Exercises: Introduction to Java EE

Problems for exercises and homework for the "Java Web Development Basics" course @ SoftUni. Submit your solutions on the **course page** of the **current instance**.

Let's exercise a little the knowledge we've accumulated during the lesson for Java EE Introduction. As you have seen only the basics, and theory, the problems won't be that hard. But they will definitely be fun and productive.

We will implement a simple application, only with Servlets, exploring the most interesting exploits of the Servlet API.

Fluffy Duffy Munchkin Cats (FDMC)

Fluffy Duffy Munchkin Cats is an application in which you register cats, with several properties. It has many versions, and you will most probably see it several times.

You will have to create a simple multi-Servlet application which has several pages and 1 object entity.

1. Data

This is the data layer of the application. There is 1 data object for you to implement.

Cat

Create a **`Cat`** class, which holds the following properties:

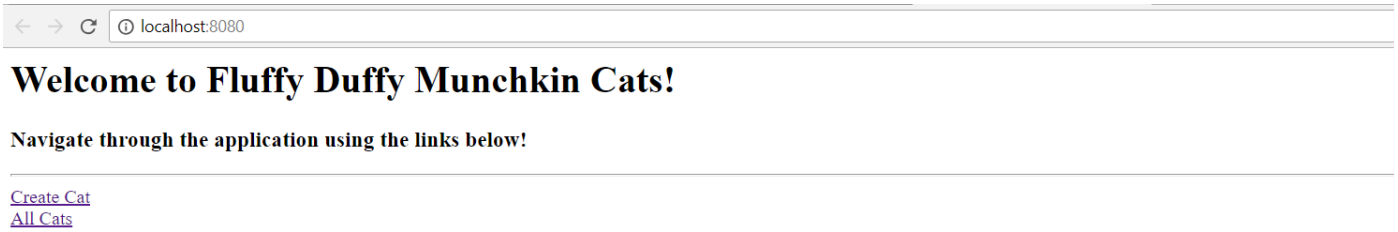
- **`name`** – a **String**.
- **`breed`** – a **String**.
- **`color`** – a **String**.
- **`numberOfLegs`** – an **Integer**.

2. Home Servlet

Implement a **Servlet**, which listens on the **index route** ("**`/`**").

It should support only a **GET** request.

It should return the following HTML page, upon a **GET** request.



There are **2 links** on this page, which must lead to the following **routes**:

Create Cat – "**`/cats/create`**"

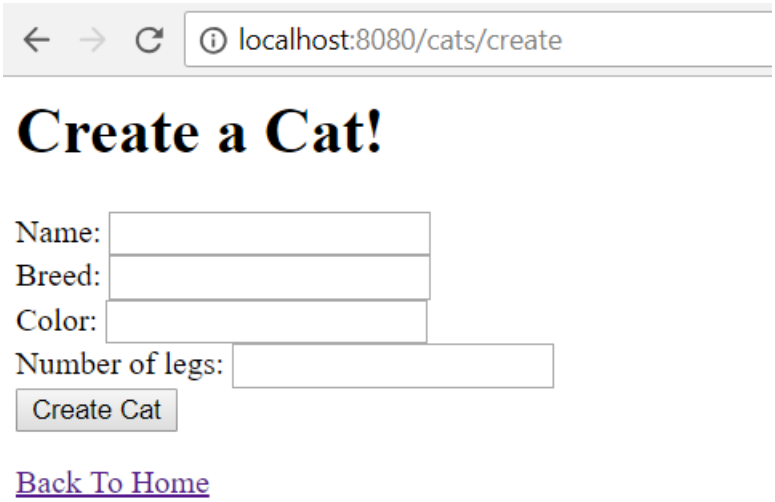
All Cats – "**`/cats/all`**"

3. Cat Create Servlet

Implement a **Servlet**, which listens on **route** ("**`/cats/create`**").

It should support **GET** & **POST** requests.

It should return the following HTML page, upon a **GET** request.



There is **1 link** on this page, which must lead to the following **route**:

Back To Home – "**/**"

As you can see there is a **form**, with several **input fields** on the page. That form should submit a **POST** request to the route ("**`/cats/create`**"). The **Cat Create Servlet** should handle the **POST** request, **instantiating** a **Cat** from it, and **populating** the **object** with the **provided data**.

Upon **creation** of a **Cat**, the **Servlet** should **redirect** to **route** ("**`/cats/profile?catName={catName}`**"), where the **catName** parameter is the **created Cat's name**.

4. Data Storage

We reached a point where we are processing **POST** requests. Most **POST** requests aim to **create data**, which should **be stored afterwards** in some kind of **data storage**.

However, we haven't seen much of how to implement data storage, when working with Servlets.
Find a way to store the created cats inside the application, however, there are some constraints:

1. You are **not allowed** to use any dedicated database, or ORM frameworks.
2. You are **not allowed** to use any back-end as a service, or API for storing data.
3. You are **not allowed** to use flat-files or anything like that.

Try to store the data **using only the help** of the **Servlet API**. Research a little about how to do that. There is this idea, that most of the answers are hidden in the first 3 StackOverflow results in Google, when you search for the right thing.

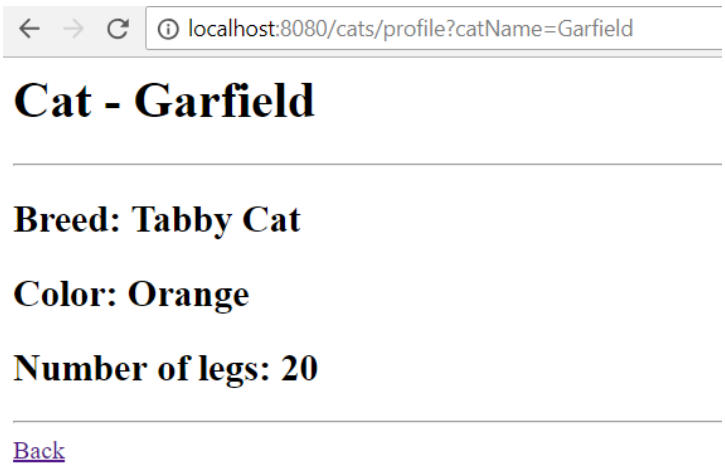
Hint: Google "**`Shared data between Servlets`**".

5. Cats Profile Servlet

Implement a **Servlet**, which listens on **route** ("**`/cats/profile`**").

It should support only a **GET** request.

It should return the following HTML page, upon a **GET** request.



There is **1 link** on this page, which must lead to the following **route**:

Back – "**`/cats/all`**"

The page should **render data** about **the** cat with **the given name** in the **query**. Extract that from the **request parameters**. The **route** will look like this ("**`/cats/profile?catName=Garfield`**").

The **cat** should be in the **data storage**.

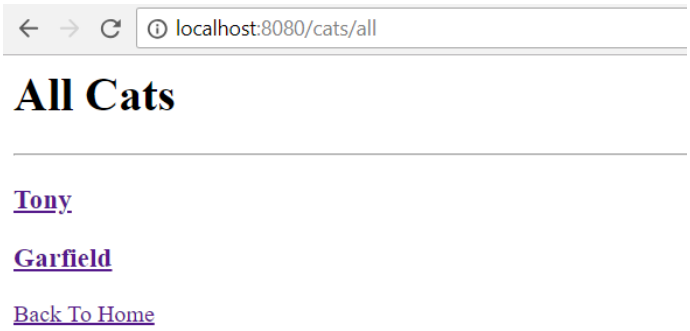
If there is **no cat**, with the **given name**, in the **data storage**, you should return the following page.

6. All Cats Servlet

Implement a **Servlet**, which listens on **route** ("**`/cats/all`**").

It should support only a **GET** request.

It should return the following HTML page, upon a **GET** request.



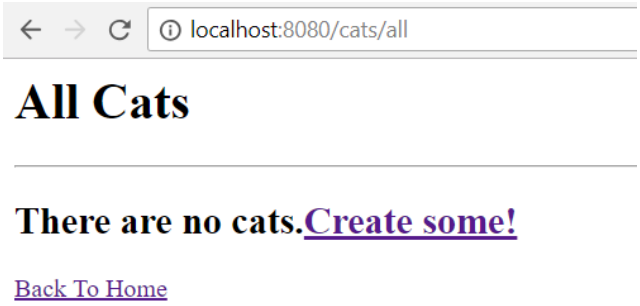
There is **1 navigational link** on this page, which must lead to the following **route**:

Back To Home – "**/**"

All of the created **Cats** in the **data storage**, should be rendered with their **names**, like it is shown above. Upon clicking on a **cat's name**, you should be **redirected** to its **profile page**. ("**`/cat/profile?catName={catName}`**").

The **cat's names** should be **links**.

If there are no cats, you should render the following HTML page:



There are **2 navigational links** on this page, which must lead to the following **routes**:

Create Some! – "**`/cats/create`**"

Back To Home – "**/**"