# JS Apps Exam – Pet My Pet Single Page Application

The app keeps **users** and **pets**. Users can **register**, **login**, **logout** , view a **dashboard** with all **pets** sorted by **likes** (a helper function will be given). They will also be able to view the pets by **category**, **add** a new pet, **edit/delete** their **own** pets, show other pets **details** and a **section** where they can view their **own** **pets**.

### NavBar (5 pts)

Implement a **NavBar** for the app: navigation links should correctly change the current screen (view).

* Clicking on the links in the **NavBar** should display the view behind the link (views are sections in the HTML code).
* Your application may **hide** by CSS (display: none) or **delete** from the DOM all unneeded elements or just display the views it needs to display.
* The given „**Dashboard,** **My Pets, Add Pet, Welcome and logout**" should be visible **only** for logged in users.

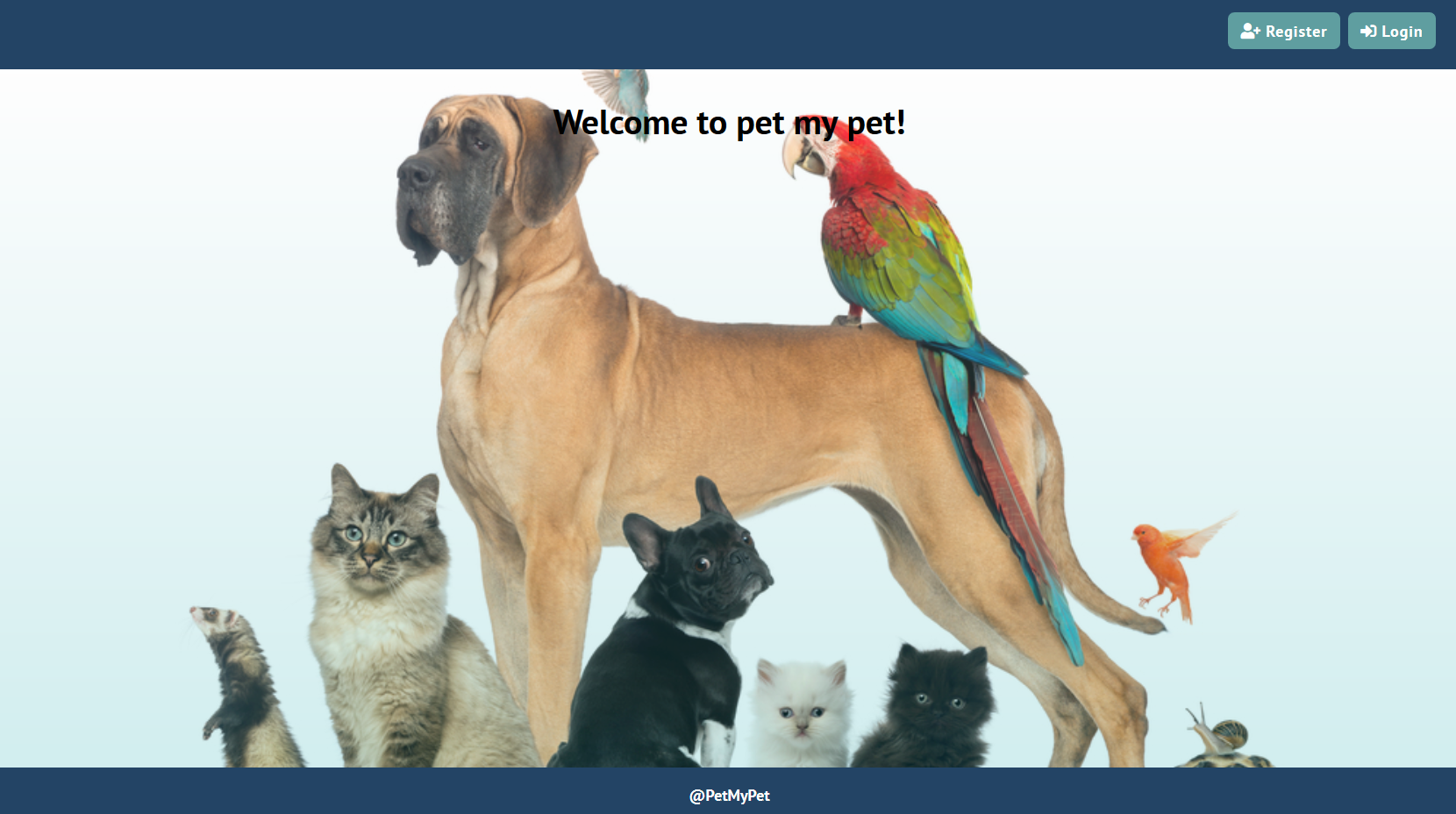


* Anonymous users can **only** view the **login and register**.



### Home Screen (5 pts)

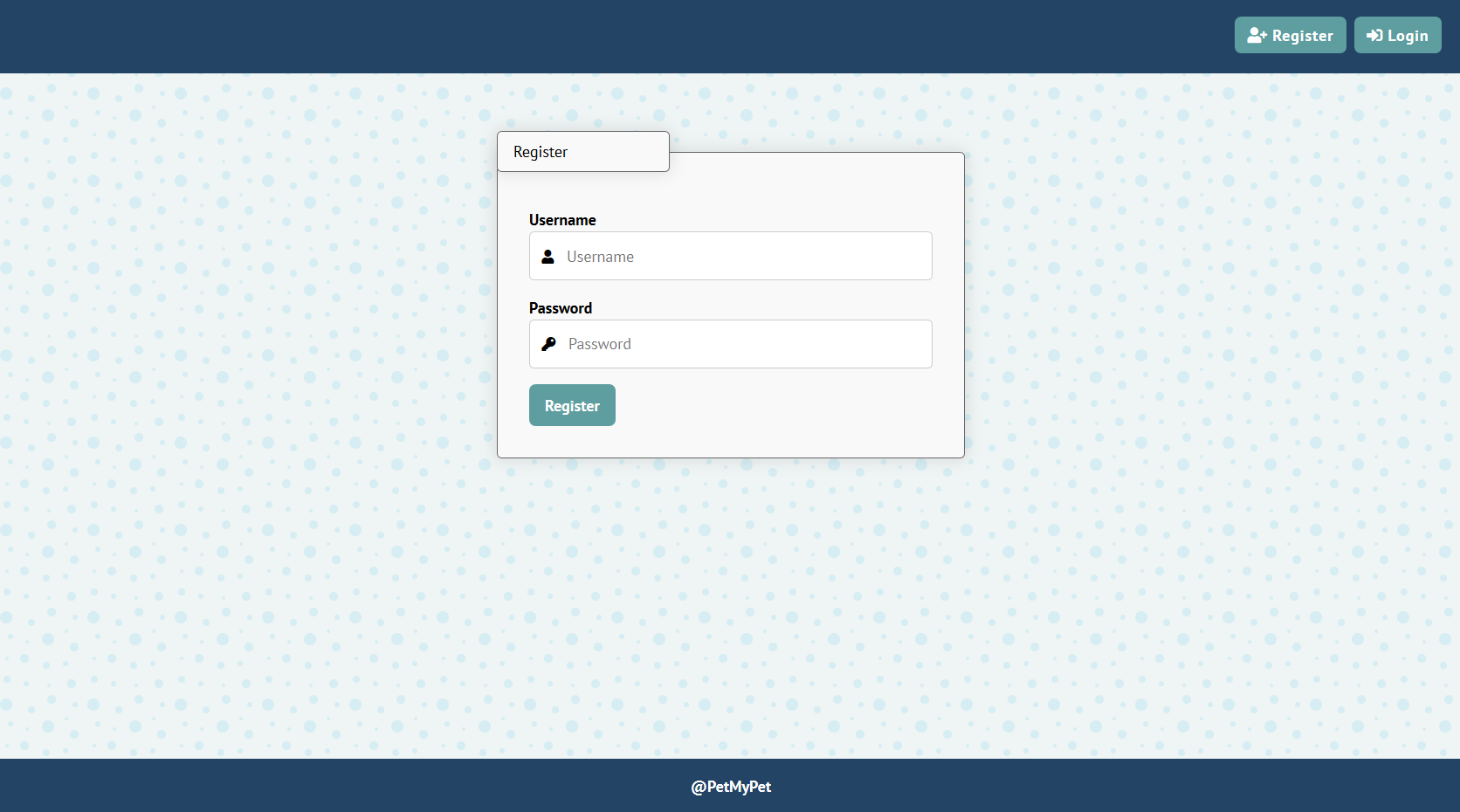
The initial screen should display the register, login and the initial image + footer.



### Register User (10 pts)

By given **username** and **password** the app should register a new user in the system.

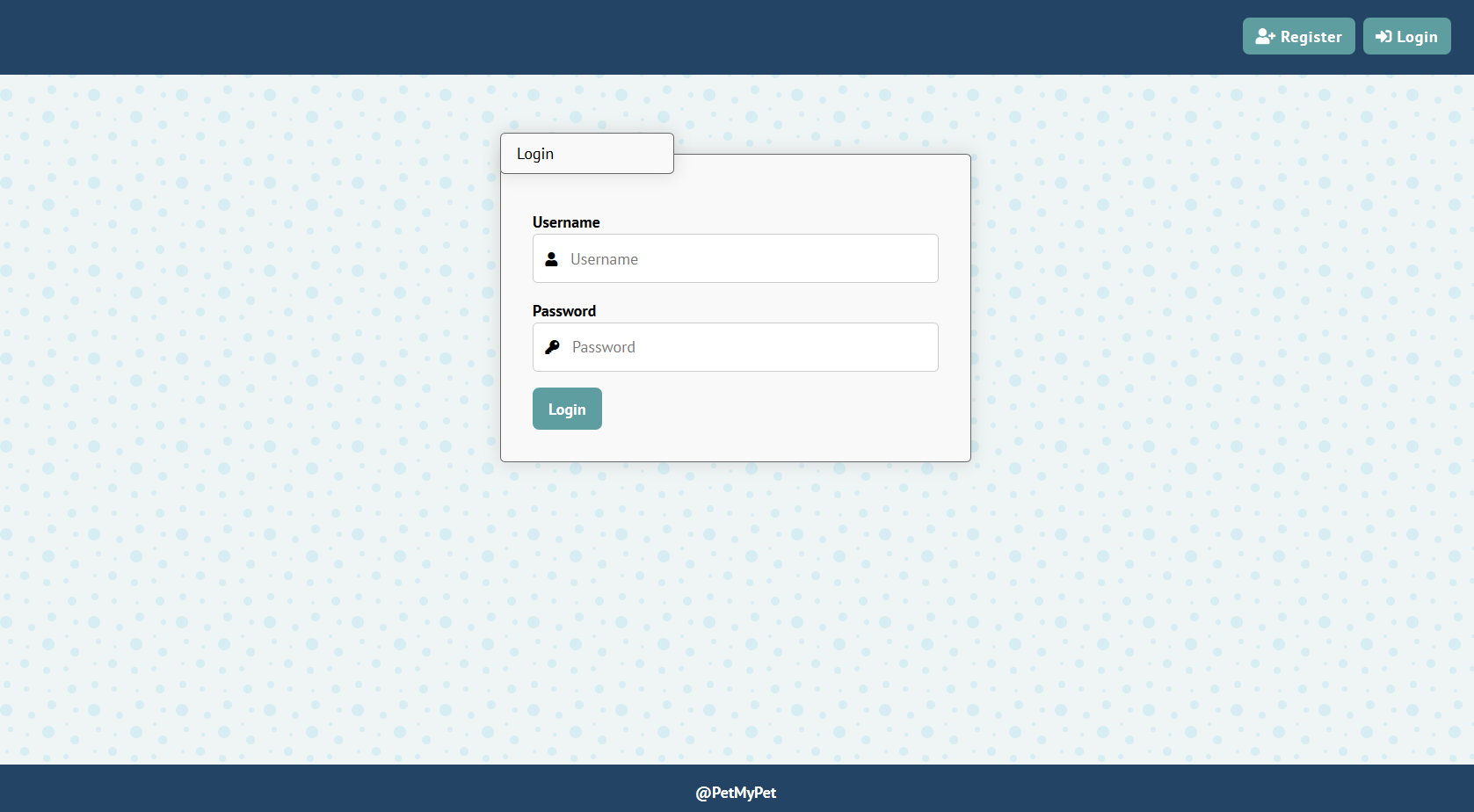
* The fields should be **non-empty**. The **username** must be **at least 3 symbols**, and the **password** **at least 6**.
* After a **successful registration**, a notification message **"User registration successful."** should be displayed and the **home page should be displayed again but with the right navbar**.
* In case of **error** (eg. invalid username/password), an appropriate error **message ("Username must be at least 3 symbols", "Password must be at least 6 symbols")** should be displayed and the user should be able to **try** to register again.
* Keep the user session data in the browser’s **session storage**.



### Login User (5 pts)

By given **username** and **password** the app should be able to login an existing user.

* The validations of the fields must be as in **"Register"**
* After a **successful login**, a notification message **"Login successful."** should be displayed and the user home screen should be displayed.
* In case of **error**, an appropriate error message should be displayed and the user should be able to fill the login form again.
* Keep the user session data in the browser’s **session storage**.
* Clear **all** input fields after **successful** login.



### Logout (5 pts)

Successfully logged in user should be able to **logout** from the app.

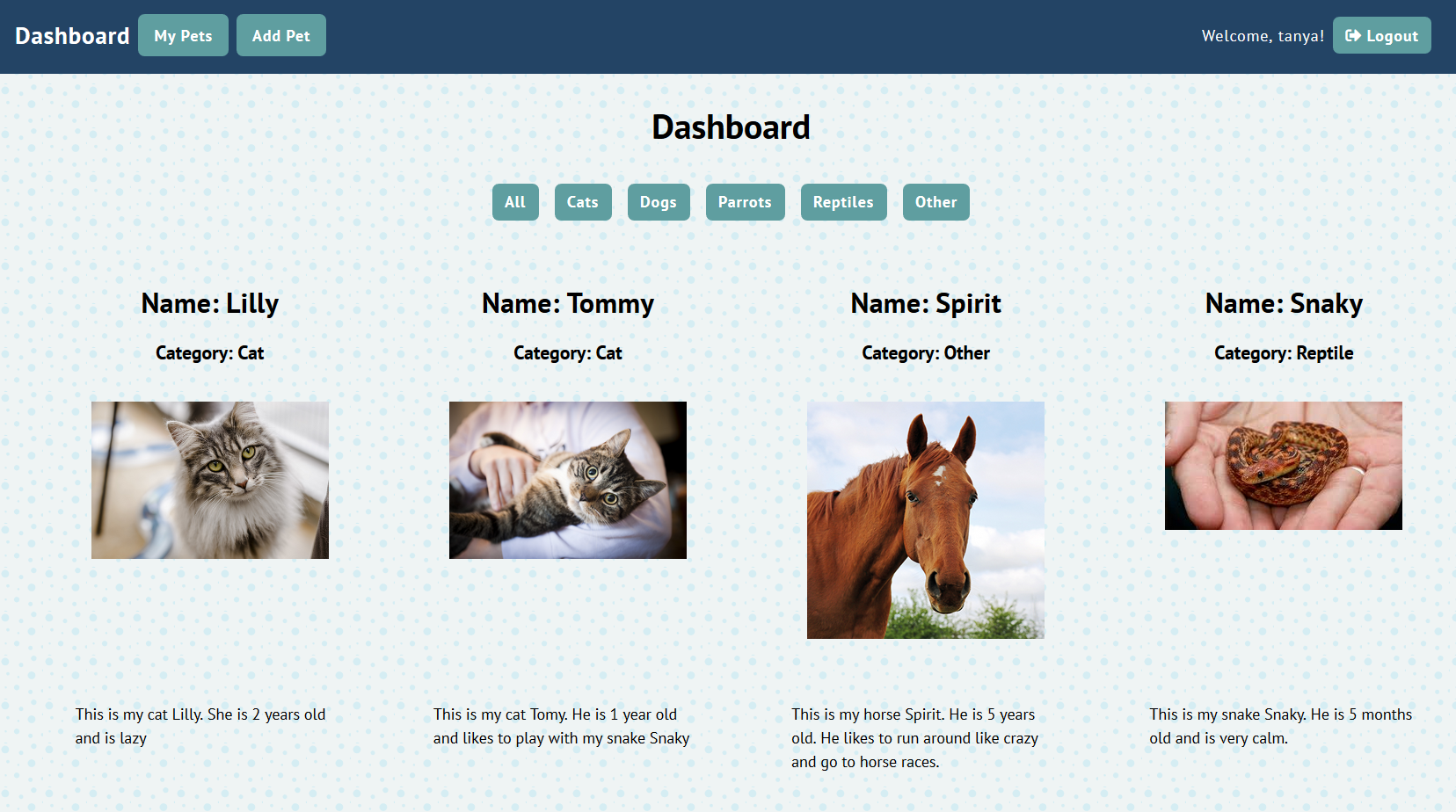
* After a **successful** logout, a **notification** message **"Logout successful."** should be displayed.
* After successful logout, the **anonymous screeen** should be shown.
* The **"logout" REST service** at the back-end should be obligatory called at logout.
* All local information in the browser (**user session data**) about the current user should be deleted.

### Logged In Dashboard (30 pts)

#### All Pets (15 pts)

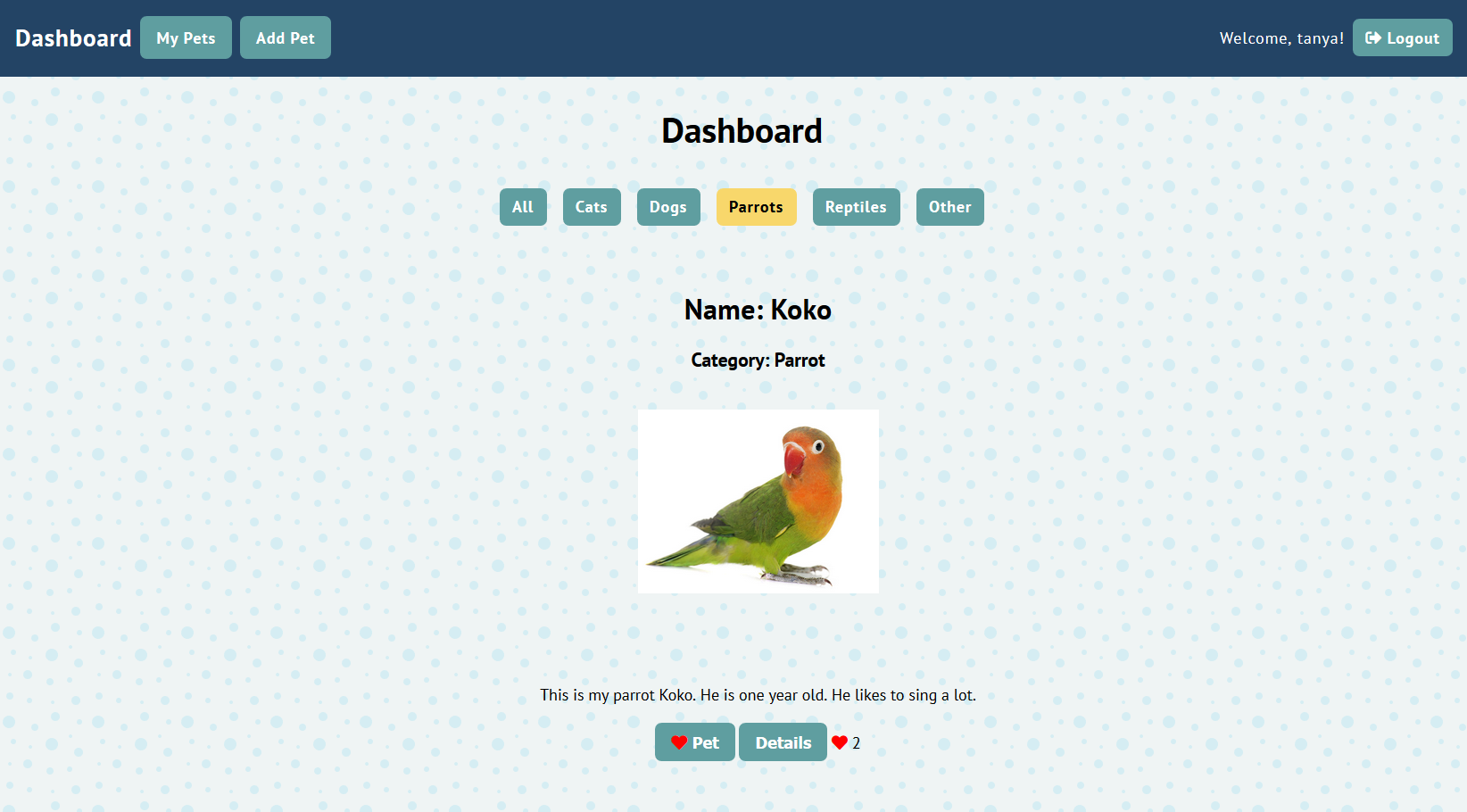
Successfully logged users should be welcomed by the **home screen**. They should also be able to see the dashboard, which must have the following:

* The **pets** should be listed in the **format** as shown in the Web design (see the screenshot below).
* Only the pets that are **not created by the current user** should be listed!
* Each **pet** has **name**, **image,** **description**, **category**.
* When the **details** link is clicked, your app should **redirect** to the details **section** of the pet.
* When the **pet** link is clicked, the **"pet counter" of the pet should be increased (it should start from 0)**



#### Categories (15 pts)

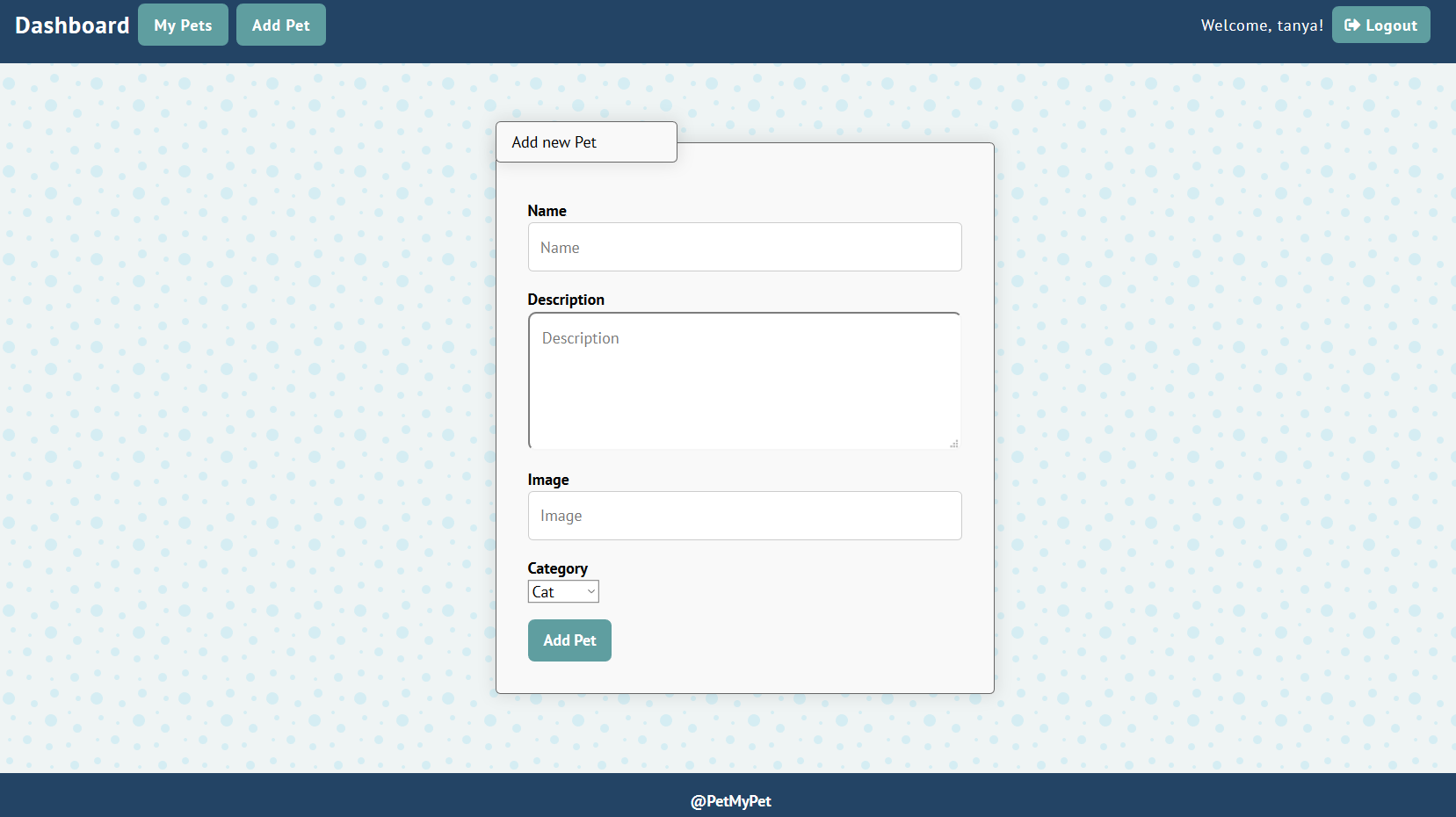
* When clicking on some of the **categories**, the pets should be **filtered by the selected category** and **oredered by their "pet counter" in descending**



### Add Pet Screen (10 pts)

Logged in users can **add** pets. Clicking the **[Add Pet]** link should open a form.

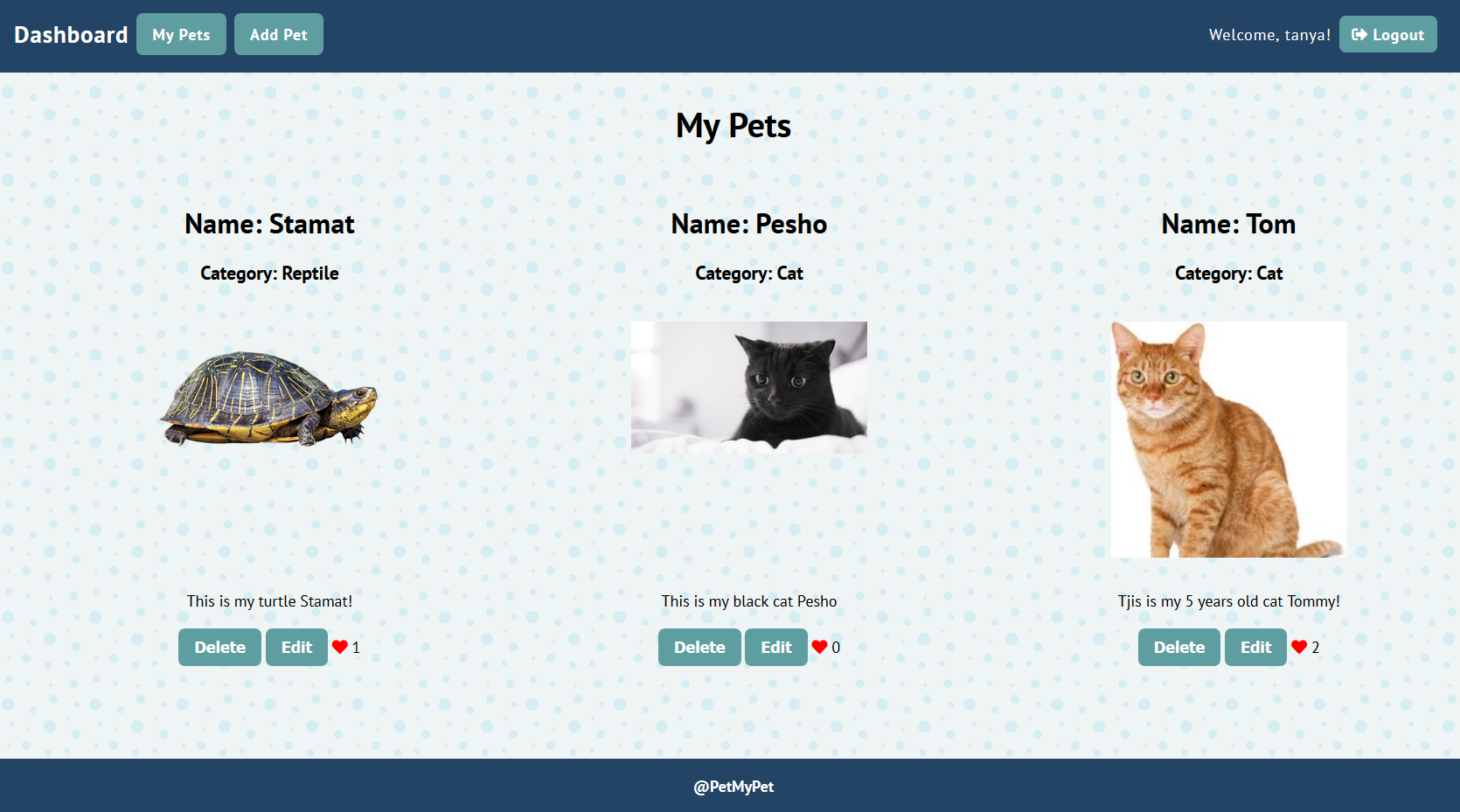
* After a **successful** pet creation, a notification message **"Pet created."** should be displayed and the **home page** should be shown.



### My Pets (10 pts)

Each **user** should be able to view his own pets by clicking [**My Pets**].

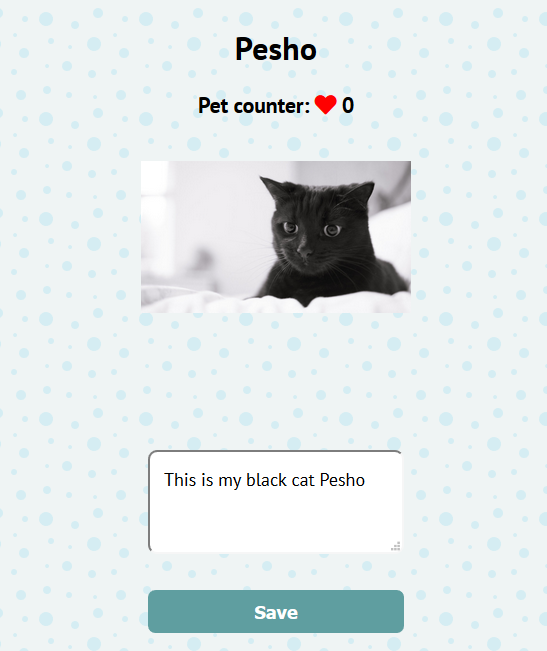
* The **pets** should be listed like in the **dashboard** section, except the **"Pet"** button is replaced with **"Delete"** button (users cannot pet their own pets) and the **"Details"** button **is now "Edit"** button. **No ordering needed**.



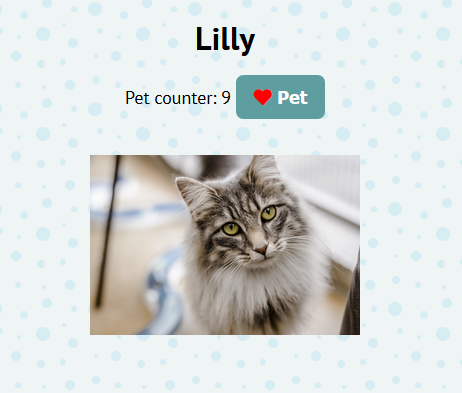
### Edit/Details Pet Screen (5 pts)

Users **should** be able to **edit** their own **pet's description**. Clicking the [**Edit**] link on each pet should open a details page. Inside there should **not be a "Pet" button** and the **description should be displayed in a text box**. There should also be a **"Save"** button to save the changes.

* After a **successful** pet update, a notification message **"Updated successfully!"** should be displayed and the **dashboard** should be shown.



* Users should be able to see **[Details]** button on **other pets** and they should be able to **"Pet"** them, **but not edit** them:



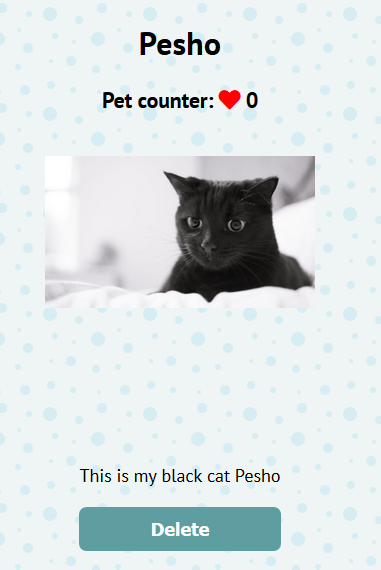
### Pet a pet (5 pts)

Every user should be able to **pet other pets**, but **not his own**. By clicking on the **[Pet]** button, **the counter of each pet increases**.

### Delete Pet (5 pts)

Owners **should** be able to **delete** their **own** pets by clicking the [**Delete**] button.

* When the button is clicked, display the following page:



* When the button **"Delete"** is clicked, the pet should be deleted
* After **successful** pet delete a notification message **"Pet removed successfully!"** should be displayed and the **home catalog** should be **shown**.

## Subtmitting Your Solution

Place in a **ZIP** file your project folder. Exclude the node\_modules folder. Upload the archive to Judge.