ADC Projeto: Edição 2023/24
Enunciado do exercício de avaliação Individual

Âmbito, objetivo da avaliação, entrega do trabalho e datas importantes

Âmbito da avaliação individual

A avaliação individual tem como âmbito a demonstração de competências individuais adquiridas no decurso das sessões de formação, com enfoque em:

i. desenvolvimento com a plataforma Google App. Engine e gestão de dados persistentes com utilização de tecnologias Google Storage e Google Datastore;
ii. desenvolvimento de serviços REST com possível demonstração com a ferramenta Postman
iii. desenvolvimento de aplicações-cliente Web programadas em Javascript ou Flutter e tecnologia.

Objetivo.

O objetivo do trabalho é o desenvolvimento, teste e demonstração de uma pequena aplicação Web, cobrindo os requisitos apresentados no presente enunciado, utilizando as tecnologias acima indicadas.

Materiais a desenvolver para a entrega do trabalho.

Para entrega do trabalho deverão ser preparados para entrega os seguintes materiais que deverão ser submetidos individualmente até à data limite de entrega:

- **Projeto com o código fonte** (que deve estar em conta individual GitHub) sendo submetido o URL do projeto GitHub que na data de entrega deve estar partilhado como se indicará oportunamente
- **Preenchimento de um formulário (Google Form)** com respostas a questões e que servirão para que cada estudante caracterize os objetivos que realizou no trabalho individual e que pode demonstrar, com base nos requisitos enunciados abaixo. O URL GitHub do repositório individual com o código fonte (acima referido) será fornecido numa das questões do enunciado

Nota: a não entrega dos materiais acima na data de submissão (conforme quadro abaxo) implica em FALTA na avaliação individual

Avaliação do trabalho.

A avaliação do trabalho decorre da avaliação dos materiais entregues:

1. Entrega atempada dos materiais na data de submissão do trabalho
2. Demonstração presencial e discussão dos objetivos realizados. Para esta sessão presencial (a decorrer em laboratório), o escalonamento dos alunos será feito em convocatória individual. Na sessão presencial os alunos serão confrontados com um guião de desmonstração, com vários passos em que deverão demonstrar que o trabalho consegue

Datas importantes:

- **25/Março/2024:** disponibilização do presente enunciado
- **4/Abril/2024:** disponibilização do formulário de submissão do trabalho
- **4/Abril/2024:** disponibilização do guião de teste para verificação/avaliação do exercício
- **6/Abril/2024 (23h59):** Entrega do trabalho (formulário de submissão onde será indicado o GitHub Repo)
- **Demonstração/Discussão individual do trabalho**: terá lugar nos dias 11 e 12 de Abril. O escalonamento será anunciado dia 10/Abril.

1. Estrutura do projeto (GitHub) com o código a desenvolver para o projeto

Cada aluno deverá desenvolver e apresentar um demonstrador, desenvolvido como uma aplicação Web, cujo código deve estar disponível em repositório GitHub (conta individual de cada aluno). No repositório GitHub (individual de cada estudante) o projeto será identificado com o título **APDC-2024-Individual**, tendo a seguinte organização (estrutura hierárquica):

- **Raiz do projeto e elementos obrigatórios:**
  - **README.md**
  - **projeto**. Onde estará o projeto (código fonte) de acordo com a estrutura do seu desenvolvimento, podendo esta seguir a organização de acordo com o ambiente IDE usado (Eclipse ou IntelliJ, por exemplo) devendo ser um projeto Maven
  - **opcional.** Aqui podem ser opcionalmente incluídos quaisquer elementos opcionais adicionais ou informação complementar sobre o exercício realizado que se entendam serem relevantes para efeitos da avaliação do trabalho feito.

1. Requisitos a desenvolver e demonstrar o projeto

O projeto tem os seguintes requisitos funcionais mínimos (aos quais cada aluno pode acrescentar, se pretender, tudo o que entender ser algum aspeto valorativo para demonstração dos conhecimentos adquiridos)

1.1 Aplicação: demonstrador a desenvolver

A aplicação a desenvolver terá um componente CLIENTE (preferencialmente já em ambiente Web-Browser ou Flutter) e componente SERVIDOR (Cloud App Engine e dados persistentes (Google Datastore e/ou Google Storage). O demonstrador deverá ser desenvolvido de forma a poder ser compilado e *deployed* na nuvem (Google Cloud), durante a sessão oral de demonstração. A demonstração será feita a partir do guião de avaliação. No início desta sessão de demonstração cada aluno partirá de um projeto "vazio" no IDE que for utilizado, passando-se pelos seguintes passos:

(1) criação e instalação do projeto recorrendo aos *sources* "entregues" e respetivo descarregamento a partir da plataforma GitHub e do repositório do individual
(2) compilar localmente demonstrando não ter erros de compilação
(3) *deployment* e demonstração de prontidão para execução na nuvem (GAE+Datastore+Storage)
(4) demonstração dos requisitos a seguir enunciados com base em passos de um guião de teste que cobrirá a funcionalidade que está indicada nos requisitos seguintes
(5) apresentar ou comprovar algum extra ou aspecto opcional valorativo que se pretenda mostrar
(6) na sessão de demonstração cada estudante deve estar capaz de dar resposta a qualquer esclarecimento ou questão sobre o código desenvolvido que seja questionado. Prevê-se que as sessões individuais de demonstração decorram em cerca de 20 minutos como referência, para se completar todos os passos definidos no guião de avaliação.

1.1.1 Requisitos do componente Cliente

Aplicação cliente: aplicação JAVASCRIPT/AJAX/FLUTTER (demonstrável com utilização em pelo menos dois browsers diferentes, podendo estes serem selecionados de entre os seguintes (Google *Chrome*, Mozilla *Firefox*, *Apple Safari* ou MS Edge)

1.1.2 Componente Servidor

Deverá executar no ambiente Cloud (*Google App Engine* + *Persistência*), sendo demonstrado em conta Google

Cloud Platform do estudante. Na sessão de demonstração os estudantes vãos ser chamados a realizar o deployment local do sistema e deployment remoto no momento da demonstração

1.1.3 Sobre o uso de *tokens* para controlo de acesso nas operações

Notar que todas as operações a implementar devem ser feitas com pedidos POST, usando-se sempre que necessário tokens de autenticação, com a seguinte estrutura:

<Output> = operation(<Input>, <Token>) isto é: {

operation: <opid> // Código de operaão
Input: input JSON data // Data de envio da operação
Token: JSON token format // Token (sessão) para controlo do pedido da operação
Outor: output JSON result data // Resultado (estado) de realização (ou não) da operação
}

Se o <*Token*> estiver correto e válido (correspondendo a uma sessão válida do utilizador), esta será realizada pelo servidor (desde que o utilizador seja suposto ter permissões para essa operação dado o seu Role). Se não é esse o caso ou se a operação resultar em erro ou exceção (seja por não validade da sessão, *token* inválido, *token* de sessão expirada, por exemplo, por Role não adequado), o resultado refletirá o erro. Quando uma sessão é expirada o utilizador será redirecionado para a página de LOGIN. O utilizador não pode ficar confrontado com um estado indeterminado sobre o que possa ter acontecido.

Para construção dos tokens, usar a inspiração que foi fornecida nas sessões de formação, devendo os tokens conter os elementos que sejam necessários para garantir a segurança no controlo de acesso às operações.

1.1.4 Funcionalidade a desenvolver

O demonstrador deverá conter as **operações OP1 a OP10 indicadas abaixo.**

OP1: Criação e registo de utilizadores (*account + perfil + role de utilizador*).

Deve suportar-se o registo e criação de até 4 contas de utilizadores na aplicação (devendo ser mostrada a possibilidade de registo de alguns utilizadores). O registo deve permitir que na criação de contas os utilizadores registem atributos (à escolha dos alunos), mas que devem incluir pelo menos os seguintes dados (independentemente de outros que os alunos considerem como valorização):

- Atributos obrigatórios para criação das contas
  - *Username* (ou UserID), exemplo: petermurphy3456
  - *Email* do utilizador: terá que ter formato <string>@<string>. …. .<dom>
    - <dom> representa um *top-level domain* DNS
    - Exemplo: petermurphy3456@campus.fct.unl.pt
  - Nome
    - Nome do utilizador (ex., Nome Apelido)
  - Telefone: número de telefone (exemplo +3512895629
  - Password: PWD (invisível): **********
    - A inserção da PWD deve sempre ser confirmada duas vezes
    - As passwords podem ter uma regra de validação à escolha dos alunos (ex., comprimento mínimo, conter letras e números ou um ou mais caracteres de pontuação, etc.), exemplo; 654%strongp?

Notas: *não vai ser preciso no exercício confirmarem a conta com confirmação do Email como é habitual em muitas aplicações, para efeitos de ativação da conta. Também não será preciso verificar se no Email fornecido o domínio DNS é válido), nem testar que é*

possível mandar EMail ao utilizador da conta registada ou enviar ou SMS para o telefone fornecido). Todos estes aspetos ou demais fatores de confirmação da abertura de contas, a serem suportados, serão sempre elementos opcionais se os estudantes os quiserem considerar.

- Atributos complementares da conta (**OPCIONAIS** no momento do registo para criar uma conta) mas que podem ser adicionados a posteriori pelos utilizadores, após login em sessão criada e devidamente autenticada) e com base numa operação de atualização/preenchimento dos respetivos atributos.
    - Perfil: Público ou Privado
    - Ocupação (e.g. Professor, Estudante, Investigador,)
    - Local de trabalho (e.g. empresa, escola, etc...)
    - Morada: (ex., "Rua dos alunos de APDC2324, No 100, Piso 2, Porta 116)
    - CP: código postal XXXX-XXX, ex: 2987-546)
    - NIF: Nº de identificação fiscal de contribuinte (STRING), ex 876784563
    - Foto: uma foto-passe (ou imagem-icon) (formato à escolha): ex: BMP, JPEG)

  Nota: Não é necessário validar os dados registados (ex., códigos postais, endereços existentes ou não, etc., devendo apenas validar-se a correção do formato esperado dos inputs.
  Quando um utilizador se regista no sistema, a conta será criada com dois atributos suplementares: ROLE e ESTADO

  As contas após serem criadas terão sempre estes dois atributos com os seguintes valores:

  ROLE: USER (representando um utilizador "normal")

  ESTADO: INATIVO

  Enquanto o estado estiver INATIVO não será possível fazer login na conta. Para o efeito, o estado deve ser mudado na base de dados de INATIVO para ATIVO.

  O ROLE inicialmente criado com o valor USER, pode depois ser modificado para um dos seguintes roles: SU (Role Super-User) , GA (Role de gestão da aplicação) ou GBO (Role de Gestão Back Office)

  ROLES:

    - A hierarquia de autorizações para mudança de ROLES em contas criadas está indicada abaixo

- **Nota importante**

  Como estratégia *Bootstrap*, em tempo de *deployment* da aplicação, deve logo ser criado uma conta de um utilizador com *username* "root", *password pré*-inicializada bem como os restantes campos obrigatório como acima, devendo a respetiva conta ficar logo registada como conta em estado ATIVO na base de dados e de modo que esse utilizador possa fazer operações como as que se referem a seguir.

OP2: Mudança de role de utilizador

Esta operação permite mudar o role de utilizadores de contas criadas, mas só pode ser executada de acordo com as seguintes permissões:

- SU: pode passar qualquer conta de qualquer role para qualquer role
- GA: pode passar contas USER para GBO (ou vice-versa)

- GBO: não pode mudar roles
- USER: não pode mudar roles

OP3: Mudança de estado de conta

Esta operação permite mudar o estado de contas de INATIVO para ATIVO (ou vice-versa) mas só pode ser executada com as seguintes permissões a partir do role do utilizador que a executa

- SU: pode passar o estado de qualquer conta de INATIVO para ATIVO (ou vice-versa)
- GA: pode mudar o estado de contas de roles USER ou GBO de INATIVO para ATIVO ou vice-versa
- GBO: pode mudar o estado de contas de role USER de INATIVO ara ATIVO ou vice-versa
- USER: não pode mudar estado de contas

OP4: Remoção de utilizadores.

Esta operação serve para remover contas da aplicação, devendo obedecer aos seguintes requisitos de autorização:

- SU: pode remover contas de qualquer utilizador e de qualquer role
- GA: pode remover quaisquer contas de roles GBO ou de USER
- GBO: não pode remover contas
- USER: só pode remover a sua própria conta (o próprio utilizador)

OP5: Listar utilizadores.

Esta operação serve para mostrar os utilizadores existentes na aplicação, mostrando os seguintes atributos:

- Utilizadores com role USER: listar os utilizadores registados e que têm perfil USER, mostrando apenas os atributos: *username*, *email* e nome (mas apenas os que têm role USER) e só se as respetivas contas estiverem com perfil público e as contas em estado ATIVO.
- Utilizadores com role GBO listar todos os atributos de utilizadores registados (mas apenas os que têm role USER) e independentemente de estarem com perfil público ou privado ou do estado da conta
- Utilizadores com role GA listar todos os atributos de utilizadores registados (que tenham role USER, GA ou GBO) e independentemente de estarem com perfil público ou privado ou do estado da conta
- Utilizadores SU: listar todos os atributos de utilizadores registados ou do estado da conta (de qualquer role)

OP6: Modificação de atributos de contas de utilizadores.

Operação executável por utilizadores com qualquer role, que permite modificar os respectivos atributos da conta, do seguinte modo:

- USER: pode modificar todos os atributos da sua própria conta, exceto os atributos *Username* (ou UserID), *Email* ou Nome (inicialmente registados na criação da conta), assim como quaisquer atributos de controlo tal como ROLE e ESTADO
- GBO: só pode modificar atributos de utilizadores USER (exceto o respectivo *Username* (ou *UserID*)
- GA: só pode modificar atributos de utilizadores GBO ou USER (exceto os respectivos *Username* (ou *UserID*)
- SU: pode modificar atributos de utilizadores GA, GBO ou USER (exceto os respectivos *Username* (ou *UserID*)

OP7: Modificação de *password*

Operação de mudança de *password*. Exige a *password* atual e a inserção da nova *password*, com dupla verificação (inserção da password duas vezes com comparação entre elas). Esta operação só pode ser suportada por cada utilizador na sua própria conta (independentemente do seu papel)

OP8: Login

Operação de "*Login*" de sessão

O utilizador fornece o IDENTIFICADOR (ex., *Username* ou *UserId* ou *Email*) e a *password*. Se as credenciais são corretas (autenticação pelo servidor): recebe um *token* para a sessão que entre quaisquer outros atributos terá:

- USER: o identificador (ou *username*, *UserID* ou *Email*)
- ROLE: o role (ou código de role)
- VALIDITY: validade do token (correspondente à validade da sessão), com as propriedades:
    - VALID_FROM: data da emissão (tempo do servidor)
    - VALID_TO: data de expiração (tempo do servidor)
    - VERIFICADOR: VERIF VERIFICADOR: ex., um "*magic (random) number*" emitido pelo servidor que traduz uma prova de autenticidade da emissão pelo servidor dos dados do *token*, para realização de operações na sessão de LOGIN. A forma de criar este VERIFICADOR é opcional, devendo ser definida por cada aluno.

Quando o utilizador conclui com sucesso o LOGIN tem acesso a uma "página de boas-vindas", que indica de forma visível qual o PAPEL em que pode atuar e em que pode realizar as restantes operações suportadas para o seu papel (ver abaixo **Operações exemplificativas em sessões de utilizadores)**

OP9: Mostrar o token de sessão (após LOGIN) e respectivos elementos de controlo

Mostrar o conteúdo dos elementos do token de sessão que foi emitido no LOGIN

OP10: LOGOUT

Operação de "*Logout*" da sessão
Cliente executa a operação e como resultado o servidor revoga o TOKEN (mesmo que este esteja ainda esteja válido) e a sessão é redirecionada no cliente para uma página que notifica LOGOUT (desaparecendo a informação que se "via" na sessão aberta). Após o *Logout*, não é possível realizar operações em nome do utilizador (mesmo que se reutilize um token que ainda estava válido), a não ser que o utilizador faça de novo LOGIN com sucesso em nova sessão e obtenha assim um novo TOKEN válido.

Operações extra

Para além das operações anteriores, podem fazer até duas operações (funcionalidade à escolha de cada aluno) que podem ser demonstradas e executadas numa sessão LOGIN de um utilizador. Ver as seguintes operações como referência indicativa. Os alunos podem escolher e implementar qualquer uma das indicadas ou qualquer outra à sua escolha.

2. Critérios de Classificação

Como indicado (*ver quadro inicial de Datas Importantes*) será depois disponibilizado um guião associado a este enunciado. Este guião estará associado a passos de testes para demonstradores que cobrirá as funcionalidades indicadas (tendo como referência base os requisitos obrigatórios). Assim, os requisitos serão demonstrados "passo a passo" obedecendo a tempos definidos. O sucesso de realização de cada passo da demonstração e o cumprimento do tempo definido para cada passo são fatores que determinam a avaliação individual. Cada passo realizado e demonstrado no guião de avaliação terá uma pontuação que será como referência a seguinte:

- De início: ter o código submetido e pronto para uso na sessão de demonstração (10%)

- Conseguir fazer download do projeto e fazer a sua compilação a partir de um projeto vazio criado no IDE: (10%)
- Conseguir fazer *deployment* para o ambiente *Cloud* (GAE + Persistência) e mostrar que está pronto a operar (20%)
- Demonstrar em sequência as operações de um utilizador do tipo USER (mostrando as operações que pode realizar de acordo com as autorizações corretas) (20%)
- Na sessão presencial poderão ser realizadas observações equivalentes que incluirão os outros papéis (ROLES) (30%)
- Se forem demonstradas operações extra propostas pelos alunos (10%)

A avaliação final terá por bases os seguintes critérios (em escala de 0 a 20 valores):

NOTA: a não realização desta atividade implica reprovação à unidade curricular (frequência). Esta avaliação conta 20% na nota final de PIPP+ADC (nota única combinada das duas UC).