Maddie’s Sprint 1 Research for task [TASK 410](https://dev.azure.com/F22-Team02-Wilson-Bondura-Clay-Hayes-Saad/F22-Team02-Wilson.Bondura.Clay.Hayes.Saad/_workitems/edit/410): Entity framework (EFCore) and ASP.NET videos/notes

STORY: As a developer, I must research our backend technology choices.

* [TASK 410](https://dev.azure.com/F22-Team02-Wilson-Bondura-Clay-Hayes-Saad/F22-Team02-Wilson.Bondura.Clay.Hayes.Saad/_workitems/edit/410): Entity framework (EFCore) and ASP.NET videos/notes
  + RDS db, create with efcore, deploy e3:  <https://www.youtube.com/watch?v=tlXlMk28vEU>
    - What video covers: .net core, EF core, VS and AWS toolkit setup, and docker
    - Need to make sure dotnet and efcore downloaded to terminal
      * Text

        Description automatically generated
    - \*confusing why the mac version looks so different then the windows: seems like you can't do as much
    - Step 2: create AWS RDS database
      * Created and DB connection gained
      * Graphical user interface, website

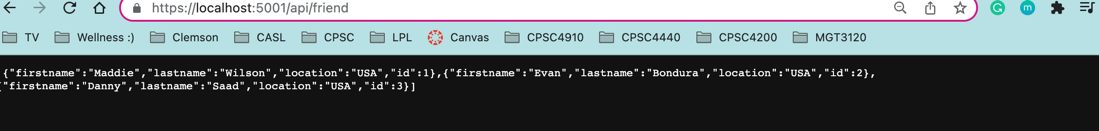
        Description automatically generated

* Graphical user interface

  Description automatically generated
* \*\*didn't do the SQL demo part because have DB stopped and unsure of how resources work waiting to ask professor but followed along still on the visual studio part on connecting
* Can right click on create table and quickly create tables even without coding
  + Then right click on the table and send to sql editor insert statement
* Stage 3 restful web API
  + Add new models folder
  + Right click on API and do manage nuget packages
    - I added .sql server but design wouldn’t add with my netcore app?
    - Graphical user interface, text

      Description automatically generated
  + Then can google connection strings for mysql
    - Server=myServerAddress;Database=myDataBase;Uid=myUsername;Pwd=myPassword;
    - Replace the server address with the endpoint from RDS database in AWS, the db name, username and password
  + Get values dotnet ef dbcontext in terminal so can reference
    - Want to use scaffold command to link the entity types and dbcontext for the db
      * Dotnet ef dbcontext scaffold --help for more info USE the connection string we replaced above
      * Dotnet ef dbcontext scaffold Server=myServerAddress;Database=myDataBase;Uid=myUsername;Pwd=myPassword PROVIDER -o .\Models
        + Is the provider for Pomelo.EntityFrameworkCore.MySQL is what he used I think that is ours too ??
        + And then put in our new models folder
    - Now they are linked so go into the startup file and add the services
      * Services.AddDbContext<FoodStoreContext>
  + Add into controller
    - Api controller with EF framework
    - Data context is our new context we linked together
    - Check the api in local host -> our DB info is there
* STAGE 4: publish the web app to AWS ECS
  + He did ecs but I think we will do ec2??
  + Visual studio -> view -> aws explorer -> choose your service
  + Then right click on the API and do publish container to AWS
    - Add your security groups
  + Can see the cluster and info in visual studio
  + Copy and paste the URL and you can see the db info

* RESTful API with ASP.net:  <https://www.youtube.com/watch?v=LMrW9lkX-TA>
  + \*\*\*This was hard to follow along so switched to the 3 videos below
  + [2. REST API in NET with C# and Visual Studio - PUT, POST AND DELETE](https://www.youtube.com/watch?v=-AKTAi979gw) && [0. How to Setup Visual Studio in MacOS and Build a REST API with C# - Beginner Tutorial](https://www.youtube.com/watch?v=5nlR3Yh5Q_c)
    - \*\*make sure use NetCore
    - API BUILT
    - Graphical user interface, text, application

      Description automatically generated
    - Add classes by right clicking on the API
    - \*right click on empty space
      * Quick actions and refactoring to get constructor
    - New scaffolding at controller
      * Add in lists
    - When done stop and re-run
      * Go to original API but then go to the slash of friends to access friends list we created
      * <https://localhost:5001/api/friend>
    - POST, PUT, DELETE
      * Easy once have model
      * Generate buttons and everything
        + Add scaffolding to generate those values
      * Adding the commands able to get list
      * And then can pull a specific friend by id
    - 

* Graphical user interface, text

  Description automatically generated
* POSTMAN
  + Copy the URL
  + Https request call get, put, post delete (remove items, see, add)
  + Graphical user interface, text, application

    Description automatically generated

* Graphical user interface, text

  Description automatically generated

* Graphical user interface, text

  Description automatically generated

* [Stored Procedures with Entity Framework Core in .NET 6 🚀](https://www.youtube.com/watch?v=nHv9jCPtwT0)
  + Ef core limited on stored procedures
  + Used his starter code to follow along with tutorial
  + Swagger UI I try it out and execute get info BUT not stored procedure
  + Graphical user interface, text, application, email

    Description automatically generated
  + Method fromsqlraw is how we get stored
    - Just use the function and then the procedure want
  + In database
    - Right click on it and create script/query to get the procedure
  + \*\*\*Always use the ExecuteSqlInterpolatedAsync and the FromSqlInterpolated over some showed in this video to defend API from injection attack
  + When used default stored procedures method
    - Set nocount on
      * Method will not return the amount of affected rows
  + NOTE: this guy is great super helpful tutorial and he has a lot we can watch