STAGE
Full-stack Application Development Challenge

End Stage & Exit

00D : 03H : 58M : 42S

Parking Lot Design

Design a simple parking lot application, which allows users to park a vehicle and get information on parked vehicles.

STEP 1 (Spring Boot + MySQL) :

Open the file `ParkingLot.java` in the project under the package `com.skillenza.parkinglotjava` and Create a parking lot model with following properties.

Field	Type	Null	Key	Default	Extra
id	bigint(20)	NO	PRI	NULL	auto_increment
created_at	datetime	NO		NULL	
lot	int(11)	NO	UNI	NULL	
parking_amount	int(11)	NO		NULL	
parking_duration	int(11)	NO		NULL	
updated_at	datetime	NO		NULL	
vehicle_number	int(11)	NO	UNI	NULL	

Name the table as `parkinglots` and also apply all the Getters and Setters methods for this model.

```
import javax.persistence.*;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import java.util.Date;

@Entity
@Table(name = "parkinglots")
@EntityListeners(AuditingEntityListener.class)
@JsonIgnoreProperties(value = {"createdAt", "updatedAt"},
        allowGetters = true)
public class ParkingLot {

    //your code goes here

}
```

STEP 2:

STAGE

Full-stack Application Development Challenge

End Stage & Exit

00D : 03H : 58M : 42S

This class implements a `@RestController` with the following mapping

1. `@GetMapping("/api/parkings")`
   - Will get all the parked vehicle data in the below format.

```
[
{
"createdAt": "2018-06-12T10:00:00.000+0000",
"id": 1,
"lot": 2,
"parkingDuration": 60,
"parkingAmount": 20,
"updatedAt": "2018-06-12T11:00:00.000+0000",
"vehicleNumber": 1
}
]
```

2. `@PostMapping("/api/parkings")`
   - Will parking the vehicle and calculate the parking amount and save that in database.
   - Min Parking charge is 20 INR, for 1 hour and for subsequent hours per minute cost will be applied 0.333 INR/Min. Calculate the total cost based on the duration vehicle was parked and save that in the database under the column heading `parking_amount (int)` as shown in STEP 1 Schema.

```
import org.springframework.web.bind.annotation.*;

import javax.validation.Valid;
import java.util.List;


@RestController
@CrossOrigin(origins = "http://localhost:8080")
@RequestMapping("/api")
public class ParkingLotController {

    // your code goes here
}
```

STAGE

Full-stack Application Development Challenge

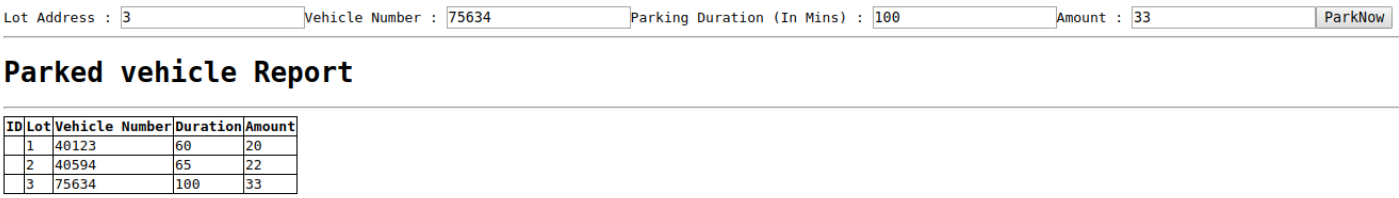
End Stage & Exit

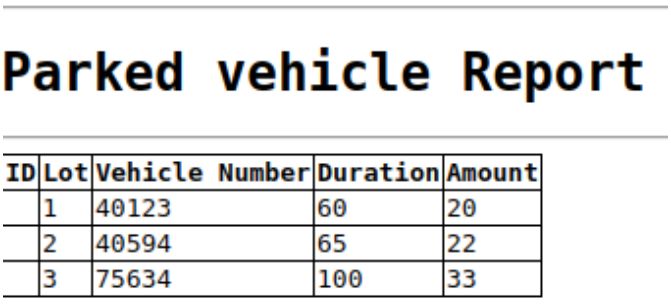
00D : 03H : 58M : 42S

```
import org.springframework.test.context.junit4.SpringRunner;

@RunWith(SpringRunner.class)
@SpringBootTest
public class ParkinglotjavaApplicationTests {


 //your test goes here


}
```

STEP 2 (Angular 7.1.0):

Design the front-end for the above use-case using Angular. Sample screenshot of front-end is given below.





1. You are not allowed to update/add a new dependency on `package.json`.
2. We have already provided you with `app.component.html`.
3. You have to modify `app.component.ts` you are required to fix the following issues.
    - Create `vehicleForm: FormGroup` with all form fields from `app.component.html` using `ReactiveFormsModule`.
    - Complete `getAllParking()` method to get all parking data from our REST APIs developed above in `STEP 1`.
    - Complete the `calculateAmount(event)` method to calculate parking cost based on the explanation given in `step 1` whenever parking duration updated in from.
    - Complete `parkNewvehicle(vehicle)` method to park new vehicle by making a `POST` request to RESTAPIs developed above in `STEP 1`.

STAGE

Full-stack Application Development Challenge

End Stage & Exit

00D : 03H : 58M : 42S

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  // your code goes here

}
```

- Finally complete the `onSubmit()` method that will park the vehicle and if on error show display an error message as shown below, if the vehicle is already parked and also update `app.component.spec.ts` for all possible test for above use-case (`ParkeNewVehicleTest` , `VehicleALreadyParkedTest`).

```
      ],
    }).compileComponents();
  }));

  it('should create the app', () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.debugElement.componentInstance;
    expect(app).toBeTruthy();
  });

  // your test cases goes here

});
```

Lot Address : 3 Vehicle Number : 75634

VEHICLE ALREADY PARKED.

Parked vehicle Report

ID	Lot	Vehicle Number	Duration	Amount
17	1	40123	60	20
18	2	40594	65	22
19	3	75634	100	33

STAGE

Full-stack Application Development Challenge

End Stage & Exit

(https://cdn.skillenza.com/files/61e20661-6b90-4068-bb7c-2c43e9a57cdd/problem.zip)

00D : 03H : 58M : 41S

Feel Free to Reach Us, In case of any query related to the problem statement. click on **HELP BUTTON** on your right.

NOTE: You are not allowed to make changes to other files, not allowed to install extra dependancy. Once you are done with the task, Please share the required files given in the below form.

All the Best!

01. **Mandatory**

Feedback

02. **Mandatory**

ParkingLot.java

Select file

03. **Mandatory**

ParkingLotController.java

Select file

04. **Mandatory**

ParkinglotjavaApplicationTests.java

Select file

05. **Mandatory**

app.component.ts

Select file

STAGE

Full-stack Application Development Challenge

End Stage & Exit

00D : 03H : 58M : 42S

Select file

07. **Mandatory**

GitHub Public Repository URL

Submit