

# Avaliação do Desempenho de Carregadores de Veículos Elétricos em Condições de Temperatura Controlada

Relatório Final do Projeto Integrador

Vasco Ferreira da Rocha Melo



Licenciatura em Engenharia Informática e Computação

**Tutor na U.Porto:** Rolando Martins  
**Orientador na empresa:** Pedro Pascoal

Data: 2024/2025

# Conteúdo

<b>1</b>	<b>Lista de abreviações e siglas</b>	<b>2</b>
<b>2</b>	<b>Glossário</b>	<b>3</b>
<b>3</b>	<b>Introdução</b>	<b>4</b>
3.1	Enquadramento . . . . .	4
3.2	Objetivos e resultados esperados . . . . .	4
3.3	Estrutura do relatório . . . . .	4
<b>4</b>	<b>Metodologia utilizada e principais atividades desenvolvidas</b>	<b>6</b>
4.1	Metodologia utilizada . . . . .	6
4.2	Intervenientes, papéis e responsabilidades . . . . .	7
4.3	Atividades desenvolvidas . . . . .	7
<b>5</b>	<b>Desenvolvimento da solução</b>	<b>10</b>
5.1	Requisitos . . . . .	10
5.2	Arquitetura e tecnologias . . . . .	11
5.2.1	Componentes de hardware . . . . .	11
5.2.2	Componentes de software . . . . .	12
5.3	Solução desenvolvida . . . . .	14
5.3.1	Sensores de temperatura DS18B20 e circuito . . . . .	15
5.3.2	Interface gráfica . . . . .	17
5.3.3	Comunicação . . . . .	18
5.3.4	Base de dados . . . . .	18
5.3.5	Migração de dados e scheduling . . . . .	19
5.3.6	Dashboard . . . . .	20
5.3.7	Instalação e configuração . . . . .	21
5.4	Validação . . . . .	23
<b>6</b>	<b>Conclusões</b>	<b>25</b>
6.1	Resultados alcançados . . . . .	25
6.2	Lições aprendidas . . . . .	25
6.3	Trabalho futuro . . . . .	25
6.4	Agradecimentos . . . . .	25

## 1 Lista de abreviações e siglas

- **SGEV** - Laboratório de Redes Elétricas Inteligentes e Veículos Elétricos
- **CPES** - Centro de Sistemas de Energia
- **INESC TEC** - Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência
- **VEs** - Veículos Elétricos
- **CA** - Corrente Alternada
- **GUI** - Interface Gráfica de Usuário
- **RPI** - Raspberry Pi
- **EVSE** - \*\*Equipamento de Carregamento de Veículos Elétricos.
- **Base de dados** -
- **IoT** - Internet das coisas. Refere-se às tecnologias que conectam objetos físicos à internet, permitindo que eles coletem e troquem dados automaticamente para operar de forma mais eficiente e inteligente.
- **GPIO** - General Purpose Input/Output, refere-se a pinos de entrada/saída que podem ser utilizados para controlar dispositivos eletrônicos.

## 2 Glossário

- **stakeholders** - Partes interessadas que influenciam ou são influenciadas pelo projeto
- **GitHub e GitLab** - Plataformas de hospedagem de código-fonte e controle de versões
- **MainPC** - Computador principal utilizado para controlar os testes de temperatura realizados. É nele que está presente a GUI e a base de dados onde o Grafana está configurado para coletar e visualizar dados.
- **Grafana** - Plataforma de visualização e análise de dados que permite criar dashboards interativos e monitorar métricas em tempo real.
- **Dashboard** - Painel de controle para visualização dos dados
- **TCP/IP** - Protocolo de Controle de Transmissão/Protocolo de Internet
- **Base de dados** -
- **EVSE** - Equipamento de Carregamento de Veículos Elétricos.
- **software** -
- **hardware** -

### **3 Introdução**

Este relatório tem como objetivo apresentar todo o trabalho desenvolvido durante o estágio curricular, onde foi desenvolvido um protocolo de teste de desempenho de carregadores de energia.

#### **3.1 Enquadramento**

O estágio curricular foi realizado no Laboratório de Redes Elétricas Inteligentes e Veículos Elétricos (SGEV) do Centro de Sistemas de Energia (CPES) do Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência (INESC TEC). Para realização do estágio teve como orientador da instituição o engenheiro Pedro Pascoal e como orientador da FEUP/FCUP o professor Rolando Martins.

O SGEV é um espaço dedicado à pesquisa e desenvolvimento de tecnologias inovadoras na área dos sistemas de energia, com principal foco na eficiência energética, mobilidade elétrica, redes elétricas inteligentes ou energias renováveis. Com a crescente demanda por soluções de carregamento de veículos elétricos (VEs), a eficiência e a segurança dos carregadores são aspectos cruciais. O INESC TEC atualmente desenvolve carregadores CA que operam em ambientes externos sob exposição de intempéries, podendo ocasionar uma elevação significativa na sua temperatura interna, especialmente no verão. Diante desse cenário, é fundamental realizar testes para avaliar o impacto das variações de temperatura externa sobre o desempenho dos carregadores. Esses testes permitem compreender como a temperatura afeta o funcionamento e a eficiência dos carregadores, assegurando a confiabilidade dos produtos em condições reais de operação.

#### **3.2 Objetivos e resultados esperados**

Desenvolver e implementar um sistema de medição de temperatura com sensores DS18B20 e Raspberry Pi 3, que será utilizado para monitorar o desempenho térmico de carregadores de veículos elétricos em diferentes condições de temperatura, simuladas em uma câmara climática.

Ao final do estágio, espera-se que o protocolo de medição esteja implementado e validado, permitindo a coleta de dados precisos sobre o desempenho térmico dos carregadores, os quais serão apresentados em uma dashboard.

#### **3.3 Estrutura do relatório**

O relatório está estruturado da seguinte forma:

- **Capítulo 1: Introdução** - Apresenta o contexto do estágio, os objetivos e a estrutura do relatório;

- **Capítulo 2: Metodologia** - Descreve a metodologia utilizada durante o estágio curricular, os intervenientes (seu papel e responsabilidade) e atividades desenvolvidas;
- **Capítulo 3: Desenvolvimento** - Detalha o desenvolvimento do protocolo , requisitos (funcionais e não funcionais), arquitetura e tecnologias do sistema, solução desenvolvida (dividida nos diferentes módulos) e validação;
- **Capítulo 4: Conclusão** - Apresenta os resultados alcançados, lições aprendidas e trabalho futuro.

## **4 Metodologia utilizada e principais atividades desenvolvidas**

Esta secção descreve a metodologia adotada durante o estágio, o que inclui o desenvolvimento iterativo, ferramentas de comunicação e colaboração, *stakeholders* e suas responsabilidades e as atividades que foram realizadas durante o estágio.

### **4.1 Metodologia utilizada**

O estágio foi conduzido com base em uma abordagem de desenvolvimento iterativo, caracterizada pela definição de objetivos diários e pela execução das atividades no ambiente do laboratório. Ao final de cada jornada, era realizado o recebimento de *feedback* por parte do orientador, o que possibilitou ajustes contínuos e aprimoramentos no trabalho desenvolvido.

Adicionalmente, algumas tarefas foram realizadas fora do horário regular de estágio, com o intuito de dar continuidade ao progresso do projeto. Essas atividades incluíram ajustes pontuais em tarefas previamente desenvolvidas, elaboração de documentação e realização de pesquisas complementares relacionadas ao tema.

Ao longo do dia, eram realizadas, em média, três reuniões principais. A primeira acontecia pela manhã, com o objetivo de realizar um ponto de situação do projeto, fornecer *feedback* sobre o trabalho eventualmente desenvolvido durante a semana e definir os principais objetivos a serem alcançados no decorrer do dia. A segunda reunião ocorria no início da tarde, momento em que se discutia o progresso das atividades realizadas durante a manhã. A terceira reunião era realizada ao final do dia, com o propósito de revisar as tarefas executadas e planejar as próximas etapas do projeto para a semana seguinte. Nessa reunião, também eram definidos os trabalhos complementares a serem realizados fora do horário de estágio.

O orientador esteve sempre disponível para oferecer suporte e esclarecimentos, tanto durante os dias presenciais no laboratório quanto fora do horário regular, por meio de diferentes canais de comunicação.

Em casos mais técnicos ou que exigiam uma análise mais aprofundada, também havia sempre a possibilidade de consultar outros engenheiros do laboratório, que tiveram sempre disponibilidade para ajudar.

Toda a comunicação no âmbito do laboratório foi realizada presencialmente. Nos casos em que se fez necessária a comunicação à distância, utilizou-se o e-mail institucional como meio oficial.

Como o projeto era muito abrangente, a comunicação foi fundamental para o alinhamento das expectativas e progresso do trabalho. Ao longo do estágio, todos os requisitos que foram estabelecidos foram ajustados sempre às necessidades da instituição como também às expectativas do estagiário. Desta forma, todo o projeto foi desenvolvido de maneira colaborativa e adaptativa, garantindo

que as metas fossem alcançadas de forma eficaz.

Todo o código desenvolvido durante o estágio foi inicialmente armazenado em um repositório pessoal no GitHub, sendo posteriormente compartilhado com a equipe por meio de um repositório no GitLab.

#### **4.2 Intervenientes, papéis e responsabilidades**

Este estágio envolveu a colaboração de diferentes participantes, cada um com responsabilidades específicas:

- Orientador da instituição:
  - Engenheiro Pedro Pascoal: Responsável por orientar o estagiário por meio de supervisão contínua e dar *feedback* construtivo em relação ao trabalho desenvolvido. Também auxiliou na integração do estagiário às instalações da instituição.
- Tutor da FEUP/FCUP:
  - Professor Rolando Martins: Responsável por orientar o estagiário do lado mais académico representando os objetivos da unidade curricular e na elaboração do relatório final.
- Cliente:
  - Engenheiro Pedro Pascoal: Responsável por representar os interesses do INESC TEC, estabelecendo os requisitos e expectativas do projeto. Também foi responsável por validar o trabalho desenvolvido.
- Estagiário:
  - Vasco Melo: Responsável por desenvolver as diferentes tarefas propostas pelo cliente e tendo em conta a qualidade e o tempo proposto para a conclusão das mesmas. Também é responsável por relatar todo o estágio através dos diferentes elementos de avaliação propostos pela unidade curricular.

#### **4.3 Atividades desenvolvidas**

Durante o estágio, foram desenvolvidas diferentes atividades, que incluiram planeamento do projeto, pesquisa de mercado, e desenvolvimento de protótipos. Ao longo que novos requisitos foram surgindo, as atividades foram ajustadas para atender às novas demandas do projeto. Desta forma, as atividades desenvolvidas podem ser descritas da seguinte forma:

1. Planeamento do projeto e pesquisa de mercado:
  - . Apresentação ao estagiário das instalações e seus equipamentos;

- . Estabelecimento de um plano de ação para o desenvolvimento do projeto;
  - . Foi realizada uma pesquisa de mercado com base nos equipamentos disponíveis e os objetivos esperados, que foram sendo estabelecidos/ajustados ao longo do projeto.
2. Raspberry Pi 3 e Sensores DS18B20:
    - . Configuração do Raspberry Pi 3;
    - . Desenvolvimento de um sistema de leitura de dados com os sensores DS18B20.
  3. Criação de uma GUI e sistema de comunicação entre a RPI e o MainPC:
    - . Levantamento de requisitos para a interface gráfica e de tecnologias no mercado;
    - . Desenvolvimento de diferentes protótipos para a GUI;
    - . Estabelecimento de um protocolo SSL/TLS com a RPI;
    - . Criação de um protocolo de controlo de teste de temperatura com a Raspberry Pi 3 e o MainPC.
  4. Montar um sistema de gestão para armazenamento dos testes:
    - . Estudo sobre os dados a guardar;
    - . Criação do schema para a base de dados;
    - . Implementação de um sistema de controlo da base de dados.
  5. Implementação de um sistema de migração de base de dados:
    - . Criação de um sistema de migração de base de dados entre o MainPC e a RPI;
    - . Criação de um sistema de migração de base de dados entre a RPI e o EVSE.
  5. Comunicação com o EVSE:
    - . Estabelecimento da comunicação via SSH entre a RPI e o EVSE.
  7. Criar uma dashboard para visualização dos dados:
    - . Estudo sobre requisitos e funcionalidades da dashboard;
    - . Análise das tecnologias disponíveis para a implementação;
    - . Teste das diferentes tecnologias;
    - . Implementação da dashboard com as funcionalidades definidas.
  8. Sistema de schedule de migração de testes:

- . Estabelecimento de um sistema de schedule entre o MainPC e a RPI;
- . Estabelecimento de um sistema de schedule entre a RPI e o EVSE.

9. Sistema de estabilização e comunicação com a câmara termal:

- . Estudo sobre as funcionalidades da câmara termal;
- . Estabelecimento da comunicação entre a RPI e a câmara termal via serial port;
- . Investigação sobre diferentes formulações matemáticas de deteção de estabilização de temperatura da câmara termal;
- . Implementação de um sistema de deteção de estabilização de temperatura com a câmara termal.

10. Documentação :

- . Elaboração de diferentes diagramas UML para o sistema desenvolvido;
- . Criação de documentação técnica para o sistema desenvolvido.

11. Bootstrapping:

- . Elaboração de um ficheiro dockerfile para facilitar na instalação da componente MainPC;
- . Elaboração de um ficheiro .bash e um sistema ansible para facilitar a instalação da componente RPI.

Para melhor organização do trabalho desenvolvido, foi elaborado um gantt chart, que pode ser visto na Figura \_\_\_\_\_.  
TODO: Colocar o gantt chart aqui

## 5 Desenvolvimento da solução

O desenvolvimento da solução para o projeto foi um processo que envolveu várias etapas que foram estabelecidas com o tempo. Desta forma a solução foi sendo adaptada conforme os novos requisitos. O desenvolvimento não só envolveu componente software como também hardware, o que levou com que o desenvolvimento do protocolo em ambiente laboratorial fosse essencial para garantir a eficácia do sistema. Como houve parte do desenvolvimento que foi realizada fora do laboratório, também foi desenvolvido uma componente que simula o hardware implementado. A solução teve em conta não só a usabilidade do mesmo, como também as dificuldades e limitações encontradas durante o processo de desenvolvimento.

### 5.1 Requisitos

Nesta secção irão ser identificados todos os requisitos funcionais e não funcionais estabelecidos ao longo do estágio:

Requisitos funcionais:

- Leitura de dados de temperatura: O estagiário deverá desenvolver um sistema de leitura de dados de temperatura;
- Interface gráfica: A solução deve constar com uma interface gráfica que permitirá ao usuário controlar o teste;
- Vizualização dos dados obtidos em tempo real: o sistema desenvolvido deve permitir ao usuário visualizar os dados obtidos do teste em tempo real;
- Base de dados: Todos os valores obtidos do teste devem ser guardados numa base de dados local;
- Comunicação: O sistema deve suba guardar a comunicação entre os dispositivos (entre a RPI e o EVSE e entre a RPI e o MainPC).

Requisitos não funcionais:

- Desempenho: O sistema deve ser capaz de processar dados em tempo real sem atrasos significativos;
- Escalabilidade: O sistema deve ser capaz de suportar um aumento no volume de dados sem comprometer o desempenho;
- Reabilidade: O sistema deve garantir a integridade e a disponibilidade dos dados durante a operação;
- Segurança: O sistema deve garantir o padrão mínimo de segurança dos dados e a proteção contra acessos não autorizados;
- Manutenibilidade: O sistema deve ser projetado para facilitar a manutenção e atualização ao longo do tempo.

Restrições:

- Todo o sistema desenvolvido deve estar limitado ao material e recursos disponíveis no laboratório;
- O sistema deve ser testado em ambiente laboratorial para garantir fidelidade dos resultados;
- O sistema deve ser capaz de operar em diferentes ambientes sem perder a sua funcionalidade.

## 5.2 Arquitetura e tecnologias

Ao longo do estágio, enumera-se tecnologias e arquiteturas foram levadas em conta para corresponder aos requisitos do Cliente. Como os requisitos foram mudando ao longo do projeto, a estrutura do projeto acabou por ser ajustada. Nesta secção irão ser abordadas as diferentes arquiteturas e tecnologias abordadas e os seus motivos para serem (ou não) selecionadas.

### 5.2.1 Componentes de hardware

Os componentes de hardware utilizados foram limitados à disponibilidade de recursos e materiais no laboratório. Para a implementação da solução foram utilizados uma Raspberry Pi 3 e 4 sensores DS18B20.

Para a validação do protocolo, foi utilizado um carregador EVSE desenvolvido pelo Laboratório e uma câmara termal (incubadora) IMP 400 da HERA-TERM (disponível no Laboratório).

A Raspberry Pi 3 é um microcomputador de baixo custo e alta versatilidade, ideal para projetos de prototipagem e automação. Outra vantagem é a sua capacidade de se conectar a diversos dispositivos e sensores, facilitando a integração em projetos de IoT. Foi instalado na RPI o sistema operacional Raspbian, que fornece uma interface amigável e suporte a diversas bibliotecas necessárias para o desenvolvimento do projeto, nomeadamente um conjunto de drives e configurações para utilizar a entrada GPIO da Raspberry. A entrada GPIO foi essencial para a comunicação entre os sensores DS18B20 e a Raspberry Pi, permitindo a leitura e o controle dos dispositivos conectados. Outra vantagem da Raspberry é a integração de uma placa de rede no sistema, o que facilitou na comunicação com outros dispositivos e a troca de dados entre eles. Para toda a configuração e estabelecimento da comunicação entre os dispositivos, foi utilizado o livro [5].

Os sensores DS18B20 são utilizados para medir a temperatura e são conectados à Raspberry Pi através da entrada GPIO, permitindo a leitura precisa dos dados de temperatura. Foi selecionado este sensor, pelos seguintes motivos:

- Versatibilidade: O sensor apresenta um fio comprido o que ajuda no posicionamento dos sensores dentro da câmara termal. Além disso o sensor é encapsulado, o que o torna resistente a ambientes adversos.

- Leitura da temperatura digitalmente: A entrada GPIO da RPI só lê valores digitais e não valores analógicos (muito comum em sensores de temperatura). O sensor DS18B20 lê a temperatura digitalmente, o que facilita na obtenção dos dados (não necessita de um módulo analógico).
- Custo: Os sensores são acessíveis e disponíveis no mercado.

O carregador EVSE desenvolvido pelo INESCCTEC é um carregador AC (alternate current), monofásico (230VAC), Potência nominal dos carregadores =  $7.4\text{kW} = 32\text{A}$ , que tem instalado diversos sensores que permitem medir valores de corrente, tensão, frequência e temperatura (os valores de temperatura remetem ao circuito). Pelo fato de ler tensão e corrente, pode-se calcular potência e o consumo de potência (W) no tempo nos dar o consumo ou então, carregamento em energia (W/h), o que é essencial para ver a eficiência do carregamento do carro elétrico ao longo do tempo. Dentro do carregador tem uma Raspberry Pi que faz o gerenciamento de controle do carregamento e possibilita a comunicação do EVSE com uma base de dados do INESC para envio dos dados de carregamento.

A IMP 400 é uma câmara termal que permite controlar a temperatura interna, garantindo condições controladas para a realização dos testes. Tem a capacidade de manter a temperatura interna constante entre os  $5^{\circ}\text{C}$  e os  $70^{\circ}\text{C}$  (o que acaba por abranger o intervalo de temperaturas sentidas em Portugal). Internamente ela possui uma sensor de temperatura que permite saber a temperatura da câmara durante do teste. Por fim, a câmara termal possui ainda uma entrada RS 232 que permite a comunicação serial port com a Raspberry Pi. Devido a uma limitação da própria câmara termal, apenas comandos de query estão disponíveis (permite por exemplo obter a temperatura do sensor interno da própria câmara), mas comandos de set (por exemplo alterar a temperatura da câmara) não estão disponíveis. Para uma melhor compreensão sobre as funcionalidades da câmara termal foi utilizada o seu manual [2] e para questões mais específicas a equipa de suporte da HERATERM.

Além disso, também foram utilizados outros componentes como resistências, cabos e conectores para garantir a comunicação entre os dispositivos e a Raspberry Pi.

### 5.2.2 Componentes de software

Os componentes de software utilizados foram os principais alvos no estudo de arquiteturas e tecnologias. O sistema desenvolvido é composto por um conjunto de módulos que trabalham em conjunto para garantir a funcionalidade do sistema. Os principais componentes de software utilizados foram:

- . Sistema Operacional: O sistema operacional utilizado na RPI foi o Raspbian, que é uma distribuição do Linux otimizada para a Raspberry Pi. O Raspbian fornece uma interface amigável e suporte a diversas bibliotecas necessárias para o desenvolvimento do projeto;

- . Linguagem de Programação: A linguagem de programação utilizada foi o Python, que é amplamente utilizada em projetos de automação e IoT devido à sua simplicidade e versatilidade. O Python possui diversas bibliotecas que facilitam a comunicação com os sensores e a manipulação dos dados obtidos;
- . Base de Dados: Para armazenar os dados obtidos dos sensores, foi utilizado o MySQL, que é um sistema de gerenciamento de base de dados relacional amplamente utilizado. O MySQL foi escolhido devido à sua robustez, escalabilidade e facilidade de uso, além de ser compatível com o Python através da biblioteca MySQL Connector. Outro fator importante é facto de ser o principal sistema de base de dados utilizado no laboratório, o que facilitou a integração com outros sistemas já existentes;
- . Interface Gráfica: Para criar a interface gráfica do sistema, foi utilizada a biblioteca PyQt, que é uma biblioteca de desenvolvimento de interfaces gráficas para Python;
- . Dashboard: Para a visualização dos dados obtidos em tempo real, foi utilizado o Grafana, uma plataforma de visualização de dados que permite criar dashboards interativas e personalizadas open source.
- . Bootstrapping: Para garantir a inicialização correta do sistema, foi utilizado o assemble para a instalação e configuração dos componentes de software necessários na RPI e dockerfile com integração devcontainer no MainPC.

Além destes componentes, foram ainda utilizados outros softwares que ajudaram no desenvolvimento do protocolo, nomeadamente:

- . Git: Para o controle de versão do código-fonte e facilitar o envio de ficheiros do computador de desenvolvimento para a RPI;
- . Visual Studio Code: Como IDE para o desenvolvimento do código-fonte;
- . MySQL Workbench: Para a gestão da base de dados MySQL, permitindo a criação e manipulação de tabelas, consultas e outros objetos das bases de dados;
- . RealVNC: Para o acesso remoto à Raspberry Pi, permitindo a visualização e controlo da interface gráfica do sistema a partir de outro computador;
- . Drawio e Mermaid: Para a criação de diagramas que ajudam na visualização da arquitetura do sistema e no planeamento do projeto;
- . Fritzing: Software de prototipagem eletrónica que foi utilizado para desenhar o circuito de ligação dos sensores DS18B20 à Raspberry Pi, facilitando a compreensão do circuito e a sua implementação.

### 5.3 Solução desenvolvida

Um teste de temperatura corresponde a um processo de obtenção de dados de temperatura durante um período de tempo que podem apresentar características como a temperatura inicial da câmara termal e a temperatura final da mesma. O protocolo tem como objetivo principal garantir as condições necessárias para a realização do teste de temperatura, além de ter em conta outros fatores como a comunicação entre os dispositivos, a leitura dos dados dos sensores e a visualização dos dados obtidos.

Para ajudar na interpretação dos dados obtidos do teste, foi utilizado Mean Absolute Error (MAE) e Root Mean Square Error (RMSE) para calcular o instante onde a temperatura lida pelo sensor de controlo da câmara termal e o sensor de controlo estabilizaram.

A solução desenvolvida pode ser dividida em duas componentes principais:

- . **RPI:** Corresponde à parte do sistema que é responsável por controlar os sensores de temperatura DS18B20, obter os dados de temperatura da câmara termal e do EVSE e comunicar com o MainPC sobre o estado do teste;
- . **MainPC:** Corresponde à parte do sistema que é responsável por servir de interface para o usuário, através de uma interface gráfica. Além disso é responsável por comunicar com a RPI sobre as ações a serem realizadas, como iniciar ou parar o teste. É no MainPC que está instalado uma dashboard (Grafana) que permite visualizar os dados obtidos em tempo real e uma base de dados (MySQL) que armazena os dados obtidos do teste.

Ambas as componentes têm uma base de dados MySQL que armazena os dados obtidos do teste, permitindo a consulta e análise posterior dos dados. O schema de ambas as bases de dados é o mesmo. Para conseguir ver os resultados obtidos em tempo real, ambas as componentes têm um sistema de migração de dados através de scheduling. Para tal ser possível, a comunicação entre a RPI e o MainPC é essencial.

Todos subsistemas estão integradas com paralelismo via threads, o que permite que as diferentes componentes do sistema possam funcionar de forma independente e simultânea, além de ser fácil de desativá-las via .env. No MainPC é ainda utilizado sinais (via pyqtSignal) para a thread da interface gráfica comunicar com as restantes threads do MainPC.

Desta forma, o esquema do protocolo pode ser traduzido no seguinte diagrama:

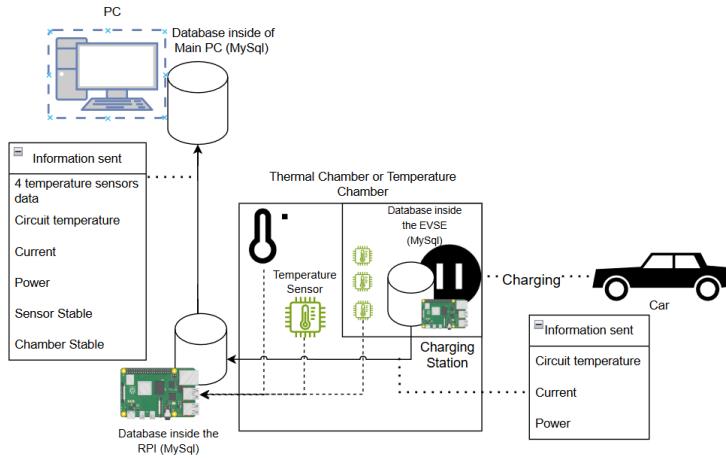


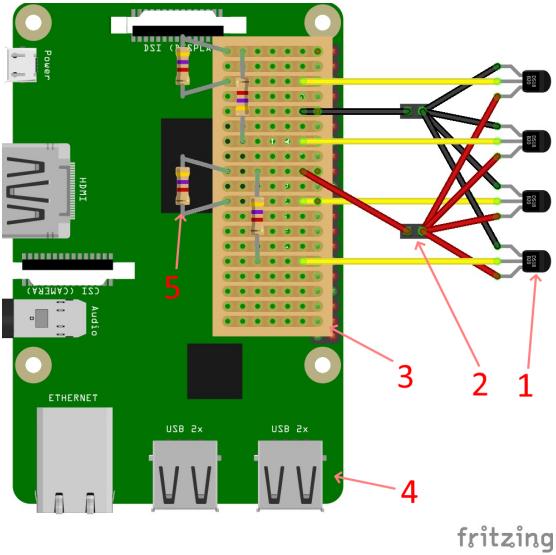
Figura 1: Diagrama do protocolo de comunicação entre a RPI e o MainPC

Nas próximas secções, serão abordadas as diferentes componentes essenciais do sistema, assim como sua instalação.

Outro ponto importante a ter em conta é que o sistema é modular, que permite facilmente desativar features, alterar configurações (IP's, portas, etc.) e alterar parâmetros. Para tal, foi utilizados ficheiros .env que permitem definir as variáveis de ambiente do sistema, facilitando a configuração. Mais informações sobre podem ser encontradas na documentação geral do protocolo [DOCUMENTAÇÃO].

### 5.3.1 Sensores de temperatura DS18B20 e circuito

Para a leitura dos dados de temperatura dos sensores DS18B20, foi desenvolvido um circuito que permite a conexão dos sensores à Raspberry Pi através da entrada GPIO. Como o circuito é possível de ser soldado, permite uma maior robustez e fiabilidade na conexão dos sensores.



**Legenda:**

1. Sensor DS18B20
2. Conector
3. Busboard de ligação
4. RPI
5. Resistência de  $4.7\text{ k}\Omega$

fritzing

Figura 2: Diagrama do circuito de ligação dos sensores DS18B20 à RPI

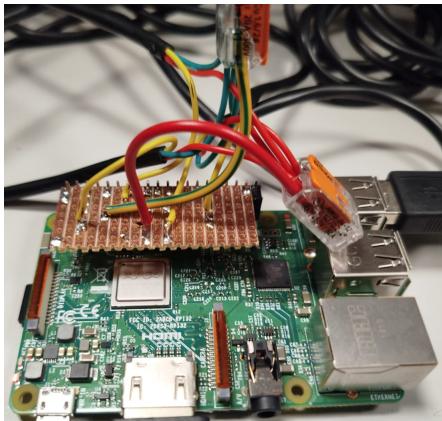


Figura 3: Vista de cima do circuito de ligação dos sensores DS18B20 à RPI

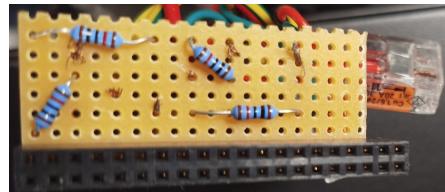


Figura 4: Vista das resistências do circuito de ligação dos sensores DS18B20 à RPI

Este circuito permite a conexão de mais sensores, contudo decidimos apenas instalar 4 sensores DS18B20, um dos sensores é utilizado para medir a temperatura da câmara termal (servindo como controlo da temperatura da câmara) e os outros três são posicionados no interior do EVSE em diferentes locais, para medir a temperatura do carregador em diferentes pontos.

Para a sua implementação de hardware e software (leitura dos sensores), foi utilizada os sites [3] e [4], a datasheet do sensor DS18B20 [1].

### 5.3.2 Interface gráfica

A interface gráfica foi desenvolvida utilizando a biblioteca PyQt e tem como objetivo principal ajudar o utilizador a controlar o teste de temperatura, permitindo criar (estabelecer tempo, temperatura do teste e descrição do teste), iniciar o teste de temperatura, parar o teste forçadamente e prolongar a duração do teste. A interface foi desenvolvida para ser fácil de implementar outras funcionalidades no futuro.



Figura 5: Interface gráfica do protocolo

O botão Connection permite ao utilizador estabelecer a conexão com a RPI (com os parametros de IP e porta definidos no ficheiro .env). Após a conexão ser estabelecida, o utilizador pode criar um teste de temperatura, através do botão Create a Test, aparecendo em pop-up janela () .



Figura 6: Pop Up de criação de um teste de temperatura

Se o formulário for preenchido corretamente (o tempo do teste ser maior que 0), o teste é criado comunicado para a RPI (dos sensores DS18B20). Mais informações sobre o processo de comunicação em Subsubseção 5.3.3.

Para comunicar as interações do utilizador com as outras threads do MainPC (comunicação e scheduling), foi utilizado o pyqtSignal (nativo do PyQt), que permite enviar sinais entre as diferentes threads do MainPC.

### 5.3.3 Comunicação

A comunicação entre os dois dispositivos é essencial para a criação/controlo dos testes de temperatura, mas como também principal via para manter a sincronização do sistema de schedule dos dois dispositivos (MainPC e RPI). Também ajuda para notificar o sistema schedule do MainPC quando o sensor de controlo e o sensor da câmara termal estabilizaram.

A comunicação entre a RPI e o MainPC é feita através de um protocolo SSL/TLS para garantir a segurança dos dados transmitidos. Para tal existe a necessidade de criar um certificado SSL/TLS que é utilizado para estabelecer a conexão segura entre os dois dispositivos.

Apesar da comunicação funcionar peer to peer, para facilitar a implementação a RPI comporta-se como um servidor (abre uma porta onde espera por uma conexão) e o MainPC como um cliente (conecta-se à RPI). Após a conexão ser estabelecida, e a autenticação por certificados validada a RPI não aceita mais nenhuma conexão, acabando por estabelecer um sistema peer to peer.

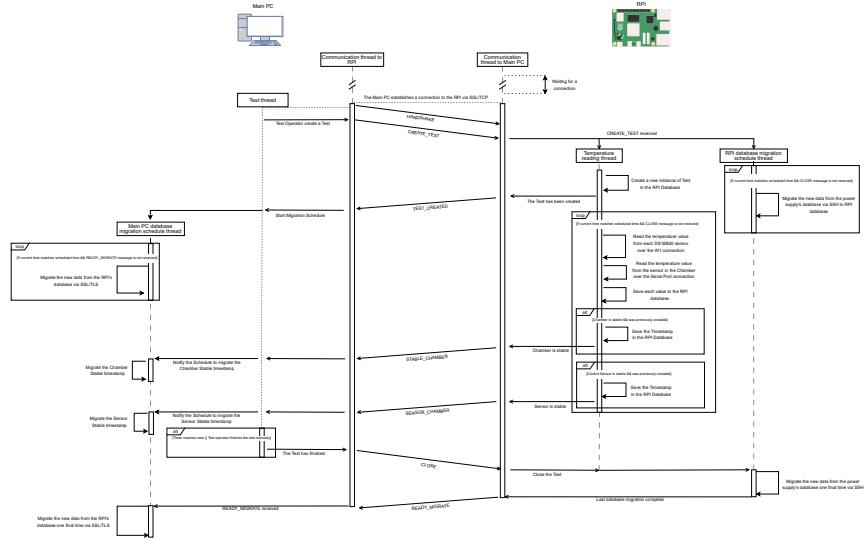


Figura 7: Diagrama do protocolo de comunicação entre a RPI e o MainPC

Mais informações sobre o processo de comunicação entre a RPI e o MainPC podem ser encontradas na documentação geral do protocolo [DOCUMENTAÇÃO].

### 5.3.4 Base de dados

Para salvaguardar a integridade dos dados obtidos do teste de temperatura, foi configurada uma base de dados MySQL no MainPC e na RPI com o mesmo

schema. O schema tem como base as tabelas da base de dados do EVSE (que já existia no laboratório) e foi adaptado para incluir a estrutura teste. Desta forma o schema da base de dados é o seguinte:

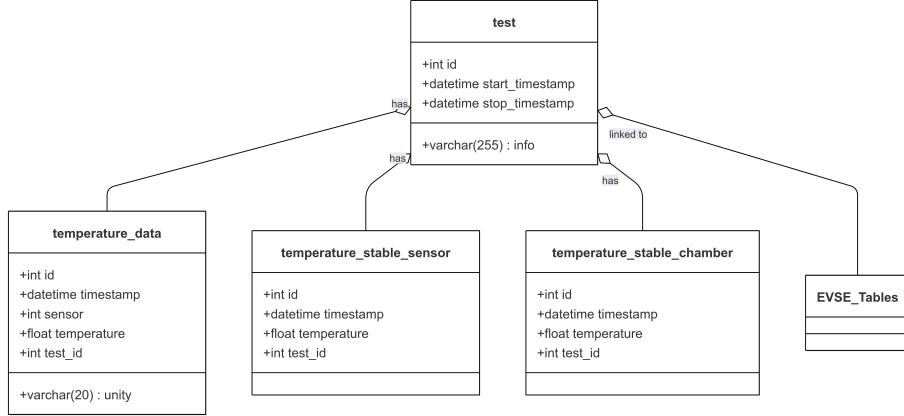


Figura 8: Diagrama de classes da base de dados do protocolo

As base de dados são acedidas através de uma biblioteca de Python chamada MySQL Connector, que permite a conexão e manipulação dos dados da base de dados MySQL. A tabela *test* é a tabela principal que armazena os dados do teste de temperatura, enquanto que a tabela *temperature<sub>data</sub>* é a tabela que armazena os dados de temperatura obtidos dos sensores DS18B20. As tabelas *temperature<sub>stable</sub>,sensor* e *temperature<sub>stable</sub>,chamber* são tabelas auxiliares que armazenam o momento que a temperatura lida pelo sensor de controlo e o sensor da câmara termal estabilizou, respetivamente.

### 5.3.5 Migração de dados e scheduling

Scheduling é um sistema de agendamento de tarefas que permite a execução de tarefas em momentos específicos ou em intervalos regulares.

Como os testes de temperatura podem durar várias horas, foi necessário implementar um sistema de migração de dados entre as três componentes do sistema (RPI, MainPC e EVSE), para permitir a visualização dos dados obtidos em tempo real. Para tal, é importante garantir a sincronização entre os diferentes componentes através da comunicação entre eles. É utilizado um sistema de scheduling através de threads para realizar a migração de dados entre as diferentes componentes do sistema.

O sistema de scheduling na RPI é responsável por obter os dados do EVSE. O sistema de scheduling no MainPC é responsável por obter os dados da presentes na RPI (valores de temperatura dos sensores DS18B20, sensor de controlo da câmara termal e dados do EVSE).

Além disso, após a RPI notificar o MainPC que o sensor de controlo da câmara termal e o sensor de controlo estabilizaram, o sistema de scheduling do

MainPC é responsável na próxima trigger realizar a migração desses dados para a base de dados do MainPC.

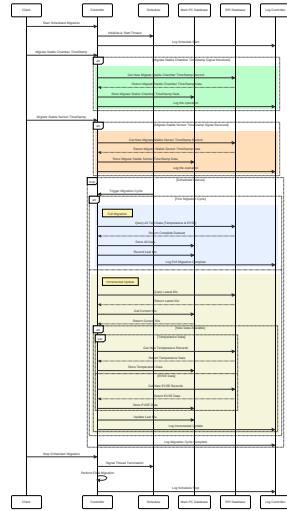


Figura 9: Diagrama de sequência do sistema de scheduling do MainPC

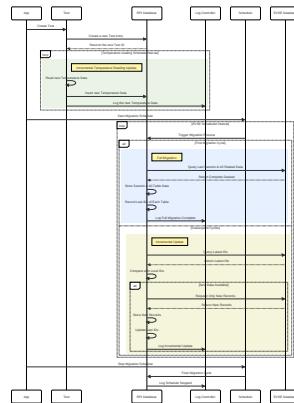


Figura 10: Diagrama de sequência do sistema de scheduling da RPI

### 5.3.6 Dashboard

Uma dashboard é uma interface gráfica que permite visualizar dados de uma forma interativa e personalizada. O objetivo principal da dashboard é permitir ao utilizador visualizar os dados obtidos do teste de temperatura em tempo real e do carregador em tempo real.

O Grafana não só permite a visualização dos dados obtidos em tempo real de forma gráfica, como também a criação automática de anotações para mostrar os momentos onde a temperatura estabilizou (tanto do sensor de controlo da câmara termal como do sensor de controlo do EVSE).

Além disso, é possível inserir o valor teórico de potência do carregador para calcular a eficiência do carregamento do carro elétrico ao longo do tempo, permitindo uma análise mais detalhada do desempenho do carregamento.

Outro fator importante é a fácil troca de visualização dos testes, permitindo ao utilizador selecionar o teste que deseja visualizar e analisar os dados obtidos.



Figura 11: Dashboard de visualização dos dados do teste

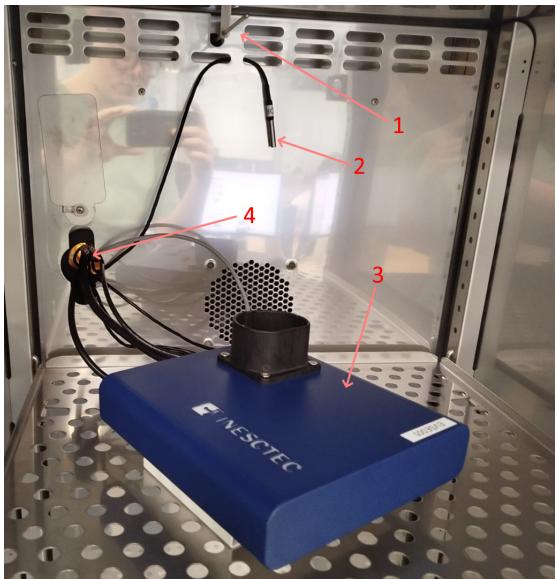
#### Legenda:

1. Exemplo de gráfico de visualização dos dados obtidos do teste
2. Dropdown de seleção do teste (via `test_id`)
3. Input box para o valor teórico de potência do carregador
4. Seleção do intervalo de tempo para visualização dos dados
5. Anotação automática de estabilização da temperatura do sensor da câmara termal
6. Anotação automática de estabilização da temperatura do sensor de controlo

#### 5.3.7 Instalação e configuração

Uma das tarefas desenvolvidas durante o estágio foi o Bootstrapping do sistema, para permitir a sua rápida instalação e configuração. Para a instalação da componente RPI, foi utilizado um ansible enquanto que para a instalação da componente MainPC foi utilizado um dockerfile com integração devcontainer (mais informações na [DOCUMENTAÇÃO]).

Após o seu setup inicial e configuração do sistema, as diferentes componentes de hardware têm que ser devidamente conectadas e posicionadas. Nas próximas figuras podem ser vista a sua instalação (mais informações na [DOCUMENTAÇÃO]):



**Legenda:**

1. Sensor da câmara termal
2. Sensor DS18B20 de controlo da temperatura da câmara
3. EVSE
4. Entrada para o exterior da câmara termal

Figura 12: Imagem do EVSE dentro da câmara termal



**Legenda:**

1. Sensores DS18B20 posicionadas dentro do EVSE
2. Raspberry do EVSE

Figura 13: Posicionamento dos sensores DS18B20 dentro do EVSE

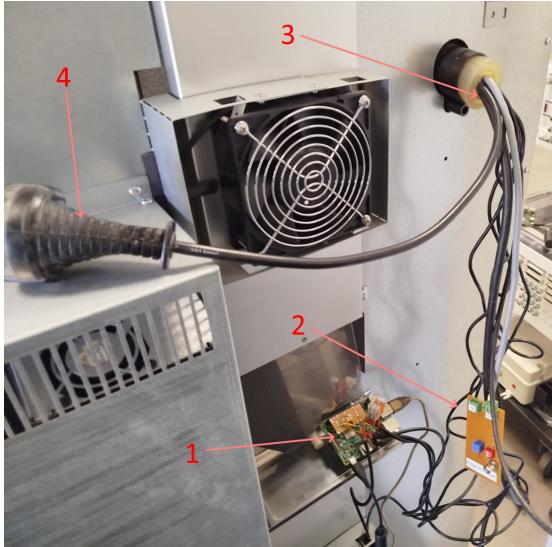


Figura 14: Parte externa traseira da câmara termal

**Legenda:**

1. RPI de controlo do teste
2. ... TODO
3. Entrada para o interior da câmara termal
4. ... TODO



Figura 15: Parte externa traseira da câmara termal

**Legenda:**

1. Câmara termal IMP 400
2. Cabo de carregamento do carro

## 5.4 Validação

Para validação do protocolo foram realizados vários testes do mesmo ao longo do estágio. Os testes mais relevantes realizados foram os seguintes:

- Precisão e exatidão dos sensores DS18B20: Colocar os sensores DS18B20 dentro da câmara termal e verificar se os valores lidos pelos sensores são coerentes com a temperatura da câmara termal (foram realizadas diversas iterações com diferentes temperaturas na câmara);

- Schedulling: Verificar se o sistema de schedulling estiver a funcionar corretamente com todas as componentes do sistema (MainPC, RPI e EVSE). Para tal foram colocados os sensores DS18B20 da câmara termal enquanto o EVSE estava fora da câmara a enviar dados sem ter nenhum carregamento a decorrer;
- Logistica: Verificar a disposição de três sensores DS18B20 dentro do EVSE dentro da câmara termal. Também serviu para verificar a nossa hipótese de que o carregador estaria a uma temperatura inferior da câmara termal enquanto a temperatura da câmara termal não estivesse estabilizada (a aumentar a temperatura, por exemplo), mas que eventualmente o carregador iria atingir uma temperatura superior à da câmara termal;
- Teste com um carro elétrico: Para validar o protocolo em condições reais, foi realizado um teste com diferentes veículos elétricos que requerem diferentes potências de carregamento. Os resultados obtidos não só validaram o protocolo, como também permitiram verificar a eficiência do carregamento do carro elétrico ao longo do tempo.

Os resultados obtidos dos testes realizados em condições reais revelaram que o carregador EVSE em temperaturas altas (acima dos 60°C) o carregador consegue manter a eficiência de carregamento acima dos 95% (o que é considerado aceitável para o carregamento de veículos elétricos).

(TODO adicionar mais informações sobre os testes realizados e os resultados obtidos)

## **6 Conclusões**

### **6.1 Resultados alcançados**

O protocolo de teste de temperatura desenvolvido atinjui os objetivos propostos, permitindo a monitorização e controlo do carregamento de veículos elétricos, monitorização em tempo real e fácil dos resultados obtidos em tempo real. Além disso, foi possível verificar a eficiência do carregamento do carro elétrico ao longo do tempo, permitindo a análise de desempenho do carregador EVSE em diferentes condições de temperatura. Além disso, o protocolo foi desenvolvido de uma forma modular o que permite que seja facilmente adaptado a diferentes tipos de dispositivos, sensores, base de dados, etc. Para um contexto de laboratório de investigação, o protocolo é facilmente adaptável a diferentes contextos de investigação.

Este protocolo irá ser uma ajuda importante no desenvolvimento de novos carregadores de veículos elétricos, como forma de validação do mesmo.

### **6.2 Lições aprendidas**

O estágio permitiu ter uma visão mais aprofundada sobre o trabalho de investigação em um contexto de laboratório, onde vários projetos estão a ser desenvolvidos em simultâneo. Além disso, como o problema proposto era um problema aplicável a um contexto real e ser desenvolvido do zero, foi possível aprender várias lições sobre o desenvolvimento de software, como a importância de ter uma boa documentação, a importância de ter uma boa comunicação com a equipa e a importância de ter uma boa organização do trabalho. Estou muito grato pela oportunidade de ter trabalhado com uma equipa tão experiente e dedicada, que me ajudou a crescer como profissional.

### **6.3 Trabalho futuro**

O trabalho futuro passaria por melhorar o protocolo de forma a fornecer mais informações relevantes sobre o carregamento que ajudariam na análise de desempenho do carregador EVSE, como por exemplo, poder realizar diversas sessões de carregamento durante o mesmo teste. Outra featura interessante seria a possibilidade de comparar os resultados obtidos com os resultados de outros testes realizados. Além disso, seria interessante implementar uma REST API para permitir que o protocolo seja facilmente integrado com outros sistemas e aplicações.

### **6.4 Agradecimentos**

Nesta secção gostaria de agradecer a todos que contribuíram para que o estágio fosse um sucesso. Em primeiro lugar, gostaria de agradecer ao INESC TEC pela oportunidade de realizar o estágio no laboratório SGEV, onde tive a oportunidade de aprender e crescer como profissional. Também gostaria de agradecer

ao meu orientador na instituição, engenheiro Pedro Pascoal, pelo seu apoio e orientação ao longo do estágio e por me ter proporcionado os melhores meios e recursos para o desenvolvimento do trabalho, além de toda a sua paciência e compreensão ao longo do processo. Gostaria de agradecer ao meu orientador na FEUP/FCUP, professor Rolando Martins, pelo seu apoio e orientação ao longo do estágio na vertente académica e por me ter ajudado na aprovação e sugestão de melhorias no projeto. Gostaria de agradecer a todos os meus colegas de estágio, que me ajudaram e apoiaram ao longo do estágio, e que tornaram o ambiente de trabalho mais agradável. Por último, gostaria de agradecer à faculdade de Engenharia da Universidade do Porto (FEUP) e à Faculdade de Ciências da Universidade do Porto (FCUP) pela oportunidade de desenvolver este projeto e de aprender com os melhores durante a minha licenciatura.

## Referências

- [1] Ds18b20 temperature sensor datasheet. URL: <https://www.circuitbasics.com/wp-content/uploads/2016/03/DS18B20-Datasheet.pdf>.
- [2] Heratherm refrigerated incubators imp 180/ imp 400 manual [en]. URL: <https://assets.thermofisher.com/TFS-Assets/LSG/manuals/LED-Heratherm-Refrigerated-Incubators-50150875-IMP180-400-Manual-EN.pdf>.
- [3] Brian, Mark, Benoit, Sara Santos, and Alessandro. Raspberry pi: Temperature readings with ds18b20 (python), Oct 2023. URL: <https://randomnerdtutorials.com/raspberry-pi-ds18b20-python/>.
- [4] Scott Campbell. Raspberry pi ds18b20 temperature sensor tutorial, Mar 2025. URL: <https://www.circuitbasics.com/raspberry-pi-ds18b20-temperature-sensor-tutorial/>.
- [5] Derek Molloy. *Exploring raspberry pi: Interfacing to the real world with embedded linux*. John Wiley and Sons, Inc, 2016.