Martial Arts S.A.T Documentation

By Kendra Allen and Jasmyne Boggs

S.A.T Requirements

An internal app for our scheduling team

1. Data driven ASP.NET MVC App
2. Two tiers
3. Full crud for all tables
4. Students, scheduled classes, enrollments, student statuses (lookup table), courses, course statuses (look up table/ no controller)
5. Three level of users, identity
    a. Anonymous user
    b. Scheduling user (support staff)
    c. Admin user (manger)
6. Controllers
    a. Home Controller
        i. Index (Anonymous, scheduler, admin)
        ii. Contact (Anonymous, scheduler, admin)
    b. Courses Controller
        i. Views
            1. Edit (Admin)
            2. Create (Admin)
            3. Delete (Admin)
            4. Index (Admin)
            5. Details (Admin)
    c. Students Controller
        i. Views
            1. Index (Scheduler) (Admin)
            2. Create (Admin
            3. Edit (Admin)
            4. Delete (Admin)
            5. Details (Scheduler) (Admin)
    d. Scheduled Classes Controller
        i. Views
            1. Index (scheduler) (admin)
            2. Create (scheduler) (admin)
            3. Edit (scheduler) (admin)
            4. Details (scheduler) (admin)
            5. Delete (admin)
    e. Student Status Controller
        i. Views
            1. Index (admin)
            2. Create (admin)

3. Edit (admin)
4. Details (admin)
5. Delete (admin)

f. Enrollments Controller
    i. views
        1. Index (admin, scheduler)
        2. Edit (admin, scheduler)
        3. Details (admin, scheduler)
        4. Delete (admin, scheduler)
        5. Create (admin, scheduler)

Team Members

- Kendra Allen – assisted converting template and view styling.
    - Completed:
        - Filtering/Searching/Paging
        - Metadata
        - Custom Properties
        - Soft Delete
        - Scaffolded and locked down courses, enrollments, student statuses controller

- Jasmyne Boggs – assisted converting template and view styling
    - Completed:
        - A functioning contact form
        - Image upload
        - Tile layout and toggle for Student Status index
        - Scaffolded and lock down the students controller
        - Styling controller views
        - Implemented Identity

Development Process

This web application was completed by pair programming.

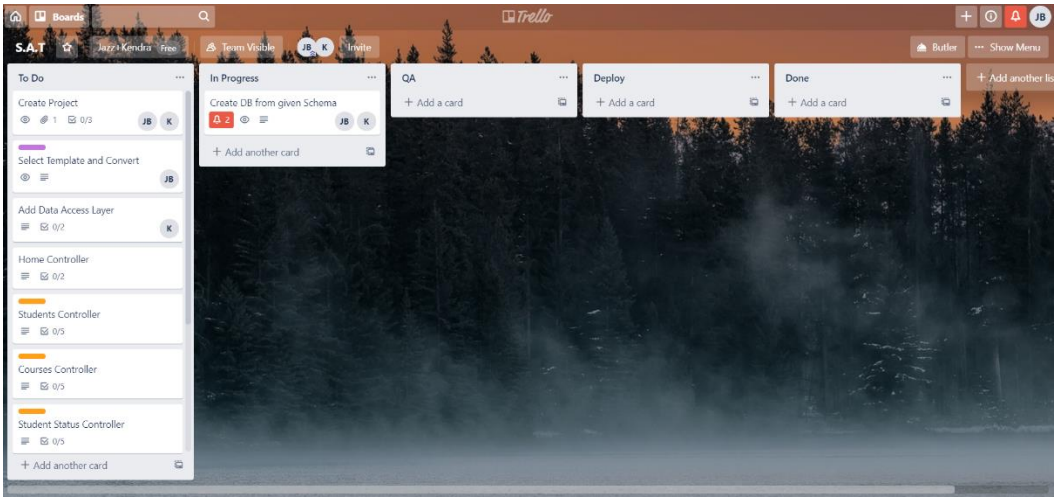
Trello was used to plan and divide tasks to tackle initial requirements. We used the Agile methodology and implemented it via Kanban management system. The team added more specificity to the cards by adding colored labels to cards.

Green – template conversion

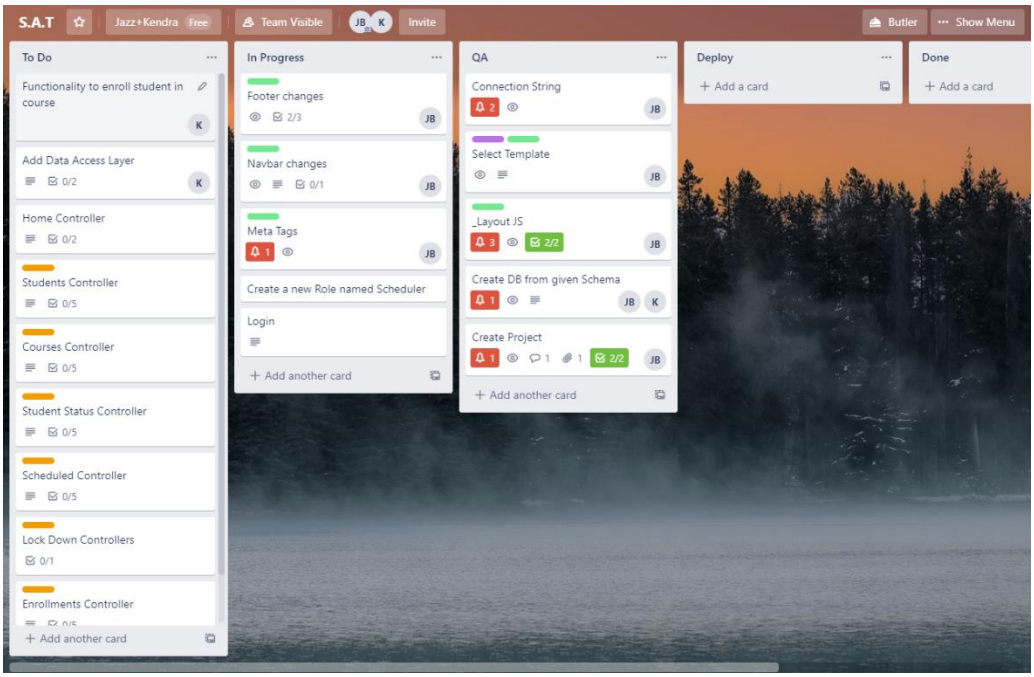
Blue – styling

Yellow – required data access label

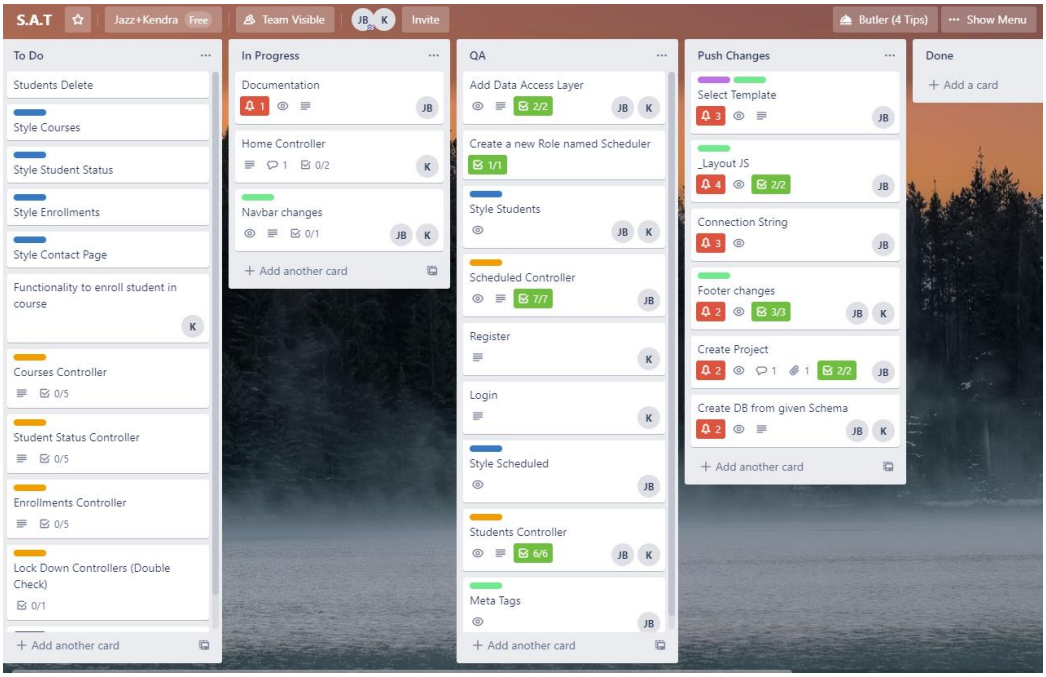
Navy Blue – bonus functionality



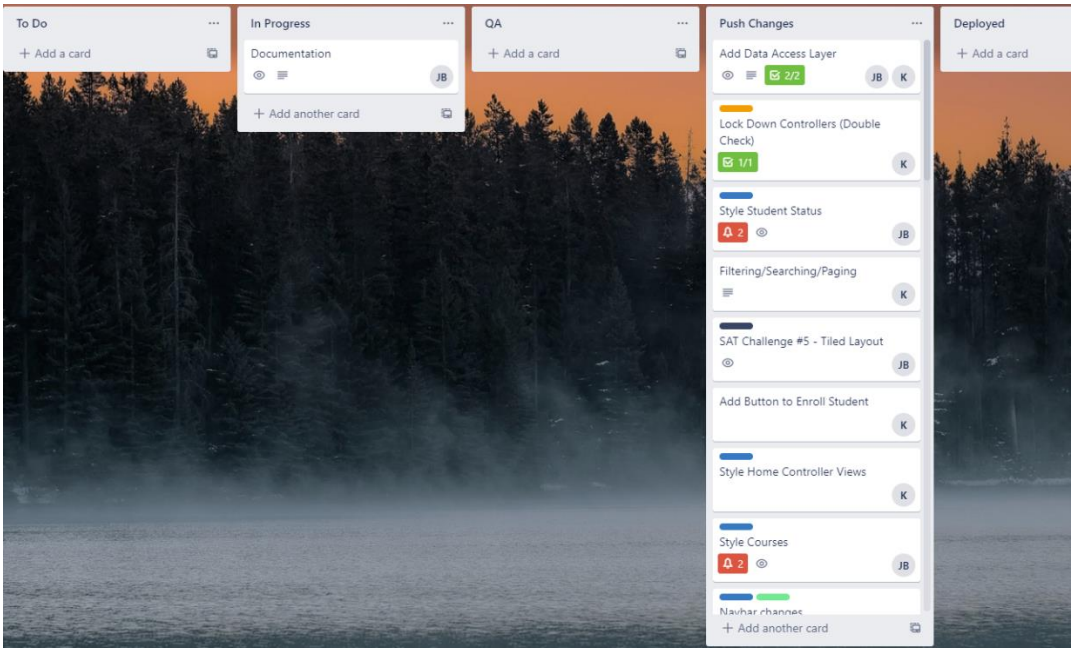
As the project progressed the larger tasks were split into additional cards to tackle specific features. The cards moved to right which meant they were completed.



To continually improve we changed around our Trello board to include a column for all the changes we have pushed to our remote repository.



Once the project was complete all of cards/task were pushed to the remote repository.



Technical

This a data driven ASP.NET MVC 5 application with identity NuGet package.

We completed base functionality by scaffolding out our controllers; however, we went beyond with a few challenges.

1. Custom Properties: Kendra added a customer property to format the first and last name together

```
[MetadataType(typeof(StudentMetadata))]
public partial class Student
{
    [Display(Name = "Name")]
    public string FullName
    {
        get { return FirstName + " " + LastName; }
    }

}
```

2. Image Upload: Jasmyne added functionality to upload a profile picture of the student

```
string imgName = "default.jpg";

if (profilePic != null)
{
    string[] goodExts = { ".jpg", ".jpeg", ".gif", ".png" };

    string ext = imgName.Substring(imgName.LastIndexOf('.'));

    if (goodExts.Contains(ext.ToLower()))
    {
        imgName = Guid.NewGuid() + ext;

        profilePic.SaveAs(Server.MapPath("~/Content/images/" + imgName));
    }
    else
    {
        imgName = "default.jpg";
    }
}

student.PhotoUrl = imgName;
```

3. Soft Delete: Kendra added a soft delete

```
Student student = db.Students.Find(id);
if (student.SSID == 2) //if student status is current
{
    db.Students.Find(id).SSID = 3;
    db.SaveChanges();
    return RedirectToAction("Index");
}
else if (student.SSID == 3) //if student is withdrawn
{
    db.Students.Find(id).SSID = 2;
    db.SaveChanges();
    return RedirectToAction("Index");
}
else
{
    return RedirectToAction("Edit", id);
}
```

4. Tiled Layout: Jasmyne added this to the student status index with the ability to toggle back to list view

```
@foreach (var item in Model)
{

    <div class="menus d-flex ftco-animate">
        <div class="text">
            <div class="d-flex">
                <div class="one-half">
                    <a href="@Url.Action("Edit", "StudentStatus", new { id = item.SSID})">
                        <h3>@Html.DisplayFor(modelItem => item.SSName)</h3>
                    </a>
                </div>
                <div class="one-forth ">
                    @Html.ActionLink("Delete", "Delete", new { id = item.SSID }, new { @class = "btn btn-default" })
                </div>
            </div>
            <p>  @Html.DisplayFor(modelItem => item.SSDescription)</p>
        </div>
    </div>

}
```

```
public ActionResult TiledIndex()
{
    return View(db.StudentStatuses.ToList());
}
// GET: StudentStatus/Details/5
```

5. Kendra implemented filtering, searching, and pagination on all controller's index views by adding the jQuery Data Table plugin.

```
@section scripts {
    <script>
        $(document).ready(function () {
            $('.data-table').DataTable();
        });
    </script>
}
```

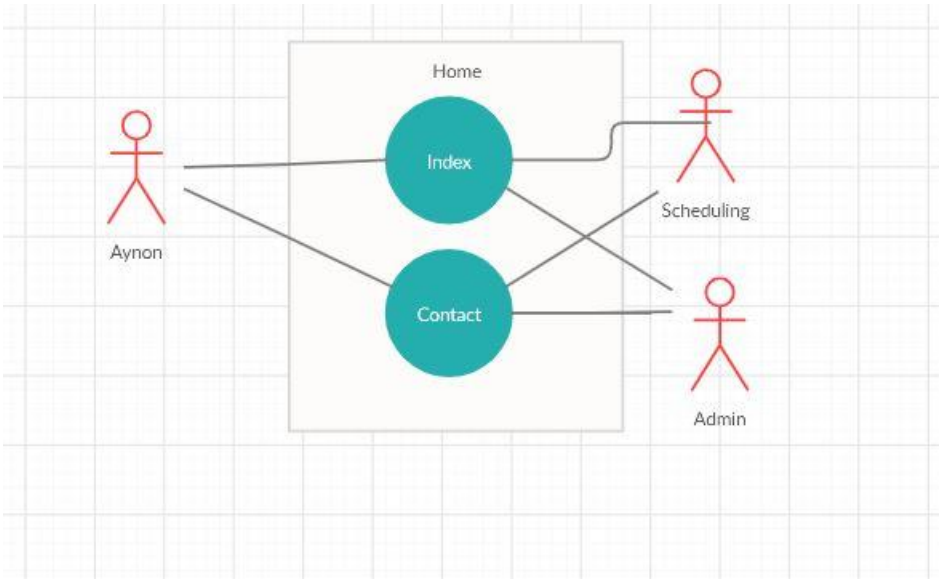
User

- Users

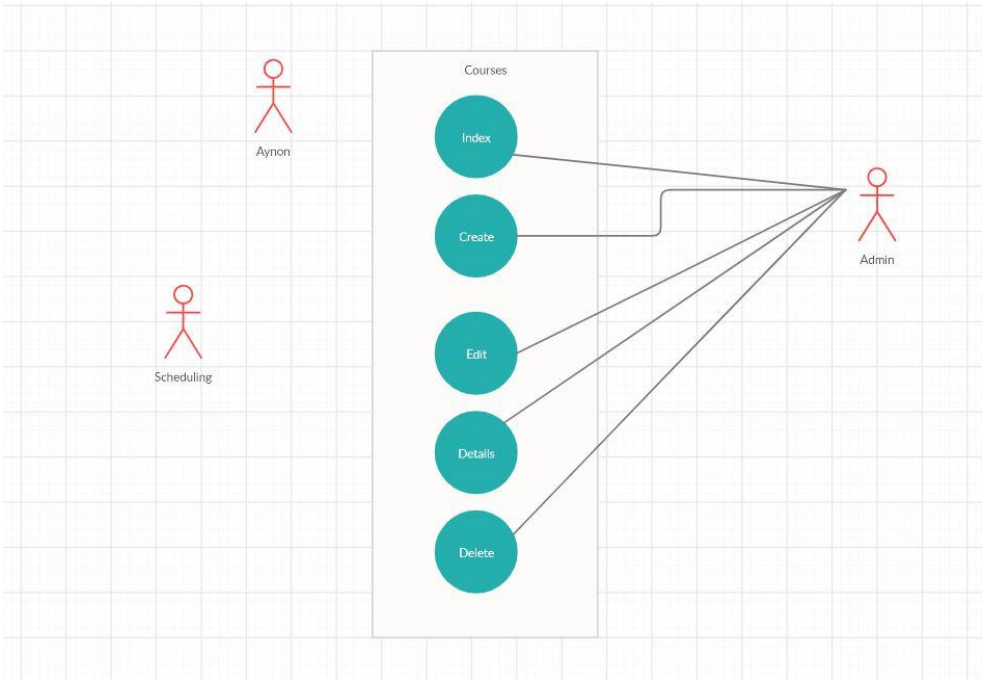
- Anonymous
- Scheduler
- Admin

The following a visual representation of user access:

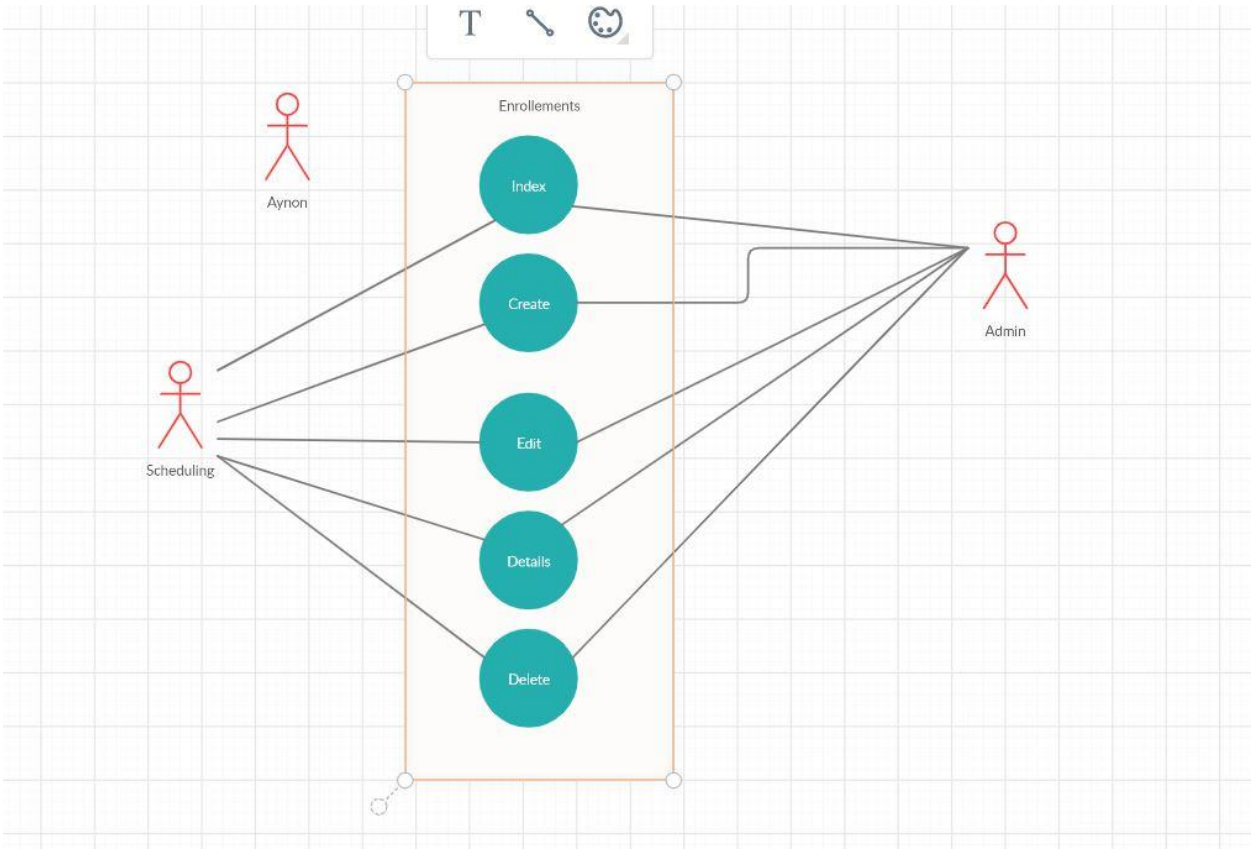
Home Controller Views



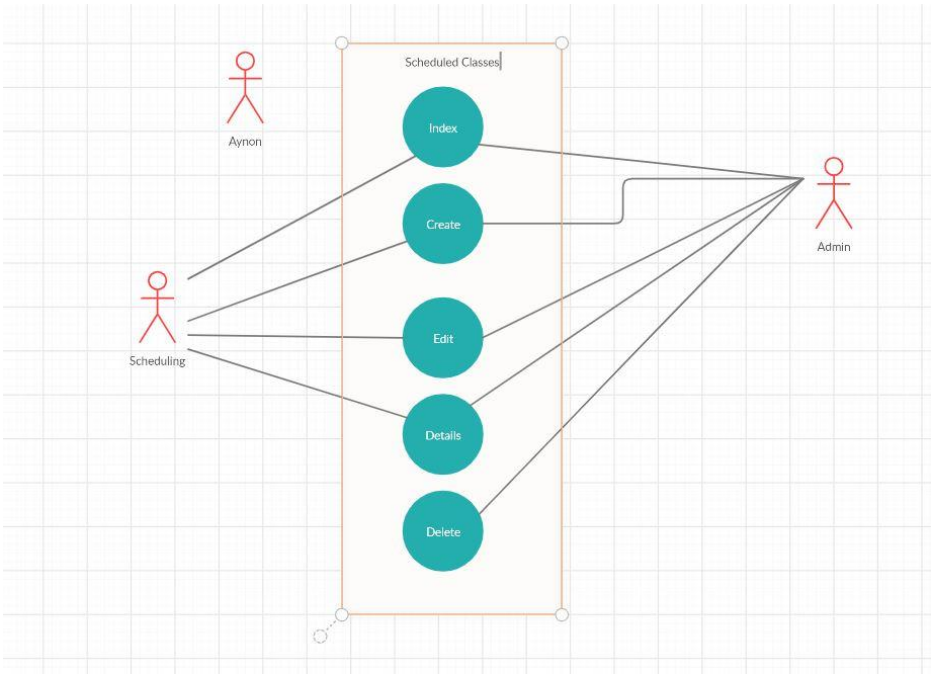
Courses Controller Views



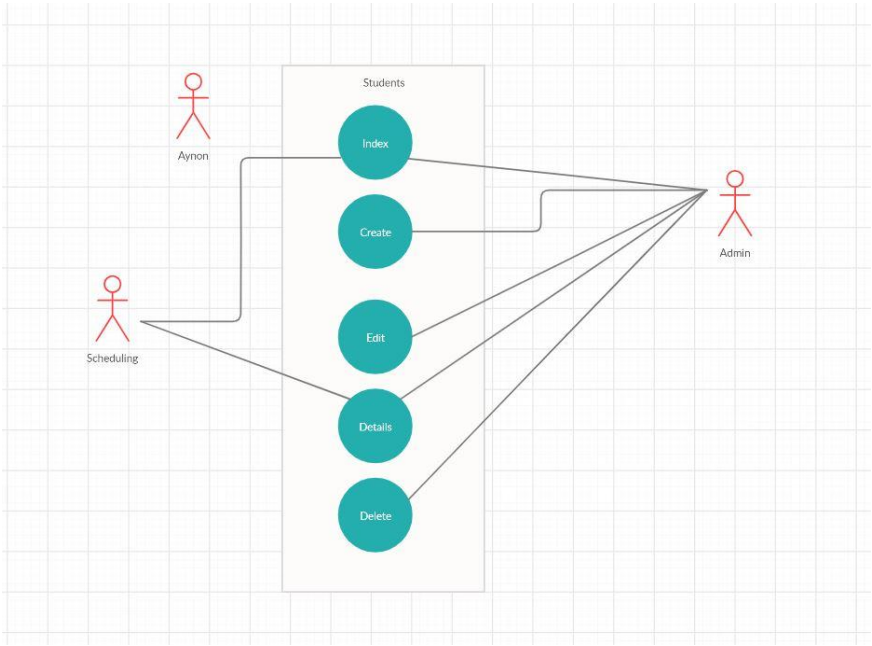
Enrollments Controller Views



Scheduled Classes Controller View



Students Controller Views



Student Status Controller Views