Problems we came across and what we did to resolve them:

**Choosing the Project Framework**

Our first major decision was which project framework to choose: SPA, WebForms, MVC or

With a little research we quickly eliminated

**Doing the first walkthrough of deploying to Azure**

We did this very early in the semester hoping to gain experience and generate a little confidence. We spent most of a day doing the walkthrough. There was a slight euphoria when it turned out to be pretty straightforward. While a great group bonding experience, it was a little demotivating in that it made it seem like everything in the project was going to be that easy…and it wasn’t.

**Integrating JQuery plugins into MVC project**

Each of the JQuery plugins specified their own set of .js, .css (Albert, others?) files. Some files were .min files. We had to understand where the files lived, what code executed them, when and how and them integrate them into the project. In addition, we had to learn about minification and choose a tool, (What did you use?).

**The GIT Bash Learning curve**

**Living in GITHub and working in GIT**

**Database**

**Integrating the MVC “free” User Authentication**

**Integrating javascript code into MVC/Razor/HtmL Model**

Circliful, Tree and schedule, all JQuery plugins are provided with static demonstration objects.

Integrating them into an MVC project required providing the model data to the plugin so that it could be rendered in the view.

For Tree this is the provided code:

**var** items = [

**new** primitives.orgdiagram.ItemConfig({

                    id: 0,

                    parent: **null**,

                    title: "Beth's Van Belle",

                    description: "Project Tree",

                    image: "demo/images/photos/a.png"

                }),

**new** primitives.orgdiagram.ItemConfig({

                    id: 1,

                    parent: 0,

                    title: "CSC 311",

                    description: "Ugh! C#",

                    image: "demo/images/photos/b.png"

                }),

**new** primitives.orgdiagram.ItemConfig({

                    id: 2,

                    parent: 0,

                    title: "Senior Project",

                    description: "Fabulous",

                    image: "demo/images/photos/c.png"

                }),

**new** primitives.orgdiagram.ItemConfig({

                    id: 3,

                    parent: 0,

                    title: "Mom",

                    description: "Visit and Love",

                    image: "demo/images/photos/d.png"

                }),

**new** primitives.orgdiagram.ItemConfig({

                    id: 4,

                    parent: 3,

                    title: "Bake Cake",

                    description: "More Fabulous",

                    image: "demo/images/photos/d.png"

                })

            ];

To this:

var items;

var newItem ;

var items = [ ];

@foreach(var item in Model)

{

@:newItem = new primitives.orgdiagram.ItemConfig({ id: '@(item.ID)', parent: '@(item.ParentID)', title: '@(item.Title)', description: '@(item.Note)' });

@:if('@item.ParentID' == '0') //needed to make javascript implementation compatible with Project Model

@: { //specifically, project model has int for parentid and sets root to zero

@: var headparent = 0 ;

@: newItem.parent = null ;

@: }

@:items.push(newItem);

}

You have a mixture of C#, razor and javascript, figuring out what needed to be done and the syntax of it was quite time consuming. Circliful and schedule had similar issues. After completing the first, experience made subsequent integrations a little easier. Despite how challenging it was, it saved timed and allowed greater functionality in the project.