Backend Engineer assessment

- Assessment estimated time, two days.
- Make sure to check all assessment, including below table.
- Note all assessment will be discussed at the interview, as a why we do so, pros and cons beside adding more use cases to the system as we go through the interview.
- Check points marked as (Should be done completely) to consider the assessment to be successful, other points will give you bonus during the interview.
- Interviewer, Mahmoud Abel Hakam

Topic	Comment
Problem solving	All test cases should pass
Database query	Query should return correct result
System design	We will address these topics: Database design, Clean code, API restful design, SOLID, Docker
Interview rules	Make sure to read them

Problem solving (Should be done completely)

- Given a string, we need to reverse substrings between each pair of parentheses, then printing the original with reversed substrings. ○ Test cases:

  Input: abd(jnb)asdf
  Output: abd(bnj)asdf

  Input: abdjnbasdf
  Output: abdjnbasdf

  Input: dd(df)a(ghhh)
  Output: dd(fd)a(hhhg)

  ○ Constraints:

1. 1 <= s.length <= 2000
2. String only contains lower case English characters and parentheses.
3. It is guaranteed that all parentheses are balanced.

Database query (Should be done completely)

Given the below tables:

```
User table:
user_id  username
1        John Doe
2        Jane Don
         3        Alice Jones
         4        Lisa Romero

Training_details table:
user_training_id  user_id  training_id  training_date 1
1        1            "2015-08-02"
2                 2        1            "2015-08-03"
3                 3        2            "2015-08-02"
4                 4        2            "2015-08-04"
5                 2        2            "2015-08-03"
6                 1        1            "2015-08-02"
7                 3        2            "2015-08-04"
8                 4        3            "2015-08-03"
9                 1        4            "2015-08-03"
10                3        1            "2015-08-02"
11                4        2            "2015-08-04"
12                3        2            "2015-08-02"
```

```
13                  1         1             "2015-08-02" 14
                    4         3             "2015-08-03"
```

Query to get the list of users who took the training lesson more than once at the same day, grouped by user and training lesson, each ordered from the most recent lesson date to oldest date.

Output:

user_id	username	training_id	training_date	count
4	Lisa Romero	2	2015-08-04	2
4	Lisa Romero	3	2015-08-03	2
1	John Doe	1	2015-08-02	3
3	Alice Jones	2	2015-08-02	2

System design

Consider design a system for managing upload/download files, with the below criteria

1. The application should be running on docker, Dockerfile or dockercompse.yml should be attached with the project.
2. Database should be provided as a docker file, and application should connect to it while starting as a docker container.
3. This application should be uploaded to GitHub. (Should be done completely)

System should be managed as a tree data structure, space as the parent then folders and files as children, for example:

 Documents ❼ (Space) ○
Folder-1 ❼ (Folder)
 File789.xls ❼ (File)
 File-98.docx ❼ (File) ○
Folder-2 ❼ (Folder)
 File-ss-795.msg ❼ (File)
○ File-124.pdf ❼ (File) ○ File-457.JPG ❼ (File)

Consider design database using JPA entities as below: (Should be done completely)

- Item table with type column (Space – Folder - File), to hold item meta data, expected columns (id, type, name, permission_group_id)
- Files table to hold binary files, expected columns (id, binary, item_id)
- Permission groups table, expected columns (id, group_name), one group can be assigned to many items.
- Permissions table, expected columns (id, user_email, permission_level, group_id)

Consider API implementation as below: (Only clean coding, API restful design, design patterns)

1. API to create space called "stc-assessments", and assign a permission group like admin group, containing a user with VIEW access and another one with EDIT access.
2. API to create folder under "stc-assessments" space, called "backend", making sure that the user has EDIT access on this space, blocking user that has VIEW access.
3. API to create file under "backend" folder, called "assessment.pdf", making sure that the user has EDIT access on this folder, blocking user that has VIEW access.
4. API to view file metadata, making sure that the user has access on this file, using native SQL query. (Making this as a graphql query will give you bonus point)
5. API to download file as a binary, making sure that the user has access on this file (Bonus API)

Technology: Spring boot – PostgreSQL - Docker

We will judge according to using spring boot, clean code, API restful design, database design, docker.

Interview rules: (In case of assessment passed)

1. Do not waste your time reviewing online interview questions, it will be use cases interview.
2. The interview will go around the assessment, why you do such and such.
3. You can search on google during the interview.