

**GRUPO 1 [7 valores]**

1. [1,5] A especificação do protocolo HTTP indica que o método DELETE é não seguro e idempotente. Justifique.
2. [1,5] Indique o *status code* a usar em cada uma das seguintes situações:
  - a. [0,25] Criação de recurso com sucesso
  - b. [0,25] Actualização de recurso com sucesso
  - c. [0,25] Recurso não encontrado
  - d. [0,25] *Query String* com formato incorreto
  - e. [0,25] Recurso mudou de localização
  - f. [0,25] Erro no servidor
3. [1,5] O que é um seletor CSS? Apresente exemplos de 3 tipos de seletores diferentes.
4. [1,5] Considere o seguinte URL: <http://somehost:8080/api/#users?page=5&limit=5>. Sobre este, responda às seguintes questões, justificando:
  - a. [0,75] O URL está mal formado?
  - b. [0,75] Se o parâmetro page for alterado para 6 o browser vai fazer um novo pedido HTTP?
5. [1] Para cada uma das funções seguintes, indique justificando, se têm uma implementação síncrona ou assíncrona.

```
function a(s1,s2, cb){ cb(s1 + s2) }

function b(s1,s2,cb){ setTimeout(() => cb(s1 + s2), 20) }

function c(s1,s2){return s1 + s2}
```

**GRUPO 2 [6 valores]**

6. [4,5] Considere o módulo Node *filter-request.js*. Este módulo exporta uma função que retorna um *middleware* compatível com o Express. O objetivo deste *middleware* é filtrar o *body* dos pedidos, de modo que o objeto ou objetos recebidos apenas contenham as propriedades passadas à função exportada.

NOTAS:

Considere que já existem as funções:

- *filterProperties(propNames, obj)* que recebe um *array* de *strings* em *propNames* e um objeto em *obj*, retornando um novo objeto com as propriedades de *obj* cujos nomes estão presentes em *propNames*.
- *filterPropertiesN(propNames, objs)* que recebe um *array* de *strings* em *propNames* e um *array* de objetos em *objs*, retornando um novo *array* de objetos correspondente à aplicação da função *filterProperties* com *propNames* a cada um dos elementos de *objs*.

NOTAS:

- *obj instanceof Array* permite determinar se *obj* é uma instância de *Array*
- Exemplo de utilização do módulo:

```
const app = require('express')()
const filterReq = require('./filter-request.js')(['a', 'c', 'e'])
// Other Express initialization code
app.use(filterReq)
```

- Considere os exemplos seguintes:

Body: { "a": 123, "b": 234, "c": 345, "d": 456 }

Body produzido para os outros *middlewares*: { "a": 123, "c": 345 }

Body: [ { "a": 123, "b": 234, "c": 345, "d": 456 }, { "a": "aa", "b": "bb", "c": "cc", "d": "dd" } ]

Body produzido para outros *middlewares*: [ { a: 123, c: 345}, { a: "aa", c: "cc" } ]

- a. [3] Implemente o módulo *filter-request.js*.

- b. [1.5] Considere que o *middleware* foi utilizado conforme o exemplo anterior e foi realizado o pedido HTTP apresentado abaixo. Constatou-se que, para este pedido, o *body* produzido para os outros *middlewares* foi `{ }`. Justifique a causa deste comportamento.

```
POST /api/resource HTTP/1.1
Host: somehost
Content-Length: 43

{ "a": 123, "b": 234, "c": 345, "d": 456 }
```

7. [1.5] Considere a listagem abaixo. Indique e justifique as mensagens apresentadas na consola chamando `test(true)` e `test(false)`.

```
function createPromise(success) {
    return success ? Promise.resolve(200) : Promise.reject(500)
}

function process(resolved) {
    return createPromise(resolved)
        .then(v => console.log(v + " Success"))
        .catch(v => console.log(v + " Error"))
}

function test(resolved) {
    process(resolved)
        .then(() => console.log("Finished with Success"))
        .catch(() => console.log("Finished with Error"))
}
```

### GRUPO 3 [7 valores]

4. [7] Pretende-se acrescentar à aplicação CIBORG, desenvolvida no trabalho prático, a funcionalidade de remoção de um utilizador. Quando um utilizador é removido, todos os dados que lhe estão associados são também removidos, incluindo as grupos de jogos por este criados.

O endpoint para a remoção é o seguinte:

Pedido

```
DELETE /api/users/:id-user
- Request:
  - Path parameters:
    - {id-user} - User unique identifier
  - Body: none
- Response:
  - Success:
    - Status code: 200
    - Content-Type: application/json
    - Body:
      {
        "status": "OK",
        "description": "User with name lfalcao and id 1 removed and all its data"
      }
  - User Not Found:
    - Status code: 404
    - Content-Type: application/json
    - Body:
      {
        "status": "NOT FOUND",
        "description": "User with name lfalcao and id 888 not found"
      }
  - Other errors (default Express behaviour):
    - Status codes: 400 and 500
    - Body: none
```

- a. [1,5] Implemente o método `deleteUser(req, rsp)` do módulo `ciborg-web-api`, que chama o método `deleteUser(userId)` do módulo `ciborg-services`, que recebe o identificador do utilizador em `userId` e retorna uma `Promise` com o mesmo objeto retornado pelo método `deleteUser` de `ciborg-db` (ver descrição deste método na alínea seguinte).
- b. [2,5] Implemente o método `deleteUser(userId)` do módulo `ciborg-services`. Na implementação deste método use os seguintes métodos do módulo `ciborg-db`:
  - `deleteGroupsOfUser(userId)`: Recebe o identificador do utilizador em `userId` e retorna um array de `Promise` sem qualquer valor.
  - `deleteUser(userId)`: Recebe o identificador do utilizador em `userId` e retorna uma `Promise` com um objeto que tem as propriedades `userId` e `username`, com o identificador e o nome do utilizador removido, respectivamente.. Caso o utilizador não seja encontrado, o objeto com que a `Promise` é resolvida tem a propriedade `error` com o valor "NOT FOUND".
- c. [3] Considere o Anexo 1. Na tabela 1, a imagem à esquerda e acima representa uma página HTML com a listagem dos utilizadores e abaixo a janela de diálogo que aparece quando se prime um dos botões "Delete User" (neste exemplo o 1º). À direita está um excerto do HTML dessa página.

A listagem 1 inclui excertos do código em JavaScript de cliente necessário para que, quando se prime um dos botões Delete User, seja mostrada uma janela de diálogo igual à da figura, para o utilizador correspondente. Caso o utilizador prima Yes, é realizado o pedido à API para remover o utilizador e é removida a linha na tabela correspondente a esse utilizador, caso contrário nada acontece. Complete o código nos locais assinalados com TO DO.

Duração: 2 horas  
ISEL, 10 de Fevereiro de 2020

## Anexo 1

Tabela 1

Username	Actions
User1	<button>Delete user</button>
User2	<button>Delete user</button>
User3	<button>Delete user</button>

Delete user with id 1

Are you sure you want to delete User1?

Yes Cancel

```
...<table class="table table-striped"><tr><th>Username</th><th>Actions</th></tr><tr><th id="user1">User1</th><th><button class="deleteBtn" id="1" data-toggle="modal" data-target="#exampleModal">Delete user</button></th></tr><tr><th id="user2">User2</th><th><button class="deleteBtn" id="2">Delete user</button></th></tr><tr><th id="user3">User3</th><th><button class="deleteBtn" id="3">Delete user</button></th></tr></table>...
```

Listagem 1

```
window.onload = function () {
    const deleteButtons = document.querySelectorAll(".deleteBtn")

    deleteButtons.forEach(btn => {
        // TO DO 1
    });

    function deleteUser() {
        const userId = this.id;
        // TO DO 2
        async function deleteUserOnServer() {
            // TO DO 3
            function processResponse(response) {
                if (response.status == 200) {
                    removeUser(id)
                } else {
                    // TO DO 4
                }
            }
        }
    }

    /**
     * Mostra uma janela de diálogo.
     * @title - Título da janela
     * @text - Texto da janela
     * @cbOk - Função a chamar caso seja premido o botão Yes
     * @cbCancel - Função a chamar caso seja premido o botão Cancel
     */
    function showModal(title, message, okCb, cancelCb) { // Função já implementada. }
    /**
     * Removes the table row for the user with the given id
     */
    function removeUser(id) { // Função já implementada. }
    /**
     * Shows the given message in a status area inside the page.
     */
    function showErrorMessage(message) { // Função já implementada. }
}
```