Assignment 2 – School Community

# Description

This assignment builds off the techniques learned in the first 6 labs of the course and allows you to get creative and create a School Community Web App.

# Estimated Time

This assignment will take an estimated 20 hours to complete.

# Deliverable

1. Create a private git repo and push your code to it and submit the link to Brightspace.
   1. NOTE: A public git repo will result in zero. No exceptions.
2. Deploy your website to Windows Azure and submit the link to Brightspace.
   1. NOTE: Your app must work to get the mark. If your app is failing and you cannot complete all features, make sure you have a partial system working. Use git to version control your code so in case you destroyed a working app you can restore it.
3. Record a video between 2 and 3 minutes and showcase your app. You can create a zoom meeting, demo the lab, and record it. Since this is a lab demo, your webcam should be on and recorded in the submission. Upload it to the cloud provider of your choice (OneDrive, GoogleDrive, Dropbox, etc.) and share the link in Brightspace. Note that if you work as a team, both team-mates should present in the demo. Your video should cover:
   1. What works, and what does not work
   2. Showcase your app. For example,
      1. doing the following actions
         1. Add a community and show the change in community table in the database
         2. Delete a student and show the data in membership and students tables before and after deletion
         3. Upload a new ad and showcase the ad table before and after upload
         4. Delete that ad and show the data changed in ad table and community table
      2. Explaining how you coded the new additions to the students controller

# Notes

* Read the entire assignment document before you get started.
* Use Lab 4 as a starting point.
* Be sure to have completed lab 4 before attempting this assignment. You will find it very difficult and waste a lot of time otherwise.
* Work in teams of maximum two if needed. If you choose to work in teams, you must send me an email with the list of team members before 9am on July 31st.
* This application framework can be intimidating when you are starting out – use all the help you can get without cheating. Plagiarism will be prosecuted.
* Make your app your own by modifying CSS, and playing with different features, without modifying the core functionality.
  + This is optional, but highly recommended. There is a bonus mark of 1 point (this is full 1 point towards your final mark)
* Ask for help. I am quick about responding to questions. However, I will not reply emails that are not accompanied with the github repo link. If you need help, you need to give me enough information so I can help you. If you did not receive any reply after 48 hours, ask yourself whether the information you gave me was enough to help you.
* Start early. You will not be able to get any one-on-one help from me after the last lab session, and it will be only over emails after, and might not be as efficient/effective.
* This assignment is intentionally vague in some parts. You know all the technology required to complete it. This assignment is intended to give flexibility, so you use what you learned in action, the way you see it fit.
* Demo provided at: <http://afrasialab4.azurewebsites.net/>
  + Note that I intentionally deactivated all create, edit, upload, delete actions.

# Application Logic

From here on I will describe functionality and it will be up to you to implement it the best you see fit.

You need to start off from Lab 4, so everything that was needed for lab 4 is a requirement for this assignment. I will not repeat them below. Please review Lab 4 document if you need to. Also, note that if you just submit lab 4 again, you will not receive any marks for this assignment!

## Views

1. Students/Index
   1. You need to add options for the user to:
      1. Select each student and see the communities s/he is a member of
      2. Think whether you need one of the ViewModels in the hints section. My wild guess is that you need it!
      3. Edit the list of community memberships for each student, which happens in Students/EditMemberships
2. Students/EditMemberships
   1. You need to enable the user to register/unregister users to communities
      1. Hint: Clicking Register/Unregister should invoke different actions in StudentsController, called Addmembership and RemoveMembership respectively. These actions should get studentId and communityId as parameters. Well, after all, these form the composite key for the CommunityMembership table. So, in total, you need one view, but three controller actions handling it. You can combine it into one controller action, but why complicate it when there is no need for!?
   2. Hmmm….Looks like a perfect candidate for a ViewModel. Don’t you think so?
   3. The way I have it, the communities that the student is a member of go first, followed by the rest of communities sorted alphabetically. You can change this sorting order, but there must be sorting logic, and your demo should specify how you decided to go with sorting.
      1. Hint: Sorting can be either in your query or using Linq after you extracted results into lists. It is up to you to choose the easier solution (the latter one) 😉
3. Communities/Index
   1. You need to add a link “Ads” that will direct to a page that has a list of images that the community posted on their forum. (we call them ads)
   2. Where do you need to make the change? Controller? View? I think it is more of a View change.
4. Advertisements/Index/{id}
   1. Notice that, for simplicity, we do not have Advertisements/Index; I mean Index not followed by {id}. You need to pass the id of the community you want to see the ads for. This view will show all the ads for community with id in the path ({id}).
   2. You should be able to delete an ad and that will take you to Advertisements/Delete/{ad id} for verification and completing the delete action. Needless to say that you need a delete view here as well.
      1. Let us make it simple, so we do not have to deal with DB constraints. We already have a lot to take care of. For simplicity, let us make sure that we delete the community only if all its ads are deleted. So, we will not allow deletion if the community already has ads.
5. Advertisements/Create/{id}
   1. The user should be able to upload an ad for a community whose id is in the path ({id}).
   2. Do you wonder how to path additional data along with the file you upload? Maybe a quick look into hints and googling hidden input tags would help.

## Hints

1. ViewModels are your best friend. Use them as much as possible. You will probably need another 4 of them for this assignment (if you choose to use ViewModels). Remember that you can put anything in ViewModels, they are just simple classes. For instance, all the followings are valid ViewModels. It does not mean that you need all these and only these viewmodels. It is up to you to decide whether they are needed, or you need more:

public class FileInputViewModel

{

public string CommunityId { get; set; }

public string CommunityTitle { get; set; }

public IFormFile File { get; set; }

}

public class AdsViewModel

{

public Community Community { get; set; }

public IEnumerable<Advertisement> Advertisements { get; set; }

}

public class StudentMembershipViewModel

{

public Student Student { get; set; }

public IEnumerable<CommunityMembershipViewModel> Memberships { get; set; }

}

public class CommunityMembersipViewModel

{

public string CommunityId { get; set; }

public string Title { get; set; }

public bool IsMember { get; set; }

}

1. Opps! Part of the solution is out. But still you need to figure out where and how to use them. Disclaimer: I find the above ViewModels make understanding the solution easier. There are lots of different way of approaching the solution. Using action parameters and model binding, or using viewdata, just to name a few. Just saying that these ViewModels are not the only solution, and you can be creative.
2. Of course, you need to have a model for Advertisement, you should know this by now. Do you remember Lab 5, or you need to look at it again?
3. So how do we add Advertisements to the database we created for Lab 4? You are probably thinking about migrations. You are correct!
   1. Since you already have a db, before making any changes to lab 4, just create a migration, maybe call it “Init”, and delete the content of its Up and Down, so the methods are empty, and then run update-database. Now, your migrations are up to speed with existing database, and the migration history in database is updated as well. Now, you can change your model, create new migrations, and your new migrations will happily work.
   2. If you are intimidated by migrations, as a work around it is ok to create a new db. I do not recommend it though. You will end up spending some time to fix your db after any change to your models. Please learn migrations. Migrations are part of your final exam.
4. Talking about Advertisement table, if you reverse engineer the sample I posted online, looks like it is many-to-one relationship with Community table.
5. How do we show Advertisements now? Do we need AdvertisementController and Upload, Delete, and Index views for Advertisements? I believe you do.
6. Last hint. Google, read different sources, and do not forget about your previous labs and amazing Microsoft Docs website. Also, even if you find the exact solution somewhere, do not copy and paste, learn it, and create your own version out of it.

## Good Luck!