Internet Application Design and Implementation – 2019/2020

MidTerm Test November 2, 2020

Notes: The test is closed book and has a duration of 1h30m. You may bring 2 handwriten A4 pages. There are 10 multiple choice answer questions and 3 open answer questions that should be answered in the provided answer sheets. Correct answers in multiple choice questions add 1 point, the first 3 wrong answers have a penalty of 1/8 of the value of a correct answer, the following wrong answers have a penalty of 1/4 of the value of a correct answer. Open answers are awarded 3, 4 and 3 points respectively. Do not unstaple the questions. You may use a pencil.

Version: RA

Name:_____Number:_____

Part 1

Q-1 [3 pts] Describe the ER model of the resulting database after mapping the following JPA Entities.

```
@Entity
data class Hotel(
    @Id
    @GeneratedValue
    var id: Long,
    var name:String,
    @OneToMany
    var rooms:List<Room>
)
```

```
@Entity
data class Room(
    @Id
    @GeneratedValue
    var id: Long,
    var number:Int,
    var name:String,
    @ManyToOne
    var hotel:Hotel
)
```

Q-2 [4 pts] Complete the following test to a service that provides data about a hotel room. Consider the skeleton of the test below where `rooms` is a reference to a repository for the data class `Room`. Select the fragment that correctly completes the test (without mocking). Make sure that the request is successful, that the result exists and that the name of the room is as expected.

```
@Test @WithMockUser(username = "user1", roles = "REGISTERED")
fun ‘testing the retrieval of one Room‘() {
  val roomId = rooms.searchByName(BIG_SUITE_NAME)[0].id
  mockMvc.perform(get("/rooms/" + roomId ))

  ...

}
```

Q-3 [3 pts] Consider the data model presented in **Q-1** extended with the entity `HotelVisit` defined below:

```
@Entity
data class HotelVisit(
    @Id
    @GeneratedValue
    var id: Long,
    var name:String,
    @OneToMany
    var rooms:List<Room>
)
```

Define a repository for hotel visits implement a custom JPQL query that shows the rooms that were visited between two given dates.

Name:_____ Number:_____

Part 2

System Description: Consider a system for the management of expense reports of employees of a company. The users of the system are employees of the company, and they are organized in departments. The system allows for employees of a company to create new expense reports for their activities within the context of a department. An expense report consists of a description, a date, and an amount. The statuses of an expense report can be ”created”, ”submitted”, ”approved”, and ”paid”. Employees can always see their expense reports, but can only edit their reports while the status has value ”created”. Heads of department can only approve expense reports when they on they have the status ”submitted”. Each expense report must be approved by the head of the department where the employee belongs to. The main resource here is an expense report.

The system described above is the development context for all questions below.

Q-4 [4 pts] Define a RESTful interface, Level 2 in the Richardson maturity scale, that defines the operations on the available expense reports to both employees and heads of departments. List all the endpoints by means of a Kotlin interface using data classes for DTO objects. No swagger annotations are needed.

Q-5 [3 pts] Define the JPA (data) classes for the system described above and relate them correctly using JPA annotations.

Q-6 [3 pts] Define two security policies, annotations and corresponding services, that regulates the access for reading, and also for changing the status of the main resource of the scenario above.