1. **REST API Design**You are tasked with designing an API for a simple task management system. The system should allow users to create tasks, mark them as completed, and retrieve a list of tasks.

Define the API endpoints, methods, and request/response structures for the following operations:

1. Create a new task.
2. Mark a task as completed.
3. Retrieve a list of tasks.
4. **Authentication**Explain how you would implement authentication for your API. Consider security best practices, and mention any tokens or mechanisms you would use.

**Jawaban:**

**Pada TaskApplication saya akan menggunakan spring-boot-starter-security dependency dan membuat API POST /login dengan request username dan password. API itu akan menggenerate token (JWT) jika username dan password user benar. Lalu terbentuklah token yang berisikan username dan role dari user. Pada API task, akan disematkan @PreAuthorize annotation untuk menentukan role access. Apabila role yang ada pada bearer token sesuai, user dapat mengakses API tersebut, jika tidak sesuai maka akan ditampilkan error unauthorized.**

1. **Database Design**

Design a database schema for the task management system described in task number 1. Include tables for tasks and any additional information you find relevant. Specify primary and foreign keys.

**Jawaban:**

Table task:

* Id sebagai primary key berupa varchar
* Employee\_id sebagai foreign key berupa varchar
* task\_description berupa text sebagai wadah untuk deskripsi task yang akan dikerjakan
* status berupa varchar yang berisikan CREATED, INPROCESS, dan COMPLETED, ini berfungsi untuk memberikan kondisi apabila task masih CREATED, maka tidak bisa diselesaikan (diubah menjadi COMPLETED) sedangkan INPROCESS adalah status yang menandakan task sedang dikerjakan, status INPROCESS dapat diselesaikan.
* created\_at berupa Date yang merupakan tanggal dibuatnya task
* created\_by berupa varchar yang merupakan nama pembuat task
* updated\_at berupa Date yang merupakan tanggal task tersebut diedit
* updated\_by berupa varchar yang merupakan nama pengedit task

Table employee:

* Id sebagai primary key berupa varchar
* name berupa varchar
* role\_id foreign key berupa varchar
* created\_at berupa date
* created\_by berupa varchar
* updated\_at berupa date
* updated\_by berupa varchar

Table role:

* id sebagai primary key berupa varchar
* role\_name berupa varchar
* created\_at berupa date
* created\_by berupa varchar
* updated\_at berupa date
* updated\_by berupa varchar

Dengan relation diatas, kita dapat mengetahui siapa yang mengerjakan task karena ada employee\_id yang ada pada table task, dan untuk table role itu sendiri untuk mengetahui role dari employee itu apa.

1. **SQL Queries**Write an SQL query to retrieve all incomplete tasks from the database.

**Jawaban:**

Dengan code saya diatas, kita bisa execute query yang berupa:

SELECT \* FROM task WHERE STATUS != ‘COMPLETED’

1. **Problem-Solving**Write a function that takes a list of integers as input and returns the sum of all even numbers in the list.

**Jawaban:**

private Integer getTotalSum(List<Integer> numberList){  
 return numberList.stream().filter(x -> x % 2 == 0).mapTo(Integer::intValue).sum;  
}

1. **Error-Handling**

Imagine you have a function that connects to an external service. Implement error handling for potential issues such as network errors or service unavailability.

**Jawaban:**

Untuk error handling saya akan gunakan try-catch:

try{

//Hit external API

}catch(IOException ie){

log.error(ie.getMessage);

throw new IOException(“Failed to connect external service”);

}catch(Exception e){

log.error(e.getMessage);

throw new Exception(“Failed to connect external service”);

}

\*Use programming languages as required by the company.