All the requirements listed in the task are fulfilled.

Although you mentioned that the front-end should have been simple, I decided to use Bootstrap and write some css code, so the software UI looks more compact and modern.

Besides, you mentioned that the registration/login might be done using VS auto generated authentication model. That model was working with Identity(Entity Framework) object and had bunch of unnecessary tables and columns in those tables. I decided to throw it away and create a simpler DB object - corresponding to the application needs.

The description of the use cases follows:

1. **Register use case**

User fills in his/her personal information and creates an account in the system.
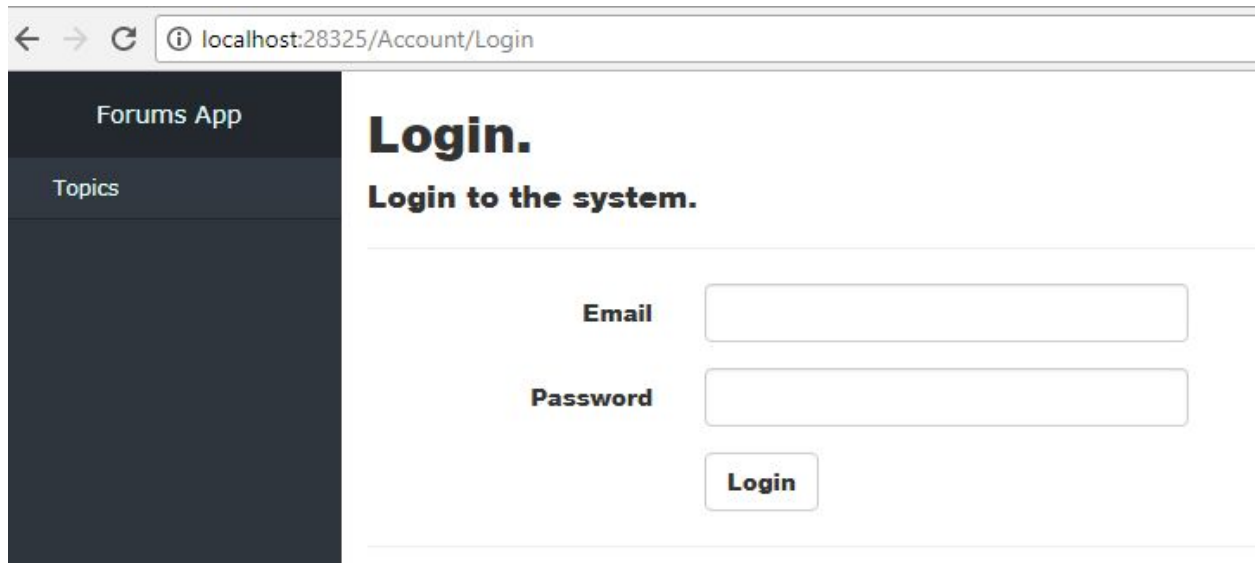
By pressing 'register' button he/she automatically logs-in to the system. New session is opened for that user.

## 2. Login/logout use cases
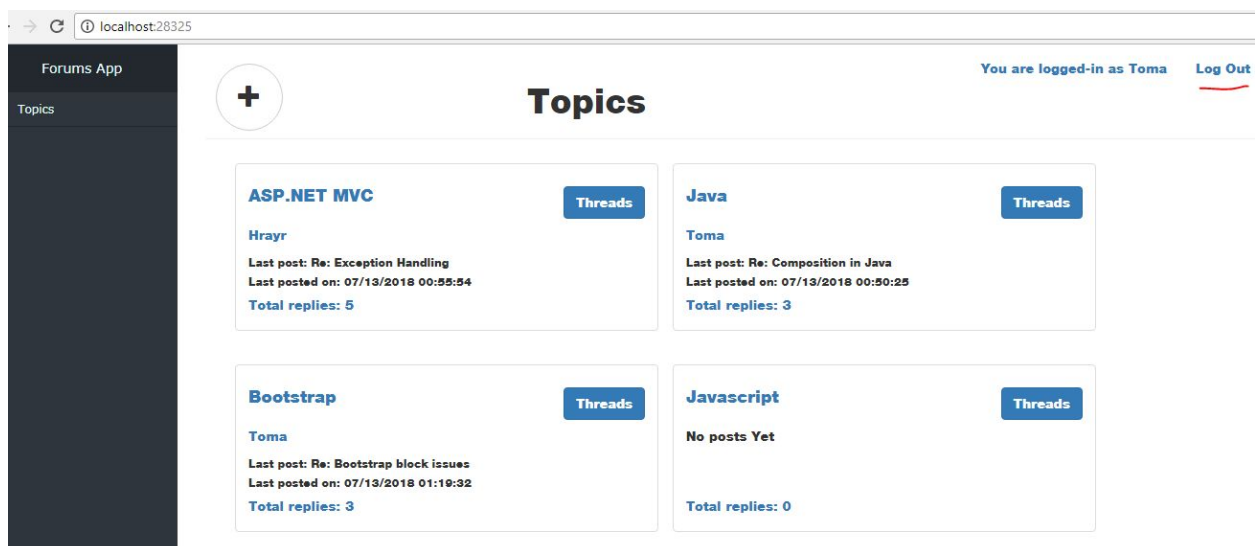
User fills in his email and password and logs into the system.



On the top right of the screen his/her Nickname appears. Near it there's a logout button which clears the current user from the session and allows only read access to topics/threads/posts.
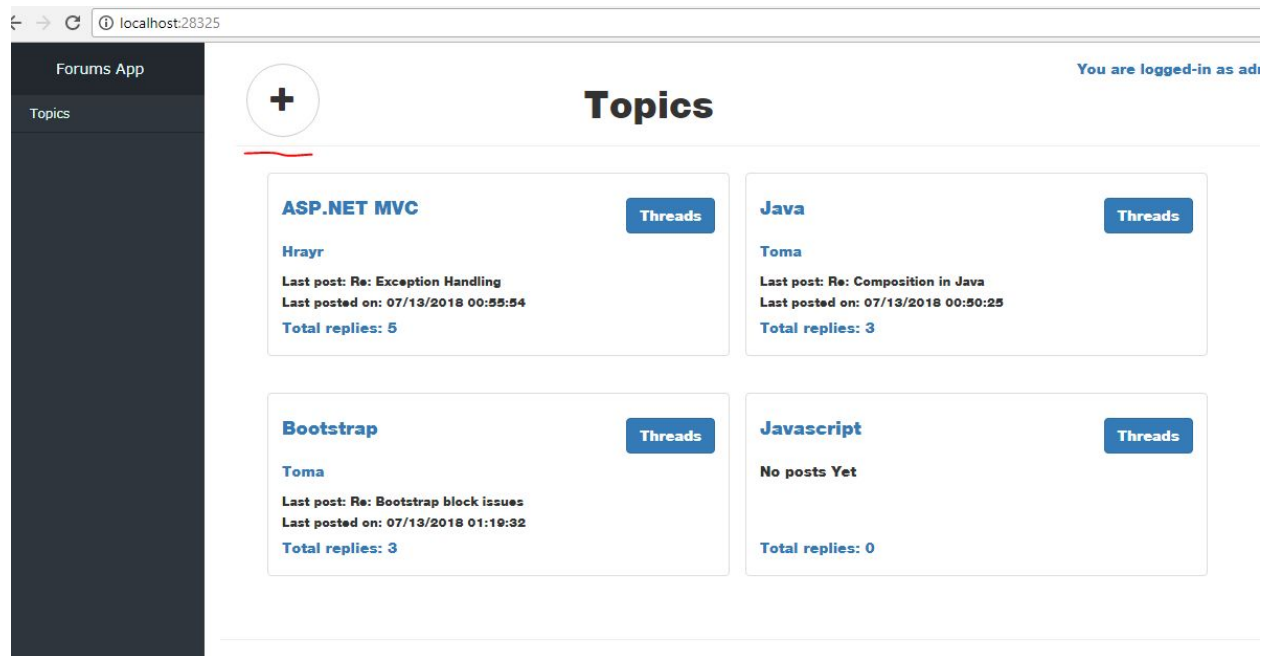
## 3. Add topic use case

Add Topic button is always there, BUT only users with the role 'Admin' may add topics.



Otherwise, if the user is not registered or not an Admin, system shows error message.
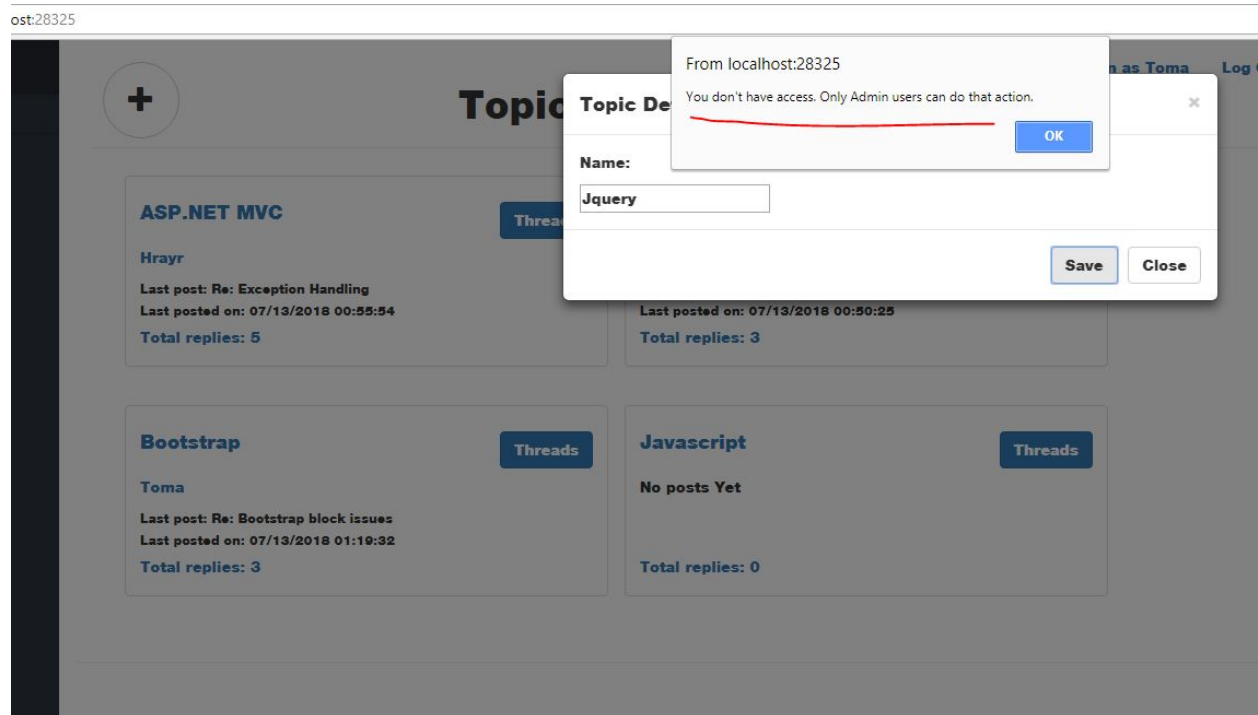
## 4. Add/stop/start thread use case

Threads can be added by any registered user (there's no user restriction for adding threads in the task). Only admin users may stop/archive or start the thread. The stop/start buttons appear only for admin users. Also there's a server side authentication attribute allowing only admins to stop/start threads.



Each thread has information about the total replies (posts) of it, name of the user who last posted in the thread and the date of posting.

## 5. Add/delete/edit post use case

After navigating to the thread details, all the posts of that thread appear on the screen in creation date descending order.

Navigation to thread link appears on the left menu. Thread name and description appear in the title.

There's also a button which sorts the posts by ascending/descending order.

In each post the name of the thread appears with 'Re:', the actual post text, its creation date and information about the user who created it.



User can edit/delete only his/her <u>own posts</u>.

# of Exception Handling thread

### System throws exception. I can't understand the problem.

**Sort**

**Re: Exception Handling**

It depends on your project. If you want to do some custom action, than its better to inherit from Authorization attribute and override its methods

| | |
|---|---|
| Poted By: | Hrayr |
| Total posts: | 5 |
| Registered: | 07/13/2018, 00:43 |
| Country: | Armenia |
| City: | Yerevan |

Posted on: 07/13/2018, 00:55

**Re: Exception Handling**

Should I write custom attribute or use the built it attribute of the framework?

| | |
|---|---|
| Poted By: | Toma |
| Total posts: | 6 |
| Registered: | 07/07/2018, 13:32 |
| Country: | Armenia |
| City: | Yerevan |

Posted on: 07/13/2018, 00:54

**Re: Exception Handling**

Yes. That's what I've exactly done, Thank you

| | |
|---|---|
| Poted By: | Hrayr |
| Total posts: | 5 |
| Registered: | 07/13/2018, 00:43 |
| Country: | Armenia |
| City: | Yerevan |

Posted on: 07/13/2018, 00:52

**Re: Exception Handling**

You should use try catch block. In catch block you can see the inner exception where the details are described.

| | |
|---|---|
| Poted By: | Toma |
| Total posts: | 6 |
| Registered: | 07/07/2018, 13:32 |
| Country: | Armenia |
| City: | Yerevan |

Posted on: 07/13/2018, 00:35

I'd like to mention that I haven't encrypted/decrypted the password as there's no module for password retrieval. It'll be easier to take a look into the database in case if you forget your password.