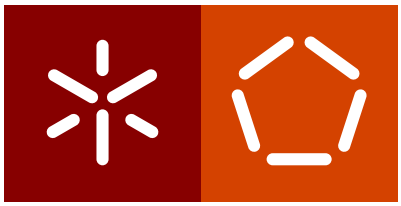
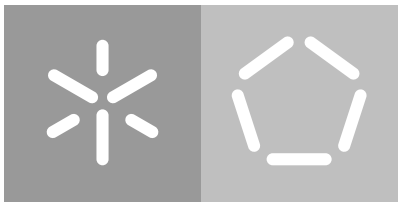


Universidade do Minho
Escola de Engenharia
Departamento de Informática

Matias Nicolau Araújo

Desmaterialização de documentos de identificação

Novembro 2019



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Matias Nicolau Araújo

Desmaterialização de documentos de identificação

Dissertação de Mestrado
Mestrado em Engenharia Informática

Dissertação sob a orientação de
José Bacelar Almeida

Novembro 2019

Despacho RT - 31 /2019 - Anexo 3

Declaração a incluir na Tese de Doutoramento (ou equivalente) ou no trabalho de Mestrado

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

Licença concedida aos utilizadores deste trabalho

Atribuição
CC BY

https://creativecommons.org/licenses/by/4.0/

AGRADECIMENTOS

A realização desta dissertação de mestrado não seria possível sem o apoio de certas pessoas às quais estou muito grato.

Em primeiro lugar queria agradecer ao José Eduardo Pina Miranda pela oportunidade de realizar a dissertação em ambiente de estágio profissional na Devise Futures, e por toda a orientação e sugestões durante o desenvolvimento do projeto. E ao meu orientador, Professor José Carlos Bacelar Almeida pela ajuda na elaboração da estrutura da tese, bem como na escrita da mesma, assim como pela disponibilidade ao longo do ano.

De seguida queria agradecer à minha família e amigos que me apoiaram durante o desenvolvimento do projeto e escrita da tese. Em particular aos meus colegas que me ofereceram sugestões, e à minha tia e ao meu avô que se disponibilizaram para ler a dissertação e fazer sugestões para a melhoria da mesma.

Por fim, queria agradecer à minha namorada por todo o apoio que me ofereceu durante este ano, e por acreditar sempre em mim e me incentivar a fazer mais e melhor.

Despacho RT - 31 /2019 - Anexo 4

Declaração a incluir na Tese de Doutoramento (ou equivalente) ou no trabalho de Mestrado

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

RESUMO

Os telemóveis com os quais andamos hoje em dia já não são utilizados apenas para chamadas e mensagens. Em vez disso utilizamo-los também para ver vídeos, ler o jornal, ou até consultar a nossa conta bancária. Dada esta grande variedade de utilizações, os dispositivos móveis têm tido um enorme avanço tecnológico, o que leva a ainda mais possibilidades. Por isso, porque não utilizá-los também como uma carteira dos nossos documentos de identificação?

Esta dissertação tem como objetivo apresentar alguns dos projetos que estão a ser desenvolvidos na área da identificação pessoal, assim desenvolver uma alternativa possível aos documentos de identificação físicos com que estamos acostumados. Esta alternativa tem o nome de documentos de identificação desmaterializados, que são no fundo, versões digitais dos documentos de identificação que já utilizamos.

Durante este trabalho, foi desenvolvido um protótipo de um sistema de documentos de identificação desmaterializados, de forma a mostrar que estes sistemas funcionam e são seguros. Este tipo de sistema pode vir a ser utilizado por qualquer pessoa, para o armazenamento de todos os documentos de identificação em dispositivos móveis.

Keywords: EletronicID, Aplicações Móveis, Documentos de Identificação Desmaterializados, Segurança da Informação, Mobile driver's license.

ABSTRACT

The mobile phones we use in the current days are no longer used just for calling or messaging. Instead, we also use them for watching videos, read the newspaper or even check our bank account. Given this high variety of uses, mobile phones have been suffering a huge technological advance, which leads to even more possibilities. So why not use them as a wallet for our identification documents too?

This dissertation aims to present some of the projects that are currently being developed in the area of personal identification, and also develop a possible alternative to the physical IDs we are currently accustomed with. This alternative is named dematerialized identification documents. Which are basically digital versions of the identification documents we currently use.

During the project, a prototype of a dematerialized identification document was developed, to show that these systems are functional and secure. This type of system might be used in the future by anyone, to store all the identification documents on mobile devices.

Keywords: EletronicID, Mobile Applications, Dematerialized Identification Documents, Information Security, Mobile driver's license

CONTEÚDO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÓNIMOS

AAMVA American Association of Motor Vehicle Administrators.
AES Advanced Encryption Standard.

CBOR Concise Binary Object Representation.

ECDH Elliptic Curve Diffie-Hellman.
ECDSA Elliptic Curve Digital Signature Algorithm.
EEE Espaço Económico Europeu.
eID Documento de Identificação Desmaterializado.
ENISA European Union Agency for Cybersecurity.

FIPS Federal Information Processing Standard.

HMAC Hash-based Message Authentication Code.

ICAO International Civil Aviation Organization.
IETF Internet Engineering Task Force.
ISO International Organization for Standardization.

JSON JavaScript Object Notation.

mDL Mobile Driver's License.
MEI Mestrado em Engenharia Informática.

NFC Near Field Comunication.

PDF Portable Document Format.
PIN Personal Identification Number.

RFC Request for Comments.
RGPD Regulamento Geral sobre a Proteção de Dados.
RN React Native.
RSA Rivest–Shamir–Adleman.

SHA Secure Hash Algorithm.

TLS Transport Layer Security.

UE União Europeia.

UM Universidade do Minho.
URL Uniform Resource Locator.

WSGI Web Server Gateway Interface.

1

INTRODUÇÃO

1.1 ENQUADRAMENTO

Os dispositivos móveis, mais especificamente, os *smartphones*, têm vindo a ser cada vez mais adotados como um bem essencial, que quase todos temos, e sem o qual já não se sai de casa para qualquer lado a que se vá. Esta ampla adoção leva a que seja muito provável que qualquer pessoa que encontremos no nosso dia a dia tenha em sua posse um *smartphone*. Para além disto, tem-se assistido também a um desenvolvimento enorme destes dispositivos, desenvolvimento este que permite que os *smartphones* sejam utilizados para outros fins que não apenas a comunicação, como por exemplo fotografia, *streaming* de vídeos, navegação na Internet, gestão de contas bancárias, etc, sendo comparável a um computador de bolso ao qual qualquer pessoa tem acesso.

1.2 MOTIVAÇÃO E OBJETIVOS

Tendo em conta o que foi falado anteriormente, uma função possível para um *smartphone* pode ser a de documento de identificação/carteira de documentos de identificação. Esta dissertação apresenta uma proposta para a concretização desta nova funcionalidade baseando-se em técnicas e algoritmos criptográficos para proceder à desmaterialização destes documentos de identificação, guardando-os à distância de um toque no nosso dispositivo móvel. Os objetivos principais deste trabalho são assim:

- Fazer o ponto de situação da desmaterialização de documentos de identificação, nomeadamente no âmbito de novos standards internacionais que estão a ser desenvolvidos;
- Contribuir para os standards internacionais que estão a ser desenvolvidos, através da Comissão Técnica portuguesa;
- Avaliar tecnologia e ferramentas *open source* que possam ser utilizadas na desmaterialização de documentos de identificação;

- Desenvolver um protótipo de documento de identificação desmaterializado (protótipo das componentes *app* mobile e componentes servidor), baseado nos (pré)-standards existentes;
- Formalizar o modelo dos documentos de identificação desmaterializados e explicar porque é que os mesmos garantem as características de segurança desejadas.

1.3 Estrutura da dissertação

Esta dissertação está organizada em 6 capítulos. No primeiro capítulo é apresentado o contexto do trabalho, assim como a motivação e objetivos do mesmo, sendo também apresentada a estrutura do documento corrente.

No segundo capítulo, o estado da arte, são explicados os conceitos base e os fundamentos necessários para a melhor compreensão do trabalho desenvolvido, sendo apresentada também a situação atual dentro do tema.

No capítulo três, estão especificados os requisitos do sistema a ser desenvolvido, assim como alguns aspetos a ter em conta durante o desenvolvimento do mesmo, de forma a explicar quais foram os principais desafios durante o trabalho realizado.

No quarto capítulo, é discutido o desenvolvimento do trabalho, começando-se pela modulação do sistema, passando pela escolha das tecnologias utilizadas e terminando com a implementação de um protótipo..

No quinto capítulo é apresentada uma prova de funcionalidade, explicando as razões do sistema garantir as características de segurança desejadas.

Por fim, o capítulo seis, é a conclusão desta dissertação, contendo uma pequena avaliação dos resultados finais e do trabalho desenvolvido, incluindo também uma pequena projeção possível de trabalhos futuros que possam surgir a partir deste.

2

ESTADO DA ARTE

A utilização de documentos de identificação é totalmente indispensável nos dias correntes. Seja para viajar, conduzir ou até mesmo ir às compras, necessitamos de documentos que possam comprovar a nossa identidade. Portanto, qual será a importância do desenvolvimento de um sistema de documentos de identificação desmaterializados?

A importância é tal que, em Portugal, já existem soluções para a utilização de documentos de identificação utilizando o telemóvel. Um exemplo simples será a aplicação do cartão continente, que é uma aplicação móvel que permite ao cliente deste hipermercado ter o seu cartão no telemóvel. O slogan desta App é "Andar com o cartão sempre atrás é tão século passado"e , segundo o site criado para a aplicação[1], tem já mais de 600 mil utilizadores, provando que para os titulares deste cartão, este tipo de soluções é útil e pertinente.

Focada mais no tema da identificação governamental, temos ainda a aplicação "id.gov.pt", uma aplicação criada pela Agência de Modernização Administrativa, que permite ao seu utilizador associar alguns dos seus cartões, sendo estes o cartão de cidadão, a carta de condução e o cartão do Instituto de Proteção e Assistência na Doença (ADSE). Esta aplicação tem as suas limitações, dado que, para permitir a consulta de documentos de identificação, exige que o utilizador entregue o dispositivo onde os documentos estão guardados para permitir que os mesmos sejam consultados, ou seja, não existe maneira de transferir documentos para outros dispositivos, para consulta. Para além disto, é uma aplicação do governo, o que fará com que só possa guardar documentos oficiais do governo.

2.1 MOBILE DRIVER'S LICENSE

Como exemplo concreto de uma proposta atual para a desmaterialização de documentos recorreu-se à mDL, visto que já foi alvo de análise cuidada no âmbito de processos de standardização.

1 https://www.cartaocontinente.pt

2.1.1 *Definição*

A *Mobile Driver's license* é atualmente o principal exemplo deste tipo de soluções de desmaterialização de documentos de informação. Este tipo de sistema consiste num titular (o dono do dispositivo móvel que contém a carta de condução desmaterializada), um emissor (que como o nome indica é a autoridade emissora do documento), o consumidor (que é a pessoa ou entidade que quer ter acesso ao documento) e um leitor (que é o dispositivo do consumidor). A Figura 1 apresenta uma visão simples deste sistema mostrando as interações entre as várias entidades.

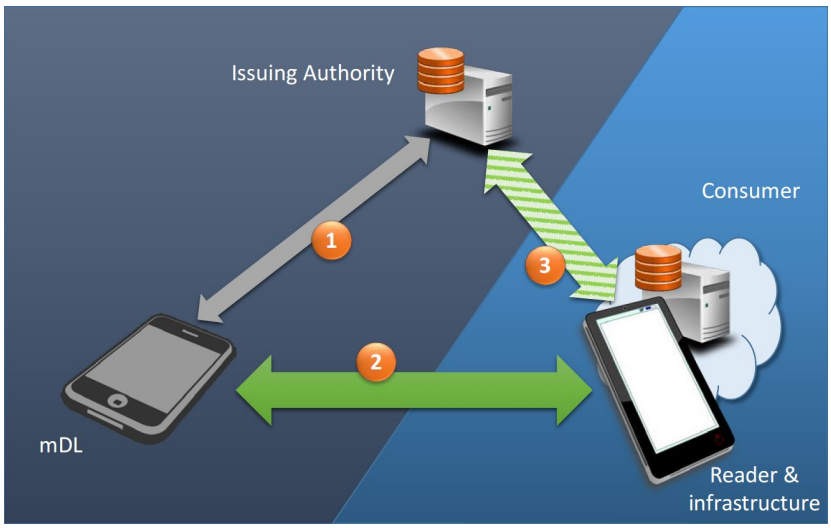


Figura 1: Exemplo de sistema de mDL[9]

Nesta figura podemos observar três interações. A primeira, representada pelo número 1 na figura, é a interação entre a mDL e a autoridade emissora, que controla entre outros a maneira como a informação da mDL é guardada no dispositivo e como são feitas atualizações, sendo representada na figura com o número 1. A interface entre a mDL e o leitor, que servirá para a mDL comunicar ao leitor, o cartão e as propriedades do mesmo que o leitor queira consultar, é a interação representada pelo número 2. E, finalmente, representada pelo número 3, temos a interface entre a autoridade emissora e o consumidor/leitor, que facilita a troca de informação necessária para permitir que o leitor confirme a autenticidade de uma dada mDL.

2.1.2 *Requisitos principais de sistema mdl*

Em [9], são apresentados os principais requisitos para um sistema de mDL. Segundo este whitepaper criado pela American Association of Motor Vehicle Administrators (AAMVA), um mDL deve:

1. Ser capaz de funcionar num ambiente *offline*

   Segundo este ponto, tanto a mDL como o *reader* estarão *online* durante a maioria das ações de verificação de identidade. No entanto, em alguns casos um dos dois dispositivos poderá estar *offline*. O *reader* pode estar *offline* durante a interação com o *holder*, desde que se ligue com uma certa frequência à autoridade emissora, de modo a manter-se atualizado. Já o *holder* deve poder estar *offline* até ao fim da sua validade, precisando apenas de estar *online* no momento de configuração inicial da mDL.

2. Incluir mecanismos que permitam aos utilizadores estabelecer confiança na informação providenciada pela mDL

   De maneira a haver confiança existem alguns pre-requisitos:

   - A informação na mDL foi emitida pela autoridade suposta;
   - A informação na mDL não foi alterada após a sua emissão a não ser que tenha sido atualizada pela autoridade responsável.

   Tendo em conta estes pre-requisitos, o processo de verificação pode explicar-se da seguinte forma: o consumidor confia no dispositivo leitor, este dispositivo obtém a informação da mDL proveniente da aplicação e autenticada pelo emissor, e por fim, o leitor passa a informação validada ao consumidor.

3. Confirmar a identidade do titular e atestar a aptidão de condução do titular;

   A confirmação de identidade tem o objetivo de confirmar que uma dada mDL pertence ao seu dono. Numa carta de condução normal isto é feito através de uma fotografia tipo retrato que se encontra no documento e pode ser vista pelo consumidor, e comparada com seu titular de forma a confirmar que é a mesma pessoa.

   Uma mDL deve, no mínimo, oferecer também esta opção. Mas dada a prevalência de outros métodos de autenticação em dispositivos móveis (por exemplo impressão digital e PINs) é expectável que estas soluções também sejam aplicáveis, sendo de esperar no entanto que a fotografia continue a constituir uma parte importante do processo.

   a) Fotografia tipo retrato

      A mDL tem de incluir uma fotografia do seu titular. Opcionalmente, caso a comunicação em tempo real esteja disponível, a fotografia pode ser disponibilizada diretamente pela autoridade emissora em tempo real. O equipamento de leitura tem de ter acesso ao retrato e ser capaz de o mostrar:

      - Ou o equipamento onde a mDL está guardada funciona como leitor, mostrando a fotografia no ecrã,

- ou a mDL reside no dispositivo do seu titular, sendo transmitida ao dispositivo leitor através de NFC/*Bluetooth*/etc, sendo depois mostrada do ecrã do leitor.

b) O retrato e o seu titular têm de ser visíveis ao mesmo tempo neste caso, significando que na maioria dos casos o leitor tem de ser um dispositivo móvel (por exemplo numa operação STOP).

c) Como dito acima é possível que o equipamento leitor seja o próprio dispositivo móvel do titular. Neste caso alguns pontos relevantes são:

- Visto que o dispositivo leitor não está sob o controlo do consumidor têm de ser estabelecidos meios adicionais de confiança;
- Surgem alguns problemas caso o titular tenha de entregar o seu dispositivo ao consumidor;
- Considerando a possibilidade do titular não querer entregar o seu dispositivo ao consumidor e apenas mostre o retrato sem que o consumidor possa pegar no dispositivo, mesmo que tenha sido estabelecida confiança na mDL, o caso é comparável a mostrar a carta de condução física sem a entregar ao consumidor.

d) Autenticação biométrica

Dados os requisitos, o uso de tecnologia biométrica para ligar uma mDL ao seu titular implica:

- O equipamento de leitura tem de ser capaz de ler os dados biométricos da mDL (ou ir buscá-los à autoridade emissora caso haja a possibilidade de comunicação em tempo real);
- Tem de haver equipamento de leitura biométrica juntamente com o equipamento leitor da mDL;
- Tem de ser feita a comparação entre os dados biométricos na mDL e os dados lidos do titular, sendo que isto tem de ser feito em tempo real na maior parte dos casos;
- Tanto o equipamento de leitura biométrica como o equipamento leitor da mDL têm de estar na posse do consumidor, sendo preferencial que se encontrem no mesmo dispositivo;
- A não ser que não haja falsos positivos ou negativos, esta solução tem de ser suportada por comparação visual do titular da mDL e do retrato nela presente.

e) Personal Identification Number

Outro possivel metodo de ligar o titular de uma mDL à sua mDL é através de um PIN. Este PIN estaria guardado na mDL de maneira segura de modo a que o processo de autenticação utilize tanto o PIN na mDL como o PIN inserido na hora pelo titular para confirmar a sua igualdade. Este conceito tem os seguintes problemas:

- Uma possível falha de segurança (PIN comprometido) pode permanecer não detetada a não ser que se use leitura biométrica em simultâneo.
- O processo de autenticação tem de ser de confiança quer por parte do consumidor quer por parte do titular da mDL. Isto é um problema porque nem todos os titulares vão querer introduzir o PIN num dispositivo do consumidor por medo que este seja comprometido. Por outro lado o consumidor pode não ter confiança caso todo o processo seja realizado no dispositivo do titular.

4. Permitir a leitura de informação entre várias jurisdições

   Já há standards para cartas de condução físicas que mostram a necessidade da utilização das mesmas através de várias jurisdições. Um leitor de mDLs deve ser capaz de ler mDLs emitidas por outras autoridades emissoras.

5. Permitir ao titular autorizar seletivamente o acesso à informação no documento;

   a) Consentimento do titular e libertação seletiva de informação

      O consumidor só deve ter acesso a uma mDL com o consentimento do seu titular. Para além disso, no caso de uma interação com um consumidor não oficial, o titular deve ser capaz de controlar a informação que é enviada, e deve ter a hipótese de limitar a informação a algo dentro de um dos campos da mDL, e não apenas limitar quais campos mostrar. Por exemplo, deve ser possível mostrar que se tem mais de 18 anos, em vez de mostrar a data de nascimento.

      No caso de uma autoridade (por exemplo policia) ou a própria entidade emissora, deverá ser possível ter acesso sem restrições (isto é, acesso não autorizado) a uma dada mDL em caso de necessidade.

   b) outras considerações de proteção de dados
      Uma mDL deve dar acesso ao titular a toda a informação pessoal guardada.

6. Suportar gestão remota da mDL

   Uma autoridade emissora deve ter a capacidade de gerir remotamente uma mDL. Isto inclui adicionar, atualizar, revogar privilégios de condução ou até revogar inteiramente uma mDL. Para além disso deve permitir atualizações técnicas à aplicação

(como por exemplo se uma mDL comprometer outra aplicação, ou se a própria aplicação da mDL precisar de ser atualizada).

Este acesso remoto não deve estar restrito à entidade emissora, sendo também desejável que o titular tenha acesso remoto à sua mDL.

Para suportar revogação remota de privilégios de condução, especialmente em casos parcialmente offline, uma mDL deve expirar automaticamente a não ser que o dispositivo em que se encontre fique online dentro de certos intervalos de tempo.

Caso uma mDL esteja ligada a uma carta de condução física, mudar remotamente a mDL pode levar a discrepâncias entre a mDL e o documento físico. Ações possíveis remotamente:

a) Revogar uma mDL
   Uma autoridade emissora pode precisar de revogar completamente uma mDL caso haja suspeita de fraude. Um titular pode precisar de o fazer também, por exemplo em caso de perda ou roubo do dispositivo em que a mDL está guardada. Apesar do titular poder dirigir-se à entidade emissora com essa finalidade, seria desejável que o pudesse fazer diretamente.
   - Revogar temporariamente uma mDL ou;
   - Revogar permanentemente uma mDL

b) Adicionar privilégios de condução

c) Mudar de uma carta de condução para um cartão de identificação

   Isto pode acontecer, por exemplo, caso os privilégios de condução sejam retirados ao titular mas este queira manter o documento para identificação pessoal.

d) Mudar de dispositivo

   É provável que o titular queira mudar o dispositivo da sua mDL em algum momento. Neste caso é desejável que o titular possa fazê-lo sem ter de se dirigir à entidade emissora só com este propósito.

7. Fornecer facilidade de utilização

   É desejável que estas condições sejam cumpridas:

   - Não ser necessário que o consumidor tenha de manusear o dispositivo do titular para ler a mDL.
   - Não ser negativamente afetado por condições meteorológicas como a intensidade da luz solar, chuva ou neve.
   - Funcionar a qualquer altura do dia ou noite.
   - Funcionar num ambiente de escritório.

- Ser suficientemente robusto para funcionar no exterior, por exemplo numa operação STOP.
- Minimizar o equipamento extra necessário que um agente das autoridades teria de carregar para poder ler e processar uma mDL.
- Minimizar equipamento adicional necessário para que entidades privadas consigam ler e processar uma mDL.

8. Ter um tempo de processamento aceitável

   Qualquer solução à volta de uma mDL deve permitir que a mesma seja processada e comparada com o seu titular num intervalo de tempo semelhante ao que demoraria a fazer o mesmo com um documento físico.

9. Utilizar infraestrutura de leitura existente

   Para o melhor funcionamento de um sistema de mDL a infraestrutura de leitura deve ao máximo utilizar equipamento já existente.

2.1.3 *ISO/IEC CD 18013-5*

O ISO/IEC CD 18013-5 [2], um standard ainda em desenvolvimento pela International Organization for Standardization (ISO), foca-se em aspetos mais técnicos ligados à mDL.

Este standard tem como base os standards ISO/IEC 18013-01 a ISO/IEC 18013-4, que estão ligados à carta de condução física. Ao longo do standard é explicada uma interação utilizando uma mDL, fala-se de quais os possíveis protocolos de transmissão entre o dispositivo do titular e o leitor e de protocolos de segurança que poderão ser usados para a transmissão e armazenamento de dados relativos à mDL.

Nessa norma é especificada ainda a formatação dos dados para uma mDL, que deverá ser armazenada por *data groups*, que provêm dos documentos de viagem (passaportes) e documentos de identificação físicos, criados originalmente pela ICAO. Nesta secção estará presente um pequeno resumo da interação com uma mDL, os atributos de segurança necessários, e a estrutura dos *data groups*.

Interação com uma mDL

A norma estudada especifica como uma interação utilizando uma mDL deve ocorrer, estando esta especificação representada na figura 2. Esta interação é iniciada pelo consumidor (o utilizador do leitor), que pede ao titular da mDL para consultar a sua mDL. Enquanto o consumidor espera que o titular prepare a sua mDL, o seu dispositivo deve estar já preparado. O titular vai então pegar no seu dispositivo e abrir a sua mDL. Caso a mDL esteja em

modo "*NFC ONLY*"os dispositivos entram em contacto imediato, caso contrário o titular deve, ou não, selecionar os dados que quer partilhar com o consumidor.

Passando à interação entre os dois dispositivos, caso o consumidor queira informação específica, é feito um pedido de dados adicionais, que vai ser comunicado ao titular. Por sua vez, o titular aceita ou não partilhar essa informação extra.

Após a primeira interação entre dispositivos, o primeiro conjunto de dados é transferido. Tendo acesso aos dados transferidos, o consumidor pode, ou não, pedir ainda mais informação ao titular da mDL. Caso não seja pedida informação extra, o consumidor pode consultar e validar a informação obtida. Caso o consumidor queira informação extra deve repetir o pedido de informação adicional, que por sua vez vai ser aceite ou recusado pelo titular.

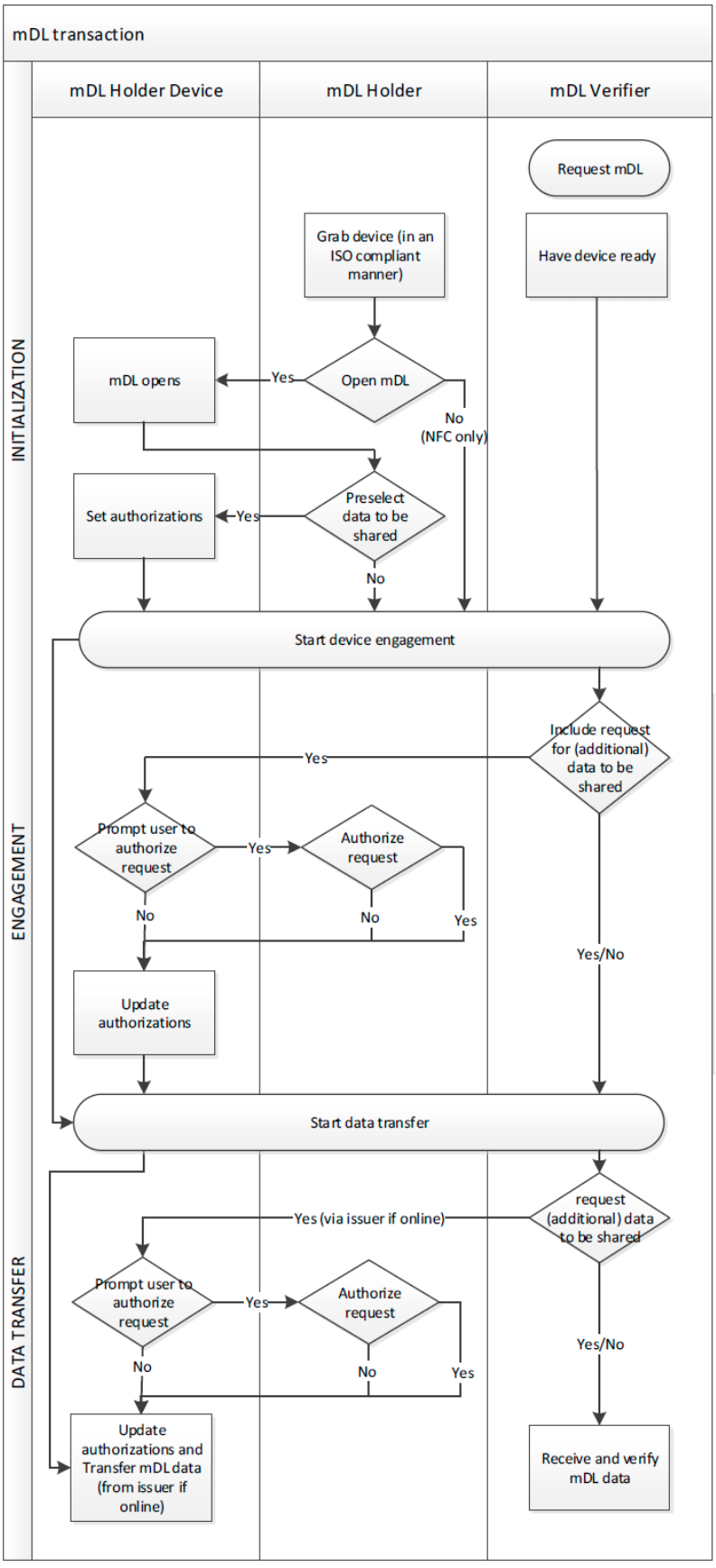


Figura 2: Interação utilizando uma mDL [2]

Atributos de segurança

Visto que uma mDL não tem presença física é importante que haja autenticação no dispositivo. A utilização de um mecanismo que suporte a criação de chaves efémeras fortes (PACE/EAC) deve ser implementado.

- Funções de hash:

  As autoridades emissoras devem escolher de entre as seguintes funções de hash: Secure Hash Algorithm (SHA)-1, SHA-224, SHA-256, SHA-384 ou SHA-512, sendo recomendada a utilização de SHA-256, e mantendo-se SHA-1 para compatibilidade com o Doc 9303 da International Civil Aviation Organization (ICAO)[11].

- Assinaturas:

  Para assinar deve ser usado ECDSA, escolhendo como curva uma das curvas apresentadas na tabela 1

Curve Name	Specification	Object Identifier
P-224 P-256 P-384 P-521	FIPS 186-4	RFC 5480 clause 2.1.1.1 [10]
brainpoolP224r1 brainpoolP224t1 brainpoolP256r1 brainpoolP256t1 brainpoolP320r1 brainpoolP320t1 brainpoolP384r1 brainpoolP384t1 brainpoolP512r1 brainpoolP512t1	RFC 5639	RFC 5639 clause 4.1 [8]

Tabela 1: Curvas para ECDSA segundo [2]

- Acordo de chaves:

  Para acordos de chaves que sejam necessários deve ser usado Elliptic Curve Diffie-Hellman (ECDH), utilizando as mesmas curvas mostradas acima, removendo as primeiras e últimas 4 linhas.

- Proteção de dados:

  O acesso aos dados numa mDL só deve ser feito com o consentimento do titular. Os métodos de consentimento não estão especificados no documento pelo que estarão à

responsabilidade das autoridades emissoras. Com a intenção de proteger a privacidade dos titulares de uma mDL, as instituições governamentais não devem rastrear o movimento de utilizadores individuais através de dados de *geo tagging*, armazenados na mDL ou recuperados através de um leitor.

Data Groups

Os *data groups* explicitam o modo como estão guardados em cada dispositivo os dados de uma mDL. Existem 127 *data groups*, sendo os grupos 1-12 os principais. Os grupos 32-127 referem apenas a combinações de dados parciais dos outros grupos para possibilitar ao utilizador a escolha de quais dados mostrar. Os grupos 1, 2, 3, 4 e 7 provêm da norma ISO 18013-2 [1].

1. Data group 1 - Dados Obrigatórios

   Este grupo é constituído por nove elementos:

   - Nome Próprio
   - Apelido
   - Data de Nascimento
   - Data de emissão
   - Data de validade
   - País de emissão
   - Autoridade emissora
   - Número de identificação
   - Categorias de veículos/restrições/condições

2. Data group 2 - Detalhes sobre o titular

   Grupo constituído por sete elementos:

   - Género
   - Altura
   - Peso
   - Cor dos olhos
   - Cor do cabelo
   - Local de nascimentos
   - Local de residência habitual

3. Data group 3 - Detalhes da Autoridade emissora

   Grupo constituído por quatro elementos:

   - Número administrativo
   - Via do documento (01 para o original, 02 para segunda via, etc)
   - Número de atualizações realizadas ao documento
   - Identificador da autoridade emissora

4. Data group 4 - Imagem tipo retrato

   Este grupo é constituído por as seguintes informações relativamente ao retrato:

   - *Timestamp* da imagem
   - Formato da imagem, WSQ, JPEG ou JPEG2000
   - Imagem

5. Data group 5 - Verificação de idade

   Este grupo contém apenas uma idade igual ou inferior à do titular.

6. Data group 6 - Reconhecimento facial

7. Data group 7 - Impressão digital

8. Data group 8 - Reconhecimento de íris

9. Data group 9 - Outras opções biométricas

10. Data group 10 - Política de atualização
    Este grupo é constituído por três elementos:

    - Versão
    - Última atualização
    - Data de validade
    - Próxima atualização esperada

11. Data group 11 - Uso doméstico

    Este grupo serve para qualquer entidade emissora ter algum parâmetro personalizado que seja usado apenas por essa entidade, ou qualquer outra informação que a entidade queira guardar e não esteja especificada nos outros grupos.

12. Data group 12 - Assinatura

2.1.4 *ISO/IEC CD 18013-5 - 2019*

Uma nova versão do SO/IEC CD 18013-5 [2], é a versão ISO/IEC CD 18013-5 - 2019 [3], em que são descartadas algumas das propriedades provenientes dos documentos de identificação criados pela ICAO, e que já estavam datadas no tempo. Estas alterações pretendem otimizar, tanto a quantidade de dados guardada nos dispositivos *holder* e nos servidores, como a quantidade de dados enviada em cada comunicação do sistema.
Nesta versão, os dados são guardados num formato JSON, o que facilita tanto a leitura (que fica em formato *pretty*) como a interpretação e consulta dos dados, e as transferências de dados são feitas num formato Concise Binary Object Representation (CBOR), de acordo com o RFC 7049[7], desenvolvido pela Internet Engineering Task Force (IETF).

2.1.5 *Trabalhos relacionados*

EUA - Colorado, Maryland e Washington DC

O Colorado, Maryland e Washington DC foram os foram primeiros estados americanos a lançar testes piloto de um sistema mDL (Julho de 2017). Estes testes terão uma duração prevista de dois anos e são desenvolvidos pela Gemalto. A primeira fase decorreu entre Julho e Agosto de 2017 e a segunda fase começou em Dezembro de 2017 e está ainda a decorrer. Os primeiros resultados foram facultados em Setembro de 2017 e podem ser consultados no site da gemalto[2].

EUA - Delaware

O Delaware foi um dos primeiros estados americanos a iniciar o desenvolvimento de uma mDL, e iniciou o seu teste piloto em Março de 2018. O teste inclui cerca de 200 utilizadores e tinha a duração planeada de 6 meses, mas desde o seu inicio não houveram mais atualizações.

Finlândia

Na Finlândia já foi aprovada uma lei que permite a utilização de uma mDL como carta de condução primária. Está neste momento a decorrer um teste piloto que teve *feedback* positivo. Dados os resultados positivos do teste, foi lançada a versão *Beta* da aplicação oficial, que já pode ser utilizada e é aceite pelas autoridades. Mais pormenores podem ser consultados no site da traficom[3].

2 https://www.gemalto.com/ddlpilot
3 https://www.traficom.fi/fi/autoilija-sovellus

2.2 REGULAMENTO GERAL SOBRE A PROTEÇÃO DE DADOS

A proteção de dados num sistema com documentos pessoais dos utilizadores é essencial. E sendo este sistema desenvolvido na Europa há um regulamento especial a ter em conta.

O RGPD[5] é um regulamento do direito europeu, criado em 2018, e que entrou em vigor a 25 de Maio de 2018, focado na privacidade de dados pessoais, e que é aplicável a qualquer sujeito na União Europeia ou no Espaço Económico Europeu. Regulamentando ainda qualquer exportação de dados para fora da União Europeia (UE) ou Espaço Económico Europeu (EEE). Este regulamento tem como principal objetivo entregar a cada individuo o controlo sobre os seus dados pessoais.

Um dos desenvolvimentos recentes mais importantes na legislação de privacidade e segurança, tem sido o aumento do foco no risco. O "princípio de risco"baseia-se na ideia de que as organizações que processam e utilizam dados pessoais devem dedicar mais recursos às ameaças mais significativas, e que a lei deve promover uma abordagem diferenciada em vez de uma regulação única, ou seja, devem haver vários níveis de risco. O RGPD aborda este principio de duas maneiras, vê o risco para os direitos e liberdades das pessoas como um *continnuum*, esperando que as empresas se esforçem mais à medida que o processamento dos dados pessoais aumenta a possibilidade de danos para o titular desses dados. Ou divide o risco em dois escalões, "risco"e "risco elevado", que terão obrigações diferentes.

Segundo o regulamento, as empresas, e organizações públicas e privadas, passam a ser responsáveis pela proteção de dados pessoais na sua posse, pelo que terão de tomar medidas para obedecer à Lei, sob a pena de multas. As empresas têm de adaptar os seus produtos que tratam dados pessoais para estarem de acordo com o RGPD relativo à proteção de dados pessoais de pessoas singulares e à livre circulação desses dados.

Os dados pessoais a serem protegidos englobam qualquer informação relativa a uma pessoa que possa ser utilizada para identificar direta ou indiretamente o titular dessa informação. Sendo estes dados, entre outros, nome, fotografia, endereço de e-mail, número de telefone, dados bancários, mensagens, informações médicas ou endereço de IP de computador.

O RGPD aplica-se a dados pessoais de qualquer cidadão da União Europeia (UE) ou Espaço Económico Europeu (EEE) independente de onde more ou da nacionalidade do cidadão, onde quer que estes estejam guardados (dentro ou fora da UE), que estejam guardados em ficheiros que sejam tratados manual ou automaticamente. O incumprimento deste regulamento leva a sanções que podem chegar aos 20 milhões de euros para grandes empresas, podendo as empresas ser adicionalmente penalizadas por eventuais danos causados pela individa aplicação do RGPD.

Segundo o regulamento, o titular dos dados deve ter alguns direitos fundamentais. São estes:

- Direito de acesso - a titular dos dados tem o direito de obter a confirmação de que os seus dados estão ou não a ser objeto de tratamento. Sendo que a pedido do titular, devem ser-lhe fornecidos todos os dados pessoais guardados, gratuitamente, de forma eletrónica.
- Direito de retificação - o titular dos dados tem o direito de obter a retificação dos dados inexatos que lhe digam respeito.
- Direito a ser esquecido - a pedido do seu titular, quaisquer dados pessoais armazenados devem ser apagados. Com excepções relacionadas com obrigações legais e/ou o interesse público e/ou a saúde pública.
- Direito de oposição - o titular dos dados tem o direito de se opor a qualquer momento ao tratamento dos dados pessoais que lhe digam respeito.
- Direito de portabilidade - implica a implementação de medidas para permitir o download direto desses dados.

Empresas que queiram cumprir com o RGPD devem adotar medidas de "*Privacy by Design*"e "*Data Minimization*. Sendo que para "*Privacy by Design*", as empresas devem adotar medidas internas, técnicas e organizacionais, que definam, de forma transparente e criteriosa, todo o processo de tratamento dos dados pessoais "desde a concepção". Para "*Data Minimization*"as empresas devem assegurar, por via de procedimentos técnicos claros, que só registam e tratam os dados pessoais estritamente necessários para cada fim estipulado.

No caso de um sistema de documentos desmaterializados, isto significa que qualquer dado dos utilizadores guardado pelas entidades envolvidas deve estar guardado de forma segura, principalmente caso estes dados de documentos estejam guardados fora dos dispositivos dos seus titulares ("*Privacy by Design*"). Para obedecer ao RGPD, um sistema deste tipo deve ainda fornecer aos seus utilizadores maneiras de diminuir os dados mostrados, escolhendo apenas os dados que queiram que estejam disponíveis, de forma a tratar os dados pessoais estritamente necessários ("*Data Minimization*").

3

LEVANTAMENTO DE REQUISITOS

O objetivo deste trabalho é desenvolver um protótipo de um sistema de documentos de identificação desmaterializados, que ofereça a possibilidade de guardar qualquer documento numa carteira digital

Para que este sistema funcione corretamente, todos os seus elementos devem funcionar corretamente em conjunto, isto implica que sejam compatíveis entre si. Neste capítulo apresenta-se a especificação de cada um dos componentes, detalhando alguns aspetos necessários ao funcionamento dos eIDs.

3.1 ALTERAÇÕES EM RELAÇÃO À NORMA

Visto a norma utilizada como base ser focada no documento de carta de condução existem algumas características a alterar e/ou acrescentar para a utilização geral de qualquer tipo de documento.

No caso das propriedades obrigatórias referidas na norma, estas devem ser reduzidas ao menor número possível, optando-se neste caso por manter apenas o número de identificação do cartão como propriedade obrigatória, isto porque é a única propriedade comum a todos os cartões de identificação. Para além disso deve haver informação que identifique o cartão (cartão de cidadão, carta de condução, etc).

3.2 ESTRUTURA DO SISTEMA

Para este sistema pretende-se utilizar uma estrutura semelhante à especificada em [2], que inclui diferentes entidades.

Em primeiro lugar inclui uma carteira de eIDs, que será uma aplicação móvel onde estarão guardados os documentos desmaterializados, esta aplicação deverá ser instalada no *smartphone* do titular dos documentos, que poderá consultá-los, ou partilhá-los com quem tiver a aplicação leitor.

A carteira de eIDs interage com servidor pertencente à autoridade emissora (cada autoridade emissora terá o seu servidor) , também chamado de servidor emissor, que que disponibiliza uma API e armazena os dados dos cartões emitidos por essa entidade. A API inclui os métodos de geração e envio dos cartões tanto para a carteira, como para o verificador..

Por último, a infraestrutura do leitor, que neste caso deve ser dividida em dois elementos, o leitor e o verificador. O leitor será a aplicação móvel que está instalada nos dispositivos das entidades que queiram ler eIDs (o consumidor), e irá comunicar com o emissor através de um servidor verificador, para que não exista comunicação direta entre o leitor e o emissor, isto serve para que o leitor não tenha conhecimento direto dos servidores emissores. Cada servidor verificador funcionará como um *proxy*, que terá guardados os vários links e certificados dos servidores emissores, de modo a que haja confiança mútua entre os verificadores e os emissores.

A estrutura do sistema pensado tem por isso 4 elementos, apresentados na figura 3.

No caso do sistema a ser desenvolvido pretende-se ainda que seja possível a sua utilização caso o dispositivo com a carteira de eIDs esteja *offline*, sendo que todos os outros componentes do sistema têm de estar obrigatoriamente *online*.

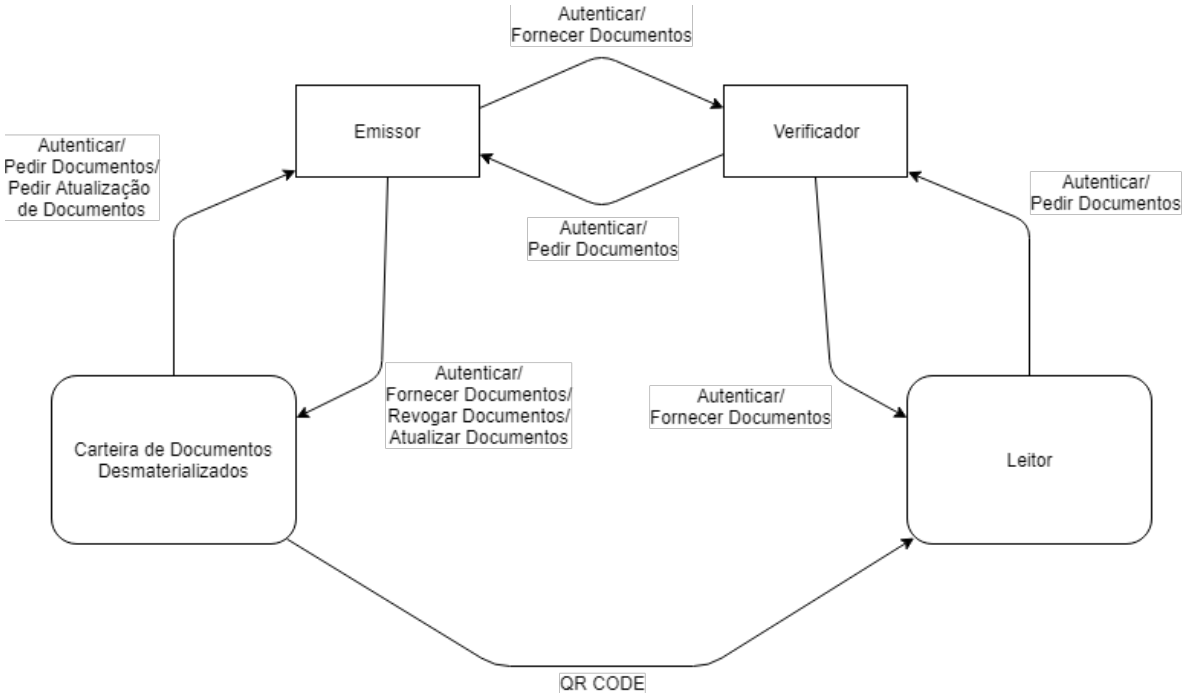


Figura 3: Esquema do sistema de documentos desmaterializados

3.3 CARTEIRA DE DOCUMENTOS DESMATERIALIZADOS

A carteira é o componente deste sistema direcionado ao utilizador comum, por isso tem de ser uma aplicação simples e intuitiva. Para tal deve obedecer aos seguintes requisitos funcionais:

- A aplicação deve estar bloqueada por uma *password* ou por impressão digital;
- O utilizador pode ter vários cartões guardados;
- O cartão a ser consultado deve ser mostrado num menu inicial tipo carteira;
- No menu inicial deve haver um botão para adicionar um novo cartão;
- A inserção de novos cartões deve ser simples e exigir o mínimo de informação possível;
- Ao selecionar um cartão o utilizador deve ter como possibilidade escolher quais os atributos do mesmo que quer mostrar;
- Deve ser possível apresentar no ecrã um QR que tenha o pedido de um cartão que mostre apenas os atributos escolhidos pelo utilizador;
- A aplicação tem de estar disponível tanto para *android* como para *ios*.

Os requisitos não funcionais já identificados são:

- A carteira deve ter de raiz as chaves públicas dos emissores de confiança, de forma a poder validar as informações recebidas;
- Todos os cartões guardados devem estar cifrados e não acessíveis fora da aplicação;
- Para cada cartão, para além do próprio cartão guardado, devem ser guardados alguns metadados sobre o cartão e o servidor do emissor, para facilitar a realização de pedidos;
- Para manter a validade dos seus cartões, um utilizador da aplicação da carteira tem de ficar *online* e fazer um *refresh* do documento periodicamente.

3.4 EMISSOR

O servidor emissor terá de conter tanto os tipos de dados incluídos nos documentos de identificação da entidade à qual pertence (campos de cada documento), como os dados de todos os cartões (os dados de cada cartão individual). Isto significa que os dados relativos a cada tipo de documento (por exemplo cartão de cidadão), devem estar separados no

template de documento e dados do documento, de forma a diminuir o espaço de armazenamento necessário. Os seus requisitos funcionais são:

- A inserção de documentos e templates deve ser simples e standardizada;
- Deve ser possível adicionar certificados de novos verificadores de confiança.

Já os requisitos não funcionais identificados são:

- O emissor deverá armazenar os certificados dos verificadores de confiança;
- Os dados pessoais guardados neste servidor devem estar inacessíveis a terceiros, a não ser que haja autorização prévia dos utilizadores;
- A procura de cartões na base de dados deve ser rápida;
- Os dados sensíveis de cada cartão devem estar cifrados;
- A comunicação tanto com a carteira como com o verificador deve ser segura;
- O emissor deve ser capaz de receber e responder a pedidos de documentos;
- Os documentos enviados devem estar no formato PDF;
- O emissor deve ser capaz de gerar o PDF de um documento a partir dos dados de um template e de um documento individual;
- O emissor deve ser capaz de assinar os documentos criados de modo a ser possível validar a fonte dos mesmos.

3.5 LEITOR

A aplicação leitor será utilizada por entidades que queiram consultar e validar cartões guardados nas carteiras de eIDs, como por exemplo a polícia em operações stop, ou em qualquer situação que requeira a validação da identidade de um cidadão. Deste modo, pretende-se uma aplicação simples e que ofereça ao utilizador a confiança de que os dados que está a receber provêm de uma fonte legitima e segura. Para o leitor temos os seguintes requisitos funcionais:

- Ao abrir a aplicação, o utilizador terá de se autenticar com o seu verificador;
- Caso o utilizador seja uma autoridade, pode forçar o emissor a fornecer todos ou alguns dos atributos ocultados pelo titular;
- Deve ser capaz de ler códigos QR.

E ainda os seguintes requisitos não funcionais:

- Nenhum dado deve ficar guardado no dispositivo no final da interação com a carteira de eIDs;
- O leitor confia sempre no verificador e esta confiança deve ser assegurada pelo sistema;
- O leitor deve ser capaz de fazer pedidos de documentos ao verificador, através de um pedido criado pela carteira.

3.6 VERIFICADOR

O verificador funciona como um *middle man* de confiança tanto do leitor como do emissor, este serve para que o leitor não precise de ter contacto direto como emissor. Cada entidade ou conjunto de entidades terá um verificador em comum que poderá ter as suas próprias regras de segurança e confiança em vários emissores, como por exemplo um verificador que exija todos os campos de um dado cartão do emissor, no caso da polícia. Os requisitos funcionais do verificador são:

- Deve ser possível adicionar certificados de novos emissores de confiança;
- Deve ter mecanismos de autenticação para os leitores.

E os seus requisitos não funcionais são:

- Tem de ter os certificados dos emissores de confiança atuais guardados, de modo a validar os dados recebidos;
- A comunicação tanto com o leitor como com o emissor tem de ser segura;
- Deve ser capaz de receber pedidos do leitor;
- Recebendo pedidos do leitor, o verificador deve reencaminhá-los para o emissor, devolvendo a resposta do mesmo ao leitor;
- Os dados reencaminhados nunca são tratados pelo verificador.

3.7 CLAIMS

Algumas das propriedades de um determinado cartão são mostradas com mais frequência que as outras, como por exemplo a nacionalidade, a idade, ou até o nome em casos em que só se queira comprovar a identidade de um titular.

As *claims* provêem do SO/IEC CD 18013-5 (tanto da versão de 2018[2] como da de 2019[3]) e funcionam exatamente para estes casos. As *claims* são versões dos cartões que mostram apenas algumas das propriedades e são específicas de cada cartão, por exemplo no caso do cartão de cidadão poderá existir uma *claim* intitulada de "nacionalidade"que o utilizador poderá escolher quando quiser mostrar apenas a sua nacionalidade.

A existência de *claims* agiliza a interação entre o titular de uma carteira de documentos desmaterializados e o utilizador da aplicação do leitor, pois permite ao utilizador da carteira escolher uma *claim* a mostrar, ao invés de ter de escolher todas as propriedades a que pretende que o leitor tenha acesso. Para além disto, as *claims* permitem a diminuição dos dados mostrados em cada interação, sendo apenas mostrados os dados necessários, como é indicado no RGPD.

Quando é escolhida uma determinada *claim* deve ter-se em atenção que a fotografia do utilizador seja sempre mostrada pois esta é a principal forma de associação entre um eID e o seu titular.

Para além das referidas *claims*, o sistema a ser desenvolvido deve suportar *claims* que tenham propriedades específicas não presentes no cartão base, por exemplo, caso um titular queira provar que é maior de idade, deve haver uma *claim* especial que permita apresentar, no caso de um cartão de cidadão português, um cartão que diga apenas "maior de dezoito"e que confirme a sua maioridade sem precisar de mostrar a sua data de nascimento (data essa presente no cartão de cidadão base original). Estas *claims* devem ser criadas pela entidade emissora, e visam diminuir ao máximo a quantidade de informação que o titular precisa de mostrar em certas situações.

3.8 Questões de Segurança

Como referido anteriormente, o sistema desenvolvido deve obedecer ao RGPD, fornecendo medidas para a segurança dos dados dos seus utilizadores.

Essas medidas de segurança traduzem-se em:

- No servidor emissor, para além dos certificados dos verificadores, a informação dos eIDs deve estar separada em duas partes, templates e dados.
- Do lado dos templates deve estar a informação sobre o cartão no geral, ou seja, os campos (Nome próprio, Apelido, Número de identificação, etc), e onde estão localizados no cartão (Coordenadas (x,y)), bem como as *fonts* usadas no texto do cartão, e mais alguns metadados. Esta informação pode ou não ser cifrada dependendo da entidade responsável, caso seja uma entidade que não queira que o seu template de documentos esteja acessível.

Em relação aos dados, estes contêm, como o nome indica, os dados de cada campo do eID, e devem estar todos cifrados.

- No que toca à carteira de eIDs esta deve ter os vários cartões cifrados e inacessíveis de fora da *app*, sendo que quaisquer metadados pode em princípio estar em texto simples, visto não ser informação sensível. Sempre que é recebido um documento, a sua assinatura, proveniente do emissor, deve ser verificada, para validar a fonte do documento.
- Em relação ao leitor não há outras preocupações de segurança para além da garantia de que nenhum dado é guardado durante uma interação com uma carteira, e que os documentos recebidos são válidos, este último ponto traduz-se na validação da assinatura que vem juntamente com os documentos recebidos.
- Já o verificador deve ter apenas algum mecanismo de autenticação dos leitores (como por exemplo um sistema de *login*), e certificados dos emissores como garantia da sua segurança.
- Por fim toda a comunicação online neste sistema deve ser cifrada para garantir a segurança da informação transmitida.

4

DESENVOLVIMENTO

Para o desenvolvimento do protótipo do sistema de documentos de identificação desmaterializados optou-se por fazer a modulação das interações do sistema, assim como desenhar a arquitetura dos vários elementos do sistema, e decidir qual a estrutura de dados a utilizar para o sistema. Terminada esta modulação, passou-se à seleção das tecnologias a utilizar para o desenvolvimento do sistema e ao seu desenvolvimento propriamente dito.

4.1 MODULAÇÃO DO SISTEMA

4.1.1 *Interações*

Existem duas interações principais no sistema de documentos desmaterializados. O pedido de um cartão novo por parte da carteira e o pedido de consulta de eID por parte do leitor. Estas interações devem ter sempre o mesmo comportamento, tendo vários resultados possíveis.

Pedido de Cartão Novo

Este é um pedido realizado pela aplicação da carteira e tem como objetivo guardar um novo cartão que ainda não esteja na mesma mas pertença ao utilizador da aplicação. Para pedir um novo cartão o dono da carteira precisa de saber o URL do servidor emissor, o identificador do cartão que está a pedir (por exemplo, no caso de um cartão de cidadão português, "prtcc"), o número do documento em questão, e um *token* único fornecido pela entidade emissora, que irá identificar e autenticar o titular do documento. Após inserir estes dados no formulário de pedido de cartão, estes serão enviados para o emissor com o URL indicado, que irá validá-los. Caso os dados enviados não sejam válidos é recebida como resposta uma mensagem de erro a informar o utilizador da aplicação de que os dados inseridos não estão corretos. Caso os dados sejam válidos o servidor irá gerar o cartão pedido, juntamente com algum metadado que seja necessário enviar, e enviá-lo para

a aplicação da carteira, que irá validar por sua vez se os dados recebidos estão assinados pelo emissor de confiança, e por fim guardar todos os dados recebidos e o respetivo cartão.

Esta interação exige que o dispositivo que contém a carteira tenha ligação à internet. Para além disso é ainda necessário que o utilizador faça atualizações frequentes ao documento para o manter válido.

Este processo está demonstrado no diagrama apresentado na figura 4.

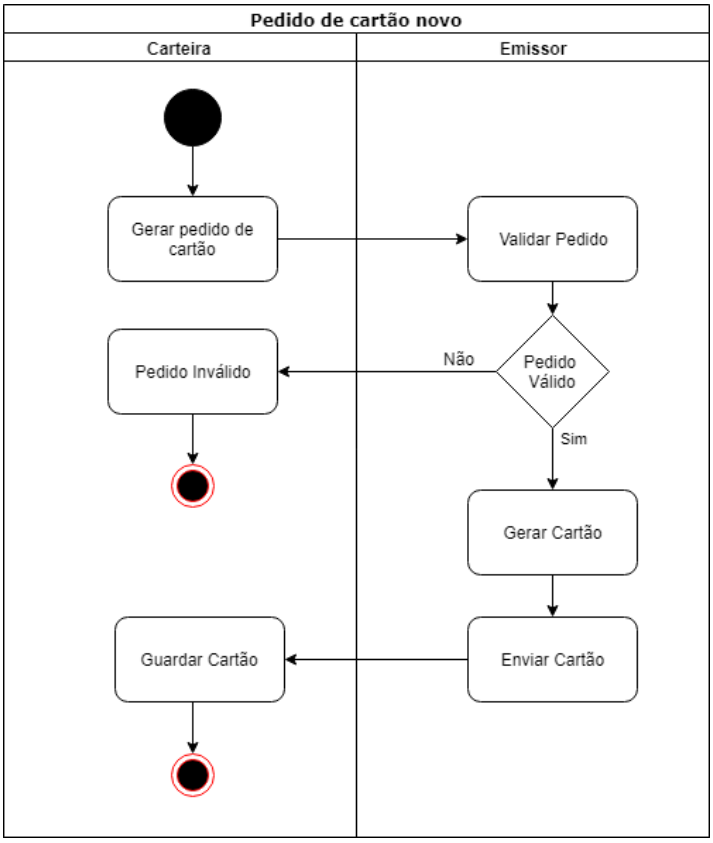


Figura 4: Diagrama de atividade do pedido de um cartão novo

Pedido de consulta de cartão

Esta interação será a mais complexa e mais frequente do sistema, e tem como objetivo validar um determinado documento guardado numa carteira de eIDs.

Para começar o utilizador da aplicação leitor deve comunicar ao utilizador da aplicação carteira quais os campos do cartão que precisa de validar. De seguida o utilizador da carteira deve escolher na aplicação os campos a passar e apresentar o QR respetivo.

A aplicação do leitor deve então ler o código QR e verificar se este é válido (isto é, se está no formato certo e se pode ser pedido ao seu verificador), e encaminhar o pedido ao verificador.

Chegando ao verificador será validado tanto o leitor como o emissor que consta do pedido. Caso algum destes seja inválido é enviada uma mensagem de erro ao leitor, caso contrário, o pedido será encaminhado para o emissor.

Quando o emissor recebe um pedido de verificação de cartão, a primeira coisa que deve fazer é verificar se este vem de um verificador de confiança. Em caso negativo é enviada uma mensagem ao verificador a comunicar a sua invalidade. Caso contrário procede-se à validação do próprio pedido. Para validar o pedido serão consultados os números de identificação do cartão pedido e o *token* enviado no pedido. Se o pedido for inválido o emissor manda uma mensagem ao verificador que por sua vez comunica ao leitor este facto. Se o pedido for validado com sucesso, o emissor passará ao processo de montar um cartão com as propriedades especificadas, e a enviá-lo de seguida ao verificador, que irá encaminhá-lo para o leitor, para que este possa consultá-lo.

Este processo está demonstrado no diagrama apresentado na figura 5.

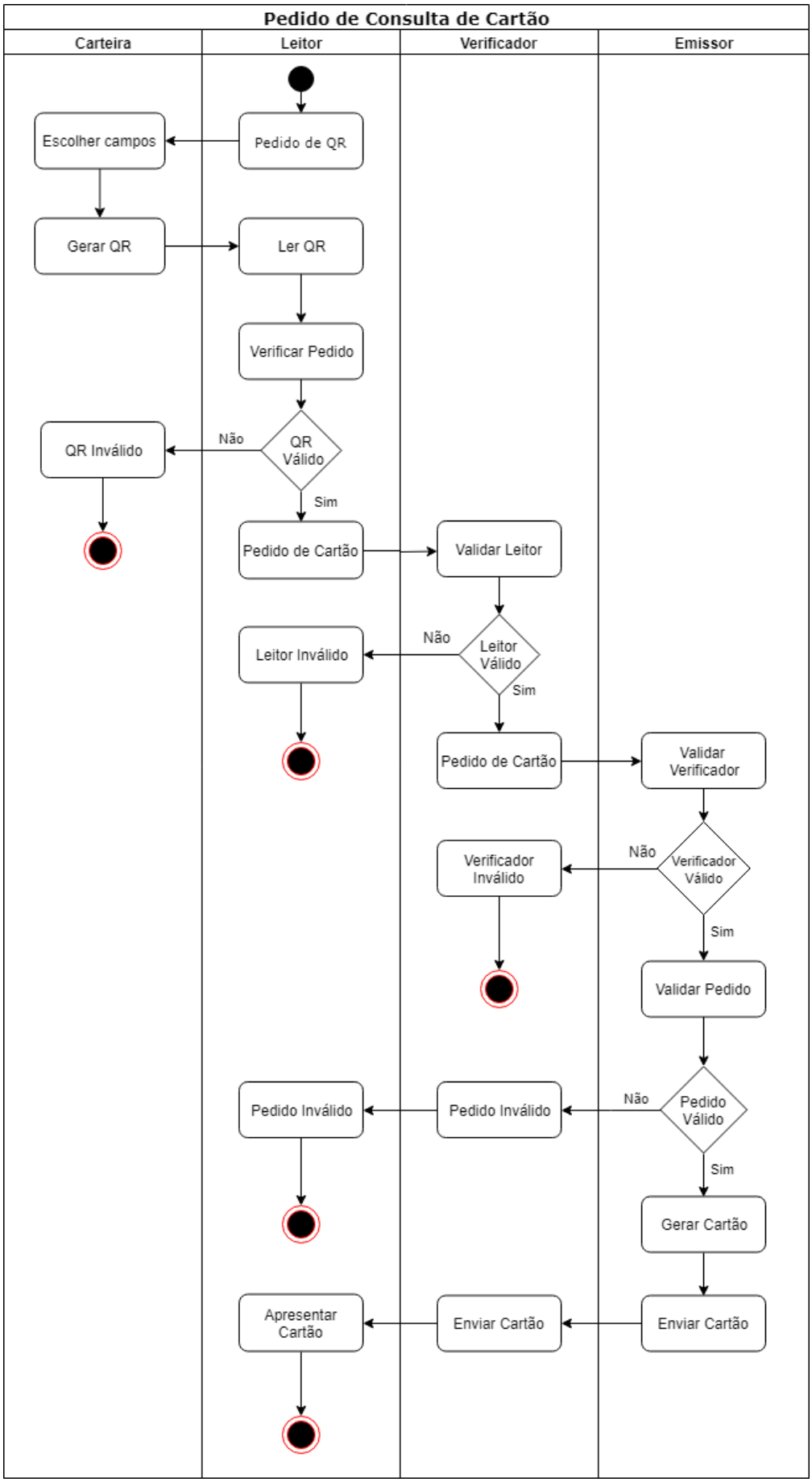


Figura 5: Diagrama de atividade do pedido de consulta de um cartão

4.1.2 *Dados*

Para que seja possível a utilização de eIDs as interações têm que ter aproximadamente a mesma duração que teriam no caso de documentos físicos. Para que isto aconteça, os dados que estão guardados em cada um dos elementos do sistema deve ser o menor possível, sem comprometer a validade e funcionalidade dos eIDs.

Dado o requisito de diminuir ao máximo os dados obrigatórios de cada cartão, os dados obrigatórios serão a identificação do documento, a versão do documento, a entidade emissora, o número do cartão, a última atualização, a próxima atualização obrigatória, e a validade, todas as outras propriedades do documento ficam à escolha da entidade emissora.

Com o objetivo de diminuir o espaço de armazenamento necessário no servidor, os cartões foram divididos em template e dados, como discutido no capitulo de levantamento de requisitos.

Os templates devem ter os seguintes dados:

- Identificação do documento
- Versão do documento
- Entidade emissora
- Fundo do documento:
  - Altura do documento
  - Comprimento do documento
  - Imagem de fundo frontal
  - Imagem de fundo posterior
- Propriedade frontais, uma lista de propriedades, na qual cada propriedade contém:
  - Identificação
  - Altura
  - Comprimento
  - Tipo (Texto, imagem, etc)
  - Coordenada X
  - Coordenada Y
  - *Font*
  - Tamanho da *font*
- Propriedades posteriores, uma lista de propriedades, cada propriedade contém:

  - Identificação
  - Altura
  - Comprimento
  - Tipo (Texto, imagem, etc)
  - Coordenada X
  - Coordenada Y
  - *Font*
  - Tamanho da *font*

- Propriedades estáticas (não dependentes do titular), contém uma lista de propriedades frontais e posteriores com os seguintes campos:
  - Identificação
  - Altura
  - Comprimento
  - Tipo (Texto, imagem, etc)
  - Coordenada X
  - Coordenada Y
  - *Font*
  - Tamanho da *font*
  - Conteúdo (o único campo diferente em relação às propriedades comuns)

- *Claims* normais, uma lista com as *claims* normais existentes, estas devem conter:
  - Descrição/Nome da *claim*
  - Lista de propriedades dessa *claim*

- *Claims* especiais são as *claims* que contêm propriedades que não estão presentes no cartão base. Estas *claims*, por serem basicamente um novo cartão, irão conter uma lista de propriedades iguais às das listas de propriedades acima especificadas. Isto é, as *claims* especiais funcionam como um template para um cartão personalizado, criado a partir de um documento base, guardado dentro do template dessse documento base.

- Lista com o nome das propriedades

- Lista com o nome das *claims* normais

- Lista com o nome das *claims* especiais

Os dados de cada cartão devem conter:

- Identificação do documento
- Versão
- Entidade emissora
- Última atualização
- Próxima atualização obrigatória
- *Token* de autenticação
- Número do cartão
- Validade
- Propriedades frontais, que contém uma lista com o nome da propriedade e o conteúdo da mesma
- Propriedades posteriores, que contém também uma lista com o nome da propriedade e o conteúdo da mesma
- *Claims* especiais, que contém uma lista de listas de propriedades, sendo cada uma dessas listas uma *claim* especial

Por sua vez, a carteira deve ter alguns dados para além dos próprios cartões, estes são:

- *Claims* do cartão, separadas em normais e especiais
- Propriedades do cartão
- URL base da API do emissor
- Versão da API do emissor
- *Token* de autenticação
- Identificador do cartão
- Identificador interno do cartão (para a própria aplicação)
- Próxima atualização obrigatória
- Assinaturas dos *hashs* das várias claims

Como foi dito anteriormente, tanto o leitor como o verificador não necessitam de quaisquer dados sobre os cartões, precisando apenas dos certificados do servidor emissor para o verificador e de mecanismos de autenticação para o leitor.

4.1.3 *Arquitetura do sistema*

Em termos de estrutura geral do sistema a estrutura utilizada será a da Figura 3, com uma aplicação de carteira de eIDs, um servidor emissor, um servidor verificador, e uma aplicação de leitor.

Tendo isto em conta, cada elemento terá a sua própria arquitetura, que deve ser especificada.

Carteira

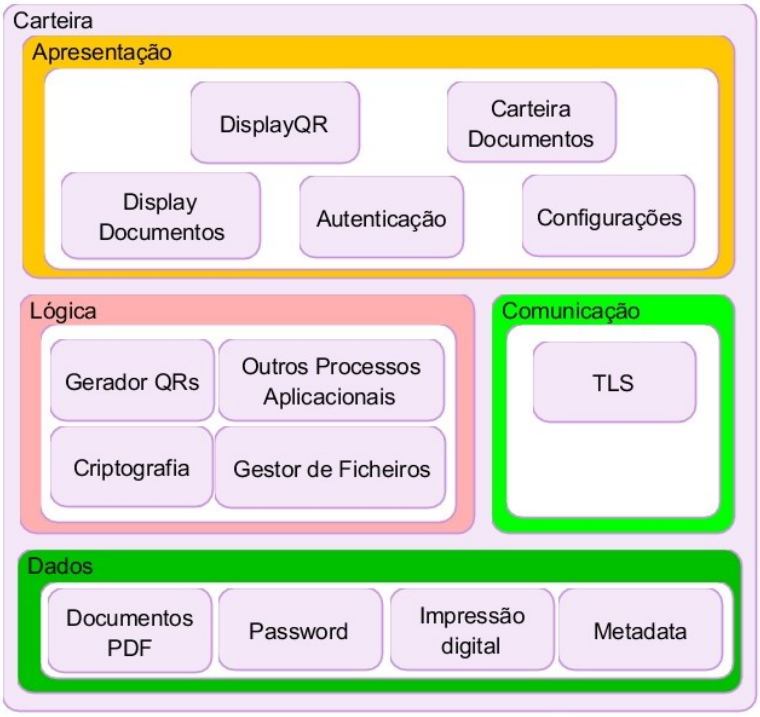


Figura 6: Arquitetura da aplicação Carteira

A carteira é a aplicação onde são guardados os documentos de identificação desmaterializados de um certo utilizador. Esta aplicação permite a inserção de novos documentos, assim como a apresentação dos mesmos e a impressão de QRs para que o leitor possa consultar os documentos e validá-los. A carteira está separada em quatro camadas, representadas na Figura 6. A camada superior é a camada de apresentação, que é a camada responsável pela interação com o utilizador, e que terá portanto um método de autenticação, um menu com os vários cartões, um ecrã de documentos, um *display* de códigos QR e um menu de configurações. A camada lógica contém todos os elementos lógicos do sistema,

como o gerador de QRs, o gerador de chaves de criptografia, a função de acordo de chaves e o gestor de ficheiros. Esta camada está ligada a todas as outras e é a parte central da aplicação. A camada de comunicação é responsável está responsável, como o nome indica, pela comunicação com o exterior, por TLS com o servidor emissor, e através de um QR com o leitor. Já a camada dos dados deve armazenar todos os dados importantes para o funcionamento da aplicação, como os eIDs da carteira, bem como os seus metadados e os dados de autenticação da aplicação, como palavra passe ou impressão digital.

Emissor

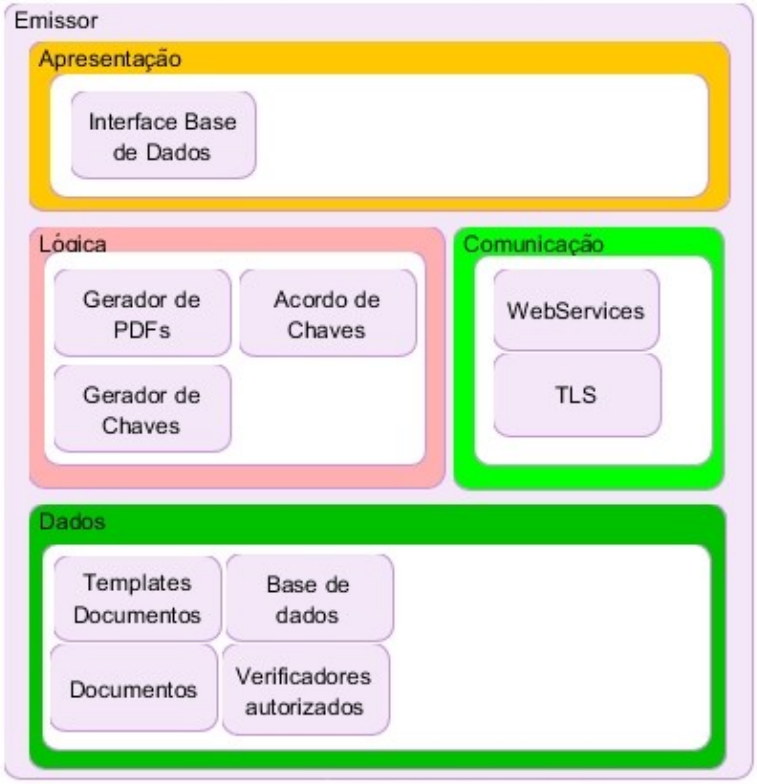


Figura 7: Arquitetura do Servidor Emissor

O servidor emissor, cuja arquitetura é apresentada na figura 7, é o servidor onde estão guardados todos os documentos de todos os utilizadores, para uma dada entidade emissora. Este servidor deve responder a pedidos de documentos por parte dos titulares dos mesmos (através da aplicação da carteira), assim como por parte de quem queira consultá-los e validá-los(através de um servidor verificador). Este elemento está dividido em dois, a API e a base de dados. Na API teremos a interface, a camada lógica e a comunicação, e na base de dados todos os dados armazenados neste servidor. Para o emissor, a interface com o

utilizador é apenas o gestor da base de dados, que permite adicionar novos documentos e templates de documentos, assim como atualizar informações dos mesmos, e até mesmo revogar cartões. Na parte lógica teremos todos os processos de geração de cartões, assim como as funções criptográficas, a parte lógica é a parte central do emissor e está em contacto com a interface, a base de dados e os *webservices* da comunicação. A parte da comunicação é responsável por toda a comunicação externa. E a parte dos dados contém uma base de dados com todos os documentos, templates dos mesmos, e certificados dos verificadores de confiança.

Leitor

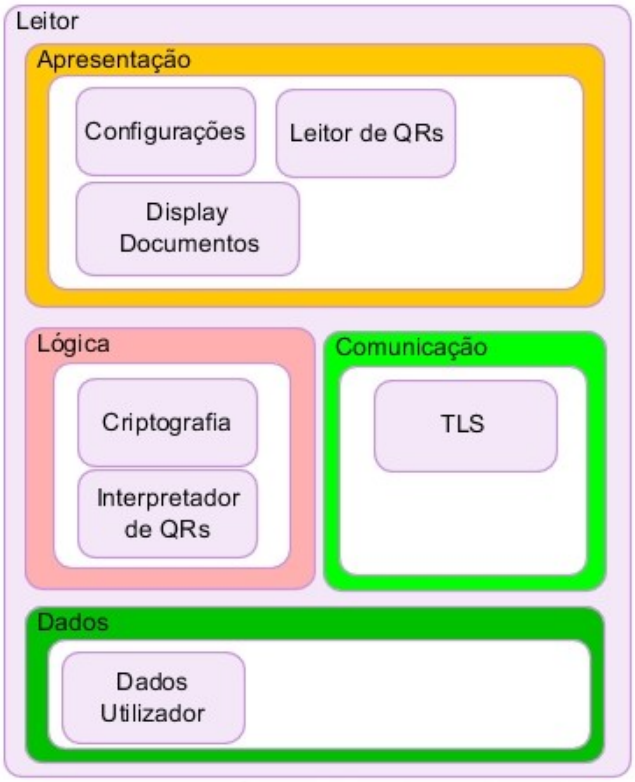


Figura 8: Arquitetura da Aplicação Leitor

A aplicação do leitor, cuja arquitetura está apresentada na figura 8 é a aplicação que permite ao seu utilizador ler QRs da aplicação da carteira, de modo a consultá-los no seu próprio dispositivo. Esta aplicação tem apenas três componentes de apresentação, o leitor de QRs, o *display* de documentos e as configurações. Na camada lógica está o interpretador de QRs, assim como a parte criptográfica. Na comunicação está o TLS para comunicação

com o verificador. E por fim, nos dados, existem apenas os dados do utilizador para autenticação com o verificador.

Verificador

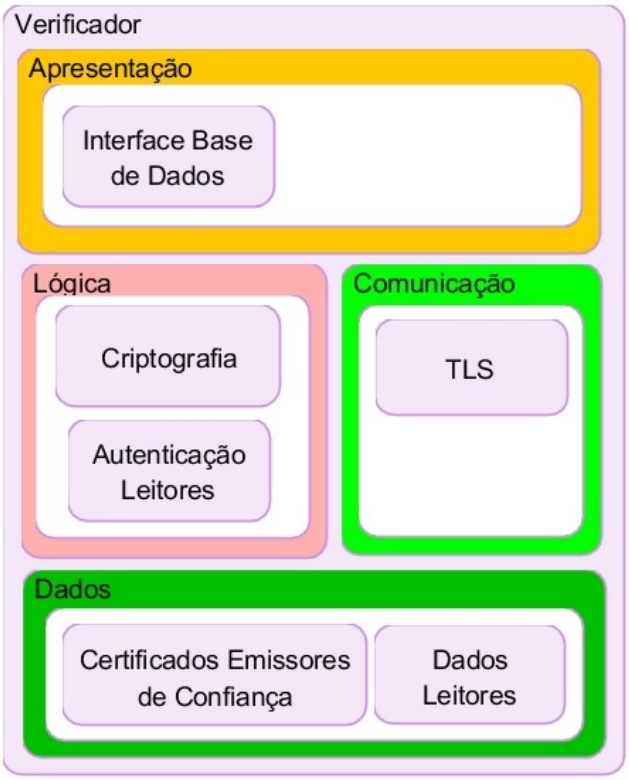


Figura 9: Arquitetura do Servidor Verificador

O servidor verificador funciona como um *middle-man* entre o servidor emissor e a aplicação leitor. Isto permite autenticar os leitores sem ocupar o servidor, e permite que haja confiança na informação que o leitor recebe do emissor, tudo isto sem que o leitor conheça diretamente o emissor, e vice-versa. A arquitetura deste servidor é apresentada na figura 9, e contém como interface o gestor da base de dados, que permite adicionar novos emissores de confiança, e novos leitores, assim como revogar os mesmos. Na parte lógica temos a autenticação dos leitores assim como toda a criptografia do servidor. O verificador comunica tanto com o emissor, como com o leitor, por TLS. Em termos de dados o verificador armazena os certificados dos emissores de confiança, assim como os leitores a si associados.

4.2 TECNOLOGIAS

Durante a tomada de decisão sobre as tecnologias a usar foram tomados em conta os requisitos abordados no capítulo anterior. Para as aplicações é necessário que possam correr tanto em *android* como em *ios*, para além de terem de suportar criptografia para a segurança dos dados. Para o servidor é necessário que a inserção de documentos na base de dados seja simples, a procura de cartões rápida e a comunicação com a carteira e verificador segura. Além disso deve ser capaz de responder a pedidos de documentos, vindos da carteira ou do verificador. Para uma maior consistência, optou-se por utilizar a mesma tecnologia para as duas aplicações móveis (carteira e leitor), assim como a mesma tecnologia para os dois servidores (emissor e verificador).

4.2.1 *Carteira e Leitor*

Há duas opções principais em relação ao desenvolvimento das aplicações móveis, utilizar código nativo, o que obriga ao desenvolvimento de duas aplicações idênticas de forma separada, uma para *android* (por exemplo em *android studio* ou *kotlin*), e uma para *ios* (em *swift*), ou utilizar uma das muitas ferramentas de desenvolvimento multi plataforma disponíveis.

Para agilizar o processo de desenvolvimento optou-se por desenvolver utilizando uma ferramenta multi plataforma, neste caso o *React Native*[1]. O React Native (RN), é uma *framework* de *javascript* desenvolvida pelo *Facebook*, baseada em *React*. Segundo [4], a implementação em *React Native* resulta num menor tempo de desenvolvimento comparado com o tempo combinado de desenvolvimento em código nativo, o que é essencial no caso deste trabalho, que tem tempo limitado para o desenvolvimento de duas aplicações. Outras opções seriam o *Xamarin*, o *Flutter* ou o *PhoneGap*, mas acabou por se optar pelo RN dado que no início desta dissertação era a ferramenta mais popular entre as quatro (como pode ser observado na Figura 10, proveniente da *Google Trends*). Para além disso, após uma pesquisa, verificou-se que oferecia bibliotecas para todos os elementos necessários.

1 https://facebook.github.io/react-native/

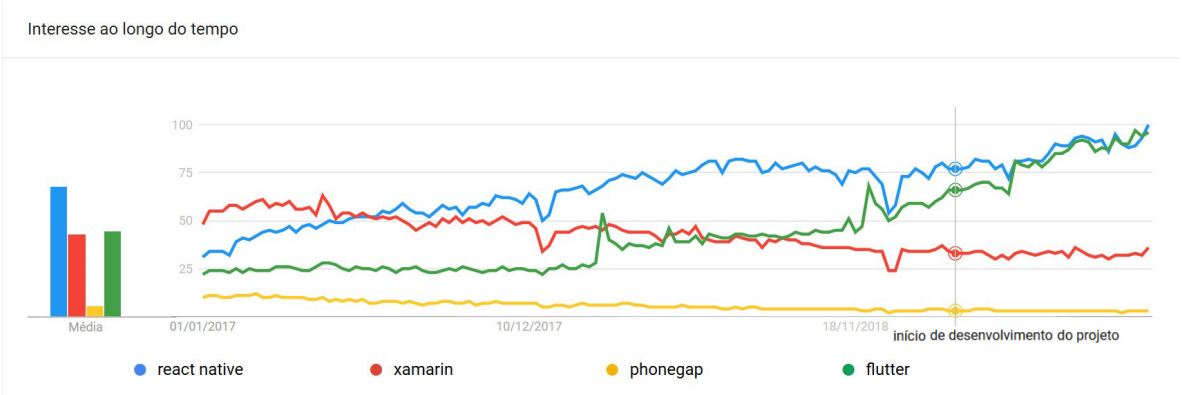


Figura 10: Gráfico de popularidade *Google Trends*

4.2.2 *Emissor e Verificador*

Com a finalidade de escolher a tecnologia utilizada para desenvolver os servidores foram pesquisadas várias alternativas. A primeira hipótese foi utilizar *Django*, uma *framework* de *python* para *web* que é utilizada por vários sites conhecidos, como o *Instagram*, o *Pinterest* e o *Disquss*. No entanto dado que não há muitos tipos de pedidos aos servidores (o emissor só recebe 4 tipos de pedido diferente e o verificador só tem um tipo de pedidos), e a simplicidade dos pedidos optou-se por um servidor *Flask*[2]. O *flask* é uma Web Server Gateway Interface (WSGI) bastante leve feita para um simples e fácil *deployment* e não necessita de nenhuma dependência instalada.

Visto o projeto ser baseado em [3], decidiu-se que a estrutura de dados devia ser o mais próxima possível com a norma, ou seja, JSON. Com este intuito optou-se por utilizar *mongoDB*[3]. Esta base de dados é formada inteiramente por coleções de JSONs, e suporta funções de cifra de acordo com os requisitos do servidor emissor, que explicam a necessidade de todos os dados dos utilizadores estarem cifrados.

Para além disto decidiu-se que os cartões gerados e enviados, tanto para o leitor como para a carteira, deviam estar em PDF, para permitir a assinatura dos dados enviados por parte do emissor.

4.3 Implementação do protótipo

A implementação de um protótipo de um sistema deste tipo, com diversos elementos, implicou a construção de um quadro de referência de modo a dividir o problema em vários problemas mais pequenos.

2 https://palletsprojects.com/p/flask/
3 https://www.mongodb.com/

Começou-se portanto, pela definição de uma ordem de prioridade de desenvolvimento para os vários elementos do sistema. Neste caso decidiu-se que o elemento com o maior grau de prioridade deveria ser o servidor emissor, por ser o elemento central do sistema, e por conter toda a estrutura de dados do sistema.

Como segunda prioridade está a aplicação do leitor, que, após desenvolvida, permite a consulta de documentos do emissor, mesmo que a aplicação da carteira não esteja terminada, através da leitura de QRs válidos, gerados por qualquer gerador de QRs.

De seguida temos a aplicação da carteira, que é no fundo, a parte principal para um potencial utilizador de um sistema de documentos desmaterializados. Apesar do seu grau de importância tendo em conta o desenvolvimento completo do sistema, a aplicação da carteira fica em terceiro na lista de prioridades de desenvolvimento, visto que os dois elementos a cima (o emissor e o leitor) podem ser testados sem que a carteira esteja terminada.

Por último, decidiu-se deixar o verificador em último na lista de prioridades, tomando-se ainda a decisão de que o verificador não seria necessário para o protótipo inicial, visto que é possível demonstrar o funcionamento do sistema sem a sua existência, deixando o desenvolvimento deste elemento para uma fase mais próxima de uma possível produção. Isto resulta num sistema protótipo mais simples, cuja estrutura está apresentada na figura 11.

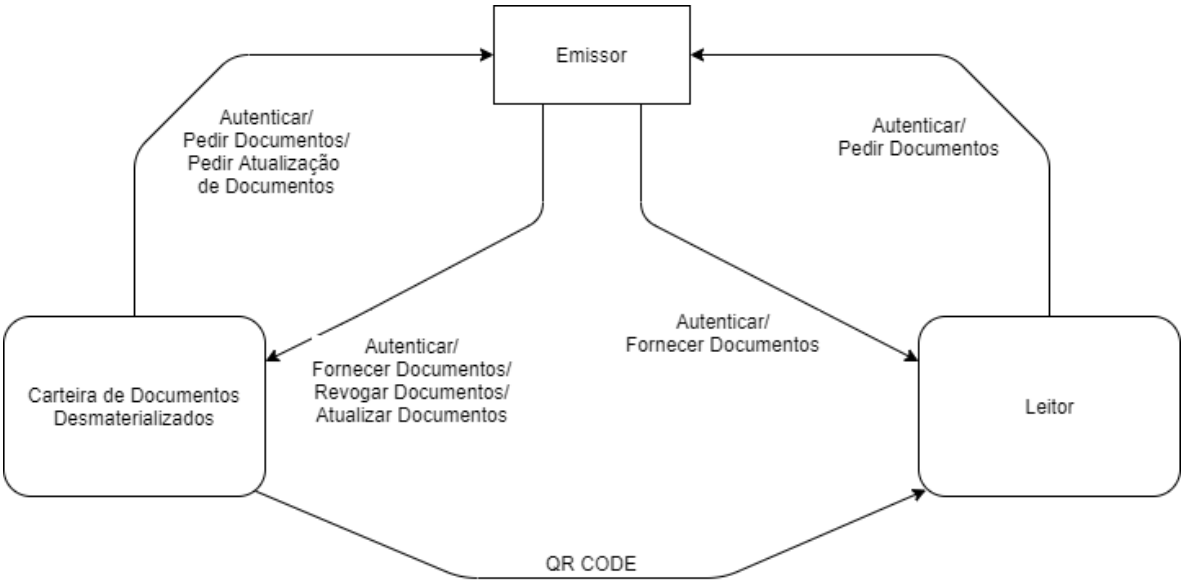


Figura 11: Estrutura do protótipo desenvolvido

Apesar desta abordagem ao desenvolvimento em separado, é importante que todos os elementos desenvolvidos em separado sejam compatíveis e consigam comunicar uns com os outros da forma desejada. Portanto deve-se ter atenção ao formato dos PDFs criados no servidor e aos dados dos cartões, de forma a estes poderem ser consultados nas aplicações.

4.3.1 *Emissor*

O emissor é responsável pelo armazenamento dos dados de todos os cartões de uma entidade emissora e para além disto tem também de responder a pedidos de acesso a estes dados quer para consulta do leitor, quer para armazenamento numa carteira. Este servidor tem vários componentes, dois dos elementos principais são a base de dados e a API. A base de dados deve conter todos os templates dos documentos de uma dada entidade emissora, assim como os dados desses documentos. E a API deve incluir os métodos para responder a todos os pedidos ao servidor.

Base de dados

Uma base de dados em *mongodb* é constituída por um conjunto de coleções, estas coleções são por sua vez, um conjunto de JSONs com dados.

A base de dados desenvolvida tem dois tipos de coleção, os *templates* e os dados. Os *templates* contêm uma lista com todos os *templates* dos cartões da entidade à qual o servidor pertence. Os dados são um conjunto de coleções com os dados dos cartões de cada *template*. Ou seja, para cada *template* na coleção de *templates*, existe uma coleção com os dados de todos os cartões com esse *template*, sendo o nome dessa coleção o nome do documento no template.

Assim sendo, um template tem a seguinte estrutura:

```
{
    "documentName" : "",
    "metaData" : {
            "version" : "",
            "issuer" : ""
    },
    "background" : {
            "sizeX" : "",
            "sizeY" : "",
            "contentFront" : "",
            "contentBack" : ""
    },
    "propertiesFront" : [],
    "propertiesBack" : [],
    "StaticFront" : [],
    "StaticBack" : [],
    "NormalClaims" : {
            "claimName" : {
```

```
                "desctription" : "",
                "fields" : []
        }
    },
    "SpecialClaims" : {
            "specialClaimName" : []
    },
    "PropertyNames" : [],
    "ClaimNames" :[],
    "SpecialClaimNames" : []
}
```

Sendo que cada propriedade (nas listas de propriedades) deve ter a seguinte estrutura:

```
{
    "id" : "",
    "sizeX" : "",
    "sizeY" : "",
    "type" : "",
    "coord" : {
                "x" : "",
                "y" : "",
                "layer" : ""
    },
    "font" : "",
    "fontSize" : ""
}
```

Por sua vez, os dados estão assim organizados:

```
{
    "metaData" : {
        "documentName" : "",
        "version" : "",
        "issuer" : "",,
        "lastRefresh" : "",
        "nextRefresh" : "",
```

```
        "token" : ""
    },
    "identificationNumber" : "",
    "validFrom" : "",
    "validTo" : "",
    "propertiesFront" : [],
    "propertiesBack" : [],
    "SpecialClaims" : {
        "specialClaimName" : []
    }
}
```

Tendo as propriedades da seguinte forma:

```
{
    "id" : "",
    "content" : ""
}
```

Como já dito anteriormente, esta separação entre os templates e os dados permite uma maior eficiência no armazenamento de dados visto que os dados dos templates só precisam de ser guardados uma vez, poupando espaço por cada cartão adicionado.

Para o armazenamento seguro dos dados dos documentos, *mongodb* oferece diversas possibilidades, a melhor opção, em que ficam cifrados todos os dados guardados, está limitada à versão *enterprise*, de modo a que a segunda melhor opção, um sistema de autenticação, foi a escolhida. Esta função de autenticação fornece acesso aos dados guardados na base de dados apenas aos utilizadores autorizados, que devem ser inseridos na interface do *mongodb*.

API

A API recebe os pedidos do servidor, e responde recorrendo aos geradores de cartões necessários. Para isto é necessário definir os pedidos possíveis e o formato dos mesmos. Neste caso o servidor deve responder a três tipos de pedido, o pedido de consulta de um cartão por parte do verificador, o *setup* inicial de um cartão por parte de uma carteira e a atualização de um cartão por parte de uma carteira também.

O encaminhamento utilizando *flask* é feito utilizando *paths*, que podem conter variáveis. Um URL de flask tem o seguinte aspeto:

```
@app.route('/hello/<name>')
```

Sendo o que está entre os delimitadores , que no caso do *flask* são os sinais de menor e maior, é uma variável.

Os pedidos para esta API são então os seguintes:

- Pedido de consulta de cartão por um leitor:

```
@app.route('/<template>/<card>/<token>/<claim>/<properties>')
```

- Pedido de *setup* de um documento por parte da carteira:

```
@app.route('/setup/<template>/<card>/<token>')
```

- Atualização de documento por parte da carteira:

```
@app.route('/refresh/<template>/<card>/<token>')
```

Para montar os PDFs é utilizada a biblioteca *fpdf*[4], que utilizando as informações nos templates e nos dados cria um PDF com o documento pedido. No caso do pedido de *setup* inicial são criados cartões para todas as *claims* para um certo documento e enviadas em conjunto com o documento base para a carteira. No caso do pedido de consulta é especificado se o documento pedido é o documento base, se é uma *claim* normal ou especial, ou se é pedido um documento personalizado, com certas propriedades escolhidas pelo leitor.

A cada PDF criado é aplicada uma função de *hash* (para as funções de *hash* é utilizada a bliblioteca *hashlib* [5]) com Secure Hash Algorithm (SHA)-256, este *hash* é depois assinado com Rivest–Shamir–Adleman (RSA) (para a assinatura com RSA utiliza-se a biblioteca *cryptography* [6]), utilizando a chave privada do servidor emissor de 3076 bits. Optou-se por uma chave de 3076, que deve ser suficiente para um futuro próximo, segundo a European Union Agency for Cybersecurity (ENISA) [6]. A assinatura criada é enviada juntamente com o documento gerado, de modo a validar o emissor.

O token utilizado pelo sistema para identificação do documento, quer no pedido de *setup*, quer no pedido de consulta, é atualizado cada vez que o utilizador faz uma atualização do cartão (que tem uma periodicidade obrigatória de sete dias).

Tendo em conta o que se falou anteriormente sobre manter a compatibilidade entre os dados e documentos gerados no servidor foi preciso eliminar o último e os dois primeiros

4 https://pyfpdf.readthedocs.io/en/latest/
5 https://docs.python.org/3/library/hashlib.html?highlight=hashlib
6 https://cryptography.io/en/latest

caracteres dos PDFs gerados pelo servidor, de modo a que as aplicações consigam lê-los e armazená-los conforme o desejado. Isto porque python acrescenta estes caracteres ao serializar os documentos em PDF para base64.

4.3.2 *Leitor*

A aplicação do leitor será operada por utilizadores que queiram ter acesso a documentos dos quais não sejam os titulares. Como por exemplo as forças policiais ou várias outras entidades governamentais. Para isso a aplicação do leitor é capaz de ler QRs que são apresentados na aplicação da carteira, e que permitem ao leitor fazer um pedido de consulta ao emissor. Visto que os documentos recebidos vêm diretamente do emissor, os dados neles contidos podem ser tomados como fidedignos, isto porque os documentos vêm juntamente com uma assinatura, que permite à aplicação do leitor validar a proveniência do documento recebido. Há ainda que referir que esta aplicação permite que o utilizador da aplicação da carteira não precise de entregar o seu dispositivo a quem queira consultar os seus documentos. A aplicação desenvolvida fornece uma navegação simples com apenas dois ecrãs para a consulta de um documento, não armazenando nenhum dos dados do cartão recebido para consulta.

Para o desenvolvimento desta aplicação foram usadas cinco bibliotecas principais. Uma para a navegação, uma para a leitura de QRs, uma para a apresentação dos documentos, e duas para a validação dos documentos recebidos.

Navegação

Para a navegação na aplicação, a biblioteca utilizada foi "react-navigation"[7], esta biblioteca possibilita a divisão da aplicação em vários ecrãs, criando uma classe para cada ecrã. O exemplo de uma estrutura de aplicação com dois ecrãs é:

```
class HomeScreen extends React.Component {
    //code for HomeScreen here
}

class DetailsScreen extends React.Component {
    //code for DetailsScreen here
}

const AppNavigator = createStackNavigator(
    {
```

7 https://reactnavigation.org/

```
            Home: HomeScreen,
            Details: DetailsScreen,
        },
        {
            initialRouteName: 'Home',
        }
    );
```

No exemplo apresentado existem dois ecrãs, identificados na função de "createStackNavigator", são estes o ecrã "Home"e o ecrã "details". Para além disso é especificado ainda o ecrã inicial ("initialRouteName").

No caso da aplicação do leitor temos dois ecrãs, um inicial para leitura do código QR e um ecrã para a consulta de um cartão, tendo portanto uma estrutura semelhante à do exemplo.

Leitura de QRs

Para ler códigos QR utilizou-se a biblioteca "react-native-camera-kit"[8], que é uma biblioteca com um conjunto de utilitários da câmara do dispositivo, entre eles a leitura e interpretação de QRs.

Consulta de documentos

Os documentos provenientes do emissor estão em formato PDF, e para lê-los é necessária uma biblioteca que permita a leitura deste formato. Neste caso optou-se pela biblioteca "react-native-pdf"[9], que permite consultar PDFs provenientes de fontes com vários formatos, como demonstrado na tabela 2.

Neste caso optou-se por utilizar uma string em base64, visto que funciona para *android* e *ios*, e ainda possibilita a utilização do mesmo formato dos documentos enviados pelo servidor.

Não foi utilizado como fonte o URL do pedido ao servidor, pois esta utilização do URL implica utilizar todos os dados recebidos diretamente como fonte para o PDF, e neste caso há mais informação enviada pelo servidor para além do próprio documento, como por exemplo o *hash* do documento com a assinatura do servidor para validação da fonte do documento. Após realizar o pedido de documento, a resposta é armazenada numa variável que é passada como fonte (*source*) à biblioteca de leitura dos PDFs, para apresentar o documento para consulta no ecrã.

8 https://github.com/wix/react-native-camera-kit
9 https://www.npmjs.com/package/react-native-pdf

Utilização	Descrição	iOS	Android
uri:"http:// xxx/xxx.pdf"	Carregar a partir de um URL	Sim	Sim
require ("./test.pdf")	Carregar a partir de um ficheiro JavaScript	Sim	Não
uri:"bundle-assets://path/to/xxx.pdf"	Carregar a partir de android/app/src/main/assets/path/to/xxx.pdf	Não	Sim
uri:"bundle-assets://xxx.pdf"	Carregar a partir do projeto em xcode	Sim	Não
uri: "base64data"	Carregar de uma string em base64	Sim	Sim
uri:"file:///absolute/path/to/xxx.pdf"	Carregar a partir de um ficheiro local	Sim	Sim

Tabela 2: Possibilidades para *source* de PDF

Validação dos documentos

Para a validação dos documentos recebidos foram utilizadas duas bibliotecas, uma para a função de hash, e uma para a função de validação das assinaturas. Foram estas: "react-native-sha256"para a função de *hash*, e "react-native-rsa-native"para a validação das assinaturas.

Ao enviar um documento para o leitor, o emissor vai enviar juntamente um hash do mesmo documento assinado com RSA. Esta assinatura pode ser validada pela aplicação do leitor, aplicando uma função de hash ao documento, e correndo uma função de *verify* com a assinatura, hash do documento, e chave pública do servidor, que vem já armazenada na aplicação do leitor.

4.3.3 *Carteira*

A aplicação da carteira irá permitir aos seus utilizadores guardar os seus documentos de identificação desmaterializados. Estes documentos podem ser documentos oficiais do governo, cartões de supermercado ou documentos de identificação de ordens, como por exemplo um cartão da ordem dos engenheiros, etc. Para pedir cartões o utilizador deve contactar o servidor emissor, realizando o pedido através de um *token* de identificação que lhe é fornecido pela entidade emissora (esta entidade deve ser uma das entidades emissoras de confiança, que terão as suas chaves públicas englobadas do código fonte da aplicação). Para passar um documento ao leitor para consulta, o utilizador da carteira deve escolher o cartão a consultar, escolhendo ainda se quer mostrar a totalidade do cartão, alguma das *claims* disponíveis, ou ainda se quer personalizar a informação a ser mostrada, escolhendo quais das propriedades do documento que quer mostrar ao leitor.

Para o desenvolvimento da aplicação do leitor foram usadas três bibliotecas principais, uma para navegação, uma para o armazenamento dos dados dos documentos, e uma para a verificação dos documentos recebidos.

Para a apresentação dos documentos armazenados foi utilizada a mesma biblioteca utilizada para o leitor, com os mesmos parâmetros (a *source* é uma string de base64).

Navegação

Para a navegação na aplicação foi utilizada a mesma biblioteca utilizada no leitor, a "react-native-navigation". No entanto, neste caso existem mais ecrãs, incluindo um ecrã "*pop-up*"para a impressão dos QRs. Isto levou à utilização de um ecrã *modal*, uma funcionalidade de "react-native-navigation", que permite que hajam multiplos *stacks* de navegação, um *stack* principal, onde estão os ecrãs de navegação base da aplicação, e um stack secundário onde está o stack primário juntamente com os ecrãs modais, neste caso apenas um. Tendo os stacks o seguinte aspeto:

```
const MainStack = createStackNavigator(
    {
        Home: HomeScreen,
        Details: DetailsScreen,
        SetUpCard: NewCardScreen,
        WaitForCard: WaitingScreen,
    },
    {
        initialRouteName: 'Home',
    }
);

const ModalStack = createStackNavigator(
    {
        Main: MainStack,
        QRScreen: PopUpQR,
    },
    {
        mode: 'modal',
        initialRouteName: 'Main',
    }
);

export default createAppContainer(ModalStack);
```

Armazenamento de dados

Para o armazenamento de dados foram utilizadas duas bibliotecas. Uma para os documentos em formato PDF, e uma para *metadata* dos documentos.

No caso da *metadata* recorreu-se a uma base de dados criada especificamente para React Native, e que é, segundo os seus criadores, até dez vezes mais rápida que as outras alternativas existentes. Esta base de dados é o "*Realm*"[10]. O "*Realm* utiliza uma estrutura de dicionário semelhante a JSON, o que facilita o armazenamento dos dados provenientes do emissor, que vêm em formato JSON. Para além disso o "Realm"possibilita a utilização de cifras para armazenar os dados utilizando Advanced Encryption Standard (AES), mais precisamente AES-256, e SHA-2 para possivel verificação com Hash-based Message Authentication Code (HMAC). Para a utilização de cifras, deve ser fornecida uma chave de cifra no momento de criação de um Realm. A utilização de Realms cifrados traz uma perda de performance de aproximadamente 10%, segundo a página do realm.

Para o armazenamento dos dados de cada cartão utilizou-se o seguinte esquema:

```
CardSchema = {
    name: 'Card',
    primaryKey: 'savedName',
    properties: {
        savedName: 'string',
        token: 'string',
        properties: 'string?[]',
        normalClaims: 'string?[]',
        specialClaims: 'string?[]',
        normalClaimSingatures: 'string?[]',
        specialClaimSigatures: 'string?[]',
        cardNumber: 'string',
        cardType: 'string',
        APIv: 'double',
        url: 'string',
        nextNeededUpdate: 'date'
    }
};
```

10 https://realm.io/docs/javascript/latest/

Destes dados, a chave primária é o nome dado pelo utilizador ao documento ao fazer o pedido ao servidor, as propriedades, as *claims* normais e especiais são usados para a escolha dos dados a apresentar. O *token*, número do cartão, tipo do cartão, versão da API e URL são usados para criar o código QR com o pedido para o emissor. A data do próximo *update* é usada para o utilizador saber quando precisa de atualizar o seu documento para o manter válido. E finalmente, as assinaturas dos *hashes* dos documentos são armazenados de forma a garantir a integridade dos documentos guardados.

No entanto o *Realm* não permite o armazenamento grandes ficheiros, e por isso recorreu-se a outro método de armazenamento para os documentos em formato PDF. A biblioteca utilizada foi a "rn-fetch-blob"[11], que permite acesso aos ficheiros privados da aplicação, criando ficheiros inacessíveis de fora da aplicação. Estes dados são também guardados cifrados com AES-256, neste caso utilizando a biblioteca "react-native-aes"[12].

Validação de documentos

Para a validação dos documentos recebidos foram utilizadas as mesmas bibliotecas utilizadas na aplicação do leitor, "react-native-sha256"e "react-native-RSA-native".

Quando é realizado um pedido ao servidor emissor, para além dos outros dados recebidos, o servidor envia ainda duas listas, uma com as assinaturas relativas às *claims* normais (que contém o cartão com todas as propriedades), e uma com as assinaturas relativas às *claims* especiais. Com estas assinaturas é possível validar a fonte dos documentos recebidos. Para isto a aplicação aplica a função de hash aos documentos recebidos e valida-os utilizando as respetivas assinaturas e a chave pública do servidor emissor, que está armazenada de raiz na aplicação.

4.3.4 *Funcionamento*

Após o desenvolvimento destes três elementos do sistema estarem completos, já é possível realizar os dois tipos principais de pedido ao emissor, utilizando a carteira para o pedido integral de um documento (para o armazenar na carteira), e utilizando o leitor para ler um QR apresentado pela carteira, de forma a realizar um pedido de consulta de um documento ao emissor.

11 https://www.npmjs.com/package/rn-fetch-blob/v/0.10.13
12 https://www.npmjs.com/package/react-native-aes

5

PROVA DE FUNCIONALIDADE

Terminado o desenvolvimento do protótipo do sistema de documentos de identificação desmaterializados, é necessária uma demonstração de que o sistema desenvolvido funciona. Com este objetivo fez-se uma filmagem da utilização das duas aplicações desenvolvidas, da qual foram retiradas imagens, que juntamente com capturas de ecrã das próprias aplicações apresenta o funcionamento do sistema.

Para além disto é importante demonstrar que este sistema confere um nível de segurança aceitável.

5.1 DEMONSTRAÇÃO

A demonstração de funcionalidade está dividida em dois, para os dois principais processos do sistema, o *setup* inicial de um documento na carteira, e a consulta de um documento pelo leitor. Neste caso utilizou-se como documento um cartão de cidadão português.

5.1.1 *Setup Inicial*

Para o *setup* inicial, o utilizador precisa de saber o URL base do servidor emissor, o identificador do documento desejado, o número de identificação do documento e um *token*, que lhe é fornecido pela entidade emissora.

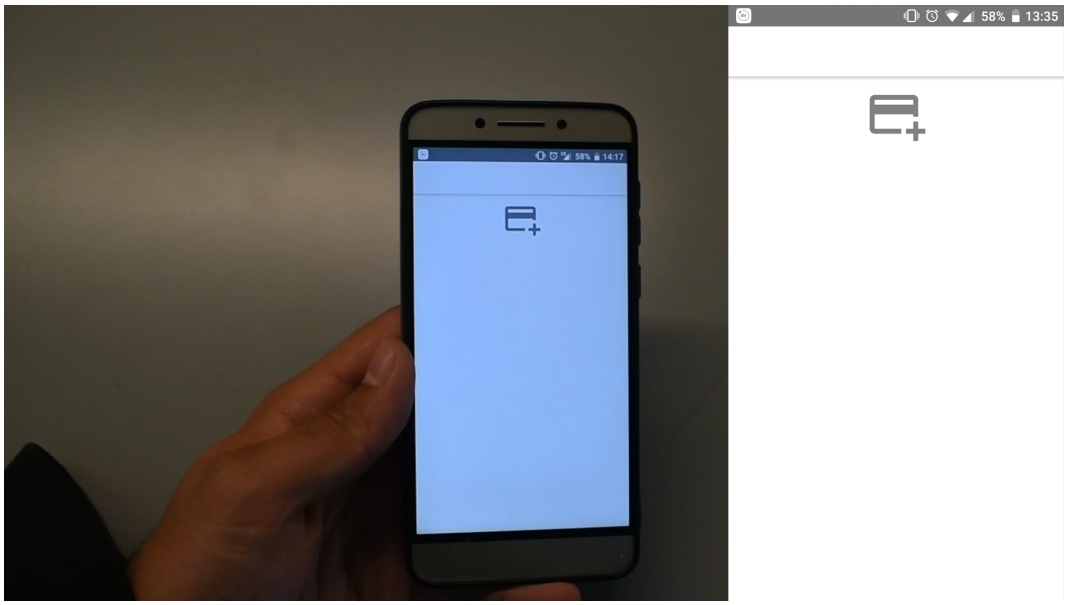


Figura 12: Menu inicial da carteira vazia

A primeira vez que a aplicação é aberta o utilizador deve autorizar o acesso ao armazenamento do dispositivo. Após autorizar este acesso é mostrado um ecrã com a carteira de eIDs, mostrada na figura 12, e que neste caso está vazia.

Para adicionar um novo documento o utilizador deve premir o botão para esse efeito, que o leva ao ecrã devido.

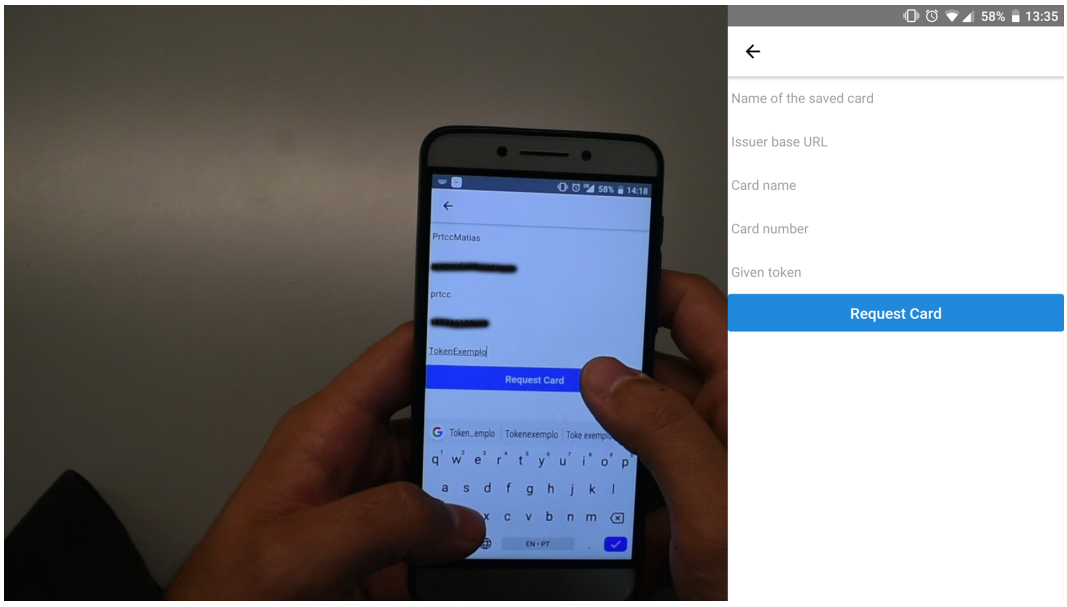


Figura 13: Inicio de *setup* de um cartão

No ecrã apresentado na figura 13, o utilizador vai inserir as informações referidas anteriormente, e que permitem criar um pedido de *setup* de um novo cartão ao servidor emissor. Para além destas informações, o utilizador deve definir um nome para o documento guardado.

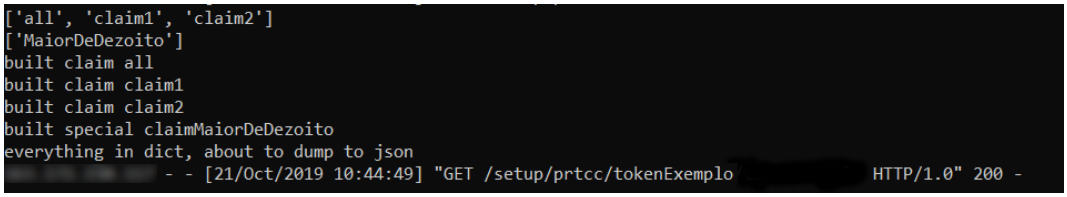


Figura 14: *Logs* do servidor emissor para o setup inicial de um documento

No lado do servidor, vai ser recebido o pedido, e vai proceder-se à geração dos documentos pedidos, criando *logs* para cada pdf gerado, apresentados na figura 14.

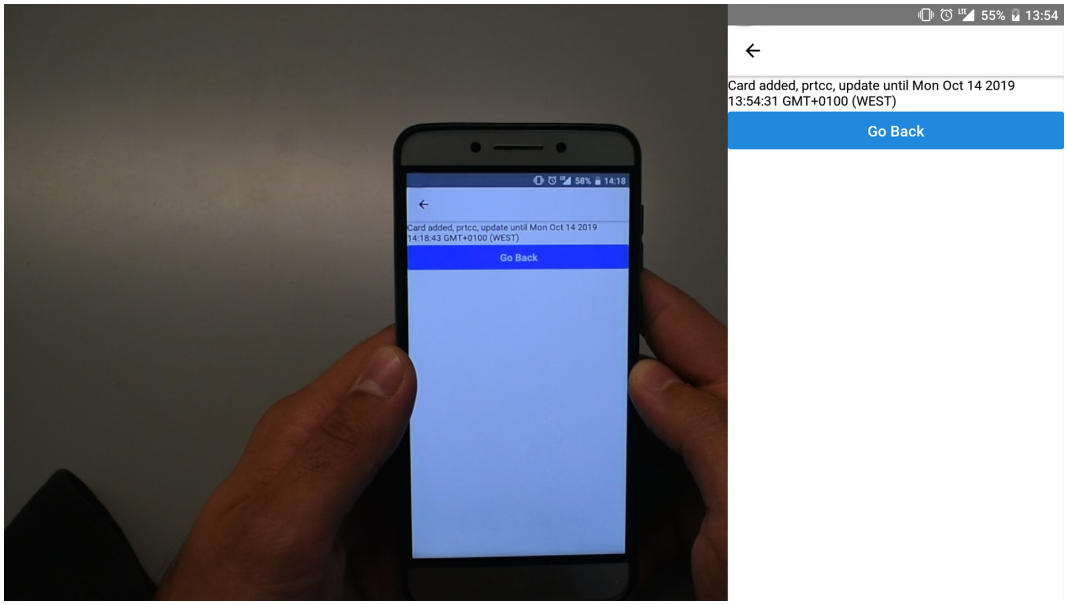


Figura 15: *Setup* de um cartão completo

Depois de preencher os campos com os dados necessários, o utilizador realiza o pedido e assim que o servidor termine de enviar todos os dados necessários, ou seja, o documento e as suas *claims*, assim como os dados sobre o documento, é apresentado o ecrã mostrado na figura 15, que confirma que o pedido teve sucesso.

5.1.2 *Consulta de documento*

Para que o utilizador da aplicação leitor possa consultar um documento no seu dispositivo, é necessário que o utilizador da aplicação da carteira tenha esse documento armazenado.

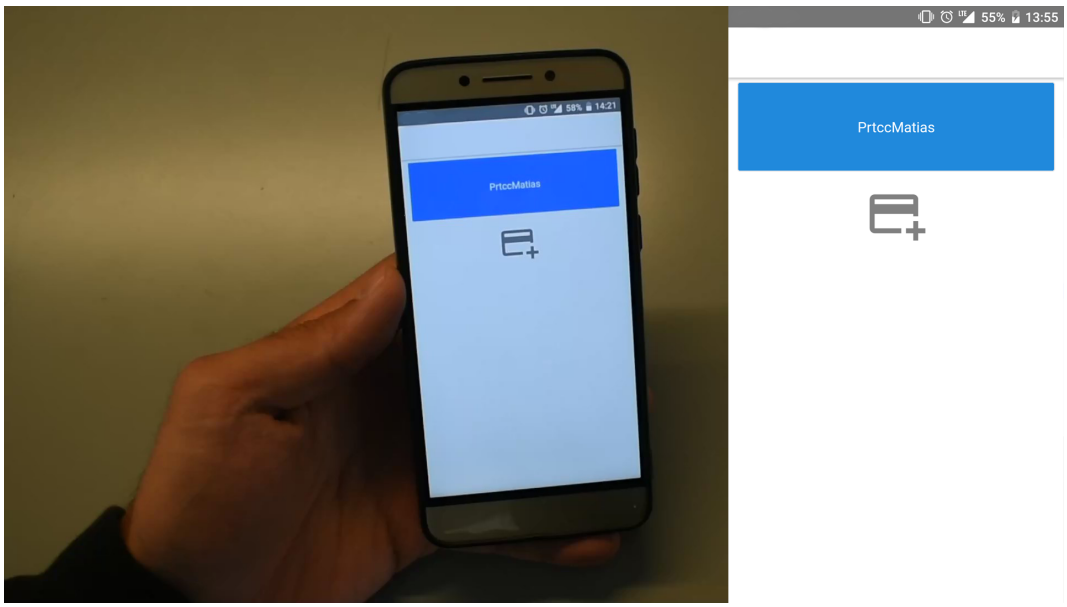


Figura 16: Menu inicial da carteira com um documento guardado

Tendo o documento armazenado no seu dispositivo, o utilizador da aplicação abre a aplicação e este documento deve aparecer no menu inicial, como é apresentado na figura 16.

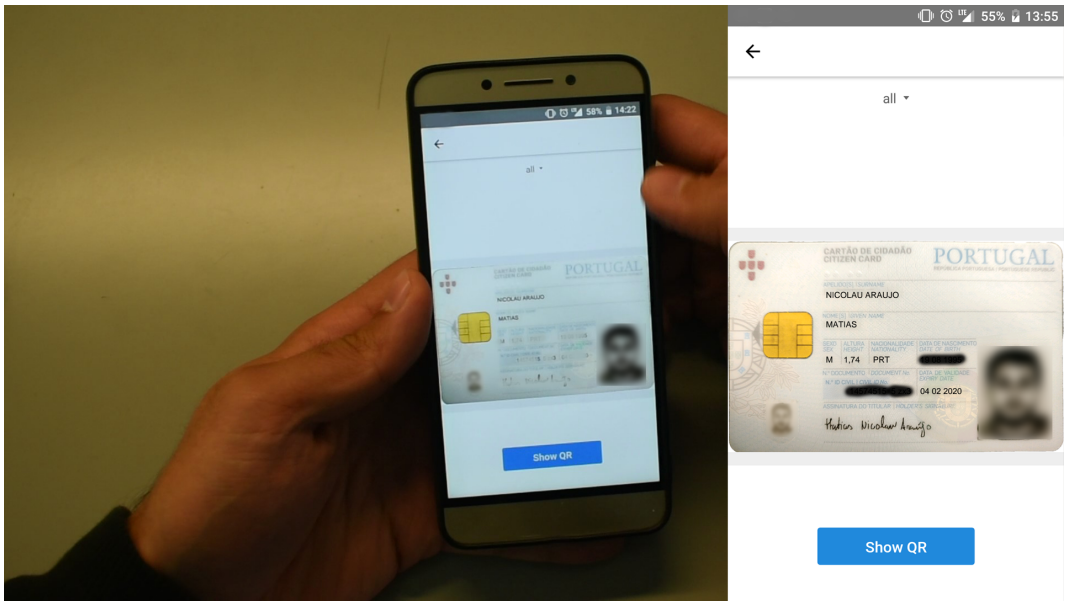


Figura 17: Documento completo guardado na carteira

Selecionado o cartão a ser consultado, o documento é apresentado no ecrã mostrado na figura 17, havendo um menu *dropdown* que permite escolher a *claim* a ser apresentada, ou ainda criar um documento personalizado para apresentar, sendo ainda possível carregar no botão "*show QR*"para apresentar no ecrã o QR a ser lido pelo leitor.

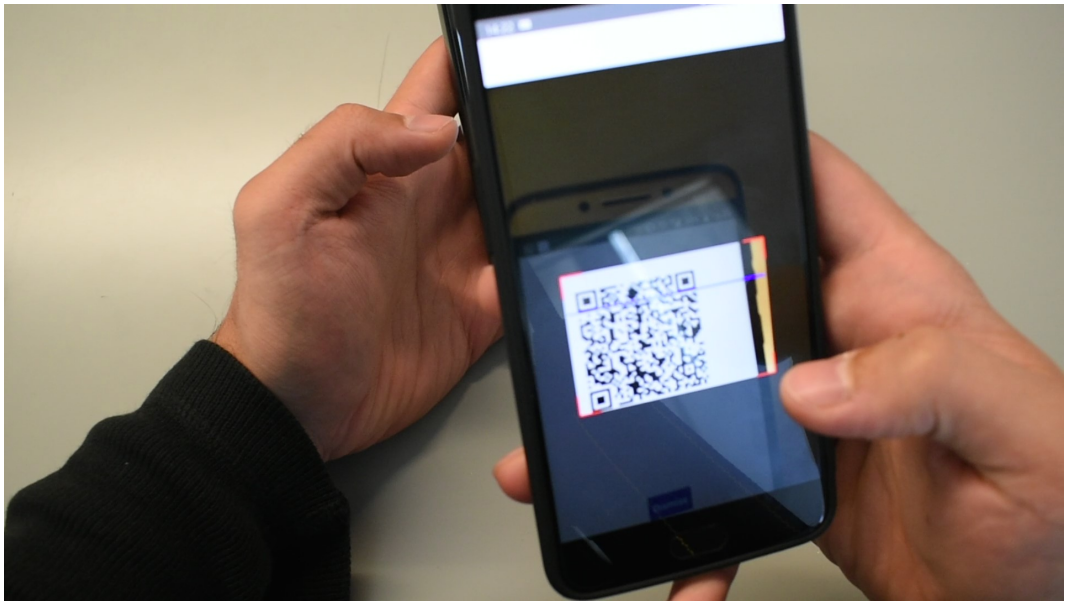


Figura 18: Leitor a ler o QR do documento completo, apresentado no ecrã da carteira

Estando este QR no ecrã, o utilizador da aplicação do leitor pode utilizar a câmara do seu dispositivo para ler o código, como demonstrado na figura 18.

Figura 19: *Log* de um pedido de consulta do lado do servidor

O servidor recebe o pedido apresentado na figura 19 e envia o documento pedido.

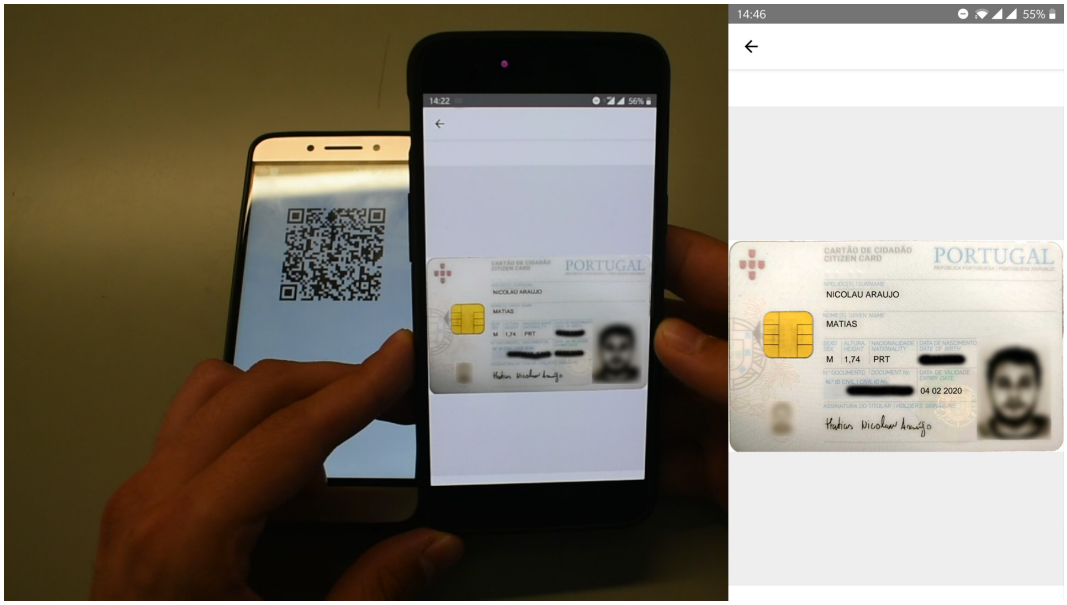


Figura 20: Documento completo, no ecrã do leitor

Assim que o servidor recebe este pedido, e enviado o documento para que o utilizador o possa consultar no seu ecrã, apresentado na figura 20.

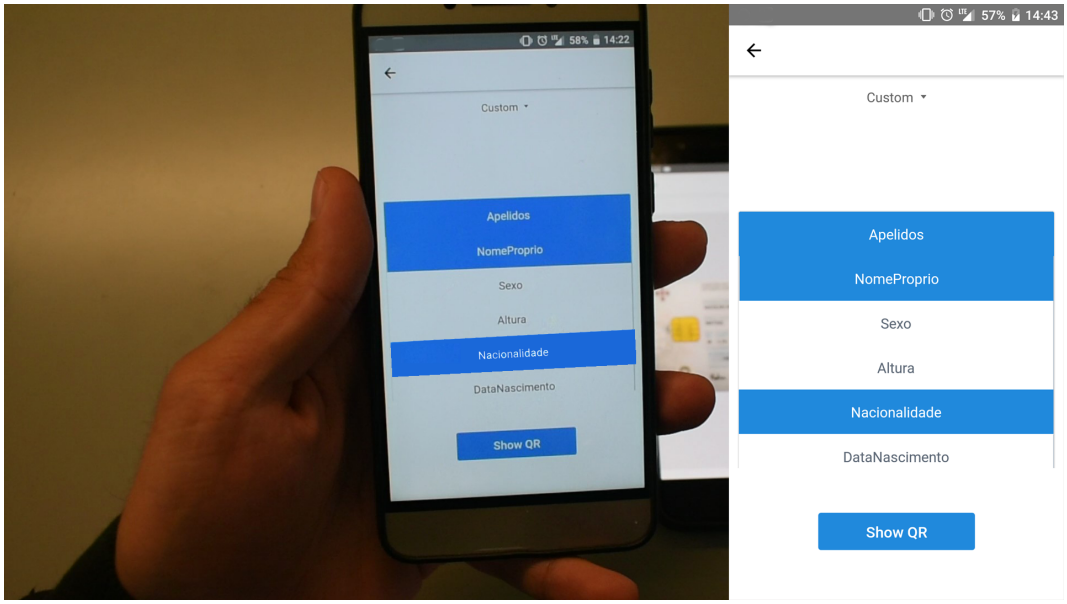


Figura 21: Escolha de propriedades para cartão

Caso o utilizador da carteira queira escolher as propriedades especificas para criar um cartão personalizado, deve escolhê-las no ecrã mostrado na figura 21, apresentando depois o QR para que o leitor possa realizar este pedido personalizado ao servidor emissor.

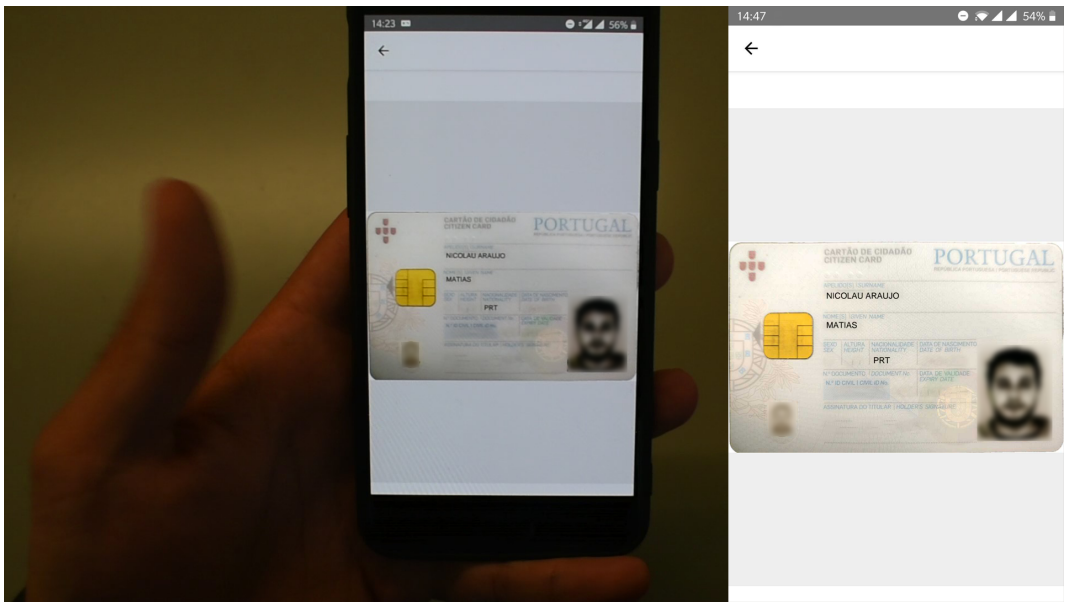


Figura 22: Documento com propriedades escolhidas, no ecrã do leitor

Feito este pedido, o leitor irá apresentar no ecrã um documento apenas com as propriedades pedidas, apresentado na figura 22. Neste caso, o nome próprio, os apelidos, a nacionalidade e a fotografia.

Na figura 23 está apresentado um exemplo de uma *claim* especial, neste caso uma prova de maioridade.

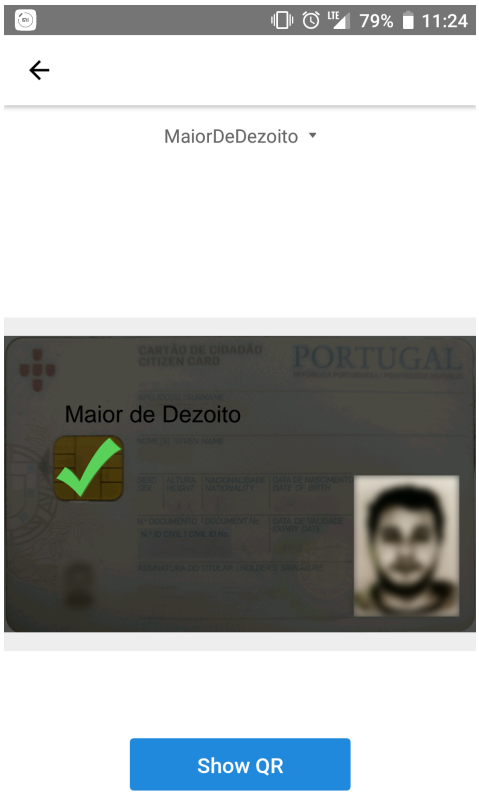


Figura 23: *Claim* especial de um cartão de cidadão português, para provar a maioridade

5.2 EXPLICAÇÃO DE ELEMENTOS DE SEGURANÇA

Neste trabalho, quer durante modulação quer durante o desenvolvimento do protótipo, a segurança da informação e a fiabilidade dos documentos foi sempre uma prioridade. Visto que se está a lidar com informação sensível e privada, todos os dados guardados quer no servidor quer nos dispositivos têm que estar seguras, de modo a oferecer confiança aos utilizadores. Para além disto, ao consultar um documento, o leitor tem de ter confiança de que o documento é fidedigno. Para que o sistema de documentos de identificação desmaterializados seja viável todas estas características têm de ser providenciadas pelo sistema.

Neste caso, todos os dados guardados estão cifrados quer no servidor quer na aplicação da carteira, e no caso do leitor não estão guardados nenhuns dados dos documentos consultados.

Em relação à confiança dos documentos recebidos no leitor, visto que estes vêm diretamente do emissor (os documentos enviados pelo servidor emissor são enviados em conjunto com uma assinatura, que permite validar se estes documentos provêm do emissor)

é possível confiar na informação recebida, estando o certificado do servidor emissor guardado tanto na aplicação do leitor, como na aplicação da carteira, permitindo a validação da informação recebida.

Este sistema está portanto de acordo com o Regulamento Geral sobre a Proteção de Dados (RGPD) no que toca ao armazenamento de dados, não sendo possível o acesso aos mesmos sem a autorização prévia dos utilizadores.

Ainda em relação ao RGPD, a existência das claims e da possibilidade de escolher as propriedades a apresentar na consulta dos documentos, permitem ainda a diminuição dos dados fornecidos a quem queira consultar um documento, oferecendo ao utilizador controlo sobre os seus próprios dados.

6

CONCLUSÃO E TRABALHOS FUTUROS

Terminado o desenvolvimento do sistema e demonstração do projeto desenvolvido, neste capitulo final encontram-se as conclusões do trabalho desenvolvido, e um conjunto de prespetivas futuras possíveis.

6.1 CONCLUSÕES

Este trabalho teve como propósito conceber um protótipo de um sistema de documentos de identificação desmaterializados (documentos de identificação oficiais em formato digital), que fosse fácil de utilizar e garantisse a segurança dos dados. Além disso pretendia-se que o distema de documentos de identificação dematerializados oferecesse ao utilizador funcionalidades e garantias semelhantes(ou ainda melhores), que o sistema utilizado atualmente, baseado em documentos de identificação físicos. Constitui também um dos principais objetivos deste trabalho, o desenvolvimento de um protótipo completo do sistema concebido de forma a ser possível demonstrar que um sistema deste tipo é funcional, seguro e viável.

Com este objetivo, numa fase inicial começou por se procurar casos de sistemas semelhantes com o sistema que se pretendia criar. O tema apesar de ser já discutido, pareceu estar pouco desenvolvido, tendo sido encontrado apenas um único projecto internacional concreto focado na matéria, específico para cartas de condução (Mobile Driver's License (mDL)). A mDL começou por ser uma iniciativa da American Association of Motor Vehicle Administrators (AAMVA), que através de um *withepaper* identificou alguns requisitos para um sistema de cartas de condução Norte Americanas. Esta iniciativa foi depois adotada pela International Organization for Standardization (ISO), que começou em 2018 o desenvolvimento de uma norma para as mDLs, a ISO18013-5, referida em [2]. Esta norma aprofunda muito mais os aspetos técnicos e os requisitos de um sistema de mDLs, em relação ao *whitepaper* desenvolvido pela AAMVA. No entanto, à data da entrega desta dissertação, a norma referida encontra-se ainda em desenvolvimento, pelo que o acesso às versões atualizadas foi dificultado. Estas dificuldades foram ultrapassadas através da reativação da

comissão técnica ”CT110”, que tem acesso a estes documentos da ISO, esta reativação da comissão técnica foi demorada dada toda a burocracia associada ao processo, pelo que o acesso aos documentos atualizados foi bastante tardio.

A norma ISO 18013-5 foi estudada aprofundadamente, selecionando-se ao longo da leitura e análise da norma, os pontos relevantes para o projeto a ser desenvolvido, descartando alguns dos requisitos, principalmente aqueles que provinham de documentos antigos, e que não eram benéficos à transformação destes documentos em formato desmaterializado. Como por exemplo a utilização de *datagroups* para o armazenamento dos dados dos documentos (explicados no capítulo 2 deste documento), que não eram optimizáveis em formato digital, e que foram descartados em troca de dados em formato JSON. Estes *datagroups* viriam mais tarde a ser alterados para o formato JSON pela própria ISO, na versão mais recente da norma, que saiu já durante o desenvolvimento deste projeto, em 2019.

Depois de estudada a norma passou-se à modulação do sistema a ser desenvolvido, utilizando como base uma estrutura parecida com a utilizada no caso de estudo, mas com o objetivo de suportar a utilização para qualquer documento, e não só a carta de condução. Este sistema foi dividido em três, sendo uma das partes um servidor pertencente à entidade emissora, outra parte seria uma carteira de documentos pertencente a um dado titular, e que poderia guardar todos os documentos desse titular, e finalmente a estrutura do leitor, que foi por sua vez dividida em dois, o próprio leitor, que seria o dispositivo utilizado para consulta dos documentos armazenados na carteira, e o servidor verificador, que iria ser utilizado para a comunicação externa do leitor.

Estando definida a estrutura base do sistema passou-se à especificação das interações com o sistema, identificando-se duas principais, uma configuração inicial de um cartão por parte da carteira, e o pedido de consulta de um documento por parte do leitor. Definindo-se ainda que o pedido de consulta por parte do leitor deveria ser o principal, pois seria a interação mais frequente do sistema.

Terminada a modulação do sistema passou-se à escolha das tecnologias utilizadas para cada componente do sistema. Esta escolha foi feita em função daquilo que pareceu, na altura, ser mais benéfico em termos de tempo de desenvolvimento, dada a limitação clara de tempo para o desenvolvimento de um sistema com quatro elementos diferentes. Com este objetivo optou-se pela ferramenta multiplataforma ”React-Native”para o desenvolvimento das duas aplicações móveis (o leitor e a carteira), que permitiria poupar o tempo de desenvolvimento separado para as duas principais plataformas móveis (android e ios). Para os dois servidores (o emissor e o verificador) optou-se por um *front-end* em *python*, com recurso a *flask* como *framework web*, e por um *back-end* em *mongodb* para o armazenamento de dados.

Passando ao desenvolvimento dos vários elementos do sistema foi necessário definir a ordem de prioridade dos elementos a desenvolver. Começando neste caso pelo servidor

emissor, passando à aplicação do leitor, seguido pela aplicação da carteira e terminando pelo servidor verificador.

O desenvolvimento do servidor emissor correu conforme o planeado, visto não ser necessário aprender *python* nem *mongodb* com as quais já havia experiência prévia, sendo maior parte do tempo usado para construir a estrutura de dados e criar o gerador de documentos a partir dos dados.

O desenvolvimento das duas aplicações móveis constituiu o maior desafio, dada a falta de experiência com este tipo de ferramentas, e também pelo facto da *framework* escolhida, o Reack-Native estar ainda em desenvolvimento, o que criou vários impasses durante o desenvolvimento de ambas as aplicações. Foi utilizado algum tempo para aprendizagem do funcionamento da *framework*, assim como para a ambientação com as várias ferramentas da mesma. Dado o desenvolvimento constante da própria *framework* durante o trabalho realizado, muito do tempo perdido derivou de atualizações às bibliotecas utilizadas, ou até mesmo de atualizações ao proprio React-Native, estas atualizações obrigaram em algumas situações à criação de uma versão própria das bibliotecas utilizadas, para assegurar a sua compatibilidade com as versões correntes da *framework*, o que dada a extensão do código de uma aplicação em react-native, atrasou consideravelmente o desenvolvimento do código. Em algumas situações foi também necessário esperar que as equipas por trás das várias bibliotecas utilizadas fizessem as correções necessárias para o funcionamento correto das mesmas.

Tendo em conta os vários atrasos no desenvolvimento das aplicações, foi tomada a decisão de criar o protótipo do sistema sem a existência do servidor verificador. Esta decisão foi tomada visto que um protótipo sem este elemento permite na mesma uma demonstração de funcionalidade do sistema como um todo, sendo o verificador um elemento pensado como uma camada extra de segurança e confiança. Isto significa que o sistema desenvolvido é constituído por apenas três dos quatro elementos modulados inicialmente. Para além disto, algumas das limitações das bibliotecas utilizadas implicam que algumas das funcionalidades funcionem apenas em *android*.

Apesar da ausência deste elemento, com o sistema desenvolvido foi possível mostrar que um sistema deste tipo funciona e que é de fácil utilização. Sendo possível que num futuro não muito longínquo um sistema de documentos de identificação desmaterializados seja usado oficialmente, em vez de cartões físicos com os quais estamos acostumados, podendo esse futuro sistema ser baseado no projeto apresentado neste documento.

6.2 TRABALHOS FUTUROS

Depois de terminado o protótipo demonstrado, surge como trabalho futuro mais evidente, o desenvolvimento do servidor verificador, que ficou fora do protótipo.

Para além disto, e de modo a uma melhor escalabilidade, é possível que a solução utilizada para a API do servidor emissor não seja a mais adequada, sendo que em ambiente de produção, uma solução mais escalável como o *Django* ou o *Ruby on Rails* fossem mais apropriadas.

Finalmente, e com o propósito de cumprir os requisitos especificados na versão mais recente da norma ISO18013-5, ter-se-ia de mudar a estrutura de dados e os métodos de comunicação de acordo com a mesma.

BIBLIOGRAFIA

[1] Technical Committee : ISO/IEC JTC 1/SC 17. Iso/iec cd 18013-2:2008 – nformation technology – personal identification – iso-compliant driving licence – part 2: Machine-readable technologies, 2008.

[2] Technical Committee : ISO/IEC JTC 1/SC 17. Iso/iec cd 18013-5 – information technology – personal identification – iso-compliant driving licence – part 5: Mobile driving licence application (mdl), 2018.

[3] Technical Committee : ISO/IEC JTC 1/SC 17. Iso/iec cd 18013-5:2019 – information technology – personal identification – iso-compliant driving licence – part 5: Mobile driving licence application (mdl), 2019.

[4] Oscar Axelsson and Fredrik Carlström. Evaluation targeting react native in comparison to native mobile development, 2016.

[5] Parlamento Europeu e o Conselho da União Europeia. Regulamento geral sobre a proteção de dados, 2018.

[6] European Union Agency for Cybersecurity. Algorithms, key size and parameters report 2014, 2014.

[7] Internet Engineering Task Force. oncise binary object representation (cbor), 2013.

[8] Secunet Security Networks. Elliptic curve cryptography (ecc) brainpool standard curves and curve generation, 2010.

[9] American Association of Motor Vehicle Administrators. Mobile driver's license functional needs white paper, r0.8, 2017.

[10] National Institute of Standards and Technology. Elliptic curve cryptography subject public key information, 2009.

[11] International Civil Aviation Organization. Doc 9303 - machine readable travel documents, 2015.