Trainee - Notepad

File Edit Format View Help

```
package com.infy.entity;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;

@Entity
public class Trainee {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
	private Integer traineeId;
	private String traineeName;
	@ManyToOne(cascade=CascadeType.ALL)
	@JoinColumn(name="classroomid")
	private Classroom classroom;
	public Integer getTraineeId() {
		return traineeId;
	}
	public void setTraineeId(Integer traineeId) {
		this.traineeId = traineeId;
	}
	public String getTraineeName() {
		return traineeName;
	}
	public void setTraineeName(String traineeName) {
		this.traineeName = traineeName;
	}
	public Classroom getClassroom() {
		return classroom;
	}
	public void setClassroom(Classroom classroom) {
		this.classroom = classroom;
```

Classroom - Notepad

File Edit Format View Help

```
package com.infy.entity;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Classroom {
	@Id
	private String classroomId;
	private String buildingName;
	private Integer capacity;
	private Integer availableCapacity;
	public String getClassroomId() {
		return classroomId;
	}
	public void setClassroomId(String classroomId) {
		this.classroomId = classroomId;
	}
	public String getBuildingName() {
		return buildingName;
	}
	public void setBuildingName(String buildingName) {
		this.buildingName = buildingName;
	}
	public Integer getCapacity() {
		return capacity;
	}
	public void setCapacity(Integer capacity) {
		this.capacity = capacity;
	}
	public Integer getAvailableCapacity() {
		return availableCapacity;
	}
	public void setAvailableCapacity(Integer availableCapacity) {
		this.availableCapacity = availableCapacity;
	}
```

ClassroomAllocationServiceImpl - Notepad

File Edit Format View Help

```
@Service(value="classroomAllocationService")
@Transactional
public class ClassroomAllocationServiceImpl implements ClassroomAllocationService {

	@Autowired
	private ClassroomRepository classroomRepository;

	@Autowired
	private TraineeRepository traineeRepository;

	/**
	  this method  calls getTrainee method of ClassroomAllocationDAOImpl class.

	  @param -  Integer traineeId
	  @return - Trainee object returned from getTrainee method of ClassroomAllocationDAOImpl class
	  @throws "Service.TRAINEE_NOT_FOUND" exception If trainee object returned from getTrainee method
	 */
	public TraineeDTO getTrainee(Integer traineeId) throws ITrainingException {
		Optional<Trainee> optional=traineeRepository.findById(traineeId);
		Trainee t=optional.orElseThrow(()->new ITrainingException("Service.TRAINEE_NOT_FOUND"));
		TraineeDTO tr=new TraineeDTO();
		tr.setTraineeId(t.getTraineeId());
		tr.setTraineeName(t.getTraineeName());
		ClassroomDTO c=new ClassroomDTO();
		c.setAvailableCapacity(t.getClassroom().getAvailableCapacity());
		c.setBuildingName(t.getClassroom().getBuildingName());
		c.setCapacity(t.getClassroom().getAvailableCapacity());
		c.setClassroomId(t.getClassroom().getClassroomId());
		tr.setClassroom(c);
		return tr;
	}

	/**
	  this method will call the addTrainee method of ClassroomAllocationDAOImpl class.
	  @param - Trainee object
	*/
```

Type here to search

omAllocationServiceImpl - Notepad

Format View Help

```
 @param - Trainee object
 @return - Integer traineeId returned from addTrainee method of ClassroomAllocationDAOImpl class.
*/
public Integer addTrainee(TraineeDTO trainee) throws ITrainingException {
        Trainee tr=new Trainee();
        tr.setTraineeName(trainee.getTraineeName());
        Classroom c=new Classroom();
        c.setAvailableCapacity(trainee.getClassroom().getAvailableCapacity());
        c.setBuildingName(trainee.getClassroom().getBuildingName());
        c.setCapacity(trainee.getClassroom().getAvailableCapacity());
        c.setClassroomId(trainee.getClassroom().getClassroomId());
        tr.setClassroom(c);
        //traineeRepository.save(tr);
        return tr.getTraineeId();
}

/**
  this method will call allocateClassroom method of ClassroomAllocationDAOImpl class, which in turn wi
  @param -  Integer traineeId, String classRoomId
  @return - Integer value returned from llocateClassroom method of ClassroomAllocationDAOImpl class
  @throws -exception as below if the return from llocateClassroom method of ClassroomAllocationDAOImpl
    0 - throw "Service.TRAINEE_NOT_FOUND" exception
   -1 - throw "Service.CLASSROOM_NOT_FOUND" exception
   -2 - throw "Service.CLASSROOM_FULL" exception.
 */
public Integer allocateClassroom(Integer traineeId, String classRoomId) throws ITrainingException {
        Optional<Trainee> optional=traineeRepository.findById(traineeId);
        Trainee t=optional.orElseThrow(()->new ITrainingException("Service.TRAINEE_NOT_FOUND"));
        Optional<Classroom> optional1=classroomRepository.findById(classRoomId);
        Classroom c=optional1.orElseThrow(()->new ITrainingException("Service.CLASSROOM_NOT_FOUND"));
        if(c.getAvailableCapacity()==0)
        {
                throw new ITrainingException("Service.CLASSROOM_FULL");
        }
        c.setAvailableCapacity(c.getAvailableCapacity()-1);
        t.setClassroom(c);
```

Type here to search

ClassroomAllocationServiceImpl - Notepad

Edit Format View Help

```
public Integer allocateClassroom(Integer traineeId, String classRoomId) throws ITrainingException {
        Optional<Trainee> optional=traineeRepository.findById(traineeId);
        Trainee t=optional.orElseThrow(()->new ITrainingException("Service.TRAINEE_NOT_FOUND"));
        Optional<Classroom> optional1=classroomRepository.findById(classRoomId);
        Classroom c=optional1.orElseThrow(()->new ITrainingException("Service.CLASSROOM_NOT_FOUND"));
        if(c.getAvailableCapacity()==0)
        {
                throw new ITrainingException("Service.CLASSROOM_FULL");
        }
        c.setAvailableCapacity(c.getAvailableCapacity()-1);
        t.setClassroom(c);
        return 1;
}
/**
  this method will call deleteTrainee method of ClassroomAllocationDAOImpl class which will return a string value
  @param -  Integer traineeId
  @return - String   value that was received from deleteTrainee method of ClassroomAllocationDAOImpl class
  @throws -Service.TRAINEE_NOT_FOUND exception if value returned by deleteTrainee method of ClassroomAllocationDA
 */
public String deleteTrainee(Integer traineeId) throws ITrainingException {
        Optional<Trainee> optional=traineeRepository.findById(traineeId);
        Trainee t=optional.orElseThrow(()->new ITrainingException("Service.TRAINEE_NOT_FOUND"));
        if(t.getClassroom()!=null)
        {
                Classroom c=classroomRepository.findById(t.getClassroom().getClassroomId()).get();
                c.setAvailableCapacity(c.getAvailableCapacity()+1);
                t.setClassroom(null);
        }

        traineeRepository.delete(t);
        return t.getTraineeName();
}

}
```

Ln 1, Co