# Exercise: Modular Applications

Exercise problem for the [“JavaScript Applications” course @ SoftUni](https://softuni.bg/trainings/3589/js-applications-february-2022).

**Working with Remote Data**

For the solution of some of the following tasks, you will need to use an up-to-date version of the **local REST service**, provided in the lesson’s resources archive. You can [read the documentation here](https://github.com/softuni-practice-server/softuni-practice-server).

## Team Manager

Create a JS application for managing teams.

### Libraries

Use the **page** and **lit-html** libraries for routing and templating. Use npm install to install them, as they are included in **package.json**.

### Data

Use the provided local REST service to store **data** and **users**.

Users are stored similarly to previous tasks in this course, in the /users service. In this app, registration requires an extra field **username** , added along with the usual **email** and **password**.

### HTML & CSS

As usual you are provided with HTML and CSS. You may change the HTML by adding attributes, adding URL's where needed and so on as you see fit.

### Application Requirements

#### Navbar

The application has a navigation bar, with **links** **to other pages**. You need to **implement** the **visibility** of the **buttons** in the header, depending on whether there is a **logged-in user** or a **guest**. This is how the header should look like for a **guest** user:

Screenshot_4.png

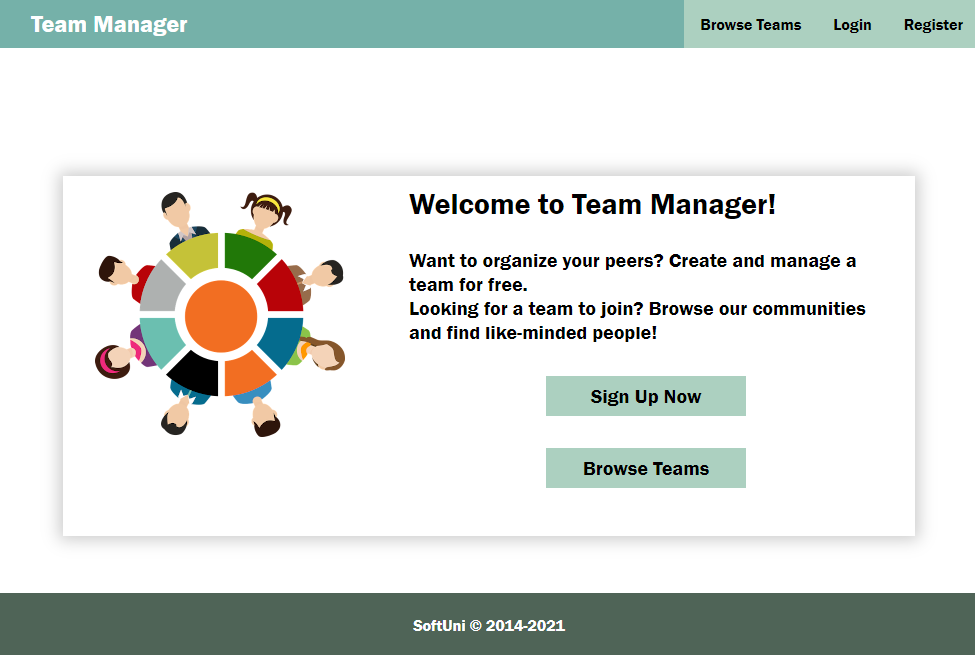
When there is a **logged-in** user, the header should look like this:

Screenshot_3.png

The "**Browse Teams**" button is **always** **visible**. The "**Team Manager**" is link to **home page**.

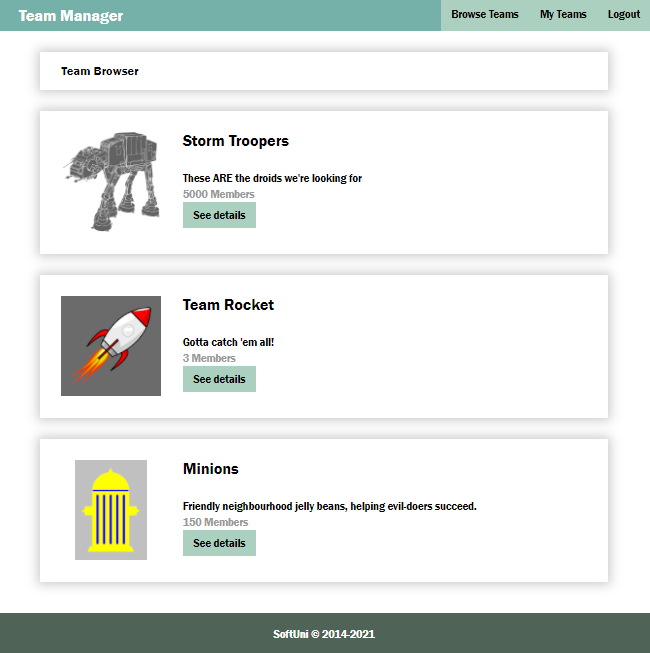
#### Home

**Guest visitors** see the "**Sign Up Now**" button, while logged-in users see the "**Browse Teams**" buttons.

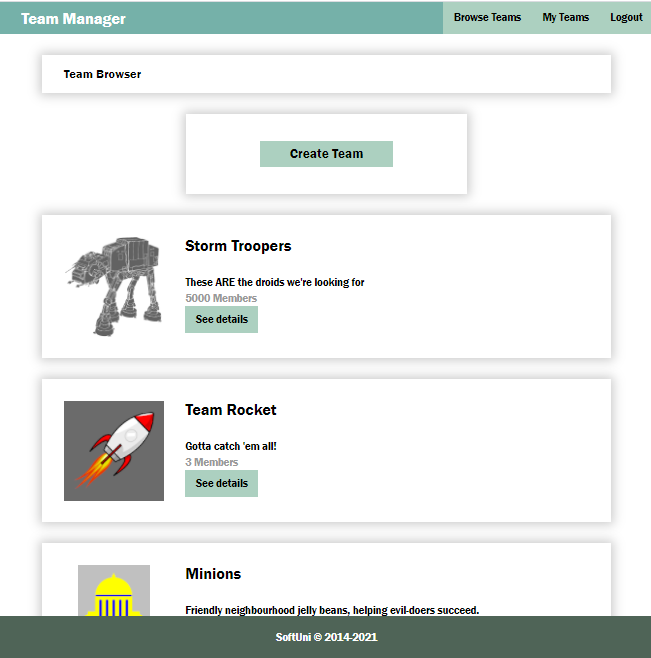


#### Browse Teams

**Guest users** can see a list of all teams. The button "Create Team" is not available for guest users.



**Logged in users** can see the button **"Create Team"**



Use the following endpoints to obtain information from the service:

* List of all teams:

GET /data/teams

* List of all members:

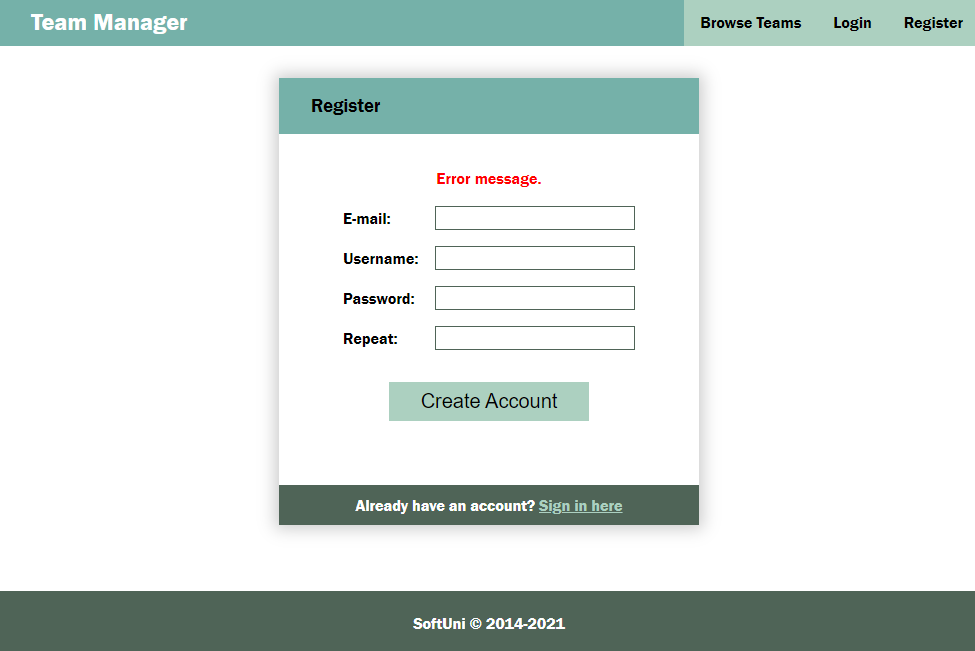
GET /data/members?where=status%3D%22member%22

#### Register

You need to **write** the **functionality** for **registration** of new user. By **clicking** the "**Register**" button you have to load the **registration form**. When the "**Create Account**" button of the **form** is clicked you need to **send a post request** to register the new user. The fields by registration have validations as follows:

* **e-mail**: required, valid e-mail;
* **username**: required, at least 3 characters;
* **password**: required, at least 3 characters/digits.

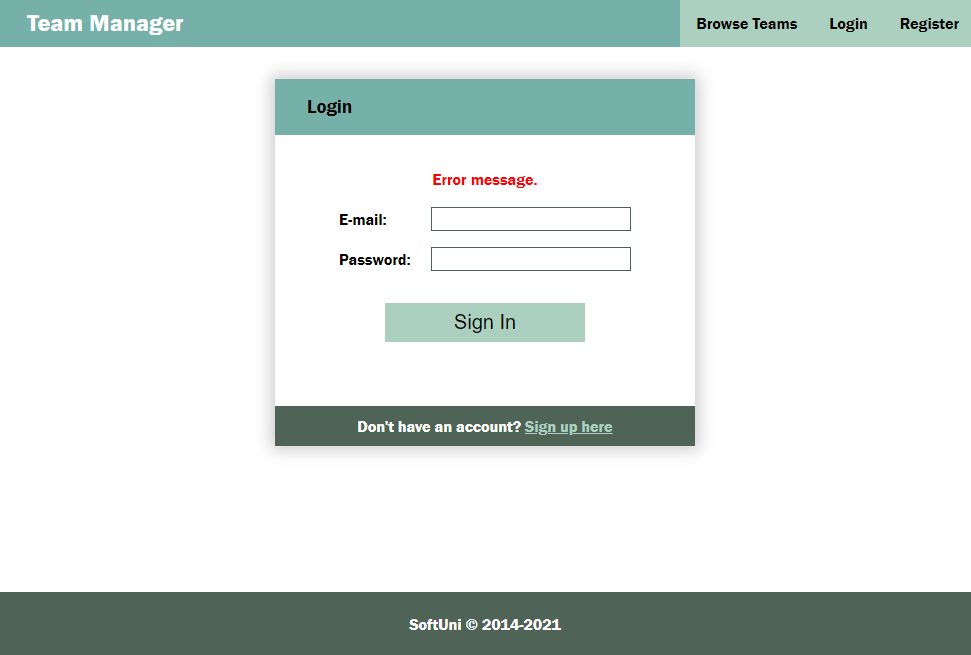
If there is a **validation** **error** you need to show the **div with class "error"** to visualize the **message**, otherwise the **div is not displayed**.



If the registration is **successful** you can redirect to **My Teams page**.

#### Login

If the user **already has registration**, they **may login** by using the **login form**. If there is an error, it should be shown the same way as described for the registration.



After **successful** login the user should be **redirected** to the **My Teams page**.

#### Logout

The **logged in** user **can be logged out** by clicking the **logout button**. Write the functionality for this action. After logout **redirect** to **Home page**.

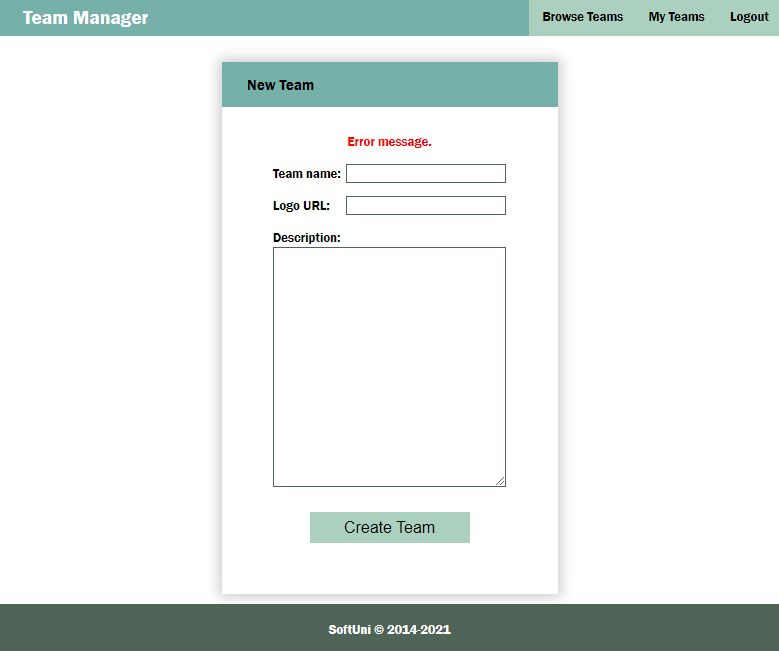
#### New Team

The Create new team page is a form with the following fields and validations:

* **name** (Team name): required, at least 4 characters,
* **logoUrl**: required,
* **description**: required, at least 10 characters

If there is an error by the creation of new team, display the error div with the corresponding message.

After successful creation of team, **redirect** to "**My Teams" page** of the new created team.



Use the following endpoint (body contains name, imageUrl, description):

POST **/data/teams**