# Express.js Exam – Project Manager

## Exam rules:

* You have 6 hours – **from 09:00 to 15:00**
* When you are ready, delete the ***node\_modules***folder, make sure all dependencies are listed in the ***package.json*** file and submit your archived project at: [SoftUni Judge](https://judge.softuni.bg/Contests/1524/ExpressJS-10-Feb-2019)
* You are required to **edit** the **provided HTML** to make it functional and **may extend** it as needed.
* Use **Express.js** as a back-end framework
* Use **MongoDB** as a database with **mongoose**
* You can use whatever **view engine** you like (Handlebars, EJS, Pug etc.…)

## Application Overview

Get familiar with the provided **HTML & CSS** and create an application for **managing projects** and the **teams** in them. The **user** should be able to **view projects and teams**, and to **leave a team**. The **admin** should be able to **distribute users into teams** and **teams into projects**. He should also be able to **create teams and projects**.

**Note: The points described in each of the sections include all the functionalities that the given task should cover!**

## Functional Requirements

The Functionality Requirements describe the functionality that the **Application** must support.

The **application** should provide Guest (not logged in) users with the functionality to login, register and **view** the Home page.

The **application** should provide Users (logged in) with the functionality to logout, and view all teams**, view all projects, view their profile, leave a team**

The **application** should provide Admins (logged in) with the functionality to logout, and distribute teams, **distribute projects, create teams, create projects, view their profile**.

The admin should be **seeded with the starting of the application**. A **registration of a new admin** user should **not be possible**!

### Users

The user should be able to **view all the teams**, **all the projects** and **his own profile**. In his profile page if he does not have teams or projects or both, you should display the following:



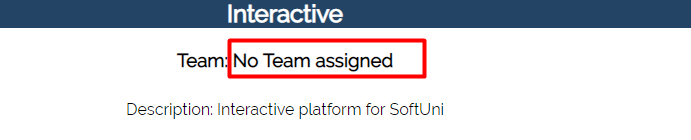
The user should **not be able** to **edit** or **delete** **teams** **and projects**

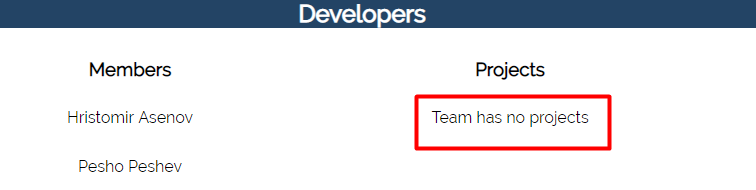
When the user **leaves a team**, the **team should be removed** from **his teams list**, and the **user should be removed** from the **members list of the team.**

### Teams and Projects

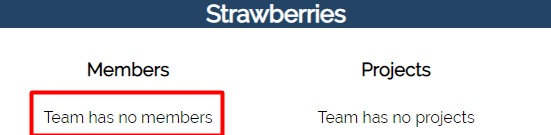
In the sections Projects and Teams, the projects and the teams should be listed as shown on the pictures below.

If there is a project without a team display the following:



If a team does not have a project, display the following:

If a team has no members, display the following:



### Admin

The admin **should not be** able to view the teams and projects like a user does. He has to be able to **distribute** users into teams and teams into projects.

## Database Models (15 pts)

The **Database** of the Project Manager application needs to support **3 entities**:

### User (5 pts)

* Username – string (required), unique
* Password – string (required)
* First Name – string (required)
* **Last Name – string (required)**
* **Teams – a collection of** Teams
* **Profile Picture – imageUrl string and choose a default**
* **Roles – array with roles ("User", "Admin")**

### Team (5 pts)

* **Name – string (required), unique**
* **Projects – a collection of Projects**
* **Members – a collection of Users**

### Project (5 pts)

* **Name – string (required), unique**
* **Description – string (required), max length of 50 symbols**
* **Team – a single Team**

Implement the entities with the **correct datatypes**.

## Application Pages (75 pts)

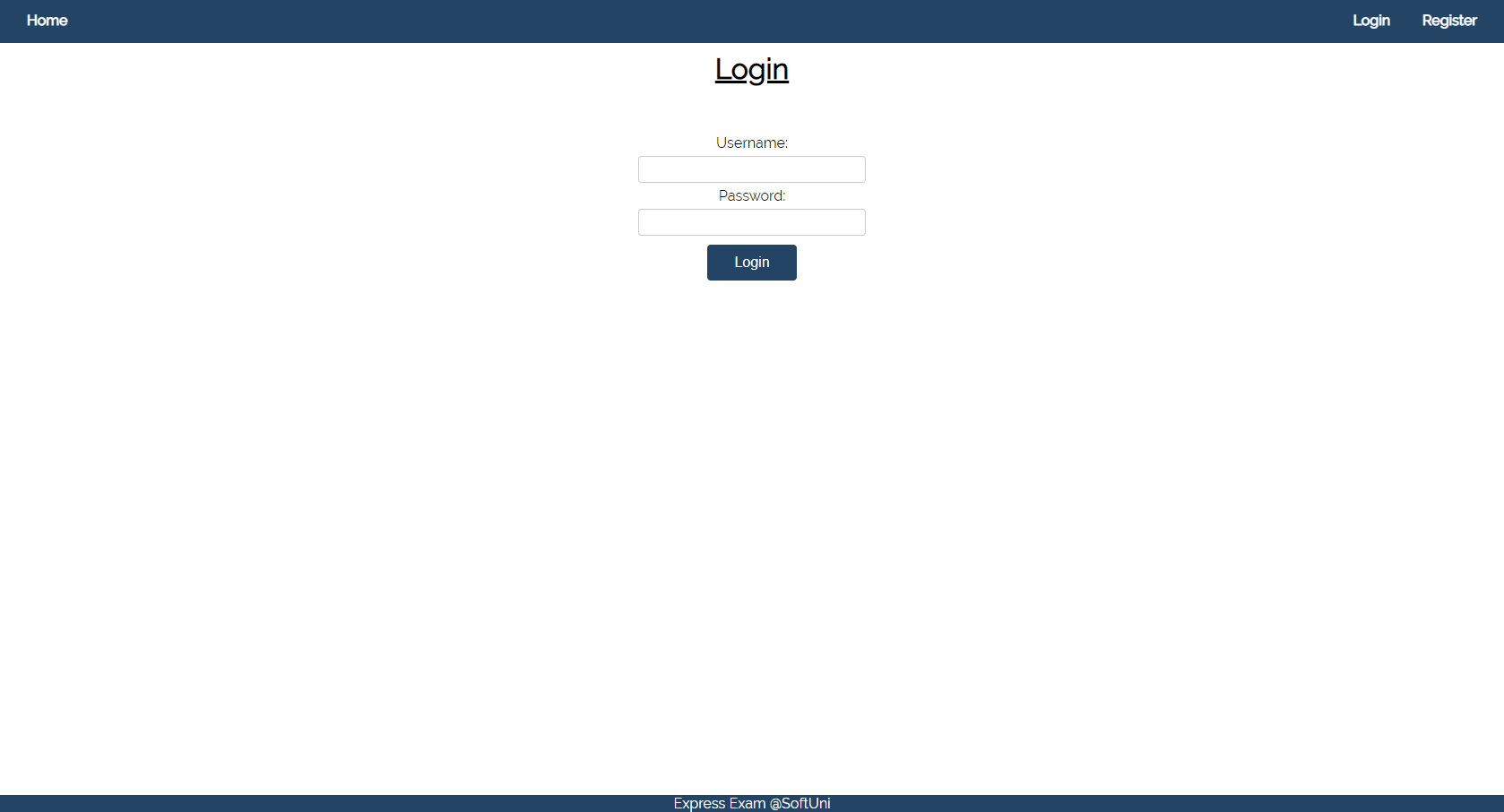
### Guest Pages (10 pts)

These are the **pages** and **functionalities**, accessible by Guests (**logged out** users).

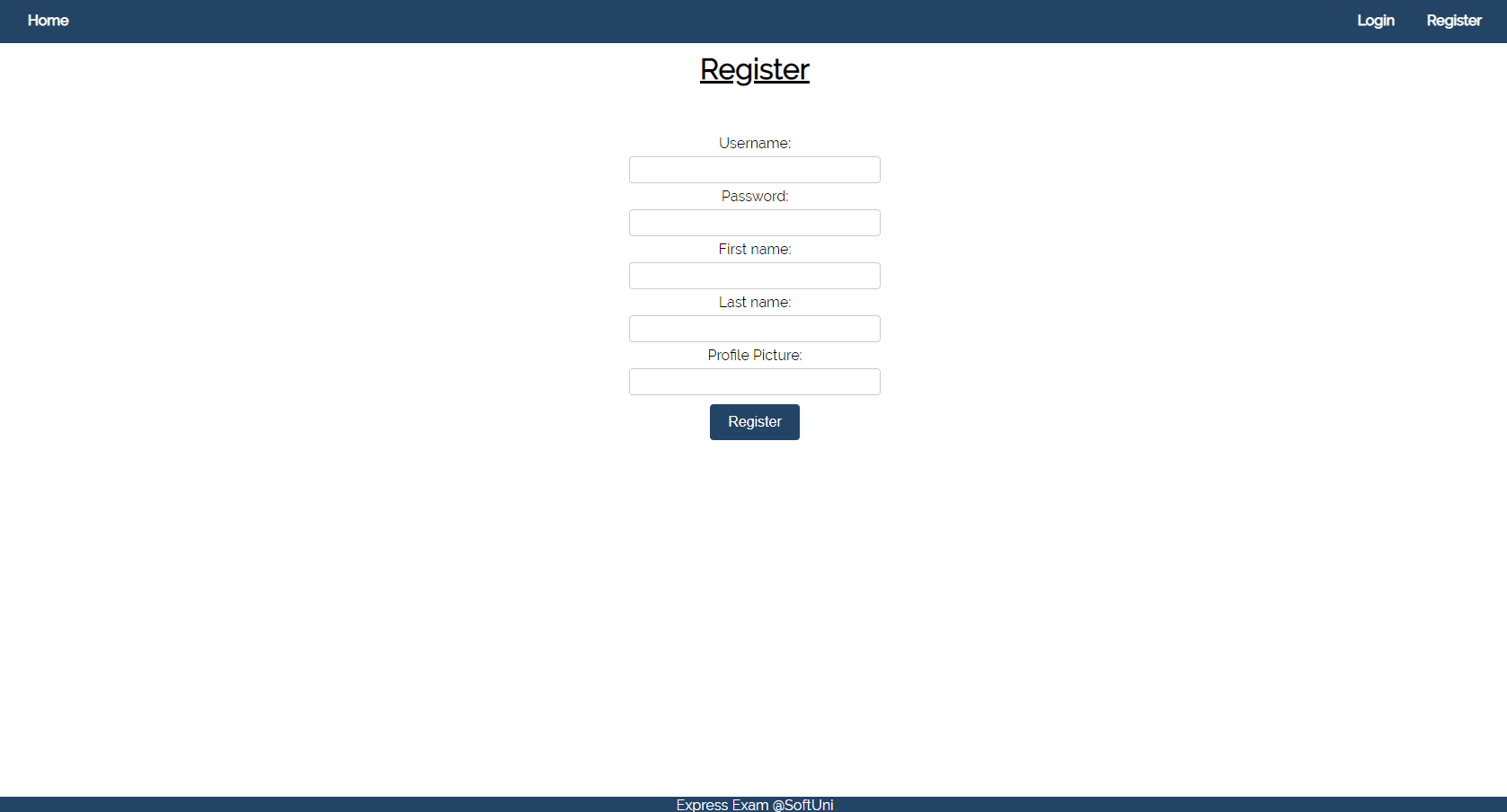
#### Index Page (logged out user)



#### Login Page (logged out user) (5 pts)



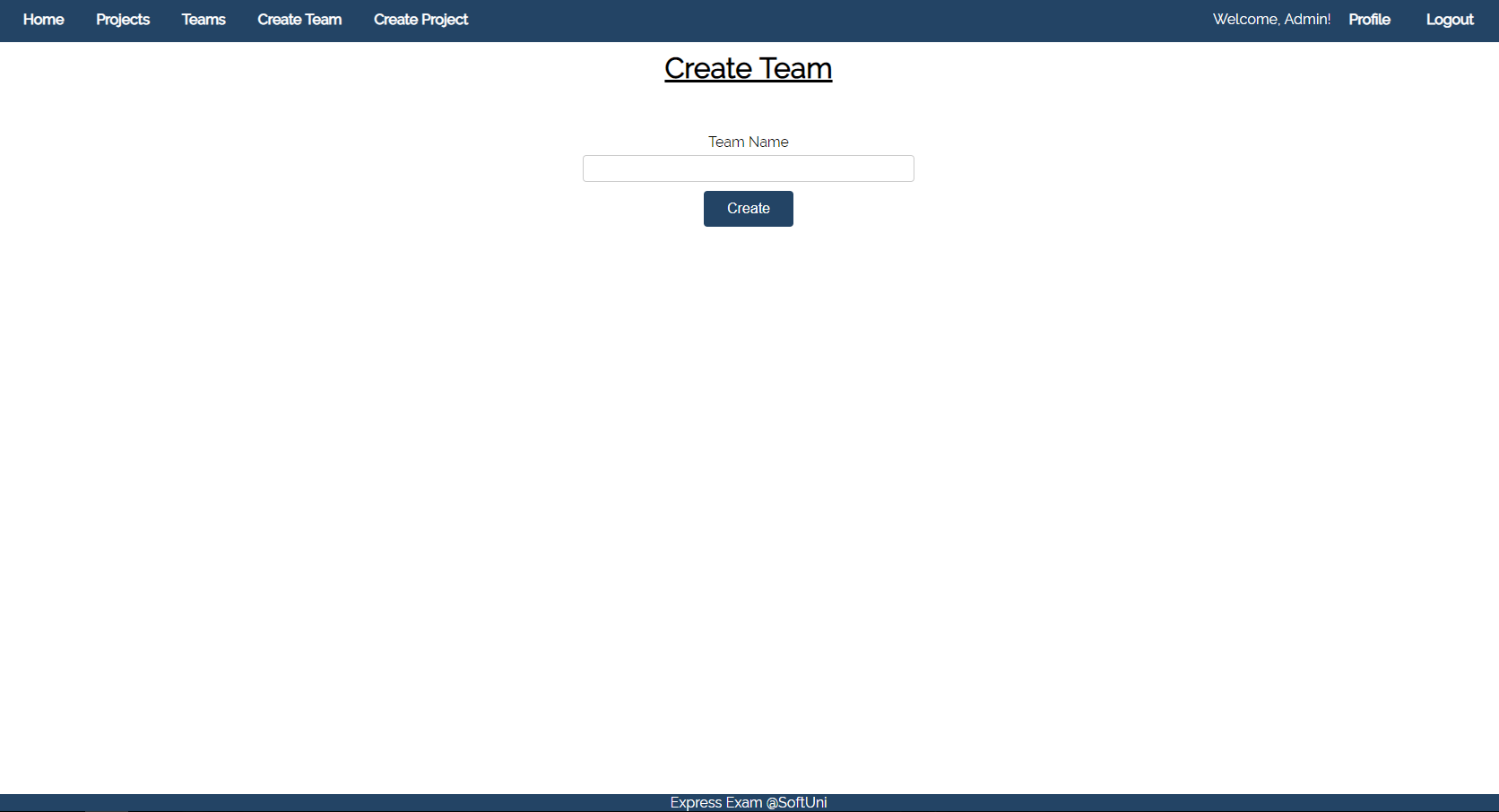
#### Register Page (logged out user) (5 pts)



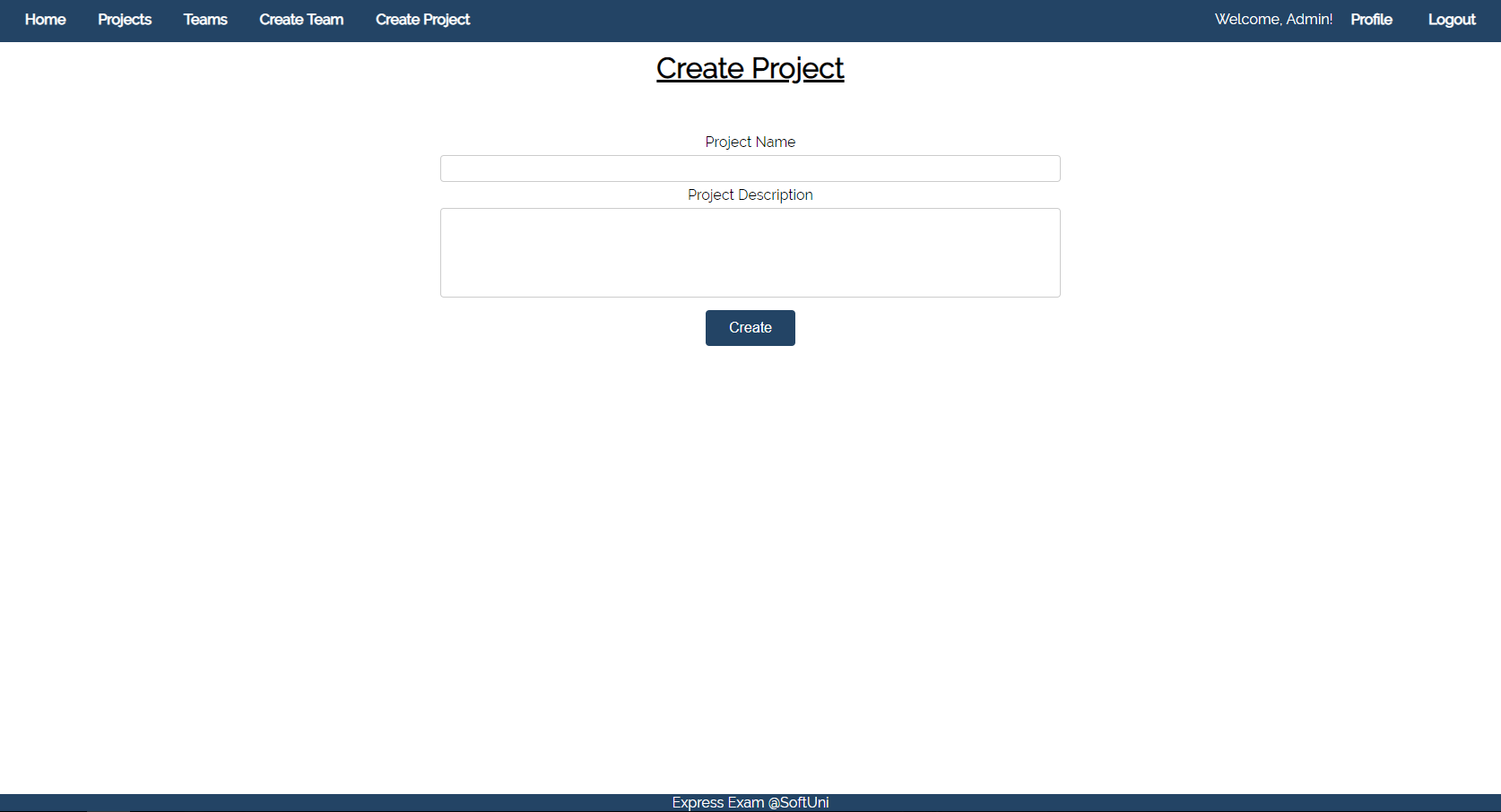
### Admin Pages (30 pts)

These are the **templates** and **functionalities**, accessible by Admins

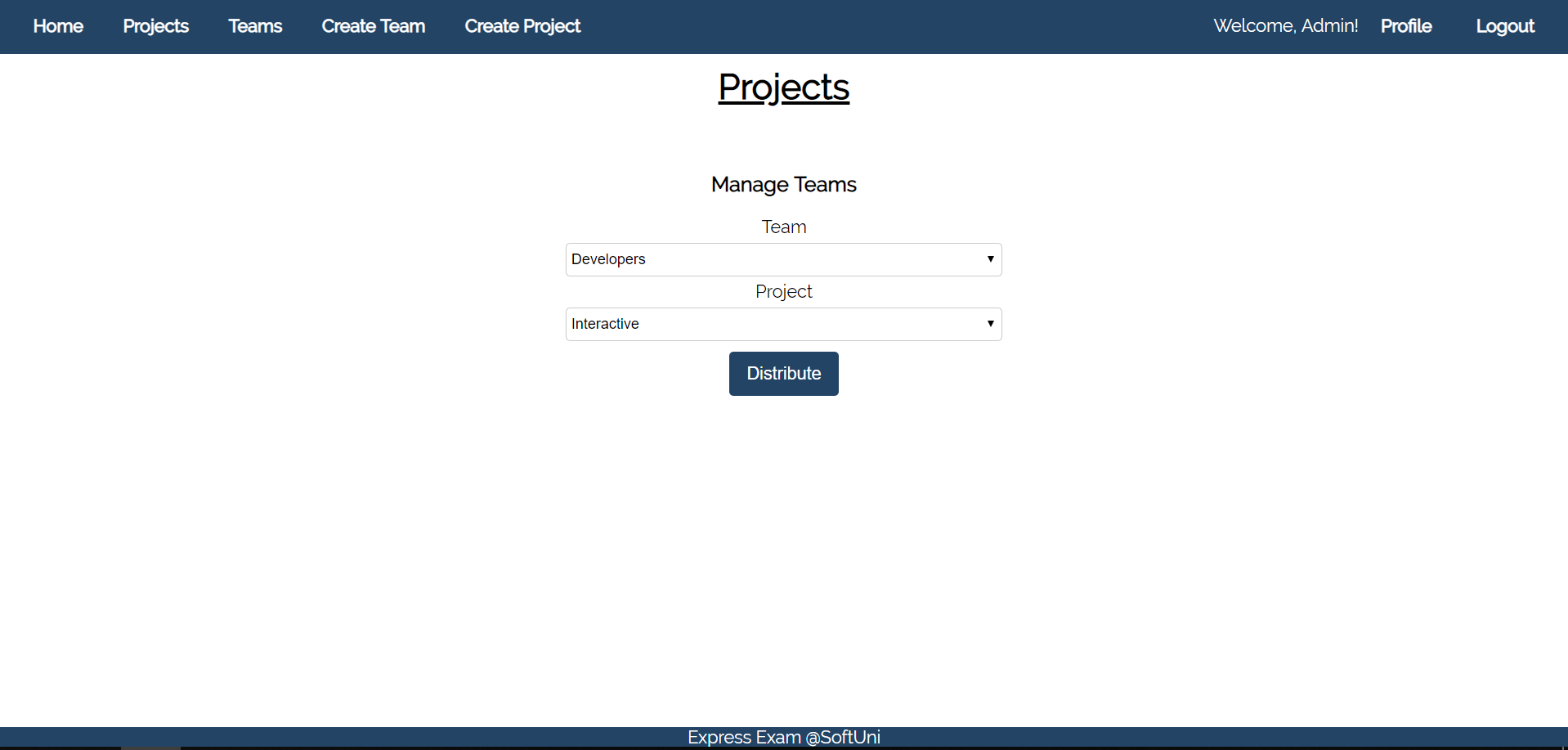
#### Admin Create Team (logged in admin) (5 pts)



#### Admin Create Project (logged in admin) (5 pts)



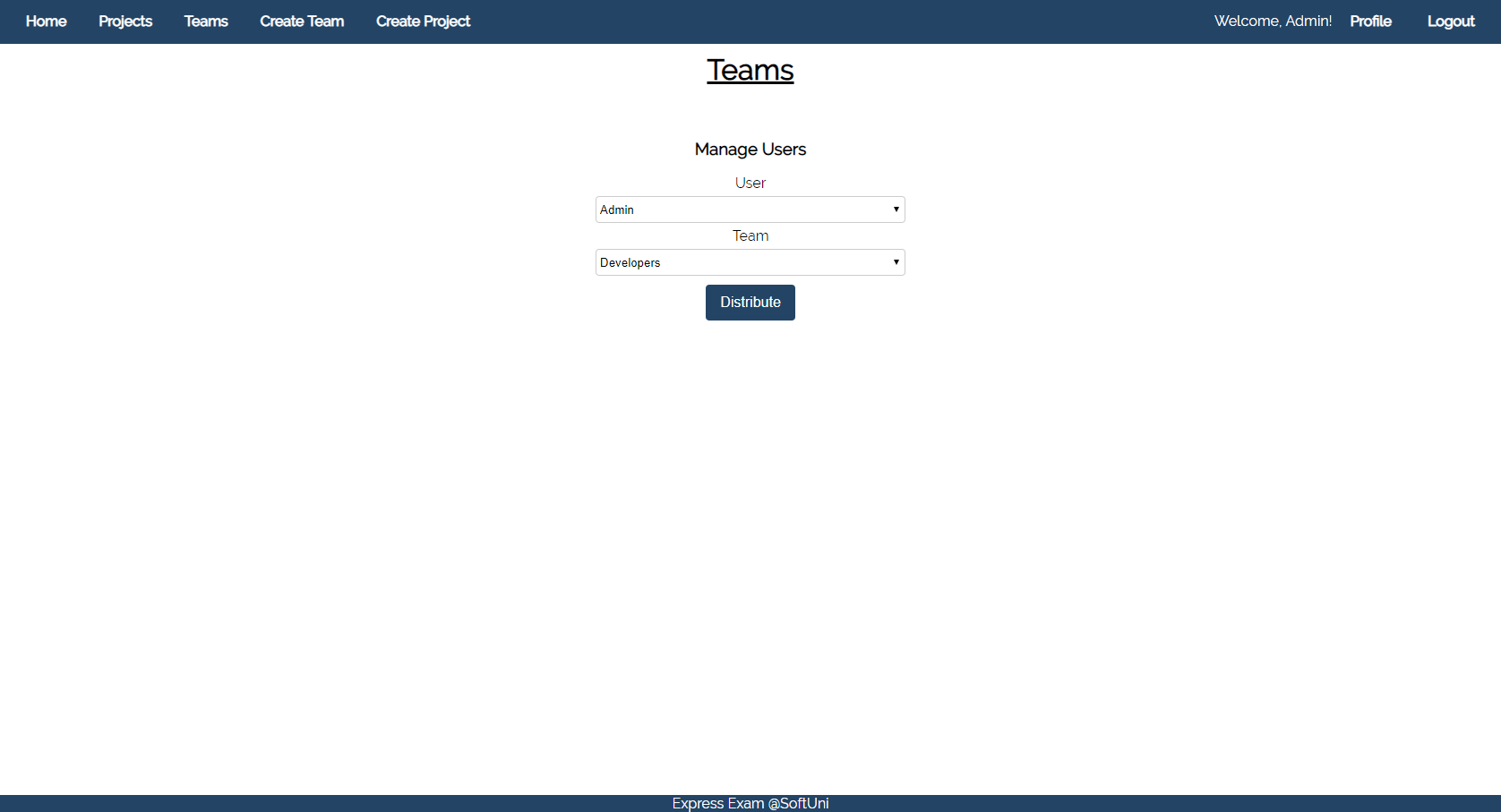
#### Admin Projects (logged in Admin) (10 pts)



The dropdown for the **teams** should include **all** the teams.

The dropdown for the **projects** should only contain projects that **do not yet** have a team assigned to them.

#### Admin Teams (logged in Admin) (10 pts)



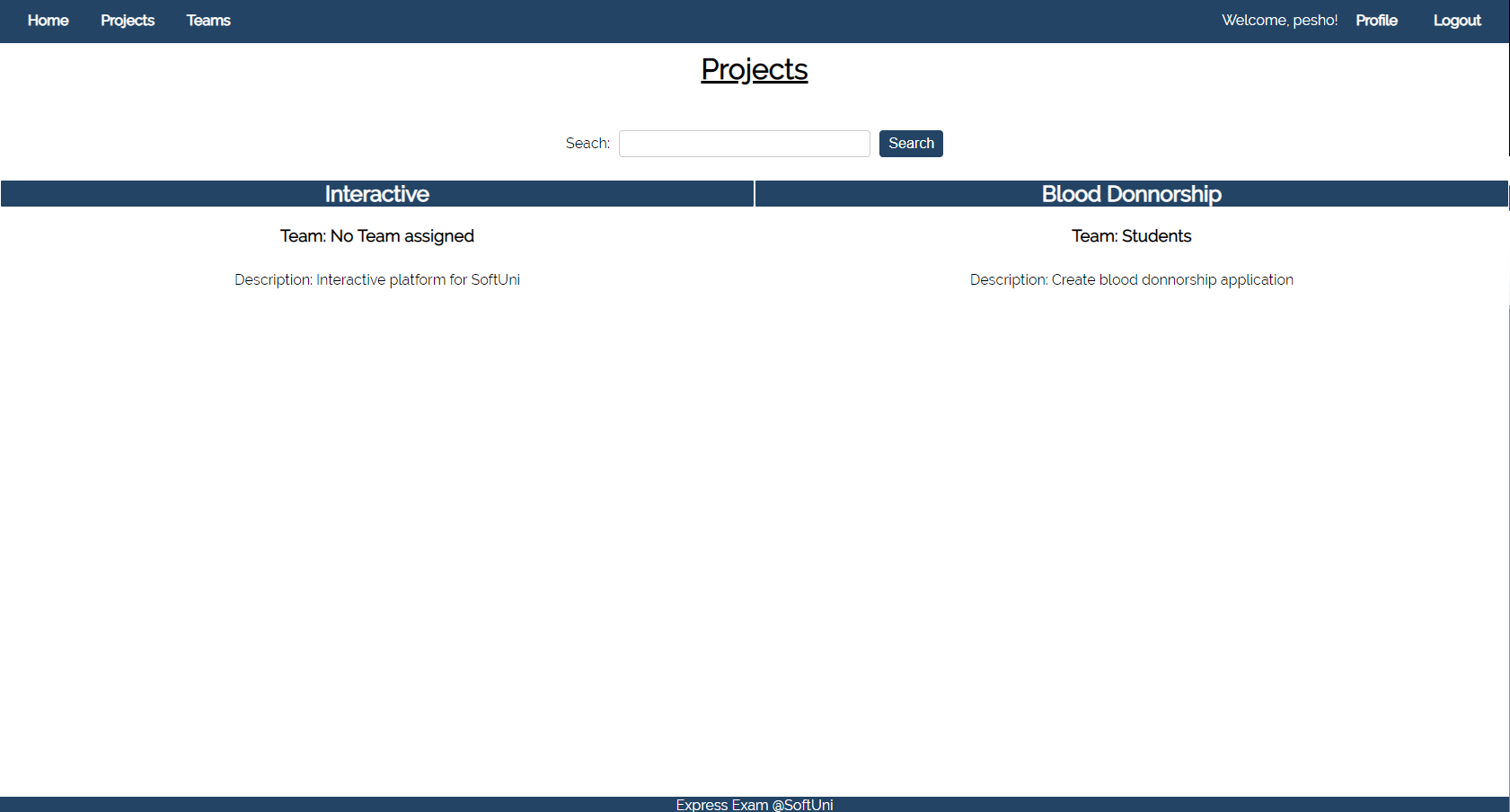
In the **users list** include **all users** in the database. Each **user** can be in **many teams**. He cannot however be added in the **same team** twice.

In the **teams list** include **all teams** in the database.

### User Pages (35 pts)

These are the **templates** and **functionalities**, accessible by Users (**logged in** users).

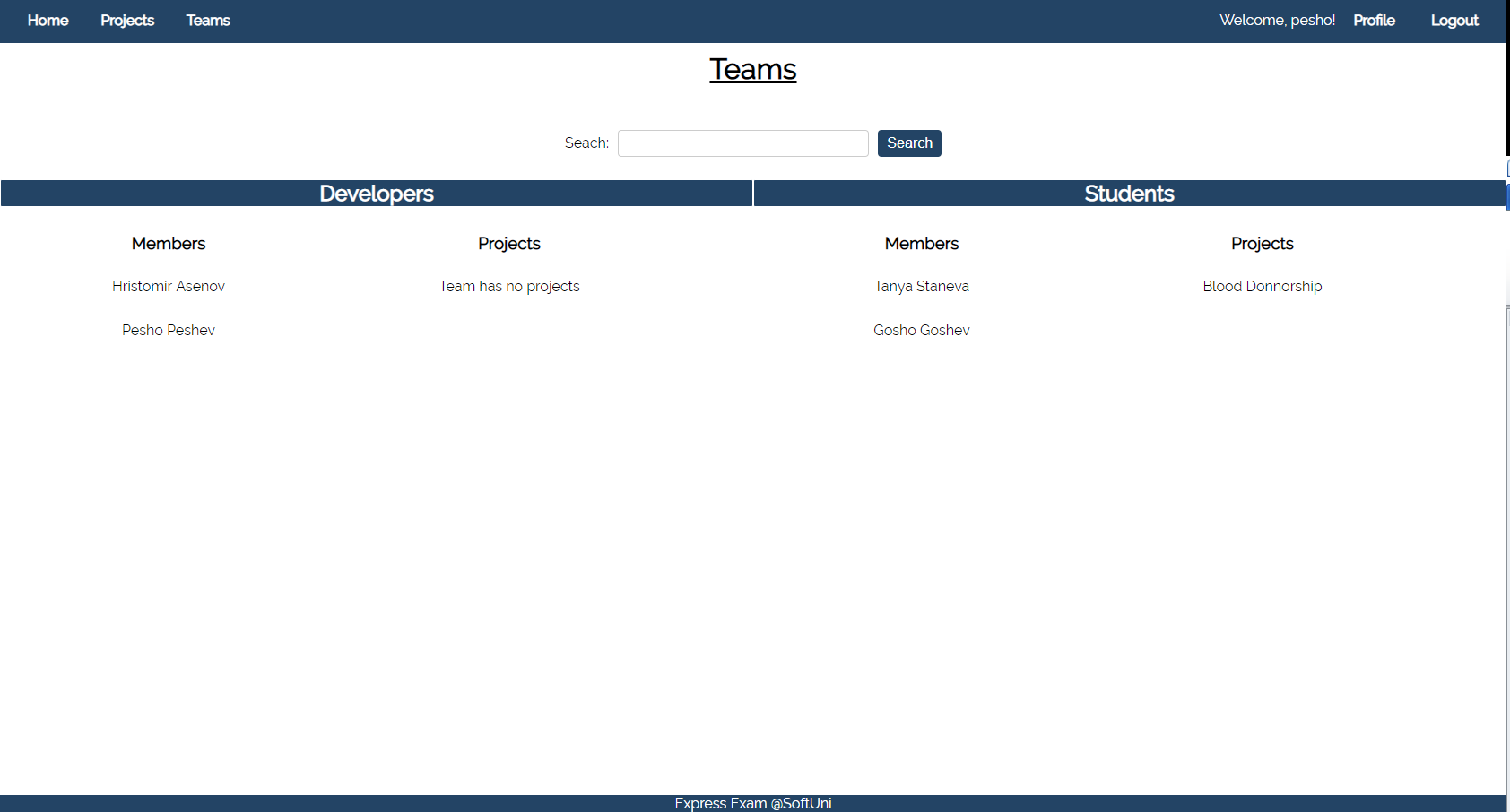
#### Logged In Projects (logged in user) (10 pts)



In the Projects section you should load **all project names** and for each project load the **team name** and **description**

If there is no team assigned yet display **"No Team assigned".**

#### Logged in Teams (logged in user) (10 pts)



In the Team section you should load all team names and for each team load all projects and members

In the **members section** display the **first and last name of each member**.

In the **projects section** load only the project **name**.

If there are **no members** assigned to a team display **"Team has no members"** instead.

If there are **no projects** assigned to a team display "**Team has no projects**" instead.

#### Logged in Profile (logged in user) (15 pts)

In **My Teams** section display the **name of each team that the user is part of**. In case of **no teams** display **"You have no teams".**

In **My Projects** display the **names of all projects that the users team/teams is/are assigned to.**

The User is able to **leave** a team if he wants to do so. If a User leaves a team it is **removed** from his **teams list** and the User is **removed** from the **members** list of a team.

## Security Requirements (10 pts)

The Security Requirements are mainly **access** requirements. Configurations about which users can access specific functionalities and pages.

* Guest (not logged in) users can access Home page and functionality.
* Guest (not logged in) users can access Login page and functionality.
* Guest (not logged in) users can access Register page and functionality.
* Users (logged in) can access Projects page and functionality.
* Users (logged in) can access Teams page and functionality.
* Users (logged in) can access the Profile functionality.
* Users (logged in) can access Logout functionality.
* Admins (logged in) can access **Projects** (but with a different view).
* Admins (logged in) can access **Teams** (but with a different view).
* Admins (logged in) can access Create Team page and functionality.
* Admins (logged in) can access the Create Project page and functionality.
* Admins (logged in) can access the Profile page and functionality.
* Admins (logged in) can access the Logout page and functionality.

## Bonus (10 pts)

When a **user searches in teams**, filter the **team names** that **include the searched string** (**case-insensitive**).

When a **user searches in projects**, filter the **project names** that **include the search string** (**case-insensitive**)