



Universidade do Minho
Escola de Engenharia

Engenharia de Segurança

Projeto de Desenvolvimento 2

Grupo 12

Bruno Filipe de Sousa Dias PG47068
Guilherme da Silva Amorim Martins PG47225
Luís Enes Sousa A89597



20 de junho de 2022

Conteúdo

1	Introdução	1
1.1	Estrutura do Relatório	1
1.2	Objetivos	1
2	Solução Desenvolvida	2
2.1	Ligação à Base de Dados	2
2.2	Login e Registo de Utilizador	3
2.2.1	Classe de Utilizador	3
2.2.2	Encriptação da Password	4
2.2.3	Novas Views	4
2.2.4	Novos Controllers	4
2.2.5	Processo de Registo	6
2.2.6	Processo de Login	6
2.3	Área de Utilizador	7
2.4	Utilização do número de telemóvel guardado na Área de Utilizador para Operações com Chave Móvel Digital	7
2.5	Funcionalidade Extra	8
3	Técnicas de Desenvolvimento de Software Seguro	9
4	Utilização e Teste da Aplicação	10
4.1	Requisitos para a Instalação	10
4.2	Processo de Instalação Windows	10
4.3	Processo de Instalação Linux	10
4.4	Demonstração	10
4.4.1	Login	11
4.4.2	Register	12
4.5	Perfil	13
4.6	Logout	14
4.7	Assinatura e Validação de um Documento	15
5	Conclusão	17

Lista de Figuras

1	Tabela de Utilizador Base de Dados	2
2	Página de Login com pedido de argumentos	11
3	Página de Login com erro de Validade	11
4	Página de Login de sucesso	11
5	Página de Register com pedido de argumentos	12
6	Página de Register com erro de Validade	12
7	Página de Registo de sucesso	12
8	Página principal com secção de Perfil	13
9	Perfil de um Utilizador sem número de telemóvel associado	13
10	Preenchimento de um novo número de telemóvel	13
11	página de confirmação de alteração do número de telemóvel	14
12	Perfil de um Utilizador com número de telemóvel associado	14
13	Perfil de um Utilizador com número de telemóvel associado	14
14	Assinatura de um Documento	15
15	Validação de assinatura de um documento	16

1 Introdução

No âmbito da Unidade Curricular Engenharia de Segurança foi-nos proposto expandir um projeto desenvolvido por colegas no ano anterior, que permitia a assinatura de documentos digitais, acrescentando a capacidade de autenticação, permitindo ao utilizador guardar dados como o número de telemóvel, na sua conta. Também foi referida a necessidade de adaptar a aplicação para uma nova versão da DSS Demo WebApp. Adicionalmente, tivemos de adicionar uma interface de ajuda para o utilizador.

1.1 Estrutura do Relatório

No presente documento iremos demonstrar e ilustrar quais as decisões tomadas para conseguirmos alcançar a implementação das novas funcionalidades pedidas pelo Docente ou ainda funcionalidades extra. Assim, este relatório foi subdividido em vários capítulos com diferentes tópicos cada um.

- **Primeiro Capítulo** - Contextualização do problema e dos seus objetivos;
- **Segundo Capítulo** - Funcionalidades implementadas com os diferentes passos e métodos utilizados;
- **Terceiro Capítulo** - Técnicas de Desenvolvimento de Software Seguro utilizadas ao longo do projeto;
- **Quarto Capítulo** - Requisitos necessários, instalação da aplicação e demonstração e teste da mesma;
- **Quinto Capítulo** - Análise global do projeto desenvolvido;

1.2 Objetivos

Este projeto tem como objetivo a reestruturação de uma aplicação, adicionando novas funcionalidades à mesma e/ou alterando o comportamento de outras. Desta forma, temos como ponto de partida código utilizado e produzido por colegas de anos anteriores, e a partir do mesmo iremos ter de cumprir alguns requisitos e funcionalidades pedidas pelo Docente, sendo estas:

- Transposição das alterações que os colegas do ano passado tinham efetuado, para a nova versão da DSS Demo WebApp (versão 5.10.1 ~~ou superior~~);
- Adição de interface de autenticação inicial (com utilizador e password);
- Adição de área de utilizador, onde o utilizador (após autenticação) possa definir qual o número de telemóvel que utiliza para a Chave Móvel Digital - sendo os dados do utilizador guardados em Base de Dados;
- Alteração do código efetuado pelos colegas do ano passado, de modo que seja utilizado o número de telemóvel guardado na área de utilizador, sempre que o utilizador efetue uma operação que utilize a Chave Móvel Digital.
- Funcionalidade Adicional (Específica do grupo 12) - Adição à interface existente, uma nova componente de "Ajuda", que disponibilizará texto de ajuda para a utilização das várias opções na componente *e-Signature*, nomeadamente para *Sign a document*, *Sign a digest*, *Sign a PDF*, *Sign with JAdES*, *Sign multiple documents*, *Counter sign a signature*;

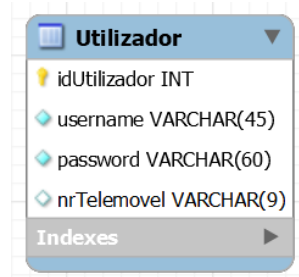
2 Solução Desenvolvida

No presente Capítulo iremos apresentar a forma como a solução foi desenvolvida, bem como os passos que foram feitos para alcançar a solução final. Desta fora, iremos seguir os mesmos moldes fornecidos pelo Docente, começando por explicar a implementação da base de dados, seguidamente o processo de Login e Registo na aplicação, implementação da Área de Utilizador e ainda a utilização do número de telemóvel guardado na área de Utilizador para efetuar operações com Chave Móvel Digital. Por fim, iremos ainda explicar a maneira como foi alcançada a implementação da funcionalidade adicional.

2.1 Ligação à Base de Dados

Inicialmente, e como podemos perceber pelo enunciado, nós teremos de possuir uma base de dados de forma a poder guardar informações sobre cada Utilizador. Nestas informações incluem-se `username`, `password` e o número de telemóvel associado à conta. Para a construção de uma base de dados recorremos a MySQL, devido a ser uma base de dados do tipo relacional (que na eventualidade de tornar a aplicação maior, é mais escalável horizontalmente) e bem conhecida publicamente, possuindo uma elevada comunidade de suporte.

Assim, construímos a seguinte tabela relativa aos Utilizadores:



Utilizador	
idUtilizador	INT
username	VARCHAR(45)
password	VARCHAR(60)
nrTelemovel	VARCHAR(9)
Indexes	

Figura 1: Tabela de Utilizador Base de Dados

Para poder implementar a conexão da Base de Dados na nossa aplicação, e por se tratar de um Projeto Maven, tivemos de adicionar uma *dependency* necessária para o devido funcionamento desta conexão. Assim, a *dependency* adicionada foi:

```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.29</version>
</dependency>
```

Uma vez adicionada a *dependency*, podemos finalmente começar a efetuar a ligação entre a nossa aplicação e a base de dados. Esta conexão está implementada no ficheiro `WebAppUtils` que vai efetuar a ligação, ou seja, a conexão entre a aplicação e a base de dados, que no nosso caso se encontra no *localhost*. Futuramente, esta base de dados poderia ser colocada num servidor externo e apenas teríamos de alterar as informações de *localhost* para o novo IP, e as informações do *user* de acesso à base de Dados.

```
public static Connection getConnectionToDB() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");

        Connection conn = DriverManager.getConnection (
            "jdbc:mysql://localhost:3306/grupo12pd2",
            "root",
            "root"
        );
        return conn;
    }
    catch (Exception e) {
        return null;
    }
}
```

2.2 Login e Registo de Utilizador

De forma a seguir uma ordem lógica do funcionamento, iremos inicialmente explicar todos os processos e passos feitos para poder efetuar um Login e Registo com Sucesso. Nestas secções incluem-se os processos de conexão à Base de Dados explicados no capítulo anterior.

2.2.1 Classe de Utilizador

De forma a poder efetuar operações sobre o Utilizador, tanto a nível de Login e Registo, bem como a nível da Área de Utilizador e da utilização automática de um número de telemóvel guardado em operações com Chave Móvel Digital. Nesta classe iremos possuir informações sobre o Utilizador, que se irão coordenar com as informações da Base de Dados. Desta forma possuímos o ID do Utilizador, o *username*, a *password* e ainda o número de telemóvel associado ao mesmo.

De forma a utilizar estas informações ao longo do projeto foram ainda produzidos construtores vazios e com todos os argumentos da Classe, *getters* e *setters* e ainda todas as funções que serão responsáveis por pesquisar informações na Base de Dados e retornar respostas ou valores que serão utilizados ao longo do projeto e da aplicação. Nestas funções encontramos as seguintes *getUtilizadorByUsername*, *getUtilizadorByUsernameAndPassword*, *addUtilizadorToDB* e ainda *setCellphoneOfUser*.

2.2.2 Encriptação da Password

Apesar de não ser um requisito neste projeto, a equipa achou por bem que as informações relativas à *password* de um Utilizador não deveriam ser guardadas como *plaintext* na Base de Dados e por isso, decidimos implementar mecanismos de encriptação. Para isso utilizamos a biblioteca *bcrypt* que nos forneceu métodos para encriptação e verificação de passwords encriptadas. Estes métodos foram utilizados nos métodos implementados no Utilizador para conexão com a Base de Dados.

Assim, quando um Utilizador se regista na nossa aplicação, iremos utilizar esta biblioteca e a sua *password* é cifrada, através do seu *plaintext* bem como de um *salt* gerado aleatoriamente. No processo de login, iremos utilizar novamente esta biblioteca e iremos comparar a password introduzida e verificar se a mesma equivala à password encriptada na base de dados.

```
<dependency>
  <groupId>org.mindrot</groupId>
  <artifactId>jbcrypt</artifactId>
  <version>0.3m</version>
</dependency>
```

2.2.3 Novas Views

Este processo de implementar novas funcionalidades faz com que, como será de esperar, tenham de ser introduzidas novas *Views* no nosso projeto. Estas views serão responsáveis tanto por mostrar informações necessárias ao Utilizador, como por apresentarem diferentes forms e opções de navegação do *website*. Para além disso, são também nestas views efetuadas operações em *Javascript* que dizem respeito ao *localStorage*, onde serão guardadas localmente informações sobre cada Utilizador. Assim, criamos algumas novas views:

- **login.html** - Contém o *form* que será preenchido pelo Utilizador para efetuar o Login;
- **login_success.html** - Informa que Utilizador se encontra com sessão iniciada e se pode ligar à nossa aplicação;
- **logout_success.html** - Informa que Utilizador se encontra com sessão terminada e se pode desligar da nossa aplicação e fazer possivelmente um novo login;
- **register_form.html** - Contém o *form* que será preenchido pelo Utilizador para efetuar o Registo;
- **register_success.html** - Informa que Utilizador se encontra registado e pode aceder à pagina de Login para iniciar a sua sessão;
- **perfil.html** - Mostra informações sobre o Utilizador e possibilita que o mesmo termine sessão ou atualize o seu número de telemóvel;
- **addPhoneSuccess.html** - Informa que Utilizador alterou com sucesso o seu número de telemóvel e pode voltar ao seu perfil;

2.2.4 Novos Controllers

Tal como a introdução de novas Views, também tiveram de ser introduzidos novos Controllers para a nossa aplicação funcionar da forma correta. Foram assim acrescentados 3 Controllers: LoginController, RegisterController e ainda o PerfilController. Estes como os nomes indicam irão controlar operação a nível do Login, Register e do Perfil do utilizador.

Para além da adição destes Controllers foi ainda feita uma alteração no HomeController de modo à pagina home deixar de ser a página inicial e passar a ser a página login (tal como pedido no enunciado).

Desta forma, os Controllers seguem uma estrutura semelhante aos já construídos na aplicação. Seguidamente poderemos ver o exemplo do **LoginController**:

```
/**
 * Login controller
 */
@Controller
public class LoginController {
    /**
     *
     * @param model
     *         The model attributes
     * @return The view name.
     */
    @RequestMapping(value = { "/" , "/login" })
    public final String showLogin(final Model model)
    {
        model.addAttribute("user", new Utilizador());
        model.addAttribute("warning", "");

        return "login";
    }

    @RequestMapping(value = { "/process_login" })
    public final String processLogin(Utilizador user, final Model model) {

        Connection conn = WebAppUtils.getConnectionToDB();

        Utilizador novoUser = Utilizador.getUtilizadorByUsernameAndPassword(
            conn, user.getUsername(), user.getPassword());

        if (novoUser == null) {

            model.addAttribute("user", new Utilizador());
            model.addAttribute("warning", "Username ou Password errados");

            return "login";
        }
        else {

            model.addAttribute("usernameLogado", novoUser.getUsername());
            model.addAttribute("telemovelLogado", novoUser.getTelemovel());

            return "login_success";
        }
    }
}
```


2.2.5 Processo de Registo

Inicialmente, e tendo como ponto de referência uma base de dados vazia, teremos de introduzir e registar o nosso primeiro Utilizador. Para efetuar esta operação, é apresentado ao Utilizador uma página de Registo com um *form* onde este poderá preencher os seus dados de registo. Na view deste form, é especificado que tanto o input de username, como o input de password devem ser obrigatoriamente preenchidos para efetuar o *submit* do *form*. Ao submeter o form, iremos então processar os dados no RegisterController.

Inicialmente, no RegisterController, recorrendo à classe Utilizador que possui as conexões e operações de sincronização com a base de dados, iremos verificar se o Utilizador criado já se encontra registado na base de dados e se esse for o caso, iremos introduzir um aviso que informa o Utilizador desse acontecimento e iremos retornar novamente a página de form de registo ao utilizador.

No caso de o Utilizador ainda não se encontrar registado na base de Dados, iremos então adicionar o utilizador à base de dados (não esquecendo que passamos pelo processo de encriptação da password antes de introduzir a mesma na base de dados). Seguidamente é apresentada uma página que confirma o sucesso do Registo do Utilizador e permite que este volte à página de Login.

2.2.6 Processo de Login

No processo de Login apresentamos ao utilizador uma página que possui um *form* ao qual o mesmo pode dar submit e dessa forma iniciar sessão. Mais uma vez, estes campos são referidos na view correspondente como parâmetros obrigatórios e o submit apenas poderá ser realizado quando todos os campos tiverem sido introduzidos. Submetendo este form iremos então processar estes dados no LoginController.

Inicialmente, no LoginController, recorrendo à classe Utilizador que possui as conexões e operações de sincronização com a base de dados, iremos verificar se o Utilizador criado se encontra registado na base de dados e se a password introduzida é a que se encontra associada a este Utilizador na base de dados. Se não existir este Utilizador na base de dados, ou se a password introduzida não for a correspondente, então iremos introduzir um aviso que informa o Utilizador desse acontecimento e iremos retornar novamente a página de form de login ao utilizador.

No caso de Utilizador se encontrar registado na base de dados, e a *password* introduzida for a correspondente na mesma, então iremos iniciar sessão redirecionando o Utilizador para uma página que lhe indica que iniciou a sessão com sucesso e que pode ir para a página principal da aplicação. Um ponto importante é que, neste momento onde o Utilizador entra na view de confirmação de *login*, as suas informações de *username* e número de telemóvel são guardadas no *localStorage*. Este ponto é importante para os passos que se seguem da área de perfil e da funcionalidade de *auto-fill* do número de telemóvel em operações com Chave Móvel Digital.

2.3 Área de Utilizador

De forma a introduzir esta funcionalidade tivemos de alterar algumas informações já presentes no projeto, mais especificamente na view do menu. Assim, acrescentamos neste menu uma nova secção que nos permite navegar para o perfil do Utilizador. Para isso introduzimos o seguinte código:

```
<div class="card">
  <div class="card-header bg-primary">Profile</div>
  <div class="list-group list-group-flush">
    <a th:href="@{/perfil}" class="list-group-item">My Profile</a>
  </div>
</div>
```

Assim poderemos navegar até à página de Perfil. Uma vez nesta página de Perfil poderemos encontrar informações relativas ao Utilizador com a sessão iniciada. Estas informações apresentadas na view, são obtidas através do *localStorage* referido anteriormente no *processo de Login*. Assim, será apresentado o *username* do Utilizador e ainda o número de telemóvel associado ao Utilizador (caso este o possua). Para além disso é apresentada um pequeno *form* onde o Utilizador poderá introduzir um novo número de telemóvel e associar o mesmo à sua conta. Este processo de alteração de número de telemóvel funciona tanto para Utilizadores que já possuam um número de telemóvel, como para aqueles que ainda não possuam nenhum número associado. Ao realizar o *submit* deste *form*, o Utilizador seria apresentado com uma página de confirmação de alteração do número de telemóvel e poderia então voltar ao seu perfil. Nesta view de confirmação de alteração do número seria, à semelhança do sucedido na página de confirmação de login, armazenado o novo número de telemóvel do Utilizador no *localStorage*.

2.4 Utilização do número de telemóvel guardado na Área de Utilizador para Operações com Chave Móvel Digital

De forma a implementar esta funcionalidade tivemos de alterar o código dos alunos do ano passado. A sua boa construção de código permitiu que o nosso grupo alterasse facilmente o mesmo de forma a conseguir alcançar este objetivo. Para isso alteramos essencialmente o ficheiro *jsSaveCmdUserId*. Aqui iríamos recorrer ao número guardado no *localStorage* para introduzir automaticamente este número em operações com uso de Chave Móvel Digital.

```
function onDocumentReady() {
  // Variables corresponding to form inputs
  const $userId = $("#userId");

  // Variables corresponding to localStorage saved preferences
  const savedCellphone = localStorage.getItem("telemovelLogado");

  // If the user has previously chosen to store his userId
  if(savedCellphone === 'null') {
    // Restore user id from saved user id
    $userId.val("+351 ");
  } else {
    // If the user hasn't chosen to store his user id, or if he hasn't
```

```

        // been on this page before, then set a default value for the user id
        $userId.val("+351 " + savedCellphone);
    }
}

$(document).ready(onDocumentReady);

```

Desta forma, transformamos o "Remember Me" dos alunos do ano passado numa componente de *auto-fill* do número de telemóvel associado a um Utilizador com sessão iniciada, quando é efetuada uma operação que recorre à Chave Movel Digital.

2.5 Funcionalidade Extra

Como funcionalidade extra, o nosso grupo implementou uma das Heurísticas de Nielsen que diz respeito à Ajuda e Documentação da nossa aplicação. Assim, construímos uma componente de "Ajuda" nas várias opções na componente *e-Signature*, nomeadamente para *Sign a document*, *Sign a digest*, *Sign a PDF*, *Sign with JAdES*, *Sign multiple documents* e ainda *Counter sign a signature*.

Assim, começamos inicialmente por construir mensagens de ajuda no ficheiro *application.properties*, com o seguinte molde:

```
label.signature.jws.documentToBeSigned = The file that is going to be signed
```

Posteriormente estas informações foram introduzidas num botão de hover nas várias secções pedidas no enunciado. Assim alteramos as views das mesmas introduzindo nos campos necessários o código HTML que se equipara ao seguinte:

```

<i class="fa fa-info-circle text-info ml-2" data-toggle="tooltip"
data-placement="top" th:title="#{label.signature.jws.documentToBeSigned}"></i>

```

Desta forma iriam nas várias páginas aparecer pequenos botões com um "i" onde poderíamos passando o rato por cima dos mesmos verificar informação de ajuda relativa a cada secção pedida.

3 Técnicas de Desenvolvimento de Software Seguro

De forma a identificar e avaliar as capacidades do grupo de trabalho em desenvolver software seguro, foi usado o **Software Assurance Maturity Model** (SAMM). Esta avaliação encontra-se no primeiro anexo (*Aula09.2022Abr26.md*).

Em relação à concordância do nosso projeto com o Regulamento Geral sobre a Proteção de Dados (RGPD), foi feita uma análise que se pode encontrar na primeira pergunta do segundo anexo (*Aula11.2022Mai10.md*).

Também foi usada uma ferramenta do CNIL (*Commission Nationale de l'Informatique et des Libertés*) para gerar o Data Protection Impact Assessment (DPIA), que permite avaliar o impacto do nosso projeto na proteção de dados. Estes resultados estão apresentados na segunda pergunta do segundo anexo.

4 Utilização e Teste da Aplicação

Uma vez introduzidos todos os tópicos anteriores, teremos de perceber agora como podemos utilizar a aplicação, ou seja, que requisitos precisamos de ter e o como podemos instalar a mesma, e por fim iremos fazer uma pequena demonstração.

4.1 Requisitos para a **Instalação**

Os requisitos mínimos para esta aplicação são:

- Versão de Java igual ou superior a 11 (testada até à versão 15 do Java);
- Versão do Maven igual ou superior a 3.6;
- Tomcat 9+ para as versões do Java iguais ou superiores a 9;
- Memória e disco: ver os requisitos mínimos para a JVM utilizada. Em geral, quando mais memória disponível melhor a performance;
- Sistema Operativo: não há requisitos específicos;

4.2 Processo de Instalação Windows

Para instalar esta aplicação em Windows, teremos de efetuar os seguintes passos:

- Obter o conteúdo do repositório da aplicação;
- Na diretoria AP2-PD/dss-demonstrations, correr o comando: `mvn clean install`;
- Descomprimir a pasta `dssdemo-bundle-5.10.1`;
- Na pasta extraída, executar o ficheiro `Webapp-Startup.bat`.

4.3 Processo de Instalação Linux

Para instalar esta aplicação em Linux, teremos de efetuar os seguintes passos:

- Instalar uma versão compatível do Java no sistema;
- Obter o conteúdo do repositório da aplicação;
- Na diretoria AP2-PD/dss-demonstrations, correr o comando: `mvn clean install`;
- Descomprimir a pasta `dssdemo-bundle-5.10.1`;
- Na pasta extraída, navegar até ao diretório `apache-tomcat-8.5.78/bin` e dar permissões de execução aos ficheiros `catalina.sh`, `startup.sh` e `shutdown.sh`;
- Definir a variável de ambiente `JRE_HOME` para apontar para o diretório do sistema que contém a JRE instalada;
- Definir a variável de ambiente `CATALINA_HOME` para apontar para o diretório `apache-tomcat-8.5.78` da pasta extraída;
- Executar o ficheiro `startup.sh` (cujas permissões foram definidas acima);

4.4 Demonstração

Seguidamente iremos mostrar os processos e diferentes funcionalidades que podemos utilizar na nossa aplicação.

4.4.1 Login

Inicialmente poderemos ver o processo Login que pede argumentos obrigatórios, uma página de Login com erro caso o Utilizador ou password estejam incorretos e ainda uma página de sucesso de iniciar sessão.

Username:

Password: Preencha este campo.

[Não tem conta? Registe-se.](#)

[Login](#)

DSS Demonstration WebApp: 5.10

Figura 2: Página de Login com pedido de argumentos

Username ou Password errados

Username:

Password:

[Não tem conta? Registe-se.](#)

[Login](#)

DSS Demonstration WebApp: 5.10

Figura 3: Página de Login com erro de Validade

You have logged in successfully!

[Click here to enter the Page](#)

DSS Demonstration WebApp: 5.10

Figura 4: Página de Login de sucesso

4.4.2 Register

Aqui poderemos ver o processo Register que pede argumentos obrigatórios, uma página de Register com erro caso o Utilizador já exista e ainda uma página de sucesso de registo.

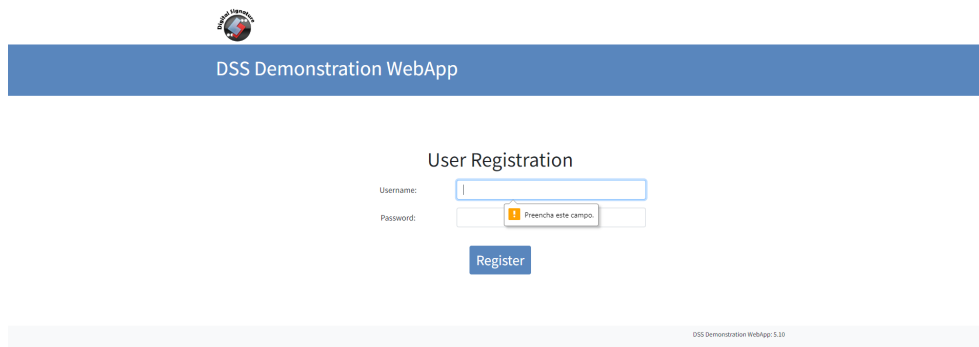


Figura 5: Página de Register com pedido de argumentos

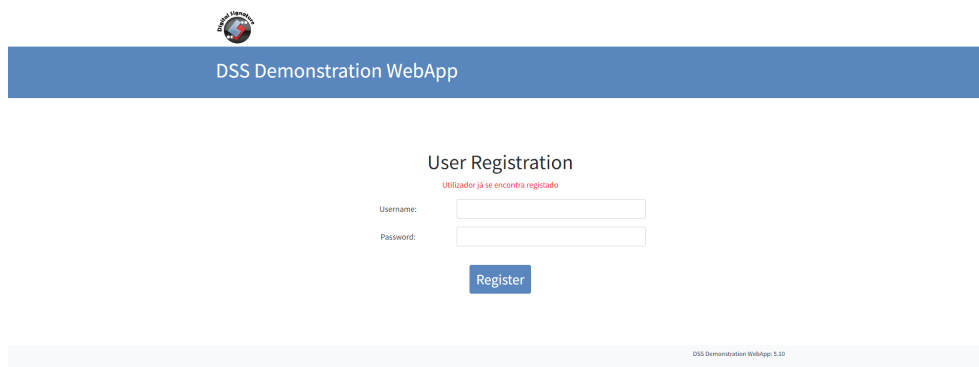


Figura 6: Página de Register com erro de Validade

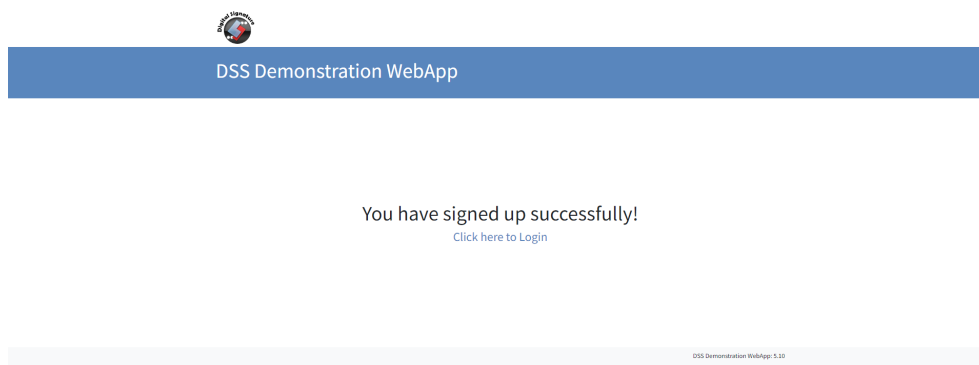


Figura 7: Página de Registo de sucesso

4.5 Perfil

Inicialmente poderemos ver como é apresentada a opção de Perfil na página inicial. Seguidamente podemos ver uma Figura que ilustra o perfil de um Utilizador sem número de telemóvel associado. Temos ainda o preenchimento de um novo número, a sua submissão com a página de confirmação e ainda o novo Perfil atualizado.

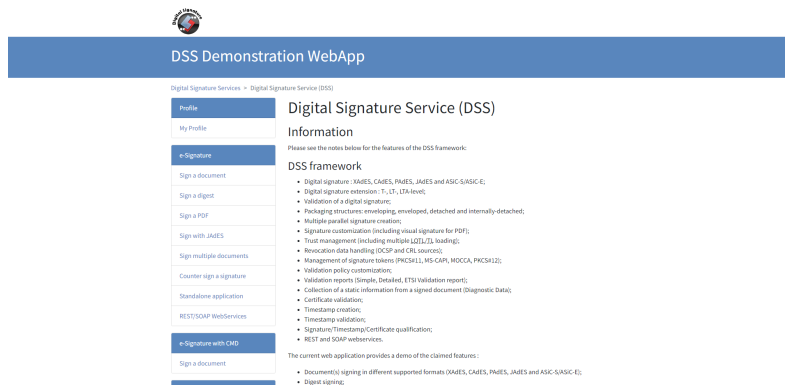


Figura 8: Página principal com secção de Perfil

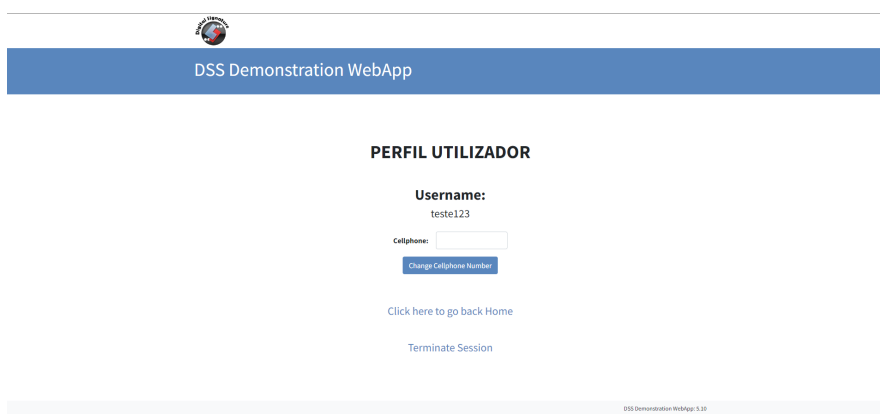


Figura 9: Perfil de um Utilizador sem número de telemóvel associado

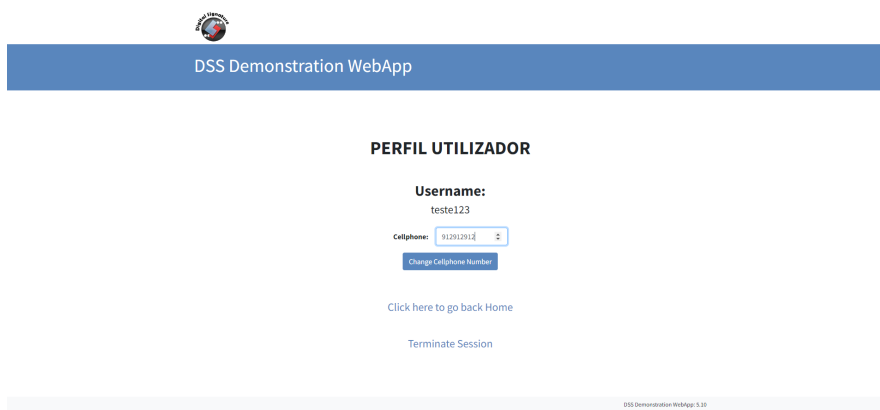


Figura 10: Preenchimento de um novo número de telemóvel

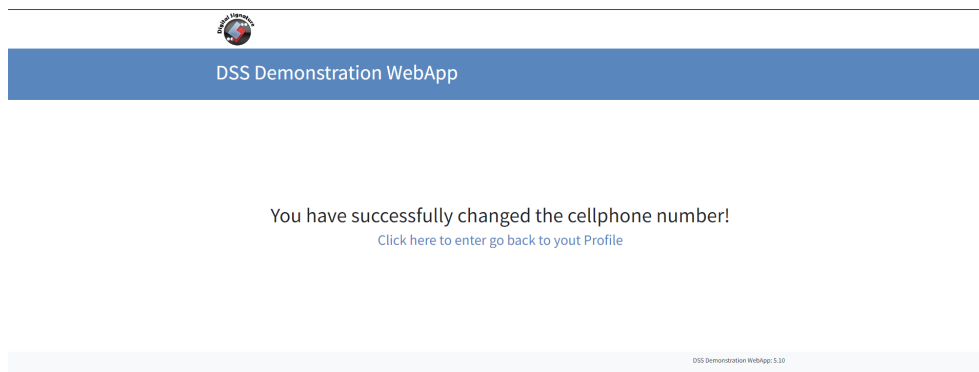


Figura 11: página de confirmação de alteração do número de telemóvel

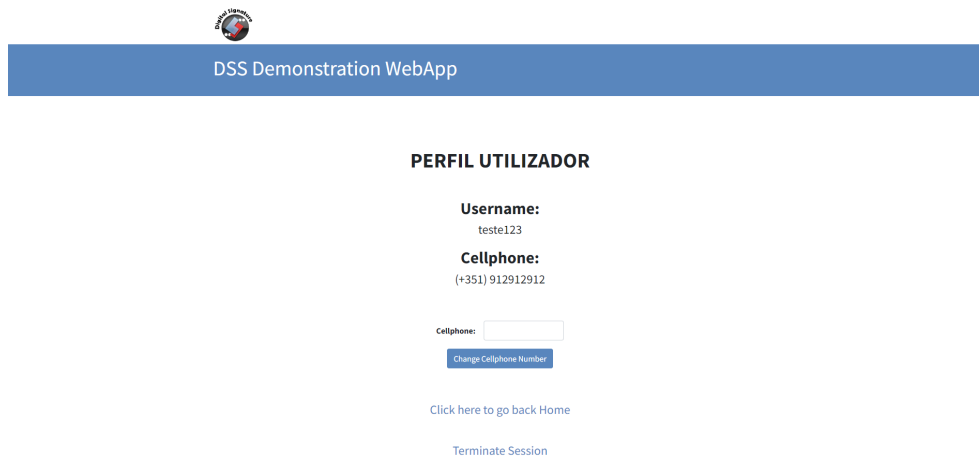


Figura 12: Perfil de um Utilizador com número de telemóvel associado

4.6 Logout

Aqui poderemos ver uma página de confirmação de Logout de um Utilizador.

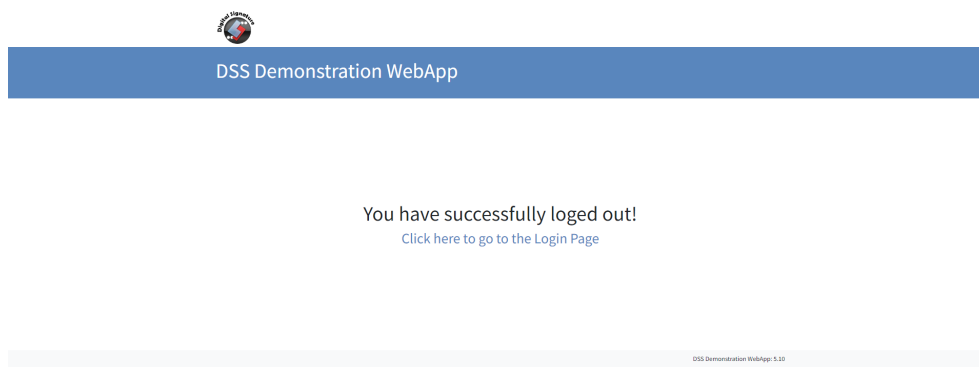


Figura 13: Perfil de um Utilizador com número de telemóvel associado

4.7 Assinatura e Validação de um Documento

Tal como os colegas do ano passado, será também necessário testar a funcionalidade de assinar um documento com recurso a CMD. Assim, escolhendo os parâmetros devidos, ou seja, o tipo de *container*, *packaging* e ainda função de *hash* utilizados poderemos então testar esta funcionalidade.

Sign a document with CMD

File to sign	<div>Escolher ficheiro</div> Nenhum ficheiro selecionado i
Container	<input checked="" type="radio"/> No <input type="radio"/> ASiC-S <input type="radio"/> ASiC-E i
Signature format	<input type="radio"/> XAdES <input type="radio"/> CAdES <input checked="" type="radio"/> PAdES <input type="radio"/> JAdES i
Packaging	<input checked="" type="radio"/> Enveloped <input type="radio"/> Enveloping <input type="radio"/> Detached i <input type="radio"/> Internally detached
Level	<div>PAdES-BASELINE-LTA</div> i
Digest algorithm	<input type="radio"/> SHA1 <input checked="" type="radio"/> SHA256 <input type="radio"/> SHA384 <input type="radio"/> SHA512 i
Allow expired certificate	<input type="checkbox"/> i
Add a content timestamp	<input type="checkbox"/> i
Phone Number (Intl. format)	<div>+351 963743669</div>
PIN	<div>....</div>
	<div>Submit Clear</div>

Figura 14: Assinatura de um Documento

Após a assinatura do documento, podemos então verificar a validade, efetuando a validação desta mesma assinatura. Assim, o resultado obtido seria:

Validation results

Simple ReportDetailed ReportDiagnostic treeETSI Validation Report

Validation Policy : QES AdESQC TL based

Print Download as PDF

Validate electronic signatures and indicates whether they are Advanced electronic Signatures (AdES), AdES supported by a Qualified Certificate (AdES/QC) or a Qualified electronic Signature (QES). All certificates and their related chains supporting the signatures are validated against the EU Member State Trusted Lists (this includes signer's certificate and certificates used to validate certificate validity status services - CRLs, OCSP, and time-stamps).

Signature SIGNATURE_António-Paulo-Rodrigues-Lacerda_20220620-1804

Qualification:	QESig
Signature format:	PAdES-BASELINE-LTA
Indication:	TOTAL_PASSED
AdES Validation Details :	Elements overlap on page(s) [3] The authority info access is not present! The checked certificate shall not appear in the OCSP Responders certificate path!
Certificate Chain:	António Paulo Rodrigues Lacerda EC de Chave Móvel Digital de Assinatura Digital Qualificada do Cartão de Cidadão 00003
On claimed time:	2022-06-20 18:04:54 (UTC)
Best signature time:	2022-06-20 18:05:45 (UTC)
Signature position:	1 out of 1
Signature scope:	Partial PDF (PARTIAL) The document ByteRange : [0, 780653, 818543, 507]

Timestamps **2**

Figura 15: Validação de assinatura de um documento

5 Conclusão

Concluído este projeto, sentimos que conseguimos ultrapassar os desafios propostos, pegando na aplicação fornecida e adicionando a capacidade de autenticação por parte do utilizador, permitindo guardar informações como o número de telemóvel. Também foi adicionada uma interface de ajuda, que explica ao utilizador como usar certas partes da nossa aplicação. De forma geral, o grupo melhorou as suas capacidades de desenvolvimento de software seguro.

Fomos capazes de alterar código de terceiros, usando frameworks e APIs de forma a simplificar o desenvolvimento do software, mas ao mesmo tempo aumentar a sua segurança. Através do modelo de maturidade (SAMM) identificamos e melhoramos as nossas capacidades no desenvolvimento de software seguro, pois avaliamos o nosso estado atual e definimos objetivos para o futuro. Por fim, analisamos o impacto do nosso projeto na proteção de dados, através de uma ferramenta que nos permitiu perceber os pontos fracos do nosso projeto neste aspeto.

Exercícios - Resolução

SAMM (Software Assurance Maturity Model)

Pergunta P2.1

As três práticas de segurança escolhidas, e cuja maturidade será analisada, pertencem à função de negócio *Construction*. Esta escolha deveu-se ao facto de considerarmos que esta é a função de negócio que mais se enquadra com o contexto do nosso grupo de trabalho. De seguida serão apresentados os resultados obtidos, para cada prática de segurança.

- *Threat Assessment*: obtivemos uma pontuação de maturidade de 0.27;
- *Security Requirements*: obtivemos uma pontuação de maturidade de 0.35;
- *Secure Architecture*: obtivemos uma pontuação de maturidade de 0.50;

No total, a função de negócio *Construction* obteve uma pontuação de maturidade de 0.37.

No ficheiro *Our_SAMM_Assessment.xlsx* encontra-se o cálculo do nível de maturidade destas práticas, usando a Toolbox disponibilizada.

Pergunta P2.2

Tendo em conta que este projeto não foi desenvolvido com o intuito de ser usado num contexto real, não achamos que o resultado nível de maturidade obtido seja problemático. Porém, de forma a preparar os elementos do grupo de trabalho para o futuro, definimos como objetivo atingir o nível de maturidade 1 nas três práticas de segurança identificadas na pergunta anterior.

Pergunta P2.3

Para a atingir o nível de maturidade 1 na prática de segurança *Threat Assessment*, o grupo irá focar-se em:

1. Documentar a maioria das possíveis ameaças ao nosso software;
2. Identificar e documentar o tipo de atacantes do qual o nosso software poderá ser alvo.

Em relação à *Security Requirements*, e de maneira a atingir o nível de maturidade 1, o foco estará em:

1. Dispender mais tempo a especificar os requisitos de segurança do software, durante a fase de desenvolvimento;
2. Criar um regulamento de melhores práticas de desenvolvimento de software, que deverá ser seguido sempre que uma aplicação esteja a ser desenvolvida.

Já para atingir o nível de maturidade 1 na *Secure Architecture*, iremos:

1. Garantir que os princípios de desenvolvimento de software seguro são respeitados sempre, por exemplo, através da criação de uma checklist de requisitos que terão de ser cumpridos;
2. Criar padrões de design prescritivos para várias arquiteturas de aplicação, que serão fornecidos à equipa de desenvolvimento, de acordo com o tipo de aplicação a ser desenvolvido.

Exercícios - Resolução

RGPD (Regulamento Geral de Proteção de Dados)

Pergunta P1.1

1.

De seguida serão listados os nove critérios que deverão ser tidos em conta, de forma a perceber se o processamento de dados pessoais irá resultar num risco elevado.

1. Quando os dados são usados para avaliação, incluindo definição de perfis e previsão, especialmente de aspetos relativos ao desempenho no trabalho, situação económica, saúde, preferências pessoais ou interesses, confiabilidade ou comportamento, localização ou movimentos do titular dos dados.
2. Quando o processamento dos dados tem como objetivo uma tomada de decisões automatizada, que tenha implicações jurídicas relativamente à pessoa singular ou que a afetem significativamente de forma similar.
3. Quando o processamento de dados é usado para observar, monitorizar ou controlar os titulares dos dados, incluindo dados recolhidos através de redes ou pelo monitoramento sistemático de zonas acessíveis ao público.
4. Quando os dados processados são considerados sensíveis ou de natureza pessoal, incluindo as categorias definidas no artigo 9 do RGPD, tais como, informação sobre as opiniões políticas ou convicções religiosas ou filosóficas do titular dos dados, bem com dados relacionados com condenações penais e infrações ou com medidas de segurança conexas.
5. Quando se trata de dados processados em larga escala. Embora o RGPD não defina o conceito de larga escala, o WP29 recomenda que os seguintes factores sejam tidos em conta:
 - a. O número de sujeitos envolvidos, quer como um número absoluto ou como uma proporção da população relevante;
 - b. O volume de dados de diferentes dados a ser processados;
 - c. A duração do processamento de dados;
 - d. A extensão geográfica do processamento de dados.
6. Quando se corresponde ou combina dois datasets distintos, por exemplo quando os dados são originários de duas ou mais operações de processamento de dados.
7. Quando os dados processados são relativos a sujeitos vulneráveis, por exemplo crianças (que se considera que não são capazes de consentir ou de se opôr, conscientemente, ao processamento dos seus dados), segmentos mais vulneráveis da população (pessoas com problemas mentais, refugiados, idosos, entre outros) ou quando existe uma relação de desigualdade entre o titular dos dados e o processador destes.
8. Quando o processamento dos dados é inovativo ou recorre ao uso de tecnologias novas (de acordo com o estado de desenvolvimento tecnológico alcançado), por exemplo combinando o uso de impressão digital e reconhecimento facial para um melhor controlo de acesso.

9. Quando o processamento dos dados em si impede o titular dos dados de exercer um direito ou de usar um serviço ou contrato, por exemplo quando um banco usa os dados de um cliente numa base de dados de referência de crédito, de forma a decidir se lhe oferece um empréstimo.

2.

Em relação ao projeto de desenvolvimento 2, este enquadra-se nos seguintes critérios:

- Critério 4: se alguém pretende assinar um documento, existe uma grande probabilidade que este seja de carácter pessoal ou contenha dados sensíveis. Tendo em conta que este projeto permite a assinatura de documentos, é muito provável que tenha de lidar com dados pessoais ou sensíveis, cumprindo este critério;
- Critério 8: a assinatura digital através da Chave Móvel Digital foi aprovada em 2018, logo pode ser considerada uma tecnologia nova. Assim, e sendo que este projeto permite assinar documentos através da assinatura digital, este critério é cumprido;

Pergunta P1.2

A resposta a esta pergunta encontra-se no ficheiro *DPIA_Grupo12.pdf*.