Projeto 1 – Aplicação Back-end REST API

1. Objetivos

O objetivo deste projeto é desenvolver uma REST API totalmente funcional de suporte a uma aplicação Front-end. A API permitirá aos utilizadores realizar operações CRUD (Create, Read, Update, Delete) em diferentes *endpoints* e será desenvolvida em Javascript com o recurso à *framework NodeJs + ExpressJS*. Para armazenamento e manipulação de dados, será utilizada a base de dados MongoDB, com uso de Queries aos documentos.

- Criar servidor com Node.js e Express.js
- Conectar a base de dados Mongo ao servidor
- Implementar os endpoints da API listados abaixo
- Testar os pedidos HTTP através da aplicação POSTMAN (https://www.postman.com/)

2. Componentes de avaliação

Aplicação Backend (20%) desenvolvida em grupo:

- Relatório em grupo sobre o trabalho realizado por cada aluno;
- Qualidade e organização do código do projeto;
- Correta definição de *routes*;
- Qualidade da solução de programação implementada;
- Correta definição das Queries MongoDB;
- Cumprimentos de todas as funcionalidades;
- Correta definição de funções;
- Correto tratamento de erros, com especificação de código de erro;
- API Documentation;
- Discussão.

3. Datas de entrega

Os alunos devem submeter as diferentes partes do projeto no Moodle e realizar as demonstrações nas seguintes datas:

- Backend Project Upload (código e documentação API) até dia **2 Novembro às 23h59;**

- Apresentação, demo e discussão individual dia **5 de Novembro.**

5. Datasets

Grupo 1 e Grupo 7	Dataset 1
Grupo 2 e Grupo 8	Dataset 2
Grupo 3 e Grupo 11	Dataset 3
Grupo 4 e Grupo 12	Dataset 4
Grupo 5	Dataset 5

4. API endpoints para avaliação

#	Route	Endpoint	Description
1	GET	/events	Lista de eventos com paginação.
2	GET	/users	Lista de *users* com paginação.
3	POST	/events	Adicionar 1 ou vários eventos
4	POST	/users	Adicionar 1 ou vários utilizadores
5	GET	/events/:id	Pesquisar evento pelo *_id* Resposta deverá incluir toda a informação do evento e o *average score*
6	GET	/users/:id	Pesquisar *user* pelo *_id* Incluir na resposta o top 3 eventos do utilizador
7	DELETE	/events/:id	Remover evento pelo *_id*
8	DELETE	/users/:id	Remover user pelo *_id*
9	PUT	/events/:id	Update evento

10	PUT	/users/:id	Update user
11	GET	/events/top/:limit	Lista de eventos com maior *score* (pela média), por ordem descendente. Mostrar na resposta toda a informação do evento. Limitar o total de eventos na resposta por {limit}
12	GET	/events/ratings/:order	Lista de eventos ordenado pelo número total de *reviews* :order - “asc” or “desc”
13	GET	/events/star	Lista de eventos com mais 5 estrelas. Mostrar toda a informação do evento e o número de reviews igual a 5
14	GET	/events/:year	Lista de eventos avaliados no ano {year}
15	POST	/users/:id/review/:event_id	Adicionar uma nova *review* a um evento. :id - user ID; :event_id – evento ID
#	A definir pelo grupo		Criar 4 endpoints relacionados com o dataset de cada grupo. Será tido em consideração a ideia adequada ao dataset; correta implementação; pertinência e utilidade dos endpoints; etc.
Nota: Todos os endpoints que retornem mais do que 20 documentos devem ter paginação.			