**Dynamic Web Api (Microsoft SQL Server)**

This is a Dynamic Web Api which supports almost any type of request input and out supported by Microsoft SQL Server and allows teams to focus only on database and UI development without even having to make any change in the service. This service does not have any service specific logic and hence could be easily used within any project having SQL Server as backend database. This supports dynamic get and dynamic post method automations.

Once deployed this service does not need any future deployment apart from queries.json.

**Authentication:** Windows Authentication (In IIS to be marked as Windows + Anonymous and as Windows Service needs no authentication to be enabled and in Linux Key Tab/Kerberos Authentication to be enabled).

**WhoAmI:** Used to authenticate users using Windows Authentication provided user is within domain and also retrieves application specific Id if configured within appsettings.json.

**WhoAmIDetailed:** Used to authenticate users using Windows Authentication provided user is within domain and also retrieves application specific Id if configured within appsettings.json and also returns the groups that’s users have access to.

**Get:** Using this one could execute any query and pass on any input parameter via the query params as part of URL.

**Post:** Using this one could input/upload individual variables, single level array of single type objects, single level array of complex objects via request body supported y JSON (string, number, boolean, datetime, array of string, array of number, array of datetime, array of boolean and mix and match of all).

The API is capable of mapping out any input to the supported query or stored procedure and get the output back by mapping the input field names ***as it is*** with sql query or stored procedure.

Supported input type is JSON only but it could take in file content as a byte array of file type Excel, CSV and BLOB.

Excel (First sheet only) and CSV file content is uploaded as byte array within request body the filed name in which byte array is passed is shared with api along with file type and a flag stating file content is present in the request. Sheet name is also shared if specific sheet within excel is needed to be read else first sheet of excel will be read. Based on file type data is converted in to xml and shared with stored procedure/query as an xml. BLOB file type is also shared same way and passed on to query/stored procedure as varbinary type.

**Note:**

Multiple file upload at a time is not supported.

Complex arrays are mapped using SQL table Types which need to follow ADO.Net/SQL Server constraint of mapping the types in the order in which the field names are defined in the type. But browsers JSON serializers tend to serialize the objects and arrange properties alphabetically which hence the type definition in database to be done alphabetically.

This api supports **hot-reload** of the queries and hence restart of the service is not required. Queries are identified using unique key passed on with the calls.

**key (Mandatory parameter)**:key value helps top identity unique query to be executed from queries.json.

**executionType (Mandatory parameter):**

* ScalarText 🡺 Executes a query and returns the count of affected rows.
* NonQueryText 🡺 Executes a query and returns success of failure.
* ScalarProcedure 🡺Executes stored procedure and returns the count of affected rows.
* NonQueryProcedure 🡺 Executes a stored procedure and returns success of failure.
* DataTableText 🡺 Executes a query and returns the first result set.
* DataSetText 🡺Executes a query and returns multiple result set.
* DataTableProcedure 🡺 Executes a stored procedure and returns the first result set.
* DataSetProcedure 🡺 Executes a stored procedure and returns multiple result set.

**outPutType (Optional parameter and default is JSON):**

* JSON 🡺JSON output is given.
* EXCEL 🡺 Data is exported via EXCEL file having one or more result set.
* CSV 🡺Data is exported via CSV file having only one result set.

**filecontentType (Optional, supported with Post method and needed when file content is shared with request body):**

* EXCEL 🡺 Shared as Byte Array with Api and mapped with query/stored procedure as xml and only first sheet data or data of sheet name shared.
* CSV 🡺 Shared as Byte Array with Api and mapped with query/stored procedure as xml.
* BLOB 🡺 Shared as Byte Array with Api and mapped with query/stored procedure as VarBinary.

**sheetName (Optional, supported with Post method and passed on if specifie sheet content is to be read from Excel shared).**

**hasFileContent (Optional, set to be true that is 1 when file content is shared as byte array within request body, default is false).**

|  |  |
| --- | --- |
| **Web Method** | **Sample URL** |
| **WhoAmI** | <http://localhost:5000/api/DynamicWebApi/WhoAmI> |
| **WhoAmIDetailed** | <http://localhost:5000/api/DynamicWebApi/WhoAmIDetailed> |
| **Get** | [http://localhost:5000/api/DynamicWebApi/Get/{key}/{executionType}/{outPutType?}](http://localhost:5000/api/DynamicWebApi/Get/%7bkey%7d/%7bexecutionType%7d/%7boutPutType?%7d%20) |
| **Post** | [http://localhost:5000/api/DynamicWebApi/Post/{key}/{executionType}/{outPutType?}/{hasFileContent?}/{fileContentType?}/{fileContentFieldName?}/{sheetName?}](http://localhost:5000/api/DynamicWebApi/Post/%7bkey%7d/%7bexecutionType%7d/%7boutPutType?%7d/%7bhasFileContent?%7d/%7bfileContentType?%7d/%7bfileContentFieldName?%7d/%7bsheetName?%7d) |

**appsettings.json sample:**

{

"Logging": {

"LogLevel": {

"Default": "Error",

"Microsoft": "Error",

"Microsoft.Hosting.Lifetime": "Error"

},

"options": {

"file": "Log\_$|Date[dd\_MMM\_yyyy]|$.log",

"size": 1073741824

},

"AllowedHosts": "\*",

"IsResolveUserId": "SELECT Id FROM User WHERE UPPER(UserName) = UPPER(@UserName)",

"Environment": "Dev1",

"ConnectionTimeOut": "3600000",

"CorsHost": "http://localhost:4200;http://localhost:4300"

}

}

**connectionstring.json sample:**

{

"Dev1": "Data Source=myServer;Initial Catalog=Dev1;Integrated Security=true;MultipleActiveResultSets=true;",

"Dev2": "Data Source=myServer;Initial Catalog=Dev2;Integrated Security=true;MultipleActiveResultSets=true;"

}

**queries.json sample:**

{

"GetEmployeeDetails\_Query": "SELECT \* FROM DBO.EMPLOYEE WHERE ID = @EmployeeId",

"GetEmployeeDetails\_StoredProcedure": "usp\_GetEmployeeDetails @EID = @EmployeeId"

}

Deployment tip (mostly for IIS): While deploying on server replace the web.config within published folder with web.server.config after renaming it as web.config.