Motor Sizing Software

Developer Manual: PBA System

Version 4

Topra World Pte Ltd

8 Eu Tong Sen Street

#14-94

The Central

Singapore 059818

**IMIT (Pvt) Ltd**

**No 82, Galle Road , Colombo 04, Sri Lanka**

**Phone: (+94)719927926 Fax: (+94)115926926**

**Web :www.imit.lk Contact: info@imit.lk**

Contents

[Motor Sizing Software 1](file:///C:\Users\lakmal\Desktop\PBA_Motor_Sizer_8_8_\DeveloperManual_MotorSizingSoftware.docx#_Toc85388241)

[Developer Manual: PBA System 1](file:///C:\Users\lakmal\Desktop\PBA_Motor_Sizer_8_8_\DeveloperManual_MotorSizingSoftware.docx#_Toc85388242)

[Revision History 3](#_Toc85388243)

[High-level Architecture 4](#_Toc85388244)

[Entity diagram 4](#_Toc85388245)

[Tools and Technology 5](#_Toc85388246)

[Data Model 7](#_Toc85388247)

[Relationships 7](#_Toc85388248)

[Special columns 8](#_Toc85388249)

[Code 10](#_Toc85388250)

[Solution Overview 10](#_Toc85388251)

[Motorsizer.Client 10](#_Toc85388252)

[Motorsizer.Core 12](#_Toc85388253)

[MotorSizer.Entities 12](#_Toc85388254)

[MotroSizer.Web 12](#_Toc85388255)

[Application Deployment 13](#_Toc85388256)

[Step 1 - Build the Solution 13](#_Toc85388257)

[Step 2 - Make sure the deployment project properties. 13](#_Toc85388258)

[Step 3 - Make sure the deployment project file system. 14](#_Toc85388259)

[Step 4 - Build the deployment project 14](#_Toc85388260)

[Other Important Task 15](#_Toc85388261)

[How to Set the Server Base URL 15](#_Toc85388262)

[How to Set the Connection String to Server Database 15](#_Toc85388263)

Revision History

|  |  |  |
| --- | --- | --- |
| **Name** | **Date** | **Remarks** |
| Lakmal | 12th September 2021 | Initial Document |
| Lakmal | 17th October 2021 | Added the code guide |
| Lakmal | 28th October 2021 | Added the content to install the setup projet and publish the web project |

# High-level Architecture

The application architecture will be multilayer architecture. The application will consist of multiple layers, as mentioned below. Also, communication between each layer will be done through DTO objects. This architecture will help the PBA team to expand the application wherever necessarily with minimum effort

## Entity diagram

Transaction

These entities will be used to acquire the user inputs and calculate the axis requirements to select the correct motor and Servo drive

Project

Profile

Axis

\*

1

11

\*

1

1

1

1

Selected Drive

Selected Motor

* One Project can have multiple Axis
* One Axis can have multiple Profiles
* One Axis can have one selected motor and selected Drive

## Tools and Technology

|  |  |  |
| --- | --- | --- |
| Development tool | Visual Studio 2019 | <https://visualstudio.microsoft.com/> |
| Server Database | MySql 8.0.25 | <https://www.mysql.com/> |
| Report Engine | Microsoft RDLC Report Designer | Download using nugget |
| Graph Engine | ScottPlot | Download using nugget |
| Design pattern | Model–view–viewmodel (MVVM) |  |
| Server Side development | ASP.net, C# |  |
| Client Side Development | WPF, C# |  |
| Target web server | IIS |  |
| JSON converter | Newtonsoft | Download using nugget |

### Nugget Packages

So far, all the third-party packages have been downloaded through the nugget

|  |  |
| --- | --- |
| Package | Version |
| Antlr | {3.5.0.2} |
| bootstrap | {3.4.1} |
| BouncyCastle | {1.8.5} |
| Google.Protobuf | {3.14.0} |
| jQuery | {3.4.1} |
| K4os.Compression.LZ4 | {1.1.11} |
| K4os.Compression.LZ4.Streams | {1.1.11} |
| K4os.Hash.xxHash | {1.0.6} |
| Microsoft.AspNet.Mvc | {5.2.7} |
| Microsoft.AspNet.Razor | {3.2.7} |
| Microsoft.AspNet.Web.Optimization | {1.1.3} |
| Microsoft.AspNet.WebApi | {5.2.7} |
| Microsoft.AspNet.WebApi.Client | {5.2.7} |
| Microsoft.AspNet.WebApi.Core | {5.2.7} |
| Microsoft.AspNet.WebApi.HelpPage | {5.2.7} |
| Microsoft.AspNet.WebApi.WebHost | {5.2.7} |
| Microsoft.AspNet.WebPages | {3.2.7} |
| Microsoft.CodeDom.Providers.DotNetCompilerPlatform | {3.6.0} |
| Microsoft.Web.Infrastructure | {1.0.0.0} |
| Modernizr | {2.8.3} |
| MySql.Data | {8.0.26} |
| Newtonsoft.Json | {12.0.2} |
| System.Buffers | {4.5.1} |
| System.Memory | {4.5.3} |
| System.Numerics.Vectors | {4.4.0} |
| System.Runtime.CompilerServices.Unsafe | {4.5.2} |
| WebGrease | {1.6.0} |
| EntityFramework | {6.4.4} |
| Newtonsoft.Json | {13.0.1} |
| Stub.System.Data.SQLite.Core.NetFramework | {1.0.113.3} |
| System.Data.SQLite | {1.0.113.7} |
| System.Data.SQLite.Core | {1.0.113.7} |
| System.Data.SQLite.EF6 | {1.0.113.0} |
| System.Data.SQLite.Linq | {1.0.113.0} |
| BouncyCastle | {1.8.5} |
| Dapper | {2.0.90} |
| DotNetProjects.WpfToolkit.DataVisualization | {6.0.90} |
| EntityFramework | {6.4.4} |
| Expression.Interaction | {3.0.40218.0} |
| Google.Protobuf | {3.14.0} |
| K4os.Compression.LZ4 | {1.1.11} |
| K4os.Compression.LZ4.Streams | {1.1.11} |
| K4os.Hash.xxHash | {1.0.6} |
| Microsoft.AspNet.WebApi.Client | {5.2.7} |
| Microsoft.ReportingServices.ReportViewerControl.Winforms | {150.1449.0} |
| Microsoft.ReportViewer.2015.Runtime | {12.0.2.2402} |
| Microsoft.ReportViewer.Common | {10.0.40219.1} |
| Microsoft.ReportViewer.VS2015.WinForms | {12.0.2.2402} |
| Microsoft.ReportViewer.WinForms | {10.0.40219.1} |
| Microsoft.SqlServer.Types | {14.0.314.76} |
| MySql.Data | {8.0.25} |
| Newtonsoft.Json | {13.0.1} |
| ScottPlot | {4.1.16} |
| ScottPlot.WPF | {4.1.16} |
| Stub.System.Data.SQLite.Core.NetFramework | {1.0.115.0} |
| System.Buffers | {4.5.1} |
| System.Configuration.ConfigurationManager | {5.0.0} |
| System.Drawing.Common | {5.0.0} |
| System.Memory | {4.5.3} |
| System.Numerics.Vectors | {4.4.0} |
| System.Runtime.CompilerServices.Unsafe | {4.5.2} |
| System.Runtime.InteropServices.RuntimeInformation | {4.3.0} |
| System.Security.AccessControl | {5.0.0} |
| System.Security.Permissions | {5.0.0} |
| System.Security.Principal.Windows | {5.0.0} |
| System.ValueTuple | {4.5.0} |

# Data Model

Data Model consists of the database that uses for the PBA Motors sizer application.

Graphical user interface, text, application

Description automatically generated

## Relationships

| Parent Table Name | PrimaryKey Column | Child Table Name | ForignKey Column |
| --- | --- | --- | --- |
| motorcategory | Id | motor | MotorCategroyId |
| motor | Id | motorparameters | MotorId |

## Special columns

This section explains different tables and columns introduced for the technical controls apart from the primary data columns.

### MotorCategory

|  |  |  |
| --- | --- | --- |
| Column Name | Values | Usage |
| MotorCategoryType | L- Linear  R – Rotary | Define whether the motor category is Linear or rotary, which will be referenced in the sizer screen to identify the same |
| ShowStroke | 1 – Show Stroke  0 – Do Not Show Stroke | Show stroke defines whether to show the Stroke field under the selected motor details section.  If the showstroke is 1 then coil length and attractive force will be hidden  If the showstroke is 0 then the Stroke will be hidden |
| MaxCoilTempConstant | Numeric value | Maximum coil temperature specification (), which used in coil temperature calculation |
| ShowSafetFactor | 1 – Show Safety Factor  0 – Do Not Show Safety Factor | Decide to whether to show the General Safety factor or other Safety factors in the Calculated Motor Values for the application section  If ShowSafetFactor = 1 then  • Below fields will be hidden  o Safety Factor Current  o Loading Factor Current  o Safety Factor Force  o Loading Factor Force  • Safety Factor will be visible  If ShowSafetFactor = 1 then  Do not Show Safety Factor  • Safety Factor will be Hidden  • Below fields will be Visible  o Safety Factor Current  o Loading Factor Current  o Safety Factor Force  o Loading Factor Force |
| MotorSelectionBase | C- Current Base  F- Force Base | **Current** – Consider the required current and motor current to retrieve the recommended motors  **Force** - Consider the required force and motor force to retrieve the recommended motors |
| DefaultLoweSafetyMargin  &  DefaultHighSafetyMargin | Numeric value | Provides the Default values for Safety Margin |
| Servo Drive Selection Base | 0 - Access to user master prohibited  1 - Can access user master | These values will decide the type of servo drive to be considered in motor sizer selection. The same field appears in the Servo driver master as well. |

### Motor

| Column Name | Usage |
| --- | --- |
| MotorSeries | Motor series is a grouping of a motor within the Motor category. This reflects during recommended motor selection, which lists a maximum of two records per motor series. |

### MotorParameters

Motor parameters are special parameters that can be used for Motor specific reasons. Users cannot add new parameters through the application. However, the user can edit the Configuration values (Numeric ) and Values power

|  |  |
| --- | --- |
| Column Name | Usage |
| MotorParameter | This is a developer-defined value as this is expected to be used in the code. |
| CalculationValue  &  CalculationValuePower | Users can edit the Configuration values (Numeric ), and Values power can set the big numbers. Currently, we use this for PIX250 & PIX150 motors  Ex: if required config value is 3x10⁸ then the value can be set as below   |  |  | | --- | --- | | **Config Value** | **Value Power** | | **3** | **8** | |
| ParameterGroup | If multiple parameters are required for one function, then this can be used to group the multiple parameters |
| ParamOrder | If there are multiple parameters under a parameter group and those need to be used in a particular order, then this can be used for such requirements |

### ServoDrive

| Column Name | Usage |
| --- | --- |
| DriveBase | Based on the selected motor category, these values will decide the type of servo drive to consider in motor sizer selection. The same field appears in the MotorCategory table as well. |

### UserMaster

| Column Name | Values | Usage |
| --- | --- | --- |
| CanAccessUserMaster | 0 - Access to user master prohibited  1 - Can access user master | If this access allows, the relevant user can access the user master and perform the basic system administration tasks, including resetting the password. |

# Code

This section will explain the usage of different projects folders and special classes under each project.

## Solution Overview

The solution consists of both the server and the client-side of the program and six projects.

Text

Description automatically generated

### Projects

|  |  |
| --- | --- |
| Project Name | Usage |
| Motorsizer.Client | This is the main (Startup) Project for the client-side. This project holds all the user interfaces and view models for the client-side executions |
| Motorsizer.Core | Core project holds the calculations for motor sizer. This runs only on client. |
| Motorsizer.Entities | This is a class with the number of Entity classes. An Entity class provide the data structure for all the data elements used within the solution. Entities are used to communicate between the different projects and layers. Therefore, this project referred in client and server both |
| MotorSizer.SecServices | This has security-related implementations, Encryption, Decryption and other security services within the application. This project is also referred to both client and server sides. |
| MotorSizer.Web | This is the main (Startup) Project for the server-side. In addition, this is a Web API project. |

## Motorsizer.Client

### MainWindow

The main window is the initial window for the project. This holds a tab control that has access to all the interfaces. The relevant ViewModel also do the preliminary checks and load the application configurations.

### Common

The common folder holds all the view models used to manipulate data. This has the ViewModels that have similar to entities in the Entities project. Further, this keeps the classes for event arguments as well. Each ViewModel has its methods. However, there are methods with the same name to do similar functionality.

1. To<EntityName>Entity(): This method will copy the values from ViewModel to a supplied or new entity instance
2. Clone(): Cline method will create a new instance with the same vie models property values

#### ValueManipulations

Valuemanipulations is a class that uses to convert from ViewModels to entities and entities view models. There are a set of methods create for every entity/ ViewModel.

1. To<Viewmodel> - This will copy the values from the selected entity to the supplied or new ViewModel
2. To<Viewmodel>Colection - This will copy the values from the selected entity array to the supplied or new ViewModel observable collection

### Controls

Controls is the folder that keeps different controls used for the application that had to be modified as per the PBA requirement.

|  |  |
| --- | --- |
| Controls | Usage |
| BindablePasswordBox.xaml | This is the password control that can bind the password to a property |
| DirectionArrow.xaml | This is the arrow control that uses to show the motion direction |
| FocusAdvancement.cs | This class is the central class to create the tab movement for the Enter button |
| LineChart.xaml | The view and implementation of the Scott plotter line graph |
| MainReport.rdlc | The report control for the sizer report |
| ProgressBar.xaml | This is the file download from server implementation with progress bar |
| RangeSlider.xaml | The range control with double axises |

### Converters

This folder holds the converters that are used to convert the values from XAML files.

### Masters

This is the folder that keeps the interfaces and client-side functionality for masters. This master will be visible only for admin users.

### Services

The service folder represents the service layer. All the data required for the application access via these classes. These classes do not hold any properties. Every class consists of several methods, which access the local data source and server data source via web API.

### Sizer

Sizer has the sizer interface and ViewModel. Also, this has one XAML, which uses for multi-profile edit screen.

### Styles

All the default styles for the client-side interfaces.

## Motorsizer.Core

Core project is the project for the calculations. This project provides all of the calculations. Mainly, this has two classes to do the axis level and profile level calculations.

### Params

Params has few classes that use as parameters for the calculations

## MotorSizer.Entities

This is the Entity layer with the number of Entity classes. An Entity class provide the data structure for all the data elements used within the solution. Entities are used to communicate between the different projects and layers. Therefore, this project referred in client and server both

### SyncMasters.cs

SyncMaster keeps the arrays of master main master entities. This class uses to generate the data files for the local database. The data file was generated by encrypting this class instance JSON object.

## MotroSizer.Web

This is the main (Startup) Project for the server-side. In addition, this is a Web API project. The documentation for the API will be provided in a separate document. This is a simple API without any security implementation.

### Admin

Admin is a folder that comes under the Web project. This folder has some admin related implementation that runs on the server.

### AutoDownload

This folder has the downloadable data files (data.Pbadata) generated with every version generation. This doesn’t directly under the web project. Therefore this folder will not generate during the API publish.

### Data

The data folder represents the data access layer. This has one class (MySQLDataAccess.cs), which uses to access the MySQLdatabase.

# Application Deployment

This section doesn’t discuss the web application deployment since there are no particular tasks apart from basic deployment procedures.

How to create an IIS web site can be access through below link

<https://www.ssl.com/how-to/create-new-website-iis-10/>

Once web project is published, content of the published folder need to copy to the IIS folder.

However, this discusses the process to create a new deployment package. PBA Motor sizer uses Microsoft setup project (PBASystemSetup) to generate the deployment package

The guide to install the deployment can be access through below link.

<https://docs.microsoft.com/en-us/cpp/windows/walkthrough-deploying-a-visual-cpp-application-by-using-a-setup-project?view=msvc-160&WT.mc_id=linkedin>

Steps to create a new deployment is explained below

## Step 1 - Build the Solution

Set the new version under Client project 🡪 Properties 🡪 Assmblyinfo.cs

A screenshot of a computer

Description automatically generated with medium confidence

Build the solution with debug mode and then build again with release mode.

## Step 2 - Make sure the deployment project properties.

Make sure that deployment project properties are below. In addition, the application product name and version should reflect the exact version set in the previous step.

Graphical user interface

Description automatically generated with medium confidence

## Step 3 - Make sure the deployment project file system.

Right-click on the deployment project and make sure all of the file systems are correctly done. If there’s a report change, the report file (**MainReport.rdlc**) under the control folder must be replaced. Also, **data.Pbadata** file needs to be replaced with latest data file

Graphical user interface, text, application, chat or text message

Description automatically generated

## Step 4 - Build the deployment project

Build the deployment project with debug mode and then release mode

Once the above steps are done, the deployment package should be generated under the release folder in the deploymemnt project, which can be distributed among the users.

# Other Important Task

## How to Set the Server Base URL

This explains that how the server base URL is set.

The server URL is set under the App.xaml.cs. GlobalVariales is a class created to hold the global variables, and it has the HttpClient (client) property, which can be referred from anywhere in the application.

A screenshot of a computer

Description automatically generated with medium confidence

## How to Set the Connection String to Server Database

PBA system web project uses my SQL to store the master data. And below explains how the connection string is set.

The connection string is set under the “web.config” file under the web project. Please note this can be updated once the application is deployed as well.

Text

Description automatically generated