

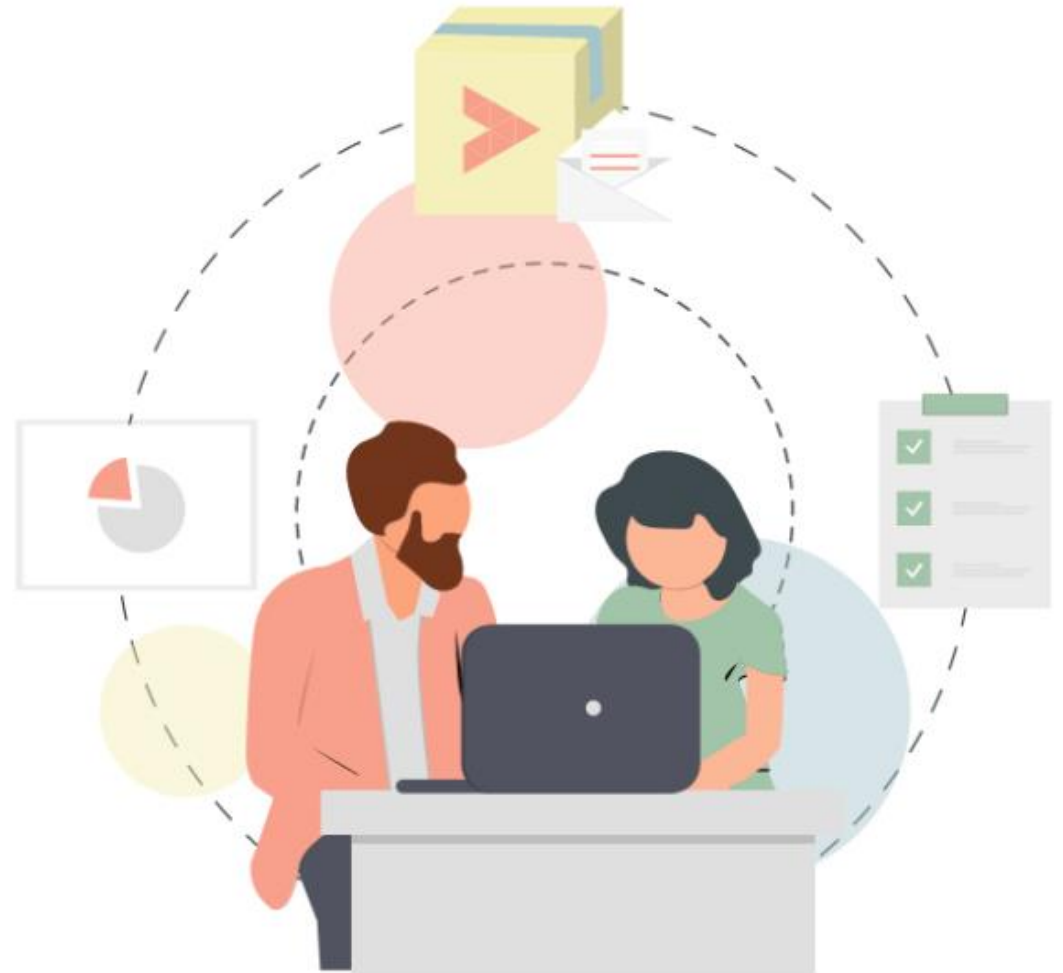


CLASS DASHBOARD WEB APP

Team: Ali Nehme, Rami Chaouki, Ling Tao

<https://studybuddyjac.azurewebsites.net/>

Nov. 14 2022



Contents

- **Background**
- **Tools**
- **Challenges and solutions**
- **What we learned**
- **Future work**
- **Summary**

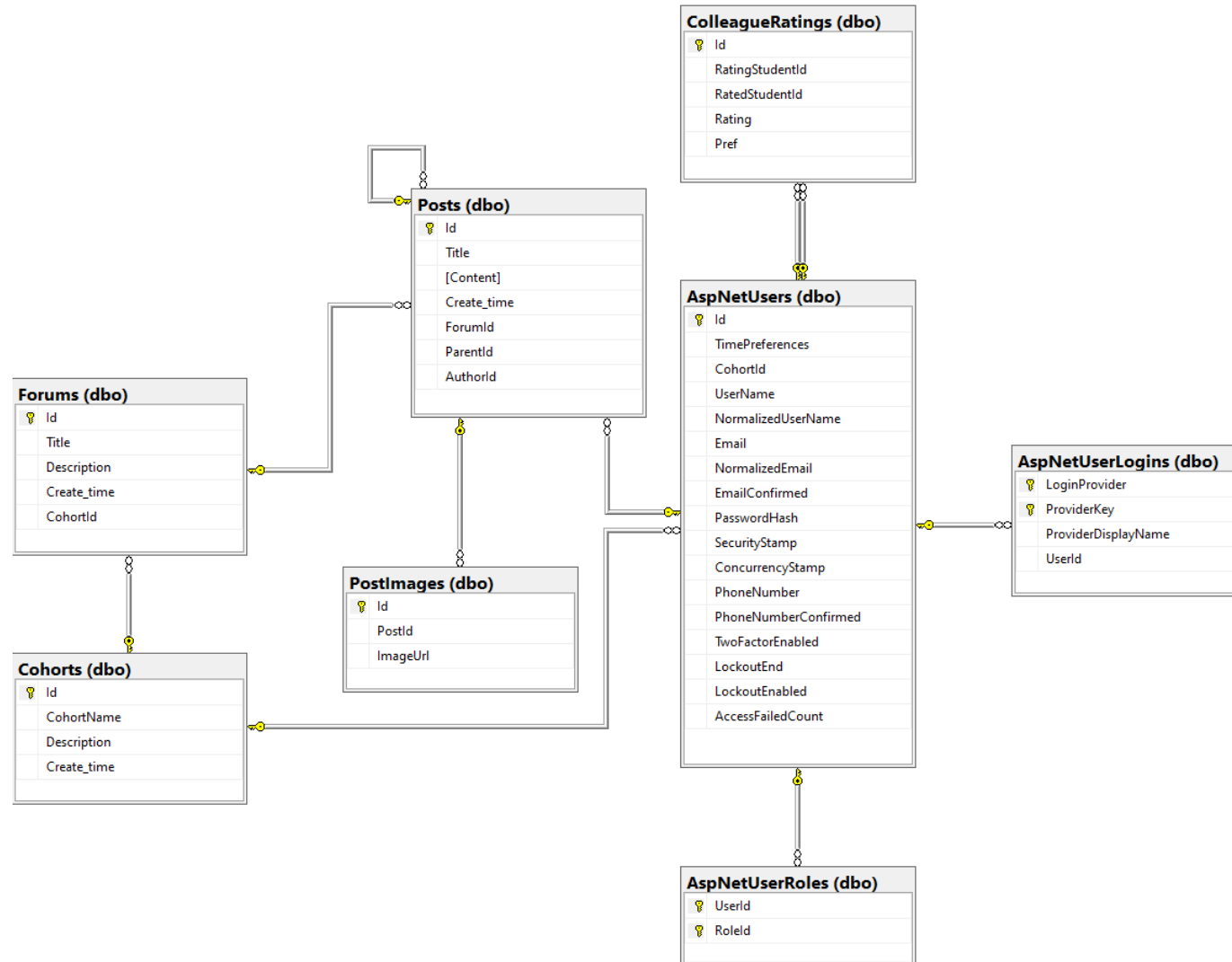


Background

- A web-app that can serve as a hub for John-Abbott students.
- Multiple Forums that supports posts with images though Azure Blobs
- A team builder app which allows students to submit their preferences
- Teachers can automatically generate the teams based on that information
- Admin CRUD functionalities.

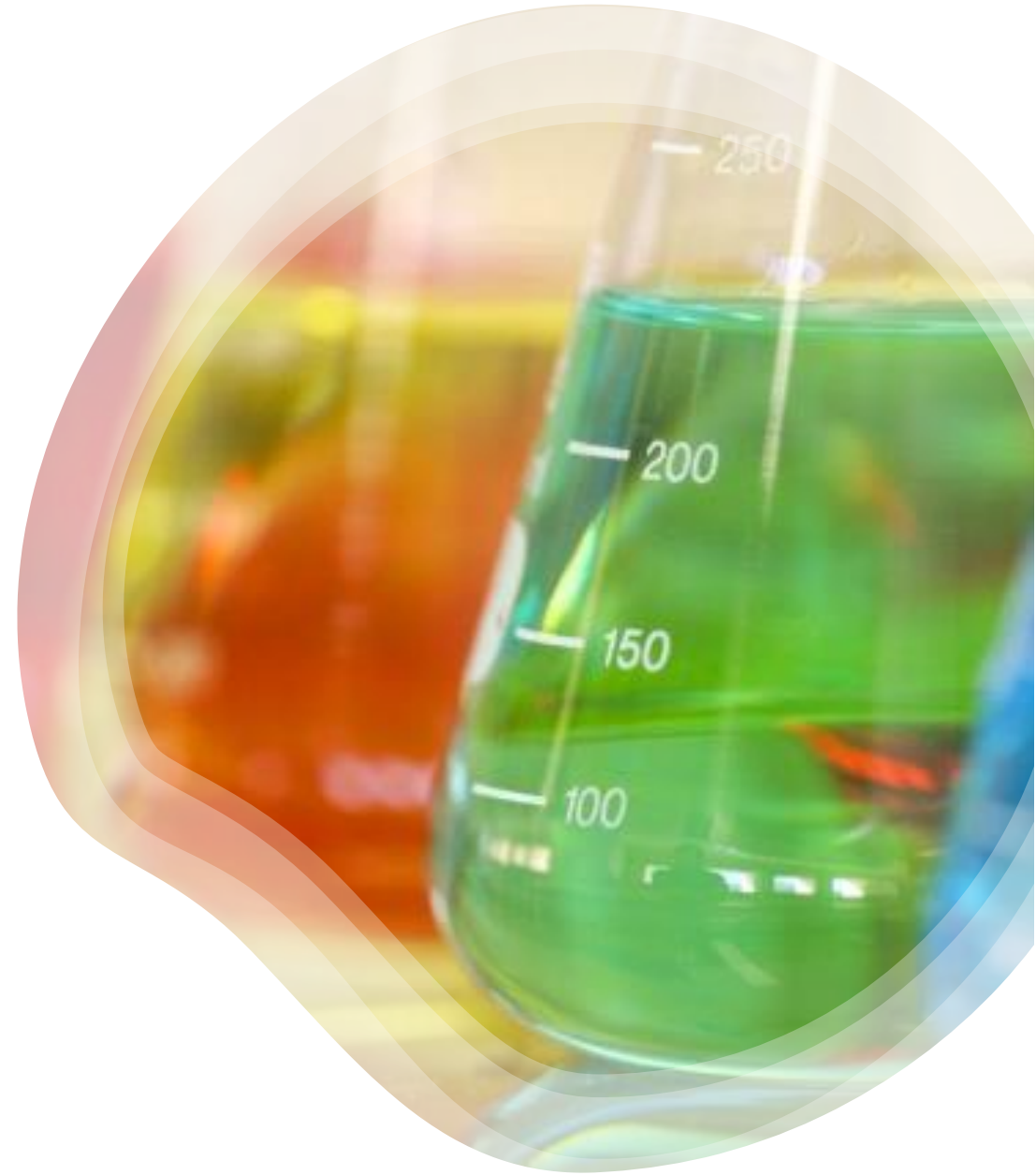


Database



Tools

- Fluent-API
- Azure Blobs
- Azure SQL Server
- Azure Virtual Machine Hosting
- Evolutionary Algorithm for team generation



Home page



[Register](#) [Login](#)

Your Place to Learn and Hang Out



ApplicationUser = IdentityUser with Custom Properties

Gaining flexibility with the powerful services of
IdentityUser

```
55 references
public class ApplicationUser : IdentityUser
{
    4 references
    public string? TimePreferences { get; set; }


    4 references
    public int? CohortId { get; set; }

    [ForeignKey("CohortId")]
    8 references
    public virtual Cohort? Cohort { get; set; }
}
```

```
// add Identity
builder.Services.AddIdentity<ApplicationUser, IdentityRole>()
                .AddEntityFrameworkStores<StudyBuddyDbContext>();
```

```
@inject SignInManager<ApplicationUser> signInManager;
@inject UserManager<ApplicationUser> userManager;
```

Forums page

 StudyBuddy

Student - TeamBuilder Forums Logout Profile

Forums

First Forum

New Post

First Post in First Forum

Second Post in First Forum

First Post in First Forum


Background for Zoom meetings

Background for Zoom meetings

11/14/2022 12:01:13 AM

Sharing background images for zoom meetings

peppas3@gmail.com



Edit Delete

ReplyTitle

ReplyContent

Images: Choose Files No file chosen

Reply

Posts

```
if (!ModelState.IsValid || db.Posts == null)
{
    return Page();
}

Post NewPost = new Post() { Title = Title, Content = Content, Create_time = DateTime.Now, ForumId = Id };
NewPost.Author = db.Users.Where(u => u.UserName == User.Identity.Name).FirstOrDefault();

await db.Posts.AddAsync(NewPost);
var result = await db.SaveChangesAsync();

if (result > 0)
{
    if (FilesToUpload == null || FilesToUpload.Count < 1)
    {
        return RedirectToPage("./NewPostSuccess", new { PostTitle = NewPost.Title });
    }

    foreach (IFormFile UploadFile in FilesToUpload)
    {
        try
        {
            string filename = "post-" + NewPost.Id + "-image-" + DateTimeOffset.Now.ToUnixTimeMilliseconds();
            Console.WriteLine("Uploading: " + filename);
            using (var stream = UploadFile.OpenReadStream())
            {
                await _containerClient.CreateIfNotExistsAsync();

                var blobClient = _containerClient.GetBlobClient(filename);

                // Upload the file to the container
                // var result = await _containerClient.UploadBlobAsync(filename, UploadFile.OpenReadStream());
                var azureResponse = await blobClient.UploadAsync(stream, true);

                Console.WriteLine(azureResponse.GetRawResponse().ReasonPhrase + ": " + filename);
                // lblUploadResults.Add("Uploaded: " + filename);
            }
        }
        catch { }
    }
}
```

Create New Post

Post

Title

Content

Images 2 files



[Back to Forums](#)

Edit Post

Post

Title

Background for Zoom meetings

Content

Sharing background images for zoom meetings

Images zoom5-png-1.webp



[Back to Forums](#)

What We Learned and Challenges

Honorable mentions:

- DeepClones
- Translating DOM manipulation with JS into Models
- Automatic deployment on push using Github Actions

Challenges + learning

5 references

```
private async Task<List<Forum>> DefaultForumsList()
```

```
<div class="@postClass">
  <form asp-page-handler="post" asp-route-id="@post.Id" method="post">
    <button class="@postBtnClass">@post.Title</button>
    <input type="hidden" name="id" value="@post.Id" />
  </form>
</div>
```

```
public async Task<IActionResult> OnPostPost(int id)
{
    ForumsList = await DefaultForumsList();
    ActivePostId = id;

    ActivePost = await db.Posts
        .Include(p => p.PostImages)
        .Include(p => p.Author)
        .Include(p => p.Replies).ThenInclude(reply => reply.PostImages)
        .Where(p => p.Id == ActivePostId)
        .FirstOrDefaultAsync();

    if (ActivePost == null)
    {
        // PostsMessage = $"Nothing to show.";
        return Page();
    }

    ActivePostReplies = ActivePost.Replies;

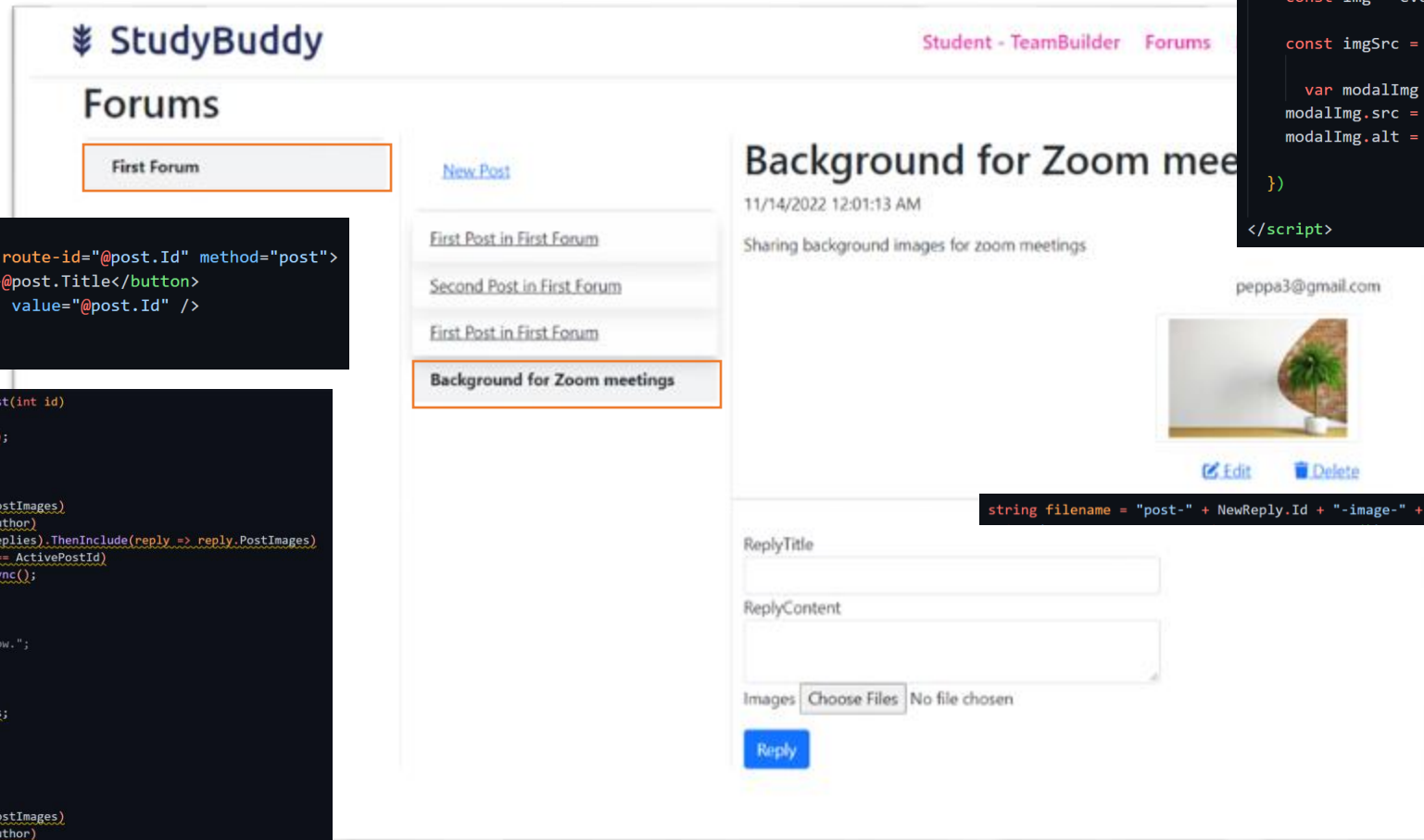
    ActiveForumId = ActivePost.ForumId;
    ActiveForum = ActivePost.Forum;

    PostsList = await db.Posts
        .Include(p => p.PostImages)
        .Include(p => p.Author)
        .Include(p => p.Replies)
        .ThenInclude(reply => reply.PostImages)
        .Where(p => p.ForumId == ActiveForumId && p.ParentId == null)
        .ToListAsync();

    ReplyTitle = "";
    ReplyContent = "";

    ModelState.Clear();

    return Page();
}
```



```
<script>
// Get the image and insert it inside the modal
const imageModal = document.getElementById('imageModal')
imageModal.addEventListener('show.bs.modal', event => {
    const img = event.relatedTarget

    const imgSrc = img.src;

    var modalImg = document.getElementById("modalImg");
    modalImg.src = imgSrc;
    modalImg.alt = imgSrc;
})
</script>
```

```
string filename = "post-" + NewReply.Id + "-image-" + DateTimeOffset.Now.ToUnixTimeMilliseconds();
```

Fluent API

```
modelBuilder.Entity<Post>()
    .HasOne(p => p.Forum)
    .WithMany(f => f.Posts)
    .HasForeignKey(co => co.ForumId)
    .OnDelete(DeleteBehavior.NoAction);

modelBuilder.Entity<Post>()
    .HasOne(p => p.ParentPost)
    .WithMany(p => p.Replies)
    .HasForeignKey(r => r.ParentId)
    .OnDelete(DeleteBehavior.NoAction);

// modelBuilder.Entity<Post>()
//     .HasOne(p => p.Author)
//     .WithMany(a => a.Posts)
//     .HasForeignKey(p => p.AuthorId)
//     .OnDelete(DeleteBehavior.SetNull);

modelBuilder.Entity<PostImage>()
    .HasOne(p => p.Post)
    .WithMany(p => p.PostImages)
    .HasForeignKey(co => co.PostId)
    .OnDelete(DeleteBehavior.Cascade);
```

- Reduce DB traffic
- On-demand fetching child entities with lazy loading
- Accessibility of grandchildren entities with one call to DB

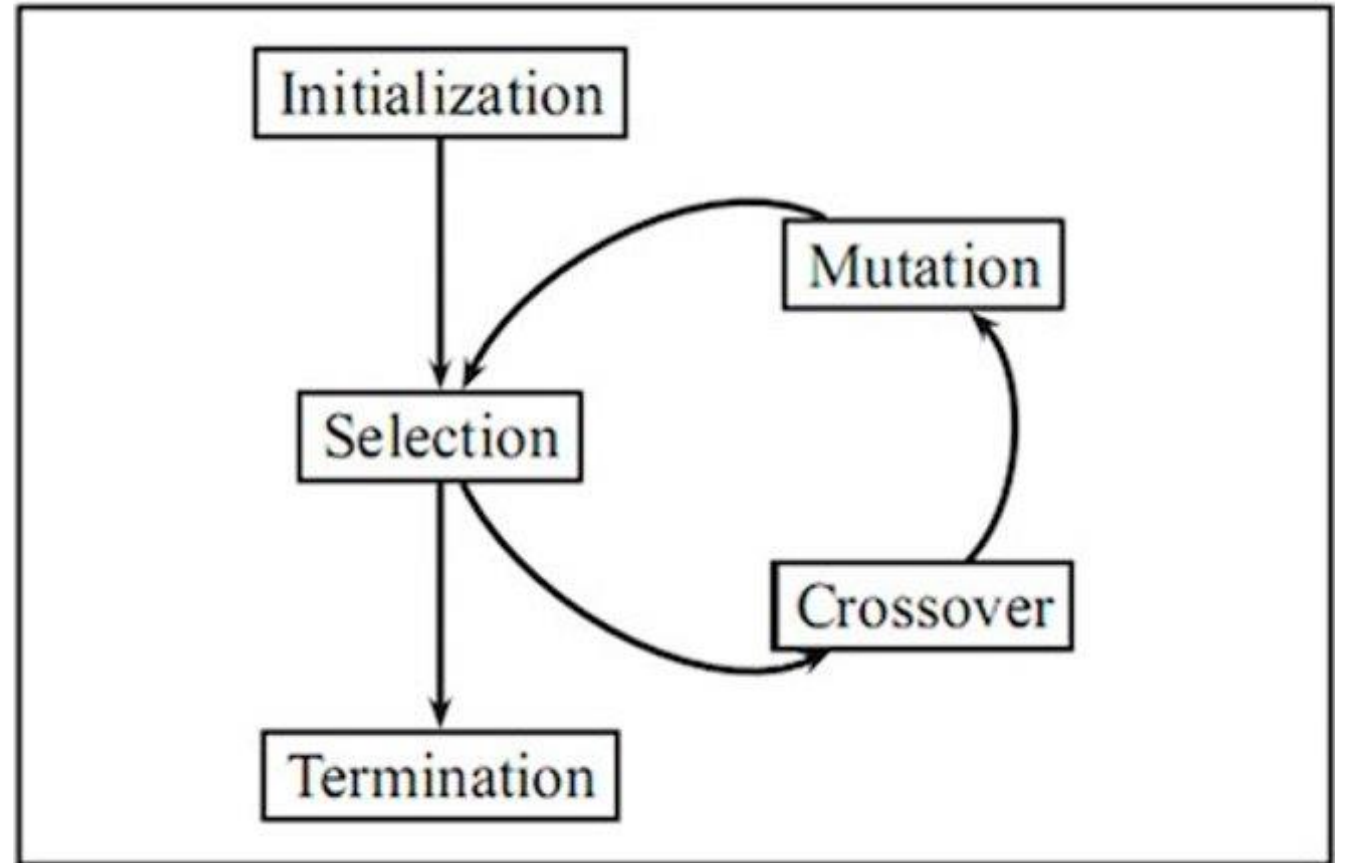
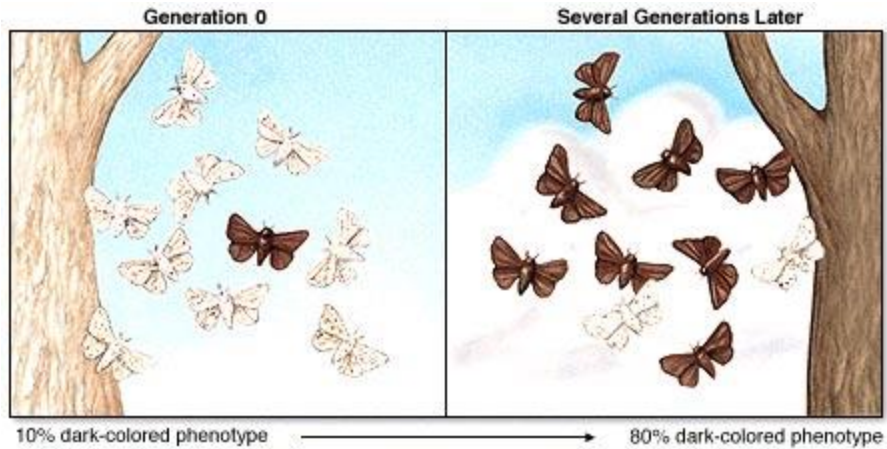
```
ActivePost = await db.Posts
    .Include(p => p.PostImages)
    .Include(p => p.Author)
    .Include(p => p.Replies).ThenInclude(reply => reply.PostImages)
    .Where(p => p.Id == ActivePostId)
    .FirstOrDefaultAsync();

if (ActivePost == null)
{
    // PostsMessage = $"Nothing to show.";
    return Page();
}

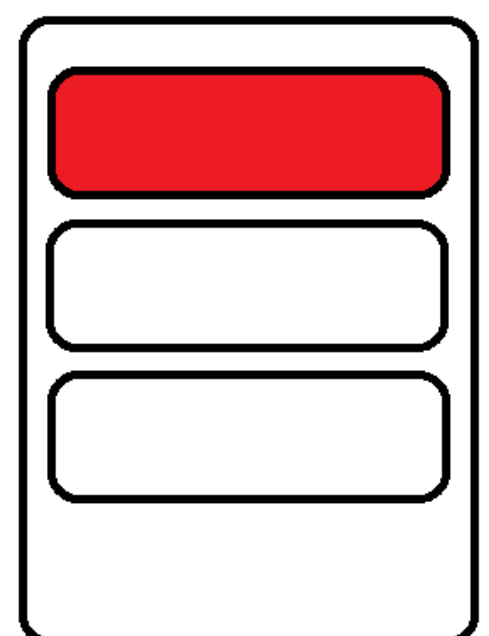
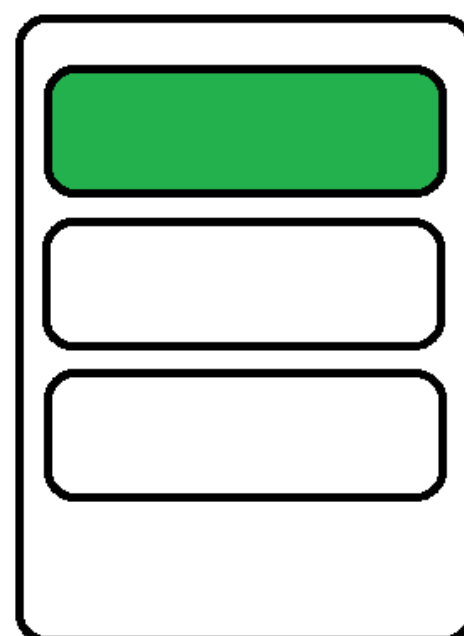
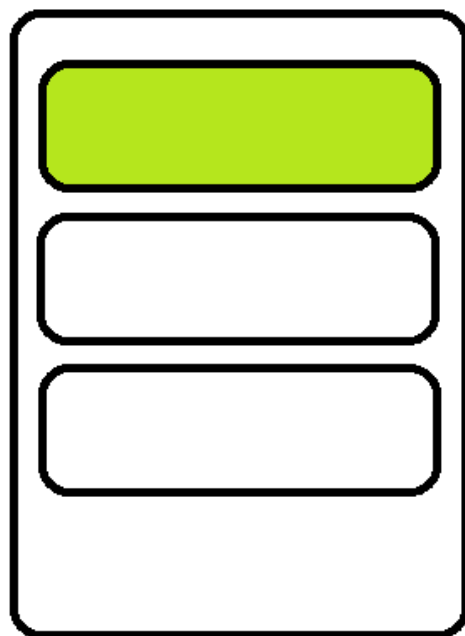
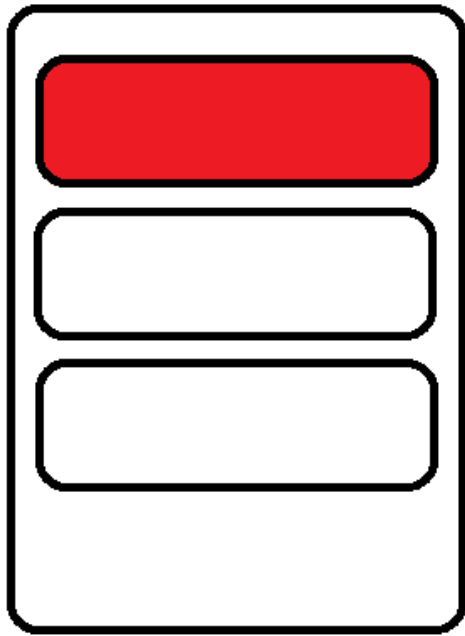
ActivePostReplies = ActivePost.Replies;
```

```
/* replies */
if (Model.ActivePostReplies == null)
{
    <div class="row">
        <hr />
        <p>Be the first to reply.</p>
    </div>
}
else
{
    @foreach (var reply in Model.ActivePostReplies)
    {
        <div class="row">
            <h3>@reply.Title</h3>
            <p>@reply.Create_time</p>
            <p>@reply.Content</p>
            <p class="@AuthorClass">@reply.Author</p>
            <div class="row d-flex flex-row-reverse">
                @if (reply.PostImages != null && reply.PostImages.Count > 0)
                {
                    foreach (var image in reply.PostImages)
                    {
```

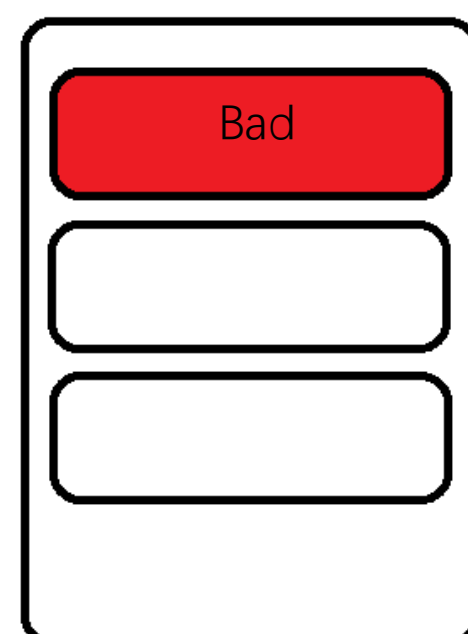
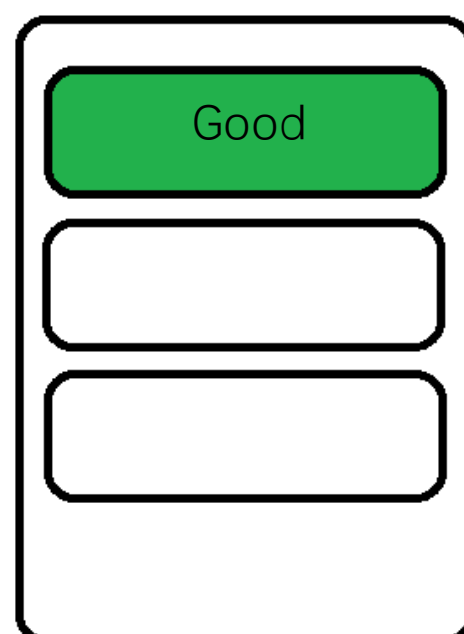
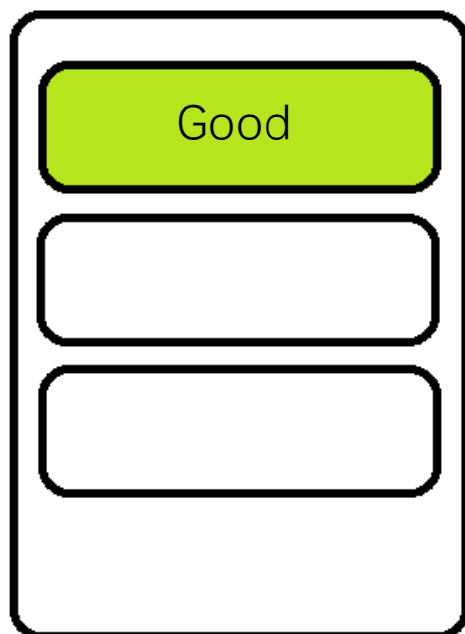
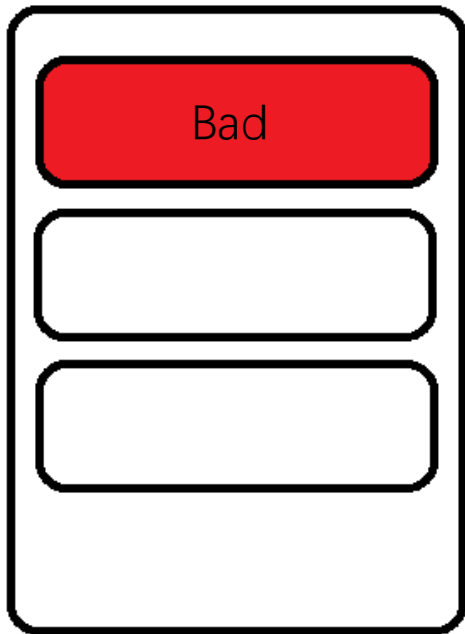
Evolutionary Algorithm



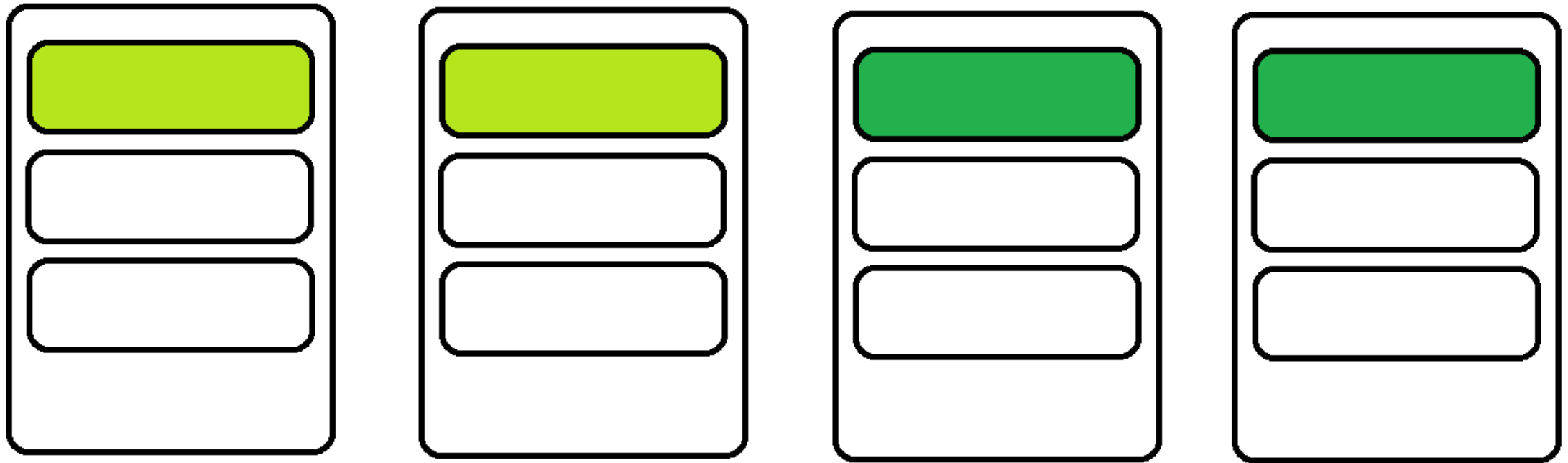
Initialization



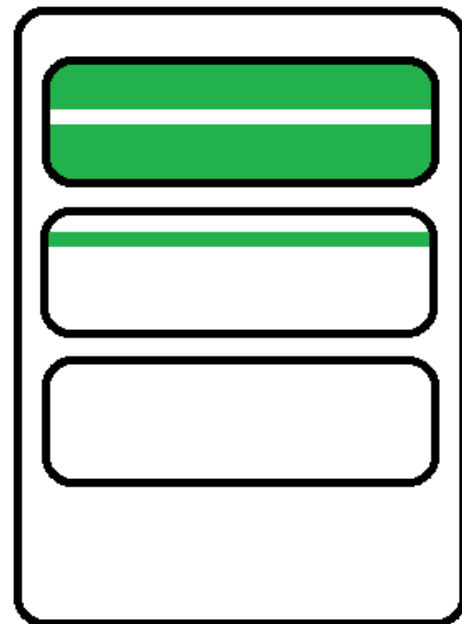
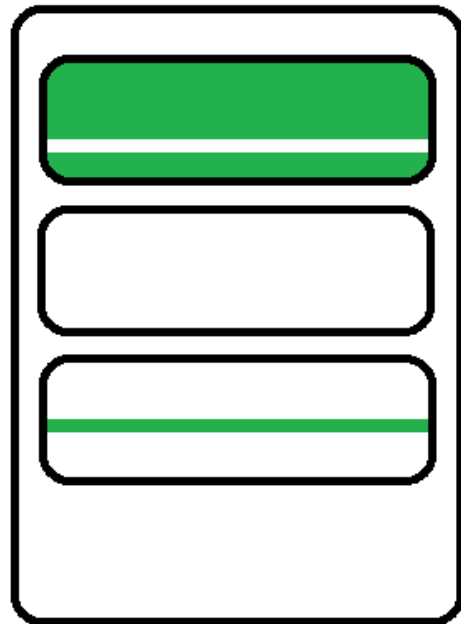
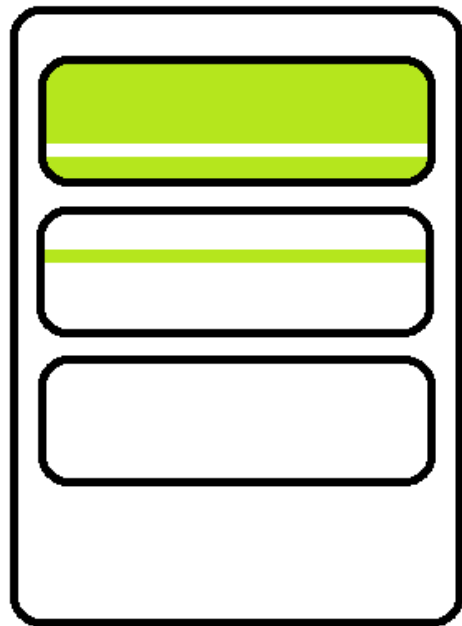
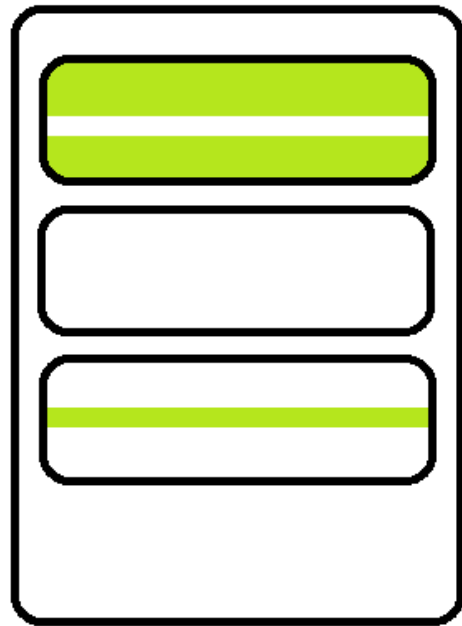
Selection



Crossover



Mutation

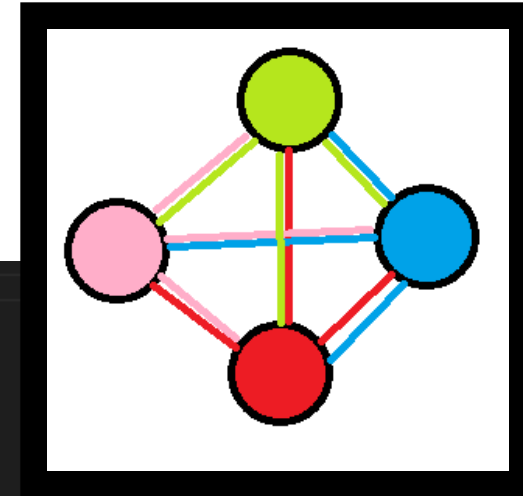


Optimizing Data Structures according to functions

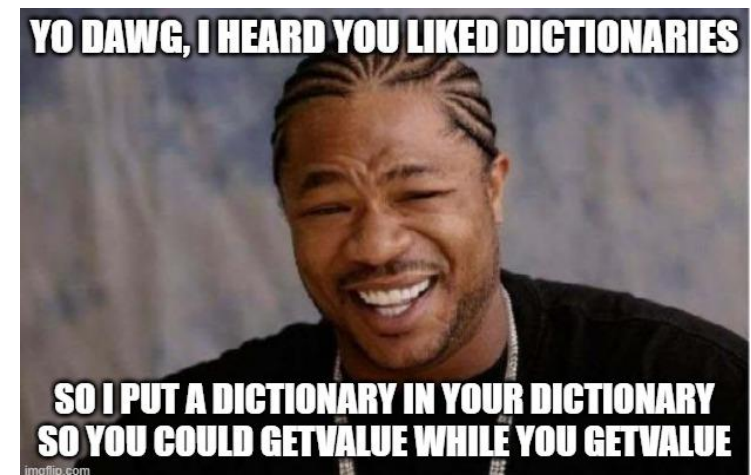
- Nested dictionaries – like the excel sheet

```
Reference
public void CalculateTeamRating(Dictionary<string,Dictionary<string,int>> ReferenceDictionary)
{
    TeamRating=0;//initializes TeamRating
    foreach(string t1 in Teammates)
    {
        foreach(string t2 in Teammates)
        {
            if(t1.Equals(t2))continue;//skips the student comparing himself

            TeamRating+=ReferenceDictionary.GetValueOrDefault(t1).GetValueOrDefault(t2); //Finds rating of t1 towards t2 in O(1) time, default of an int, which is 0
        }
    }
}
```



```
foreach(ApplicationUser s in Students)
{
    Dictionary<string,int> SubDictionary= new Dictionary<string,int>();
    StudentUsernames.Add(s.UserName);
    foreach(ColleagueRating r in s.RatingStudents)
    {
        SubDictionary.Add(r.RatedStudent.UserName,r.Rating);
    }
    ReferenceDictionary.Add(s.UserName,SubDictionary);
}
```



Optimizing Data Structures according to functions

- HashSets

```
4 references
public Classroom Crossover(int teamToKeep)
{
    //Identify index of highest rated team
    //int largestTeamRatingIndex=Int32.MinValue;
    //int largestTeamRating=Int32.MinValue;
    //Ranks teams by descending order - i.e. the largest is at index [0]
    List<Team> orderedTeam=new List<Team>(this.Teams.OrderBy(t=>-1*t.TeamRating).ToList());

    // for(int i=0;i<Teams.Count();i++)
    // {
    //     if(Teams[i].TeamRating>largestTeamRating)
    //     {
    //         largestTeamRatingIndex=i;
    //         largestTeamRating=Teams[i].TeamRating;
    //     }
    // }

    //Create a HashSet that EXCLUDES the members of the highest rated team (for loop)
    HashSet<string> studentsToExclude=new HashSet<string>(Usernames);

    //Loop through the highest rated team and removes it's students from the studentsToExclude set
    foreach(string t in orderedTeam[teamToKeep].Teammates)
    {
        studentsToExclude.Remove(t);
    }
}
```

Optimizing Data Structures according to functions

- Tuples!

```
//SANITIZE INPUT: AVAILABILITES -- AVOID -- PREFER
(string avail, HashSet<string> avoid, HashSet<string> prefer) sanitizedInput = SanitizeInputs(AreChecked,AvoidPseudoList,PreferredPseudoList);
```

```
foreach(ApplicationUser c in currUserColleagues)
{
    if(!ratingCurrUserCR.ContainsKey(c.UserName)) //if this search returns false, it means there is no ColleagueRating for this classmate
    {
        if(sanitizedInput.prefer.Contains(c.UserName)) //Uses hashsets to determine in O(1) if the user to be created is preferred, avoid or neutral
        {
            await CreateRating(currentUser.Id,c,sanitizedInput.avail,Preferences.Prefer);
        }
        else if (sanitizedInput.avoid.Contains(c.UserName))
        {
            await CreateRating(currentUser.Id,c,sanitizedInput.avail,Preferences.Avoid);
        }
        else
        {
            await CreateRating(currentUser.Id,c,sanitizedInput.avail,Preferences.Neutral);
        }

        //CREATE OPPOSITE CR WHERE RATED RATES CURR USER
        // await CreateRating(c.Id,currentUser,c.TimePreferences,Preferences.Neutral);// TOFIX: Risk of Overwriting existing CR because

    }
}

currentUser.TimePreferences=sanitizedInput.avail;
// await userManager.UpdateAsync(currentUser);
await db.SaveChangesAsync();
// await OnGetAsync(currentUser,currUserColleagues);
await OnGetAsync();
return Page();
```

Optimizing Data Structures according to functions

- Stacks

```
//Shuffle remaining students
Stack<string> studentsToExcludeStack = new Stack<string>(studentsToExclude.OrderBy(s=>rnd.Next()).ToList());
try{
    //Repopulate Teams -- Starts from second largest team i=1
    for(int i=0;i<orderedTeam.Count();i++)
    {
        //skips modifying the team to keep
        if(i==teamToKeep)continue;
        for(int teamsize=0;teamsize<orderedTeam[i].TeamSize;teamsize++)
        {
            orderedTeam[i].Teammates[teamsize]=studentsToExcludeStack.Pop();
        }
    }
}catch(InvalidOperationException e)
{
    Console.Error.WriteLine(e.Message);
}
//Create a Classroom object
Classroom crossoverOtherClassroom = new Classroom(this.Usernames,orderedTeam);
return crossoverOtherClassroom;
}
```

Admin Dashboard

Admin

Users



Cohorts



Forums



Home

Dashboard

Overview ⓘ

50

Users

[View Users](#)



4

Cohorts

[View Cohorts](#)



2

Forums

[View Forums](#)



Forums Table

ID	Title	Description	Cohort Name	Action
1	First Forum	This is the first forum	"Not assigned yet"	Delete
2	Second Forum	This is the Second Forum	"Not assigned yet"	Delete

Cohorts Table

[+ Add New Cohort](#)

ID	CohortName	Description	Create time	Action
1	FSD-1	FSD-1 class of 2022	2022-11-07 3:45:35 PM	Delete
2	FSD-2	FSD-2 class of 2022	2022-11-07 3:45:35 PM	Delete
6	FSD-3	FSD-3 class of 2022	2022-11-09 12:13:49 PM	Delete
7	PeppaTown-Primary	Students from PeppaTown-Primary School!	2022-11-13 2:42:27 PM	Delete

Users Table

ID	Username	Email	Role	Cohort	Action	
007e2...	EmilyElep...	EmilyElep...	Student	PeppaTown-Primary	<button>Update</button>	<button>Delete</button>
011ace...	MandyMo...	MandyMo...	Student	PeppaTown-Primary	<button>Update</button>	<button>Delete</button>
0d78c...	Student8...	Student8...	Teacher	FSD-1	<button>Update</button>	<button>Delete</button>
1072e...	Student2...	Student2...	Student	FSD-1	<button>Update</button>	<button>Delete</button>

[BindProperty]

2 references

```
public IList<UserRolesViewModel> model { get; set; } = new List<UserRolesViewModel>();
```

4 references

```
public class UserRolesViewModel
```

```
{
```

4 references

```
public string Id { get; set; }
```

2 references

```
public string Username { get; set; }
```

2 references

```
public string Email { get; set; }
```

3 references

```
public IEnumerable<string> Role { get; set; }
```

3 references

```
public string CohortName { get; set; }
```

```
}
```

```
public async Task OnGetAsync()
{
    usersList = await db.Users.Include("Cohort").ToListAsync();

    foreach (ApplicationUser user in usersList)
    {
        UserRolesViewModel urv = new UserRolesViewModel()
        {
            Id = user.Id,
            Username = user.Username,
            Email = user.Email,
            Role = await userManager.GetRolesAsync(user)
        };
        if (user.Cohort != null)
        {
            urv.CohortName = user.Cohort.CohortName;
        }
        model.Add(urv);
    }
}
```

Frontend Layout

Home Page

 StudyBuddy

[Register](#) [Login](#)

Your Place to Learn and Hang Out

Admin Page

 StudyBuddy

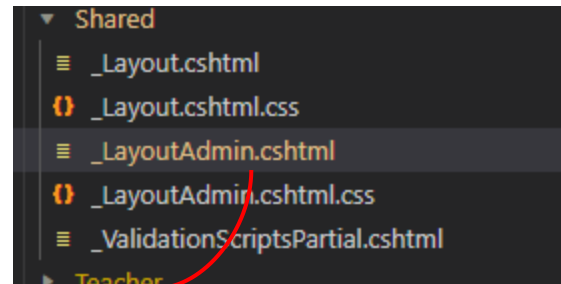
Admin

Users

Cohorts

Forums

Home



```
<!-- partial -->
<div class="container-fluid page-body-wrapper">
  <!-- partial:partials/_sidebar.html -->
  <nav class="sidebar sidebar-offcanvas" id="sidebar">...
</nav>
  <!-- partial -->
  <div class="main-panel">
    <div class="content-wrapper">
      @RenderSection("ContentSection")
      @* @RenderBody() *@
    </div>
  </div>
  <!-- main-panel ends -->
</div>
<!-- page-body-wrapper ends -->
```

```
@page
@model StudyBuddy.Pages.Admin.Users.ViewUsersModel
@{
  ViewData["Title"] = "Users List";
  Layout = "~/Pages/Shared/_LayoutAdmin.cshtml";
}

@section styles
{
  <link rel="stylesheet" type="text/css" href="~/css/admin.css" />
}

@section ContentSection {
  <div class="page-header">...
</div>

  <div class="col lg-12 grid-margin stretch-card">...
</div>
}
```




Future Work

- Beautify frontend pages
- Having a live chat
- Rich text posts
- Search posts
- Delete images in container on post delete
- Use threads to generate classrooms in parallel
- Ability to opt-out of team building

Summary

- Users can register, login.
- Students can save their team preferences
- Teachers can create teams of custom sizes taking into account the preferences
- Students and teachers can post text and images in the forums
- Admin can assign user as teacher or student
- Admin can edit user, cohort and forum info



