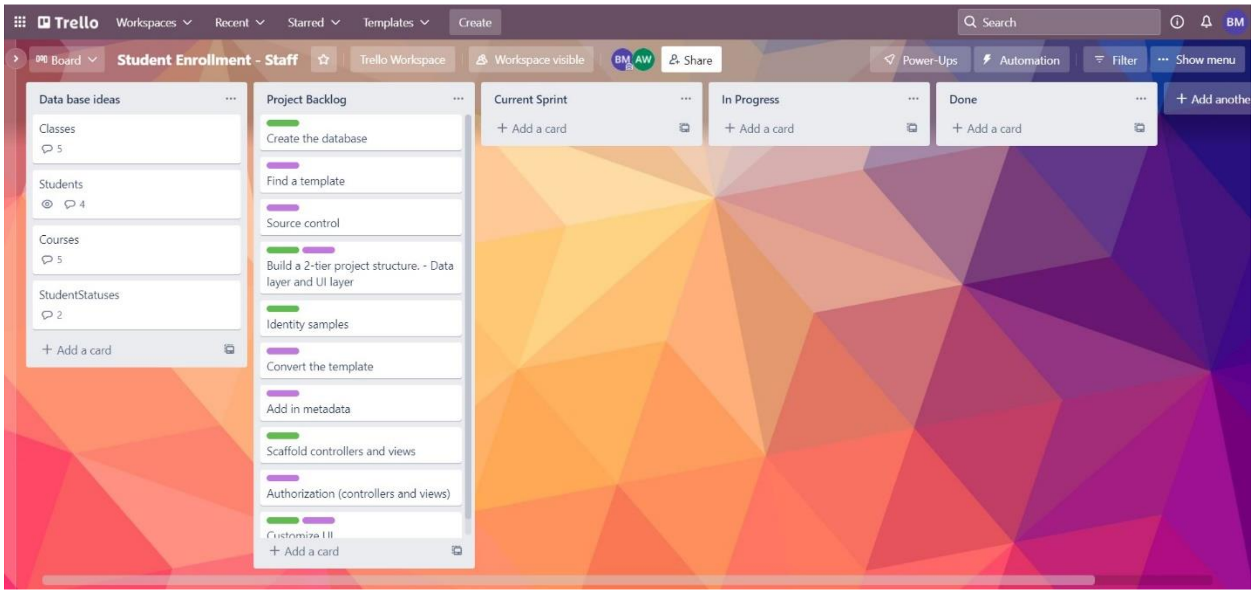
Austin and I were given an assignment from ABC University and local Funks Music store, who have partnered for their music program, to create a working web application that allows Funks employees the ability to set up and manage the courses they offer and the students enrolled. The client wanted to have access to student information, course details, scheduling and class details, and the ability to enroll students into their course. A database was created for information to be saved. We were given two types of users that would have password access to the site and the specific permissions (view, create, edit, and/or delete) for each. The client also wanted a charming home page and a contact page for students to apply for enrollment. We planned out, delegated, and tracked our work using Trello. We also documented our progress with screenshots shown below;

Trello board before initiating our work. The green bars indicate items that Benjamin would be focusing, and the purple bars indicated items that Austin would be focusing.



Data base ideas
- Classes 5
- Students 4
- Courses 5
- StudentStatuses 2
- + Add a card

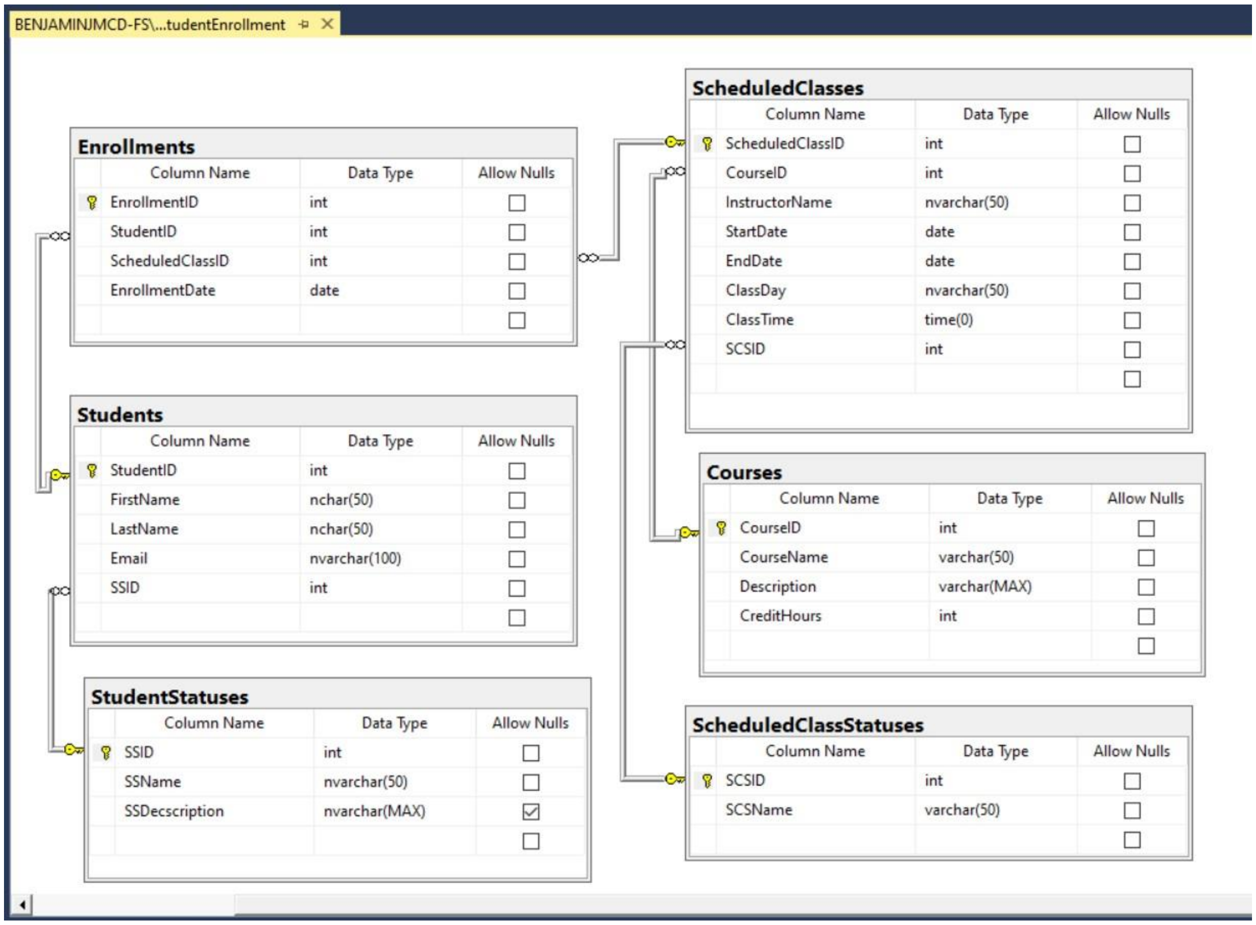
Project Backlog
- Create the database
- Find a template
- Source control
- Build a 2-tier project structure. - Data layer and UI layer
- Identity samples
- Convert the template
- Add in metadata
- Scaffold controllers and views
- Authorization (controllers and views)
- Customize UI
- + Add a card

Current Sprint
- + Add a card

In Progress
- + Add a card

Done
- + Add a card

Screenshot of our Database Schema created in SQL Server



Enrollments

Column Name	Data Type	Allow Nulls
EnrollmentID	int	☐
StudentID	int	☐
ScheduledClassID	int	☐
EnrollmentDate	date	☐
		☐

Students

Column Name	Data Type	Allow Nulls
StudentID	int	☐
FirstName	nchar(50)	☐
LastName	nchar(50)	☐
Email	nvarchar(100)	☐
SSID	int	☐
		☐

StudentStatuses

Column Name	Data Type	Allow Nulls
SSID	int	☐
SSName	nvarchar(50)	☐
SSDecscription	nvarchar(MAX)	☑
		☐

ScheduledClasses

Column Name	Data Type	Allow Nulls
ScheduledClassID	int	☐
CourseID	int	☐
InstructorName	nvarchar(50)	☐
StartDate	date	☐
EndDate	date	☐
ClassDay	nvarchar(50)	☐
ClassTime	time(0)	☐
SCSID	int	☐
		☐

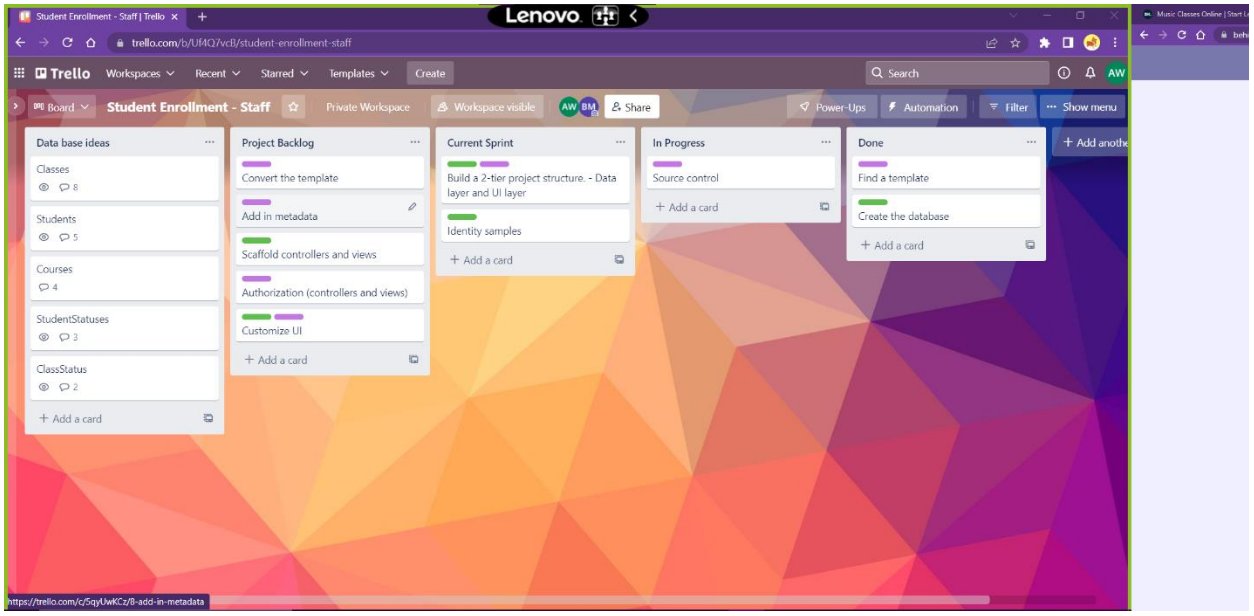
Courses

Column Name	Data Type	Allow Nulls
CourseID	int	☐
CourseName	varchar(50)	☐
Description	varchar(MAX)	☐
CreditHours	int	☐
		☐

ScheduledClassStatuses

Column Name	Data Type	Allow Nulls
SCSID	int	☐
SCSName	varchar(50)	☐
		☐

After having found a template, we began creating an MVC website backed up by a remote repo in Github and converting our template to an MVC style layout


Student Enrollment - Staff | Trello
Lenovo
trello.com/b/Uf4Q7vcB/student-enrollment-staff
Trello
Workspaces
Recent
Starred
Templates
Create
Search
AW
Board
Student Enrollment - Staff
Private Workspace
Workspace visible
AW
BM
Share
Power-Ups
Automation
Filter
Show menu
+ Add anothe
Music Classes Online | Start L
https://trello.com/c/5qyUwKCz/8-add-in-metadata

Data base ideas
- Classes 8
- Students 5
- Courses 4
- StudentStatuses 3
- ClassStatus 2
- + Add a card

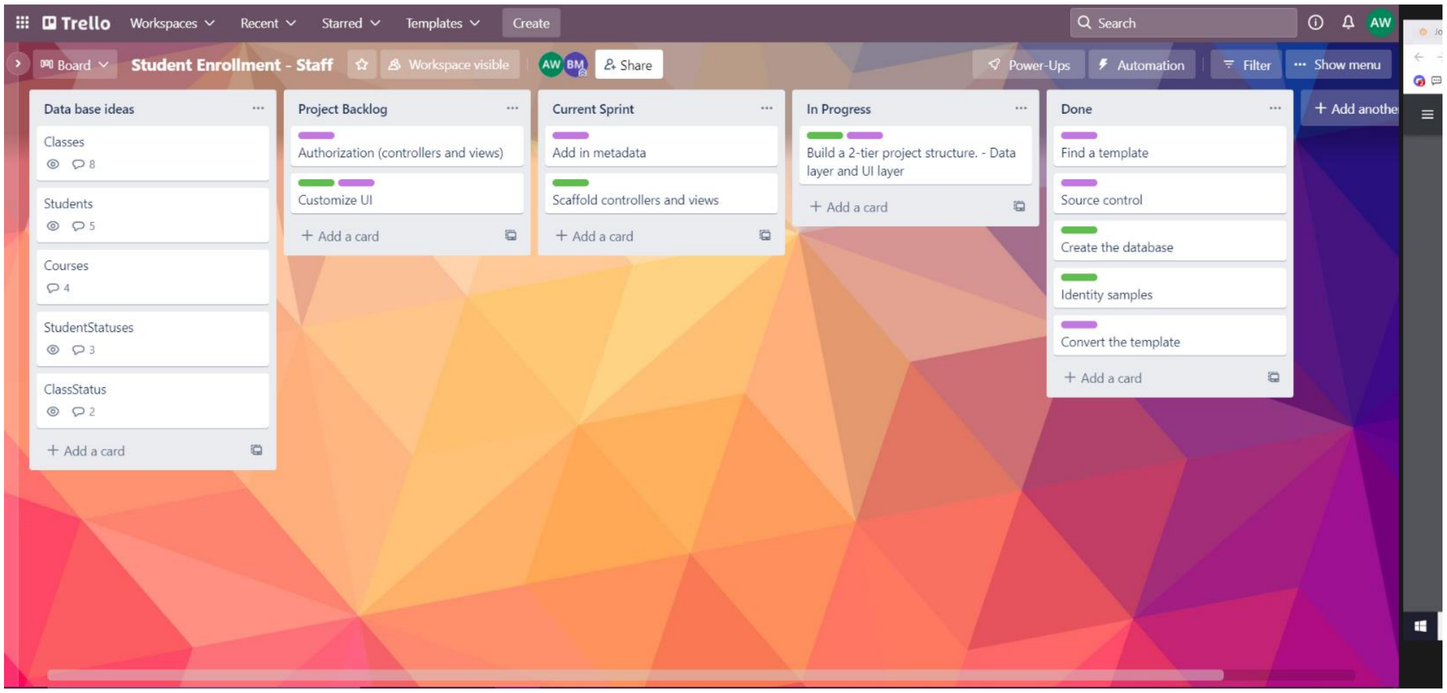
Project Backlog
- Convert the template
- Add in metadata
- Scaffold controllers and views
- Authorization (controllers and views)
- Customize UI
- + Add a card

Current Sprint
- Build a 2-tier project structure. - Data layer and UI layer
- Identity samples
- + Add a card

In Progress
- Source control
- + Add a card

Done
- Find a template
- Create the database
- + Add a card

We then added our Data layer from SQL Server and created Metadata to control how it would be displayed and express limitations from its datatypes



Data base ideas
- Classes 8
- Students 5
- Courses 4
- StudentStatuses 3
- ClassStatus 2
- + Add a card

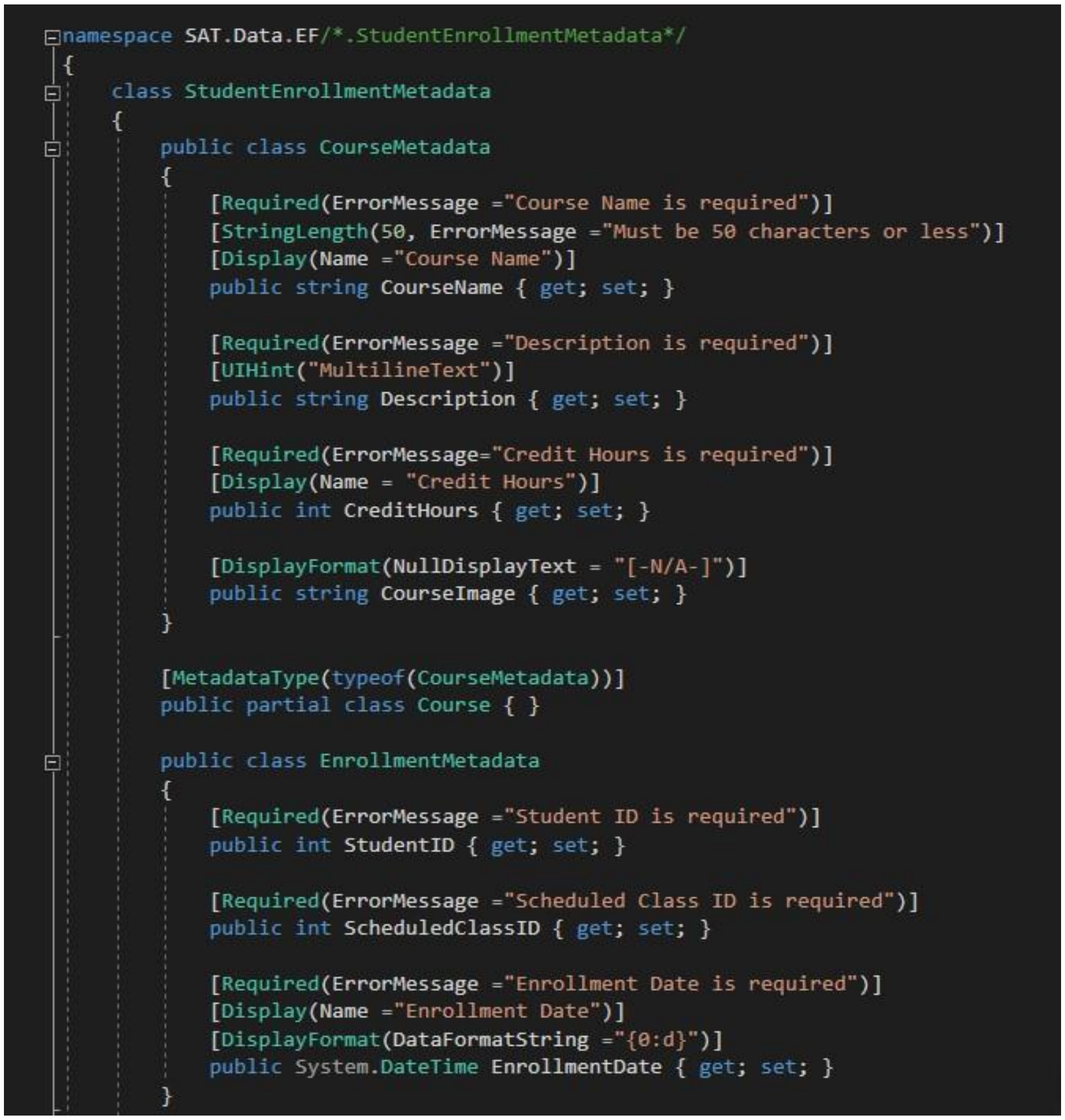
Project Backlog
- Authorization (controllers and views)
- Customize UI
- + Add a card

Current Sprint
- Add in metadata
- Scaffold controllers and views
- + Add a card

In Progress
- Build a 2-tier project structure. - Data layer and UI layer
- + Add a card

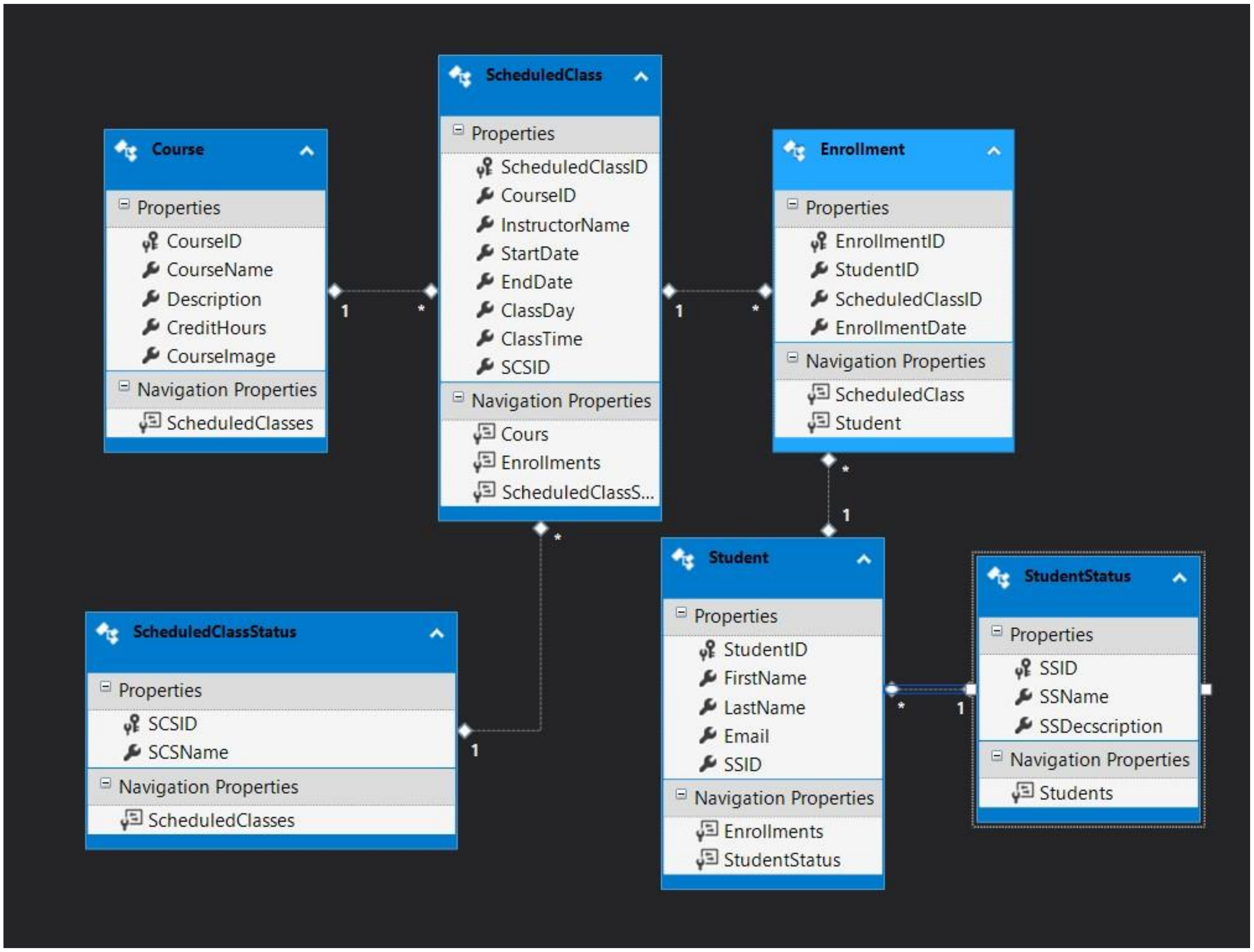
Done
- Find a template
- Source control
- Create the database
- Identity samples
- Convert the template
- + Add a card

Metadata Code

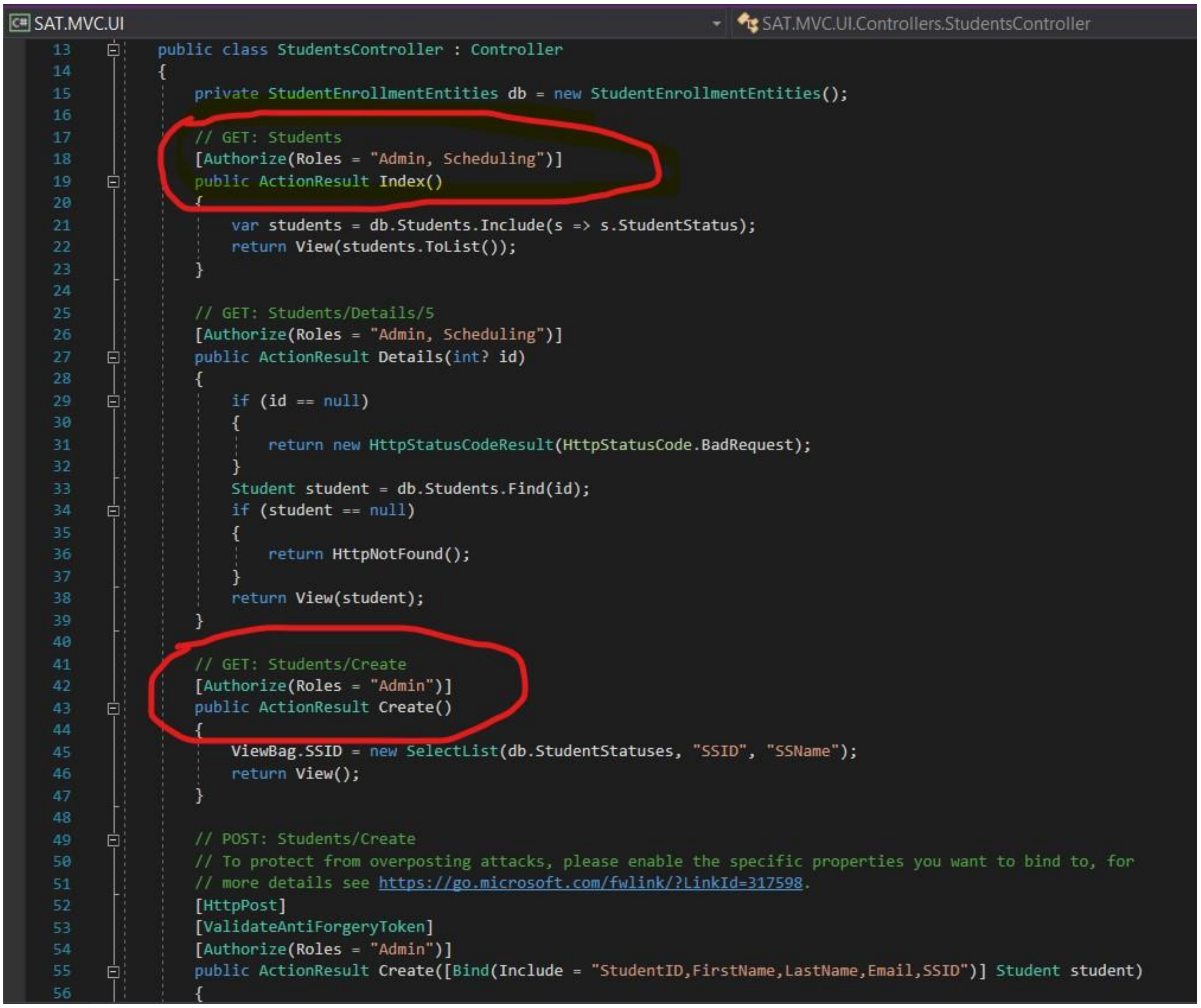


```
namespace SAT.Data.EF/*.StudentEnrollmentMetadata*/
{
    class StudentEnrollmentMetadata
    {
        public class CourseMetadata
        {
            [Required(ErrorMessage ="Course Name is required")]
            [StringLength(50, ErrorMessage ="Must be 50 characters or less")]
            [Display(Name ="Course Name")]
            public string CourseName { get; set; }

            [Required(ErrorMessage ="Description is required")]
            [UIHint("MultilineText")]
            public string Description { get; set; }

            [Required(ErrorMessage="Credit Hours is required")]
            [Display(Name = "Credit Hours")]
            public int CreditHours { get; set; }

            [DisplayFormat(NullDisplayText = "[-N/A-]")]
            public string CourseImage { get; set; }
        }

        [MetadataType(typeof(CourseMetadata))]
        public partial class Course { }

        public class EnrollmentMetadata
        {
            [Required(ErrorMessage ="Student ID is required")]
            public int StudentID { get; set; }

            [Required(ErrorMessage ="Scheduled Class ID is required")]
            public int ScheduledClassID { get; set; }

            [Required(ErrorMessage ="Enrollment Date is required")]
            [Display(Name ="Enrollment Date")]
            [DisplayFormat(DataFormatString ="{0:d}")]
            public System.DateTime EnrollmentDate { get; set; }
        }
```

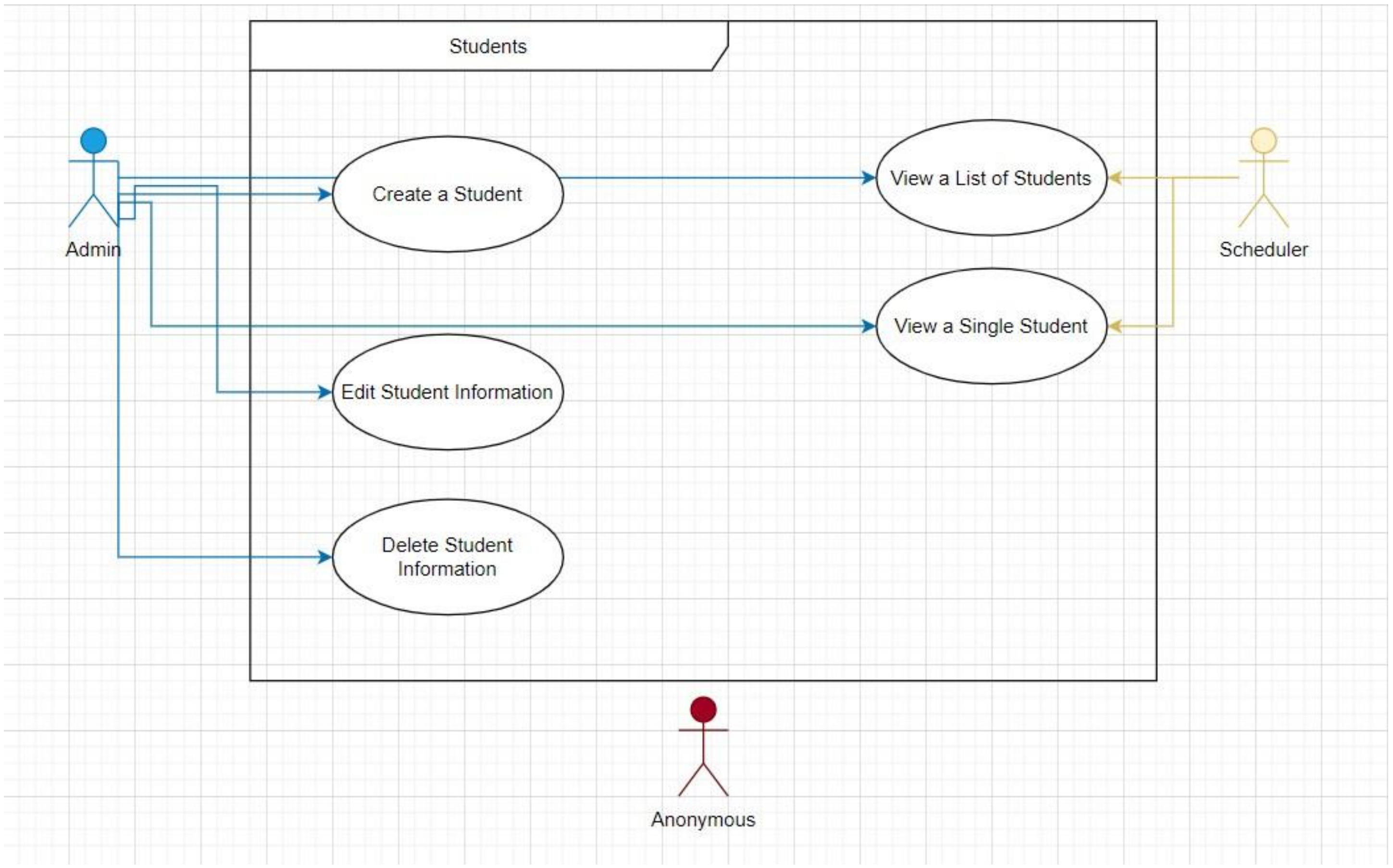
.edmx file

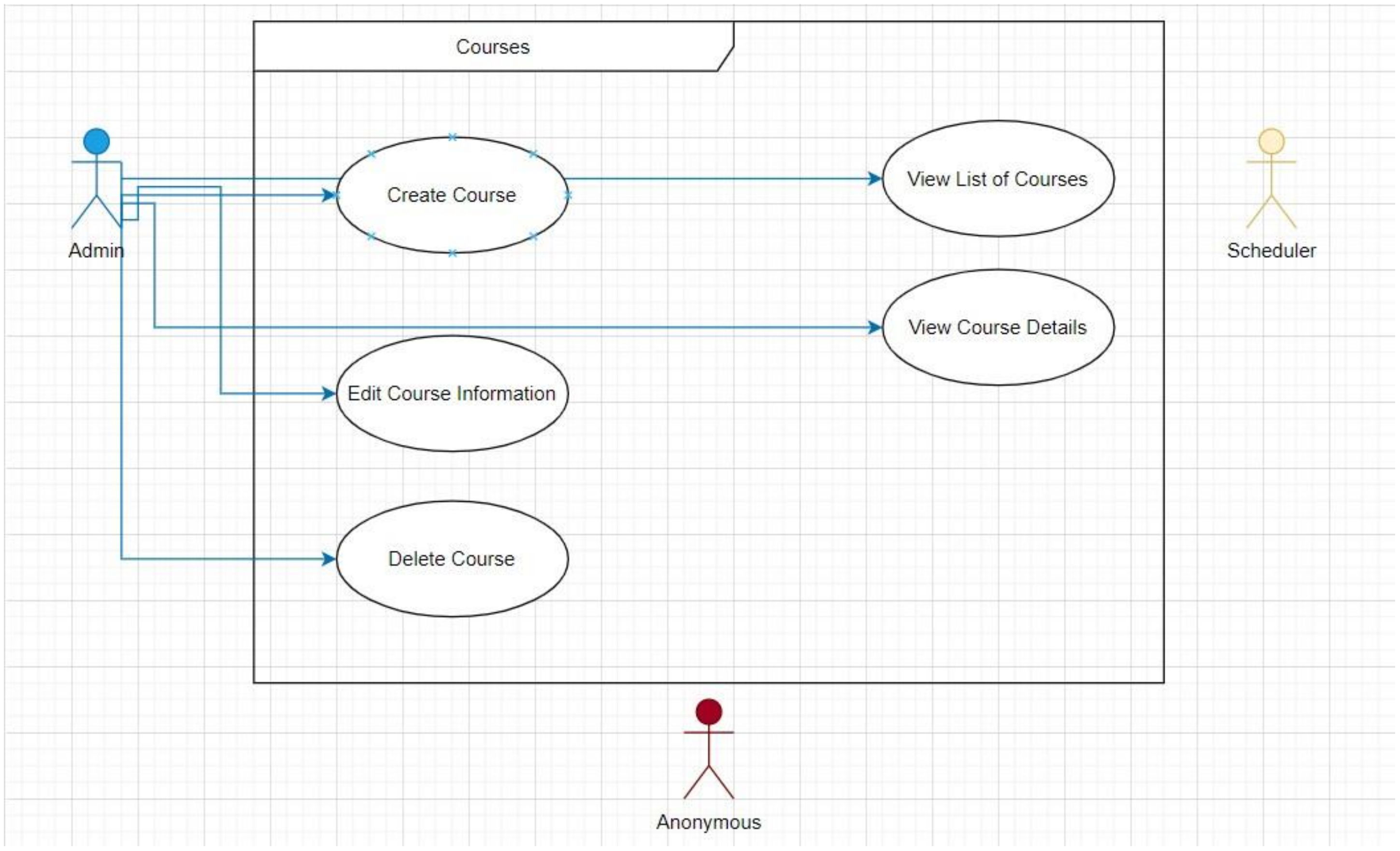

ScheduledClass
Properties
ScheduledClassID
CourseID
InstructorName
StartDate
EndDate
ClassDay
ClassTime
SCSID
Navigation Properties
Cours
Enrollments
ScheduledClassS...
Course
Properties
CourseID
CourseName
Description
CreditHours
CourseImage
Navigation Properties
ScheduledClasses
Enrollment
Properties
EnrollmentID
StudentID
ScheduledClassID
EnrollmentDate
Navigation Properties
ScheduledClass
Student
Student
Properties
StudentID
FirstName
LastName
Email
SSID
Navigation Properties
Enrollments
StudentStatus
StudentStatus
Properties
SSID
SSName
SSDecscription
Navigation Properties
Students
ScheduledClassStatus
Properties
SCSID
SCSName
Navigation Properties
ScheduledClasses
1
*
1
*
*
1
*
1
*
1

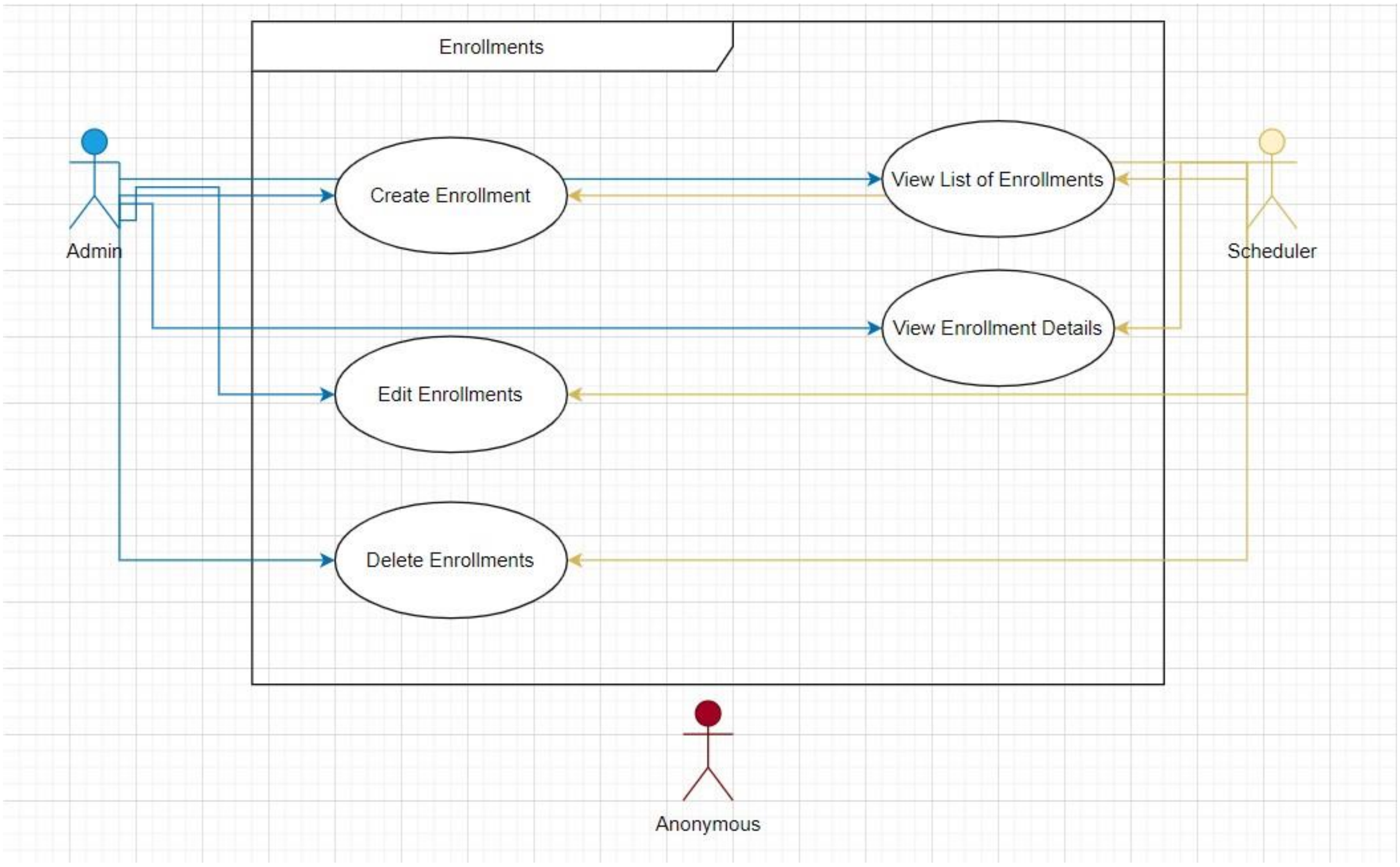
After Data was set up, we began assigning authorization to the specific users that were requested (Administrator and Scheduler). Below is code of how we performed that in the controllers followed by diagrams of the access given to these users.

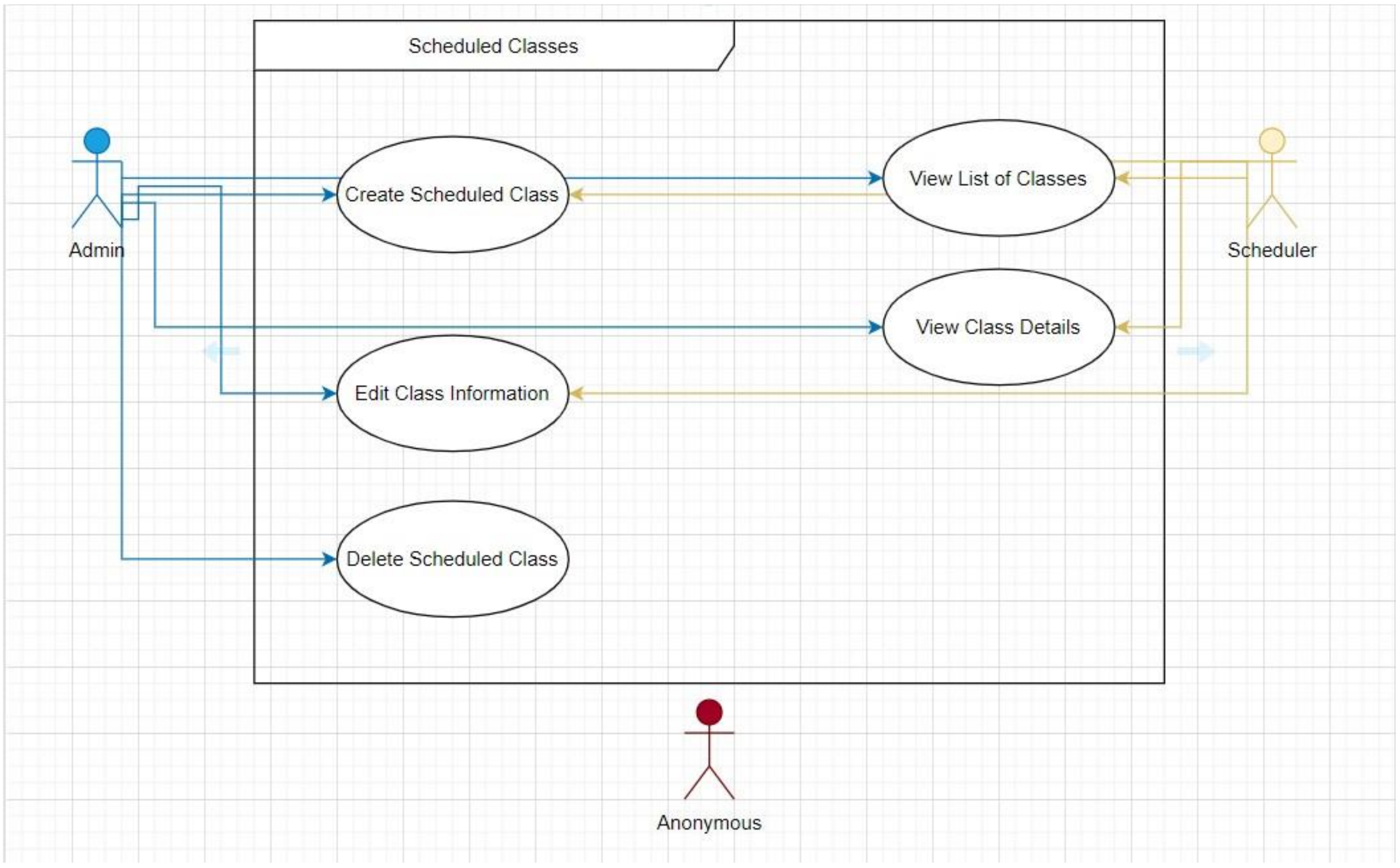


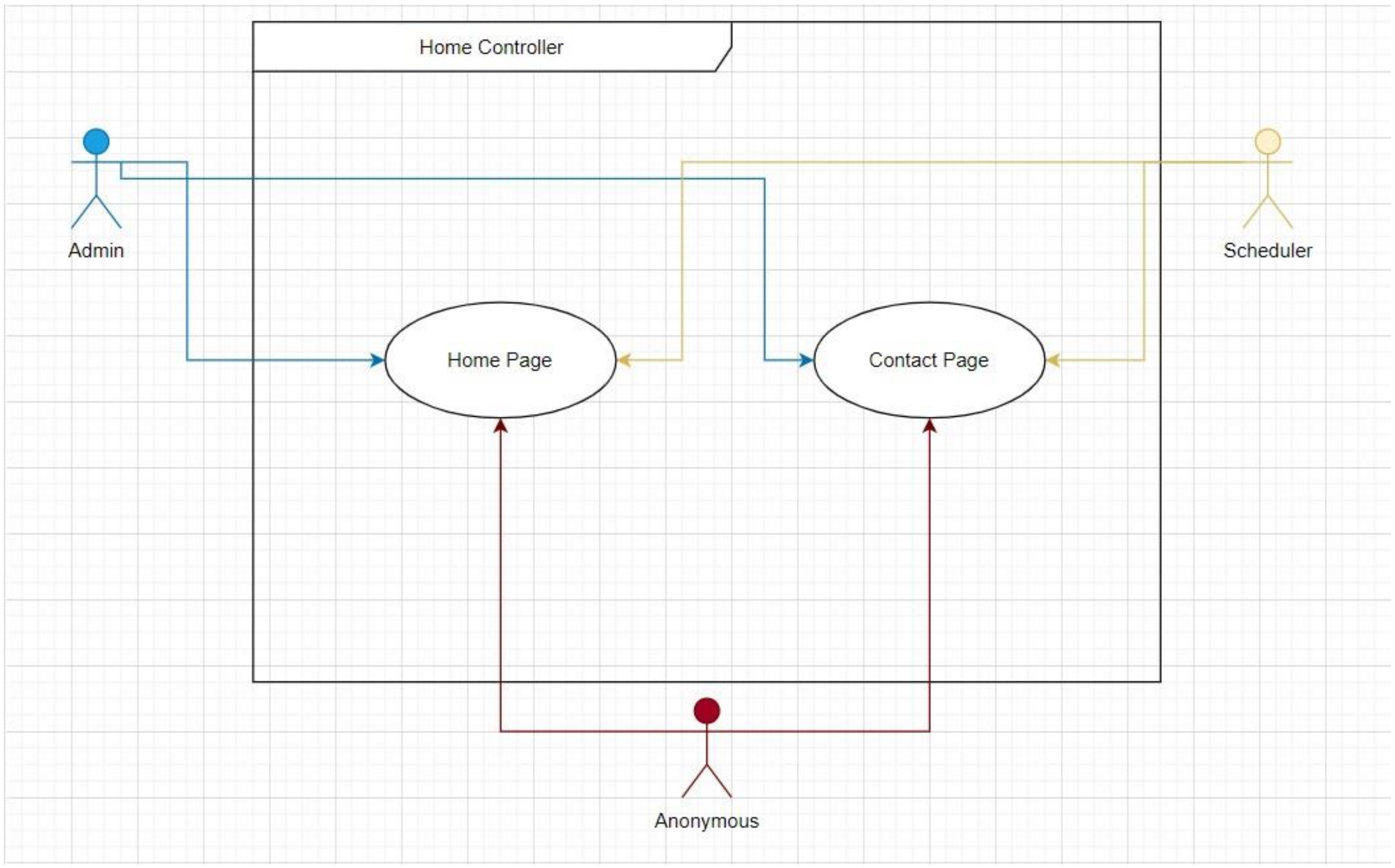
```
    public class StudentsController : Controller
    {
        private StudentEnrollmentEntities db = new StudentEnrollmentEntities();

        // GET: Students
        [Authorize(Roles = "Admin, Scheduling")]
        public ActionResult Index()
        {
            var students = db.Students.Include(s => s.StudentStatus);
            return View(students.ToList());
        }

        // GET: Students/Details/5
        [Authorize(Roles = "Admin, Scheduling")]
        public ActionResult Details(int? id)
        {
            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
            Student student = db.Students.Find(id);
            if (student == null)
            {
                return HttpNotFound();
            }
            return View(student);
        }

        // GET: Students/Create
        [Authorize(Roles = "Admin")]
        public ActionResult Create()
        {
            ViewBag.SSID = new SelectList(db.StudentStatuses, "SSID", "SSName");
            return View();
        }

        // POST: Students/Create
        // To protect from overposting attacks, please enable the specific properties you want to bind to, for
        // more details see https://go.microsoft.com/fwlink/?LinkId=317598.
        [HttpPost]
        [ValidateAntiForgeryToken]
        [Authorize(Roles = "Admin")]
        public ActionResult Create([Bind(Include = "StudentID,FirstName,LastName,Email,SSID")] Student student)
        {
```


Students
Admin
Create a Student
View a List of Students
Scheduler
View a Single Student
Edit Student Information
Delete Student Information
Anonymous

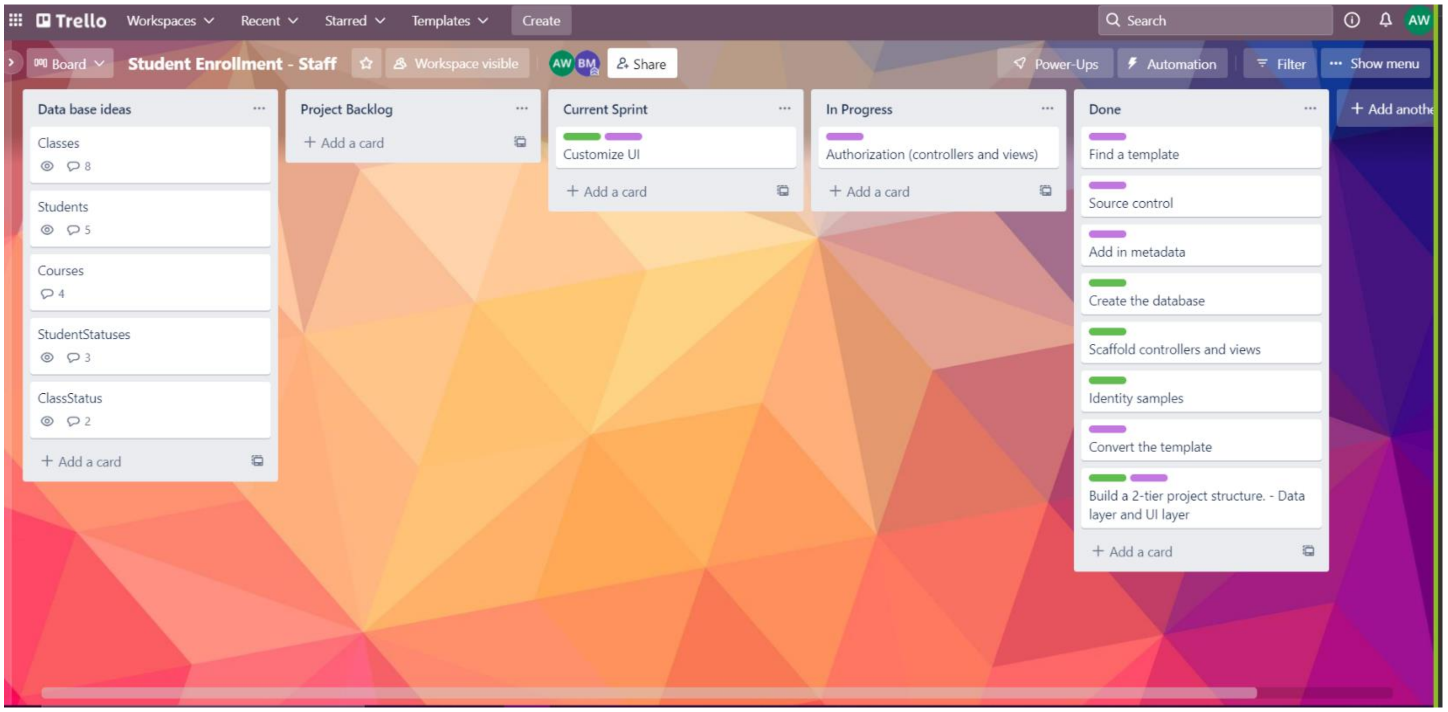

Courses
Admin
Create Course
View List of Courses
Scheduler
View Course Details
Edit Course Information
Delete Course
Anonymous


Enrollments
Admin
Create Enrollment
View List of Enrollments
View Enrollment Details
Edit Enrollments
Delete Enrollments
Scheduler
Anonymous


Scheduled Classes
Admin
Create Scheduled Class
View List of Classes
View Class Details
Edit Class Information
Delete Scheduled Class
Scheduler
Anonymous



We then customized the UI and views for each page and began testing CRUD functionality as well as authorization



Data base ideas
- Classes 8
- Students 5
- Courses 4
- StudentStatuses 3
- ClassStatus 2
- + Add a card

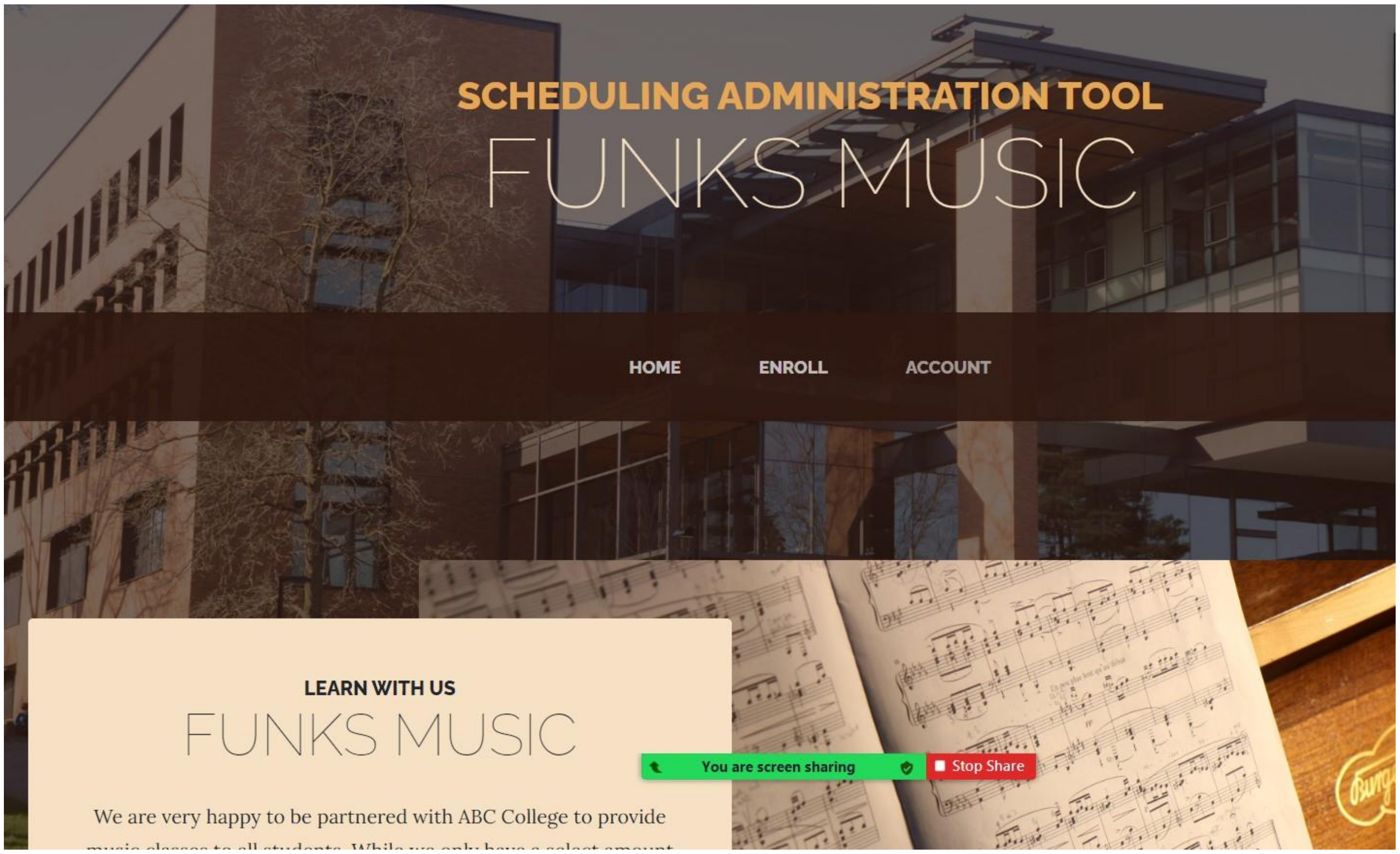
Project Backlog
- + Add a card

Current Sprint
- Customize UI
- + Add a card

In Progress
- Authorization (controllers and views)
- + Add a card

Done
- Find a template
- Source control
- Add in metadata
- Create the database
- Scaffold controllers and views
- Identity samples
- Convert the template
- Build a 2-tier project structure. - Data layer and UI layer
- + Add a card

We added a functioning contact page for enrollment and finished styling


SCHEDULING ADMINISTRATION TOOL

FUNKS MUSIC

HOME ENROLL ACCOUNT

LEARN WITH US

FUNKS MUSIC

We are very happy to be partnered with ABC College to provide