Instituto Superior de Engenharia de Lisboa
Licenciatura em Engenharia Informática e de Computadores
Desenvolvimento de Aplicações Web
Teste Final de Época Normal, Semestre de Verão, 21/22
Duração: 2 horas

---

1. (6) Para cada uma das questões sequintes, indique qual a resposta correta. Cada resposta incorrecta subtrai 1/3 pontos à classificação total do conjunto de questões deste grupo.

   1.1. No protocolo HTTP, o conceito de *interface uniforme* significa que:
   - i. A semântica de um *header* de resposta não depende do recurso alvo.
   - ii. Os identificadores de recurso estão presentes no caminho e não na *query string*.
   - iii. O mesmo *media type* tem de ser usado em todas as respostas.
   - iv. Os únicos métodos que podem ser usados são o `POST` and `GET`.

   1.2. No protocolo HTTP, uma mensagem de resposta de *status code* 400:
   - i. Não pode ter *payload*.
   - ii. Pode ter um *payload* usando um qualquer *media type*.
   - iii. Tem de ter um *payload* usando o *media type* `application/problem+json`.
   - iv. Tem de ter um *payload* usando o *media type* `application/json`.

   1.3. Uma mensagem de pedido com método `GET` para o recurso `https://example.com/projects/123.json`:
   - i. Tem de resultar numa mensagem de resposta com `Content-Type` igual a `application/json`.
   - ii. Tem de resultar numa mensagem de resposta com `Content-Type` igual a `text/json`.
   - iii. Tem de resultar numa mensagem de resposta com `Content-Type` igual a `application/json` ou a application/problem+json.
   - iv. Pode resultar numa mensagem de resposta com qualque valor de `Content-Type`.

   1.4. Na plataforma Spring MVC, por omissão, uma classe anotada com `@RestController` vai ter as seguintes instâncias criadas pelo contexto Spring:
   - i. Uma única instância.
   - ii. Uma instância distinta por pedido.
   - iii. Uma instância distinta por cada ligação de um cliente.
   - iv. O menor número de instâncias, de forma a que nunca exista uma instânca a ser usada simultaneamente por *threads* diferentes.

   1.5. No contexto de uma *single page application*, a avaliação da seguinte expressão resulta em:

   ```
   window.location.pathname = '/projects'
   ```

   - i. Em nenhum pedido HTTP executado.
   - ii. Na execução de um pedido de método `GET` para o caminho `/index.html`.
   - iii. Na execução de um pedido de método `GET` para o caminho `/projects`.
   - iv. Na execução de um pedido de método `GET` para o caminho `/projects.html`.

   1.6. A avaliação da seguinte expressão JSX `<Example a="b"/>`
   - i. É equivalente à avaliação da expressão `React.createElement(Example)({a:"b"})`.
   - ii. É equivalente à avaliação da expressão `React.createElement(Example, {a:"b"})`.
   - iii. É equivalente à avaliação da expressão `React.createElement(Example({a:"b"}))`.
   - iv. Nenhuma das anteriores.

2. (2) No *media type* Siren, quais as diferenças na estrutura e no propósito dos *links* e das *actions*.

3. (2) No contexto de uma aplicação que usa as biblioteca React e React Router, qual a diferença entre a utilização dos elementos `<a />` e `<Link />`?

4. (2) Descreva os elementos do processo de construção de aplicações usado nesta unidade curricular que permitem a utilização de bibliotecas fornecidas pelo NPM em aplicações para execução no *browser*.

5. (4) Realize um ou mais componentes para a plataforma Spring MVC, de forma a que um recurso seja exposto no caminho `/pending`. Um pedido de método `GET` para esse recurso deve retornar uma mensagem com uma representação contendo um objeto JSON. Esse objeto deve representar o número de pedidos atualmente em processamento (i.e. iniciados e não concluídos), agrupados por método HTTP.

6. (4) Realize um componente React com as seguites propriedades: um *array* de *strings* `messages`, um número `period`, e um componente `component`.

   Este componente deve apresentar cada mensagem do *array* `messages` durante `period` milisegundos, avançando automaticamente para a próxima mensagem depois de decorrido esse tempo. A seguir à última mensagem, o processo deve recomeçar na primeira mensagem.

   A apresentação das mensagens deve ser feita com recurso ao componente `component`, nomeadamente através da sua propriedade `text` com tipo `string`.

   O componente realizado deve reagir correctamente a mudanças nas suas propriedades.

Pedro Félix,
30 de junho de 2021