IPS Instituto Politécnico de Setúbal Escola Superior de Tecnologia de Setúbal	**Programação Web 2023/2024**	TPSI

Objetivo

O principal objetivo deste projeto é desenvolver, utilizando HTML, CSS e JavaScript, uma solução Web de um sistema de informação (SI) que permita a gestão de informação associada a uma determinada área de negócio. O SI tem obrigatoriamente seguir as tecnologias e técnicas ensinadas durante a aula e usar uma base de dados MySQL. O domínio de aplicação ou área de negócio deve ser preferencialmente escolhido pelos alunos.

Requisitos tecnológicos

O sistema de informação tem obrigatoriamente de ser uma aplicação web com separação entre as camadas de cliente, servidor e base de dados.
Projetos que não sigam os requisitos da UC (dados em aula) não serão aceites para avaliação.

Cliente

O lado do cliente deve usar CSS, HTML e javascript. Não podem usar qualquer framework para desenvolvimento do cliente (ex: **não podem** usar bootstrap, angular, react, view, etc).
A aplicação deve ter uma página inicial que deve ligar a múltiplas páginas com as funcionalidades necessárias à aplicação.

HTML: A nível de HTML devem usar uma estrutura clara, aproveitando ao máximo as vantagens trazidas pelo HTML5, e evitando elementos como o "div" e evitando a excessiva utilização de ids e classes.

CSS: O CSS deve ser colocado em ficheiros separados do HTML e sempre que possível organizado de forma hierárquica de forma a poder ser reutilizado em diversas páginas.

Javascript: Para o javascript devem usar o que é seguido na UC. Devem criar código modular, identificando entidades do problema que serão representadas por classes em que os dados e funcionalidade são agregados nessas classes.
Toda a manipulação das páginas tem de usar as funcionalidades de manipulação do DOM do javascript (aka: não usar o "innerHTML").

Alguns dos cuidados a ter em conta no lado do cliente:

- Desenvolver primeiro as páginas e funcionalidades que permitam demonstrar o conceito proposto como tema do projeto (MVP → Minimum Value Product).
- Devem dividir o trabalho por funcionalidade (ex: não ter um aluno só a fazer HTML+CSS, até porque o foco principal desta UC será o javascript). Páginas que não "funcionam" não são avaliadas.
- Apesar de ser importante a página "funcionar" (desempenhar a função) a apresentação da página e principalmente a usabilidade da mesma será tida em conta.

Servidor:

O lado do servidor tem de usar as tecnologias Node.js/Express.js tal como ensinado durante as aulas.
O servidor deverá estar estruturado de acordo com o que será ensinado nas aulas. Alguns dos elementos principais (outros serão apresentados nas aulas):

- Ficheiro "app.js" com as configurações do servidor Node.js/Express.js (liga aos restantes componentes do servidore).
- Pasta "www" com os ficheiros do lado do cliente (HTML, CSS, JS, imagens, etc).
- Pasta com os ficheiros de roteamento da REST API.
- Pasta com os ficheiros de "lógica de negócio". As funcionalidades de acesso a dados e cálculo (lógica de negócio) devem estar separadas do roteamento.
- Ficheiro de configuração da ligação à base de dados

O servidor tem obrigatoriamente duas funções:

- Servidor de páginas: a pasta "www" com os ficheiros a fornecer ao cliente. Estes ficheiros contêm apenas o conteúdo fixo (estático) das páginas, o conteúdo dinâmico será preenchido pelo javascript.
- REST API: caminhos que fornecem dados ao cliente. Serão usados para popular as páginas com a informação necessária (a maioria proveniente da Base de Dados).

É obrigatório a utilização da REST API para qualquer informação dinâmica. Ou seja, qualquer página com informação dinâmica seguirá os seguintes passos:

- Pedido HTTP dos ficheiros estáticos (HTML, CSS, Javascript, etc) ao servidor de páginas.
- Código de javascript do lado do cliente faz um pedido AJAX da informação dinâmica a um dos caminhos da REST API.
- A REST API do servidor corre o código necessário para verificar o pedido, obter a informação da base de dados, processar a informação e enviar a informação para o cliente.
- O código de javascript do lado do cliente obtém a informação e processa a mesma inserindo-a nos locais apropriados da página usando as funcionalidades de manipulação do DOM.

A utilização de templates (páginas dinâmicas criadas do lado do servidor) só pode ser usada para o preenchimento de conteúdo estático repetido das páginas (ex: cabeçalhos de páginas, rodapés, barras de navegação comuns, etc).

Serão também dadas na UC algumas bibliotecas que serão obrigatórias para as funcionalidades respetivas (ex: usar a biblioteca Passport.js para autenticação), seguindo o que irá ser ensinado durante as aulas.

Base de dados:

É obrigatória a utilização da uma base de dados MySQL para gestão dos dados da aplicação proposta.
A base de dados tem de ter dado suficientes para a demonstração das funcionalidades, não se devendo assumir que serão introduzidos dados pelo utilizador para teste de uma funcionalidade (a menos que seja a única forma de testar essa funcionalidade).
A base de dados pode ser maior que o que é usado na aplicação (partes que não são usadas).
Qualquer funcionalidade que não seja possível testar por falta de dados não será contabilizada na avaliação.

Requisitos Funcionais

Qualquer projeto que não tenha um conjunto de funcionalidades que permita comprovar o tema proposto ("proof of concept") não terá aprovação. Por exemplo, só conseguir visualizar informação mas não conseguir introdução informação relevante, em particular informação que relaciona entidades do problema.

Para além de seguir a estrutura pedida o projeto terá de ter um conjunto mínimo de funcionalidades que depende também da dimensão do grupo de alunos.

O projeto tem de ter uma página inicial que explica o âmbito do projeto e o papel da aplicação. Deve também ter uma página que apresenta a equipa, focando-se principalmente nas skills que cada um trouxe para o projeto.

Apesar da página inicial poder ter conteúdo dinâmico devem evitar nesta página colocar funcionalidades, deixando esta página ser um hub inicial para chegar às páginas funcionais.

No mínimo cada aluno deve ter um módulo da aplicação com as seguintes funcionalidades:

- Visualização de informação contida em múltiplas tabelas (o módulo deve relacionar informação).
    - Tal pode ser feito combinando informação na mesma página ou interligando múltiplas páginas (ex: página que mostra uma lista de produtos, página do produto ao clicar num elemento da lista, página que permite encomendar o produto ao carregar no botão de encomenda, etc).
- Introdução de dados, incluindo informação que relaciona entidades de diferentes tabelas (ex: encomendar um produto relaciona o produto com o cliente).
- Filtragem e ou agregação de informação.
- O módulo deve ser um componente relevante para o tema em questão (não um módulo geral que pode existir em qualquer tema como por exemplo registar um utilizador).

Cada aluno deve ser responsável por pelo menos um módulo (parte de cliente e servidor), se bem que os módulos podem ter intervenção de outros colegas (mas o aluno responsável deve ser o principal contribuidor e mexer em cada um componentes principais do módulo).

Cada módulo deve ser suficientemente individualizado dos restantes e ter alguma complexidade. O aluno responsável deve estar à vontade para explicar pelo menos o funcionamento principal do módulo. Desta forma será possível avaliar os conhecimentos desse aluno num contexto com dimensão aceitável para permitir pelo menos a sua aprovação na UC.

A repetição da mesma funcionalidade múltiplas vezes não acrescenta valor por si (ex: mostrar múltiplas tabelas individuais da Base de Dados).

NOTA: Todos os alunos devem desenvolver os vários componentes de um módulo (cliente, servidor, ligação à BD, etc). Qualquer aluno que não tenha desenvolvido um dos componentes principais ensinados nas aulas será penalizado na sua avaliação, mesmo que consiga explicar o que os colegas fizeram, podendo mesmo reprovar se se tiver focado demasiado num conjunto restrito de conceitos (por não poder ser avaliado nos outros).

Relatório

Devem ter um relatório (para ambas as fases) com os seguintes elementos (alguns elementos são comuns à UC de base de dados e podem aproveitar os mesmos para ambas as UCS):

- Capa (identificando o tema, nome hipotético do software e os dos elementos constituintes do grupo (número do grupo, número e nome dos estudantes);
- Índice;
- Introdução:
    - Apresentação do tema e âmbito do projeto;
    - Definição do domínio do problema.

- Funcionalidades desenvolvidas com indicação do papel de cada aluno nessa funcionalidade (**para cada funcionalidade deve sempre haver um responsável**, mas devem também indicar o papel dos outros colegas).
- Auto-avaliação dos alunos: Tabela em que cada aluno resume a sua participação no projeto devendo também colocar uma percentagem que indica a sua participação geral no projeto (deve considerar dimensão e complexidade do que foi feito).

Fases do projeto

O projeto terá duas fases de entrega e discussão.
Apesar do projeto ser em grupo a nota é individual e depende da participação e discussão de cada aluno. **Alunos que não estejam presentes na discussão do projeto terão zero no mesmo.**

Submissões

O grupo deve eleger um "líder" que irá submeter o projeto (só um dos elementos deve submeter o projeto) em cada fase no moodle. Caso seja um outro elemento a submeter deve avisar o docente por email.
A submissão deve estar contida num ficheiro zip (extensão .zip) com a pasta do projeto. **A pasta do projeto deve conter o código e o relatório (formato pdf)**.
No caso da segunda entrega **têm de apagar a pasta "node_modules"** antes de fazer o zip.

Primeira fase - até 20 de Maio – 30% do projeto

Entrega do projeto com a parte de cliente completa e o relatório (.pdf):

- Servidor node.js simples com serviço de páginas (ou servidor Node.js/Express.js equivalente conforme apresentado na UC).
- Pasta "www" com os ficheiros necessários ao lado do cliente.
- HTML e CSS com o conteúdo estático das páginas
- Ficheiros de javascript com as funcionalidades necessárias para preencher o conteúdo dinâmico das páginas e organizado em classes conforme ensinado nas aulas.
- De momento a informação a ser usada para o conteúdo dinâmico será lida a partir de ficheiros contidos na pasta www (aconselha-se o formato JSON, mas podem usar também formatos como o CSV).
    - O código para obter informação destes ficheiros deve estar separado do restante código (ficheiros de request) e será substituído no futuro por chamadas AJAX à API REST (serão dados exemplos nas aulas para ambos os casos).

Critérios de avaliação da primeira fase:

- Dimensão e complexidade da funcionalidade (10%)
    - Qualquer projeto cuja funcionalidade não permite demonstrar o tema proposto terá avaliação negativa.
- Organização e qualidade do código (10%)
    - Qualquer projeto que não siga o foi ensinado nas aulas (requisitos tecnológicos) terá avaliação negativa podendo nem ser avaliado (ter zero) se se afastar completamente do pedido.
- Usabilidade, detalhes técnicos, criatividade e adequabilidade para o tema proposto (10%).

Segunda fase - até 24 de Junho – 70% do projeto

Nesta segunda fase será incluída a parte do servidor podendo ser completadas ou acrescentadas funcionalidades.
Projetos que estejam preparados para notas mais elevadas devem considerar também a criação de funcionalidades que envolvam pesquisa por parte dos alunos. Exemplos:

- Utilização de bibliotecas ou outras tecnologias complementares ao que foi dado nas aulas e que se enquadrem no tema (ex: google maps, acesso a apis externas, etc).
    - Terão de ser complementares, não podem substituir o que foi dado na UC (ex: autenticação).
    - Terão de se enquadrar no tema (não forçar a utilização de algo).
- Desenvolvimento de funcionalidades mais complexas que usem o que aprenderam, mas de forma mais complexa do que foi exemplificado nas aulas.
    - Neste caso é importante que o código criado esteja bem organizado e sempre que possível seja genérico.

Entrega do projeto com a parte de cliente e servidor completa e o relatório (.pdf):

- Servidor node.js/express.js com parte de cliente e servidor.
- Pasta "www" com os ficheiros necessários ao lado do cliente.
    - HTML e CSS com o conteúdo estático das páginas
    - Javascript com as funcionalidades necessárias para preencher o conteúdo dinâmico das páginas e organizado em classes conforme ensinado nas aulas. Todos os dados necessários para o preenchimento devem ser pedidos através de pedidos AJAX à REST API.
- Pastas com o código necessário à REST API (roteamento, lógica de negócio e ligação à BD).
- Outras pastas/ficheiros necessários à configuração de bibliotecas (ex: autenticação).

Critérios de avaliação da segunda fase:

- Melhorias à interface desenvolvida na primeira fase e integração (20%):
    - As mesmas 3 dimensões usadas na primeira fase (5% cada).
    - Pedidos AJAX e utilização da REST API (5%)
- Lado do servidor (30%):
    - Dimensão e complexidade da funcionalidade da API REST(10%)
    - Organização e qualidade do código (10%)
    - Verificações, detalhes técnicos (ex: autenticação), criatividade das soluções e coerência geral da API (10%).

  NOTA: Podem ter mais funcionalidades na API REST do que as que serão utilizadas na aplicação, tornando a API mais coerente. As API REST devem ser genéricas e de propósito geral pois poderão ser usadas para várias aplicações (se bem que em caso de falta de tempo devem se focar no que é preciso para a aplicação Web).
- Geral (20%):
    - Coerência da aplicação web como um todo (10%)
        - Em resumo, se a aplicação serve para o pretendido, e é "agradável" de usar tanto a nível de funcionalidade como de interface.
    - Projeto demonstra a aplicação dos conceitos ensinados e a exploração de técnicas e conhecimentos por parte dos alunos (10%)
        - A nota máxima neste parâmetro dependerá da exploração de técnicas, tecnologias ou conceitos não dados nas aulas.

Exceções

Pedidos de exceção terão de ser feitos e justificados por email sendo avaliados pelo docente.