**Table of Contents**

|  |
| --- |
| SAP Financials Upgrade Administrative Appeals Tribunal Revolution IT  Proposal (Tender) |
| Version 1.0 |
| When Quality Matters |

[1. Document Control 2](#_Toc23767983)

[1.1. Reviewers & Authorisations 2](#_Toc23767984)

[1.2. Document Change History 2](#_Toc23767985)

[1.3. Document Reference Personnel 2](#_Toc23767986)

[1.4. Related Documents 2](#_Toc23767987)

[2. Introduction 3](#_Toc23767988)

[2.1. Background 3](#_Toc23767989)

[2.2. Purpose of TDM and DBLibrary 3](#_Toc23767990)

[2.3. Intended Audience 3](#_Toc23767991)

[3. Technical Details 4](#_Toc23767992)

[4. Dependencies 5](#_Toc23767993)

[5. Project Setup 6](#_Toc23767994)

[5.1. Adding Reference 6](#_Toc23767995)

[5.2. Connection String 6](#_Toc23767996)

[5.3. Dictionary to Object Converter. 6](#_Toc23767997)

[6. DBLibrary 7](#_Toc23767998)

[6.1. CommandBuilder 7](#_Toc23767999)

[6.2. Query Executor 8](#_Toc23768000)

[6.3. Excel Executor 8](#_Toc23768001)

[7. Test Data Management 9](#_Toc23768002)

[7.1. Enums 9](#_Toc23768003)

[7.2. CustomBuilders 10](#_Toc23768004)

[7.3. Adapters 10](#_Toc23768005)

[7.4. Controllers 11](#_Toc23768006)

[8. Product Information 12](#_Toc23768007)

[8.1. Swagger Document 12](#_Toc23768008)

[8.1.a. How to use Swagger Document 12](#_Toc23768009)

# Document Control

## Reviewers & Authorisations

|  |  |  |
| --- | --- | --- |
| Name | Title | Review/Authorisation |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Document Change History

| Version No. | Version Date | Author | Comment |
| --- | --- | --- | --- |
| 0.1 | 24/10/2019 | Pratap Surendran | Initial Version |
|  |  |  |  |

## Document Reference Personnel

Queries regarding this document may be directed to:

|  |  |
| --- | --- |
| Name | Title |
| Pratap Surendran | Consultant |

## Related Documents

| Document Name | Version | Document Date | Comment |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Background

Most test automation tools do not have an out-of -the-box integration with database for test data preparation and management to facilitate test execution process.

The Test Data Management project is built with the purpose of facilitating test execution process by exposing API to interact with MS-SQL Server and Oracle databases, and Excel application.

This document’s purpose is to:

* Document all resources required to utlilze DBLibrary.
* Detail the methods and architecture required to better understand; and
* Act as a sample document that will help to get the most benefit by providing example usecases and coding references.

## Purpose of TDM and DBLibrary

The main purpose of the TDM and DBLibrary is to build reusable modules that can be used across different client projects and to build maintainable components that facilitate test execution process.

The purpose of DBLibrary is to hide the low level details of database and Excel interactions, where as the motivation behind Test Data Management (TDM) project is to provide APIs to interact with database and Excel documents.

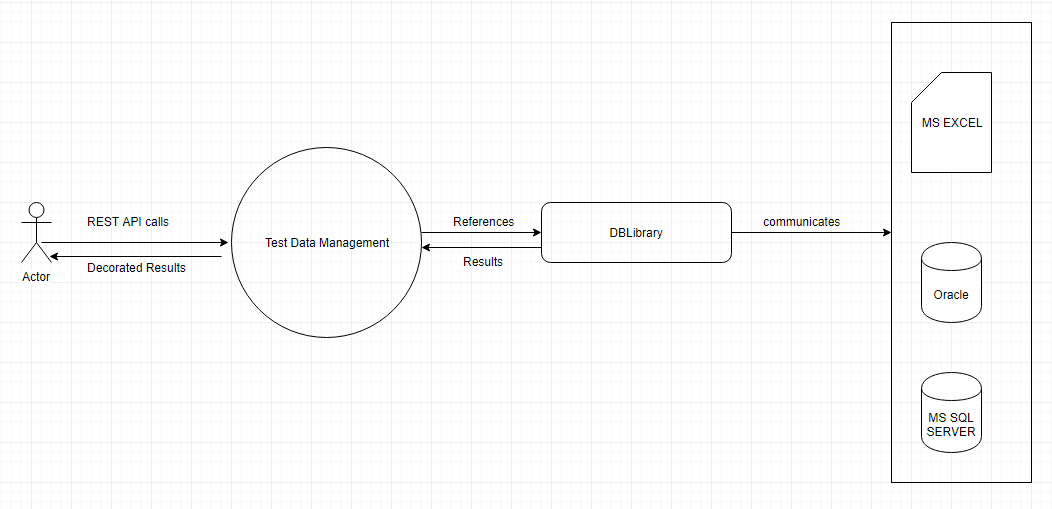
## Intended Audience

The intended audience of this document is Test Automator, Test Executor, Test Developer, DB architect and other personnel involved in the delivery of the **<ClientProject>** and related interfaces at **<ClientName>**.

# Technical Details

The purpose of Test Data Management project (TDM) , using DBLibrary, is to provide REST API to

* Retrieve the results by posting raw SQL queries and Stored Procedures.
* Fetch the results in custom business object types by passing parameters.
* Provide list of available query options that can be executed in databases.
* Retrieve the results in key-value pairs by executing the available query options, with or without passing parameters.
* Fetch the results in custom business object types from Excel documents.

DBLibrary is built with the purpose of abstracting the low level interactions with multiple databases, hiding the low level details of constucting database commands, and reading the results from execution of database commands.

Thus, DBLibrary is the abstraction layer to interact with databases and Excel application, where as Test Data Management project, using DBLibrary, provides REST API to get key-value results or business object types.

# Dependencies

DBLibrary is built with C# .NET framework with below dependencies:

* **ExcelMapper:** To interact with Excel Document.
* **Oracle.ManagedDataAccess.Core:** To interact with Oracle database.
* A screenshot of a cell phone

  Description automatically generated**System.Data.SqlClient:** To interact with MS SQL Server database.

You must install **ExcelMapper** and may require **Oracle.ManagedDataAccess.Core** and **System.Data.SqlClient** to use DBLibrary within your application.

# Project Setup

## Adding Reference

You need to add reference to **DBLibrary.dll** file in your project

## Connection String

You may pass database connection string using **appsettings.json** if you’re using .NET CORE app, or you may build API that accepts connection string as **HEADER** in JSON request.

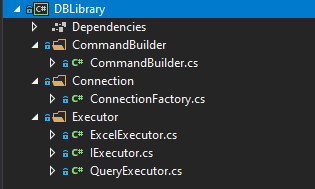
## Dictionary to Object Converter.

You need to use **DictionaryConverter.cs** to convert Dictionary values from database to custom business objects.



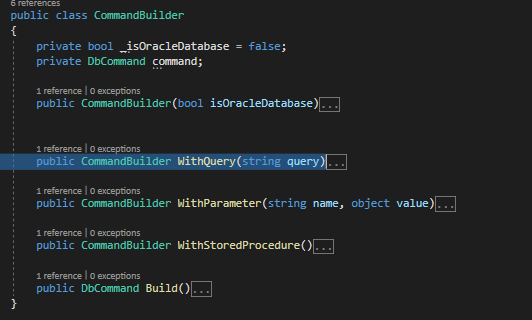
# DBLibrary

DBLibrary interacts only with MS SQL Server ,Oracle Databases, and Excel application.

This section explains the folder structure and usage details of DBLibrary.

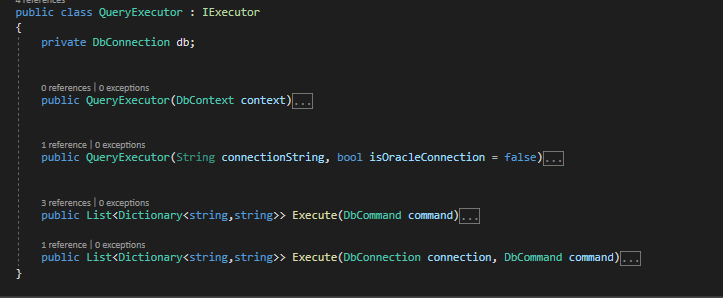
## CommandBuilder

**CommandBuilder**: Builds raw SQL commands and stored procedures,with or without parameters by taking boolean as a contructor parameter to indicate if it’s for Oracle.

Methods:

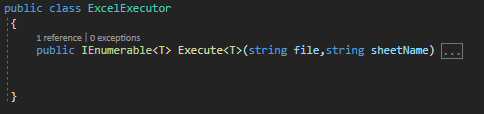
CommandBuilder is built with the fact that we need to create an instance for each SQL command or Strored Procedure that we wish to build. You get the DBCommand by calling Build() after constructing complete command using other methods exposed by CommandBuilder.

## Query Executor

**QueryExecutor:**  As the name implies, it executes Command and gives the result in key-value pairs.

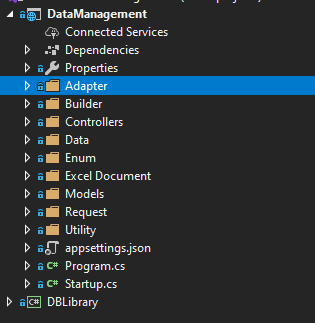
Create an instance of QueryExecutor with ConnectionString and boolean to indicate if it’s Oracle connection. You can use the same instance to execute further requests.

## Excel Executor

ExcelExector provides single method that takes file name along with its absolute location, sheet name, and then Object type to convert resutls into custom object type.

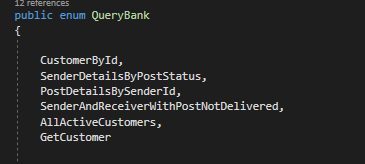
# Test Data Management

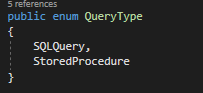
Test Data Managemet provides REST API’s that accept SQL Queries, Stored Procedures, Parameters, list of query options and gives result in key-value pairs or custom object types.

This section explains the folder structure and usage details of Test Data Management project.

## Enums

Enum contains the list of possible options that this project provides to end user.

**QueryBank:** List of query options that user can opt to execute; therefore, hiding the query or stored procedure details from user.

**QueryType:** List of query type that user can select to execute.

## CustomBuilders

This is purely a business oriented object where we use CommandBuilder from DBLibrary.dll to create a database command using step by step process. It stores all the business artificats and default parameters. Output is a database command that we wish to execute.

## Adapters

Adapter is built with the purpose of below steps:

* Create a DBCommand by calling CustomBuilder.
* Execute DBommand using QueryExecutor from DBLibrary.dll.
* Parse the result from QueryExecutor and convert using DictionaryToObject converter.
* Fetch Results from ExcelExecutor using custom business object type.



The above illustrates the sample usage where we build instance of QueryExecutor and call its Execute() to get results.

## Controllers

Controllers accepts the request from enduser and redirects to Adapters for further requests.Controllers can call the methods from Adapters based of the response that user wish to have.

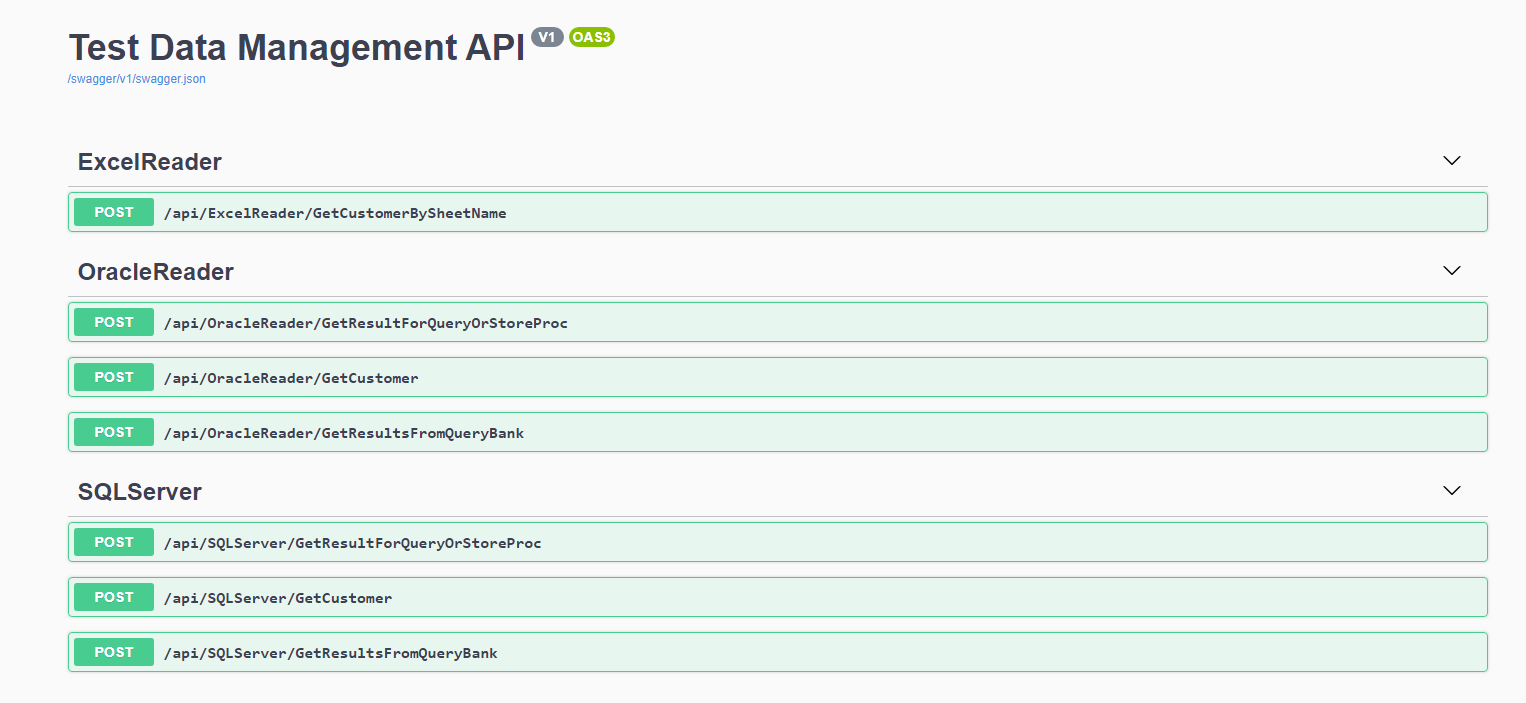
* GetDictionary()
* GetViewModel()

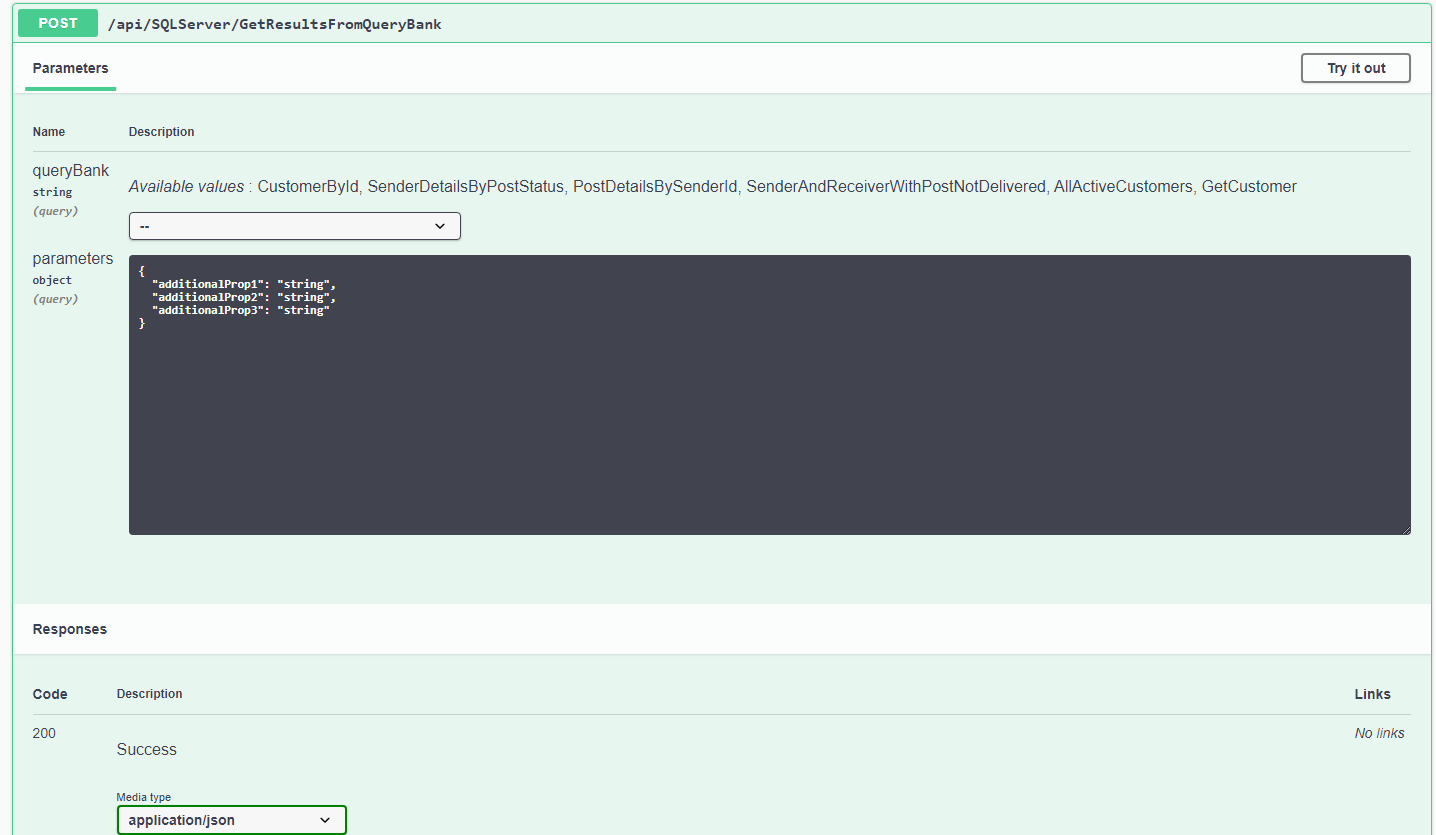
# Product Information

## Swagger Document

Test Data Management project is configured with Swagger to generate documents. The project is pushed to RevIT CICD server and live.

### 8.1.a. How to use Swagger Document

Navigate to URL <http://localhost:port/swagger/index.html>

Click on “Try it out” button for any of the API call that you wish to execute