

ANÁLISE DE REQUISITOS.com.br

ESPECIFICAÇÃO DE CASOS DE USO

DOCUMENTO X-0001

LUCAS MACHADO DE OLIVEIRA ANDRADE

ÚLTIMA ATUALIZAÇÃO: 28/04/2024

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: "[Como especificar casos de uso em 5 passos](#)", "[Como fazer um diagrama de casos de uso](#)" e "[Como documentar requisitos de software](#)".

HISTÓRICO DE REVISÕES DO DOCUMENTO

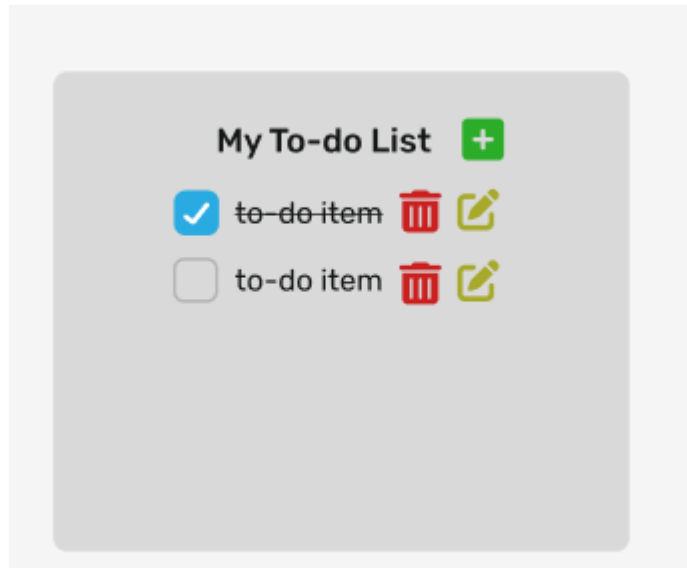
DATA	VERSÃO	DESCRIÇÃO DA ALTERAÇÃO	AUTOR
06/02/2024	1	CRIAÇÃO DESTE DOCUMENTO	LUCAS MACHADO DE OLIVEIRA ANDRADE
13/02/2024	2	CRIAÇÃO DO CASO DE USO DO TO-DO LIST	LUCAS MACHADO DE OLIVEIRA ANDRADE
13/02/2024	3	CRIAÇÃO DO CASO DE USO DESCRITIVO DO TO-DO LIST	LUCAS MACHADO DE OLIVEIRA ANDRADE
17/03/2024	4	CRIAÇÃO DO PROTÓTIPO DE BAIXA FIDELIDADE DO TO-DO-LIST	LUCAS MACHADO DE OLIVEIRA ANDRADE
17/03/2024	5	CONCLUSÃO DO DOCUMENTO	LUCAS MACHADO DE OLIVEIRA ANDRADE
28/04/2024	6	ALTERAÇÃO CASO DE USO	

IDENTIFICAÇÃO DOS ENVOLVIDOS

PAPEL	NOME	EMAIL
ANALISTA DE REQUISITOS	Lucas Machado de Oliveira Andrade	lmoandrade@sga.pucminas.br
PATROCINADOR	Cristiano de Macêdo Neto	51897@sga.pucminas.br

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

PROTÓTIPO



TL001: tela inicial



TL002: cadastro de padrão ou com Tipo Livre

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Frame 6

Descrição da Tarefa

Tarefa...

Tipo da Tarefa

▼ Data

Prazo

📅

Prioridade

▼ Alta, Média, Baixa...

SALVAR **CANCELAR**

TL003: cadastro de padrão ou com Tipo Data

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: "[Como especificar casos de uso em 5 passos](#)", "[Como fazer um diagrama de casos de uso](#)" e "[Como documentar requisitos de software](#)".

Frame 8

Descrição da Tarefa

Tarefa...

Tipo da Tarefa

▼ Prazo

Prazo

Dias...

Prioridade

▼ Alta, Média, Baixa...

SALVAR **CANCELAR**

TL004: cadastro de padrão ou com Tipo Prazo

My To-do List +

✓ to-do-item 🗑️ ✎️

Deseja mesmo excluir
a tarefa to-do-item?

SALVAR **CANCELAR**

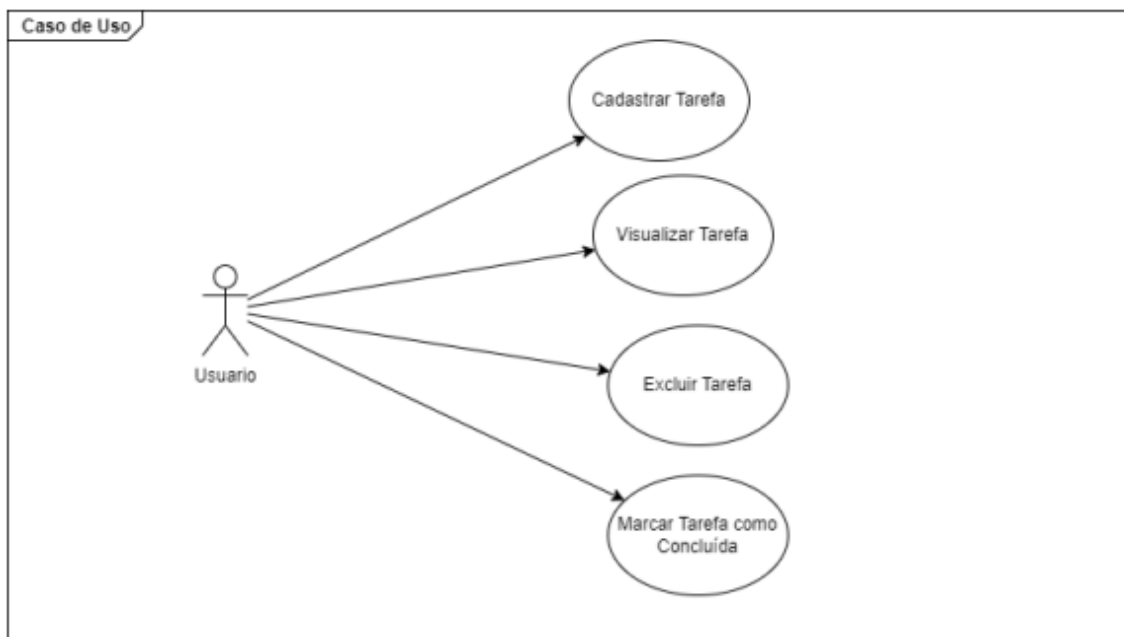
TL005: exclusão de tarefa

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: "[Como especificar casos de uso em 5 passos](#)", "[Como fazer um diagrama de casos de uso](#)" e "[Como documentar requisitos de software](#)".

DESCRIÇÃO DO CASO E USO

Possibilita a manutenção (incluir, consultar, alterar e excluir) de informações relacionadas a uma lista de tarefas

DIAGRAMAS DOS CASOS DE USO



ATORES

- *Usuário: Pessoa com acesso que utilizará as funcionalidades da lista de tarefas em sua totalidade*

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

CASOS DE USO DESCRITIVO

- **VISUALIZAR TAREFA**

FLUXO BÁSICO

FB01 VISUALIZAR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Usuário acessa o sistema	-	-	-	TL001
2	O sistema carrega tela inicial com botão para cadastrar novas tarefas	FB02	-	-	TL001
3	O sistema carrega dados de tarefas já existentes	FA01	RGN001	-	TL001

FLUXO ALTERNATIVO

FA01 NÃO HÁ TAREFAS CADASTRADAS					
ID	Passo	Fluxo	Regras	Msg	Tela
4	Sistema exibe mensagem na tela	FB01	-	Não existem tarefas cadastradas	TL001

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

REGRAS

RGN01 REGRA DE FLUXO DE VISUALIZAR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Sistema envia para fluxo alternativo FA01 caso não existam tarefas cadastradas	FB01 e FA01	-	Não existem tarefas cadastradas	TL001

● CADASTRAR TAREFA

FLUXO BÁSICO

FB02 CADASTRAR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	O usuário acessa clica no botão de criar tarefa	FB01	-	-	TL001
2	O sistema carrega o formulário	-	-	-	TL002
3	O usuário preenche os dados do formulário	FA03	-	-	TL002
3.1	Caso o usuário coloque o Tipo como Data, exibe um campo de data para ser preenchido	FA03	-	-	TL003
3.2	Caso o usuário coloque o Tipo como Prazo, exibe um campo numérico para ser preenchido	FA03	-	-	TL004
4	O usuário clica no botão de salvar	FA02	RGN02	-	TL002

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

5	Sistema atualiza os dados e retorna para tela principal	FB01	-	Salvo com sucesso	TL001

FLUXO ALTERNATIVO

FA02 FORMULÁRIO NÃO PREENCHIDO CORRETAMENTE					
ID	Passo	Fluxo	Regras	Msg	Tela
6	O Sistema exibe em vermelho os dados preenchidos erroneamente	FB02	-	Favor, preencher dados obrigatórios	TL002, TL003, TL004

FA03 USUÁRIO DECIDE NÃO CADASTRAR A TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
6	Usuário clica no botão de cancelar	FB02	-	-	TL002, TL003, TL004
7	Sistema retorna para a tela principal	FB02	-	-	TL001

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

REGRAS

RGN02 REGRA DE FLUXO ALTERNATIVO					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Sistema valida se usuário preencheu o campo "Descrição"	FB02 e FA02	-	-	TL002, TL003, TL004

- **EXCLUI TAREFA**

PRÉ CONDIÇÃO

PR01 EXCLUIR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Deve haver ao menos uma tarefa já cadastrada	FP03	-	-	TL001

FLUXO BÁSICO

FP03 EXCLUIR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	O usuário acessa clica no botão de excluir tarefa	FB01	-	-	TL001

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

2	O sistema carrega um modal de confirmação	-	-	-	TL005
3	O usuário clica no botão de confirmar	FA04	-	-	TL005
4	Sistema atualiza os dados e retorna para tela principal	FB01	-	Excluído com sucesso	TL001

FLUXO ALTERNATIVO

FA04 USUÁRIO DECIDE NÃO EXCLUIR A TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
4	Usuário clica no botão de cancelar	FB03	-	-	TL005
5	Sistema retorna para a tela principal	FB03	-	-	TL001

4.CASO DE USO CONCLUIR TAREFA

PRÉ CONDIÇÃO

PR02 CONCLUIR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Deve haver ao menos uma tarefa já cadastrada	FB04	-	-	TL001

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

FLUXO BÁSICO

FB04 CONCLUIR TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	O usuário acessa e clica no checkbox	FB01	-	-	TL001
2	O sistema altera o valor do checkbox	-	RGN03	-	TL001

REGRAS

RGN03 REGRA DE CONCLUSÃO DE TAREFA					
ID	Passo	Fluxo	Regras	Msg	Tela
1	Caso a tarefa esteja concluída, retoma o status de não concluída	FB04	-	-	TL001

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

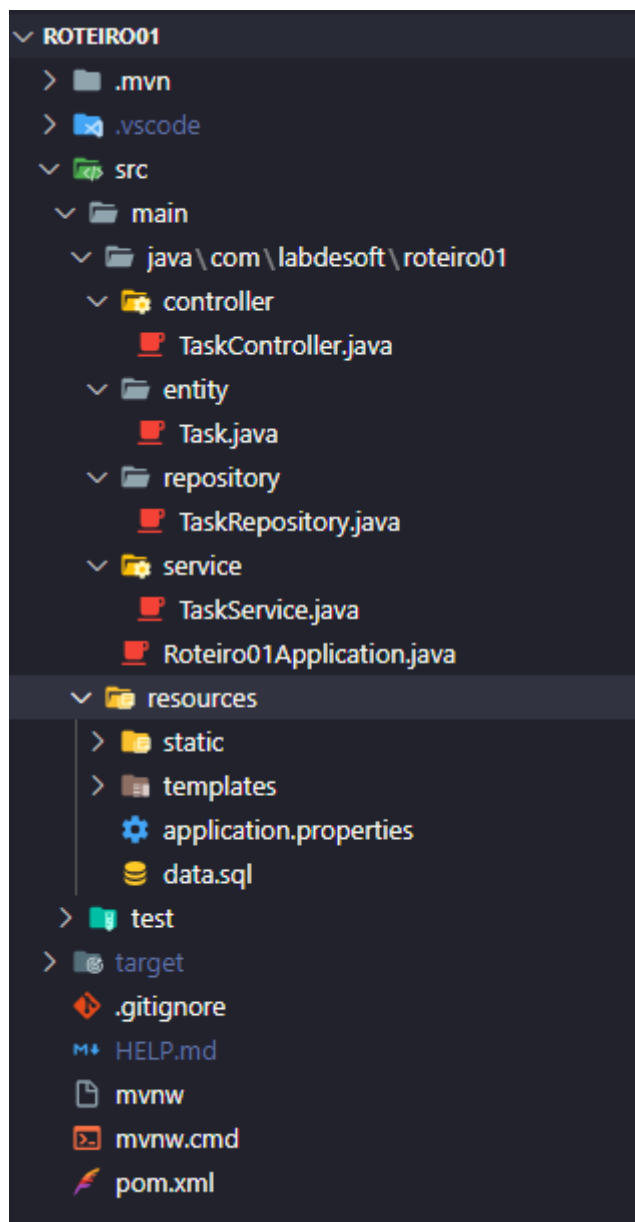
ENTREGA DE CÓDIGO

CÓDIGO FONTE

Link para conferir o código fonte:

<https://github.com/lucasandrademo/roteiro01/tree/laboratorio2>

Organização de Pastas:



¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: "[Como especificar casos de uso em 5 passos](#)", "[Como fazer um diagrama de casos de uso](#)" e "[Como documentar requisitos de software](#)".

A organização foi dividida da seguinte forma:

Entity: Pasta responsável pela organização dos arquivos referentes às tabelas, contendo nos arquivos, os dados dos campos, nomes das tabelas e os tipos de dados em cada campo;

Controller: Pasta responsável pela organização dos arquivos referentes às controladoras, contendo arquivos que recebem os endpoints e redirecionam as responsabilidades para o service.

Service: Pasta responsável pela organização dos arquivos referentes às gestões das informações, nele são feitas as regras de negócios, assim como a comunicação com os repositorys.

Repository: Pasta responsável pela organização dos arquivos referentes às interfaces que estendem comandos básicos ou complexos, envolvendo queries, busca de dados, criação de novas instâncias, etc.

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: [“Como especificar casos de uso em 5 passos”](#), [“Como fazer um diagrama de casos de uso”](#) e [“Como documentar requisitos de software”](#).

Arquivos

Entity:

Arquivo Task.java

```
import java.time.LocalDate;

import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;

import io.swagger.v3.oas.annotations.media.Schema;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.validation.constraints.FutureOrPresent;
import lombok.Data;

You, 30 seconds ago | 1 author (You)
@Entity
@Data
@Schema(description = "Todos os detalhes sobre uma tarefa. ")
public class Task {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String description;

    private Boolean completed;

    private Type type;

    private Priority priority;

    @FutureOrPresent(message = "Data superior a data de hoje")
    private LocalDate date;

    private Integer days;

    public Task(String description, Type type, Priority priority){
        this.description = description;
        this.type = type;
        this.priority = priority;
    }

    @Override
    public String toString() {
        return "Task [id=" + id + ", description=" + description + ", completed=" + completed + "];"
    }
}
```

Priority e Type são enums

```
public enum Priority {
    Alta,
    Media,
    Baixa
}
```

```
public enum Type {
    Data,
    Prazo,
    Livre
}
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Controller:

Arquivo TaskController.java

Parte das importações:

```
You, 14 minutes ago | 1 author (You)
1  package com.labdesoft.roteiro01.controller;
2
3  import org.springframework.beans.factory.annotation.Autowired;
4  import org.springframework.http.HttpStatus;
5  import org.springframework.http.ResponseEntity;
6  import org.springframework.web.bind.annotation.CrossOrigin;
7  import org.springframework.web.bind.annotation.DeleteMapping;
8  import org.springframework.web.bind.annotation.GetMapping;
9  import org.springframework.web.bind.annotation.PostMapping;
10 import org.springframework.web.bind.annotation.PutMapping;
11 import org.springframework.web.bind.annotation.RequestMapping;
12 import org.springframework.web.bind.annotation.RestController;
13
14 import com.labdesoft.roteiro01.service.TaskService;
15 import com.labdesoft.roteiro01.entity.Task;
16
17 import io.swagger.v3.oas.annotations.Operation;
18
19 import java.util.List;
20
```

Parte da definição da classe

```
You, 14 minutes ago | 1 author (You)
21 @CrossOrigin("*")
22 @RequestMapping("/task")
23 @RestController
24 > public class TaskController { ...
93
```

Parte da Autowired com o Service e endpoint GET para listar as tarefas:

```
25
26 @Autowired
27 TaskService taskService;
28
29 @GetMapping("/")
30 @Operation(summary = "Lista todas as tarefas da lista")
31 public ResponseEntity<List<Task>> listAll() {
32     try{
33         List<Task> allTasks = taskService.findAllTask();
34         if (allTasks.isEmpty()) {
35             return new ResponseEntity<> (HttpStatus.NO_CONTENT);
36         }
37         return new ResponseEntity<List<Task>> (allTasks, HttpStatus.OK);
38     } catch (Exception e) {
39         return new ResponseEntity<> (HttpStatus.INTERNAL_SERVER_ERROR);
40     }
41 }
42
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Parte do endpoint POST para listar uma única tarefa:

```
@GetMapping("/{id}")
@Operation(summary = "trás dados de uma tarefa específica")
public ResponseEntity<Task> listOne(@PathVariable Long id) {
    try{
        Task task = taskService.findTaskById(id);
        if (task == null) {
            return new ResponseEntity<>(HttpStatus.NO_CONTENT);
        }
        return new ResponseEntity<Task>(task, HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

Parte do endpoint POST para criar uma nova tarefa:

```
@Operation(summary = "Cria uma nova tarefa")
@PostMapping(path = "/")
public ResponseEntity<Object> create(@RequestBody Task task) {
    try {
        if(
            task.getDescription() == null ||
            task.getType() == null ||
            task.getPriority() == null
        ){
            return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
        }
        return new ResponseEntity<Object>(taskService.create(task), HttpStatus.CREATED);
    } catch (Exception e) {
        return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Parte do endpoint PUT para editar e atualizar uma tarefa já existente:

```
@Operation(summary = "Atualiza uma tarefa já existente")
@PutMapping(path = "/")
private ResponseEntity<Object> update(@RequestBody Task task) {
    try {
        if(
            task.getDescription() == null ||
            task.getType() == null ||
            task.getPriority() == null ||
            task.getId() == null
        ){
            return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
        }
        return new ResponseEntity<Object>(taskService.update(task), HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<Object>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

Parte do endpoint DELETE para excluir uma tarefa existente:

```
@Operation(summary = "Delete uma tarefa já existente")
@DeleteMapping(path =("/{id}")
private ResponseEntity<Object> delete(@PathVariable Long id) {
    try {
        taskService.delete(id);
        return new ResponseEntity<Object>(HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<Object>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

Parte do endpoint PUT para finalizar uma tarefa ou retirar o status de finalizada de uma tarefa:

```
@Operation(summary = "finaliza uma tarefa ou retira o status de finalizada")
@PutMapping(path = "/done/{id}")
private ResponseEntity<Object> done(@PathVariable Long id) {
    try {
        if(id == null){
            return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
        }
        return new ResponseEntity<Object>(taskService.complete(id), HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<Object>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: "[Como especificar casos de uso em 5 passos](#)", "[Como fazer um diagrama de casos de uso](#)" e "[Como documentar requisitos de software](#)".

Service:

Arquivo TaskService.java

Parte das importações:

```
1  package com.labdesoft.roteiro01.service;
2
3  import java.util.List;
4
5  import org.springframework.beans.factory.annotation.Autowired;
6  import org.springframework.stereotype.Service;
7
8  import com.labdesoft.roteiro01.entity.Task;
9  import com.labdesoft.roteiro01.repository.TaskRepository;
10
```

Parte da definição da classe:

```
10
11  @Service
12  > public class TaskService { ...
53
```

Parte da Autowired com o Repository ação de findAll para buscar as tarefas:

```
13
14  @Autowired
15  TaskRepository taskRepository;
16
17  public List<Task> findAllTask() {
18      return taskRepository.findAll();
19  }
20
```

Parte da ação de criação de uma nova tarefa:

```
20
21  public Task create(Task novaTask) {
22      return taskRepository.save(novaTask);
23  }
24
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Parte da ação de atualização de uma tarefa existente:

```
public Task update(Task task) {  
  
    Task taskToUpdate = taskRepository.findById(task.getId()).orElse(null);  
  
    if(taskToUpdate != null ) {  
        taskToUpdate.setDescription(task.getDescription());  
        taskToUpdate.setType(task.getType());  
        taskToUpdate.setCompleted(task.isCompleted());  
        taskToUpdate.setPriority(task.getPriority());  
        taskToUpdate.setPriority(task.getPriority());  
        taskToUpdate.setPriority(task.getPriority());  
        taskToUpdate.setDate(task.getDate());  
        taskToUpdate.setDays(task.getDays());  
    }  
    return taskRepository.save(taskToUpdate);  
}
```

Parte da ação de exclusão de uma tarefa existente:

```
35  
36     public void delete(Long id) {  
37         taskRepository.deleteById(id);  
38     }  
39
```

Parte da ação de finalização ou retirada da finalização de uma tarefa existente:

```
public Task complete(Long id) {  
    Task task = taskRepository.findById(id).orElse(null);  
    if(task != null ) {  
        if(task.isCompleted()){  
            task.setCompleted(completed:false);  
        }else{  
            task.setCompleted(completed:true);  
        }  
    }  
    return taskRepository.save(task);  
}
```


¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).

Parte da ação de obter dados de uma única task:

```
public Task findTaskById(Long id) {  
    return taskRepository.findById(id).orElse(null);  
}
```

Repository:

Arquivo TaskRepository.java

```
You, 13 minutes ago | 1 author (You)  
1 package com.labdesoft.roteiro01.repository; You, 13 minutes ago  
2   
3 import org.springframework.boot.autoconfigure.domain.EntityScan;  
4 import org.springframework.data.jpa.repository.JpaRepository;  
5 import org.springframework.stereotype.Repository;  
6  
7 import com.labdesoft.roteiro01.entity.Task;  
8  
You, 13 minutes ago | 1 author (You)  
9 @EntityScan  
10 @Repository  
11 public interface TaskRepository extends JpaRepository<Task, Long> {  
12  
13 }  
14
```

¹ Para entender melhor o que são requisitos de software e casos de uso, aconselhamos a leitura dos artigos: ["Como especificar casos de uso em 5 passos"](#), ["Como fazer um diagrama de casos de uso"](#) e ["Como documentar requisitos de software"](#).