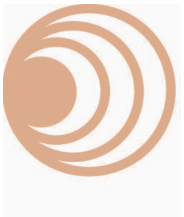
Tiago Alexandre Vicente Horta

APLICAÇÃO WEB PARA CONTROLO DO CRESCIMENTO DE MICROALGAS



UNIVERSIDADE DO ALGARVE
Instituto Superior de Engenharia
2024

Tiago Alexandre Vicente Horta

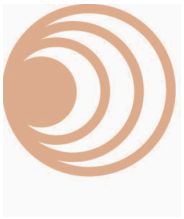
APLICAÇÃO WEB PARA CONTROLO DO CRESCIMENTO DE MICROALGAS

Mestrado em Engenharia Eletrotécnica e de Computadores

(Especialidade em Tecnologias de Informação e Telecomunicações)

Trabalho realizado sob a orientação de:

Doutor Roberto Célio Lau Lam



UNIVERSIDADE DO ALGARVE
Instituto Superior de Engenharia
2024

APLICAÇÃO WEB PARA CONTROLO DO CRESCIMENTO DE MICROALGAS

Declaração de autoria da obra

Declaro ser o autor desta obra, que é original e inédita. Os autores e obras consultados são devidamente citados no texto e constam da listagem de referências incluídas.

_____

(Nome Completo do Autor)

AGRADECIMENTOS

Queria primeiramente agradecer ao laboratório colaborativo *GreenCoLab* pelo convite para trabalhar neste projeto e por acreditarem nas minhas capacidades dando-me assim a oportunidade de desenvolver o *software* onde futuros utilizadores farão várias experiências e farão crescer diversas espécies de microalgas.

Em segundo lugar, agradecer aos diversos professores que conheci na Universidade do Algarve, mais especificamente no departamento DEE, professores esses que sempre me ajudaram, apoiaram e acreditaram nas minhas capacidades ao longo do meu percurso académico.

Por fim, agradecer de forma geral à minha família que sempre acreditou em mim e encheu o meu coração de orgulho por todas as minhas conquistas.

Resumo

Este projecto destina-se ao desenvolvimento de um *software* para o controlo de um ou vários fotobiorreatores, o referido *software* é composto por uma aplicação Web e um ou vários microcontroladores. A aplicação Web tem como principal objetivo o fornecimento de uma interface de utilizador na qual será possível a visualização e realização de experiências para o controlo do crescimento de microalgas. Para a realização de experiências, a aplicação Web irá enviar comandos através de uma API, para um microcontrolador que estará ligado a um fotobiorreator realizando assim uma ação ou consulta dos valores nos sensores no fotobiorreator.

Será possível simular diferentes condições ambientais através de ajustes nos parâmetros dos sensores disponíveis, sendo estes luzes LED brancas e RGB, temperatura, pH, dióxido de carbono e oxigénio no meio de cultivo. O *software* inclui funcionalidades como a utilização da aplicação através de um navegador Web não sendo necessária qualquer instalação, interface responsiva para a utilização quer em telemóveis quer em computadores, a possibilidade de controlo de vários fotobiorreatores em simultâneo e a possibilidade de acesso ao *software* por parte de vários utilizadores em paralelo.

Ao realizar experiências de controlo do crescimento de microalgas, serão mostrados na aplicação os valores dos parâmetros de temperatura e pH sob a forma de gráfico no interface de utilizador, valores esses que serão também gravados numa base de dados. A aplicação Web irá permitir também o descarregamento de um *dataset* de qualquer experiência em formato CSV de modo a possibilitar uma posterior análise.

Palavras Chave: Aplicação Web, Software como serviço, Nuvem, Análise de dados

ABSTRACT

This project is aimed at developing software for controlling one or more photobioreactors. The software consists of a web application and one or more microcontrollers. The main objective of the web application is to provide a user interface in which it will be possible to visualize and carry out experiments to control the growth of microalgae. To carry out experiments, the web application will send commands via an API to a microcontroller that will be connected to a photobioreactor, thus carrying out an action or consulting the values of the sensors in the photobioreactor.

It will be possible to simulate different environmental conditions by adjusting the parameters of the available sensors, these being white and RGB LED lights, temperature, pH, carbon dioxide and oxygen in the growing medium. The software includes features such as using the application via a web browser, with no installation required, a responsive interface for use on both mobile phones and computers, the possibility of controlling several photobioreactors simultaneously and the possibility of several users accessing the software in parallel.

When carrying out experiments to control the growth of microalgae, the application will display the values of the temperature and pH parameters in graphical form on the user interface, these values will also be saved in the database. The web application will also make it possible to download a dataset of any experiment in CSV format so that it can be used for analysis later.

Keywords: Web Application, Software as a Service, Cloud, Data Analysis

ÍNDICE

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

SaaS	Software as a service
PBR	Photobioreactor
API	Application Programming Interface
USB	Universal Serial Bus
CD	Compact Disk
CSV	Comma-separated value
IOT	Internet of things
ML	Machine Learning
AI	Artificial Intelligence
IDE	Integrated Development Environment
SPA	Single Page Application
SDK	Software development kit
CSS	Cascading Style Sheets
UI	User Interface
REST	Representational State Transfer
URI	Uniform Resource Identifier

1 INTRODUÇÃO

As microalgas contêm ingredientes valiosos tais como, proteínas, carboidratos, lipídos e crescem rapidamente precisando apenas de luz solar, dióxido de carbono e minerais, por estas mesmas razões são vistas como matéria prima promissora podendo ser utilizadas em vários setores industriais diferentes. O custo de produção de microalgas ainda é alto, limitando as suas aplicações comerciais, no entanto parte da redução desses custos pode ser obtida através quer na concepção mais eficiente de fotobiorreatores (PBR)[1] quer no *software* utilizado para o controlo do mesmo.

O foco deste projeto é o desenvolvimento de um *software* de alta eficiência e baixo custo, onde estará presente uma visão de inovação adicionando funcionalidades que permitirão aos laboratórios que trabalham com microalgas serem mais eficientes no controlo e cultivação do crescimento das mesmas. O *software* desenvolvido é uma aplicação Web que tem como destaque funcionalidades que aumentam a produtividade neste sector, funcionalidades como a ausência da necessidade de qualquer tipo de instalação num computador, tablet ou telemóvel tornando assim versátil a sua utilização em qualquer tipo de dispositivo que tenha um navegador Web, uma interface responsiva para suportar qualquer tamanho de ecrã, a possibilidade para controlar vários fotobiorreatores em simultâneo e acesso local ou remoto em simultâneo de vários utilizadores. Em alternativa às tradicionais aplicações de computador, esta não necessita de qualquer tipo de instalação. Normalmente este tipo de aplicação reside numa rede remota na *cloud*, onde os utilizadores têm um ou vários pontos de acesso, normalmente através de uma API. Tendo em conta este tipo de aplicações existem várias vantagens do seu uso, sendo as principais, a redução do tempo de acesso a atualizações do *software*, por parte dos utilizadores, a redução de custos face à alternativa tradicional e uma maior escalabilidade e flexibilidade.

No subcapítulo 1.1, objetivos e características do projeto, serão descritos os objetivos principais e a arquitetura do projeto, detalhando de que maneira as diferentes partes do sistema interagem entre si e também os diferentes sensores e parâmetros ajustáveis existentes no PBR. Irá ser detalhado o modo de funcionamento de uma experiência de crescimento de microalgas, as

condições que precisam de ser reunidas para a sua realização, as características específicas do *software* e será feito um enquadramento com os objetivos principais. Por fim, será descrito de forma breve o modo de como serão analisados os *datasets* gerados pela realização de experiências de crescimento de microalgas e quais serão os mecanismos de inteligência artificial utilizados com o objetivo de detetar anomalias no sistema.

No subcapítulo 1.2, contexto do trabalho, será feita uma breve descrição de onde surgiu a ideia do projeto, que necessidades o mesmo irá cobrir e que inovações irá trazer possivelmente para o mercado de fotobiorreatores.

No subcapítulo final 1.3, será feita uma breve descrição da organização, estrutura e tópicos abordados, com o objetivo de dar a conhecer de forma resumida o documento.

1.1 Objetivos e Características do Projeto

O objectivo principal do projeto é fornecer uma interface de utilizador para possibilitar a configuração de um ou vários fotobiorreatores para a realização de experiências no controlo do crescimento de microalgas. A configuração escolhida pelo utilizador será gravada numa base de dados para possibilitar a sua reutilização em experiências distintas. Ao iniciar uma experiência serão lidos através dos sensores do fotobiorreator e mostrados sob a forma de gráficos no interface de utilizador, os valores de temperatura e pH no decorrer da experiência. Existe também a necessidade de pausar a experiência para verificação, reposição ou remoção de parte da biomassa no meio de cultivo, em modo de pausa será possível também alterar as configurações da experiência, caso a experiência esteja a decorrer não será possível alterações. Ao terminar uma experiência o objectivo será possibilitar uma análise de forma geral, através dos gráficos fornecidos, da ocorrência de erros ou mudanças bruscas nos valores lidos, subidas ou descidas de temperatura, subidas ou descidas de pH ou mesmo falha na leitura de valores deverão ser fáceis de serem identificados. Para uma análise mais profunda será possível fazer *download* do *dataset* em formato CSV da experiência, que por sua vez terá dados com mais precisão gravados de 4 em 4 segundos, o mesmo dataset poderá ainda ser utilizado para análises estatísticas por parte dos laboratórios.

O projeto contará com um microcontrolador ligado a um PBR onde é estabelecido um dos pontos de comunicação entre máquinas, o microcontrolador utilizado será um *Arduino* que corre um *software* próprio desenvolvido pelo laboratório *GreenCoLab*, que tem como objetivo alterar e ler valores dos sensores do fotobiorreator através de comandos específicos. Serão enviados comandos de comunicação através da aplicação Web e será possível também a configuração de diferentes parâmetros, tais como, a cadência em que serão mostrados os valores nos gráficos de temperatura e

pH, os valores de luz LED branca e RGB, temperatura, pH, dióxido de carbono e oxigénio no meio de cultivo. A aplicação Web permitirá ainda acompanhar toda a duração da experiência monitorizando de forma constante, consoante a escala (1 a 60 minutos) escolhida pelo utilizador, os valores de temperatura e pH. Cada configuração escolhida na aplicação Web pelo utilizador enviará um comando específico para o microcontrolador, é importante ressaltar que ao carregar dados de configuração, denominados por perfis de utilizador outrora utilizados numa outra experiência, estes serão enviados 1 por 1 de forma sequencial para garantir uma configuração correta dos parâmetros no PBR.

Em termos de características específicas, a aplicação Web terá duas componentes principais, uma aplicação cliente desenvolvida utilizando uma *framework* de *JavaScript* denominada por *React* que terá a finalidade de fornecer o interface de utilizador e uma aplicação servidor desenvolvida utilizando *Flask*, uma Web *framework* em *Python*, que tem como principal objetivo o fornecimento de uma RESTful API[2] para estabelecer o ponto principal de comunicação entre a interface de utilizador e o microcontrolador. Uma RESTful API é uma interface de programação que segue os princípios da arquitetura REST, onde REST é um estilo de arquitetura principalmente utilizado em serviços Web, onde recursos são identificados por URLs e as operações que são realizadas através dos métodos de HTTP padrão (GET, POST, PUT, PATCH e DELETE). O fluxo de comunicação, ilustrado na Figura 1, partirá da aplicação cliente dado que cada configuração ou alteração feita pelo utilizador na interface enviará um comando para a API, esse mesmo comando será enviado para o microcontrolador e por fim a alteração desejada será aplicada no PBR.

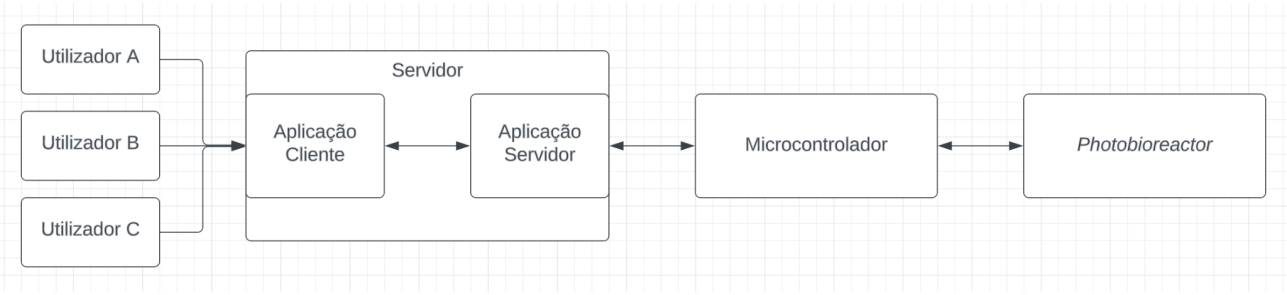


Figura 1 - Arquitetura do fluxo de comunicação entre os componentes do sistema

A comunicação poderá também fluir de forma contrária onde o microcontrolador enviará mensagens para a porta série ao qual estará conectado, será então responsabilidade da aplicação servidor fazer a gestão dessas mesmas mensagens, um exemplo deste tipo de comunicação é o começo e decorrer de uma experiência, onde será enviado um comando para começar a experiência

e serão gravados os dados dessa mesma experiência quando recebidos. Ao mesmo tempo que estarão a ser lidas as mensagens recebidas e gravados os dados da experiência num processo à parte, a aplicação servidor poderá receber outras chamadas por parte da aplicação cliente não interferindo com o processo de gravação dos dados como ilustrado na Figura 2.

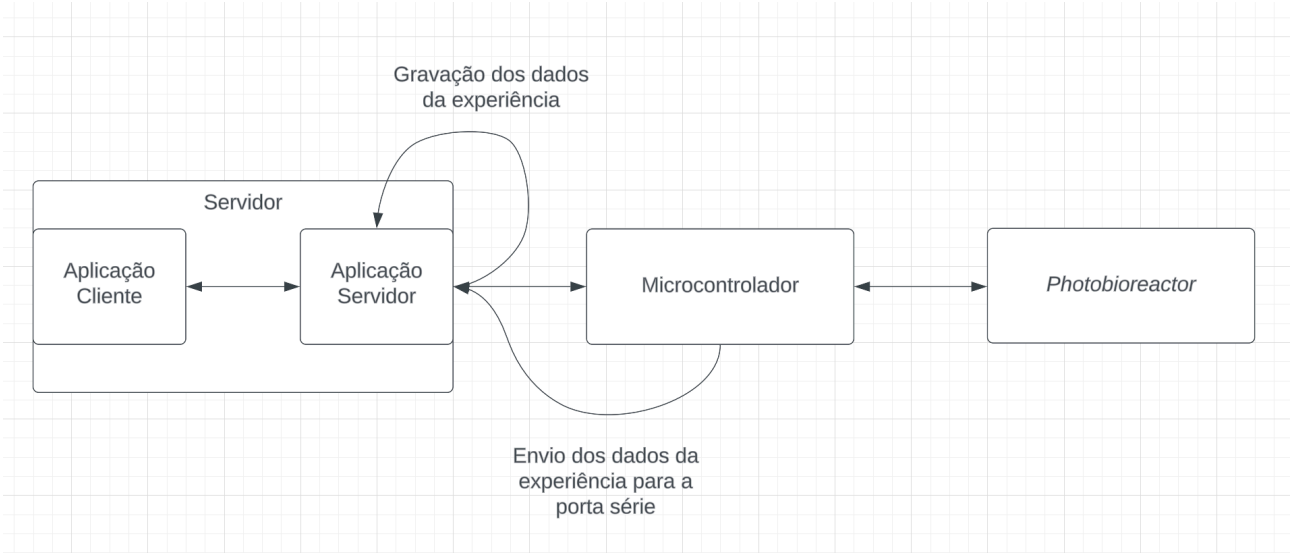


Figura 2 - Fluxo de gravação dos dados de uma experiência a decorrer

A aplicação cliente ficará então com a responsabilidade de requisitar os dados consoante configuração da cadência definida pelo utilizador, caso seja de 1 minuto, será feito um pedido para ir buscar os dados a cada 1 minuto refletindo essa mesma escala nos gráficos. Será possível também aceder aos dados de uma experiência anterior e carregar o *dataset* da mesma com uma cadência diferente para visualização e análise dos dados, a mesma escala será aplicada ao fazer *download* do *dataset* em formato CSV.

Após a realização de vários testes experimentais com o protótipo, existe ainda a possibilidade de analisar os dados dos resultados obtidos em cada experiência para a deteção de anomalias no sistema e para a realização de alguns estudos estatísticos.

1.2 Contexto do Trabalho

O *GreenCoLab* é um laboratório colaborativo de investigação na área das microalgas[3] e tem como um dos principais objetivos o desenvolvimento de projetos no seu campo de investigação. Dado o custo elevado de um fotobiorreator que tem como propósito a cultivação de microalgas, os responsáveis do laboratório decidiram desenvolver um protótipo com o objetivo de alcançar futuramente o mercado com um preço mais competitivo.

Com o protótipo do fotobiorreator já desenvolvido pelo laboratório, o objetivo principal deste projeto é o desenvolvimento de um *software* para o controlo do mesmo. As funcionalidades essenciais do *software* serão a possibilidade de realizar experiências de controlo do crescimento de microalgas onde a configuração de um PBR deverá ser feita totalmente a partir de um interface de utilizador, a visualização do decorrer da experiência na sua totalidade onde os valores de temperatura e pH deverão ser apresentados em tempo real na forma de gráficos, a possibilidade de pausar a experiência para eventuais mudanças no meio de cultivo sendo possível depois retomar a experiência a partir do ponto de paragem, a possibilidade de visualizar e analisar experiências realizadas anteriormente e a possibilidade de fazer *download* da experiência a decorrer ou de experiências anteriormente realizadas em formato CSV para posterior análise de dados estatísticos de forma detalhada.

Por fim, será feita uma avaliação do custo do desenvolvimento e produção deste tipo de protótipo por parte do laboratório, desde o *hardware* ao *software*, para determinar a possibilidade de desenvolver um produto escalável e comercializável.

1.3 Organização do Documento

Como parte fundamental do desenvolvimento deste projeto foi necessária uma investigação na área das microalgas, mais especificamente, com foco no funcionamento de um fotobiorreator e no *software* utilizado para controlo do mesmo.

No capítulo 2 revisão bibliográfica / estado de arte, será feita uma contextualização e revisão de diferentes desenvolvimentos técnicos e científicos para por fim destacar a importância do trabalho desenvolvido.

No capítulo 3 desenvolvimento do trabalho realizado, serão detalhadas as diferentes partes do sistema fazendo uma separação entre o *hardware* e o *software*. Começando pelo *hardware*, será feita uma contextualização do desenvolvimento do protótipo do PBR, o modo de funcionamento do microcontrolador e será detalhada a comunicação entre máquinas. Seguindo depois para o *software*, será detalhado o modo de comunicação com o microcontrolador, a arquitetura do sistema, as suas principais características, será descrito em detalhe o modo de funcionamento, principais funcionalidades, a comunicação entre o utilizador e o fotobiorreator e por fim será descrito o modo de funcionamento da realização de experiências para o controlo do crescimento de microalgas.

No capítulo 4 resultados experimentais, serão analisados os *datasets* obtidos através da realização de experiências utilizando diferentes configurações consoante a cultivação de espécies diferentes de microalgas. Serão analisados objetivos tais como a realização de experiências de

crescimento de microalgas por mais de 24 horas, análise dos valores de temperatura e pH na totalidade da duração de uma experiência, análise do efeito de pausa e mudança de parametrização em experiências de longa duração (vários dias ou semanas), análise da eficácia e rapidez do crescimento das microalgas, a análise e comparação dos dados de diferentes experiências.

No capítulo 5 conclusões e trabalhos futuros, será feita uma análise geral do projeto e do trabalho realizado, será feita uma comparação entre os objetivos propostos e os objetivos alcançados e será feita também uma projeção de trabalhos futuros possíveis para a melhoria do atual sistema ou desenvolvimento de funcionalidades que poderão ter ficado em falta.

2 Revisão Bibliográfica / Estado da Arte

O desenvolvimento de um sistema de controlo automatizado para fotobiorreatores é um campo que envolve a integração de biotecnologia, engenharia de controlo e tecnologias de informação. Sendo um fotobiorreator um dispositivo projetado para a cultivação controlada de microalgas, existe uma dependência para a gestão das variáveis ambientais de modo a otimizar a produção de biomassa e de outros bioprodutos de interesse industrial. A necessidade de controlar parâmetros como luz, temperatura, pH e CO_2 no meio de cultivo de microalgas em tempo real torna essencial a implementação de um software robusto e eficiente.

Este capítulo apresenta uma revisão de estudos existentes na área do desenvolvimento de *software* para o controlo de fotobiorreatores, esta revisão é estruturada e dividida em vários subcapítulos onde são explorados os fundamentos dos fotobiorreatores, descritos os componentes essenciais e algumas tecnologias de automação. Esta abordagem permite uma compreensão das bases teóricas e tecnológicas que sustentam o projeto, bem como os desafios e oportunidades relacionadas associadas com a mesma área.

No subcapítulo 2.1 introdução ao fotobiorreator, serão apresentados conceitos fundamentais relacionados com fotobiorreatores, com foco na descrição do seu modo de funcionamento. Será dado também um ênfase à importância da automação, pois a cultivação de microalgas é garantidamente mais estável e otimizada sendo portanto mais eficaz, minimizando também a necessidade de intervenção manual.

No subcapítulo 2.2 componentes do fotobiorreator e parâmetros de controlo, serão explorados os principais componentes físicos e químicos utilizados pelo fotobiorreator. Será descrito o modo de como os parâmetros luz, temperatura, pH e CO_2 afetam o crescimento das microalgas, descrevendo também os sensores necessários para monitorizar e controlar esses mesmos parâmetros de forma eficaz.

No subcapítulo 2.3 s*oftware* para o controlo de fotobiorreatores, será feita uma análise da arquitetura adequada para controlar fotobiorreatores, exploradas algumas soluções de *internet of*

things (IoT), serão destacados alguns elementos essenciais para a implementação de um sistema eficiente e detalhada a parametrização eficiente dos diferentes sensores de um fotobiorreator.

No subcapítulo 2.4 estado da arte no desenvolvimento de fotobiorreatores automatizados, serão apresentadas algumas das pesquisas mais recentes e projetos inovadores na área dos fotobiorreatores. Serão exploradas algumas das tecnologias emergentes, como a artificial intelligence (AI) e a *internet of things* (IoT) que estão a transformar a maneira de como são implementados os sistemas para o controlo de fotobiorreatores, oportunidades futuras e desafios serão discutidos fornecendo uma visão clara das várias direções possíveis para a evolução na área.

Finalmente no subcapítulo 2.5 conclusão, serão sintetizados os pontos abordados anteriormente destacando as contribuições significativas para o campo do desenvolvimento de *software* para fotobiorreatores, será contextualizado por fim, o projeto proposto.

2.1. INTRODUÇÃO AO FOTOBIORREATOR

Um fotobiorreator é um sistema biotecnológico que permite a cultivação de organismos fotossintéticos, como microalgas, em condições controladas. Os fotobiorreatores utilizam luz na forma de radiação artificial ou natural para impulsionar a fotossíntese, que é o processo em que organismos como plantas ou algas captam a luz solar e a transformam em energia química. Esta energia é utilizada para produzir compostos orgânicos, como carboidratos ou açúcares, a partir de água e dióxido de carbono. Existem dois grandes grupos de fotobiorreatores, os sistemas abertos que podem ser lagoas para a produção em larga escala de algas mas onde existe menos controlo sobre as condições ambientais e mais suscetibilidade a contaminações e os sistemas fechados onde se inserem os reatores tubulares, planos e colunas de bolhas, estes reatores oferecem um controlo mais preciso sobre os parâmetros de cultivo, como a luz, temperatura, pH e a composição do meio e são ideais para a produção de microalgas de alto valor agregado.

A automação nos fotobiorreatores é essencial para garantir o controlo constante dos parâmetros críticos que influenciam o crescimento das microalgas, sistemas automatizados podem ajustar dinamicamente a intensidade da luz, a temperatura do meio de cultivo, o fluxo de CO_2 para aumentar a produtividade e eficiência do sistema, sem a automação seria difícil manter as condições ótimas de cultura por períodos longos. A automação facilita também a escalabilidade, permitindo que operações sejam realizadas em grande escala com precisão e reduzindo a necessidade de intervenção manual constante no meio de cultivo, este ponto é particularmente importante em indústrias que exigem alta reprodutibilidade e controlo, como é o caso da produção de biocombustíveis, alimentos e compostos farmacêuticos.

2.2. Componentes do fotobiorreator e parâmetros de controlo

Os sensores de um fotobiorreator são fundamentais para monitorizar as condições do meio de cultura das microalgas e fornecer dados a tempo real para os sistemas de controlo. São portanto essenciais os sensores de luz que medem a intensidade luminosa em diferentes pontos do fotobiorreator, sensores de temperatura que servem para monitorizar o controlo da temperatura do meio de cultivo, sensores de pH que servem para monitorizar continuamente o nível de acidez do meio e sensores de CO_2 que controlam a taxa de injeção de dióxido de carbono essencial para o crescimento das microalgas. A integração de sensores e atuadores automatizados permitem uma operação contínua de um fotobiorreator em condições ótimas aumentando a sua eficiência, podem ser então utilizadas bombas de circulação, válvulas de controlo, sistemas de iluminação LED ajustáveis e dispositivos de aquecimento e arrefecimento.

Os parâmetros de controlo essenciais são exatamente os mesmos componentes físico-químicos que os sensores monitorizam no fotobiorreator, a luz é o parâmetro mais crítico do fotobiorreator pois fornece a energia necessária para a realização da fotossíntese por parte das microalgas. A distribuição de luz dentro do fotobiorreator é um dos maiores desafios especialmente em reatores de grande volume, a intensidade da luz deve ser alta o suficiente para suportar a fotossíntese, mas não intensa que cause a fotoinibição onde a taxa de fotossíntese diminui devido ao excesso de luz. A qualidade da luz (espectro de comprimentos de onda) afeta também o crescimento das microalgas, sendo que as faixas de luz ótima são o azul e o vermelho, é portanto utilizado na maior parte das vezes luzes LED brancas e RGB[4].

O parâmetro da temperatura afeta diretamente a eficiência metabólica dos organismos, cada espécie de microalgas tem uma faixa de temperatura ideal para o seu crescimento e a variação fora dessa faixa pode reduzir significativamente a produtividade. Nos fotobiorreatores o controlo da temperatura pode ser feito através de mantas térmicas ou sistemas de arrefecimento do ar, como ventoinhas. A integração de sensores de temperatura é essencial para garantir um controlo preciso no meio de cultivo[5].

O pH da cultura das microalgas influencia a solubilidade de nutrientes e a disponibilidade de dióxido de carbono, ambos essenciais para o crescimento das microalgas. Controlar o pH em fotobiorreatores envolve a injeção de CO_2 que ao dissolver na água forma ácido carbônico, podem ainda ser adicionados ácidos ou bases diretamente no meio para controlar o mesmo[6].

O parâmetro do dióxido de carbono é essencial para a fotossíntese, sendo convertido em biomassa pelas microalgas. Nos fotobiorreatores, o controlo preciso da injeção de CO_2 é necessário para manter os níveis ótimos da cultura sem causar acidificação excessiva do meio. Um sistema

automatizado pode ajustar a taxa de injeção de CO_2 com base nos níveis de pH e nas taxas de crescimento das microalgas, garantindo assim um fornecimento constante e eficiente para a produtividade no meio de cultura[7].

2.3. Software para o controlo de fotobiorreatores

O *software* para o controlo de fotobiorreatores requer uma arquitetura que permita a integração de múltiplos componentes de *hardware* e *software*. A escolha da arquitetura depende da complexidade do sistema, da necessidade de escalabilidade e de requisitos de precisão.

Um *Embedded System* é um sistema de computação dedicado a realizar funções específicas como parte de um sistema maior, este tipo de sistemas pode ser utilizado em fotobiorreatores de pequena escala onde a simplicidade e o custo baixo são prioridade. Microcontroladores como um *Arduino* ou um *Raspberry Pi* são utilizados muitas vezes para tal, este tipo de sistema tem a capacidade de monitorizar sensores e controlar atuadores diretamente mas sofrem de algumas limitações em termos de poder de processamento.

Os sistemas *Supervisory Control and Data Acquisition* (SCADA)[8] são mais comuns em aplicações industriais, estes permitem o controlo centralizado e a monitorização de processos em tempo real. Existe a possibilidade de integrar dados de múltiplos fotobiorreatores e fornecer interfaces gráficas para os funcionários, utilizadores finais. A arquitetura SCADA é modular, permitindo adicionar novos sensores, atuadores e algoritmos de controlo em fotobiorreatores.

Os sistemas baseados em *Internet of Things* (IoT) estão a ser cada vez mais utilizados e integrados em fotobiorreatores para permitir a monitorização e controlo remoto através de dispositivos conectados à internet. Este tipo de sistema utiliza plataformas na *cloud* para processar e armazenar dados, possibilitando assim a implementação de algoritmos de controlo mais complexos, o uso de *Artificial Intelligence* (AI) e *Machine Learning* (ML) baseados nos *datasets* obtidos através das experiências no controlo do crescimento das microalgas[9].

A interface de utilizador é a plataforma que permite a interação entre o funcionário de um laboratório ou indústria e o sistema de controlo do fotobiorreator, esta representa um papel crucial na operação do fotobiorreator pois facilita a monitorização, controlo e ajuste dos parâmetros do sistema, garantindo assim que o processo de controlo do crescimento de microalgas seja mantido dentro das condições ideais.

Uma interface bem projetada melhora a eficiência operacional, reduz o risco de erros humanos, simplifica a manutenção e ajuda a maximizar a produtividade do sistema. No contexto dos fotobiorreatores, o controlo preciso de parâmetros como luz, temperatura, pH e CO_2 é crucial, por

essa mesma razão a interface deve ser fácil de usar e interpretar mesmo que os funcionários não tenham um conhecimento técnico profundo. O uso de gráficos, painéis de controlo e componentes visuais deverão refletir de forma clara os estados operacionais dos fotobiorreatores, indicadores de estado, alarmes visuais e auditivos poderão melhorar a usabilidade e controlo do sistema.

Deverá ser possível a monitorização em tempo real dos parâmetros acima referidos, isso é essencial para a tomada de decisões caso algum parâmetro saia fora dos limites predefinidos. Não só a monitorização em tempo real mas também a possibilidade de exibir dados de experiências realizadas anteriormente é fundamental para identificar padrões, comparar diferenças ao cultivar duas espécies iguais em experiências diferentes e otimizar processos.

Em ambientes de produção mais atuais, deverá ser possível o controlo remoto permitindo que os funcionários acedam ao sistema em qualquer lugar através de interfaces baseadas na Web, isto é especialmente útil para a realização de experiências que exigem uma supervisão contínua mesmo fora do horário de trabalho regular. A interface deverá facilitar a configuração e ajuste de parâmetros operacionais, definindo limites de segurança, sugestões para as configurações ou valores por defeito nos painéis de controlo, minimização do risco de erros, confirmações de ação e prevenção de configurações incorretas que possam comprometer a segurança ou eficiência do sistema. A segurança e autenticação é outro dos pontos importantes para garantir um acesso de forma segura remoto, garantindo que apenas funcionários autorizados possam aceder e fazer alterações críticas no sistema.

O interface de utilizador não serve apenas para controlo do fotobiorreator, este é um ponto central na operação eficiente e segura, contribuindo significativamente para a maximização de produtividade, minimização de erros e a facilidade para realizar experiências para o controlo do crescimento de microalgas.

2.4. Estado de arte no desenvolvimento de fotobiorreatores automatizados

O desenvolvimento de fotobiorreatores automatizados tem evoluído rapidamente, impulsionados por avanços nas áreas de IoT, AI e biotecnologia. Estas tecnologias estão a redefinir a maneira como os fotobiorreatores são controlados e operados para o crescimento das microalgas.

A utilização de tecnologias na área de IoT permite a integração de sensores e atuadores com a possibilidade de serem monitorizados e controlados remotamente, fotobiorreatores que utilizem dispositivos deste tipo podem recolher dados em tempo real sobre os parâmetros essenciais, intensidade da luz, temperatura, pH e CO_2, permitindo ajustes automáticos otimizando assim o

crescimento das microalgas no meio de cultivo. Para além disso, a IoT facilita o uso de modelos preditivos para a identificação de anomalias que possam causar falhas no sistema[10].

No âmbito de AI, *Machine Learning* e redes neuronais, são cada vez mais utilizados para melhorar o controlo de fotobiorreatores. A *Artificial Intelligence* pode analisar grandes volumes de *datasets* de experiências de crescimento de microalgas e em tempo real otimizar o funcionamento do fotobiorreator, ajustando automaticamente os parâmetros para aumentar a produtividade[11].

A automatização avançada de fotobiorreatores está a permitir o desenvolvimento de novas aplicações em diversas áreas industriais de produção de compostos através das microalgas pois existe uma garantia de condições ótimas de cultivo, aumentando a eficiência e viabilidade econômica da produção. Na área dos biocombustíveis, os fotobiorreatores estão a ser utilizados para a produção em grande escala de microalgas ricas em lipídios, que podem ser convertidos em biodiesel[12]. Alguns fotobiorreatores são projetados para a produção de compostos bioativos, como antioxidantes, pigmentos ou outros metabólitos de alto valor. As microalgas cultivadas em fotobiorreatores podem ainda ser usadas para remover nutrientes e contaminantes de águas residuais, contribuindo para uma remediação ambiental, a automatização facilita o ajuste dinâmico dos parâmetros para maximizar a remoção dos poluentes.

Apesar dos avanços tecnológicos e industriais, existem desafios significativos que ainda precisam de ser enfrentados para que a automatização de fotobiorreatores alcance o seu potencial máximo. Esses desafios incluem a escalabilidade de sistemas, o custo dos sensores de alta precisão e a complexidade de integração de novas tecnologias avançadas como a *Artificial Intelligence* e a *Internet of Things* em ambientes de produção industrial. Por outro lado, as oportunidades para inovar são grandes, o desenvolvimento de novos materiais para reatores, a aplicação de biotecnologia para a engenharia de microalgas mais produtivas e a integração de tecnologias de *blockchain* para manter o histórico e segurança dos dados, são todas áreas promissoras que podem transformar o campo dos fotobiorreatores e das microalgas nos anos futuros.

2.5. Conclusão

Neste capítulo foi feita uma revisão geral sobre a literatura existente e o estado da arte relativamente ao desenvolvimento de *software* para o controlo de fotobiorreatores, destacando as tecnologias emergentes e algumas soluções para otimização de sistemas de produção de microalgas. A automatização de fotobiorreatores é essencial para alcançar uma operação eficiente e escalável, especialmente em aplicações industriais que exigem uma alta produção e precisão da parametrização de controlo do crescimento das microalgas.

Para o projeto proposto será descrito o desenvolvimento de um *software* de controlo de crescimento de microalgas, onde através de um fotobiorreator será possível realizar experiências de cultivação de diferentes espécies de microalgas. Serão descritas as características do fotobiorreator e do microcontrolador que compõem a parte do *hardware* do sistema, será detalhada a infraestrutura do sistema, a interface do utilizador e a RESTful API. Por fim, será detalhada a implementação de um mecanismo de deteção de anomalias do sistema através do uso de inteligência artificial.

3 DESENVOLVIMENTO DO TRABALHO REALIZADO

São poucas as condições necessárias para o desenvolvimento de microalgas, as maiores fontes para o seu crescimento são a luz, dióxido de carbono e certos minerais, em retorno as microalgas são uma fonte de diferentes nutrientes tais como proteínas, carboidratos e lipídos, por essa mesma razão existe uma grande procura em diferentes setores industriais. Suplementos alimentares, indústria farmacêutica, aquacultura e tratamento de águas residuais, são os principais responsáveis pela grande demanda de microalgas[13].

O custo de produção de microalgas ainda é alto e o preço de um fotobiorreator pode variar significativamente dependendo de vários fatores como o tamanho, a capacidade, os materiais e a tecnologia utilizada para o seu controlo. Existem fotobiorreatores de pequena escala utilizados em laboratórios e de escala industrial como é o caso do fotobiorreator ilustrado na Figura 3, que foi utilizado no projeto ALGACYCLE pela empresa Necton para o combate da escassez e contaminação da água[14].

Figura 3 - Fotobiorreator tubular de 19 m3

Uma grande referência utilizada maioritariamente em laboratórios de investigação do mercado de fotobiorreatores é o *Algem Pro*, um fotobiorreator concebido para melhorar a investigação na área da biologia das algas, pode gerar até meses de dados contínuos de forma fiável, otimizada e previsível na cultivação de microalgas[15]. O *Algem Pro* foi utilizado como modelo exemplo para o desenvolvimento do protótipo do fotobiorreator por parte do laboratório *GreenCoLab* e o *software* de controlo do *Algem Pro* chamado *Algenious*, como base para o desenvolvimento da aplicação Web para este projeto.

Foi feita uma análise e escolha das funcionalidades essenciais a serem implementadas, a gestão de perfis de parametrização, onde é possível gravar os parâmetros escolhidos pelo utilizador de luz LED, temperatura, pH, modo de controlo da válvula de CO_2 e modo de controlo das válvulas de ar, a gestão dos dados gerados pelas experiências realizadas no controlo do crescimento das microalgas e a visualização em tempo real dos parâmetros mais relevantes, a temperatura e o pH, no decorrer total de cada experiência em forma de gráficos onde no eixo do Y são apresentados os valores com uma precisão de 3 casas decimais quer da temperatura quer do pH e no eixo do X é apresentado uma escala temporal do decorrer total da experiência.

Tendo em conta que cada experiência pode ter uma duração que varia entre horas até meses e o intervalo de tempo em que são gravados numa base de dados os dados recolhidos dos valores dos sensores é de 4 segundos, o sistema deve estar preparado para lidar com uma grande quantidade de dados que serão ilustrados em tempo real nos gráficos de temperatura e pH ao utilizador.

Este projeto consiste no desenvolvimento da parte tecnológica que, como referido anteriormente, servirá para controlar o fotobiorreator e o crescimento de microalgas através de um interface de utilizador que irá ilustrar todo o decorrer de uma experiência e permitirá que o utilizador interaja com os diferentes sensores presentes no PBR.

No subcapítulo 3.1 explicação do projeto, serão apresentadas e descritas as características do protótipo do fotobiorreator desenvolvido pelo laboratório *GreenCoLab*, será explicado a forma de comunicação entre máquinas e o *software* utilizado no microcontrolador *Arduino*, será detalhado o *software Algenious*, de modo a contextualizar o projeto serão descritas as decisões de arquitetura e implementação da aplicação Web proposta e por fim será descrita a abordagem proposta para detetar anomalias no sistema após a realização de experiências para o crescimento de microalgas.

No subcapítulo 3.2 infraestrutura do sistema, será detalhada a escolha feita para a implementação e hosting aplicação Web através da utilização do modelo SaaS, será introduzida a motivação da escolha das tecnologias utilizadas, quais as vantagens e desvantagens do tipo de *frameworks* escolhidas, serão exploradas as vantagens e desvantagens que possam existir em relação a uma

aplicação instalável de computador, será contextualizada a escolha deste tipo de modelo com a necessidade de utilização e demanda por parte de um laboratório de microalgas.

No subcapítulo 3.3 desenvolvimento do projeto, serão detalhados todos os componentes da aplicação Web e será feito um enquadramento do projeto para o controlo do crescimento de microalgas através de um fotobiorreator. Irá ser também descrita de forma breve, o modo de funcionamento entre o microcontrolador *Arduino* e protótipo do fotobiorreator e contextualizada a forma de como ambos se enquadram no sistema. Este subcapítulo será subdividido em duas partes, 3.3.A interface de utilizador e 3.3.B RESTful API, de modo a facilitar a leitura, compreensão e interligação dos detalhes do sistema.

Começando pelo subcapítulo 3.3.A interface de utilizador, será feita uma descrição detalhada da arquitetura, implementação e modo de funcionamento do interface de utilizador, serão detalhadas as funcionalidades, melhorias e inovações em relação ao *software* existente do fotobiorreator *Algem Pro*.

Já no subcapítulo 3.3.B RESTful API, será explorada em detalhe a arquitetura da API, será descrita a forma de intercomunicação entre as diferentes partes do sistema, desde o interface de utilizador até ao fotobiorreator, será feita uma análise do tratamento de dados enviados pelo utilizador para o PBR e pelo PBR para o interface de utilizador.

A realização de experiências de controlo da cultivação de microalgas será feita utilizando o protótipo do fotobiorreator e o *software* desenvolvidos no projeto, onde os objetivos principais e os resultados dessas mesmas experiências serão explorados no capítulo 4 resultados experimentais.

3.1 Explicação do Projeto

Um fotobiorreator tem como objetivo principal a cultivação de microalgas num meio controlado e é através de um *software* específico que será possível esse controlo. Existem parâmetros como a temperatura, o pH, os níveis de dióxido de carbono e oxigênio, que serão alterados consoante as condições necessárias para a cultivação de uma espécie de algas específica, é então assim possível uma simulação de ambientes diferentes para a cultivação de diferentes tipos microalgas. O fotobiorreator é constituído por um tubo de vidro transparente, uma bomba mecânica, válvulas de controlo dos diferentes parâmetros e um conjunto de luzes LED que envolvem um tubo de vidro garantindo assim uma total penetração de luz, como se pode observar pela Figura 4, maximizando o crescimento das microalgas[16].

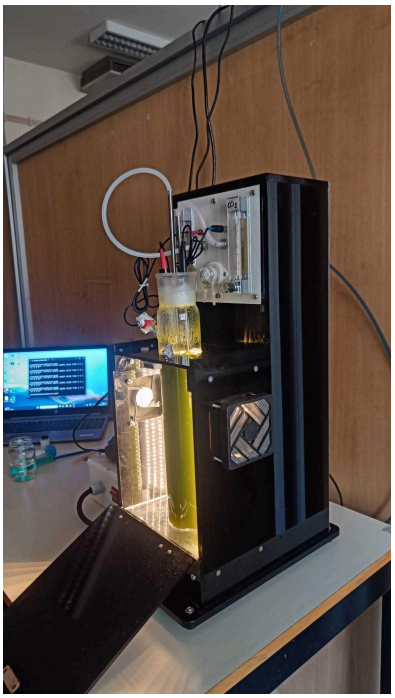


Figura 4 - Protótipo do fotobiorreator desenvolvido pelo laboratório GreenCoLab

Para o controlo total do fotobiorreator é necessário um componente que permita a comunicação entre máquinas para o envio de comandos, leitura de valores e respetiva alteração de parâmetros no fotobiorreator, foi então utilizado um microcontrolador *Arduino* de modo a permitir a alteração dos parâmetros configuráveis por um conjunto de comandos específicos ilustrados abaixo na Tabela 1.

A comunicação com o PBR é efetuada através do envio de comandos segundo uma sintaxe própria criada para o aparelho, as instruções são constituídas por cadeias de carateres ASCII maiúsculos e existem dois tipos básicos de instruções comandos e consultas. Os comandos instruem o aparelho a realizar alguma ação, enquanto que as consultas interrogam o *Arduino* sobre o estado do fotobiorreator. Os comandos são compostos por uma mnemónica constituída por 3 carateres e, eventualmente, um ou vários parâmetros separados do comando e entre si por um espaço. Os comandos de consulta são constituídos por uma mnemónica de 3 caracteres seguida de um ponto de interrogação. Os valores devolvidos pelo PBR são formados por cadeias de caracteres ASCII terminadas por um carater de mudança de linha (nl)[17].

Tabela 1 - Comandos de controlo do fotobiorreator

Comando	Explicação	Parâmetros	Notas
IDN?	Interroga o PBR sobre os seus detalhes	-	1
STA?	Interroga o PBR sobre o seu estado atual	-	2
RUN	Inicia uma experiência	-	
STP	Para uma experiência	-	
PAU	Pausa uma experiência	-	
CLP	Inicia a calibração do pH	-	
CMD	Define o modo de operação do CO_2	0 / 1 / 2	3
LMD	Define o modo de operação dos LEDs	0 / 1 / 2 / 3	4
TMD	Define o modo de operação da temperatura	0 / 1 / 2 / 3	4
MMD	Define o modo de operação do regime	0 / 1 / 2 / 3	5
MOD?	Interroga os modos de operação atuais	-	
WLD	Define os parâmetros primários da luz LED branca	MAX MIN	6
RLD	Define os parâmetros primários da luz LED vermelha	MAX MIN	6
GLD	Define os parâmetros primários da luz LED verde	MAX MIN	6
BLD	Define os parâmetros primários da luz LED azul	MAX MIN	6
LED?	Interroga os parâmetros dos LEDs atuais	W / R / G / B	6
SLD	Define os parâmetros secundários dos LEDs	PERIOD START RISE WIDTH FALL	
SLD?	Interroga os parâmetros secundários dos LEDs	-	
TMP	Define os parâmetros primários da temperatura	MAX MIN	7
TMP?	Interroga os parâmetros primários da temperatura	-	
STP	Define os parâmetros secundários da temperatura	PERIOD START RISE WIDTH FALL	

STP?	Interroga os parâmetros secundários da temperatura	-	
MED	Define os parâmetros do regime	VOL PERIOD START ODSP	
MED?	Interroga os parâmetros do regime	-	
PHS	Define um *setpoint* para o valor de pH	PHSP	8
PHS?	Interroga o *setpoint* do valor de pH	-	
VCO	Liga ou desliga a válvula de CO_2	ON / OFF	
VAI	Liga ou desliga a válvula de ar	ON / OFF	
PM1	Liga ou desliga a bomba número 1	ON / OFF	
PM2	Liga ou desliga a bomba número 2	ON / OFF	

Notas da Tabela 1

1. O PBR responde com um conjunto de carateres (*string*) “GREENCOLAB,PBR,X,Y,Z”, onde “X” é o número de série do fotobiorreator, o “Y” é a versão do *hardware* e o “Z” a data do *firmware*.
2. 0 - Parado; 1 - A Decorrer; 2 - Pausado.
3. 0 - Desligado; 1 - Ligado; 2 - Controlado pelo valor *setpoint* do pH.
4. 0 - Constante; 1 - Ciclo diário; 2 - Onda sinusoidal; 3 - Onda pulso.
5. 0 - Lote; 1 - Semi contínuo; 2 - Pseudo contínuo; 3 - Turbidostato.
6. MAX, MIN - Máximo e mínimo do valor de intensidade dos LEDs, em percentagem, 0(%) até 100(%).
7. MAX, MIN - Máximo e mínimo do valor de temperatura, em graus celsius, 0(Cº) até 100(Cº).
8. pH *setpoint* x 100, exemplo, se o valor for 7.35, enviar 735.

Estabelecida assim a primeira parte da comunicação entre máquinas, PBR e microcontrolador, é necessário então enviar os comandos ilustrados na Tabela 1 para o microcontrolador de modo a que seja possível controlar e realizar experiências de crescimento de microalgas. É necessário então o desenvolvimento de um interface de utilizador intuitivo de modo a tornar o controlo do PBR mais acessível e eficaz por parte dos funcionários de um laboratório na área das microalgas. O protótipo do fotobiorreator, o microcontrolador e o respectivo *software* foram desenvolvidos pelo laboratório *GreenCoLab* e utilizados neste projeto.

De como a avaliar e introduzir melhorias no desenvolvimento do *software* para este projeto, foi feita uma análise do interface de utilizador de um fotobiorreator comercial chamado *Algem Pro*,

ilustrado na Figura 5, com objetivo de determinar melhorias que podem ser implementadas no *software* que será utilizado para o controlo do protótipo do fotobiorreator do laboratório *GreenCoLab*.

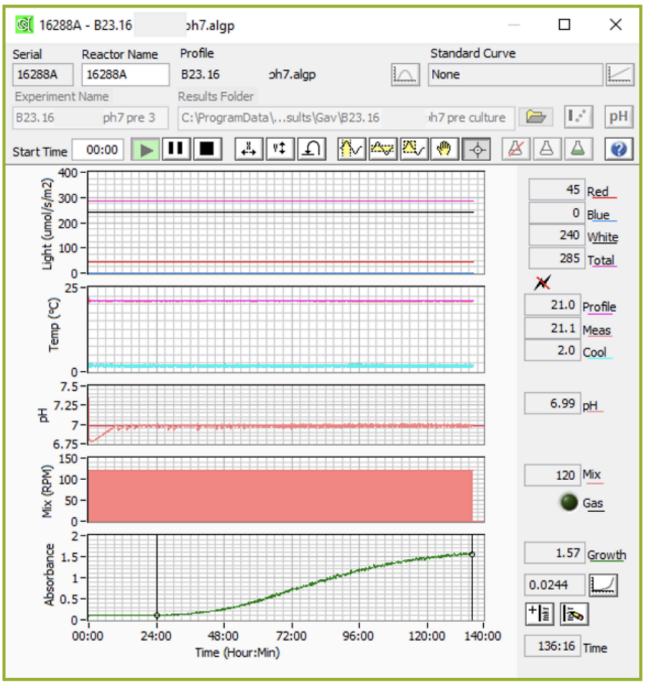


Figura 5 - Interface de utilizador do fotobiorreator Algem Pro

De um modo geral, as funcionalidades essenciais para o controlo de um fotobiorreator e para a realização de experiências de crescimento de microalgas através de um interface de utilizador, são a possibilidade de identificação um ou vários PBRs que estão conectado ao sistema, a porta série e o respectivo nome, a possibilidade de criar perfis de configuração para a reutilização de parâmetros de configuração em outras experiências, a possibilidade de realizar experiências com o fotobiorreator para o crescimento de microalgas onde terá de ser feito um tracking do tempo decorrido, temperatura e pH em forma de gráficos e por fim a possibilidade de análise dos dados, quer através da utilização dos gráficos apresentados na interface, quer através da geração de um ficheiro CSV que contém os dados da experiência realizada. O *software* do fotobiorreator *Algem Pro* tem todas as funcionalidades descritas anteriormente e muitas mais específicas que são possíveis devido aos sensores presentes no fotobiorreator. O fotobiorreator *Algem* e o seu *software* apesar de muito completo, sofre de algumas falhas ou necessidades que são essenciais para aumentar a eficiência na sua utilização e controlo para o crescimento de microalgas. Um dos seus maiores problemas é que o seu *software* é instalável através de um CD, sendo este uma aplicação de computador a sua utilização fica restrita ao computador onde foi feita a instalação, tendo isto em

conta, um dos principais pontos de inovação do desenvolvimento do *software* para este projeto é a possibilidade de acesso e controlo do PBR em vários computadores ou telemóveis que tenham um navegador Web.

Tendo a possibilidade de aceder ao *software* através de um navegador Web, será servida uma aplicação Web desenvolvida utilizando uma livraria de interfaces de utilizador chamada *React*, que tem por base a criação de componentes Web de modo a facilitar a escalabilidade e reutilização. Isso facilita a organização do código, tornando-o mais modular e fácil de manter. Outra das suas vantagens é a utilização de um modelo de programação reativa, onde as alterações no estado da aplicação atualizam automaticamente a interface simplificando assim o desenvolvimento de aplicações mais dinâmicas às interações por parte do utilizador[18].

A leitura de dados e o envio de comandos para o microcontrolador *Arduino*, é feita através de uma *framework* Web chamada *Flask*, que visa facilitar o desenvolvimento de aplicações Web de forma rápida e simples[19]. Considera-se então este o ponto de comunicação central do sistema, onde o fluxo de comunicação ilustrado na Figura 1, se inicia pela interação do utilizador com a aplicação Web através de um navegador Web, onde serão enviados comandos referidos na Tabela 1, através da chamada a *endpoints* disponibilizados pela RESTful API, em que esses mesmos comandos serão redirecionados para o microcontrolador, que por sua vez, comunicará com o PBR acionando assim a função pretendida para o seu controlo.

Para a gestão de configurações ou perfis de utilizador para a realização de experiências através do fotobiorreator, é necessário a utilização de uma base de dados onde serão gravados esses mesmos dados de configuração. Não só os dados de configuração mas também os detalhes de identificação do fotobiorreator e os dados relativos às experiências realizadas com o protótipo. A base de dados escolhida para a gestão dos dados foi *SQLite*, que é um sistema leve e autônomo onde não existem dependências de um servidor para o seu funcionamento[20].

3.2 Infraestrutura do sistema

A arquitetura do sistema necessita satisfazer as necessidades exigidas por parte de um laboratório de microalgas de modo a aumentar a sua eficiência e produtividade, o objetivo no fundo é desbloquear a possibilidade de controlar e realizar experiências de controlo do crescimento de microalgas através de um fotobiorreator de uma maneira simples e clara para o utilizador.

Os utilizadores do sistema necessitam de compreender com facilidade as possibilidades e funcionalidades que o *software* oferece através do ponto de partida que é a interface de utilizador, é através da interface que os utilizadores farão as configurações dos parâmetros do PBR, realizaram

experiências de crescimento de microalgas e irão analisar os resultados obtidos em todo o seu decorrer. A comunicação parte da interface do utilizador onde serão realizadas chamadas a uma RESTful API, estabelecendo assim o segundo ponto de comunicação do sistema, serão feitas trocas de informação entre pedidos do utilizador e respostas por parte de leituras dos sensores presentes no PBR, a aplicação servidor gravará numa base de dados a identificação e o estado do PBR que está conectado ao sistema, as configurações efetuadas ao PBR sob a forma de perfis de utilizador para possibilitar a reutilização de tais configurações em diferentes experiências e gravará os dados das experiências de crescimento de microalgas.

A troca de informação com o PBR é feita através da utilização de um microcontrolador *Arduino*, onde a API é responsável pela leitura e envio de comandos (ver Tabela 1) através de uma porta série presente no computador onde estará a correr a aplicação Web. Assim estabelecido o terceiro ponto de comunicação do sistema ilustrado na Figura 6, um comando é enviado pelo interface de utilizador para a API através de um pedido POST de HTTP, a API envia esse mesmo comando para a porta série em que o *Arduino* está conectado e por fim a mudança pretendida é refletida no fotobiorreator. Existem dois tipos de comandos que são utilizados no sistema, comandos de consulta que servem para ler dados dos sensores ou parâmetros do PBR e comandos de ação que servem para realizar alguma ação ou mudança no modo de funcionamento do fotobiorreator, como por exemplo a alteração do valor de pH.

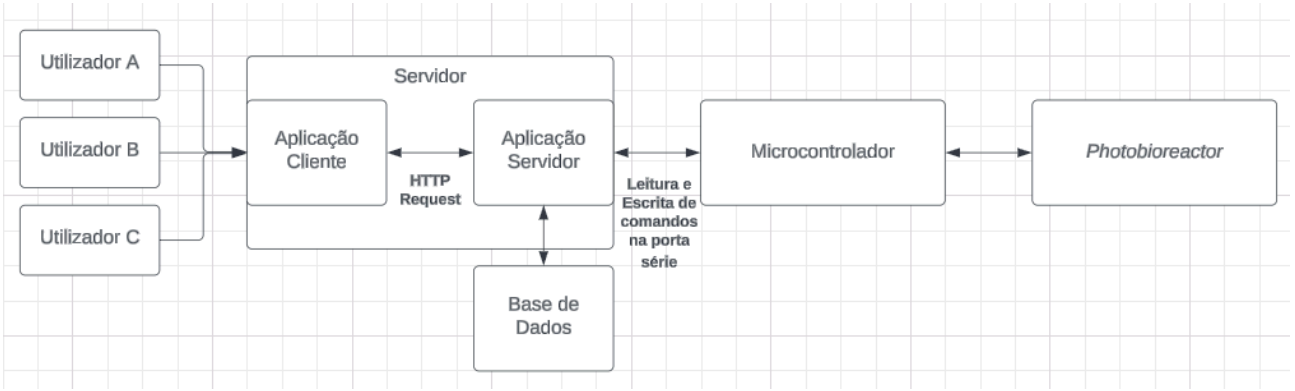


Figura 6 - Tipo de interação entre os componentes do sistema

Descrita então a estrutura do sistema, foram exploradas várias possibilidades para a escolha das tecnologias utilizadas em cada um dos componentes. Para o interface de utilizador (aplicação cliente) foram exploradas algumas opções como *Angular*, *Vue* e *React*, todas frameworks de *JavaScript* para o desenvolvimento de aplicações Web. A tecnologia escolhida foi o *React* devido à sua simplicidade, flexibilidade e desempenho, onde a sua abordagem minimalista combinada com a

grande comunidade e suporte entre desenvolvedores, torna-a opção mais atraente para soluções escaláveis e eficientes[21].

Para a API (aplicação servidor) foram exploradas algumas opções como *NestJS* e *Flask*, a primeira sendo uma framework de *JavaScript Node.js* para o desenvolvimento de aplicações Web robustas, eficientes e escaláveis e a segunda sendo uma micro framework para *Python* onde a sua abordagem é mais minimalista, simples e flexível. Para este projeto, um dos objetivos a ter em consideração é a possibilidade de maior flexibilidade e adaptação para eventuais mudanças conforme as necessidades dos laboratórios de microalgas, por essa mesma razão a tecnologia escolhida foi o *Flask* devido muito à sua simplicidade, liberdade de organização do código e fácil adaptação consoantes as necessidades exigidas. Existe também uma preocupação com a possível integração com ferramentas de aprendizagem de máquinas para a análise dos *datasets* das experiências ou deteção de anomalias no sistema, *Flask* é então uma excelente escolha para alcançar esse mesmo objetivo trazendo vantagens muito superiores em relação à rapidez de desenvolvimento e adaptação para este projeto[22].

Ao desenvolver uma aplicação Web utilizando *Python* e *Flask*, a escolha da base de dados é uma decisão crucial que pode impactar o desempenho, manutenção e a escalabilidade do sistema. O *SQLlite* é uma das opções mais populares para este tipo de projetos onde são atendidas necessidades essenciais dada a sua simplicidade e facilidade de utilização, a sua integração com *Python*, o facto de ser leve e eficiente, a sua portabilidade e o seu baixo consumo de recursos, por essas mesmas razões foi a escolha para este projeto[23].

Desenvolver uma aplicação Web em relação a uma aplicação de *desktop* convencional oferece várias vantagens que podem ser decisivas dependendo do projeto, público alvo e das necessidades da aplicação. Para este projeto foi desenvolvida uma aplicação Web pelas vantagens, necessidades gerais de um laboratório de microalgas e fácil adaptação das funcionalidades da aplicação. As aplicações Web podem ser acedidas em qualquer dispositivo que tenha um navegador Web e conexão à internet, eliminando assim a necessidade de instalação do *software* em cada máquina e permitindo a utilização em diferentes dispositivos tais como, computadores, tablets ou telemóveis.

Para contextualização das necessidades de um laboratório de microalgas, este tipo de aplicações deverá funcionar em qualquer sistema operativo, deverá permitir que diferentes utilizadores trabalhem em simultanêo, as atualizações deverão ser centralizadas pois a aplicação será hospedada num servidor central, existirá uma redução de custos de desenvolvimento face à criação de diferentes versões para diferentes sistemas operativos e como os dados são armazenados no servidor, todos os utilizadores deverão ter acesso às informações mais atualizadas do sistema, todas

estas funcionalidades e possibilidades são essenciais para uma maior eficiência de desenvolvimento e utilização face a uma aplicação de *desktop* convencional.

3.3 Desenvolvimento do projeto

A aplicação Web proposta para este projeto contém 2 componentes essenciais que garantem o controlo e a possibilidade de realização de experiências através de um fotobiorreator. Uma aplicação cliente que serve um interface de utilizador intuitivo que possibilita a configuração geral do fotobiorreator, a realização de experiências para o controlo do crescimento de microalgas e uma aplicação servidor que serve uma API que é o ponto central de comunicação do sistema onde são enviados pedidos e em troca respostas utilizando o protocolo de HTTP. São geridos também processos no background para quando realizadas experiências de forma longa e contínua, serem geridos e gravados os dados numa base de dados *SQLite*. É por fim, estabelecida a comunicação com o microcontrolador *Arduino* através da ligação com uma porta série onde é feita uma troca de comandos e mensagens.

Para o bom funcionamento de um laboratório de microalgas utilizando o *software* proposto e o protótipo desenvolvido, é necessário ter em consideração a rotina da realização de uma experiência de crescimento de microalgas. É necessário que o *software* forneça de forma clara o modo de configuração do fotobiorreator pré-experiência, que o perfil de configuração introduzido seja possível de ser gravado para reutilização em experiências futuras, seja possível dar um nome a uma experiência para facilitar a identificação, que exista a possibilidade de pausar uma experiência para a troca de compostos no meio de cultivo ou verificação manual do mesmo, é necessário visualizar em todo o decorrer da experiência os parâmetros de temperatura, ph sob a forma de gráficos na interface de utilizador e por fim, ter a possibilidade de acesso ao interface de utilizador e à experiência que estará a decorrer em múltiplos dispositivos dentro do laboratório e não só ao computador que estará ligado diretamente ao fotobiorreator.

O computador que estará ligado ao fotobiorreator exerce uma função essencial pois corre a aplicação Web e estabelece a ligação ao microcontrolador através de um cabo USB que está ligado a uma porta série, o microcontrolador corre um *software* próprio desenvolvido pelo laboratório *GreenCoLab* onde através de uma sintaxe criada para o aparelho, é feita uma troca de comandos de leituras e pedidos possibilitando assim a alteração e leitura de valores nos sensores do fotobiorreator.

3.3. A Interface de utilizador

A aplicação cliente serve um interface de utilizador que foi implementada utilizando como livraria o *React* e como linguagem de programação *JavaScript*. A escolha desta tecnologia foi feita com base na sua simplicidade de sintaxe, vasta comunidade, facilidade de encontrar soluções, abordagem minimalista e utilização de componentes reutilizáveis.

A criação desta aplicação foi feita através de um *setup* simples onde a única dependência foi a necessidade de instalação de uma versão igual ou superior à 14.0.0 da framework *Node.JS*. O output ou estrutura do projeto após correr o comando para criar a aplicação, npx create-react-app my-app, está preparada para a escalabilidade do projeto, onde configurações complexas não são necessárias como ponto de partida, apenas fazer uma *build* da aplicação é o suficiente para correr a mesma.É importante referir que o projeto foi desenvolvido utilizando *TypeScript*, que é uma forma de adicionar tipos ou definições a *codebases* de *JavaScript*, isto trás uma grande vantagem pois as estruturas que são utilizadas na aplicação são detectadas pelo o IDE utilizado e por essa mesma razão ficam mais claros os tipos e interfaces utilizados na aplicação e são reduzidos o números de erros cometidos por acesso a variáveis inexistentes[24].

O interface de utilizador tem como objetivo principal fornecer de forma clara o meio de configuração e controlo de um ou vários fotobiorreatores conectados ao sistema. Ao correr a aplicação Web, o interface do utilizador irá mostrar a identificação do número de série e estado dos fotobiorreatores conectados ao sistema na barra de tarefas principal da aplicação, irá também fornecer a possibilidade de configuração do fotobiorreator numa fase pré-experiência através do menu expansível lateral esquerdo onde perfis de utilizador podem ser criados com base nas configurações escolhidas, irá possibilitar a escolha de um nome para a experiência que será realizada onde serão mostrados os valores lidos de temperatura e pH no seu decorrer total na barra de tarefas secundária, possibilitará também a pausa de uma experiência para troca de compostos manualmente no meio de cultivo e disponibilizará duas formas de análise dos dados da experiência, uma através dos gráficos apresentados no centro da aplicação e outra através da geração de um ficheiro de CSV que contém os dados da totalidade da duração da experiência, reduzida ou não, a uma escala à escolha do utilizador. Ilustradas na Figura 7 estão identificadas as diferentes partes que compõem a aplicação.

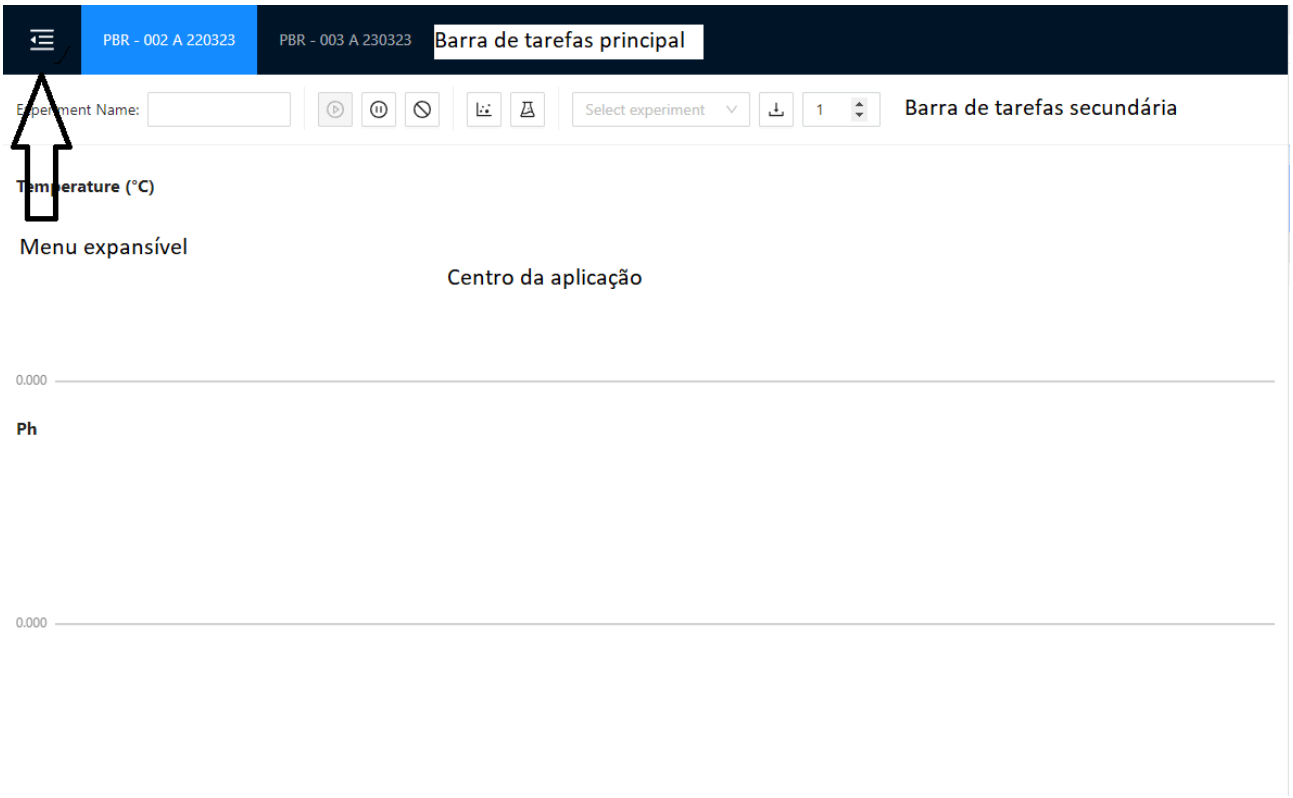


Figura 7 - Diferentes secções que compõem o interface de utilizador

A implementação da interface de utilizador e estruturação do projeto precisam de estar organizados de maneira a tornar fácil o desenvolvimento de novas funcionalidades ou implementar alterações necessárias a funcionalidades existentes. Foram então criadas diferentes diretorias com propósitos diferentes como forma de manter uma estrutura clara ao nível de código do projeto como mostra a Figura 8.

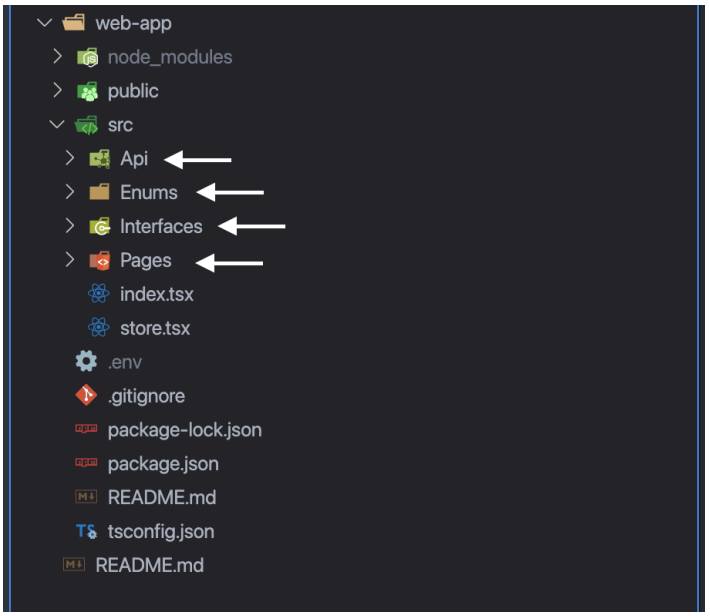


Figura 8 - Estrutura da aplicação cliente

Destacadas com setas a branco na Figura 8, estão a diretoria “Api” que tem como propósito guardar um ficheiro com todos os URIs (rotas) de pedidos que poderão ser feitos do lado aplicação servidor, estas rotas são utilizadas onde é necessário fazer um pedido ao servidor para requisitar algum tipo de informação ou realizar alguma ação no fotobiorreator, é estabelecido assim um ponto único e central onde caso seja necessário alguma alteração ou adição de uma rota nova essa mudança seja facilitada. As rotas que estão presentes no ficheiro vão de encontro aos comandos que estão presentes no microcontrolador, começando por pedidos de identificação e estado do fotobiorreator, pedidos relacionados com as configurações que o utilizador pode fazer pré-experiência e pedidos para realizar e controlar uma experiência de crescimento de microalgas, algumas dessas rotas estão ilustradas na Figura 9 seguidas de um comentário com o comando específico esperado pelo microcontrolador.

```
// Information
export const identificationRoute = `${API_URL}/identification`; // IDN?
export const statusRoute = `${API_URL}/status`; // STA?

// Experiment
export const startExperimentRoute = `${API_URL}/run`; // RUN
export const getExperimentRoute = `${API_URL}/run/get-data`;
export const stopExperimentRoute = `${API_URL}/stop`; // STP
export const pauseExperimentRoute = `${API_URL}/pause`; // PAU
export const getExperimentFilesRoute = `${API_URL}/run/get-csvs`;
export const getExperimentFileRoute = `${API_URL}/run/get-csv`;

// Ph Calibration
export const phCalibrationRoute = `${API_URL}/calibration`; // CLP
export const phCalibrationNextRoute = `${API_URL}/calibration/next`;
export const phValueRoute = `${API_URL}/ph/value`;

// Mode
export const co2ModeRoute = `${API_URL}/co2/mode`; // CMD
export const ledModeRoute = `${API_URL}/led/mode`; // LMD
export const temperatureModeRoute = `${API_URL}/temperature/mode`; // TMD
export const regimeModeRoute = `${API_URL}/regime/mode`; // MMD
export const modeRoute = `${API_URL}/mode`; // MOD?

// Leds
export const ledColorRoute = `${API_URL}/led/color`; // WLD | RLD | GLD | BLD
export const ledParamRoute = `${API_URL}/led`; // LED?
export const ledSecondaryParamRoute = `${API_URL}/led`; // SLD | SLD?
```

Figura 9 - Ficheiro das rotas para pedidos HTTP

A diretoria “Enums” serve para armazenar estruturas de dados do tipo enumeradores que são reutilizadas em diferentes pontos da aplicação, mais uma vez para facilitar a alteração ou adição de novos enumeradores foi criado um ponto central para esse propósito. Estes enumeradores detalham o tipo de opções que são possíveis de escolher para configurar os diferentes tipos de sensores presentes no fotobiorreator, como por exemplo os 3 estados possíveis de uma experiência “running”, “paused” e “stopped”. A diretoria “Interfaces” tem como propósito agregar as várias interfaces utilizadas em estruturas de dados na aplicação, foram criadas diferentes estruturas com tipos bem definidos para manter a consistência de identificação ao longo do projeto. O objetivo principal da criação de interfaces é a organização dos dados que são utilizados na aplicação como mostra a Figura 10, a estrutura e tipo dos dados do PBR conectado, da configuração do fotobiorreator, de leitura dos valores de temperatura e pH, entre a definição de outros tipos mais genéricos utilizados para facilitar o desenvolvimento da aplicação.

```ts
export interface IExperimentData {
  timelapse: string;
  timeInSeconds: string;
  timeTotal: string;
  temperature: IOptionCurrentSetValue;
  ph: IOptionCurrentSetValue;
  od: string;
  co2Valve: string;
  pump: string;
  whiteLed: string;
  redLed: string;
  greenLed: string;
  blueLed: string;
}

export interface ITemperatureData {
  timestamp: string;
  category: IOptionCurrentSetLabel;
  temp: number;
}

export interface IPhData {
  timestamp: string;
  category: IOptionCurrentSetLabel;
  ph: number;
}
```

Figura 10 - Ficheiro de interfaces

Por último, a diretoria "Pages" serve para guardar os diferentes ficheiros referentes a ecrãs específicos, componentes visuais genéricos, estilos dos componentes (ficheiros de CSS) e ficheiros de armazenamento de estados utilizados de forma global na aplicação. A estrutura do projeto a nível de ecrãs está organizada de maneira simples, sendo este tipo de aplicação uma *Single Page App* (SPA), tem apenas um ecrã principal chamado "Home" e vários componentes que o compõem como é possível observar pela Figura 11.

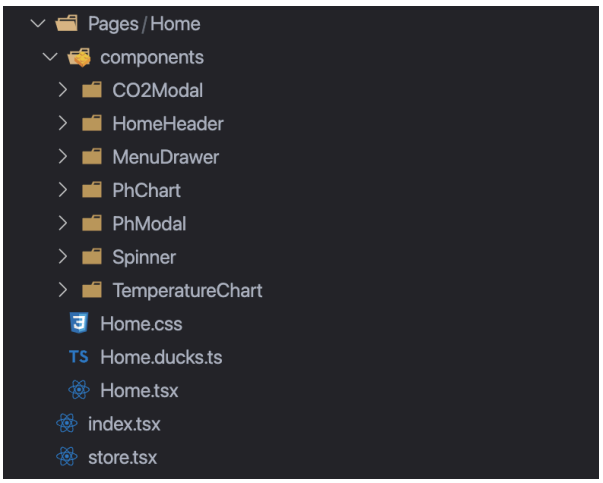


Figura 11 - Estrutura dos diferentes componentes da interface de utilizador

Detalhados os pontos principais da estruturação do projeto, será feita uma descrição das funcionalidades implementadas de forma individual de modo a facilitar a leitura.

3.3.a.1. Identificação e estado dos fotobiorreatores conectados

O ponto de partida da interface da interface do utilizador é o ecrã "Home" onde são feitos dois pedidos principais para obter informações sobre os fotobiorreatores conectados e o seu respectivo estado. Estes pedidos são feitos através da utilização de uma livraria chamada *axios* que foi instalada no projeto e que é utilizada para fazer pedidos de HTTP, onde são retornados os dados de resposta. Os dados de resposta são depois mostrados na barra de tarefas principal da interface de utilizador, tal como se pode observar pela Figura 12 abaixo, com informação dos nomes dos fotobiorreatores conectados e do estado de cada um deles, estado esse referente a se uma experiência está a correr, em pausa ou parada.

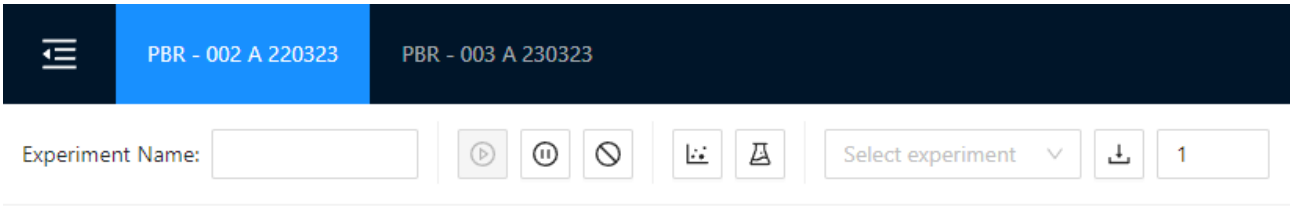


Figura 12 - Identificação dos fotobiorreatores

A função que se encarrega de fazer tais pedidos está ilustrada na Figura 13, onde se pode observar também a utilização das variáveis de rotas definidas e importadas da diretoria "Api" referida anteriormente e a utilização de um mecanismo de *dispatch* para fazer uma alteração a um estado global da aplicação.

```
async function fetchReactorsData() {
  try {
    dispatch(setIsLoading(true));

    const identificationStatus = await axios.get(identificationRoute);
    dispatch(selectReactor(identificationStatus.data));

    const statusResponse = await axios.get(statusRoute);
    dispatch(setExperimentRunning(!!statusResponse.data));
  } catch (error) {
    alert(error);
  } finally {
    dispatch(setIsLoading(false));
  }
}
```

Figura 13 - Função que inclui pedidos de informação dos fotobiorreatores

O mecanismo de *dispatch* é utilizado com o objetivo de fazer uma ação a nível global na aplicação, para possibilitar a sua utilização é necessário a instalação de uma livraria chamada *Redux* e a implementação da configuração de uma *store* que irá funcionar como um armazém de estados de diferentes partes da aplicação. A grande vantagem da utilização de uma *store* é a possibilidade do acesso a um estado definido a nível global por parte de qualquer componente da aplicação sem a necessidade de repetir código ou transferir parâmetros de um nível mais alto para um nível mais baixo entre funções e componentes[25]. Implementada então a *store*, foi utilizada uma abordagem chamada *ducks* onde é proposto que o ficheiro que será utilizado para a gestão do estado global associado a um componente esteja sempre ao mesmo nível da definição do próprio componente como pode ser observado na Figura 14.

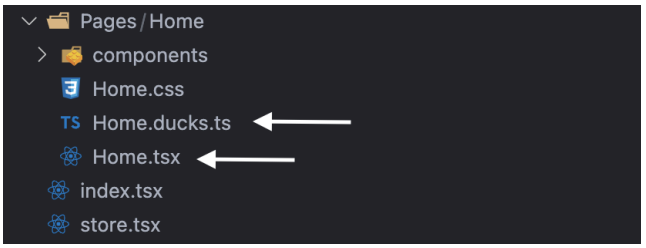


Figura 14 - Ecrã Home e ficheiro ducks definidos ao mesmo nível

O ficheiro "Home.ducks.ts" guarda estruturas de dados, ilustradas na Figura 15, que facilitam uma gestão central e distribuição para qualquer ponto da aplicação. São guardados dados referentes a detalhes dos fotobiorreatores conectados, leituras dos parâmetros de temperatura e pH, detalhes de uma experiência de crescimento de microalgas e por fim estados de controlo do sistema, como por exemplo, estado "isLoading" para controlar o tempo de espera de resposta após um pedido à API.

No caso específico da informação de identificação dos fotobiorreatores, esta é guardada na variável "reactors", que é uma lista do tipo "IReactor" onde cada elemento é composto por 4 propriedades sendo elas o "id" identificador único do PBR, "active" determina se um PBR está conectado ou não, "name" identificador do nome do PBR e "selected" que determina se pelo interface de utilizador o PBR selecionado.

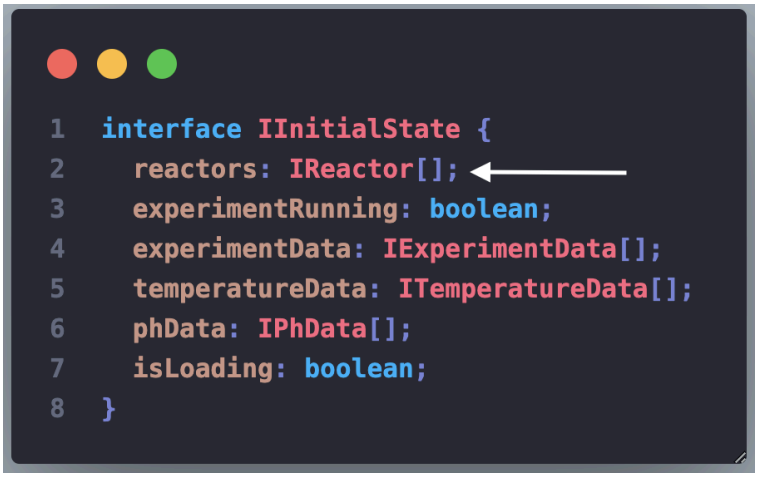


Figura 15 - Estrutura do estado central da aplicação

3.3.A.2. Gestão dos perfis e configurações do fotobiorreator

Após obtida a informação sobre os fotobiorreatores conectados, o próximo passo será configurar o PBR selecionado para posteriormente começar uma experiência de crescimento de microalgas. Foi implementado um menu expansível na lateral esquerda do interface de utilizador com o objetivo de fornecer a possibilidade de configurar o fotobiorreator ao nível de parametrização dos sensores e gerir os diferentes perfis de configuração para a reutilização em futuras experiências.

A configuração dos valores dos LEDs de cor branca, vermelho, azul e verde é feita através de um *slider* onde o utilizador escolhe o valor de cada cor que quer aplicar aos LEDs presentes no protótipo do fotobiorreator. Ao nível da temperatura o utilizador pode escolher entre 4 opções diferentes, "Constant", "Daily Cycle", "Sine Wave" e "Pulse" onde os valores e a lógica de funcionamento desta configuração é gerida pelo microcontrolador *Arduino*, ao selecionar uma das opções o valor da temperatura pode manter-se constante ou mudar conforme uma lógica específica, por exemplo, na opção "Daily Cycle" a temperatura irá mudar conforme a transição do dia para a noite. A configuração da válvula de CO_2 pode ser feita de forma manual, sempre ligada ou sempre desligada (ON/OFF), ou de forma automática onde esta é ativada quando um valor de pH definido pelo utilizador é atingido como se pode observar pela Figura 16.

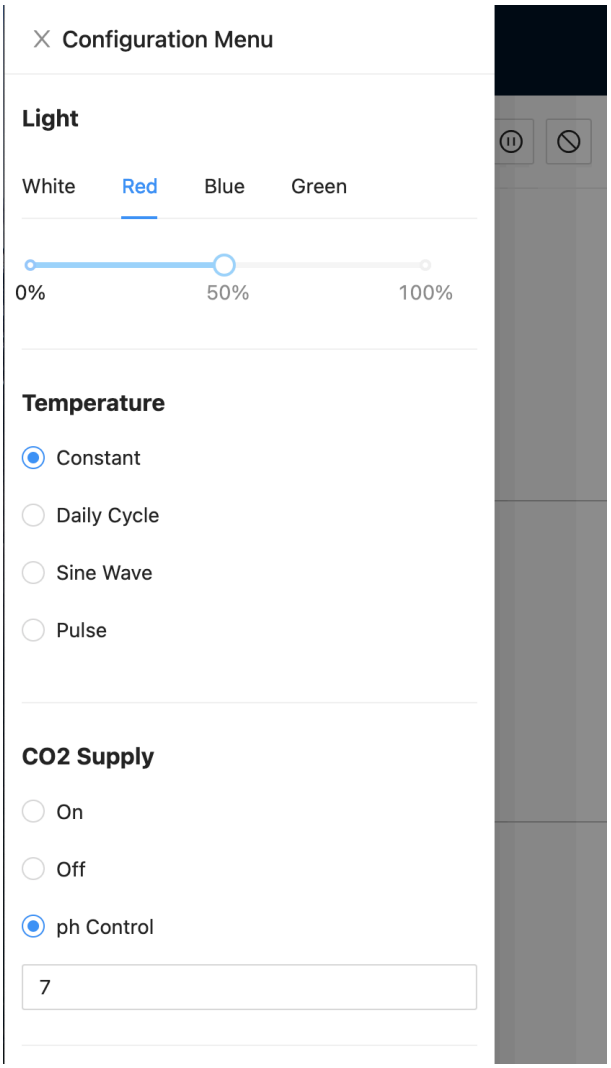


Figura 16 - Menu de configuração e escolha do valor de pH

A válvula de ar pode ser controlada da mesma forma manual onde a mesma está sempre ligada ou desligada (ON/OFF). A configuração do modo de cultivação é semelhante ao da temperatura, onde uma das 4 opções diferentes podem ser selecionadas para definir o modo de cultivação das microalgas, "Batch", "Semi-Continuous", "Continuous" e "Turbidostat", cada uma das opções aplica um método diferente de gestão da cultura no meio de cultivação onde a ação que acontece em cada uma das opções é responsabilidade do microcontrolador, normalmente este processo serve para adicionar ou retirar meio de cultivo de modos diferentes. Existem ainda duas bombas que funcionam de forma semelhante às válvulas de ar e CO_2, onde a sua configuração é feita de forma manual com o mecanismo de ON/OFF como mostra a Figura 17.

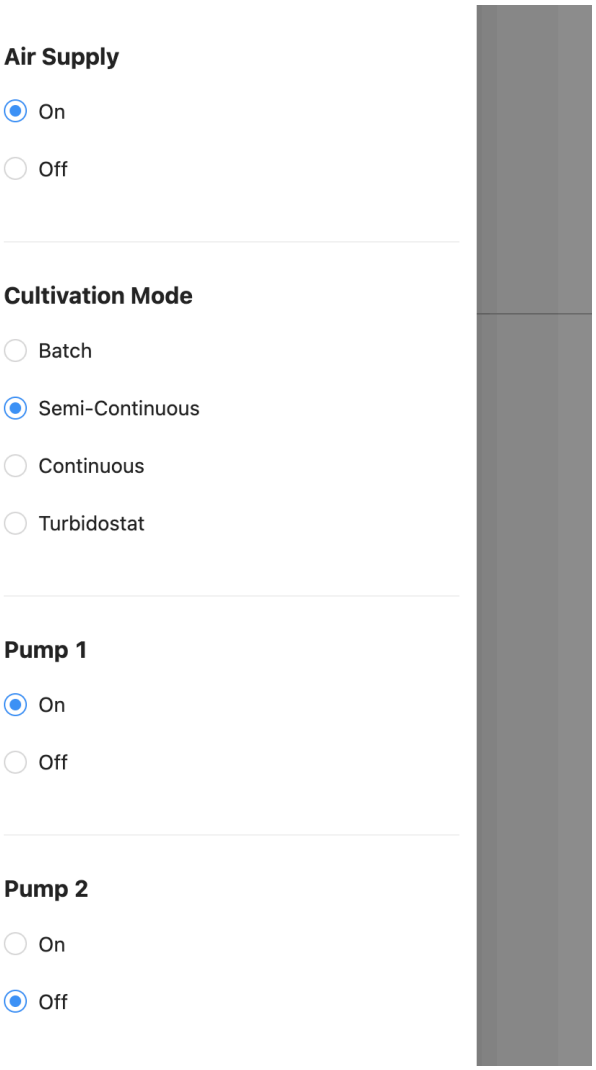


Figura 17 - Menu de configuração e escolha do valor de cada uma das bombas

No menu é possível também definir uma escala que irá controlar a cadência de pedido de dados e escala temporal dos valores de temperatura e pH no decorrer da experiência para preencher os respectivos gráficos. Após o utilizador definir a configuração pretendida, existe ainda a possibilidade de gravar o perfil criado com um nome à escolha como mostra a Figura 18 abaixo ou importar um perfil existente que irá preencher as opções descritas anteriormente com valores associados ao perfil escolhido.

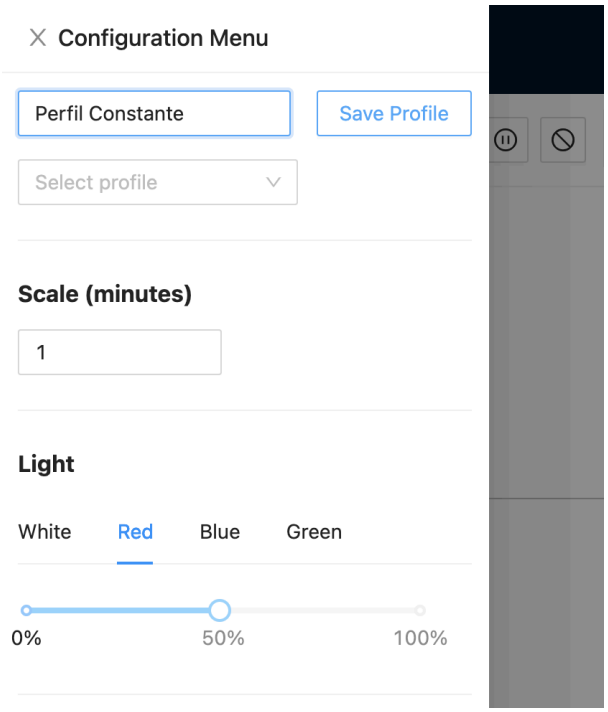


Figura 18 - Menu de configuração e criação de um novo perfil de configuração

Cada uma das opções de configuração no menu, envia um pedido específico para a API onde posteriormente é enviado um comando para o microcontrolador *Arduino* e refletida a configuração pretendida no fotobiorreator. Ao selecionar um perfil de configuração existente, cada uma das opções será preenchida com os valores configurados anteriormente e será feito um encadeamento de pedidos à API de forma sequencial, isto porque cada pedido equivale a um comando específico no fotobiorreator (os comandos existentes podem ser consultados através da Tabela 1).

É importante referir que todas as configurações ou possibilidades referidas anteriormente estão apenas disponíveis para uma fase pré-experiência, quando uma experiência está a decorrer a alteração de valores de configuração não é possível.

3.3.a.3. Calibração da sonda de pH

Normalmente uma experiência de crescimento de microalgas dura no mínimo 12 dias, dada a longa duração a sonda que mede o valor de pH precisa de ser calibrada. Através da interface de utilizador é possível fazer o processo de calibração da sonda, ilustrada na Figura 19 abaixo, numa fase pré-experiência ou com uma experiência em pausa, onde ao começar a calibração o valor de pH será lido de forma constante e o utilizador como primeiro passo irá selecionar o valor de pH 7 para o ajuste da sonda a esse valor, após o sinal de calibração completa para o valor de pH 7, o mesmo processo será repetido para a calibração da sonda para os valores de pH 4 e pH 10.

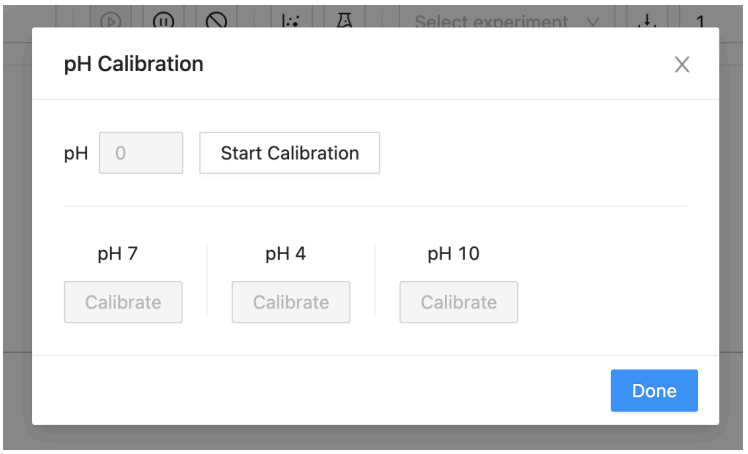


Figura 19 - Interface de calibração da sonda de pH

Em termos de implementação, ao começar a calibração da sonda de pH é feito um pedido à API específico e é iniciado um temporizador de 2 em 2 segundos de forma constante que irá ser monitorizado e onde será feito um outro pedido para requisitar informação sobre o atual valor de pH da sonda. O pedido feito à API retorna dois valores diferentes, uma mensagem que poderá estar vazia e um valor de pH, quando é recebida uma mensagem é decifrado o valor de pH nela incluído, como mostra a Figura 20, determinando assim qual o próximo valor de pH a ser calibrado ou quando a calibração da sonda está concluída.

```
async function getPhValue() {
  try {
    const response = await axios.get(phValueRoute);

    const phMessage = response?.data?.ph_message;
    setPhMessage(phMessage);

    const phStep = phMessage.match(/\d+/)[0];
    handlePHStep(phStep);

    const phValue = response?.data?.ph_value;
    setPhValue(phValue);
  } catch (error) {
    alert(error);
  }
}
```

Figura 20 - Função para a calibração da sonda de pH

3.3.A.4. Gestão de uma experiência de crescimento de microalgas (UI)

Para a realização de uma experiência de controlo do crescimento de microalgas, o primeiro passo necessário é a configuração do fotobiorreator referida no subcapítulo 3.3.3 gestão de perfis e configurações do fotobiorreator, onde são escolhidas configurações ajustadas ao crescimento de

uma espécie específica através do menu expansível lateral. Após escolhidas as configurações pretendidas, é necessário preencher apenas o nome da experiência que irá ser realizada e clicar no botão de "play" na barra de tarefas secundária ilustrada abaixo na Figura 21.

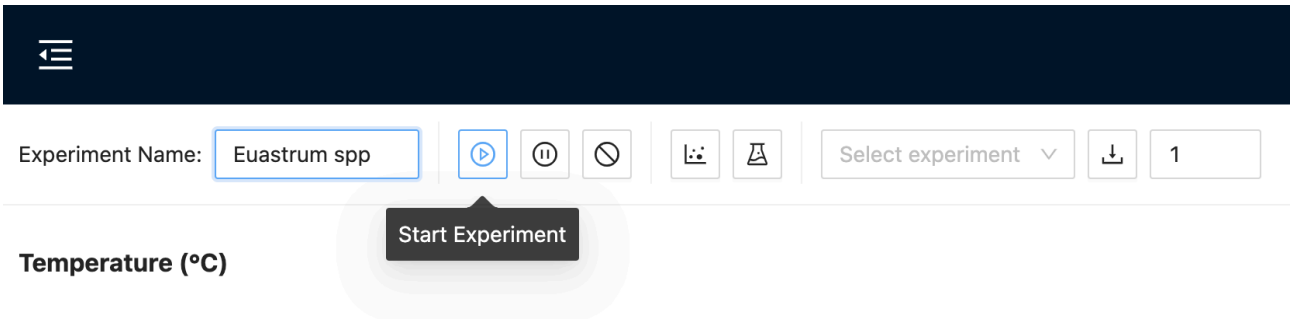


Figura 21 - Começo de uma experiência

Para começar uma experiência existe uma função específica que se encarrega fazer uma série de processos diferentes, ilustrados também na Figura 22, sendo eles:

1. Verificar se foi escolhida uma configuração para o fotobiorreator no menu expansível;
2. Através do mecanismo de *dispatch*, modificar no estado global o valor da variável "loading" para *true*;
3. Fazer o pedido à API para começar uma experiência através da rota específica /run, onde é passado no URI o número de série do fotobiorreator que está selecionado e como parâmetros o nome escolhido para a experiência, o "id" do reator selecionado e o "id" do perfil selecionado;
4. Através do mecanismo de *dispatch*, modificar no estado global o valor da variável "experimentRunning" para *true*;
5. Começar um contador onde de 4 em 4 segundos onde será feito um outro pedido para requisitar informação dos dados referentes à temperatura e pH na mesma cadência temporal;
6. Caso aconteça algum erro em alguns dos processos, o mesmo será mostrado ao utilizador através de uma mensagem de alerta;
7. Após todos os processos serem bem sucedidos, é feita novamente uma modificação no estado global o valor da variável "loading" para *false*.


```
async function startExperiment() {
    if (!selectedProfileId) {
      return alert('Please select a profile before starting an experiment!');
    }

    try {
      dispatch(setIsLoading(true));
      await axios.post(`${startExperimentRoute}/${selectedReactor?.serial}`, {
        experimentName: experimentName,
        reactorId: selectedReactor?.id,
        profileId: selectedProfileId,
      });
      dispatch(setExperimentRunning(true));
      setTimeout(() => setStartTimer(true), 4000);
    } catch (error) {
      alert(error);
    } finally {
      dispatch(setIsLoading(false));
    }
  }
```

Figura 22 - Função para começar uma experiência

Após o contador estar ativo, será feito um pedido de acordo com a escala escolhida pelo utilizador em minutos para requisitar os dados de temperatura e pH de forma constante, onde após a receção desses mesmos dados serão preenchidos os respectivos gráficos presentes no centro da aplicação. Ambos os gráficos foram implementados através da importação, modificação e configuração do componente "Line" da livraria "@ant-design/plots"[26], onde no eixo do X é representada a escala temporal e no eixo do Y o valor da temperatura lido no momento que foi requisitado. Tendo em conta que as experiências podem demorar dias, mesmo com uma escala de 1 minuto, os dados geridos pelo interface de utilizador podem ser demasiados e podem causar problemas de performance pela vasta quantidade de registos, por essa mesma razão foi implementado um processo de *downsampling* de modo a reduzir a quantidade de dados processados limitando-os a um máximo de 1000 registos. O utilizador tem ainda a possibilidade de fazer *zoom* num período temporal específico através do ajuste dos *handlers* presentes na parte inferior dos gráficos para uma possível análise com maior foco.

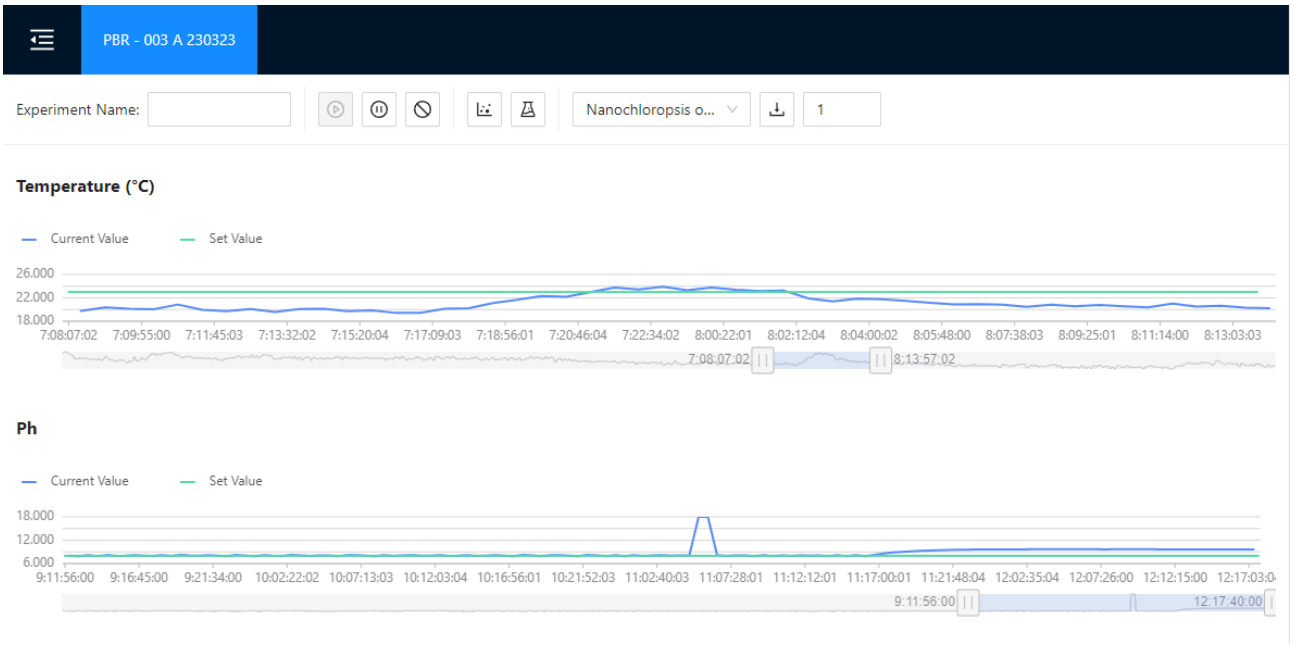


Figura 23 - Zoom em períodos temporais específicos através dos handlers dos gráficos

Existe por vezes a necessidade de pausar a experiência para uma análise manual do meio de cultivo por parte do utilizador, é possível que seja necessário alguma troca de compostos no meio de cultura e é essencial que a experiência volte a correr a partir do momento em que foi pausada. Ao pausar e recomeçar uma experiência, são efetuados os mesmos processos de começo de experiência descritos anteriormente e ilustrados na Figura 22, com a única diferença de começarem num ponto específico temporal.

Após terminada uma experiência, poderá ser feita uma análise geral dos dados através dos gráficos de temperatura e pH ou para uma análise mais detalhada poderá ser feito o descarregamento dos dados gravados em formato de CSV com uma escala temporal em minutos escolhida pelo utilizador como mostra a Figura 24 abaixo.

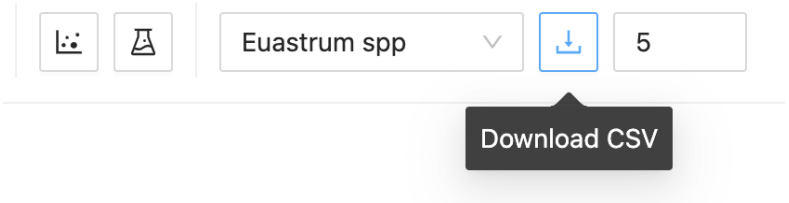
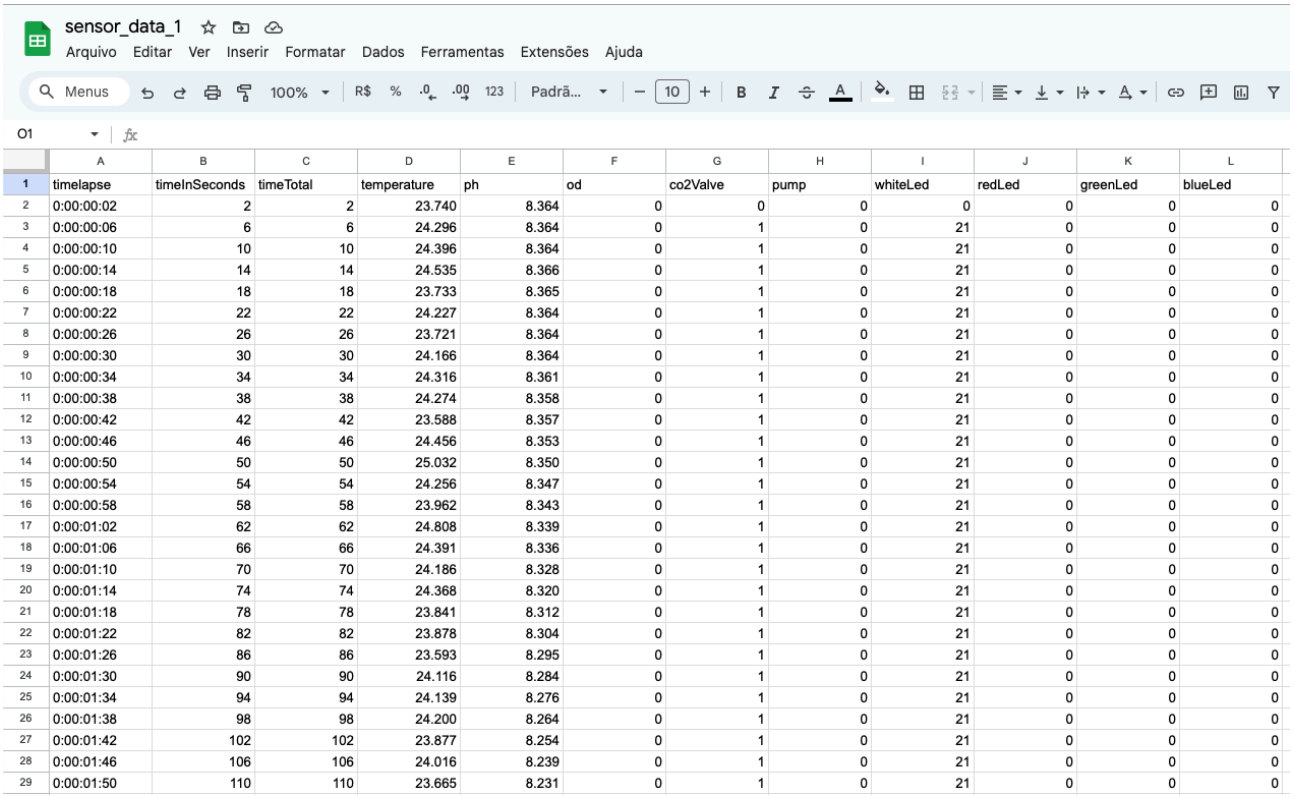


Figura 24 - Descarregamento da experiência em formato CSV

Um ficheiro de CSV de uma experiência tem registos dos seguintes parâmetros:

1. timelapse - Tempo decorrido de uma experiência no formato DD:HH:MM:SS;
2. timeInSeconds - Tempo decorrido de uma experiência em segundos;
3. timeTotal - Tempo total decorrido de uma experiência em segundos (removido o tempo de eventuais pausas);
4. temperature - Valor lido pelo sensor de temperatura presente no fotobiorreator;
5. ph - Valor lido pela sonda de pH presente no fotobiorreator;
6. od - Valor lido pelo sensor de densidade ótica presente no fotobiorreator, este valor determina o crescimento de microalgas sem a necessidade de uma análise manual;
7. co2Valve - Valor entre 0 ou 1, onde 0 representa que a válvula de CO_2 está desligada e 1 que está ligada;
8. pump - Valor entre 0 ou 1, onde 0 representa que a bomba de fluxo do meio está desligada e 1 que está ligada;
9. whiteLed - Intensidade de luz branca aplicada nos LEDs do fotobiorreator;
10. redLed - Intensidade de luz vermelha aplicada nos LEDs do fotobiorreator;
11. greenLed - Intensidade de luz verde aplicada nos LEDs do fotobiorreator;
12. blueLed - Intensidade de luz azul aplicada nos LEDs do fotobiorreator.

A Figura 25 abaixo mostra um exemplo de um *dataset* gerado a partir de um teste de uma experiência de 3 minutos.



sensor_data_1

Arquivo Editar Ver Inserir Formatar Dados Ferramentas Extensões Ajuda

	A	B	C	D	E	F	G	H	I	J	K	L
1	timelapse	timeInSeconds	timeTotal	temperature	ph	od	co2Valve	pump	whiteLed	redLed	greenLed	blueLed
2	0:00:00:02	2	2	23.740	8.364	0	0	0	0	0	0	0
3	0:00:00:06	6	6	24.296	8.364	0	1	0	21	0	0	0
4	0:00:00:10	10	10	24.396	8.364	0	1	0	21	0	0	0
5	0:00:00:14	14	14	24.535	8.366	0	1	0	21	0	0	0
6	0:00:00:18	18	18	23.733	8.365	0	1	0	21	0	0	0
7	0:00:00:22	22	22	24.227	8.364	0	1	0	21	0	0	0
8	0:00:00:26	26	26	23.721	8.364	0	1	0	21	0	0	0
9	0:00:00:30	30	30	24.166	8.364	0	1	0	21	0	0	0
10	0:00:00:34	34	34	24.316	8.361	0	1	0	21	0	0	0
11	0:00:00:38	38	38	24.274	8.358	0	1	0	21	0	0	0
12	0:00:00:42	42	42	23.588	8.357	0	1	0	21	0	0	0
13	0:00:00:46	46	46	24.456	8.353	0	1	0	21	0	0	0
14	0:00:00:50	50	50	25.032	8.350	0	1	0	21	0	0	0
15	0:00:00:54	54	54	24.256	8.347	0	1	0	21	0	0	0
16	0:00:00:58	58	58	23.962	8.343	0	1	0	21	0	0	0
17	0:00:01:02	62	62	24.808	8.339	0	1	0	21	0	0	0
18	0:00:01:06	66	66	24.391	8.336	0	1	0	21	0	0	0
19	0:00:01:10	70	70	24.186	8.328	0	1	0	21	0	0	0
20	0:00:01:14	74	74	24.368	8.320	0	1	0	21	0	0	0
21	0:00:01:18	78	78	23.841	8.312	0	1	0	21	0	0	0
22	0:00:01:22	82	82	23.878	8.304	0	1	0	21	0	0	0
23	0:00:01:26	86	86	23.593	8.295	0	1	0	21	0	0	0
24	0:00:01:30	90	90	24.116	8.284	0	1	0	21	0	0	0
25	0:00:01:34	94	94	24.139	8.276	0	1	0	21	0	0	0
26	0:00:01:38	98	98	24.200	8.264	0	1	0	21	0	0	0
27	0:00:01:42	102	102	23.877	8.254	0	1	0	21	0	0	0
28	0:00:01:46	106	106	24.016	8.239	0	1	0	21	0	0	0
29	0:00:01:50	110	110	23.665	8.231	0	1	0	21	0	0	0

Figura 25 - *Dataset* de uma experiência de teste

3.3.B API

Em resumo uma API (*Application Programming Interface*) é um conjunto de regras e protocolos que permitem diferentes aplicações comunicarem e interagirem entre si, onde são definidos métodos de comunicação e estruturas de dados para tais interações. Uma RESTful API é um tipo de API que segue os princípios REST (Representational State Transfer), muito utilizada em serviços Web, com recursos identificados através de URLs. São utilizados os métodos standard HTTP (GET, POST, PUT, DELETE) para a realização de operações em recursos como pedidos de informação específicos ou alteração de dados.

A aplicação servidor serve uma RESTful API que foi implementada utilizando como livraria o *Flask* e como linguagem de programação *Python*. A escolha desta tecnologia foi feita com base na sua abordagem minimalista, simplicidade de desenvolvimento, escalabilidade eficiente, fácil integração com plataformas da nuvem como o *Azure Cloud*, compatibilidade com frameworks de frontend como o *React* e pela sua fácil integração com bases de dados como o *SQLite*.

O objetivo principal da API é a gestão da comunicação entre a interface do utilizador e o fotobiorreator, onde são fornecidas primeiramente rotas de comunicação para a interação com a interface de utilizador onde pedidos são efetuados e dados de resposta são retornados, onde é realizada também a gravação de dados numa base de dados, dados esses tais como a identificação

dos fotobiorreatores conectados ao servidor, parâmetros de configuração ou perfis de utilizador que servem por sua vez para a reutilização de configurações efetuadas anteriormente pelo utilizador em experiências diferentes e por fim são gravados os dados das experiências realizadas no âmbito de controlo do crescimento de microalgas. É nesta parte da aplicação que é estabelecida também a comunicação com o microcontrolador *Arduino* onde comandos (consultar Tabela 1) são enviados e ações são efetuadas ao nível dos sensores do fotobiorreator, sejam elas ações como a alteração de um valor de uma luz LED ou consulta como a leitura do valor atual do pH. A conexão e a troca de comandos é possível através da utilização de uma livraria chamada *pySerial*, que é uma ferramenta em *Python* que facilita a comunicação entre um computador e dispositivos externos como microcontroladores via portas série como por exemplo portas USB.

Detalhados os objetivos principais da aplicação servidor, será feita uma descrição das funcionalidades implementadas de forma individual de modo a facilitar a leitura.

3.3.b.1. Inicialização da aplicação e funcionalidades associadas

Ao inicializar a aplicação servidor o primeiro processo realizado é o reconhecimento dos fotobiorreatores conectados ao sistema, onde é utilizada uma função que reconhece as portas série disponíveis e por cada uma delas é estabelecida uma conexão com cada um dos microcontroladores de modo a disponibilizar uma troca de comandos e realizar ações nos respectivos fotobiorreatores, este processo está ilustrado na Figura 26 abaixo.

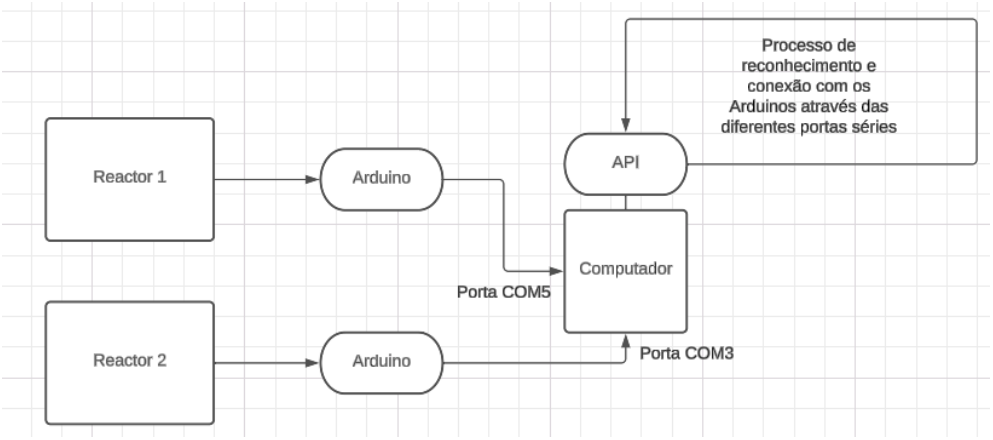


Figura 26 - Processo de inicialização da API

Tal como referido anteriormente, como primeiro passo foi implementada então uma função que faz o reconhecimento dos microcontroladores conectados e retorna uma lista de portas série disponíveis a estabelecer uma conexão, como mostra a Figura 27 abaixo.

```python
def list_serial_ports():
    ports = serial.tools.list_ports.comports()
    available_ports = [port.device for port in ports]
    return available_ports
```

Figura 27 - Função que retorna a lista de portas série disponíveis

Por cada uma das portas série disponíveis retornadas, é criado um processo que corre no *background* da aplicação onde é realizada a inicialização de cada uma das portas série. Esta função de inicialização denominada por "initialize_ports" tem como processos a abertura da conexão com cada uma das portas série disponíveis através da utilização de um método específico da livraria *pySerial*, a criação de uma lista que facilitará o acesso e a interação a nível global com os microcontroladores conectados e a inicialização de um outro processo de leitura constante de possíveis mensagens recebidas nas diferentes portas série. Cada um dos processos referidos anteriormente da função de inicialização estão ilustrados na Figura 28.

```python
def initialize_ports(serial_port):
    global serial_ports
    # Initialize serial port
    serial_port_initializer = serial.Serial(serial_port, 9600)
    serial_port_dict = {'name': serial_port_initializer.name, 'serial': serial_port_initializer,
                        'command': None, 'message': None, 'experiment_id': None}
    serial_ports.append(serial_port_dict)

    # Start the reading thread
    read_thread = threading.Thread(
        target=read_serial, args=(serial_port_dict,))
    read_thread.daemon = True
    read_thread.start()
```

Figura 28 - Função de inicialização de cada uma das portas série

A função "read_serial", ilustrada na Figura 29, inicializada a partir do processo anterior, corre numa *thread* à parte e tem como objetivo principal a gestão das diferentes possíveis mensagens recebidas. Existem dois comandos em particular que precisam de ser geridos de maneira específica, os comandos RUN e CLP, onde no primeiro estamos presentes na inicialização de uma experiência de controlo de crescimento de microalgas logo os dados da mesma serão gravados na base de dados

e posteriormente retornados para o interface de utilizador. No caso do segundo comando, o processo de calibração da sonda de pH foi inicializado, este processo consiste numa leitura constante do valor de pH e retorno desse mesmo valor para o interface do utilizador acompanhado de uma mensagem de controlo do estado do processo, de modo a informar o utilizador de qual ação a ser realizada em cada um dos passos até à conclusão do mesmo. Caso o comando recebido não for nenhum dos dois em particular, apenas é feita a atualização na estrutura da lista de portas série definida a nível global da aplicação com o objetivo de fornecer sempre a informação mais atualizada sobre a última mensagem recebida e o último comando enviado.

```python
def read_serial(serial_port_dict):
    while True:
        if serial_port_dict['serial'] and serial_port_dict['serial'].in_waiting > 0:
            current_message = serial_port_dict['serial'].readline().decode()

            global serial_ports, ph_value, ph_message
            for serial in serial_ports:
                # Match the serial that is receiving a message
                if serial['name'] == serial_port_dict['name']:
                    # Updates received message
                    serial['message'] = current_message

                    # Saves the experiments data in case we are running an experiment
                    if serial['command'] == 'RUN':
                        message = current_message.strip().split(" ")
                        data = [element for element in message if element]
                        saveExperimentData(data, serial)

                     # PH Calibration Running - CLP or EMPTY for next value to calibrate
                    if serial['command'] == 'CLP' or serial['command'] == ' ':
                        if current_message[0].isdigit():
                            ph_value = current_message.strip()
                        elif "pH" in current_message:
                            ph_message = current_message
```

Figura 29 - Função de gestão de mensagens recebidas

3.3.B.2. ENDPOINTS DISPONÍVEIS NA API

No subcapítulo anterior foram descritos os processos que correm no passo de inicialização da aplicação servidor, como também a maneira como são geridas as trocas de comandos e mensagens entre o interface de utilizador, a API, o microcontrolador e por fim o fotobiorreator. Foram exploradas as soluções de implementação para a conexão com diferentes microcontroladores, gestão

das estruturas de dados, variáveis de controlo da aplicação e processos que correm no *background* de modo a providenciar uma constante conexão com as portas série com o propósito de monitorizar e processar qualquer tipo de comando ou mensagem recebida. Tendo em conta que para este projeto em específico um microcontrolador *Arduino* corre um *software* próprio desenvolvido pelo laboratório *GreenCoLab*, apenas um comando é processado de cada vez então a abordagem principal para a implementação da aplicação servidor foi uma API que contém 1 rota por comando, a lista de comandos e os parâmetros de controlo associados a cada comando podem ser consultados na Tabela 1 e nas respectivas notas.

Foram desenvolvidas várias rotas de modo a disponibilizar a troca de recursos, satisfazer pedidos por parte da aplicação cliente e a realizar diferentes ações no fotobiorreator. As primeiras rotas implementadas foram as de identificação e estado dos fotobiorreatores conectados, onde é feito um pedido aos microcontroladores conectados através das portas série disponíveis sendo depois retornados para a interface de utilizador os dados de identificação referentes aos fotobiorreatores conectados. Para o caso do pedido de identificação (Método GET, rota /identification, comando IDN?) é retornada uma lista de *strings*, como por exemplo ["GREENCOLAB", "PBR", "001", "A", "230323"], que contém informações diferentes em cada *string* constituindo em conjunto a identificação de um fotobiorreator. Para o caso do pedido de estado (Método GET, rota /status/serial_name, comando STA?) é incluído no URI do pedido o nome da porta série obtida anteriormente e é retornado um dos valores inteiros entre 0 e 2, onde 0 - Parado, 1 - A Decorrer, 2 - Pausado.

Após obtidas as informações sobre os PBRs conectados, o próximo conjunto de rotas implementadas estão relacionadas com a configuração e calibração do fotobiorreator numa fase pré experiência. Tal como referido anteriormente no subcapítulo Interface de utilizador, existe um menu expansível lateral que permite a configuração do PBR onde cada uma das opções disponíveis fará um pedido à API para a configuração específica em questão.

Foram então implementadas rotas individuais, onde em alguns casos são enviados comandos para o microcontrolador, com os seguintes propósitos:

1. Gravação de um perfil de configuração identificado através de um nome escolhido pelo utilizador, onde as opções de configuração escolhidas são gravadas numa base de dados. (Método POST, rota /profile/save/profile_name);
2. Consulta de informação de todos os perfis de utilizador existentes no sistema identificados através do nome. (Método GET, rota /profiles);

3. Definição da intensidade do valor das luzes LED, entre as cores, branco, vermelho, verde e azul. (Método POST, rota /led/color/serial_name, comandos WLD, RLD, GLD, BLD);
4. Definição do modo operacional do valor da temperatura, onde o valor inteiro presente no URI varia entre 0 e 4 sendo 0 - Constante, 1 - Ciclo diário, 2 - Onda sinusoidal, 3 - Onda pulso. (Método POST, rota /temperature/mode/serial_name/value, comando TMD);
5. Definição do modo operacional da válvula de CO_2, onde o valor inteiro presente no URI varia entre 0 e 4 sendo 0 - Desligado, 1 - Ligado, 2 - Controlado pelo valor *setpoint* do pH. (Método POST, rota /co2/mode/serial_name/value, comando CMD);
6. Definição do valor do *setpoint* de pH para o meio de cultivo. (Método POST, rota /ph/serial_name/value, comando PHS);
7. Definição do estado da válvula de oxigênio (ligada ou desligada), onde o valor inteiro presente no URI será 0 ou 1 sendo 0 - Desligado, 1 - Ligado. (Método POST, rota /air/serial_name/value, comando VAI);
8. Definição do modo operacional do regime de funcionamento no meio de cultivo, onde o valor inteiro presente no URI varia entre 0 e 4 sendo 0 - Lote, 1 - Semi contínuo, 2 - Pseudo contínuo, 3 - Turbidostato. (Método POST, rota /regime/mode/serial_name/value, comando MMD);
9. Definição do estado das bombas de circulação do meio de cultivo (ligadas ou desligadas), onde ambos os valores presentes no URI serão 0 ou 1 sendo 0 - Desligado, 1 - Ligado. (Método POST, rotas /pump1/serial_name/value e /pump2/serial_name/value, comandos PM1 e PM2).

Para a maior parte das rotas onde é enviada uma mensagem para uma porta série, o mesmo tipo de função foi implementada ao nível de código, ilustrada na Figura 30, onde o identificador da porta série e um valor associado estão normalmente presentes no URI. A partir do identificador e acedendo à variável global "serial_ports" que guarda em memória a lista de controladores disponíveis, é enviada uma mensagem para a porta série específica onde uma ação ou uma consulta de valores é realizada através dos sensores do PBR.

```python
@app.route('/temperature/mode/<serial_name>/<value>', methods=["POST"])
def setTemperatureOperatingMode(serial_name, value: int):
    global serial_ports
    for serial in serial_ports:
        if serial_name == serial['name']:
            command = f'TMD {value}'
            serial['serial'].write(f'{command}\n'.encode())
            time.sleep(0.5)

    return f'Temperature operating mode {value} set'
```

Figura 30 - Função para definir o modo operacional do valor da temperatura

É importante referir que por cada comando que é enviado para uma porta série, o valor da propriedade "command" presente na lista global de portas série conectadas é atualizado de modo a disponibilizar uma maneira de consultar o último comando enviado.

Definidas então as rotas para a configuração do fotobiorreator numa fase pré-experiência, o próximo passo será disponibilizar rotas para a realização de uma experiência de controlo do crescimento de microalgas. Rotas para a calibração do sensor de pH, começo, pausa e paragem de uma experiência foram disponibilizadas, assim como rotas para facilitar uma possível análise dos dados após a realização de uma experiência entre outras abaixo enumeradas:

1. Começo de uma experiência, onde são recebidos como recursos o nome da experiência definida pelo utilizador, o identificador do fotobiorreator onde será realizada a experiência e o identificador do perfil de configuração que será aplicado no decorrer da experiência. (Método POST, rota /run/serial_name, comando RUN);
2. Pausa de uma experiência, onde através do nome da experiência (recurso enviado no pedido), o valor das propriedades "command" e "experiment_id" presentes na lista global são atualizados de modo a identificar uma pausa na experiência com maior facilidade. (Método POST, rota /run/pause/serial_name, comando PAU);
3. Paragem de uma experiência. (Método POST, rota /stop/serial_name, comando STP);
4. Consulta dos dados de uma experiência através da escala temporal definida pelo utilizador, onde através do nome da experiência são consultados na base de dados e retornados os valores mais atualizados da mesma. (Método GET, rota /run/get-data/experiment_name);

5. Começo do processo de calibração da sonda de pH, onde é apenas enviado o comando CLP através de uma implementação semelhante à da função definida. (Método POST, rota /calibration/serial_name, comando CLP);
6. Continuação para o passo seguinte do processo de calibração da sonda de pH, onde é enviado um espaço em branco para a porta série (modo definido para passar para o próximo passo no *software* do microcontrolador). (Método POST, rota /calibration/next/serial_name);
7. Consulta do valor de pH atual através da sonda. (Método GET, rota /ph/value);
8. Mudança do estado da válvula de CO_2 (ligar ou desligar). (Método POST, rota /co2/serial_name/value, comando VCO);
9. Consultar a lista de experiências realizadas anteriormente. (Método GET, rota /get-experiments-list, comando GET);
10. Consultar os dados de uma experiência específica utilizando uma escala escolhida pelo utilizador. (Método GET, rota /get-experiment-data/experiment_id/scale);
11. Consultar os dados de uma experiência específica utilizando uma escala escolhida pelo utilizador onde é retornado um ficheiro de CSV. (Método GET, rota /get-experiment/csv/experiment_name/experiment_id/scale).

Apenas algumas rotas efetuam gravações ou consultas à base de dados, estas serão detalhadas e analisadas no subcapítulo seguinte.

3.3.b.3. Base de dados

As funcionalidades cruciais da aplicação dependem dos dados ou recursos que são trocados entre o interface de utilizador e o fotobiorreator. Sem a gravação de tais dados numa base de dados não seria possível manter consistência e fiabilidade na aplicação, é crucial gravar dados referentes a informações sobre os fotobiorreatores conectados, perfis de utilizador que permitem a configuração do PBR e experiências de crescimento de microalgas.

Para alcançar tais objetivos foi utilizada uma base de dados *SQLite* pela facilidade de integração com a arquitetura da API implementada em *Python*, sem dependências de um servidor e pela simplicidade de configuração, implementação e utilização em projetos deste género.

Foram então criadas quatro tabelas diferentes para alcançar os objetivos referidos anteriormente, começando pela informação referente aos fotobiorreatores conectados, foi criada a tabela “reactors” com as seguintes colunas:

1. id - Chave primária, única, do tipo INTEGER que serve para a identificação dos diferentes PBRs;
2. name - Coluna única, não nula, do tipo TEXT que contém o nome do PBR;
3. serial - Coluna do tipo TEXT que contém o nome da porta série que o PBR está conectado.

Tal como foi analisado no subcapítulo anterior, ao inicializar a aplicação servidor, o primeiro processo que decorre é a identificação e conexão com os diferentes PBRs através das portas série onde por sua vez os dados da conexão são gravados na base de dados de modo a registar em que porta série foi conectado um PBR pela última vez.

Após termos informação sobre os PBRs, o próximo passo é a configuração do mesmo, para tal foi criada a tabela "profiles" com o propósito de registar a configuração efetuada pelo utilizador onde a mesma será gravada através do nome do perfil de configuração único. A tabela referida contém as seguintes colunas:

1. id - Chave primária, única, do tipo INTEGER que serve para a identificação dos diferentes perfis de configuração;
2. name - Coluna única, não nula, do tipo TEXT que contém o nome atribuído pelo utilizador ao perfil de configuração;
3. whiteLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz branca;
4. redLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz vermelha;
5. greenLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz verde;
6. blueLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz azul;
7. temperature - Coluna do tipo INTEGER que contém um valor entre 0 e 4, associado ao modo operacional da temperatura;
8. co2Valve - Coluna do tipo INTEGER que contém um valor entre 0 e 1, onde é definido se a válvula de CO_2 ficará aberta ou fechada durante a experiência;
9. ph - Coluna do tipo INTEGER que contém o valor de *setpoint* do pH;
10. airValve - Coluna do tipo INTEGER que contém um valor entre 0 e 1, onde é definido se a válvula de oxigênio ficará aberta ou fechada durante a experiência;
11. cultivationMode - Coluna do tipo INTEGER que contém um valor entre 0 e 4, onde é definido modo operacional do regime de funcionamento no meio de cultivo;

12. pump1 - Coluna do tipo INTEGER que contém um valor entre 0 e 1, onde é definido se a bomba de circulação de meio número 1 ficará ligada ou desligada durante a experiência;
13. pump2 - Coluna do tipo INTEGER que contém um valor entre 0 e 1, onde é definido se a bomba de circulação de meio número 2 ficará ligada ou desligada durante a experiência;
14. scale - Coluna do tipo INTEGER em que por defeito o valor é 1, onde é definida a escala escolhida pelo utilizador para a cadência temporal com que os dados serão retornados para a interface de utilizador (valor em minutos).

De modo a conectar um fotobiorreator, um perfil de configuração e uma experiência de crescimento de microalgas foi criada a tabela "experiments" onde é feita essa mesma conexão. O nome de cada experiência escolhido pelo utilizador no interface, é gravado juntamente com os identificadores das tabelas "reactors" e "profiles", constituindo assim a seguinte estrutura:

1. id - Chave primária, única, do tipo INTEGER que serve para a identificação das diferentes experiências de crescimento de microalgas;
2. name - Coluna do tipo TEXT que contém o nome escolhido pelo utilizador para a experiência que será realizada;
3. reactor_id - Chave estrangeira, única, do tipo INTEGER que guarda o valor do identificador primário da tabela "reactors";
4. profile_id - Chave estrangeira, única, do tipo INTEGER que guarda o valor do identificador primário da tabela "profiles";

Por fim, para a realização de uma experiência e para uma possível análise dos *datasets* obtidos ao longo do seu decorrer, foi criada uma tabela "experimentsData" onde são guardados os dados enviados pelo microcontrolador sob a forma de mensagens constantes (4 em 4 segundos) para uma porta série. Dados esses referentes às leituras de alguns dos sensores do PBR, onde foram criadas as seguintes colunas para o seu armazenamento:

1. id - Chave primária, única, do tipo INTEGER que serve para a identificação de um registo de uma leitura feita no PBR a um determinado momento numa experiência específica;
2. timelapse - Coluna do tipo TEXT que contém o tempo decorrido da experiência até ao momento com o seguinte formato DD:HH:MM:SS, onde DD significa dias, HH horas, MM minutos e SS segundos;
3. timeInSeconds - Coluna do tipo TEXT que contém o tempo decorrido da experiência até ao momento com o formato em segundos (considerando o tempo de eventuais pausas);

4. timeTotal - Coluna do tipo TEXT que contém o tempo total decorrido de uma experiência até ao momento em segundos (não considerando o tempo de eventuais pausas);
5. temperature - Coluna do tipo REAL que contém o valor, em graus celsius, da temperatura atual no meio de cultivo;
6. ph - Coluna do tipo REAL que contém o valor atual do pH no meio de cultivo;
7. co2Valve - Coluna do tipo INTEGER que contém o valor 0 - desligado ou 1 - ligado, da válvula de CO_2;
8. pump - Coluna do tipo INTEGER que contém o valor 0 - desligado ou 1 - ligado, da bomba de circulação do meio;
9. od - Coluna do tipo INTEGER que contém o valor da densidade ótica que serve para identificar o crescimento de microalgas no meio de cultivo;
10. whiteLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz branca;
11. redLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz vermelha;
12. greenLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz verde;
13. blueLed - Coluna do tipo INTEGER que contém o valor da intensidade do LED de luz azul;
14. experiment_id - Chave estrangeira, única, do tipo INTEGER que guarda o valor do identificador primário da tabela “experiments”.

No subcapítulo seguinte será feita uma conexão geral entre o interface de utilizador, a API, a base de dados, o microcontrolador e o fotobiorreator de modo a tornar mais explícito o processo de realização de uma experiência de controlo do crescimento de microalgas.

3.3.B.4. Gestão de uma experiência de crescimento de microalgas (API)

Para a gestão de uma de uma experiência eficaz no controlo do crescimento de microalgas através do sistema implementado, é necessário garantir a obtenção das informações relativas aos fotobiorreatores conectados e realizar uma configuração específica desses mesmos fotobiorreatores adequada para garantir uma simulação com a maior proximidade das condições ideais de crescimento de uma espécie de microalgas específica.

Após obtidas as informações sobre os fotobiorreatores conectados às diferentes portas série e escolhidas as configurações pretendidas ajustadas às condições mais próximas das ideais para o

crescimento de uma espécie específica (ambos os processos estão detalhados no subcapítulo 3.3.b.2 Endpoints disponíveis na API), o próximo passo será fazer a calibração da sonda de pH.

A calibração da sonda de pH serve para aumentar a precisão da leitura da sonda do valor de pH no meio de cultivo, é através do comando CLP enviado para o microcontrolador e da rota /calibration/serial_name que será iniciado o processo de calibração onde a sonda será ajustada aos valores 4, 7 e 10 por um período determinado de tempo.

Após calibrada a sonda do pH, uma experiência de crescimento de microalgas pode ser iniciada através do interface de utilizador onde quando escolhido um nome para a experiência e quando clicado o botão para iniciar uma experiência, um pedido é feito na API para a rota específica /run/serial_name, onde para a porta série conectada e posteriormente para o microcontrolador é enviado o comando RUN começando assim a experiência no fotobiorreator. Na mesma sequência de processos, são gravados os dados de ligação entre a experiência, o fotobiorreator e o perfil de configuração na tabela "experiments", tal como ilustrado na Figura 31 abaixo.

```
# Only save a new experiment if the experience was not paused before
if paused_experiment == False:
    # Open database connection
    connection = sqlite3.connect("GreenColabDB.db")
    connection.row_factory = sqlite3.Row
    cursor = connection.cursor()

    # Get 1 experiment by name
    cursor.execute("SELECT * FROM experiments WHERE name=?",
                   (experimentName,))
    experiment = cursor.fetchone()
    if experiment:
        raise Exception("Experiment name already exists!")

    # Add a new experiment to the database
    cursor.execute("INSERT INTO experiments (name, reactor_id, profile_id) VALUES (?,?,?)",
                   (experimentName, reactorId, profileId))
    experiment_id = cursor.lastrowid
```

Figura 31 - Verificação e gravação dos dados de ligação de uma experiência

Como referido anteriormente, é através da tabela "experiments" que é feita a ligação entre o fotobiorreator, o perfil de configuração e a experiência em si, disponibilizando assim uma forma simples de identificação dos diferentes componentes numa única tabela.

Ao enviar o comando RUN para o microcontrolador, será inicializada uma *thread* que irá correr no *background* que será responsável pela gestão das mensagens recebidas (de 4 em 4 segundos) na

API. Ao ser detetado que o comando atual é o comando RUN será utilizada uma função com o propósito de gravar os dados da experiência na base de dados, mais especificamente na tabela "experimentsData" onde é estabelecida uma ligação com a tabela "experiments" referida anteriormente, através da chave estrangeira "experiment_id".

Os dados da experiência são gravados no lado da API com uma cadência de 4 em 4 segundos conforme a implementação do *software* específico do microcontrolador, os utilizadores podem consultar os dados da experiência através da rota /get-experiment-data/experiment_id/scale com uma escala mínima de 1 minuto. Cada experiência tem uma duração mínima de 12 dias, para a gravação de dados na base de dados de 4 em 4 segundos ao longo do decorrer total da experiência o sistema não terá qualquer problema, mas para o caso do retorno de uma grande quantidade de dados para a aplicação cliente, onde são utilizados nos respectivos gráficos os dados das leituras obtidas dos parâmetros de temperatura e pH, podem existir problemas de performance consoante o dispositivo que está a aceder à experiência em questão. Para resolver possíveis problemas na interface de utilizador, a quantidade de dados será restringida a um máximo de 1000 registos processados em ambos os gráficos (mecanismo de *downsampling*) e o utilizador pode ainda aumentar a escala que controla a cadência de dados retornados da experiência, em por exemplo 30 minutos, controlando assim os eventuais problemas de performance do sistema. Na Figura 32 abaixo, está ilustrada a função que tem como objetivo retornar os dados da experiência para a interface de utilizador, onde é possível observar a utilização da escala e a *query* à base de dados que inclui o filtro temporal baseado na escala.

```python
@app.route('/get-experiment-data/<experiment_id>/<scale>', methods=["GET"])
def getExperimentData(experiment_id, scale):
    try:
        # Open database connection
        connection = sqlite3.connect("GreenColabDB.db")
        connection.row_factory = sqlite3.Row
        cursor = connection.cursor()

        scale_seconds = int(scale) * 60

        # Fetch experiments data
        cursor.execute("SELECT * FROM experimentsData WHERE experiment_id=? AND timeInSeconds % ? <= 4",
                       (experiment_id, scale_seconds))
        experiment_data = [dict(row) for row in cursor.fetchall()]

        return json.dumps(experiment_data)
    except Exception as error:
        print(str(error))
        return str(error)
    finally:
        # Save changes and close database connection
        connection.commit()
        connection.close()
```

Figura 32 - Função que retorna os dados da experiência

No decorrer da experiência pode haver necessidade de pausar a mesma para realizar alguma verificação manual no meio de cultivo, para tal através da rota /pause/serial_name é enviado para o microcontrolador o comando PAU que fará com que os dados retornados de 4 em 4 segundos sejam enviados de igual maneira para a porta série, tal como quando uma experiência está a decorrer, com o detalhe que para as propriedades “timelapse” e “timeInSeconds” os valores não serão alterados. A API consegue identificar que a experiência está em pausa, através do código ilustrado na Figura 33, onde o último comando enviado é atualizado na estrutura global que contém informação sobre as portas série.

```python
# Checks if the experiment was paused before
 paused_experiment = False
 for serial in serial_ports:
     if serial_name == serial['name']:
         if serial['command'] == 'PAU':
             paused_experiment = True
```

Figura 33 - Identificação que uma experiência está em pausa

Apesar de continuarem a ser enviadas mensagens de 4 em 4 segundos pelo microcontrolador, se uma experiência estiver em pausa não serão gravados dados na base de dados, podendo gerar uma

falha temporal na coluna “timeTotal” pois o valor continua a aumentar e só será gravado um novo valor ao recomeçar a experiência. Uma maneira de identificar que o sistema esteve em pausa é verificar se existe uma diferença maior que 4s entre registos da coluna “timeTotal”.

Ao recomeçar uma experiência, voltamos ao mesmo processo referido anteriormente com a diferença de começarmos num ponto específico no tempo. Quando alcançado o crescimento de microalgas pretendido no meio de cultivo, podemos parar a experiência clicando no botão de stop no interface de utilizador, onde será enviado o comando STP para o microcontrolador através da rota /stop/serial_name terminando assim a experiência na sua totalidade.

Poderá ser feita uma análise da experiência por parte do utilizador na própria aplicação através dos dados presentes nos gráficos de temperatura e pH, tendo em conta que esses mesmos dados podem ser restritivos devido aos ajustes efetuados na quantidade de dados processados de modo a evitar problemas de performance, para a realização de uma análise mais precisa, poderá ser descarregado o *dataset* da experiência com uma escala à escolha do utilizador, através de um pedido à rota /get-experiment/csv/experiment_name/experiment_id/scale. Este pedido retornará um ficheiro de CSV, que poderá ser posteriormente analisado em detalhe através da utilização de outras ferramentas específicas como o *Microsoft Excel*.

4 Resultados Experimentais

Para a realização de experiências de crescimento de microalgas no protótipo do fotobiorreator existem certas condições que têm que ser garantidas para não haver problemas com o sistema no geral. Uma experiência tem uma duração mínima de 12 dias, tempo mínimo para garantir algum resultado de crescimento de uma espécie de microalgas. O controlo dos parâmetros como a temperatura, o pH e o dióxido de carbono presente no meio de cultivo são essenciais para conseguir alcançar um resultado positivo.

Infelizmente o protótipo do fotobiorreator ainda não possui os componentes que permitem alterar o valor da temperatura e medir o valor da densidade ótica no meio de cultivo, impossibilitando o crescimento de microalgas que não sobrevivem à temperatura ambiente da região (Algarve) e como não existe forma de avaliar o crescimento de biomassa ao longo do tempo, não foi possível realizar uma experiência na sua totalidade apenas pequenos testes foram feitos no sistema. Ambos os componentes serão implementados futuramente no protótipo do fotobiorreator por parte do laboratório *GreenCoLab*.

Os testes que foram realizados no sistema foram inicialmente testes funcionais para garantir a eficiência, fiabilidade e consistência do sistema. Uma das partes mais importantes para todo o sistema funcionar de forma uniforme é garantir a solidez no fluxo de comunicação entre os diferentes componentes. Para este processo o utilizador através do interface de utilizador seleciona um fotobiorreator para realizar uma experiência, configura os diferentes parâmetros pré-experiência, calibra a sonda de pH e começa uma experiência. Ao começar uma experiência são retornados os valores das leituras efetuadas nos sensores do fotobiorreator, onde são lidos e enviados para o utilizador os valores da temperatura e pH. Estes valores são depois mostrados em tempo real no interface de utilizador sob a forma de gráficos onde será possível fazer uma análise no decorrer total da experiência ou numa porção temporal escolhida e filtrada pelo utilizador, completando assim o fluxo de comunicação entre os diferentes componentes do sistema.

As grandes vantagens da implementação do *software* desenvolvido para este projeto são a possibilidade de aceder à aplicação Web e a qualquer experiência através de um navegador Web, a

responsividade do interface de utilizador que suporta qualquer tamanho de ecrã permitindo assim a utilização de um telemóvel que possua um navegador Web para controlar o sistema e o acesso ao sistema a partir de qualquer dispositivo dentro da mesma rede local (normalmente um laboratório que de produção de microalgas). Uma das melhorias que poderá ser feita no futuro é a implementação do sistema na nuvem permitindo assim não só a centralização do sistema num único lugar como a possibilidade de acesso remoto fora de uma rede local.

Ao realizar uma experiência de controlo do crescimento de microalgas, é gerado um *dataset* que poderá ser analisado tanto na aplicação Web através dos gráficos de temperatura e pH presentes no interface de utilizador, como pela geração de um ficheiro CSV que o utilizador pode descarregar. Uma outra melhoria que poderá ser implementada na aplicação é o fornecimento de dados estatísticos sob a forma de diferentes gráficos, de modo a facilitar uma possível análise manual por parte do utilizador que está a realizar experiências de crescimento de biomassa.

Tendo em conta o tempo mínimo de 12 dias para fazer crescer uma espécie de microalgas, no decorrer de uma experiência desse tipo milhares de dados serão gerados e o sistema poderá ter algumas falhas nas leituras de algum dos parâmetros, por exemplo, ao haver uma falha de luz as leituras podem ficar a 0 por um período determinado de tempo voltando depois ao normal quando a luz regressar. Uma grande melhoria que ficará para o futuro, é a implementação de um mecanismo de deteção de anomalias em tempo real e após término de uma experiência.

Por fim, foi realizada uma experiência com o propósito de testar o controlo do pH no meio de cultivo. As algas consomem carbono dissolvido no meio tornando-o mais básico, foi então implementado e testado o sistema de automação de controlo do pH onde é injetado CO_2 comprimido consoante um valor de *setpoint* de pH definido pelo utilizador. Através da Figura 34 ilustrada abaixo, é possível observar a duração total (12 horas) da experiência teste, onde através da análise da evolução do valor de pH no respectivo gráfico podemos concluir que a automação funcionou como esperado.

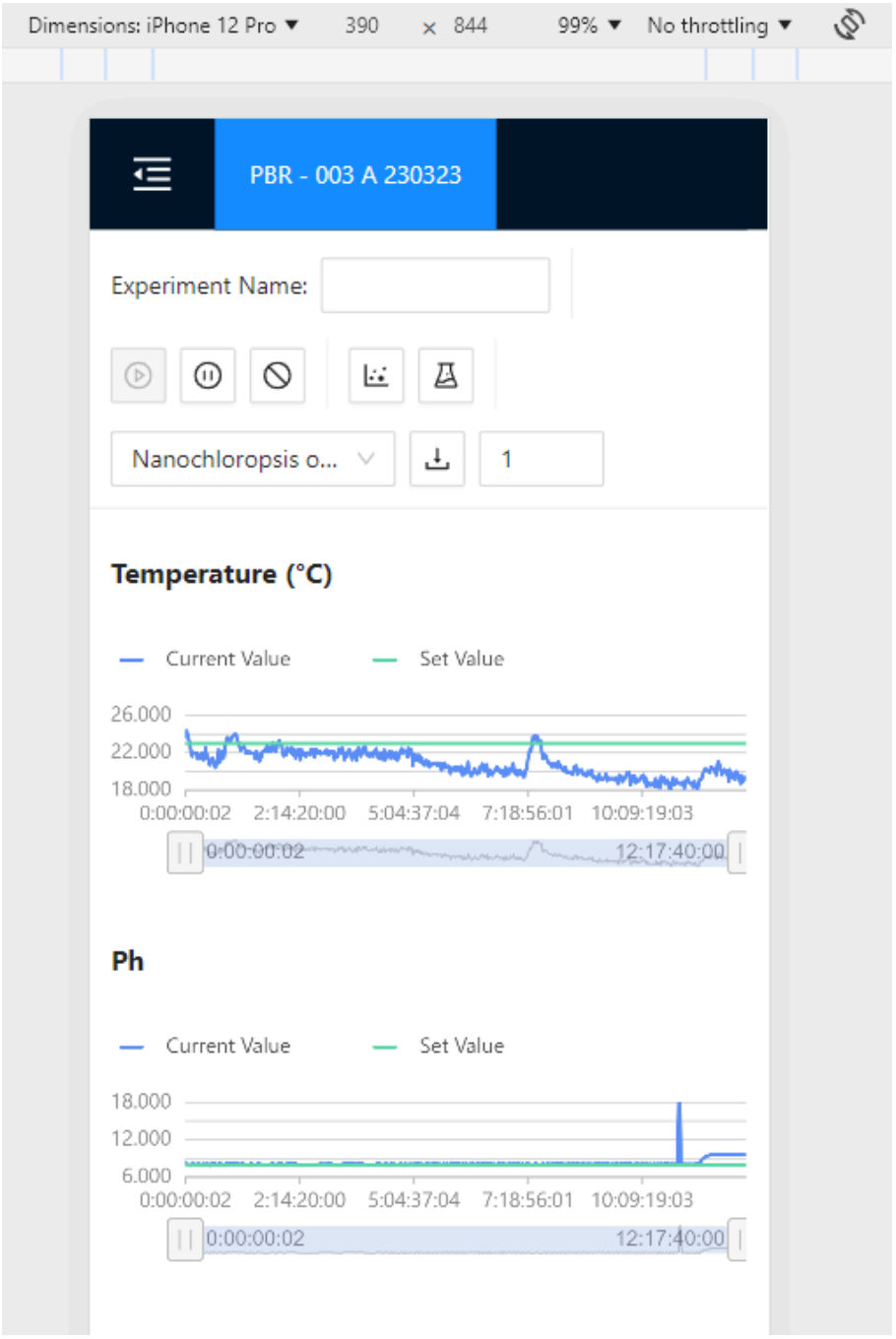


Figura 34 - Teste de controlo do pH (iPhone 12 Pro)

Em conclusão, um dos grandes impedimentos para a realização de experiências de crescimento de microalgas através da utilização do sistema desenvolvido neste projeto, foi a falta dos componentes no protótipo do fotobiorreator para a alteração de temperatura e medição da densidade ótica. Mesmo tendo em conta esse impedimento, foi possível realizar testes de modo a garantir que o fluxo de comunicação entre a interface de utilizador, API, microcontrolador e fotobiorreator está funcional. Algumas das melhorias identificadas anteriormente serão detalhadas no capítulo seguinte de modo a promover uma proposta de implementação e melhoria do sistema para o futuro.

5 Conclusões e Trabalhos Futuros

Uma das grandes vantagens da utilização de fotobiorreatores que permitam controlar condições ambientais de forma simulada é a possibilidade de cultivar espécies de microalgas onde as condições específicas de um país ou local não são favoráveis para o seu crescimento. Para simular tais condições é necessário um *software* que permita o controlo do PBR, onde é possível alterar os valores e os modos de funcionamento dos sensores disponíveis. O interface de utilizador deve ser intuitivo e acessível de modo a facilitar o trabalho de quem irá realizar experiências e controlar o crescimento das microalgas. Para a produção de espécies de microalgas específicas, os parâmetros a ser alterados são as luzes LED, a temperatura, o PH, o valor de CO_2 e o valor de oxigénio no meio de cultivo, as alterações desses mesmos parâmetros podem ser feitos de forma manual e constante, onde os valores escolhidos pelo utilizador não variam com o decorrer de uma experiência ou através de modos de funcionamento pré-programados que farão variar os valores no decorrer da experiência. Estes valores podem ir alterando no decorrer da experiência seguindo, por exemplo, um ciclo diário de um local específico onde os valores dos parâmetros mudam conforme o número de horas de dia e noite.

Neste projeto foram implementadas funcionalidades que permitirão melhorar muito o fluxo de trabalho com o PBR num laboratório na área das microalgas, o fácil acesso aos dados de uma experiência quer pela base de dados quer através do descarregamento de um ficheiro de CSV, a criação de perfis de utilizador onde os parâmetros configuráveis são reutilizáveis em qualquer nova experiência e a possibilidade de aceder à interface de utilizador através de vários dispositivos que tenham um navegador Web, tais como computadores, tablets ou telemóveis. O fluxo de trabalho para a realização de uma experiência de controlo de crescimento de microalgas anteriormente referido foi testado utilizando o *software* implementado neste projeto onde seguindo a mesma ordem de descrição foram testados os processos de configuração parâmetros do fotobiorreator através da interface do utilizador, configuração da experiência para o controlo do crescimento de uma espécie de microalga específica com uma duração total de 12 dias e uma escala de gravação de

1 minuto referenciada no capítulo dos resultados experimentais, pausa da experiência para troca de compostos no meio de cultura e retoma da mesma, término da experiência e análise dos resultados através do interface do utilizador (gráficos de temperatura e pH) e ficheiro de CSV descarregado que contém os dados da experiência com a mesma cadência de gravação por defeito (1 minuto) ou com uma escala à escolha.

Todos os passos deste fluxo podem ser realizados em qualquer dispositivo presentes na mesma rede do laboratório que tenham acesso a um navegador Web, sendo esta uma das maiores vantagens face ao *software Algenious* analisado anteriormente, ficou ainda por implementar uma funcionalidade de acesso remoto ao sistema, onde o objetivo principal seria aceder ao decorrer da experiência e fazer mudanças de parâmetros caso necessário remotamente.

Como funcionalidade futura, existe a possibilidade de implementação do sistema (aplicação Web) na nuvem, onde o acesso é permitido dentro e fora de um laboratório mas com algumas condicionantes a nível de infraestrutura do sistema. Com o design atual do protótipo do fotobiorreator existe a necessidade de uma ligação USB a uma porta série de um computador ou outro dispositivo, que tenha disponível o mesmo tipo de ligação, de modo a estabelecer a comunicação com sistemas externos como é o caso da aplicação Web desenvolvida. Em alternativa a um computador a estabelecer este tipo de ligação poderia ser utilizado um *Raspberry Pi*, onde a sua função principal seria estabelecer a ligação com o sistema implementado na nuvem e o redirecionamento de comandos transferidos entre sistemas (interface de utilizador e PBR). Isto significa que os comandos ou pedidos enviados através do interface de utilizador seriam feitos à API na nuvem onde estaria hospedado o sistema, sistema esse com uma ligação ao *Raspberry Pi* que por sua vez estaria ligado diretamente ao PBR completando assim o do fluxo de comunicação.

O *Azure IoT Hub* é uma das possíveis soluções para a implementação prática para este tipo de sistema específico na nuvem[27], a preparação e configuração do *Raspberry Pi* é necessária para disponibilizar a comunicação com o fotobiorreator via porta série, são necessários seguir alguns passos para configurar o sistema:

1. Criação de um *IoT Hub* no *Azure* e o registo do *Raspberry Pi* como dispositivo através de uma string de conexão;
2. Instalação do *SDK* do *Azure IoT* no *Raspberry Pi* para possibilitar o envio e recepção de dados no *IoT Hub*;
3. Redirecionamento de dados do *Raspberry Pi* para a API (hospedada na nuvem *Azure*) no *IoT Hub* através da utilização de uma *Azure Function*;

4. Comunicação bidirecional através da recepção de mensagens provindas da API para o *Raspberry Pi* e por sua vez para o PBR.

Em resumo, o fluxo de comunicação bidirecional ilustrado na Figura 35, parte do interface de utilizador onde são feitos pedidos à API que são transmitidos ao *Raspberry Pi* via *Azure IoT Hub* e por sua o *Raspberry Pi* lê dados do fotobiorreator via porta série e envia-os para o *Azure IoT Hub* que os direciona para a interface de utilizador via API.

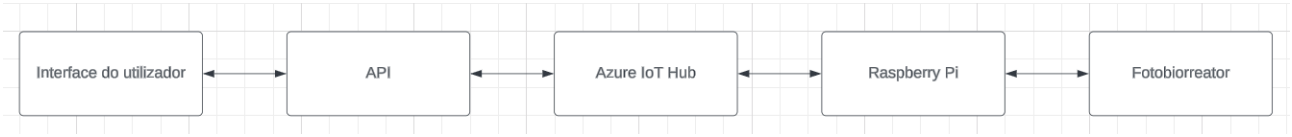


Figura 35 - Fluxo de comunicação bidirecional

Dado que no *software* implementado não existe maneira de detetar falhas no sistema sem analisar os resultados obtidos dos dados de cada experiência de controlo do crescimento de microalgas, outra das funcionalidades futuras poderá ser a implementação de um mecanismo de deteção de anomalias em tempo real. Na implementação atual da aplicação Web, a API corre um processo à parte de forma contínua ao iniciar uma experiência, este processo tem como objetivo fazer a leitura dos valores dos sensores do fotobiorreator com uma cadência de 4 segundos onde os dados da leitura são gravados numa base de dados e enviados para o interface de utilizador quando requisitados para preencher os gráficos de temperatura e pH.

Tendo uma precisão de 4 segundos entre leituras, existe a possibilidade de implementar um sistema de deteção de anomalias simples no decorrer do processo referido anteriormente onde são definidos os limites aceitáveis para a temperatura (entre 15º a 30º) e pH (entre 6.5 a 8.5) e validados os valores lidos de modo a determinar se estão dentro dos limites aceitáveis, caso não estejam pode ser disparada uma ação como enviar uma notificação ou gravar um registo na base de dados.

Para uma solução mais complexa mas alinhada com a proposta da implementação do sistema na nuvem referido anteriormente, é possível implementar um mecanismo de deteção de anomalias através da utilização do *Azure IoT Hub*, mais especificamente utilizando o serviço *Azure Anomaly Detector*. Este serviço baseado em inteligência artificial da *Microsoft Azure*, utiliza técnicas de *Machine Learning* para detetar automaticamente anomalias em séries temporais, é projetado para identificar padrões inesperados em dados ao longo do tempo, como variações fora do comum que podem indicar problemas, falhas ou eventos relevantes em sistemas monitorizados[28].

Existe ainda outra possível solução, a de treinar um modelo de *Machine Learning* de modo a detetar padrões anômalos com base em dados históricos. Para obter esses dados históricos seria necessário a realização de várias experiências com o protótipo do fotobiorreator onde fossem categorizados por espécie de microalgas os limites de temperatura e pH.

Os modelos mais comuns para este tipo de objetivo são:

1. *Isolation Forest* - Onde é feita uma detecção de pontos isolados fora dos dados normais;
2. *Autoencoders* - Que são redes neuronais que tentam reproduzir os dados de entrada onde erros de reconstrução altos poderão indicar anomalias.

Tendo em conta que a API está desenvolvida em Python, a opção que melhor se adequa é a utilização da *framework scikit-learn* para a utilização de um modelo *Isolation Forest* onde ilustrado na Figura 36 abaixo está um exemplo de uma possível implementação.

```python
from sklearn.ensemble import IsolationForest
import numpy as np

# Dados históricos de temperatura e pH
X = np.array([[22, 7.0], [23, 7.2], [25, 6.9], [24, 7.1], [26, 6.8], [23, 7.0]])

# Treina o modelo
clf = IsolationForest(contamination=0.1)
clf.fit(X)

# Nova leitura de temperatura e pH
new_reading = [[27, 8.5]]
if clf.predict(new_reading)[0] == -1:
    print("Anomalia detectada!")
```

Figura 36 - Exemplo de implementação de um mecanismo de deteção de anomalias

Dado que existiram alguns impedimentos para o desenvolvimento de certas funcionalidades como as referidas anteriormente neste capítulo, de forma geral os objetivos do projeto foram cumpridos e foram implementadas as funcionalidades essenciais no *software* proposto para a realização de experiências de controlo do crescimento de microalgas no protótipo do fotobiorreator desenvolvido pelo laboratório *GreenCoLab*.

Referências

[1] Pires, J. C. M., Avon Ferraz, M. C. M., & Martins, F. G. (2017). Photobioreactor design for microalgae production through computational fluid dynamics: A review. Disponível em https://hdl.handle.net/10216/104789. Acesso em 17 de março de 2024.

[2] Richardson, L., Amundsen, M., & Ruby, S. (2013). *RESTful Web APIs*. Sebastopol, CA: O'Reilly Media.

[3] CCMAR, & GreenCoLab. (2018). GreenCoLab. Disponível em https://ccmar.ualg.pt/en/page/greencolab. Acesso em 21 de março de 2024.

[4] Pulz, O., & Scheibenbogen, K. (1998). Photobioreactors: Design and performance with respect to light energy input. *Advances in Biochemical Engineering/Biotechnology, 59*, 123–152. https://doi.org/10.1007/3-540-49793-1_4

[5] Tredici, M. R. (2010). Photobioreactors. In *Algae Biotechnology* (pp. 289–311). Berlin, Alemanha: Springer.

[6] Richmond, A. (2004). *Handbook of microalgal culture: Biotechnology and applied phycology*. Oxford, Reino Unido: Wiley-Blackwell.

[7] Chisti, Y. (2007). Biodiesel from microalgae. *Biotechnology Advances, 25*(3), 294–306. https://doi.org/10.1016/j.biotechadv.2007.02.001

[8] McCrady, S. G. (2013). *Designing SCADA application software: A practical approach*. Oxford, Reino Unido: Butterworth-Heinemann.

[9] Maayan, G. (2021). The IoT rundown for 2021: Stats, risks, and solutions. *Security Today*. Disponível em https://www.securitytoday.com.

[10] Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for smart cities. *IEEE Internet of Things Journal, 1*(1), 22–32. https://doi.org/10.1109/JIOT.2014.2306328

[11] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, 61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

[12] Chisti, Y. (2007). Biodiesel from microalgae. *Biotechnology Advances, 25*(3), 294–306. https://doi.org/10.1016/j.biotechadv.2007.02.001

[13] Phytoalgae. (n.d.). Microalgas. Disponível em https://phytoalgae.pt/microalgas/. Acesso em 2 de abril de 2024.

[14] Postal. (2023, 21 de julho). Projeto liderado pela empresa algarvia Necton combate escassez e contaminação da água. Disponível em https://postal.pt/sociedade/projeto-liderado-pela-empresa-algarvia-necton-combate-escassez-e-contaminacao-da-agua/. Acesso em 25 de março de 2024.

[15] Algenuity. (n.d.). Algem brochure. Disponível em https://irp-cdn.multiscreensite.com/3aa121e5/files/uploaded/286-19%20Algem%20Brochure_190819.pdf. Acesso em 2 de abril de 2024.

[16] InfoEscola. (n.d.). Produção de biomassa por microalgas: Fotobiorreatores x sistema de lagoas raceway. Disponível em https://www.infoescola.com/biologia/producao-de-biomassa-por-microalgas-fotobiorreatores-x-sistema-de-lagoas-raceway/. Acesso em 2 de abril de 2024.

[17] GreenCoLab. (2023). Operation via RS232 Interface [Documento interno não publicado].

[18] Modan, S. (2023, outubro). The benefits of ReactJS and reasons to choose it for your project. Disponível em https://www.peerbits.com/blog/reasons-to-choose-reactjs-for-your-web-development-project.html. Acesso em 3 de abril de 2024.

[19] Derry, M. (2023, 30 de agosto). What is Flask and how do developers use it? A quick guide. Disponível em https://careerfoundry.com/en/blog/web-development/what-is-flask/. Acesso em 3 de abril de 2024.

[20] Xu, L., & Li, Y. (2020). Data management in bioprocessing and prototyping. *Journal of Bioprocessing Science and Engineering*, 12(4), 45–58.

[21] Silva, M. A., & Oliveira, L. F. (2021). Comparative analysis of modern JavaScript frameworks: React, Vue, and Angular. *Journal of Web Development Technologies*, 8(2), 15–30.

[22] Rossum, G., & Pereira, J. (2020). Comparative study of web frameworks for rapid application development. *Journal of Software Engineering*, 34(3), 78–92.

[23] Jones, N. (2021). Choosing the right database for your Python web app. *Database Insights Weekly*.

[24] Heemskerk, B. (2019). *Pro TypeScript: Application-scale JavaScript development*. Berkeley, CA: Apress.

[25] Abramov, D., & Clark, A. (2016). *Redux: A predictable state container for JavaScript apps*. Disponível em https://redux.js.org/.

[26] Ant Design Charts. (n.d.). Basic line plot. Disponível em https://ant-design-charts.antgroup.com/en/examples/statistics/line/#basic. Acesso em 17 de setembro de 2024.

[27] Kobeissi, A., Berta, R., Bellotti, F., & De Gloria, A. (2020). IoT ubiquitous edge engine implementation on the Raspberry Pi. *Journal of IoT Applications*, 5(1), 12–19.

[28] Microsoft. (n.d.). *Anomaly Detector documentation*. Disponível em https://learn.microsoft.com/en-us/azure/cognitive-services/anomaly-detector. Acesso em 14 de setembro de 2024.