

Escola Superior de Tecnologia e Gestão

Relatório

Escalonamento hospitalar com férias

Grupo 2:

Berat Devecioğlu

Diogo Sá

Miguel Almeida - nº 1011311

Disciplina: Engenharia de Software II

Docente: Maria Clara Silveira

Tabela de Conteúdo

[Definição do Tema do Projeto 3](#_Toc531379114)

[Descrição dos atores 3](#_Toc531379115)

[Casos de Uso 4](#_Toc531379116)

[Lista de Casos de Uso 4](#_Toc531379117)

[Diálogos de Casos de Uso 4](#_Toc531379118)

[Requisitar ferias 4](#_Toc531379119)

[Diagrama de Contexto 6](#_Toc531379120)

[Diagrama de Casos de Uso 7](#_Toc531379121)

[Descrição de Padrões de Caso de Uso 8](#_Toc531379122)

[Detectable conditions (condições detetáveis) 8](#_Toc531379123)

[Anexo 9](#_Toc531379124)

[Algoritmo gerar horário 9](#_Toc531379125)

# Definição do Tema do Projeto

## Descrição dos atores

Medico: O medico pode requisitar ferias, consultar o seu horário e as ferias, pode pedir a alteração do horário, a alteração ou cancelamento das ferias e é lhe enviado o seu horário.

Enfermeiro: o enfermeiro pode requisitar ferias, consultar o seu horário e as ferias, pode pedir a alteração do horário, a alteração ou cancelamento das ferias e é lhe enviado o seu horário.

Diretor de Serviço: o diretor de serviço pode aprovar ferias, aprovar alterações nos horários, propor marcação de ferias, alterar os horários, editar as regras de ferias, consultar o seu próprio horário e gerar horários.

# Casos de Uso

Na Engenharia de Software, um **caso de uso** (do inglês *use case*) é um tipo de classificador representando uma unidade funcional coerente provida pelo sistema, subsistema, ou classe manifestada por sequências de mensagens intercambiáveis entre os sistemas e um ou mais atores.

Especificações de casos de uso são narrativas em texto, descrevendo a unidade funcional, e são amplamente utilizados para representar requisitos funcionais nos sistemas.

## Lista de Casos de Uso

* Requisitar ferias
* Consultar Horários e ferias
* Pedir alteração de horário
* Pedir alteração ou cancelamento das ferias
* Enviar horários
* Aprovar ferias
* Aprovar alteração de horários
* Propor marcação de ferias
* Alterar horários
* Editar regras de horários
* Editar regras de ferias
* Gerar horários

## Diálogos de Casos de Uso

### Requisitar ferias

Descrição: Formulário do sistema para que médicos e enfermeiros possam efetuar um pedido de férias.

Pré-Condições

Login Valido

Caminho principal

1. O utilizador seleciona a opção para requerer férias
2. O sistema devolve um formulário com a data de início e fim para preencher
3. O utilizador insere as datas que pretende ter como férias
4. O sistema guarda o formulário.

Caminhos alternativos

* 2a. O sistema diz ao utilizador que não tem direito a férias.
* 3a. O sistema envia um alerta de que o utilizador está a inserir um período de tempo que excede o permitido.
* 3b. O sistema diz que o formulário não é valido, pois, a data é demasiado próxima a férias já marcadas pelo mesmo utilizador.
* 3c. O sistema diz que o formulário não é valido, pois, a data coincide com um período já indisponível, ocupado por outros utilizadores(funcionários)

Pós-Condição

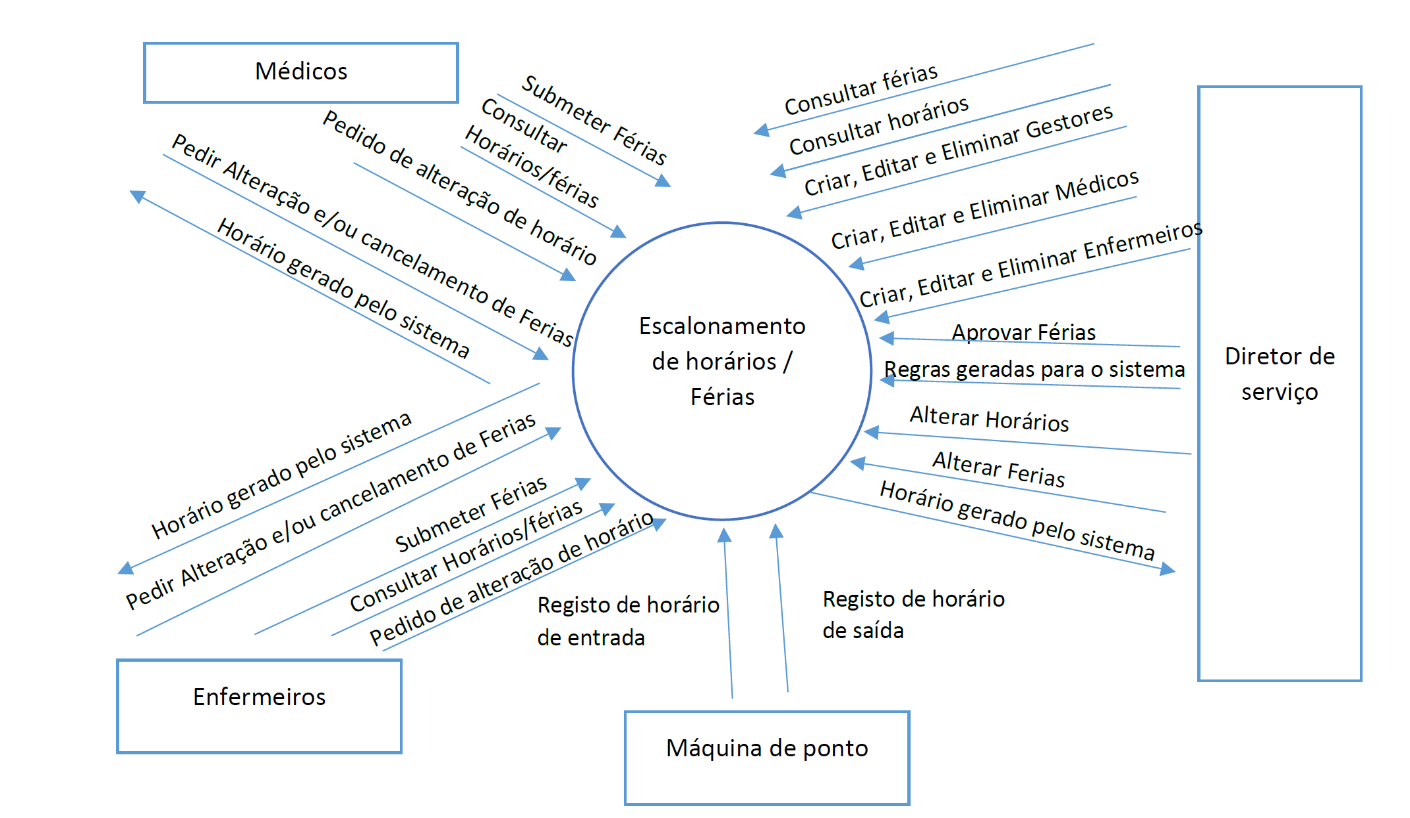
O Sistema notifica um supervisor para revisão.

​

# Diagrama de Contexto

**Diagrama de Contexto** é a representação do sistema como um único processo.

A sua representação baseia-se em fluxos de dados que nos mostram as interfaces entre o **Sistema,** Sistema de Trilhos, e as **Entidades Externas**.



# Diagrama de Casos de Uso

# Descrição de Padrões de Caso de Uso

## Detectable conditions (condições detetáveis)

Escritores estão sempre a discutir quantas e quais condições devem ser incluídas.

Um sistema não consegue lidar com eventos que não consegue detetar.

exemplo se um código for escrito para o computador de bordo de um carro, perder uma chave, não ter combustível, etc, vai tudo levar à mesma condição, o carro não funciona, sendo a resolução para cada circunstância diferente o computador não quer saber, só sabe que o carro não funciona.

Casos de uso eficazes podem fornecer aos desenvolvedores uma visão completa do comportamento do sistema, enumerando todas as condições que se espera que o sistema encontre, com descrições das suas resoluções.

Não dar conta de condições que deviam ser detetadas pelo sistema pode tornar-se caro se só for detetado tarde durante ou pós o desenvolvimento, mas também muitas condições que aparentam ser diferentes acabam por ser a mesma condição, isto pode levar a conflitos de código.

Em conclusão, devemos apenas incluir condições detetáveis pelo sistema e juntar condições que tenham o mesmo impacto no sistema.

​

# Anexo

## Algoritmo gerar horário

**Escolher funcionário para turno da manha: fazer até preencher vagas mínimas**

Verificar se funcionário já fez 40h semanais

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já fez 3 dias de trabalho seguidos

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já foi atribuído no mesmo dia

se sim: Escolher outro funcionário

se não: Atribuir funcionário

**Escolher funcionário para turno da tarde: fazer até preencher vagas mínimas**

Verificar se funcionário já fez 40h semanais

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já fez 3 dias de trabalho seguidos

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já foi atribuído no mesmo dia

se sim: Escolher outro funcionário

se não: Atribuir funcionário

**Escolher funcionário para turno da noite: fazer até preencher vagas mínimas**

Verificar se funcionário já fez 40h semanais

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já fez 3 dias de trabalho seguidos

se sim: Escolher outro funcionário

se não:

Verificar se funcionário já foi atribuído no mesmo dia

se sim: Escolher outro funcionário

se não:

Verificar se funcionário tem alguma restrição com horários noturnos

se sim: Escolher outro funcionário

se não: Atribuir funcionário

//não incluído regras especiais(restrições)

Para cada dia na semana fazer:

**Escolher o primeiro ID\_funcionário para o turno da manha em determinada data** (respeitando um limite mínimo)

Verificar se funcionário nas últimas 72horas (3 dias) o funcionário trabalhou 24 horas (3 turnos de 8horas)

se sim: Funcionário ignorado (no dia)

se não:

verificar se nos últimos 7 dias o funcionário fez 40 horas

se sim: Funcionário ignorado (no dia)

se não:

Verificar se funcionário fez noite dia anterior

se sim: ignorar funcionário (neste turno)

se não:

verificar se existe restrições para a manhã

se sim: ignorar funcionário (neste turno)

se não:

atribuir horário da manha ao ID\_Funcionario nesta data

​