Projeto em Engenharia de Automação

Departamento de Engenharia Mecânica

ClassQuiz Plus - Sistema de Inquérito e Avaliação Rápida em Sala de Aula

Orientadores:

Prof. Vítor Santos

Prof. Pedro Fonseca

Autor:

Manuel Mamede

89070

MEAI

Índice

Índice de Figuras

1. Introdução

Nos sistemas de avaliação pedagógica modernos é necessário um envolvimento continuo dos estudantes, o que implica uma avaliação continua. Em certas aulas, como teóricas e teórico-práticas, a realização de pequenas avaliações ao longo do ano letivo pode tornar-se num grande desafio dado o elevado número de alunos neste tipo de aulas. Assim, é necessário um método que permita uma avaliação rápida e com resultados imediatos.

Um método que permite alcançar uma avaliação rápida é a realização de um teste de escolha múltipla projetado diretamente para todos os alunos, evitando dessa forma o recurso a respostas escritas. Uma solução passa por um terminal individual que permita uma resposta direta a cada pergunta; este dispositivo estará ligado a uma rede *wireless* que transmite diretamente a resposta para um computador com *software* capaz de interpretar e armazenar toda essa informação.

Este problema já foi abordado num projeto anterior [1] tendo por base a comunicação rádio. Esse projeto serviu para provar o conceito, mas carece de desenvolvimento para evoluir para um sistema mais fiável e simples.

A solução apresentada neste relatório consiste num terminal desenvolvido com base num NodeMCU (semelhante a um Arduino) com comunicação Wi-Fi e num computador a correr *software* que funciona como estação-base para a receção das mensagens do terminal. A função do terminal é enviar as respostas. É composto por vários botões, que o aluno prime de modo a escolher uma resposta.

2. Objetivos

Na figura 1 está ilustrado o princípio para o funcionamento deste sistema. O terminal terá uma interface física, botões, e um módulo que permita autenticação, RFID. O terminal comunica com a estação-base e, se necessário, a estação com o terminal.

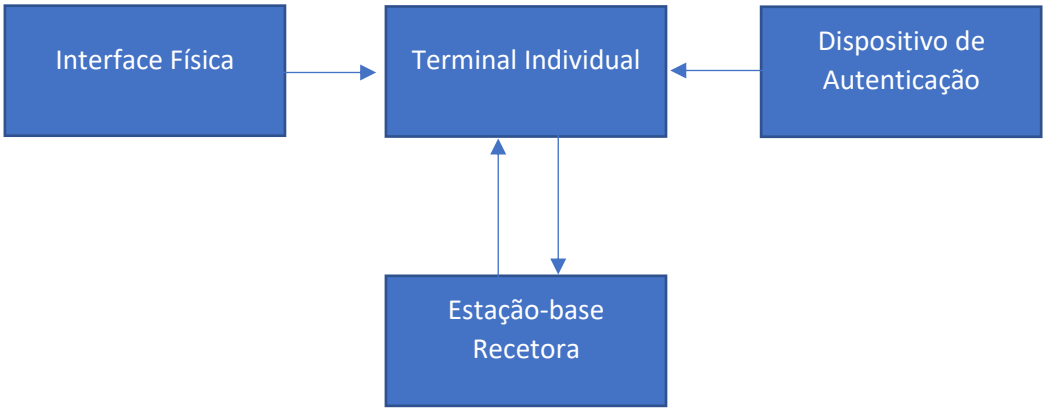

Figura 1 - Diagrama de Blocos do Sistema a Implementar

Este projeto tem como principais objetivos:

- Escolha do modo de comunicação entre o terminal individual e a estação-base recetora;
- Escolha do método de autenticação físico, que permita verificar se o utilizar pode aceder ao terminal;
- Escolha do *hardware* a utilizar;
- Escolha do método de interface com o terminal;
- Programação do modo de comunicação do terminal individual;
- Programação do modo de autenticação do aluno;
- Programação da estação-base recetora;
- Elaboração do circuito elétrico do terminal;
- Montagem do protótipo.

3. Planificação do projeto

3.1 Fases de desenvolvimento

O plano de trabalho foi dividido em 3 fases, com complexidade crescente. As fases não completadas podem, eventualmente, ser aplicadas num projeto baseado neste, uma continuação, ou até numa tese que desenvolva este sistema até estar preparado para a produção em série e comercialização.

1ª Fase:

• Estabelecer o *hardware* para o terminal individual;

• Definição do protocolo de comunicação entre terminal e estação-base recetora;

• Estabelecimento de uma comunicação entre o terminal e a estação-base;

• Adicionar uma interface física à comunicação do terminal, com recurso a botões e LED como forma de atuação e *feedback*, respetivamente;

• Desenvolvimento de uma interface gráfica simples para a estação-base recetora.

A primeira fase é a base para um projeto funcional onde terão de ser tomadas todas as decisões acerca de *software* e *hardware,* bem como a definição dos protocolos de comunicação.

2º Fase:

• Acrescentar autenticação à mensagem que o terminal individual envia para a estação-base;

• Restringir a utilização do terminal ao espaço do evento e aos indivíduos autorizados;

• Desenvolvimento de um protótipo;

• Teste com um programa que imite um possível *quiz* de escolha múltipla.

A segunda fase foca-se no desenvolvimento de um terminal físico pronto a usar com uma interface simples, assim como o desenvolvimento de um programa que gere as respostas já de uma forma adequada para o proposto.

3º Fase:

• Encriptação de dados para evitar interseções de comunicações ou tentativas de sabotagem;

• Comunicação com vários terminais e uma estação-base;

• Criação de uma base de dados para reconhecimento do aluno.

A terceira fase é para um projeto mais avançado e próximo de uma solução real e, por isso, não chegou a ser completada. Exige conhecimentos mais específicos, nomeadamente na criação de bases de dados *MySQL* e encriptação de comunicações.

4. Modo de comunicação

Num projeto anterior foi aplicada a comunicação por sinais de rádio [1], esse projeto provou-se funcional e adequado para o objetivo, no entanto a quantidade de ruído nos sinais, entre outras limitações, não tornaram o sistema fiável. Foi assim necessário explorar outros métodos de comunicação.

O método adotado para a comunicação entre terminal e estação-base foi o Wi-Fi, é um método eficaz e já estabelecido em todo o sistema de ensino, tornando-o acessível. É uma comunicação de baixa potência e praticamente imune a ruído, mas poderá haver algumas perdas de dados ou atrasos entre o envio e a receção.

O Wi-Fi recorre ao *Internet Protocol* (IP), e dentro deste podem se estabelecer comunicações por UDP ou TCP/IP. Para este projeto optou-se pelo UDP (*User Datagram Protocol*).

O TCP (*Transmission Control Protocol*) é o protocolo mais usual e assegura sempre que a mensagem chega ao destino e que, se for divida em vários pacotes, estes são ordenados no recetor.

O UDP é um protocolo mais simples e não garante que a mensagem chegue ao destinatário; qualquer mensagem perdida deixa simplesmente de existir, para o destinatário. Como neste projeto apenas serão realizados testes com um único terminal, foi adotado o protocolo UDP de modo a ter uma comunicação simples de implementar.

Para comunicar via UDP ou TCP/IP é necessário estabelecer um IP local e remoto (de onde se envia e para onde se envia). No caso do UDP é necessário atribuir uma porta. O IP normalmente é atribuído pelo *router* ao estabelecer a ligação, via DHCP como acontece com os dispositivos comuns (computar ou smartphone), mas a porta UDP é definida pelo utilizador.
Como este projeto terá, eventualmente, de suportar múltiplos terminais ligados, o que implica múltiplos endereços IP e portas, é necessário implementar um método que permita a atribuição

destes, de forma controlada. A porta UDP pode ser gerada aleatoriamente a cada sessão, tendo o cuidado de não repetir portas.

Uma solução poderá ser utilizando DHCP (*Dynamic Host Configuration Protocol*), que é um protocolo de serviço TCP/IP que oferece configuração dinâmica de terminais, com concessão de endereços IP de *host* (terminais), máscara de sub-rede, *default gateway*, IP de um ou mais servidores DNS, entre outros.

Com o acréscimo de terminais, o *router* pode não ser capaz de gerir a quantidade de informação a ser enviada, ou seja, o nó de rede entra em sobre carga. Pode ser necessário, em aulas de grande afluência, ter vários AP (Access Points). O *access point* é um dispositivo que permite interligar uma rede a vários dispositivos, ou seja, criar novos pontos de acesso à rede.

5. Modo de funcionamento do terminal

5.1 Escolha do *Hardware*

O terminal será composto por um microcontrolador com interface física composta por botões e LED. Para a autenticação torna-se prático utilizar o próprio cartão de aluno como método de validação, ao invés de se criar novos cartões ou *tags* para o efeito. Estes cartões são *Mifare* e funcionam por RFID (*Radio-Frequency Identification*).

Para o terminal individual será necessário um microcontrolador de dimensões reduzidas capaz de gerir a comunicação Wi-Fi; para isso foi utilizado o microcontrolador *Lolin NodeMCU V3* [Figura 2]. O *NodeMCU* é uma plataforma *open source*. Inclui firmware que corre no ESP8266 e hardware baseado no módulo ESP-12. O termo *NodeMCU* refere-se ao *firmware* e não ao kit de desenvolvimento, este *firmware* baseia-se na linguagem de *script Lua*. Neste projeto a programação foi feita em Arduino IDE.

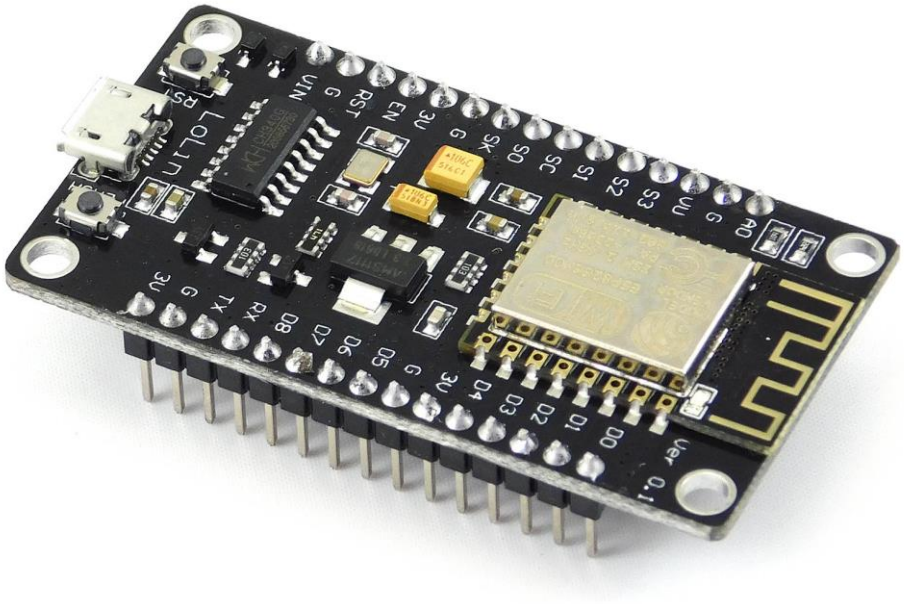

Figura 2- Microcontrolador Lolin NodeMCU V3 com Módulo Wi-Fi ESP8266 integrado

Este é um microcontrolador compacto, mas com capacidades semelhantes a um Arduino UNO. Possui 128kBytes de memória e 4MBytes de memória de armazenamento. Para além disso, possui já integrado um módulo Wi-Fi ESP8266. É um microcontrolador acessível em termos de preço e uso, e existem disponíveis vários projetos, bibliotecas e tutoriais que tornam a sua

utilização simples de aprender [8]. O *chip* ESP8266 segue a norma Wi-Fi IEEE 802.11 b / g / n com, capacidade para redes de 2.4 GHz.

O *NodeMCU* possui 9 entradas/saídas digitais e uma entrada analógica, aceita uma tensão de entrada de 5 até 20V e consome 800mA. Possui 3 saídas de 3.3V que podem ser usadas para alimentar dispositivos externos. Para programar e alimentar o dispositivo recorre-se à ligação micro-USB. Estas entradas/saídas podem ser visualizadas na figura 3.

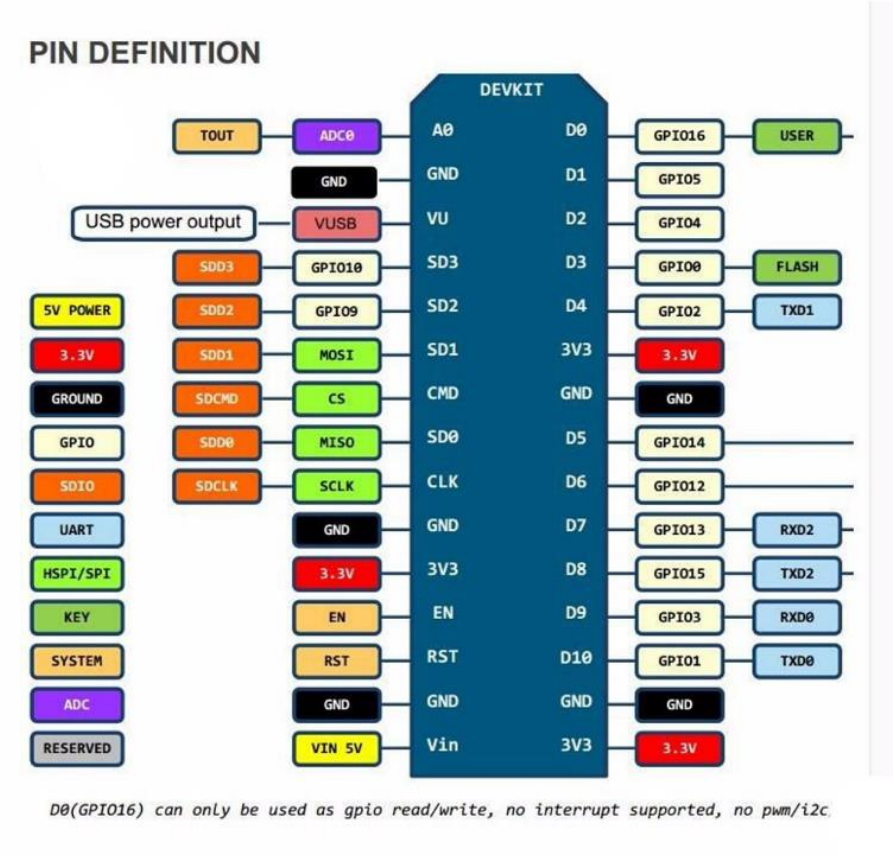

Figura 3- PinOut do NodeMCU

Para a autenticação será necessário um leitor de cartões RFID compatível com o *NodeMCU,* foi utilizado o *Mifare* RFID – RC522 [Figura 4]. Este módulo comunica com cartões a distâncias de até 1cm através de um campo magnético de 13.56MHz, e envia para o microcontrolador os dados através de comunicação SPI (*Serial Peripheral Interface*).

Figura 4- Módulo Mifare RFID RC522

O leitor *Mifare* é um dos mais utilizados para projetos em Arduino e similares [7], e é bastante compacto e simples de utilizar. Possui uma vasta documentação, exemplos e bibliotecas disponíveis online. Este cartão permite ainda ler e escrever em *data blocks* (secções dos cartões *Mifare* onde se pode escrever informações), com isso uma solução possível para a autenticação e atribuição de IP's poderia ser escrever num *data block* o endereço IP a atribuir e uma chave (palavra/*keyword*) que confirme que o cartão foi validado. [5]

Para a interface física serão utilizados 5 botões de pressão de 6x6mm e 4 pinos [Figura 5]. É premindo estes botões que o utilizador pode enviar a sua resposta.

Figura 5 - Push Button de 6x6mm Utilizado na Interface do Terminal

Como *feedback* para as ações que estão a decorrer no terminal utiliza-se um LED vermelho e um verde de 5mm [Figura 6].

Figura 6 - LED's de 5mm para o Feedback do Terminal

Para os botões foram utilizadas resistências de pulldown de 10kΩ, que levam a entrada digital, a que está conectado o botão, a zero (LOW) quando o botão não está premido.

Para os LED's foram usadas resistências de 510Ω de modo a ter uma boa intensidade do LED.

De modo a ser possível utilizar os 5 botões, 2 LED's e as 5 saídas do RFID, são necessárias 12 entradas/saídas no *NodeMCU*. Como este apenas possui 9, a solução é utilizar um *encoder* de 8:3 para codificar os 5 botões em apenas 3 entradas digitais usando também a entrada analógica. O *encoder* escolhido foi o *Philips* HEF4532BP [Figura 7]. Este é um *encoder* CMOS prioritário de 8 *bits*, ou seja os pinos de entrada de número maior têm prioridade sobre os de número menor. A tecnologia deste *encoder* permite-lhe ser alimentado diretamente com os 3.3V fornecidos pelo microcontrolador.

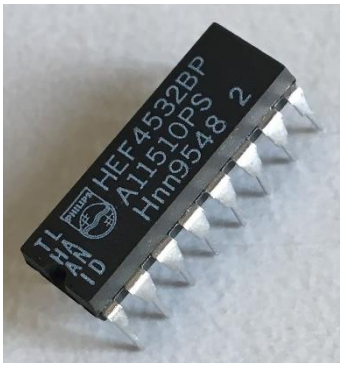

Figura 7 - Encoder CMOS Philips HEF4532BP

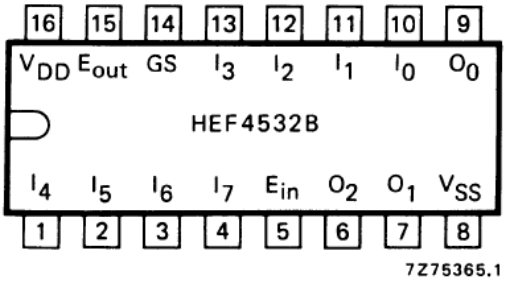

Figura 8 - PinOut do Encoder

Os pinos do *encoder* podem ser visualizados na figura 8.

Para os botões foram utilizados os pinos de D1 a D5, não foi possível utilizar o D0 pois sem outra entrada analógica para o pino Gs não seria possível distinguir o código de D0 com o de *encoder* desligado, como é possível ver na tabela de verdade da figura 10. Na figura 9 está descrito o significado de cada pino do *encoder*.

PINNING

I_0 to I_7	priority inputs
E_{in}	enable input
E_{out}	enable output
GS	group select output
O_0 to O_2	outputs

Figura 9 - Pinning do Encoder CMOS

TRUTH TABLE

INPUTS									OUTPUTS				
E_{in}	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	GS	O_2	O_1	O_0	E_{out}
L	X	X	X	X	X	X	X	X	L	L	L	L	L
H	L	L	L	L	L	L	L	L	L	L	L	L	H
H	H	X	X	X	X	X	X	X	H	H	H	H	L
H	L	H	X	X	X	X	X	X	H	H	H	L	L
H	L	L	H	X	X	X	X	X	H	H	L	H	L
H	L	L	L	H	X	X	X	X	H	H	L	L	L
H	L	L	L	L	H	X	X	X	H	L	H	H	L
H	L	L	L	L	L	H	X	X	H	L	H	L	L
H	L	L	L	L	L	L	H	X	H	L	L	H	L
H	L	L	L	L	L	L	L	H	H	L	L	L	L

Notes

1. H = HIGH state (the more positive voltage)
2. L = LOW state (the less positive voltage)
3. X = state is immaterial

Figura 10 - Tabela de Verdade do Encoder Philips

Finalmente para que tudo isto funcione é necessário um *router* Wi-Fi ao qual os terminais e a estação-base estarão ligados.

5.2 Interface

A figura 11 mostra o diagrama de blocos do *hardware* do terminal individual.

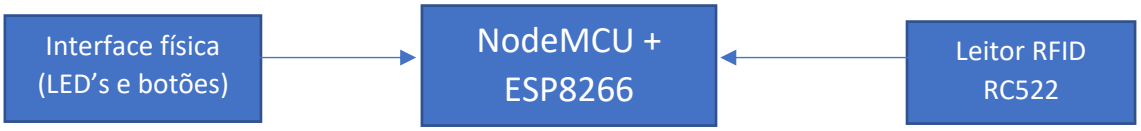

Figura 11 - Diagrama de Blocos do Terminal

A seguir estão descritos todos os passos para a utilização do sistema.

1. Ativação do cartão de aluno para a sessão, ou seja, passar o cartão por um leitor RFID, à entrada da sala. Deste modo é restringida a utilização apenas aos alunos dentro da sala;
2. Ligação do terminal individual;
3. Inserção do cartão de aluno no terminal;
4. Adesão do terminal à rede local Wi-Fi, o LED verde permanecerá acesso até estabelecer a ligação;
5. Verificação, com a estação-base, do cartão. Caso não esteja autorizado o LED vermelho acende e o terminal desliga a conexão à rede. Se o cartão for válido o terminal permanece conectado;
6. Carregar no botão pretendido (botão 1 a 5). O LED verde acenderá se a informação tiver sido enviada para a estação-base;
7. Desligar o terminal individual.

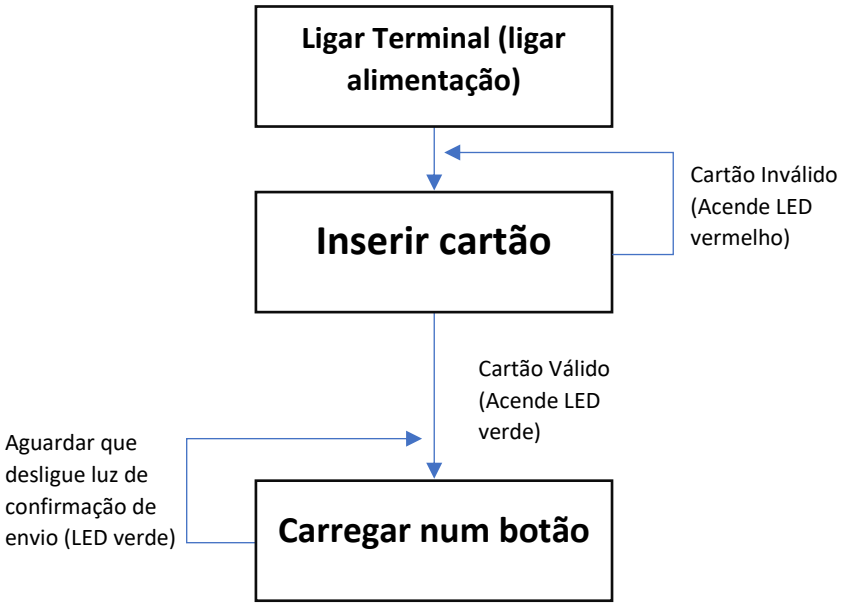

Figura 12 - Fluxograma de Funcionamento do Terminal

Na figura 12 está representado o fluxograma simplificado dos passos descritos no parágrafo anterior.

A validação à entrada pode ser feita recorrendo a um leitor de cartões do género da figura 13. Este dispositivo pode ler o UID e enviar para o computador, este valida o cartão na base de dados de alunos autorizados a aceder ao terminal. Este tipo de dispositivos pode ser integrado com, por exemplo, uma aplicação em *Visual Basic*.

Figura 13 - Leitor de Cartões RFID HID OMNIKEY 5321

5.3 Circuito do terminal

Recorrendo ao *software Eagle* foi desenhado o circuito eletrónico do Terminal.

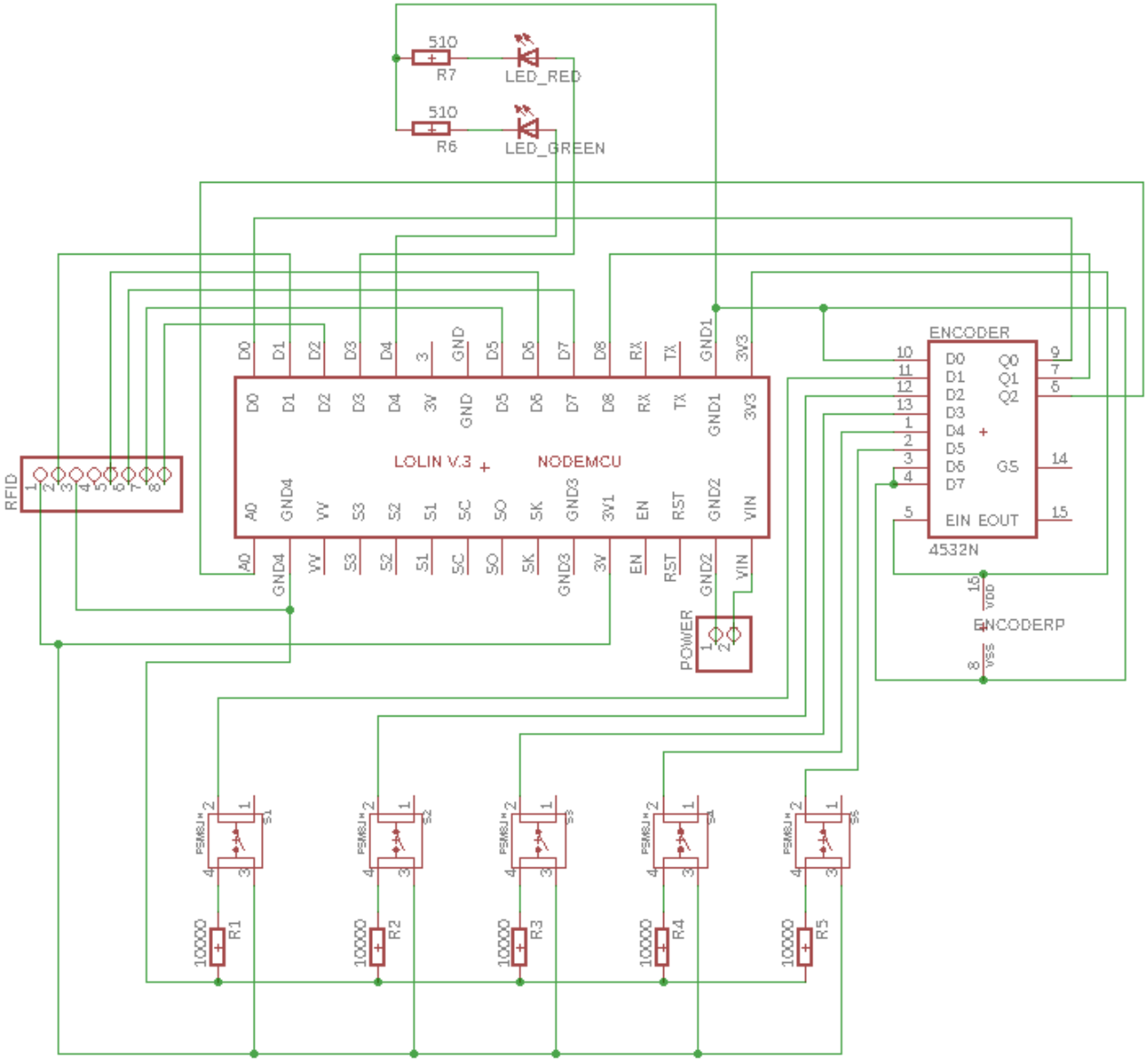

Figura 14 - Esquema de ligações do Microcontrolador NodeMCU

Na figura 14 estão representadas todas as ligações do circuito. As ligações do RFID foram feitas como recomendado na biblioteca do RFID RC522.

A partir deste circuito foi fabricada uma placa PCB com 2 *layers* [Figura 15] [Figura 16] de modo a reduzir as dimensões.

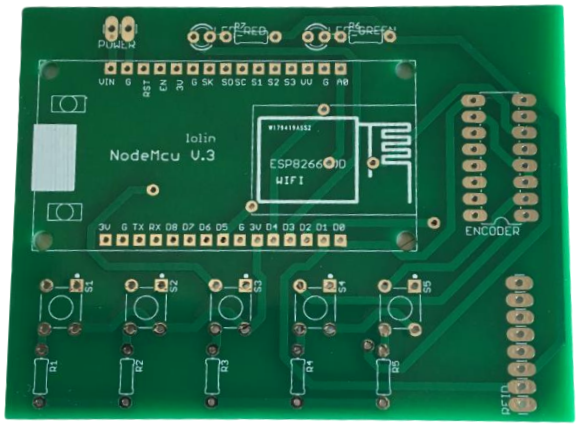

Figura 15 - Frente da Placa PCB

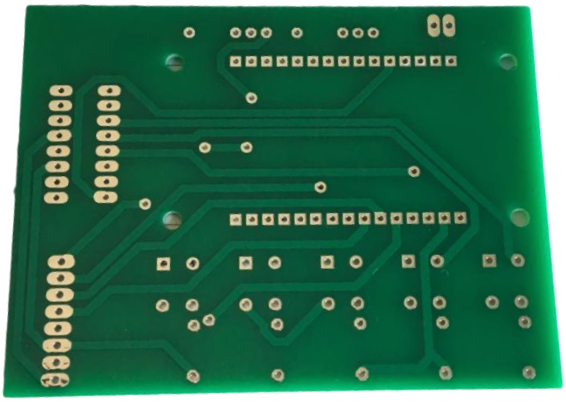

Figura 16 - Verso da Placa PCB

Estando fabricada a placa, foram soldados todos os componentes e foi montado o circuito eletrónico do terminal [Figura 17].

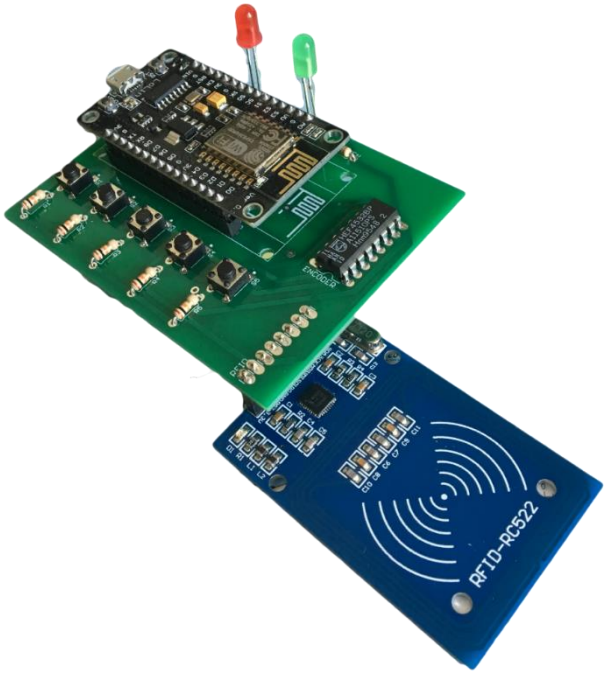

Figura 17 - Circuito Final do Terminal

5.4 Software do terminal

Toda a programação do terminal foi feita em Arduino IDE, com as entradas/saídas do NodeMCU ligadas como na tabela da figura 18.

Entrada/Saída *NodeMCU*	**Função**
D0	*Encoder* - Q0
D1	RFID - RST
D2	RFID - SDA
D3	LED Vermelho
D4	LED Verde
D5	RFID - SCK
D6	RFID - MISO
D7	RFID - MOSI
D8	*Encoder* - Q1
A0	*Encoder* - Q2

Figura 18 - Tabela de Ligações ao NodeMCU

Foi determinado que os pinos D3 e D4 não podiam ser utilizados como entradas digitais pois estão associados à comunicação e *flash* do *NodeMCU*, daí serem utilizados apenas como saídas para os LED's. Estes pinos estão normalmente a 3.3V, e ao usar como entrada estes pinos poderão ser forçados a zero durante o processo de *upload* ou *reset* do programa. Enquanto que como saída durante o *upload* ou *reset* os pinos podem se manter a 3.3V. Ao fazer o *upload* do programa para o *NodeMCU* observa-se que ambos os LED's acendem por este mesmo motivo.

O módulo RFID RC522 possui comunicação SPI e I2C, mas todos os pinos têm de estar ligados mesmo que apenas se use um dos modos de comunicação. Neste caso o método utilizado foi SPI. Para utilizar este módulo foram usadas as seguintes bibliotecas:

$$\#include\ <SPI.h>$$

$$\#include\ <MFRC522.h> [2]$$

Para iniciar a leitura faz-se:

$$SPI.begin();// Initiate\ \ SPI\ bus$$

$$mfrc522.PCD\_Init();$$

A partir daqui espera um cartão e lê a sua UID (*Unique Identifier).* Os cartões de aluno da Universidade de Aveiro testados possuem uma UID única de 7 números hexadecimais, que se convertem em 7x4 bits. (UID de cartões *Mifare* é sempre representado em hexadecimal).

Foi explorada ainda a possibilidade de escrever e ler do cartão informações e, embora o módulo o permita, é necessária uma chave de escrita para cartões que não possuam essa chave com a codificação de fábrica, 0xFF. Com recurso às funções da biblioteca RFID foi possível escrever num *data block* livre de um cartão com codificação de fábrica, assim como ler essa mesma informação. De notar que para ler de um cartão é necessário saber o *data block* onde se encontra. O *software* em anexo com este relatório, aceita apenas os cartões autorizados numa lista dentro do programa, numa aplicação real, a UID era verificada através de uma base de dados na estação-recetora e só depois de receber a autorização é que o terminal ficaria ativo.

Para a ligação Wi-Fi, através do módulo ESP8266, incluiu-se a seguinte biblioteca:

$$\#include\ < ESP8266WiFi.h >$$

E para a comunicação UDP:

$$\#include\ < WiFiUdp.h >$$

A definição do IP do terminal foi feita manualmente, visto que os testes foram feitos apenas entre um computador e um terminal foi possível manter o IP fixo em cada dispositivo. A porta UDP não depende do IP usado e essa manteve se constante ao longo de todos os testes.

Numa aplicação real o método mais eficaz de atribuição de IP's seria através de DHCP, o *NodeMCU* suporta esta negociação com recurso a um módulo *wifi.ap.dhcp* [9] ou outra biblioteca compatível. Isto permitia a ligação de múltiplos terminais, onde cada um negociava o seu IP.

Numa tentativa de evitar os endereços IP foi criado um mDNS (*Multicast Domain Name System*) para que seja possível obter o endereço IP a partir do nome de um dispositivo na rede local. O mDNS é usada em redes locais sem servidor DNS. A partir da linha de comandos do *Windows* é possível resolver o nome num enderenço IP como na figura 19. [6]

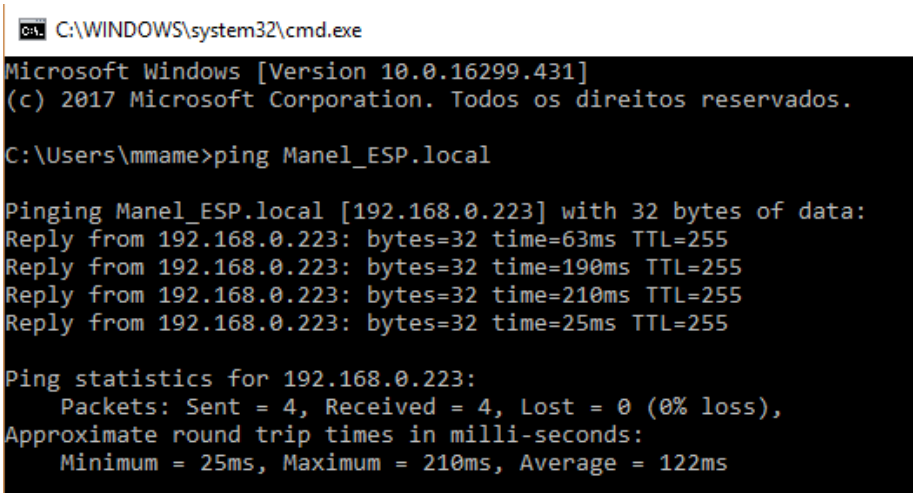

Figura 19 - Obtenção do Endereço IP através de mDNS

5.4.1 Mensagens

As mensagens enviadas pelo terminal são do tipo: “bt2 07 62 79 33 23 65 77”; onde “bt2” corresponde ao botão premido e os restantes dígitos são a UID do cartão inserido. Para o software isto corresponde a um pacote de dados de 8x24 bits por mensagem. [3]

Os dados são enviados como *string,* da seguinte forma:

$$char\ combinedArray[sizeof(buttonPacket2) + sizeof(UID) + 1];$$
$$sprintf(combinedArray, "\%s\ \%s", buttonPacket2, UID);$$
$$Udp.beginPacket(ip, remotePort);$$
$$Udp.write(combinedArray);$$
$$Udp.endPacket();$$

Esta ação é repetida cada vez que um botão é atuado.

5.5 Caixa do Terminal Individual

Após o circuito estar montado numa placa PCB recorreu-se ao *software Autodesk Inventor* para desenhar uma caixa para o terminal.

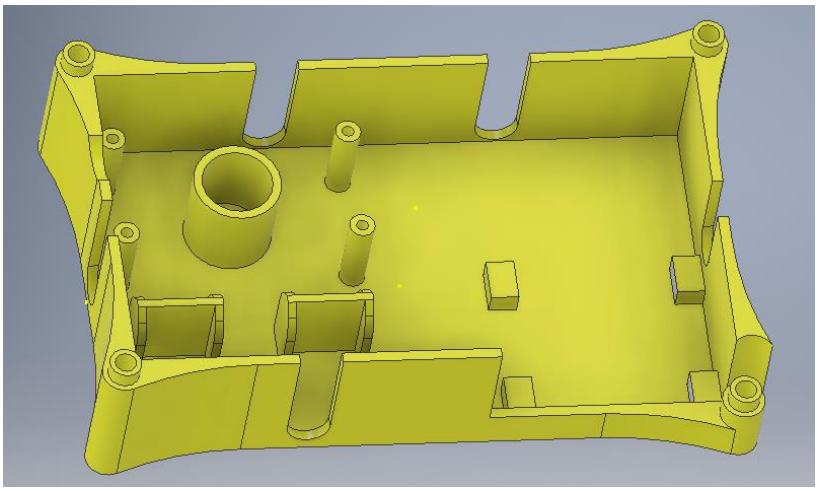

Figura 20 - Fundo da caixa

A figura 20 mostra o fundo da caixa com o suporte para a PCB, um espaço para o leitor de cartões, uma ranhura para inserir o cartão e um espaço para uma pequena bateria ou pilha.

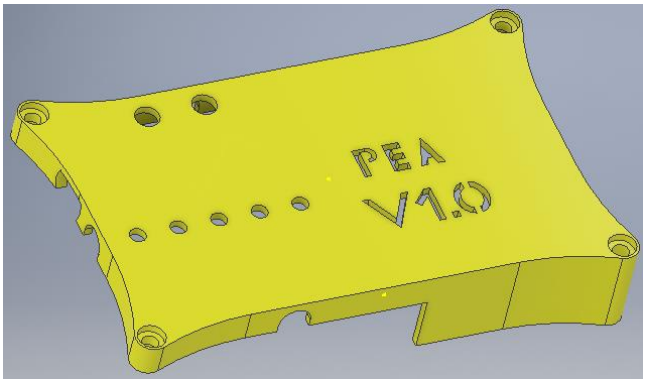

Figura 21 - Topo da caixa

A figura 21 mostra a tampa da caixa com furos para os LED e para os botões.

Figura 22 - Extensor dos Botões

A figura 22 apresenta uma extensão para os botões, pois é necessário que do topo da caixa se consiga premir os botões.

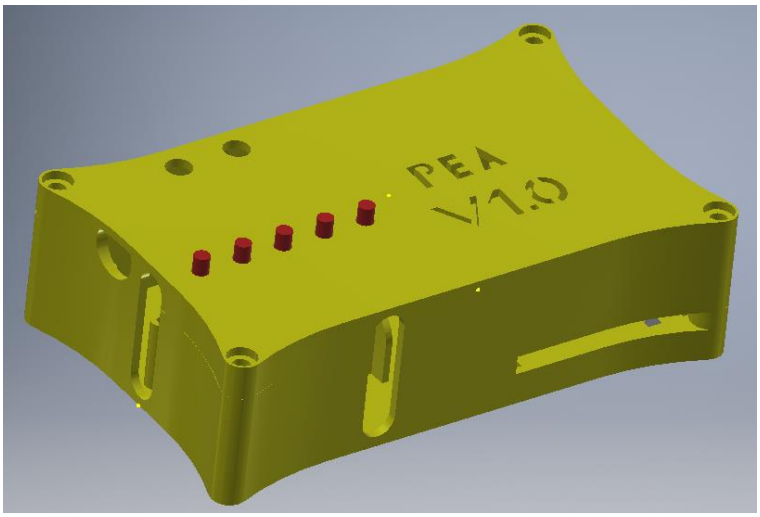

Figura 23 - Caixa Montada

A figura 23 mostra o modelo 3D da caixa fechada com um furo circular lateral de modo a ser possível ligar um cabo micro USB ou premir os botões de reset e flash do NodeMCU. Nas figuras 24 e 25 é possível ver a caixa final.

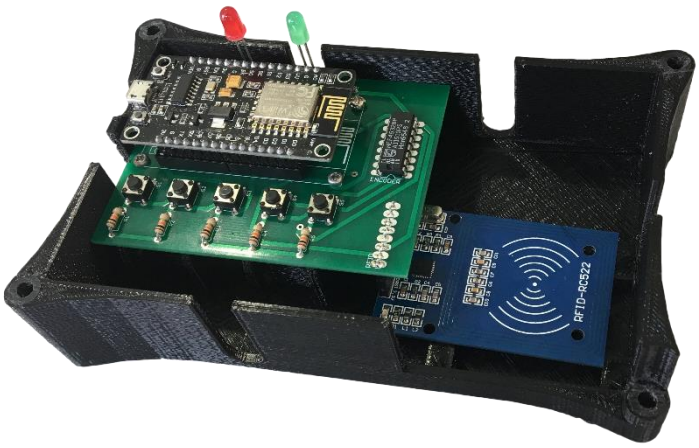

Figura 24 - Caixa Real com o Circuito Eletrónico Montado

Figura 25 - Caixa Real Fechada

6. Modo de funcionamento da estação-base recetora

6.1 *Hardware* da estação-base

Neste projeto como o modo de comunicação usado é o Wi-Fi, é possível usar um computador, sem qualquer dispositivo extra, como o recetor. A comunicação será gerida apenas por software, e para isso foi utilizado o *Visual Basic.*

6.2 *Software* da estação-base recetora

Para gerir as mensagens do terminal individual foi criada uma aplicação em *Visual Basic* capaz de interpretar e registar as mensagens.

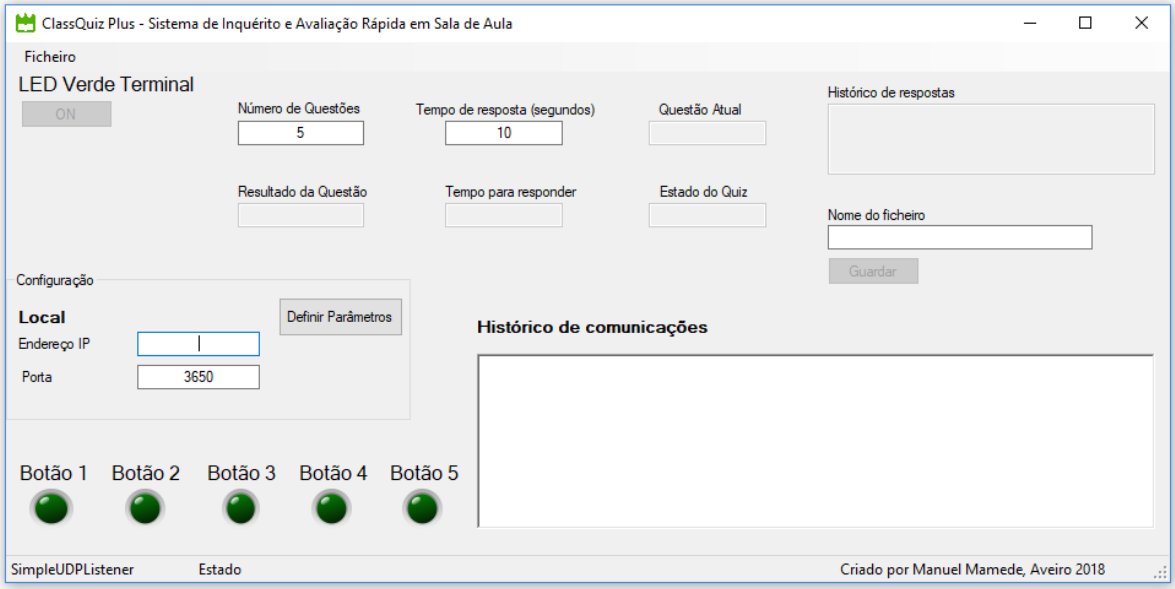

Figura 26 - Aplicação em VB do ClassQuiz

Ao iniciar a aplicação é necessário definir os parâmetros [Figura 26], o endereço IP é atualizado automaticamente com o IP local do computador e a porta é definida pelo utilizar. No menu "Ficheiro" existe a opção "Iniciar" e a opção "Sair". Ao clicar em "Iniciar" é iniciado o temporizador para um teste de escolha múltipla com os parâmetros definidos em "Número de Questões" e "Tempo de Resposta". Após terminar o tempo para todas as questões é apresentado o histórico das respostas validadas durante esse tempo e é desbloqueada a opção

"Guardar" abaixo de "Nome do Ficheiro" que permite guardar o histórico de respostas num ficheiro ".txt" com o nome inserido na caixa de texto.

A aplicação aceita para resposta à questão atual o valor do último botão premido durante o tempo de resposta. Quando um botão é premido o corresponde acende no canto inferior esquerdo da aplicação e apaga-se ao fim de meio segundo.

Numa fase inicial da programação desta aplicação ao iniciar um servidor UDP não era possível realizar nenhuma outra função na aplicação, como apresentar resultas ou clicar sobre o botão de sair. A aplicação ficava sobrecarregada com o servidor, e enquanto corria era como se o programa deixasse de responder, de modo a resolver esse problema foi utilizado um *BackgroundWorker.* A classe *BackgroundWorker* permite executar uma operação num *thread* dedicado separado, ou seja, pode-se executar o servidor UDP, sempre à escuta de pacotes de dados, e outras operações da aplicação ao mesmo tempo. Como esta aplicação é apenas de demonstração esta solução foi eficaz, mas numa aplicação real com centenas de comunicações simultâneas este problema poderá persistir. [4]

6.3 Funcionamento da aplicação

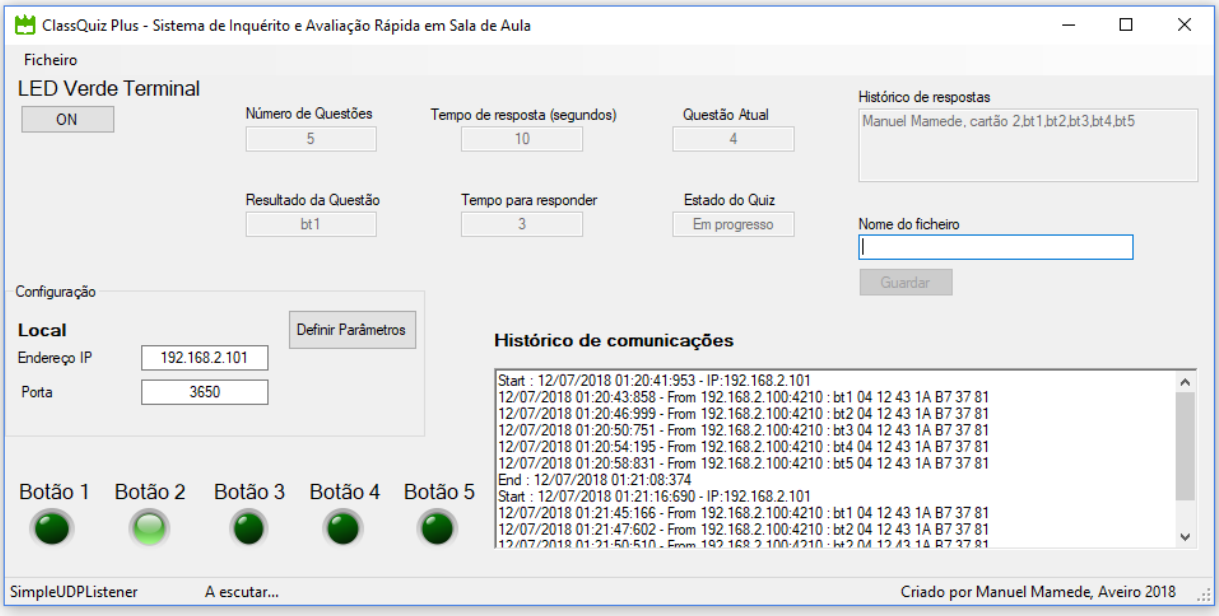

Figura 27 - Aplicação em Funcionamento

Durante o funcionamento [Figura 27] é possível observar o temporizador a decrescer e as questões a avançarem, no histórico estão registadas todas as comunicações.

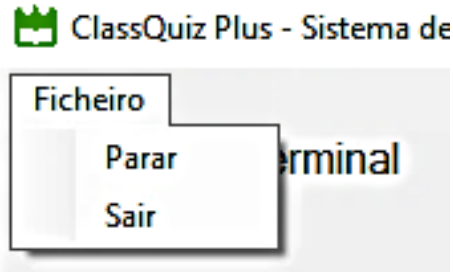

Figura 28 - Menu "Ficheiro" da Aplicação VB Durante o Funcionamento

Através do menu da figura 28, é possível iniciar e parar a execução e sair da aplicação.

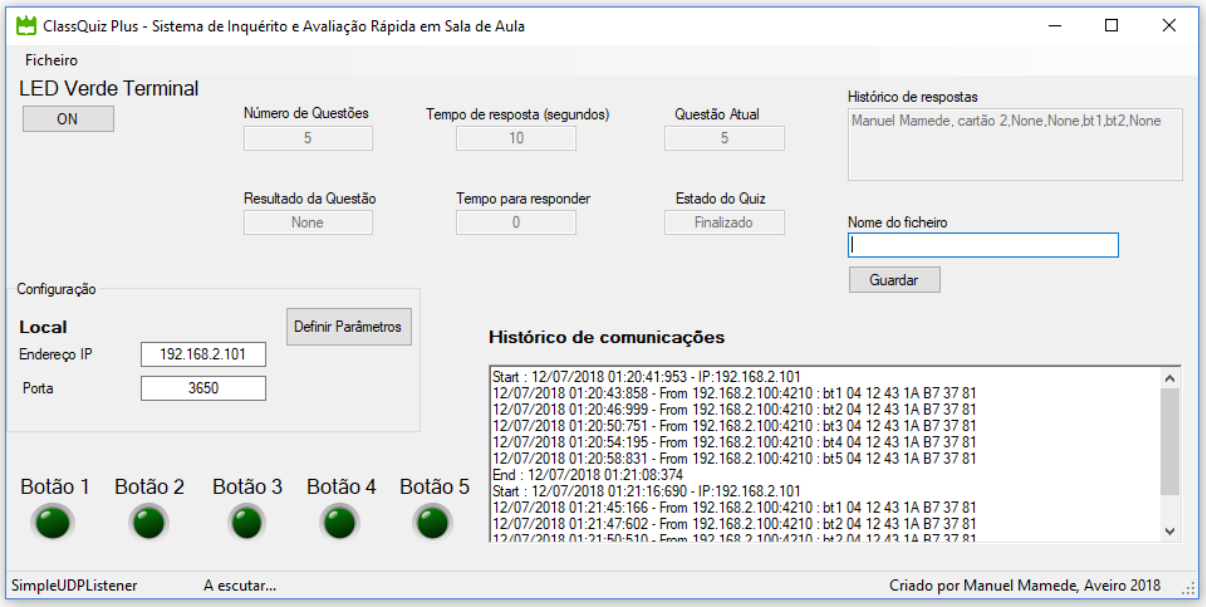

Figura 29 - Aplicação Após Fim de um Questionário

Após o término de um *quiz* de escolha múltipla [Figura 29] é apresentado no histórico de respostas, o utilizador associado ao UID e o resultado do *quiz*.

7. Conclusão

Neste projeto foram usados conhecimentos de programação em Arduino e *Visual Basic*, assim como de desenvolvimento de circuitos elétricos em *Eagle* e modulação 3D em *Autodesk Inventor*.

A comunicação por Wi-Fi provou-se funcional e prática para o desenvolvimento do projeto, tendo se observado apenas alguns atrasos entre o envio e receção e alguma troca de botões em raras ocasiões numa fase menos recente deste trabalho. O atraso pode dever-se diretamente ao computador ou a uma sobre carga no router. Na versão final do software não foram detetadas quaisquer trocas de botões ao enviar a mensagem.

Foram completadas a primeira e segunda fase de planeamento de projeto, ficando a terceira fase para uma aplicação futura na continuação deste projeto. Algumas melhorias futuras poderão passar por reduzir a dimensão da placa PCB e assim reduzir a dimensão da caixa. Outra melhoria poderá ser a passagem para comunicação por TCP/IP de forma a ter confirmação de entrega de mensagens, como descrito no capítulo 4, e modificar a aplicação em *VB* de forma a ser capaz de gerir eficientemente mais do que um terminal a enviar mensagens durante uma sessão. E implementando a terceira fase deste projeto já é possível ter acesso a uma base de dados que associa o UID do cartão ao respetivo aluno, determinando assim se está autorizado a ativar o terminal, e encriptando a comunicação permite evitar que alguém crie um dispositivo semelhante com o objetivo de sabotar a avaliação.

Um possível inconveniente para a implementação prática deste projeto em sala de aula será o facto de ser necessário ter um *router* dedicado ao efeito, visto que a rede da Universidade de Aveiro, a *eduroam*, não permite a ligação ou envio de mensagens a este tipo de dispositivos. Este *router* terá de ter alcance suficiente para uma sala de aula ou anfiteatro, assim como terá de ser capaz de gerir até uma possível centena de terminais constantemente a enviar mensagens. Este poderá ser um problema para o computador responsável pela gestão das respostas, podendo ser necessário que armazene as mensagens em buffer para processar mais tarde.

A inclusão de uma bateria ou pilha permitirá ao terminal ser independente não necessitando de conexão por USB. Chegaram a ser realizados testes com uma *power bank* que provaram que o terminal é capaz de funcionar completamente *wireless* com a aplicação *VB.*

Em relação à placa PCB, seria possível reduzir as suas dimensões reduzindo a largura das vias até ao mínimo que fosse possível fabricar. Os componentes soldados, para uma produção mais eficaz, devem estar todos rentes à placa, ao contrário dos LED's que estão elevados para chegar ao cimo da caixa. O módulo RFID poderia ter ficado diretamente por baixo da placa e não a sair desta, assim como ficar alinhado com a placa de forma a não exceder a sua largura.

Por consequência das melhorias descritas no parágrafo anterior a caixa do projeto também ficaria mais pequena. E a necessidade de existirem extensores para os botões poderia ser eliminada utilizando botões mais altos.

Para o projeto atual foi implementada a comunicação entre um terminal e uma estação-base, neste caso o computador. A autenticação é feita localmente, isto é, dentro do *NodeMCU* e do *VB* existe uma lista com os cartões que possuem acesso ao terminal. A comunicação é feita por UDP, sem confirmações de entrega. Para uma versão real a comunicação terá de ser entre múltiplos terminais e uma estação-base. A autenticação será feita com recurso a uma base de dados em MySQL acessível ao programa na estação-base. Para segurança extra, ou seja, evitar terminais duplicados, o cartão deverá permanecer no terminal de modo a que este funcione. Se o cartão for retirado a ligação Wi-Fi é cortada de imediato, e é necessário voltar a repetir o processo de autenticação. A comunicação deverá ter alguma forma de confirmação de chegada à estação-base, como é o caso do protocolo TCP/IP.

Obteve-se com este projeto uma prova de que este conceito funciona e pode ter aplicações em situações reais.

8. Referências bibliográficas

[1] – TEIXEIRA, António. Class Quiz. Aveiro, 2017

[2] – Biblioteca RFID RC522 para Arduino – Disponível em:

https://github.com/miguelbalboa/rfid

[3] – Exemplo de envio de *strings* por UDP – Disponível em:

https://www.arduino.cc/en/Tutorial/UDPSendReceiveString

[4] – Classe BackgroundWorker – Disponível em:

https://msdn.microsoft.com/en-us/library/system.componentmodel.backgroundworker(v=vs.110).aspx

[5] – Tutorial de escrita e leitura de cartões com o RFID RC522 – Disponível em:

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

[6] – Tutorial de criação de *Multicast DNS* – Disponível em:

https://tttapa.github.io/ESP8266/Chap08%20-%20mDNS.html

[7] – Exemplo de interface entre o RFID e *NodeMCU* – Disponível em:

http://www.instructables.com/id/MFRC522-RFID-Reader-Interfaced-With-NodeMCU/

[8] – Tutorial de *NodeMCU* – Disponível em:

https://www.hackster.io/Aritro/getting-started-with-esp-nodemcu-using-arduinoide-aa7267

[9] – Módulos *Lua* Wi-Fi do *NodeMCU* – Disponível em:

https://nodemcu.readthedocs.io/en/master/en/modules/wifi/