

EAD 2022 – CA2 Mini Project – May 12th

Project Title: Chemical Management MVC

Team:

Johnathan Sheehan - X00083305

Martyna Pacula - X00118313

Project Links to Website and Source Code

Azure Website	https://chemical-management.azurewebsites.net/
MVC Project Repo (Source)	https://github.com/MartynaP14/Chemical-Management
Client Code Repo	https://github.com/JohnnySheehan/Chemical-Management-Client

Project Goals and Functionality

- To create an MVC ASP.NET Core Application to allow CRUD operations across Lab, Reagent and Reagent Supply models, to form a chemical management system.
- The chemical management system can only be used by authorized and authenticated users.
- The App should be deployed on an Azure App Service and thus should be accessible to the users via a client.
- The App will host and maintain its data in a cloud database (Azure SQL).

Table of Contents

1. Project Plan: Pages 1 – 2
2. Database Design: Pages 2 - 6
3. URI Addressing Scheme: Page 7 - 8
4. Unit Testing: Page 8
5. MVC App and Console App Screenshots: Page 9 - 12

1. Project Plan:

For CA2, we agreed to adopt an iterative software development lifecycle. During Scrum meetings, we would do planning around our sprint tasks. The goal of this was to thoroughly understand how to represent our requirements in code before starting development.

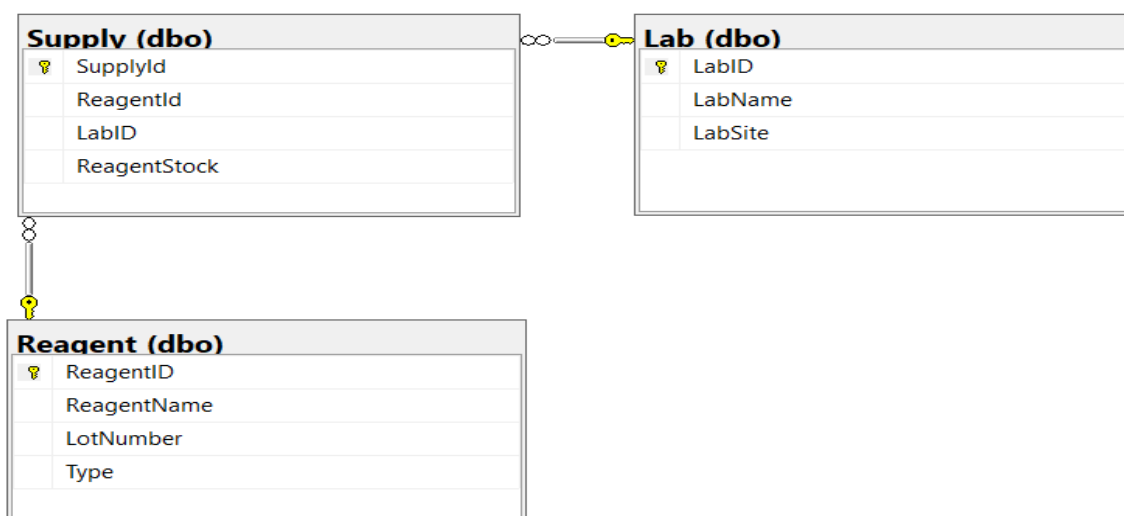
After coding, we would user test the code representing our requirements and take the learnings from our testing to refactor our code. This helped us adjust and in one case refactor our project scope to achieve a successful running ASP.NET Core MVC app for a chemical management system.

<i>Development Schedule</i>		
Scrums	Sprint Tasks	Outcome
Team Creation: 04/04/22		Team Assigned by Dermot
Scrum 1: 08/04/22	<ul style="list-style-type: none">- Project Proposal- Set Up GitHub Repo- Create Models- Create Controllers	Granted by Dermot: Scope meets requirements Completed Completed Completed
Scrum 2: 15/04/22	<ul style="list-style-type: none">- User test Models- User Test Controllers- Set Up Azure Resource Group- Add SQL Server to Resource Group with shared IPs	Completed Completed Completed Completed
Scrum 3: 22/04/22	<ul style="list-style-type: none">- Request permission to refactor project scope- Refactor project (Models, Controllers, Views)	Granted by Dermot Refactor Completed (Milestone: ASP.NET Core App completed locally)
Scrum 4: 26/04/22	<ul style="list-style-type: none">- Add Cookie based Login Service- User test new MVC	Completed Completed
Scrum 5: 30/04/22	<ul style="list-style-type: none">- Drop 'Type' MVC + DB set- Final MVC Development Iteration- Move from Local DBs to Production Database on Azure SQL Server.	Completed Completed Completed (Milestone: Azure SQL Database with three tables)
Scrum 6: 02/05/22	<ul style="list-style-type: none">- Deploy App Service & Publish Project	Completed (Milestone: Website hosted in Azure cloud)

Scrum 7: 07/05/22	<ul style="list-style-type: none"> - Bootstrap CSS theme and Navbar - Unit Testing 	<p>Completed</p> <p>Completed (Milestone: Unit Testing Completed)</p>
Scrum 8: 11/05/22	<ul style="list-style-type: none"> - Project Documentation - RESTFUL Web Service and Http Client Console App - Seed Data via SSMS - Add Icons, Pictures, Carousel. 	<p>Ongoing</p> <p>Completed (Milestone: RESTFUL Web Service and Client)</p> <p>Completed</p> <p>Completed (Milestone: Nice intuitive front-end for website)</p>
	<ul style="list-style-type: none"> - DEMO Practice - Final Front-End Clean Up - Final Publish to App Service - Finalise Documentation 	<p>Completed</p> <p>Completed</p> <p>Completed (Milestone: Finalised ASP.NET Core App Completed)</p> <p>Completed (Milestone: Doc Completed)</p>

2. Database Design

Below is a diagram of our database captured from SSMS.



Our Database Logic

The Reagent table is needed to hold information (properties) about this object.

The Supply Table is needed, as it is a child table of Reagent and Lab but has the unique property of 'Reagent Stock.' This breaks a many-to-many two-table relationship (Reagent-Lab) into a three-table relationship consisting of a one-to-many relationship between Reagent (1) to

Supply (M) and a many-to-one relationship between Supply (M) and Lab (1). Therefore, each reagent can have many supplies and many Supplies are stored in a Lab.

Code First Design (Entity Framework Core)

To help our go-to object relationship mapper (EF Core) understand our models and their relationships, we need to add constraint properties to certain models. These properties represent Foreign Keys.

These Foreign Key act as a reference to a parent table/model within a child table/model. In our project, the Supply model needed to contain a reference to both the Reagent and the Lab objects to create the 'one' side of the entity relationship. This was completed by adding a virtual class reference and Foreign Key Identifiers for both Reagent and Lab.

Likewise, our Reagent and Lab classes needed an 'ICollection' reference for Supply to model the 'many' sides of the relationship.

Database Constraints

Our models incorporated Primary Keys as unique identifiers for each model (INTs that increment in 1) and Foreign Keys to link related tables.

Using Data Annotations from the component model data annotation namespace allowed us to enforce required entry for model properties. This allowed us to make model properties non-nullable meaning that user input is required for the creation of the three object types in our models. This solves the challenge of our model relationship design for Supply, as it must have a reagent and lab to be constructed (Foreign Key enforcement).

Model Code

Reagent Model:

```
public class Reagent
{
    [Key]
    public int ReagentID { get; set; }
    [Required(ErrorMessage = "Please enter Reagent Name")]
    public string ReagentName { get; set; }

    [Required(ErrorMessage = "Please enter Reagent the lot number")]
    public int LotNumber { get; set; }

    //represents many side of reagent to supply
    public virtual ICollection<Supply> Supplies { get; set; }

    public ReagentType Type { get; set; }
}
```

Supply Model:

```
public class Supply //represents relationship between reagent and lab with added property of stock/supply
{
    [Key]
    public int SupplyId { get; set; }

    [Required]
    public int ReagentId { get; set; } //FK to tie to Reagent ref in supply class / also allows us to see which labs have certain reagents
    public int LabID { get; set; } //FK to tie Lab reference in supply class (needed to see what labs have certain stocks)
    public int ReagentStock { get; set; }

    //Navigation
    public virtual Reagent Reagent { get; set; }
    public virtual Lab Lab { get; set; }
}
```

Lab Model:

```
public class Lab
{
    [Key]
    public int LabID { get; set; }

    [Required]
    [Display(Name = "Lab name")]
    public string LabName { get; set; }

    [Required]
    [Display(Name = "Site the lab belongs to")]
    public string LabSite { get; set; }

    //represents many side of the lab to supply relationship
    public virtual ICollection<Supply> Supplies { get; set; }
}
```

Adding Tables and Data Seeding our Database

We decided to create our models using a code-first approach. The database tables for each model were created by using a database context and model database sets using EF Core.

This code first method gave us the initial benefit of database version control (locally) while testing our models in the early stages of the project design. In hindsight, this was key to the outcome of our final database design, as we refactored our model design to a more refined yet interconnected one.

Although, we created our database with a code first approach; for our database seeding, we decided to add data to our models using SQL statements in SQL Server Management Studio. Our logic behind this decision was to prove our database context class within our application was working correctly (executing the SQL statements in SSMS to see the data appear on web application views).

Another reason we chose to add data manually via SQL, is that after testing on local databases, we figured by the very nature of the project (two programmers working on separate project instances, at the same time, pulled from GitHub, and both are editing and seeding data to one cloud database) that it could cause concurrency issues. For example, programmer one executes an edit in the web app and programmer two re-seeds the data and updates before programmer one's changes are written to the database. *However, concurrency could still happen as ANSI -> Set Implicit_Transactions is set to off automatically in Azure SQL Server and no locks were added (therefore, more than one transaction per connection on the same data item could happen).*

Reagents Table Insert:

```
set IDENTITY_INSERT Reagent ON
insert into Reagent(ReagentID, ReagentName, LotNumber, Type)
values
(1, 'Nitrous Oxide', 0973, 5),
(2, 'Hydrochloric Acid', 2825, 2),
(3, 'Acetone', 3159, 0),
(4, 'Iodine', 0105, 1),
(5, 'Methanol', 5261, 0),
(6, 'Nitric Acid', 1010, 2),
(7, 'Lithium Hydroxide', 5874, 3),
(8, 'Sodium Carbonate', 1359, 0),
(9, 'Isopropanol', 5262, 0),
(10, 'Hydrogen Peroxide', 3535, 1),
(11, 'Sodium Hydroxide', 4915, 3),
(12, 'Potassium Hydroxide', 4800, 3),
(13, 'Hydrofluoric Acid', 4195, 2),
(14, 'Acetic Anhydride', 2900, 2),
(15, 'Sulphuric Acid', 0083, 2),
(16, 'Imidazole', 9112, 4)
set IDENTITY_INSERT Reagent OFF;
```

Supply Table Insert:

```
set IDENTITY_INSERT Supply ON
insert into Supply(SupplyID, ReagentID, LabID, ReagentStock)
values
(1, 1, 1, 10),
(2, 2, 1, 20),
(3, 3, 1, 15),
(4, 4, 1, 25),
(5, 2, 2, 25),
(6, 5, 2, 15),
(7, 6, 2, 20),
(8, 7, 2, 25),
(9, 8, 3, 30),
(10, 9, 3, 25),
(11, 10, 3, 16),
(12, 11, 3, 23),
(13, 12, 4, 33),
(14, 13, 4, 40),
(15, 14, 4, 50),
(16, 15, 4, 25),
(17, 10, 4, 6)
set IDENTITY_INSERT Supply OFF;
```

Lab Table Insert:

```
set IDENTITY_INSERT Lab ON
insert into Lab(LabID, LabName, LabSite)
values
(1, 'Production Lab', 'Site A'),
(2, 'R&D Lab', 'Site B'),
(3, 'Quality Testing Lab', 'Site C'),
(4, 'Cleanroom 1', 'Site D')
set IDENTITY_INSERT Lab OFF;
```

3. URI Addressing Scheme

To test URI, please log in with Username: Dermot & Password: Dermot2022!

The App uses cookie-based authentication, so options will not appear if not logged in.

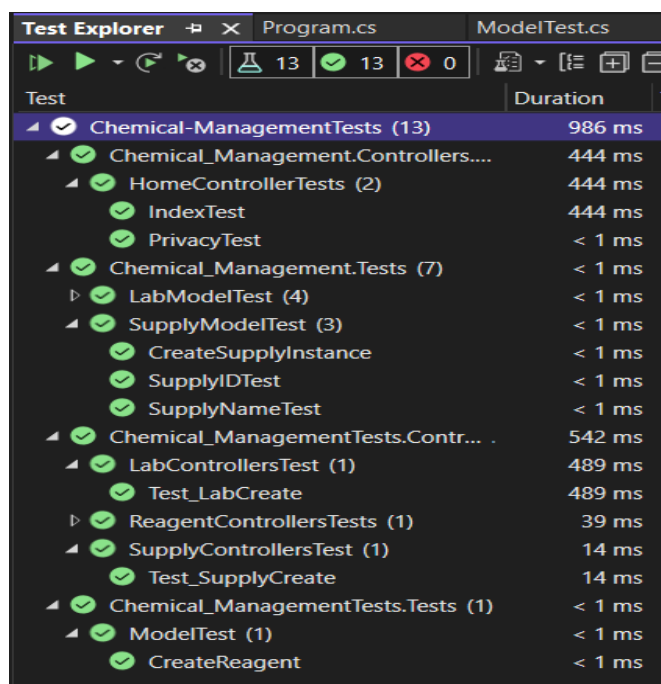
URI Addressing Scheme		
Model / Action	Description	URI
User / Login / POST	Login page for hard-coded users	https://chemical-management.azurewebsites.net/Login
Reagent / GET	Returns All Reagents	https://chemical-management.azurewebsites.net/Reagents
Reagent / POST	Create Reagent	https://chemical-management.azurewebsites.net/Reagents/Create
Reagent / POST	Edit Reagent (ID = 1)	https://chemical-management.azurewebsites.net/Reagents/Edit/1
Reagent / GET	Returns details for single Reagent (ID = 1)	https://chemical-management.azurewebsites.net/Reagents/Details/1
Reagent / DELETE	Deletes Reagent (ID = 1)	https://chemical-management.azurewebsites.net/Reagents/Delete/1
Lab / GET	Returns All Labs	https://chemical-management.azurewebsites.net/Labs
Lab / POST	Create Lab	https://chemical-management.azurewebsites.net/Labs/Create
Lab / POST	Edit Lab (ID = 1)	https://chemical-management.azurewebsites.net/Labs/Edit/1
Lab / GET	Returns details for single Lab (ID = 1)	https://chemical-management.azurewebsites.net/Labs/Details/1
Lab / DELETE	Deletes Lab (ID = 1)	https://chemical-management.azurewebsites.net/Labs/Delete/1
Supply / GET	Returns all Supplies	https://chemical-management.azurewebsites.net/Supplies

Supply / POST	Creates Supply	https://chemical-management.azurewebsites.net/Supplies/Create
Supply / POST	Edit Supply (ID = 1)	https://chemical-management.azurewebsites.net/Supplies/Edit/1
Supply / GET	Returns details for single Supply (ID = 1)	https://chemical-management.azurewebsites.net/Supplies/Details/1
Supply / DELETE	Deletes Supply (ID = 1)	https://chemical-management.azurewebsites.net/Labs/Delete/1

4. Unit Testing:

For Unit Testing, we decided the best strategy was to focus on our models and controllers. For the models, we tested using Assert to see if the models behaved in an expected manner. The first asserts given to our models were for the purpose of proving they (the models) were instances of their types, and the following asserts were used to prove their properties matched with identical types and values given.

For controllers, we tested their 'create' functionality by creating a new instance of the database context from an in-memory database and then passing this to a new model controller instance. A new model instance was created and added to the in-memory database via the controller's 'create' function and stored in a variable. This variable was then tested to assert whether it was null or not, to prove the 'create' function in the controller. All tests created passed.



5. Application and Console Http Client Screenshots

Chemical Management Application

Home Page:

Chemical_Management

Home

Login

Welcome to Lab Chemical Management Application





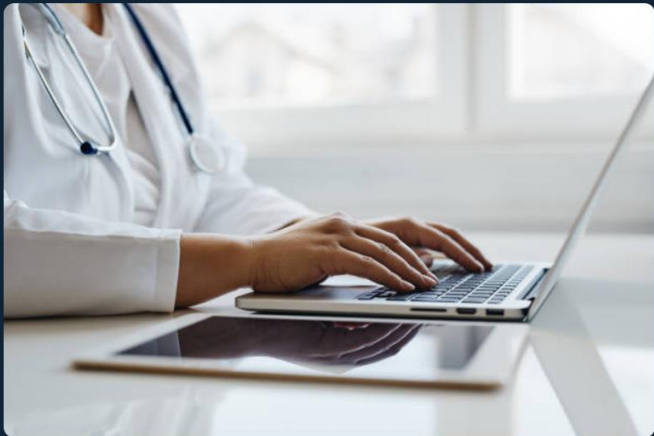
USER GUIDE TO CHEMICAL MANAGEMENT PROGRAM

- 1. Login: Please select the 'Login' link in the top right corner of the navbar. With the Username and Password supplied by the program administrator, please enter the details into the textbox and select 'login'. If your password is incorrect you will be prompted to reenter your details. If correct, all app options will appear for your lab work.
- 2. You will now see options for Labs, Reagents, Stock, Chemical Type, User Management and a button to log out. This program has cookie-based authorization; Therefore, once you are logged in you will have access to these options. However, if your session is interrupted or you logout, these options will no longer be accessible.

Login Page:

Chemical_Management

Home



Please Log-in

Username

Password

login

Once Logged In (options available in navbar):



Reagents Page:

Chemical_Management Home Reagents Labs Supply User Management			
Reagents Registered			
Create New			
Reagent Name	Lot number	Type	
Nitrous Oxide	973	Toxic	Edit Details Delete
Hydrochloric Acid	2825	Acid	Edit Details Delete
Acetone	3159	Organic	Edit Details Delete
Iodine	105	Inorganic	Edit Details Delete
Methanol	5261	Organic	Edit Details Delete
Nitric Acid	1010	Acid	Edit Details Delete
Lithium Hydroxide	5874	Base	Edit Details Delete
Sodium Carbonate	1359	Organic	Edit Details Delete
Isopropanol	5262	Organic	Edit Details Delete
Hydrogen Peroxide	3535	Inorganic	Edit Details Delete
Sodium Hydroxide	4915	Base	Edit Details Delete
Potassium Hydroxide	4800	Base	Edit Details Delete
Hydrofluoric Acid	4195	Acid	Edit Details Delete

Reagent Edit:

Edit

Supply

Reagent ID

Nitrous Oxide

Lab ID

Production Lab

Reagent Stock

10

Save

Reagent Details:

Details

Supply

Reagent Stock

10

Reagent

Nitrous Oxide

Lab

Production Lab

Edit | Back to List

© 2022 - Chemical_Management - Privacy

Reagent Delete:

Delete

Are you sure you want to delete this?

Supply

Reagent Stock

10

Reagent

Nitrous Oxide

Lab

Production Lab

Delete

Back to List

Reagent Create:

Create

Reagent

Reagent Name

Lot number

Type

Organic

Create

Back to List

Labs Page:

Chemical_Management

Home

Reagents

Labs

Supply

User Management

All Labs

Create New

Lab name	Site the lab belongs to	
Production Lab	Site A	Edit Details Delete
R&D Lab	Site B	Edit Details Delete
Quality Testing Lab	Site C	Edit Details Delete
Cleanroom 1	Site D	Edit Details Delete

Labs Edit:

Edit

Lab

Lab name

R&D Lab

Site the lab belongs to

Site B

Save

[Back to List](#)

Lab Details:

Details

Lab

Lab name

Production Lab

Site the lab belongs to

Site A

Edit | [Back to List](#)

Lab Delete:

Delete

Are you sure you want to delete this?

Lab

Lab name

Production Lab

Site the lab belongs to

Site A

Delete | [Back to List](#)

Lab Create:

Create

Lab

Lab name

Site the lab belongs to

Create

[Back to List](#)

Supply Page:

Chemical_Management

[Home](#)[Reagents](#)[Labs](#)[Supply](#)[User Management](#)

Reagent Inventory

Create New

Reagent Stock	Reagent	Lab	
10	Nitrous Oxide	Production Lab	Edit Details Delete
20	Hydrochloric Acid	Production Lab	Edit Details Delete
15	Acetone	Production Lab	Edit Details Delete
25	Iodine	Production Lab	Edit Details Delete
25	Hydrochloric Acid	R&D Lab	Edit Details Delete
15	Methanol	R&D Lab	Edit Details Delete
20	Nitric Acid	R&D Lab	Edit Details Delete
25	Lithium Hydroxide	R&D Lab	Edit Details Delete
30	Sodium Carbonate	Quality Testing Lab	Edit Details Delete
25	Isopropanol	Quality Testing Lab	Edit Details Delete
16	Hydrogen Peroxide	Quality Testing Lab	Edit Details Delete
23	Sodium Hydroxide	Quality Testing Lab	Edit Details Delete
33	Potassium Hydroxide	Cleanroom 1	Edit Details Delete

Supply Edit:

Edit

Supply

Reagent ID

Hydrochloric Acid

Lab ID

Production Lab

Reagent Stock

20

Save

[Back to List](#)

Supply Details:

Details

Supply

Reagent Stock

15

Reagent

Acetone

Lab

Production Lab

Edit | [Back to List](#)

Supply Delete:

Delete

Are you sure you want to delete this?

Supply

Reagent Stock

10

Reagent

Nitrous Oxide

Lab

Production Lab

Delete

 | [Back to List](#)

Supply Create:

Create

Supply

Reagent ID

Nitrous Oxide

Lab ID

Production Lab

Reagent Stock


Create

[Back to List](#)

User Management (no data, as users are hardcoded in the asp.net core project)

Chemical_Management

[Home](#) [Reagents](#) [Labs](#) [Supply](#) [User Management](#)



User Management

[Create New](#)

User name

User Type

© 2022 - Chemical_Management - [Privacy](#)

Console App output from RESTFUL web Service Interaction:

Http Client calling api/[Controller]

Three GET calls.

```
List of Reagents:
Reagent Name:Nitrous Oxide | Reagent Type: Toxic | Assigned ID: 1
Reagent Name:Hydrochloric Acid | Reagent Type: Acid | Assigned ID: 2
Reagent Name:Acetone | Reagent Type: Organic | Assigned ID: 3
Reagent Name:Iodine | Reagent Type: Inorganic | Assigned ID: 4
Reagent Name:Methanol | Reagent Type: Organic | Assigned ID: 5
Reagent Name:Nitric Acid | Reagent Type: Acid | Assigned ID: 6
Reagent Name:Lithium Hydroxide | Reagent Type: Base | Assigned ID: 7
Reagent Name:Sodium Carbonate | Reagent Type: Organic | Assigned ID: 8
Reagent Name:Isopropanol | Reagent Type: Organic | Assigned ID: 9
Reagent Name:Hydrogen Peroxide | Reagent Type: Inorganic | Assigned ID: 10
Reagent Name:Sodium Hydroxide | Reagent Type: Base | Assigned ID: 11
Reagent Name:Potassium Hydroxide | Reagent Type: Base | Assigned ID: 12
Reagent Name:Hydrofluoric Acid | Reagent Type: Acid | Assigned ID: 13
Reagent Name:Acetic Anhydride | Reagent Type: Acid | Assigned ID: 14
Reagent Name:Sulphuric Acid | Reagent Type: Acid | Assigned ID: 15
Reagent Name:Imidazole | Reagent Type: Corrosive | Assigned ID: 16

List of Labs:
Lab Name:Production Lab | Site Name: Site A
Lab Name:R&D Lab | Site Name: Site B
Lab Name:Quality Testing Lab | Site Name: Site C
Lab Name:Cleanroom 1 | Site Name: Site D

List of Supplies:
Reagent Name: Nitrous Oxide | Stock Level: 10 | Reagent ID: 1
Reagent Name: Hydrochloric Acid | Stock Level: 20 | Reagent ID: 2
Reagent Name: Acetone | Stock Level: 15 | Reagent ID: 3
Reagent Name: Iodine | Stock Level: 25 | Reagent ID: 4
Reagent Name: Hydrochloric Acid | Stock Level: 25 | Reagent ID: 2
Reagent Name: Methanol | Stock Level: 15 | Reagent ID: 5
Reagent Name: Nitric Acid | Stock Level: 20 | Reagent ID: 6
Reagent Name: Lithium Hydroxide | Stock Level: 25 | Reagent ID: 7
Reagent Name: Sodium Carbonate | Stock Level: 30 | Reagent ID: 8
Reagent Name: Isopropanol | Stock Level: 25 | Reagent ID: 9
Reagent Name: Hydrogen Peroxide | Stock Level: 16 | Reagent ID: 10
Reagent Name: Sodium Hydroxide | Stock Level: 23 | Reagent ID: 11
Reagent Name: Potassium Hydroxide | Stock Level: 33 | Reagent ID: 12
Reagent Name: Hydrofluoric Acid | Stock Level: 40 | Reagent ID: 13
Reagent Name: Acetic Anhydride | Stock Level: 50 | Reagent ID: 14
Reagent Name: Sulphuric Acid | Stock Level: 25 | Reagent ID: 15
Reagent Name: Hydrogen Peroxide | Stock Level: 6 | Reagent ID: 10

C:\Users\johna\source\repos\Chemical-Management Client\Chemical-Management C
Press any key to close this window . . .
```

Helpful Links

Cookie-Based Auth: <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/cookie?view=aspnetcore-6.0>

Carousel: <https://fontawesome.com/docs/apis/javascript/configuration>

CSS Bootstrap: <https://bootswatch.com/superhero/>