

How to Publish ASP.NET Core MVC/Web API application to Azure and Consume it using jQuery

In this tutorial I'm going to show how to:

- Create a SQL Database
- Create a .Net Core Web API App and .Net Core Web App
- Deploy Web API to Azure
- Deploy Web App to Azure
- Consume the Web API using jQuery

You can find the complete source code here:

<https://github.com/patricksameerajayalath/Net-Core-WebAPI>

Step 01: Create a SQL Database

We are going to create new SQL Database and add a table called Employee. We will be having bellow structure for an employee.

Column Name	Type
Name	String
Age	Int
DOB	Date
Manager	Bool

Create a SQL Database.

- Resource group: patsam-rg
- Database name: patsam-sqldb
- Server: patsam
- Region: Central US

The screenshot shows the 'Create SQL Database' wizard in the Microsoft Azure portal. The page title is 'Create SQL Database' under 'Microsoft'. The top navigation bar includes 'Home > SQL databases > Create SQL Database'. Below the title, there are tabs: 'Basics' (selected), 'Networking', 'Additional settings', 'Tags', and 'Review + create'. A note says: 'Create a SQL database with your preferred configurations. Complete the Basics tab then go to Review + Create to provision with smart defaults, or visit each tab to customize.' A 'Learn more' link is provided. The 'Project details' section asks to select a subscription and resource group. The 'Subscription' dropdown is set to 'Visual Studio Enterprise – MPN'. The 'Resource group' dropdown is set to 'patsam-rg' with a 'Create new' link below it. The 'Database details' section asks to enter database name and server. The 'Database name' field is filled with 'patsam-sqldb' and has a green checkmark. The 'Server' dropdown is set to '(new) patsam ((US) West Central US)' with a 'Create new' link below it. A question 'Want to use SQL elastic pool? *' has a radio button for 'Yes' (unchecked) and 'No' (checked). The 'Compute + storage' section shows 'Basic' configuration with '2 GB storage' and a 'Configure database' link. At the bottom, there are 'Review + create' and 'Next : Networking >' buttons.

Home > SQL databases

SQL databases

RDP Services Ltd

+ Add Reservations Edit columns Refresh Assign tags Delete

(?) Try our new Azure SQL resource browser! This experience offers a unified view of all your SQL Server resources in Azure as well as improved sorting and filtering. Click here to go to the new experience.

Subscriptions: Visual Studio Enterprise – MPN – Don't see a subscription? Open Directory + Subscription settings

Filter by name... All resource groups All locations All tags No grouping

1 items

Name	Status	Replication role	Server	Pricing tier	Location	Subscription	...
<input type="checkbox"/> patsam-sqldb	Online	None	patsam	Basic	West Central US	Visual Studio Enterprise – MPN	...

Home > SQL databases > patsam-sqldb (patsam/patsam-sqldb)

SQL databases

RDP Services Ltd

+ Add Reservations More

(?) Try our new Azure SQL resource browser! This experience offers a unified view of all your SQL Server resources in Azure as well as improved sorting and filtering. Click here to go to the new experience.

Filter by name... Name patsam-sqldb ...

patsam-sqldb (patsam/patsam-sqldb)

SQL database

Copy Restore Export Set server firewall Delete Connect with... Feedback

Resource group (change) : patsam-rg Status : Online Location : West Central US Subscription (change) : Visual Studio Enterprise – MPN Subscription ID : 3822b9c0-bf1e-455e-a1d1-8717b1b4150b Tags (change) : Click here to add tags

Server name : patsam.database.windows.net Elastic pool : No elastic pool Connection strings : Show database connection strings Pricing tier : Basic Earliest restore point : No restore point available

Show data for last: 1 hour 24 hours 7 days Aggregation type: Max

Compute utilization

Database storage

Notifications (0) Database features (6)

All Security (4) Performance (1) Recovery (1)

- Transparent data encryption** Encryption at rest for your databases, backups, and logs. CONFIGURED
- Advanced data security** Data Discovery & Classification, Vulnerability Assessment and Advanced Threat Protection. NOT CONFIGURED

Configure the Firewall settings to allow accessing it.

Home > SQL databases > patsam-sqldb (patsam/patsam-sqldb) > Firewall settings

Firewall settings

patsam (SQL server)

Save Discard Add client IP

i Connections from the IPs specified below provides access to all the databases in patsam.

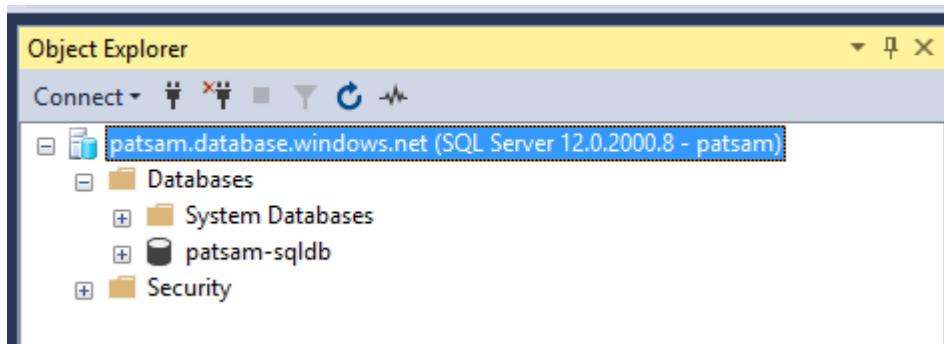
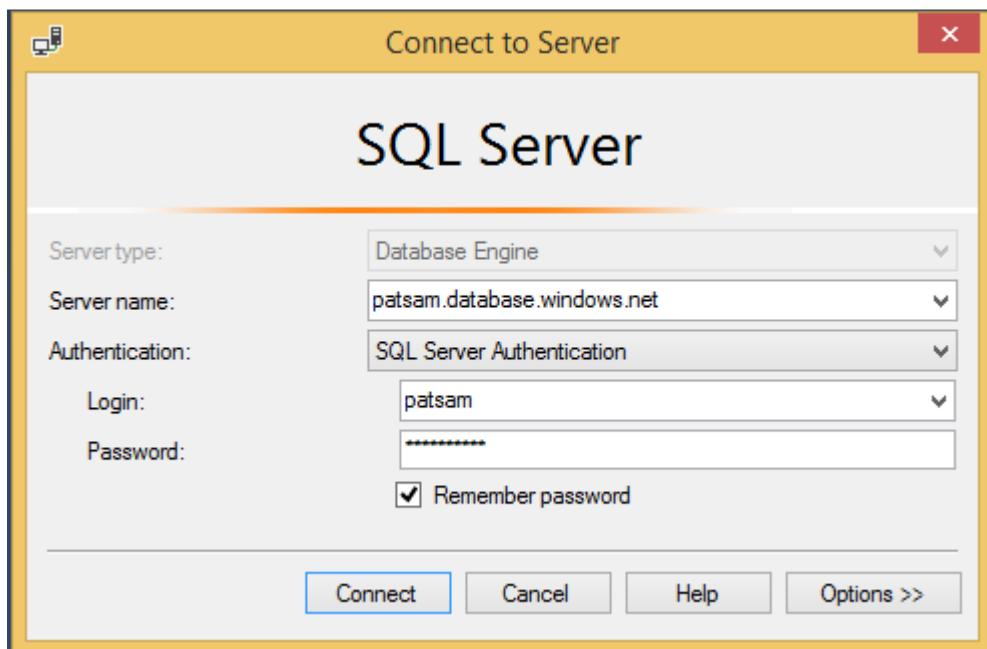
Allow Azure services and resources to access this server ON OFF

Client IP address

Rule name	Start IP	End IP	...
All	0.0.0.0	255.255.255.255	***

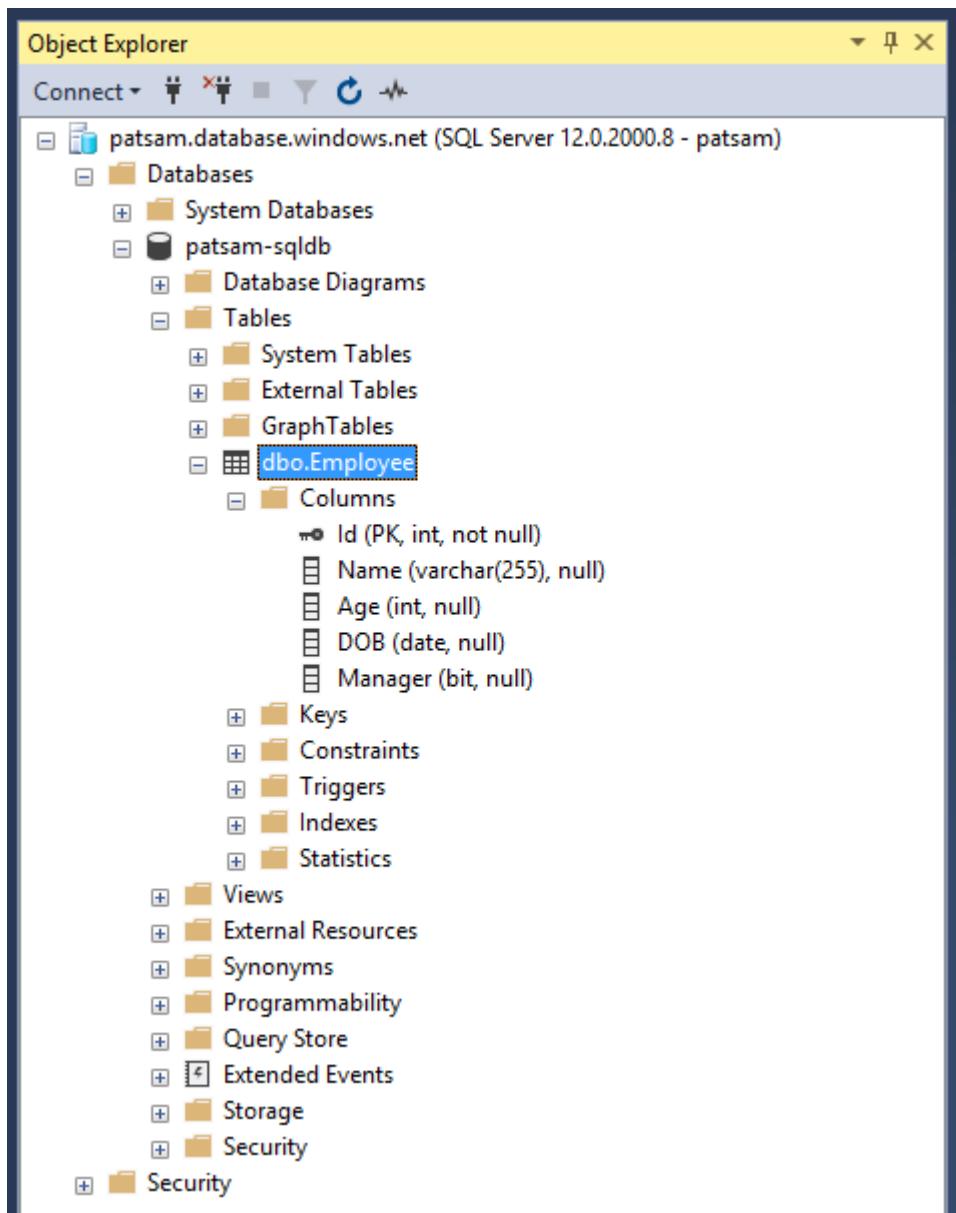
Patrick Sameera Jayalath

Connect to the Database through SQL Server Management Studio.



Next create a new table called Employee.

```
CREATE TABLE Employee (
    Id Int IDENTITY(1,1) PRIMARY KEY,
    Name Varchar(255),
    Age Int,
    DOB Date,
    Manager Bit
);
```



Insert some data to the table.

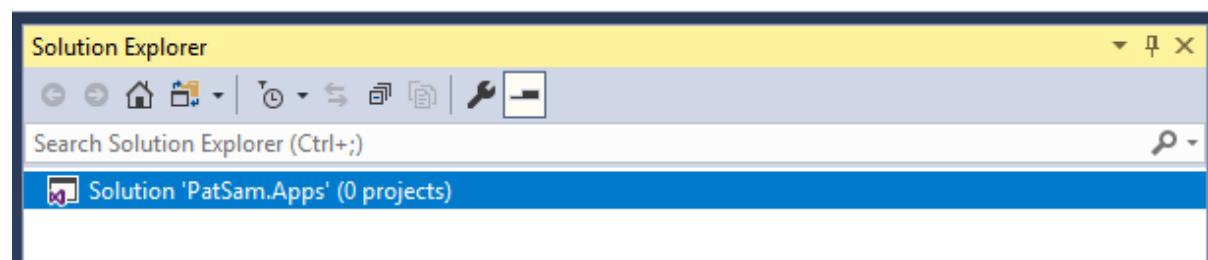
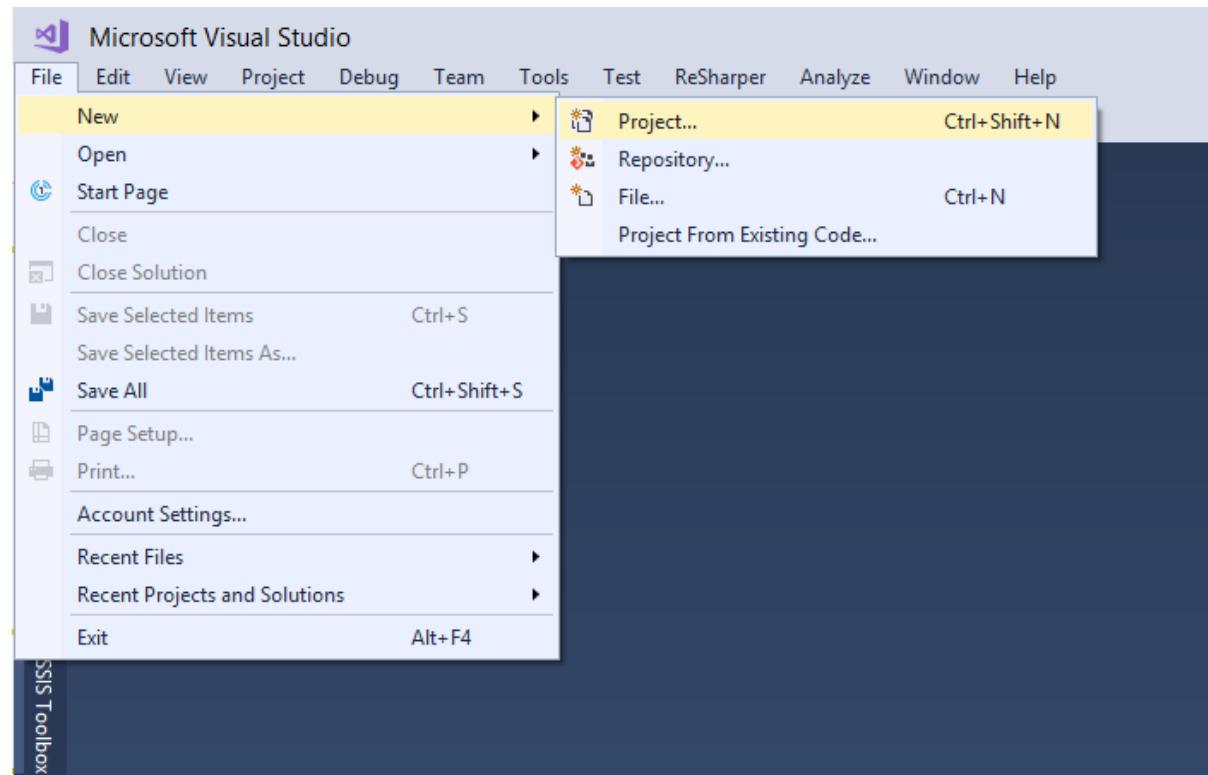
```
Insert Into Employee  
Values ('Patrick Sameera', 30, '05/05/2000', 1)
```

The screenshot shows the SQL Server Management Studio interface with the Object Explorer on the left and a results grid on the right. The results grid displays a single row of data inserted into the Employee table: Id 1, Name 'Patrick Sameera', Age 30, DOB '2000-05-05', and Manager 1.

	Id	Name	Age	DOB	Manager
1	1	Patrick Sameera	30	2000-05-05	1

Step 02: Create a .Net Core Web API App and .Net Core Web App

First, we need to create an empty Project.



Under that we are going to create 3 projects.

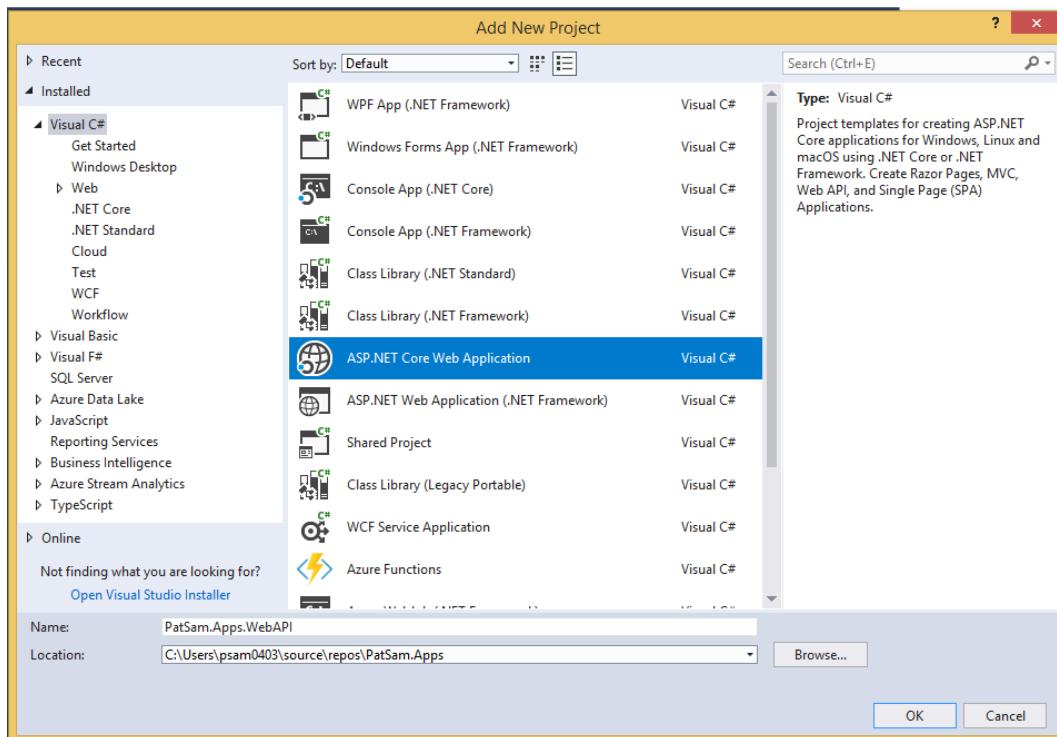
- API/PatSam.Apps.WebAPI
- Shared/PatSam.Apps.Data
- Web/PatSam.Apps.Web

Create 3 folders under that.

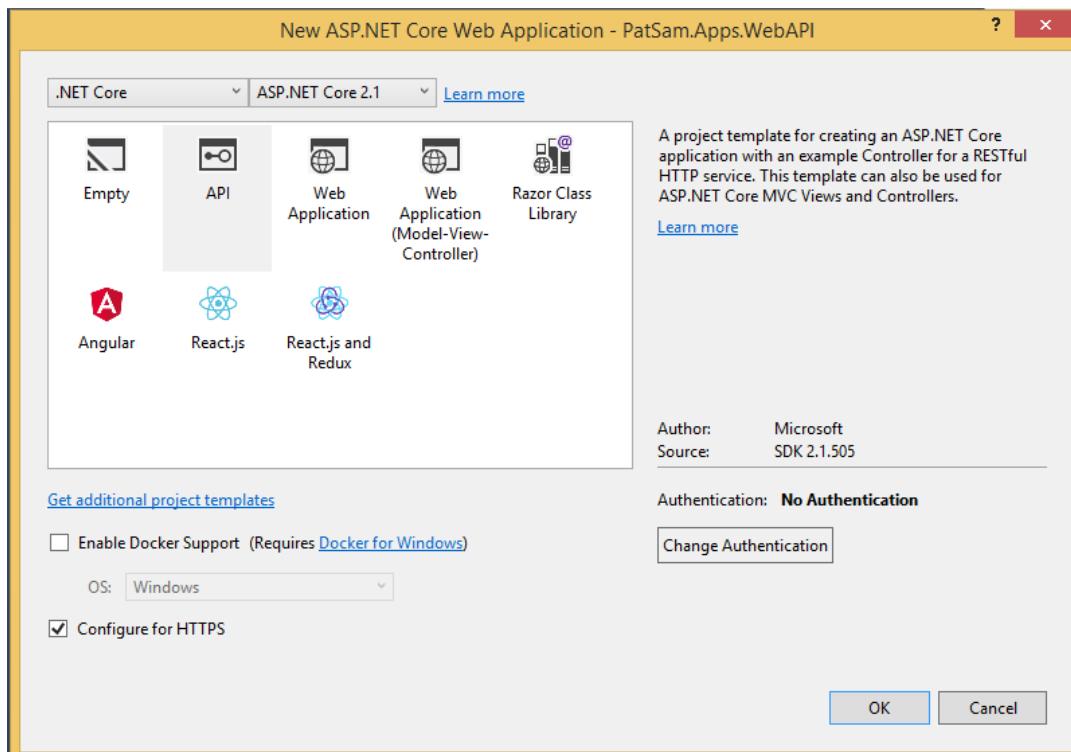
- API
- Shared
- Web

Under API folder create a ASP.NET Core Web Application.

- PatSam.Apps.WebAPI

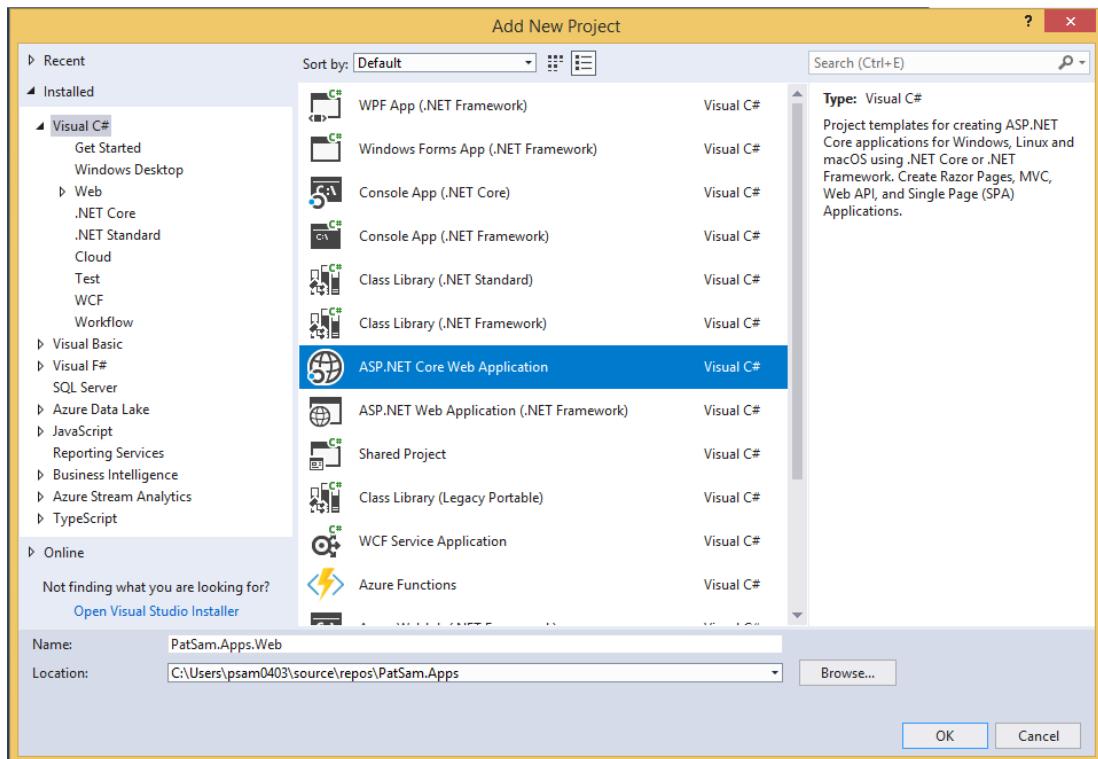


- ASP.NET Core 2.1
- API

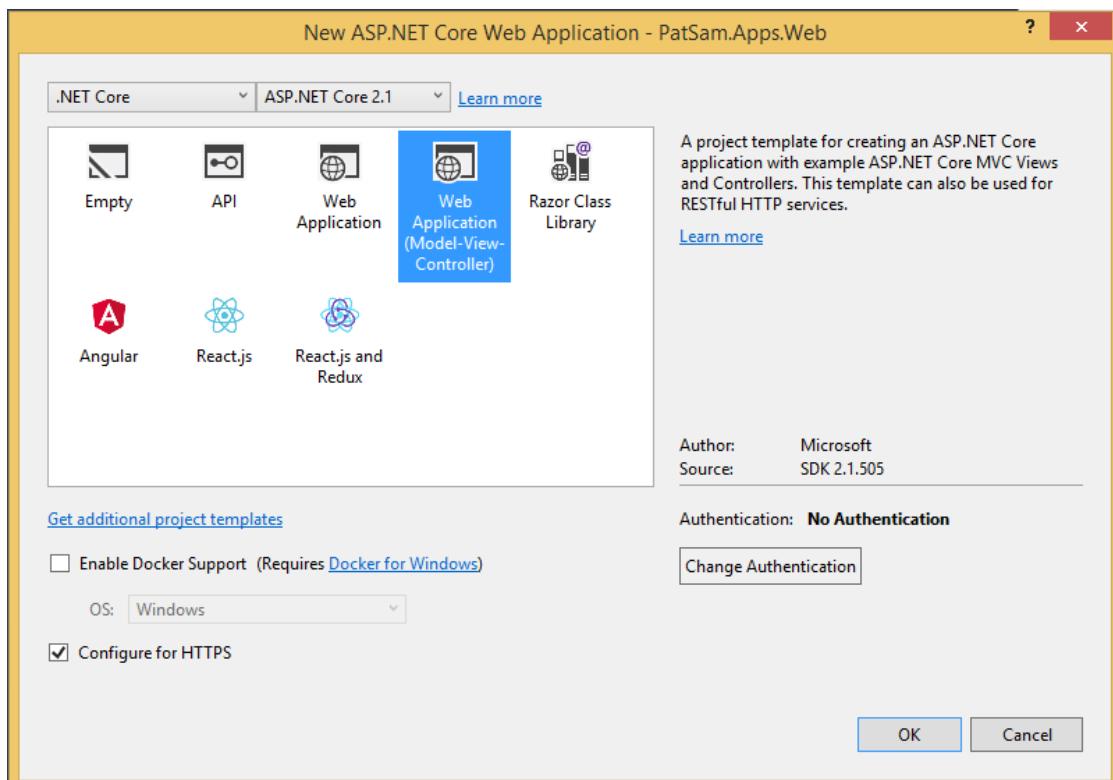


Under Web folder create a ASP.NET Core Web Application.

- PatSam.Apps.Data

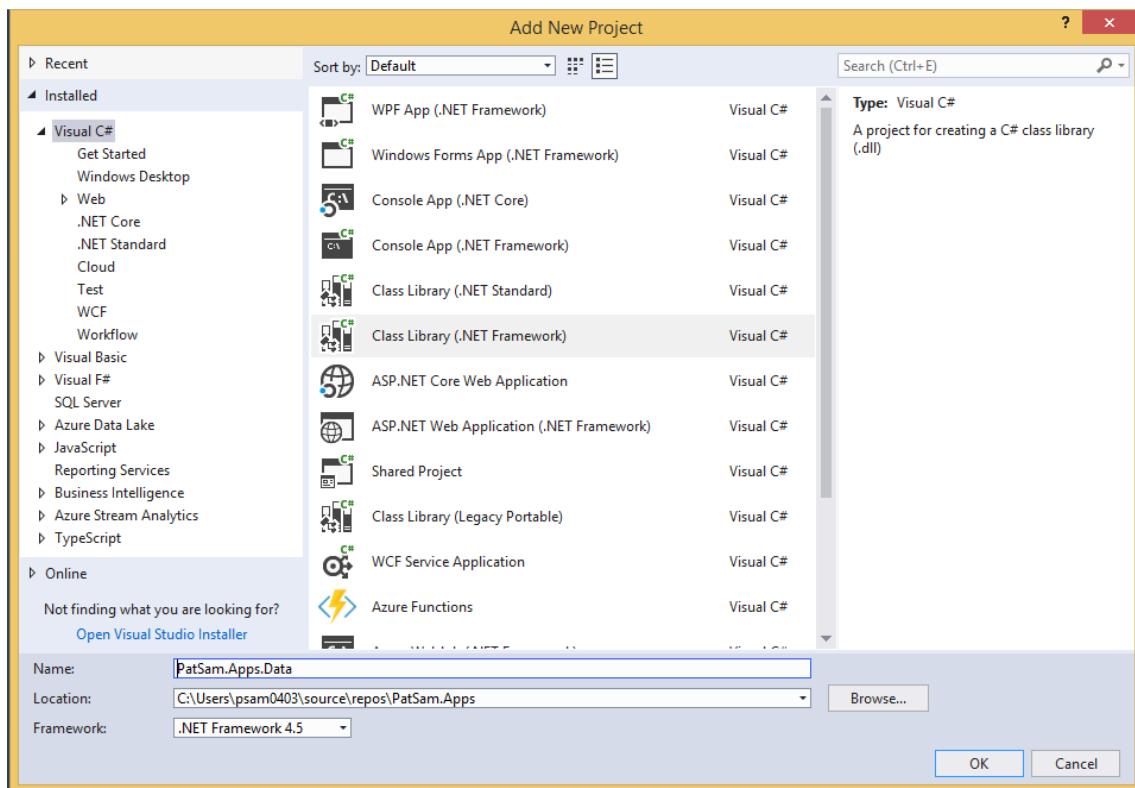


- ASP.NET Core 2.1
- Web Application (MVC)

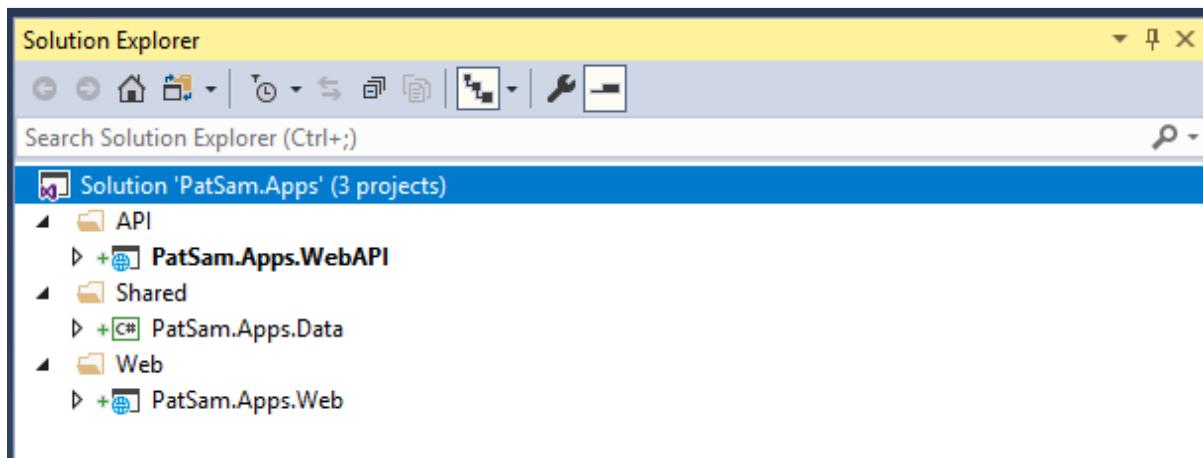


Under Shared folder create a .Net Class Library.

- PatSam.Apps.Data



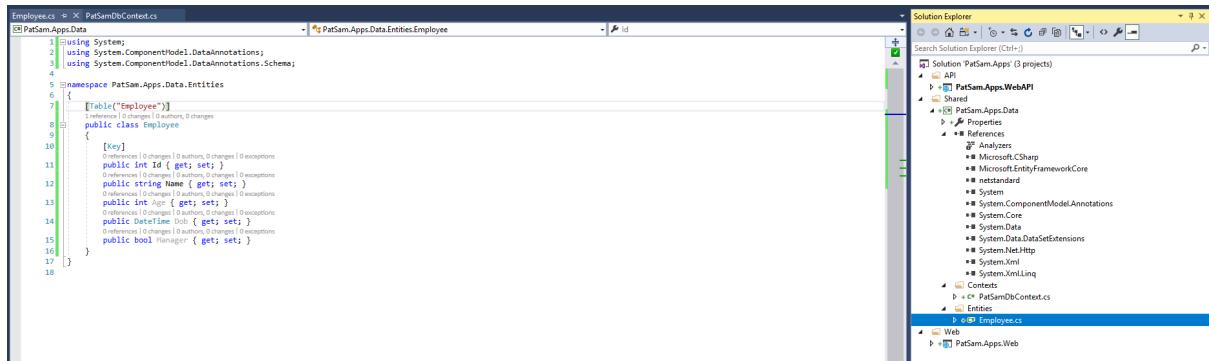
At the end we should have bellow structure.



PatSam.Apps.Data

Under project Shared/PatSam.Apps.Data, create a folder called Entities and inside that create file called Employee.cs and add bellow code.

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.Data/Entities/Employee.cs>

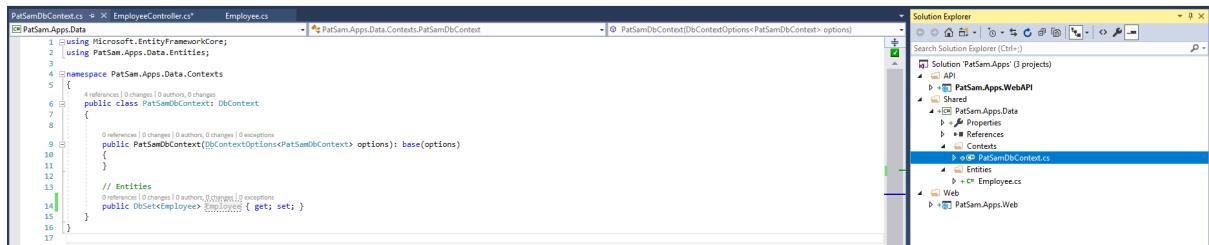


```
Employee.cs  X  PatSamDbContext.cs
[PatSam.Apps.Data]
1  using System;
2  using System.ComponentModel.DataAnnotations;
3  using System.ComponentModel.DataAnnotations.Schema;
4
5  namespace PatSam.Apps.Data.Entities
6  {
7      [Table("Employee")]
8      [Index("Id", IsUnique = true)]
9      public class Employee
10     {
11         [Key]
12         [Column("Id")]
13         public int Id { get; set; }
14
15         [Column("Name")]
16         public string Name { get; set; }
17
18         [Column("Age")]
19         public int Age { get; set; }
20
21         [Column("Manager")]
22         public int Manager { get; set; }
23     }
24 }
```

The screenshot shows the Visual Studio IDE with the Employee.cs file open in the editor. The file contains C# code defining a class named Employee with properties Id, Name, Age, and Manager. The Id property is marked as the primary key. The Manager property is a foreign key reference. The code uses Entity Framework annotations like [Table] and [Column]. The Solution Explorer on the right shows the project structure with the Employee.cs file selected.

Under project Shared/PatSam.Apps.Data, create a folder called Contexts and inside that create file called PatSamDbContext.cs and add bellow code.

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.Data/Contexts/PatSamDbContext.cs>



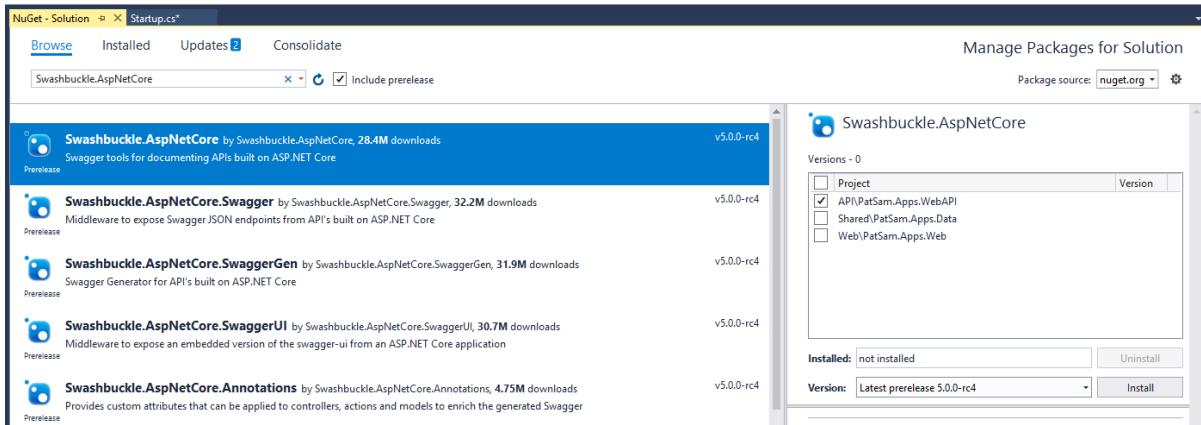
```
PatSamDbContext.cs  X  EmployeeController.cs  Employee.cs
[PatSam.Apps.Data]
1  using Microsoft.EntityFrameworkCore;
2  using PatSam.Apps.Data.Entities;
3
4  namespace PatSam.Apps.Data.Contexts
5  {
6      public class PatSamDbContext : DbContext
7      {
8
9          public PatSamDbContext(DbContextOptions<PatSamDbContext> options) : base(options)
10         {
11
12             // Entities
13             OnModelCreating(ModelBuilder => modelBuilder.Entity<Employee>().HasIndex(e => e.Id));
14         }
15     }
16 }
```

The screenshot shows the Visual Studio IDE with the PatSamDbContext.cs file open in the editor. The file defines a DbContext named PatSamDbContext that inherits from Microsoft.EntityFrameworkCore.DbContext. It includes a constructor that takes an DbContextOptions<PatSamDbContext> parameter and calls the base constructor. Inside the constructor, it uses the OnModelCreating method to add an index on the Employee entity's Id column. The Solution Explorer on the right shows the project structure with the PatSamDbContext.cs file selected.

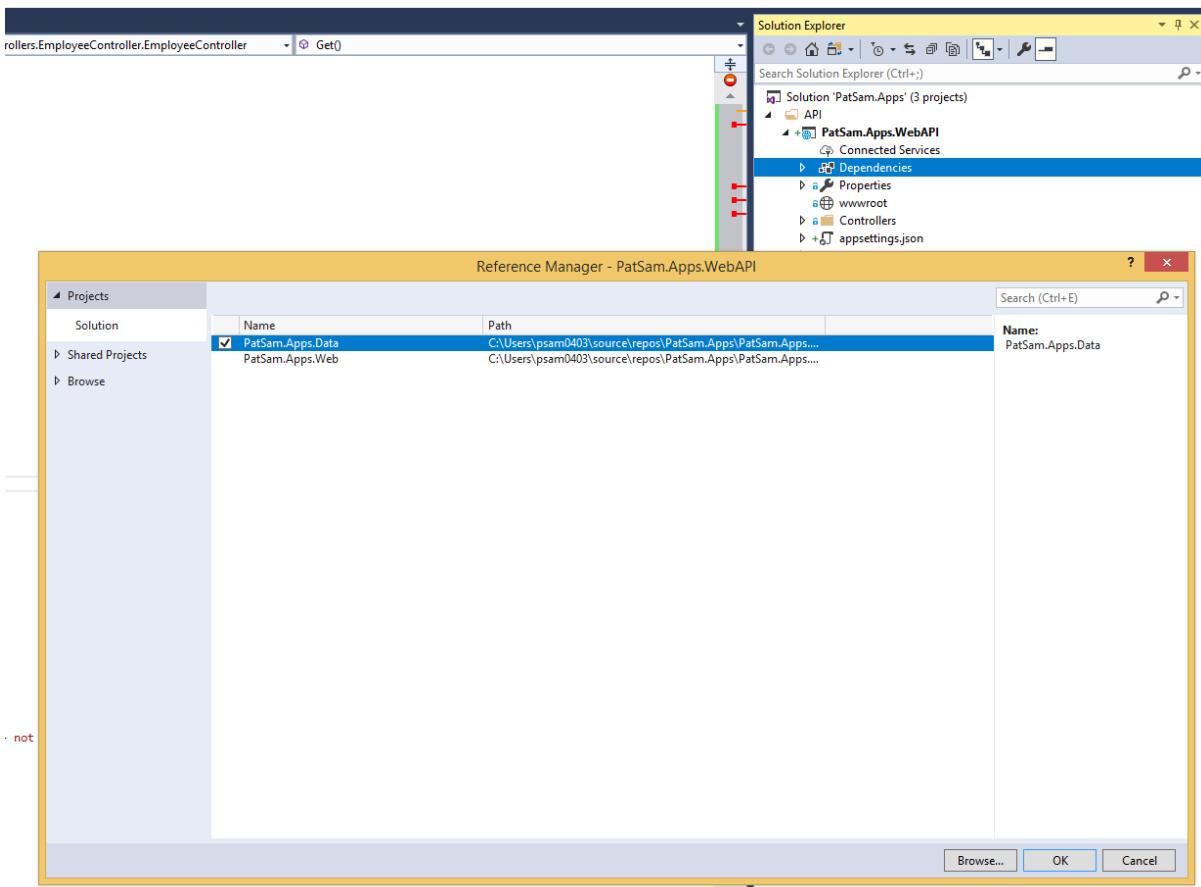
PatSam.Apps.WebAPI

Make sure to install Swashbuckle.AspNetCore through NuGet on PatSam.Apps.WebAPI. This helps us to test out the Web API easily through Swagger.

- Tick “Include prerelease ticked”
- Swashbuckle.AspNetCore -Version 5.0.0-rc4



Then add the PatSam.Apps.Data DLL reference to the PatSam.Apps.WebAPI project.



Under project Shared/PatSam.Apps.WebAPI, under the folder Controllers and create file called EmployeeController.cs and add bellow code.

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.WebAPI/Controllers/EmployeeController.cs>

```

EmployeeController.cs
1  using Microsoft.AspNetCore.Mvc;
2  using Microsoft.EntityFrameworkCore;
3  using PatSam.Apps.Data.Contexts;
4  using PatSam.Apps.Data.Entities;
5  using System.Threading.Tasks;
6
7  namespace PatSam.Apps.WebAPI.Controllers
8  {
9      [Produces("application/json")]
10     [Route("api/[controller]")]
11     [ApiController]
12     [EnableDependencyInjection]
13     [UnitOfWork]
14     [Authorizes]
15     public class EmployeeController : ControllerBase
16     {
17         private readonly PatSamDbContext _context;
18
19         [References]
20         [Changes]
21         [Authors]
22         [Exceptions]
23         public EmployeeController(PatSamDbContext context)
24         {
25             _context = context;
26         }
27
28         [HttpGet]
29         [References]
30         [Changes]
31         [Authors]
32         [Exceptions]
33         public async Task<ActionResult> Get()
34         {
35             var people = await _context.Employee
36                 .ToListAsync()
37                 .ToListAsync();
38
39             return await Task.FromResult(new JsonResult(people));
40         }
41
42         [HttpGet("{id:int}")]
43         [References]
44         [Changes]
45         [Authors]
46         [Exceptions]
47         public async Task<ActionResult> Get(int id)
48         {
49             var employee = await _context.Employee.FindAsync(id);
50
51             if (employee == null)
52             {
53                 return new NotFoundObjectResult(value $"Employee with Id {id} not found.");
54             }
55
56             return await Task.FromResult(new JsonResult(employee));
57         }
58
59         [HttpDelete("{id:int}")]
60         [References]
61         [Changes]
62         [Authors]
63         [Exceptions]
64         public async Task<ActionResult> Delete(int id)
65         {
66             var employee = await _context.Employee.FindAsync(id);
67
68             if (employee == null)
69             {
70                 return new NotFoundObjectResult(value $"Employee with Id {id} not found.");
71             }
72
73             _context.Employee.Remove(employee);
74             await _context.SaveChangesAsync();
75         }
76     }
77 }

```

Do bellow changes in Startup.cs

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.WebAPI/Startup.cs>

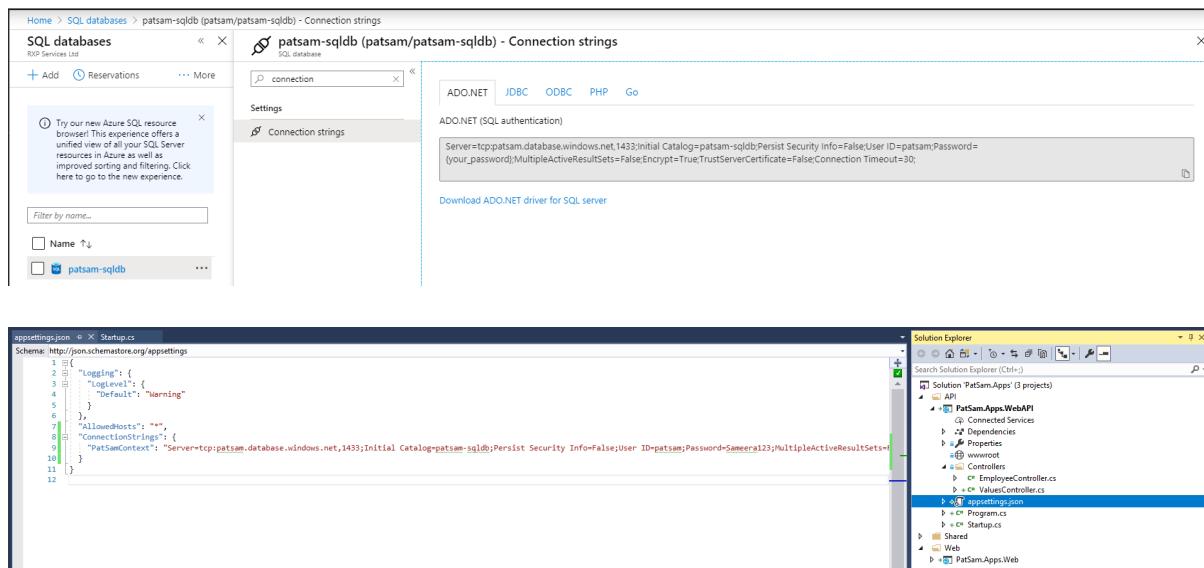
```

Startup.cs
1  using Microsoft.AspNetCore.Builder;
2  using Microsoft.AspNetCore.Hosting;
3  using Microsoft.AspNetCore.Mvc;
4  using Microsoft.EntityFrameworkCore;
5  using Microsoft.Extensions.Configuration;
6  using Microsoft.Extensions.DependencyInjection;
7  using Microsoft.Extensions.Hosting;
8  using PatSam.Apps.Data.Contexts;
9
10 namespace PatSam.Apps.WebAPI
11 {
12     public class Startup
13     {
14         private const string CorsAllowedOrigins = "CorsAllowedOrigins";
15
16         public IConfiguration Configuration { get; }
17
18         // This method gets called by the runtime. Use this method to add services to the container.
19         // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkID=398940
20         public void ConfigureServices(IServiceCollection services)
21         {
22             services.AddCors(option =>
23             {
24                 option.AddPolicy(name: CorsAllowedOrigins, configurePolicy: builder =>
25                 {
26                     builder.AllowAnyOrigin()
27                         .AllowAnyMethod()
28                         .AllowAnyHeader();
29                 });
30             });
31
32             services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_1);
33
34             // *****
35             // Database connection
36             services.AddDbContext<PatSamDbContext>(options =>
37                 options.UseSqlServer(Configuration.GetConnectionString(name: "PatSamContext")));
38
39             // Register the Swagger generator
40             services.AddSwaggerGen(c =>
41                 {
42                     c.SwaggerDoc(name: "v1", new OpenApiInfo { Title = "Employee API", Version = "v1" });
43                 });
44             // *****
45
46             // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
47             // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkID=398941
48             public void Configure(IApplicationBuilder app, IHostingEnvironment env)
49             {
50                 if (env.IsDevelopment())
51                 {
52                     app.UseDeveloperExceptionPage();
53                 }
54             }
55         }
56     }
57 }

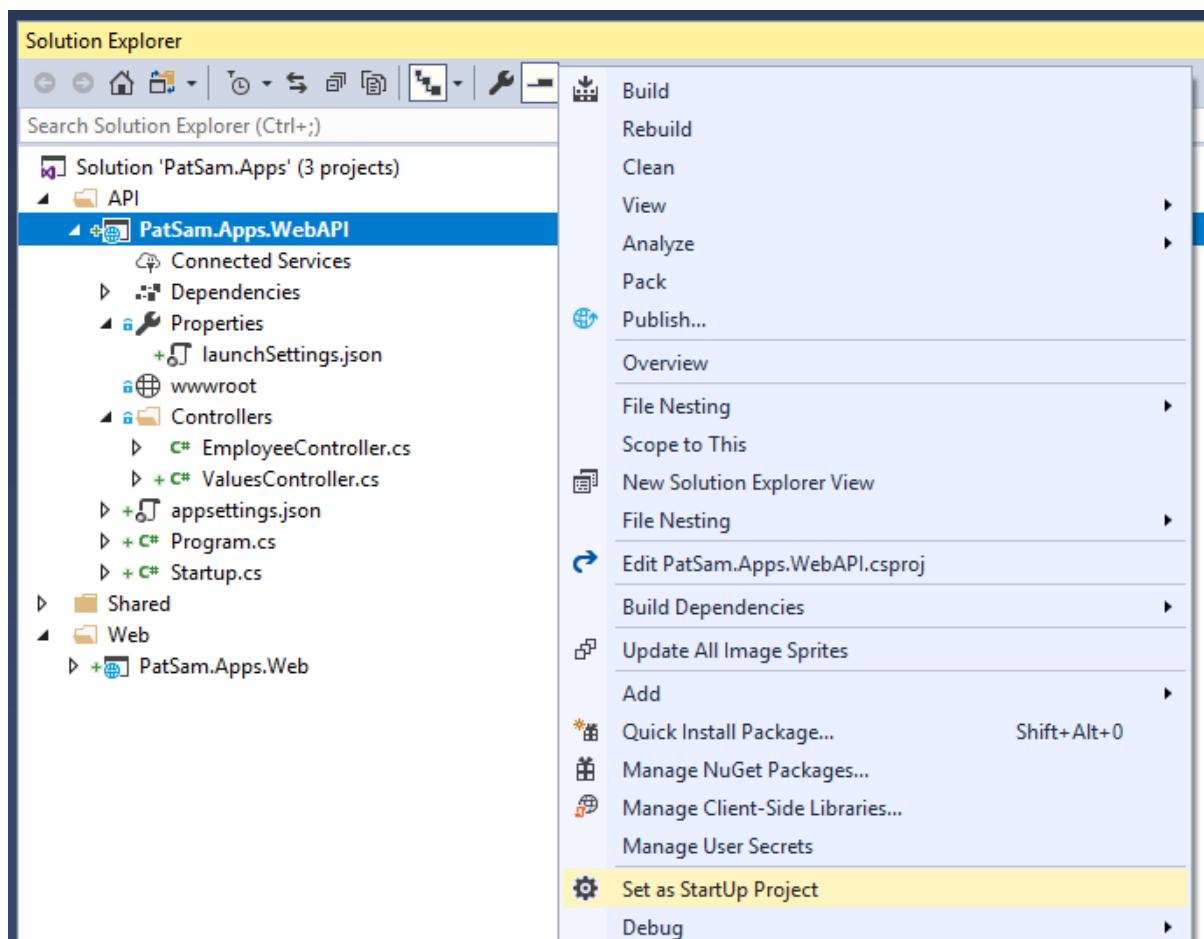
```

Add the Database connection string to the appsettings.json file.

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.WebAPI/appsettings.json>



Clean build the solution and set PatSam.Apps.WebAPI – Set as Startup Project and run it.



Browse bellow URL:

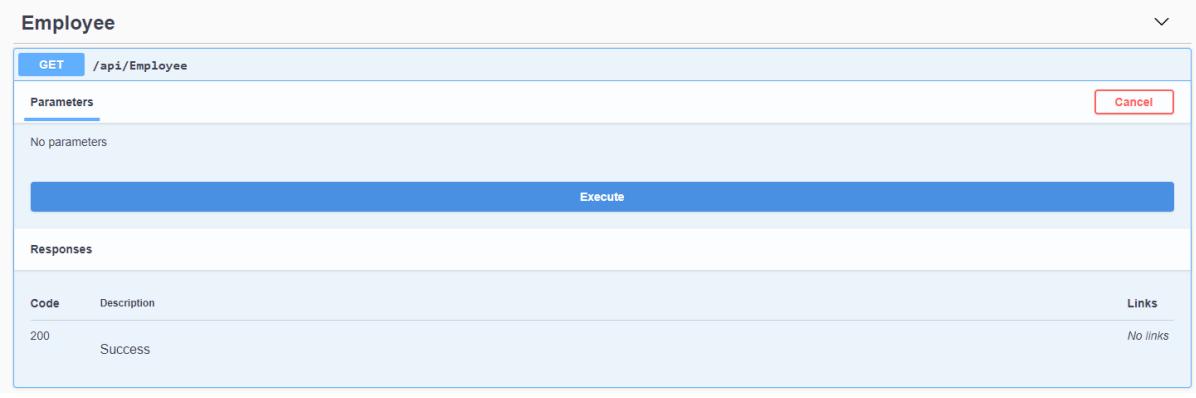
<https://localhost:44321/swagger/index.html>

The screenshot shows the Swagger UI interface for the Employee API v1. At the top, there's a navigation bar with links like Apps, Azure, DevOps, TimeSheets, RXP HR, MSN, OzBargain, CommSec, Udemy, Motley, Azure Code Samples, C# to VB.NET, FAC - DEV, Azure PowerShell, ASP.NET Web Dev..., Upload and Download, and Azure Tips and Tricks. Below the navigation bar, the main title is "Employee API v1" with an OAS3 badge. A dropdown menu "Select a definition" is set to "Employee API v1". The main content area is titled "Employee" and contains two sections: "Employee" and "Values". The "Employee" section has four operations: GET /api/Employee (blue), POST /api/Employee (green, highlighted), GET /api/Employee/{id} (blue), and DELETE /api/Employee/{id} (red). The "Values" section has five operations: GET /api/Values (blue), POST /api/Values (green), GET /api/Values/{id} (blue), PUT /api/Values/{id} (orange), and DELETE /api/Values/{id} (red). At the bottom, there's a "Schemas" section with a link to "Employee".

Click on GET /api/Employee and click on Try it out.

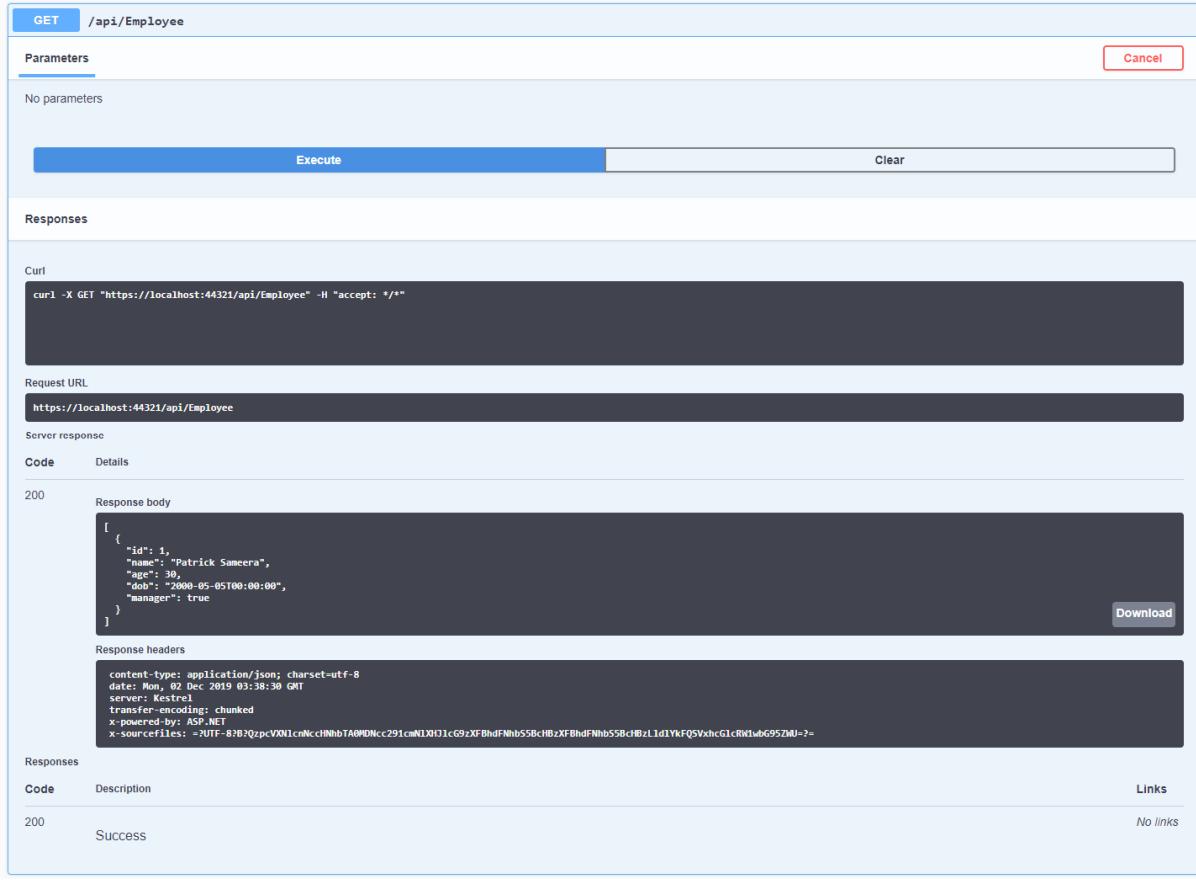
This screenshot shows the detailed view for the GET /api/Employee operation. The top header includes the API title "Employee API v1" and the OAS3 badge. Below the header, the method "GET /api/Employee" is listed. To the right of the method, there's a "Try it out" button. Under the method, there's a "Parameters" section which states "No parameters". In the "Responses" section, there's a table with one row for status code 200 and description "Success". The "Links" column for this row indicates "No links".

Click on Execute.



The screenshot shows a REST API tool interface. At the top, it says "Employee". Below that, a blue bar indicates a "GET /api/Employee" request. Underneath, there's a "Parameters" section with a note "No parameters". On the right, there's a red "Cancel" button. In the center, a large blue "Execute" button is prominent. Below the execute button is a "Responses" section. It contains a table with two rows: one for "Code" (200) and "Description" (Success), and another for "Links" (No links). The entire interface has a light blue background.

It will call the service.



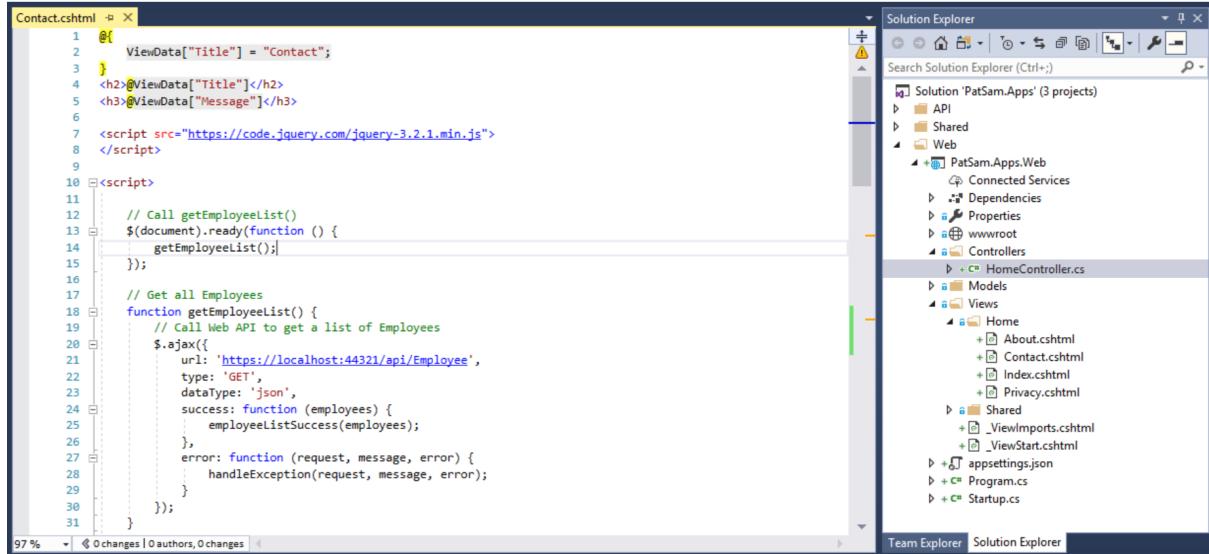
This screenshot shows the same REST API tool after the "Execute" button was clicked. The "Responses" section is expanded and contains several parts: a "Curl" block with the command "curl -X GET "https://localhost:44321/api/Employee" -H "accept: */*""; a "Request URL" block with the URL "https://localhost:44321/api/Employee"; a "Server response" block; and a "Response body" block which displays a JSON array: [{"id": 1, "name": "Patrick Sameera", "age": 30, "dob": "2000-05-05T00:00:00", "manager": true}]. There is a "Download" button next to the response body. The "Responses" section also includes a "Code" (200) and "Description" (Success) row, and a "Links" (No links) row. The overall layout is identical to the first screenshot but with more detailed information in the responses section.

PatSam.Apps.Web

We will be using jQuery AJAX Http Calls to call the Web API and bind data to the page.

We are going to retrieve the data from the Web API and display the data in Contact.cshtml page. Copy bellow code in to Contact.cshtml page.

<https://github.com/patricksameerajayalath/Net-Core-WebAPI/blob/master/PatSam.Apps.Web/Views/Home/Contact.cshtml>



The screenshot shows the Visual Studio IDE interface. On the left is the code editor window titled "Contact.cshtml" containing C# and JavaScript code. On the right is the "Solution Explorer" window showing the project structure for "PatSam.Apps". The "HomeController.cs" file is selected in the "Views\Home" folder. The code in the editor is as follows:

```
1  @{
2     ViewData["Title"] = "Contact";
3 }
4 <h2>@ViewData["Title"]</h2>
5 <h3>@ViewData["Message"]</h3>
6
7 <script src="https://code.jquery.com/jquery-3.2.1.min.js">
8 </script>
9
10 <script>
11     // Call getEmployeeList()
12     $(document).ready(function () {
13         getEmployeeList();
14     });
15
16     // Get all Employees
17     function getEmployeeList() {
18         // Call Web API to get a list of Employees
19         $.ajax({
20             url: 'https://localhost:44321/api/Employee',
21             type: 'GET',
22             dataType: 'json',
23             success: function (employees) {
24                 employeeListSuccess(employees);
25             },
26             error: function (request, message, error) {
27                 handleException(request, message, error);
28             }
29         });
30     }
31 }
```

Step 03: Deploy Web API to Azure

Create API App.

- Reaource group: patsam-rg (exisiting)
- App service plan: patsam-sp (exisiting)

The screenshot shows the 'API App' creation interface in the Azure portal. At the top, there's a breadcrumb navigation: Home > API App. Below it, the title 'API App' is displayed with a 'Create' button. The main form fields include:

- App name ***: The input field contains 'patsamapi' with a green checkmark icon to its right.
- .azurewebsites.net**: A dropdown menu showing 'Visual Studio Enterprise – MPN'.
- Subscription ***: A dropdown menu showing 'Visual Studio Enterprise – MPN'.
- Resource Group * ⓘ**: A section with two radio buttons: 'Create new' (unselected) and 'Use existing' (selected). Below it is a dropdown menu showing 'patsam-rg'.
- *App Service plan/Location**: A section with a dropdown menu showing 'patsam-sp(Central US)' followed by a right-pointing arrow.
- Application Insights**: A section with a dropdown menu showing 'patsamapi' followed by a right-pointing arrow.

At the bottom of the form are two buttons: a blue 'Create' button and a light blue 'Automation options' button.

Once the API App got created sucessfully go inside the resource.

Click Browse to API App.

<https://patsamapi.azurewebsites.net>

The screenshot shows the Azure portal interface for the 'patsamapi' App Service. The left sidebar contains navigation links for Home, Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, Deployment (Quickstart, Deployment slots, Deployment Center), Settings (Configuration, Authentication / Authorization), and Help. The main content area displays the following details:

- Resource group:** patsam-rg
- Status:** Running
- Location:** Central US
- Subscription:** Visual Studio Enterprise – MPN
- Subscription ID:** 3822b9c0-bf7e-455e-a1d1-8717b1b4150b
- Tags:** Click here to add tags
- URL:** https://patsamapi.azurewebsites.net
- App Service Plan:** patsam-sp (F1: Free)
- FTP/deployment username:** patsamapi\deploy-patrick-sameera
- FTP hostname:** ftp://waws-prod-dm1-155.ftp.azurewebsites.windows.net
- FTPS hostname:** https://waws-prod-dm1-155.ftp.azurewebsites.windows.net

Below the main details, there are three cards:

- Diagnose and solve problems:** Our self-service diagnostic and troubleshooting experience helps you identify and resolve issues with your web app.
- Application Insights:** Application Insights helps you detect and diagnose quality issues in your apps, and helps you understand what your users actually do with it.
- App Service Advisor:** App Service Advisor provides insights for improving app experience on the App Service platform. Recommendations are sorted by freshness, priority and impact to your app.

Make sure it works properly.

The screenshot shows the Microsoft Azure website at https://patsamapi.azurewebsites.net. The page has a header with the Microsoft Azure logo and a search bar. Below the header, there is a main message:

Hey, App Service developers!
Your app service is up and running.
Time to take the next step and deploy your code.

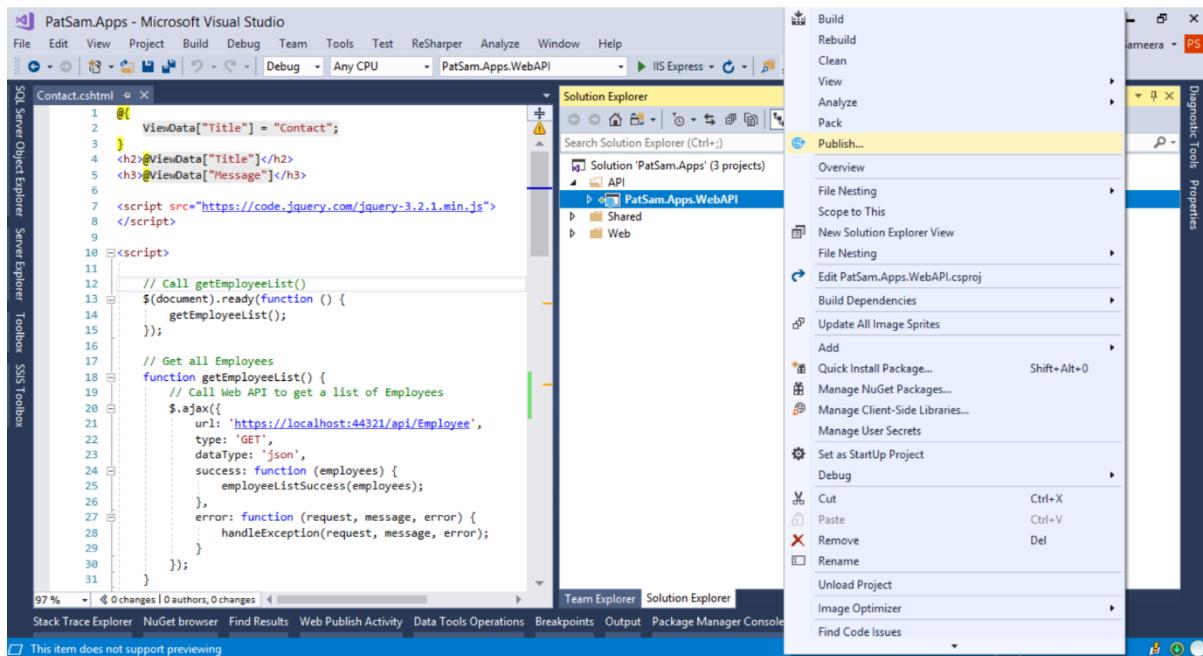
On the left, there are two sections:

- Have your code ready?**
Use deployment center to get code published from your client or setup continuous deployment.
- Don't have your code yet?**
Follow our quickstart guide and you'll have a full app ready in 5 minutes or less.

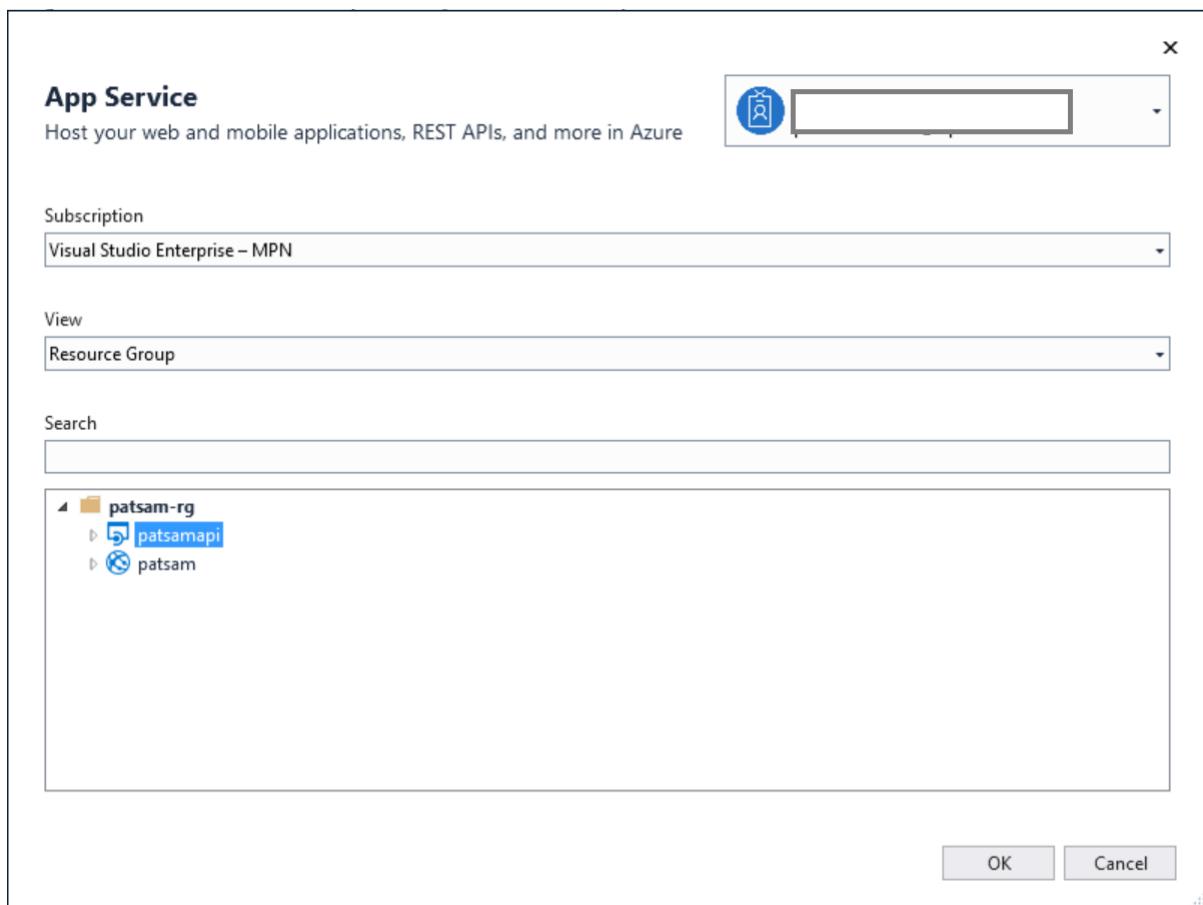
At the bottom of the page are two buttons: **Deployment Center** and **Quickstart**. To the right of the message, there is an illustration of a person working on a laptop, surrounded by various programming language icons (PHP, Python, Java, Node.js, Ruby, .NET Core) in bubbles.

Next publish the Web API to the newly created API App.

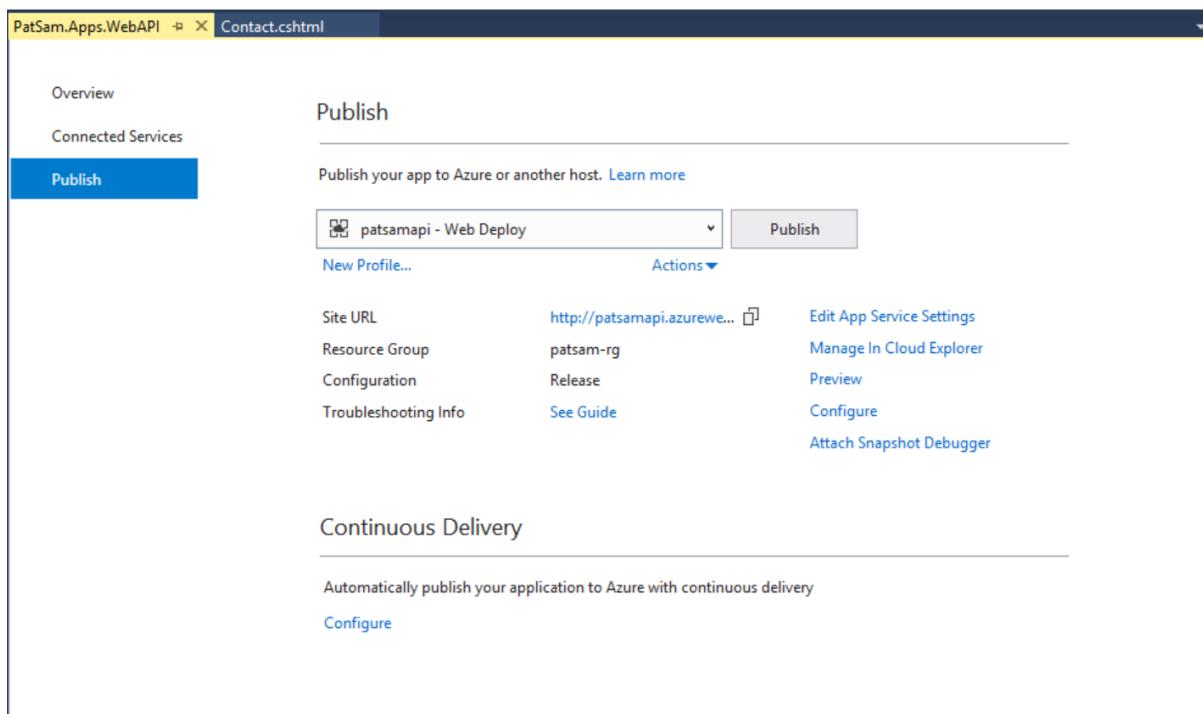
- PatSam.App.WebAPI



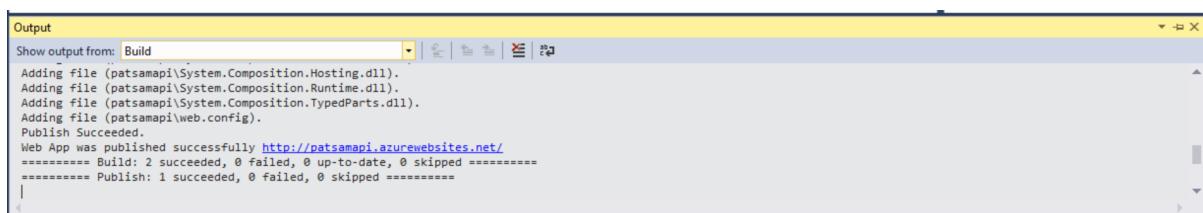
Select the API App we created earlier.



Once the profile been set publish it.

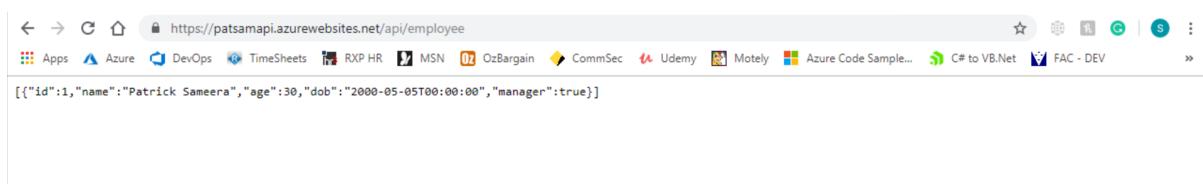


Make sure it gets published sucessfully.



Make sure the Web API works fine after the publish.

<https://patsamapi.azurewebsites.net/api/employee>



Step 04: Deploy Web App to Azure

Create Web App.

- Reaource group: patsam-rg (exisiting)
- App service plan: patsam-sp (exisiting)
- .Net Core 2.1

The screenshot shows the 'Create Web App' wizard in the Azure portal. The current step is 'Web App'. The configuration includes:

- Subscription:** Visual Studio Enterprise – MPN
- Resource Group:** patsam-rg
- Name:** patsamwebapp (suffix: .azurewebsites.net)
- Publish:** Code (selected)
- Runtime stack:** .NET Core 2.1 (LTS)
- Operating System:** Windows
- Region:** Central US (Note: Not finding your App Service Plan? Try a different region.)
- App Service Plan:** patsam-sp (F1) (Windows Plan)
- Sku and size:** Free F1 (Shared infrastructure, 1 GB memory)

At the bottom, there are buttons for 'Review + create' and navigation links: '< Previous' and 'Next : Monitoring >'.

Once the Web App got created sucessfully go inside the resource.

Click Browse to Web App.

<https://patsamwebapp.azurewebsites.net/>

Home > Microsoft.Web-WebApp-Portal-22086de3-a495 - Overview > patsamwebapp

patsamwebapp App Service

Search (Ctrl+ /)

Browse Stop Swap Restart Delete Get publish profile Reset publish profile

Click here to access our Quickstart guide for deploying code to your app →

Resource group (change)
patsam-rg

Status
Running

Location
Central US

Subscription (change)
Visual Studio Enterprise – MPN

Subscription ID
3822b9c0-bf7e-455e-a1d1-8717b1b4150b

Tags (change)
Click here to add tags

URL
<https://patsamwebapp.azurewebsites.net>

App Service Plan
patsam-sp (F1: Free)

FTP/deployment username
patsamwebapp\deploy-patrick-sameera

FTP hostname
ftp://waws-prod-dm1-155.ftp.azurewebsites.windows.net

FTPS hostname
ftps://waws-prod-dm1-155.ftp.azurewebsites.windows.net

Diagnose and solve problems

Application Insights

App Service Advisor

https://patsamwebapp.azurewebsites.net

Microsoft Azure

Hey, App Service developers!

Your app service is up and running.
Time to take the next step and deploy your code.

Have your code ready?
Use deployment center to get code published from your client or setup continuous deployment.

Don't have your code yet?
Follow our quickstart guide and you'll have a full app ready in 5 minutes or less.

Deployment Center

Quickstart

PHP

Python

Java

Node.js

.NET Core

Ruby

Before the publish next we need change the Web API URL on the Contact.cshtml page.

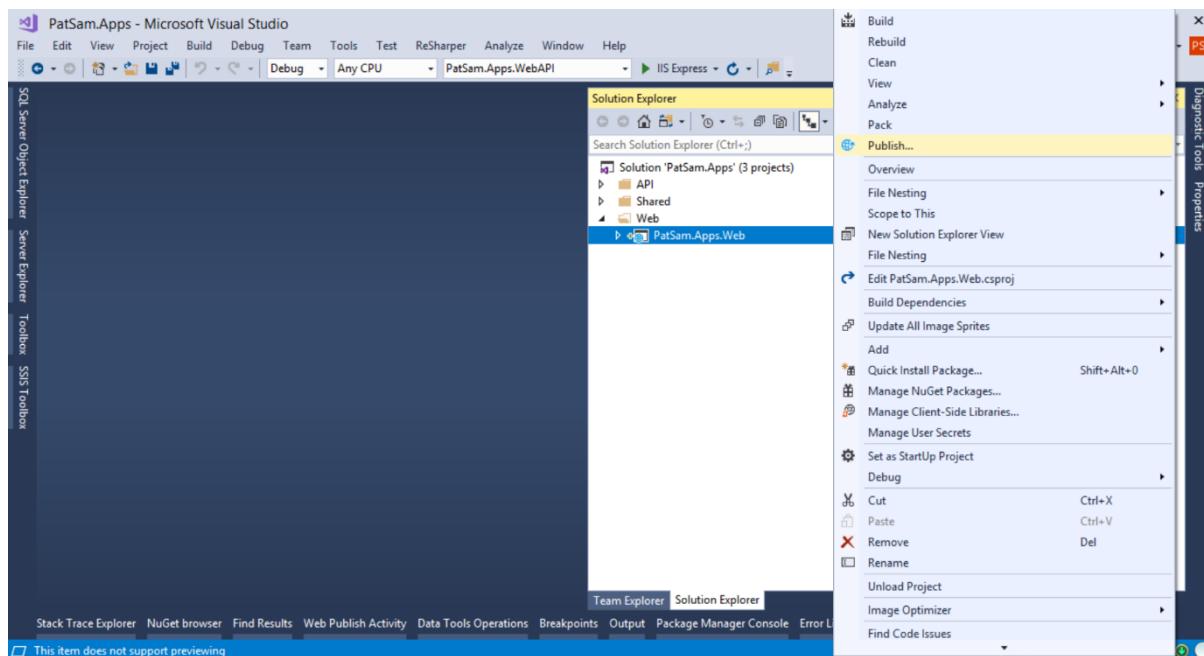
<https://patsamapi.azurewebsites.net/api/employee>



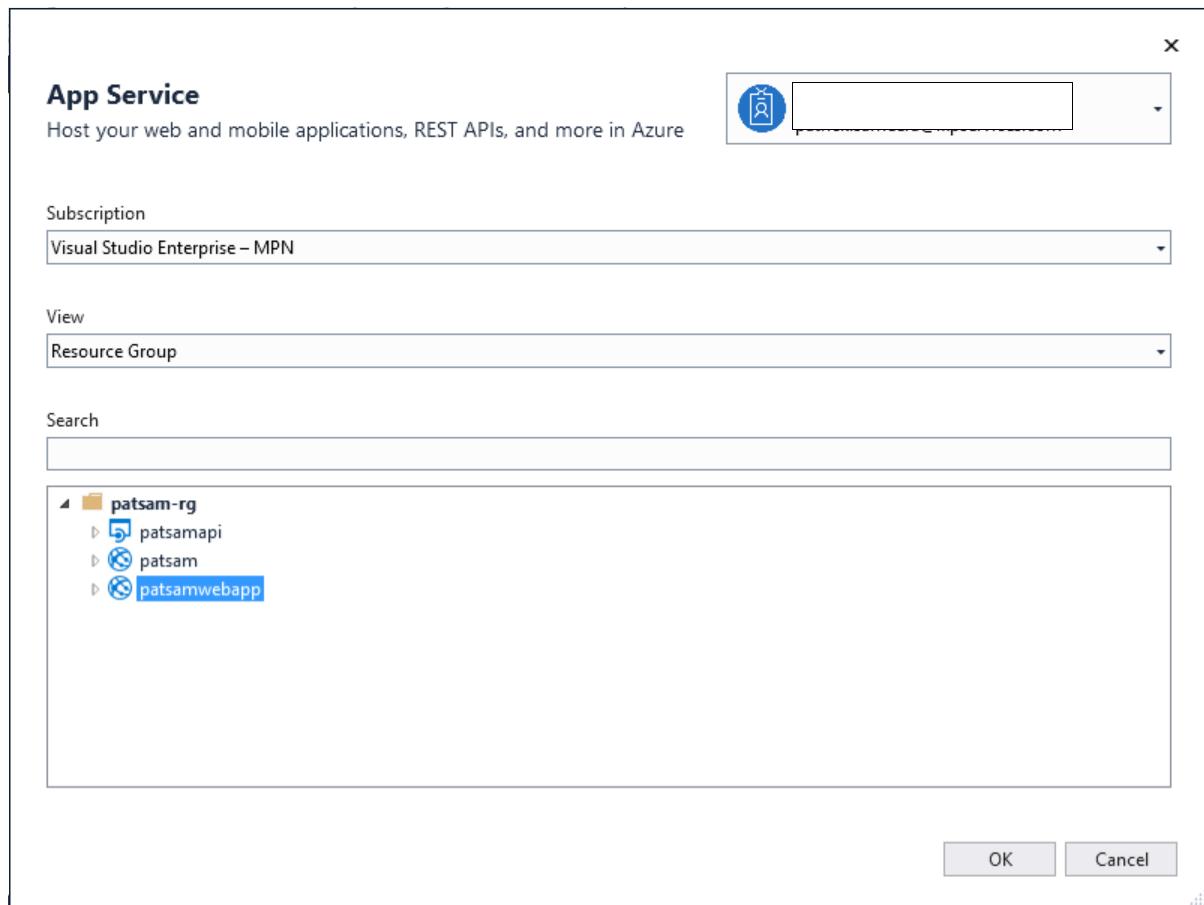
```
16 // Get all Employees
17     function getEmployeeList() {
18         // Call Web API to get a list of Employees
19         $.ajax({
20             url: 'https://patsamapi.azurewebsites.net/api/employee',
21             type: 'GET',
22             dataType: 'json',
23             success: function (employees) {
24                 employeeListSuccess(employees);
25             },
26             error: function (request, message, error) {
27                 handleException(request, message, error);
28             }
29         });
30     };
31 }
```

Next publish the web app to the newly created Web App.

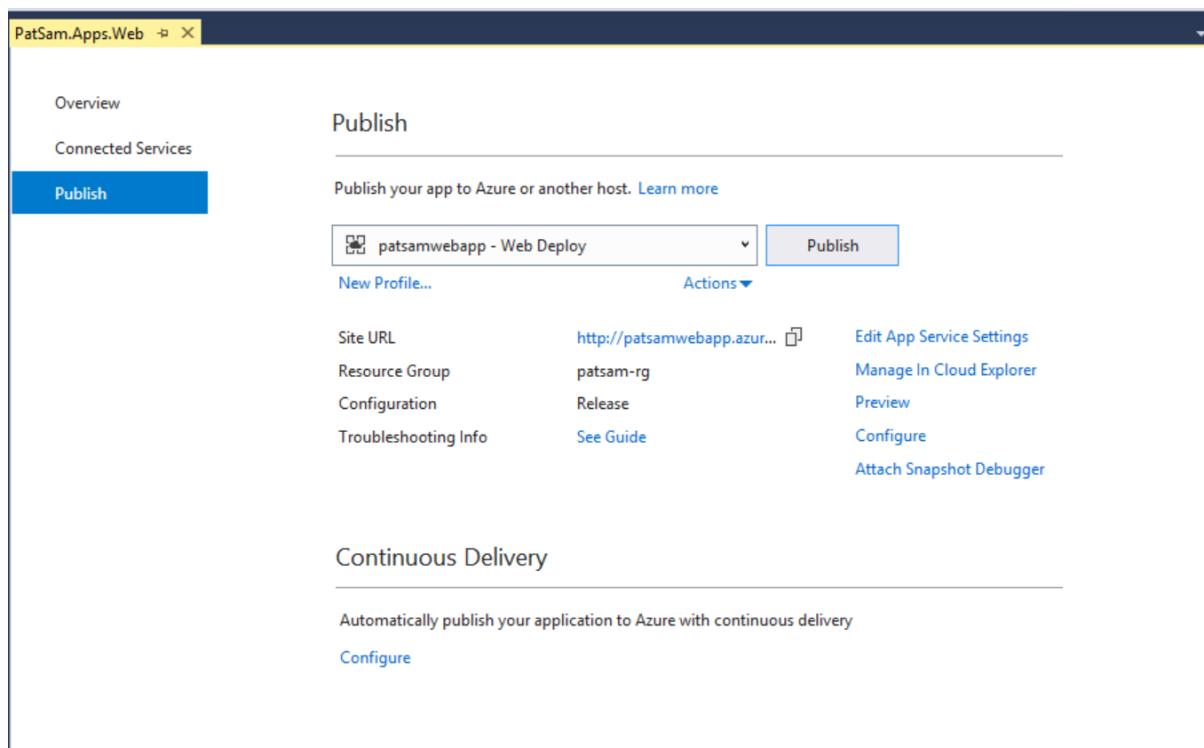
- PatSam.App.Web



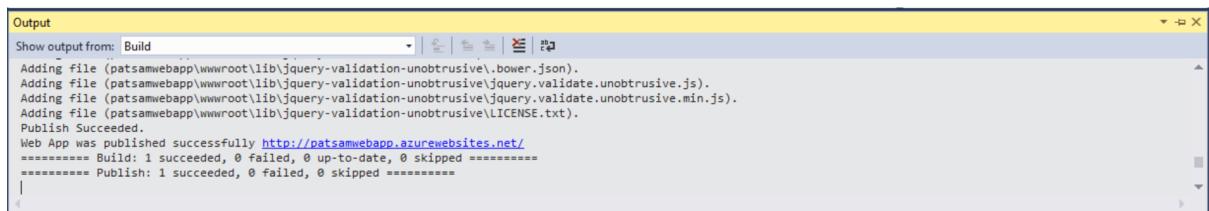
Select the Web App we created earlier.



Once the profile been set publish it.



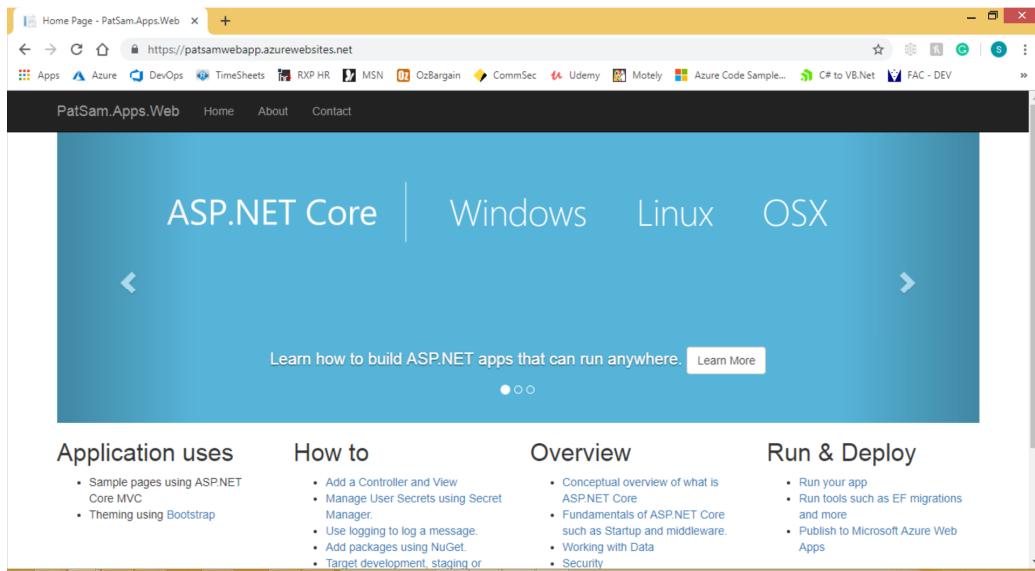
Make sure it gets published sucessfully.



```
Output
Show output from: Build
Adding file (patsamwebapp\wwwroot\lib\jquery-validation-unobtrusive\bower.json).
Adding file (patsamwebapp\wwwroot\lib\jquery-validation-unobtrusive\jquery.validate.unobtrusive.js).
Adding file (patsamwebapp\wwwroot\lib\jquery-validation-unobtrusive\jquery.validate.unobtrusive.min.js).
Adding file (patsamwebapp\wwwroot\lib\jquery-validation-unobtrusive\LICENSE.txt).
Publish Succeeded.
Web App was published successfully http://patsamwebapp.azurewebsites.net/
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

Browse the Web App after the publish.

<https://patsamwebapp.azurewebsites.net/>

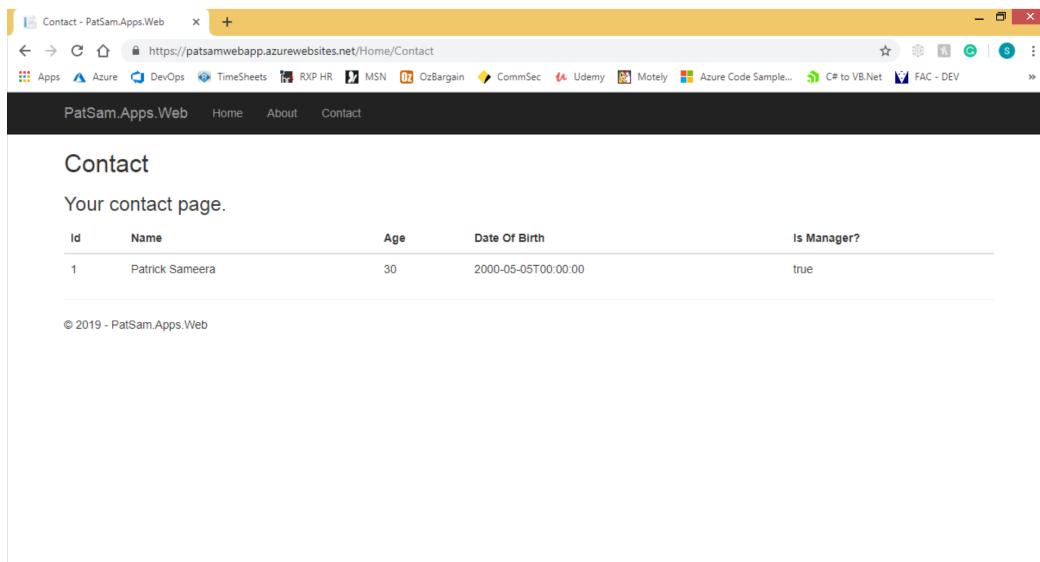


The screenshot shows a web browser window with the URL <https://patsamwebapp.azurewebsites.net/>. The page title is "Home Page - PatSam.Apps.Web". The main content area features a blue banner with the text "ASP.NET Core | Windows | Linux | OSX" and a call-to-action button "Learn how to build ASP.NET apps that can run anywhere. Learn More". Below the banner, there are four sections: "Application uses", "How to", "Overview", and "Run & Deploy", each with a list of links.

Application uses	How to	Overview	Run & Deploy
<ul style="list-style-type: none">Sample pages using ASP.NET Core MVCTheming using Bootstrap	<ul style="list-style-type: none">Add a Controller and ViewManage User Secrets using Secret ManagerUse logging to log a message.Add packages using NuGet.Target development, staging or	<ul style="list-style-type: none">Conceptual overview of what is ASP.NET CoreFundamentals of ASP.NET Core such as Startup and middleware.Working with DataSecurity	<ul style="list-style-type: none">Run your appRun tools such as EF migrations and morePublish to Microsoft Azure Web Apps

Click on Contact link.

<https://patsamwebapp.azurewebsites.net/Home/Contact>



The screenshot shows a web browser window with the URL <https://patsamwebapp.azurewebsites.net/Home/Contact>. The page title is "Contact". The content displays a table with one row, showing a contact named Patrick Sameera with age 30 and birth date 2000-05-05, and the value "true" for the "Is Manager?" column. At the bottom of the page, there is a copyright notice: "© 2019 - PatSam.Apps.Web".

ID	Name	Age	Date Of Birth	Is Manager?
1	Patrick Sameera	30	2000-05-05T00:00:00	true

Patrick Sameera Jayalath