Project guide for IoTDemo API’s

Table of Contents

[1. About the Solution 2](#_Toc482024699)

[2. About the Project structure 2](#_Toc482024700)

[3. Publishing & Hosting in IIS 2](#_Toc482024701)

[3.1. Publishing 2](#_Toc482024702)

[3.2. Hosting 3](#_Toc482024703)

[4. Testing the API 3](#_Toc482024704)

Document Versioning

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Purpose | Date |
| V0.1 | Sagar Yerva | Created the guide | May 8, 2017 |
|  |  |  |  |
|  |  |  |  |

# About the Solution

\* The IoTDemo solution is developed using latest Visual Studio 2017 edition, it has only 1 project at this time.

\* The project name is IoTDemo.API and the project type is Web API

\*The stack followed by the solution is as follows:

1. Asp.Net core – version 1.1.1 (<https://docs.microsoft.com/en-us/aspnet/core/>)
2. Asp.Net core MVC – 1.1.2
3. Entity Framework core – 1.1.1
4. Entity Framework core SQLServer – 1.1.1

All of the above mentioned version were the latest at the time of developing the application.

# About the Project structure

* 1. **Controller** -The IoTDemo.API project uses a single controller IoTDataController, which inherits from Controller class in Asp.Net core MVC.

In Asp.Net core, every public method in the controller is an HTTP endpoint so that each method can be called as a web service, this comes out of the box by Asp.Net core since it focuses on micro services.

* 1. **Data** – We have created 3 models as per the requirement, placed in Models folder.

We have IoTDemoDbContext class in Data folder which is a context for read and write operations on the database.

* 1. **Database Initializing** – We have a SeedData class in Data folder which creates a database for the provided connection string and inserts seed data if not existing.

The Initialize method in SeedData is called in Startups’ configure method.

* 1. **Logging** – We are using a 3rd party logging framework called Serilog , (<https://www.nuget.org/packages/Serilog/>) this logging framework extends the functionalities of out of the box logging provided by Asp.Net core, (<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/logging>)

Our implementation is logging every request to the methods in controller and the log data includes timestamp and method body as well.

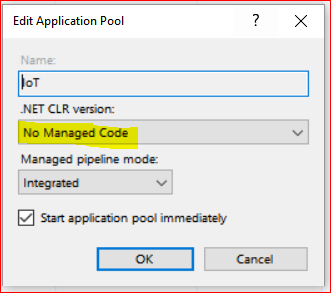
The logs are stored in *Logs* folder at the root of published package.

# Publishing & Hosting in IIS

## Publishing

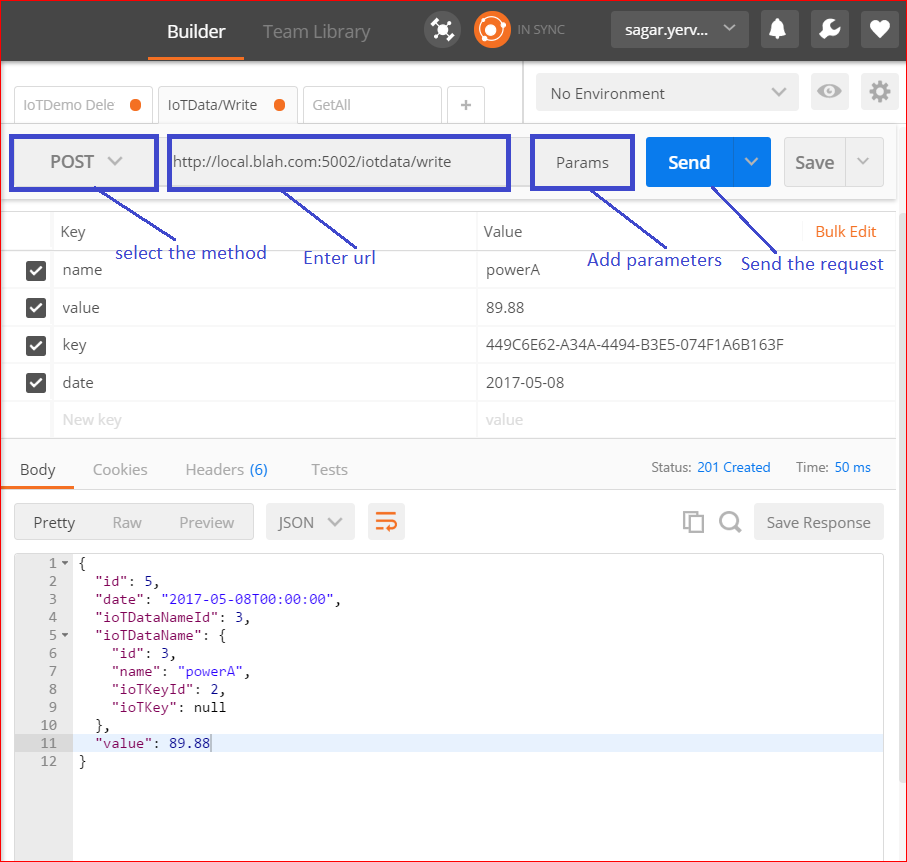
1. Make changes to the IoTDemoDbContext connection string in appsetings.json file, and set the database name, user id & password for the SQL Server (supported version is SQLServer 2012 and above).
2. In case using SQL Server localDB instead of SQL server instance, there is a commented connection string, just uncomment it and rename the database as per your needs.
3. Right click on the IoTDemo.API project and click publish.
4. Select Publish Method as File System and select the directory where you want to publish the application.
5. Target framework is netcoreapp 1.1, click on publish (In Asp.Net core, the published package will include all the framework related dll’s/exe apart from the dll of our application – IoTDemo.API).

## Hosting

1. Now that we have the published directory, we want to host the application in IIS
2. Create a new website in IIS with the name – IoTData
3. Set the physical path to the published directory.
4. Name the application as per your needs, I have named it as IoT.
5. Set the Host name as local.blah.com and set any valid port number if you are not using default port 80.
6. Set the .NET CLR version of this sites’ application pool to No Managed Code as shown below.
7. 
8. You need to add the Asp.Net core module to your IIS in order to run the Asp.Net core applications.
9. You can install this module from Web platform installer, search for “Asp net core module” and install it, after the installation is complete, you should see the entry for *AspNetCoreModule* in the modules list of your IIS.
10. After installing the module, reset the IIS and then browse the site.
11. ***Note*:** Since the project is Web API and there is no UI, you will get a not found error when browsing the url, got to /iotdata/getall to see the available data.

# Testing the API

1. The API’s can be tested by easy to use 3rd party API tools like Postman/SOAP UI.
2. I have tested the API’s using Postman tool, here is a download link - <https://www.getpostman.com/>
3. Postman has a simple UI to hit the requests.



1. Following are the methods provided by IoTData controller (All the responses are in JSON).

In case validation fails, custom messages are returned by the API along with a status code 422 (unprocessable entity), these messages can be configured from code.

1. /iotdata/getall –

Method: GET

Parameters: none

Response: Returns all the records in IotData table.

1. /iotdata/write –

Method: POST

Parameters:

1. name\* – Name of the data to be entered, if this already exists in IoTDataNames table, the IoTDataNameId is set to the existing Id from IoTDataNames table else a new record is created in IoTDataNames table and generated Id is used in the IoTData table.
2. value\* – Value to be inserted against the data, this value should be a valid floating point value.
3. key\* – An unique identifier which is used to identify the user who created the record, the keys are present in IotKey table and a user with a key can add a record only when the Enabled flag in this table is set to true.
4. date- An optional valid date time value should be provided to insert data, if this field is not provided or empty, current date time value on the server is insertd against the record.

Note: All fields marked with \* are mandatory.

There is another method iotdata/writedata which is a GET version of this same feature but POST method is preferred since we are posting the data to server and it includes a key which should not be exposed in the GET request url.

1. /iotdata/delete

Method: DELETE

Parameters:

1. id\* - The Id of the data to be deleted, if this id is not found, a 404 response will be received else the record will be deleted and deleted record will be returned.
2. key\* - Id parameter is provided in the url, but this parameter has to be provided in the request body, if key is not present or disabled, data will not be deleted, API will give a validation error message.