Escola Secundária Rainha Dona Leonor | AERDL

PROVA DE APTIDÃO PROFISSIONAL – PAP

CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

RELATÓRIO

JOÃO BEIRÃO

TURMA 12º 11ª – ALUNO 14

MARÇO.2021

AGRADECIMENTOS
E DEDICATÓRIA

Durante a criação e desenvolvimento do meu projeto fui sempre apoiado por um conjunto de pessoas que tiveram um papel direto e indireto, muito importante.

Em primeiro lugar gostaria de agradecer aos meus amigos e colegas mais chegados. Foram eles que me apoiaram diariamente na construção do meu projeto dando-me apoio moral e dicas de alguns detalhes que poderia incluir no meu projeto. Tiveram desde o início do curso um papel muito importante na minha vida, tanto a nível profissional como pessoal, ajudando-me a crescer como indivíduo e dando-me um motivo mais do que suficiente para dar o meu melhor em todos os trabalhos que realizei.

Gostaria de agradecer também a todos os Professores que durante os 3 anos do Curso, dedicaram o seu esforço e empenho em todos os momentos. Um agradecimento particular ao núcleo de professores de informática, pois apoiaram-me desde o primeiro momento na realização do meu projeto. Mostraram-se sempre bastante acessíveis e disponíveis para qualquer eventualidade, tendo mesmo cedido algumas aulas para a realização dos projetos. Gostaria de agradecer em particular à Professora Paula Pinheiro, não só pelo seu profissionalismo, mas pela disponibilidade demonstrada, sobretudo em momentos críticos do meu projeto, e me ajudou a manter uma atitude positiva face ao enorme desafio enfrentado.

Agradeço também ao grupo de auxiliares da escola Rainha Dona Leonor, por me terem tratado com enorme respeito e amizade durante a realização de todo o curso.

Por fim, mas não menos importante, agradeço à minha família pelo enorme apoio que me têm dado em todo o meu percurso escolar até agora, e neste ambicioso projeto.

Gostaria ainda de destacar o papel do meu estimado pai, que desde o início do meu percurso escolar sempre me mostrou qual o melhor caminho a seguir para se alcançar a excelência. Desde cedo, sempre teve uma plena noção das minhas capacidades e dos meus pontos fracos, tendo-me sempre incutido uma mentalidade vencedora, que dura até hoje e me motiva a dar o meu melhor em tudo o que faço.

Para além da enorme ajuda que me deu neste projeto, esteve ao meu lado nos piores momentos vivenciados, acentuando ainda mais a profunda admiração e estima que lhe tenho como pai e ser humano.

Quero assim dedicar-lhe este trabalho, como prova do meu reconhecimento e admiração, para com os valores que me transmite diariamente.

INTRODUÇÃO

Este relatório tem o propósito de apresentar o projeto final do Curso Profissional de Técnico de Informática – Sistemas.

O projeto tem como objetivo demonstrar todos os conhecimentos obtidos durante os 3 anos de formação teórica e prática do respetivo Curso.

Escolhi o projeto "BGREEN", que é constituído por uma estufa de criação de plantas, controlada por telemóvel e automatizada.

A escolha recaiu neste tema, porque consegue criar a ligação entre o meio digital e as preocupações ambientais atuais.

Atualmente, é frequente a preocupação com as alterações climáticas, ao excesso de energias não renováveis consumidas pelo homem e às consequentes questões ambientais que, felizmente, se têm vindo a evidenciar.

A mudança de mentalidade, e a consequente prática de pequenas ações por parte da população mundial têm se revelado bastante positivas, porém, é necessário que o Homem adote ideias mais inovadoras e globais, pois caso contrário, este enfrentará grandes alterações a nível marítimo, atmosférico, e até mesmo de extinção.

Um dos problemas, se não o maior, que se opõe a esta mudança ideológica global é o facto de existirem pouquíssimos indivíduos que possuam a capacidade de manipular tão agigantadas massas populacionais.

Eu, como singelo jovem estudante que sou, tenho um impacto relativamente pequeno na sociedade face a estes gigantes das economias mundiais. Mesmo assim, penso que devo alertar para estas questões e colaborar em tudo o que estiver ao meu alcance para as mudanças necessárias.

Como tal, decidi dedicar o meu projeto a este tema, sendo este uma tentativa de resolver algumas destas questões no meu quotidiano.

Tudo começou com um pequeno equipamento que encontrei numa loja de produtos eletrónicos, que captou a minha atenção nesse mesmo instante. De repente deparei-me com uma criação de plantas dentro de um conceito todo ele eletrónico, e fez todo o sentido.

Se podemos obter melhores resultados e eficiência ambiental através das novas tecnologias, era esse o meu caminho.
Seguiram-se algumas pesquisas do que poderia eu fazer dentro daquele modelo, não só aplicando alguns dos conhecimentos adquiridos, como também tentar abordar outros conceitos, aprendendo outras matérias, sair um pouco da zona de conforto e confesso que foram muitos os desafios que encontrei, consolidando tudo num único projeto.
Existiu também a preocupação de que se tratasse de algo que qualquer um pudesse utilizar no seu dia-a-dia e até compatível com a vertente comercial de produto.

Assim surge a "BGREEN", uma estufa que idealizei de raiz, desde a conceção da estrutura física, aos componentes eletrónicos, controlo, programação e automatização.

ÍNDICE

ESTUFA

COMO TUDO COMEÇOU

Toda a minha jornada relacionada com este projeto, começou com o vislumbre de um produto que me agradou bastante.

Numa tarde em que eu precisei de me dirigir a uma loja de dispositivos eletrónicos para comprar algo que necessitava, acabei por ir com o meu pai a uma loja "Worten", situada no centro comercial Vasco da Gama.

Enquanto me dirigia à secção que dizia respeito ao produto que eu procurava, deparei-me com um produto em destaque que me chamou particularmente à atenção. Este produto consistia numa espécie de vaso inteligente que geria autonomamente algumas plantas contidas no mesmo. Esta ideia ficou-me presa na cabeça, uma vez que nunca me tinha deparado com uma fusão tão crua entre tecnologia e natureza.

O produto, a meu ver, era uma ideia genial, uma vez que conseguia relacionar o problema que são os cuidados constantes para cuidar de uma planta, com a capacidade atual de robotizar e automatizar esses processos. Deste modo, o utilizador deste produto irá sempre, ao contrário do que diz o bom ditado, "colher os frutos do que não plantou".

Dado o meu conhecimento já elevado no que diz respeito à construção e programação de circuitos eletrónicos fui capaz de avaliar rapidamente a complexidade do mesmo. Pude então compreender que também eu, com bastante esforço e trabalho, podia criar um projeto semelhante, porém com algumas funcionalidades adicionais que não estavam incluídas no produto. Sendo que o produto custava várias centenas de euros, e eu não necessitava de ganhar qualquer lucro, poderia então investir uma pequena fração dessa quantia na produção de um produto que poderia até ser superior.

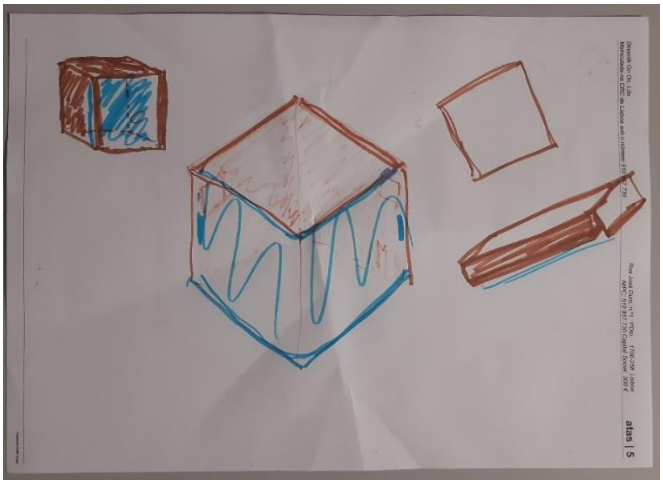
Começaram então a surgir algumas ideias relativas à forma que o meu projeto iria tomar.

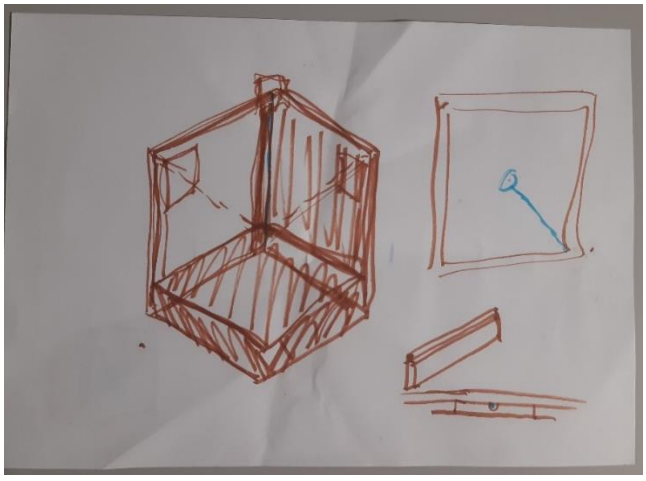
Tratando-se de crescimento de plantas, uma coisa estava clara, seria necessário assegurar as componentes essenciais, ou seja, água, luz e temperatura correta.

Estas funcionalidades obrigatórias foram o eixo principal para projetar todos os restantes componentes. Para a água tinha de conter um reservatório e respetivo processo de rega. Para a luz, teria de colocar alguma luz artificial que obedecesse às preocupações energéticas e fosse ao mesmo tempo capaz de potenciar o crescimento das plantas. A regulação da temperatura seria também essencial, pois tratando-se de um ambiente fechado seria necessário assegurar a ventilação adequada não deixando a temperatura subir demasiado.

DESIGN DA ESTUFA

No mesmo dia, comecei por desenhar alguns esboços do formato físico de todo o suporte. Uma vez que estes me agradaram bastante, não só a nível de eficácia como também esteticamente, decidi prosseguir com neste sentido.





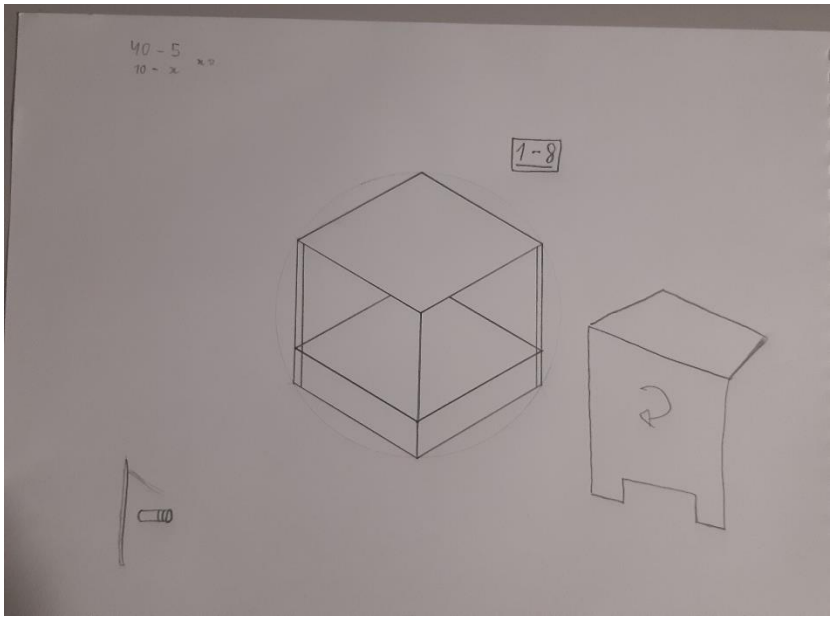
Com o avançar das ideias, tinha que ter algumas preocupações no design da caixa. A primeira delas seria ter a planta no interior da estufa exposta à maior quantidade de luz solar possível, para tal, seria necessário que alguns dos lados da estufa não bloqueassem entrada de luz e fossem transparentes

Como seria um projeto eletrónico e todo ele automatizado, seria também necessário que todos os componentes eletrónicos a colocar, estivessem devidamente isolados da planta para não ficarem danificados. Por outro lado, teria de ter acesso a todos eles para proceder à instalação e montagem.

Antecipando já quais os componentes eletrónicos que seriam necessários, preocupei-me de início em deixar aberturas laterais na caixa para que fosse possível a introdução de ventoinhas para circulação de ar.

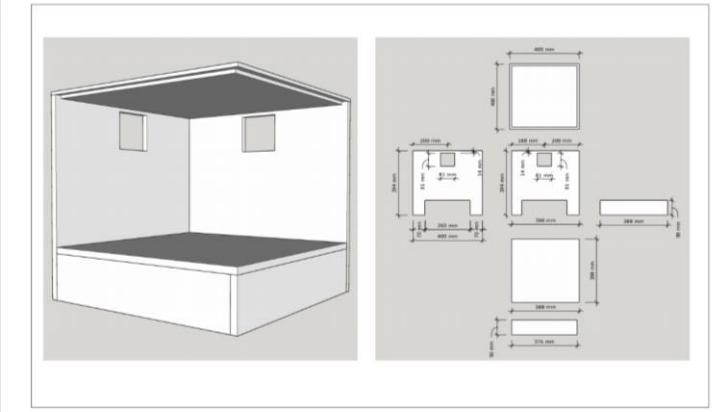
Uma vez que grande parte destas partilham as mesmas medidas, foi possível antecipar a abertura que necessitariam.

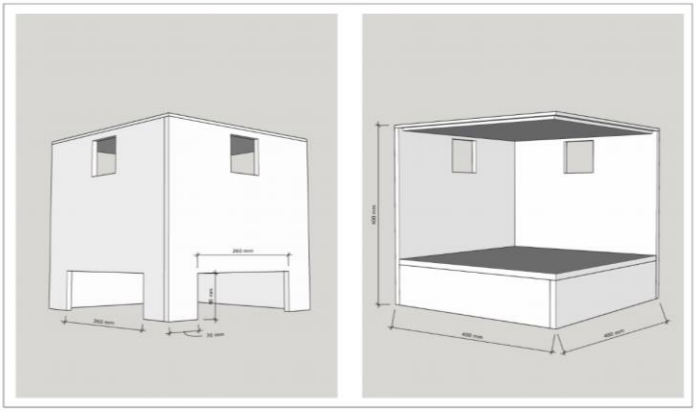
Acabei então por criar um modelo mais profissional e detalhado do formato que a caixa da minha estufa deveria ter.



Sendo que as minhas capacidades de carpintaria não são as melhores, decidir recorrer a alguma ajuda profissional.

Dirigi-me a uma carpintaria próxima do meu local de residência, onde, após explicar e descrever a minha ideia, foi possível efetuar um desenho mais técnico para proceder à criação em madeira do modelo que eu pretendia.





No período de cerca de uma semana todo o modelo estava concluído. Fiquei bastante agradado com o resultado do meu projeto, uma vez que correspondia na perfeição ao que eu tinha idealizado.

Sendo que o trabalho que pedi na carpintaria estava sem acabamento, seria necessário pintá-lo, pelo que acabei por escolher a cor branca, uma vez que é visualmente simples, agradável e encaixava perfeitamente com a ideia do projeto.



Aplicada a pintura, resultou num modelo visualmente bastante apelativo.


Posteriormente foram apenas encaixadas 2 portas de plástico que abrem ao centro, e outros detalhes estéticos menos relevantes. Uma vez que o desenho foi pensado ao pormenor, tudo encaixou na perfeição, dando um ar muito profissional ao meu projeto.

NOME E ÍCONE

Com o projeto a ganhar forma, houve a necessidade de lhe atribuir um nome. Após algumas ideias relacionadas com a temática ambiental e tecnologia surgiu então o nome "BGreen", que desde o primeiro momento fez todo o sentido.

Em primeiro lugar porque estabelece uma conotação com a língua inglesa e para a expressão "Be Green", que traduzido para português se lê "Sê Verde", ou "Seja Verde". Para além disso, este nome estabelecia também uma relação direta com o meu nome de família, e alcunha que me costumam atribuir: "Beirão", através do destaque na letra "B". Uma vez pude contar com um enorme apoio por parte da minha família, o nome adequava se perfeitamente.

Criei de seguida o ícone representativo do meu projeto. Para tal, utilizei uma plataforma online chamada "Pixlr". Decidi incluir o nome do meu projeto no ícone, dando destaque a letra "B", como planeado anteriormente. Criei então algo relacionado tanto com a vertente ambiental como com a tecnologia, tendo resultado num formato bem original.

Durante a criação de outros componentes visuais do meu projeto tive sempre em atenção as cores principais do meu logotipo: branco (#FFFFFF) e verde-escuro (#005015), pelo que utilizei exatamente os mesmos tons de cor em vários componentes diferentes.

Com a construção da caixa e a parte visual concluída, estava então na altura de dar início a construção do restante projeto.

ELETRÓNICA

BASES DO SISTEMA ELETRÓNICO

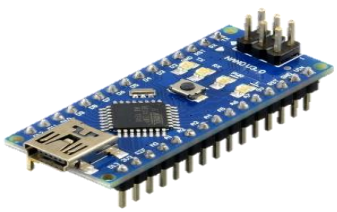
Apesar do sistema eletrónico ter sido o último a ser terminado, este foi construído em conjunto com a aplicação. Deste modo possibilitou-me ir testando o funcionamento de ambas as partes em tempo real.

O sistema eletrónico é o que comanda diretamente os componentes da minha estufa, sejam eles sensores ou outros elementos. Para dar as respetivas instruções a cada componente, é necessário que o sistema contenha uma placa central onde é processada toda a informação.

Neste contexto existiam duas opções principais: uma placa "Rasberry", ou uma "Arduino". As Placas Raspberry são muito utilizadas neste tipo de projetos, uma vez que possuem uma alta capacidade de processamento de informação e uma enorme versatilidade. No entanto estas placas apresentam muitas vezes especificações desnecessariamente elevadas e um consequente aumento do preço. Por outro lado, as placas Arduino têm uma arquitetura focada no baixo custo e recursos estritamente necessários, sendo muitas vezes indicados para projetos escolares.

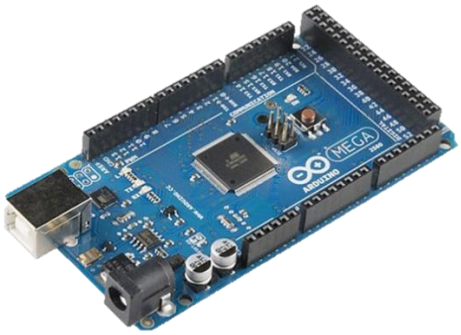
PLACA DE CIRCUITOS

Colocando ambas as opções lado a lado, e dado a minha experiência anterior com placas Arduino, decidi usar uma destas placas no meu projeto. inicialmente optei por utilizar um Arduino "Nano", uma placa bastante pequena e barata, porém com pouquíssimos recursos.



Infelizmente, a falta de recursos veio a evidenciar-se nos últimos momentos. Durante o processo de soldadura de todos os componentes, a uma semana da entrega do projeto, a minha placa principal deixou de funcionar misteriosamente. Tive então que readaptar todo o meu projeto para um modelo superior que já havia utilizado em projetos anteriores.

Trata-se de uma placa "Mega", com muitos mais recursos do que a placa anterior. infelizmente o custo destas placas também é ligeiramente mais elevado, porém, no contexto de pandemia mundial, torna-se muito difícil adquirir componentes num tão curto espaço de tempo.



Foi durante a minha adaptação do projeto a esta placa, que me ocorreu que as falhas sucedidas anteriormente poderiam ou estavam com certeza relacionadas com a falta de recursos.

SENSORES

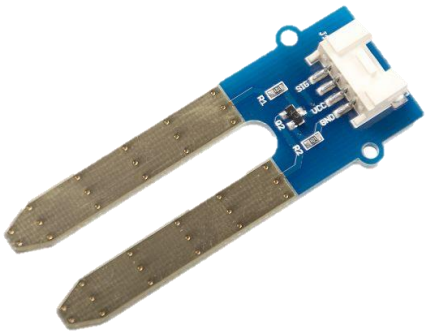
Alguns dos dados que estavam a sobrecarregar a minha placa são provenientes dos sensores que implementei na minha estufa. O primeiro deles é um sensor de luz. Uma vez que existem várias diversidades de plantas, seria necessário adaptar a quantidade de luz pelo utilizador.

Um dos dados que seria necessário monitorizar na minha estufa era a quantidade de luz recebida. Para tal, necessitei de utilizar um sensor de luz, onde está presente uma célula fotossensível, ou seja, capaz de medir a intensidade de luz que incide na mesma.

Pera além da intensidade de luz, é também necessário medir a temperatura dentro do interior da estufa, assim como a humidade do solo onde se inserem as plantas.

Para medir a temperatura utilizei um sensor de luz e humidade que também já possuía de projetos anteriores. Uma vez que a humidade do ar é completamente irrelevante neste contexto, acabei apenas por utilizar os dados relativos à temperatura.

Por último, mas não menos importante, implementei um sensor de humidade de solo. Este, quando espetado na terra onde está contida a planta, é capaz de medir a percentagem de água presente na mesma, o que nos permite saber se é necessária mais ou menos rega na respetiva planta.



Para além dos sensores para monitorizar os dados, necessitei também de alguns componentes maiores e mecânicos, capazes de produzir alterações no microclima dentro da minha estufa.

ELEMENTOS MECÂNICOS

Para que fosse possível fazer circular o ar dentro da estufa utilizei 2 ventoinhas, sendo uma de entrada de ar e uma outra de saída. Deste modo seria possível criar um fluxo de ar dentro da caixa.

Estas ventoinhas foram posicionadas nas aberturas laterais e antecipadamente preparadas para tal.

A minha estufa dispõe também de uma rega, uma vez que é necessário fornecer água às plantas no modo automático. Para tal, necessitei também de acoplar um pequeno depósito de água por baixo da minha estufa. Este pode ser facilmente removido para facilitar o enchimento. Dentro deste depósito está acoplada uma bomba de água, que tem como função empurrar a água pelo tubo de irrigação até à planta.

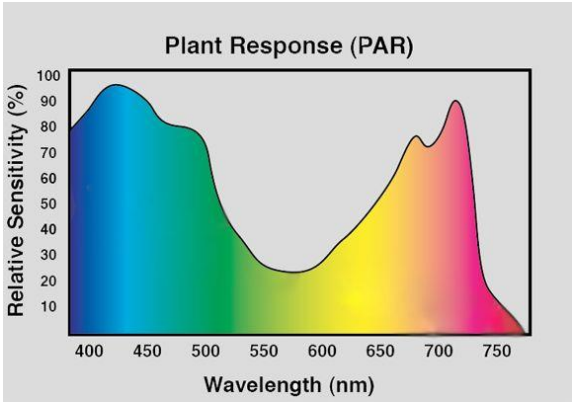
Para além de tudo isto a estufa conta também com luzes LED laterais com o intuito de fornecer não só luz, mas também de aumentar ligeiramente a temperatura no interior da caixa caso necessário.

LUZES

O processo de escolha das luzes LED que seriam utilizadas, foi bastante mais complexo que a escolha de outros componentes, uma vez que existem luzes próprias para o crescimento e desenvolvimento de plantas.

Após alguma investigação acerca do assunto pude concluir que as plantas são muito mais sensíveis a algumas frequências de luz. Deste modo, fornecer à planta luz desnecessária seria apenas desperdício energético, o que me ajuda a atingir um dos meus objetivos iniciais, contribuir para as causas sustentáveis.

As gamas de cores de luz que mais afeta as plantas situam-se tanto em tons de azul como de vermelho. Dado que a luz vermelha tem uma menor frequência que a luz azul e um consequente consumo energético menor, grande parte destas luzes contêm cerca de três quartos da energia total gastos em luz vermelha e o restante quarto em luz azul, o que lhe atribui uma tonalidade de cor-de-rosa.



De acordo com a minha investigação, acabei por investir numas luzes que otimizassem o consumo energético face às necessidades das plantas.

TRANSFERÊNCIA DE ENERGIA

Para que fosse possível ligar todos estes componentes, seria necessária uma alimentação muito maior do que aquela que a placa Arduino oferece.

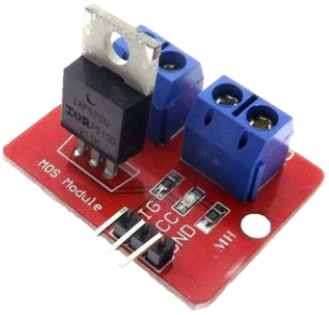
Necessitaria de introduzir algum tipo de alimentação externa para alimentar os componentes de maior necessidade energética.

Uma vez que o circuito eletrónico presente funciona com energia proveniente de corrente contínua, contrariamente à corrente alternada fornecida por uma tomada vulgar, optei por inserir um adaptador externo que se adequasse à voltagem necessária.

Tendo em conta os vários componentes presentes neste contexto acabei por optar por um transformador de 4 Amperes, tendo este sido mais do que suficiente para alimentar todo o sistema.

Sendo que os sinais emitidos pela placa Arduino para ativar os diversos componentes são apenas de 5 volts e os componentes de maior carga energética funcionam a 12 volts, necessitei de introduzir um componente intermédio que fosse capaz de transferir a carga.

Após mais alguma pesquisa, acabei por utilizar transístores do tipo “Mosfet”, sendo estes capazes de receber diferentes voltagens como carga principal e carga controlo. Assim, consegui fornecer à placa Arduino a capacidade de controlar estes componentes maiores sem que a carga do transformador e a da placa entrem em conflito.

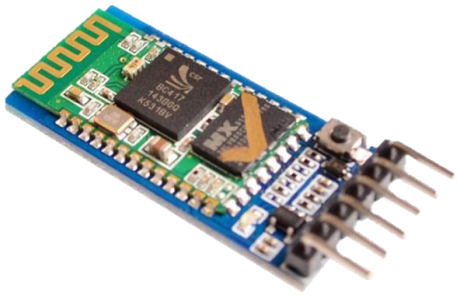


Estes transístores funcionam como um interruptor, ou seja, um dos cabos de energia de cada componente é passado através do transístor, e este apenas permite passagem de corrente quando a respetiva carga energética do Arduino é ativada.

SENSOR BLUETOOTH

Por fim o sensor mais importante de todo o projeto: o sensor Bluetooth. Este sensor é responsável por realizar toda a comunicação entre o telemóvel e a placa Arduino - Estufa.

Para o correto funcionamento deste sensor foi necessário configurar a placa para que a comunicação da mesma fosse realizada através deste sensor, uma vez que, por defeito, esta está configurada para comunicar através de USB.



Este sensor, apesar de muitíssimo eficaz, traz também alguns desafios. O principal deles é o facto de apenas transmitir linhas de texto, ou num contexto de programação - Strings. Deste modo foi necessária uma adaptação de todo o código apenas em torno deste sensor.

PROGRAMAÇÃO ARDUINO

Para programar a placa de circuitos utilizei, como seria de esperar, o ambiente de desenvolvimento do próprio Arduino. Este ambiente exige a ligação através de USB à placa, para conseguir programá-la e executar o respetivo código.

No meu código comecei por importar todas as bibliotecas necessárias e declarar as variáveis para os diferentes mecanismos (rega, luz e ventilação). Declarei também uma variável com o nome "data", responsável por receber as informações vindas do telemóvel via Bluetooth.

```
#include <SoftwareSerial.h>
#include <dht.h>
#include <Wire.h>
//#include <BH1750.h>
//Bibliotecas


long int data;

int buzzer = 10;
int LED1 = 9;
int LED2 = 8;
int LED3 = 7;
```

De seguida criei duas variáveis chamadas "password" 1 e 2, com o objetivo de guardar valores inteiros. Quando o valor da variável *data* corresponder a um destes valores, é porque o utilizador premiu um botão específico na aplicação. Criei também uma variável "code", que deverá adquirir o mesmo valor da variável *data*, e seja possível manipulá-lo sem alterar o valor da variável inicial.

```
long int password1 = 1111;
long int password2 = 2222;

int code;
```

Seguidamente, criei variáveis para todos os valores que possam ser lidos dentro estufa, ou seja, a intensidade de luz, a temperatura e a humidade do solo. Declarei também os pinos da placa onde os respetivos sensores estariam ligados. Declarei em seguida algumas variáveis necessárias para o funcionamento do modo automático.

```
boolean automatico = false;

float temp = 0;

int num1=0;
int num2=0;
int num3=0;

int horas = 0;
int minutos = 0;
int seg = 0;

int humSensor = A7;
int humSensorVal = 0;

int luzSensor = A8;
int lux=0;


dht DHT;
#define DHT11_PIN 3
```

Após a declaração das variáveis dá-se o começo da função "*setup* ()", responsável por inicializar os pinos e algum tipo de comunicação existente.

```
void setup() {  //-----------

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(buzzer,OUTPUT);

  digitalWrite(LED1, LOW);
  digitalWrite(LED2, LOW);
  digitalWrite(LED3, LOW);
  Serial.begin(9600);
}
```

Esta função é seguida de uma outra chamada "loop ()", onde está contido o código que será corrido repetidamente pela placa durante a utilização da estufa. O primeiro processo a ser realizado dentro desta função principal é a leitura de todos os dados presentes nos sensores ligados à placa, que poderão ser, ou não, enviados para o telemóvel.

```
void loop() { //----------------------------------------


  int chk = DHT.read11(DHT11_PIN);
  if ((DHT.temperature > -25) && (DHT.humidity > -25)) {
    temp = constrain (DHT.temperature, -25, 70);
  }

  humSensorVal = analogRead(humSensor) /8 ;
  lux = analogRead(luzSensor);
```

De seguida, segue-se a secção onde os dados são recebidos. Sempre que o utilizador fizer alguma alteração na aplicação Android, serão enviados os respetivos dados para a placa e consequentemente haverá uma leitura desses mesmos dados, cuja confirmação se realiza pela emissão de um sinal sonoro efetuado pela buzzer acoplado para o efeito.

```
if (Serial.available()>0){
  data = Serial.parseInt();

if(data > 0){

  //buzzer
 for(int i=0;i<60;i++)
 {
  digitalWrite(buzzer,HIGH);
  delay(1);
  digitalWrite(buzzer,LOW);
  delay(1);
  }
  delay(50);
     for(int i=0;i<60;i++)
 {
  digitalWrite(buzzer,HIGH);
  delay(1);
  digitalWrite(buzzer,LOW);
  delay(1);
  }
}
```

A variável *code* é então igualada à variável *data* para que possa ser manipulada. São também verificadas as variáveis *password* anteriormente mencionadas.

A primeira serve para atualizar os dados da estufa, pelo que é necessário fazer o envio dos dados de todos os sensores. A segunda *password* serve para desativar o modo automático. Para tal, são desligados todos os componentes que poderão estar ativos, e definidos como zero os valores relativos ao cronómetro do modo automático, sobre o qual falarei mais à frente.

```
if (data==password1){

  Serial.write("");
  Serial.print((int)temp);
  Serial.write(".");
  Serial.print(humSensorVal);
  Serial.write(".");
  Serial.print(lux);
  Serial.write(".");

}

if (data==password2){

  automatico = false;
  seg=0;
  minutos=0;
  horas=0;

  digitalWrite(LED1,LOW);
  digitalWrite(LED2,LOW);
  digitalWrite(LED3,LOW);
}
```

Posteriormente há uma verificação de correspondência entre os dados recebidos e os estados que se enquadram no início do modo automático. Sendo que a placa apenas recebe números inteiros, o início do modo automático está definido para qualquer número entre 3000 e 4999, para que seja possível que o número comece pelo algarismo 3 e contenha mais 3 algarismos relativos aos parâmetros que deverão ser recebidos.

Deste modo cada algarismo, além do primeiro, corresponderá a um parâmetro do modo automático, ou seja, às horas introduzidas pelo utilizador no telemóvel.

```
if (data>3000 && data <4000){     //automatico ON   //  da
  code = code - 3000;
  automatico = true;

  if(code>200 && code <400){code = code-200; num1 = 2;}
  if(code>400 && code <600){code = code-400; num1 = 4;}
  if(code>600 && code <800){code = code-600; num1 = 6;}
  if(code>800 && code <900){code = code-800; num1 = 8;}

  if(code>20 && code <40){code = code-20; num2 = 2;}
  if(code>40 && code <60){code = code-40; num2 = 4;}
  if(code>60 && code <80){code = code-60; num2 = 6;}
  if(code>80 && code <90){code = code-80; num2 = 8;}

  if(code==2){num3 = 2;code=0;}
  if(code==4){num3 = 4;code=0;}
  if(code==6){num3 = 6;code=0;}
  if(code==8){num3 = 8;code=0;}
}
```

De modo semelhante à secção anterior, existe uma segunda verificação, porém, ao invés de se tratar da inicialização do modo automático, trata-se do processo de ligar ou desligar um único componente individualmente.

Nesta verificação, os números recebidos deverão variar entre 4000 e 4999 e os parâmetros recebidos após o primeiro algarismo, que deverá ser 4, servem para indicar qual o componente em questão e se deverá ser ligado ou desligado.

```
if(code > 4000 && code < 5000){
  code =  code - 4000;

  if(code > 99 && code < 200){
    code=code-100;
    if(code == 10){
       digitalWrite(LED1,HIGH);
    }
    else{
      digitalWrite(LED1,LOW);
    }
  }
  if(code > 199 && code < 300){
    code=code-200;
    if(code == 10){
       digitalWrite(LED2,HIGH);
      delay(2000);
      digitalWrite(LED2,LOW);
    }
  }
  if(code > 299 && code < 400){
    code=code-300;
    if(code == 10){
       digitalWrite(LED3,HIGH);
    }
    else{
      digitalWrite(LED3,LOW);
    }
```

Segue-se então o modo automático, sendo efetuada a verificação deste a cada volta do ciclo, estando em funcionamento ou não. Caso esteja em funcionamento, estará também a decorrer um cronómetro referente ao tempo decorrido desde a inicialização do modo. Este cronómetro apenas adiciona um segundo a cada passagem do ciclo, que deverá ter um atraso de 1 segundo para que o cronómetro funcione. O cronómetro fornece um pequeno sinal sonoro à passagem de cada hora, e permite que sejam cumpridas rigorosamente, as horas respetivas ao modo automático.

```
delay(1);

if(seg==59){
  seg = 0;
  if(minutos == 59){
    minutos = 0;
    if(horas == 23){
      horas = 0;
    }
    else{
          horas = horas + 1;

            for(int i=0;i<30;i++)
            {
             digitalWrite(buzzer,HIGH);
             delay(1);//wait for 1ms
             digitalWrite(buzzer,LOW);
             delay(1);//wait for 1ms
             }

    }
  }
  else{
    minutos = minutos + 1;
    Serial.print(horas);
    Serial.print(":");
    Serial.print(minutos);
    Serial.print("\n");
  }
}
else{
  seg = seg + 1;
```

Dá-se então início às verificações do modo automático.

A cada mudança de hora existe uma averiguação dos parâmetros introduzidos pelo utilizador como horas a cumprir, por dia, por cada componente. Para que as horas sejam distribuídas igualmente durante o dia, os únicos valores possíveis de ser introduzidos serão 2, 4, 6 e 8, uma vez que todos estes números são divisores de 24. Cada um destes valores tem horas específicas para ligar e desligar durante um ciclo de 24 horas, para que seja possível que as horas sejam cumpridas.

A rega é uma exceção para este caso, uma vez que esta não poderá permanecer ligada tanto tempo. Nas horas onde seria suposto ser ligada, esta é ligada e desligada, com intervalo de 1 segundo entre os processos. Deste modo, utilizador não deverá introduzir as horas de rega, mas sim a quantidade de regas por dia.

```
if(seg==1 && minutos ==0){

if(automatico && horas==0){    //HORA 0
    digitalWrite(LED1,HIGH);
    digitalWrite(LED3,HIGH);
    digitalWrite(LED2,HIGH);
    delay(1000);
    digitalWrite(LED2,LOW);
}

if(automatico && horas==1){    //HORA 1
    digitalWrite(LED1,LOW);
    digitalWrite(LED3,LOW);
}

                               //HORA 2 n/a

if(automatico && horas==3){    //HORA 3
  if(num1 == 8){
    digitalWrite(LED1,HIGH);
  }
  if(num2 == 8){
    digitalWrite(LED2,HIGH);
    delay(1000);
    digitalWrite(LED2,LOW);
  }
  if(num3 == 8){
    digitalWrite(LED3,HIGH);
  }
}
```

APLICAÇÃO DE TELEMÓVEL – APP

PORQUÊ UMA APLICAÇÃO?

Com o início do projeto, tentei adequar os diversos modelos que encontrei à minha realidade, e como tal, foram necessárias algumas adaptações.

O maior problema que tinha, a meu ver, era a questão financeira, porque todos os projetos que tinha idealizado encontram-se financiados por grandes empresas. No meu caso o budget não cobria exatamente todos os aspetos destes projetos pelo que foi necessária uma readaptação dos mesmos.

Durante a minha investigação verifiquei que um dos componentes que causava o elevado preço destes produtos era me display touch screen. sendo que estes tipos de componentes são geralmente de elevado custo, pensei em readaptar a utilização dos mesmos. Deste modo ponderei várias soluções.

A primeira delas, que foi durante muito tempo a minha ideia principal, foi a utilização de um display simples sem a capacidade de adaptação ao toque. Neste caso seria necessário um meio de introdução de dados. Para tal, utilizaria no meu projeto um conjunto de botões que já possuía de projetos anteriores.

Esta ideia, porém, apresentava alguns problemas, como a difícil introdução de dados por parte do utilizador. Para além disso, seria necessária uma larga quantidade destes componentes para que fosse possível cobrir as necessidades do sistema.

Para além disso, os displays que estariam disponíveis mediante o meu orçamento seriam de reduzida qualidade, o que iria causar algum desconforto e complicação na leitura dos dados. Seria também necessária a calibração do mesmo, uma vez que este tipo de displays necessita em grande parte de uma calibração regular.

Deste modo a utilização deste produto por parte de um Público seria dificultada, pois este necessitaria de algum conhecimento e capacidade de gestão do dispositivo por parte do utilizador. Ponderei então outras soluções que facilitassem a utilização do produto por parte indivíduos menos experientes, como por exemplo display touch screen de

baixo custo, porém dada a experiência negativa que tenho com os mesmos, julgo serem um pouco frustrantes quando não funcionam corretamente.

O meu problema mantinha-se. Foi num destes momentos de reflexão que me ocorreu uma ideia bastante interessante. Uma vez que, nos dias de hoje, qualquer pessoa tem consigo um dispositivo móvel onde está presente um display touch screen de altíssima qualidade, e que se adequa perfeitamente às minhas necessidades, por que não tentar introduzir a utilização do mesmo no meu projeto?

Deste modo, no caso de eu conseguir conciliar este tipo de dispositivos na minha ideia, seria possível, não só visualizar dados provenientes do dispositivo físico, como também controlá-lo remotamente. Poderia então, embarcar noutras funções secundárias que tinha colocado como hipótese.

MODO AUTOMÁTICO - ARDUINO

Sendo que o problema da introdução de dados no dispositivo estava resolvido, previ que teria bastante facilidade em criar uma variedade de comandos diferentes que pudessem ser introduzidos. Assim sendo os dados que toda a componente eletrónica iria receber poderiam ser muito mais variados, e assim sendo, poderia não só introduzir dados binários, como se um botão está permitido ou não, como também dados mais completos.

Existiria assim a possibilidade de dar instruções mais complexas, como por exemplo, a inclusão de parâmetros numéricos. Assim sendo, ocorreu-me a opção de criar um modo completamente automatizado. Este necessitaria não sou de nível mais complexo de programação como também de instruções mais elaboradas por parte do utilizador.

Ponderei vários métodos de automatização deste produto, de entre eles, um modo que se auto regulasse mediante os dados obtidos. Porém, sendo este projeto dedicado a criação de plantas este modo deveria ser o mais volátil possível, uma vez que, diferentes tipos de plantas requerem diferentes tipos de condições.

Foi então que instaurei um modo automático com parâmetros completamente escolhidos pelo utilizador. Este seria responsável pelas condições climáticas dentro da estufa.

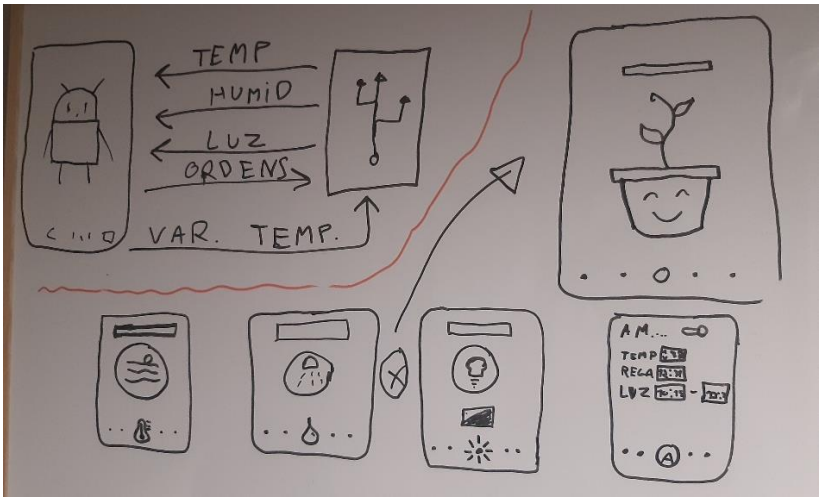
Existiam 3 variáveis a considerar: a iluminação, a ventilação e a rega.

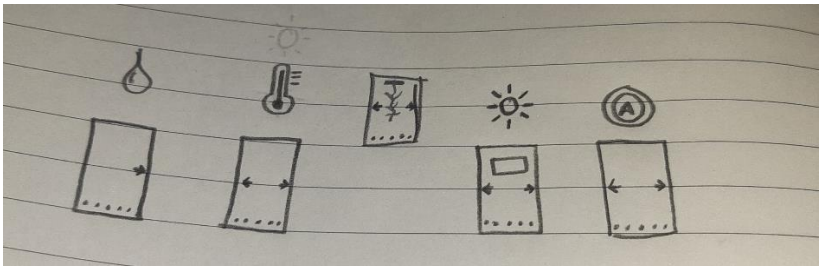
Como se pode observar estes 3 parâmetros são reguláveis em duração, sendo que a duração da rega seria bastante mais curta. Decidi proporcionar ao utilizador do meu produto a capacidade de gerir, em horas, a quantidade de energia que seria despendida nestas funcionalidades. Seria possível também preservar bastante energia, desligando os componentes, contribuindo consideravelmente para o meu objetivo inicial de relacionar o meu projeto às causas ambientais.

A partir do momento em que a quantidade de horas de funcionamento do dispositivo passa a estar dependente do utilizador do produto, as horas, diárias, que cada componente está ligado pode ser executada de 2 formas distintas: ou as horas são executadas de seguida logo após a indicação do utilizador, ou são distribuídas ao longo do dia ciclicamente, porém este processo nada tem que ver com a aplicação, mas sim com a programação posterior dos circuitos eletrónicos.

INTERFACE APP

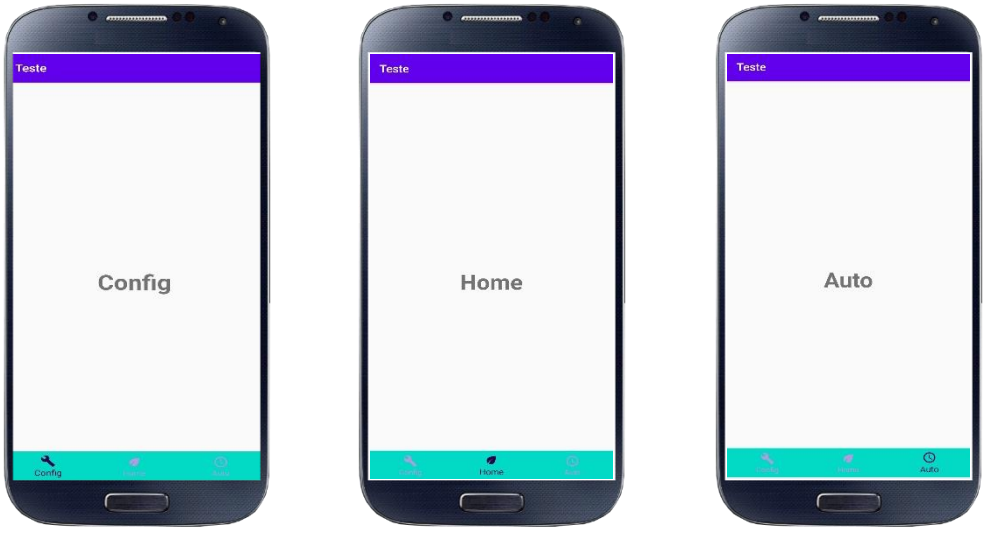
O design do esquema de navegação e da própria interface visual da aplicação foi um processo bastante agradável e interativo, onde obtive o auxílio dos meus familiares mais próximos para me darem algumas opiniões e críticas construtivas. Não sendo pessoas ligadas diretamente às ciências informáticas, a sua opinião foi muito mais válida, uma vez que esta não seria influenciada por conhecimento relativo ao funcionamento destas tecnologias. Deste modo pude adotar uma postura muito mais criativa e que não fosse limitada pelas ferramentas que estavam à minha disposição.





Necessitaria de criar um esquema de navegação que fosse simultaneamente simples, apelativo e que dispusesse toda a informação necessária. Com base nestes pressupostos estipulei que a minha aplicação móvel seria constituída por 3 painéis, sendo eles: uma página principal que apresentasse toda a informação, uma página dedicada a toda a automação, e uma terceira página volátil, tendo esta o objetivo inicial de ser dedicada a algum tipo de configuração necessária.

A minha primeira decisão relativa a estrutura desta aplicação foi a criação de um menu inferior que permitisse a troca entre painéis (Definições, Home e Automático). Deste modo o utilizador poder-se-ia movimentar livremente entre painéis sem recorrer a um menu muito complexo. A fluidez da minha aplicação é um dos aspetos mais positivos da mesma, sendo que tem um visual muito simples e agradável.



Durante este processo de design do interface, deparei-me com o maior problema do meu projeto até ao momento, que consequentemente, me obrigou a tomar a maior decisão do meu projeto: onde seria construída a minha aplicação.

Até ao momento, tinha ponderado criar uma aplicação relativamente simples utilizando uma ferramenta online chamada "MIT Inventor", uma plataforma onde é possível desenhar e criar aplicações simples para Android. Todo o design inicial da minha aplicação foi realizado nesta plataforma, porém durante o esse processo de criação reparei que esta instrumento tem bastantes limitações quando se pretende criar algo mais profissional. Fui assim obrigado a pesquisar por outras ferramentas de criação e programação de aplicações.

Após vários dias a investigar e experimentar diversos softwares, deparei-me com a ferramenta "Android Studio", um ambiente de desenvolvimento integrado (IDE), muito famoso na criação deste tipo de interface.

FERRAMENTA ANDROID STUDIO

Após uma pesquisa intensiva sobre o funcionamento desta ferramenta e dos seus fundamentos, comecei a assistir variados vídeos educativos relativos ao tema. Para minha surpresa a ferramenta é relativamente simples de se utilizar, porém, qualquer

indivíduo que não possua conhecimentos base na área da informática, e mais concretamente na programação, deparar-se-á com um ambiente bastante intimidante.

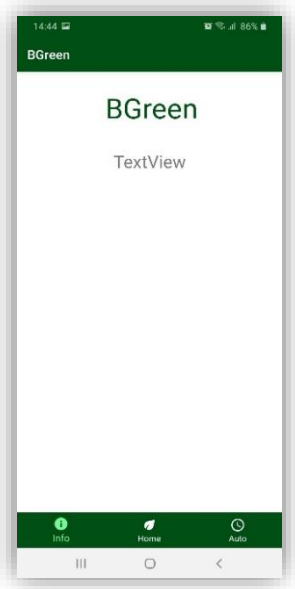
Para mim todo o funcionamento deste software pareceu-me bastante familiar, uma vez que a linguagem utilizada neste programa varia entre Kotlin, e Java, uma linguagem já lecionada no nosso curso. Desta forma entrei "com o pé direito" na minha atividade.

Após alguma investigação base e algumas horas de pesquisa no Google, decidi realizar o conteúdo de alguns tutoriais relativos a esta matéria na plataforma YouTube.

Tenho a mencionar que fiquei genuinamente surpreendido com o quão positiva e amigável esta comunidade se revela, comparativamente com outros tipos de matérias, os profissionais desta área são bastante ativos em fóruns e discussões sobre os tópicos, o que se revela uma ajuda imensa a pessoas novas neste tema.

A própria ferramenta Android Studio, tem por base um software com o qual eu já tinha tido contato. Este é proveniente de um outro IDE, com o nome "IntelliJ", uma ferramenta que permite a criação e programação de programas na linguagem Java. Como é óbvio, também a minha experiência prévia com este software me deu algumas luzes de como a nova ferramenta funcionava.

Após algumas tentativas de desenhar uma aplicação através da interface visual Android Studio decidi começar definitivamente o meu projeto nesta plataforma, pois esta revelou-se bastante agradável de utilizar.



Comecei por desenhar e criar os 3 elementos principais constituintes da minha aplicação, sendo cada um deles uma classe de Java independente das outras. Dei início à implementação das ideias que tinha tido anteriormente, introduzindo novas ferramentas que tornavam a aplicação mais consistente e sólida.

Foi durante este processo de embelezar o visual criado que a minha família me deu a sua opinião, que tive em grande consideração. Durante o processo de criação aprendi bastante sobre o conceito de design em geral, como por exemplo, aprendi que um design não deve ser apenas apelativo para aquele que o cria, mas sim para o maior número de pessoas possível, uma vez que quanto mais pessoas tiverem uma opinião positiva acerca do design, mais este se aproximará do design ideal que deve ser apresentado ao mercado. Esta ideia ficou bastante presente na minha cabeça durante o restante do processo de criação do produto, não só a nível de design, como também a nível de funcionalidades e funcionamento das mesmas.

Após todo o trabalho de edição da minha aplicação, esta ficou na sua forma final, tendo eu adotado um esquema de cores, muito intuitivo e agradável, tendo por base as cores representativas do meu projeto.

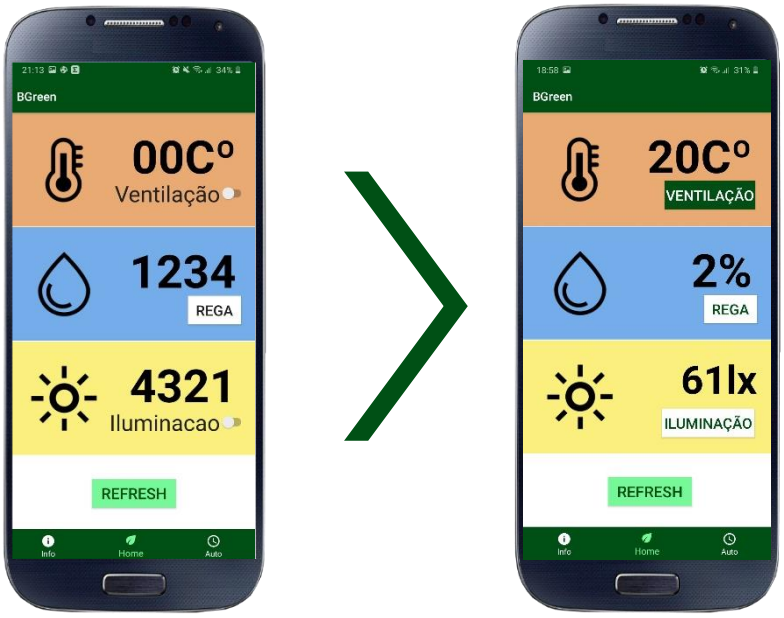
Dei assim por terminado o desenho da minha aplicação. Após este processo comecei a idealizar o respetivo funcionamento que toca à programação das respetivas ferramentas.

PROGRAMAÇÃO APP

PÁGINA PRINCIPAL

Com a aposta na simplicidade e praticidade da aplicação, todo o painel principal foi pensado em conformidade com estes objetivos. O meu foco foi torná-lo visualmente apelativo, uma vez que este se trata do painel principal da aplicação, aquele que será aberto sempre que a aplicação for executada.

Procedi à criação dos botões para o painel principal, em vez de recorrer à ferramenta "Switch" que tinha utilizado inicialmente, processo esse que mantive para os restantes separadores da aplicação.



Graças à ferramenta responsável pelas preferências do sistema, foi-me possível guardar as alterações feitas na página principal. Estes botões têm por objetivo ligar e desligar componentes isoladamente, ou seja, ao contrário do modo automático no qual os componentes ligam e desligam automaticamente em horas pré-programadas, os botões da página principal são responsáveis por ligar cada componente individualmente.

Desta forma, no caso de o utilizador premir o botão responsável por ligar as luzes da estufa, estas permanecerão ligadas até ordem em contrário.

Consequentemente, também os botões responsáveis por estas ações deverão mudar o seu visual consoante o estado do componente em questão. Esta mudança tornou a aplicação muito mais intuitiva e atual.

A rega, ao contrário dos outros componente, não tem a possibilidade de permanecer ligada, pois inundaria por completo a estufa. Neste caso, optei por deixar o botão relativo à rega incapaz de mudar o seu visual, e ficará ligado apenas por breves segundos.

Sendo que iriam existir 3 painéis na minha aplicação, estava na altura de pôr em funcionamento o menu inferior responsável por fazer a troca entre os diferentes fragmentos.

BOTTOM NAVIGATION VIEW

Para criação desta funcionalidade utilizei uma ferramenta chamada "Bottom Navigation View". Esta ferramenta veio a ser a base da navegação dentro da minha aplicação, tornando muito simples a movimentação do utilizador entre fragmentos. Esta, organiza um conjunto de botões inseridos num friso junto à parte inferior do monitor.

A sua implementação revelou-se bastante complexa de início, uma vez que nunca tinha tido contato com esta ferramenta anteriormente. Comecei então a pesquisar em vários fóruns e vídeos, para tentar compreender o funcionamento desta ferramenta. Acabei por encontrar alguns vídeos educativos na plataforma YouTube, que me ajudaram bastante na compreensão do instrumento.

Deste modo, tive de executar um conjunto de procedimentos para que fosse possível importar esta ferramenta para o meu projeto, uma vez que esta não se encontra disponível por defeito no ambiente de desenvolvimento Android Studio.

Comecei por incluir todas as bibliotecas e dependências necessárias. Os vídeos aos quais assisti ensinaram-me tudo o que precisava de saber para implementar a minha ideia, pelo que, após algum tempo a navegar pelas pastas e subpastas de ficheiros no meu projeto, achei finalmente a localização dos parâmetros que pretendia personalizar,

de entre os quais, a adequação dos nomes dos respetivos botões, a modificação de imagens ilustrativas para cada um deles e a palete de cores presente nesta parte da minha interface visual.

```
bottomNavigationView.setOnNavigationItemSelectedListener(new BottomNavigationView.OnNavigationItemSelectedListener() {
@Override
    public boolean onNavigationItemSelected(@NonNull MenuItem menuItem){

    switch(menuItem.getItemId()){
            case R.id.info:
                startActivity(new Intent(getApplicationContext(),Info.class));
                overridePendingTransition( enterAnim: 0, exitAnim: 0);
                return true;
            case R.id.home:
                return true;
            case R.id.auto:
                startActivity(new Intent(getApplicationContext(),auto.class));
                overridePendingTransition( enterAnim: 0, exitAnim: 0);
                return true;

        }
        return false;
    }
```

O funcionamento desta ferramenta baseia-se num conjunto de botões com diferentes identificações. Quando um botão é premido, a identificação do mesmo será interpretada e com base no resultado, será aberta a classe que corresponde ao botão premido. Para além disto existe uma variável nos parâmetros da minha aplicação que varia consoante a classe aberta no momento, o que faz com que o menu inferior destaque a página em que o utilizador se encontra, tornando-o assim muito mais dinâmico.

MODO AUTOMÁTICO - APP

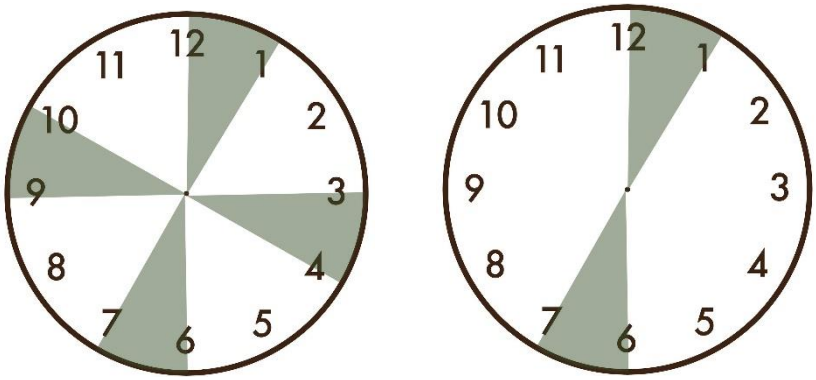
A primeira coisa que fiz foi inserir todos os botões e caixas de texto existentes, para que a interface visual da minha aplicação fosse dinâmica e estivesse constantemente a adaptar-se às instruções do utilizador.

```
ImageButton Automatico;
ImageButton horasLuzDiaMais;
ImageButton horasLuzDiaMenos;
ImageButton regasDiaMais;
ImageButton regasDiaMenos;
ImageButton horasVentilacaoDiaMais;
ImageButton horasVentilacaoDiaMenos;
TextView horasLuzDia;
TextView regasDia;
TextView horasVentilacaoDia;
```

Para que fosse possível a introdução de valores como parâmetros no modo automático, necessitaria de caixas de texto onde fosse possível visualizar previamente os dados que seriam enviados. Criei também botões laterais e essas caixas de texto para que fosse possível a alteração dos valores nas mesmas. Defini um limite nas caixas de texto para que fosse impossível a introdução de valores inválidos.

Foi durante este processo que tomei uma decisão bastante importante, decidi que durante o modo automático, as horas de luz, ventilação ou rega, seriam igualmente distribuídas durante um período de 24 horas, ou seja, se o utilizador introduzisse o valor 2, por exemplo, como horas de ventilação por dia, a estufa ligaria 1 hora a cada 12 horas correntes, em vez de 2 horas seguidas logo após a inicialização deste modo.

Assim sendo tive de adaptar os valores que seriam válidos na sua introdução. Para tal, defini que os valores possíveis seriam 2, 4, 6 e 8. Como é possível observar, todos estes números são divisores de 24, o que faz com que num período de 24 horas seja possível equilibrar cada um destes valores num ciclo de hora a hora.



A partir daí pude definir estes valores como únicos valores de possível introdução nas caixas de texto através dos botões, de cada vez que o valor dentro de uma caixa de texto

se igualasse ao limite, o botão ficaria sem efeito de modo a impedir que o valor se aumente ou diminua.

Mais tarde implementei o desaparecimento dos botões por completo, caso o valor seja um dos valores limitantes (2 ou 8), o botão de mais ou menos deverá desaparecer por completo não dando sequer a opção ao utilizador de exceder esse limite.

Invertendo este processo, programei os botões para reaparecer caso o valor em questão se altere, ou seja, se um dos valores for definido como 2, o botão que diminuirá esse valor deverá desaparecer, porém, caso esse mesmo valor se altere novamente para um valor superior, o botão de diminuição do valor deverá reaparecer.

```
horasLuzDiaMais.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        CharSequence s = horasLuzDia.getText();
        String s1 = s.toString();
        int s2 = Integer.parseInt(s1);
        if(s2 == 2){
            String s3 = Integer.toString( i: 4);
            horasLuzDia.setText(s3);
            horasLuzDiaMenos.setVisibility(View.VISIBLE);
        }
        if(s2 == 4){
            String s3 = Integer.toString( i: 6);
            horasLuzDia.setText(s3);
        }
        if(s2 == 6){
            String s3 = Integer.toString( i: 8);
            horasLuzDia.setText(s3);
            horasLuzDiaMais.setVisibility(View.GONE);
        }
        saveState();
    }
});
```

Foi durante este período que me comecei a deparar com alguns dos problemas relativos ao modo automático, sendo o primeiro deles o facto de aplicação ter de estar sincronizada com o respetivo dispositivo físico. Para isso criei um botão principal neste painel com o objetivo de indicar se o modo automático está ligado ou desligado.

Este botão tem por base uma variável *booleana*, binária, que permita identificar o estado tanto da aplicação como do dispositivo. Achei por bem programar o botão para que alterasse o seu visual em função do estado do modo automático, esse botão deverá fornecer indicação visual relativamente ao seu estado.

Acrescentei também a indicação de que todos os botões que permitem alterar valores fiquem indisponíveis quando o modo automático é ativado, para que não seja possível alteração de valores durante o funcionamento do respetivo modo.

Esta indicação visual gerou um problema de conflito com o código previamente criado uma vez que todos os botões da alteração de valores devem reaparecer quando o modo automático é desligado. Assim sendo todos os botões tornariam a aparecer, independentemente do valor contido nas caixas de texto.

Para resolver esta questão, programei o botão responsável por ligar o modo automático para que este lê-se todos os valores presentes nas caixas de texto durante a desativação do modo, com a finalidade de fazer desaparecer todos os botões que deveriam estar ocultos.

```
modoAutomatico = "0";
Automatico.setImageResource(R.mipmap.autooff_foreground);

CharSequence s = horasLuzDia.getText();           //--
String s1 = s.toString();                        // Horas Luz Dia ...
int s2 = Integer.parseInt(s1);                  //--

CharSequence t = regasDia.getText();              //--
String t1 = t.toString();                        //
int t2 = Integer.parseInt(t1);                  //--

CharSequence u = horasVentilacaoDia.getText();    //--
String u1 = u.toString();                        //
int u2 = Integer.parseInt(u1);                  //--

if(s2 != 8){
    horasLuzDiaMais.setVisibility(View.VISIBLE);
}
if(s2 != 2){
    horasLuzDiaMenos.setVisibility(View.VISIBLE);
}

if(t2 != 8){
    regasDiaMais.setVisibility(View.VISIBLE);
}
if(t2 != 2){
    regasDiaMenos.setVisibility(View.VISIBLE);
}
```

Surgiu então outro problema.

Quando um utilizador desliga a aplicação,o todos os dados temporariamente armazenados nesta, como variáveis, são eliminados. Assim sendo, quando um utilizador fechasse a aplicação o modo automático seria desativado e todos os botões relativos ao mesmo começariam também a apresentar falhas. Para resolver esta questão comecei a investigar formas de armazenar os dados permanentemente no dispositivo.

Durante algum tempo considerei a criação de ficheiros de armazenamento de dados a que só a aplicação tivesse acesso, deste modo os dados estariam sempre corretos e em conformidade como modo em que o dispositivo físico se encontra. Após uma árdua investigação comecei a duvidar da minha escolha uma vez que a criação de ficheiros numa aplicação Android é bastante complexa.

Após mais alguma pesquisa decidi então implementar no meu projeto uma ferramenta vulgarmente conhecida por preferências do sistema.

PREFERÊNCIAS DO SISTEMA

Preferências do sistema consistem em pequenos dados armazenados como configurações de uma aplicação. Para meu espanto, esta ferramenta é muito utilizada para guardar valores respetivos a recordes em jogos e armazenamento do progresso dos jogadores nos mesmos. Esta ferramenta permite armazenar dados na própria aplicação sem que estes sejam apagados sem o consentimento do utilizador.

Para que estes dados sejam eliminados, é necessário que o utilizador, através das definições do dispositivo, apague os dados da aplicação, o que é praticamente impossível de acontecer por acidente.

Assim sendo, fui capaz de armazenar não só o estado do modo automático dentro da aplicação, como também os valores introduzidos pelo utilizador na inicialização do mesmo.

Para isso criei 3 funções com os nomes *saveState* (), *loadState* () e *showState* ().

A primeira função tem por objetivo armazenar os dados presentes na aplicação sempre que há alguma alteração. Para que esta função funcione corretamente, é acionada cada vez que há alguma alteração no conteúdo mostrado, ou seja, sempre que um botão e premido.

```
private void saveState(){

    SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFS, MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();

    editor.putString(MODO_AUTOMATICO, modoAutomatico);
    editor.putString(NUMERO_1, horasVentilacaoDia.getText().toString());
    editor.putString(NUMERO_2, regasDia.getText().toString());
    editor.putString(NUMERO_3, horasLuzDia.getText().toString());

    editor.apply();
}
```

As outras 2 funções servem respetivamente para carregar e mostrar os dados.

A primeira delas vai buscar os dados armazenados pela aplicação em utilizações anteriores, e é executada no inicializar do painel do modo automático. Esta função atribui os valores previamente guardados em variáveis temporárias que serão utilizadas pela função seguinte para alterar os valores da aplicação em funcionamento. Isto impõe que ambas as funções sejam utilizadas seguidamente para que não haja falhas no carregamento e visualização dos dados.

```
private void loadState(){

    SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFS, MODE_PRIVATE);
    modoAutomatico = sharedPreferences.getString(MODO_AUTOMATICO, defValue: "0");

    hVD = sharedPreferences.getString(NUMERO_1, defValue: "4");
    rD  = sharedPreferences.getString(NUMERO_2, defValue: "4");
    hLD = sharedPreferences.getString(NUMERO_3, defValue: "4");

}
```

```
horasVentilacaoDia.setText(hVD);
regasDia.setText(rD);
horasLuzDia.setText(hLD);
```

Uma vez que todos os botões aparecem na inicialização do painel do modo automático, implementei nesta última função uma cópia do esquema de verificação dos valores limites, para que no caso de algum dos parâmetros ter sido previamente definido como um valor limite, o respetivo botão da alteração do valor fica indisponível com o inicializar do painel.

Após este processo realizei uma larga quantidade de testes, através dos quais pude comprovar que o modo automático permanecia ligado ou desligado mesmo após desativar a aplicação. Para além disso, também os valores ficavam corretamente armazenados. Pude verificar ainda que os botões desapareciam caso o valor nas caixas de texto não puder ser alterado, tal como tinha planeado previamente.

Durante o meu período de programação deste software deparei-me com a dificuldade em seguir processos não visuais, ou seja, a verificar o correto funcionamento de operações de processamento de informação não visíveis. Para tal criei uma pequena função que me permite mostrar “Toasts”, pequenas indicações visuais que aparecem no ecrã durante a utilização da aplicação.

```
private void showToast (String msg){
    Toast.makeText( context: this, msg, Toast.LENGTH_SHORT).show();
}
```

Apesar de relativamente simples, esta função ajudou-me bastante durante o meu trabalho, dando-me a possibilidade de mostrar valores durante a execução do código.

A função recorre a uma ferramenta do Android para mostrar as mensagens, porém, eu próprio adaptei a mesma para que fosse mais simples e rápida de utilizar.

Com 2 dos 3 painéis terminados comecei então a programação do sistema Bluetooth responsável pela transferência de dados entre o meu telemóvel e a componente eletrónica do meu projeto.

PÁGINA DE INFORMAÇÃO

Esta página tem por objetivo principal oferecer ao utilizador algum tipo de informação relativa ao projeto e à Aplicação, servindo também como uma página de créditos.

Esta é apenas composta por uma grande secção de texto, onde é possível dar scroll, que apresenta informações relativas ao criador do trabalho e a aplicação em si. Adicionei também uma versão aumentada do logotipo do meu projeto, apenas por questões de estética.

MÓDULO BLUETOOTH

O módulo de Bluetooth foi provavelmente a etapa mais complicada e frustrante de todo o meu projeto. Uma vez que, durante a minha experiência com as ferramentas online de criação de aplicações, me habituei a programar sistemas de Bluetooth através duma interface visual bastante agradável, que grande parte das vezes se resumia a juntar e organizar blocos visuais que representavam código de programação. No entanto vi-me confrontado com procedimentos e estruturas com as quais nunca antes tinha trabalhado.

Dei início a uma longa e exaustiva pesquisa na internet relativa a este Tema. Durante um largo período de tempo tentei variados métodos de comunicação via Bluetooth.

Depois de inúmeras experiências, incontáveis reconstruções do meu código, e uma longa aprendizagem por tentativa e erro, acabei por criar uma versão concatenada de várias funções que encontrei na internet.

Com base em projetos semelhantes que encontrei em diversos fóruns, websites e vídeos educativos, juntei alguns aspetos comuns que fossem realmente eficientes, simples, e que se adequassem ao funcionamento que eu havia previsto.

Comecei por criar 5 funções principais para poder efetuar a minha comunicação, e 3 outras secundárias com o propósito de interpretar os dados recebidos.

Durante o processo de criação das mesmas descobri que tinha a possibilidade de conectar o meu dispositivo eletrónico ao telefone sem a necessidade de qualquer configuração durante o uso da aplicação. Para que este sistema funcione corretamente o utilizador deverá apenas conectar o seu dispositivo telefónico ao sensor de comunicação presente na placa de circuitos uma única vez antes de qualquer utilização da aplicação. Para tal, basta aceder às configurações do dispositivo e conectar-se ao sensor como se tratasse de qualquer outro dispositivo Bluetooth, tipo áudio ou outros periféricos.

Resumindo o funcionamento da técnica que adotei, esta tem por base 2 parâmetros de um dispositivo Bluetooth: o nome e o endereço Mac do componente. Sendo que o módulo que utilizei tem o nome “HC-05” por defeito, pude adaptar as minhas funções para que se ligassem automaticamente ao componente com este nome.

As 2 primeiras funções que criei têm os nomes: *findBT* () e *openBT* () respetivamente.

A primeira função, *findBT* (), tem como objetivo verificar o acesso ao Bluetooth do dispositivo por parte da aplicação e definir o dispositivo que será conectado, neste caso, o módulo Bluetooth da componente eletrónica.

A função começa por verificar se existe um adaptador de Bluetooth no dispositivo móvel, e se está ou não ligado. No caso de uma destas condições não se verificar, a aplicação mostrará a respetiva mensagem. De seguida, a aplicação cria um objeto do tipo "BluetoothAdapter", ao qual é atribuído o registo do componente físico do telemóvel. A função recorre então a uma outra função secundária do adaptador de Bluetooth para receber os dados relativos a todos os dispositivos emparelhados ao dispositivo telefónico. Após efetuado este processo, a função deverá correr todos os dispositivos conectados em busca daquele em que o nome coincida com o nome estabelecido, neste caso, "HC-05".

Após encontrado o respetivo dispositivo, a função deverá atribuir o seu valor a um outro objeto do tipo "BluetoothDevice", capaz de controlar o dispositivo Bluetooth que será conectado, para que seja possível estabelecer uma comunicação com o mesmo nas seguintes funções.

```
Set<BluetoothDevice> pairedDevices = mBluetoothAdapter.getBondedDevices();
if(pairedDevices.size() > 0)
{
    for(BluetoothDevice device : pairedDevices)
    {
        if(device.getName().equals(nome))
        {
            mmDevice = device;
            break;
        }
    }
}
```

A segunda função, OpenBT (), tem por objetivo estabelecer uma comunicação entre o dispositivo Android e o aparelho Bluetooth definido como "device" na função anterior. Esta função recorre à identificação padrão da porta Bluetooth do dispositivo para conectar o dispositivo selecionado ao socket pré-definido. Esta função prepara também o próprio socket para a comunicação definindo os canais de input e output do mesmo.

```
void openBT() throws IOException {
    UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb"); //Standard SerialPortService ID
    mmSocket = mmDevice.createRfcommSocketToServiceRecord(uuid);
    mmSocket.connect();
    mmOutputStream = mmSocket.getOutputStream();
    mmInputStream = mmSocket.getInputStream();

}
```

A terceira função, "*beginListenForData* ()", é provavelmente a função mais complexa das 5, uma vez que deverá receber e interpretar todas as mensagens recebidas via Bluetooth.

Esta começa por declarar alguns parâmetros necessários para o funcionamento das subfunções. De seguida começa um ciclo de modo a interromper o funcionamento desta função caso a comunicação seja interrompida ou a mensagem termine.

```
void beginListenForData() {
    final Handler handler = new Handler();
    final byte delimiter = 10; //This is the ASCII code for a newline character
    final String[] dataFinal = {"null"};

    stopWorker = false;
    readBufferPosition = 0;
    readBuffer = new byte[1024];
    workerThread = new Thread((Runnable) () -> {
            while(!Thread.currentThread().isInterrupted() && !stopWorker)
            {
```

Dentro desse mesmo ciclo é criada uma variável chamada "bytesAvailable", que irá limitar a quantidade de bytes a serem recebidos pela função. De seguida, no caso de o número de bytes disponíveis ser superior a zero, será criado um vetor de variáveis do tipo byte que irão adquirir os valores de cada byte individualmente à medida que estes vão sendo recebidos. Estes serão então compilados para uma string com nome "data", que será então enviada no final da leitura para uma variável global chamada mensagem.

```
int bytesAvailable = mmInputStream.available();
if(bytesAvailable > 0)
{
    byte[] packetBytes = new byte[bytesAvailable];
    mmInputStream.read(packetBytes);
    for(int i=0;i<bytesAvailable;i++)
    {
        byte b = packetBytes[i];
        if(b == delimiter)
        {
            byte[] encodedBytes = new byte[readBufferPosition];
            System.arraycopy(readBuffer, srcPos: 0, encodedBytes, destPos: 0, encodedBytes.length);
            final String data = new String(encodedBytes, charsetName: "US-ASCII");
            readBufferPosition = 0;

            handler.post(() -> {
                    mensagem = elimina(data);
            });
        }
```

Posteriormente criei uma função denominada "*sendData* ()", que tem por objetivo, como o nome indica, enviar dados para a componente eletrónica. Apesar de simples, esta é provavelmente a função mais importante de todo o programa, uma vez que fornecerá informação para a placa de circuitos com as ordens sobre quais os componentes que devem ser inicializados. Esta função é também utilizada quando se pretende receber dados da placa, uma vez que necessitará de uma descrição relativa ao teor de informação que deve fornecer.

Durante a utilização do painel referente ao modo automático, a informação que será enviada à placa apenas dirá respeito aos parâmetros a introduzir na inicialização deste modo. Para isso, serão enviados diferentes números representantes de diferentes ordens.

A função é apenas constituída por 2 subfunções iguais. Uma delas serva para reconhecer e enviar os dados, enquanto a segunda é dedicada a limitar a Mensagem.

```
void sendData(String msg) throws IOException {

    mmOutputStream.write(msg.getBytes());
    mmOutputStream.write( b: 'A');
}
```

A última função que criei tem o nome de "*closeBT* ()", que serve para terminar a conexão entre o telemóvel e a placa de circuitos. Sendo que cada fragmento da minha aplicação tem uma classe de Java individual, foi necessária a implementação de uma reconexão cada vez que a página é alterada. Para isso, a conexão Bluetooth é encerrada antes da página ser trocada, e reaberta na página seguinte, o que implica que esta função seja utilizada recorrentemente.

O método de funcionamento desta função é bastante simples, uma vez que se resume ao término de todos os processos relacionados com a comunicação.

```
void closeBT() throws IOException {
    stopWorker = true;
    mmOutputStream.close();
    mmInputStream.close();
    mmSocket.close();
}
```

Durante os testes que realizei às minhas funções, pude reparar que os dados enviados da placa Arduino para a aplicação Android estavam sempre contidos numa única string e numa única linha de texto. Este problema teria de ser contornado caso eu quisesse apresentar vários dados no meu telemóvel.

Uma vez que era praticamente impossível distinguir qual dos valores dizia respeito a que parâmetro, necessitaria de algum tipo de marcação no texto para que pudesse distinguir estes valores, uma vez que os algarismos dos números seriam praticamente aleatórios.

Foi então que, após algum tempo a criar este esquema, acabei por me recordar de um processo aprendido na escola. Este processo seria o protocolo IP, uma vez que também este necessita de marcações para distinguir números.

Criei então um sistema de pontuação que me permitisse enviar apenas uma linha de dados que descrevesse vários números distintos. Necessitaria então de uma função dentro do código Android que fosse capaz de interpretar o código enviado pela placa.

Foi então que criei a função “separar ()”, capaz de dividir a minha linha de texto e retirar da mesma os diversos valores que deverão ser apresentados quando os dados são atualizados.

A função começa por criar 4 mensagens e de seguida define uma variável para a parte do texto em que a leitura se encontra. Seguidamente a função corre cada um dos caracteres da Mensagem e armazena como a primeira Mensagem. Assim que um dos caracteres lidos seja um ponto (.), a função deverá delimitar a primeira parte da Mensagem, dando início a parte seguinte. Deste modo, no final da leitura de todos os caracteres a função deverá ter até um máximo de 4 mensagens, sendo que 3 deles

servirão para transportar os dados, e a última delas para algum tipo de informação adicional que seja necessário descrever.

```
private String separar(int ordem, String msg){

    String msg1 = "";
    String msg2 = "";
    String msg3 = "";
    String msg4 = "";

    int parte = 1;

    for(int i=0;i < msg.length();i++){
       if(msg.charAt(i)=='.'){
          parte++;
       }
       else{
           switch(parte){
               case 1: msg1 += msg.charAt(i);
               break;
               case 2: msg2 += msg.charAt(i);
               break;
               case 3: msg3 += msg.charAt(i);
               break;
               case 4: msg4 += msg.charAt(i);
               break;
           }
       }
    }
```

Como parâmetro para esta função devemos inserir um número inteiro, correspondente à parte do texto que pretendemos que seja devolvida. Isto requer que a função seja utilizada cada vez que é necessário atualizar um dado individualmente. Devemos também inserir a mensagem recebida via Bluetooth nos parâmetros da função, para que esta tenha uma mensagem para separar.

```
if(ordem == 1){
    return msg1;
}
else if(ordem == 2){
    return msg2;
}
else if(ordem == 3){
    return msg3;
}
else{
    return msg4;
}
```

Criei também uma função com o nome “elimina ()”. Esta função tinha o intuito de remover o último caracter da Mensagem recebida por Bluetooth uma vez que neste método de comunicação existe sempre um caracter que limita a mensagem. Durante o processo de criação do meu projeto, esta função foi bastante útil, porém, não foi mais necessária a sua utilização após a criação da função “separar ()”, uma vez que estes caracteres adicionais sobrariam para a tal quarta parte, não sendo interpretados.

```
public String elimina(String str) {
    if (str != null && str.length() > 0) {
        str = str.substring(0, str.length() - 1);
    }
    return str;
}
```

Sendo que a aplicação se conecta automaticamente ao Arduino após a sua inicialização, a página pré criada para configurações Bluetooth não foi necessária. Decidi então readaptar esta página de modo a tornar-se um fragmento descritivo.

WEBGRAFIA

https://www.youtube.com/watch?v=TLXpDY1pItQ&t=497s

https://www.youtube.com/watch?v=GwgTlxLSeZ8&t=20s

https://www.youtube.com/watch?v=TLXpDY1pItQ&t=110s

https://forum.arduino.cc/index.php?topic=567431.0

https://www.circuito.io/app?components=9442,11022,11285,11285,12022,987654

https://stackoverflow.com/questions/36119183/android-sending-data-to-arduino-via-bluetooth

https://www.youtube.com/results?search_query=turning+on+led+with+arduino+bluetooth

https://www.instructables.com/How-to-Receive-Arduino-Sensor-Data-on-Your-Android/

https://www.arduino.cc/en/software

https://www.buildcircuit.com/bluetooth-communication-between-android-and-arduino-using-free-applications/

https://www.buildcircuit.com/how-to-send-data-from-android-to-arduino/

https://www.allaboutcircuits.com/projects/communicate-with-your-arduino-through-android/

https://create.arduino.cc/projecthub/azoreanduino/simple-bluetooth-lamp-controller-using-android-and-arduino-aa2253

https://www.youtube.com/watch?v=GwgTlxLSeZ8&t=20s

https://forum.arduino.cc/index.php?topic=103066.0

https://forum.arduino.cc/index.php?topic=567431.0

https://www.youtube.com/watch?v=GwgTlxLSeZ8&t=20s

https://www.electroniclinic.com/android-app-development-to-control-arduino-over-bluetooth-using-android-studio/#Android_app_development

https://howtomechatronics.com/tutorials/arduino/arduino-and-hc-05-bluetooth-module-tutorial/

https://www.youtube.com/watch?v=JScZd7AAJwI&t=414s

https://www.arduino.cc/en/software

https://www.arduino.cc/en/donate/

https://create.arduino.cc/projecthub/infoelectorials/project-017-arduino-bh1750-light-sensor-project-640075

https://www.arduino.cc/reference/en/libraries/bh1750/

https://github.com/claws/BH1750

https://create.arduino.cc/projecthub/afsh_ad/measure-lux-with-arduino-using-bh1750-91dad1

https://www.arduino.cc/reference/en/language/functions/communication/serial/write/

https://forum.arduino.cc/index.php?topic=140299.0

https://forum.arduino.cc/index.php?topic=302961.0

https://stackoverflow.com/questions/14479349/receive-data-from-arduino-bluetooth-device

https://forum.arduino.cc/index.php?topic=645280.0

https://forum.arduino.cc/index.php?topic=264634.0

https://create.arduino.cc/projecthub/mayooghgirish/arduino-bluetooth-basic-tutorial-d8b737

https://forum.arduino.cc/index.php?topic=89401.0

https://support.arduino.cc/hc/en-us/articles/360013825259-My-sketch-seems-to-upload-successfully-but-does-nothing

https://www.filipeflop.com/blog/tutorial-arduino-bluetooth-hc-05-mestre/

https://www.botnroll.com/pt/bluetooth/2581-m-dulo-bluetooth-hc-05-para-arduino.html

https://create.arduino.cc/projecthub/electropeak/getting-started-with-hc-05-bluetooth-module-arduino-e0ca81

https://www.filipeflop.com/blog/tutorial-modulo-bluetooth-com-arduino/

https://www.instructables.com/Connecting-HC-05-Bluetooth-Module-to-Arduino/

https://ae01.alicdn.com/kf/HTB1WLLmXqLN8KJjSZFmq6AQ6XXap/Superior-mosfet-bot-o-irf520-mosfet-driver-m-dulo-3-3v-5v-pot-ncia-para-arduino.jpg_50x50.jpg_.webp

https://www.arduino.cc/reference/en/language/functions/communication/serial/write/

https://www.electrofun.pt/sensores-arduino/sensor-temperatura-humidade-dht11

https://lastminuteengineers.com/dht11-module-arduino-tutorial/

https://forum.arduino.cc/index.php?topic=89143.0

https://forum.arduino.cc/index.php?topic=596935.0

https://www.arduino.cc/en/Tutorial/BuiltInExamples/StringAdditionOperator

https://forum.arduino.cc/index.php?topic=244118.0

https://forum.arduino.cc/index.php?topic=58531.msg1912859#msg1912859

https://www.instructables.com/Arduino-and-DHT11-Temperature-Measurement/

https://github.com/adafruit/DHT-sensor-library

https://www.arduino.cc/reference/en/libraries/bh1750/

https://github.com/claws/BH1750

http://arduinolearning.com/code/arduino-bh1750-sensor.php

https://theorycircuit.com/bh1750-sensor-arduino-interfacing/

https://create.arduino.cc/projecthub/afsh_ad/measure-lux-with-arduino-using-bh1750-91dad1

https://www.arduino.cc/reference/en/libraries/dht-sensor-library/

https://arduino.stackexchange.com/questions/61407/dht-11-temp-and-hum-sensor-always-0-values

https://forum.arduino.cc/index.php?topic=277430.0

https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/

https://forum.arduino.cc/index.php?topic=507803.0

https://arduino.stackexchange.com/questions/65850/dht11-sensor-providing-999-outputs

https://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

https://www.reddit.com/r/arduino/comments/7c2foz/dht11_only_reading_at_0c_and_0_humidity/

https://stackoverflow.com/questions/40874880/getting-nan-readings-from-dht-11-sensor

https://stackoverflow.com/questions/40874880/getting-nan-readings-from-dht-11-sensor

https://forum.arduino.cc/index.php?topic=244118.0

https://forum.arduino.cc/index.php?topic=89143.0

https://forum.arduino.cc/index.php?topic=596935.0

https://forum.arduino.cc/index.php?topic=103066.0

https://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/

https://create.arduino.cc/projecthub/infoelectorials/project-017-arduino-bh1750-light-sensor-project-640075

https://www.arduino.cc/en/Tutorial/BuiltInExamples/StringAdditionOperator

https://lastminuteengineers.com/dht11-module-arduino-tutorial/

https://forum.arduino.cc/index.php?topic=596935.0

https://www.youtube.com/watch?v=TLXpDY1pItQ&t=497s

https://forum.arduino.cc/index.php?topic=567431.0

https://www.circuito.io/app?components=9442,11022,11285,11285,12022,987654

https://www.instructables.com/How-to-Receive-Arduino-Sensor-Data-on-Your-Android/

https://www.arduino.cc/en/software/

https://www.buildcircuit.com/bluetooth-communication-between-android-and-arduino-using-free-applications/

https://www.allaboutcircuits.com/projects/communicate-with-your-arduino-through-android/

https://forum.arduino.cc/index.php?topic=171320.0

https://stackoverflow.com/questions/44236642/my-hc-05-bluetooth-module-is-not-receiving-correct-data

https://create.arduino.cc/projecthub/electropeak/getting-started-with-hc-05-bluetooth-module-arduino-0ca81

https://forum.arduino.cc/index.php?topic=416704.0

https://arduino.stackexchange.com/questions/66810/my-hc-05-bluetooth-module-is-not-working

https://efundies.com/midi-controlled-solenoids-with-arduino-and-ableton-live-part-1/

https://dronebotworkshop.com/transistors-mosfets/

https://www.youtube.com/watch?v=TLXpDY1pItQ&t=497s

https://www.youtube.com/watch?v=3dDlw9A1PM8

https://www.youtube.com/watch?v=fJEFZ6EOM9o

https://gossipfunda.com/run-button-not-working-in-android-studio/

https://medium.com/@okann/android-button-needs-to-click-twice-problem-solved-f08c193c05d0

https://www.tutorialspoint.com/how-to-change-color-of-button-in-android-when-clicked

https://www.journaldev.com/19850/android-button-design-custom-round-color

https://abhiandroid.com/ui/button

https://www.geeksforgeeks.org/how-to-change-the-background-color-of-button-in-android-using-colorstatelist/

https://stackoverflow.com/questions/3310603/android-how-to-programmatically-set-button-color

https://stackoverflow.com/questions/32671004/how-to-change-the-color-of-a-button/55954610

https://stackoverflow.com/questions/15116393/fit-image-in-imagebutton-in-android

https://stackoverflow.com/questions/11688689/save-variables-after-quitting-application/11688753

http://samir-mangroliya.blogspot.com/p/android-shared-preferences.html

https://developer.android.com/training/data-storage/shared-preferences

https://stackoverflow.com/questions/43669651/how-to-keep-two-versions-of-an-app-installed

https://medium.com/swlh/managing-multiple-apps-within-a-single-android-studio-project-659eb33f5f0e

https://stackoverflow.com/questions/4834750/how-to-get-the-selected-item-from-listview

https://www.youtube.com/watch?v=jxNdBy-LIqc&t=152s

https://www.youtube.com/watch?v=iFtjox9_zAI&t=775s

https://www.youtube.com/watch?v=wLRQ9EClYuA&t=1028s

https://android.gadgethacks.com/how-to/change-your-home-screen-icon-shapes-android-10-0194758/

https://www.youtube.com/watch?v=OCceWupZ_Ik

https://stackoverflow.com/questions/12128331/how-to-change-fontfamily-of-textview-in-android

https://stackoverflow.com/questions/1748977/making-textview-scrollable-on-android

https://www.youtube.com/results?search_query=linear+layout+tutorial+android+studio

https://stackoverflow.com/questions/4127725/how-can-i-remove-a-button-or-make-it-invisible-in-android

https://stackoverflow.com/questions/9280981/convert-float-to-charsequence-in-android

https://developer.android.com/guide/topics/resources/runtime-changes

https://stackoverflow.com/questions/3723823/i-want-my-android-application-to-be-only-run-in-portrait-mode

https://stackoverflow.com/questions/21473509/android-how-to-close-an-activity-on-button-click/21473578

https://developer.android.com/studio/run/oem-usb

https://www.youtube.com/results?search_query=change+image+android+studio+

https://www.youtube.com/watch?v=EMxYdhoRqlg

https://www.c-sharpcorner.com/UploadFile/1e5156/how-to-set-image-in-a-image-view-on-click/

https://stackoverflow.com/questions/5223656/how-can-i-change-the-image-of-an-imageview-on-android/5223718

https://software.intel.com/content/www/us/en/develop/articles/intel-usb-driver-for-android-devices.html

https://stackoverflow.com/questions/27697018/where-to-find-or-download-usb-driver-folder

https://stackoverflow.com/questions/15393899/how-to-close-activity-and-go-back-to-previous-activity-in-android

https://stackoverflow.com/questions/38987354/android-stop-onpause-from-being-called-in-parent-class

https://developer.android.com/studio/write/create-java-class

https://developer.android.com/reference/android/app/Activity

https://stackoverflow.com/questions/16612548/project-in-android-studio-wont-start-activity-class-not-specified

https://stackoverflow.com/questions/17373176/overridependingtransition-when-launching-an-intent-via-preferences

https://stackoverflow.com/questions/14671897/super-oncreatesavedinstancestate

https://www.studioks.net/en-androidviewbackorfront/

https://stackoverflow.com/questions/35882755/android-custom-view-move-back-to-original-position-when-refreshing

https://stackoverflow.com/questions/3496269/how-do-i-put-a-border-around-an-android-textview

https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/connectivity/bluetooth.html

https://stuff.mit.edu/afs/sipb/project/android/docs/reference/android/bluetooth/BluetoothSocket.html

https://www.tutorialspoint.com/android/android_bluetooth.htm

http://www.londatiga.net/it/programming/android/how-to-programmatically-pair-or-unpair-android-bluetooth-device/

https://www.youtube.com/results?search_query=conect+to+a+bluetooth+device+android+studio

https://www.youtube.com/watch?v=iFtjox9_zAI

https://www.youtube.com/results?search_query=android+studio+bluetooth+spinner

https://www.youtube.com/results?search_query=android+studio+bluetooth+slider

https://developer.android.com/reference/com/google/android/material/slider/Slider

https://developer.android.com/studio/write/image-asset-studio

https://stackoverflow.com/questions/33820260/how-to-delete-corresponding-mipmap-icon-images-all-at-once

https://developer.android.com/codelabs/basic-android-kotlin-training-birthday-card-app-image#2

https://www.youtube.com/watch?v=JjfSjMs0ImQ&list=PL1LKvysk2fdhG84shdg6LVBI0_zbMkKsP&index=13&t=1s

https://www.youtube.com/watch?v=NXXeb0ORocE

https://stackoverflow.com/questions/16632428/android-change-value-in-strings-xml

https://www.youtube.com/watch?v=Xju34v1qEDY

https://developer.android.com/guide/topics/ui/controls/radiobutton

https://developer.android.com/training/data-storage/room

https://developer.android.com/training/basics/network-ops/xml

https://stackoverflow.com/questions/33450657/how-to-add-icon-image-button-in-android-studio

https://www.youtube.com/watch?v=510WE8CmiXI

CONCLUSÃO

Este projeto proporcionou-me uma enorme aprendizagem, tanto de conhecimentos teóricos sobre diversas matérias que fui obrigado a pesquisar e aprender, como também a nível de competências técnicas, com a exigência de manipulação dos componentes eletrónicos e mecânicos com os quais tive de lidar.

Fui capaz, através deste processo, de aprofundar bastante os meus conhecimentos relativos à criação de circuitos elétricos, e à programação dos mesmos. Aprofundei ainda bastantes conhecimentos nas linguagens de programação Java e C/C++, uma vez que os conteúdos lecionados durante o decorrer do curso não foram suficientes para a realização do projeto.

Todos os conhecimentos que tenho acerca da programação Android foram também adquiridos durante a realização deste projeto, porque nunca tinha tido nenhum tipo de contacto com este sistema. Consequentemente, aprendi também como utilizar e manipular a linguagem XML, que esta foi utilizada para criação de toda a estrutura visual da minha aplicação.

Este projeto permitiu-me também aprender mais sobre mim como indivíduo, e sobre o mundo que me rodeia. Sabia da dificuldade que seria finalizar um projeto desta dimensão, sabendo dos conhecimentos e perícias que detinha. No entanto nunca desviei desse objetivo final. Levou-me a níveis de pressão extremos e situações de stress, que nunca tinha sentido. E como o que não nos mata torna-nos mais fortes, foi precisamente isso que aconteceu. Agora sei que não há impossíveis e como reagir em determinadas situações menos boas. Foi também uma experiência muito enriquecedora como ser humano, conheci limites, estabeleci relações pessoais, aprendi a gerir tempos, prioridades e fundamentalmente a importância de um bom planeamento.

Estou mais FORTE.