Week 7 Pair Exercises: Cat Cards

Context

Congratulations on your new job at *HazBro*. You'll be working on their latest project, *Cat Cards*. To facilitate development, the VP in charge of this project split the development staff into three teams: Database, Frontend, and Backend. The VP placed you on the Backend team.

The Frontend team developed their part of the application. However, they need you to provide the data so it works properly. They gave you the following API documentation:

- `GET /api/cards`: Provides a list of all Cat Cards in the user's collection.
- `GET /api/cards/{id}`: Provides a Cat Card with the given ID.
- `GET /api/cards/random`: Provides a new, randomly created Cat Card containing information from the cat fact and picture services.
- `POST /api/cards`: Saves a card to the user's collection.
- `PUT /api/cards`: Updates a card in the user's collection.
- `DELETE /api/cards`: Removes a card from the user's collection.

Cat Card JSON object structure

Here's the JSON object structure for a Cat Card:

```
{
    "id" : an integer that represents this particular card's unique identifier,
    "imgUrl" : "A string containing the full URL to the cat image",
    "fact" : "A string containing a cat fact",
    "caption" : "A string containing the caption for this particular card"
}
```

Cat Card collection example

Here's an example collection of Cat Cards:

```
[
    {
        "id" : 17,
        "imgUrl" : "https://purr.objects-us-east-1.dream.io/i/8M3AW.jpg",
        "fact" : "Cats sleep 70% of their lives.",
        "caption" : "Aww, this reminds me of Lefty! He slept CONSTANTLY."
    },

    {
        "id" : 38,
        "imgUrl" : "https://purr.objects-us-east-1.dream.io/i/image.jpeg",
        "fact" : "People who own cats have on average 2.1 pets per household,
whereas dog owners have about 1.6.",
```

```
        "caption" : "Bartender, I'll take a Salty *Cat*"
    }
]
```

Once you provide implementation for the above endpoints and return properly formatted JSON objects in the agreed-upon schema, the application works.

> Hint: Consider starting with the controller method that provides a new, randomly created card.

Database

The Database team provided you with a script to create a local database and the DAO files to retrieve the data. You should be able to implement your code without reviewing the implementation details of the DAO files. The interface provides you with enough information to complete your tasks.

Cat APIs

There are two web APIs that will provide you with sources of random cat fact and cat picture data.

You can use `https://random-cat-api.netlify.app/.netlify/functions/api/` to retrieve the URL of a random cat picture as a JSON object that looks like this:

```
{"file":"https:\/\/purr.objects-us-east-1.dream.io\/i\/VEcIJ.jpg"}
```

You should implement the `RestCatPicService` to call this endpoint and return the data as a `CatPic` object.

You can use `https://cat-fact.herokuapp.com/facts/random` to retrieve random cat facts as a JSON object that looks like this:

```
{"used":false,
 "source":"user",
 "type":"cat",
 "deleted":false,
 "_id":"5a038bb98e3dbc001f71978d",
 "updatedAt":"2020-05-10T20:20:11.457Z",
 "createdAt":"2018-01-30T21:20:04.144Z",
 "user":"5a9ac18c7478810ea6c06381",
 "text":"The domestic cat (Felis catus) is a small, typically furry, carnivorous
mammal.",
 "__v":0,
 "status":{"verified":true,"sentCount":1}}
```

You will find additional documentation for this endpoint at `https://alexwohlbruck.github.io/cat-facts/docs/endpoints/facts.html`.

You should implement the `RestCatFactService` to call this endpoint and return the data as a `CatFact` object.

Getting started

1. To install your database, run the command `database/create_db.sh`.

   Note: You'll see a message that says that the "catcards" table doesn't exist the first time you run the `create` script. You can ignore the message.

2. Launch this project by running it as a Spring Boot application and navigate to `http://localhost:8080`.