# Workshop Part One

# Judge v2

Judge V2 is an application, which organizes the exercises in the Java DB course. In the courses, each homework is required to be in a zip format, which will be evaluated by your peers. The application will have an admin functionality. Only the administrators can add new exercises and give out admin roles to other users. The rest of the users can submit solutions to the specific exercise, as well as evaluate the solutions of your peers.

### Role

Create a Role class, which holds the following properties:

* **id** – UUID **String or Long**
* **name** – a **String**.
  + **USER and ADMIN** (add the two roles manually, when create DB)

### User

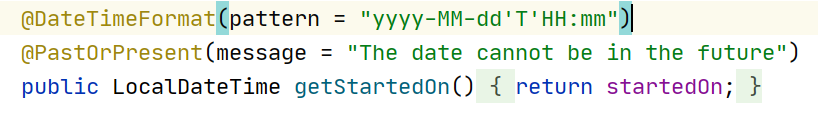
Create a User class, which holds the following properties:

* **id** – UUID **String or Long**
* username – a **String**.
  + username length must be minimum two characters**!**
* **password** – a **String**
  + password length must be minimum two characters**!**
* **email** – a **String**
  + email must contains '@'
* **git** – a **String**
  + git must be a valid github address in pattern: https:/github.com/{username}/SpringTestData/…
* **role** – a **Role**
  + on register every be user.

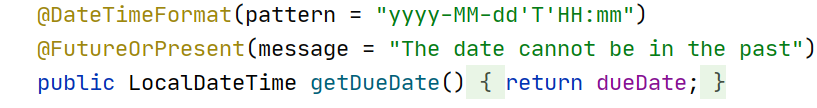
### Exercise

Create an Exercise class, which holds the following properties:

* **id** – UUID **String or Long**
* name – a **String**.
  + name length must be minimum two characters**!**
* **startedOn** – a **Date and Time**
  + startedOn cannot be in the future
    - Little Hint



* **dueDate** – a **Date and Time**
  + dueDate cannot be in the past
    - Little Hint



### Homework

Create a User class, which holds the following properties:

* **id** – UUID **String or Long**
* addedOn – a **Date and Time**.
  + auto set time and date now
* **gitAddress** – a **String**
  + gitAddress must be a valid github address in pattern: https:/github.com/{username}/{homeworkExample}/
* **author** – a **User**
  + logged in user
* **exercise** – an **Exercise**
  + exercise that owns it

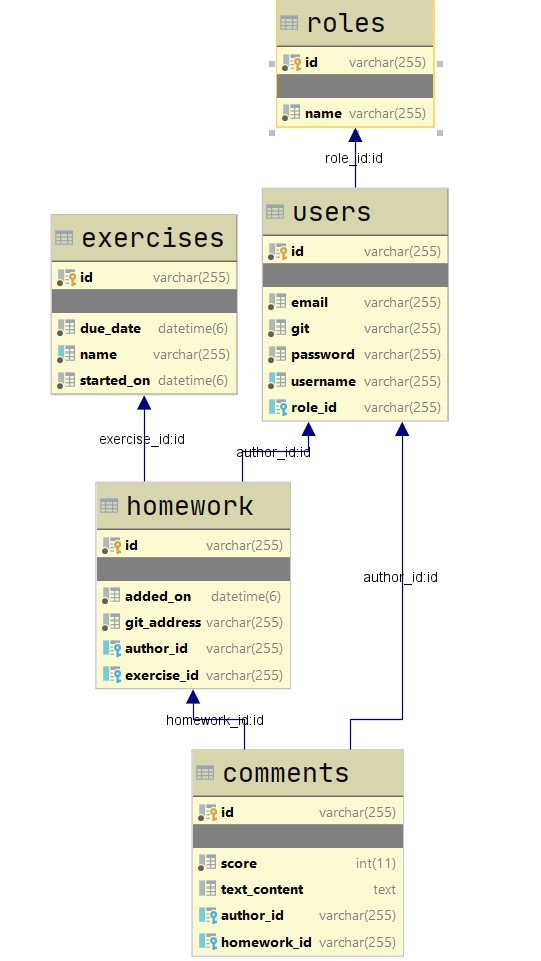
### Comment

Create a User class, which holds the following properties:

* **id** – UUID **String or Long**
* score – an **Integer**.
* **textContent** – a very long **String**
* **author** – a **User**
* **homework** – a **Homework**

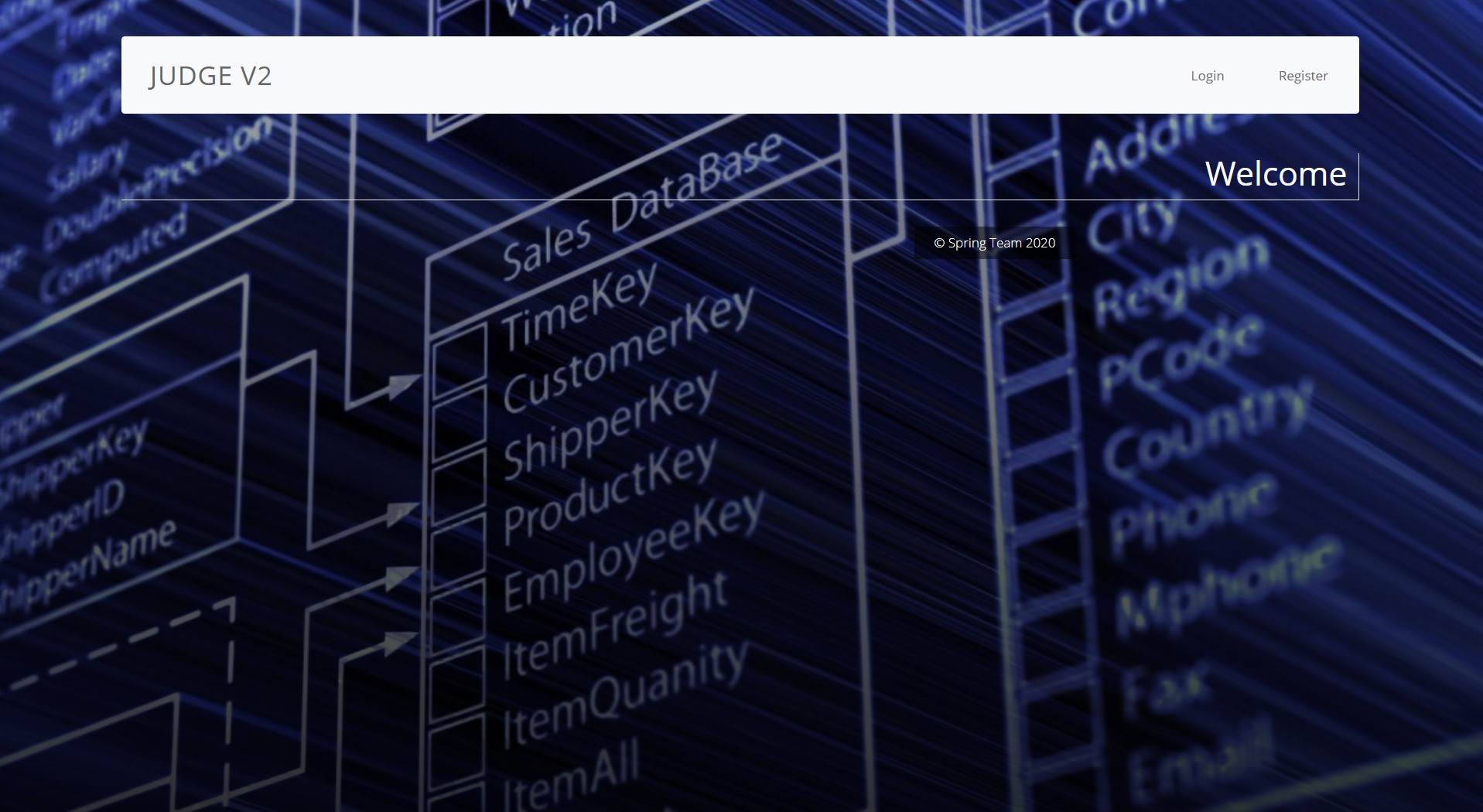
You can change the database and entities so far as the project works according to the task.

**Example for ER Diagram**



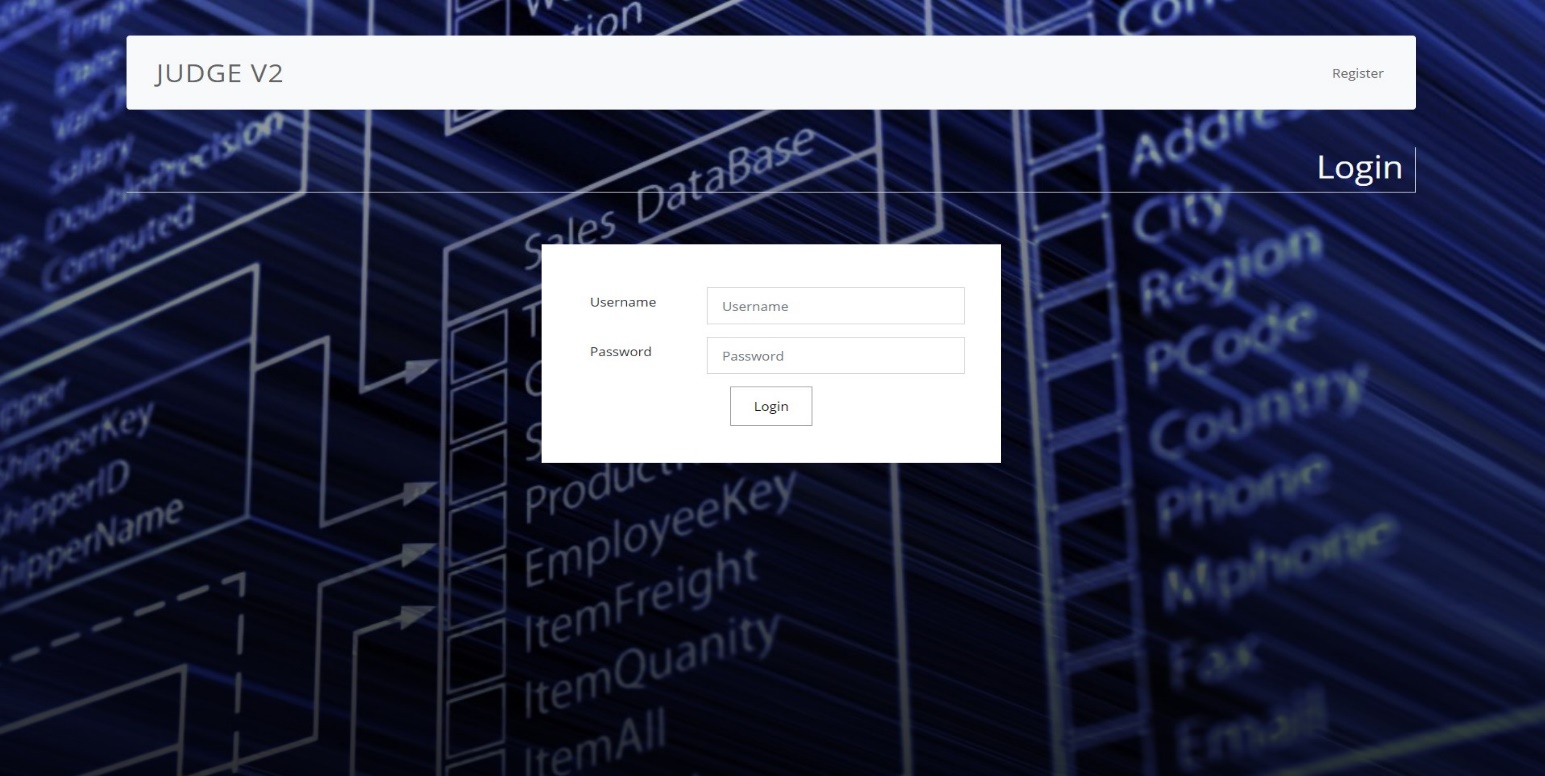
## index - route ("/")

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

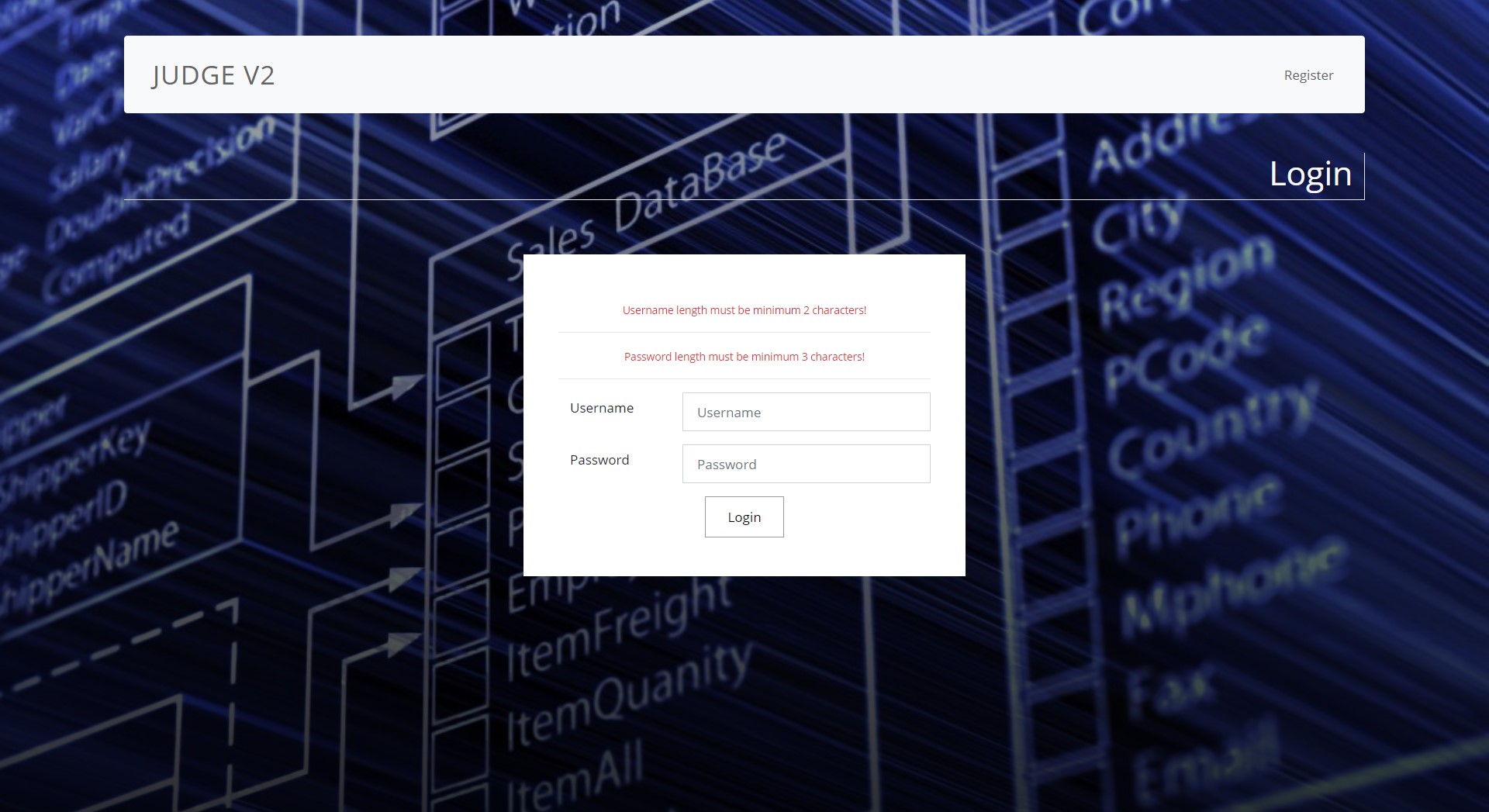


## Login page - route ("/users/login").

* It should support **GET** & **POST** requests.
* It should return the following HTML page, upon a **GET** request.
* When login successfully redirect to "**/home**".

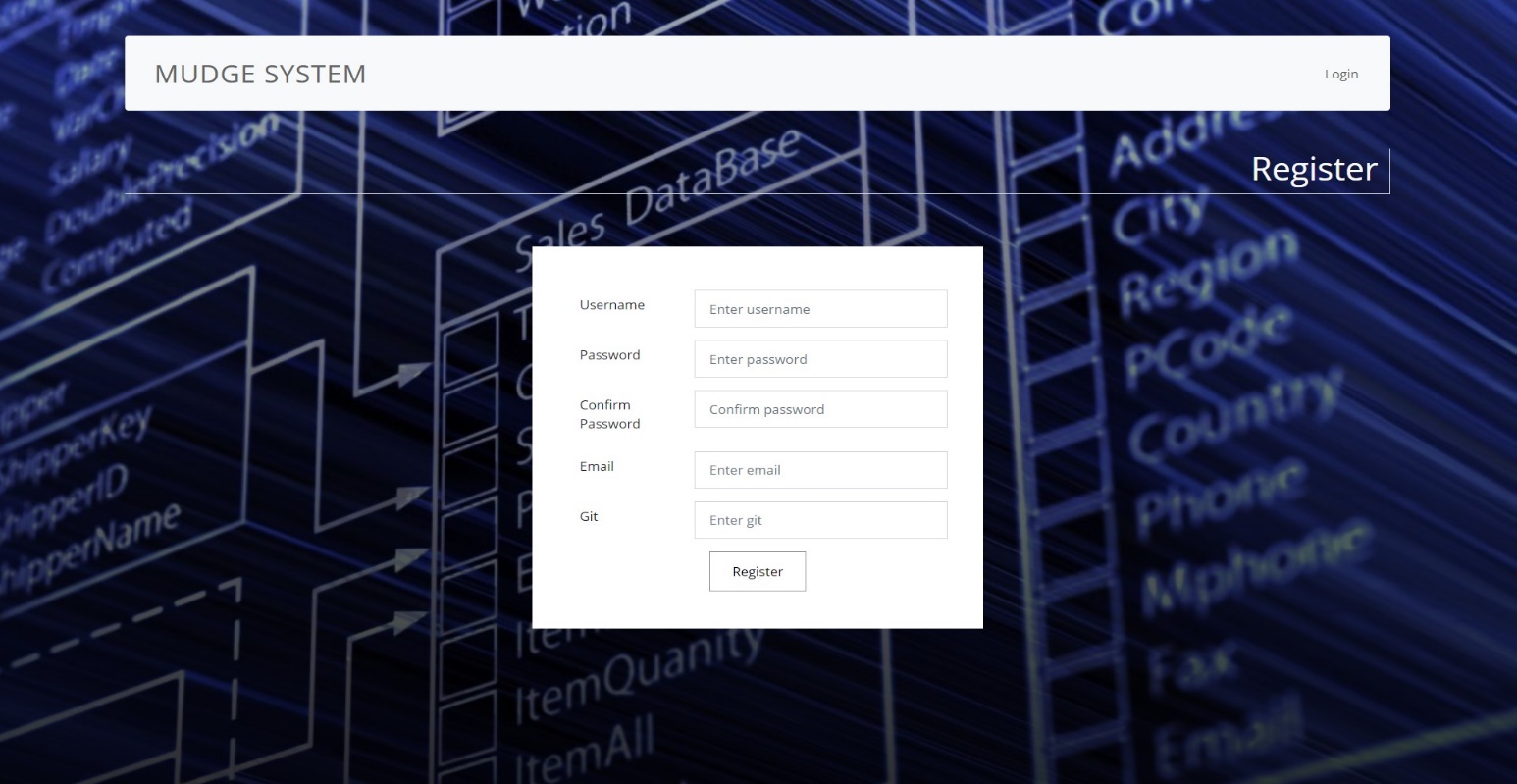


* When the users enter the wrong username or password, you must show the appropriate error message.

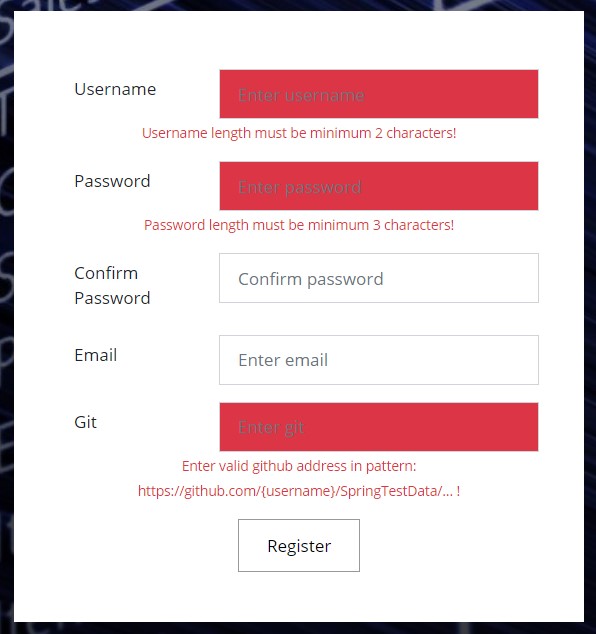


## Register - route ("/users/register").

* It should support **GET** & **POST** requests.
* It should return the following HTML page, upon a **GET** request.
* When register successfully redirect to "**/users/login**".



* When the users enter the wrong username or password, you must show the appropriate error message.



## Home page - route ("/home").

* It should support a **GET** request.
* It should return the following HTML page, upon a GET request.
* For now, just create home page for the logged in user, who are not admins.
* Later we will explain all for this page in details, for now just show it to the user on this route.
* Guest users cannot access it. They will redirect to login page.

