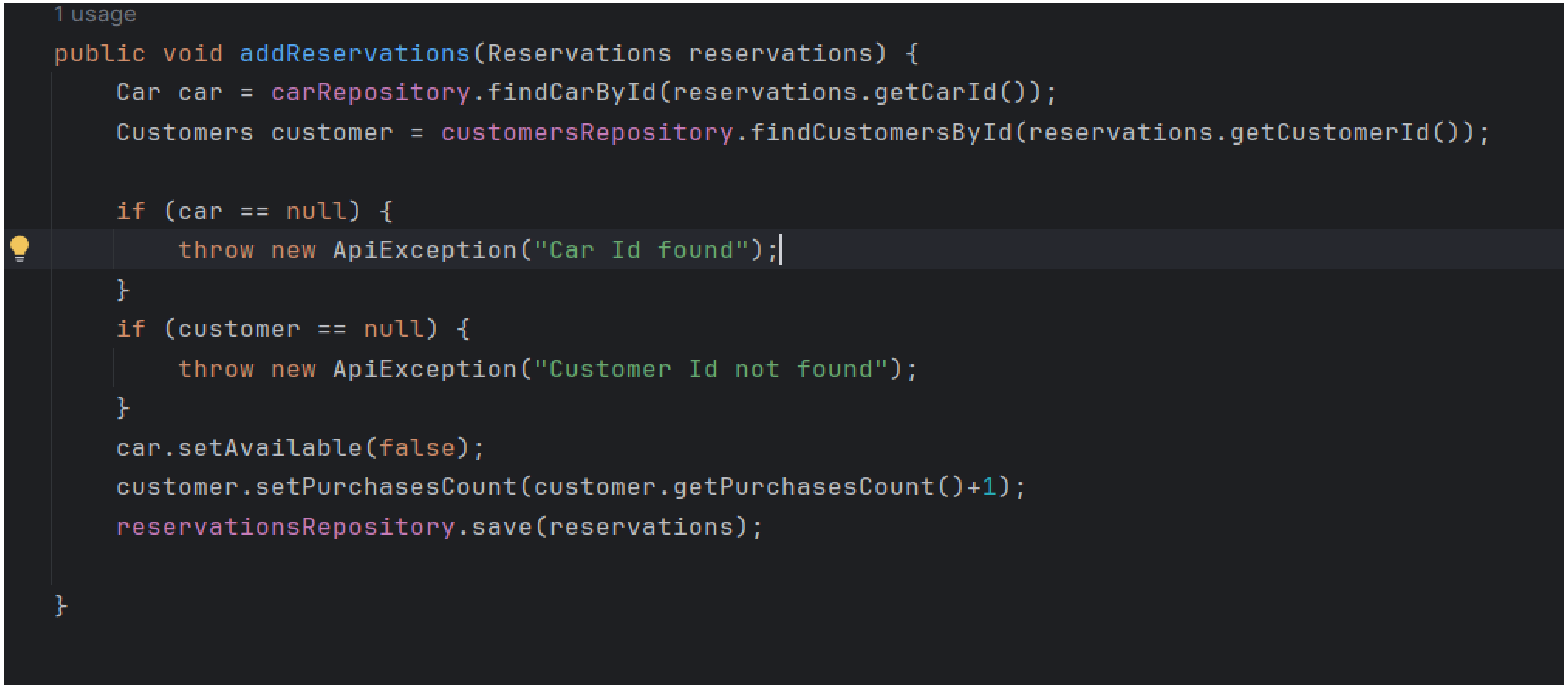
Invoices

03

Car rental

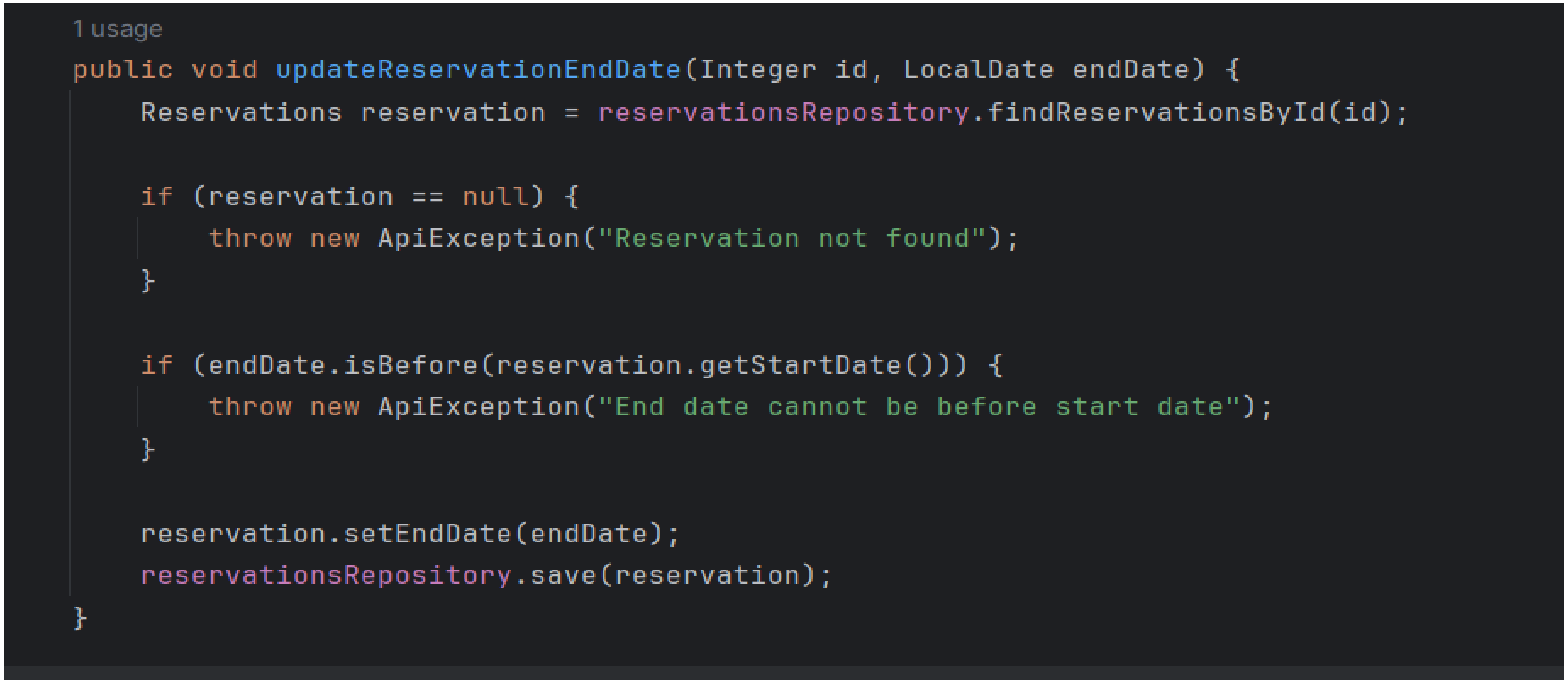
- Provides a convenient and flexible transportation solution for various purposes such as travel.
- Extending the reservation period and displaying the available cars that can be booked. Customers can reserve and rent the available car.

Reservations



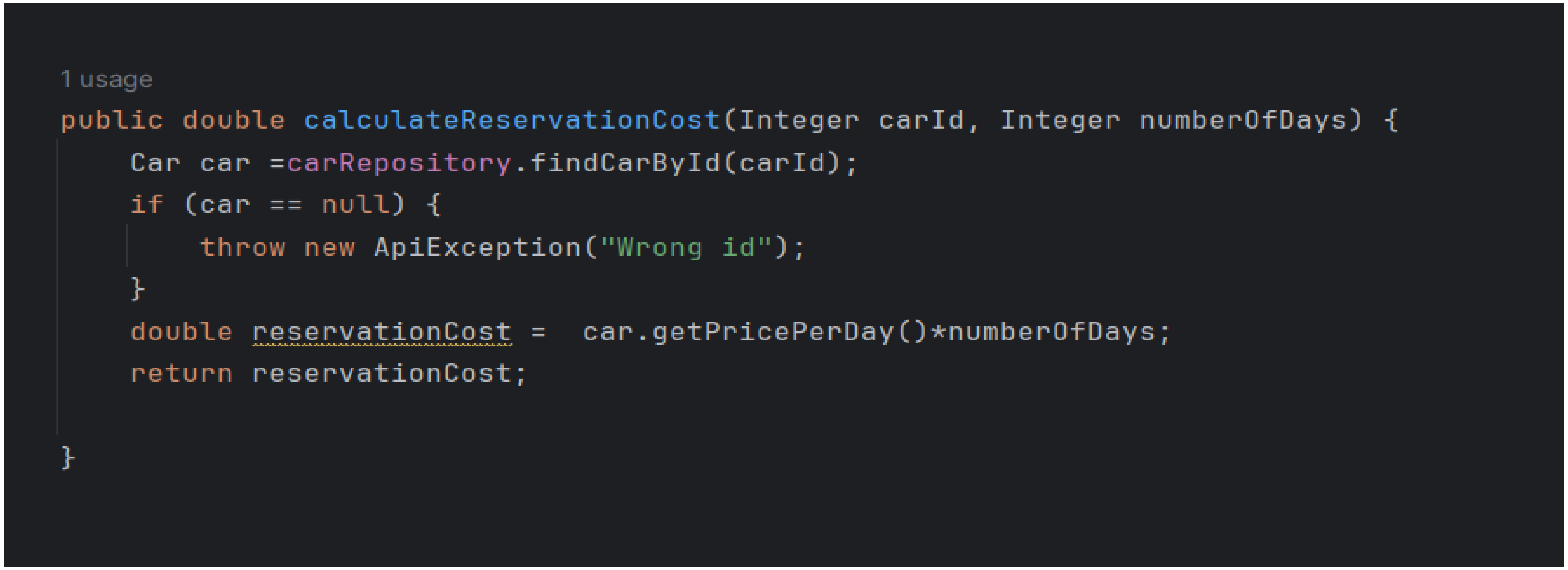
```
1 usage
public void addReservations(Reservations reservations) {
    Car car = carRepository.findCarById(reservations.getCarId());
    Customers customer = customersRepository.findCustomersById(reservations.getCustomerId());

    if (car == null) {
        throw new ApiException("Car Id found");
    }
    if (customer == null) {
        throw new ApiException("Customer Id not found");
    }
    car.setAvailable(false);
    customer.setPurchasesCount(customer.getPurchasesCount()+1);
    reservationsRepository.save(reservations);

}
```

Extend the reservation



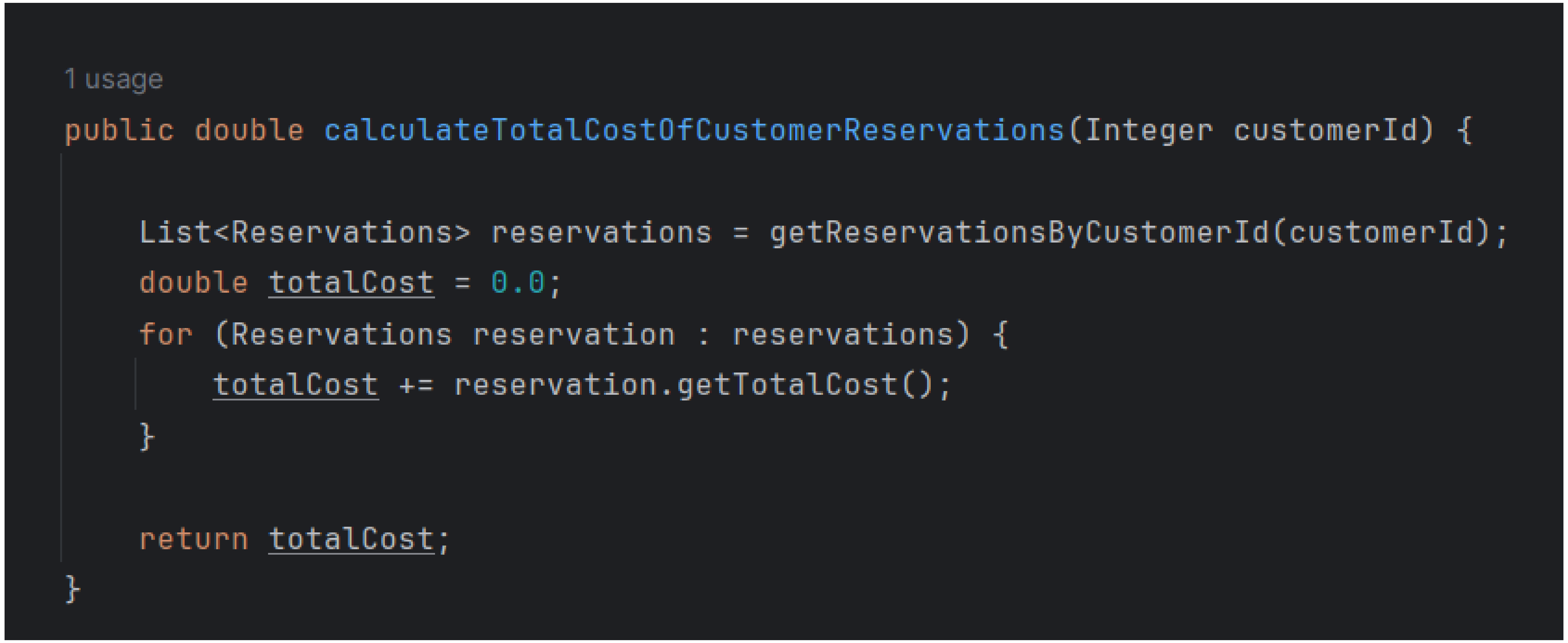
```
1 usage
public void updateReservationEndDate(Integer id, LocalDate endDate) {
    Reservations reservation = reservationsRepository.findReservationsById(id);

    if (reservation == null) {
        throw new ApiException("Reservation not found");
    }

    if (endDate.isBefore(reservation.getStartDate())) {
        throw new ApiException("End date cannot be before start date");
    }

    reservation.setEndDate(endDate);
    reservationsRepository.save(reservation);
}
```

Calculate the cost based on the number of days (without discount)



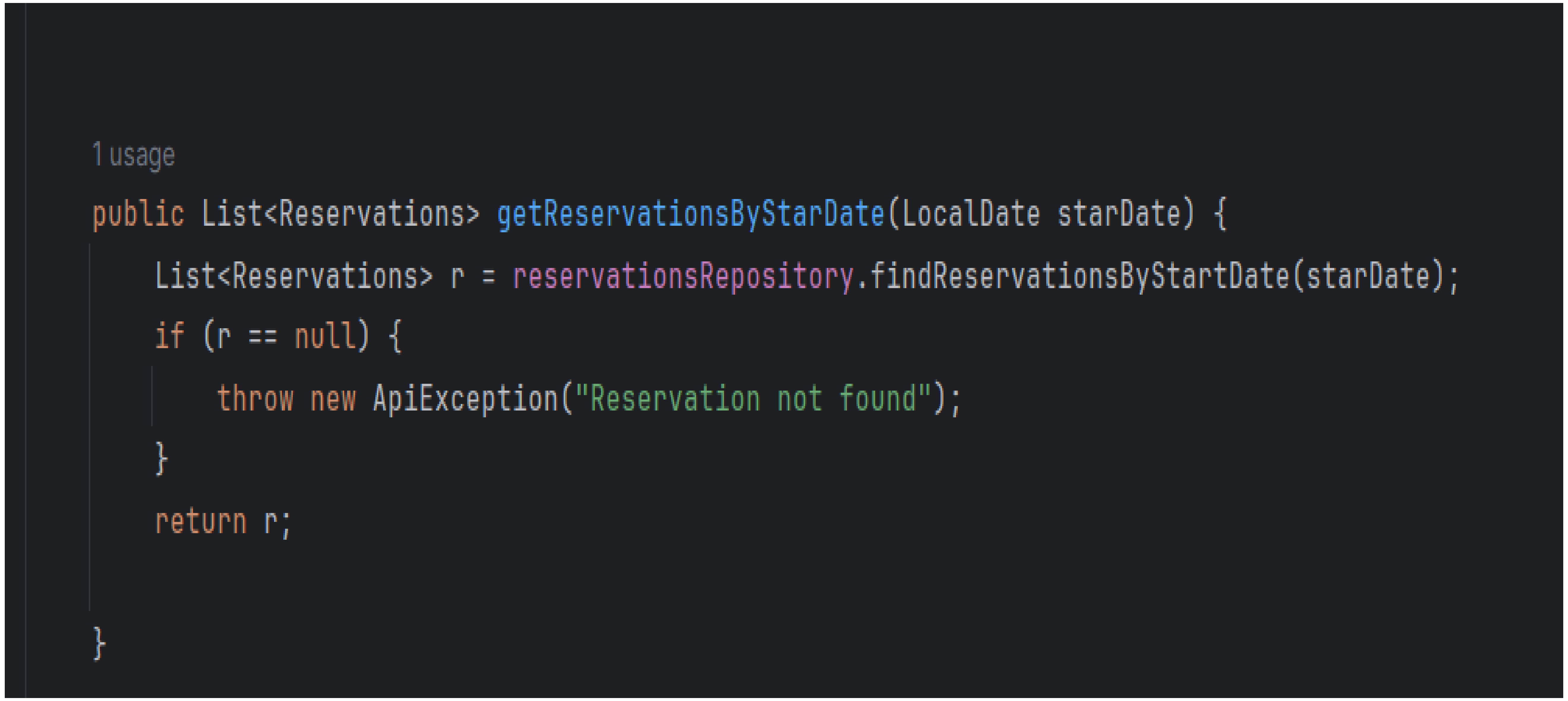
```
1 usage
public double calculateReservationCost(Integer carId, Integer numberOfDays) {
    Car car =carRepository.findCarById(carId);
    if (car == null) {
        throw new ApiException("Wrong id");
    }
    double reservationCost =  car.getPricePerDay()*numberOfDays;
    return reservationCost;

}
```

Calculate total cost of the reservation for a specific customer.



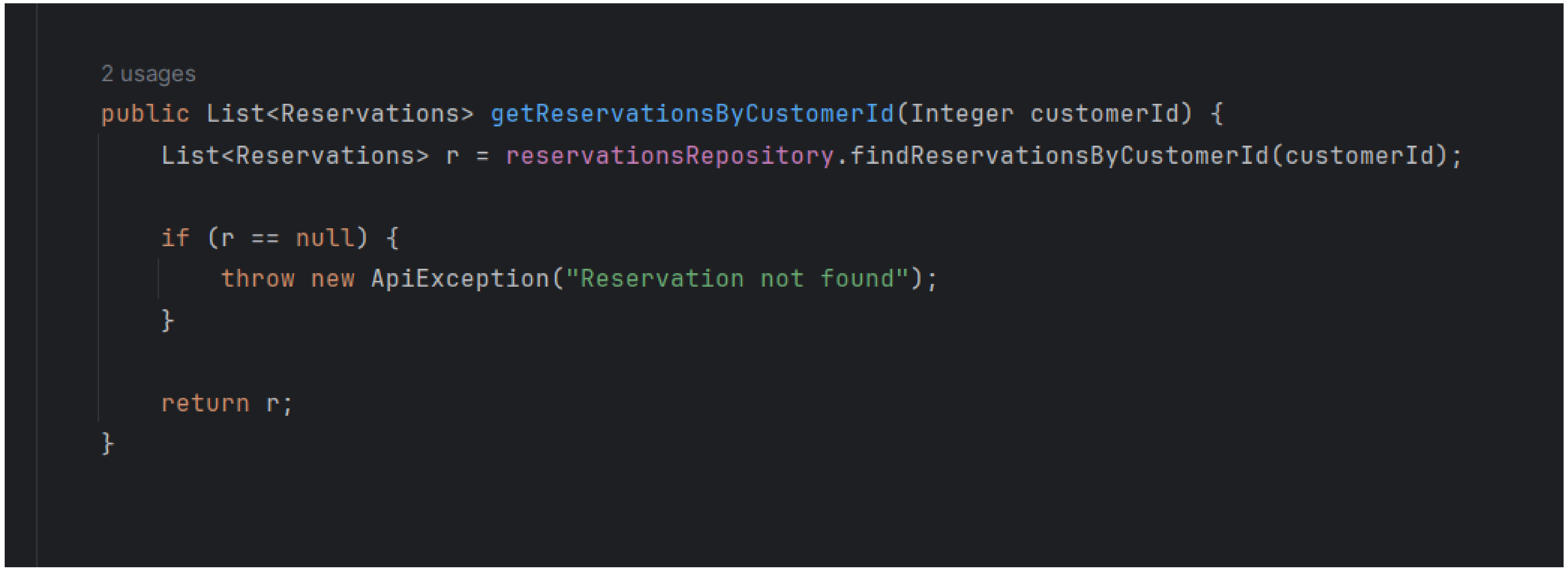
```
1 usage
public double calculateTotalCostOfCustomerReservations(Integer customerId) {

    List<Reservations> reservations = getReservationsByCustomerId(customerId);
    double totalCost = 0.0;
    for (Reservations reservation : reservations) {
        totalCost += reservation.getTotalCost();
    }

    return totalCost;
}
```

Get Reservation By start date



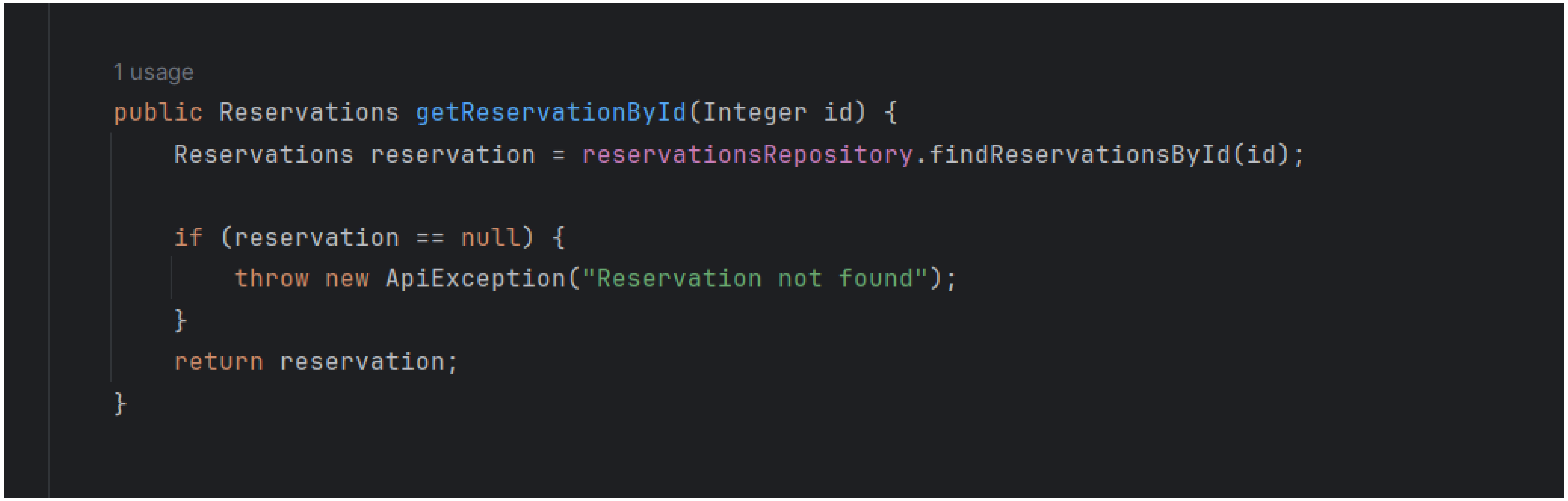
```
1 usage
public List<Reservations> getReservationsByStarDate(LocalDate starDate) {
    List<Reservations> r = reservationsRepository.findReservationsByStartDate(starDate);
    if (r == null) {
        throw new ApiException("Reservation not found");
    }
    return r;

}
```

Get Reservation By customer id



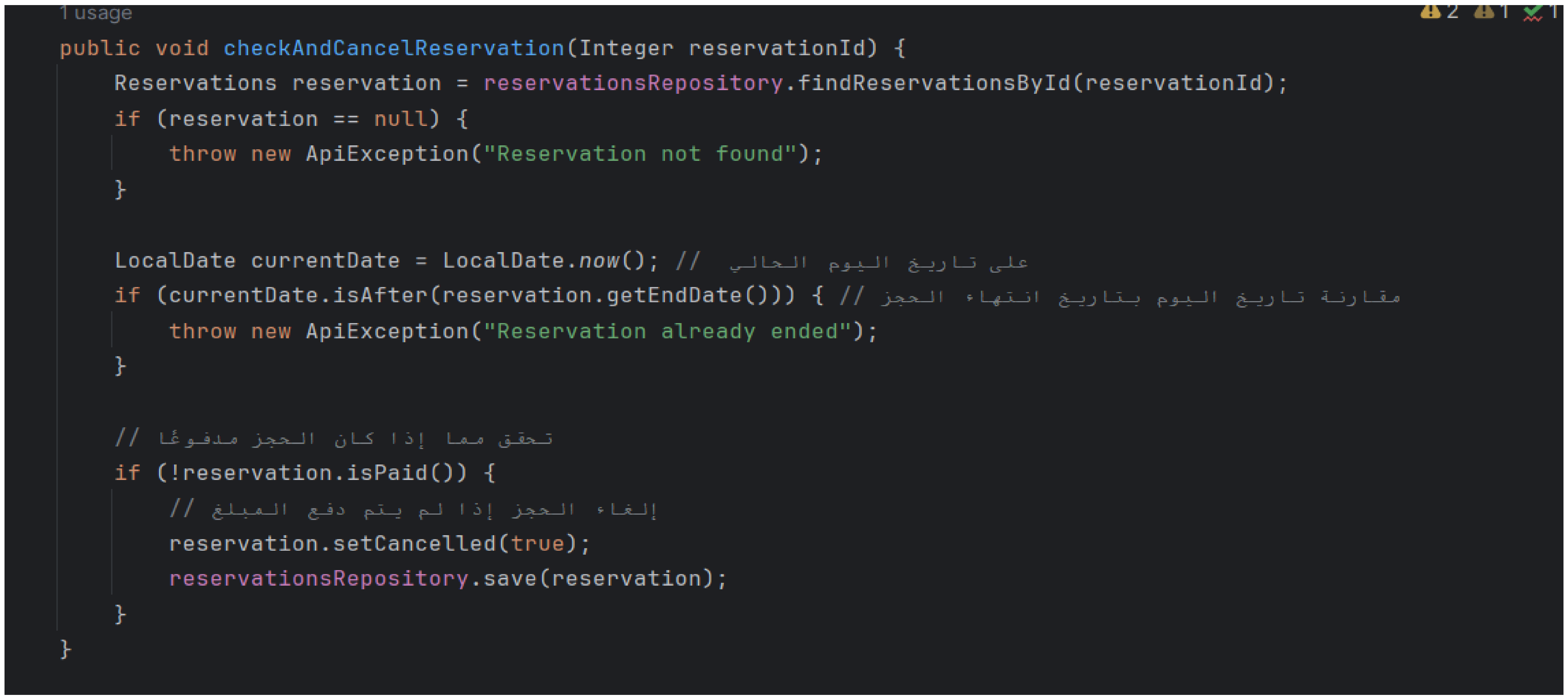
```
2 usages
public List<Reservations> getReservationsByCustomerId(Integer customerId) {
    List<Reservations> r = reservationsRepository.findReservationsByCustomerId(customerId);

    if (r == null) {
        throw new ApiException("Reservation not found");
    }

    return r;
}
```

Get Reservation By id



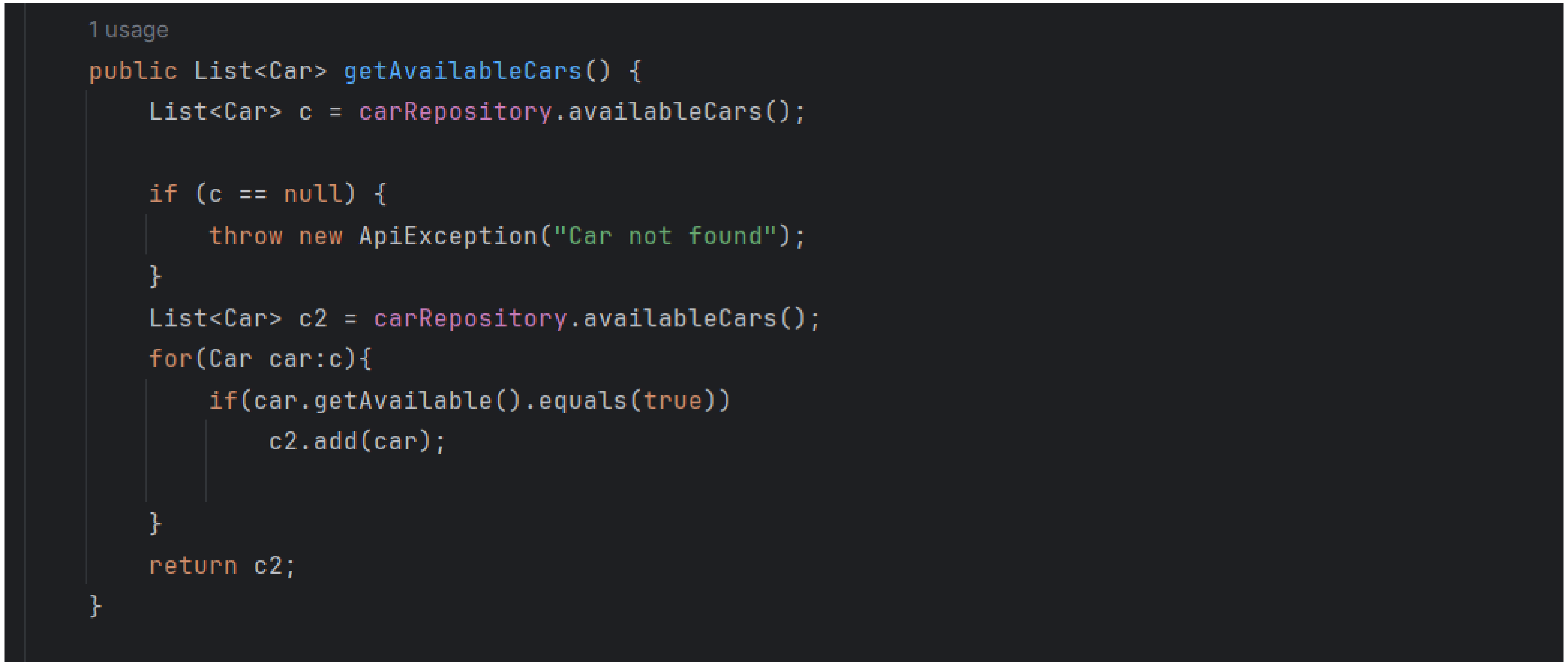
```
1 usage
public Reservations getReservationById(Integer id) {
    Reservations reservation = reservationsRepository.findReservationsById(id);

    if (reservation == null) {
        throw new ApiException("Reservation not found");
    }
    return reservation;
}
```

Check and cancel Reservation



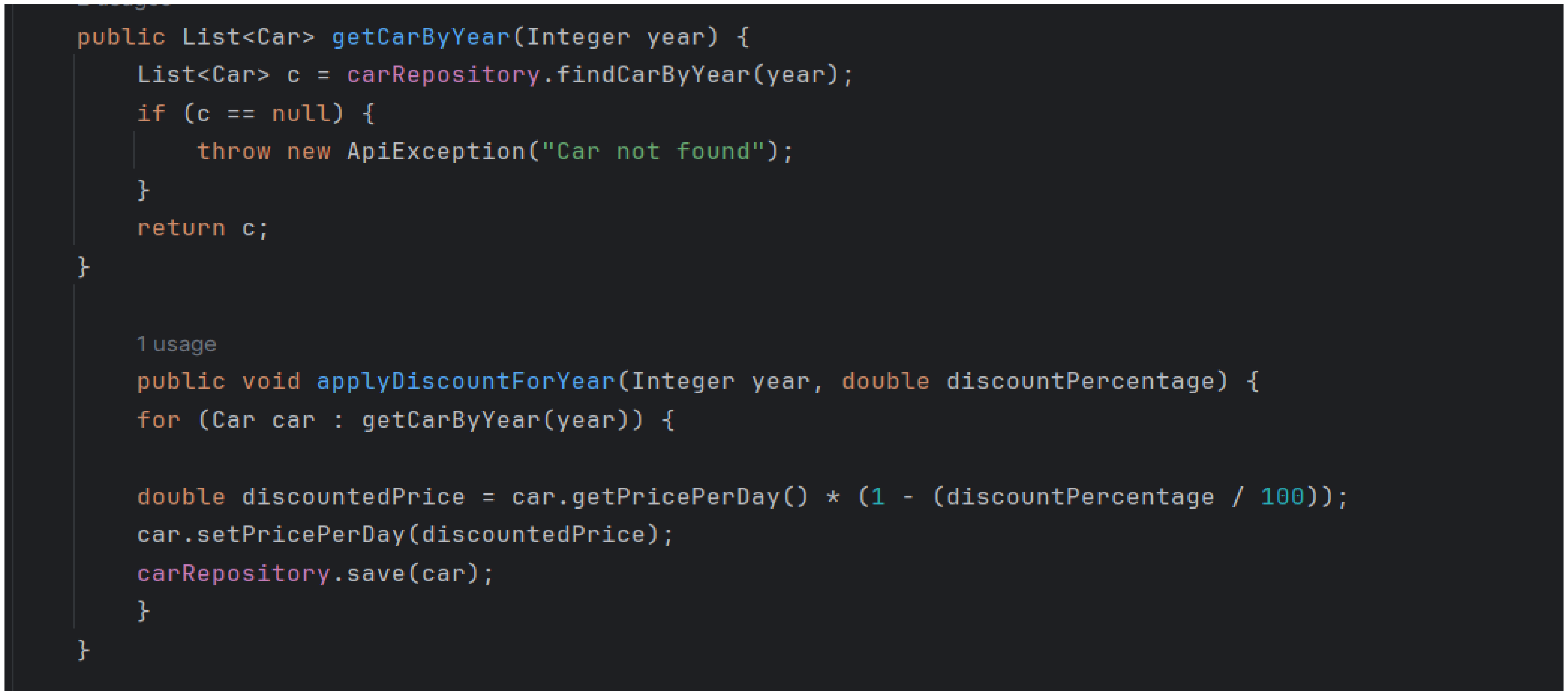
```
1 usage
public void checkAndCancelReservation(Integer reservationId) {
    Reservations reservation = reservationsRepository.findReservationsById(reservationId);
    if (reservation == null) {
        throw new ApiException("Reservation not found");
    }

    LocalDate currentDate = LocalDate.now(); // على تاريخ اليوم الحالي
    if (currentDate.isAfter(reservation.getEndDate())) { // مقارنة تاريخ اليوم بتاريخ انتهاء الحجز
        throw new ApiException("Reservation already ended");
    }

    // تحقق مما إذا كان الحجز مدفوعًا
    if (!reservation.isPaid()) {
        // إلغاء الحجز إذا لم يتم دفع المبلغ
        reservation.setCancelled(true);
        reservationsRepository.save(reservation);
    }
}
```

Available Cars



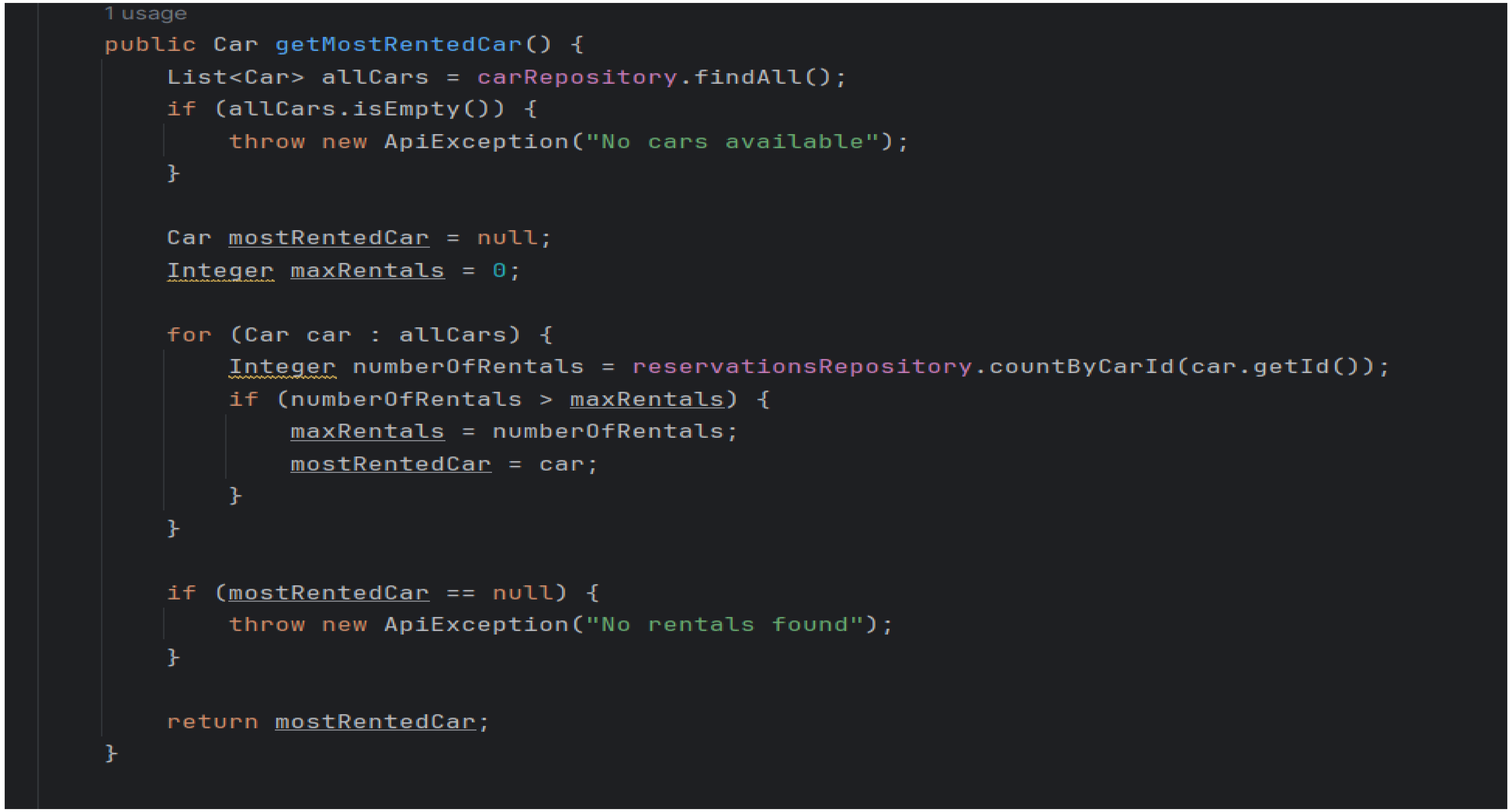
```
1 usage
public List<Car> getAvailableCars() {
    List<Car> c = carRepository.availableCars();

    if (c == null) {
        throw new ApiException("Car not found");
    }
    List<Car> c2 = carRepository.availableCars();
    for(Car car:c){
        if(car.getAvailable().equals(true))
            c2.add(car);

    }
    return c2;
}
```

Discount on cars of a specific year



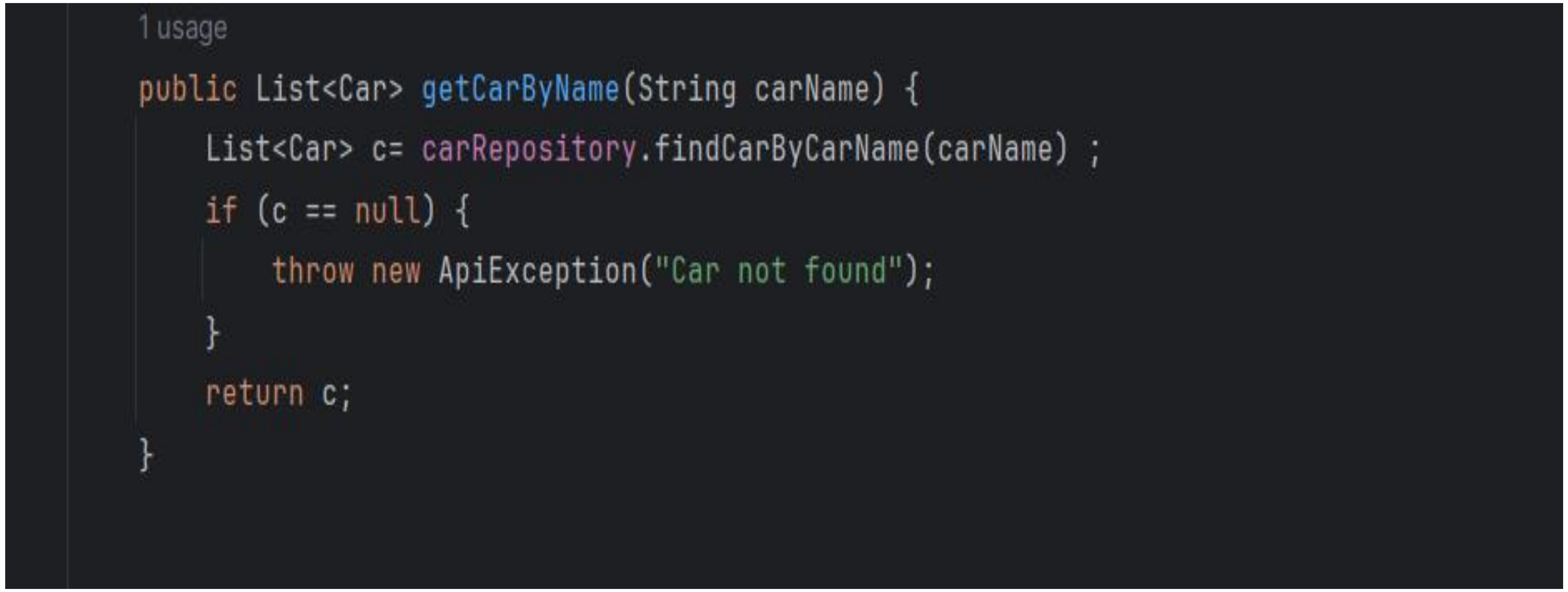
```java
public List<Car> getCarByYear(Integer year) {
    List<Car> c = carRepository.findCarByYear(year);
    if (c == null) {
        throw new ApiException("Car not found");
    }
    return c;
}

    1 usage
    public void applyDiscountForYear(Integer year, double discountPercentage) {
    for (Car car : getCarByYear(year)) {

    double discountedPrice = car.getPricePerDay() * (1 - (discountPercentage / 100));
    car.setPricePerDay(discountedPrice);
    carRepository.save(car);
    }
}
```

Most rented car



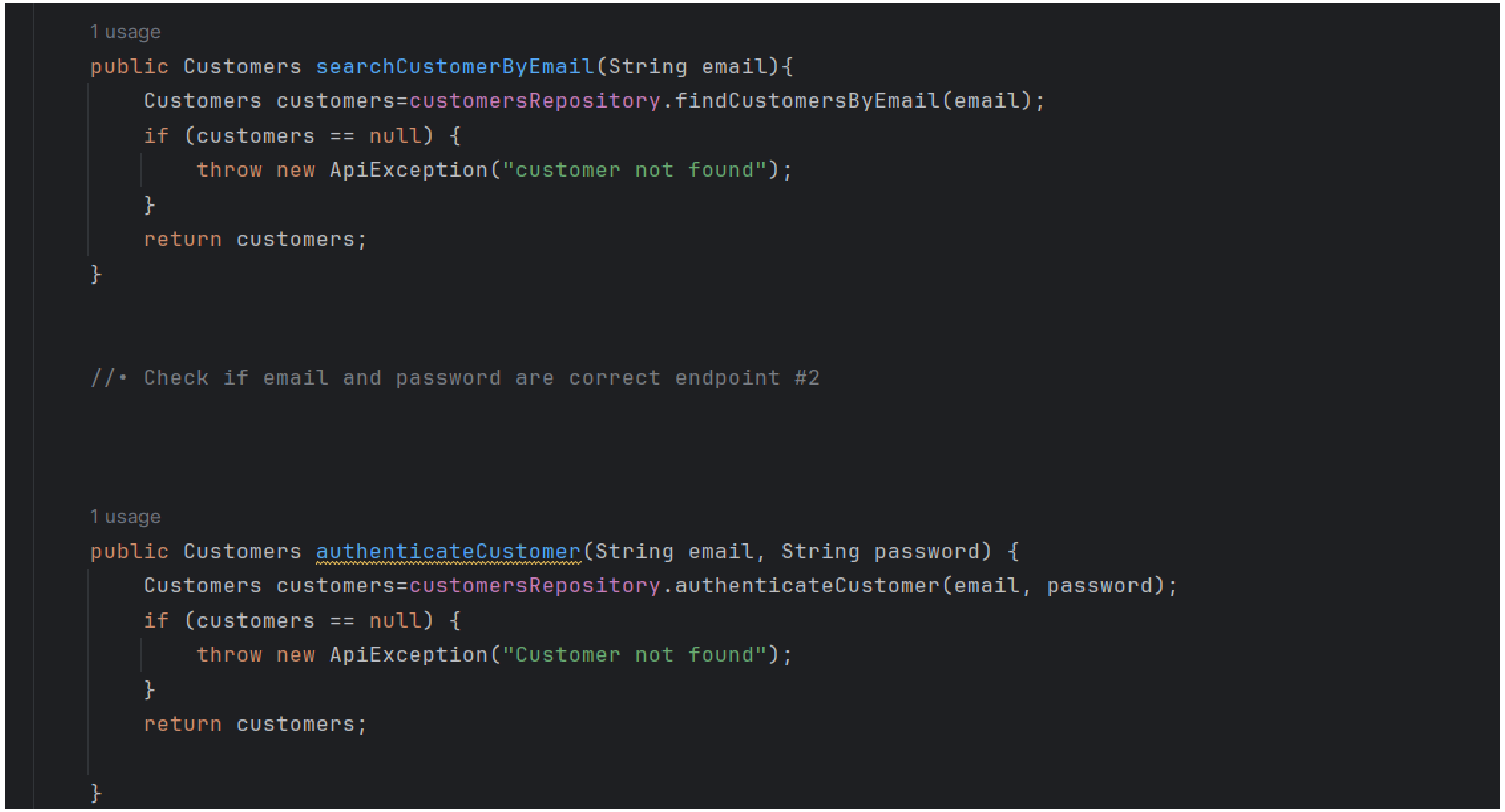
```
1 usage
public Car getMostRentedCar() {
    List<Car> allCars = carRepository.findAll();
    if (allCars.isEmpty()) {
        throw new ApiException("No cars available");
    }

    Car mostRentedCar = null;
    Integer maxRentals = 0;

    for (Car car : allCars) {
        Integer numberOfRentals = reservationsRepository.countByCarId(car.getId());
        if (numberOfRentals > maxRentals) {
            maxRentals = numberOfRentals;
            mostRentedCar = car;
        }
    }

    if (mostRentedCar == null) {
        throw new ApiException("No rentals found");
    }

    return mostRentedCar;
}
```

Car by name



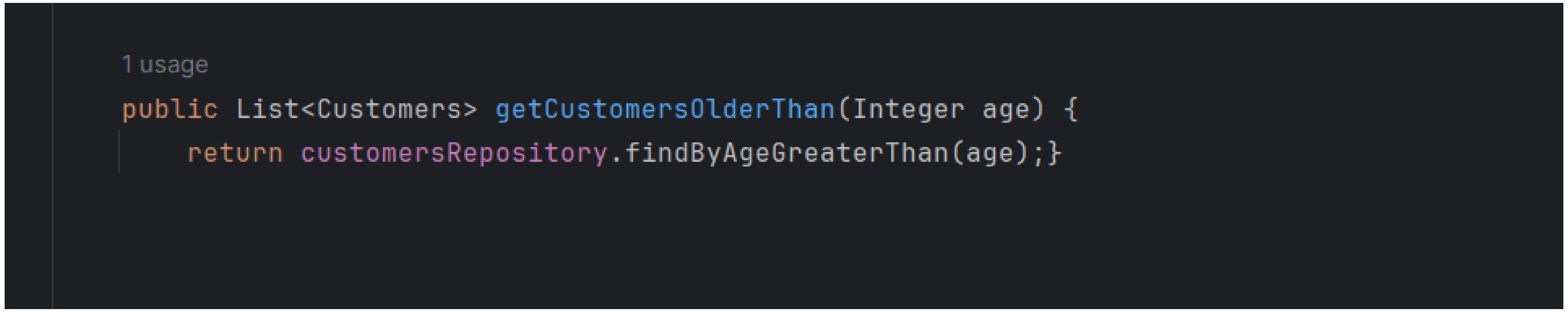
```
1 usage
public List<Car> getCarByName(String carName) {
    List<Car> c= carRepository.findCarByCarName(carName) ;
    if (c == null) {
        throw new ApiException("Car not found");
    }
    return c;
}
```

Authenticate Customer& search by email

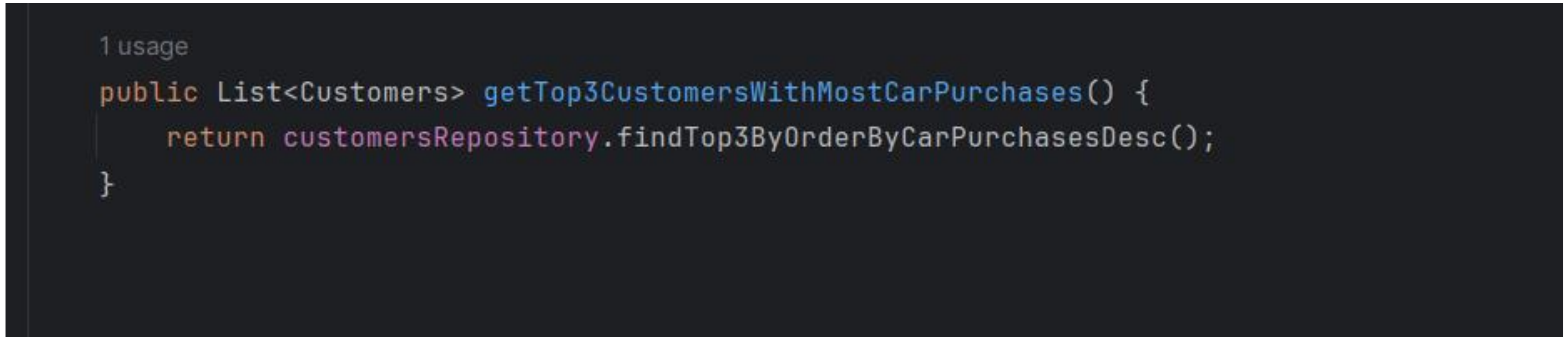


```
1 usage
public Customers searchCustomerByEmail(String email){
    Customers customers=customersRepository.findCustomersByEmail(email);
    if (customers == null) {
        throw new ApiException("customer not found");
    }
    return customers;
}


//• Check if email and password are correct endpoint #2



1 usage
public Customers authenticateCustomer(String email, String password) {
    Customers customers=customersRepository.authenticateCustomer(email, password);
    if (customers == null) {
        throw new ApiException("Customer not found");
    }
    return customers;

}
```

Customer older than& Top Customer



```
1 usage
public List<Customers> getCustomersOlderThan(Integer age) {
    return customersRepository.findByAgeGreaterThan(age);}
```



```
1 usage
public List<Customers> getTop3CustomersWithMostCarPurchases() {
    return customersRepository.findTop3ByOrderByCarPurchasesDesc();
}
```

Thank You

Do you have any questions ?