Laboratório 3 – Aula 04

Diagrama de Sequência

Agora será necessário fazer um diagrama de sequência para cada um dos endpoints que você criou na sua aplicação. É sempre importante verificar se o diagrama esta 100% compatível com o seu código fonte, uma vez que ele demonstra o aspecto dinâmico da sua aplicação.

O diagrama deverá ser criado utilizando a linguagem PlantUML e deverá ter os arquivos com a extensão PUML salvo na pasta **docs/diagramas_sequencia**. Para cada um dos endpoints, deveremos criar então um arquivo com a extensão puml.

O exemplo a seguir apresenta o **Diagrama de Sequência referente ao CDU001 - Listar todas as Tarefas.**

O diagrama pode ser gerado:

- instalando um plugin PlantUML para o VSCode
- diretamente no site oficial https://www.plantuml.com/plantum
- Em editores gratuitos disponíveis online, como por exemplo: https://www.planttext.com

O material de apoio se encontra no endereço oficial: https://plantuml.com/guide.

Código Exemplo

```
@startuml
title Diagrama de Sequencia referente ao CDU001 - Listar todas as Tarefas
actor Usuário
Boundary Frontend
Boundary SpringBoot
Control taskController
Entity Task
Participant taskService
Participant taskRepository
Participant pagedResourcesAssembler
Boundary TypedQuery
Participant Hibernate
Database TodoList
activate SpringBoot
Usuário -> Frontend: Listar todas as Tarefas
activate Usuário
activate Frontend
activate TodoList
Frontend -> SpringBoot: GET api/tasks/
group Roteiro01Application [Spring Boot Java Application]
SpringBoot -> taskController: listAll(TaskDto, Pageable, PersistentEntityResourceAssembler)
activate taskController
taskController -> taskService: listAll(pageable)
activate taskService
taskService -> taskRepository: findAll(pageable)
```

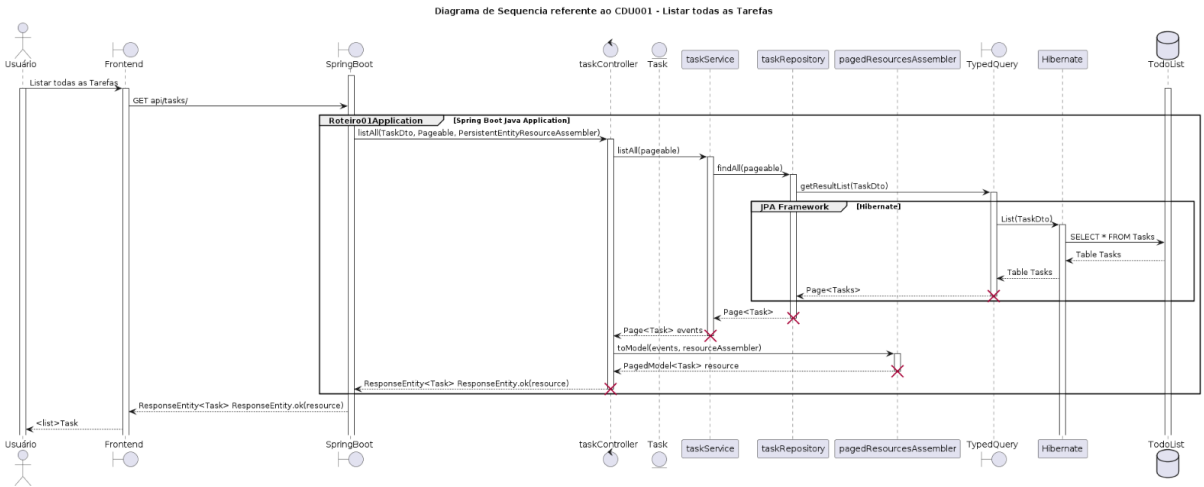
```
activate taskRepository
taskRepository -> TypedQuery: getResultList(TaskDto)
group JPA Framework[Hibernate]
Activate TypedQuery
TypedQuery -> Hibernate: List(TaskDto)
Activate Hibernate
Hibernate -> TodoList: SELECT * FROM Tasks
TodoList --> Hibernate: Table Tasks
Hibernate --> TypedQuery: Table Tasks
TypedQuery --> taskRepository: Page<Tasks>
destroy TypedQuery
end
taskRepository --> taskService: Page<Task>
destroy taskRepository
taskService --> taskController: Page<Task> events
destroy taskService
taskController -> pagedResourcesAssembler: toModel(events, resourceAssembler)
activate pagedResourcesAssembler
pagedResourcesAssembler --> taskController: PagedModel<Task> resource
destroy pagedResourcesAssembler
taskController --> SpringBoot: ResponseEntity<Task> ResponseEntity.ok(resource)
destroy taskController

end

SpringBoot --> Frontend : ResponseEntity<Task> ResponseEntity.ok(resource)
Frontend --> Usuário: <list>Task

@enduml
```

Diagrama gerado pelo código acima



[Gerar diagrama novamente]

Exemplo utilizando o PlantText