



Universidade do Minho
Escola de Engenharia

METI - Emulação e Simulação de Redes de Telecomunicações

Relatório Final

Grupo 2

Alunos:

João Pedro Costa Bastos - pg57564

Bruno Miguel Fernandes Araújo - pg55806

Docentes:

Adriano Jorge Cardoso Moreira

Bruno Daniel Mestre Viana Ribeiro

José Augusto Afonso

Conteúdo

Lista de Símbolos	iii
Lista de Figuras	iii
Lista de Tabelas	iii
1 Introdução	1
2 Objetivos	1
3 Síntese de conceitos	2
4 Arquitetura geral do sistema	3
4.1 Infraestrutura da Rede no GNS3	3
4.1.1 Encaminhamento	4
4.1.2 Endereçamento	5
4.2 Rede de sensores no Cupcarbon	6
4.2.1 Protocolo e formato das mensagens	7
4.2.2 Gerador de Eventos	8
4.3 Conectividade entre o GNS3 e o CupCarbon.	9
4.3.1 Publisher	9
4.3.2 Subscriber	9
4.4 Base de dados	10
4.5 Servidor Web	11
4.5.1 Serviço de autenticação	11
4.5.2 Gestão de utilizadores	12
4.5.3 Visualização dos Dados	13
5 Ferramentas utilizadas	14
5.1 <i>Software</i>	14
5.2 <i>Hardware</i>	14
6 Conclusão	15

Lista de Símbolos

Acrónimos

DNS Domain Name System

HTTP HyperText Transfer Protocol

IoT Internet of Things

IP Internet Protocol

MQTT Message Queuing Telemetry Transport

OSPF Open Shortest Path First Protocol

Lista de Figuras

1	Arquitetura da rede simulada no GNS3.	4
2	Rede de Sensores simulada no Cupcarbon.	6
3	Modelo lógico da base de dados.	10
4	Interface para o login	11
5	Interface para criar conta	11
6	Interface de Gestão de Utilizadores	12
7	Interface de visualização de informação	13

Lista de Tabelas

1	Tabela com o endereçamento de todas as interfaces de todos os dispositivos.	5
2	Tabela na base de dados que representa o estacionamento.	10
3	Tabela na base de dados que representa os utilizadores.	10

1 Introdução

Este relatório descreve o desenvolvimento e implementação de um sistema de gestão de estacionamento inteligente (Smart Parking), integrando conceitos de emulação e simulação de redes de telecomunicações e utilizando ferramentas como GNS3 e CupCarbon, que permitem simular topologias de rede e sistemas de sensores, respetivamente.

A solução proposta tem como objetivo monitorizar e gerir a ocupação de um parque de estacionamento em tempo real, garantindo acesso a informações precisas por meio de um webserver.

Este sistema interliga diferentes componentes:

- GNS3 emula a topologia de rede, utilizando protocolos de encaminhamento dinâmico como o OSPF (Open Shortest Path First).
- CupCarbon simula os sensores no parque de estacionamento, que recolhem dados sobre a ocupação das vagas.
- Publisher e Subscriber que dão uso de um broker mosquitto MQTT para estabelecer conexão entre o Cupcarbon e o GNS3.
- Base de Dados e Webserver, que armazenam e apresentam as informações do estacionamento aos utilizadores através de uma interface acessível por múltiplos dispositivos.

Adicionalmente, foram implementados mecanismos de autenticação, permitindo que utilizadores registados acessem ao sistema, com permissões específicas para utilizadores normais e administradores. Este relatório detalha a arquitetura, os componentes, e os resultados obtidos, demonstrando a integração entre as tecnologias utilizadas e a eficiência do sistema desenvolvido.

2 Objetivos

Neste projeto tivemos os seguintes objetivos principais:

- O desenvolvimento de uma rede de interligação utilizando a ferramenta GNS3.
- O desenvolvimento de uma rede de sensores utilizando a ferramenta CupCarbon.
- Criação de um serviço de aquisição de dados de IoT com base num protocolo de aplicação adequado (e.g., HTTP, MQTT)
- Estabelecer um meio de ligação entre a rede de sensores e a rede simulada no GNS3.
- Criação de uma base de dados para armazenamento dos dados
- Criação de uma plataforma de gestão de utilizadores.
- Criação de um serviço de autenticação.
- Elaboração de uma interface gráfica para a visualização dos dados de IoT.
- Replicação do serviço de IoT.
- Criação de um serviço de load balancing.
- Configuração do serviço de load balancing para encaminhamento dos dados de IoT com os diversos protocolos para o servidor

3 Síntese de conceitos

- **Plataforma IoT** - É um ambiente integrado que fornece as ferramentas e serviços necessários para conectar, gerenciar e controlar dispositivos inteligentes que fazem parte do ecossistema da Internet das Coisas (IoT). Estas plataformas ajudam a conectar dispositivos físicos (sensores, atuadores, máquinas, etc.) à internet, recolher dados, analisá-los, e permitir que os dispositivos interajam uns com os outros ou com aplicações.
- **Sensores** - São dispositivos que detetam mudanças no ambiente ou em objetos e convertem essas variações em dados digitais, que podem ser transmitidos e analisados. Eles desempenham um papel fundamental na recolha de informações que são utilizadas para monitorizar, controlar e otimizar processos, e são essenciais para aplicações de IoT.
- **Atuadores** - São dispositivos que recebem comandos para executar uma ação física ou mecânica em resposta aos dados processados de sensores ou comandos diretos do sistema. Eles atuam como a "parte de ação" dos sistemas de IoT, enquanto os sensores representam a "parte de coleta de dados".
- **Load balancer** - Load Balancer (Balanceador de Carga) É um componente essencial para escalabilidade e alta disponibilidade em sistemas distribuídos. Ele distribui tráfego de rede ou solicitações de clientes entre vários servidores, otimizando o desempenho, reduzindo o risco de sobrecarga num único servidor e garantindo redundância.
- **HTTP** - É o protocolo principal para comunicação na web, utilizado para transferir dados entre clientes (navegadores, por exemplo) e servidores. Ele segue um modelo request/response, no qual o cliente faz uma solicitação (request) ao servidor, que retorna uma resposta (response). É amplamente utilizado devido à sua simplicidade e suporte global.
- **MQTT** - É um protocolo leve de comunicação projetado para dispositivos com recursos limitados e redes instáveis. Este utiliza um modelo publish/subscribe para troca de mensagens entre dispositivos (clientes) e um broker central, é ideal para aplicações de IoT (Internet das Coisas).
- **Routers** - Os routers são dispositivos responsáveis por direcionar pacotes de dados entre diferentes redes de computadores. Eles atuam como intermediários, definindo o melhor caminho para a transmissão de dados entre dispositivos conectados a redes distintas.
- **Cloud** - Refere-se a um modelo de computação que permite o acesso a recursos de computação (como servidores, armazenamento, software e serviços) através da internet, em vez de depender de hardware físico local. Isso significa que os dados podem ser armazenados e executados em servidores remotos, acessíveis via internet, em vez de em computadores locais.

4 Arquitetura geral do sistema

4.1 Infraestrutura da Rede no GNS3

A topologia ilustrada esquematicamente na Figura 1 é composta por uma estrutura hierárquica interconectada, organizada com os seguintes elementos principais:

Dispositivos:

- Um pc com o sistema operacional Ubuntu (**VisualizadorWebPage**), que será usado para visualizar a pagina web criada pelo webserver.
- Dois contentores ubuntu docker, com os nomes **Subscriber-1** e **Subscriber-2**. O Subscriber-1 contém o broker Mosquitto e o código do subscriber, enquanto que o Subscriber-2 é uma réplica deste.
- Dois contentores ubuntu docker, com o nome **DatabaseWebserver-1** e **DatabaseWebserver-2**. O DatabaseWebserver-1 representa o dispositivo que contém uma base de dados e o webserver que analisa esta , enquanto que o DatabaseWebserver-2 é uma réplica deste.
- Um servidor DNS (**DNS-1**) responsável por resolver nomes dentro da rede ou de fora (reencontra o pedido para o servidor da google com ip **8.8.8.8**).

Estrutura Física:

- Switches: **Switch1**, **Switch3** e **Switch4** realizam a interconexão local entre os PCs e outros dispositivos.
- Routers: Estabelecem comunicação entre os diferentes segmentos da rede interna, temos presente na nossa topologia 5 routers (**R1**, **R2**, **R3**, **R4**, **R5**)
- Ponto de Acesso Externo: A rede possui conexão com um sistema externo, representado pelo **Cloud2**, que pode ser utilizado para troca de informações com dispositivos fora da rede local.

Esta topologia garante a organização eficiente dos dispositivos, com comunicação centralizada entre os elementos internos e conectividade externa através de um ponto de acesso dedicado.

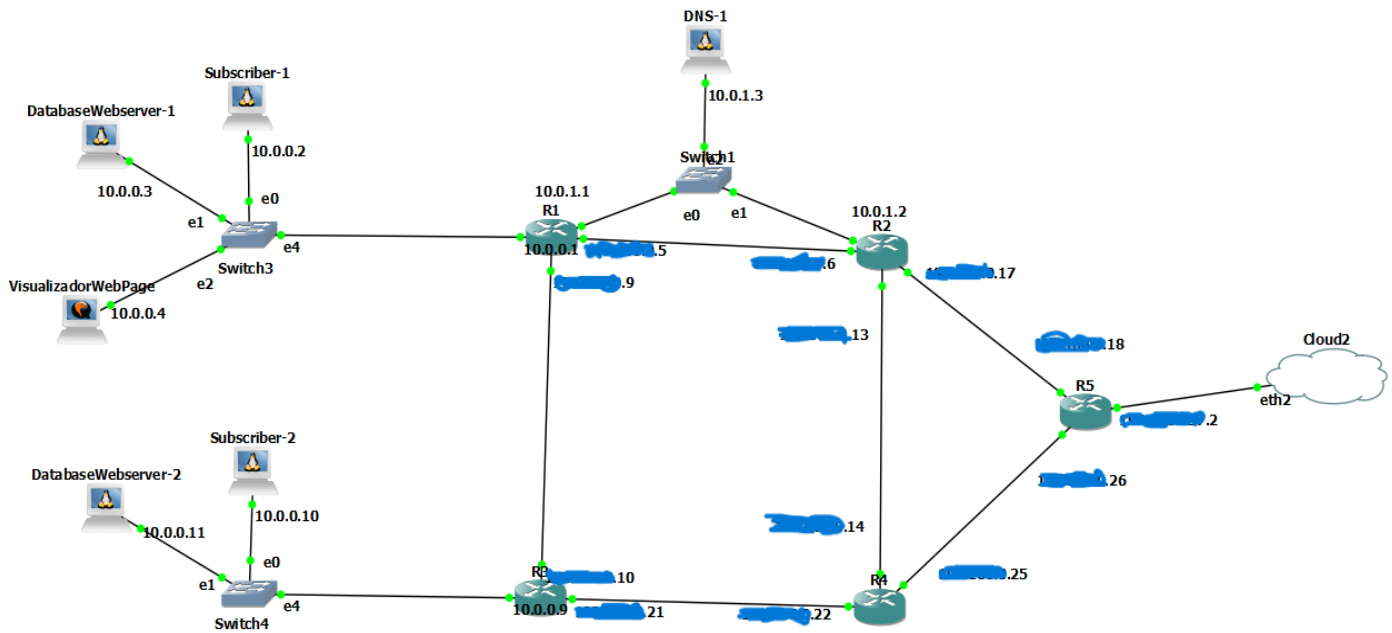


Figura 1: Arquitetura da rede simulada no GNS3.

4.1.1 Encaminhamento

Foi utilizado OSPF um protocolo de encaminhamento dinâmico baseado no algoritmo de estado de ligação (Link-State), amplamente utilizado em redes para calcular o caminho mais curto entre routers. Ele garante uma comunicação eficiente e fiável em redes de grande escala.

Funcionamento do OSPF:

- Descoberta de Vizinhos
- Troca de Informações de Estado de Link (LSA)
- Construção da Base de Dados de Estado de Link
- Cálculo do Caminho Mais Curto
- Atualizações Incrementais

4.1.2 Endereçamento

Dispositivo	Endereço IP	Máscara
R5	.2	255.255.255.0
R5	.18	255.255.255.252
R5	.26	255.255.255.252
R4	.25	255.255.255.252
R4	.14	255.255.255.252
R4	.22	255.255.255.252
R2	.17	255.255.255.252
R2	.13	255.255.255.252
R2	.6	255.255.255.252
R2	10.0.1.2	255.255.255.248
R1	.5	255.255.255.252
R1	.9	255.255.255.252
R1	10.0.1.1	255.255.255.248
R1	10.0.0.1	255.255.255.248
R3	.10	255.255.255.252
R3	.21	255.255.255.252
R3	10.0.0.9	255.255.255.248
DNS-1	10.0.1.3	255.255.255.248
Subscriber-1	10.0.0.2	255.255.255.248
DatabaseWebserver-1	10.0.0.3	255.255.255.248
VisualizadorWebPage	10.0.0.4	255.255.255.248
Subscriber-2	10.0.0.10	255.255.255.248
DatabaseWebserver-1	10.0.0.11	255.255.255.248

Tabela 1: Tabela com o endereçamento de todas as interfaces de todos os dispositivos.

4.2 Rede de sensores no Cupcarbon

Utilizando a ferramenta Cupcarbon, simulámos uma rede de sensores que representa um parque de estacionamento com 10 lugares, bem como carros e os seus respectivos trajetos, representados por 'Mobiles' e 'routes', disponíveis nesta ferramenta.

Nesta rede, temos 10 sensores direcionais, agrupados em conjuntos de 5 e associados a um dispositivo multihop, responsável por redirecionar a informação de ocupação ou desocupação de um lugar para o gateway. Existe também um placar (atuador) que receberá e exibirá a informação sobre o número de lugares vazios.

Por fim, há um gateway que tem conhecimento do estado de todos os lugares do estacionamento, sendo responsável por registar a informação das ocupações e desocupações num ficheiro e por enviar o número de lugares livres para o placar. A arquitetura geral da rede de sensores está ilustrada esquematicamente na Figura 2.

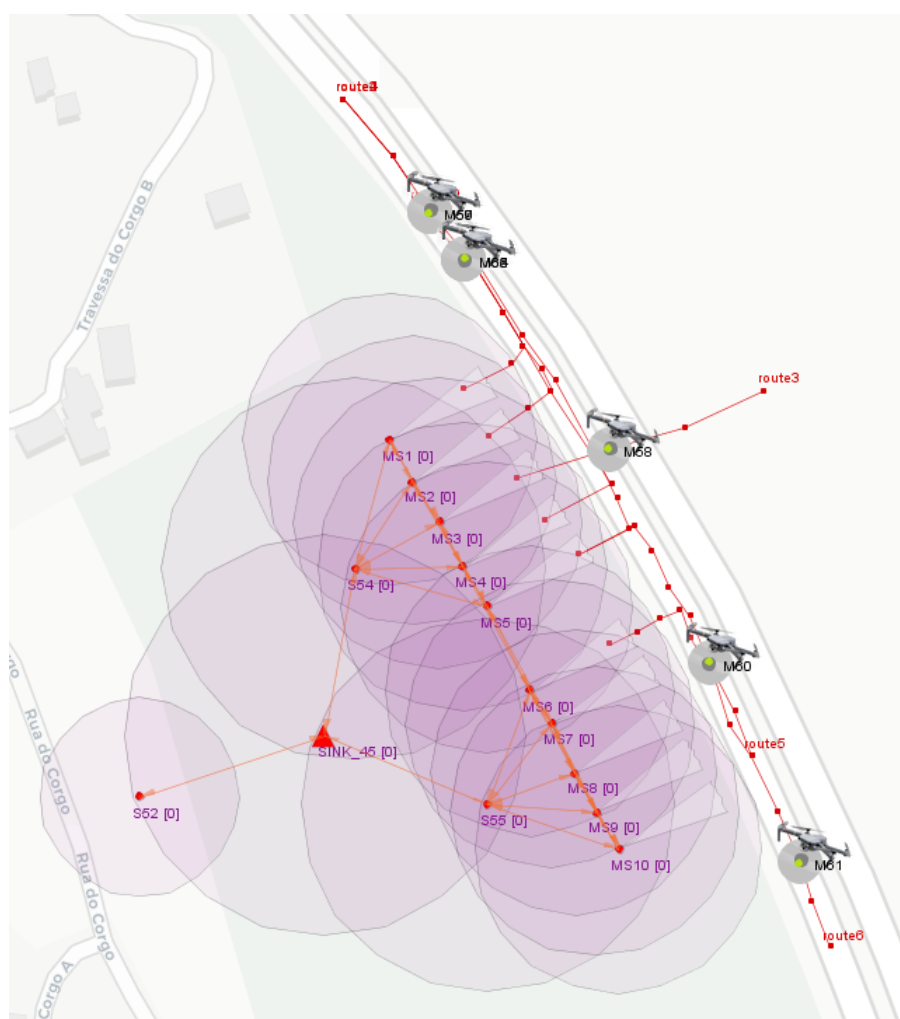


Figura 2: Rede de Sensores simulada no Cupcarbon.

4.2.1 Protocolo e formato das mensagens

As mensagens seguem um protocolo de leitura por eventos, sendo que temos presente um parque de estacionamento, os eventos são as entradas e saídas dos veículos nos lugares deste. O formato das mensagens enviadas entre os dispositivos desta rede de sensores, varia dependendo de quem é que envia, o sensor ou o gateway.

Para a comunicação **Sensor** → **Multihop** → **Gateway** temos:

type	id	id_gateway	v	tpacote	tenvio
------	----	------------	---	---------	--------

- **type**: Tipo da mensagem, neste caso é sempre 'S' pois vem do sensor.
- **id**: Id do sensor que enviou a mensagem.
- **id_gateway**: Id do destino, o id do gateway.
- **v**: Valor da alteração do espaço que foi feita no estacionamento, 0 se o lugar foi desocupado e 1 se foi ocupado.
- **tpacote**: Timestamp da criação do pacote.
- **tenvio**: Timestamp do envio do pacote.

Para a comunicação **Gateway** → **Multihop** → **Sensor** ou **Gateway** → **Placar** temos:

type	id	id_gateway	valor	t
------	----	------------	-------	---

- **type**: Tipo da mensagem, 'R' para notificar o sensor sobre uma reserva no local, 'C' para informar o sensor que houve uma colisão, exigindo o reenvio da mensagem, e 'P' para o placar.
- **id**: Id do destino, id do sensor ou do placar.
- **id_gateway**: Id de onde veio a mensagem, id do gateway.
- **valor**: Se for enviado para o sensor é 1 (reserva) para este depois dar mark(altera a cor do led), se for para o placar é um valor entre 0 a 10 , dependendo do número de lugares livres.
- **t**: Timestamp de quando esta foi enviada.

4.2.2 Gerador de Eventos

O gerador de eventos é uma funcionalidade extra que desenvolvemos em Python, para facilitar o processo de criar diferentes exemplos de caminhos que os veículos possam tomar. Resumidamente este programa vai abrir os ficheiros dos caminhos destes veículos e vai alterar os pontos dos trajetos, ou seja ele vai gerar "eventos". Infelizmente não deu para expandir mais neste conceito e manteve-se o mesmo desde a fase B.

Além desta funcionalidade, este tem outras, analisaremos com maior detalhe uma a uma:

Opção 1:Exemplos.

- Exemplo predefinido em que temos uma reserva num lugar que se tornou livre e poderá haver uma colisão entre 1-5.(Este é o default, o CupCarbon vem com este exemplo)
- Exemplo predefinido em que o estacionamento acaba por ficar cheio e consequentemente não será possível reservar um novo lugar.

Opção 2:Associar um lugar a um veículo.

Existem rotas previamente definidas para cada lugar, permitindo indicar para qual lugar o veículo deve ir. Caso seja inserido o valor 0, o veículo permanece parado e não se desloca para nenhum lugar.

Opção 3:Definir uma rota manualmente.

Com base nas coordenadas de diferentes pontos, é traçado um caminho que os conecta. Além disso, é possível decidir se o trajeto será repetido em loop após a sua conclusão.

Opção 4:Alterar o número de mensagens para haver uma reserva.

As reservas ocorrem após o gateway receber um número definido de mensagens, conforme especificado no seu script. Com esta opção pode alterar este número, sem ter de ir diretamente ao script altera-lo.

Opção 5:Alterar a presença ou não de tratamento de colisões.

Os scripts dos dispositivos do CupCarbon têm em consideração as colisões. Com esta opção, poderá escolher se deseja ou não o tratamento de colisões. Isto é possível porque as linhas de código relacionadas com o tratamento de colisões ficarão ou não comentadas, dependendo do que deseja.

4.3 Conectividade entre o GNS3 e o CupCarbon.

A informação sobre o estado do estacionamento, a ser enviada para a base de dados no GNS3, encontra-se no gateway do nosso projeto no CupCarbon. Assim, optámos por recorrer ao broker MQTT **Mosquitto** e desenvolvemos um **publisher** e um **subscriber**.

4.3.1 Publisher

O Publisher é o componente responsável por enviar mensagens para um tópico num broker MQTT. Nesse caso, o gateway (simulado pelo sistema CupCarbon) publica informações de sensores simulados. Ele age como a fonte de dados para o sistema MQTT.

Funcionamento:

- **Recolha de Dados:** O gateway, recolhe informações dos sensores simulados sendo estas referentes a ocupação dos lugares do parque e escreve-as num ficheiro.
- **Publicação de Dados:** O publisher lê esse ficheiro e envia mensagens para um tópico com o nome de "parking" referentes a atualização da ocupação dos lugares.
- **Conexão:** O publisher vai publicar a sua informação para o tópico MQTT "parking", cuja subscrição vai ser feita por parte de um "subscriber" na topologia simulada no GNS3 estabelecendo a conexão.

4.3.2 Subscriber

O Subscriber é o componente dentro da rede (um contentor ubuntu docker) que recebe as mensagens publicadas no tópico pelo Publisher. Ele adquire os dados enviados para armazená-los.

Funcionamento:

- **Assinatura do Tópico:** O Subscriber é configurado para assinar o tópico usado pelo Publisher, sendo este, "parking". Assim, ele recebe automaticamente todas as mensagens enviadas para o tópico.
- **Receção de Dados:** Quando o Publisher publica uma mensagem, o Subscriber recebe os dados em tempo real.
- **Processamento:** Armazenada a informação numa base de dados SQL localizada noutro contentor ubuntu docker, que será utilizada posteriormente.
- **Conexão:** O Subscriber está configurado na rede interna do GNS3, conectando-se ao broker MQTT via o seu endereço IP.

4.4 Base de dados

A **Base de Dados** foi programada em SQL, esta tem duas tabelas com a seguinte estrutura:

Variável	Tipo	Descrição
spot_id	Inteiro	Identificador do lugar no parque de estacionamento (chave primaria)
status	Inteiro	Se o lugar se encontra livre (0) ou ocupado (1)
last_update	Data	Indica o dia e a hora da última atualização do espaço.

Tabela 2: Tabela na base de dados que representa o estacionamento.

Variável	Tipo	Descrição
user_id	Inteiro	Identificador do utilizador(chave primária)
username	String	Nome do utilizador
password_hash	String	Palavra-passe da conta do utilizador
user_type	Enum	Se o utilizador é apenas um usuário (normal) ou um administrador (admin).
create_at	Data	Indica o dia e a hora da criação da conta.

Tabela 3: Tabela na base de dados que representa os utilizadores.

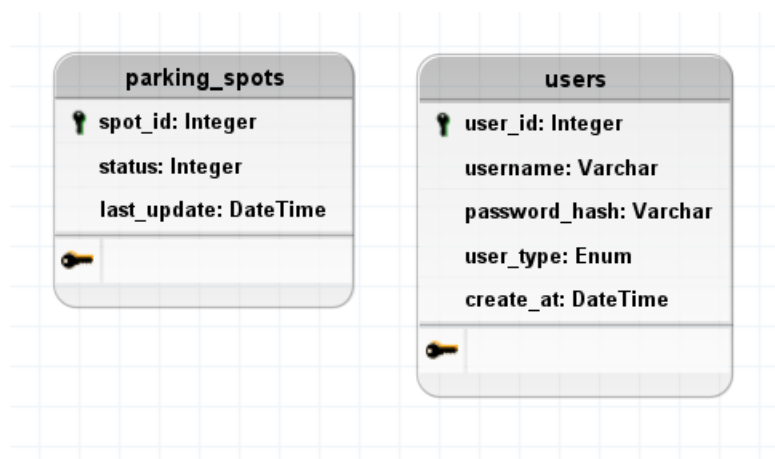


Figura 3: Modelo lógico da base de dados.

4.5 Servidor Web

4.5.1 Serviço de autenticação

O **sistema de autenticação** permite o acesso seguro ao webserver através de um formulário simples e intuitivo. Os utilizadores podem inserir o seu nome de utilizador e palavra-passe para efetuar login. Caso não tenham uma conta, é disponibilizada a opção de registo com um link para criar uma nova conta.

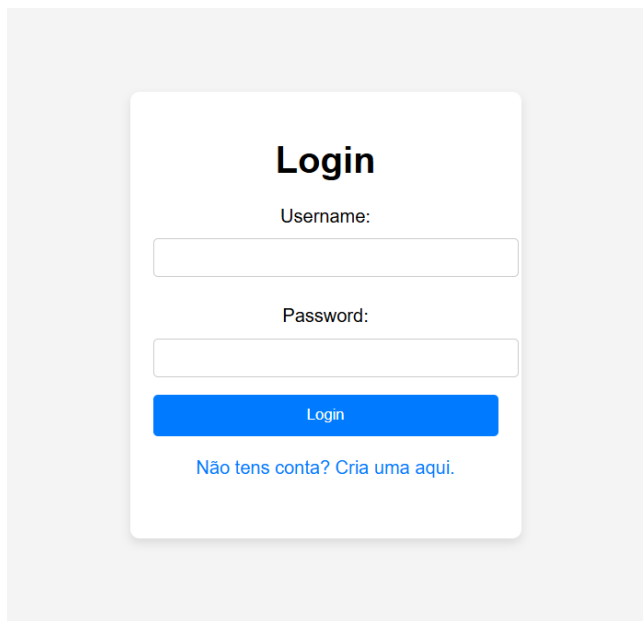
A screenshot of a web application's login interface. It features a white rectangular form with rounded corners centered on a light gray background. At the top of the form, the word "Login" is displayed in a bold, black, sans-serif font. Below this, the label "Username:" is positioned above a white text input field with a thin gray border. Further down, the label "Password:" is positioned above another white text input field with a thin gray border. Below the password field is a solid blue button with the word "Login" in white, centered text. At the bottom of the form, there is a link in blue text that reads "Não tens conta? Cria uma aqui."

Figura 4: Interface para o login

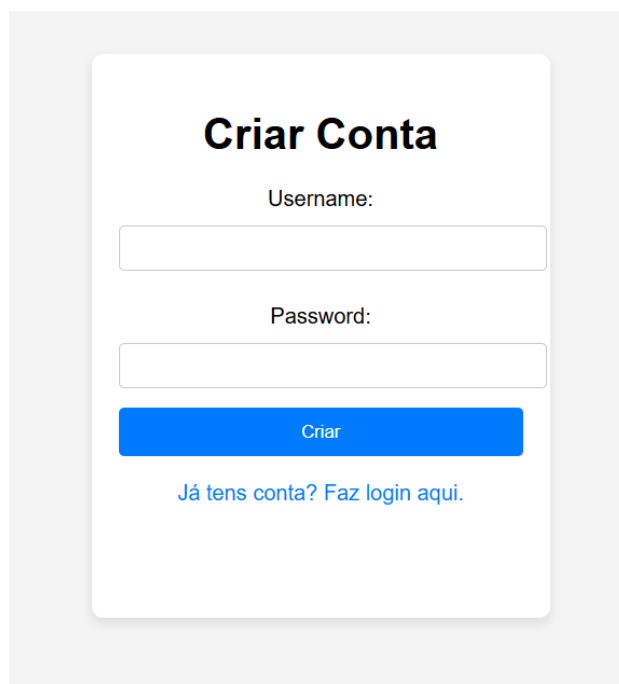
A screenshot of a web application's "Create Account" interface. It features a white rectangular form with rounded corners centered on a light gray background. At the top of the form, the words "Criar Conta" are displayed in a bold, black, sans-serif font. Below this, the label "Username:" is positioned above a white text input field with a thin gray border. Further down, the label "Password:" is positioned above another white text input field with a thin gray border. Below the password field is a solid blue button with the word "Criar" in white, centered text. At the bottom of the form, there is a link in blue text that reads "Já tens conta? Faz login aqui."

Figura 5: Interface para criar conta

4.5.2 Gestão de utilizadores

A funcionalidade de gestão de utilizadores permite aos administradores controlar os acessos ao sistema. A interface apresenta uma tabela com a lista de utilizadores, os seus tipos (normal ou administrador) e ações disponíveis.

Ações disponíveis:

- Promover: Elevar um utilizador normal ao nível de administrador.
- Despromover: Rebaixar um administrador para utilizador normal.
- Remover: Eliminar um utilizador da base de dados.

Além disso, a interface inclui botões para efetuar logout ou navegar para o estado do estacionamento, permitindo uma gestão integrada e fácil de usar.

Bem-vindo, Bruno.

Gestão de Utilizadores

Username	Type	Actions
Bruno	admin	<button>Despromover</button>
Luis	normal	<button>Promover</button> <button>Remover</button>
Carlos	normal	<button>Promover</button> <button>Remover</button>
João	admin	<button>Despromover</button>

Logout Estado do Estacionamento

Figura 6: Interface de Gestão de Utilizadores

4.5.3 Visualização dos Dados

A interface de visualização do estado do estacionamento permite aos utilizadores monitorizar, em tempo real, a ocupação das vagas de estacionamento.

Representação visual:

- Cada vaga é representada por um quadrado numerado.
- Cor verde indica vaga livre, enquanto outras cores (e.g., vermelho) indicariam vaga ocupada.

Atualização de dados:

- A informação é atualizada automaticamente a cada segundo, garantindo precisão.
- Existe também um botão para atualização manual caso o utilizador o deseje.

Navegação e ações adicionais:

- Botões para realizar logout ou navegar para a gestão de utilizadores, mantendo uma experiência de utilização simples e eficiente.

Este sistema garante que a ocupação do estacionamento é sempre visível e acessível de forma intuitiva.



Figura 7: Interface de visualização de informação

5 Ferramentas utilizadas

Apresentamos as ferramentas utilizadas, listadas conforme a sua categoria: *Software* ou *Hardware*.

5.1 *Software*

As ferramentas a nível de software usamos as seguintes:

- Programa **GNS3**, para a simulação de redes, permite o teste de cenários e de configurações antes da implementação prática.
- Programa **Oracle VM Virtualbox**, para a utilização da virtual machine como servidor local para estabelecer ligação á internet.
- Plataforma **Discord**, para a comunicação e partilha de ficheiros entre o grupo.
- Plataforma **OverLeaf**, para a elaboração de relatórios em \LaTeX .
- Programa **CupCarbon**, para a simulação e monitorização da rede de sensores.
- Programa **Visual Studio Code**, para a programação de códigos complementares .
- Programa **MySQL Workbench**, para a criação de bases de dados.
- Programa **brModelo**, para a criação do modelo lógico da base de dados.
- Programa **Wireshark**, para a análise do tráfego entre dispositivos da rede simulada no GNS3.

5.2 *Hardware*

Ao nível de hardware tivemos presentes 2 computadores , um para cada estudante.

6 Conclusão

Durante o desenvolvimento deste projeto, enfrentamos diversas dificuldades, sendo as principais relacionadas à ligação com a internet e à comunicação entre o CupCarbon e o GNS3. Além disso, não foi possível implementar o load balancer no GNS3, o que limitou parcialmente nossos objetivos iniciais.

Grande parte do tempo dedicado ao desenvolvimento deste projeto foi consumido com problemas de ligação. O computador parecia ter uma mente própria e, por vezes, simplesmente perdia a ligação, bloqueando o processo e exigindo um reinício. Isto resultou numa enorme perda de tempo, que poderia ter sido investido no desenvolvimento das funcionalidades extras solicitadas.

Apesar dos desafios, o trabalho realizado proporcionou aprendizagens significativas. Conseguimos criar e configurar topologias no GNS3 e desenvolver simulações de sistemas de sensores no CupCarbon. Combinando esses conhecimentos, foi possível projetar uma plataforma de smart parking funcional, capaz de simular um sistema real aplicável no mundo atual.

Esses avanços demonstram que, mesmo diante das limitações encontradas, os conceitos estudados e aplicados têm grande potencial para o desenvolvimento de soluções tecnológicas inovadoras e práticas.