Instituto Superior de Engenharia de Lisboa
Licenciatura em Engenharia Informática e de Computadores
Programação na Internet
Verão de 2020/2021
Teste Global de Época Normal (2 de Julho de 2021) - Duração 1h30

GRUPO 1 [6 valores]

Classifique as seguintes afirmações de verdadeiras (V) ou falsas (F). Uma resposta classificada corretamente conta 0,25 valores, incorrectamente desconta 0,25 valores ao total do grupo, sem resposta conta 0 valores.

1. [1] Segundo as regras definidas no protocolo HTTP, para criar um recurso:
   ___ O método POST pode sempre ser usado.
   ___ O método GET pode ser usado mas não é aconselhável.
   ___ O método PUT pode ser sempre usado.
   ___ O método PUT pode ser usado quando o URI do recurso a criar é conhecido previamente.

2. [1] Um método HTTP idempotente significa que:
   ___ Repetindo o mesmo pedido várias vezes, o resultado é sempre o mesmo.
   ___ Repetindo o mesmo pedido várias vezes, é equivalente a ter realizado apenas um pedido.
   ___ O mesmo pedido pode ser realizado várias vezes.
   ___ O mesmo pedido repetido várias vezes não altera o estado da aplicação.

3. [1] A linguagem JavaScript:
   ___ É não tipificada, o que significa que não tem tipos.
   ___ É dinâmica, porque todas as operações são avaliadas em tempo de execução.
   ___ Tem características de uma linguagem funcional, nomeadamente o suporte para funções de ordem superior.
   ___ É uma linguagem orientada a objetos, porque inclui suporte para objectos.

4. [1] A plataforma Node.js:
   ___ É vocacionada para server-side, o que significa que não é possível criar aplicações cliente nesta plataforma.
   ___ É um ambiente de execução para código em JavaScript.
   ___ Acrescenta comportamento à linguagem JavaScript standard (aka ECMASCript).
   ___ Inclui um ambiente de web browser.

5. [1] O módulo Express:
   ___ Pode ser usada tanto no desenvolvimento de web sites, como de web APIs.
   ___ É uma framework para desenvolvimento de aplicações web em Node.js.
   ___ Permite registo de rotas independentemente do tipo de pedido.
   ___ É baseado em middlewares, que são funções que têm acesso aos objetos Request e Response.

6. [1] No contexto do *web browser*:
   ___ O HTML é uma linguagem para definir o aspeto das páginas web.
   ___ O HTML é uma linguagem para definir a estrutura e conteúdo das páginas Web.
   ___ Documento HTML é um sinónimo para página web.
   ___ Uma CSS pode ser aplicada de forma independente de um documento HTML.

GRUPO 2 [9 valores]

7. [2] Considere o seguinte excerto de código em JavaScript:

```
1  function Coll(items) {
2     this.removeItem = items.pop
3  }
4  const coll = new Coll([2,4,6,8,10])
5  console.log(coll.removeItem())
```

   a. [1] A instrução da linha 5 não apresenta o valor 10, como seria esperado, correspondente à remoção do último elemento do *array* `items`. Justifique.

   b. [1] Realize as alterações necessárias ao código, de modo a que o método `removeItem` tenha o comportamento esperado, ou seja, remova do *array* `items` o elemento que está na última posição.

(continua)

8. [5] Considere o excerto de código em node.js, onde `mwGen` é uma função geradora de *middlewares* que, quando executado, retorna ao cliente a string recebida como parâmetro.

```
1  const PORT = process.argv[2] || 1904
2  const app = require('express')()
3  app.use("/a/b", mwGen("mw1"))
4  app.use("/a/:b", mwGen("mw2"))
5  app.use("/a/c", mwGen("mw3"))
```

a. [1.5] Executanto os seguintes comandos curl, obtemos as respectivas respostas. Justifique-as.

	Pedido	Resposta
1	curl localhost:1904/a/a	mw1
2	curl localhost:1904/a/b	mw2
3	curl localhost:1904/a/c	mw2

b. [2.5] Implemente a função `mwGen` de modo a que o *middleware* por ele gerado retorne para o cliente a string recebida como parâmetro.

c. [1] Indique que pedido poderia ser realizado de modo a que a resposta fosse `mw3`, sem ~~alterar~~ o código anterior.

9. [2] Indique, justificando, o erro grave presente na função get e apresente uma versão corrigida:

```
1  const urllib = require('urllib')
2  function get(url, cb) {
3    let result = null;   let err = null;
4    urllib(url, (err, data, res) => { err = err; result = data })
5    while (!(result || err) ) /* waiting*/
6    cb(err, result)
   }
```

GRUPO 3 [5 valores]

10. [5] Considere a aplicação TRINKAS desenvolvida no trabalho prático de PI.

a. [1,5] Pretende-se adicionar a funcionalidade de associar uma nota (com comentários) a uma receita presente num grupo. Especifique o recurso a disponibilizar na API da componente servidora da aplicação para suportar esta funcionalidade. Na definição do recurso, indique o que for necessário para que quem usa a API tenha a informação para aceder a esse recurso e conseguir adicionar apenas uma nota a uma receita presente num grupo, bem como para saber identificar e interpretar as situações de erro.
NOTAS:
- O pedido tem o formato Json
- A resposta tem o formato Json, quer indique conclusão com sucesso ou não.
- Os erros suportados são: pedido inválido, recurso não encontrado e erro no servidor.

b. [1] Tendo em consideração a estrutura de módulos aconselhada para o trabalho prático, que módulos teriam que ser alterados para implementar esta funcionalidade? Justifique.

c. [2,5] Implemente o método `addNoteToReceipe(req, rsp)` do módulo `trinkas-web-api`, que chama o método `addNoteToReceipe(groupId, receipeId, note)` do módulo `trinkas-services`. Este último recebe o identificador do grupo em `groupId`, o identificador da receita em `receipeId` e um string com a nota em `note`.
NOTA: Não implemente o método `addNoteToReceipe(groupId, receipeId, note)`. Assuma que este já está implementado.

O docente,
Luís Falcão