My employer struggles with documentation of systems and servers. My project will start to fix that problem. It will allow them to track the servers, what systems belong to, and what their role (server type) is. Users will be able to populate a list of servers, a list of systems, and maintain a list of server types (app, web, database, etc…). They will be able to maintain the relationships between them. The solution should enforce referential integrity (can’t delete a server unless it is no longer associated with a system). It should be easy to maintain the names and short descriptions and it should be easy to see the relationships and navigate between them. This solution is meant as a phase 1 that allows the users to gather the information. Future phases will be expanded to allow the users to store email distribution groups for systems and allow for triggered notifications of planned and accidental outages. This is not mean to be a server monitor. It is mean to compliment a system monitor, issue tracking system and planned projects by allowing the organization to document the relationships between the servers and systems and get a full view of the impact of outages so that ALL appropriate users can be notified.

Links I used:

<https://stackoverflow.com/questions/38334376/entity-framework-composite-key-not-allowing-duplicate-data>

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>

<https://stackoverflow.com/questions/37166098/how-to-make-ef-core-use-a-guid-instead-of-string-for-its-id-primary-key>

<https://forums.asp.net/t/2117352.aspx?Populate+DropDownListFor+using+Entity+Framework>+

I ran into a few minor problems when creating this project. First of all, when I created the MVC views and controllers, I specified to use Async, which causes performance issues and complexities. I deleted them and recreated without using the Async option. Next, I had an issue with one of my table names. Don’t use “System” as a table name. EF gets it confused with the System namespace. I had to tweak the code to handle it. The next issue I ran into was I had created foreign keys in the Link table using String instead of Uniqueidentifier. EF sees these as two different things that don’t work together. I recreated the Link table without its own primary key (used a composite key instead) and changed the data types to Uniqueidentifier. I ran into a few issues that I had to fix regarding EF and composite keys. It doesn’t seem to like those. It may have been easier to store a primary key, but it was unnecessary. EF is also a little finicky about the link table, but I was able to find examples and make it work.

This project will make my life easier. As a Systems Analyst, I support the application systems that are used throughout the global operations of a 1000+ employee organization. There are many servers (specifically database and web servers) that support a number of different systems from COTS applications to custom in-house apps. There is no way to easily document and report on what systems a server will impact or vice versa. During system upgrades, system migrations or server maintenance, it is frequently my job to coordinate down-times with users in order to set expectations regarding system availability. I will use this and share with my coworkers so that we can maintain it together and have complete and accurate information.

2/3/2021:

* Create code repo in github and shared with instructor.
* Submitted Project Proposal
* Created Azure database.
* Created Journal.
* Created application solution in Visual Studio
* Synced solution with github
* Installed EntityFramework and MVC
* Created database schema in azure
* Used code-first for EF to create data models
* Fixed issue with schema (note to self: When using EF, don’t create a table with the name System. VS gets confused between the System class and the System namespace.)
* Modified Index and Detail pages for Systems, System Type, Servers (display fields on screen for Index and links to edit from both)
* Fixed issue with primary key fields in EF and models they generated
* Tested Create for systems, system type, servers
* Tested edit for systems, system type, servers

2/4/2021:

* Set Name and Description fields to nvarchar so it doesn’t pad. Fixed EF model.
* Played around with Link table.

2/6/2021:

* Deleted ID for Link to make it a composite key. Changed keys to uniqueidentifier instead of string.
* Deleted and rebuilt Models, Controllers and Views (not async)
* Reimplemented fixes from 2/3
* Added properties to the ViewBag for drop-downs for Link Create and Edit pages
* Added validation for Link create/edit pages to prevent duplicate composite keys and require valid selections (views and controller)
* Fixed editing Links (EF doesn’t like updating keys)
* Added Details pages with listings of the related items
* Added links to Details pages
* Adding sorts to index pages

2/7/2021:

* Added comments
* Cleaned up code
* Tweaked Detail pages to make it look a little better
* Tested thoroughly