Documentation for Scheduling Administration Tool Application

Team Members: Mindi Ford, Michael Tullis and Justin Vignone

Welcome to the Scheduled Administration Tool ("S.A.T."). As a team, we were tasked with creating a site that a school may use for scheduling classes. This web application is used to manage Students, Student Statuses, Courses, Scheduled Classes, and Enrollments. Depending on your Identity role, you may be able to view, create, update, or remove these items.

This application uses:

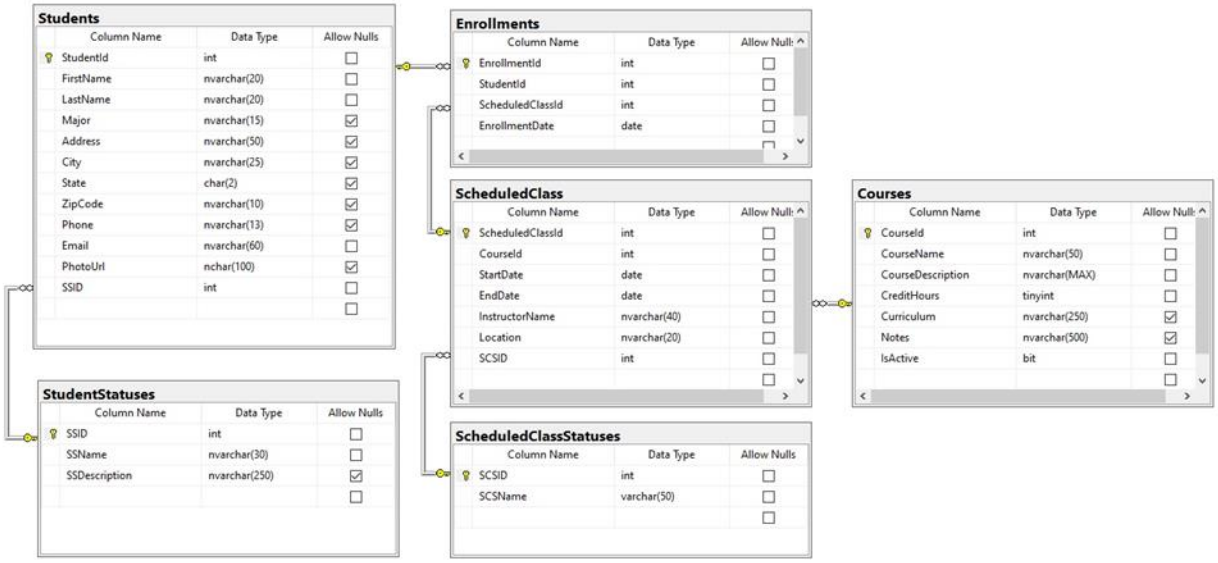
- HTML and CSS for structure and style
- Javascript for front-end logic
- MS (Microsoft) SQL Server data structure to manage and persist the data
- Identity Framework to manage Users/Roles
- Model-View-Controller (MVC) design pattern architecture to manage the building of the site

The following roles will be used in this application:

- Anonymous (unauthenticated) users will be able to view the following information:
    - Base Pages (home, contact)
- Users with the role of Scheduling will be able to view the following information:
    - Base Pages (home, contact)
    - Students (index, details)
    - Current Scheduled Classes (NO delete)
    - Enrollments (FULL CRUD)
- Users with the role of Administrator will be able to view the following information:
    - Base Pages (home, contact)
    - Courses (FULL CRUD)
    - Student Statuses (FULL CRUD)
    - Enrollments (FULL CRUD)
    - Scheduled Classes (FULL CRUD)
    - Students (FULL CRUD)

Below, we have provided documentation with photos as we build out this site.

Database Creation:



Logo Design:

Scaffolding Controllers with Views to display information in database

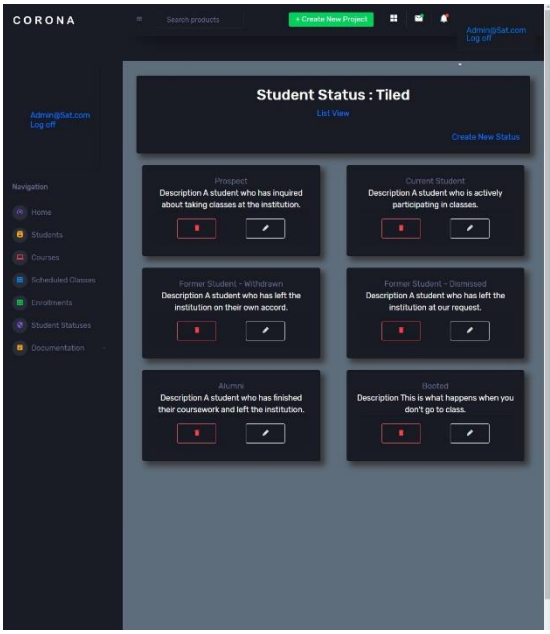
```
Index.cshtml
1   @model IEnumerable<SAT.DATA.EF.ScheduledClass>
2
3   @{
4       ViewBag.Title = "Scheduled Classes";
5   }
6
7   <h2>@ViewBag.Title</h2>
8
9   <p>
10      @Html.ActionLink("Create New", "Create")
11  </p>
12  <table class="table">
13      <tr>
14          <th>
15              @Html.DisplayNameFor(model => model.StartDate)
16          </th>
17          @*<th>
18              @Html.DisplayNameFor(model => model.EndDate)
19          </th>
20          <th>
21              @Html.DisplayNameFor(model => model.InstructorName)
22          </th>*@
23          <th>
24              @Html.DisplayNameFor(model => model.Location)
25          </th>
26          <th>
27              @Html.DisplayNameFor(model => model.Course.CourseNa
28          </th>
29          <th>
30              @Html.DisplayNameFor(model => model.ScheduledClassS
31          </th>
32          @if (User.IsInRole("Admin"))
33          {
34          <th></th>
35          }
36      </tr>
37
38  @foreach (var item in Model) {
39      <tr>
40          <td>
41              @Html.DisplayFor(modelItem => item.StartDate)
42          </td>
43          @*<td>
44              @Html.DisplayFor(modelItem => item.EndDate)
45          </td>
46          <td>
```

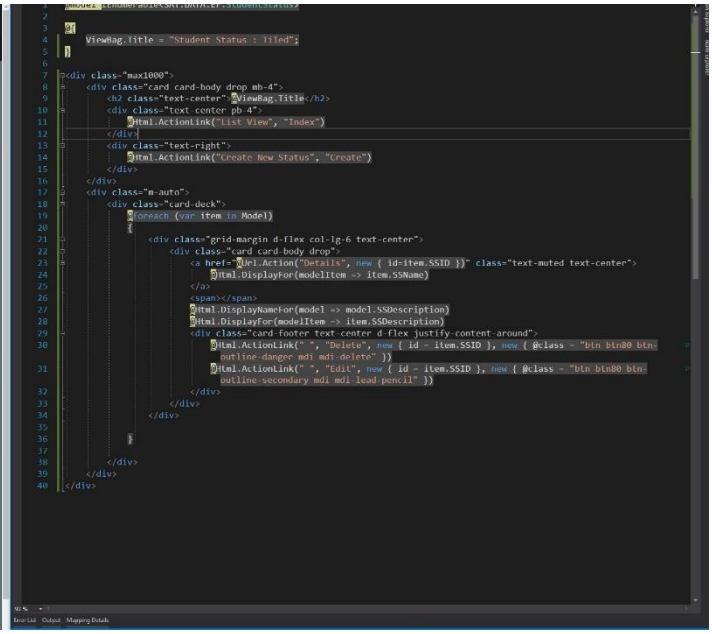
```
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'SAT' (2 projects)
  Solution Items
  SAT.DATA.EF
    Properties
    References
    bin
    Metadata
      SATMetadata.cs
    obj
    App.Config
    Class1.cs
    packages.config
    SAT.edmx
  SAT.UI.MVC
    Connected Services
    Properties
    References
    _Archive
    App_Data
    App_Start
    configs
    Content
    Controllers
      AccountController.cs
      CoursesController.cs
      EnrollmentsController.cs
      HomeController.cs
      ManageController.cs
      RolesAdminController.cs
      ScheduledClassesController.cs
      StudentsController.cs
      StudentStatusController.cs
      UserAdminController.cs
    fonts
    Models
    Scripts
    Views
      Account
      Courses
      Enrollments
      Home
      Manage
      RolesAdmin
      ScheduledClasses
        Create.cshtml
        Delete.cshtml
        Details.cshtml
        Edit.cshtml
        Index.cshtml
      Shared
      Students
        Create.cshtml
        Delete.cshtml
        Details.cshtml
        Edit.cshtml
        Index.cshtml
      StudentStatus
      UsersAdmin
      _ViewStart.cshtml
      Web.config
    favicon.ico
    Global.asax
    packages.config
    Startup.cs
```

Implementing soft delete functionality with a dropdown menu to change class status.

```
Delete.cshtml  Index.cshtml  ScheduledClassesController.cs  Edit.cshtml  ImageUtility.cs  Edit.cshtml  SATMetadata.cs  Delete.cshtml  StudentsController.cs  Student.cs

45          <td>
46              @Html.DisplayFor(modelItem => item.Course.CourseName)
47          </td>
48
49          <td>
50              <div class="dropdown d-inline">
51                  <div href="#" class="btn btn-outline-primary btn94 d-inline-block dropdown-toggle" data-
                      toggle="dropdown">
52                      @* dropdown-toggle adds the down arrow *@
53                      @Html.DisplayFor(modelItem => item.ScheduledClassStatus.SCSName)
54                  </div>
55                  <div class="dropdown-menu" aria-haspopup="true" aria-expanded="false">
56                      @Html.ActionLink("Active", "Delete", new { id = item.ScheduledClassId, @status = 1 }, new { @class
                          = "dropdown-item text-white" })
57                      @Html.ActionLink("Pending", "Delete", new { id = item.ScheduledClassId, @status = 2 }, new
                          { @class = "dropdown-item text-white" })
58                      @Html.ActionLink("Completed", "Delete", new { id = item.ScheduledClassId, @status = 3 }, new
                          { @class = "dropdown-item text-white" })
59                      @Html.ActionLink("Cancelled", "Delete", new { id = item.ScheduledClassId, @status = 4 }, new
                          { @class = "dropdown-item text-white" })
60                  </div>
61              </div>
62          </td>
63
64          <td class="text-right">
65              @if (User.IsInRole("Admin"))
66              {
100 %
Error List  Output
```

Creating alternate tiled view to customize user experience.

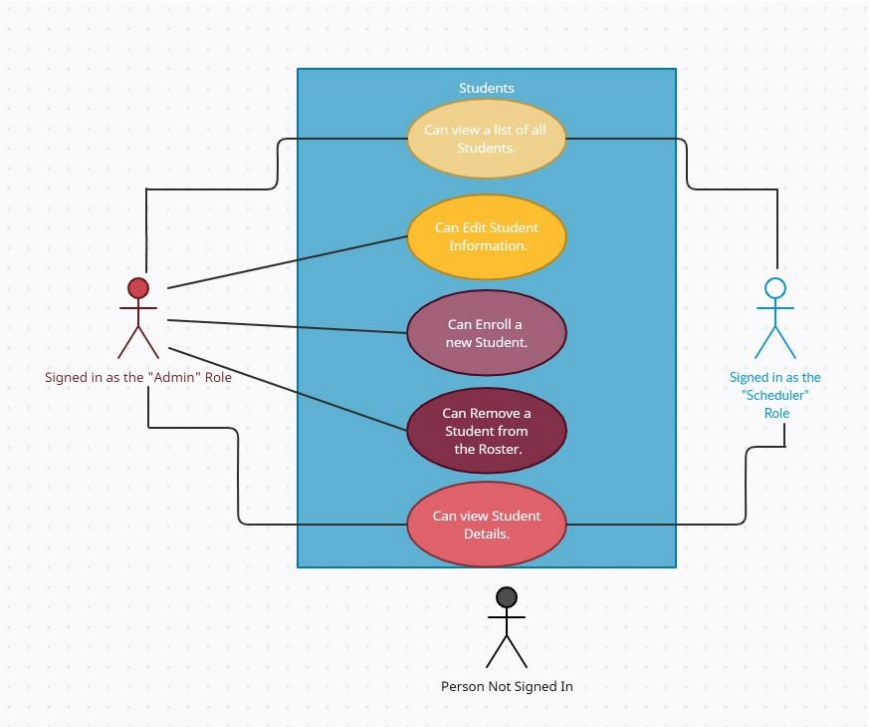


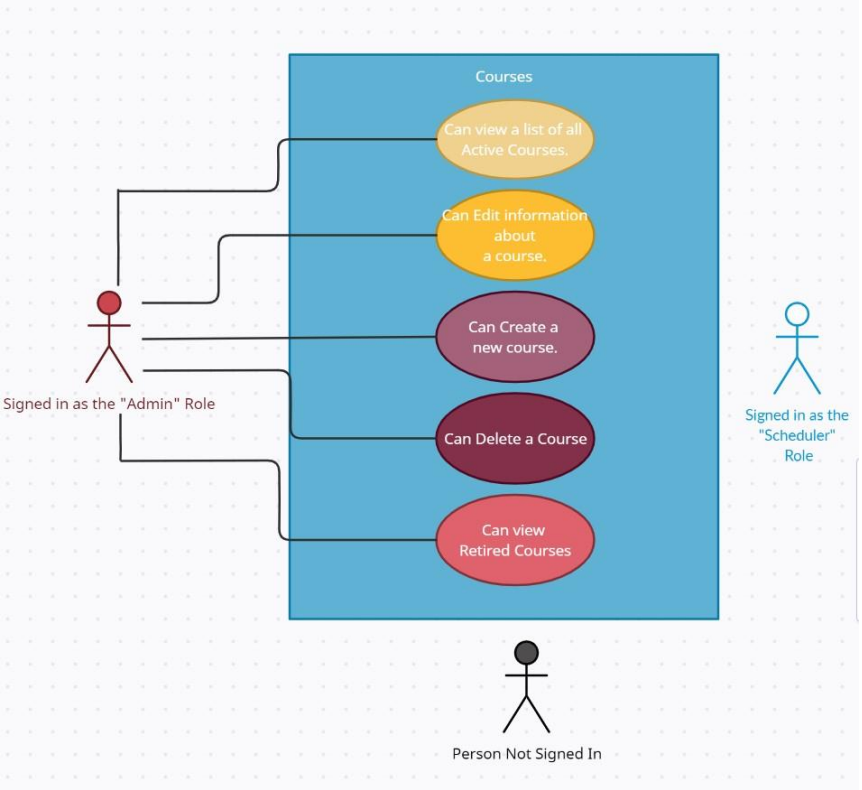


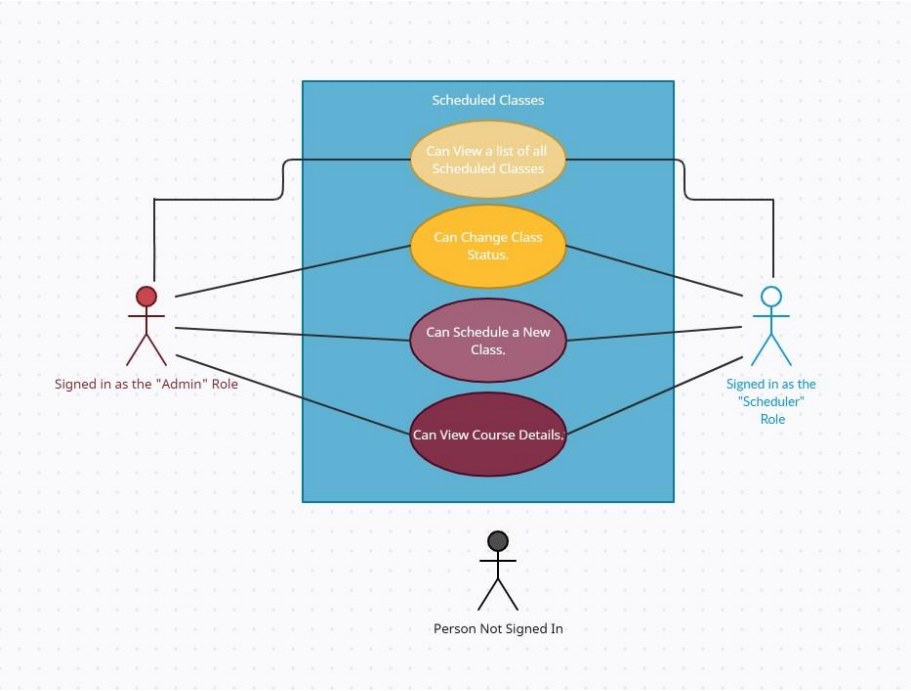
Implementing file upload utility.

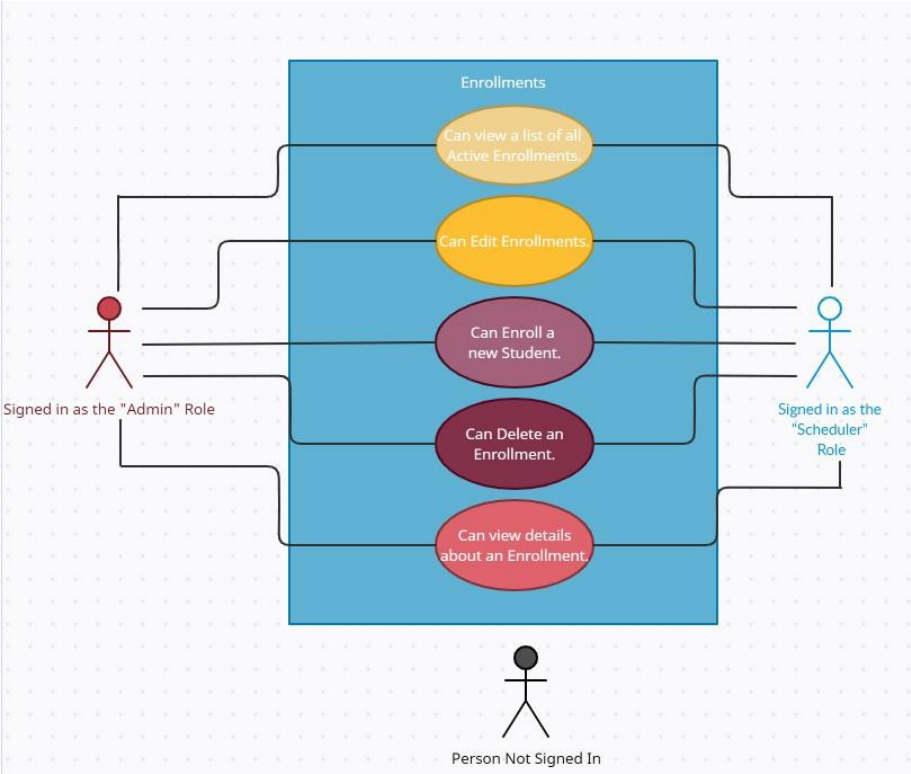
```
imageUtility.cs   StudentsController.cs*   Index.cshtml   CoursesController.cs   Index.cshtml
SAT.UI.MVC   SAT.UI.MVC.Controllers.StudentsController   Edit(Student student, HttpPostedFileBase studentPhoto)

        // 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.
        [Authorize(Roles = "Admin")]
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Edit([Bind(Include = "StudentId,FirstName,LastName,Major,Address,City,State,ZipCode,Phone,Email,PhotoUrl,SSID")] Student
          student, HttpPostedFileBase studentPhoto)
        {
            if (ModelState.IsValid)
            {
                #region File Upload

                string imageName = "noImage.jpg";

                if (studentPhoto != null)
                {
                    imageName = studentPhoto.FileName;

                    string ext = imageName.Substring(imageName.LastIndexOf("."));

                    string[] goodExts = new string[] { ".jpeg", ".jpg", ".png", ".gif" };

                    if (goodExts.Contains(ext.ToLower()) && studentPhoto.ContentLength <= 4194304)
                    {
                        imageName = Guid.NewGuid() + ext;

                        #region Resize Image Functionality
                        string savePath = Server.MapPath("~/Content/assets/images/students/");

                        Image convertedImage = Image.FromStream(studentPhoto.InputStream);

                        int maxImageSize = 300;

                        int maxThumbSize = 65;

                        ImageUtility.ResizeImage(savePath, imageName, convertedImage, maxImageSize, maxThumbSize);

                        #endregion

                    }

                }

                student.PhotoUrl = imageName;

                #endregion
```

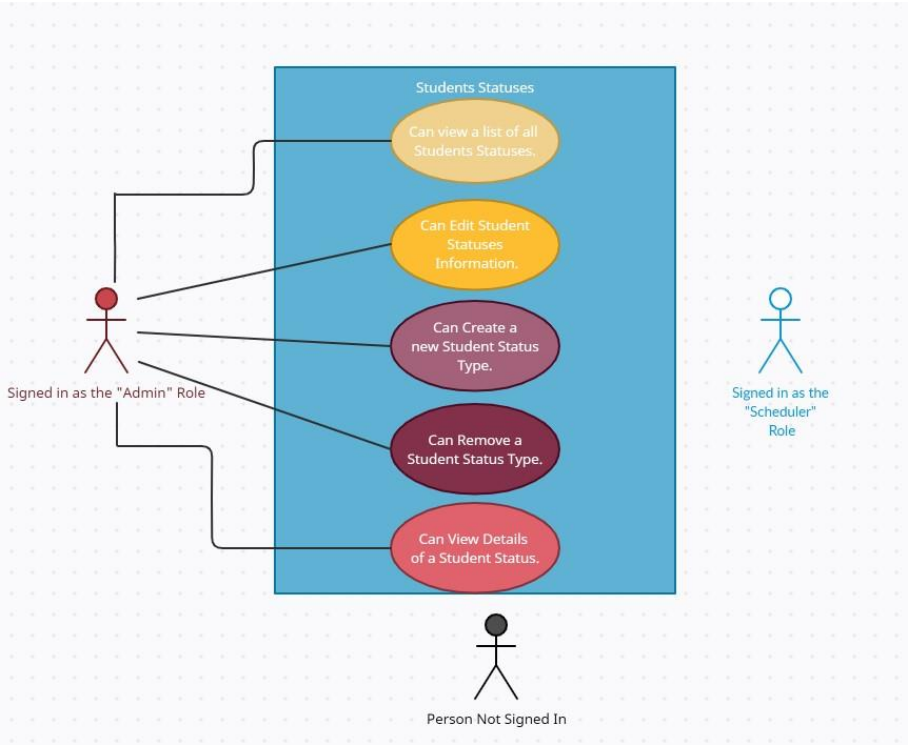
Below, we have our UML Use Case Diagrams. These detail the different roles and the functionality each has access to.






Scheduled Classes
Can View a list of all Scheduled Classes
Can Change Class Status.
Can Schedule a New Class.
Can View Course Details.
Signed in as the "Admin" Role
Signed in as the "Scheduler" Role
Person Not Signed In


Enrollments
Can view a list of all Active Enrollments.
Can Edit Enrollments.
Can Enroll a new Student.
Can Delete an Enrollment.
Can view details about an Enrollment.
Signed in as the "Admin" Role
Signed in as the "Scheduler" Role
Person Not Signed In


Students Statuses
Can view a list of all Students Statuses.
Can Edit Student Statuses Information.
Can Create a new Student Status Type.
Can Remove a Student Status Type.
Can View Details of a Student Status.
Signed in as the "Admin" Role
Signed in as the "Scheduler" Role
Person Not Signed In