

# Controle de Férias

Guilherme Eduardo Bom Guse

Janaina Mai

Lucas Walim

Thiago Luiz Barbieri

Willian Kenji Nishizawa



# Review da Sprint

## 0 que foi feito

- Refatoração da Sprint anterior
- Separação de camadas em MVC
- Design Pattern: DAO, Builder e Singleton
- Orientação a Interfaces
- Conceitos de Código Limpo
- Princípios de SOLID

## 0 que foi adicionado

- Divisão de tarefas por branches no Git
- Implementamos Builder e Director para todas as classes
- Persistência de dados utilizando Singleton
- TDD nas classes DAO

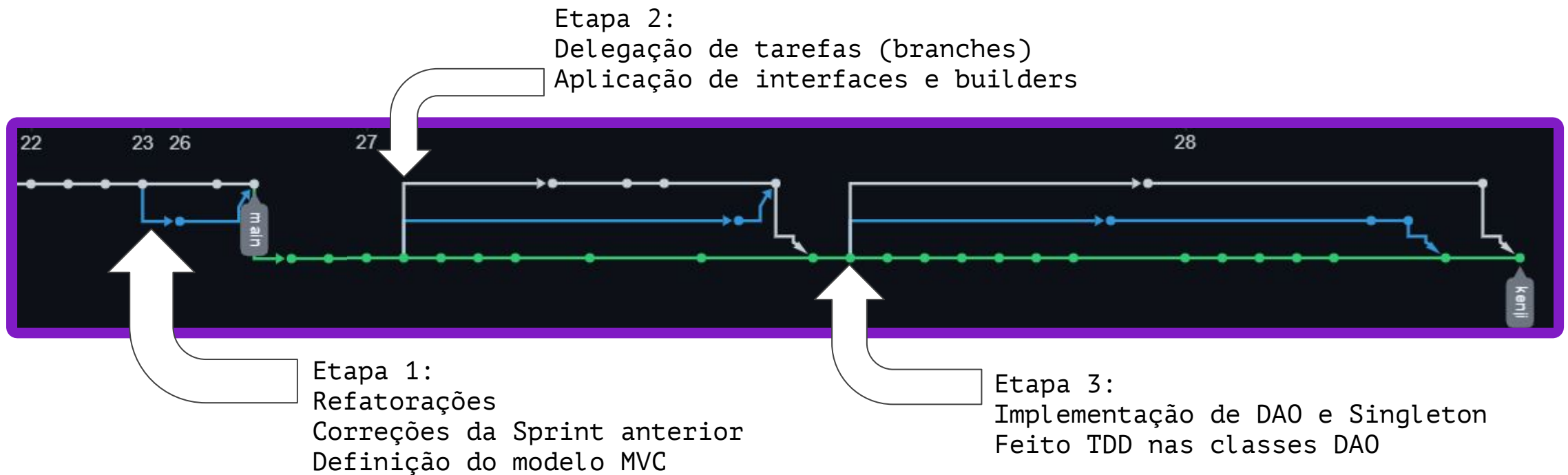
## 0 que não foi feito

- Diagramas de sequência UML
- Alguns casos de teste
- Controller da classe Ferias

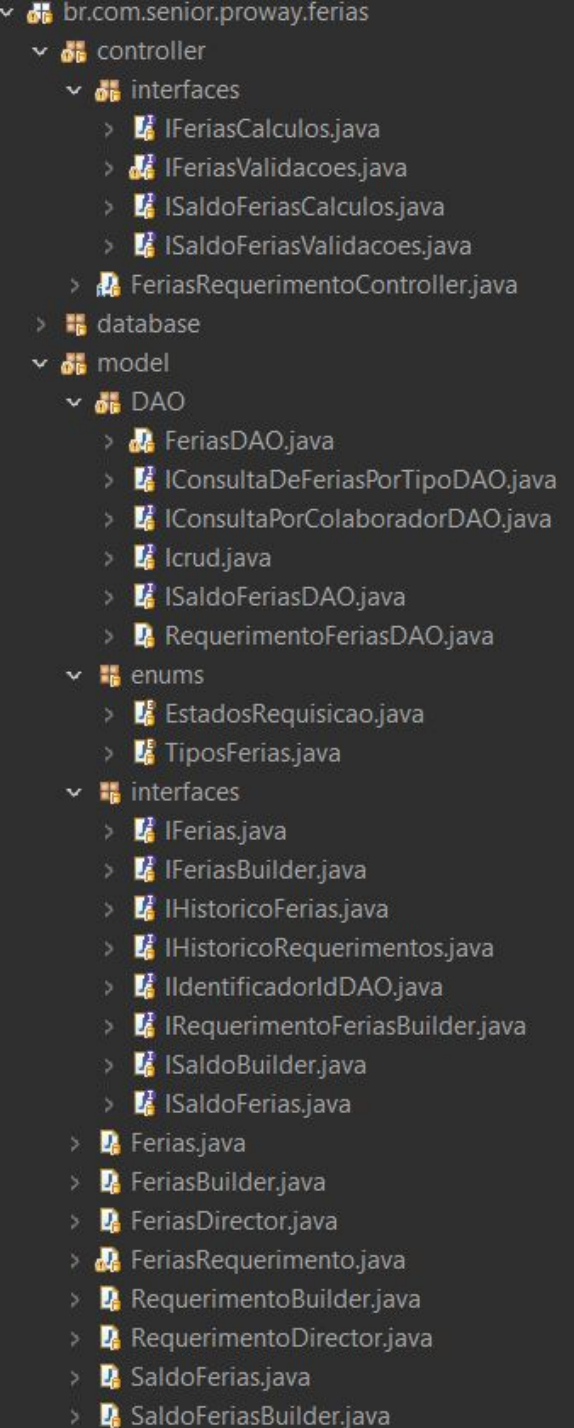
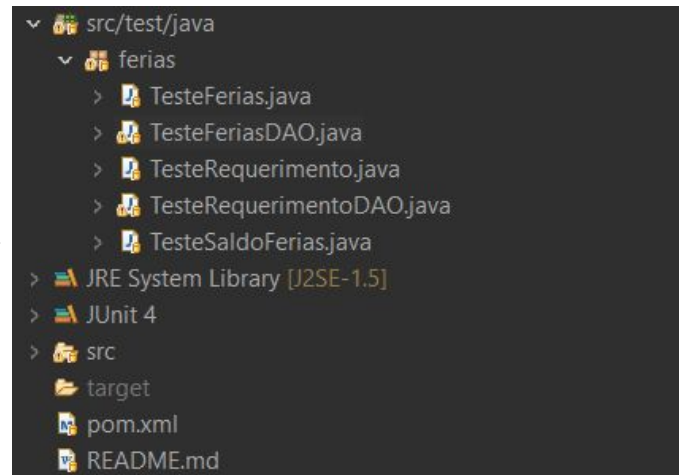
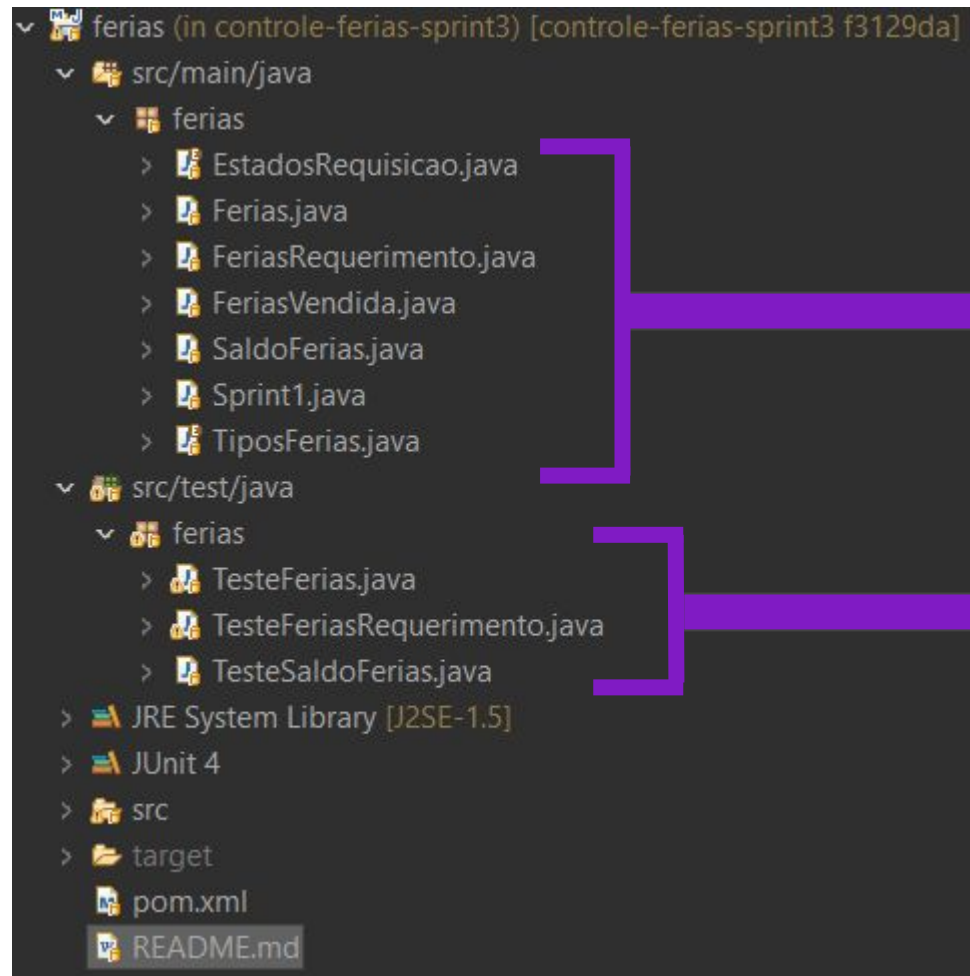
## 0 que foi removido

- Heranças entre classes do tipo Férias

# Linha do tempo - Github

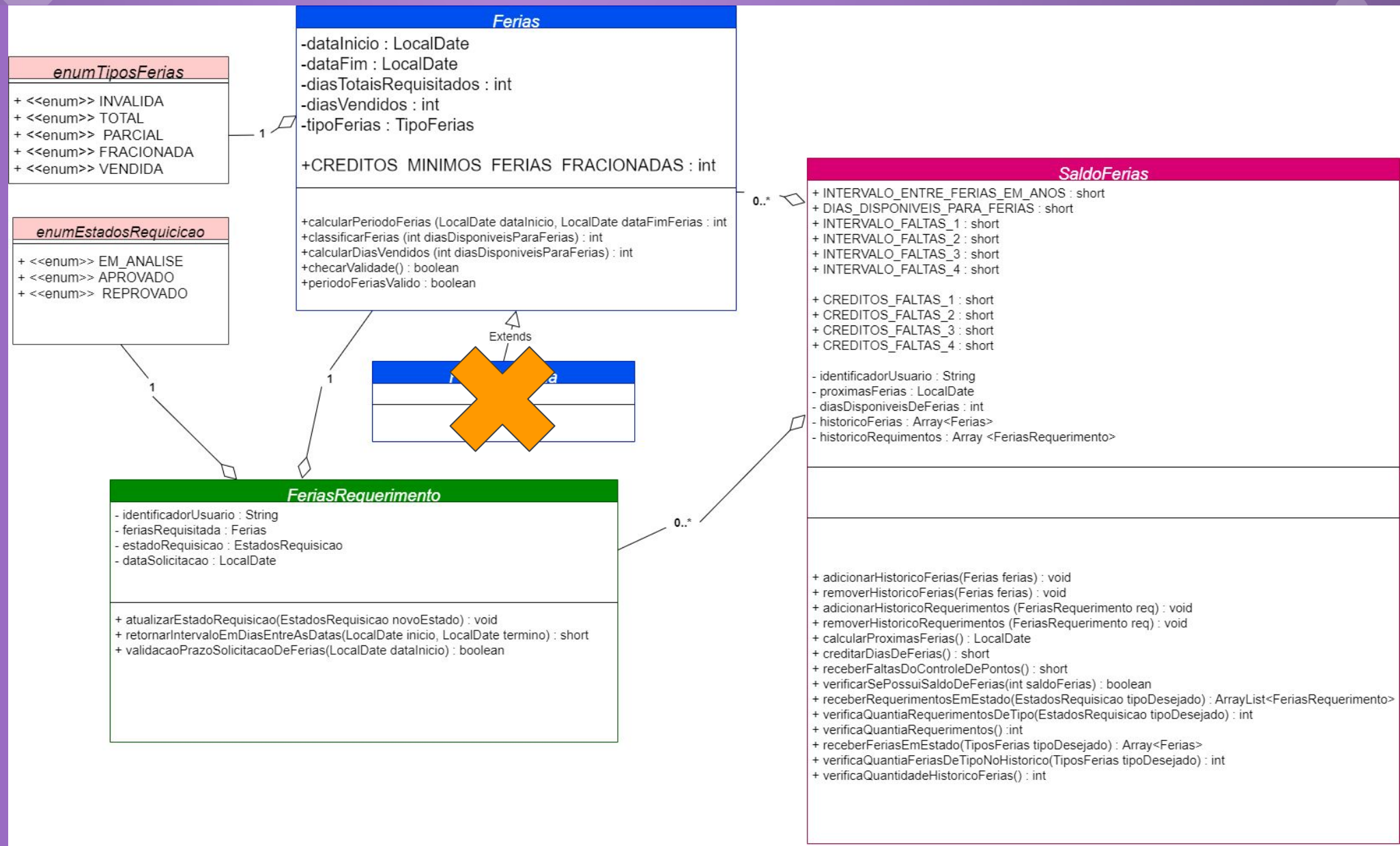


# Camadas: antes e depois



# Diagramas de classe

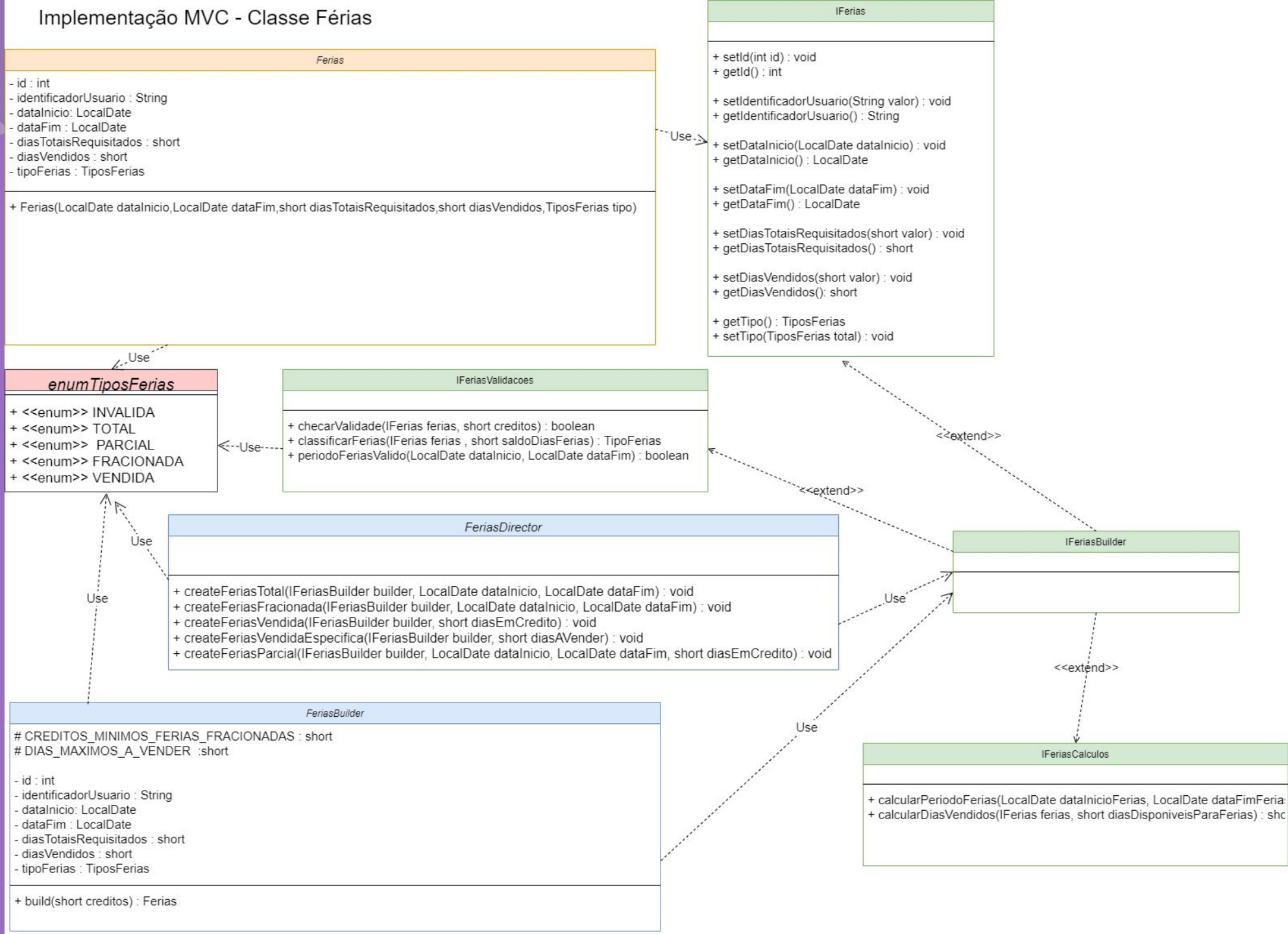
Sprint anterior





# Diagramas de classe

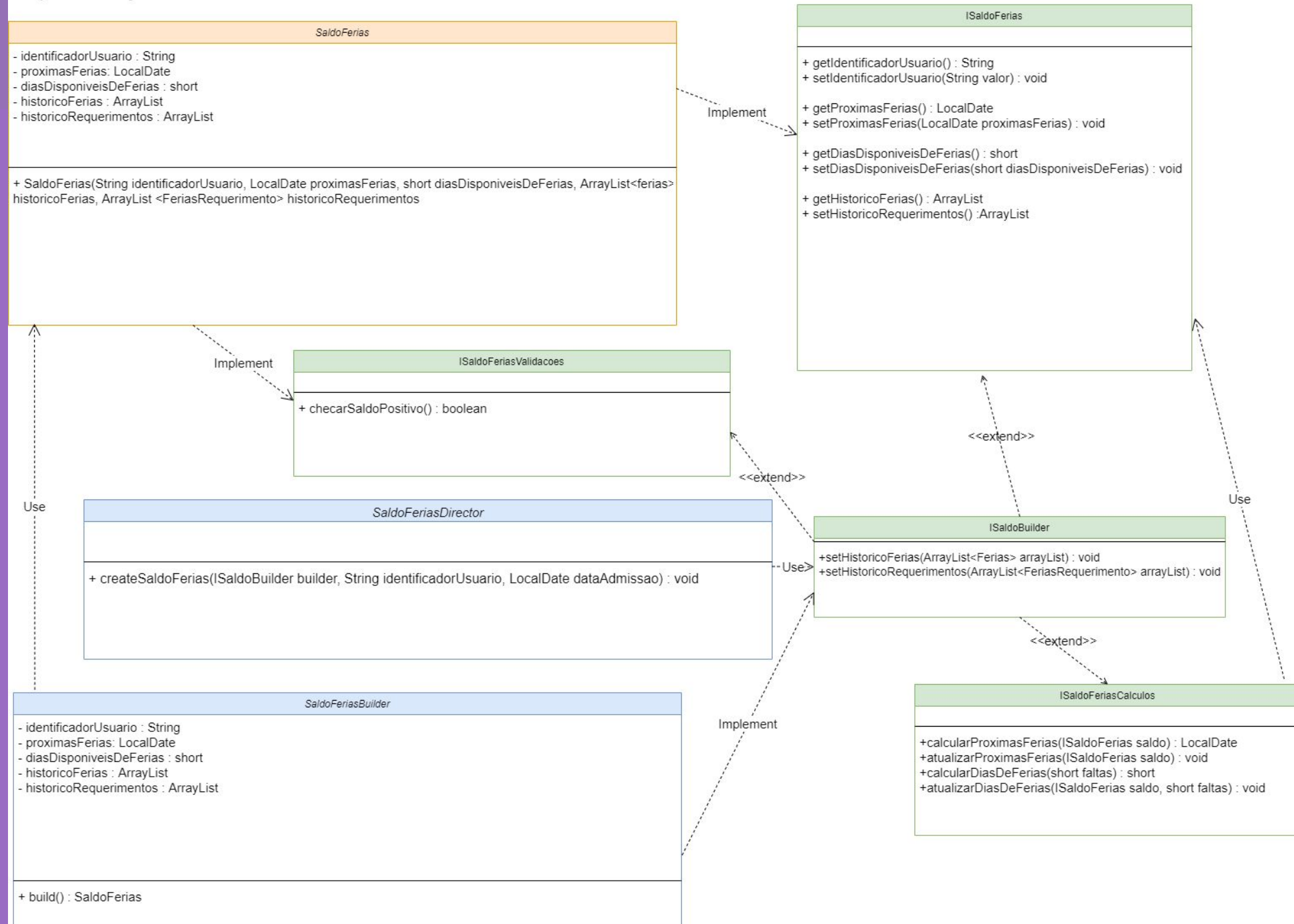
## Classe: Férias



# Diagramas de classe

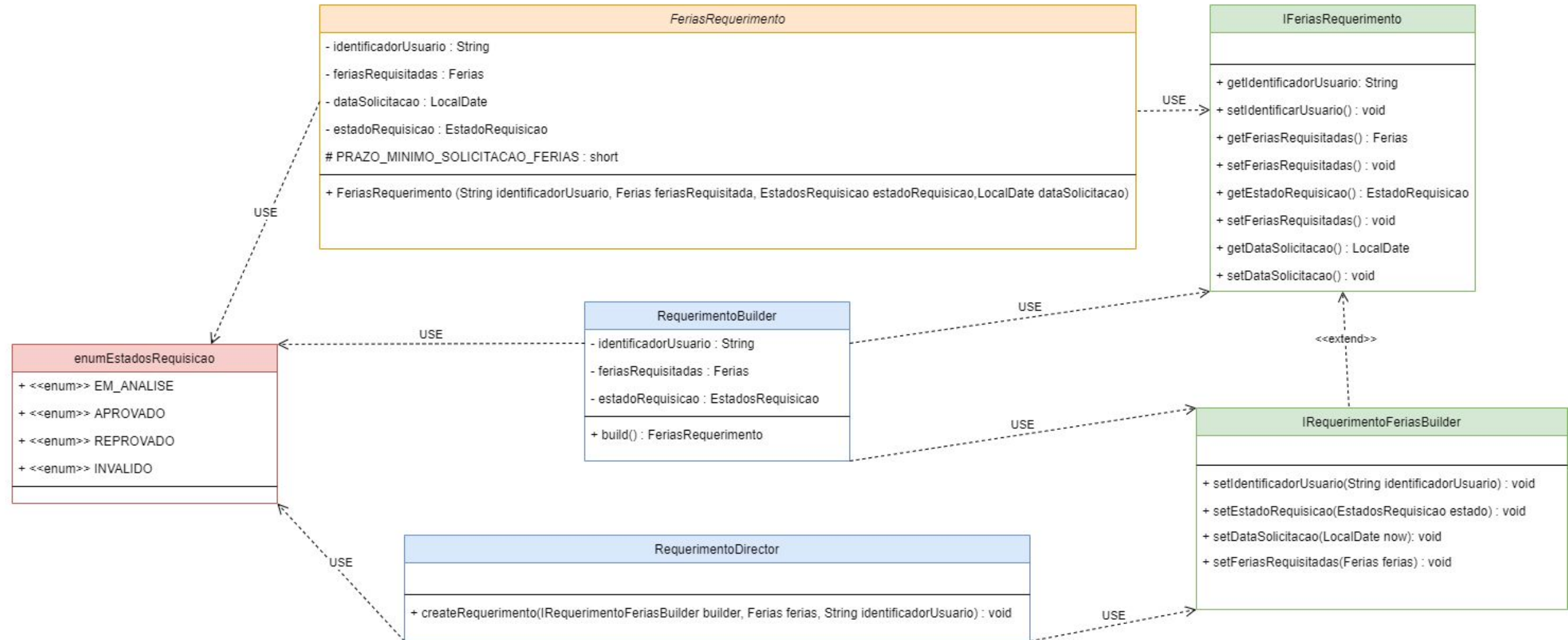
## Classe: Saldo de Férias

### Implementação MVC - Classe SaldoFerias



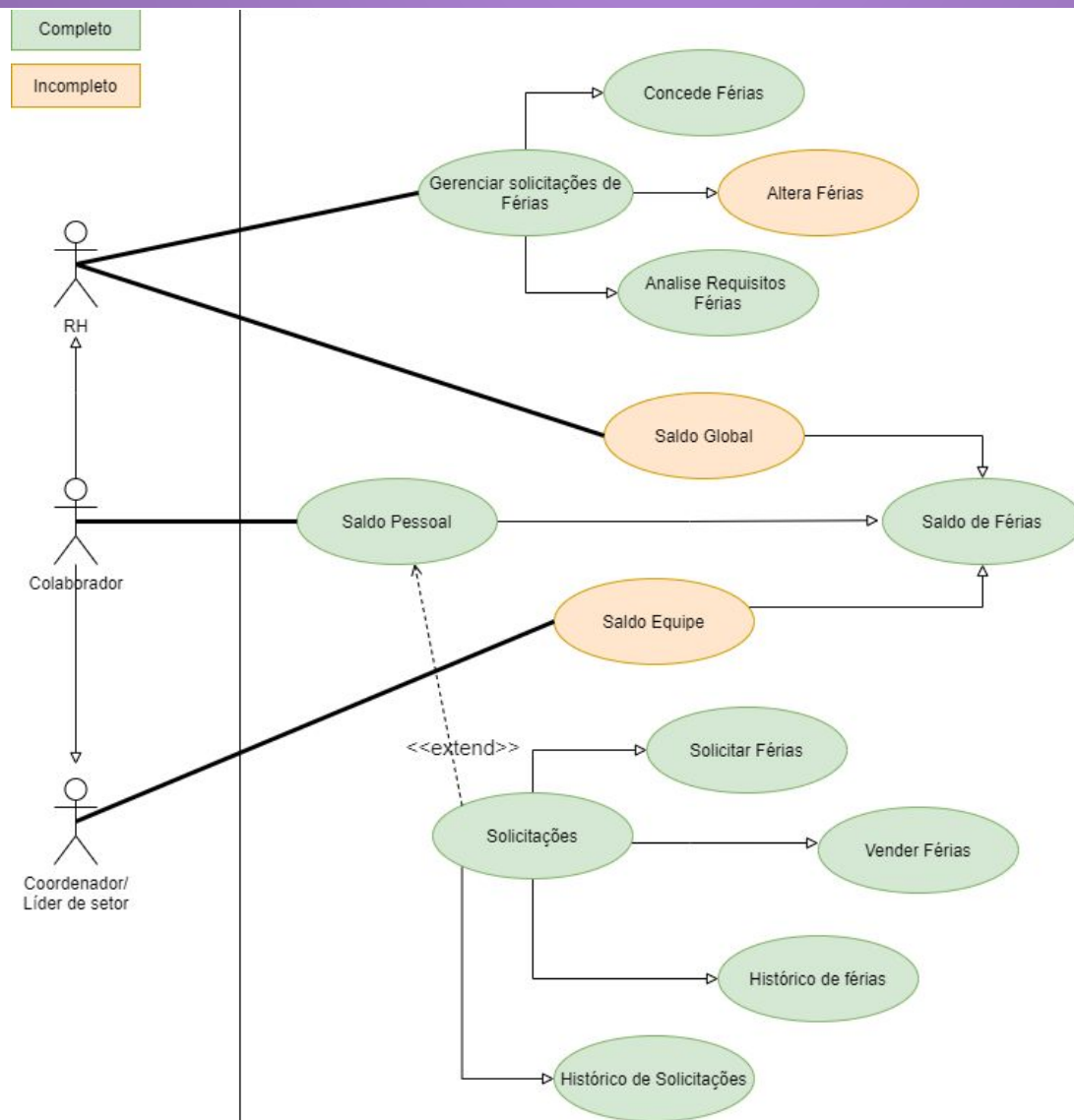
# Diagramas de classe

## Classe: Requerimento





# Diagrama de Casos



# Trello

## Documentação

SOLID



GG

Anexos e Informações ~wk3



8

Anexos e informações ~wk2



7

Instruções - Instanciando as classes  
Férias



Instruções - Concessão de férias



+ Adicionar outro cartão



## Backlog da Sprint

+ Adicionar um cartão



## Fazendo

Refatorar/Criar Diagramas UML

2/3

GG

LW

TB

WN

Criar Slides

2



GG

JM

WN

+ Adicionar outro cartão



## Pronto

"DB" Singleton

3



GG

JM

LW

TB

Classe Ferias

1



6/7

WN

Classe SaldoFerias

6/7

WN

Classe RequerimentoFerias

0/7

LW

TB

Classe DAO - Ferias

1



2/2

GG

JM

Classe DAO - RequerimentoFerias

LW

TB

Classe DAO - SaldoFerias

LW

TB

Definir modelo MVC para o Sistema

1



3/3

GG

JM

LW

TB

WN

+ Adicionar outro cartão



## Bloqueado

Issue: Singleton de Persistência de  
Dados não está limpando os próprios  
dados quando requisitado.

Verificar Injeção de Dependências no  
SaldoFerias

+ Adicionar outro cartão



# Retrospectiva da Sprint



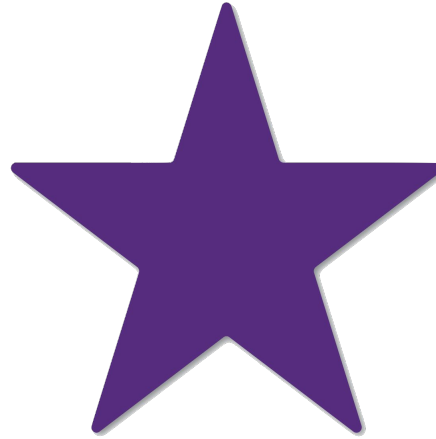
## Fazer mais:

1. Daily Meeting.
2. Uso do Trello.
3. Conferência diária de todos os códigos.
4. Interação entre os membros.
5. Utilizar as ferramentas de debug



## Fazer menos:

1. Tentar resolver problemas sem pedir ajuda.



## Parar de fazer:

1. Testar posteriormente.
2. Levantamento de requisitos, parcial.
3. Documentação com caracteres especiais.
4. Versionamento com pendrive



## Continuar fazendo:

1. Foco no aprendizado.
2. Troca de conhecimento.
3. *Pair programming*.
4. Aplicação de todos os conceitos apresentados em aula.

## Começar a fazer:

1. Rodízio do *pair programming*.
2. Preparar diagramas antes de *codar*.

# 0 que aprendemos

- Separação de camadas em MVC
- Design Patterns: DAO, Builder e Singleton
- Orientação a Interfaces
- Conceitos de Código Limpo
- Princípios de SOLID
- Ordem em que os testes são executados (randômica ou personalizada)
- Cobertura dos testes

## Utilizando Builder

```
@Test
public void feriasTotal() {
    short credits = 30;
    LocalDate data1 = LocalDate.of(2021, 4, 15);
    LocalDate data2 = LocalDate.of(2021, 5, 15); // 30 dias

    FeriasDirector feriasDirector = new FeriasDirector();
    FeriasBuilder Bob = new FeriasBuilder();
    feriasDirector.createFeriasTotal(Bob, data1, data2);
    Ferias ferias = Bob.build(credits);

    assertEquals(ferias.getTipo(), TiposFerias.TOTAL);
    assertEquals(0, ferias.getDiasVendidos());
}
```



## Métodos do Director

```
public void createFeriasFracionada(
    IFeriasBuilder builder, LocalDate dataInicio, LocalDate dataFim
) {
    builder.setDataInicio(dataInicio);
    builder.setDataFim(dataFim);
    builder.setTipo(TiposFerias.FRACIONADA);
    builder.setDiasTotaisRequisitados(
        builder.calcularPeriodoFerias(builder.getDataInicio(), builder.getDataFim())
    );
    builder.setDiasVendidos((short) 0);
}

/** Instancia um objeto de Ferias classificado como Ferias VENDIDA
 * @param builder Objeto Builder responsavel por instanciar Ferias
 * @param diasEmCredito Saldo disponivel de creditos para ferias
 */
public void createFeriasVendida(
    IFeriasBuilder builder, short diasEmCredito
) {
    builder.setDataInicio(null);
    builder.setDataFim(null);
    builder.setTipo(TiposFerias.VENDIDA);
    builder.setDiasTotaisRequisitados(
        builder.calcularPeriodoFerias(builder.getDataInicio(), builder.getDataFim())
    );
    builder.setDiasVendidos(builder.calcularDiasVendidos(builder, diasEmCredito));
}
```

# Utilizando Singleton

```
@FixMethodOrder(MethodSorters.NAME_ASCENDING)
public class TesteRequerimentoDAO {
    DataBase dbSingle = DataBase.getInstance();

    @Test
    public void testeACreate() {
        short credits = 30;

        LocalDate inicio = LocalDate.of(2021, 04, 01);
        LocalDate fim = LocalDate.of(2021, 04, 28);

        FeriasDirector feriasDiretor = new FeriasDirector();
        FeriasBuilder feriasBuilder = new FeriasBuilder();

        feriasDiretor.createFeriasParcial(feriasBuilder, inicio, fim, credits);
        Ferias ferias = feriasBuilder.build(credits);









        RequerimentoDirector directorRequerimento = new RequerimentoDirector();
        RequerimentoBuilder builderRequerimento = new RequerimentoBuilder();

        directorRequerimento.createRequerimento(builderRequerimento, ferias, "Roberto");
        RequerimentoFerias feriasRequerimento = builderRequerimento.build();

        RequerimentoFeriasDAO DAOFerias = new RequerimentoFeriasDAO();
        boolean resultado = DAOFerias.create(feriasRequerimento);

        assertTrue(resultado);
        assertEquals(dbSingle.requerimentos.size(), 1);
        assertEquals(dbSingle.requerimentos.get(0).getIdentificadorUsuario(), "Roberto");
    }
}
```

## Cobertura dos testes (resumido)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
▼  ferias	 76,9 %	3.092	929	4.021
▼  src/main/java	 53,6 %	905	784	1.689
>  br.com.senior.proway.ferias.model.DAO	 26,9 %	155	422	577
>  br.com.senior.proway.ferias.controller	 42,8 %	122	163	285
>  br.com.senior.proway.ferias.database	 58,0 %	29	21	50
>  br.com.senior.proway.ferias.model	 73,8 %	501	178	679
>  br.com.senior.proway.ferias.model.enums	 100,0 %	98	0	98
▼  src/test/java	 93,8 %	2.187	145	2.332
>  ferias	 93,8 %	2.187	145	2.332



# Cobertura dos testes (detalhado)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
▼ ferias	76,9 %	3.092	929	4.021
▼ src/main/java	53,6 %	905	784	1.689
▼ br.com.senior.proway.ferias.model.DAO	26,9 %	155	422	577
> SaldoFeriasDAO.java	0,0 %	0	161	161
> FeriasDAO.java	3,0 %	8	261	269
> RequerimentoFeriasDAO.java	100,0 %	147	0	147
▼ br.com.senior.proway.ferias.controller	42,8 %	122	163	285
> FeriasRequerimentoController.java	27,0 %	33	89	122
> SaldoFeriasController.java	54,6 %	89	74	163
▼ br.com.senior.proway.ferias.database	58,0 %	29	21	50
> DataBase.java	58,0 %	29	21	50
▼ br.com.senior.proway.ferias.model	73,8 %	501	178	679
> SaldoFerias.java	41,7 %	35	49	84
> Ferias.java	41,8 %	28	39	67
> RequerimentoFerias.java	67,4 %	31	15	46
> RequerimentoBuilder.java	70,6 %	36	15	51
> SaldoFeriasBuilder.java	71,2 %	37	15	52
> FeriasBuilder.java	79,6 %	176	45	221
> FeriasDirector.java	100,0 %	114	0	114
> RequerimentoDirector.java	100,0 %	19	0	19
> SaldoFeriasDirector.java	100,0 %	25	0	25
▼ br.com.senior.proway.ferias.model.enums	100,0 %	98	0	98
> EstadosRequerimentos.java	100,0 %	44	0	44
> TiposFerias.java	100,0 %	54	0	54
▼ src/test/java	93,8 %	2.187	145	2.332
▼ ferias	93,8 %	2.187	145	2.332
> TesteFeriasDAO.java	86,6 %	923	143	1.066
> TesteRequerimento.java	98,4 %	126	2	128
> TesteFerias.java	100,0 %	278	0	278
> TesteRequerimentoDAO.java	100,0 %	597	0	597
> TesteSaldoFerias.java	100,0 %	263	0	263

*Obrigado(a)!*

