Uma imagem com texto, Tipo de letra, logótipo, Gráficos

Descrição gerada automaticamenteUma imagem com Tipo de letra, texto, Gráficos, logótipo

Descrição gerada automaticamente

**ASIST**

**SPRINT 2**

**Realizado por:**

**Diogo Ferreira, 1220829**

**Francisco Osório, 1220846**

**Sérgio Moreira, 1220890**

**Rafael Ferraz, 1221104**

**11/2024**

[Divisão Tarefas 3](#_Toc181963875)

# Divisão Tarefas

Tabela - Divisão das Tarefas

|  |  |
| --- | --- |
| User Stories | Aluno |
| 1 | Francisco Osório |
| 2 | Diogo Ferreira |
| 3 | Rafael Ferraz |
| 4 | Sérgio Moreira |
| 5 | Francisco Osório |
| 6 | Diogo Ferreira |
| 7 | Rafael Ferraz |
| 8 | Sérgio Moreira |

# User Stories

## User Story 1

## User Story 2

## User Story 3

Para esta User Story, temos de fazer com que seja possível definir os clientes do DEI, simplesmente mudando um ficheiro de texto.

Para isso, começamos por criar o tal ficheiro de texto, que vai conter os endereços permitidos. Fazemos isso com o comando “*nano ips.txt*”. Dentro do ficheiro colocamos o IP da máquina (linha 1) e o IP da rede interna do DEI, que os clientes usariam (linha 2):

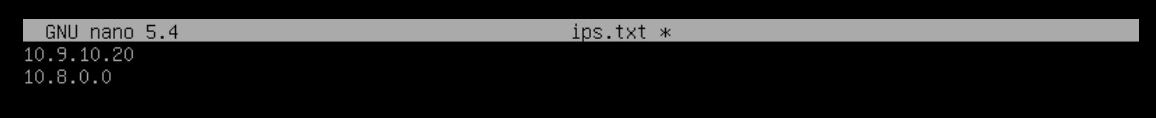


Figura ips.txt

Em seguida, criamos um pequeno script que permite acesso aos clientes presentes no “*ips*.*txt*”, usamos “*nano script\_us6-4-3.sh*” para criar o ficheiro, que contem um ciclo *for* para cada *ip* encontrado em “*ips.txt”*, adiciona uma regra no *iptables* que permite tráfego TCP na porta 22, que corresponde a *SSH. Finalmente, é adicionada uma regra que bloqueia o trafico de outras ligações*:

*Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente*

Figura Script para as iptables

Não nos podemos esquecer de dar permissão de execução ao ficheiro com o comando “*chmod +x script\_us6-4-3.sh*”. Agora, corremos o script com “*./script\_us6-4-3.sh*”. Podemos verificar que o script fez o pretendido:

Uma imagem com texto, captura de ecrã, Tipo de letra, preto

Descrição gerada automaticamente

Figura iptables depois de correr o script

Por último, devemos guardar as mudanças de modo a se manterem mesmo após reiniciar o sistema. Para isso, corremos “*sudo netfilter-persistent save*”

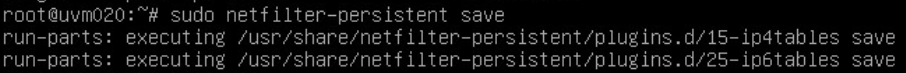


Figura Guardar permanentemente os iptables

Como podemos ver abaixo, após reiniciar o sistema as configurações ainda estão presentes:

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura iptables depois de reiniciar o sistema

## User Story 4

## User Story 5

Nesta user story era pedido que se defini-se o *MBCO (Minimum Business Continuity Objective)* a propor aos stakeholders.

O *Mininum Business Continuity Objective (MBCO)* especifica o nível mínimo de operacionalidade que deve ser mantida durante uma disrupção na infraestrutura.

A aplicação que está a ser desenvolvida encontra-se divida em vários módulos: o *backend*, o *frontend*, incluindo dentro o *3d visualization*, e o *planning*. Sendo que cada módulo tem uma função distinta e consequentemente maior importância, é necessário ter em atenção ao *Maximum Tolerable Downtime (MTD)* e ao *Maximum Tolerable Period of Disruption (MTPD)* para cada um.

Começando pelo módulo de *backend*, este é o responsável por processar e armazenar os dados, permitindo a gestão de pacientes, staffs, tipos de cirurgias, pedidos de cirurgias e marcação destas, sendo assim considerado o core da nossa aplicação. Desta forma, foi definido que o *MTD* para este módulo seria de 1 hora, e após esse tempo este módulo apresentará funcionalidades mínimas como gestão de pacientes, pedidos de cirurgias e suas marcações até ao *MTPD,* que foi definido para 12 horas.

Passando agora para o módulo de *frontend*, este é o que permite aos utilizadores interagirem com o sistema e realizarem funcionalidades como gestão de pacientes, staffs, tipos de cirurgias, pedidos de cirurgias e respetivas marcações, funcionalidades baseadas no módulo *backend*. Sendo assim, o *frontend* é considerado um módulo essencial para a continuidade das operações da aplicação. Assim, como no *backend*, definiu-se também um *MTD* de 1 hora. Após esse tempo, funcionalidades como gestão de pacientes, pedidos de cirurgias e suas marcações, estarão disponíveis, podendo estar outras ainda indisponíveis. Para recuperar todas as funcionalidades foi definido um período também de 12 horas, ou seja, para o *MTPD*.

O módulo *3d visualization* encontra-se integrado no *frontend*, e este permite aos utilizadores uma visão em tempo real do hospital, sendo possível ver os quartos e as respetivas operações que se encontram a decorrer. Apesar deste módulo ser útil, é secundário em relação aos módulos anteriores. Desta forma, foi definido um *MTD* de 6 horas, sendo que após este já deve ser possível os utilizadores verem um layout simplificado com o status de ocupação dos quartos, ainda com poucos detalhes. Já o *MTPD* foi definido para 48 horas, permitindo que este seja restaurado integralmente dentro desse prazo.

Por fim, o módulo *planning* é o que é responsável por gerar o agendamento das cirurgias e otimizar estes horários de acordo com diferentes critérios e disponibilidade de profissionais médicos e quartos. Este módulo também tem a sua importância, como a otimização, apesar disso, pode também ser considerado secundários, pois é possível realizar agendamentos manuais temporariamente. Assim, determinou-se que o tempo de inatividade deste módulo será de 3 horas, ou seja, o *MTD*. Após o *MTD* e até o *MTPD* a sua funcionalidade mínima, permitirá o agendamento básico de cirurgias, deixando otimização automática desativada temporariamente, sendo que o definiu-se o *MTPD* para 24 horas.

Concluindo, garante-se que as funcionalidades críticas da aplicação, continuem disponíveis, com tolerâncias razoáveis para restabelecimento de cada módulo em caso de disrupção, minimizando assim os impactos.

## User Story 6

## User Story 7

Nesta US é-nos pedido que se define uma pasta pública para todos os utilizadores registados no sistema, de modo a poderem ler o que lá está.

Primeiro, precisamos de criar a tal pasta, que depois será configurada como pública. Para isto usamos o comando “*mkdir*”:



Figura 6 Criar a pasta partilhada

Em seguida, precisamos de mudar as permissões da pasta, o objetivo é que todos tenham permissão de leitura, para isso, temos de usar o comando “*chmod -R a+r /shared”*:



Figura 7 Dar permissão de leitura para ficheiros na pasta partilhada

O comando “*chmod*” é usado para mudar permições de acesso, leitura ou execução. “*-R*” muda as permissões recursivamente (a tudo o que se encontra dentro do diretório), “*a+r*” indica que todos (a) devem receber (+) permição de leitura (r), e depois metemos a nossa pasta. Podemos rapidamente ver se as permissões estão a funcionar corretamente criando um ficheiro de texto e correndo o comando “*ls -l*”:



Figura 8 Criar um ficheiro de teste

Uma imagem com texto, Tipo de letra, captura de ecrã, preto

Descrição gerada automaticamente

Figura 9 Ficheiro de teste com permissão de leitura para todos

O ficheiro criado tem permissões de leitura e escrita para o dono, e apenas de leitura para o grupo e para todos, tornando isto uma pasta que todos podem aceder.

## User Story 8