Problems Encountered: SimpleBlog

Adding Role check boxes to edit page:

In userController.Edit() on the return view, I had to create a new UsersEdit object which contained the name, email and checkboxRoles which the user I was editing had on it. This was so that I could display that name, email and all the roles(with check boxes beside them) of the user.

Inside of my new UsersEdit object which I was returning in the view I had to fill the CheckBoxRoles with a list of RoleCheckBox’s (these objects have a name, id and IsChecked field). This is where I ran into a problem:

return View(new UsersEdit

{

Email = user.Email,

Username = user.Username,

CheckBoxRoles = db.Roles.Select(role => new RoleCheckBox

{

Id = role.RoleID,

IsChecked = user.Roles.Contains(role),

Name = role.RoleName

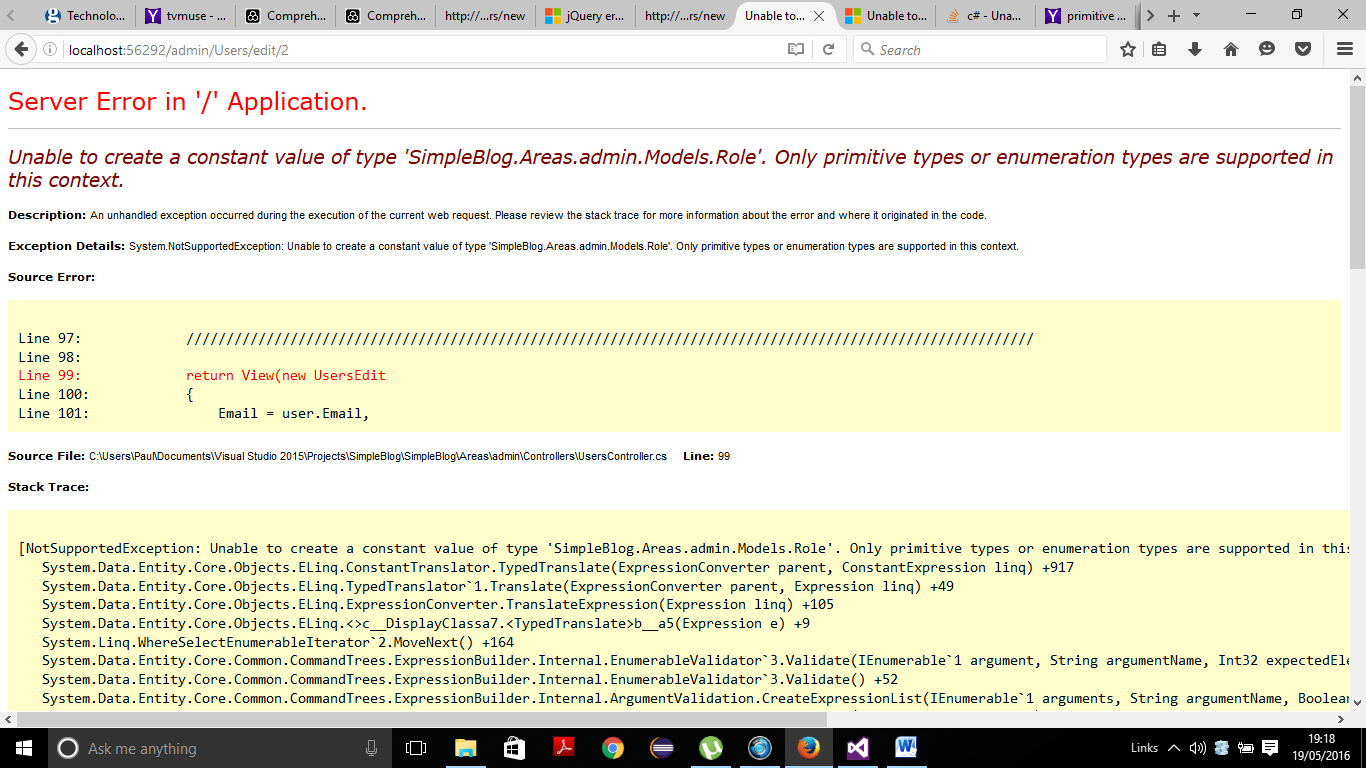
}).ToList()

});

}

The problem occurred on line starting IsChecked.

The error I got was the one shown below:



Essentially, only primitive types are allowed in the lambda query I wrote. I don’t fully understand this. I think it had something to do with: user.Roles.Contains(role)

I think that maybe I couldn’t do this because whilst querying a list of objects might be allowed outwith a lambda, it maybe isn’t allowed inside one.

**SOLUTION:**

My solution does not make much sense to me at the moment. I replaced line:

IsChecked = user.Roles.Contains(role),

With

IsChecked = variableName.Contains(role.RoleName)

Where the variableName was a variable I declared just above the return(View). This variable was the query:

IEnumerable<string> variableName = user.Roles.Select(r => r.RoleName);

This query takes in a list of roles (which the user has) and sends in r (role object) and converts the role to a list of role names. So this returns a list of strings essentially. And then in the lambda above I look to see if role.RoleName is contained within this list of strings. And if the role.RoleName matches the roleName of the user roles, then the query returns a bool. If the bool is true, then IsChecked is true and the box is ticked. If false, then the checkbox is empty.

The weird thing about this is that if you replace variableName with the “user.Roles.Select(r => r.RoleName)”, in the lambda, we get the primitive types error.

**CONCLUSION:**

This leads me to conclude that the problem lies with the “user.Roles” part of the query. This in itself is an IList<Role>. So maybe, you cannot have a list of non-primitive objects performing a query within your lambda expression. But this still does not make any sense to me yet, as we aren’t taking the user.Roles out of the equation, we are just sticking it into a variable.