```
static void Main()
        {
            Application.EnableVisualStyles();
            Application.DoEvents();
            Application.Run(new MainForm());
        }

        private void startButton_Click(object sender, System.EventArgs e)
        {
            if (runningTask == null)
            {
                try
                {
                    stopButton.Enabled = true;
                    startButton.Enabled = false;

                    // Create a new task
                    myTask = new Task();

                    // Create a virtual channel
                    myTask.AIChannels.CreateVoltageChannel(physicalChannelComboBox.Text, "",
                        (AITerminalConfiguration)(-1), Convert.ToDouble(minimumValueNumeric.Value),
                        Convert.ToDouble(maximumValueNumeric.Value), AIVoltageUnits.Volts);

                    // Configure the timing parameters
                    myTask.Timing.ConfigureSampleClock("", Convert.ToDouble(rateNumeric.Value),
                        SampleClockActiveEdge.Rising, SampleQuantityMode.ContinuousSamples, 1000);

                    // Verify the Task
                    myTask.Control(TaskAction.Verify);

                    // Prepare the table for Data
                    InitializeDataTable(myTask.AIChannels, ref dataTable);
                    acquisitionDataGrid.DataSource = dataTable;

                    runningTask = myTask;
                    analogInReader = new AnalogMultiChannelReader(myTask.Stream);
                    analogCallback = new AsyncCallback(AnalogInCallback);

                    // Use SynchronizeCallbacks to specify that the object
                    // marshals callbacks across threads appropriately.
                    analogInReader.SynchronizeCallbacks = true;
                    analogInReader.BeginReadWaveform(Convert.ToInt32(samplesPerChannelNumeric.Value),
                        analogCallback, myTask);
                }
                catch (DaqException exception)
                {
                    // Display Errors
                    MessageBox.Show(exception.Message);
                    runningTask = null;
                    myTask.Dispose();
                    stopButton.Enabled = false;
                    startButton.Enabled = true;
                }
            }
        }

        private void AnalogInCallback(IAsyncResult ar)
        {
            try
            {
                if (runningTask != null && runningTask == ar.AsyncState)
                {
                    // Read the available data from the channels
                    data = analogInReader.EndReadWaveform(ar);

                    // Plot your data here
                    dataToDataTable(data, ref dataTable);

                    analogInReader.BeginMemoryOptimizedReadWaveform(Convert.ToInt32(samplesPerChannelNumeric.Value),
                        analogCallback, myTask, data);
                }
            }
            catch (DaqException exception)
            {
                // Display Errors
                MessageBox.Show(exception.Message);
                runningTask = null;
                myTask.Dispose();
                stopButton.Enabled = false;
                startButton.Enabled = true;
            }
        }

        private void stopButton_Click(object sender, System.EventArgs e)
        {
            if (runningTask != null)
            {
                // Dispose of the task
                runningTask = null;
                myTask.Dispose();
                stopButton.Enabled = false;
                startButton.Enabled = true;
            }
        }

        private void dataToDataTable(AnalogWaveform<double>[] sourceArray, ref DataTable dataTable)
        {
            // Iterate over channels
            int currentLineIndex = 0;
            foreach (AnalogWaveform<double> waveform in sourceArray)
            {
                for (int sample = 0; sample < waveform.Samples.Count; ++sample)
                {
                    if (sample == 10)
                        break;

                    dataTable.Rows[sample][currentLineIndex] = waveform.Samples[sample].Value;
                }
                currentLineIndex++;
            }
        }

        public void InitializeDataTable(AIChannelCollection channelCollection, ref DataTable data)
        {
            int numOfChannels = channelCollection.Count;
            data.Rows.Clear();
            data.Columns.Clear();
            dataColumn = new DataColumn[numOfChannels];
            int numOfRows = 10;

            for (int currentChannelIndex = 0; currentChannelIndex < numOfChannels; currentChannelIndex++)
            {
                dataColumn[currentChannelIndex] = new DataColumn();
                dataColumn[currentChannelIndex].DataType = typeof(double);
                dataColumn[currentChannelIndex].ColumnName = channelCollection[currentChannelIndex].PhysicalName;
            }

            data.Columns.AddRange(dataColumn);

            for (int currentDataIndex = 0; currentDataIndex < numOfRows; currentDataIndex++)
            {
                object[] rowArr = new object[numOfChannels];
                data.Rows.Add(rowArr);
            }
        }
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data;
using System.Data.SqlClient;

namespace Data
{
    public class DatabaseAdgang
    {
        private SqlConnection conn;
        private const String DB = "F15ST2ITS2201404492";
        private DateTime Datostempel;
        private int GemtId = 0;
        public DatabaseAdgang()
        {
            conn = new SqlConnection("Data Source=webhotel10.iha.dk;Initial Catalog=" + DB + ";Persist Security Info=True;User ID=" + DB + ";Password=" + DB + "");
        }
            public int gemData(string Forsøgsnavn, List<double> Rådata)
        {
            Datostempel = DateTime.Now;
            double[] BLOBListe = Rådata.ToArray();
            byte[] BYTEliste = Rådata.SelectMany(value => BitConverter.GetBytes(value)).ToArray();

            String query = "INSERT INTO SEMPRJ3 (Forsøgsnavn, Datostempel, Blodtryksmåling) " +
                "Output Inserted.Id " +
                "VALUES(@Forsøgsnavn, @Dato, @MåleListe) ";
            conn.Open();

            SqlCommand command = new SqlCommand(query, conn);
            command.Parameters.AddWithValue("@Forsøgsnavn", Forsøgsnavn);
            command.Parameters.Add("@Dato", SqlDbType.DateTime).Value = Datostempel;
            command.Parameters.Add("@MåleListe", SqlDbType.Image).Value = BYTEliste;
            GemtId = Convert.ToInt32(command.ExecuteScalar());

            conn.Close();
            return GemtId;
        }
    }
}
```

```xml
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>
  <appSettings>
    <add key="KalibreringsKoefficient" value="50"/>
  </appSettings>
</configuration>
```