Ficha engweb2024-recurso

Avaliação: Engenharia Web

6 de Junho de 2024, 14h, Ed.2 - sala 2.09

Licenciatura em Engenharia Informática (3º ano)

Sinopsis

O objectivo principal deste teste é avaliar os conhecimentos obtidos durante as aulas no desenvolvimento de aplicações Web e outras tarefas afins.

Antes de começares, lê atentamente até ao fim para ficares com uma percepção do todo que se pretende. Vais ver que **tomarás decisões mais acertadas depois de uma leitura completa**.

Os resultados finais deverão ser enviados ao docente da seguinte forma:

- Enviar por email para: `jcr@di.uminho.pt`;
- Colocar no subject/assunto: `ENGWEB2024::Recurso::Axxxxx`, em que `Axxxxx` corresponde ao número do aluno;
- Enviar ao docente um link do github para um repositório novo criado especificamente para o exame com o seguinte conteúdo (esta preparação poderá valer 1 valor do exame):
  - O repositório no GitHub deverá chamar-se `ENGWEB2024-Recurso`;
  - Dentro do repositório deverá haver um ficheiro, `PR.md`, contendo uma descrição de como fez a persistência de dados, do setup de bases de dados, respostas textuais pedidas, instruções de como executar as aplicações desenvolvidas, descreva as ações necessárias para quem estiver de fora poder arrancar as aplicações, etc...
  - Dentro do repositório deverão existir duas pastas: `ex1`, onde colocarão a aplicação desenvolvida para responder ao primeiro exercício e, `ex2`, onde colocarão a aplicação desenvolvida para responder ao segundo exercício.

Os exercícios que envolvam criação de rotas serão testados com as rotas no enunciado, qualquer rota que seja diferente da pedida será avaliada com 0.

Ambas as aplicações serão colocadas em execução recorrendo aos seguintes comandos:

```
$ npm i
$ npm start
```

Alternativamente, se resolveres implementar o último ponto, tudo será colocado em execução com o comando:

```
$ docker-compose up -d
```

Prepara e configura as tuas aplicações para funcionarem desta forma.

Recursos

Recursos para a realização da prova:

- Lista de livros, este ficheiro tem a seguinte estrutura:


```
[
    {
        "bookId": "2767052-the-hunger-games",
        "title": "The Hunger Games",
        "series": "The Hunger Games #1",
        "author": "Suzanne Collins",
        "rating": "4.33",
        "description": "WINNING MEANS FAME AND FORTUNE.LOSING MEANS
CERTAIN DEATH.THE HUNGER GAMES HAVE BEGUN. . . .In the ruins of a place
once known as North America lies the nation of Panem, a shining Capitol
surrounded by twelve outlying districts. The Capitol is harsh and cruel
and keeps the districts in line by forcing them all to send one boy and
once girl between the ages of twelve and eighteen to participate in the
annual Hunger Games, a fight to the death on live TV.Sixteen-year-old
Katniss Everdeen regards it as a death sentence when she steps forward to
take her sister's place in the Games. But Katniss has been close to dead
before\u2014and survival, for her, is second nature. Without really
meaning to, she becomes a contender. But if she is to win, she will have
to start making choices that weight survival against humanity and life
against love.",
        "language": "English",
        "isbn": "9780439023481",
        "genres": "['Young Adult', 'Fiction', 'Dystopia', 'Fantasy',
'Science Fiction', 'Romance', 'Adventure', 'Teen', 'Post Apocalyptic',
'Action']",
        "characters": "['Katniss Everdeen', 'Peeta Mellark', 'Cato (Hunger
Games)', 'Primrose Everdeen', 'Gale Hawthorne', 'Effie Trinket', 'Haymitch
Abernathy', 'Cinna', 'President Coriolanus Snow', 'Rue', 'Flavius',
'Lavinia (Hunger Games)', 'Marvel', 'Glimmer', 'Clove', 'Foxface',
'Thresh', 'Greasy Sae', 'Madge Undersee', 'Caesar Flickerman', 'Claudius
Templesmith', 'Octavia (Hunger Games)', 'Portia (hunger Games)']",
        "bookFormat": "Hardcover",
        "edition": "First Edition",
        "pages": "374",
        "publisher": "Scholastic Press",
        "publishDate": "09/14/08",
        "firstPublishDate": "",
        "awards": "['Locus Award Nominee for Best Young Adult Book
(2009)', 'Georgia Peach Book Award (2009)', 'Buxtehuder Bulle (2009)',
'Golden Duck Award for Young Adult (Hal Clement Award) (2009)', \"Grand
Prix de l'Imaginaire Nominee for Roman jeunesse \u00e9tranger (2010)\",
'Books I Loved Best Yearly (BILBY) Awards for Older Readers (2012)',
\"West Australian Young Readers' Book Award (WAYRBA) for Older Readers
(2010)\", \"Red House Children's Book Award for Older Readers & Overall
```

```
(2010)\", 'South Carolina Book Award for Junior and Young Adult Book
(2011)', 'Charlotte Award (2010)', 'Colorado Blue Spruce Young Adult Book
Award (2010)', 'Teen Buckeye Book Award (2009)', \"Pennsylvania Young
Readers' Choice Award for Young Adults (2010)\", 'Rhode Island Teen Book
Award (2010)', \"Dorothy Canfield Fisher Children's Book Award (2010)\",
'Evergreen Teen Book Award (2011)', 'Soaring Eagle Book Award (2009)',
'Milwaukee County Teen Book Award Nominee (2010)', 'Sakura Medal for
Middle School Book (2010)', 'Michigan Library Association Thumbs Up! Award
(2009)', 'Florida Teens Read (2009)', 'Deutscher Jugendliteraturpreis for
Preis der Jugendjury (2010)', 'Iowa High School Book Award (2011)', 'New
Mexico Land of Enchantment Award for Young Adult (2011)', 'Eliot Rosewater
Indiana High School Book Award (2010)', 'The Inky Awards for Silver Inky
(2009)', 'California Young Readers Medal for Young Adult (2011)', 'Lincoln
Award (2011)', 'Kinderboekwinkelprijs (2010)', 'Missouri Truman Readers
Award (2011)', 'CYBILS Award for Young Adult Fantasy & Science Fiction
(2008)', 'Literaturpreis der Jury der jungen Leser for Jugendbuch (2010)',
'The Inky Awards Shortlist for Silver Inky (2009)', 'Prix Et-lisez-moi
(2011)', 'Missouri Gateway Readers Award (2011)', 'Oklahoma Sequoyah Award
for High School and Intermediate (2011)', 'Premio El Templo de las Mil
Puertas for Mejor novela extranjera perteneciente a saga (2009)',
\"Rebecca Caudill Young Readers' Book Award (2011)\", 'LovelyBooks
Leserpreis for Fantasy (2009)', 'LovelyBooks Leserpreis for Bestes
Cover/Umschlag (2009)', 'Premi Protagonista Jove for Categoria 13-14 anys
(2010)']",
        "numRatings": "6376780",
        "ratingsByStars": "['3444695', '1921313', '745221', '171994',
'93557']",
        "likedPercent": "96",
        "setting": "['District 12, Panem', 'Capitol, Panem', 'Panem
(United States)']",
        "coverImg": "https://i.gr-
assets.com/images/S/compressed.photo.goodreads.com/books/1586722975l/27670
52.jpg",
        "bbeScore": "2993816",
        "bbeVotes": "30516",
        "price": "5.09"
    },
    ...
]
```


Exercício 1: Livros (API de dados)

Neste exercício, irás implementar uma API de dados sobre o dataset fornecido. Encontra-se dividido em 3 partes.

1.1 Setup [1val.]

Realiza as seguintes tarefas sem alterares os nomes da base de dados e coleção fornecidos:

- Analisa o dataset fornecido;
- Introduz as alterações que achares necessárias no dataset. Este resultou de um processo de conversão e precisa de algumas limpezas, há campos que são listas e estão como string por

exemplo: tem atenção ao `id` do registo, às listas de géneros e de personagens, ao campo autor que pode conter referências a várias pessoas, etc;
- Importa-o numa base de dados em MongoDB com os seguintes parâmetros:
  - database: `-d livros`
  - collection: `-c livros`
- Testa se a importação correu bem.

1.2 Queries (warm-up) [0.5+0.5+1+1+1 = 4val.]

Especifica queries em MongoDB para responder às seguintes questões:

1. Quantos livros têm a palavra `Love` no título;
2. Quais os títulos dos livros, em ordem alfabética, em que um dos autores tem apelido `Austen`?
3. Qual a lista de autores (ordenada alfabeticamente e sem repetições)?
4. Qual a distribuição de livros por género (`genre`) (quantos livros tem cada género)?
5. Quais os títulos dos livros e respetivos isbn, em ordem alfabética de título, em que um dos personagens (`characters`) é `Sirius Black`?

- Regista estas queries num ficheiro de texto que deverás colocar na pasta `ex1` chamado `queries.txt`.

1.3 API de dados [0.5+0.5+1+1+1+1+1+1+1 = 8val.]

Desenvolve agora uma API de dados, que responde na **porta 17000** e que responda às seguintes rotas/pedidos:

- `GET /books`: devolve uma lista com todos os registos;
- `GET /books/:id`: devolve o registo com identificador `id` (em `PR.md` deves indicar o que vais usar como id);
- `GET /books?charater=EEEE`: devolve a lista dos livros em que `EEEE` faz parte do nome de um dos personagens;
- `GET /books?genre=AAA`: devolve a lista dos livros associados ao género (`genre`) `AAA`;
- `GET /books/genres`: devolve a lista de géneros ordenada alfabeticamente e sem repetições;
- `GET /books/characters`: devolve a lista dos personagens ordenada alfabeticamente e sem repetições;
- `POST /books`: acrescenta um registo novo à BD;
- `DELETE /books/:id`: elimina da BD o registo com o identificador `id`;
- `PUT /books/:id`: altera o registo com o identificador `id`.

Antes de prosseguires, testa as rotas realizadas com o Postman ou similar.

Requisitos para o Exercício 2: API de dados ou json-server

Se por algum motivo não conseguiste obter uma API de dados a funcionar que te permita continuar a realizar este teste, podes avançar para p exercício 2 colocando o dataset num `json-server` e usando este como API de dados (terá uma valorização de 20%, ou seja, há uma redução de 80% relativamente à valorização do exercício 1).

Se optares por esta via, coloca o `json-server` a responder na **porta 17000**.

Exercício 2: Contratos (Interface) [2+2+2 = 6val.]

Tendo a API desenvolvida, desenvolve agora um novo serviço, que responde na **porta 17001**. Este serviço, **sempre que precisar de dados deverá solicitá-los à API de dados**, não deverá ir diretamente à base de dados em MongoDB.

Esta nova aplicação deverá reagir da seguinte forma:

1. Se colocares no browser o endereço `http://localhost:17001` deverás obter a página principal constituída por:
    - Um cabeçalho com metainformação à tua escolha;
    - Uma tabela contendo a lista de registos, um por linha, com os campos: `id`, `title`, `author`, `publishDate`, `pages`;
    - O campo `id` deverá ser um link para a página do livro com esse identificador;
    - O campo `author` deverá ser um link para a página desse autor (arranja maneira de associar um id ao autor).
2. Se colocares no browser o endereço `http://localhost:17001/:id` deverás obter a página do livro cujo identificador foi passado na rota:
    - Esta página deverá conter todos os campos do livro e um link para voltar à página principal;
    - Deverás também inluir a imagem referenciada no campo `coverImg`, se este existir no registo.
3. Se colocares no browser o endereço `http://localhost:17001/authors/:idAutor` deverás obter a página do autor cujo `id` corresponde ao parâmetro passado na rota :
    - Na página de cada autor deverá constar este identificador e o respetivo nome;
    - Uma tabela com a lista de livros desse autor (tabela com estrutura semelhante à da página principal);
    - O somatório do total de livros;
    - E um link para voltar à página principal.

Distribuição das aplicações [1val.]

Tendo chegado a este ponto, deverás ter as duas aplicações a funcionar. Para facilitar a sua distribuição e colocação em funcionamento deverás realizar os seguintes passos:

1. Criar uma especificação `Dockerfile` para cada uma;
2. A partir destas especificações criar as imagens locais das aplicações;
3. Criar uma especificação `docker-compose` que orquestre os três serviços: MongoDB, API de dados e Interface.
4. O serviço do MongoDB deverá ser interno, não deverá exteriorizar nenhuma porta;
5. Testar tudo.

Bom trabalho e boa sorte jcr