

Mário Costa Silva

A75654

Nuno Gabriel Areal

A74714

Trabalho Prático 1

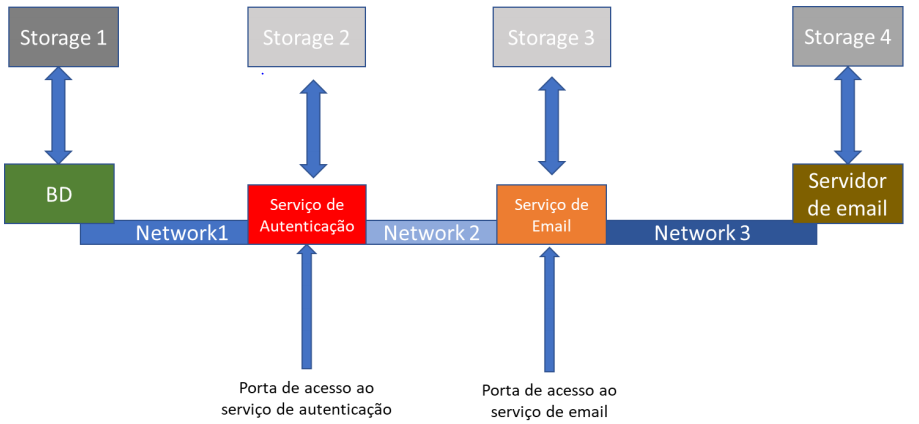
VIRTUALIZAÇÃO DE REDES

Grupo 4

# Introdução

Neste relatório pretendemos expor a nossa resolução do primeiro trabalho prático, realizado no âmbito da disciplina de Virtualização de Redes, que se baseia na implementação de um serviço de autenticação e email através de containers docker. Neste sistema estão então presentes as seguintes componentes:

* Serviço de autenticação, para efetuar login para obtenção de um token
* Base de dados, onde será armazenado as informações dos utilizadores registados e o seu token
* Serviço de email, que permitirá fazer o input da informação a enviar e recetor do mail
* Servidor de email, que efetuará o envio do mail para o destino.



# Serviço de Autenticação

Neste *container* é usada uma imagem **php:7.0-apache** que executará um servidor *Apache* com suporte a *PHP* e, consequentemente, as páginas que desenvolvemos para efetuar a autenticação e registo de utilizadores. Em suma este serviço tem duas funcionalidades, o registo de utilizadores e login. Estes dois processos efetuam operações de inserção e de verificação na base de dados, respetivamente. No caso do login, depois de verificado, é gerado um *token* de sessão que será guardado na base de dados e em cookie no browser para ser depois confirmado quando passar para o serviço de email.

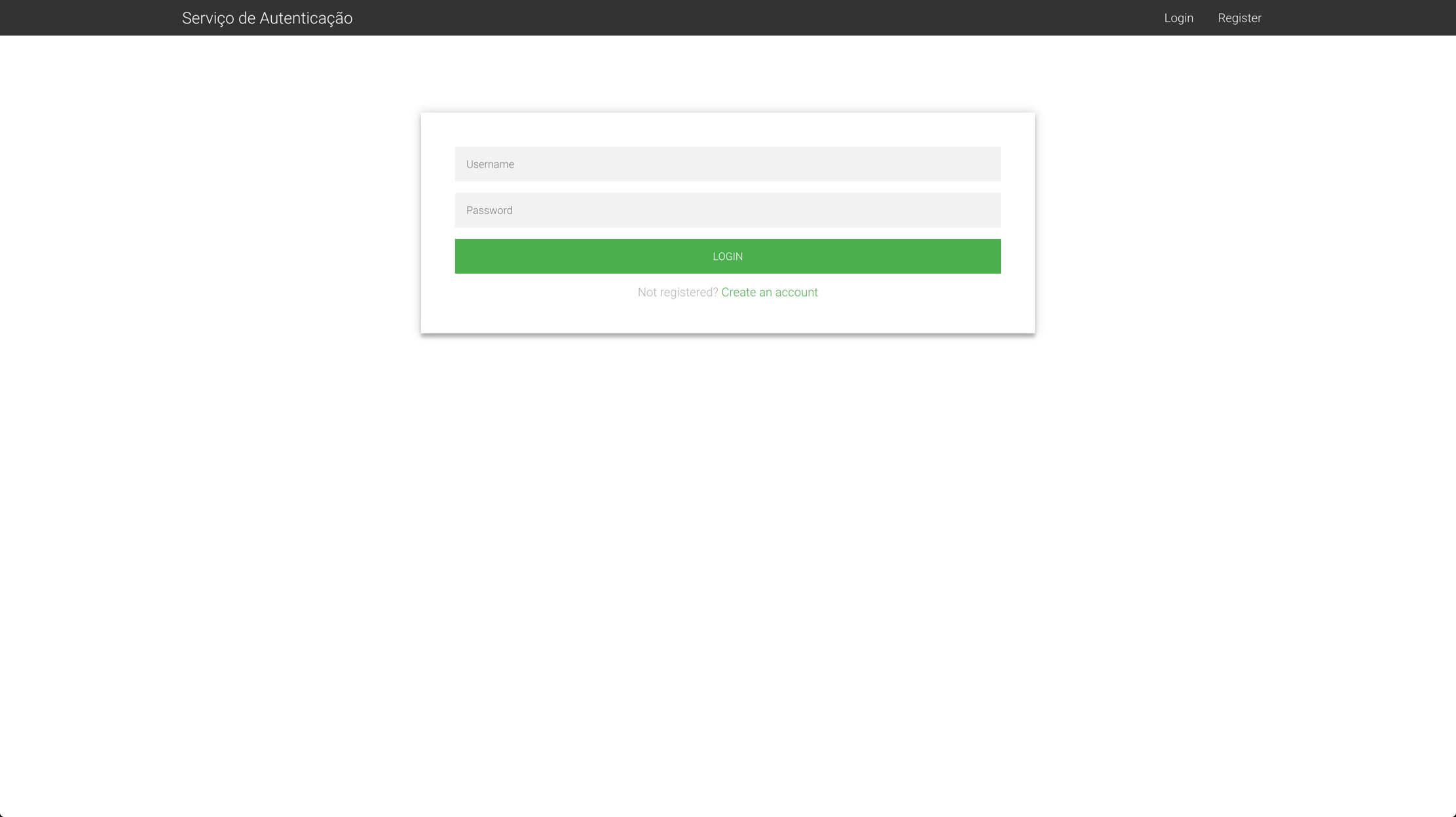


Figura 1 - Página de autenticação

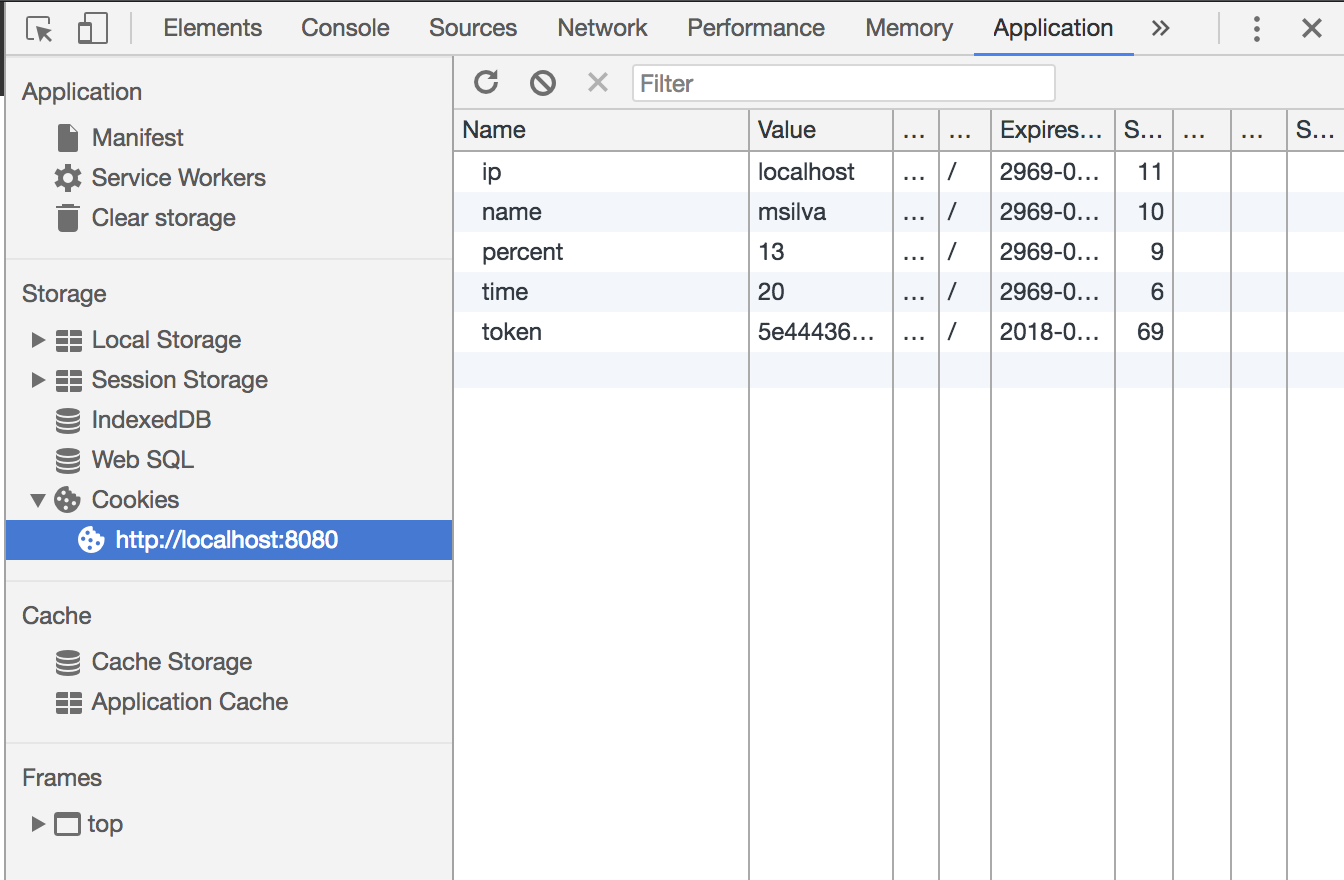


Figura 2 - Token guardado em cookies

# Base de Dados

Neste *container* é usada uma imagem **postgres:8.4** para servir de base de dados do TP e apenas contém uma tabela com os dados do utilizador e *token* de sessão. O esquema pode ser visualizado na imagem abaixo.

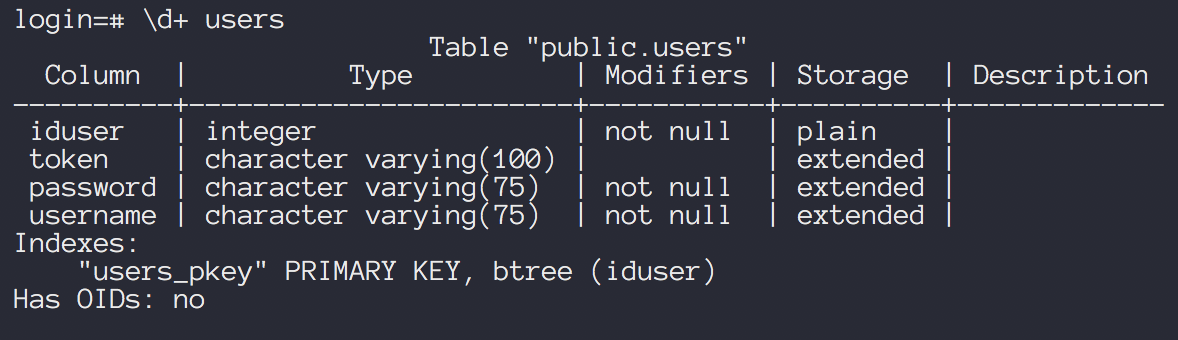


Figura 3 - Tabela de utilizadores

# Serviço de Email

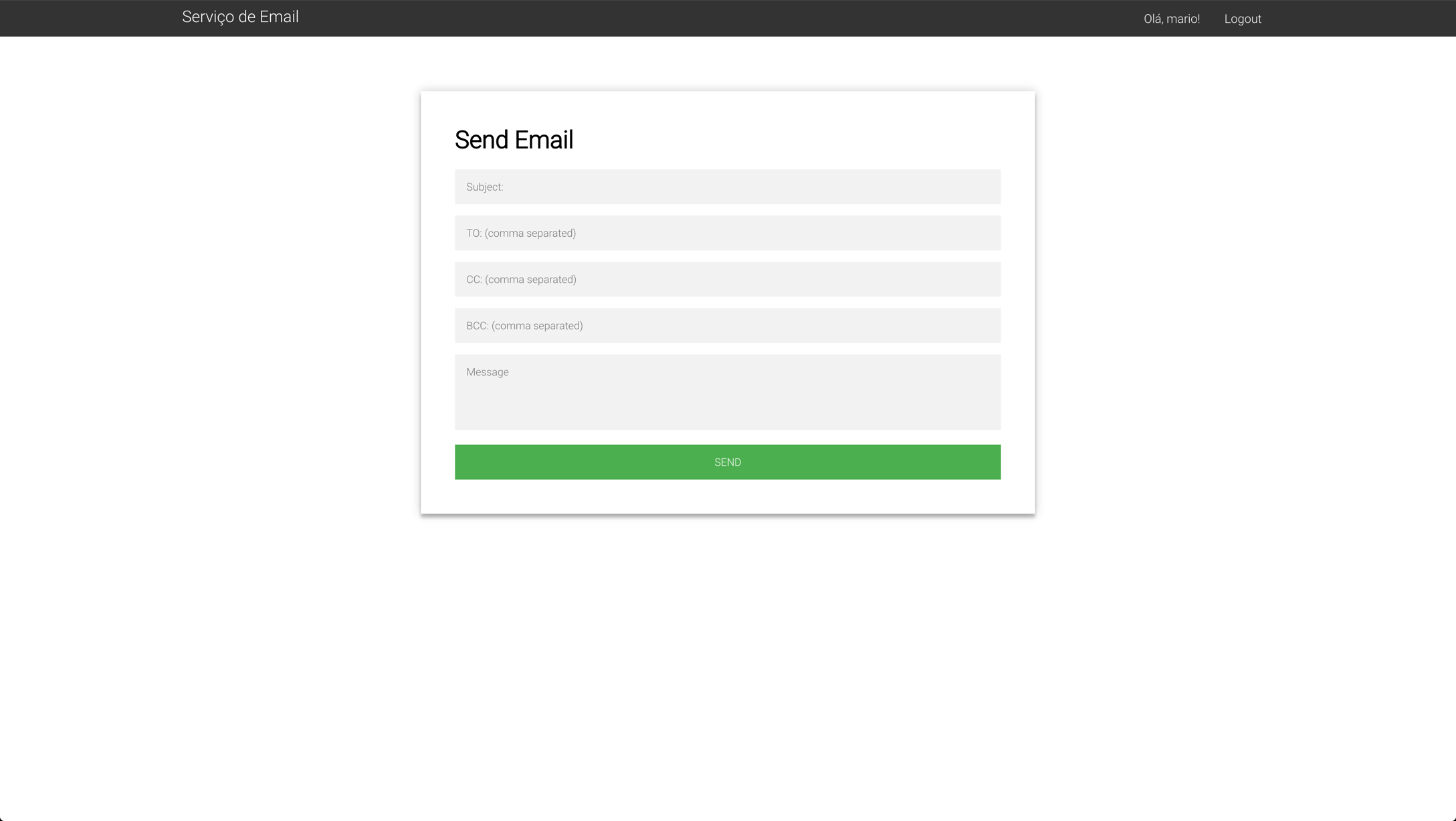
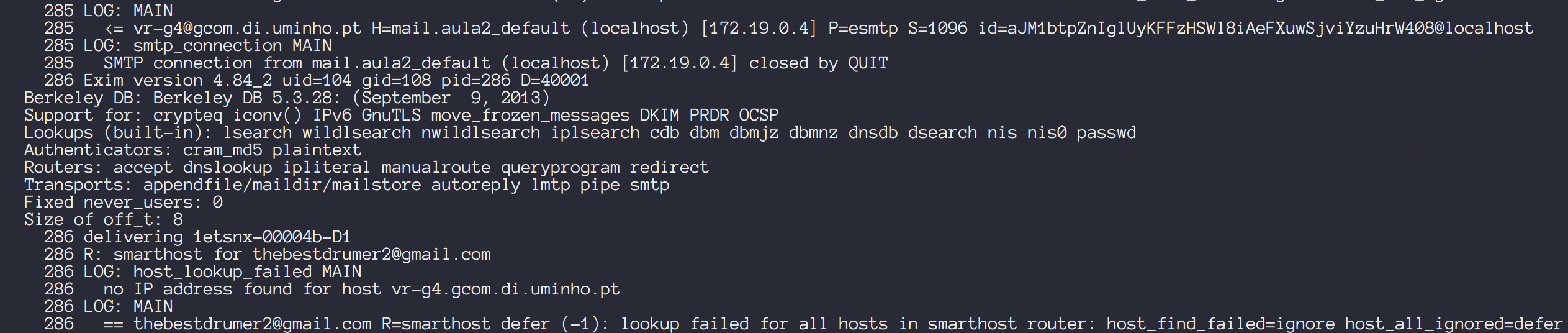
Neste *container* é também usada a imagem **php:7.0-apache**. Ao entrar na página é verificado se existe um *token* em *cookies* e se este é igual ao armazenado na base de dados. Depois de comprovado o utilizador poderá escrever um email que será enviado para o *container* com o servidor de email, que o fará chegar ao destino.

Figura 4 - Página do serviço de email

# Servidor de Email

Neste *container* é usada a imagem **namshi/smtp** que fará o *forwarding* do email para o destino. Apenas é configurada informação do SMARTHOST\_ADDRESS para **vr-g4.gcom.di.uminho.pt**  e do SMARTHOST\_ALIASES para **\*.gcom.di.uminho.pt**. Nos *logs* verificamos que recebe os dados do serviço de email mas não faz chegar ao destino devido a não ser possível resolver o endereço vr-g4.gcom.di.uminho.pt

# Nginx

Neste *container* é usada a imagem **jwilder/nginx-proxy** que permitirá aceder tanto ao serviço de autenticação como ao serviço de email na mesma porta. O acesso aos dois será feito no endereço <http://localhost:8080> mas a este será acrescentado **/auth** no caso de se pretender aceder ao serviço de autenticação e **/mail** no caso do serviço.

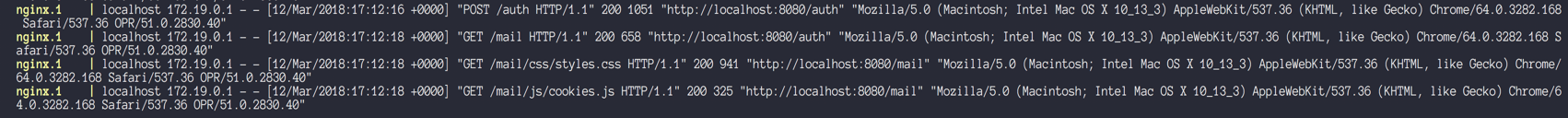
Para colocar este serviço a funcionar foi necessário escrever a configuração manualmente, uma vez que a prometida configuração automática desta imagem não funcionou.

Figura 5 - Logs do Nginx

# Mapeamento a Diretorias

Os *containers* **Serviço de Autenticação, Serviço de Email, Nginx e Servidor de Email** tem os seus volumes mapeados em diretorias físicas escolhidas no docker-compose.ymlpara facilitar o acesso aos ficheiros dos diferentes *containers*.

# Comunicação entre containers

Como objetivo principal do trabalho, a comunicação entre *containers* foi feita escrevendo apenas o nome do *container* com o qual queríamos comunicar ficando o resto a ser tratado pelo *Docker*.

Por exemplo, para a verificação do *token*, o serviço de email tem de contactar o serviço de autenticação. Para tal é efetuado um pedido através do *cURL* com o *token* enviado através do método GET para a página <http://auth/>, sendo *auth* o nome do *container* do serviço de autenticação. Existem duas respostas possíveis, 0 ou 1, caso não seja válido ou caso contrário, respetivamente.

Todas as restantes comunicações necessárias decorrem de forma idêntica como já dissemos, não sendo necessário efetuar o *cURL*, referindo-nos apenas ao *host* com o nome do *container*.