



João Carlos da Silva
Moreira

Desenvolvimento e implementação em contexto
realista do sistema CLASSQUIZ



**João Carlos da Silva
Moreira**

**Desenvolvimento e implementação em contexto
realista do sistema CLASSQUIZ**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de Vítor Manuel Ferreira dos Santos, Professor Associado C/ Agregação do Departamento de Engenharia Mecânica da Universidade de Aveiro.

O júri / The jury

Presidente / President

Prof. Doutor José Paulo Oliveira Santos

Professor Auxiliar da Universidade de Aveiro

Vogais / Committee

Prof. Doutor Gustavo Ribeiro da Costa Alves

Professor Adjunto do Instituto Superior de Engenharia do Porto (arguente)

Prof. Doutor Vítor Manuel Ferreira dos Santos

Professor Associado C/ Agregação da Universidade de Aveiro (orientador)

Agradecimentos / Acknowledgements

Ao Prof. Doutor Vítor Santos por toda a ajuda prestada ao longo da realização deste trabalho.

À equipa do Laboratório de Automação e Robótica, em especial ao Sr. Rui Heitor pela ajuda durante o período em que realizei a minha dissertação, e ao Sr. Luís Rodrigues pela ajuda na montagem dos componentes electrónicos. Agradeço também ao Tiago Gomes e ao Rafael Marques pela partilha da sua experiência em impressão 3D, e pela ajuda e atenção disponibilizadas, e a todos os que participaram nas sessões de testes realizadas.

A todos os meus amigos e colegas, dentro e fora da Universidade, pela ajuda e apoio constantes durante todo o meu percurso académico, em especial ao Pedro Abade pela ajuda na aprendizagem de umas das principais ferramentas utilizadas nesta dissertação.

À minha namorada Susana por acreditar em mim, pela companhia e amizade, e pela paciência que demonstrou para comigo ao longo deste percurso. Agradeço aos meus pais, irmãos e avós o apoio, formação e motivação dados ao longo de toda a minha vida, que contribuíram de forma crucial para os meus êxitos pessoais e académicos, e em particular ao meu irmão Nuno pela partilha de conhecimentos e apoio constante. Agradeço ainda ao José Carlos Fernandes pelo apoio e disponibilidade prestados.

Palavras-chave

Inquéritos electrónicos; Aplicação *web*; Django; Base de dados; ESP8266; Autenticação; RFID.

Resumo

Este trabalho foca o desenvolvimento de uma nova solução de sondagem e avaliação em contexto de sala de aula, especialmente em ambientes com muitos alunos, designada CLASSQUIZ. Neste documento, para além da justificação das opções tomadas durante o seu desenvolvimento, é feita uma reflexão sobre os resultados recolhidos dos participantes das sessões de teste realizadas, em ambiente realista.

O sistema CLASSQUIZ permite aos professores e formadores realizarem inquéritos, questionários e sondagens, sob a forma de questões de escolha múltipla, que podem ser usados para a avaliação dos alunos, ou meramente para efeitos estatísticos, podendo ser anónimos ou não. Assim, o sistema CLASSQUIZ permite aumentar a interactividade dos alunos numa sala de aula, promovendo a sua participação e estimulando a atenção dada à matéria leccionada.

O sistema desenvolvido consiste num servidor e numa série de terminais individuais, a partir dos quais os alunos respondem a questões de escolha múltipla postas globalmente. A autenticação dos alunos é assegurada através dos seus cartões de estudante, que deve permanecer no terminal durante a realização do inquérito.

De forma a garantir que apenas os alunos presentes na sala de aula podem participar nos inquéritos realizados, o sistema encontra-se ligado a uma rede local, cujo acesso se encontra restrito a terminais devidamente registados no sistema, prevenindo assim qualquer forma de acesso do exterior.

Em vez de precisarem de instalar *software* adicional nos seus computadores pessoais, os professores podem aceder à interface gráfica do sistema a partir de qualquer dispositivo com um navegador de Internet. Para além disso, cada questão encontra-se associada a um URL, que pode ser inserido numa qualquer apresentação existente, garantindo a fluidez da lição. O sistema possibilita ainda a apresentação dos resultados dos inquéritos realizados, permitindo aos participantes receberem um *feedback* imediato da sua participação.

Keywords

Electronic surveys; Web application; Django; Database; ESP8266; Authentication; RFID.

Abstract

This dissertation focuses on the development of a solution for polling and evaluating students in a classroom environment, named CLASSQUIZ. In this document, in addition to the justification of the options taken during its development, a reflection is made on the results collected from the participants of the test sessions performed in a realistic environment.

The CLASSQUIZ system allows teachers and trainers to carry out surveys and quizzes, in the form of multiple choice questions, which can be used for student evaluation, or merely for statistical purposes, whether anonymous or not. Thus, the CLASSQUIZ system allows the increase of students' interactivity in a class classroom, promoting their participation and stimulating the attention given to the subject being taught.

The developed system consists of a server and a set of individual terminals, through which students are able to answer multiple choice questions posed globally. Authentication of the students is ensured through their ID cards, which must remain inserted in the individual terminal during the quiz or survey.

In order to ensure that only the students present in the classroom can participate in the surveys, the system is connected through a dedicated local network, whose access is restricted to terminals properly registered in the system, thus preventing any form of access from the outside.

Instead of being required to install additional software on their personal computers, teachers can access the CLASSQUIZ User Interface from any device with a web browser. In addition, each question is associated with a URL, which can be inserted into any existing presentation, allowing teachers to keep the flow of the lesson. The system also allows the presentation of the results of the surveys, allowing the participants to receive immediate feedback on their performance.

Índice

I	Enquadramento	1
1	Introdução	3
1.1	Enquadramento e Motivação	3
1.2	Objectivos	4
1.3	Estrutura da Dissertação	4
2	Revisão Bibliográfica	7
2.1	Controlo de Acessos	7
2.2	Tecnologias de Autenticação	8
2.2.1	Métodos Tradicionais	9
2.2.2	Sistemas Biométricos	11
2.3	Internet of Things	14
2.4	Tecnologias e Protocolos de Comunicação	14
2.5	Segurança de Redes Wi-Fi	18
2.5.1	Wired Equivalent Privacy	18
2.5.2	Wi-Fi Protected Access	18
2.5.3	Wi-Fi Protected Access 2	19
2.6	Bases de Dados	19
2.6.1	Modelos de Bases de Dados	19
2.6.2	Bases de Dados Relacionais	20
2.7	Aplicações WEB	21
2.7.1	Front-end	21
2.7.2	Back-end	22
3	Trabalhos Relacionados	25
3.1	Sistemas de Gestão de Aprendizagem	25
3.2	QuizXpress	26
3.3	Versões Anteriores do Sistema CLASSQUIZ	28
3.3.1	Primeira Versão do Sistema	28
3.3.2	Segunda Versão do Sistema	29
4	Solução Proposta	33
4.1	Autenticação dos Utilizadores	33
4.2	Modelo de Comunicação	34
4.3	Ferramentas Utilizadas	34
4.3.1	MySQL	34
4.3.2	Django	35

4.3.3	AJAX	36
4.4	Considerações Adicionais	36
II Desenvolvimento e Implementação		39
5	Desenvolvimento da Aplicação WEB	41
5.1	Conceitos Importantes	41
5.2	Arquitectura da Aplicação	42
5.2.1	Configurações Gerais	44
5.2.2	Aplicações	44
5.2.3	Modelos	45
5.2.4	Vistas	46
5.2.5	Formulários	48
5.2.6	URL's	48
5.2.7	Templates	49
5.2.8	Ficheiros Estáticos	50
5.3	Relações da Base de Dados	51
5.3.1	Descrição da Base de Dados, por Aplicação	51
5.3.2	Utilizadores, Unidades Curriculares e Questões	53
5.3.3	Lições, Questões e Sessões	54
5.3.4	Resultados	55
5.3.5	Respostas	57
5.3.6	Hierarquia dos Modelos	58
5.4	Realização de um Questionário e Processamento das Respostas	58
5.4.1	Realização de um Inquérito	60
5.4.2	Recepção das Respostas	61
5.4.3	Avaliação das Respostas	62
5.4.4	Acções no Cliente e Servidor	65
6	Desenvolvimento do Terminal Individual	67
6.1	Hardware	67
6.1.1	Microcontrolador	68
6.1.2	Módulo RFID	69
6.1.3	8-bit Priority Encoder	70
6.1.4	Botões e LED's	71
6.1.5	Alimentação	72
6.1.6	PCB	73
6.1.7	Estrutura de Fixação e Protecção	73
6.1.8	Lista de Materiais	75
6.2	Firmware	75
6.2.1	Leitura da UID do Cartão	78
6.2.2	Presença Contínua do Cartão	78
6.2.3	Envio da Mensagem HTTP	79
6.2.4	Validação do Cartão	79
6.2.5	Envio das Respostas	79

III	Resultados e Discussão	81
7	Testes Realizados em Ambiente Realista	83
7.1	Sessão de Testes em Ambiente Realista	83
7.1.1	Resultados da Sessão	85
7.2	Teste de Usabilidade da Aplicação WEB	88
7.2.1	Resultados do Teste de Usabilidade	88
7.3	Sessão de Testes em Realizada na Academia de Verão	90
8	Considerações Finais	93
8.1	Conclusões	93
8.2	Trabalhos Futuros	94
	Anexos	101
	Anexo A Diagrama do Circuito Eléctrico	103
	Anexo B Desenho Técnico da Estrutura Mecânica dos Terminais	105
	Anexo C Inquérito Apresentado na Sessão de Testes em Ambiente Realista	107
	Anexo D Manual de Utilização do Sistema CLASSQUIZ	111

Lista de Tabelas

2.1	Caracterização das diferentes classes Bluetooth [20].	15
6.1	Tabela de verdades do codificador CD4532BE [57].	70
6.2	Tabela de materiais para um terminal individual.	75
6.3	Tabela de ligações do microcontrolador.	76

Lista de Figuras

2.1	Esquema de funcionamento de um sistema RFID.	10
2.2	Constituição de um cartão de proximidade.	11
2.3	Representação da região de interesse e descritores no reconhecimento facial.	12
2.4	Representação do olho humano.	13
2.5	Exemplo do controlo remoto de uma "Casa Inteligente".	14
2.6	Topologias de uma rede ZigBee.	16
2.7	Protocolos de comunicação: TCP <i>vs</i> UDP.	17
2.8	Representação esquemática dos diferentes tipos de relações [30].	21
2.9	Comunicação cliente-servidor segundo o protocolo HTTP.	22
3.1	Janela de criação de inquéritos do sistema QuizXpress.	26
3.2	Exemplo de um inquérito realizado no sistema QuizXpress.	27
3.3	Diferentes tipos de controladores do sistema QuizXpress.	27
3.4	Esquema eléctrico do terminal [48].	28
3.5	Interface gráfica da aplicação desenvolvida por Mamede [49].	29
3.6	Sistema desenvolvido por Mamede [49].	30
3.7	Esquema eléctrico do sistema [49].	30
4.1	Diagrama conceptual do sistema.	33
4.2	Diagrama representativo do envio, recepção e processamento de respostas do sistema proposto.	37
5.1	Árvore da aplicação <i>web</i>	42
5.2	Esquema relacional dos modelos da aplicação <i>users</i>	52
5.3	Esquema relacional dos modelos da aplicação <i>quiz</i>	52
5.4	Relação entre os modelos <i>User</i> , <i>Course</i> e <i>Quiz</i>	54
5.5	Relação entre os modelos <i>Course</i> , <i>Lesson</i> , <i>Quiz</i> e <i>Session</i>	55
5.6	Relação dos modelos <i>Results</i> e <i>LessonStudent</i>	56
5.7	Hierarquia dos principais modelos da aplicação.	58
5.8	Diagrama exemplificativo do processamento de respostas.	59
6.1	Comparação do mapa de pinos das duas versões do microcontrolador.	68
6.2	Comparação das dimensões dos microcontroladores NodeMCU v3 e v2.	69
6.3	Módulo Mifare RC522.	69
6.4	Codificador Texas Instruments CD4532BE.	70
6.5	Botão de pressão unipolar de uma via 6x6x13 mm.	71
6.6	LED's de 5 mm utilizados nos terminais individuais.	71
6.7	Alimentação dos terminais individuais.	72

6.8	Interruptor unipolar de uma via.	73
6.9	PCB desenvolvida.	73
6.10	Renderização do terminal individual, sem tampa (conjunto).	74
6.11	Renderização do terminal individual, com tampa (conjunto).	74
6.12	Diagrama do <i>firmware</i> dos terminais individuais.	77
7.1	Ilustração do ambiente de realização das sessões de teste do sistema.	84
7.2	Resultados médios da sessão de testes em ambiente realista.	85
7.3	Apresentação do modo de ecrã inteiro num <i>smartphone</i>	89
7.4	Página de selecção de lições da aplicação <i>web</i>	89
7.5	Sessão de testes realizada na Academia de Verão.	91
A.1	Diagrama do circuito eléctrico do terminal individual.	103
B.1	Desenho técnico da tampa (dimensões em mm) da caixa do terminal individual.	105
B.2	Desenho técnico da caixa (dimensões em mm) do terminal individual.	106

Listagens

5.1	Vista <code>QuizListView</code>	46
5.2	Vista <code>start_quiz</code>	47
5.3	Formulário <code>QuizForm</code>	48
5.4	Endereços URL.	49
5.5	Incorporação do formulário <code>course_form</code> num <i>template</i>	49
5.6	Função que evoca a vista <code>start_quiz</code>	50
5.7	Controlo do tempo restante de uma questão.	60
5.8	Vista <code>quiz_response</code>	61
5.9	Extracção da identificação do cartão da mensagem enviada pelo terminal.	62
5.10	Migração das respostas do modelo <code>Answer</code> para o modelo <code>AnswerProcessing</code>	62
5.11	Extracção da identificação da questão activa.	63
5.12	Verificação da lição activa.	63
5.13	Processamento das respostas.	64
5.14	Avaliação das respostas.	65
6.1	Obtenção da UID do cartão de estudante.	78
6.2	Verificação da presença do cartão de estudante.	79

Lista de Acrónimos

AES Advance Encryption Standard.

AJAX Asynchronous Javascript and XML.

CDN Content Delivery Network.

CMOS Complementary Metal–Oxide–Semiconductor.

CSRF Cross Site Request Forgery.

CSS Cascading Style Sheets.

DHCP Dynamic Host Configuration Protocol.

GSM Global System of Mobile Communication.

HTML HyperText Markup Language.

HTTP HyperText Transfer Protocol.

I²C Inter-Integrated Circuit.

IDE Integrated Development Environment.

IoT Internet of Things.

IP Internet Protocol.

JS JavaScript.

JSON JavaScript Object Notation.

LED Light-Emitting Diode.

LMS Learning Management Systems.

MAC Media Access Control.

MIC Message Integrity Check.

PCB Printed Circuit Board.

RFID Radio-Frequency Identification.

SPI Serial Peripheral Interface.

SQL Structured Query Language.

TCP Transmission Control Protocol.

UDP User Datagram Protocol.

UID Unique ID.

URL Uniform Resource Locator.

USB Universal Serial Bus.

WEP Wired Equivalent Privacy.

WLAN Wireless Local Area Network.

WPA Wi-Fi Protected Access.

Parte I

Enquadramento

Capítulo 1

Introdução

Neste documento é apresentado todo o trabalho efectuado na unidade curricular Dissertação, projecto de final de curso no âmbito do Mestrado Integrado em Engenharia Mecânica da Universidade de Aveiro.

Neste capítulo será apresentado o Enquadramento e Motivação para o desenvolvimento da presente dissertação, seguida dos Objectivos que se pretendem atingir. Por fim, está presente a Estrutura do documento.

1.1 Enquadramento e Motivação

Os sistemas de avaliação pedagógica modernos preconizam um envolvimento contínuo dos estudantes. Nessa linha, uma solução em certas unidades curriculares será a de promover momentos de participação/avaliação de todos os alunos presentes nas aulas ou sessões de formação. Contudo, em aulas com um número elevado de alunos, a realização de pequenos momentos de avaliação pode-se revelar um desafio. O sistema aqui proposto, designado CLASSQUIZ, procura permitir a realização de inquéritos/questionários com resultados imediatos e automáticos. O sistema é baseado em três componentes:

- Terminais individuais, portáteis e sem fios, onde cada aluno presente na sala de aula poderá responder a questões postas globalmente, usando um sistema de escolha múltipla;
- Uma estação base-receptora, presente na sala, que valida, autentica e estabelece as trocas de comunicações com os terminais individuais;
- Um *software* dedicado, a correr num computador ligado ao sistema base, que mantém e gere uma base de dados de alunos, as suas respostas, bem como os tempos de pergunta/resposta.

Uma preocupação grande é a segurança e privacidade do sistema, que deve restringir intrusões exterior, de forma a que só os presentes na sala possam participar numa dada sessão, e apenas dentro da sala. Para além disso, uma vez que se trata de um sistema que visa a avaliação de alunos, é imperativo garantir que o aluno que responde a uma dada questão é na verdade quem diz ser. A autenticação do inquirido pode ser assegurada com um cartão de identificação individual, como o cartão de estudante, ou outro devidamente registado. Para suporte das comunicações, o recurso ao Wi-Fi assume-se como a solução

mais fácil e expansível, não obstante algumas potenciais restrições. As funções principais do terminal individual são:

- Validar em contínuo a presença na sala (por cartão de estudante);
- Resposta a questões de escolha múltipla;
- Receber confirmação da estação base como meio de *feedback* ao aluno.

Sem validação individual, o sistema poderá ser potencialmente usado também para inquéritos anónimos e testes de diagnóstico, ou para efeitos estatísticos.

1.2 Objectivos

A presente dissertação tem como objectivo o desenvolvimento e implementação, em contexto realista, de um sistema de realização de inquéritos electrónicos, designado CLASSQUIZ. Um estudo e demonstração deste sistema já foi feito em trabalhos anteriores, deixando, contudo, vários problemas em aberto, nomeadamente no que toca a questões de escalabilidade e robustez do mesmo, sob o sistema Wi-Fi. Assim, torna-se necessário um refinamento do *hardware* e *firmware* do sistema já desenvolvido para um estudo da potencial limitação das comunicações em ambiente Wi-Fi. Para além disso, é necessário o desenvolvimento de uma aplicação, em ambiente realista, que teste o *software* ao mais alto nível na estação base, em especial a sua interligação com um sistema de bases de dados, para gerir os fluxos de informação e ações dos participantes.

De forma a que a aplicação desenvolvida seja agnóstica ao sistema operativo, esta deve ser desenvolvida de forma a correr num *browser*, e com potencialidade de acesso a uma base de dados. Esta aplicação deve ter as funções de gerir as questões (texto e imagem) a colocar por projecção numa tela, bem como as temporizações e toda a dinâmica de registo de respostas na base de dados.

Para além do refinamento do *hardware* e das comunicações sobre o sistema existente do CLASSQUIZ, existe a necessidade de desenvolver um *software* que não apenas faça a interface entre a leitura de cartão e a sua activação para a sessão em causa, mas que também se encarregue, através de um *browser*, de fazer o *display* dos questionários para todos os presentes na aula, bem como a sua temporização, para além da recepção e validação das respostas.

Por fim, de forma a testar a escalabilidade do sistema, devem ser realizados testes e demonstrações, em ambiente realista, tendo em operação todos os componentes previstos para uma solução funcional.

1.3 Estrutura da Dissertação

Para além da Introdução, esta dissertação contém mais 7 capítulos.

No capítulo 2 é feita uma revisão bibliográfica aos assuntos enquadrados no tema desta dissertação, tais como diferentes métodos de autenticação possíveis de implementar, tecnologias de comunicação sem fios, bases de dados e por fim sobre em que é que consiste uma aplicação *web*.

No capítulo 3 são apresentados os trabalhos anteriormente realizados no âmbito deste projecto, bem como algumas soluções alternativas.

No capítulo 4 é apresentada uma solução para o problema, tendo em conta as tecnologias mencionadas no capítulo 2, bem como os trabalhos já desenvolvidos, mencionados no capítulo 3.

O capítulo 5 descreve a implementação do *software* do sistema, a correr na estação base-receptora. Neste capítulo são abordadas as relações entre registos e a estrutura geral da base de dados do sistema. São ainda abordadas e justificadas as opções tomadas no que toca a ações que ocorrem no cliente e no servidor, para além de ser descrito o sistema de recepção, processamento e avaliação de respostas.

No capítulo 6 é descrita a implementação do terminal individual. Numa primeira instância é detalhada a constituição do terminal individual, seguindo-se da implementação do seu *firmware*, onde são expostas e justificadas algumas opções tomadas, nomeadamente no que à autenticação dos alunos e registo de respostas diz respeito.

No capítulo 7 é descrito o modelo de testes adoptado para a validação do sistema em ambiente realista. São ainda apresentados e debatidos os resultados obtidos dos mesmos, bem como o *feedback* recebido por parte dos dois tipos de utilizadores alvo: alunos e professores.

Por fim, no capítulo 8 são apresentadas as considerações finais, sendo ainda apresentadas sugestões de trabalho futuro.

Capítulo 2

Revisão Bibliográfica

Neste capítulo são apresentadas diversas tecnologias para o desenvolvimento e implementação de um sistema de inquérito electrónico.

Inicialmente, procede-se à análise e estudo de modelos de autenticação do utilizador remoto, seguindo-se um estudo às diversas formas possíveis de comunicação *wireless*, de modelos de gestão de bases de dados e, por fim, das ferramentas utilizadas na criação de uma aplicação *web*.

2.1 Controlo de Acessos

O controlo de acessos consiste num conjunto de sistemas que permitem gerir o nível de autorização de acesso de um sujeito a um determinado espaço, seja este físico ou virtual. Estes sistemas seguem um modelo de segurança pré-estabelecido, uma vez que o nível de autorização de um sujeito é lhe atribuído por uma entidade superior, e o processo de concessão de acesso ao espaço segue um conjunto de procedimentos previamente estabelecidos pela mesma. Consequentemente, é possível controlar o acesso de um sujeito a um espaço protegido, bem como monitorizar as suas acções dentro do mesmo [1].

De forma a impedir o acesso ao espaço protegido por parte de sujeitos não autorizados, e garantir que todas as acções realizadas dentro do mesmo são realizadas em conformidade com as normas de segurança definidas, é necessário implementar um conjunto de metodologias que se complementem. Assim, o processo de autorização de um determinado sujeito é composto por três processos essenciais (identificação, autenticação e autorização), que visam conceder acesso do mesmo ao espaço protegido. Para além dos processos referidos, existe ainda um quarto, a auditoria, que consiste na monitorização da atividade do sujeito, dentro do espaço protegido, por parte de uma entidade superior [2, 3].

Identificação

A identificação do sujeito é um processo preliminar que consiste na recolha de informação sobre o mesmo, de forma a assegurar a sua autenticidade. Este processo resulta na emissão de credenciais, que podem assumir a forma de um cartão de identificação ou de uma palavra-chave, que o sujeito poderá usar posteriormente para validar a sua identidade. Uma vez emitidas, as credenciais devem ser validadas como parte do processo de autenticação [2, 4].

Autenticação

Quando um sujeito apresenta as suas credenciais, estas devem ser autenticadas pelo sistema de controlo de acessos. A informação recolhida é então comparada com a informação registada no sistema (por exemplo, numa base de dados) e, caso coincidam, o sujeito é considerado autenticado [1, 2]. Existem vários métodos de autenticação, que serão discutidos com maior detalhe mais à frente.

Autorização

A um sujeito autenticado não deve ser dado acesso ao espaço protegido de forma imediata, devendo ser implementados procedimentos de controlo de forma a garantir que cada sujeito autenticado possui também autorização por parte de uma entidade superior para o aceder. O processo de verificação de autorização é necessário pois um sujeito pode ter permissão para aceder ao espaço apenas mediante determinadas condições (por exemplo, pode ter permissão o fazer num horário específico, para executar apenas um número reduzido de acções, ou para aceder apenas a determinados espaços dentro de um espaço global) [2, 3, 4].

Auditoria

Como já foi referido, a auditoria consiste na monitorização da actividade do sujeito a quem foi concedido acesso ao espaço protegido, possibilitando a criação de um registo da mesma. A criação deste registo de actividades permite verificar, posteriormente, a existência de erros no sistema, bem como detectar tentativas de acesso não autorizadas. Assim, a auditoria permite avaliar os mecanismos de controlo implementados e agir em conformidade [2, 4].

2.2 Tecnologias de Autenticação

Como referido na secção anterior, existem vários métodos de autenticação, que servem para verificar a veracidade das credenciais apresentadas pelo sujeito. De uma forma geral, é possível distinguir os métodos de autenticação em duas categorias principais: os métodos tradicionais e os biométricos.

Todas as tecnologias disponíveis apresentam vantagens e desvantagens na sua utilização. Sendo verdade que a tecnologia adoptada deve ser fácil de utilizar, isso pode levar a que seja igualmente fácil um sujeito falsificar a sua identidade. É importante referir que num sistema de controlo de acessos é possível utilizar vários métodos de autenticação distintos em simultâneo (autenticação de factor múltiplo), conduzindo a um sistema mais seguro, uma vez que existem mais barreiras que precisam de ser ultrapassadas numa tentativa de acesso não autorizado. Este aumento de segurança, no entanto, tem um custo associado à inconveniência da sua utilização, uma vez que o utilizador necessita de passar por vários momentos de autenticação para poder aceder ao espaço [3, 5].

Esta secção visa apresentar as principais tecnologias utilizadas em sistemas de controlo de acessos, com o fim de validar a identificação do sujeito que procura acesso a um espaço protegido.

2.2.1 Métodos Tradicionais

Tradicionalmente, os métodos de autenticação são baseados no *conhecimento* ou *posse* do sujeito. Os primeiros dependem de algo que o sujeito sabe à prior, como uma senha, e são talvez os métodos mais antigos, sendo também por isso dos mais vulneráveis, uma vez que se baseiam na memória do sujeito que, por natureza, estão sujeitos ao esquecimento. Por outro lado, os métodos baseados na *posse* requerem que o sujeito possuía consigo um objecto físico para proceder à sua autenticação. Estes métodos são também amplamente utilizados no quotidiano da generalidade das pessoas, sob a forma de chaves ou cartões de identificação e, tal como os restantes métodos tradicionais, são bastante susceptíveis de serem falsificados, já que o sujeito pode perder as suas credencias, ou vê-las serem roubadas [3].

Palavra Chave

Num mundo cada vez mais tecnológico, este método de autenticação encontra-se empregado em várias tarefas do dia-a-dia da população sob a forma de um *username* e *password*. De forma a garantir que um sujeito/sistema comprometido não comprometa outros sujeitos/sistemas, as palavras-chave devem ser únicas tanto ao nível do sujeito, como ao nível do sistema onde são empregues. Contudo, são tantas as diferentes palavras-chave que um sujeito tem de memorizar, que muitas vezes acabam por usar as mesmas credenciais para aceder a espaços completamente distintos, resultando assim numa perda significativa do nível de segurança que as mesmas providenciam. Para além disso, uma palavra-chave criada por um ser humano é mais facilmente descoberta por *software* adequado, ou até mesmo por tentativa erro, do que senhas geradas por um computador, uma vez que precisam de ser simples o suficiente para o utilizador as conseguir memorizar [6].

Em resultado disto foram criadas várias soluções, desde sistemas de autenticação única (isto é, o sujeito só necessita de se autenticar uma vez) a gestores de palavras-chave, que nada mais são que sistemas onde as várias senhas de um sujeito são armazenadas [7]. Uma vez que, por norma, ambas as soluções funcionam ao nível de um dispositivo pertencente ao sujeito, apesar das vantagens significativas que apresentam, ambas são susceptíveis ao mesmo problema: se um intruso conseguir acesso ao dispositivo, poderá ter também acesso às credenciais. Ainda assim, num mundo maioritariamente virtual, onde as senhas são cifradas de forma a aumentar o seu nível de segurança, ganhar acesso a um dispositivo físico não é tarefa fácil, sendo por isso que este método é um dos mais utilizados mundialmente.

RFID

A tecnologia RFID (identificação por rádio-frequência) incorpora o uso de electro-magnetismo, ou o acoplamento electro-estático na rádio-frequência, para identificar de forma única um sujeito [8]. Este método de autenticação, que se enquadra na categoria da *posse* de credenciais por parte do sujeito, não necessita da existência de contacto físico entre o leitor e o identificador, e é significativamente mais fiável que outras tecnologias existentes [3, 4, 9].

Algumas das principais vantagens desta tecnologia estão relacionadas com a comunicação e transmissão de dados, que pode ocorrer sem linha de visão óptica, uma vez que as ondas electromagnéticas de rádio-frequência são capazes de atravessar diversos tipos de

materiais, bem como com a elevada velocidade do processo de autenticação e capacidade de leitura de dados à distância [3], que pode variar entre o centímetro e vários metros dependendo do sistema utilizado [4]. Contudo, existem também algumas desvantagens relacionadas com a adopção desta tecnologia. A primeira, e mais proeminente, é o facto de se tratar de um método de autenticação baseado na *posse* de credenciais, que podem ser esquecidas, pedidas ou roubadas. Outra desvantagem, menos comum mas não menos importante, é o facto de ser possível a sua leitura à distância, o que faz com que seja possível, através de um dispositivo dedicado, ler e recolher informação de outros utilizadores sem que estes se apercebam, possibilitando a falsificação de credenciais de acesso com a informação recolhida [10].

Apesar das desvantagens referidas, e em resultado das vantagens que a autenticação por RFID apresenta em comparação com outros métodos de autenticação, associado ao seu baixo custo de implementação, esta tecnologia é utilizada amplamente no inventário de objectos e produtos alimentares [1, 9], ou mesmo para a identificação de indivíduos em organizações e empresas, através da utilização de cartões de proximidade [1, 8].

Um sistema RFID baseia-se, portanto, na emissão e recepção de sinais rádio, sendo constituído essencialmente por três componentes: uma antena, um transmissor e um *transponder* (ou etiqueta), como mostra a figura 2.1. A antena, em conjunto com o transmissor, forma o leitor, e a sua principal função é fornecer os meios de comunicação necessários para a transferência de dados. Para tal, a antena emite sinais de rádio, que activam a etiqueta e permitem a sua leitura/escrita, através da criação de um campo electromagnético que pode ser permanente, ou activado e desactivado quando necessário. A antena é constituída por um filamento metálico e impõe uma frequência de emissão na comunicação. Para além disso, a antena tem de ter uma impedância compatível com a do *transponder*, de forma a optimizar a transferência de energia entre ambas [1, 11]. O leitor é então um dispositivo que comunica com a etiqueta, de forma a identifica-la e adquirir o identificador único a si associado [3].

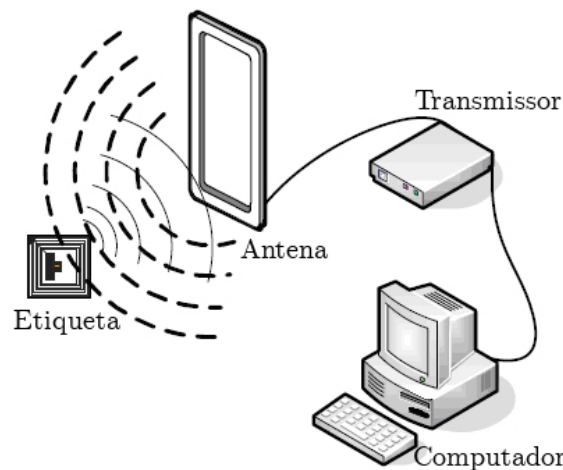


Figura 2.1: Esquema de funcionamento de um sistema RFID.

O *transponder* é o dispositivo que transporta informação num sistema RFID e consiste num circuito integrado com memória. Regra geral, não possui uma fonte de alimentação própria, possuindo bobines que geram energia enquanto se encontrarem dentro da zona de ação do leitor. Este tipo de etiquetas (passivas) são tipicamente mais leves, pequenas

e baratas do que as activas, que possuem uma bateria própria e que permitem maior alcance. Por estes motivos, e por terem um tempo de vida bastante longo, as etiquetas passivas são bastante mais utilizadas que as alternativas [3, 4, 11].

Como já foi referido, uma das principais utilizações de sistemas RFID é na identificação de indivíduos, através de cartões de proximidade. Como o próprio nome indica, este requer apenas uma aproximação a um leitor, não necessitante de contacto físico. Neste caso, tanto o cartão como o leitor possuem antenas internas, de forma a possibilitar a comunicação e troca de dados (figura 2.2). Por não possuírem bateria, estes cartões são passivos, gerando energia na presença do campo magnético gerado pelo leitor [3].

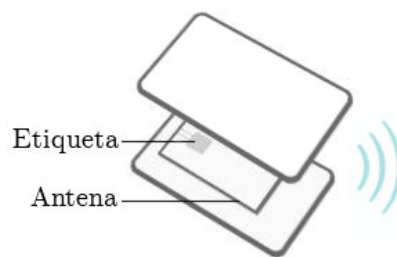


Figura 2.2: Constituição de um cartão de proximidade.

Outra característica importante da tecnologia RFID é a vasta gama de bandas de rádio-frequência em que pode operar. Os sistemas passivos operam, por norma, em bandas de frequência mais baixas, variando entre os 30 KHz e os 3000 MHz, sendo maioritariamente utilizados em sistemas de controlo de acessos, cartões de proximidade e no inventário de objectos. Já os sistemas activos podem operar em bandas que chegam aos 30 GHz e são utilizados quando é necessário um maior alcance, como é o caso das portagens electrónicas ou o rastreamento de bagagens nos aeroportos [11].

2.2.2 Sistemas Biométricos

Enquanto que os métodos tradicionais de autenticação são baseados no *conhecimento* e *posse* de um sujeito, cujas credenciais podem ser esquecidas, roubadas ou até duplicadas, os métodos biométricos focam-se nas características *físicas* ou *comportamentais* do sujeito, aumentando substancialmente o grau de segurança do sistema [12].

Apesar de ser um método utilizado em casos de identificação criminal já há vários anos, apenas recentemente começou a ser utilizado de forma mais ampla, em virtude da redução do custo associada às tecnologias empregues no mesmo [3], embora continue a ser um tema controverso devido à sua natureza intrusiva para o utilizador [13]. De forma a preservar a privacidade e identidade do sujeito, a informação biométrica recolhida aquando do registo do utilizador deve ser encriptada, evitando que seja utilizada por terceiros para fins maliciosos [1].

De seguida, são apresentadas algumas das técnicas de autenticação biométrica com maior relevo actualmente em sistemas de segurança comerciais.

Impressões Digitais

As impressões digitais, enquanto forma de identificação pessoal, são caracterizadas pelo seu elevado grau de variabilidade, de tal forma que dois indivíduos que partilhem o mesmo DNA, como os gémeos verdadeiros, possuem impressões digitais diferentes. De facto, a sua variabilidade é de tal ordem que nunca foram encontrados dois indivíduos com impressões digitais idênticas. Para além disso, a maioria da população possui impressões digitais legíveis o que, associado ao facto de estas não sofrerem alterações ao longo da vida do indivíduo (a não ser por acção de forças externas, como queimaduras ou lesões), torna-as numa forma válida de autenticação [14].

Tradicionalmente, era utilizada tinta para transpor as impressões digitais para um papel. O padrão que ficava registado no papel era depois digitalizado, utilizando um *scanner* tradicional. Actualmente, devido ao avanço tecnológico, é possível obter as impressões digitais directamente do indivíduo, através de um de quatro tipos de sensores distintos [3, 13, 14]:

- *Ópticos*: são os mais comuns e consistem na emissão de um feixe de luz, e na leitura da reflexão do mesmo para capturar uma imagem do padrão da impressão digital;
- *Capacitivos*: consistem na criação de diferentes pontos de acumulações de carga, causados pelos sulcos do dedo;
- *Térmicos*: recorrem à alteração da temperatura provocada na superfície do sensor;
- *Ultra-sónicos*: baseiam-se na reflexão de radiações ultra-som.

Reconhecimento Facial

A tecnologia associada ao reconhecimento facial permite identificar um indivíduo de forma automática, com base na sua estrutura facial, a partir de uma imagem ou vídeo. Para tal, estes sistemas procuram traços faciais bem definidos, como a posição dos olhos, nariz e boca, bem como a distância entre estes (figura 2.3). Uma vez adquirida a imagem, a região de interesse, neste caso a face do utilizador, é redimensionada para valores padrão, processada e armazenada, tipicamente numa base de dados [1, 3, 13]. Uma vez processada a imagem, os métodos de reconhecimento facial traduzem a imagem obtida num código digitalizado único, que será posteriormente comparado com a imagem adquirida aquando do momento de autenticação.

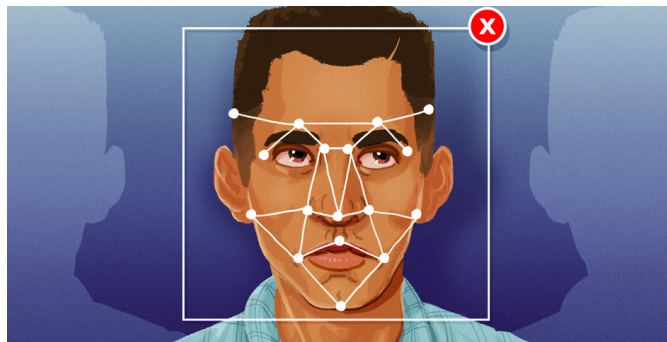


Figura 2.3: Representação da região de interesse e descritores no reconhecimento facial.

Esta tecnologia, em comparação com outros métodos biométricos, tem a vantagem de ser pouco intrusiva para o utilizador, uma vez que não existe qualquer tipo de contacto físico entre este e o dispositivo de leitura, e por a imagem do rosto de um indivíduo ser comumente utilizada em documentos de identificação. Para além disso, o equipamento necessário para a sua implementação é extremamente simples e barato, uma vez que a complexidade do sistema está contida nos algoritmos desenvolvidos [3]. Contudo, são várias as limitações associadas a esta tecnologia. Uma delas, e talvez a principal, é o facto de a face humana mudar continuamente ao longo do ciclo de vida do indivíduo. Para além disso, tal como acontece com as impressões digitais, a face pode ser danificada, alterando assim as suas propriedades e dificultando, ou até impossibilitando, a sua validação com base nos parâmetros já adquiridos. Estas limitações fazem com que possa ser necessária a actualização periódica dos descritores obtidos. Outra limitação está relacionada com a iluminação, cujas condições podem influenciar bastante a imagem recolhida [3, 5].

Reconhecimento de Íris

Outra tecnologia de autenticação biométrica, cuja utilização tem vindo a aumentar consideravelmente nos últimos anos, é o reconhecimento da íris do olho humano (figura 2.4). Tal como acontece nas impressões digitais, também o padrão da íris do olho é única para cada indivíduo, chegando a ser considerada a forma de identificação e autenticação mais segura [15]. Contudo, este sistema é bastante mais complexo, fazendo com que seja também mais dispendioso de implementar, uma vez que o tamanho da íris pode sofrer alterações, causadas pela dilatação ou contracção da pupila [13].

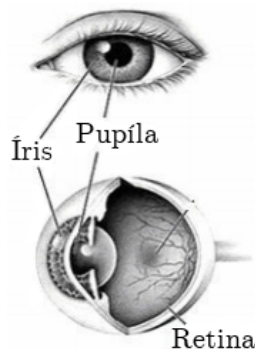


Figura 2.4: Representação do olho humano.

A imagem da íris é obtida por câmaras que utilizam luz, tipicamente no espectro do infravermelho, para adquirir uma imagem de alta qualidade da íris [4]. A luz é emitida directamente para o olho do indivíduo, que deve permanecer imóvel durante breves segundos, ao mesmo tempo que tenta manter o olhar fixo num determinado ponto. Isto faz com que seja um método extremamente intrusivo e, conseqüentemente, leva a uma baixa taxa de aceitação por parte do utilizador. Para além disso, o reconhecimento da íris pode ser problemático em pessoas que sofram de determinadas doenças, como epilepsia ou glaucoma [3].

2.3 Internet of Things

O conceito de *Internet of Things*, ou IoT, pode ser definido como sendo uma rede de equipamentos capazes de estabelecer trocas de informação entre si, com o utilizador e com aplicações existentes [1]. Apesar do seu nome, os dispositivos IoT não necessitam de ter uma conexão com a rede Internet, podendo ser utilizada uma rede local como infraestrutura para as trocas de informação [17].

Os dispositivos tipicamente associados ao conceito IoT não são os tradicionais computadores ou servidores, mas sim inúmeros sistemas electrónicos, tipicamente constituídos por sensores e actuadores e com especial foco no baixo consumo energético, com dimensões reduzidas e capazes de se conectarem a uma rede estabelecida [18]. Um exemplo da implementação deste tipo de sistemas é na transformação de uma habitação numa “Casa Inteligente”, onde todos os equipamentos são passíveis de serem controlados e monitorizados pelo utilizador através de um dispositivo com acesso à Internet ou à rede local, como mostra a figura 2.5 [1].



Figura 2.5: Exemplo do controlo remoto de uma "Casa Inteligente".

2.4 Tecnologias e Protocolos de Comunicação

São várias as tecnologias e protocolos de comunicação utilizados para estabelecer comunicações entre dispositivos, todas com as suas vantagens e limitações. Nesta secção são apresentados alguns exemplos de como é feita a comunicação entre dispositivos, sendo dada especial atenção aos métodos de baixo consumo energético.

Bluetooth

O Bluetooth é uma das principais tecnologias de comunicação sem fios de curto alcance, sendo utilizada principalmente em sistemas de baixo consumo energético. A transferência de informação é feita por rádio-frequência, numa gama de frequência a rondar os 2.4 GHz e os 2.5 GHz. Devido ao modo como é feita a comunicação entre

dispositivos, é necessário que se encontrem num espaço dentro do limite de alcance, que pode variar entre o 1 e 100 metros, dependendo da classe dos dispositivos [19, 20]. A tabela 2.1 mostra as diferentes classes de dispositivos, bem como o seu alcance e consumo energético.

Tabela 2.1: Caracterização das diferentes classes Bluetooth [20].

Classe	Alcance [m]	Consumo Energético [mW]
Classe 1	100	100
Classe 2	10	2.5
Classe 3	1	1

Em resultado do seu baixo alcance e consumo, esta tecnologia é principalmente empregue em dispositivos como auscultadores, periféricos para computador e equipamentos IoT, encontrando-se também presente na maioria dos *smartphones* [20].

Global System for Mobile Communication

Global System for Mobile Communication, ou GSM, é a tecnologia padrão utilizada nas comunicações móveis, com uma cota de mercado superior a 90% [16]. Inicialmente desenvolvido para comunicações de voz, rapidamente evoluiu de forma a permitir o envio de mensagens de texto e transferência de informação, marcando assim a passagem da tecnologia analógica para a digital na transmissão de dados na rede móvel. Consoante o local de aplicação, esta tecnologia funciona a diferentes gamas de frequência, sendo que as mais utilizadas mundialmente são 900 MHz e 1800 MHz. A privacidade da comunicação é garantida encriptando a mensagem, em conjunto com a alternância constante da frequência utilizada [16, 19].

O facto de possuir elevados níveis de segurança, associado à sua ampla cobertura, fazem do GSM uma das principais tecnologias de comunicação à escala mundial. Contudo, está severamente limitada no interior de alguns edifícios e principalmente em compartimentos subterrâneos, sendo necessária a utilização de um provedor externo para interligar dispositivos nestas circunstâncias, fazendo aumentar consideravelmente o seu custo de implementação [16].

ZigBee

ZigBee é um padrão de comunicação sem fios destinado ao controlo e monitorização de dispositivos, como relés, motores e sensores variados, caracterizados pela sua baixa taxa de transmissão [19]. Com um alcance que pode exceder os 100 metros, esta tecnologia é capaz de estabelecer trocas de informação com taxas de transferência até aos 250 kbps, dependendo da gama de frequência em que opera, que pode variar entre os 868 MHz e os 2.4 GHz. A elevada taxa de transferência, associado ao baixo consumo energético, na ordem dos 80 mW, faz desta tecnologia uma das prediletas para redes de sensores remotos que precisem de operar durante meses num mesmo ciclo de carga [19, 20].

Regra geral, uma rede ZigBee consiste num coordenador, *routers* e vários nós. O coordenador é responsável por criar a rede e gerir os nós da mesma. Uma rede deste tipo pode assumir várias topologias, como malha, árvore e estrela (figura 2.6). Numa rede

em malha, todos os dispositivos estão ligados entre si, existindo por isso vários caminhos possíveis para se estabelecer uma comunicação. Já numa rede em árvore, existe um nó principal, ao qual se ligam os nós secundários que estão ligados aos restantes nós. Por fim, numa rede em estrela todas as comunicações entre nós passam pelo coordenador, que encaminha a informação para o nó de destino. Em qualquer dos casos, uma rede ZigBee é capaz de suportar até cerca de 65500 nós [16, 19].

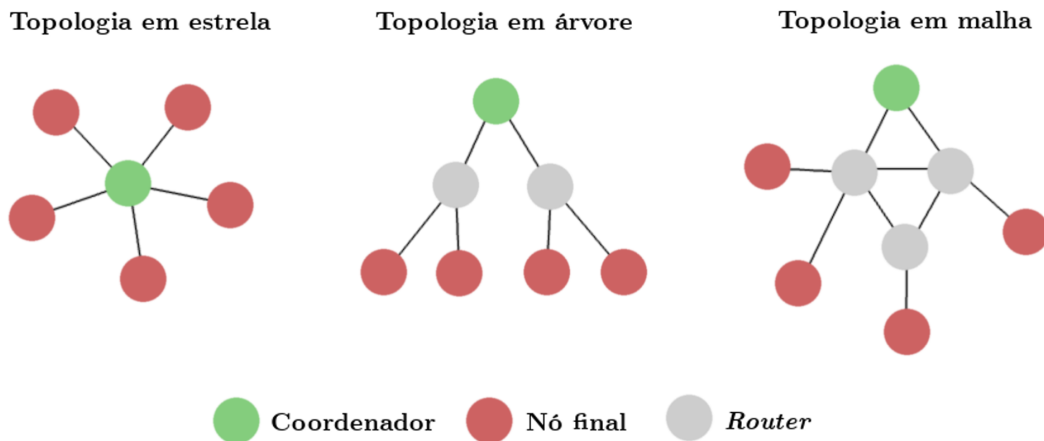


Figura 2.6: Topologias de uma rede ZigBee.

Contudo, um dos aspectos mais negativos deste tipo de redes é o facto de um dispositivo ZigBee não ser capaz de comunicar com outro não-ZigBee sem o recurso de interfaces adicionais. Esta limitação leva a que sejam procuradas alternativas sempre que seja necessário que estes dispositivos comuniquem com computadores, *smartphones*, ou outros dispositivos que não estejam preparados para o efeito [20].

Wi-Fi

A tecnologia Wi-Fi, baseada no protocolo 802.11 WLAN, permite a criação de redes sem fios e comunicação entre dispositivos via ondas de rádio. Apesar de hoje o termo estar largamente associado à Internet, a verdade é que este tipo de redes não impõe nenhuma ligação a uma rede exterior. O protocolo 802.11 define apenas um conjunto de regras da comunicação *wireless* entre dispositivos, cuja banda de frequência pode variar entre os 2.4 GHz e os 5 GHz, conforme a variante do protocolo usada [20]. Da mesma forma, de acordo com a frequência utilizada, tanto o alcance da rede como a taxa de transferência podem variar consoante a versão utilizada.

Como já foi referido, a tecnologia Wi-Fi encontra-se muito associada ao acesso à rede Internet, e em particular com o protocolo TCP, que será descrito de seguida. Contudo, particularmente no universo IoT, até há poucos anos eram poucos os microprocessadores capazes de executar acções de codificação e decodificação das mensagens usadas por este protocolo (TCP). Com o avanço tecnológico, no entanto, hoje em dia praticamente todos os fabricantes de componentes electrónicos para circuitos impressos possuem uma linha de módulos especializados para esta função, possibilitando o acesso a este tipo de redes por parte de sistemas embutidos [20].

Transmission Control Protocol

Transmission Control Protocol, ou TCP como é normalmente conhecido, é um protocolo de transferência de informação que serve de base à maioria das comunicações sem fios. A principal premissa deste protocolo é a sua fiabilidade de entrega da mensagem, em contraste com outros protocolos que se focam mais com a velocidade e o *timing* de entrega, fazendo deste protocolo o predilecto quando é necessário garantir que a mensagem chega ao destino.

Depois de uma mensagem ser enviada, o receptor deve confirmar a sua recepção e, caso tal não aconteça, ou se a mensagem for corrompida, esta será reenviada de forma automática [21, 22].

User Datagram Protocol

Ao contrário do protocolo descrito acima, o protocolo UDP não possui nenhum mecanismo de controlo de fluxos, fazendo com que não seja uma solução viável quando a garantia da recepção da mensagem é de maior prioridade [20, 22]. Em vez disso, este protocolo foca-se mais na velocidade de entrega da mensagem, sendo por isso utilizado quando é necessário enviar grandes quantidades de informação num curto espaço de tempo [21]. Um exemplo disso é o *streaming* de vídeos, onde a perda de alguns *bytes* de informação não é relevante, uma vez que a prioridade é fazer com a informação seja enviada atempadamente, principalmente tendo em atenção ao crescente aumento de qualidade dos vídeos [23].

A imagem da figura 2.7 procura exemplificar as diferenças no processo de comunicação entre estes dois protocolos (TCP e UDP).

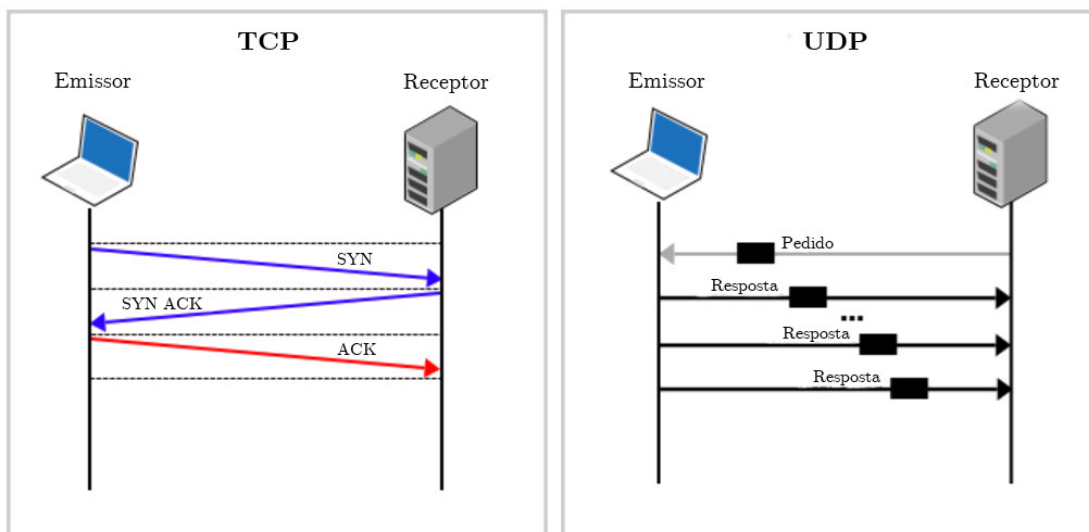


Figura 2.7: Protocolos de comunicação: TCP *vs* UDP.

2.5 Segurança de Redes Wi-Fi

Uma vez que as redes Wi-Fi transmitem informação via ondas rádio, existe o risco de esta ser interceptada por terceiros. Devido à sua natureza, não é possível implementar as mesmas soluções de autenticação e autorização de utilizadores que são aplicadas às redes cabladas. Assim, de forma a garantir que a informação transmitida não é interceptada por terceiros, e uma vez que uma rede sem fios é naturalmente mais vulnerável que uma rede cablada, é necessário implementar medidas de segurança que previnam ataques à rede [24].

Nesta secção são apresentados alguns dos mecanismos de segurança tipicamente aplicados, que visam mitigar as vulnerabilidades inerentes a este tipo de redes.

2.5.1 Wired Equivalent Privacy

WEP foi o primeiro algoritmo de encriptação aplicado às redes sem fios. Este visa proteger as comunicações de escutas por parte de terceiros, garantir a integridade da mensagem, bem como prevenir o acesso à rede por parte de utilizadores não autorizados. É importante referir que este algoritmo não protege as comunicações ponto-a-ponto, mas sim os pacotes de informação enviados [25].

A integridade e confidencialidade da mensagem é garantida através do uso de uma chave partilhada pelo emissor e receptor da mensagem. Contudo, esta chave pode ser facilmente decifrada, o que pode resultar em perdas ou alterações da informação transmitida. Para além disso, a mesma chave é utilizada durante largos períodos de tempo o que, associado à curta dimensão das mesmas (40 bits), deixa o sistema susceptível a uma busca exaustiva de chave por parte do atacante [25, 26]. Outras falhas na segurança deste sistema são a sua vulnerabilidade a ataques *man-in-the-middle*¹ e de negação de serviço, onde a rede é sobrecarregada, impossibilitando o seu acesso por parte de utilizadores legítimos [24].

2.5.2 Wi-Fi Protected Access

Devido às limitações do algoritmo anterior, que tem sido alvo de fortes críticas desde a sua primeira implementação, surgiu a necessidade de criar um novo que fosse capaz de implementar soluções de maior segurança nas comunicações sem fios. Assim surgiu o WPA, cujo principal objectivo era resolver os problemas de criptografia do WEP [26].

A principal diferença entre estes dois algoritmos prende-se com o facto de num sistema onde o WPA esteja implementado ser utilizada uma chave única, de 128 bits de dimensão, para cada mensagem, reduzindo assim a facilidade de descodificação da mesma. Para além disso, o controlo de integridade das mensagens enviadas passa a contemplar também a ordem das mesmas, o que não acontece no WEP [24]. Este protocolo integra ainda um sistema de verificação de integridade da mensagem, denominado MIC, como forma de determinar se uma mensagem foi adulterada antes de chegar ao destino.

Ainda assim, uma vez que este protocolo é baseado no WEP, não é capaz de resolver todas as falhas de segurança, sendo de notar que continua a ser susceptível a ataques de negação de serviço [24].

¹*Man-in-the-middle* é um tipo de ataque onde um intruso se coloca no meio de uma comunicação entre dois dispositivos, podendo interceptar e modificar as mensagens em trânsito.

2.5.3 Wi-Fi Protected Access 2

Este protocolo, que pode ser visto como uma versão melhorada do WPA, visa melhorar os seus níveis de segurança, incorporando na íntegra o protocolo 802.11i. Assim, o WPA2 utiliza um novo algoritmo de encriptação, designado AES [24]. Para além disso, de forma a garantir a autenticação dos clientes, é implementado um servidor DHCP² para atribuição de endereços IP. Contudo, este protocolo continua a ser susceptível a ataques de negação de serviço, para além de ser necessária a utilização de equipamentos que suportem o algoritmo AES, o que não acontece com o WPA que utiliza os mesmos equipamentos que o sistema WEP [29].

2.6 Bases de Dados

As bases de dados são uma parte estruturante de qualquer sistema de informação. De uma forma resumida, uma base de dados nada mais é do que um conjunto de informações e das relações que existem entre si. Para se conseguir gerir e manipular a informação nelas contida, são utilizados sistemas de gestão de bases de dados, que permitem guardar e gerir a informação de uma forma eficiente, salvaguardando-a também eventuais perdas [30, 31].

2.6.1 Modelos de Bases de Dados

Existem quatro categorias distintas de modelos de bases de dados: hierárquicas, em rede, relacionais e *object-oriented*. A primeira é a forma de bases de dados mais antiga, cuja estrutura assume a forma de uma árvore invertida. Nela, existe uma tabela que actua como a raiz da árvore enquanto que as restantes assumem o papel de ramos. Este tipo de base de dados é ainda caracterizado pela relação entre tabelas ser do tipo *parent/child*, onde uma tabela *parent* pode estar associada a várias tabelas *child*, e estas últimas apenas podem estar relacionadas com uma tabela *parent*. A sua estrutura simples permite uma grande rapidez no acesso a registos, não possibilitando, contudo, relações mais complexas entre as informações nela contida [31].

As bases de dados em rede têm como objectivo eliminar as restrições do modelo hierárquico, permitindo que um dado registo esteja envolvido em várias relações. Este modelo permite assim a relação entre várias tabelas, por meio de referências onde habitualmente estariam entradas de dados, formando desta forma uma rede de informação. Uma das principais vantagens deste modelo é a possibilidade de criar relações do tipo muitos para muitos, onde um dado registo pode referenciar vários outros ao mesmo tempo que é referenciado por outros registos [31, 32].

O modelo relacional de bases de dados é o mais utilizado actualmente, principalmente em soluções comerciais, e é caracterizado pela sua simplicidade e base matemática, tendo como base a teoria dos conjuntos e a álgebra relacional [30, 32]. Em resultado da sua vasta aplicação, este modelo será abordado em maior pormenor na secção que se segue.

O modelo *object-oriented* é um modelo mais complexo e de difícil utilização, sendo também por isso utilizado apenas em aplicações especializadas. Neste tipo de bases de

²DHCP é um protocolo que automatiza o processo de configuração de dispositivos numa rede TCP/IP [28].

dados, a informação é representada sob a forma de objectos, o que possibilita a manipulação de dados complexos, para além de ser capaz de suportar informação abstracta como áudio, vídeo e informação geográfica [32, 33].

2.6.2 Bases de Dados Relacionais

A estrutura fundamental do modelo relacional é a tabela. Esta é formada por diferentes conjuntos de linhas e colunas, onde a cada coluna corresponde um atributo diferente, único e indivisível, e a cada linha corresponde um registo, também conhecido por tuplo. O domínio de um atributo é o conjunto de valores que este pode assumir, seja sob a forma de um número inteiro, uma entrada de texto ou até um valor nulo.

Uma base de dados relacional deve ainda ter em conta que a informação nela contida deve ser única, ou seja, não deve existir informação repetida, nem que seja em tabelas diferentes. Para se conseguir relacionar registos em tabelas distintas, são utilizadas chaves que identificam de forma única os registos, e são responsáveis por estabelecer as relações entre tabelas [30, 31, 33].

Chaves Primárias e Estrangeiras

Uma chave primária é formada por um único atributo que pode ser, por exemplo, o seu número de identificação. Por vezes pode ser necessário utilizar mais que um atributo na constituição de uma chave, no caso de os valores do mesmo se repetirem, mas esta abordagem deve ser evitada sempre que possível.

Assim sendo, na selecção de uma chave primária deve-se ter em atenção as seguintes características: a chave deve ser única, ou seja, o seu valor não deve ser repetido na tabela; se existir um atributo que permita formar uma chave primária, este deve ser utilizado ao invés de chaves constituídas por múltiplos atributos; o atributo que constitui a chave primária deve ser *não-nulo*, pois caso isto não se verifique, não será possível identificar o registo; a chave não deve ser actualizável, de forma a não afectar relações já estabelecidas [31, 32].

As chaves estrangeiras são utilizadas para representar as relações entre registos de tabelas distintas. Isto é o mesmo que dizer que uma chave estrangeira corresponde à chave primária de uma outra tabela. Devido à importância que as chaves assumem neste modelo de bases de dados, torna-se evidente que seja necessário salvaguardar a integridade das mesmas [30, 31].

Tipos de Relações entre Tabelas

Como já foi referido, as relações entre registos de tabelas distintas é alcançado através da empregabilidade de chaves únicas de identificação. Porém, o tipo de relação entre estes pode assumir várias formas. O tipo de relação criada depende do modo como as colunas relacionadas são definidas. Existem três tipos de relações entre tabelas (figura 2.8) [31, 32]:

- *Um para um* (1:1): este tipo de relação apenas permite que a um registo de uma tabela A corresponda um e um só registo da tabela B. A utilização deste tipo de relações é pouco comum em implementações comerciais, uma vez que é mais simples colocar os dois registos na mesma tabela.

- *Um para muitos* (1:N): esta relação também é conhecida por *muitos para um* (N:1), sendo que o que as distingue é o ponto de vista, e significa que a um registo da tabela A podem corresponder 0, 1 ou vários registos da tabela B. Devido à sua versatilidade, e ao grande número de situações em que pode ser utilizada, este tipo de relações é um dos mais utilizados.
- *Muitos para muitos* (M:N): esta é a relação mais complexa das bases de dados relacionais, permitindo que um tuplo da tabela A tenha várias correspondências na tabela B, acontecendo o mesmo para os registos da tabela B. Um exemplo prático deste tipo de relações é o seguinte: um aluno pode frequentar várias disciplinas, mas as disciplinas são elas próprias frequentadas por vários alunos.

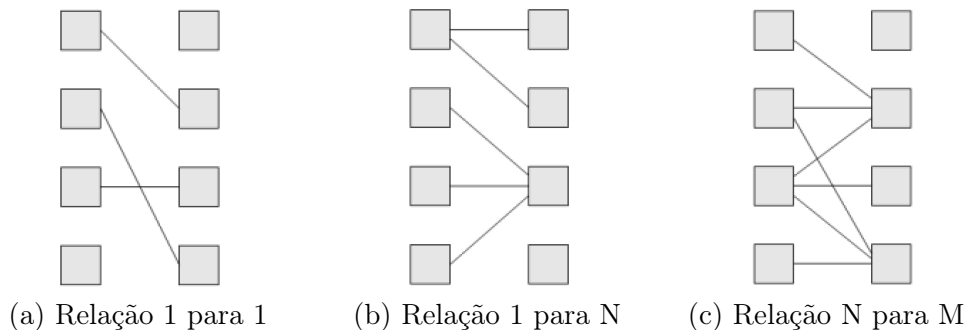


Figura 2.8: Representação esquemática dos diferentes tipos de relações [30].

2.7 Aplicações WEB

Uma aplicação *web* é uma aplicação projectada para ser utilizada através de um *browser*, que actua como cliente e que envia pedidos em HTTP para um servidor remoto, tipicamente via Internet. Neste tipo de aplicações, as operações lógicas da mesma encontram-se distribuídas entre o cliente e o servidor, sendo que a maioria da informação é guardada no servidor. Uma das principais vantagens deste tipo de aplicação, e a de maior importância no âmbito desta dissertação, é o facto de ser agnóstica ao sistema operativo onde corre o cliente, podendo ser acedida e utilizada em qualquer plataforma com acesso a um navegador de Internet [34].

2.7.1 Front-end

O *front-end* da aplicação é a parte da mesma que corre no lado do cliente. O seu desenvolvimento consiste, essencialmente, na criação de uma interface através da qual o utilizador interage com a aplicação. As principais ferramentas utilizadas na criação da interface referida são: HTML, CSS e JavaScript. Utilizando estas ferramentas é possível desenvolver o mais variado tipo de páginas, com um sem-número de acções e interacções possíveis [35].

HTML

HTML é uma linguagem de marcação que define a estrutura da página apresentada através da utilização de *tags*, que são utilizadas para definir cabeçalhos, tabelas e blocos, entre outros. O texto escrito neste formato é interpretado pelo *browser*, de forma a que o resultado seja apresentado no dispositivo devidamente formatado e estruturado [36].

CSS

De forma a dar estilo e permitir a personalização das páginas *web* é utilizada a linguagem de estilo CSS. Esta permite afinar até ao mais pequeno pormenor no design dos elementos gráficos da página, tais como *fonts*, cores e contornos. Esta linguagem foi desenvolvida com o objectivo de forçar a separação entre a estrutura de uma página e o estilo dos seus objectos, permitindo assim que várias páginas possuam o mesmo estilo, especificado num ficheiro *.css* em separado [37].

JavaScript

JavaScript é a linguagem que permite tornar uma página, por natureza estática, numa página dinâmica. Esta linguagem de programação, caracterizada por ser bastante leve, foi desenvolvida para correr directamente no *browser*, permitindo a incorporação de funções nas páginas HTML. Apesar de ser tradicionalmente utilizada no lado do cliente, hoje em dia existem já ferramentas que a permitem utilizar também na construção do servidor *web* [36, 37].

2.7.2 Back-end

O desenvolvimento do *back-end* de uma aplicação refere-se à implementação da mesma no lado do servidor, cujo foco primário passa pela aplicação da lógica da mesma ou, por outras palavras, define como esta funciona. O servidor, que trata da informação recebida do lado do cliente, tem uma relação muito íntima com a base de dados, uma vez que é responsável por gerir e guardar a maioria da informação da aplicação.

O servidor da aplicação estabelece comunicações com o cliente seguindo o protocolo HTTP, segundo o qual o cliente faz um pedido ao servidor, que por sua vez responde com a informação pedida (figura 2.9). Este protocolo impõe uma série de regras que devem ser cumpridas: o servidor só deve responder ao pedido feito pelo cliente; deve responder a todos os pedidos efectuados, nem que seja com uma mensagem de erro; o pedido HTTP, ou HTTP *request*, deve conter o endereço URL que indica o caminho para o qual o pedido deve ser enviado [34, 35].

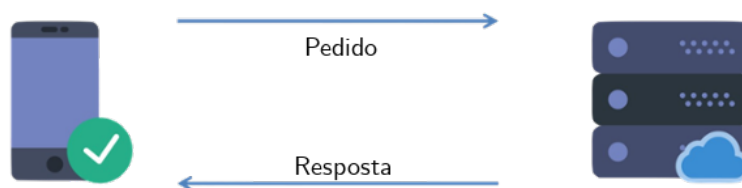


Figura 2.9: Comunicação cliente-servidor segundo o protocolo HTTP.

A parte lógica da aplicação envolve todo o processamento dos pedidos e envios de informação, para além de guardar a informação na base de dados e tomar decisões com base na informação requisitada, fazendo os *queries* necessários para recuperar a informação da base de dados. Para além disso, é responsável pela aplicação de medidas de segurança, tais como a identificação e autenticação do utilizador que pretende aceder a determinada informação [36, 38].

São várias as ferramentas existentes que visam facilitar o desenvolvimento do *back-end* de uma aplicação *web*, desenvolvidas nas mais várias linguagens de programação, como Java e PHP. Ainda assim, devido ao constante crescimento de *frameworks open source* e das comunidades envolventes, linguagens alternativas como Ruby, Python e até mesmo JavaScript têm vindo a ser cada vez mais utilizadas nesta área de desenvolvimento [34, 39].

Capítulo 3

Trabalhos Relacionados

No presente capítulo são apresentados os trabalhos previamente realizados no âmbito do sistema CLASSQUIZ, bem como algumas soluções alternativas existentes no mercado.

3.1 Sistemas de Gestão de Aprendizagem

Existem vários sistemas que permitem a avaliação electrónica de alunos existentes no mercado. Estes sistemas, designados Sistemas de Gestão de Aprendizagem (LMS), já se encontram implementados na maioria das Universidades, como é o caso do Moodle. Para além de servirem para realizar testes de avaliação, estes sistemas permitem também a partilha de conteúdos, a avaliação de Unidades Curriculares e docentes, o contacto entre professores e alunos e até a frequência de unidades curriculares exclusivas *online*, facilitando assim o ensino à distância [40, 41].

A partilha de conteúdo é, porventura, a forma de utilização mais comum destes sistemas, devido à facilidade em os professores partilharem livros, *slides* das aulas ou exercícios. É também por este meio que, normalmente, os docentes partilham informações com os alunos relativamente a trabalhos ou modelos de avaliação [40].

Outra forma de utilização bastante comum destes sistemas é a avaliação dos alunos, que pode assumir uma de duas formas: realização de testes *online* ou submissão de trabalhos realizados. Esta forma de utilização dos sistemas LMS é mais usada, tipicamente, por professores mais aptos para o uso de tecnologias e aplicações computacionais. Das duas vertentes aqui descritas, a mais comum é sem dúvida a submissão de trabalhos, uma vez que facilita bastante o processo, para além de aumentar a garantia de recepção dos mesmos [42, 43, 44].

A utilização dos Sistemas de Gestão de Aprendizagem como forma de avaliação de Unidades Curriculares e docentes é igualmente comum, principalmente se for feita de forma anónima, uma vez que incentiva à participação dos alunos. O anonimato é um factor importante de forma a garantir que os inquiridos não sintam receio de possíveis represálias ao avaliarem os seus professores [44].

A discussão de assuntos relacionados com as aulas, sejam para promover o debate de um determinado tema, seja como forma de esclarecimento de dúvidas, acaba por ter pouco destaque. Isto deve-se principalmente à falta de interactividade social que estes sistemas proporcionam. De facto, esta é uma das principais limitações destes sistemas, já que a interactividade social é um dos grandes motivadores para a aprendizagem [40, 45].

Por fim, a realização de aulas por via destas plataformas é utilizada, na sua esmagadora maioria, na realização de cursos à distância, e em que não é possível leccionar aulas presenciais. Este tipo de aulas podem ser leccionadas em tempo real, ou serem gravadas num vídeo, que pode ser acedido em qualquer altura por parte dos alunos. Como já foi referido, a falta de interactividade de que estes sistemas sofrem levam a que os resultados obtidos não se consigam equiparar aos obtidos em aulas presenciais [40].

Apesar de todas as vantagens que estes sistemas apresentam, a sua utilização está muito ligada ao ensino à distância e via Internet, não sendo, portanto, indicados para complementarem uma aula presencial, uma vez que não existe forma de restringir a sua utilização a um espaço físico delimitado. Para além disso, o facto de o ser conteúdo ser acedido através de um *browser* obriga a que os alunos possuam dispositivos computacionais modernos (computadores e/ou *smartphones*) o que, embora seja a tendência nos países mais desenvolvidos, não deve ser tido como garantia por parte da instituição educativa.

3.2 QuizXpress

Existem várias soluções comerciais de sistemas de inquéritos electrónicos. Um destes sistemas é o QuizXpress, que visa simular os sistemas de inquéritos utilizados em concursos televisivos, sendo mesmo utilizado num concurso da televisão grega. Como tal, este sistema permite a realização de inquéritos com um grupo alargado de participantes, tendo como principal foco a realização de jogos, embora possa ser também utilizado, segundo os seus desenvolvedores, na educação, particularmente dentro de uma sala de aula. Contudo, este sistema não oferece garantias de autenticação dos participantes, pelo que a sua utilização em salas de aulas está confinada à realização de questionários sem fins avaliativos. A informação contida nesta secção foi obtida a partir do manual de utilização do sistema QuizXpress [46], bem como da sua página *online* [47].

À semelhança do sistema proposto nesta dissertação, o QuizXpress é composto por duas partes: uma aplicação e um conjunto de controladores remotos. A aplicação deste sistema corre num computador, estando contudo limitado ao sistema operativo Windows (versões 7, 8 e 10). Esta aplicação permite a criação e realização de inquéritos, através de uma interface gráfica que oferece ao utilizador uma variedade de diferentes *templates* pré-feitos (figura 3.1).

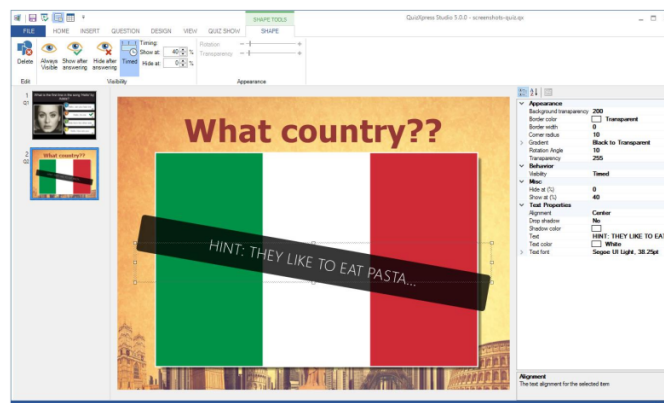


Figura 3.1: Janela de criação de inquéritos do sistema QuizXpress.

Uma vez que o principal fim deste sistema é a realização de jogos do tipo *quiz*, a aplicação do mesmo vem apetrechada de uma série de funcionalidades, como a possibilidade de realizar inquéritos em equipa e de detectar qual o participante mais rápido, para além de permitir, ao utilizador da aplicação, criar um conjunto de regras de validação de respostas para um dado inquérito. A figura 3.2 exemplifica um jogo, do género do popular “Quem Quer Ser Milionário”, a ser executado no sistema descrito nesta secção.



Figura 3.2: Exemplo de um inquérito realizado no sistema QuizXpress.

No que toca aos controladores remotos, este sistema apresenta uma variedade de soluções, desde a utilização de *smartphones* como comando, a diferentes terminais proprietários vendidos pela marca. Para além destes, o sistema é ainda compatível com o *buzzer* desenvolvido pela Sony para o famoso jogo Buzz!TM(figura 3.3).



(a) Aplicação para *smartphone* (b) Controladores QuizXpress (c) Controlador Sony

Figura 3.3: Diferentes tipos de controladores do sistema QuizXpress.

No caso da utilização de um *smartphone* como controlador remoto, a comunicação entre este e a aplicação dá-se via Internet, ao passo que no caso dos controladores dedicados, tanto da QuizXpress como da Sony, a comunicação dá-se por rádio-frequência (a 2.4 GHz), sendo necessário um adaptador ligado ao computador.

Assim, o sistema QuizXpress apresenta-se como uma solução que, do ponto de vista da implementação, se assemelha bastante ao sistema CLASSQUIZ idealizado, apesar das suas limitações no que à identificação dos participantes diz respeito. Este sistema apresenta várias soluções que podem servir de referência ao desenvolvimento do sistema proposto nesta dissertação, principalmente no que toca às funcionalidades que a aplicação proporciona, como a possibilidade de exportar resultados, ou a possibilidade de realizar pequenos jogos, já que o sistema CLASSQUIZ aqui proposto não se limita à realização de momentos de avaliação.

3.3 Versões Anteriores do Sistema CLASSQUIZ

Ao longo dos últimos anos, têm vindo a ser realizados trabalhos na Universidade de Aveiro que visam contribuir para o desenvolvimento do sistema CLASSQUIZ. O desenvolvimento deste sistema teve início em 2017, pelas mãos de António Teixeira [48], tendo sido posteriormente iterado por Manuel Mamede [49]. Apesar de os estudos e conclusões obtidas pelo primeiro terem permitido obter um foco mais preciso sobre as necessidades e limitações do sistema, pouco ou nada resta do sistema por si desenvolvido, tendo este sido completamente reformulado pelo seu sucessor.

Nesta secção serão demonstrados os trabalhos desenvolvidos por ambos, de forma a permitir um melhor enquadramento do sistema e das suas minúcias, bem como fazer um ponto da situação do mesmo à data de início da presente dissertação.

3.3.1 Primeira Versão do Sistema

O trabalho desenvolvido por António Teixeira idealizava um sistema composto por terminais individuais (um por cada aluno) e uma estação receptora. A autenticação do aluno seria realizada através do seu cartão de estudante e foi estudada a hipótese de a comunicação entre o terminal individual e a estação base-receptora se dar por rádio-frequência.

Nesta primeira versão do sistema, a estação receptora era composta por um micro-controlador Arduino Mega 2560 e um receptor de rádio-frequência, sendo alimentada por cabo a um computador. Este trabalho não contemplava a existência de uma interface gráfica de controlo do sistema, tendo-se focado principalmente na comunicação entre dispositivos.

O terminal desenvolvido era composto por um micro-controlador Arduino UNO, um módulo para leitura de cartões RFID, 5 botões e um módulo transmissor de rádio-frequência. Para além disso, a alimentação do dispositivo era conseguida através de uma bateria de 9 V, ligada ao microcontrolador, que por sua vez alimentava os restantes componentes, já que o leitor de cartões utilizado operava a 3.3 V e o transmissor a 5 V. Para o funcionamento dos botões foram utilizadas resistências *pull-down* de 100 Ω , como mostra a figura 3.4.

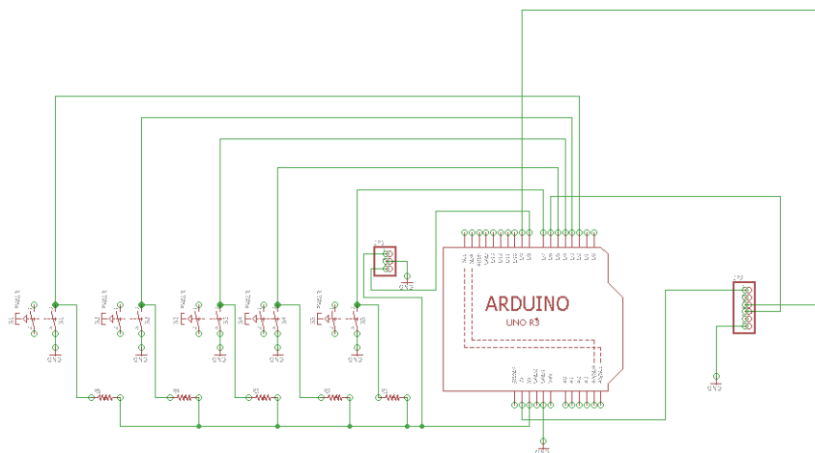


Figura 3.4: Esquema eléctrico do terminal [48].

No que à comunicação entre dispositivos diz respeito, em resultado dos testes realizados, o autor verificou que a alimentação do receptor de rádio-frequências é afectada por interferências alheias ao sistema, influenciando a comunicação entre o transmissor e o receptor, e levando a que por vezes este não fosse capaz de descodificar a informação enviada, inviabilizando desta forma a utilização do sistema [48].

3.3.2 Segunda Versão do Sistema

A segunda versão do sistema CLASSQUIZ é sustentada pela mesma ideia da sua antecessora, ou seja, o sistema é composto por terminais individuais, empregues aos alunos, que comunicam com uma estação base-receptora.

Como ficou demonstrado pelo trabalho de António Teixeira [48], uma comunicação entre dispositivos realizada por rádio-frequência acarreta fortes limitações em relação à integridade da mensagem, sendo por isso necessária uma solução alternativa. Assim, a solução apresentada por Manuel Mamede [49] prevê uma comunicação entre dispositivos via Wi-Fi (secção 2.4) e uma aplicação gráfica a correr na estação base (este último item havia sido deixado em aberto no trabalho anterior), que passou a ser constituída por um computador ao invés de um microcontrolador.

O protocolo de comunicação *wireless* adoptado pelo autor é o UDP o que, como foi referido na secção 2.4, não garante a confirmação da recepção da mensagem, tendo inclusive sido esta uma das considerações finais do autor, alertando para a necessidade de adopção de um protocolo com maior fiabilidade.

A aplicação, desenvolvida em Visual Basic para a estação base-receptora, assume apenas um papel de controlo e monitorização, não tendo sido implementada com vista a uma utilização em ambiente realista. A imagem da figura 3.5 demonstra a aplicação em funcionamento durante uma comunicação com um terminal individual.

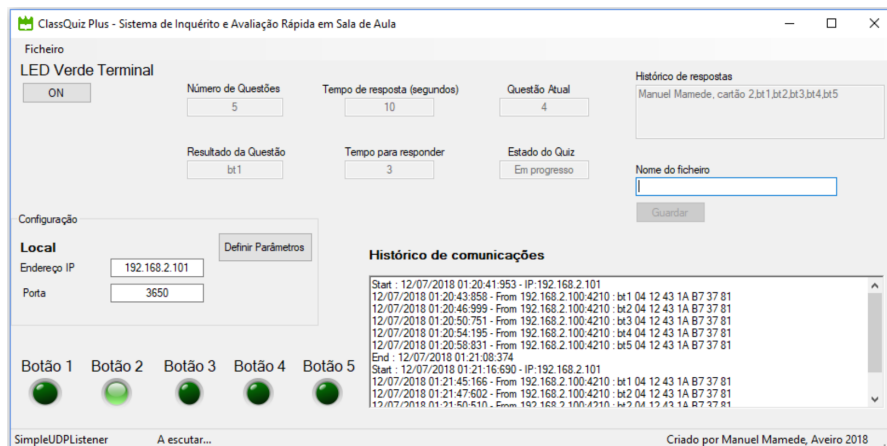


Figura 3.5: Interface gráfica da aplicação desenvolvida por Mamede [49].

Uma vez que, em relação ao trabalho anteriormente desenvolvido, o modelo de comunicação foi alterado, também o terminal individual foi alvo de formulações. O microcontrolador embutido no mesmo necessita agora de ter um módulo de comunicações Wi-Fi. Um dos módulos mais comuns em IoT é o ESP8266, podendo este ser adquirido com módulo externo para sistemas existentes, ou integrado em alguns microcontroladores. O microcontrolador adoptado pelo autor é o NodeMCU v3, caracterizado pelas suas

dimensões reduzidas e capacidades semelhantes ao Arduino UNO utilizado na primeira versão do sistema. O módulo ESP8266 segue ainda as normas 802.11b/g/n, permitindo assim trocas de informação a taxas de transferência bastante elevadas.

Este microcontrolador é alimentado a 5 V, possuindo saídas de 3.3 V para alimentar os restantes componentes do sistema. Contudo, como evidenciado pelo autor, os I/O's do microcontrolador não são suficientes para acomodar o módulo RFID em conjunto com 5 botões dedicados. Como forma de solucionar este problema, os botões encontram-se ligados a um *priority encoder* de 8 bits, de forma a codificar os 5 possíveis sinais de entrada em 3, que corresponde precisamente ao número de I/O's livres do microcontrolador. Na imagem da figura 3.6 é possível ver os vários componentes constituintes do terminal individual desenvolvido, ao passo que a figura 3.7 esquematiza as ligações do sistema.

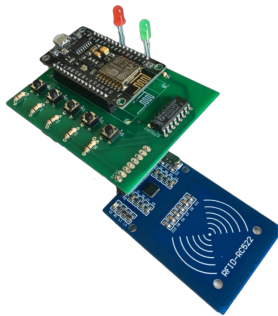


Figura 3.6: Sistema desenvolvido por Mamede [49].

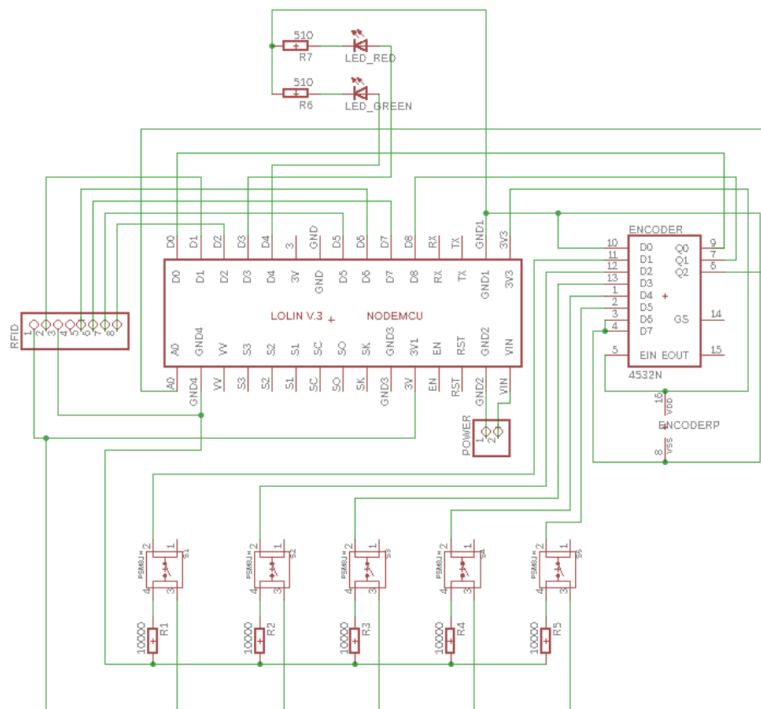


Figura 3.7: Esquema eléctrico do sistema [49].

O terminal apresentado nesta secção é muito semelhante ao implementado no âmbito desta dissertação. Assim, a sua constituição e modo de funcionamento serão temas abordados mais a frente no presente documento. Uma questão que esta versão do sistema não aborda, no entanto, é a alimentação do terminal individual. Como já foi referido, o sistema é alimentado a 5 V, o que pode acontecer por via dos pinos dedicados, ou através de um cabo USB. Esta é uma das principais modificações efectuadas ao sistema desenvolvido por Manuel Mamede, no que ao *hardware* do terminal individual diz respeito.

Capítulo 4

Solução Proposta

Neste capítulo são apresentadas as tecnologias seleccionadas para o desenvolvimento do sistema CLASSQUIZ. Numa primeira instância, são apresentados os métodos de autenticação adoptados, seguindo-se uma reflexão sobre o modelo de comunicação entre dispositivos e, por fim, serão expostas e justificadas as ferramentas utilizadas para o desenvolvimento da aplicação *web*. O princípio de funcionamento da solução proposta encontra-se representado na figura 4.2 para consulta.

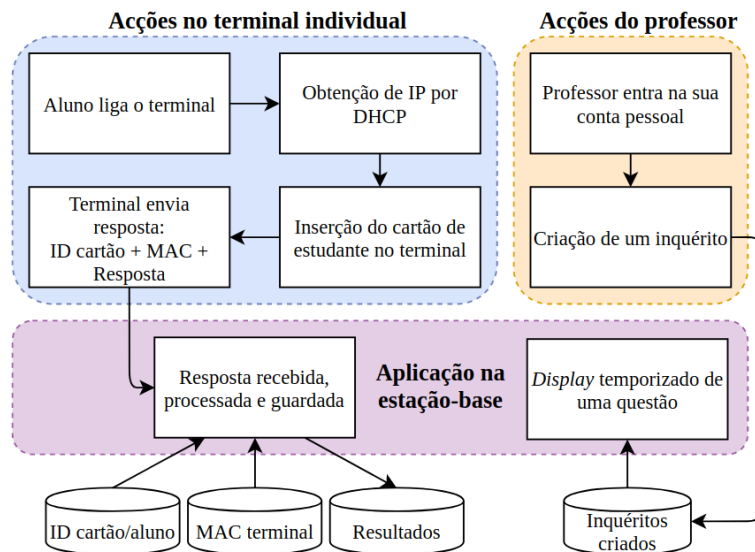


Figura 4.1: Diagrama conceptual do sistema.

4.1 Autenticação dos Utilizadores

Tratando-se este de um sistema que visa ser utilizado na avaliação de alunos, a segurança e fiabilidade do mesmo é um tema de maior importância no seu desenvolvimento e implementação. Torna-se por isso necessário adoptar métodos de autenticação do utilizador que garantam a veracidade da identificação do mesmo.

Como foi descrito na secção 2.2, os sistemas biométricos são a solução que melhor garantem que o utilizador é na realidade quem afirma ser, mas são também os mais difíceis

de implementar, principalmente por existir a necessidade de o sistema ser compacto e de baixo custo de produção, para além de serem também os mais intrusivos para o utilizador, tendo conseqüentemente um menor grau de aceitação pelo mesmo. Restam então os métodos tradicionais enunciados, que podem ser aplicados em situações distintas.

A autenticação por palavra-chave apresenta-se como uma solução viável no acesso à aplicação da estação base-receptora, cujos utilizadores são os docentes das Unidades Curriculares, uma vez que estes se assumem como utilizadores responsáveis, minorando o risco de partilha de senhas. Já para os utilizadores dos terminais individuais (alunos), a autenticação por RFID assume-se como a solução mais atraente, quer seja devido à sua facilidade de implementação (como ficou evidenciado nos trabalhos realizados por António Teixeira e Manuel Mamede), quer por dificultar a partilha de credenciais, sendo este um dos principais riscos da avaliação electrónica.

4.2 Modelo de Comunicação

A comunicação entre dispositivos deve ser por Wi-Fi, como ficou demonstrado nos trabalhos apresentados na secção 3.3, de forma a garantir a integridade da mensagem enviada. Contudo, tal como Mamede refere no trabalho que realizou [49], o protocolo adoptado deve confirmar a recepção da mensagem. Assim sendo, e uma vez que não basta que o aluno responda a uma questão, é também necessário que a estação base receba a resposta, a opção pelo protocolo TCP torna-se óbvia.

4.3 Ferramentas Utilizadas

Nesta secção são apresentadas as ferramentas adoptadas para a gestão de uma base de dados, bem como para o desenvolvimento da aplicação *web*, quer para o lado do servidor, quer para o lado do cliente.

4.3.1 MySQL

O sistema de gestão de bases de dados MySQL permite construir e utilizar bases de dados relacionais, utilizando a linguagem SQL como interface. Este apresenta algumas vantagens valorizadas no âmbito desta dissertação, como a sua simplicidade de utilização e elevada velocidade de resposta, para além de ser um dos sistemas de gestão de bases de dados mais utilizados mundialmente, o que resulta numa forte fonte de informação disponível *online*. Para além disso, existe uma versão gratuita do sistema MySQL, o que é altamente valorizado neste projecto [30, 50].

Existem, contudo, outras alternativas interessantes, tais como SQLite e PostgreSQL. O primeiro é um sistema de gestão de bases de dados muito simples, mas que se encontra limitado pelo facto de funcionar apenas localmente, não permitindo, por isso, ligações remotas. Esta limitação levanta algumas questões relacionadas com a segurança da informação guardada, uma vez que a base de dados, e conseqüentemente toda a informação nela armazenada, consistem num único ficheiro, que pode ser apagado ou substituído, ainda que acidentalmente, caso tenham permissão para tal [30].

O sistema PostgreSQL é um sistema bastante mais robusto, que excele na realização de tarefas mais complexas. Este é um sistema *open-source* bastante utilizado em sistemas e aplicações de grandes dimensões, graças à sua elevada taxa de leitura/escrita. Contudo,

apesar de oferecer mais funções e opções do que o MySQL, este é também um sistema cuja utilização é bastante mais complexa [50]. Para além disso, não é tão popular como o sistema adoptado, pelo que a informação disponível é mais escassa, potenciando assim o surgimento de dificuldades durante o desenvolvimento do projecto. Para além disso, neste projecto não são necessárias as potencialidades extra que este sistema oferece em relação ao MySQL.

A escolha recai então sobre o sistema MySQL, quer pelo elevado grau de segurança de proporciona, quer pela sua fácil utilização e vasta documentação existente.

4.3.2 Django

Django é uma *framework* desenvolvida em Python, e é baseado no padrão de arquitectura modelo-vista-controlador. Neste padrão, o modelo representa os dados da aplicação (de uma forma simplificada, um modelo corresponde a uma tabela na base de dados), enquanto que a vista é responsável por exprimir o estado do modelo, através da aceitação de pedidos e consequente envio de respostas (a vista está muitas vezes associada a uma janela de visualização, mas não é mandatário que tal aconteça). Por fim, o controlador é responsável pela interpretação das acções de entrada do utilizador, e o respectivo envio para o modelo e para a vista [51].

Esta *framework* é ainda caracterizada pela filosofia empregue pelos seus criadores, da qual se destacam os seguintes princípios [52]:

- **Menos código** - assenta na ideia de que mais código traduz-se em mais erros, o que se traduz num maior tempo de desenvolvimento;
- **Menos duplicação** - a redundância deve ser evitada, ou seja, cada conceito distinto deve estar num, e apenas um, sítio;
- **Acoplamento fraco** - consiste na ideia de que cada elemento deve ser tão independente dos restantes quanto possível, o que permite adicionar, remover e até migrar elementos e funcionalidades de uma aplicação para outra facilmente;
- **Explicitude** - deve-se evitar a complexidade do sistema. Um sistema mais simples e explícito é de mais fácil leitura, e permite uma melhor aprendizagem por parte de novos desenvolvedores.

Como referido na secção 2.7.2, o *back-end* de uma aplicação tem uma relação muito íntima com a base de dados. Neste campo, Django apresenta suporte para um grande número de sistemas de gestão de bases de dados, incluindo todos os mencionados na anteriormente, e possibilitando a substituição deste sistema de uma forma rápida e simples [53].

O Django é ainda uma ferramenta muito utilizada, principalmente nas comunidades *open-source*, o que resulta numa grande comunidade *online*. Como já foi referido na secção anterior, o facto de uma ferramenta possuir uma grande comunidade é valorizado no âmbito desta dissertação, pois existe uma forte probabilidade de existirem resoluções *online* para problemas semelhantes aos passíveis de serem encontrados. A sua vasta popularidade, associada aos seus princípios de desenvolvimento e vasta documentação, tornam esta numa das ferramentas indicadas para novos desenvolvedores de aplicações *web*.

4.3.3 AJAX

AJAX, ou Asynchronous JavaScript and XML, é uma tecnologia que permite aumentar o grau de interactividade de uma página *web*. A abordagem clássica destas páginas consistia num pedido HTTP através da interacção do utilizador. Esta tecnologia permite a realização destes pedidos de forma automática, sem que seja necessário recarregar a página para visualizar a resposta. Isto torna as páginas *web* muito mais dinâmicas, e permite a utilização destas aplicações para, por exemplo, validar a introdução de dados numa página por parte do utilizador ou monitorizar e controlar em tempo real dispositivos IoT que se encontrem ligados à mesma rede [54, 55].

4.4 Considerações Adicionais

O sistema deve ser composto por duas partes: um terminal individual, através do qual os alunos poderão responder a questões de escolha múltipla, e uma estação base-receptora, que recebe e processa as respostas vindas dos terminais, para além de servir de servidor para a aplicação *web*, através da qual os professores poderão criar e realizar questionários.

O terminal deve ser de dimensões reduzidas, uma vez que se pretende que este seja portátil. Para além disso, o terminal desenvolvido deverá ser robusto o suficiente para a realização de testes de escalabilidade em ambiente realista.

A aplicação deverá correr num *browser*, de forma a ser independente do sistema operativo da máquina e para poder ser acedido de forma remota. Esta aplicação, cujo público alvo são os docentes das Unidades Curriculares, deverá ser intuitiva e fácil de utilizar, de forma a que mesmo os utilizadores menos dotados de aptidão tecnológica consigam utilizar sem problemas. A aplicação deverá ainda possibilitar a criação de questionários, anónimos ou não, que devem estar associados ao utilizador (professor) e a uma Unidade Curricular, para além de permitir a exibição dos mesmos e processamento das respostas submetidas pelos alunos. De forma a garantir que cada utilizador da aplicação apenas tem acesso aos questionários e respostas a si associados, torna-se necessária a existência de contas de utilizador, protegidas por palavra-chave.

Uma vez que é da maior importância garantir que apenas os alunos presentes na sala de aula são capazes de responder aos questionários, é necessária a utilização de uma rede local. Esta rede deve ser criada por um *router* presente na sala, ao qual se conectam todos os terminais individuais e estação base-receptora.

De forma a dificultar a partilha de credenciais, o cartão de estudante deve permanecer presente no terminal até a resposta ser enviada, pretendendo-se com isto diminuir o risco de falsificação de identidade por parte dos inquiridos. Para evitar a partilha de terminais, para um dado questionário, apenas uma resposta deve ser validada por dispositivo, que deve ficar também associado ao aluno que o utiliza. Contudo, no caso dos questionários anónimos, deve ser dada a possibilidade aos alunos de responderem aos questionários sem a presença do seu cartão de identificação. Por consequência, o terminal deixa de estar associado a um determinado aluno, permitindo a partilha de terminais. Isto poderia levar a um número excessivo de respostas, pelo que é necessário garantir que mesmo nestas condições, apenas uma resposta será validada por terminal.

O terminal deve ainda ser, do ponto de vista da lógica de operação, o mais simples possível. Assim, todas as questões relacionadas com o anonimato do questionário ou

autenticação do aluno devem recair sobre o servidor da aplicação. O esquema da figura 4.2 procura exemplificar o modo de funcionamento e de comunicação do sistema proposto.

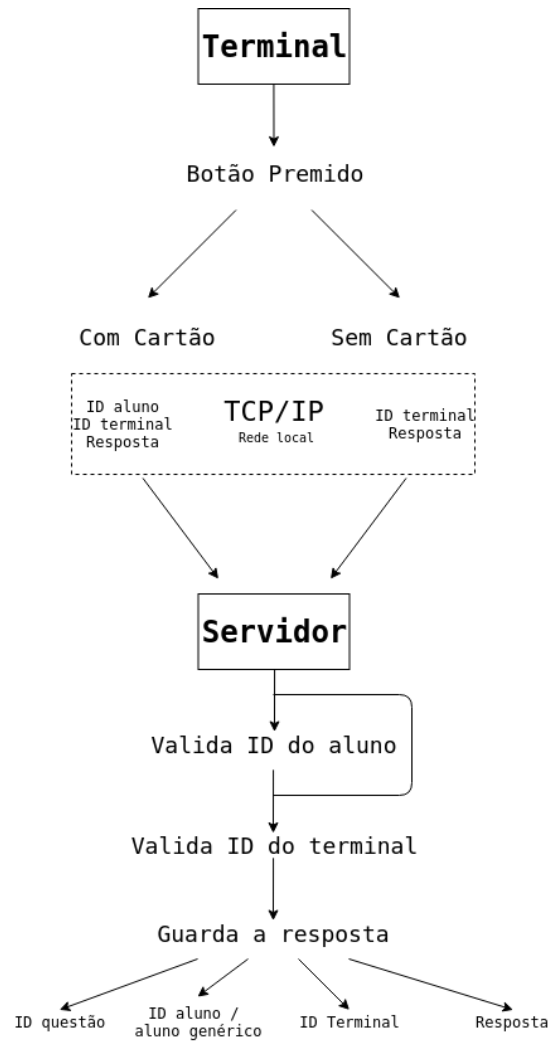


Figura 4.2: Diagrama representativo do envio, recepção e processamento de respostas do sistema proposto.

Parte II

Desenvolvimento e Implementação

Capítulo 5

Desenvolvimento da Aplicação WEB

A utilização da *framework* Django para o desenvolvimento do *back-end* da aplicação possibilita a utilização de ferramentas e funcionalidades pré-definidas, como a criação de contas de utilizador e encriptação das suas palavras-chave, ou a utilização de uma página de administração. A página de administração, que vem por defeito com esta *framework*, é particularmente interessante pois possibilita a gestão e, até certo ponto, a manutenção da aplicação sem ser necessário recorrer ao terminal. Por outro lado, a encriptação automática das palavras-chave dos utilizadores acrescenta uma camada de segurança ao sistema de autenticação, uma vez que ninguém, nem mesmo os administradores, têm acesso às senhas dos restantes utilizadores.

Contudo, para além das vantagens *out of the box* que esta ferramenta proporciona, é necessário que toda a estrutura de código siga um padrão pré-estabelecido.

Neste capítulo é exposto, de forma resumida, a arquitectura da aplicação concebida. Numa primeira fase, são explicados os principais elementos da aplicação, do ponto de vista do seu desenvolvimento, seguindo-se o modelo de relações entre registos da base de dados. É ainda apresentado o processamento lógico por detrás da realização de um inquérito e, por fim, são discutidas e justificadas algumas tomadas de decisão relativamente a acções que ocorrem no cliente e no servidor.

5.1 Conceitos Importantes

Para se perceber a estrutura da aplicação *web* desenvolvida, é necessário introduzir primeiro alguns conceitos relativos à utilização da *framework* Django. Estes conceitos são abordados em maior detalhe mais à frente, sendo também explicada a forma como são empregues. Os principais conceitos a saber são:

- Projecto - Corresponde a toda a aplicação *web* desenvolvida, composta por várias aplicações mais pequenas;
- Aplicações - Uma aplicação traduz um conjunto de funcionalidades do projecto. Por exemplo, um *blog* pode ser composto por uma aplicação responsável pela gestão de utilizadores, e outra responsável por toda a gestão dos *posts* criados;
- Modelos - Cada modelo corresponde, essencialmente, a uma tabela da base de dados e às relações a si associadas;

- Vistas - Uma vista traduz uma acção lógica da aplicação. Regra geral, uma aplicação é composta por várias vista, que podem assumir a forma de funções ou classes;
- URL - Um endereço URL indica o caminho para se aceder a uma vista. Uma vez que uma vista corresponde muitas vezes a uma função, o endereço URL pode ser visto como uma forma de evocar essa mesma função;
- *Templates* - Um *template* nada mais é que um ficheiro HTML, que pode ser utilizado numa ou mais vistas, de forma a traduzir graficamente o resultado das mesmas;
- Formulários - Um formulário traduz um conjunto de entradas de texto, botões rádio, ou outro tipo de *inputs* que permitem ao utilizador da aplicação inserir ou editar a informação presente na base de dados.

5.2 Arquitectura da Aplicação

Como foi referido no início deste capítulo, o desenvolvimento de uma aplicação em Django deve seguir uma estruturação pré-definida, como mostra a figura 5.1. Esta consiste em dividir uma aplicação (denominada *projecto*) em várias aplicações de dimensões menores (neste trabalho, *quiz* e *users*), de forma a facilitar o seu intercâmbio.

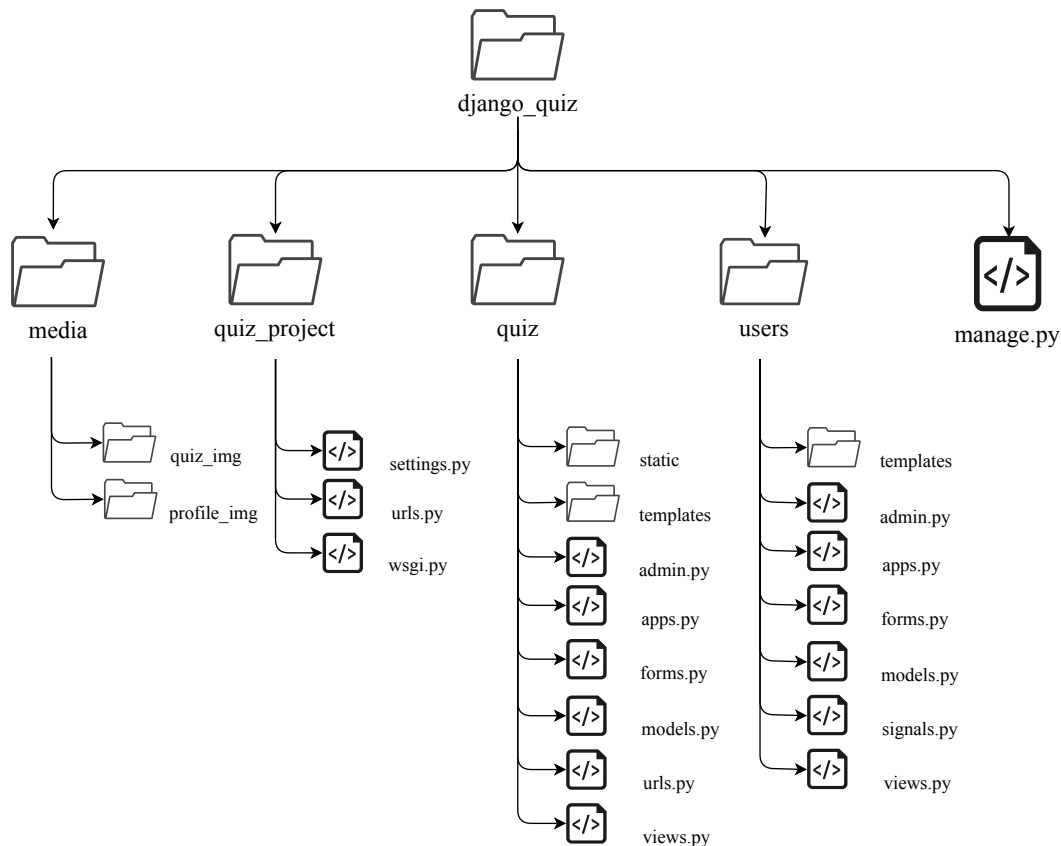


Figura 5.1: Árvore da aplicação *web*.

Ao criar pela primeira vez um projecto Django, denominado `quiz_project`, são criadas de forma automática uma série de ficheiros e pastas, seguindo a estrutura seguinte:

```
django_quiz/  
├── manage.py  
├── quiz_project/  
│   ├── settings.py  
│   ├── urls.py  
│   └── wsgi.py
```

A aplicação *web* é iniciada a partir do ficheiro `manage.py`, que se encontra na raiz do projecto. O directório `quiz_project` contém as definições transversais às várias aplicações inseridas no projecto. Estas aplicações devem ser modulares e tão independentes entre si quanto possível, de forma a que a remoção de uma aplicação não invalide o bom funcionamento das restantes. O sistema desenvolvido nesta dissertação é composto por duas aplicações: uma relacionada com os utilizadores, `users`, e outra dedicada aos questionários, designada `quiz`.

Tal como acontece com a criação de um projecto, também a criação de uma aplicação gera, de forma automática, uma série de pastas e ficheiros que estruturam a aplicação. Tipicamente, uma aplicação segue a seguinte estrutura:

```
quiz/  
├── admin.py  
├── apps.py  
├── forms.py  
├── models.py  
├── urls.py  
├── views.py  
├── templates/  
│   ├── quiz/  
├── static/  
│   └── quiz/
```

De forma a clarificar o papel de cada ficheiro, segue-se uma breve descrição das funções que desempenham. É importante referir que, apesar de ser aconselhado que se siga a estrutura descrita, tal não é obrigatório. Para além disso, algumas aplicações podem conter mais ou menos ficheiros do que os que são aqui descritos.

- `models.py` - Contém os modelos da aplicação, que correspondem às tabelas da base de dados;
- `views.py` - Contém as vistas da aplicação. Estas podem assumir a forma de funções ou classes e, por norma, correspondem a uma página da aplicação, podendo no entanto haver excepções. As vistas são ainda responsáveis por filtrarem ou processarem a informação que é enviada ou recebida pela aplicação;
- `forms.py` - Contém os formulários da aplicação, através dos quais o utilizador pode inserir dados. Por norma, não é necessário criar formulários, já que estes são criados

automaticamente com base no modelo a que se referem. Contudo, podem existir situações em que os formulários padrão não satisfaçam as necessidades pretendidas;

- `urls.py` - Contém os URL's, ou caminhos, para as páginas. É através destes que o utilizador pode aceder a uma página específica;
- `admin.py` - Contém os modelos, de uma dada aplicação, que são registados na página de administração pré-concebida.

Em cada aplicação existe ainda o directório `templates`, dentro do qual se encontram os ficheiros HTML para a renderização das páginas *web*, e o directório `static` (opcional), que contém os ficheiros CSS e JS referidos nos *templates*.

Resumindo, a aplicação desenvolvida assume então a estrutura seguinte, onde o directório `media` contém os ficheiros de imagem associados ao perfil dos utilizadores e aos questionários criados.

```

django_quiz/
├── manage.py
├── media/
├── quiz_project/
├── quiz/
│   ├── templates/
│   │   └── quiz/
│   └── static/
│       └── quiz/
├── users/
│   └── templates/
│       └── users/

```

5.2.1 Configurações Gerais

As configurações gerais do projecto, definidas no ficheiro `settings.py`, são aquelas que são transversais a todas as aplicações nele inseridas. Estas configurações correspondem às aplicações instaladas (sejam estas criadas pelo desenvolvedor ou não), à definição do gestor de bases de dados utilizado e das credenciais que a si dizem respeito, bem como ao caminho dos directórios `static` e `media`, caso existam, que contém os ficheiros estáticos associados aos *templates* e as imagens guardadas, respectivamente.

5.2.2 Aplicações

Como já foi referido, um projecto Django é constituído por uma, ou mais, aplicações. No âmbito do sistema CLASSQUIZ, foram desenvolvidas duas aplicações, `quiz` e `users`, de forma a separar os questionários dos utilizadores.

Embora não estejam previstas migrações de aplicações no futuro, a separação do sistema em duas aplicações facilita o seu desenvolvimento, para além de proporcionar uma organização mais intuitiva. Para além disso, apesar de se tratar de duas aplicações distintas, é possível referenciar uma a partir da outra, não comprometendo assim o acesso mútuo.

De forma a permitir a expansão futura da aplicação, apenas a aplicação **quiz** faz referência à aplicação **users**, e nunca ao contrário. À partida, a aplicação **users** estará sempre presente no futuro, independentemente das restantes aplicações existentes.

5.2.3 Modelos

Os modelos correspondem, no fundo, às tabelas existentes na base de dados, e estão inseridos numa das duas aplicações criadas. Começando pelos utilizadores, existem três modelos: **Profile**, **Course** e **ProfileCourse**. O primeiro corresponde ao perfil do utilizador, que é diferente da sua conta. Isto acontece porque a conta de utilizador é gerada de forma automática, e contém campos como *username*, *password*, primeiro e último nome, e-mail e permissões de utilizador. Contudo, esta informação não é suficiente para um utilizador do sistema CLASSQUIZ. Assim, o perfil do utilizador serve como uma extensão ao modelo criado por defeito, e permite associar mais informação ao utilizador.

O modelo **Course** nada mais é que uma tabela com a lista de Unidades Curriculares, e respectivas chaves de identificação única. As Unidades Curriculares e os utilizadores são associados por via do modelo **ProfileCourse**.

Já no que à aplicação **quiz** diz respeito, existem vários modelos e as relações entre si são um pouco mais complexas. Esta aplicação é composta por onze modelos distintos:

- **Answer** - Modelo onde são armazenadas as respostas submetidas pelos alunos;
- **AnswerProcessing** - Semelhante ao modelo **Answer**, onde são processadas as respostas;
- **Results** - Modelo são armazenadas as respostas dos alunos, já processadas;
- **Session** - Corresponde às sessões realizadas. Cada sessão corresponde a uma instância de um dado questionário;
- **Student** - Contém a lista de alunos;
- **Terminal** - Contém a lista de terminais autorizados;
- **Quiz** - Corresponde às questões criadas;
- **Quiz_Course** - Estabelece a relação entre uma dada questão e uma Unidade Curricular (semelhante ao modelo **ProfileCourse**);
- **Lesson** - Corresponde a uma lição (ou aula). Todos as questões realizadas ficam automaticamente associados à última lição criada pelo utilizador;
- **LessonStudent** - Estabelece a relação entre uma lição e os resultados dos alunos, de forma a que seja possível visualizar os resultados agrupados de várias questões realizadas;
- **Lesson_Course** - Estabelece a relação entre uma lição e uma Unidade Curricular (semelhante ao modelo **ProfileCourse**).

A existência do modelo **AnswerProcessing** deve-se ao facto de as respostas demorem alguns instantes a serem processadas. Assim, torna-se necessário migrar as respostas

recebidas para um outro modelo onde são processadas, de forma a evitar problemas resultantes da recepção de respostas durante o seu processamento. Este processo encontra-se explicado em maior detalhe na secção 5.3.5.

5.2.4 Vistas

Uma vista consiste numa função responsável por processar um pedido HTTP, podendo retornar, ou não, uma resposta, que contém o objecto requerido. O objecto nada mais é do que uma instância de um modelo. Assim, as vistas são responsáveis pela *lógica* da aplicação.

Regra geral, uma vista está associada a uma página acessível pelo utilizador, que pode ser acedida por via de um endereço URL. Um exemplo disto é a vista responsável pela página principal da aplicação desenvolvida (`QuizListView`) que mostra todas as questões existentes das Unidades Curriculares onde o utilizador é docente (listagem 5.1). Nesta vista, o modelo utilizado é o `Quiz`, que contém todos os questionários criados. Esta vista é ainda responsável pela renderização da página, recorrendo ao `template home.html`. Após o acesso à página por parte do utilizador, esta vista é evocada, respondendo ao pedido HTML com o objecto `quizzes`. Este objecto corresponde ao modelo `Quiz` mencionado anteriormente. No entanto, uma vez que este modelo contém todas as questões criadas, e apenas se pretende responder ao pedido com as questões das Unidades Curriculares onde o utilizador é docente, torna-se necessário filtrar o objecto através da função `get_queryset()`.

Listagem 5.1: Vista `QuizListView`.

```
class QuizListView(ListView):
    model = Quiz
    template_name = 'quiz/home.html'
    context_object_name = 'quizzes'
    ordering = ['-date_created']

    def get_queryset(self):
        auth_user = self.request.user;
        q1 = Course.objects.filter(profile=auth_user.id)
        q2 = Quiz.objects.filter(course__in=q1)
        return q2.order_by('-date_created').distinct()
```

Para se proceder à filtragem, primeiro é necessário determinar quem é o utilizador da aplicação, definido na variável `auth_user`. A variável `q1` corresponde à lista de Unidades Curriculares onde o utilizador é docente, ao passo que a variável `q2` representa o modelo `Quiz` filtrado pelas Unidades Curriculares definidas em `q1`. Por fim, a função retorna o objecto `q2`, ordenado por ordem inversa à sua data de criação. A aplicação do parâmetro `distinct()` garante que não são retornados questionários repetidos (isto só aconteceria se um questionário estivesse associado a mais que uma Unidade Curricular). O objecto `q2` retornado corresponde então ao objecto `quizzes` da vista.

Como já foi referido, nem todas as vistas estão associadas a uma página à qual o utilizador pode aceder. No âmbito do sistema `CLASSQUIZ`, estas podem corresponder às três vistas seguintes: a responsável pelo início de uma questão, a responsável pelo seu término, e a responsável pela recepção de respostas vindas dos terminais individuais. A vista exemplificada na listagem 5.2 (`start_quiz`) é a vista responsável pelo início de uma

questão, ou sessão. Esta vista, ao contrário da exemplificada anteriormente, assume a forma de uma função. A opção entre uma função ou uma classe não interfere em nada com a funcionalidade da mesma, sendo meramente uma preferência do desenvolvedor.

A primeira diferença de destaque entre esta e a vista anterior é que esta deve ser capaz de alterar os registos da base de dados. Esta permissão é atribuída à vista através da utilização do *decorator* `@require_http_methods(["POST"])`. Como tal, de forma a salvaguardar a integridade da informação, a *framework* Django oferece um serviço de protecção contra ataques CSRF¹, através um *token*. Este *token* está associado aos *cookies* do navegador de Internet do utilizador, e a sua requisição deve ser sempre efectuada num *template* quando se pretende alterar informações da base de dados da aplicação *web*. Contudo, esta vista não tem nenhum *template* associada, sendo acedida através de um pedido AJAX. Por consequência, é necessário excluir esta camada de segurança desta vista, através da utilização do *decorator* `@csrf_exempt`.

Listagem 5.2: Vista `start_quiz`.

```
@csrf_exempt
@require_http_methods(["POST"])
def start_quiz(request, *args, **kwargs):
    Answer.objects.all().delete()
    AnswerProcessing.objects.all().delete()

    body = request.body.decode('utf-8')
    m = re.search('id=(.+?)&', body)
    if m:
        quiz_id = m.group(1)

    quiz = get_object_or_404(Quiz, id=quiz_id)
    quiz.start_date = datetime.datetime.now()
    quiz.save()

    to_return = {'type': 'success', 'msg': 'done', 'code': 200}
    return HttpResponse(json.dumps(to_return),
                        content_type='application/json')
```

A primeira tarefa desta vista passa por apagar todos os objectos do modelo `Answer`. Este modelo contém as mensagens (respostas) enviadas pelos terminais individuais, pelo que é necessário garantir que esta tabela se encontra vazia no início de um qualquer questão. Da mesma forma, são apagadas todas as entradas da tabela `AnswerProcessing`.

O passo seguinte passa pela obtenção da chave primária da questão a ser realizada. Esta informação é enviada pelo método AJAX definido no *template* da questão em causa (quando o utilizador prime o botão para iniciar a sessão, evoca esta vista e passa-lhe a informação relativa à questão a ser executada). A informação chega à vista `start_quiz` em formato JSON, sendo depois descodificada e armazenada na variável `body`. Esta variável é então filtrada (variável `m`) e, por fim, a chave primária da questão é armazenada na variável `quiz_id`.

Uma vez obtida a chave primária da questão, é possível obter o seu objecto (`quiz`). A função `get_object_or_404()` retorna o objecto pretendido, ou o erro 404 caso este

¹CSRF é um tipo de ataque onde uma página maliciosa utiliza o *browser* do visitante para realizar ataques a uma página distinta [56].

não exista. É ainda necessário actualizar a data e hora de início da realização da questão (atributo `start_date` do modelo `Quiz`). Por fim, a vista responde ao pedido HTTP com o código de sucesso 200.

5.2.5 Formulários

Como já foi referido, regra geral os formulários são gerados de forma automática. Contudo existem cenários onde os formulários gerados não satisfazem as necessidades pretendidas, seja por apresentarem demasiada informação, seja por não apresentarem da forma desejada. Assim, surge a necessidade de criar formulários, ou *forms*, que satisfaçam as necessidades de casos particulares.

Um exemplo disto é o formulário para a criação/edição de uma questão (listagem 5.3). Um dos atributos das questões é a Unidade Curricular a que estão associadas. Contudo, um formulário gerado automaticamente apresenta como opção de selecção todas as Unidades Curriculares existentes, quando se pretende que apenas exiba as opções correspondentes às Unidades Curriculares onde o utilizador é docente.

Listagem 5.3: Formulário `QuizForm`.

```
class QuizForm(forms.ModelForm):
    def __init__(self, auth_user, *args, **kwargs):
        super(QuizForm, self).__init__(*args, **kwargs)
        self.fields['course'] = forms.ModelChoiceField(
            queryset=Course.objects.filter(
                profile=auth_user.id
            )
        )

    class Meta:
        model = Quiz
        fields = ['course', 'title', 'question', 'ansA', 'ansB',
                 'ansC', 'ansD', 'ansE', 'right_ans', 'duration',
                 'image', 'anonymous']
```

No formulário exemplificado na listagem 5.3 é realizada uma *query*, ou filtragem, que se encontra definida na função `__init__()`, às Unidades Curriculares existentes. O resultado desta filtragem é armazenado na variável `course`, e corresponde à lista de Unidades Curriculares onde o utilizador lecciona.

Desta forma, o atributo `course` do modelo `Quiz`, cujo formulário é definido na *class* `Meta`, não exibe todas as Unidades Curriculares existentes, mas sim aquelas onde o utilizador é docente. Para além disso, é necessário definir nesse mesmo formulário todos os restantes atributos, uma vez que este formulário é utilizado no lugar do gerado automaticamente. É importante referir que os atributos referentes à data de criação, data de começo de uma questão e autor não precisam de ser definidos, uma vez que são povoados de forma automática.

5.2.6 URL's

Os endereços URL, associados às vistas criadas, são definidos num ficheiro próprio (`urls.py`). Na listagem 5.4 são tomadas como exemplo apenas as vistas descritas anteriormente.

Listagem 5.4: Endereços URL.

```
urlpatterns = [
    path('', login_required(QuizListView.as_view()),
          name='quiz-home'),
    path('quiz/<int:pk>/start_quiz/', views.start_quiz,
          name='start_quiz')
]
```

Na listagem anterior, o primeiro endereço URL corresponde à vista da página principal (`QuizListView`), onde são apresentados todas as questões das Unidades Curriculares onde o utilizador é docente. Repare-se que o endereço desta vista está vazio. Isto acontece por esta página corresponde à página principal, ou seja, na prática o endereço corresponde ao endereço IP do servidor da aplicação *web*, ou ao seu domínio. O parâmetro `login_required()` define que é necessário o utilizador estar autenticado para aceder à página e, caso tal não aconteça, será redireccionado para a página de *login*. Por fim, o parâmetro `name='quiz-home'` define uma variável que traduz o endereço descrito.

O segundo endereço exemplificado diz respeito à vista `start_quiz`. Como já foi referido, esta vista não tem nenhum *template* associado, contudo, não dispensa da existência de um endereço URL. No endereço descrito, `<int:pk>` corresponde à chave primária da questão. Por exemplo, para o caso da questão número 36, o endereço desta página no *browser* seria `http://192.168.0.2:8000/quiz/36/start_quiz/`, onde 192.168.0.2 corresponde ao endereço IP do servidor na rede local e 8000 traduz a porta onde se encontra à escuta.

5.2.7 Templates

Os *templates* correspondem aos ficheiros HTML associados às vistas, e que serão interpretados pelo *browser* do utilizador. Apesar de cada página ser única, existem aspectos comuns à maioria, como a barra de navegação superior. Desta forma, faz sentido a criação de um *template* que contém os componentes comuns às restantes páginas, de forma a evitar a repetição de código. Neste ficheiro, de nome `base.html`, são evocados os ficheiros estáticos CSS e JS utilizados, definindo também a barra de navegação superior (que permite aceder às diferentes páginas da aplicação) e as mensagens de sucesso e alerta que possam ocorrer.

Previamente foi falado do *token* CSRF, cuja utilização é necessária quando se pretende alterar a informação presente na base de dados da aplicação. Um exemplo da aplicação deste *token* é na página de criação de questões, onde ao utilizador é apresentado um formulário ainda por preencher. A listagem 5.5 demonstra como é incorporado o formulário criado (`course_form`) nesta página.

Listagem 5.5: Incorporação do formulário `course_form` num *template*.

```
<form id="quiz_form" method="POST" enctype="multipart/form-data">
  {% csrf_token %}
  <fieldset class="form-group">
    {{ course_form|bootstrap }}
  </fieldset>
</form>
```

A inclusão de uma *tag* do tipo *form* indica ao *browser* que a informação nela contida corresponde a um formulário. Uma vez que a informação preenchida no formulário será adicionada à base de dados, o pedido HTML é do tipo *Post*, pelo que é necessário acesso ao *token* CSRF. De seguida, o formulário criado é inserido na página, sendo-lhe aplicado o estilo padrão da biblioteca Bootstrap².

Em relação à vista `start_quiz`, foi dito que esta é evocada quando o utilizador prime o botão `play_btn` para iniciar uma instância de uma questão. Este processo é realizado através da função JS, exemplificada na listagem 5.6.

Listagem 5.6: Função que evoca a vista `start_quiz`.

```
document.getElementById("play_btn").addEventListener("click",
function() {
$.ajax({
type: "POST",
url: "start_quiz/",
data: { 'id': '{{ object.id }}',
'start_date': '{{ object.start_date }}',
'right_ans': '{{ object.right_ans }}'
},
dataType: 'json',
success: function(data, textStatus) {
console.log(data);
}
});
});
```

O excerto de código anteriormente representado actua no *template* de uma dada questão que possui, entre outros, um botão para dar início à sua realização. Neste caso, a questão encontra-se caracterizada pela variável `object`. Como se pode ver na função descrita, ao clicar no botão `play_btn` é enviado um *Post request* para o endereço `start_quiz`, enviando também a informação referente à questão, em formato JSON.

5.2.8 Ficheiros Estáticos

Os ficheiros estáticos correspondem às bibliotecas JavaScript e CSS utilizadas pela aplicação. Tipicamente, estas bibliotecas são obtidas por CDN³, contudo, uma vez que o sistema CLASSQUIZ se encontra restringido a uma rede local, tal não é possível. Assim, todas as bibliotecas utilizadas têm de estar armazenadas no servidor da aplicação. Para além das bibliotecas desenvolvidas por terceiros, também os ficheiros `.css` desenvolvidos ficam armazenados no directório `static/`. Estes ficheiros contêm os estilos criados especificamente para aplicação, como por exemplo a barra de navegação.

Atendendo à estrutura apresentada no início da secção 5.2, verifica-se que o directório `static/` se encontra dentro da aplicação `quiz`, o que se deve ao facto de tanto o *template* base, como a maioria dos *templates* que recorrem a estas bibliotecas corresponderem a vistas desta aplicação, facilitando assim o *routing* entre ficheiros.

²A inclusão de variáveis do Django nos *templates* é feita através do uso de duplas chavetas. Exemplo: `{{ course_form|bootstrap }}`.

³CDN é uma forma de distribuição de conteúdo através da Internet. No caso das aplicações *web*, estas redes são utilizadas de forma a obter os *scripts* necessários directamente dos seus desenvolvedores, ao invés de estarem armazenados no servidor da aplicação.

No que respeita ao estilo da aplicação, é utilizada a *framework* Bootstrap, composta por uma série de objectos gráficos como botões e entradas de texto, e otimizada para a utilização em dispositivos móveis, como é o caso dos *smartphones* e *tablets*. Contudo, esta *framework* não se limita ao estilo dos objectos, possuindo um conjunto de *scripts* JS que tornam a página mais dinâmica e fluente.

O sistema CLASSQUIZ utiliza várias bibliotecas JS, de forma a incorporar funcionalidades adicionais no navegador da Internet do utilizador, tais como o *display* de fórmulas matemáticas, ou a possibilidade de importar/exportar tabelas para formato Excel. Assim, as bibliotecas JS utilizadas pela aplicação desenvolvida são:

- MathJax - Permite a exibição de notação matemática em navegadores da Internet, utilizando para isso marcação MathML, LaTeX e ASCIIMathML;
- FileSaver.JS e SheetJS - São utilizadas em conjunto para permitir a importação e exportação de tabelas para um ficheiro Excel;
- Popper e JQuery - São bibliotecas genéricas que fornecem uma vasta gama de funcionalidades. A primeira, Popper, é utilizada particularmente em conjunto com a *framework* Bootstrap, enquanto que a JQuery permite a implementação de métodos AJAX.

5.3 Relações da Base de Dados

A base de dados é uma das partes fundamentais da aplicação. É nela que fica armazenada a maior parte da informação, e é através dela que é possível estabelecer relações entre modelos. Estas relações são particularmente importantes uma vez que, normalmente, as vistas apenas têm acesso a um modelo. Esta secção tem como objectivo detalhar os vários modelos criados durante o desenvolvimento da aplicação, bem como as relações entre eles existentes.

5.3.1 Descrição da Base de Dados, por Aplicação

Como referido anteriormente, a presente secção visa detalhar os vários modelos implementados na aplicação *web* do sistema CLASSQUIZ. De forma a contextualizar melhor as secções que se seguem, são apresentados de seguida, nas figuras 5.2 e 5.3, os esquemas relacionais das bases de dados, por aplicação (**users** e **quiz**). Nos esquemas apresentados, os modelos representados com a cor amarela integram a aplicação **auth**, gerada automaticamente pelo Django e que contém a informação referente aos utilizadores da aplicação, ao passo que a cor verde representa os modelos da aplicação **users** e a cor azul a aplicação **quiz**.

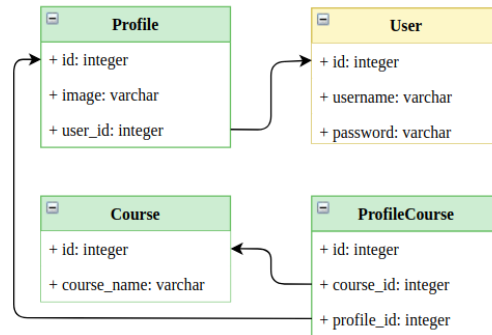


Figura 5.2: Esquema relacional dos modelos da aplicação users.

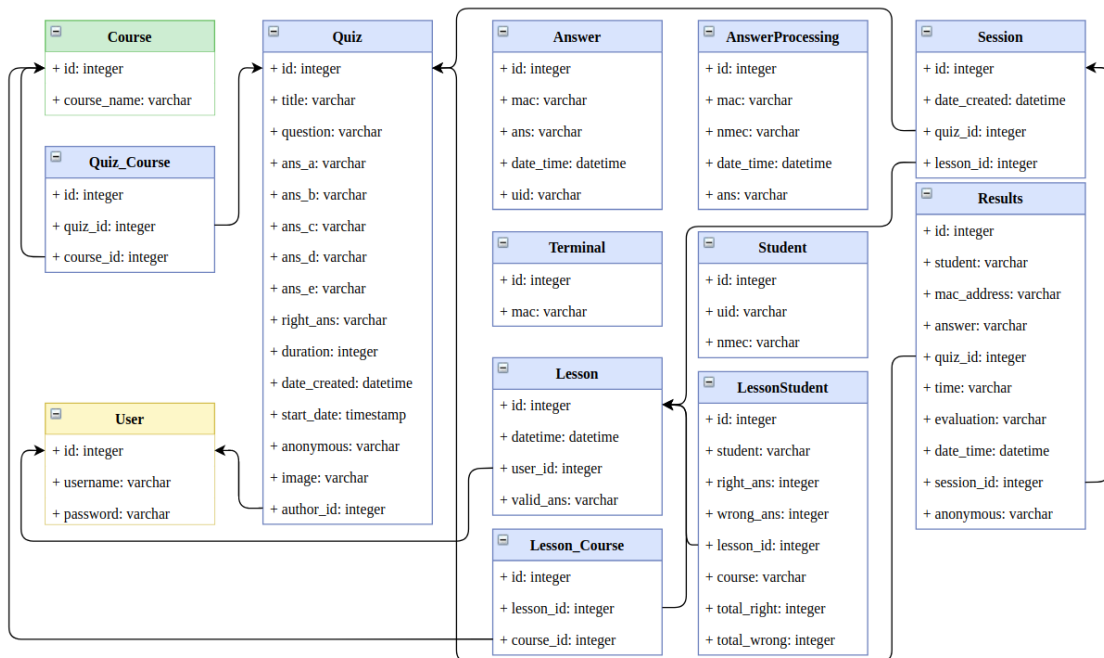


Figura 5.3: Esquema relacional dos modelos da aplicação quiz.

5.3.2 Utilizadores, Unidades Curriculares e Questões

Uma das relações principais da aplicação é a relação entre uma questão, o seu autor e a sua Unidade Curricular, representada na figura 5.4. De uma forma resumida, um utilizador pode estar associado a várias questões, que por sua vez estão associadas a um único utilizador (autor). Assim, a relação entre os questões e o seu autor é do tipo *um para muitos* (1:N). Por outro lado, um utilizador pode estar associado a várias Unidades Curriculares, e estas podem também estar associadas a vários utilizadores, tratando-se portanto de uma relação do tipo *muitos para muitos* (M:N). Contudo, uma questão apenas pode estar associado a uma Unidade Curricular, e da mesma forma, uma Unidade Curricular pode ter vários questionários associados a si. Esta seria então uma relação do tipo 1:N. Do ponto de vista teórico, esta seria a forma correcta de estabelecer esta relação, mas do ponto de vista da implementação leva ao aparecimento de vários problemas resultantes da forma como as alterações à base de dados são implementadas pelo Django. A solução passa então por estabelecer uma relação M:N entre os questionários e as Unidades Curriculares, sendo que a selecção da Unidade Curricular, aquando da criação de uma nova questão, encontra-se limitada à selecção de apenas uma opção. Assim, apesar de se tratar na prática de uma relação M:N, comporta-se como que se de uma relação 1:N se tratasse, do ponto de vista do utilizador.

Analisando o esquema da figura 5.2, verifica-se que a relação entre os utilizadores e os seus perfis é estabelecida através de uma chave estrangeira, que corresponde a uma chave primária do modelo `User`. Contudo, a relação entre os utilizadores e as Unidades Curriculares não pode ser estabelecida da mesma forma, uma vez que se trata de uma relação M:N. Assim, a relação é feita através de um modelo auxiliar, `ProfileCourse`, onde cada registo (ou *objecto*) corresponde a uma relação entre um utilizador e uma Unidade Curricular. É importante referir que, de forma a não alterar o modelo `User`, que é criado de forma automática pelo Django, a relação entre os utilizadores e as Unidades Curriculares dá-se através do seu perfil, que por sua vez possui uma relação do tipo 1:1 para com o utilizador. Assim, um utilizador possui um único perfil, que por sua vez se relaciona com o modelo `Course` por via do modelo `ProfileCourse` (figura 5.2). Desta forma, se uma qualquer vista da aplicação tiver acesso a um objecto do modelo `Profile`, tem também acesso ao utilizador correspondente e às Unidades Curriculares a si associadas.

De forma a realçar a importância das relações entre modelos, o exemplo seguinte toma a página principal da aplicação como referência. Esta página, como visto na secção 5.2.4, apenas retorna o modelo `Quiz`. Contudo, uma vez que são apresentados somente as questões das Unidades Curriculares associadas ao utilizador, é necessário ter acesso tanto ao modelo dos utilizadores como das Unidades Curriculares a ele associadas. Um dos campos do modelo `Quiz`, como se pode ver no esquema da figura 5.4, corresponde ao utilizador que criou a questão, através de uma chave estrangeira (que corresponde à chave primária do modelo dos utilizadores). Outro campo do modelo `Quiz` diz respeito à Unidade Curricular que fica associada à questão. Mais uma vez, tratando-se de uma relação M:N, não é possível estabelecer esta relação através de uma chave estrangeira, mas sim através de um modelo complementar (`Quiz_Course`). Desta forma, facilmente se verifica que as relações M:N aumentam, de uma forma considerável, a complexidade das relações.

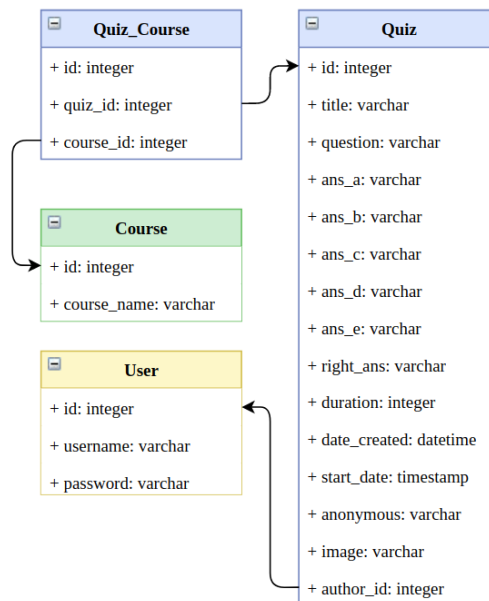


Figura 5.4: Relação entre os modelos User, Course e Quiz.

5.3.3 Lições, Questões e Sessões

Existe uma série de modelos associados à realização de questionários e consequente processamento das respostas recebidas dos terminais individuais. Cada vez que um questão é executada, é criada uma nova sessão (**Session**) que corresponde, no fundo, a uma instância dessa mesma questão. Desta forma, é possível realizar a mesma questão várias vezes, como se de questões diferentes se tratassem. As questões realizadas ficam também associados a uma lição (**Lesson**), que deve ser criada previamente pelo utilizador.

O modelo **Lesson** é caracterizado pelos campos de identificação (**id**), que corresponde à chave primária do objecto, de data e hora de criação (**datetime**), pelo utilizador que cria a lição (**user**) e pelo campo **valid_ans**, que corresponde à resposta que deve ser considerada válida durante a realização dos questionários (se a primeira submetida pelos alunos, ou a última). Para além dos campos referidos, tal como acontece no modelo **Quiz** descrito anteriormente, a relação entre as Unidades Curriculares e as lições é feita através do modelo auxiliar **Quiz_Course**. Para cada utilizador, será sempre utilizada a última lição criada e, uma vez que cada lição se encontra associada a uma Unidade Curricular, o utilizador só poderá executar questões dessa mesma Unidade Curricular. Assim, para executar questões de outras disciplinas, o utilizador deve criar uma nova lição, associada a essa Unidade Curricular. As lições assumem o papel de agrupar questões executadas (sessões), de forma a que se possa apresentar os resultados acumulados. O esquema da figura 5.5 procura exemplificar as relações descritas neste parágrafo.

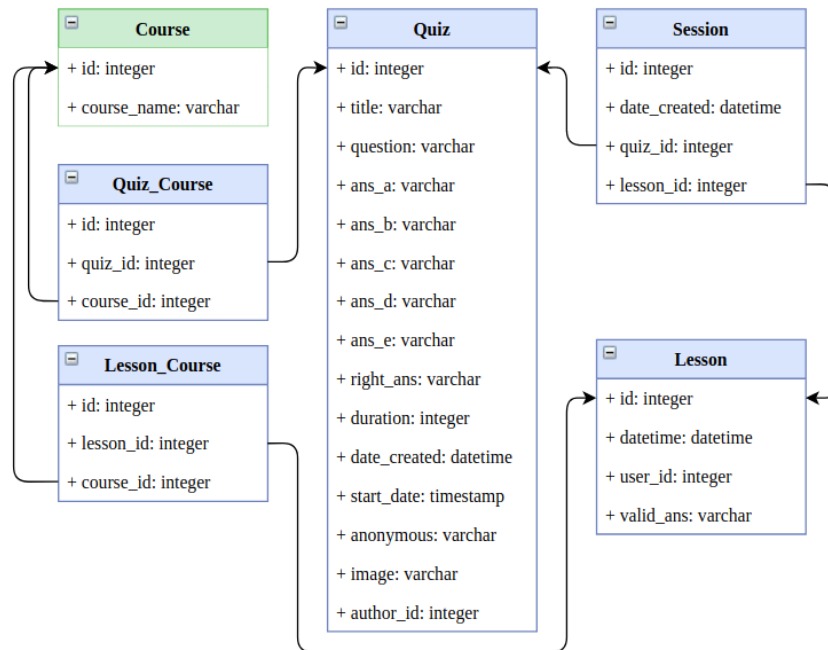


Figura 5.5: Relação entre os modelos Course, Lesson, Quiz e Session.

5.3.4 Resultados

Como já foi referido, existem duas formas de apresentar os resultados dos questionários: uma é apresentando os resultados acumulados da lição, outra é apresentando os resultados de cada questão individual (sessão). Para cada uma destas formas é utilizado um modelo diferente. O diagrama da figura 5.6 exemplifica, de forma simplificada, as relações existentes entre os modelos Lesson, LessonStudent, Session, Results e Quiz, ao mesmo tempo que identifica os campos que compõem os mesmos.

O modelo Results traduz os resultados de cada sessão, e é constituído pelos seguintes campos:

- **id**: Chave primária;
- **student**: Número mecanográfico do aluno;
- **mac_address**: Endereço MAC do terminal;
- **answer**: Resposta dada pelo aluno;
- **quiz_id**: Chave estrangeira da questão;
- **time**: Tempo que o aluno demorou a responder;
- **evaluation**: Avaliação da resposta (certo ou errado);
- **date time**: Data e hora em que a resposta foi guardada na base de dados;
- **session_id**: Chave estrangeira da sessão;
- **anonymous**: Identifica se se trata, ou não, de uma questão anónima.

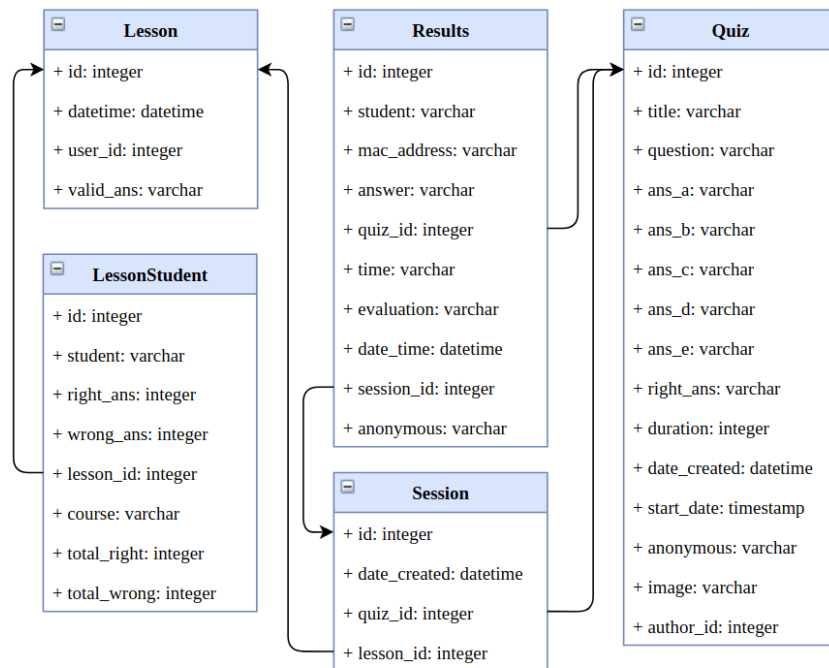


Figura 5.6: Relação dos modelos Results e LessonStudent.

Alguns dos campos descritos acima podem parecer redundantes, tais como o campo `anonymous`, já que se pode obter essa informação a partir da questão. No entanto, a questão pode vir a ser editada no futuro, e uma questão que era anônima pode deixar de o ser. Assim, a opção por repetir informação deve-se à necessidade de salvaguardar os parâmetros da questão no momento da sua realização, de forma a que futuras alterações não interfiram com os resultados já obtidos. Também a identificação do Quiz em evidência, definido pelo campo `quiz_id`, poderia ser omitida, uma vez que essa informação pode ser obtida por via do campo `session_id`. Contudo, na eventualidade de a sessão vir a ser apagada no futuro, é importante que os resultados continuem a estar associados a um questionário existente.

É no modelo `LessonStudent` que ficam acumulados os resultados de uma dada lição. Este modelo traduz, para cada lição, o número de respostas certas e erradas dadas por cada estudante. Este modelo é constituído pelos seguintes campos:

- `id`: Chave primária;
- `student`: Número mecanográfico do aluno;
- `right_ans`: Indica se a resposta dada a uma dada questão está certa;
- `wrong_ans`: Indica se a resposta dada a uma dada questão está errada;
- `lesson_id`: Chave estrangeira da lição;
- `course`: Unidade Curricular a que a lição se encontra associada;
- `total_right`: Número de respostas certas dadas pelo aluno;
- `total_wrong`: Número de respostas erradas dadas pelo aluno.

Tal como acontece com o modelo `Results`, também o modelo `LessonStudent` é composto por campos que podem parecer, à primeira vista, redundantes. Um exemplo disto é o campo `Course`, que descreve informação que pode ser obtida através do campo `lesson_id`. No entanto, uma vez que se trata de um campo que traduz uma relação do tipo M:N, a obtenção dessa informação torna-se bastante mais complexa. Assim, a opção por definir esta informação num campo dedicado resulta da simplicidade na forma como a informação é depois tratada pela aplicação, bem como pela redução do tempo despendido no desenvolvimento da mesma.

A existência dos campos `right_ans` e `wrong_ans` deve-se à forma como são criados novos registos deste modelo. Após a realização de uma questão, e depois de ser criado um novo registo no modelo `Results`, é criado um novo registo no modelo `LessonStudent` que indica se o aluno acertou, ou não, na resposta correcta da questão. Da mesma forma, após cada questão executada, é efectuado num somatório de todas as respostas certas e erradas de cada aluno, cujo resultado ocupa os campos `total_right` e `total_wrong`.

5.3.5 Respostas

Para se proceder ao processamento das respostas recebidas pelos terminais, são utilizados quatro modelos adicionais: `Answer`, `AnswerProcessing`, `Student` e `Terminal`. Todas as respostas enviadas pelos terminais, independentemente de estar a ocorrer, ou não, a realização de um questionário, ficam registadas no modelo `Answer`, caracterizado pelos seguintes campos:

- `id`: Chave primária;
- `mac`: Endereço MAC do terminal que submeteu a resposta;
- `ans`: Opção de resposta seleccionada pelo utilizador do terminal;
- `date_time`: Data e hora de submissão da resposta;
- `uid`: Identificação única do cartão de estudante presente no terminal.

Este modelo serve então o propósito de armazenar, momentaneamente, todas as mensagens recebidas dos terminais. Contudo, uma vez que todas as mensagens recebidas são armazenadas aqui, independentemente de estar a decorrer, ou não, um inquérito, torna-se necessário proceder ao seu processamento num outro modelo, de forma a que não estejam a ser registados novos dados no modelo durante o processamento das respostas. Para tal, é utilizado o modelo `AnswerProcessing`. Após a realização de uma questão, e antes de se iniciar o processamento das suas respostas, toda a informação contida no modelo `Answer` é migrada para o modelo `AnswerProcessing`, com a diferença de que, agora, o autor de uma resposta é caracterizado pelo seu número mecanográfico em vez do seu cartão de estudante.

A correspondência entre um cartão de identificação e o aluno ao qual corresponde é feita por comparação com o modelo `Student`. Este modelo nada mais é que uma lista de todos os alunos, identificados pelo seu número mecanográfico, e a identificação única do seu cartão de estudante, como se pode ver o esquema da figura 5.3.

Uma das acções à qual são submetidos os registos do modelo `AnswerProcessing` é a verificação do terminal utilizado para submeter a resposta, através do seu endereço

MAC. Assim, o modelo `Terminal` representa uma lista de todos os terminais registados no sistema. A existência desta lista, e conseqüente utilização na validação dos terminais, assume especial importância devido à necessidade em assegurar que as questões colocadas apenas podem ser respondidas utilizando o *hardware* adequado, impedindo que possam ser submetidas respostas a partir de outros dispositivos, como um computador ou telemóvel.

5.3.6 Hierarquia dos Modelos

Os modelos descritos anteriormente apresentam relações que se podem tornar bastante complexas, principalmente uma vez que não existe uma hierarquia clara entre os mesmos. Isto resulta, em parte, da natureza da *framework* utilizada para o desenvolvimento da aplicação, que incentiva ao desenvolvimento de uma aplicação por módulos. Para além disso, a forma como se encontra desenvolvida a aplicação atribui igual importância aos utilizadores e às Unidades Curriculares. Contudo, assumindo um exemplo onde exista apenas um utilizador e uma Unidade Curricular é possível representar, de forma simplificada, as várias relações descritas nas secções anteriores, como apresentado na figura 5.7, que visa ilustrar a hierarquia dos principais modelos da aplicação, no que à criação e realização de um questionário diz respeito.

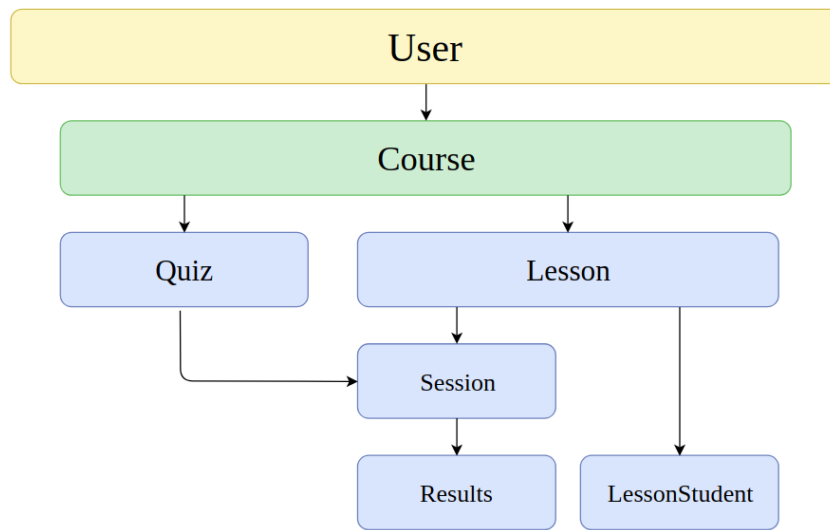


Figura 5.7: Hierarquia dos principais modelos da aplicação.

5.4 Realização de um Questionário e Processamento das Respostas

A principal função da aplicação desenvolvida é possibilitar a realização de questionários directamente a partir do navegador de Internet do professor.

Esta secção visa explicar como são realizados os questionários na aplicação, desde as acções que o utilizador (professor) deve realizar na interface gráfica à lógica aplicada pelo servidor para tal. O diagrama da figura 5.8 exemplifica todo o processo de recepção e processamento de respostas por parte da aplicação *web*.

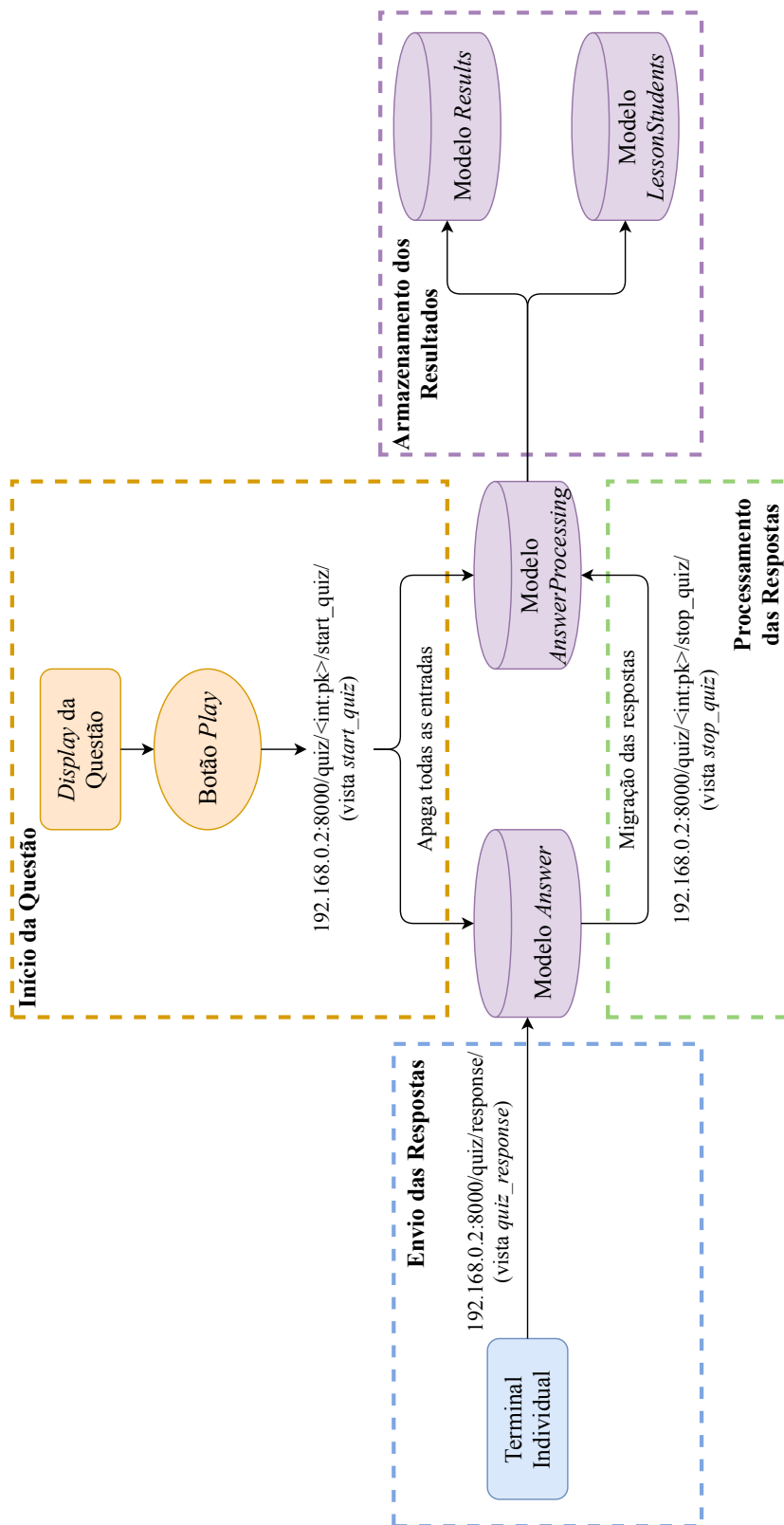


Figura 5.8: Diagrama exemplificativo do processamento de respostas.

5.4.1 Realização de um Inquérito

Uma vez criada uma questão, e iniciada uma lição, o utilizador da aplicação *web* pode proceder à realização de perguntas, desde que estas pertençam à mesma Unidade Curricular da lição activa e tenham sido criadas por si. Uma vez cumpridos estes requisitos, o utilizador pode navegar até à página de uma dada questão, cujo endereço URL assume a forma `http://192.168.0.2:8000/quiz/<int:pk>/`, onde `<int:pk>` corresponde à chave primária da questão, e lança-la em modo *fullscreen*.

O utilizador pode, de seguida, iniciar a realização dessa mesma questão, criando assim uma nova instância (ou *session*) da mesma. Esta acção efectua um pedido AJAX ao servidor, activando, de forma dinâmica, a vista `start_quiz` exemplificada na secção 5.2.4. Como já foi referido, ao carregar esta vista, todos os registos dos modelos `Answer` e `AnswerProcessing` são apagados, assegurando que não existem na base de dados respostas dadas antes do começo da questão.

Uma vez iniciada a instância da questão, é também iniciada a contagem decrescente de um temporizador, cuja duração corresponde à duração definida no momento de criação da questão. Esta contagem do tempo restante ocorre apenas no *browser* do utilizador e resulta da evocação da função JavaScript exemplificada na listagem 5.7.

Listagem 5.7: Controlo do tempo restante de uma questão.

```
function startTimer(duration, display) {
    var start = Date.now(), diff, minutes, seconds;

    function stopTimer() {
        $.ajax({
            type: "POST",
            url: "stop_quiz/",
            data: { 'id': '{{ object.id }}',
                  'start_date': '{{ object.start_date }}',
                  'right_ans': '{{ object.right_ans }}' },
            dataType: 'json',
            success: function(data, textStatus) {
                console.log(data);
            }
        });
        clearInterval(time);
    }

    function timer() {
        diff = duration - (((Date.now() - start) / 1000) | 0);
        if (diff <= 0) {
            diff = 0;
            stopTimer();
        }
        display.textContent = diff;
    };

    timer();
    time = setInterval(timer, 1000);
}
```

Analisando o excerto de código da listagem 5.7, verifica-se que este cumpre duas funções: a primeira é fazer a contagem decrescente do tempo restante da questão, e actualizar o *timer* apresentado na página (`display.textContent = diff;`); a segunda é, após o contador chegar a zero, evocar a vista `stop_quiz`, responsável por processar as respostas recebidas durante a realização da questão. Ao evocar esta vista, a função passa-lhe, em formato JSON, a informação referente à questão a ser realizada, através da variável `object.id`, que corresponde à chave primária do objecto `quiz` em execução. Uma vez que tanto a vista `start_quiz` como a vista `stop_quiz` são evocadas pelo método AJAX, não é necessário actualizar a página para que as acções acima mencionadas tomem efeito, permitindo uma maior fluidez da realização dos inquéritos.

5.4.2 Recepção das Respostas

A qualquer instante, independentemente de estar a decorrer, ou não, a realização de um inquérito, a aplicação recebe e regista as respostas vindas dos terminais individuais. Estas respostas são enviadas para a vista `quiz_response`, identificada pelo URL `http://192.168.0.2:8000/quiz/response/`, e armazenadas no modelo `Answer`. Esta vista encontra-se detalhada na listagem 5.8, onde se verifica que, mais uma vez, é necessário não utilizar o *token* CSRF, já que as respostas são enviadas a partir dos terminais, e não a partir de um *browser*, ou seja, não têm *cookies*. Verifica-se também que é definido o método *Post* para o pedido HTTP, uma vez que se pretende alterar a informação da base de dados.

Listagem 5.8: Vista `quiz_response`.

```
@csrf_exempt
@require_http_methods(["POST"])
def quiz_response(request, *args, **kwargs):
    body = request.body.decode('utf-8')

    m = re.search('uid: (.+?)', body)
    if m:
        card_id = m.group(1)

    m = re.search('mac: (.+?)', body)
    if m:
        mac_id = m.group(1)

    m = re.search('ans: (.+?)}', body)
    if m:
        ans = m.group(1)

    answer = Answer(uid=card_id,
                   mac=mac_id,
                   ans=ans,
                   date_time=datetime.datetime.now())
    answer.save()

    to_return = {'type': 'success', 'msg': 'done', 'code': 200}
    return HttpResponse(json.dumps(to_return),
                       content_type='application/json')
```

A mensagem é enviada pelos terminais em formato JSON, e fica armazenada na variável `body` da função descrita. É então necessário retirar a informação pretendida do corpo da mensagem, nomeadamente a identificação única do cartão de estudante do aluno que submeteu a resposta, o endereço MAC do terminal utilizado e por fim qual a resposta submetida. Um exemplo de uma mensagem recebida é: "{uid: 04 43 C6 32 34 31 80, mac: B4:E6:2D:3F:08:30, ans: D}", onde 43 C6 32 34 31 80 corresponde à identificação do cartão de estudante do aluno, B4:E6:2D:3F:08:30 diz respeito ao endereço MAC do terminal utilizado e D traduz a opção seleccionada pelo aluno. A listagem 5.9 traduz o processo de extracção da identificação do cartão do aluno da mensagem recebida, onde se extrai o conteúdo da mensagem entre `uid` e a vírgula seguinte da variável `body`. O mesmo processo é adoptado para a extracção do endereço MAC do terminal e resposta seleccionada.

Listagem 5.9: Extracção da identificação do cartão da mensagem enviada pelo terminal.

```
m = re.search('uid: (.+?) ,', body)
if m:
    card_id = m.group(1)
```

Uma vez extraída a informação pretendida da mensagem, é criado um novo registo no modelo `Answer` com essa informação, registando ainda a data e hora do momento de registo na base de dados. Por fim, é enviada uma mensagem de sucesso de volta para o terminal, de forma a confirmar que a resposta foi devidamente recebida e armazenada.

5.4.3 Avaliação das Respostas

Uma vez terminada a duração de uma questão, ou seja, quando o contador apresentado em modo *fullscreen* chega a zero, é evocada a vista responsável por processar as respostas recebidas (`stop_quiz`).

A primeira acção executada nesta vista corresponde à migração das respostas para o modelo `AnswerProcessing`, de forma a que alguma resposta que chegue durante esta etapa não interfira no processamento das respostas da questão, como se pode ver na listagem 5.10.

Listagem 5.10: Migração das respostas do modelo `Answer` para o modelo `AnswerProcessing`.

```
answers = Answer.objects.all().order_by('id')
for answer in answers:
    try:
        student = Student.objects.get(uid=answer.uid)
        mac_id = Terminal.objects.get(mac=answer.mac)
        copy = AnswerProcessing(nmec=student.nmec,
                                mac=mac_id,
                                ans=answer.ans,
                                date_time=answer.date_time)

        copy.save()
    except Student.DoesNotExist:
        pass
    except Terminal.DoesNotExist:
        pass

Answer.objects.all().delete()
```

Analisando o excerto de código anterior, verifica-se que não é copiada a identificação do cartão de estudante do aluno, mas sim o seu número mecanográfico, obtido através da expressão `student = Student.objects.get(uid=answer.uid)`. De seguida, é verificado se o endereço MAC do terminal consta da tabela de terminais registados (`mac_id = Terminal.objects.get(mac=answer.mac)`). Caso tanto o aluno como o terminal se encontrem registados na base de dados do sistema, a resposta é copiada para o modelo `AnswerProcessing` onde será posteriormente processada e avaliada. Caso tal não se verifique, a resposta é ignorada. Por fim, são apagados todos os registos do modelo `Answer`, de forma a que não existam registos prévios na realização de uma nova questão.

Uma vez migradas as respostas, é verificada qual a questão em execução, cuja chave primária é passada para a vista aquando da sua evocação, como descrito na secção 5.4.1. Uma vez que esta informação é transmitida em formato JSON, o processo de extracção da informação relevante é em tudo semelhante ao descrito na secção anterior, como demonstra o excerto de código da listagem 5.11, onde a expressão `quiz = get_object_or_404(Quiz, id=quiz_id)` retorna o registo do modelo `Quiz` pretendido, caso exista, ou uma mensagem de erro quando se verifica o contrário.

Listagem 5.11: Extracção da identificação da questão activa.

```
body = request.body.decode('utf-8')
m = re.search('id=(.+?)&', body)
if m:
    quiz_id = m.group(1)

quiz = get_object_or_404(Quiz, id=quiz_id)
```

De seguida, é verificada qual a lição (`lesson`) activa do utilizador, ou professor, e criada uma nova instância (`session`) da questão em execução, como mostra a listagem 5.12. A determinação da lição activa é importante para o processamento das respostas, uma vez que é nela que se encontra definida qual a resposta que deve ser contabilizada, se a primeira ou a última submetida pelo aluno.

Listagem 5.12: Verificação da lição activa.

```
lesson = Lesson.objects.filter(user=quiz.author).latest('id')
session = Session(quiz=quiz, lesson=lesson)
session.save()
```

Segue-se o processamento das respostas. O código da listagem 5.13 exemplifica como é processada uma resposta, no caso de se tratar de uma questão não anónima, ou autenticada. A primeira etapa passa por verificar qual a resposta definida como válida. Se for a última submetida pelo aluno, os registos do modelo `AnswerProcessing` são ordenados, primeiro por número mecanográfico, e depois por ordem decrescente da sua chave de identificação, criando uma lista designada `ans_inverse`. Caso a resposta válida seja a primeira submetida pelos alunos, é criada a mesma lista, mas ordenada de forma crescente da chave primária dos registos do modelo.

De seguida, é realizada uma verificação do número mecanográfico do aluno a quem corresponde cada resposta, para a qual é percorrida a lista de registos `ans_inverse`. Uma vez que se trata de uma questão autenticada, é necessário apagar todas as respostas submetidas pelo aluno 00000, que correspondem a uma resposta submetida sem o cartão de estudante estar presente no terminal. De seguida, é necessário verificar se já foi

registada uma resposta do aluno em causa, ou seja, se é a primeira vez que o algoritmo vê um aluno ou não. Se o algoritmo já tiver registado uma resposta do mesmo aluno, apaga então essa resposta. Desta forma, assegura-se que para cada aluno é apenas registada uma resposta. Caso se tratasse de uma questão anónima, a verificação do aluno seria ignorada, uma vez que todas as respostas ficariam associadas ao aluno 00000. Por fim, é adoptado o mesmo procedimento para a verificação do endereço MAC do terminal individual que submeteu a resposta, de forma a que apenas a primeira resposta submetida por terminal seja validada.

Listagem 5.13: Processamento das respostas.

```

if quiz.anonymous == "No":

    lastSeenNMEC = float('-Inf')
    if lesson.valid_ans == "Last":
        ans_inverse = AnswerProcessing.objects.all().order_by(
            'nmecc',
            '-id'
        )
    else:
        ans_inverse = AnswerProcessing.objects.all().order_by(
            'nmecc',
            'id'
        )
    for answer in ans_inverse:
        if answer.nmecc == "00000":
            answer.delete()
        elif answer.nmecc == lastSeenNMEC:
            answer.delete()
        else:
            lastSeenNMEC = answer.nmecc

    lastSeenMAC = float('-Inf')
    answers_processing = AnswerProcessing.objects.all().order_by('id')
    for answer in answers_processing:
        if answer.mac == lastSeenMAC:
            answer.delete()
        else:
            lastSeenMAC = answer.mac

```

Dá-se então por concluído o processamento das respostas. Segue-se a avaliação das respostas, e conseqüente registo dos resultados. O modelo `AnswerProcessing` contém, nesta fase, apenas uma resposta dada por aluno e por terminal, e serão estas as respostas que serão avaliadas. É necessário criar uma nova lista de respostas, através da expressão `answers_processed = AnswerProcessing.objects.all().order_by('id')`.

Para cada item desta lista é determinado, em primeiro lugar, o tempo que decorreu entre o começo da questão e a recepção da resposta, determinado a partir da expressão `answer.date_time - quiz.start_date`, onde `answer` corresponde ao item em análise da lista mencionada acima. Esta diferença de tempo é de seguida convertida para segundos, de forma a poder ser depois registada no modelo `Results`. Segue-se a comparação entre

a resposta submetida pelo aluno e a resposta correcta definida pelo professor aquando da criação da questão, como mostra a listagem 5.14.

Listagem 5.14: Avaliação das respostas.

```
if answer.ans == quiz.right_ans:
    evaluation = "right"
else:
    evaluation = "wrong"
```

Por fim, são criados os registos das respostas no modelo `Results`, com base na informação processada. No caso de se tratar de uma questão anónima, é registado o valor 00000 da identificação do aluno, ao passo que no caso de se tratar de uma questão autenticada, é registado o seu número mecanográfico. Para além disso, neste tipo de questões é ainda criado um registo, por resposta avaliada, no modelo `LessonStudent`, onde é indicado o aluno em questão e se submeteu, ou não, a resposta certa.

Uma vez avaliadas todas as respostas, é calculado o número de respostas certas e erradas dadas pelo aluno às questões dessa mesma lição, armazenando o resultado no modelo `LessonStudent`. Uma vez concluído, são apagados todos os registos do modelo `AnswerProcessing`, e é enviada uma mensagem de sucesso de volta para o *browser* do utilizador da aplicação (professor).

5.4.4 Acções no Cliente e Servidor

A maioria das acções lógicas da aplicação ocorrem no servidor, tais como o registo de novas entradas nos modelos descritos. Contudo, existem certas acções que ocorrem no cliente, em particular no *browser* do utilizador da aplicação.

Uma destas acções diz respeito à exportação dos resultados dos inquéritos, quer se trate dos resultados de uma questão em particular, ou os resultados acumulados de uma dada lição. Aquando da visualização dos resultados, é possível exportar os mesmos em formato Excel. Para tal, em vez de se aceder à base de dados para exportar os resultados pretendidos, a solução implementada faz uso dos resultados renderizados na página, apresentados sob a forma de uma tabela, designada `myTable`. A exportação é assegurada através de uma função JS, embutida no *template* da página, que percorre cada linha da tabela, e formata o seu conteúdo de forma a corresponder a uma tabela Excel. Esta abordagem permite simplificar o processo de exportação, uma vez que não é necessário filtrar os resultados a exportar, já que são considerados apenas os resultados apresentados no ecrã.

Outra acção que ocorre no cliente é a determinação de quando chega ao fim a duração de uma questão. Uma vez que ao iniciar uma nova instância (`session`) de uma questão é registada a data e hora de criação desse registo, seria intuitivo utilizar essa informação para determinar se um aluno respondeu, ou não, dentro do tempo limite. Contudo, não é isso que acontece. Uma vez que o *display* do tempo ocorre no *browser* do professor, é possível que ocorram atrasos na apresentação e contabilização do tempo restante. Este atraso será mais significativo em máquinas com menor capacidade de processamento, o que poderia induzir em erro os alunos participantes. É muito comum os *browsers* usarem recursos computacionais excessivos, o que pode levar a pequenos congelamentos da página. Assim, seria possível a duração de uma questão terminar enquanto são ainda apresentados alguns segundos no ecrã. De forma a combater este cenário, a

vista `stop_quiz` é apenas evocada quando o tempo apresentado no ecrã chegar a zero, mesmo que isso implique mais alguns segundos de duração do que aqueles definidos durante a criação da questão. Para além disso, o tempo de resposta calculado aquando do processamento das respostas é meramente informativo, não sendo contabilizado na avaliação da resposta. Esta abordagem salvaguarda a transparência dos inquéritos realizados, uma vez que em circunstância alguma os alunos são induzidos em erro quanto ao tempo restante.

Capítulo 6

Desenvolvimento do Terminal Individual

Embora o terminal desenvolvido seja baseado no trabalho realizado por Mamede [49], tal como referido na secção 3.3.2, é necessário atender a algumas questões que haviam sido deixadas em aberto. Uma dessas questões diz respeito à alimentação dos terminais. O sistema é alimentado a uma tensão de 5 V, o que pode acontecer através de um cabo USB. Contudo, esta solução não é ideal, uma vez que obriga a que o terminal assuma dimensões maiores ou, em alternativa, requer uma fonte de alimentação externa. Para além disso, um dos objectivos da presente dissertação passa por desenvolver uma solução para fixar os botões e o cartão de identificação, quando inserido no terminal. Assim, torna-se necessário desenvolver uma estrutura mecânica que sirva tanto de suporte como de protecção aos componentes internos do sistema.

Outra questão abordada diz respeito à necessidade de actualizar o *firmware* do terminal individual já desenvolvido de forma a que, por um lado, seja capaz de comunicar com o servidor seguindo o protocolo TCP, e por outro permita o envio de respostas tanto quando o cartão de estudante se encontra inserido, como em situações que tal não aconteça. Além disso, o terminal deve ser capaz de identificar se o cartão se encontra inserido no momento da submissão da resposta, de forma a aumentar a fiabilidade do sistema, no que respeita à correcta identificação dos inquiridos.

O presente capítulo visa detalhar a constituição dos terminais desenvolvidos, realçando e justificando as alterações feitas à versão desenvolvida por Mamede, no que à selecção de componentes diz respeito. Neste capítulo é ainda justificada a solução adoptada para a alimentação dos terminais, seguindo-se o esquema eléctrico dos mesmos e a estrutura de suporte, ou caixa de protecção, projectada para acomodar os componentes eléctricos que os constituem. Por fim, é exposta a lógica de funcionamento dos terminais, através de uma análise detalhada ao seu *firmware*.

6.1 Hardware

Tendo como base o terminal desenvolvido por Mamede [49], em 2018, a abordagem relativa ao desenvolvimento de novos terminais passa por construir em cima do que já foi feito, ao invés de criar um terminal novo completamente de raiz. Assim, a selecção de componentes, tais como o microcontrolador, módulo leitor de cartões, *encoder*, bo-

tões, resistências e LED's coincide com os componentes constituintes da segunda versão do terminal. Esta abordagem permite reduzir o tempo despendido no seu desenvolvimento, sem sacrificar o seu correcto funcionamento, como ficou demonstrado no trabalho realizado pelo seu anterior desenvolvedor do sistema, possibilitando ainda que seja dedicada mais atenção às questões que haviam sido deixado em aberto, ou cuja completa reformulação era imperativa.

As secções seguintes descrevem, numa primeira instância, os componentes constituintes do terminal individual, cuja selecção havia sido feita por Mamede. Como tal, a informação presente nas secções referidas resultam da leitura do relatório do trabalho por si desenvolvido [49]. De seguida, é abordada a alimentação dos terminais, seguindo-se uma representação do esquema eléctrico dos mesmos, bem como da PCB e estrutura de fixação e protecção concebidas.

6.1.1 Microcontrolador

O microcontrolador presente no terminal individual é responsável por todas as ações lógicas do mesmo, bem como por estabelecer as comunicações *wireless* com o servidor. O microcontrolador NodeMCU v3, utilizado por Mamede na sua versão do terminal, é integrado com o módulo de comunicações Wi-Fi ESP8266. Este microcontrolador possui ainda a vantagem de ter 128 kB de memória e um armazenamento de 4 MB, que se comprovou ser mais do que suficiente para a aplicação em questão. Mais ainda, este microcontrolador é alimentado a 5 V e 800 mA, e possui três saídas de 3.3 V, utilizadas para alimentar os restantes constituintes do terminal. Contudo, não foi este o microcontrolador utilizado nos terminais desenvolvidos, tendo-se optado pelo NodeMCU v2. Este é, basicamente, uma versão mais pequena do terminal utilizado por Mamede, sendo que a principal diferença, para além das suas dimensões, reside na falta de uma saída de 5 V. As imagens da figura 6.1 ilustram o mapa de pinos destes dois microcontroladores, onde se verifica que, para além da saída de 5 V mencionada e da existência de mais um pino de massa, não existem quaisquer outras diferenças entre ambos.

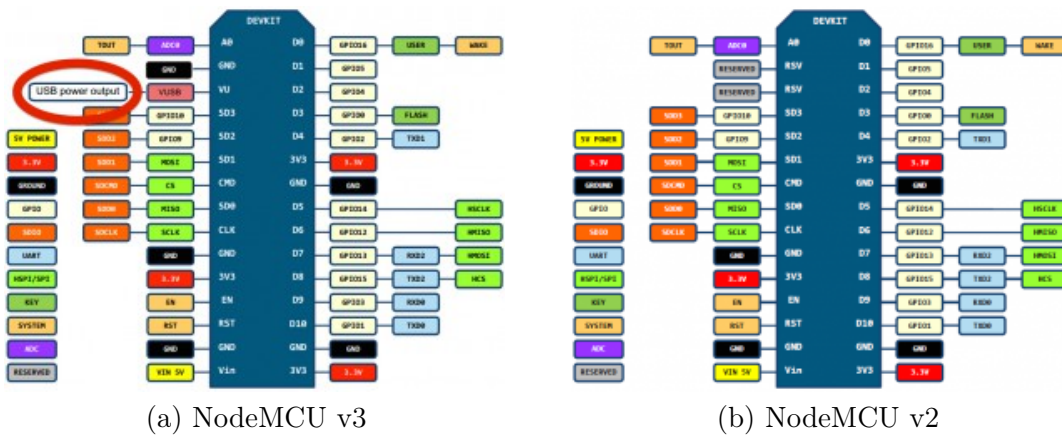


Figura 6.1: Comparação do mapa de pinos das duas versões do microcontrolador.

Como referido acima, a opção pelo microcontrolador NodeMCU v2 reside no facto de este ter dimensões mais reduzidas, o que por sua vez permite reduzir as dimensões do terminal individual desenvolvido. A imagem da figura 6.2 visa permitir a comparação das dimensões de ambos.

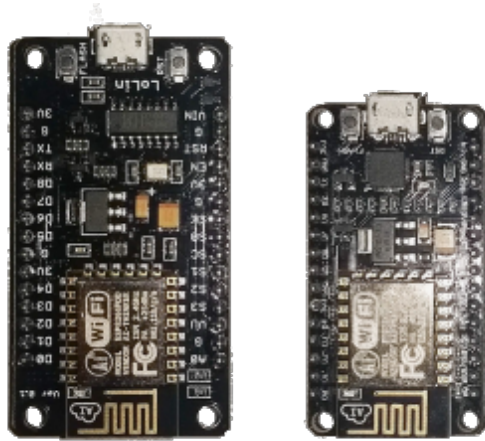


Figura 6.2: Comparação das dimensões dos microcontroladores NodeMCU v3 e v2.

6.1.2 Módulo RFID

A leitura dos cartões RFID é alcançada através do módulo Mifare RC522 (figura 6.3), capaz de comunicar com cartões até uma distância de 1 cm, através da criação de um campo magnético de 13.56 MHz, e que comunica com o microcontrolador por SPI. A escolha deste módulo leitor de cartões reside no facto de ser um dos mais utilizados para projectos desenvolvidos em Arduino, resultando numa documentação muito vasta e inúmeros exemplos e bibliotecas disponíveis na Internet.

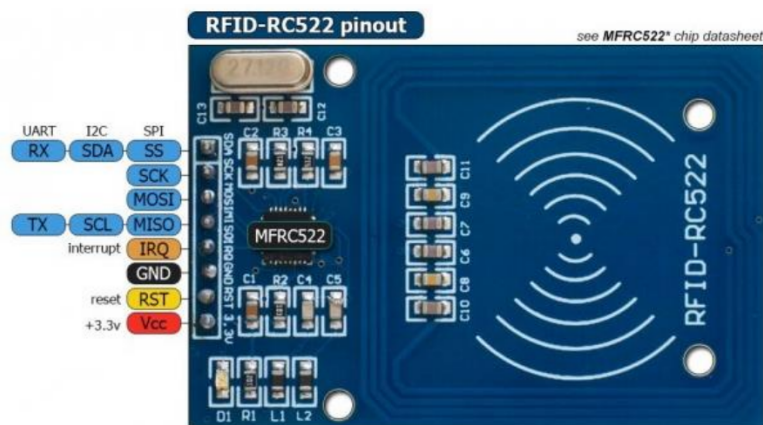


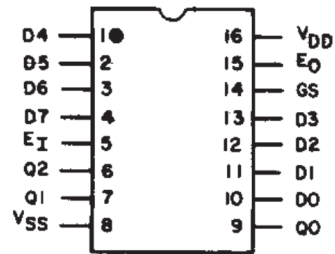
Figura 6.3: Módulo Mifare RC522.

6.1.3 8-bit Priority Encoder

A interacção entre o utilizador e o terminal é realizada através da presença de cinco botões, que correspondem às cinco opções de resposta das questões colocadas através da aplicação *web*. Contudo, existem apenas três entradas livres no microcontrolador, uma das quais é analógica e as restantes digitais. Assim, torna-se necessária a utilização de um codificador de 8 bits para 3, de forma a codificar os cinco botões existentes no terminal em apenas três entradas analógicas. O codificador escolhido para o desenvolvimento dos terminais é o CD4532BE da Texas Instruments (figura 6.4), que é equivalente ao Philips HEF4532BP, utilizado no terminal desenvolvido por Mamede. Este é um codificador CMOS prioritário de 8 bits, o que significa que os pinos de entrada de número maior têm prioridade sobre os de número menor. Para além disso, este codificador pode ser alimentado directamente com os 3.3 V fornecidos pelo microcontrolador.



(a) Codificador CD4532BE



(b) Mapa de pinos do codificador [57]

Figura 6.4: Codificador Texas Instruments CD4532BE.

Os botões são ligados aos pinos D1 a D5. O motivo por que nenhum se encontra ligado ao pino D0 deve-se ao facto de, sem outra entrada analógica disponível no microcontrolador, não seria possível distinguir o cenário em que nenhum botão é premido, como se pode ver a partir da tabela de verdades do codificador (tabela 6.1).

Tabela 6.1: Tabela de verdades do codificador CD4532BE [57].

Input									Output				
E _I	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	GS	Q ₂	Q ₁	Q ₀	E _O
0	X	X	X	X	X	X	X	X	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	1
1	1	X	X	X	X	X	X	X	1	1	1	1	0
1	0	1	X	X	X	X	X	X	1	1	1	0	0
1	0	0	1	X	X	X	X	X	1	1	0	1	0
1	0	0	0	1	X	X	X	X	1	1	0	0	0
1	0	0	0	0	1	X	X	X	1	0	1	1	0
1	0	0	0	0	0	1	X	X	1	0	1	0	0
1	0	0	0	0	0	0	1	X	1	0	0	1	0
1	0	0	0	0	0	0	0	1	1	0	0	0	0

X = Don't Care Logic 1 ≡ High Logic 0 ≡ Low

6.1.4 Botões e LED's

A interface física entre o utilizador e o terminal é alcançada através de cinco botões de pressão, de dimensões 6x6x13 mm e 4 pinos, como mostra a imagem da figura 6.5. Mais uma vez, a diferença entre os botões utilizados no desenvolvimento de novos terminais e os utilizados por Mamede reside apenas nas suas dimensões. O facto de estes botões terem uma altura de 13 mm permite que não seja necessária a utilização de uma extensão, como acontece na versão anterior do terminal. Esta altura permite que o botão fique ligeiramente de fora da caixa prototipada, permitindo assim uma mais fácil utilização dos terminais e maior *feedback* da acção. A cada botão é ainda associada uma resistência de *pulldown* de 10 k Ω , que força a entrada digital do codificador, a que o botão se encontra ligado, a zero (0) quando este se encontra no seu estado de repouso.



Figura 6.5: Botão de pressão unipolar de uma via 6x6x13 mm.

De forma a dar aos utilizadores dos terminais algum *feedback* do funcionamento dos mesmos, são utilizados dois LED's (figura 6.6), um vermelho e outro verde, que indicam o estado da conexão ao servidor e retorno das acções lógicas do terminal, respectivamente. Em conjunto com os LED's, são utilizadas resistências de 510 Ω , uma por LED, de modo a se obter uma boa intensidade de iluminação.

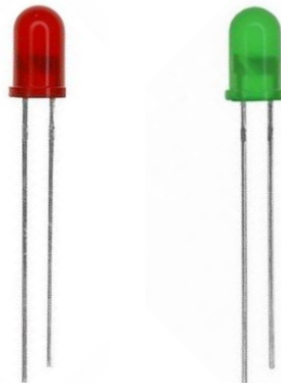


Figura 6.6: LED's de 5 mm utilizados nos terminais individuais.

6.1.5 Alimentação

Como referido no início deste capítulo, o microcontrolador é alimentado a uma tensão de 5 V, que por sua vez alimenta os restantes componentes do terminal a uma tensão de 3.3 V. Como tal, torna-se necessária uma solução de alimentação capaz de assegurar 5 V constantes. Assim, os terminais são alimentados por uma bateria de íon lítio 18650 recarregável, com uma tensão nominal de 3.7 V. As baterias, por natureza, não têm uma tensão constante, uma vez que esta depende da sua carga, o que, associado ao facto de se tratar de uma bateria de tensão mais baixa que a desejada, requer a utilização de um conversor de tensão, de forma a se poder obter os 5 V pretendidos.

As baterias lítio requerem cuidados especiais, uma vez que existe sempre o risco de explosão durante a sua utilização. Para além disso, não se deve permitir que descarreguem completamente, uma vez que se tornam inutilizáveis caso tal aconteça. Assim, é necessário tomar precauções de forma a salvaguardar o bom funcionamento das mesmas e a segurança de quem as utiliza. A solução encontrada para este problema passa pela utilização de uma *battery shield* da WEMOS (figura 6.7), que para além de salvaguardar todas as questões relativas à segurança e integridade da bateria, fornece um sistema de conversão de tensão, possuindo três saídas de 5 V, e outras tantas de 3.3 V. Para além disso, permite a alimentação de dispositivos via USB, possuindo até um interruptor associado, permitindo uma fácil utilização da mesma nos períodos de desenvolvimento dos terminais. Outra vantagem deste escudo é o facto de servir também para recarregar a bateria através de uma entrada micro-USB, mesmo com o sistema em funcionamento. Assim, utilizando uma das saídas de 5 V do escudo, fica assegurada uma tensão de alimentação constante, sendo que a mesma é cortada quando tal não for possível, salvaguardando que a bateria nunca fica completamente descarregada. É ainda utilizado um interruptor unipolar de uma via (figura 6.8), de forma a permitir ligar e desligar o terminal quando necessário. Este interruptor encontra-se situado entre a saída de 5 V do escudo da bateria e a respectiva entrada na PCB projectada.



(a) Escudo da bateria



(b) Bateria de íões de lítio 18650

Figura 6.7: Alimentação dos terminais individuais.

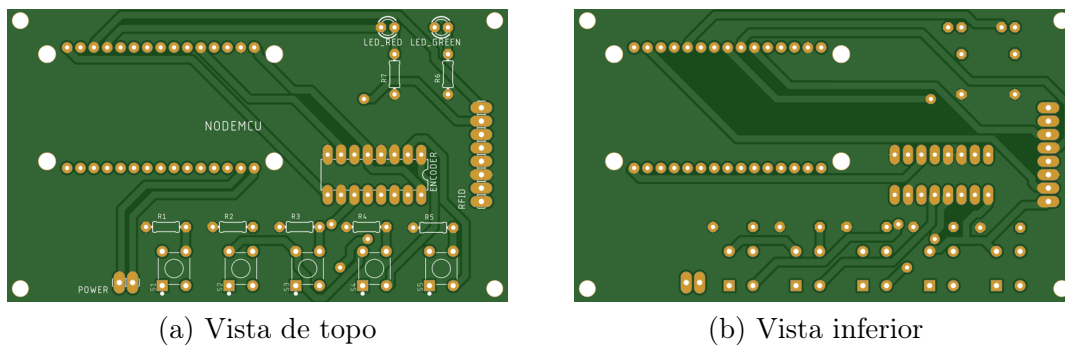
Tudo isto, associado ao seu baixo custo e fácil implementação, tornam esta uma solução bastante viável para o desenvolvimento de protótipos dos terminais individuais. Não obstante das vantagens referidas, a principal desvantagem desta solução prende-se à sua grande dimensão, resultante das baterias utilizadas. Embora nesta fase de desenvolvimento do sistema esta não seja uma questão crucial, existirá sempre a necessidade de ser revista futuramente em estágios mais avançados do projecto.



Figura 6.8: Interruptor unipolar de uma via.

6.1.6 PCB

O último passo para a completar o circuito eléctrico do terminal consiste no desenho e impressão de uma PCB, responsável por estabelecer as ligações entre os componentes e assegurar a correcta fixação dos mesmo. Para o desenho da PCB foi utilizado o *software* EAGLE, tendo como base o desenho criado por Mamede. Quanto à sua impressão, foi necessário ter em atenção as dimensões da mesma, de forma a manter o seu preço de produção baixo (as dimensões deveriam ser inferiores a 100x100 mm), e reduzir as dimensões do terminal face à versão desenvolvida por Mamede. As imagens da figura 6.9 ilustram a PCB desenvolvida. O diagrama do circuito eléctrico encontra-se disponível no anexo A para consulta.



(a) Vista de topo

(b) Vista inferior

Figura 6.9: PCB desenvolvida.

6.1.7 Estrutura de Fixação e Protecção

De forma a acomodar os componentes eléctricos dos terminais desenvolvidos, torna-se necessário o projecto de uma caixa que, para além de proteger os mesmos, serve o propósito de os fixar. Esta caixa foi projectada de forma a ser fabricada por impressão 3D e, para além dos propósitos já mencionados, deve ainda prever a fixação do cartão de estudante e fixação dos LED's e botões do terminal. Por fim, deve ainda contemplar uma entrada para alimentação da bateria e um espaço para acomodar o interruptor presente no terminal. A imagem da figura 6.10 visa ilustrar os componentes eléctricos de um terminal individual montados na caixa desenvolvida, ao passo que na figura 6.11 é possível ver o terminal completamente montado. O desenho técnico da mesma encontra-se disponível do anexo B para consulta.

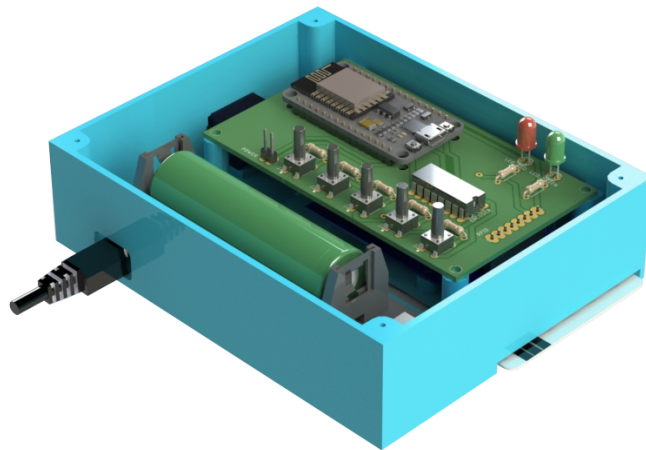


Figura 6.10: Renderização do terminal individual, sem tampa (conjunto).

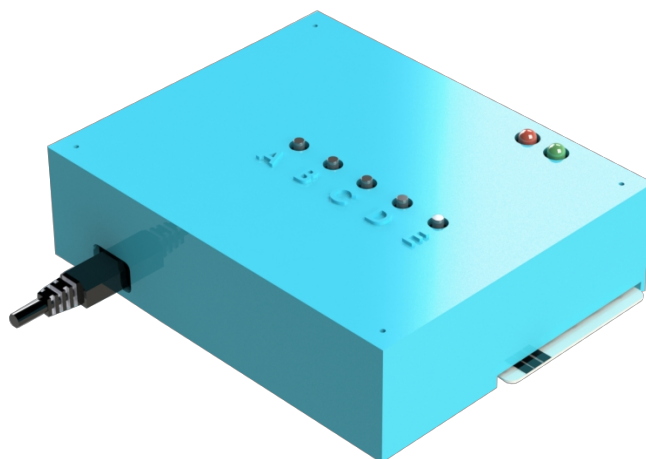


Figura 6.11: Renderização do terminal individual, com tampa (conjunto).

6.1.8 Lista de Materiais

Uma das premissas do desenvolvimento dos terminais individuais é que estes devem ter um custo de produção reduzido. Apesar de os terminais desenvolvidos serem apenas protótipos, foi possível obter um custo aproximado de 46 € por cada um. É importante referir que, de forma a agilizar o processo de desenvolvimento dos terminais, foram tomadas opções na escolha dos materiais e fornecedores que não são ótimas do ponto de vista económico, tais como a opção por uma bateria de iões de lítio 18650, ou a escolha do fornecedor dos microcontroladores e leitores RFID (é possível encontrar estes componentes no mercado a cerca de um terço do preço). Também o custo de produção das caixas protectoras dos terminais pode ser reduzido, uma vez que o mesmo depende do preço de compra do filamento utilizado na impressão 3D. Assim, embora o custo de produção por caixa obtido não seja ideal, não é de todo exorbitante, podendo mesmo a vir a ser reduzido na produção de séries maiores.

Na tabela 6.2 encontram-se discriminados todos os componentes constituintes dos terminais individuais, bem como as quantidades necessárias para a construção de um terminal e o preço indicativo dos seus componentes. Os preços apresentados têm como base o custo de aquisição dos componentes durante a realização desta dissertação.

Tabela 6.2: Tabela de materiais para um terminal individual.

Componente	Quantidade	Preço unitário
Microcontrolador NodeMCU v2	1	12,55 €
Leitor de cartões Mifare RC522	1	5,54 €
Conector 1x8 macho-fêmea	1	1,69 €
Botão de pressão unipolar de uma via 6x6x13 mm	5	0,09 €
Resistência 10 k Ω	5	0,01 €
LED vermelho	1	0,37 €
LED verde	1	0,19 €
Resistência 510 Ω	2	0,02 €
Codificador CD4532BE	1	0,39 €
PCB	1	3,20 €
Interruptor unipolar de uma via	1	0,71 €
Bateria de iões de lítio 18650	1	10,94 €
WEMOS battery shield 18650 v3	1	2,59 €
Impressão da caixa protectora	1	3,60 €
Impressão da tampa da caixa protectora	1	0,90 €

6.2 Firmware

O *firmware* do terminal individual é responsável por definir todas as acções. Uma vez que o microcontrolador utilizado vem incorporado com o módulo ESP8266, é possível utilizar o IDE da Arduino para desenvolver, compilar e transferir o código para o microcontrolador.

A primeira etapa do desenvolvimento do *firmware* passa por definir as entradas e saídas do microcontrolador utilizado, que correspondem às mesmas utilizadas por Mamede no trabalho por si desenvolvido [49], e pode ser consultadas na tabela 6.3.

Tabela 6.3: Tabela de ligações do microcontrolador.

I/O's (microcontrolador)	Componente	Pino
D0	<i>Encoder</i>	Q0
D1	Módulo RFID	RST
D2	Módulo RFID	SDA
D3	LED vermelho	
D4	LED verde	
D5	Módulo RFID	SCK
D6	Módulo RFID	MISO
D7	Módulo RFID	MOSI
D8	<i>Encoder</i>	Q1
A0	<i>Encoder</i>	Q2

De acordo com Mamede, os pinos D3 e D4 não podem ser utilizados como entradas digitais, uma vez que se encontram associados à comunicação e *flash* do microcontrolador, sendo por esse motivo apenas utilizados como saídas para os LED's. Ainda segundo o mesmo, o módulo leitor de cartões é capaz de comunicar por SPI e I²C, sendo, contudo, utilizado o primeiro protocolo como modo de comunicação.

Uma vez definidos os I/O's do sistema, e o protocolo de comunicação com o módulo leitor de cartões, é estabelecida a conexão à rede, seguindo-se a conexão ao servidor, através do seu endereço IP e da porta 8000. Uma vez conectado ao servidor, o terminal entra em *loop*, até que seja desligado.

O diagrama da figura 6.12 ilustra a lógica de funcionamento dos terminais, onde *card_read* representa se um cartão foi lido (=1) ou não (=0), *card_id* traduz a UID do cartão lido e *control* indica se o cartão ainda se encontra presente no terminal (se <= 2).

Nas secções que se seguem é explicado em maior detalhe algumas das funções e procedimentos do *firmware*, como a verificação da presença do cartão RFID, seguindo-se uma reflexão sobre algumas alternativas aos métodos implementados, nomeadamente no que respeita à validação do cartão e ao envio das mensagens.

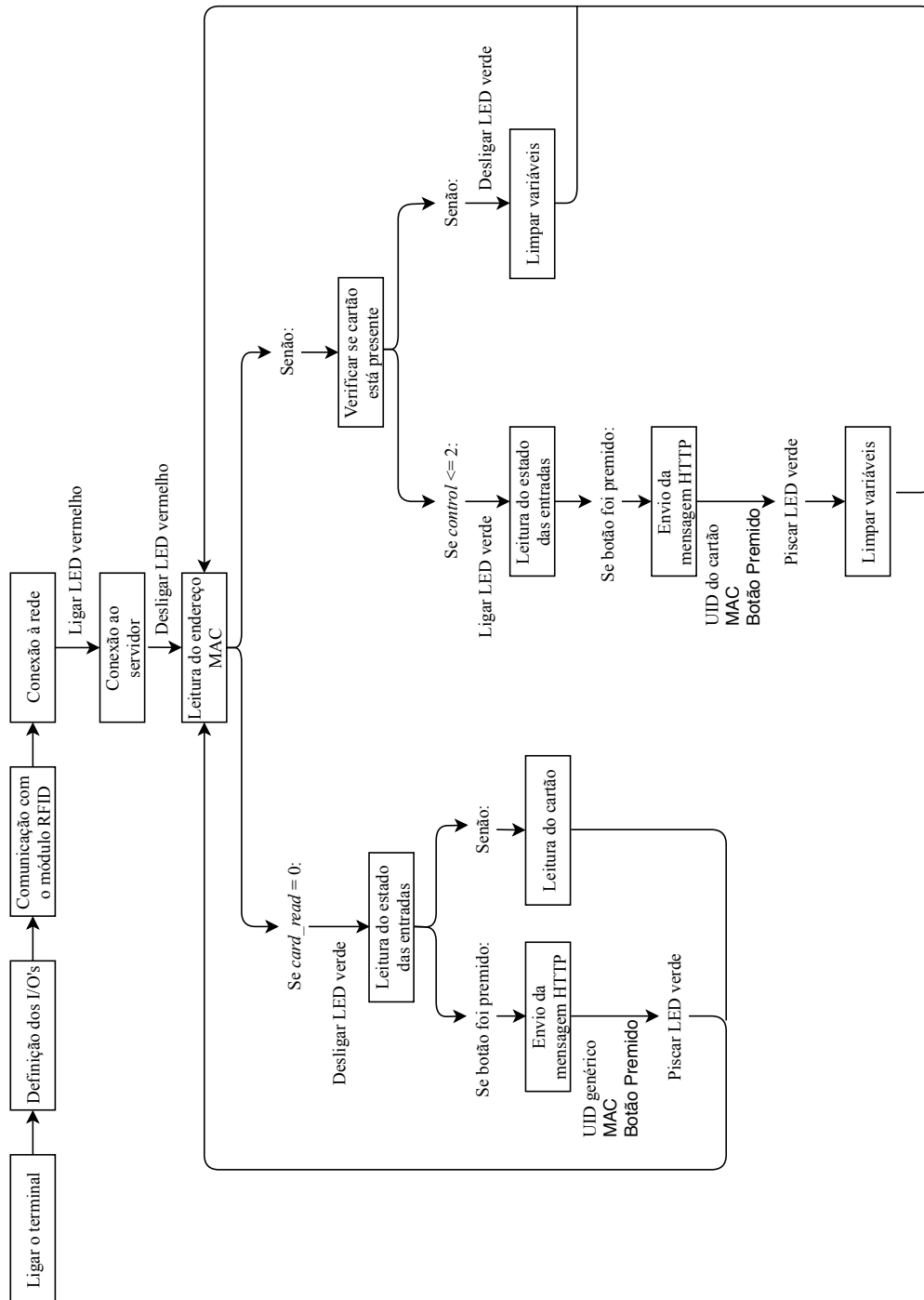


Figura 6.12: Diagrama do *firmware* dos terminais individuais.

6.2.1 Leitura da UID do Cartão

A função descrita na listagem 6.1 permite obter a identificação única do cartão inserido no terminal, formatando ainda a mesma para o formato que será posteriormente processado pelo servidor. Esta função deriva do *firmware* desenvolvido por Mamede, em 2018, e começa por limpar as variáveis `content`, que contém a informação lida do cartão RFID, e `UID`, que contém essa informação já devidamente formatada. De seguida, é verificado se existe um cartão presente no terminal o que, caso se verifique, leva à leitura imediata a sua identificação única, armazenando-a na variável `content`. Por fim, a informação é transformada para o formato desejado (ex: 0F FF F0 00 0F FF F0), e convertida em *string*.

Listagem 6.1: Obtenção da UID do cartão de estudante.

```
String read_card(void) {
    String content = "";
    char UID[] = "";

    if (!mfr522.PICC_IsNewCardPresent())
        return "";
    if (!mfr522.PICC_ReadCardSerial())
        return "";

    for (byte i=0; i<mfr522.uid.size; i++) {
        content.concat(
            String(mfr522.uid.uidByte[i]<0x10 ? " 0" : " ")
        );
        content.concat(String(mfr522.uid.uidByte[i], HEX));
    }

    content.toUpperCase();
    (content.substring(1)).toCharArray(UID, 21);

    String card_id(UID);

    return card_id;
}
```

6.2.2 Presença Contínua do Cartão

A verificação da presença do cartão de estudante no terminal é realizada através da função `is_card_present()`, descrita na listagem 6.2. A biblioteca utilizada para controlar o módulo leitor de cartões contém a função `mfr522.PICC_IsNewCardPresent()`, que permite saber se existe algum cartão presente no raio de acção do leitor. Desta forma, a função criada permite detectar se o cartão foi removido, retornando assim o valor 3. Isto acontece porque, a cada iteração do ciclo `for`, o valor retornado da função `mfr522.PICC_IsNewCardPresent()` é sempre 1 se o cartão tiver sido removido. Caso contrário, o valor pode alternar entre 0 e 1. Assim, caso a variável `control` assumira um valor menor que 3 no final do ciclo iterativo, significa que o cartão ainda se encontra presente no terminal.

Listagem 6.2: Verificação da presença do cartão de estudante.

```
uint8_t is_card_present(uint8_t control) {  
  
    control = 0;  
    for (byte i=0; i<3; i++) {  
        if (!mfrc522.PICC_IsNewCardPresent())  
            control = control + 1;  
    }  
    return control;  
}
```

6.2.3 Envio da Mensagem HTTP

Quando um botão é premido, é enviada uma mensagem HTTP para o servidor. Esta acção é realizada a partir da função `post_request()`, que aceita como argumentos três *strings*, que correspondem à identificação do cartão (UID), endereço MAC do terminal e botão premido, respectivamente.

A função começa por verificar se o terminal ainda se encontra conectado ao servidor. Caso tal não aconteça, este irá reestabelecer a conexão, tal como acontece quando se liga o mesmo. Se o terminal e o servidor se encontrarem devidamente conectados, é então iniciado um cliente HTTP, que envia uma mensagem para o endereço `http://192.168.0.2:8000/quiz/response/` com a seguinte mensagem "{uid: 04 43 C6 32 34 31 80, mac: B4:E6:2D:3F:08:30, ans: D}", como descrito na secção 5.4.2. A execução da função termina apenas quando o terminal receber a mensagem de resposta vinda do servidor, o que faz piscar o LED verde do terminal, indicando que a mensagem foi enviada com sucesso.

6.2.4 Validação do Cartão

Durante o desenvolvimento da *firmware* dos terminais, foram testadas diferentes abordagens no que respeita à validação do cartão de identificação. Inicialmente, a validação era realizada no terminal. Ao inserir-se um cartão no mesmo, era feita uma verificação numa base de dados, alojada no servidor, de forma a verificar se o cartão de estudante se encontrava registado no sistema, só permitindo que fossem executadas acções caso tal se verificasse. Contudo, esta abordagem fazia aumentar o tempo de inutilização do terminal, uma vez que era necessário aguardar pela verificação do cartão.

A solução adoptada passa por considerar, do ponto de vista do terminal, que todos os cartões são válidos, sendo que a sua autenticação é realizada posteriormente pelo servidor no momento de processar as respostas recebidas. Esta abordagem permite tornar o terminal bastante mais responsivo, ao mesmo tempo que o torna mais ligeiro do ponto de vista de processamento.

6.2.5 Envio das Respostas

Tal como aconteceu com a validação dos cartões de identificação, também foram testadas diferentes abordagens quanto ao envio da mensagem para o servidor. Numa primeira abordagem, a resposta era escrita diretamente na base de dados do servidor. Apesar de, funcionalmente, não existir grande diferença entre esta e a solução adoptada,

tal abordagem não permitia garantir uma precisão do tempo de resposta menor que um segundo. Tal se deve ao facto de, ao introduzir informação diretamente na base de dados, o gestor utilizado, MySQL, não ser capaz de assegurar uma maior precisão na data e hora de registo. Por sua vez, o Django determina primeiro a data e hora antes de registar informações na base de dados, sendo assim capaz de alcançar uma precisão até aos milissegundos.

Outro motivo que levou a que a mensagem seja enviada, através de um pedido HTTP, para o servidor, e só posteriormente seja registada na base de dados, prende-se com facto de ser boa prática deixar o servidor lidar com todas as entradas na mesma. Desta forma, apenas uma entidade tem acesso à base de dados, garantindo assim uma maior fiabilidade da informação nela contida.

Parte III

Resultados e Discussão

Capítulo 7

Testes Realizados em Ambiente Realista

De forma a testar, tanto a escalabilidade do sistema, como o seu correcto funcionamento, foram realizados testes no Laboratório de Robótica e Automação do departamento de Engenharia Mecânica, da Universidade de Aveiro, de modo a simular um ambiente realista de sala de aula, onde uma matéria é exposta aos alunos através da projecção de *slides*, nos quais são inseridas algumas questões, que os alunos devem responder através dos terminais individuais disponibilizados. Para além disso, foi realizado um teste de usabilidade da aplicação *web*, para o qual foram convidados vários professores para testar a aplicação desenvolvida. Por fim, foram realizados testes em ambiente realista no âmbito da Academia de Verão, realizada na Universidade de Aveiro.

Neste capítulo são apresentados os moldes em que foram realizados os testes mencionados, seguindo-se uma reflexão sobre os resultados dos mesmos, obtidos tanto através de um inquérito respondido pelos participantes das sessões de teste, como das opiniões recolhidas do contacto directo com os mesmos.

7.1 Sessão de Testes em Ambiente Realista

De forma a simular um ambiente de sala de aula, a sessão de testes realizada contou com uma apresentação do sistema e das minúcias do seu funcionamento. No início da apresentação foi descrito aos participantes o funcionamento geral do sistema, seguindo-se uma explicação de como utilizar os terminais individuais fornecidos, para que pudessem responder às questões colocadas ao longo da apresentação.

A implementação do sistema CLASSQUIZ utilizada nesta sessão de testes contou com um servidor da aplicação, um *router* responsável por criar uma rede local e uma série de terminais individuais, um por cada participante. Tanto o servidor da aplicação como o cliente (navegador da Internet do docente), encontravam-se alojados na mesma máquina física. Contudo, a comunicação entre o cliente e o servidor deu-se por Wi-Fi, através do *router*, por via de pedidos HTTP. Desta forma, não existem diferenças significativas entre a abordagem adoptada e um cenário onde o servidor e cliente se encontrem em máquinas distintas, no que à fiabilidade e velocidade da comunicação diz respeito. A imagem da figura 7.1 ilustra a disposição dos componentes do sistema nos testes realizados.

No final da apresentação, e da realização de algumas questões, foi ainda feita uma demonstração da utilização da aplicação *web*, exemplificando tanto as acções que o uti-



Figura 7.1: Ilustração do ambiente de realização das sessões de teste do sistema.

lizador (professor) deve cumprir para a realização de inquéritos electrónicos, como das funções do administrador da aplicação, cuja principal função passa por associar as contas de utilizadores a Unidades Curriculares existentes.

À data da sessão realizada, apenas tinha sido possível imprimir, em 3D, uma caixa protectora para os terminais individuais, pelo que apenas um se encontrava na sua versão final. Os restantes cinco terminais construídos eram constituídos apenas pelo seu circuito eléctrico e fonte de alimentação, não tendo, portanto, uma estrutura que lhes servisse de apoio. Como tal, devido ao peso da PCB, e ao facto de estar se situar por cima do módulo leitor de cartões, existiram casos de mau contacto eléctrico, levando a que os terminais não fossem capazes de ler os cartões de identificação. Ainda assim, na maioria dos casos isto acabou por não se revelar um problema de força maior tendo, a maioria dos participantes, sido capaz de responder, de forma autenticada, às questões colocadas globalmente. Para além disso, em virtude da falta de terminais adicionais, foi também utilizado o terminal desenvolvido por Mamede. Este último havia sido actualizado com a nova versão de *firmware* desenvolvida, fazendo com que funcionasse da mesma maneira que os restantes.

A sessão de testes contou com a participação de cinco alunos, todos finalistas do curso de Engenharia Mecânica da Universidade, e dois professores, que participaram nos testes realizados da mesma forma que os alunos presentes, ou seja, respondendo às questões colocadas através dos terminais.

Ao todo, foram realizadas cinco questões, inseridas na apresentação através do seu URL, onde foram abordadas questões relacionadas com a informação apresentada. De forma a simular uma aula realista, todas as questões realizadas foram englobadas numa mesma lição (capítulo 5), sendo ainda todas elas autenticadas. Para a autenticação dos alunos foram fornecidos cartões RFID, previamente registados no sistema. As questões apresentadas tinham todas a duração de 30 segundos, que se verificou ser tempo suficiente para que os participantes pudessem reflectir um pouco sobre a opção de resposta a seleccionar. O *browser* utilizado durante a sessão de testes para a realização das questões foi o Google Chrome, versão 74.0.3729.169 para Linux.

No final da sessão, foi pedido aos participantes que respondessem a um inquérito, disponível para consulta no anexo C, de forma a se poder avaliar, tanto a facilidade de

utilização e fiabilidade do sistema, como possíveis formas de implementação do mesmo. Aos professores participantes na sessão, foi pedido que respondessem apenas às questões 1, 2 e 14 do inquérito, uma vez que as restantes procuram traduzir a forma como os alunos preferem ver o sistema a ser implementado no futuro.

Assim, de uma forma resumida, o ambiente em que foi realizado o primeiro teste consiste em:

- Um computador, que serve ao mesmo tempo de servidor e cliente;
- Um *router* presente na sala, responsável por criar a rede local, à qual se ligam todos os restantes componentes do sistema;
- Sete terminais individuais, um por participante;
- Sete cartões RFID devidamente registados no sistema, utilizados pelos participantes para se autenticarem.

7.1.1 Resultados da Sessão

De seguida são apresentados os resultados obtidos dos inquéritos, aos quais os participantes responderam, sendo ainda realizada uma reflexão sobre os mesmos, considerando também as opiniões recolhidas do contacto directo com os participantes da sessão de testes. Os resultados médios do inquérito encontram-se apresentados no gráfico da figura 7.2.

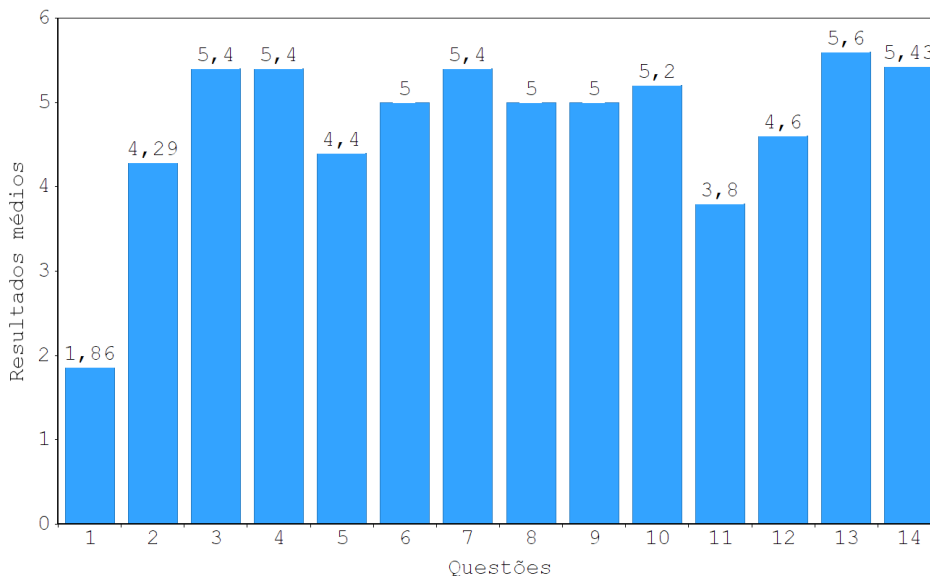


Figura 7.2: Resultados médios da sessão de testes em ambiente realista.

Dificuldade Sentida na Utilização dos Terminais Individuais

A maioria dos participantes não sentiu dificuldades na utilização do terminal individual. Houve, contudo, um participante que teve muitas dificuldades na sua utilização. Esta dificuldade está, provavelmente, associada ao facto de se tratar de um terminal sem caixa envolvente, pelo que a sua utilização pode ter ficado comprometida devido a algum mau contacto entre o módulo leitor de cartões e a PCB.

Avaliação do *Feedback* Proporcionado pelos LED's

Quanto ao *feedback* proporcionado pelos LED's, a avaliação média dos participantes foi de 4,29 valores em 6. As principais críticas recebidas em relação a este item do inquérito relacionam-se, por um lado, com o facto de só existirem dois LED's no terminal, o que leva a que uma mesma luz tenha vários significados, e por outro o facto de a confirmação de registo da resposta poder levar alguns segundos a acontecer. Em relação a este último não existe muito a fazer, uma vez que essa é uma das contrapartidas do modelo de comunicação adoptado.

Autenticação dos Participantes

Houve, por parte dos participantes, uma boa aceitação do cartão de estudante como forma de autenticação dos alunos. De facto, os únicos participantes que não concordam plenamente com este método de autenticação são os que preferem um sistema mais robusto, como a autenticação por leitura das impressões digitais. Contudo, as opiniões recolhidas durante a sessão de testes levam a concluir que não os mesmos não se opõem a um modelo de autenticação tradicional na falta de alternativa, tendo apenas preferência por um menos falível.

Utilização do Sistema para a Realização de Inquéritos Anónimos e Autenticados

Ao contrário do que seria expectável, existe uma maior aceitação dos participantes para a realização de inquéritos autenticados, em vez de inquéritos anónimos. Para além disso, os participantes tendem a preferir a utilização do sistema para fins de avaliação. O esperado seria precisamente o oposto, uma vez que inquéritos estatísticos, principalmente quando anónimos, são menos comprometedores para o inquirido.

Com base no *feedback* obtido presencialmente, não é possível concluir se de facto os participantes preferem participar em questionários que contem para a sua avaliação, ou se simplesmente vêm maior potencial do sistema quando utilizado dessa forma. Ainda assim, todos os métodos de utilização sugeridos no inquérito foram bem recebidos, com uma aprovação média de 4,4, 5 e 5,4 valores em 6 no que respeita à utilização do sistema para a realização de inquéritos anónimos, autenticados para fins estatísticos e autenticados para efeitos de avaliação, respectivamente.

Incentivo à Assistência e Participação nas Aulas

Todos os inquiridos mostraram-se confiantes que a utilização do sistema CLASSQUIZ pode ser importante para melhorar os índices da assistência às aulas, bem como para aumentar a atenção e participação dos alunos nas mesmas, tendo, ambos os itens, obtido

uma avaliação média de 5 valores em 6. Não obstante, um dos participantes da sessão não se mostrou confiante que a utilização do sistema pudesse aumentar significativamente o nível de participação dos alunos nas aulas, tendo atribuído apenas uma avaliação de 3 valores a esta questão. Ainda assim, todos os inquiridos acreditam que a utilização deste sistema, particularmente a realização de questionários durante as aulas, pode levar a um aumento do grau de aprovação às disciplinas onde o sistema seja implementado, tendo este item obtido uma aprovação média de 5,2 valores.

Utilização do Sistema como Instrumento de Avaliação

Como esperado, os inquiridos encontram-se mais receptivos à utilização do sistema como ferramenta de avaliação facultativa, em oposição a um modelo de avaliação obrigatório, tendo estas propostas obtido uma aprovação de 76,67% e 63,33% respectivamente. Contudo, estes valores são baixos quando comparados com os obtidos à questão da utilização do sistema para inquéritos de carácter avaliativo, que obteve uma aprovação média de 90%. Esta diferença leva a concluir que os participantes da sessão, apesar de preferirem ver o sistema a ser aplicado para avaliação de alunos, não se encontram tão certos que este deva ser implementado de todo. Ainda assim, o grau de aprovação à utilização do sistema como instrumento de avaliação indica que, pelos menos, não se opõem à sua implementação, em particular se for utilizado na realização de inquéritos que não constituam momentos de avaliação de carácter obrigatório.

Acesso à Aplicação pelos Alunos

Uma sugestão proposta no inquérito respondido pelos participantes da sessão de testes aborda o acesso à aplicação *web* por parte dos alunos, de forma a poderem visualizar os resultados por si obtidos. Tal funcionalidade implicaria a criação de contas de utilizador para os alunos, e permitiria apenas que estes tivessem acesso aos resultados, não tendo sido especificado, no entanto, se seriam apenas os seus, ou de todos os participantes dos questionários. Esta proposta obteve uma excelente aprovação por parte dos participantes, com uma avaliação média da proposta de 5,6 valores em 6. Esta avaliação, contudo, embora significativa, não permite concluir se os alunos consideram que tal funcionalidade seria imperativa para a implementação do sistema, ou se seria apenas uma funcionalidade bem vista pelos alunos.

Avaliação Geral do Sistema

O sistema obteve, no seu todo, uma avaliação média de 90,48% por parte dos participantes da sessão de testes, nos quais são incluídos os dois professores presentes. Este elevado grau de aprovação faz transparecer a confiança que os mesmos têm no sucesso da implementação do sistema, embora careça ainda de algumas melhorias, como o o *feedback* proporcionado pelo terminal. Para além disso, ficam em aberto algumas questões quanto à forma como o sistema poderá vir a ser implementado numa sala de aula, principalmente quanto ao carácter avaliativo dos questionários através de si realizados.

7.2 Teste de Usabilidade da Aplicação WEB

De forma a testar a usabilidade da aplicação *web* desenvolvida, foi pedido a um grupo de professores do Departamento de Engenharia Mecânica da Universidade de Aveiro que experimentassem a aplicação, seguindo a secção laranja do Manual do Utilizador (anexo D). O teste decorreu entre os dias 26 de Junho e 1 de Julho de forma a permitir uma maior flexibilidade de participação.

Com vista a facilitar a participação neste teste, a página da aplicação foi disponibilizada na Internet, através do endereço `http://lars.mec.ua.pt:8000`. Foi ainda criada, previamente, uma conta de utilizador para cada participante, sendo que todas se encontram associadas a uma única Unidade Curricular, designada LAR. Desta forma, simula-se um ambiente de uma disciplina leccionada por vários docentes.

O principal objectivo deste teste é o de verificar se os utilizadores sentem dificuldades na utilização da aplicação *web*, bem como recolher sugestões de melhoria e opiniões sobre que tipo de funcionalidades gostariam de ver implementadas no futuro.

7.2.1 Resultados do Teste de Usabilidade

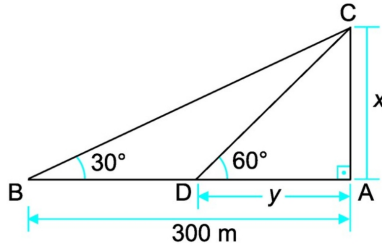
Os resultados do teste de usabilidade da aplicação *web* foram obtidos a partir do contacto directo com os participantes. No geral, não foram verificadas dificuldades na utilização da aplicação, contudo, foram várias as dúvidas e sugestões que surgiram sobre a implementação do sistema.

A principal questão colocada pelos participantes encontra-se relacionada com a possibilidade de se poder realizar várias questões em simultâneo, uma vez que num ambiente real decorrem várias aulas ao mesmo tempo, ainda que de disciplinas diferentes. Para além disso, uma mesma Unidade Curricular pode ter várias turmas com aulas a decorrer ao mesmo tempo, principalmente se se tratarem de aulas práticas. Assim, a necessidade de arranjar uma solução que permita uma centralização do sistema torna-se evidente para uma implementação bem sucedida do sistema CLASSQUIZ.

Uma funcionalidade sugerida durante o teste de usabilidade diz respeito à optimização da aplicação para dispositivos móveis, como *smartphones* e *tablets*, nomeadamente no que respeita ao modo *fullscreen*. Em resultado da reduzida dimensão de ecrã e do *aspect ratio* que estes dispositivos apresentam, e uma vez que no modo *fullscreen* é necessário garantir que toda a informação é apresentada no ecrã, não é possível garantir que uma mesma estrutura de apresentação satisfaça as necessidades de todos os dispositivos, como se pode ver na figura 7.3. Desta forma, será necessário, futuramente, reformular a forma como é apresentada a informação de uma dada questão em modo de ecrã inteiro, tendo em conta a utilização de dispositivos móveis por parte dos professores.

Outra funcionalidade sugerida diz respeito à ausência de um modo de edição na versão actual da aplicação *web*, tal como existe noutras aplicações de criação de inquéritos, como é o caso do Moodle. Este modo de edição deveria fazer aparecer, ou desaparecer, todas as opções de criação e edição de inquéritos, podendo ser alterado a qualquer instante pelo utilizador. Desta forma, no momento de realização de uma questão, não seria possível premir por engano o botão de edição da questão, o que permitiria aos alunos verem demasiada informação sobre a questão a realizar.

What is the value of y ?



a) $y = 300 \cdot \frac{\tan 60^\circ}{\tan 30^\circ}$
 b) $y = 300 \cdot \frac{\tan 30^\circ}{\tan 60^\circ}$
 c) $y = \frac{300 \cdot \tan 60^\circ}{\tan 30^\circ}$
 d) $y = \frac{300 \cdot \tan 30^\circ}{\tan 60^\circ}$
 e) None of the above

19

Figura 7.3: Apresentação do modo de ecrã inteiro num *smartphone*.

Também a página de selecção de lições já criadas foi alvo de algumas críticas, tendo sido considerado que a forma como estas são apresentadas ao utilizador (figura 7.4) não é ideal, uma vez que se encontram identificadas apenas pela sua Unidade Curricular e data e hora de criação. Assim, é aconselhada uma reestruturação desta página, de forma a que seja mais fácil a identificação e selecção de uma dada lição.

ClassQuiz New Question My Questions New Lesson My Lessons Results Profile Account Logout

June 26, 2019 - 00:33:59
LAR

June 26, 2019 - 00:33:51
Teste

June 26, 2019 - 00:27:33
LAR

June 26, 2019 - 00:24:38
LAR

June 26, 2019 - 00:20:17
LAR

June 14, 2019 - 15:59:50
LAR

Figura 7.4: Página de selecção de lições da aplicação *web*.

Quanto ao correcto funcionamento da aplicação, embora este teste não contemple a interação do servidor com os terminais individuais, não foram detectados quaisquer anomalias, à excepção do *display* de uma questão em dispositivos móveis, como já foi mencionado. Apesar das questões levantadas pelos docentes participantes do teste de usabilidade, estes mostraram-se receptivos a uma eventual implementação do sistema desenvolvido nesta dissertação.

7.3 Sessão de Testes em Realizada na Academia de Verão

A Academia de Verão destina-se a alunos do ensino secundário (10^o ao 12^o ano) e consiste num programa de actividades científicas, desportivas e de lazer, realizadas num campus universitário.

No contexto das actividades científicas, foram realizadas no Laboratório de Robótica e Automação da Universidade de Aveiro, nos dias 8 e 9 de Julho, actividades relacionadas com a manipulação robótica de objectos e com o projecto de condução autónoma (ATLASCAR), para além de uma sessão de testes do sistema CLASSQUIZ. Os participantes foram alunos do ensino secundário aos quais, depois de participarem nas actividades propostas, foram sujeitos a seis questões, com a duração de 30 segundos cada, sobre as actividades que haviam realizado (figura 7.5).

Devido ao elevado número de participantes em relação ao número de terminais disponíveis, os alunos, agrupados em 6 grupos de três elementos e com um terminal individual por grupo, responderam às questões projectadas. De resto, a configuração do sistema implementado nesta sessão de testes foi em tudo semelhante à da primeira sessão em ambiente realista, ou seja, foi utilizada a mesma máquina como servidor e cliente da aplicação *web*, sendo que a comunicação entre estes dois elementos se deu através de um *router* presente no laboratório, responsável por criar a rede local à qual se ligam todos os dispositivos. Contudo, ao contrário do que aconteceu na primeira sessão, as questões não foram inseridas numa apresentação. Em vez disso, as questões foram realizadas em sequência, tendo sido apenas explicado de uma forma muito resumida como é que os participantes deveriam utilizar os terminais para responder às questões que lhes iriam ser colocadas. No que aos terminais individuais diz respeito, ao contrário do que havia acontecido na primeira sessão de testes em ambiente realista, os terminais já se encontravam devidamente montados na caixa protectora, minimizando assim drasticamente os problemas de falha de comunicação entre componentes dos terminais.

Em geral, a sessão decorreu sem problemas, com a excepção de uma resposta submetida que não foi devidamente validada pelo servidor. Desde então, não foi possível replicar o problema, pelo que o mais provável é que tenha ocorrido alguma falha de comunicação entre o módulo leitor de cartões e o microcontrolador, impossibilitando assim a leitura do cartão de identificação e conseqüentemente o seu registo da resposta na base de dados, ou alguma falha de comunicação entre o terminal e o microcontrolador (neste caso, a causa mais provável é que o microcontrolador se tenha desconectado momentaneamente da rede, não tendo conseguido reestabelecer a ligação durante o decorrer da questão).

Ao nível da facilidade de utilização dos terminais, não existiu qualquer tipo de dificuldades sentidas por parte dos participantes, existindo até algum entusiasmo visível pela participação nos inquéritos.

Para concluir, embora tenha ocorrido um problema com um dos terminais individuais no decorrer de uma questão, considera-se que os resultados são bastante satisfatórios, tanto pela ausência de outros problemas, como pela aceitação e facilidade de utilização dos terminais demonstrados pelos participantes da Academia de Verão. Ainda assim, fica demonstrado que o sistema CLASSQUIZ carece da realização de mais testes, a fim de identificar possíveis falhas, tanto na montagem de componentes, como na implementação do sistema.



Figura 7.5: Sessão de testes realizada na Academia de Verão.

Capítulo 8

Considerações Finais

Neste capítulo são apresentadas as principais conclusões desta dissertação, bem como descritas as perspectivas de desenvolvimentos futuros que podem surgir na sua sequência.

8.1 Conclusões

Um dos principais objectivos desta dissertação consistia no desenvolvimento de uma aplicação *web*, capaz de comunicar com os terminais individuais desenvolvidos, e que permitisse a criação e realização de questionários electrónicos, para além de permitir a obtenção de resultados imediatos. Outro grande objectivo passava pela criação dos terminais individuais, capazes de autenticar, em contínuo, o aluno participante nos questionários postos globalmente.

Em relação ao desenvolvimento da aplicação *web*, a opção pela utilização da *framework* Django para o desenvolvimento do *back-end* mostrou-se acertada. Graças à sua vasta documentação *online*, e em virtude da estruturação rígida que esta impõe ao projecto, a implementação de novas funcionalidades tornou-se bastante simples, uma vez que uma mesma ideia é implementada de forma muito semelhante nos mais diversos projectos encontrados na Internet. Além disso, a forte comunidade existente em torno desta *framework*, em especial a comunidade *open-source*, provou ser uma mais valia inigualável tanto na aprendizagem da utilização da ferramenta, como na resolução de problemas.

Outra vantagem da utilização desta *framework* diz respeito ao facto de oferecer, logo à partida, um sistema de contas de utilizador bastante robusto e fiável, que contempla inclusive a encriptação de senhas, e uma página de administrador. Esta última, para além das vantagens que proporciona na utilização regular do sistema, serviu ainda como um dos principais métodos de *debugging* durante o desenvolvimento da aplicação, uma vez que a aplicação consiste, essencialmente, em modelos que podem ser registados na página do administrador.

No que ao desenvolvimento dos terminais diz respeito, foi necessário reformular todo o seu *firmware*, de forma a, por um lado, permitir comunicações com o servidor segundo o protocolo TCP, e por outro permite a verificação, em contínuo, da presença do cartão de identificação no mesmo. Para além disso, foi necessário garantir que este permita o envio de respostas com e sem cartão, de forma a ser possível a realização de inquéritos anónimos.

Uma questão que foi necessário abordar diz respeito ao sistema de alimentação, uma vez que tal não havia sido equacionado na versão anterior do terminal. A solução adotada permitiu solucionar rapidamente este problema, embora não seja a solução ideal, tanto ao nível do seu custo (as baterias utilizadas correspondem ao componente mais caro dos terminais) como ao nível das suas dimensões. Ainda assim, a solução provou ser ideal para a alimentação de terminais protótipos.

Quanto à acomodação do cartão de estudante e fixação dos botões e LED's dos terminais, verifica-se que a solução encontrada cumpre com os requisitos previamente estabelecidos. Principalmente no que respeita aos botões, o facto de o utilizador do terminal premir directamente os mesmos permite-lhe receber um *feedback* táctil que não existia na versão anterior do terminal, onde o aluno fazia actuar os botões por intermédio de uma extensão em plástico. O principal problema relativo à estrutura mecânica desenvolvida prende-se com o seu método de produção, uma vez que as estruturas desenvolvidas foram obtidas por impressão 3D. Este método de fabrico, apesar de excelente para prototipagem, torna-se muito demorada para o fabrico de uma série de terminais, ainda que pequena (uma caixa demora cerca de 5 horas a ser impressa, e requer mais algumas para acabamentos e montagens dos componentes).

Por fim, os resultados dos testes realizados evidenciam um elevado grau de aceitabilidade do sistema tanto por parte dos alunos, que o vêem inclusive como uma forma de poderem melhorar o seu aproveitamento escolar, como por parte dos professores. Esta receptividade é imperativa, uma vez que para o sistema CLASSQUIZ ter sucesso, não basta a aceitação por parte dos professores, é necessário que os alunos sintam que este traduz um modelo válido e justo de avaliação.

O sistema desenvolvido deu origem a um poster digital que foi submetido, com sucesso, à participação no Fórum de Ensino e Aprendizagem@UA, que se caracteriza como um espaço de debate de estratégias, iniciativas e inovação no ensino e na aprendizagem. Durante o decorrer do evento, o poster digital encontrou-se exposto ao público em local próprio, definido pela organização do evento.

8.2 Trabalhos Futuros

Apesar de todo o trabalho desenvolvido nesta dissertação, a solução alcançada não passa de um protótipo, carecendo ainda da realização de testes mais aprofundados. Para além disso, existem uma série de funcionalidades que podem vir a ser incorporadas no sistema futuramente.

Com vista à continuidade deste projecto, são apresentadas de seguida algumas sugestões de trabalho futuro.

Aplicação WEB

No que respeita à aplicação *web*, existem várias funcionalidades que podem ser implementadas. Uma destas funcionalidades diz respeito a um sistema de acesso à aplicação por parte dos alunos, de forma a estes terem acesso aos seus resultados e avaliações. A proposta de implementação desta funcionalidade obteve resultados muito positivos no inquérito respondido pelos participantes da primeira sessão de testes, conforme discutido na secção 7.1.

Uma questão relacionada com a aplicação *web* diz respeito à necessidade de se arranjar uma solução de centralizar o sistema. Neste momento, o sistema está desenvolvido de forma a que exista um servidor por sala, e os vários servidores que possam existir não conhecem nenhuma forma de comunicação entre si. Esta solução, embora viável para testes iniciais, não serve para uma solução a longo prazo, uma vez que uma mesma disciplina não é leccionada sempre na mesma sala de aula. Assim, uma possível solução poderá passar por existir apenas um servidor centralizado para todo o departamento ou instituição, sendo necessário, contudo, arranjar forma de se poder realizar vários questionários ao mesmo tempo. Esta será, por ventura, a principal dificuldade de tal implementação, uma vez que neste momento, se forem realizadas mais do que uma questão em simultâneo, o servidor não tem forma de saber a que questão corresponde cada resposta recebida.

Por fim, seria interessante arranjar um alojamento na Internet para a aplicação, permitindo tanto aos professores, como possivelmente aos alunos, acederem à mesma a partir de qualquer lugar, sem terem de se encontrar no mesmo espaço físico do servidor. A implementação desta solução, contudo, deverá salvaguardar que para a realização de questões e questionários, continua a existir uma limitação do espaço físico, onde só os terminais podem aceder ao sistema.

Terminal Individual

Em relação aos terminais individuais, propõe-se que, futuramente, seja reformulado o sistema de alimentação, de forma a se encontrar uma alternativa mais barata e compacta à solução adoptada. Na mesma linha, de forma a reduzir as dimensões dos terminais, propõe-se que, ao invés da utilização do microcontrolador NodeMCU, seja desenvolvida uma placa com o módulo ESP8266, ou outro equivalente. Esta alternativa, embora mais trabalhosa, poderá permitir reduzir as dimensões dos terminais individuais, especialmente se adoptada em conjunto com a sugestão anterior.

Relativamente à autenticação dos alunos, propõe-se o estudo de alternativas à autenticação através do cartão de estudante. Como se pode concluir dos resultados da primeira sessão de testes, parece existir indiferença dos alunos sob a forma como é realizada a sua autenticação. Assim, embora o modelo adoptado seja perfeitamente funcional, uma vez que se trata de um modelo baseado na *posse*, está sujeito ao esquecimento ou roubo das credenciais de acesso ao sistema. Desta forma, um modelo de autenticação biométrico, como a leitura de impressões digitais, assume-se como sendo uma solução atractiva, aumentando o nível de fiabilidade do sistema de autenticação e identificação de utilizadores do sistema.

Para além disso, de forma a aumentar a fiabilidade do sistema, sugere-se que os terminais individuais fiquem associados apenas a uma sala, ou, em alternativa, que sejam validados no início de cada aula. Com esta abordagem, uma vez que o endereço MAC dos terminais é utilizado para validar as respostas submetidas, evita-se que alguém possa utilizar outros dispositivos para enviar mensagens para o servidor com um endereço falso, uma vez que a lista de endereços válidos seria diferente para cada aula.

Por fim, para questionários onde a precisão do tempo de resposta assuma uma importância maior, propõe-se a implementação de um sistema em que os relógios internos dos terminais sejam sincronizados no início de cada aula com o relógio do servidor, ou de forma periódica durante o decorrer da aula, de forma a garantir uma maior precisão no tempo de resposta. O *timestamp* da resposta seria enviado na mensagem em conjunto

com a resposta seleccionada, o endereço MAC do terminal individual e a UID do cartão de identificação, garantindo assim que os possíveis atrasos que ocorram na recepção da mensagem por parte do servidor não prejudiquem o aluno.

Referências

- [1] CALÇADO, Samuel Fernando Jorge - *Automatização e controlo de reservas na indústria hoteleira*. Universidade de Aveiro, 2017. Dissertação de Mestrado.
- [2] *Access Control in Support of Information Systems - STIG*. Version 2, Release 1. Defense Information Systems Agency (DISA) for the Department of Defense (DoD), 2007.
- [3] MOREIRA, Pedro Alexandre Teixeira - *Gestão de controlo de acessos*. Faculdade de Engenharia da Universidade do Porto, 2008. Dissertação de Mestrado.
- [4] CARDOSO, David Rafael Pimentel Cardoso - *Controlo de acessos no sector turístico*. Universidade de Aveiro, 2015. Dissertação de Mestrado.
- [5] AMIN, Reham *et al.* - Biometric and Traditional Mobile Authentication Techniques: Overview and Open Issues. In *Intelligent Systems Reference Library*, 70, 2014.
- [6] WOORDS, Naomi e SIPONEN, Mikko - Improving password memorability, while not inconveniencing the user. In *International Journal of Human-Computer Studies*, 128, 2019.
- [7] PINHEIRO, Nuno Miguel Grilo Fonseca - *Secure Password Management Using Smartcards*. Instituto Superior Técnico de Lisboa, 2013. Tese de Mestrado.
- [8] MHAYIDIN, Mohd Firdaus Bin - *Student Attendance Using RFID System*. University Malaysia Pahang, 2008. Tese de Licenciatura.
- [9] KHAST, Negar - *Overview of Radio Frequency Identification - Security Issues and Suggesting a Solution*. Helsinki Metropolia University of Applied Sciences, 2017. Tese de Licenciatura.
- [10] KAPOOR, Gaurav e PIRAMUTHU, Selwyn - Vulnerabilities in Chen and Deng's RFID mutual authentication and privacy protocol. In *Engineering Applications of Artificial intelligence*, 24 (7) 2011.
- [11] HOFMAYR, Stefan - *Analysis and comparison of the potential of RFID-technology in European and U.S. retail supply chains*. Vienna University of Economics and Business Administration. Tese de Mestrado.
- [12] BELHADJ, Foudil - *Biometric system for identification and authentication*. Ecole nationale Supérieure en Informatique Alger, 2017. Tese de Doutoramento.

- [13] BHATTACHARYYA, Debnath *et al.* - Biometric Authentication: A Review. In *International Journal of u- and e- Service, Science and Technology*, 2 (4), 2009.
- [14] CHIKKERUR, Sharat - *Online Fingerprint Verification System*. State University of New York, 2005. Tese de Mestrado.
- [15] MASEK, Libor - *Recognition of Human Iris Patterns for Biometric Identification*. University of Western Australia, 2003. Tese de Licenciatura.
- [16] CUNHA, Marco Soudo - *Domotic Module for the Internet-of-Things*. Instituto Superior Técnico de Lisboa, 2017. Tese de Mestrado.
- [17] McEWEN, Adrien e CASSIMALLY, Hakin - *Designing the Internet of Things*. John Wiley & Sons, 2013.
- [18] ABADE, Pedro António Carvalho - *VR_Banway: deploying a body area network gateway on single-board computers and mesh networks*. Universidade de Aveiro, 2017. Tese de Mestrado.
- [19] MARQUES, Rafael Fernandes - *Sistema de monitorização de veículos*. Universidade de Aveiro, 2016. Tese de Mestrado.
- [20] PUKHANOV, Alexander - *WiFi Extension for Drought Early-Warning Detection System Components*. Linköping University, 2015. Tese de Mestrado.
- [21] NARAYAN, Shaneel - *Improving Network Performance: An Evaluation of TCP/UDP on Networks*. UNITEC Institute of Technology, 2014. Tese de Doutoramento.
- [22] KARAS, John e PESCHKE, Roland - *TCP/IP Tutorial and Technical Overview*. IBM Redbook, 2006.
- [23] ELLIS, Martin - *Understanding the Performance of Internet Video over Residential Networks*. University of Glasgow, 2012. Tese de Doutoramento.
- [24] SARI, Arif e KARAY, Mehmet - Comparative Analysis of Wireless Security Protocols: WEP vs WPA. In *International Journal of Communications, Network and System Sciences*, 8, 2015.
- [25] MILLER, Stewart - *WiFi Security*. McGraw-Hill Networking Professional, 2003. ISBN 0-07-142917-4
- [26] OSTERHAGE, Wolfgang - *Wireless Network Security*. CRC Press, 2018. ISBN 9781138093799
- [27] LASHKARI, Habibi *et al.* - *A survey in wireless security protocols (WEP, WPA and WPA2/802.11i)*. ICCSIT, 2009
- [28] DROMS, Ralph e LEMON, Ted - *The DHCP Handbook*. Sams, 2002.
- [29] WONG, Stanley - *The evolution of wireless security in 802.11 networks: WEP, WPA and 802.11 standards*. SANS Institute, 2003

- [30] RAMOS, Bruno Manuel de Moura - *Sistema de Recolha e Armazenamento Remoto de Informação Sensorial de um Processo Industrial usando Bases de Dados Múltiplas*. Universidade de Aveiro, 2018. Tese de Mestrado.
- [31] HARRINGTON, Jan L. - *Relational database design and implementation : clearly explained*. Morgan Kaufmann Publishers, 2009. ISBN 978-0-12-374730-3
- [32] RIBEIRO, Aurelio - *Princípios de Sistema de Base de Dados*. Universidade Virtual Africana.
- [33] SING, Pramod *et al.* - Studies and Analysis of Popular Database Models. In *International Journal of Computer Science and Mobile Computing*, 4, 2015.
- [34] JASNÝ, Vojtěch - *Trends in web application development*. University of Economics, Prague, 2007. Tese de Licenciatura.
- [35] VU, Dao - *Development of a front-end application using AngularJS: 1UP Media company case*. Laurea University of Applied Sciences, 2016. Tese de Licenciatura.
- [36] REIS, João Miguel Próspero Marques - *Desenvolvimento de Aplicação WEB para Visualização e Análise Genética Microbiana*. Universidade de Lisboa, 2012. Tese de Mestrado.
- [37] MOORE, Jonathan Ian - *Building a reusable application with Django*. Laurea University of Applied Sciences, 2016. Tese de Licenciatura.
- [38] BÄCK, Fredrik - *Back-end development of mobile application for the collection of dietary data*. Umeå University, 2012. Tese de Mestrado.
- [39] SHARMA, Gaurav - *Web Application Development for San Diego Cricket Club*. San Diego State University, 2012. Tese de Mestrado.
- [40] CHUNG, Chih-hung e PASQUINI, Laura - Web-based Learning Management System Considerations for Higher Education. In *Learning and Performance Quarterly*, 1 (4), 2013.
- [41] ALBARRAK, Ahmed *et al.* - Evaluating learning management systems for University medical education. In *Education and Management Technology (ICEMT)*, 2010.
- [42] WANG, Tzu-Hua - Web-based dynamic assessment: Taking assessment as teaching and learning strategy for improving students' e-Learning effectiveness. In *Computers & Education*, 54, 2010.
- [43] SEALE, Jane e COOPER, Martyn - E-learning and accessibility: An exploration of the potential role of generic pedagogical tools. In *Computers Education*, 54, 2010.
- [44] FRITZ, John - Classroom wall that talk: Using online course activity data of successful students to raise self-awareness of underperforming peers. In *The Internet and Higher Education*, 14 (2), 2011.
- [45] CHEN, Ling-Hsiu - Web-based learning programs: Use by learners with various cognitive styles. In *Computers Education*, 54, 2010.

- [46] *QuizXpress 6 - User Manual*. Game Show Crew, 2018.
- [47] QuizXpress - URL: <https://www.quizxpress.com/en/>. Visitado em Maio de 2019.
- [48] TEIXEIRA, António - *CLASSQUIZ*. Universidade de Aveiro, 2017.
- [49] MAMEDE, Manuel - *ClassQuiz Plus - Sistema de Inquérito e Avaliação Rápida em Sala de Aula*. Universidade de Aveiro, 2018.
- [50] XIAOJIE, Yang - *Analysis of DBMS: MySQL Vs PostgreSQL*. Kemi-Tornio University of Applied Sciences, 2011. Tese de Licenciatura.
- [51] LEE, Hsien-Yu e WANG, Nai-Jian - Cloud-based enterprise resource planning with elastic model-view-controller architecture for Internet realization. In *Computer Standards Interfaces*, 64, 2019.
- [52] THEBE, Laxmi - *Community Parenting Platform - Development and Dployment using the Django Framework*. Turku University of Applied Sciences, 2016. Tese de Licenciatura.
- [53] BENNETT, James - *Practical Django Projects*. Apress, 2009. ISBN 978-1-4302-1939-2
- [54] CURRAN, Kevin *et al.* - Javascript, XML, E4X and AJAX. In *Understanding the Internet*. Chandos Publishing, 2009. ISBN 978-1-84334-499-5
- [55] TURC, Traian - AJAX Technology for Internet of Things. In *Procedia Manufacturing*, 32, 2019.
- [56] BURNS, Jesse - *Cross Site Request Forgery - And introduction to a common web application weakness*. Information Security Partnes LLC, 2007.
- [57] Texas Instruments - URL: <http://www.ti.com/product/CD4532B/>. Visitado em Junho de 2019.

Anexos

Anexo A

Diagrama do Circuito Eléctrico

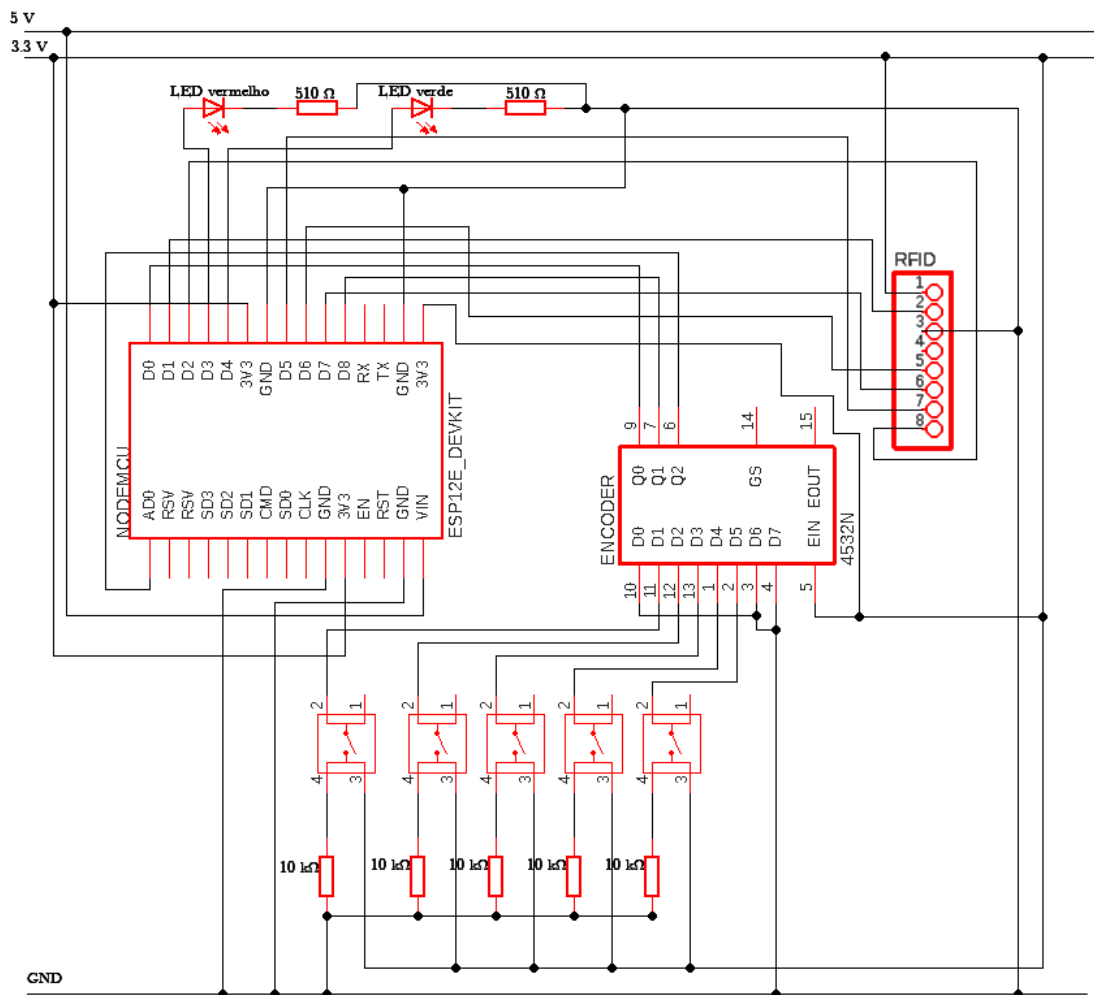


Figura A.1: Diagrama do circuito eléctrico do terminal individual.

Anexo B

Desenho Técnico da Estrutura Mecânica dos Terminais

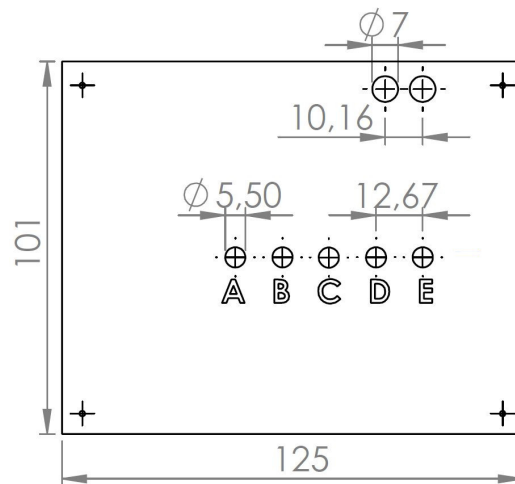


Figura B.1: Desenho técnico da tampa (dimensões em mm) da caixa do terminal individual.

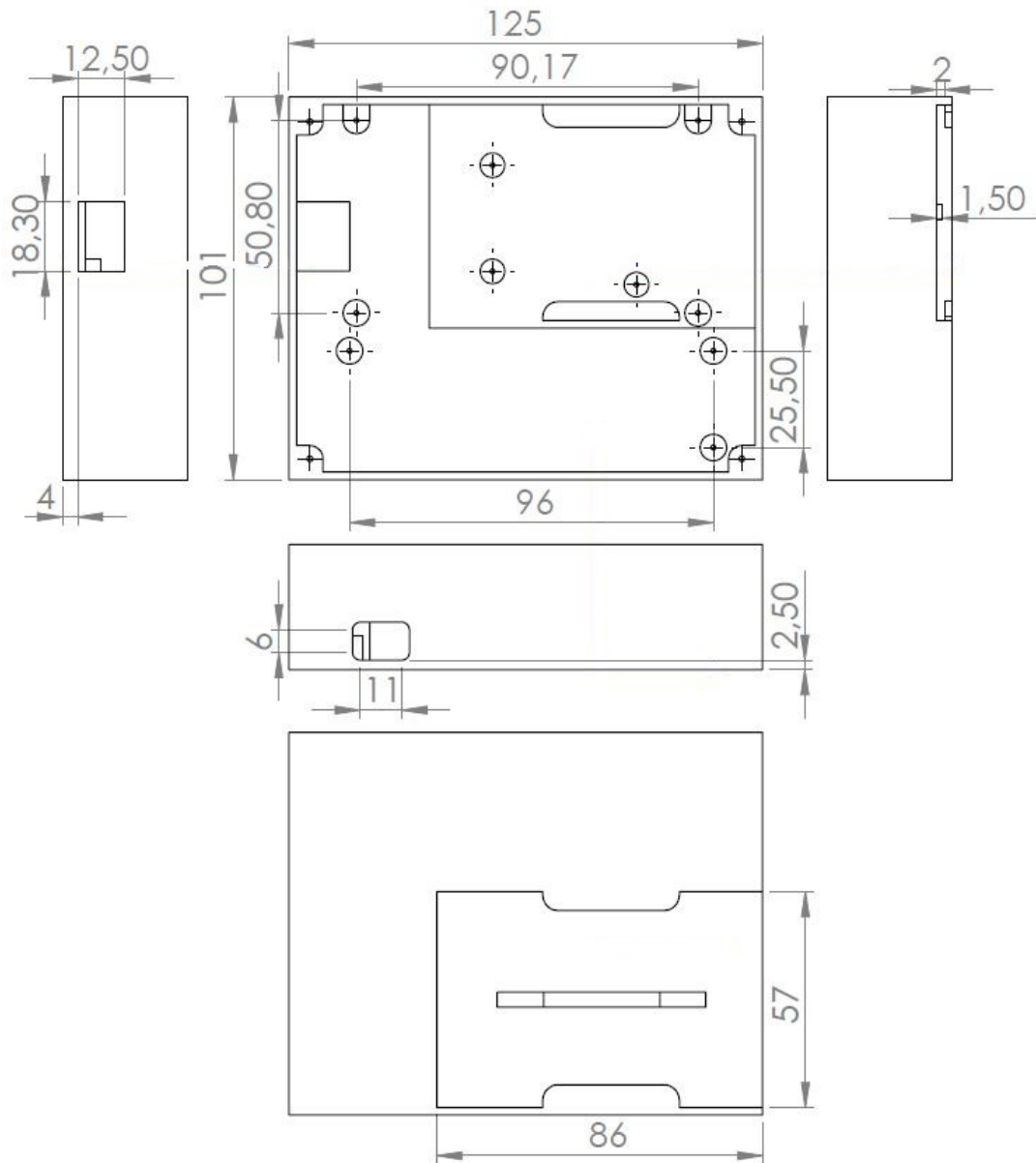


Figura B.2: Desenho técnico da caixa (dimensões em mm) do terminal individual.

Anexo C

Inquérito Apresentado na Sessão de Testes em Ambiente Realista

Sistema CLASSQUIZ

29/05/2019

Numa escala de 1 a 6, avalie:

1) O nível de dificuldade sentido na utilização do terminal individual:

1 2 3 4 5 6

2) O *feedback* proporcionado pelo terminal individual através dos LED's:

1 2 3 4 5 6

3) O nível de conforto que sente em a autenticação do aluno ser feita através do cartão de estudante:

1 2 3 4 5 6

4) O nível de conforto que sentiria se a autenticação fosse feita através da leitura de impressões digitais:

1 2 3 4 5 6

5) O seu nível de receptividade para que o sistema CLASSQUIZ seja utilizado na realização de inquéritos anónimos:

1 2 3 4 5 6

6) O seu nível de receptividade para que o sistema CLASSQUIZ seja utilizado na realização de inquéritos autenticados, para fins estatísticos:

1 2 3 4 5 6

7) O seu nível de receptividade para que o sistema CLASSQUIZ seja utilizado na realização de inquéritos autenticados, para fins de avaliação:

1 2 3 4 5 6

8) A importância da realização de inquéritos durante as aulas, para fins estatísticos ou de avaliação, enquanto incentivo para assistência às mesmas:

1 2 3 4 5 6

9) A importância da realização de inquéritos durante as aulas, enquanto ferramenta para ajudar a aumentar os níveis de atenção e participação dos alunos nas mesmas:

1 2 3 4 5 6

10) A importância da realização de inquéritos durante as aulas, enquanto ferramenta para ajudar a melhorar o grau de aprovação às disciplinas:

1 2 3 4 5 6

11) O seu nível de receptividade para que os inquéritos electrónicos constituam uma parte obrigatória da avaliação:

1 2 3 4 5 6

12) O seu nível de receptividade para que os inquéritos electrónicos constituam um método de avaliação facultativo:

1 2 3 4 5 6

13) A importância do acesso à aplicação por parte dos alunos, de forma a terem acesso aos resultados dos inquéritos realizados:

1 2 3 4 5 6

14) De uma forma global, como avalia o sistema CLASSQUIZ?

1 2 3 4 5 6

Anexo D

Manual de Utilização do Sistema CLASSQUIZ

CLASSQUIZ

Manual do Utilizador



Índice

Comandos

- 2 Iniciar o servidor
- 5 Criar *superuser*

Administração

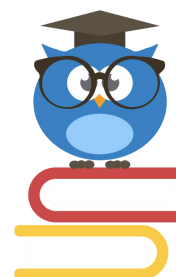
- 7 Página de Administração
- 8 Terminais Individuais
- 11 Alunos
- 14 Unidades Curriculares
- 16 Unidades Curriculares e Utilizadores

Aplicação WEB

- 18 Ligação à rede local
- 19 Página de *Login*
- 20 Home Page
- 21 Criar uma nova questão
- 23 Visualizar uma questão
- 24 Criar uma nova lição
- 25 Realizar uma questão
- 29 Resultados de uma questão
- 32 Resultados de uma lição

Terminal Individual

- 35 Terminal individual
- 37 Submissão de respostas
- 39 Inserção do cartão de estudante
- 40 Recarregar a bateria



CLASSQUIZ

Comandos



Iniciar o Servidor

2

Abra um terminal novo.

Shortcut: **ctrl + alt + T**

A screenshot of a Linux terminal window. The title bar at the top reads 'joao@joao-linux: ~'. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main area of the terminal shows the prompt 'joao@joao-linux:~ \$' in green text on a dark background.

Iniciar o Servidor

Digite o comando `dq`.

Este comando activa o *virtual environment* Python utilizado para correr o servidor, e define o caminho onde se encontram os ficheiros do servidor.

O caminho pode ser diferente, dependendo do computador utilizado.

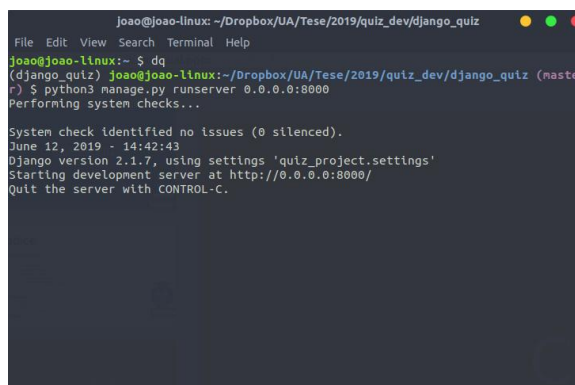


```
Joao@Joao-Linux: ~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz
File Edit View Search Terminal Help
Joao@Joao-Linux:~$ dq
(django_quiz) Joao@Joao-Linux:~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz (mas...
r *) $
```

Iniciar o Servidor

Digite o comando `python3 manage.py runserver 0.0.0.0:8000`

Este comando inicia o servidor da aplicação *web*, que fica à escuta na porta 8000.



```
Joao@Joao-Linux: ~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz
File Edit View Search Terminal Help
Joao@Joao-Linux:~$ dq
(django_quiz) Joao@Joao-Linux:~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz (mas...
r) $ python3 manage.py runserver 0.0.0.0:8000
Performing system checks...

System check identified no issues (0 silenced).
June 12, 2019 - 14:42:43
Django version 2.1.7, using settings 'quiz_project.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Criar *superuser*

Digite o comando `python3 manage.py createsuperuser`

Este comando cria um novo super-utilizador, com permissões de administrador.

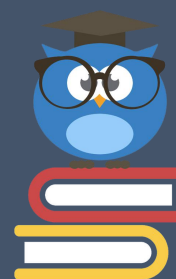
É necessário existir pelo menos um *superuser*, uma vez que são os únicos que podem aceder à página de administração.

Esta conta pode também ser utilizada como conta de utilizador.

```
Joao@Joao-linux: ~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz
File Edit View Search Terminal Help
(django_quiz) Joao@Joao-linux:~/Dropbox/UA/Tese/2019/quiz_dev/django_quiz (master)
r) $ python3 manage.py createsuperuser
Username (leave blank to use 'joao'):
```

CLASSQUIZ

Administração



Página de Administração

Através de um *browser*, acesse ao URL **192.168.0.2:8000/admin**

Apenas *superusers* podem acessar à página de administração.

Nesta página é possível:

- Registrar terminais individuais (**Terminals**)
- Registrar alunos (**Students**)
- Criar Unidades Curriculares (**Courses**)
- Associar utilizados (professores) a Unidades Curriculares (**Profile courses**)

Django administration
WELCOME, JOAO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ add	change
Users	+ add	change

GRRZ

Students	+ add	change
Terminals	+ add	change

USERS

Courses	+ add	change
Profile courses	+ add	change

Recent actions

My actions

- [Question: 56, Student: 00000](#) [undo](#)
- [Course: LAR meeting, User: Lar Meeting, Profile course](#)
- [Course: LAR meeting, User: Lar Meeting, Profile course](#)
- [Course: LAR meeting, User: Jobo Moreira, Profile course](#)
- [LAR meeting](#) [undo](#)
- [LAR meeting](#) [undo](#)
- [Course: LAR meeting, User: Test User, Profile course](#)
- [LAR meeting](#) [undo](#)
- [Course: LAR meeting, User: Test User, Profile course](#)
- [LAR meeting](#) [undo](#)
- [Course: Dissertação, Lesson: 2019-09-19 20:02:22.640891+00:00, Student: 00006](#) [undo](#)
- [Course: Dissertação, Lesson: 2019-09-19 20:02:22.640891+00:00, Student: 00006](#) [undo](#)

Terminais Individuais

Na página de administração, clique em **Terminals**.

Nesta página é possível editar a lista de terminais individuais registrados no sistema.

Para adicionar um terminal, clique em **ADD TERMINAL** + (1), para registrar vários terminais ao mesmo tempo, clique em **IMPORT** (2).

Django administration
WELCOME, JOAO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Quiz > Terminals

Select terminal to change

[IMPORT](#) [ADD TERMINAL](#)

Action: 0 of 7 selected

<input type="checkbox"/>	TERMINAL
<input type="checkbox"/>	B4.E6:2D:3F:08:30
<input type="checkbox"/>	B4.E6:2D:4A:99:83
<input type="checkbox"/>	B4.E6:2D:3F:08:93
<input type="checkbox"/>	B4.E6:2D:3F:08:24
<input type="checkbox"/>	B4.E6:2D:3F:04:AB
<input type="checkbox"/>	BC.DD:C2:9E:14:6D
<input type="checkbox"/>	2C:3A:E8:14:9B:C9

7 terminals

Terminais Individuais

- 1) Introduza o endereço MAC do terminal na caixa de texto
- 2) Clique em **SAVE** para guardar

Django administration
WELCOME, JOAO / VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Quiz > Terminais > Add terminal

Add terminal

Mac: 1

Save and add another Save and continue editing **2** SAVE

Terminais Individuais

- 1) Clique em **Choose File** para selecionar o ficheiro que contém os endereços MAC dos terminais
- 2) Seleccione o formato do ficheiro (**Format**)
- 3) Clique em **SUBMIT** para finalizar

Django administration
WELCOME, JOAO / VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Quiz > Terminais > Import

Import

This importer will import the following fields: id, mac

File to Import: 1 no file chosen

Format: 2

3

Alunos

Na página de administração, clique em **Students**.

Nesta página é possível editar a lista de alunos registados no sistema.

Para adicionar um aluno, clique em **ADD STUDENT** + (1), para registar vários alunos ao mesmo tempo, clique em **IMPORT** (2).

Alunos

- 1) Introduza a UID do cartão de estudante do aluno
- 2) Introduza o número mecanográfico do aluno
- 3) Clique em **SAVE** para guardar

Alunos

- 1) Clique em **Choose File** para selecionar o ficheiro que contém os números mecanográficos dos alunos, bem como a UID dos seus cartões de estudante.
- 2) Selecione o formato do ficheiro (**Format**)
- 3) Clique em **SUBMIT** para finalizar

Unidades Curriculares

Na página de administração, clique em **Courses**

Nesta página é possível editar as Unidades Curriculares registadas no sistema.

Para adicionar uma Unidade Curricular, clique em **ADD COURSE + (1)**.

Unidades Curriculares

- 1) Introduza o nome da Unidade Curricular
- 2) Clique em **SAVE** para guardar

2

Unidades Curriculares e Utilizadores

Na página de administração, clique em **Profile courses**

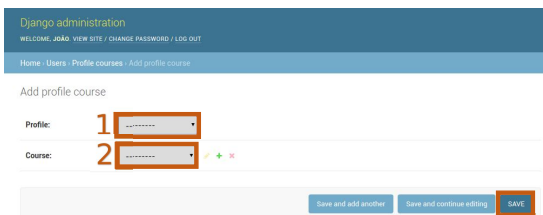
Nesta página é possível editar as relações entre Unidades Curriculares e utilizadores.

Para associar um utilizador a uma Unidade Curricular, clique em **ADD PROFILE COURSE** + (1).

Unidades Curriculares e Utilizadores

17

- 1) Seleccione o utilizador
- 2) Seleccione a Unidade Curricular
- 3) Clique em **SAVE** para guardar



The screenshot shows the Django administration interface for adding a profile course. The page title is "Django administration" with links for "WELCOME, JOAO", "VIEW SITE", "CHANGE PASSWORD", and "LOG OUT". The breadcrumb trail is "Home > Users > Profile courses > Add profile course". The form is titled "Add profile course" and contains two dropdown menus: "Profile:" with the value "1" and "Course:" with the value "2". There are also "+", "x", and "SAVE" buttons. The "SAVE" button is highlighted with a red box.

3

CLASSQUIZ

Aplicação WEB



Ligação à rede local

Conecte-se à rede local:

SSID: CLASSQUIZ

PASSWORD: password

Uma vez conectado, aceda ao endereço 192.168.0.2:8000, a partir de um *browser*.

Página de *Login*

Para aceder à aplicação, é necessário fazer *Login* com a sua conta de utilizador. De forma a simplificar o processo, pode utilizar a conta de administração **admin**.

NOTA: se optar por criar uma nova conta de utilizador, não estará associado a nenhuma Unidade Curricular!

1) Introduza as credenciais da sua conta de utilizador:

Username: admin

Password: admin

2) Clique em **Login**.

ClassQuiz Register Login

Log In

Username

Password

1

2 Login Forgot Password?

Need An Account? Sign Up Now

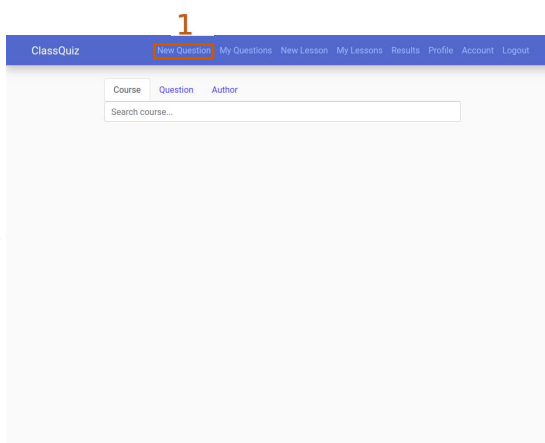
Home Page

Nesta página aparecem todas as questões das Unidades Curriculares a que o utilizador se encontra associado.

A conta de utilizador **admin** só se encontra associada à Unidade Curricular **LAR**, pelo que só terá acesso às questões associadas à mesma. Da mesma forma, apenas poderá criar questões e lições da Unidade Curricular **LAR**.

Através da caixa de texto, é possível filtrar as questões apresentadas por Unidade Curricular, título da questão e autor.

Para criar uma nova questão, clique em **New Question (1)**.



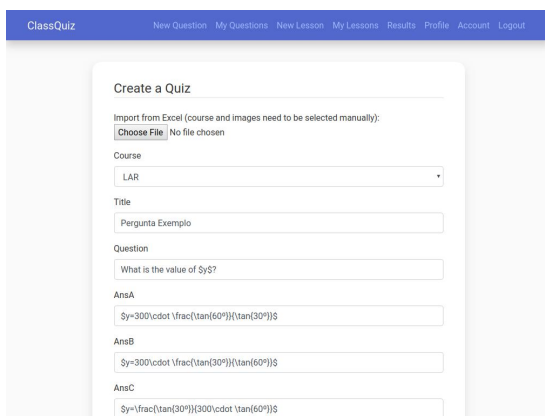
Criar uma nova questão

É possível importar uma questão a partir de um ficheiro Excel (**upload_template.xlsx**), formatado de acordo com o *template* disponibilizado.

Nos campo de texto, é possível incorporar fórmulas matemáticas, escritas em formato LATEX, como mostra a imagem.

Para criar uma questão, preencha os campos seguintes:

- **Course:** Unidade Curricular da questão
- **Title:** Título da questão, serve apenas como referência ao utilizador
- **Question:** Pergunta
- **AnsA - AnsE:** Opções de resposta



Criar uma nova questão

- **Right ans:** Resposta certa
- **Duration:** Duração da questão, em segundos
- **Image:** Imagem da questão (opcional)
- **Anonymous:** Indica se a questão é anónima ou autenticada

Por fim, clique em **Submit (1)** para guardar a questão criada.

ClassQuiz New Question My Questions New Lesson My Lessons Results Profile Account Logout

AnsC

$$y = \frac{\tan(30^\circ)}{\tan(60^\circ)} \cdot 300$$

AnsD

$$y = \frac{\tan(60^\circ)}{\tan(30^\circ)} \cdot 300$$

AnsE
 None of the above

Right ans
 b)

Duration
 4

Image
 Currently: quiz_img/Trigonometria.jpg
 Change: [Choose File](#) No file chosen

Anonymous
 No

1 **Submit** Cancel

Visualizar uma questão

Nesta página é possível pré-visualizar a questão (**Show (1)**).

Para além disso, é possível realizar a questão, através do botão **Start (2)**. Contudo, este botão pode estar inactivo, significando que o utilizador não tem nenhuma lição activa da mesma Unidade Curricular da questão.

Assim, é necessário criar uma nova lição. Para tal, clique em **New Lesson (3)**.

ClassQuiz New Question My Questions **New Lesson** My Lessons Results Profile Account Logout

João Moreira March 29, 2019 [Delete](#) [Edit](#) [Show](#) [Start](#)

Pergunta Exemplo

What is the value of y ?

a) $y = 300 \cdot \frac{\tan 60^\circ}{\tan 30^\circ}$
 b) $y = 300 \cdot \frac{\tan 30^\circ}{\tan 60^\circ}$
 c) $y = \frac{300 \cdot \tan 30^\circ}{\tan 60^\circ}$
 ...

10

3

1 2

Criar uma nova lição

A lição irá agrupar todas as questões realizadas.

Para criar uma nova lição:

- Selecione a Unidade Curricular (1);
- Selecione a resposta válida submetida pelos alunos (2);
- Clique em **Submit** (3) para concluir a criação da lição.

Realizar uma questão

A questão criada aparece agora na página principal da aplicação.

Para poder realizar a questão, clique no título da mesma (1).

Realizar uma questão

O botão **Start** (1) aparece agora activo. Clique nele para iniciar a questão em modo *fullscreen*.

ClassQuiz New Question My Questions New Lesson My Lessons Results Profile Account Logout

João Moreira March 29, 2019 Delete Edit Show **Start** 1

Pergunta Exemplo

What is the value of y ?

Diagram description: A right-angled triangle ABC with the right angle at A. Point D lies on the base BA. The length of BA is 300 m. The angle at B is 30° and the angle at D is 60° . The height CA is x and the distance DA is y .

Options:

- a) $y = 300 \cdot \tan 60^\circ$
- b) $y = 300 \cdot \tan 30^\circ$
- c) $y = \frac{300 \cdot \tan 60^\circ}{\tan 30^\circ}$
- d) $y = \frac{300 \cdot \tan 30^\circ}{\tan 60^\circ}$

10

Realizar uma questão

Ao clicar em **Start** na janela anterior, passa para o modo *fullscreen*, onde aparece apenas um botão, **Play**.

Ao premir este botão, a questão irá aparecer no ecrã, e o tempo de realização irá começar imediatamente a contar!

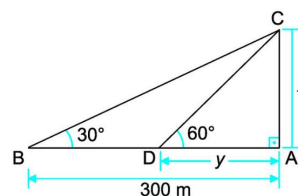


Realizar uma questão

Após o tempo chegar a zero (0), clique na tecla **Esc**, ou **F11**, para sair do modo *fullscreen*.

NOTA: as ações para sair do modo fullscreen podem variar consoante o *browser* utilizado.

What is the value of y ?



- a) $y = 300 \cdot \frac{\tan 60^\circ}{\tan 30^\circ}$
- b) $y = 300 \cdot \frac{\tan 30^\circ}{\tan 60^\circ}$
- c) $y = \frac{300 \cdot \tan 60^\circ}{\tan 30^\circ}$
- d) $y = \frac{300 \cdot \tan 30^\circ}{\tan 60^\circ}$
- e) None of the above

8

Resultados de uma questão

Para visualizar os resultados de uma dada questão, clique em **Results (1)** a partir de qualquer janela.

ClassQuiz New Question My Questions New Lesson My Lessons **Results** Profile Account Logout

João Moreira March 29, 2019 Delete Edit Show Start

Pergunta Exemplo

What is the value of y ?

What is the value of y ?

a) $y = 300 \cdot \frac{\tan 60^\circ}{\tan 30^\circ}$

b) $y = 300 \cdot \frac{\tan 30^\circ}{\tan 60^\circ}$

c) $y = \frac{300 \cdot \tan 60^\circ}{\tan 30^\circ}$

d) $y = \frac{300 \cdot \tan 30^\circ}{\tan 60^\circ}$

e) None of the above

10

Resultados de uma questão

Nesta janela são apresentados os resultados de todas as questões realizadas. Uma vez que uma mesma questão pode ser realizada várias vezes, os seus resultados encontram-se identificados pela data e hora a que a questão foi realizada.

Para visualizar os resultados de uma dada questão, clique na data e hora a que foi realizada (1).

The screenshot shows the ClassQuiz interface. At the top, there is a navigation bar with links: New Question, My Questions, New Lesson, My Lessons, Results, Profile, Account, and Logout. Below the navigation bar, there is a search bar with the text 'Search course...'. A card titled 'Pergunta Exemplo' is displayed, showing 'LAR' and a date and time 'Date 14, 2019 - 16:05:53' which is highlighted with a red box and labeled with a red '1'.

Resultados de uma questão

Os resultados da questão são apresentados numa tabela.

A cor de cada resultado indica a aprovação, ou não, da resposta.

No caso de se tratar de uma questão anónima, é omitida da tabela tanto a identificação do aluno, como a avaliação da resposta.

Para exportar os resultados para Excel, clique em **Export to Excel** (1).

Para visualizar os resultados de uma lição, clique em **My Lessons** (2) a partir de qualquer janela.

The screenshot shows the ClassQuiz interface. At the top, there is a navigation bar with links: New Question, My Questions, New Lesson, My Lessons, Results, Profile, Account, and Logout. The 'My Lessons' link is highlighted with a red box and labeled with a red '2'. Below the navigation bar, there is a table with the following data:

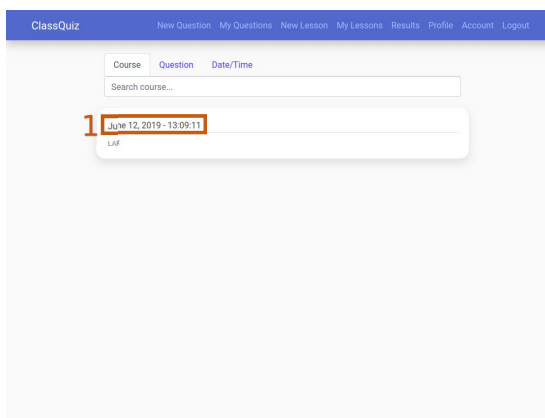
Course	Quiz ID	NMec	Answer	Time	Evaluation
LAR	37	00001	B	6.954199	right
LAR	37	00002	A	4.597493	wrong

Below the table, there is a button labeled 'Export to Excel' which is highlighted with a red box and labeled with a red '1'.

Resultados de uma lição

Nesta janela são apresentadas todas as lições criadas, identificadas pela data e hora a que foram criadas, bem como da Unidade Curricular a que estão associadas.

Para visualizar os resultados de uma dada lição, clique na data e hora a que foi criada (1).



Resultados de uma lição

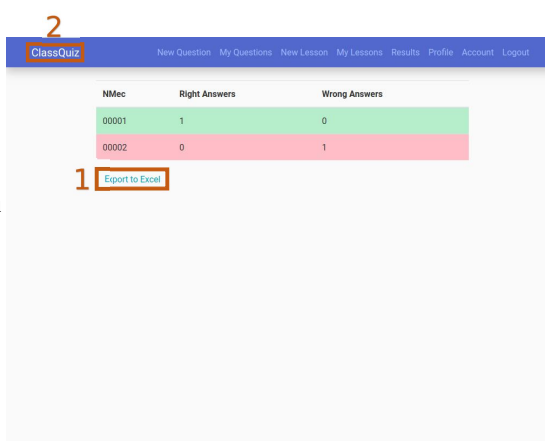
Os resultados de uma dada lição são apresentados numa tabela.

A cor de cada resultado indica a aprovação, ou não, de cada aluno.

Os resultados das questões anónimas não são contabilizados nem apresentados nesta tabela.

Para exportar os resultados para Excel, clique em **Export to Excel** (1).

Para voltar à página principal, clique em **ClassQuiz** (2) a partir de qualquer janela.



CLASSQUIZ

Terminal Individual

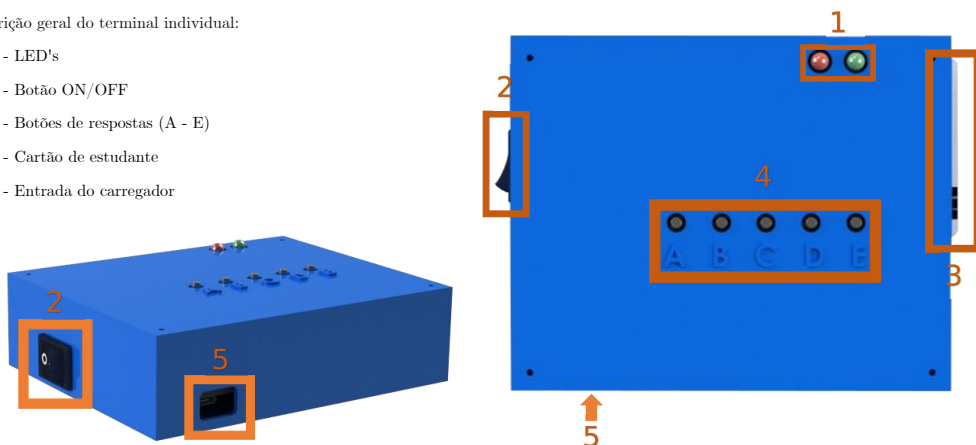


35

Terminal individual

Descrição geral do terminal individual:

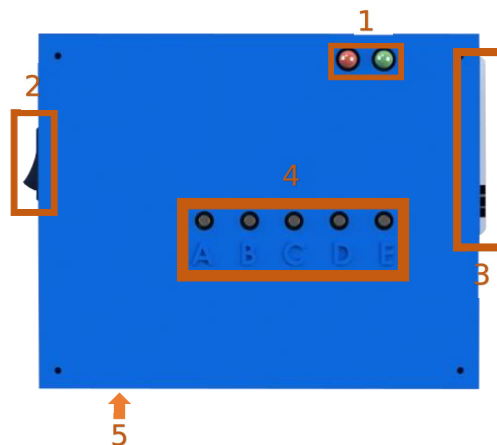
- 1 - LED's
- 2 - Botão ON/OFF
- 3 - Botões de respostas (A - E)
- 4 - Cartão de estudante
- 5 - Entrada do carregador



Terminal individual

Ao ligar o terminal individual (1), acende o LED vermelho (2), indicando que o terminal ainda não se encontra ligado ao servidor.

Quando a conexão tiver sido estabelecida, o LED vermelho apaga-se. Se a qualquer instante a luz vermelha se voltar a acender, significa que a conexão caiu. Neste caso, poderá reiniciar o terminal, ou premir qualquer botão e aguardar alguns instantes.



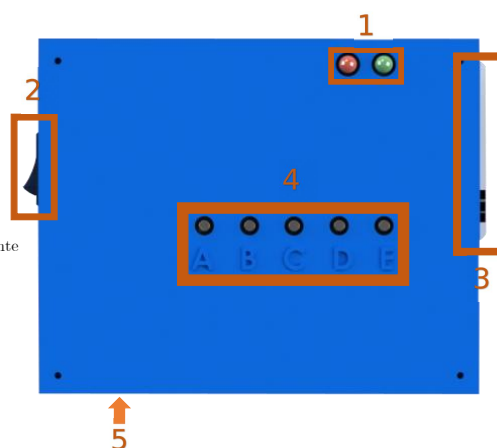
Submissão de respostas

Uma vez estabelecida a conexão, o LED vermelho apaga-se (2).

O terminal encontra-se então pronto a enviar respostas para o servidor.

No caso de se tratar de uma questão autenticada, será necessária a presença do cartão de estudante no terminal para que as respostas sejam validadas.

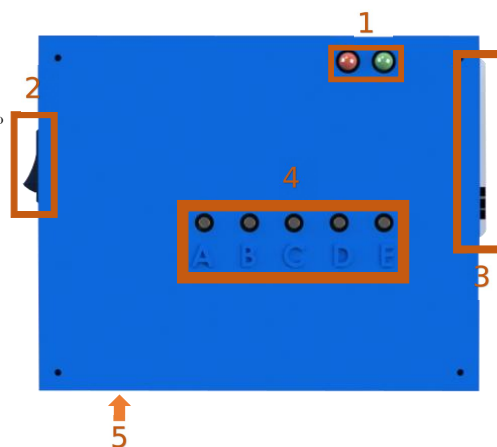
Caso se trate de uma questão anónima, a presença do cartão de estudante é opcional.



Submissão de respostas

Alguns instantes depois de premir um botão (4), o LED verde pisca, significando que a resposta foi recebida com sucesso pelo servidor.

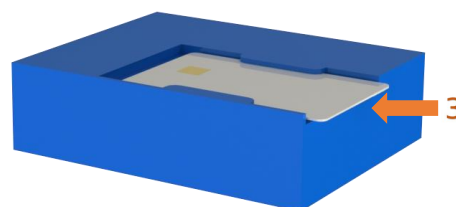
As respostas só serão validadas quando submetidas durante a realização de uma questão, ou seja, após o seu começo e antes de o tempo chegar a zero.



Inserção do cartão de estudante

Para inserir o cartão de estudante no terminal, basta fazê-lo deslizar na ranhura existente na parte inferior do mesmo (3).

Uma vez inserido, o LED verde acende, indicando que existe um cartão presente.



Recarregar a bateria

Para recarregar a bateria do terminal, ligue um cabo micro-USB à entrada **5**, como se de um telemóvel se tratasse.

Por questões de segurança, aconselha-se que o terminal se encontre desligado durante o recarregamento da bateria (**2**).

O processo de carregamento é bastante demorado (pode demorar várias horas), e não existe nenhum tipo de *feedback* sobre o estado do carregamento.

Após algum tempo (1 - 2 horas), retire o carregador.

