Java Backend – Advanced

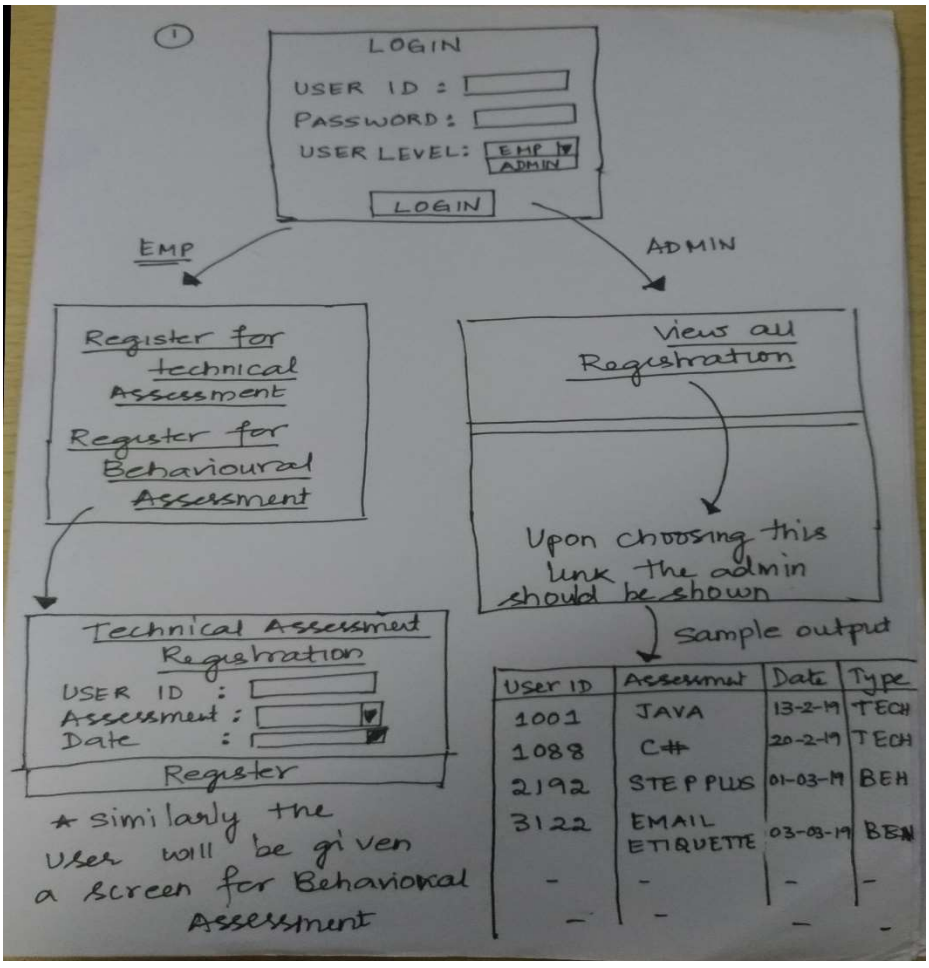
Assessment Management – Case Study

Assessment Management system helps the Employees and the Assessment Team Administrator to manage assessment related tasks.

Broad Functionalities:

a. Employees, once successfully logged in can register for either “Technical” or “Behavioral” Assessments.
b. Assessment Team Admininstrator, once successfully logged in can view a tabular list of all the registrations done by the employees.

Flow and Sample screens:



Expectation is that this application will grow exponentially in time and may be required to be managed/scaled independently in future. Hence, the requirement is to architect and implement it as Java based Microservices. Use Spring Boot framework.

Apart from implementing the Microservices based on the functionalities discussed above,

a. Use Eureka as your Registry and Discovery service so that Microservices can find/discover each other and interact with each other.
b. Implement Client-side load balancer wherever applicable (Spring Cloud Load Balancer)
c. Identify some important methods that can lead to cascading failures and guard them by implementing Software Circuit Breaker/Fault Tolerance mechanism (Spring Cloud Circuit Breaker with Resilience4J and optionally can also setup a dashboard like Prometheus)
d. Use Thymeleaf based dynamic views wherever needed (do not use JSPs)
e. Create an external configuration server (file/github) and applications should get the configuration information from the configuration server.
f. Enable distributed log tracing with micrometer brave and configure the zipkin dashboard to monitor and trace the logs.

Assumptions:

a. The users are already registered. So, you can use a static list of registered Employees/Admins for this application.
b. The choice of how many number of Microservices you want to implement for the given scenario is left to you. Keep in mind – Nano services are anti-patterns.
   Hint: You can have 3 microservices:
   1. Login Microservice
   2. Employee Microservice
   3. Admin Microservice
c. You can use a static list of "Technical" or "Behavioral" assessments for registration.
   E.g.: Technical -> "Java", "C#", "PHP", "PERL" etc.
   Behavioral -> "Step Plus", "Email Etiquette", "Learning Agility" etc.
d. Implement appropriate validations to ensure that only upon providing correct userId and password and user type, the user will be logged on successfully and further allowed to proceed with the application. Otherwise appropriate Error messages are to be displayed.
e. Screen designs used in the above image is just as a suggestion. You could design and implement the user-interface as per your choice.

Other Details:

a. You can use any IDE of your choice (STS/IntelliJ/Eclipse) etc.,
b. You can use any in-memory database like H2/HSQL as your data store.
c. Try to use good programming constructs wherever applicable.