REST APIs using Spring Data REST Assignment Solutions

Assignment 1 : Implement a department-api

Create a department table and expose the domain model as a rest API.

```
        creating a dept table

        create table department(

        ID int NOT null AUTO_INCREMENT,

        dept_name varchar(20),

        PRIMARY KEY(ID)

);
```

```
@Entity

public class Department {

   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   private long ID;
   private String deptName;

   public long getID() {
       return ID;
```

```
    }

    public void setID(long ID) {

    this.ID = ID;

    }

    public String getDeptName() {

    return deptName;

    }

    public void setDeptName(String deptName) {

    this.deptName = deptName;

    }

}
```

Assignment 2 : Paging and Sorting

```
        @Entity
        public class Department {
        @Id
        @GeneratedValue(strategy = GenerationType.IDENTITY)
         private Long id;
        private String name;

    public Long getId() {
        return id;
}

    public void setId(Long id) {
    this.id = id;

}

public String getName() {
        return name;
}

   public void setName(String name) {
        this.name = name;
}

}
```

Assignment No : 3 JSON Serialization

Event entity:

```
@JsonPropertyOrder({"started","resourceId"})
```

Sample event from Postman test:

```
    "started": null,

        "resourceId": 20,
    "name": "Blade Runner Bootcamp 5",

    "description": "From basics to expert in hunting replicants",

    "startTime": "2017-08-29T11:46:18-03:00",

    "endTime": "2017-08-29T13:46:18-03:00",

    "zoneId": "US/Central",

    "venue": {

    "resourceId": 20,

    "name": "Los Angeles Police Department",

    "streetAddress": "12345",

    "streetAddress2": "Spice Wood Springs",

    "city": "Lost Angeles",

    "state": "California",

    "country": "United States",

    "postalCode": "71250"

},

    "_links": {
```

```
    "self": {

    "href": "http://localhost:8080/eventmanagement-api/events/20"

},

    "event": {

    "href": "http://localhost:8080/eventmanagement-api/events/20"

},

    "participants": {

"href": "http://localhost:8080/eventmanagement-api/events/20/participants"

},

"organizer": {

"href": "http://localhost:8080/eventmanagement-api/events/20/organizer"

}

}

}
```

Assignment 4 : Custom Finder Methods Assignment

```
        public interface VenueRepository extends
PagingAndSortingRepository<Venue, Long> {

    List<Venue> findByPostalCode(@Param("postalCode") String
postalCode);

    //or

        //Page<Venue> findByPostalCode(@Param("postalCode") String
postalCode, Pageable pageable);

}
```

Request:

http://localhost:8080/eventmanagement-api/venues/search/findByPostalCo
de?postalCode=78750

Given an email Id find the participant

```
        public interface ParticipantRepository extends
PagingAndSortingRepository<Participant, Long> {

    List<Participant> findByIdAndEmail(@Param("id") Long id,
@Param("email") String email);

    //or

        //Page<Participant> findByIdAndEmail(@Param("id") Long id,
@Param("email") String email, Pageable pageable);
```

}

Request:

http://localhost:8080/eventmanagement-api/participants/search/findByIdAndEmail?id=2&email=johnf@gmail.com

Assignment 5 : Custom Controller Methods Assignment

```
        @RepositoryRestController

        @RequestMapping("/events")

        public class CheckOutController {

   @Autowired

   private ParticipantRepository participantRepository;

   @PostMapping("/checkout/{id}")

   public ResponseEntity<PersistentEntityResource>
checkout(@PathVariable Long id,PersistentEntityResourceAssembler
assembler) {

   Participant participant = participantRepository.findById(id).get();

   if (participant != null) {

   if (!participant.getCheckedIn()) {
```

```
   throw new NotCheckedInException();

}

        participant.setCheckedIn(false);

    participantRepository.save(participant);

        }

    return ResponseEntity.ok(assembler.toFullResource(participant));

}

}
```

Assignment 5 : Projections Assignment

```
        @Projection(name = "partial", types = { Event.class })

        public interface PartialParticipantProjection {

    String getName();

    Boolean getCheckedIn();

}

@RepositoryRestResource(excerptProjection=PartialParticipantProjection.
class)

        public interface ParticipantRepository extends
PagingAndSortingRepository<Participant, Long> {

        Page<Participant> findByEmail(@Param("email") String email,
Pageable pageable);

}

        http://localhost:8080/eventmanagement-api/events/8/participants

{

   "_embedded": {

   "participants": [

{

   "name": "John ",

   "checkedIn": false,

   "_links": {

   "self": {

   "href": "http://localhost:8080/eventmanagement-api/participants/3"

},

   "participant": {
    "href": "http://localhost:8080/eventmanagement-api/participants/3"

},

    "event": {

    "href": "http://localhost:8080/eventmanagement-api/participants/3/event"

}

}

}

]

},

        "_links": {

    "self": {

        "href":
"http://localhost:8080/eventmanagement-api/events/8/participants"

}

}

}
```