Projeto Final

Sistemas de Computação Móvel e Ubíqua 2021/22

Diogo Ramos 56174, Diogo Rodrigues 56153, José Murta 55226

Smart Gym

1 Introdução e objetivos

O objetivo principal deste projeto foi criar um sistema de computação móvel e ubíquo que tente melhorar a qualidade de vida dos seus utilizadores. Neste sistema, optámos por uma vertente de "Smart Gym"de forma a melhorar a experiência dos seus clientes.

Enquanto utilizadores frequentes de ginásios, surgiu a ideia de tentar tornar o ginásio algo mais inteligente facilitando a utilização do mesmo pelos clientes, através de implementações úteis.

O presente documento tem como objetivo principal apresentar uma descrição detalhada da arquitetura do sistema e sua implementação, bem como documentar e fundamentar a evolução do desenvolvimento deste sistema.

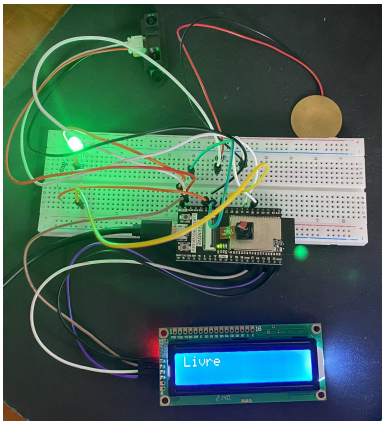
2 Arquitetura

Como indicado no documentos previamente redigidos do projeto, o sistema desenvolvido consiste numa aplicação móvel que comunica com aplicação ubíqua constituída pelo microcontrolador e devidos sensores e atuadores. A comunicação entre os dois subsistemas é realizada com auxílio a um serviço de Cloud.

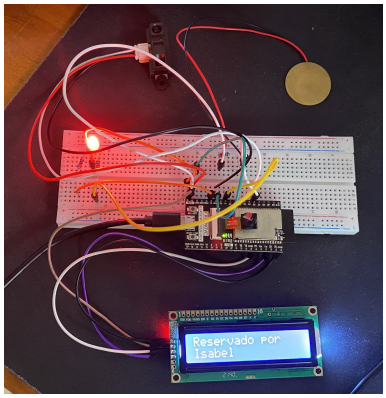
Na Fig.1 pode-se observar o setup dos sensores e atuadores com o microcontrolador e respetivamente o comportamento quando a máquina está livre e reservada pela sócia do ginásio "Isabel".

Aplicação móvel

A aplicação móvel é destinada a dispositivos Android e por isso foi desenvolvida em Dart, e para tal optámos por utilizar a framework Flutter que nos



(a) Setup em modo máquina livre



(b) Setup em modo máquina reservada

Figura 1: Setup dos sensores, atuadores e microcontrolador

irá facilitar no processo de desenvolver a mesma. Esta aplicação tem como principais funcionalidades:

- Auxílio no controlo de entradas e saídas do ginásio
- Consulta do nivel de lotação do ginasio
- Reserva e consulta do estado de equipamentos
- Comportamento conforme o contexto

Microcontrolador

Como microcontrolador para este projeto, selecionámos o ESP32 visto que contém um recetor de WiFi integrado no mesmo e disponibiliza interfaces para comunicar desta forma.

Sensores

Para as funcionalidades descritas são necessários dois tipos de sensores:

- Infrared Sensor Sharp - este é um sensor de Infra-vermelhos que mede a distância e tem um range de 10cm até 80cm. As medições deste sensor servem para registar o número de pessoas que saíram do ginásio, e com

essa informação, calcular a lotação do mesmo. Para a implementação ideal do nosso projeto num ginásio real, provavelmente seria necessário um sensor que medisse distância num maior range, mas tendo em conta os sensores disponíveis, esta opção foi a mais adequada.

- Force Sensitive Resistor - este é constituído por uma resistência que varia a sua medição consoante a pressão aplicada na área de deteção. Através deste sensor será possível aferir se um equipamento está a ser ocupado.

Atuadores

Os atuadores do sistema serão

- RGB Led - este atuador tem como objetivo indicar visualmente a disponibilidade de um equipamento. Apresentará a cor verde se a máquina estiver livre ou a cor vermelha caso a mesma esteja ocupada ou reservada.

- Display LCD - este atuador irá indicar visualmente o nome do utilizador que reservou um determinado equipamento ou indica se está a ser utilizado ou livre.

3 Implementação do programa para Microcontrolador

Para desenvolver o código que vai ser executado pelo ESP32 utilizamos a Arduino IDE, onde foi necessário instalar novas bibliotecas no gestor de placas para que a IDE e o microcontrolador fossem compatíveis. Para respeitar a arquitetura do projeto, onde os dados registados pelos sensores precisam de ser observados pela aplicação e vice versa (a aplicação regista as ações do utilizador e os atuadores respondem às mesmas, explicação detalhada nas secções seguintes) utilizamos uma *FireBase RealTime Database*. O código é composto por dois métodos principais: setup e loop.

Setup method

Durante o método setup, executado apenas uma vez, é realizado a inicialização das variáveis usadas durante o código, explicadas detalhadamente nos próximos parágrafos, o setup do LCD e Firebase e a caracterização dos pins do LED como output. Para bom funcionamento do LCD utilizamos a interface *LiquidCrystal_I2C.h* onde inicialmente definimos o endereço I2C, número de linhas e colunas (16 e 2, respetivamente, no nosso caso) e utilizamos as funções *init()* (como o nome indica, inicia o LCD display) e *backlight()* (acende a luz azul de fundo do atuador).

Para contacto com o Firebase utilizamos uma biblioteca disponibilizada pelo professor durante as aulas práticas e disponível no Github por *ohad32* com o nome *FireBase32*, e incluída no código através interface *FireBase32.h*. O objetivo principal com o uso desta biblioteca é simplificar o uso da original mascarando dificuldades, por exemplo diminuindo o numero de argumentos por função. Tivemos também de ultrapassar dois erros durante o desenvolvimento, o primeiro era um erro de compilação do ficheiro interface onde faltava os tipos de return e os métodos para escrever e ler strings não funcionavam pelo que resolvemos simplesmente adaptando os nomes no ficheiro .cpp e interface da biblioteca. Na fase inicial apenas é preciso definir a bd, utilizando o url da real time database disponivel na consola web do FireBase e executar o metodo *wifi()* (neste método é realizada a ligação à rede wifi necessária para a comunicação com a rede, os argumentos são o SSID e a password).

Loop method

Durante o método loop, executado em ciclo durante a execução do programa, é definido o comportamento desejável que os sensores e atuadores tenham durante o uso do sistema ubíquo. No inicio é lida a distância sentida pelo sensor de distância e analisada se se encontra acima de um determinado threshold (devidamente testado para detetar a presença de um corpo a mover-se). Se isso acontecer é porque algum sócio está a sair do ginásio por isso através do firebase lemos a lotação atual, subtraímos uma unidade, e voltamos a escrever para a firebase atualizando a lotação atual do ginásio (isto é feito utilizando os métodos *GetData e WriteData* passando o path e a variável a escrever ou para onde o valor é lido). Devido às leituras do sensor serem muito rápidas a passagem de um corpo iria retirar mais que uma pessoa da lotação, por

isso utilizando uma variável counter em que a condição apenas executa se for menor que 0, induzimos um pequeno delay (a variável é iniciada a 10 por fazendo com que a condição apenas execute a cada 10 iterações).

Caso exista uma reserva (esta condição é confirmada através do contato com a base de dados e verificar se a variável nomeReserva se encontra com um nome diferente de null, assumindo à priori que nenhum utilizador tem o nome null) o LED RGB deve-se encontrar vermelho e no LCD ser mostrado o nome do sócio que tem a reserva realizada. Para isso o LED é acionado com as devidas intensidades nos pinos certos e no LCD é definido a zona do cursor e executado o print. Para bom funcionamento da aplicação é enviado um write para a firebase para mudar o estado da máquina reservada. Para finalizar de forma a que a reserva não seja eterna, cada uma tem um timeout à volta de 20 segundos em que a reserva do utilizador realizada na app desaparece e fica disponível para ser renovada (este timeout numa situação real teria de ser um pouco maior mas para questões de teste e demonstração foi fixado à volta de poucos segundos).

Caso não exista nenhuma reserva efetuada no momento então é medida a pressão efetuada no Force Sensitive Resistor e guardada numa variável detalhada para o efeito. Se esta medição for superior a um determinado threshold (devidamente calibrado para obtenção de resultados desejados) então significa que o equipamento está ocupado. Nesta situação, para os clientes no ginásio saberem o estado de ocupação da máquina, a cor do LED RGB deve ser alterada para vermelho e o LCD deve apresentar o texto "Busy". De forma a a que os utilizadores consigam também obter esta informação através da aplicação mobile é necessário enviá-la para a base de dados Firebase (isto é feito utilizando o método *WriteData* e passando o path desejado e a variável a escrever). Se a medição realizada pelo Force Sensistive Resistor for inferior ao threshold mencionado então o LED RBG deve apresentar a cor verde, o LCD deve indicar o texto "Livre"e da mesma forma também é necessário disponibilizar esta informação para os utilizadores da aplicação (realizado da forma mencionada anteriormente).

4 Implementação da Aplicação Android

A aplicação Android desenvolvida está dividida em 4 screens diferentes sendo que o principal é a página inicial, HomePage. Esta é a página que permite aceder às restantes.

Para utilizar os serviços de FireBase na aplicação é necessário importar a package disponibilizada pelo FireBase de desenvolvimento Android, bem como registar a aplicação na plataforma para interligação com a base de dados. Para além disso sempre que se pretende fazer alterações à base de dados é necessário criar um referência para a mesma, diferindo apenas no path onde se pretende ler ou escrever dados.

HomePage

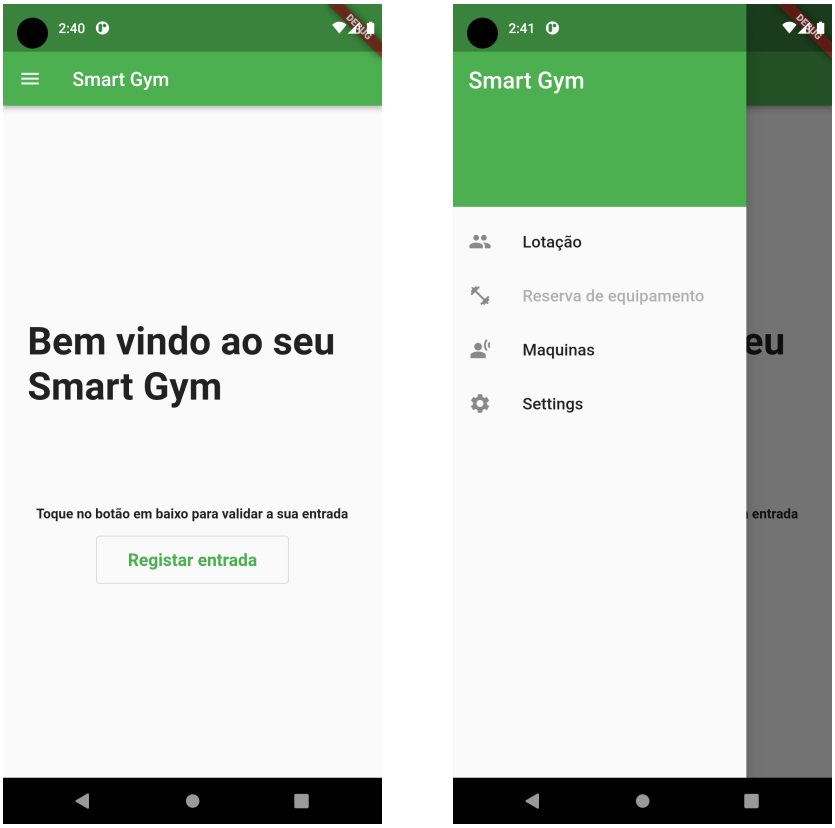
A HomePage é constituída por blocos de texto simples e um botão com a funcionalidade de registar a entrada do utilizador. É importante mencionar que optámos por não implementar nenhum tipo de autenticação nem funcionalidades e características específicas de utilizadores visto que para as funções traçadas para o sistema e para efeitos de demonstração, tendo em conta a quantidade de recursos como atuadores e sensores, não haveria necessidade de tal. No entanto, temos a clara noção que num sistema que fosse utilizado num ginásio real, teríamos de desenvolver detalhadamente esta funcionalidade com mecanismos de autenticação e consequentemente mecanismos de segurança adequados. Por esta razão o botão de registo de entrada apenas incrementa o número de clientes no ginásio, incrementando o valor indicado para este fim armazenado na base de dados, mas não regista propriamente uma entrada de um utilizador com mecanismos de autenticação.

Nesta tab existe também um botão no canto superior esquerdo para aceder a um menu lateral com ligações para acesso às restantes tabs. A página de reserva de equipamento apenas fica disponível para acesso caso o utilizador já tenha registado a sua entrada no ginásio, pois não faria sentido um cliente poder reservar uma máquina sem estar no ginásio. As restantes páginas podem ser acedidas normalmente.

Na Fig.2, é possível visualizar screenshots da HomePage e do menu lateral da mesma.

Página de visualização da lotação em tempo real do ginásio

Esta página é constituída apenas por um bloco de texto simples e um widget que disponibiliza ao utilizador a percentagem de lotação do ginásio em tempo



(a) HomePage (b) Menu lateral

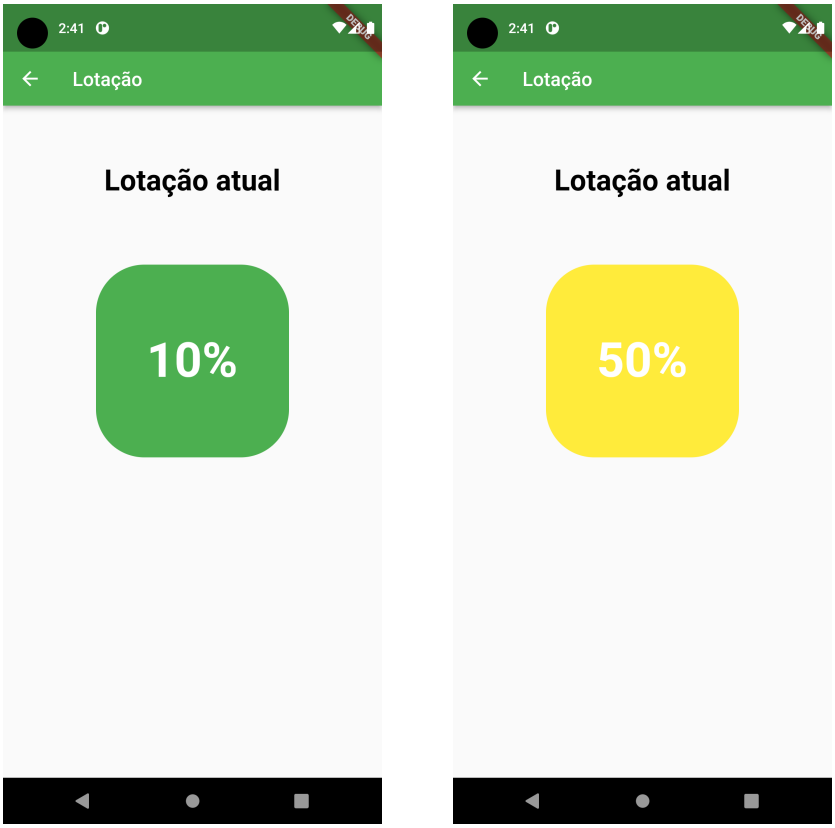
Figura 2: Página Principal da aplicação

real. Esta funcionalidade é implementada através de uma função Listener de um evento de alteração do valor desejado na base de dados Firebase. Esta função monitoriza o valor desejado e quando existem alterações ao mesmo, muda o estado das variáveis da aplicação. Caso o valor de ocupação seja alterado, quer seja por entrada ou saída de um cliente, então o valor disponibilizado também é alterado instantaneamente.

Consoante a percentagem de ocupação do ginásio a cor do widget é modificada: caso o ginásio estiver pouco lotado, até 50% de lotação será apresentada a cor verde; caso esteja, pelo menos metade do ginásio ocupado então a cor disponibilizada é o verde; e caso esteja muito ocupado (mais de 80% de ocupação) o widget apresenta a cor vermelha.

Na Fig.3, é possível visualizar screenshots desta página com diferentes

percentagens de lotação.



(a) Lotação baixa (b) Lotação média

Figura 3: Página de visualização da ocupação do ginásio

Página de visualização da ocupação de cada equipamento

Neste screen é possível visualizar o estado de ocupação de cada equipamento do ginásio em tempo real. Devido ao número reduzido de sensores e atuadores apenas é possível simular o comportamento de um equipamento, e para este efeito escolhemos o primeiro equipamento da lista, a "Passadeira". As

restantes máquinas não possuem comportamento, foram apresentadas apenas para contexto.

Para implementar a funcionalidade desejada, foi utilizada uma função Listener de um evento de alteração do valor desejado na base de dados, realizada de forma similar à apresentada anteriormente. Assim caso o equipamento esteja ocupado ou tenha sido reservado, é apresentada a cor vermelha na máquina em questão e caso esteja livre é apresentada a cor verde.

Na Fig.4 pode-se visualizar um screenshot desta página.

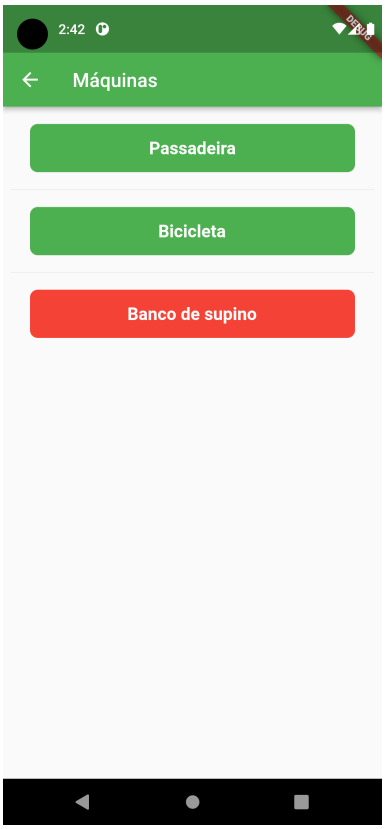


Figura 4: Página

Página de reserva de equipamento

Neste screen é possível o utilizador realizar uma reserva de uma máquina. Esta ficará reservada durante cerca de 20 segundos (este valor é tão baixo

apenas para efeitos de demonstração). Devido aos motivos mencionados anteriormente, nesta página também só é possível reservar uma máquina.

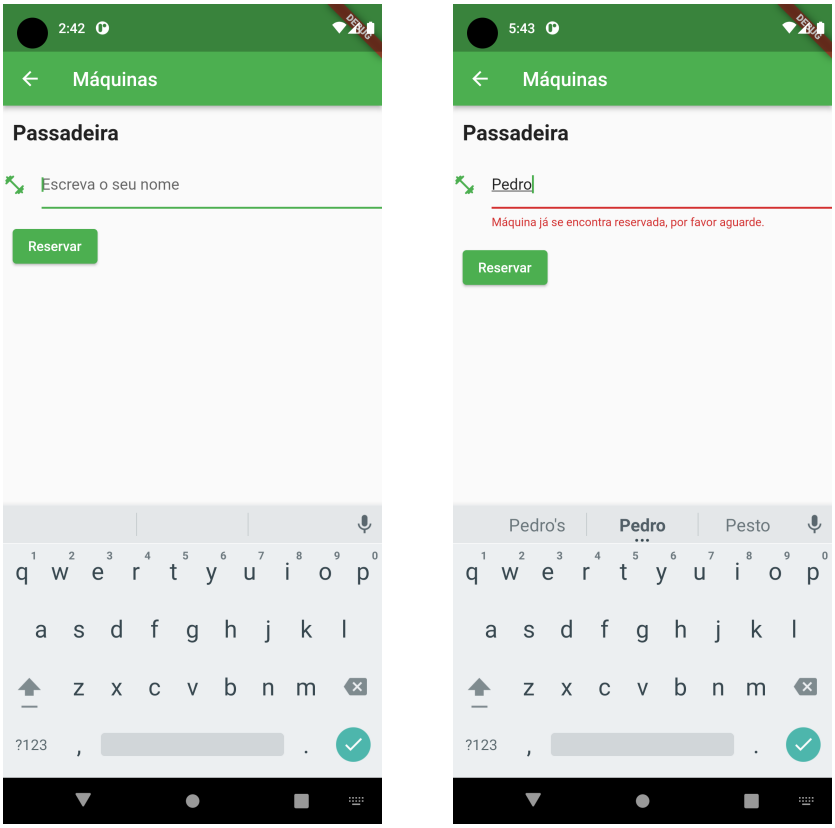
Para implementar esta funcionalidade, foi utilizado um formulário constituído por um campo de texto, onde um utilizador digita o seu nome (que estará disponível no LCD), e um botão para efetivamente reservar o equipamento. As condições de validação deste formulário são apenas duas: o utilizador tem de obrigatoriamente digitar um nome (não pode deixar o campo de texto vazio) e a máquina não se pode encontrar reservada no momento (esta informação é obtida com uma função Listener). O formulário precisa ainda de fazer uso de um TextEditingController para conseguir guardar, numa variável do programa, o nome inserido pelo utilizador. O nome do cliente que reserva a máquina é escrito no Firebase através do método Update disponibilizado pela biblioteca de Firebase para Android.

Na Fig. 5 é possível visualizar screenshots deste screen. A figura (a) demonstra o formulário vazio e a (b) o comportamento e respetiva mensagem de erro caso a máquina já se encontre reservada.

Firebase

O Sistema de armazenamento em Cloud FireBase é usado para comunicação e armazenamento de todos os dados da aplicação Android e do uso do Arduino. A base de dados utilizada foi a RealTime Database pois é a mais eficaz e simplifica o processo de interligação entre a aplicação e a base de dados, tendo em conta as necessidades e funcionalidades do projeto. Os dados são guardados como JSON e sincronizados em tempo real com cada cliente que esteja conectado. Os dados armazenados são a lotação e as máquinas de forma a ser usados pela aplicação para poder interpretar de forma eficiente o funcionamento do ginásio.

- Lotação - A lotação é composta por uma constante que é chamada Max, que tem a lotação máxima do ginásio, e a variável atual que indica a lotação atual do ginásio. Para controlar as entradas, é preciso validar a entrada para a variável poder ser incrementada e para registar a saída é utilizado o Infrared Sensor Sharp para poder detetar a saída dos utilizadores, decrementando o valor da lotação.



(a) Formulário vazio (b) Formulário sem sucesso

Figura 5: Página de reserva de equipamentos

- Máquinas - Na base de dados, temos apenas uma máquina uma vez que apenas possuímos um Force Resistive sensor para puder detetar que a máquina se encontrava a ser utilizada. Na máquina temos se está ocupada, o que vai levar a que esteja a 1, ativando a luz Led e apresentando uma mensagem que indica que está ocupada. Na parte da reserva estará presente então o nome do utilizador que reservou o equipamento não sendo possível que outro utilizador reserve o mesmo. Para utilizar mais máquinas, seria necessário criar novos valores para pudermos os dados de várias máquinas de forma eficiente.

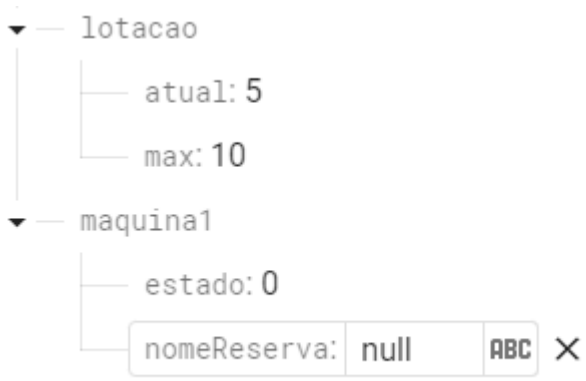


Figura 6: Estrutura da Base de dados