EA Project Description – March 2023

Goal

The goal of this project is to give you hands-on, practical experience with building a RESTful system of microservices using Spring, Spring MVC, Spring Boot, Spring Data JPA and Spring Cloud Services.

Working as a team

You are required to work with your team to create a project. But you should be aware that everyone still must give their own presentation. There are no “team” presentations, although I do ask that members of the same team present consecutively (no gaps / other people in between). You need to work together and share responsibilities and workload.

Badge & Membership System

We are trying to create a RESTful application (you only need to write the backend part of it; no UI is necessary) for a Badge & Membership System. Imagine a university with thousands of students, staff, and faculty. Each member of this community has a “badge” that identifies that member uniquely. Each member can also have many “memberships” that allow the member to make use of different areas of university. For example, a member can have a membership to the library or the gym and his/her badge allows the person to access those areas.

We can expand this concept of a membership to include a broader range of things. Basically, anything that needs access control (dining hall, meditation hall, residence hall, etc.) could also be a membership plan and can be controlled by the badge. When you do that, then you realize that there are fundamentally two types of memberships: limited and unlimited.

- Gym membership is unlimited. Meaning that members can come and go to the gym as many times as they want.
- Meditation hall membership is also unlimited for the same reason.
- Dining hall membership is limited. Each member is only allowed a fixed number of meals per week. Even students with an “all-meals” plan, can use the dining hall a maximum of 20 times a week (Sundays are two meals only). Faculty and staff may have only 20, 30 or 50 meals/month.

Another issue to be considered is the fact that each member can have multiple badges. This happens when a member loses his/her badge. University issues a new badge and the old one should be marked inactive. The information of the old badge and it’s unique ID will be kept in the system for future reference and historical records.

Also, in this system you need to audit all the “transactions”, which is basically every time a badge is swiped.

EA Project Description – March 2023

Step 1 – Requirements (Domain Driven Design)

Create a UML class diagram of the domain based on the concepts below:

- Use a surrogate id for all entities.
- A member has first and last names, email address and multiple roles (student, staff, faculty, …)
- A membership has a start date and end date and belongs to a plan
- A plan has a name, description, list of allowed roles
- A transaction has a datetime, is linked to a member, a membership and a location
- Every location has a name and description, capacity, and a location "type".
- Location type can be one of the following: DINING_HALL, MEDITATION_HALL, FLYING_HALL, CLASSROOM, GYMNASIUM, DORMITORY
- Every location also has a list of "time-slots" which are slots during which the location is open
- Each time-slot has a start time, an end time "for every day of the week".
- Member's badges only work during time-slots and otherwise are rejected.

Step 2 - Architectural Analysis and Design

Sit with your team and agree on a high-level architecture for your application. Your decision should include your choice of technologies. How do you secure the application? What methodology do you use for software development? Do a proof of concept (POC).

Step 4 – Divide and Conquer

Carefully read the requirements and divide the tasks (use-cases) among team members. Each team member is responsible for designing, developing, and testing his/her use-case.

Step 5 – Integration Testing

At least once a day sit together and integrate your code and test together and iterate (correct mistakes and refactor your design and development).

Extra Credit

Here are some ideas for extra credit:

- Deploy your application to the cloud
- See if you can us Spring Security using JWT or the Spring Authorization Server
- Create an automated pipeline for CI/CD (continuous integration/continuous delivery)

EA Project Description – March 2023

Simple Use-cases

1. Provide CRUD services over “Member”
2. Provide CRUD services over “Badge”
3. Provide CRUD services over “Plan”
4. Provide CRUD services over “Membership” (the link between member and plan)
5. Provide CRUD services over “Location”
6. Provide CRUD services over “Transaction”
7. For a member, return the list of all transactions (GET /members/{memberId}/transactions)
8. For a member, return the list of all Plans (GET /members/{memberId}/plans)
9. For a member, return the list of all Badges (GET /members/{memberId}/badges)
10. For a member, return the list of all Memberships (GET /members/{memberId}/memberships)
11. For a plan, return the list of Locations (GET /plans/{planId}/locations)

Notes:

- There are 4 main actors (user roles). Admin, Student, Staff and Faculty.
- Transaction is a link between Badge, Location and Plan (you can do it other ways too, but only if it answers to use-cases 6 and onwards

Complex Use-cases

1. Provide a service for login. I suggest: (POST /authentication) and send username, password. It should return a Member object or return a 404 NotFound
2. For a member, using memberId, then return the list of badges (#9 above), then filter based on the only active badge and then display that badge as the active badge
3. For a door checker, after the checker logs in
    a. return a list of memberships using #10 above.
    b. Then filter for Memberships of type “CHECKER” (so, now you have three types of memberships: LIMITED, UNLIMITED and CHECKER). The last type allows the member to check other members for that plan.
    c. Now that you have a list of memberships with type CHECKER, you will present the checker with the list of plans and let him/her choose which plan
    d. Then you call #11 above and bring the list of locations for that plan and let the checker choose which location
    e. At this point the checker is ready to scan badges and allow or decline users
    f. Successful badge scan results in a transaction (You have Plan, Location and Member). Just put them together and add a timestamp!
    g. Maybe you can also track unsuccessful badge swipe attempts. Then you will need to add one more column/attribute to your transaction. Transaction Type = ALLOWED, DECLINED.

EA Project Description – March 2023

Presentation Delivery

I will schedule your presentations from 9:30-12:30 and 1:30 – 4:30 on Thursday. You will have to present individually. Each group will have one hour, which means each member of the group gets to present for about 5 minutes and I get to ask questions for a couple of more minutes.

I will evaluate you based on the following factors:

1. Clarity of Speech – Your presentation should be coherent and understandable. It is ok if you have an accent. It is ok if your English is not as fluent as a native speaker. However, it is ***not*** ok if you talk too fast! Enunciate and speak clearly.
2. Knowledge of Your Application – You are expected to be knowledgeable about the overall design of your app. All layers and all use-cases, not just yours.
3. Ability to Answer Questions – You need to be able to answer questions about the design and coding of your app. It always shows when your team members have done all the work and you have been mostly observing. Try to be an active participant and you will get full grade for this category.
4. Working Demo – Your app needs to work (obviously!). So, if you succeed to show me working features, you will get maximum grade.
5. Code quality – You need to write readable, good quality code. Hint: use "Sonar Lint" or similar plug-ins for your IDE to check the quality of your code.
6. Extra credit – you can get up to 10% of extra credit based on how well you design your application and how well you integrate it.