**Exercise 1**

Create a REST Service for your application.

Use the REST/Web API class resources on Blackboard for this exercise.

Week9\_WebApp1 on the class repo has code for a REST API to use as an example.

1. Copy the files from a previous lab that has a real database (week 4 or later) to a working folder for this weeks lab.  Don't worry about changing names of the files in the copy of the lab.
2. Select the "ASP.NET Web API" link on the REST/WebAPI resources page.  On the resulting page, select the "Getting Started with ASP.NET Web API 2 (C#)" link.  Follow the section titled: "Create a Web API Project".  However, add a new project to your existing copied solution instead of creating a separate solution and project.  Name the new project something that reflects that it is a API (e.g. MyAppApi).
3. Add a reference to your new API project to your existing Web application project.  This allows you to reference various common pieces of your existing Web application.  In a "real" project, you would factor the common pieces (models, database access, etc.) out into a separate library project that both the Web application project and the API application project could reference.
4. You can read the section titled "Adding a Model" but don't add a model.  You'll use a model from your existing Web application project.
5. Follow the section titled: "Adding a Controller".  Instead of creating a controller for Products, create the controller for one of your applications entities (e.g. Pets).
6. Create 2 methods in this controller similar to the ones for Products except use the entity from your application that you chose.
7. Use the data repository you created in the Web application project to access your entity data.  If you don't have a repository class in the lab you chose then just use the DbContext class directly.  In a "real" application, you would use dependency injection to inject a instance of the repository into your service controller.  You can do that if you want or you can just "new" an instance in your controller.
8. Go to NuGet manager and install EntityFramework for your new API project.
9. Find the connection string setting (<connectionStrings>... connectionStrings> in your Web application's web.config file.  Copy it to the web.config file in your new API project.
10. Add the following line to the  Application\_Start method (see global.asax.cs) in your API project (the path should point to your database file in your web application's App\_Data folder):  
      
    AppDomain.CurrentDomain.SetData("DataDirectory", @"D:\Classes\Oregon Tech\Web Development\GitHub\webdev\examples\Week7App\Week7App\App\_Data");
11. Build the solution.  Right click on the new API project in Visual Studio and select: "Set as Startup Project".  Start the application in Visual Studio.  You should see an  error page (HTTP Error 403.14 - Forbidden).  That is OK as we are not expecting for the API app to run in the browser.
12. For testing in the following steps, add a few instances of the entity you chose using your Web application.
13. If you haven't already installed Postman then install it from: https://www.getpostman.com/.
14. Once you have Postman running, attempt to access the list for your entity:
    1. Select the "GET" method in Postman.
    2. Enter the URI for your entity (in the box next to the HTTP method selection).  The URL should look something like: "http://localhost:57170/api/pet" where the number is the port number you see in the browser address bar when you attempted to "run" the API project.  "Pet" in this URL should be the name of your entity.
    3. Select the "Headers" section just underneath the URL input.  Add a header with the key being "Accept" and the value being "application/json".
    4. Click "Send" in the upper right.  You should see a list of your chosen entities.
    5. Try retrieving a single entity using an existing id.  You should see the entity displayed that has the id you specified.
    6. Try retrieving a single entity using a non existing id.  You should get a 404 error in Postman.
15. You now have a REST service that either your Web application or applications (mobile, etc.) could use to access the data stored for you application.

**Exercise 2 (Extra Credit)**

Create a REST client for your new REST service.  Use the REST/Web API class resources on Blackboard for this exercise.

1. Create a new Visual Studio solution.
2. Select the ".NET HTTP Client HowTo" link on the REST/WebAPI resources page.
3. Follow the steps through the section titled "Sending a GET request to retrieve a resource".  Use your entity instead of Product.  Write the code to display a list for your entity.