**SpotfireDemo App**

**Documentation**

**(Windows Forms)**

**(IoT Background App)**

**(TIBCO Spotfire Visuals)**

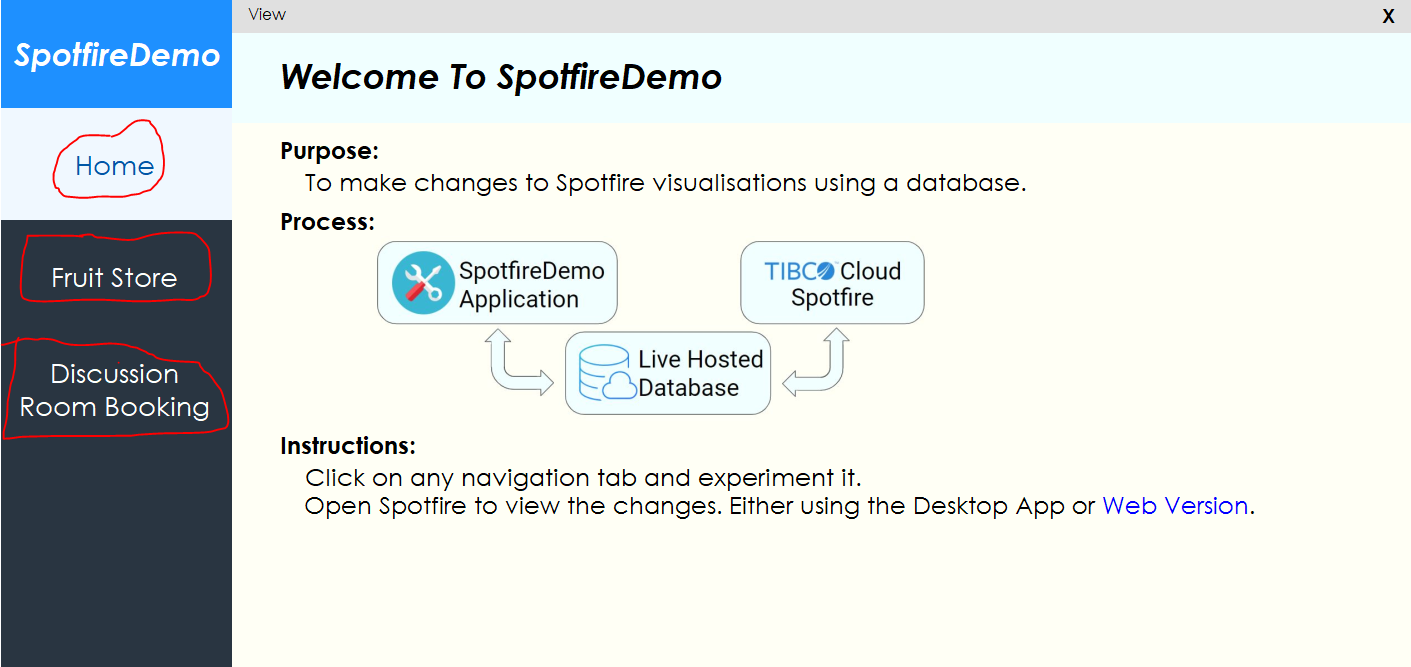
**Table Of Contents**

1. **Introduction……………………………….………….. 2**
   1. **Important Notes………………………………………...…… 2**
2. **Project Set-up…………………………………..……. 3 - 5**
   1. **Create Local Database……………………………..………. 3 - 5**
3. **Creating The Project………………………….….…. 6 - 12**
   1. **Fruit Store Demo………………………………….………… 7 - 9**
   2. **Discussion Room Booking Demo…………….………….. 10 - 12**
      1. **IoT Background Application…………….……….. 10**
      2. **Windows Forms Application……………………... 11 - 12**
4. **Alternatives And Fixes……………………………... 13 - 16**
   1. **IoT Hardware Connection…………………………………. 13**
   2. **IoT Multithreading…………………………………………… 14**
   3. **IoT Project Startup Error……………………………………. 14**
   4. **Enhance Waiting Function………………………………… 15**
   5. **TIBCO Spotfire Automatic Data Reload…………………. 16 - 18**
5. **How To Use....…………………………………………. 19 - 21**
   1. **Fruit Store Demo…………………………………………….. 20**
   2. **Discussion Room Booking Demo………………………… 21**
6. **References....…………………………………………. 22**
   1. **General……………………………………………...………. 22**
      1. **Prerequisites………………………….……………. 22**
      2. **Utilities………………………………...……………. 22**
      3. **Potential Bugs……………………………………… 22**
   2. **C# Coding………………………………………………..…. 23**
      1. **TIBCO Spotfire JavaScript API (Never used) 23**
      2. **Database…………………………..…………...….. 23**
      3. **General…………………………....………………… 23**
   3. **C# Windows Forms…………………………………..……. 24 - 25**
      1. **Sections……………………………………………... 24**
      2. **Controls……………………………...…………….... 24**
      3. **Programmatic Controls……………...………….... 25**

**1. Introduction**

With the access to Spotfire, came the idea of creating a SpotfireDemo App. The SpotfireDemo App was made to discover the possibilities of leveraging Spotfire’s visualisations through using external applications.

In the SpotfireDemo App contains navigation tabs at the side. Each tab will display a screen, creating a test environment for each demo.



**1.1 Important Notes**

The SpotfireDemo App requires the access of Spotfire Desktop App and Visual Studio. Visual Studio Enterprise 2015 was used.

The database created must be from the “SQL Server Object Explorer” view. This is because we will be testing using locally hosted databases to make SpotfireDemo and Spotfire sync with each other when a change is made to the database.

Another note is that in this project, IoT technology was also introduced. When using this IoT technology, additional requirements are needed to run the SpotfireDemo App.

IoT technology requirements are as follows:

1. Visual Studio 2015 Updates 3
2. IoT equipment (Raspberry Pi and Grove Pi)
3. IoT resources (Practical files, extensions, libraries)
4. Windows 10 operating system, enable developer mode

For item 2 and 3, they can be obtained by requesting the necessary components from your supervisor in NYP.

**2. Project Set-up**

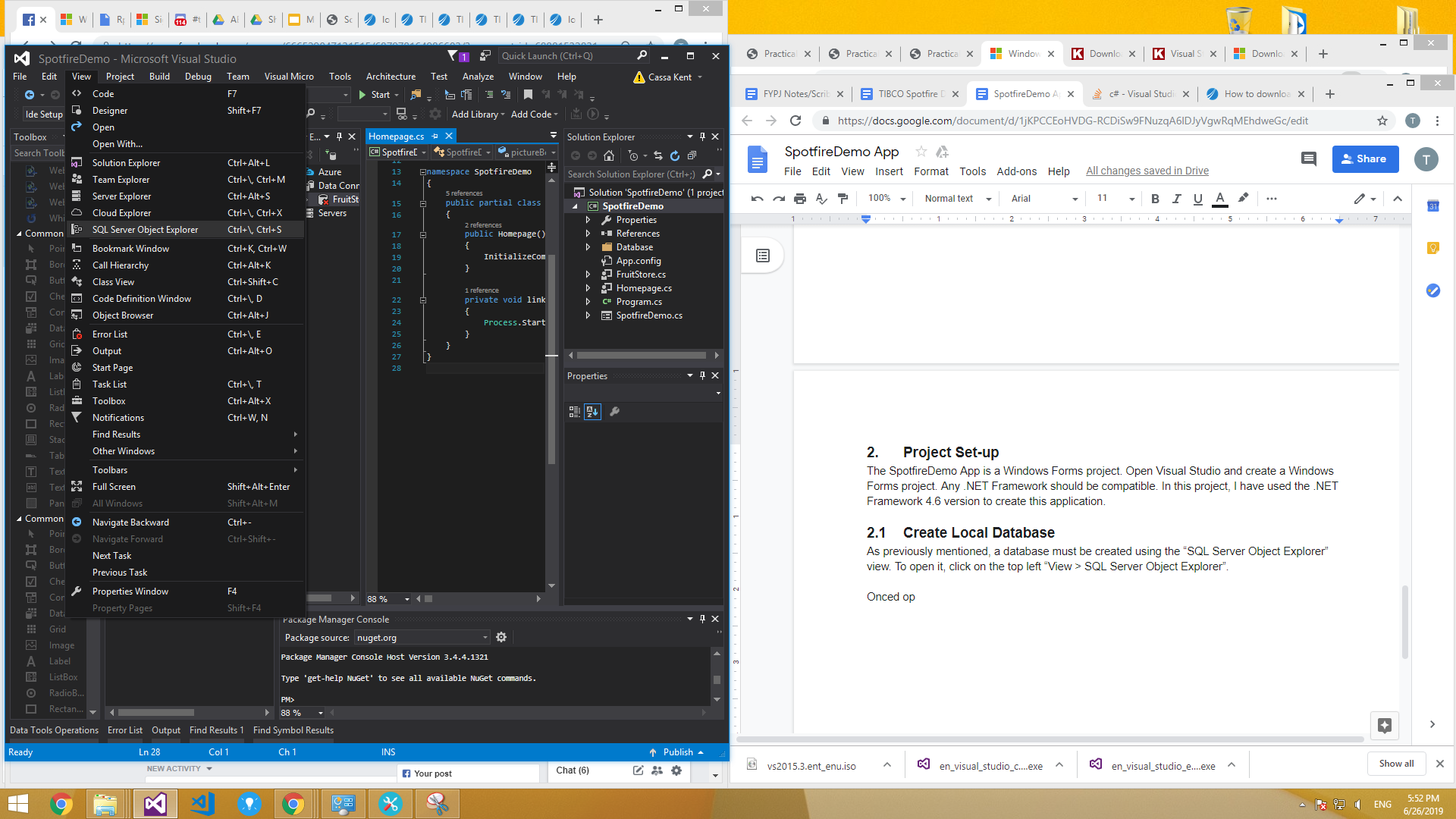
The SpotfireDemo App is a Windows Forms project. Open Visual Studio 2015 and create a Windows Forms project. Any .NET Framework should be compatible. In this project, I have used the .NET Framework 4.6 version to create this application.

If you are planning to also create an IoT project, a different setup is required. Visit the “Raspberry Pi/ Visual Studio 2015 Setup Guide” documentation to view the prerequisite steps required.

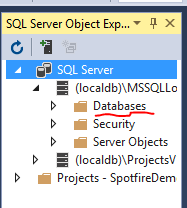
.

**2.1 Create Local Database**

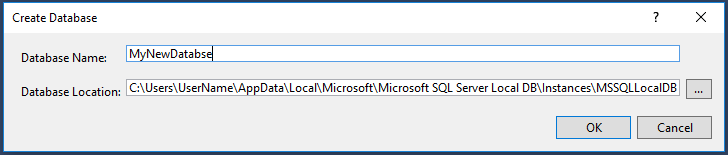
As previously mentioned, a database must be created using the “SQL Server Object Explorer” view. To open it, click on the top left “View > SQL Server Object Explorer”.



Once opened, will display a “SQL Server”. Navigate to “SQL Server > (localdb)MSSQLLocalDB > Databases”. Using (localdb)MSSQLLocalDB is preferred as it will be the default server for most users.

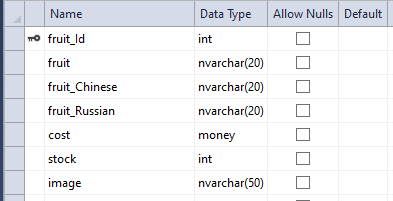


Finally, right click the “Databases” folder and “Add New Database”. Proceed to give your database name, and select a destination to save the database.



Once your local SQL database server is created, you must always use this database as it will be essential in connecting to the TIBCO Spotfire Application later on. Alternatively, if an online database is already used and is supported by TIBCO Spotfire, you can ignore this step.

As you are recreating the database, here is a snapshot of the table schema and data of my database.



**Database Table Schema**

CREATE TABLE [dbo].[Fruits] (

[fruit\_Id] INT IDENTITY (1, 1) NOT NULL,

[fruit] NVARCHAR (20) NOT NULL,

[fruit\_Chinese] NVARCHAR (20) NOT NULL,

[fruit\_Russian] NVARCHAR (20) NOT NULL,

[cost] MONEY NOT NULL,

[stock] INT NOT NULL,

[image] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([fruit\_Id] ASC)

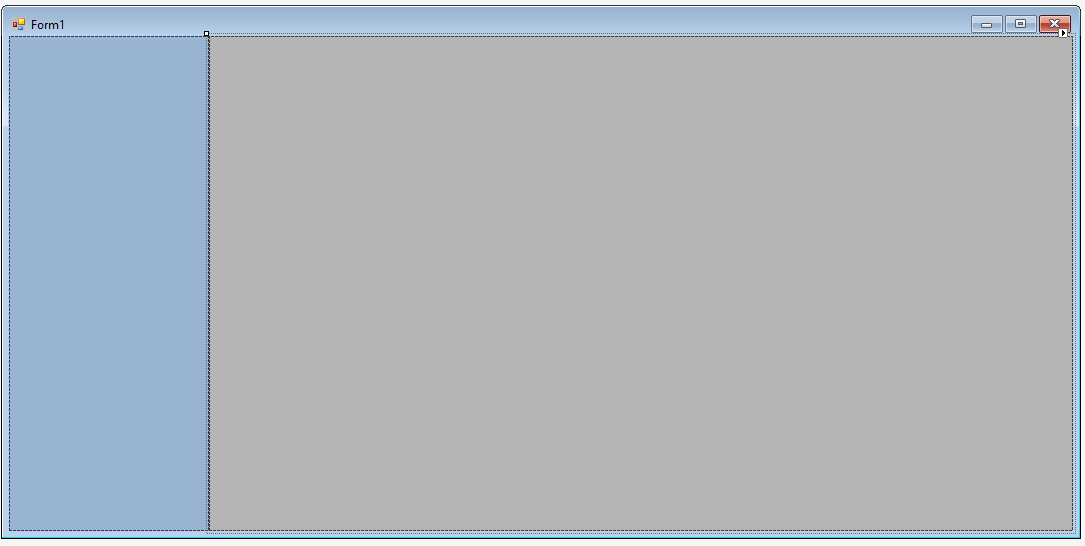
);

**Database Table Data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Apple | 苹果 | яблоко | 1.0000 | 50 | apple.png |
| 2 | Pear | 梨 | груша | 0.9000 | 40 | pear.png |
| 3 | Orange | 橙子 | апельсин | 1.0000 | 40 | orange.png |
| 4 | Kiwi | 猕猴桃 | киви | 2.0000 | 20 | kiwi.png |
| 5 | Watermelon | 西瓜 | арбуз | 10.0000 | 10 | watermelon.png |
| 6 | Dragon Fruit | 火龙果 | плод дракона | 8.0000 | 0 | dragonfruit.png |
| 7 | Banana | 香蕉 | банан | 1.0000 | 50 | banana.png |
| 8 | Lemon | 柠檬 | лимон | 0.8000 | 10 | lemon.png |
| 9 | Mango | 芒果 | манго | 3.0000 | 30 | mango.png |
| 10 | Durian | 榴莲 | дуриан | 8.0000 | 0 | durian.png |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**3. Creating The Project**

Create a windows forms project and add user interfaces to navigate through the application.



You can create a side navigation tab as shown in the picture. It will serve as an empty area for navigation buttons to be placed. The grey area will be the content. The content will change every time a navigation tab is clicked. This technique will require the use of a “User Control” item provided by VS Windows Forms.

First, add a new item “User Control”. For each “User Control” created, it will serve as a single page, and it will also serve as a single test environment for various demos. You can refer to this two links [here](https://www.codeproject.com/Questions/156146/How-to-load-UserControl-into-panel) and [here](https://stackoverflow.com/questions/10871565/how-to-make-winforms-usercontrol-fill-the-size-of-its-container) for a better reference on how to use “User Controls” as a new page.

Additionally, here are some UI ideas on how to create neat projects: <https://www.youtube.com/watch?v=tgqKd7l7_s8>

<https://www.youtube.com/watch?v=K9Ps66GoD-k>

<https://www.youtube.com/watch?v=r8s35leUR6A>

You should be able to create a simple homepage by yourself, and with these links above.

From later development of the project, I also introduced foreign languages into the demo. That is where the Chinese and Russian characters come into play.

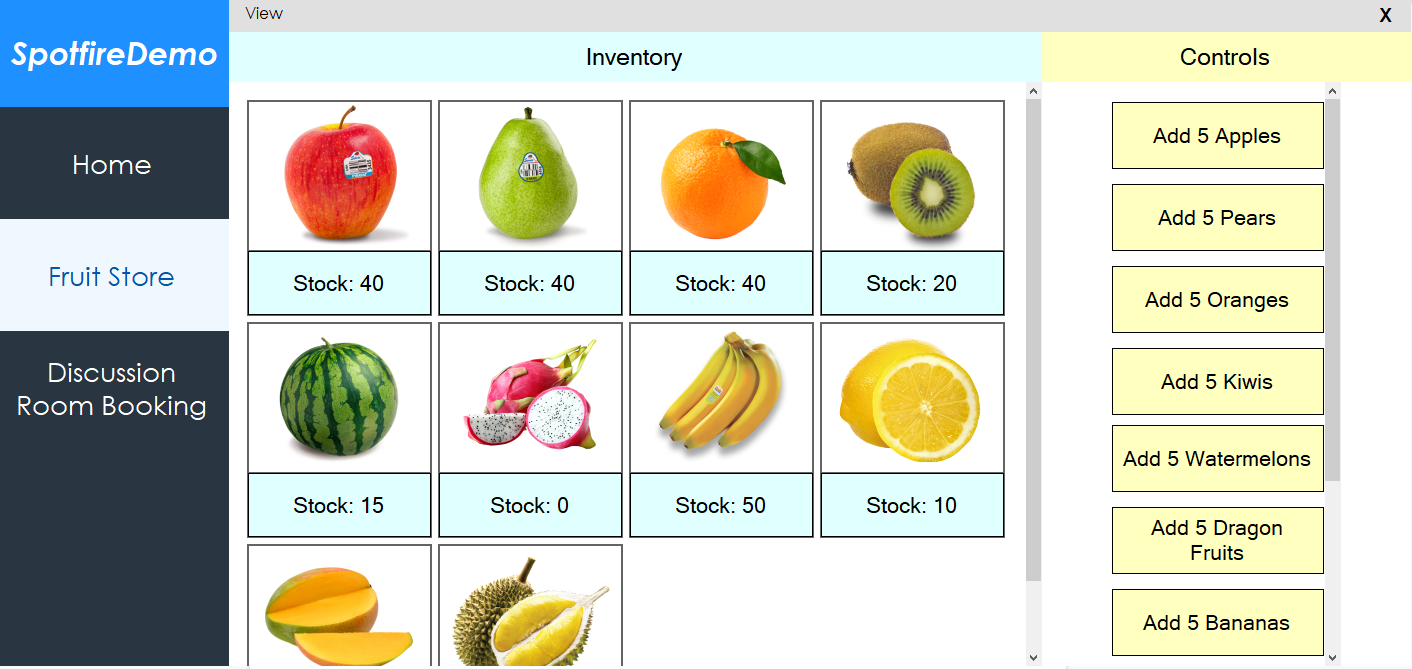
**3.1 Fruit Store Demo**

The “Fruit Store Demo” was aimed to create a simple test connection between Spotfire and a Windows Forms project. To recreate this project, first connect the previously created local SQL database server onto the same “User Control” page that will be using the database.

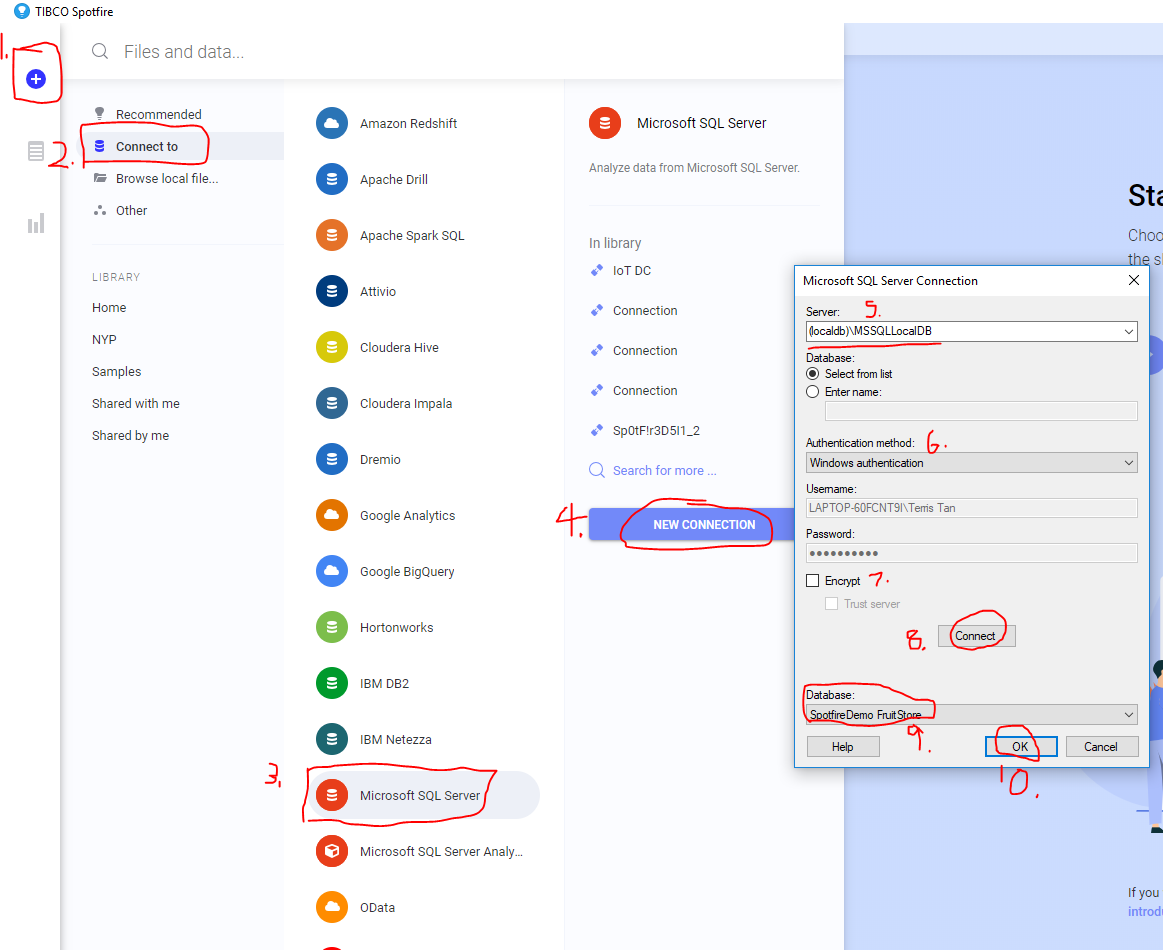


Next, add in some simple data tables and entries. It can be as simple as inventory checking.

Create the demo project normally as a normal practical lesson. By adding basic buttons and SQL commands, you should be to achieve something similar as this image.



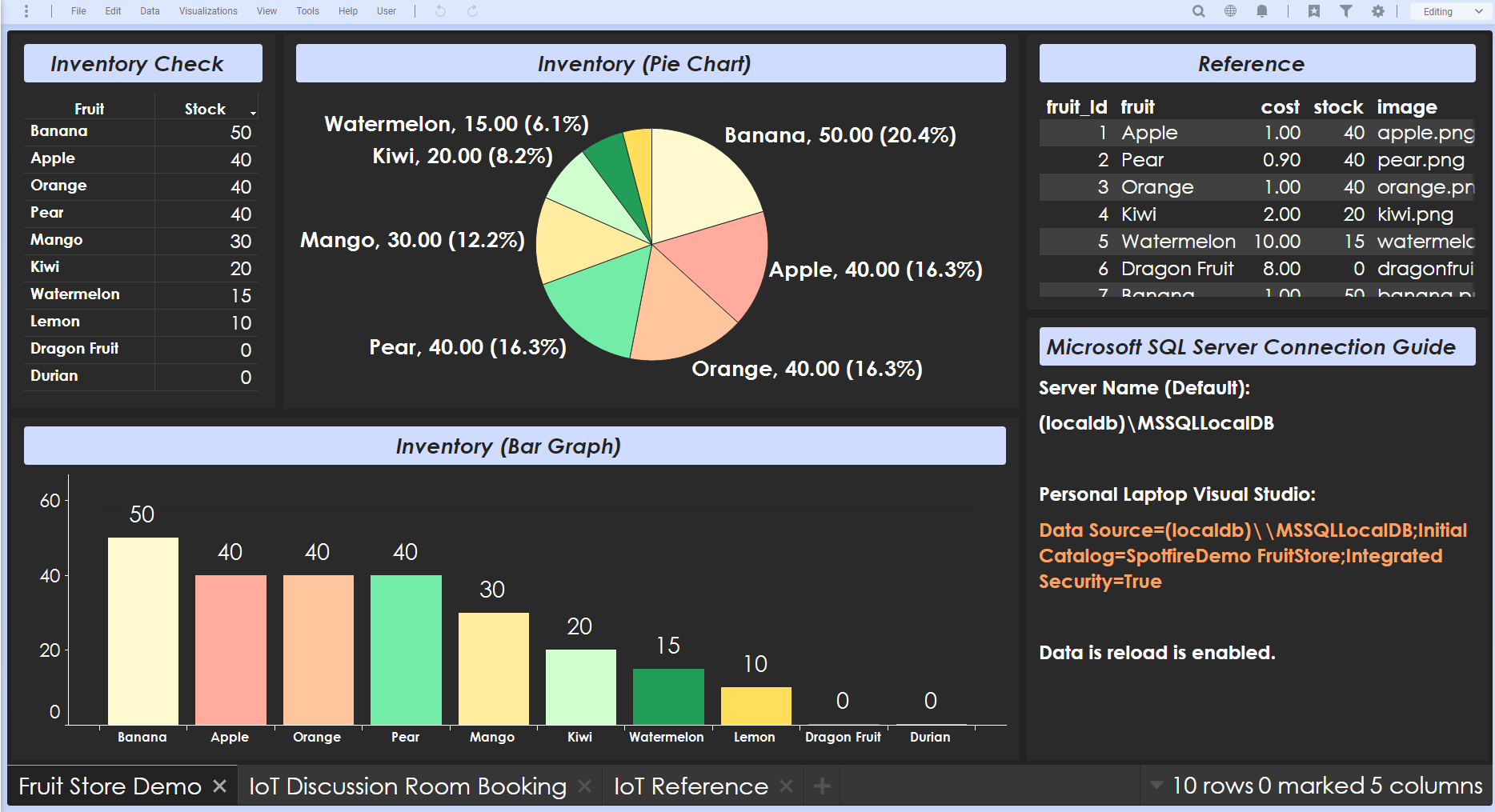
The final step is to connect this demo project into the TIBCO Spotfire Application. By opening your TIBCO Spotfire Desktop Application, making a connection to your local SQL database server.



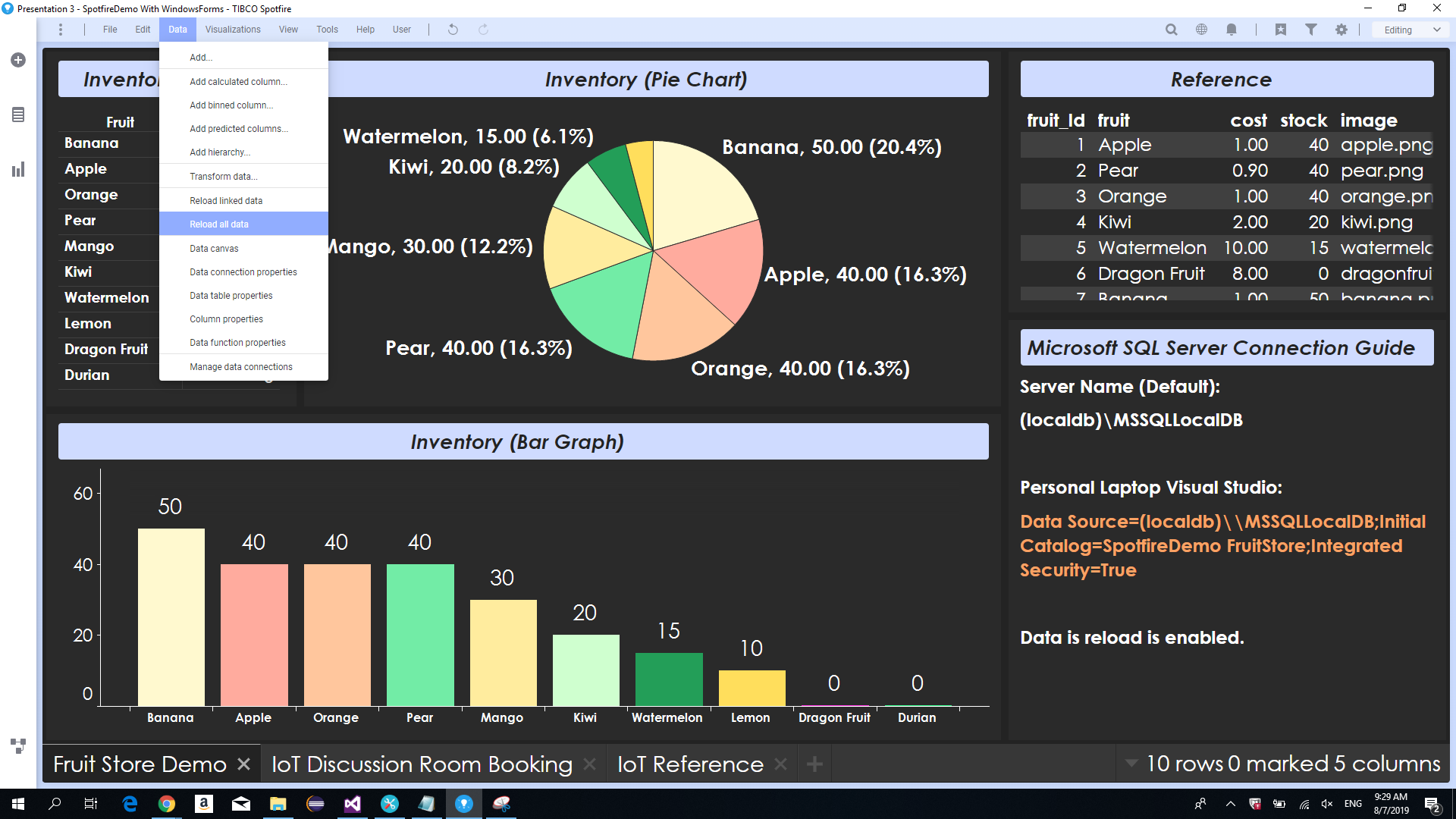
Following the steps in the image, you should be able to connect your local SQL database server onto Spotfire. Note that this database connection will only be retrievable by using the same PC as it is a local database.

Once connected, depending when you clicked “Connect to”, it will either create a new analysis with the new connection, or create a new database connection in the data canvas of the opened analysis.

Next is to create simple visualisations to test if the connection is successful. You can create a simple visualisation set as shown:



Now, test both your Windows Forms application and Spotfire visualisation to see if the data will be able to be updated and change the visualisations accordingly. To refresh the data connection in Spotfire, click on “Data > Reload all data”.



**3.2 Discussion Room Booking Demo**

The “Discussion Room Booking Demo” aims to replicate the same objective as the “Fruit Store Demo”, with an addition to IoT technology. As previously mentioned, please ensure that you have the essential items.

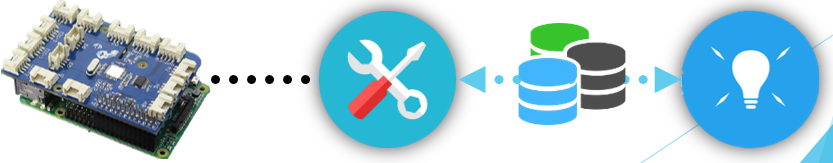
You must have the IoT hardware, the IoT practical resources, Visual Studio 2015 Updates 3, and an operating system with Windows 10 to enable developer mode.

Read the “Raspberry Pi/Visual Studio Setup Guide” on how to set up your IoT Background Application in Visual Studios.

Once the setup of the IoT is complete, we will start by making an IoT Background Application project. This section of the demo requires two different Visual Studio projects, one for IoT and the other for UI presentation.

**3.2.1 IoT Background Application**

The IoT Background Application project will send IoT sensor data to the Windows Forms project, then it will write to the database to reflect onto TIBCO Spotfire visualisations.



Inside your IoT Background Application, refer to the IoT practical resources heavily. Refer to P09 and add dataComms to the project. This “dataComms” will be the tool used to enable communication between the IoT Background Application and your Windows Forms Application.

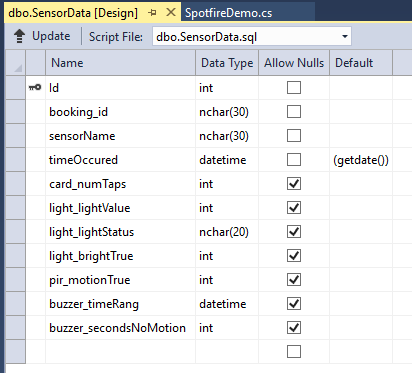
After successfully connecting them, slowly add more sensors into the project using the IoT practical resources as reference. While handling with multiple sensors, you may have to learn multithreading to avoid the bugs that comes with handling multiple sensors.

**3.2.2 Windows Forms Application**

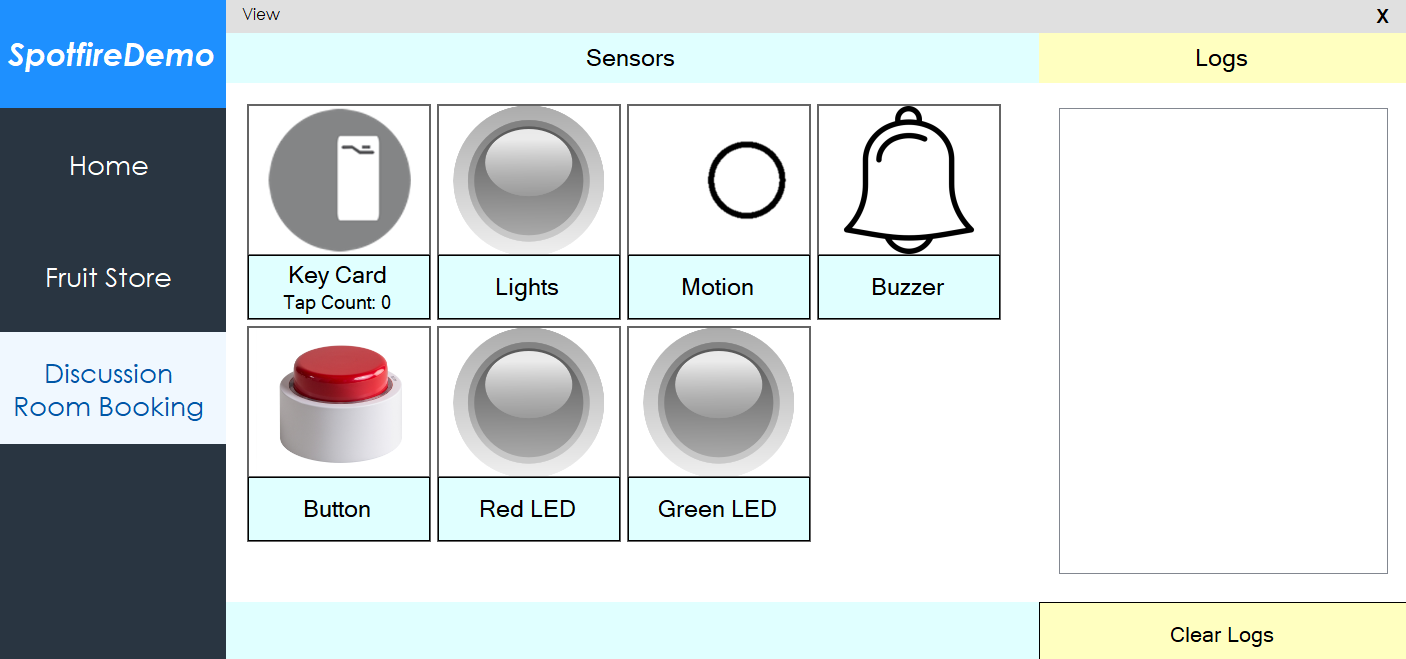
Completed with the IoT Background Application, go back to the Windows Forms Application and create a new “User Control” page. Ensure that you have also added the dataComms inside and create a new local SQL database server with SQL Server Object Explorer.

With the Windows Forms project setup finished, ensure that your database design suits the needs of TIBCO Spotfire. TIBCO Spotfire requires data sources to meet a specific set of requirements in order to have an easier experience creating visualisations.

From this article [here](https://techdifferences.com/difference-between-normalization-and-denormalization.html), creating a denormalized database design will help provide an easier time when making Spotfire visualisations. Below is a screenshot of my database design. You may modify changes to make a better database design.



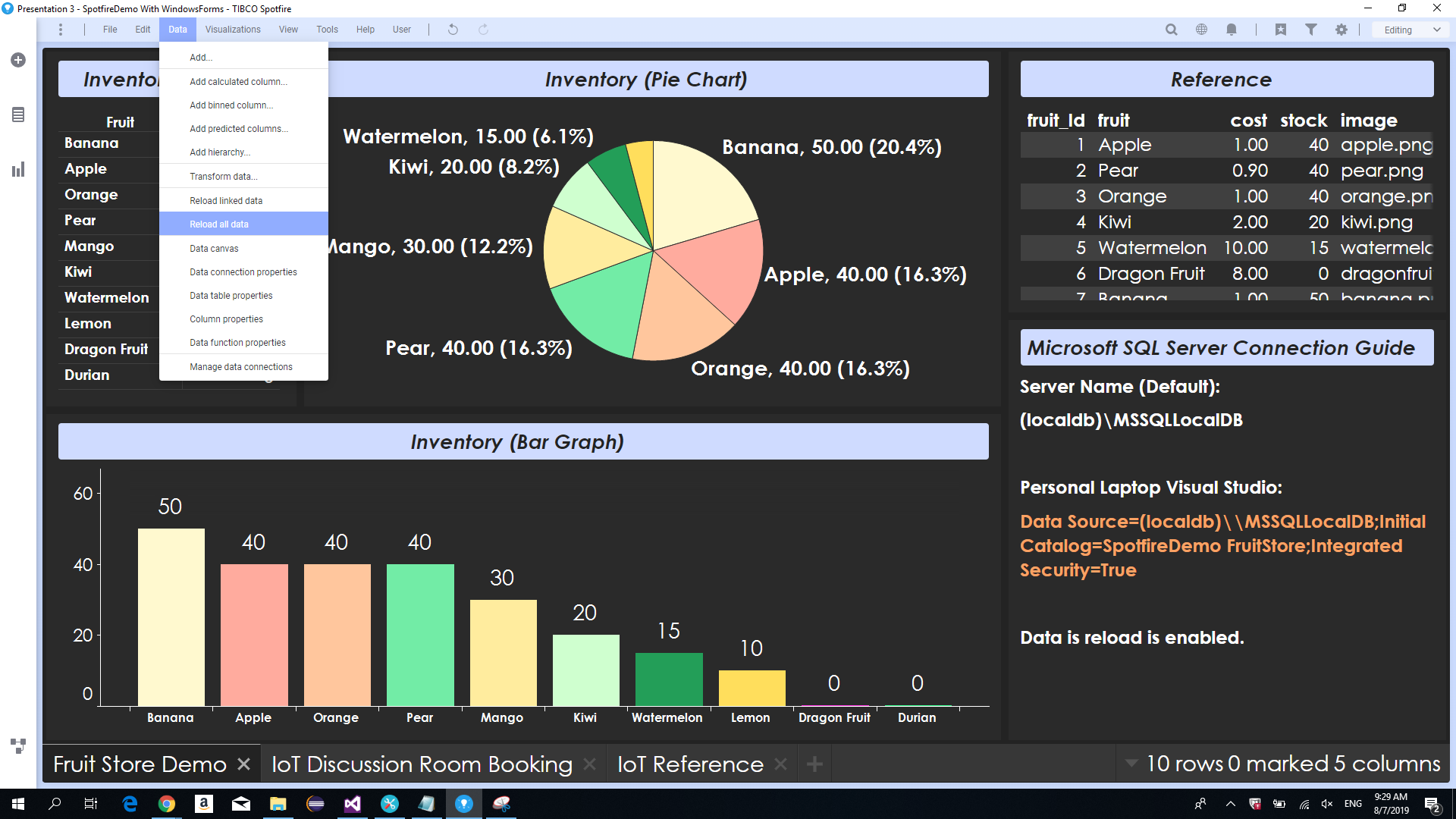
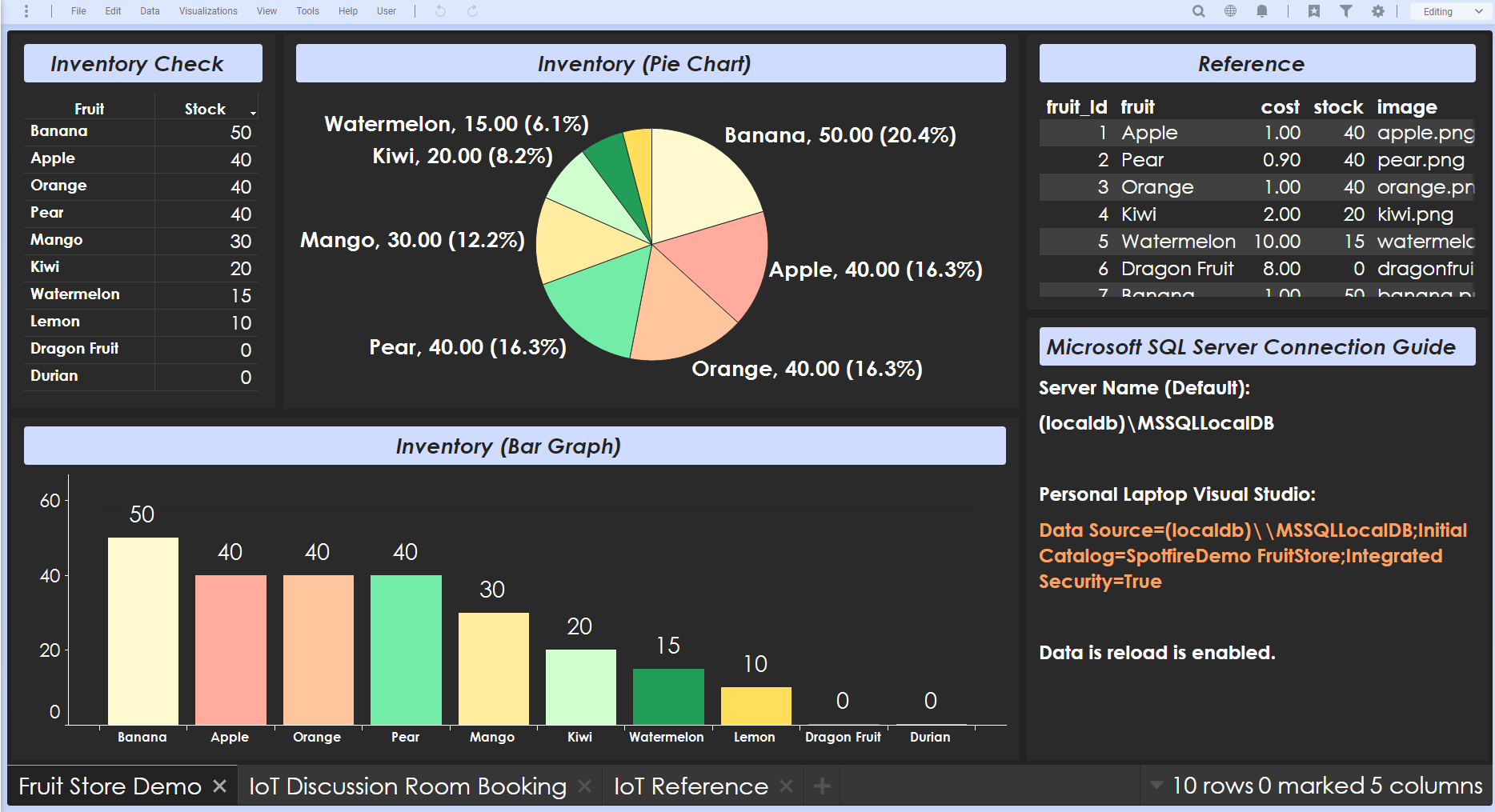
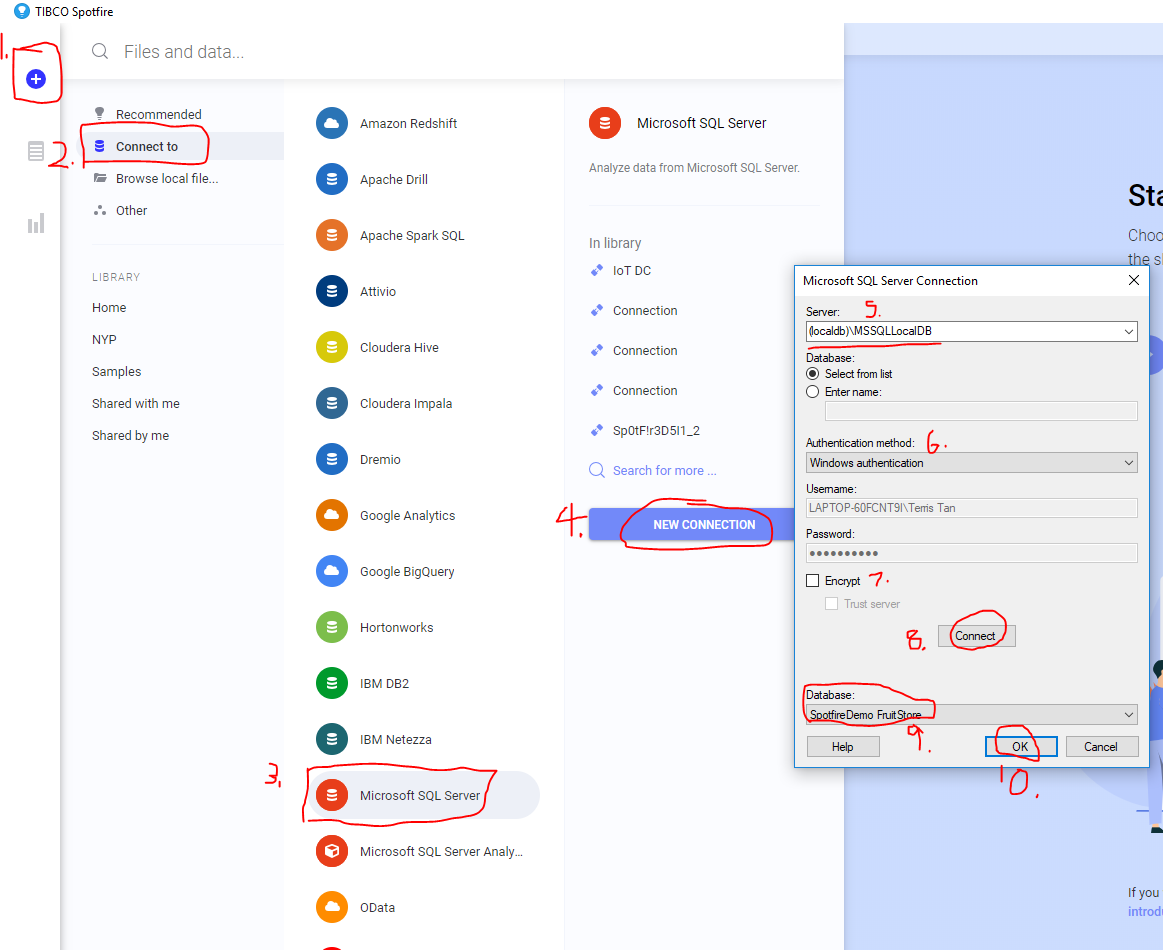
Next is to start with the UI, the purpose of this UI will be to make simple visualisations to display the IoT sensors current status. Below is a screenshot of my UI. It is best to avoid learning towards a similar UI to the “Logs” section as normal users may not understand how to read it.



After the UI, finish up adding the backend code to connect to the IoT Background Application. What is to be achieved are as follows:

1. Able to retrieve sensor data
2. Able to write to database
3. Able to display the status of the sensors in UI

Once finished, the last step is to test the connection to TIBCO Spotfire visualisations. As previously mentioned, connect your database to TIBCO via “Connect to > Microsoft SQL Server”. Then create a simple visualisation set to check of data is able to be viewed as expected, and changes are updated when data reloads.



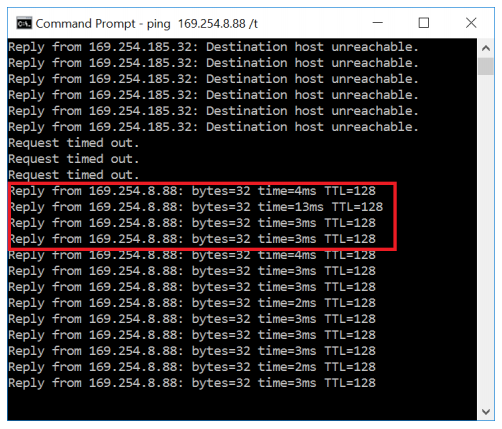
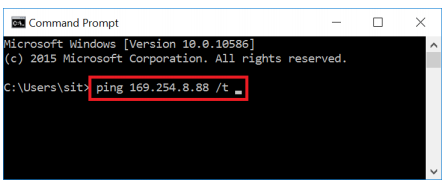
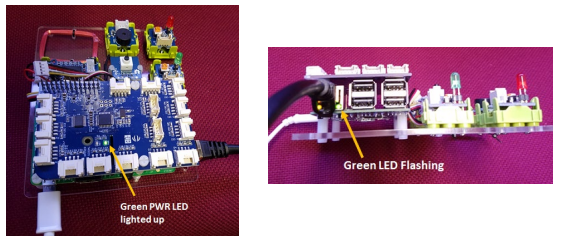
**4. Alternatives And Fixes**

**4.1 IoT Hardware Connection**

With IoT technology, errors may occur due to missing components or problems in connection.

Steps to ensure that your IoT Background Application is working properly:

1. Ensure that the IoT hardware is connected properly.
2. Refer to P00, practical 0. From the IoT practical resources for a guide to setup.
3. Refer to P01, practical 1. To test IoT hardware connection to your operating system.
4. Refer to the “Raspberry Pi/Visual Studio Setup Guide” and follow the steps accordingly.
5. Ensure that all relevant references are made, such as for dataComms and grovePi.
6. Ensure the Windows Forms also has the relevant references.
7. Using P01, perform the command prompt /ping.

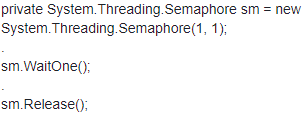


If all else fails, approach your supervisor to check the hardware for any faults.

**4.2 IoT Multithreading**

If your IoT Background Application is facing difficulties using multiple sensors, it may be due to the program being overworked. To solve this issue, you must use multithreading.

Multithreading allows the program to run multiple instances of codes by creating workers for each task. This can help reduce stress on your application and run smoother, especially useful for handling with multiple sensors due to having all sensors to be constantly running along with each other. Documentation can be found [here](https://www.tutorialspoint.com/csharp/csharp_multithreading.htm) and [here](https://docs.microsoft.com/en-us/dotnet/api/system.threading.semaphore?view=netframework-4.8).



**4.3 IoT Project Startup Error**

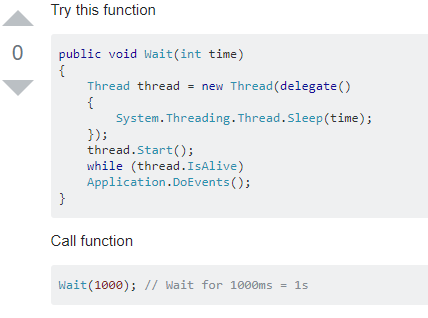
Occasionally, if you were to run the program too fast despite the green lights being turned on and connections are all fixed, the project may still come across a startup error. To fix this, just re-attempt to start the program again. This will usually work due to the connection not readily connected despite all the notice it gave. If not, check if your IoT hardware is properly connected.

**4.4 Enhance Waiting Function**

In Windows Forms, using the wrong waiting function can cause problems in loading the UI. By using basic functions like Thread.Sleep or Task.Delay, the outcome will be different from expectations. An article about the difference between these two can be found [here](https://stackoverflow.com/questions/20082221/when-to-use-task-delay-when-to-use-thread-sleep).

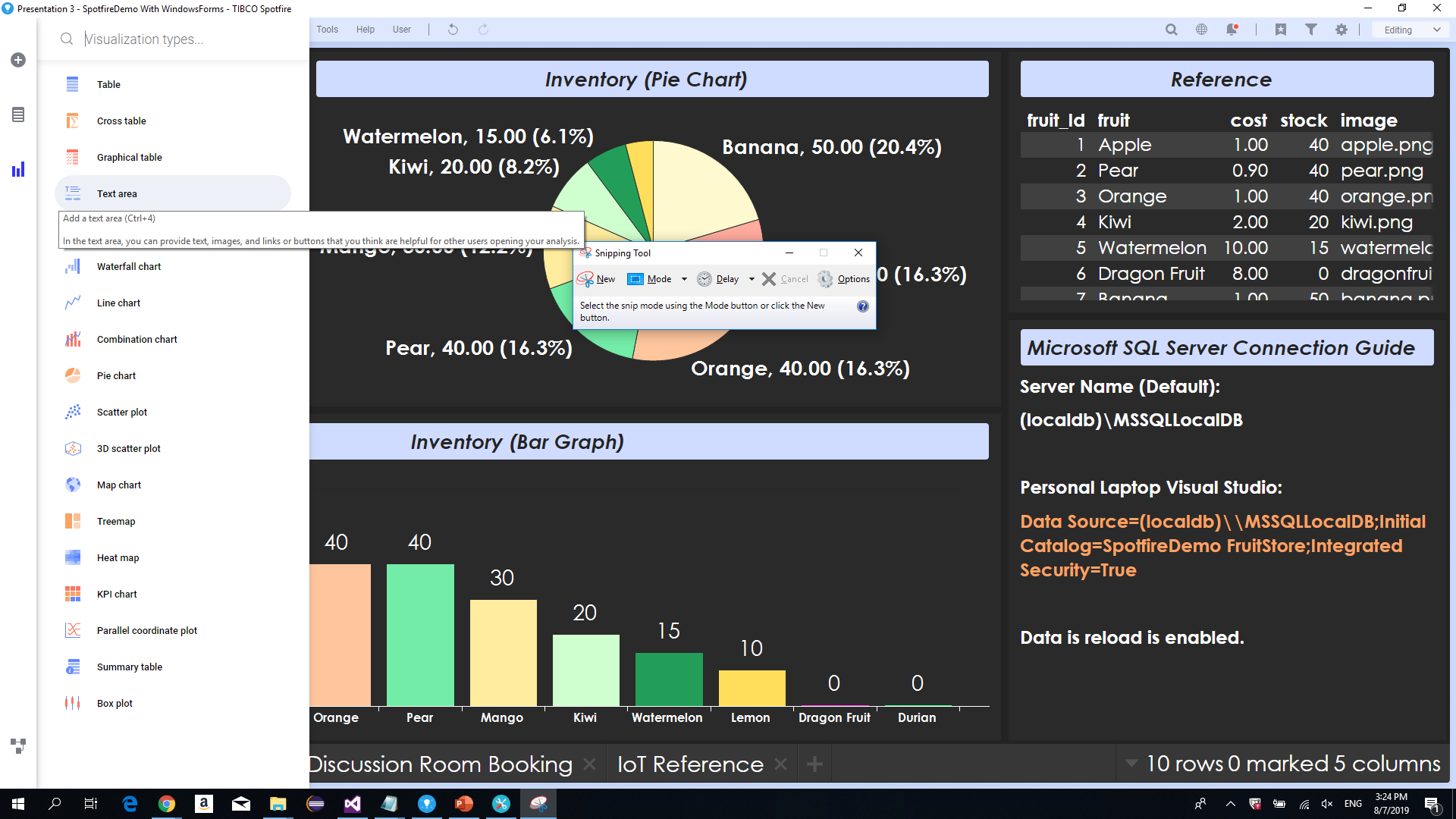
Despite both serving the same purpose of forcing the application to wait for a specified amount of time. Using either of these functions will actually cause the UI to load with the specified amount of time as well.

An example being, Wait(10000) will cause any rendering of UI codes before this line of code, to take 10 seconds to load the UI. To solve this issue, I have come across a potential fix in this link [here](https://stackoverflow.com/questions/10458118/wait-one-second-in-running-program).

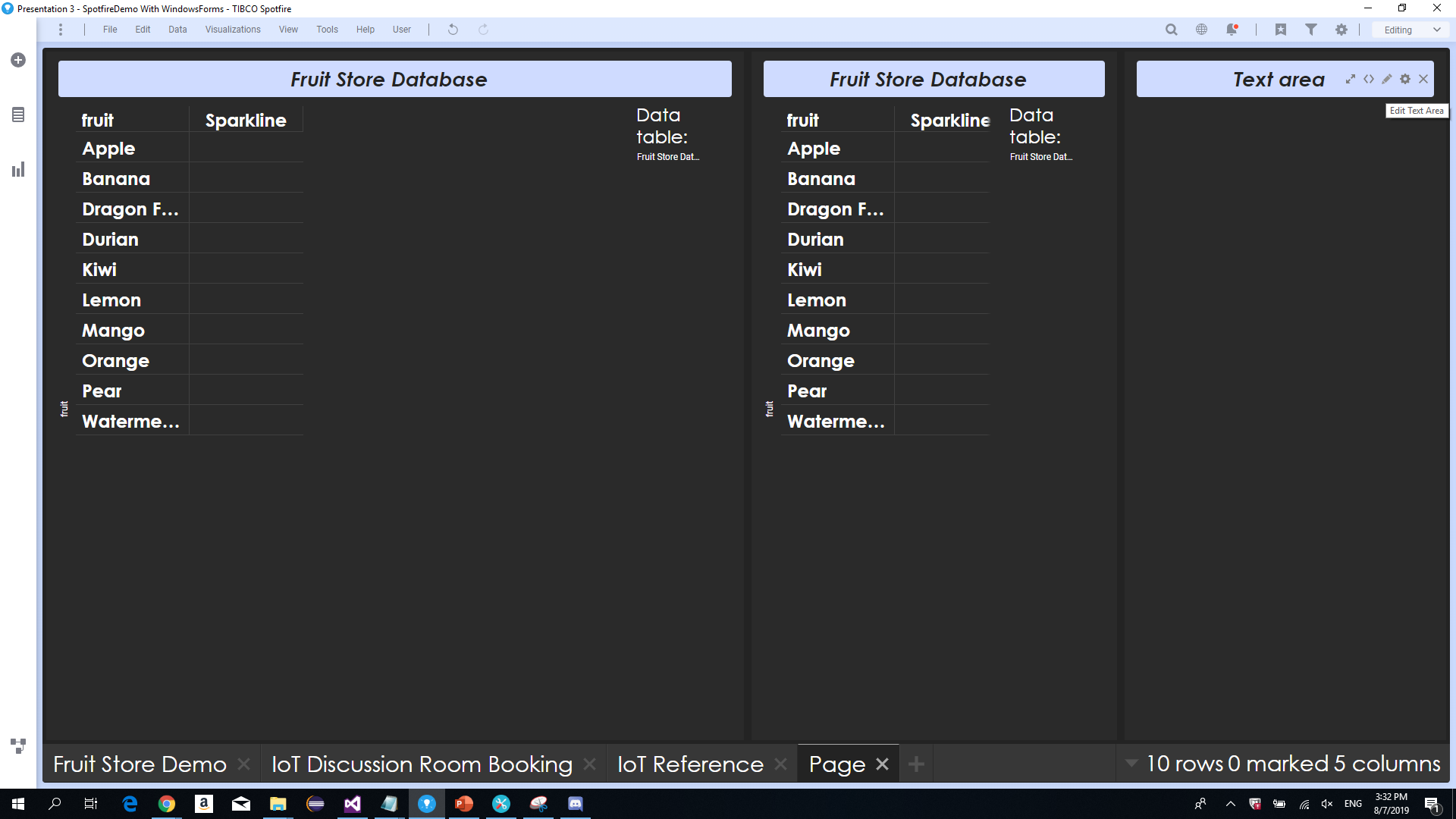


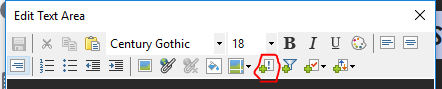
**4.5 TIBCO Spotfire Automatic Data Reload**

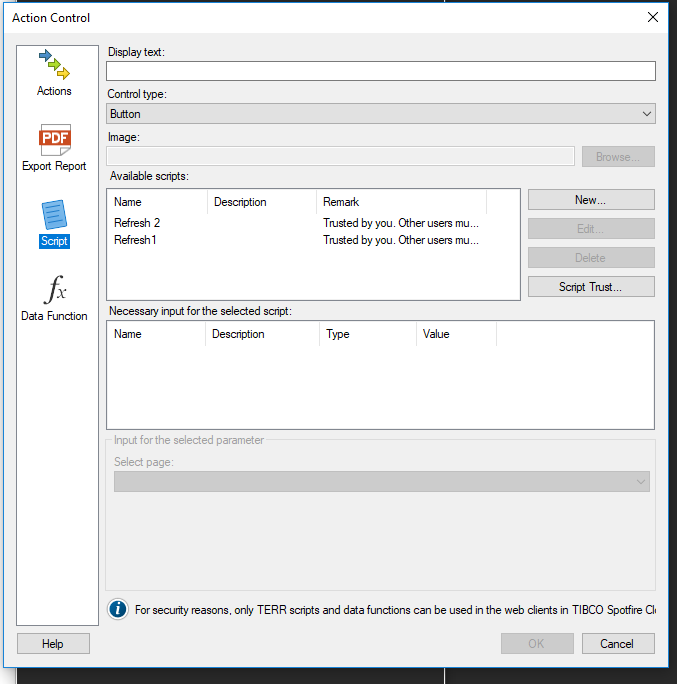
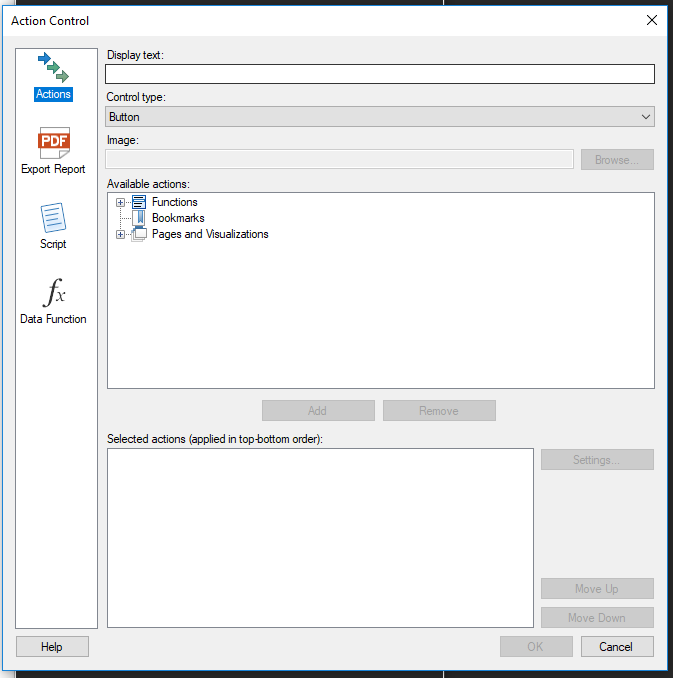
To enable automatic data reload, we must first learn how TIBCO Spotfire scripts work. First open an analysis and create a text area visualisation.

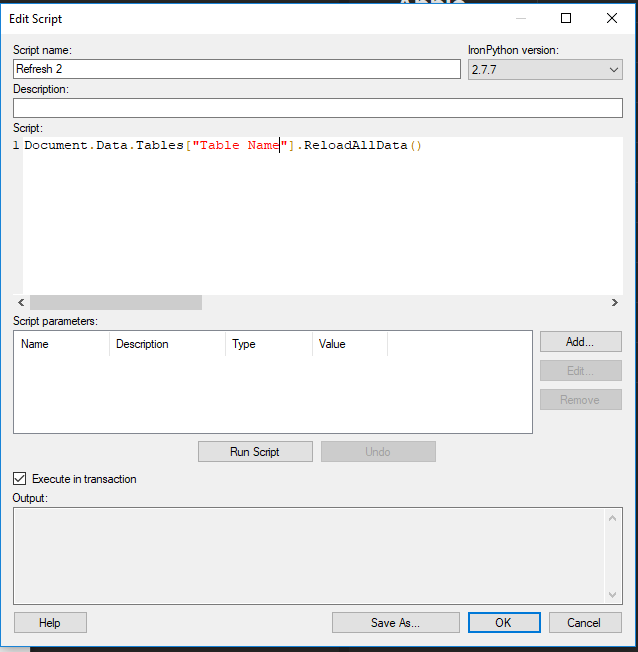


After creating, hover over your mouse to the pencil icon to edit the text area. A popup will appear, select the “Insert an action control” icon. When selected, another popup will appear to configure the action control. Give it a name and select “Button” control type. Then go to scripts, then add a new script. Copy the code as shown below.





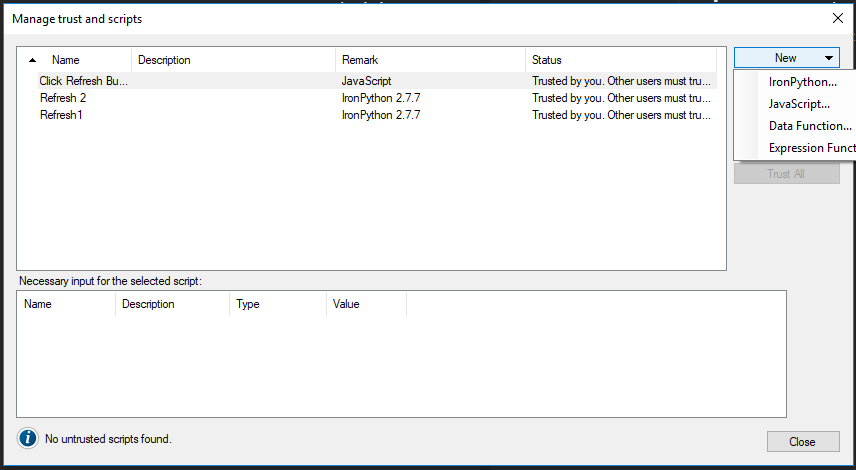
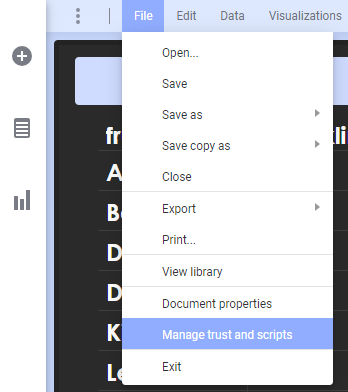


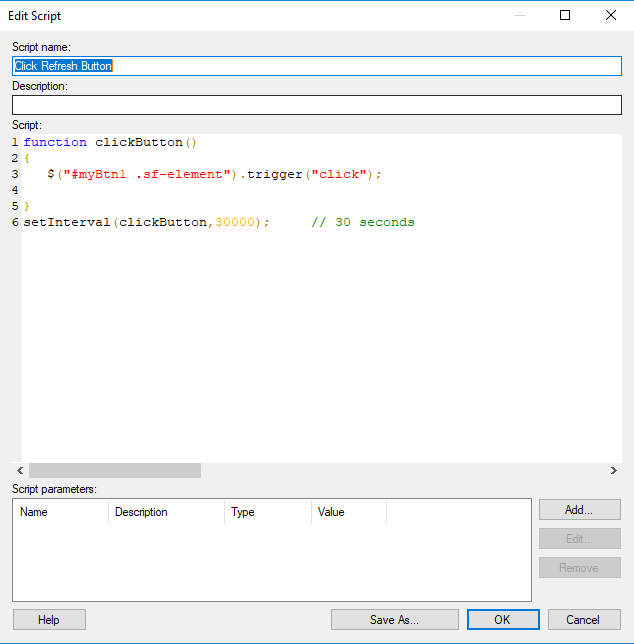


This line of script is in IronPython programming language. By creating this action button with the script, you have created a button. When clicked, will refresh the data source link.

Now we will add passive JavaScript as a background script. This script will trigger every given time period, to trigger the button. Hence, reload the data source link every given time period.

At the top left, click on “File > Manage trust and scripts”. A popup will appear, displaying all previously created scripts. Click on “New > JavaScript”. Then add the following lines of code to set an interval timer to trigger the button every given time period.





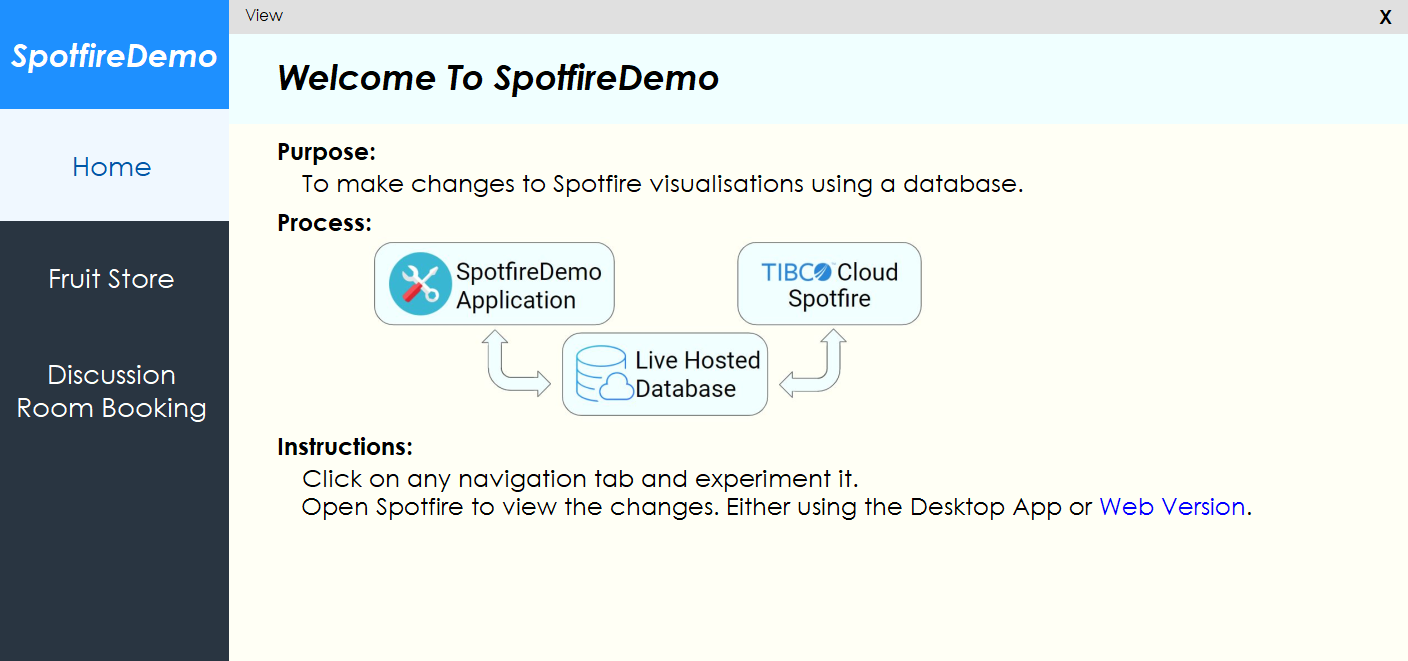
**Note**: After adding this script, be wary of the data reload interval. Although it is good practice to reload the data quickly, it also takes time to reload the data, especially when new data is being constantly added for IoT. Hence, set an appropriate interval for reloading the data source link.

**5. How To Use**

To use the already existing demo created, first download the project folders for the IoT Background Application and Windows Forms Application. These two projects should be in separate zip folders as they are two separate projects despite being reliant on each other.

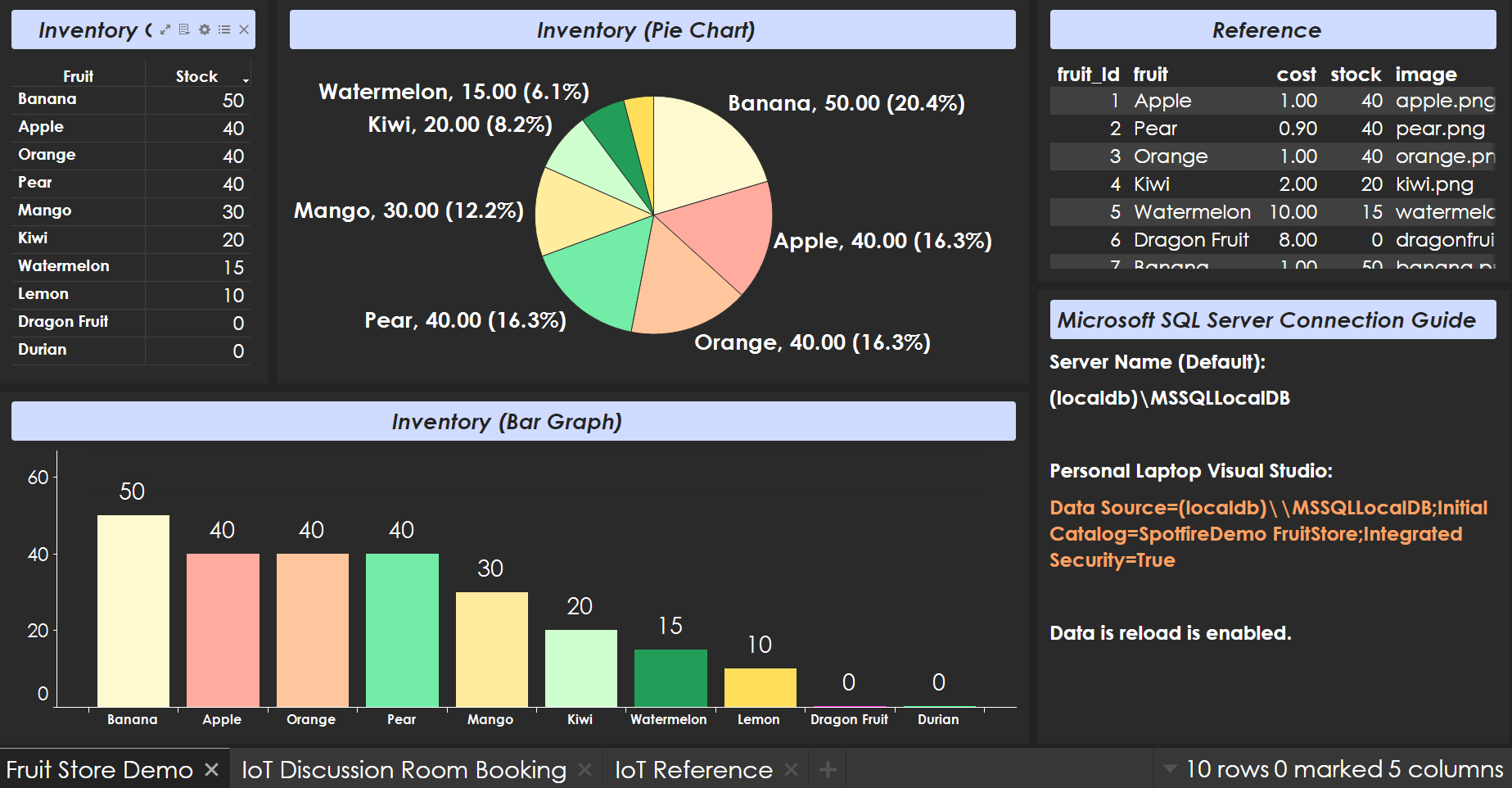
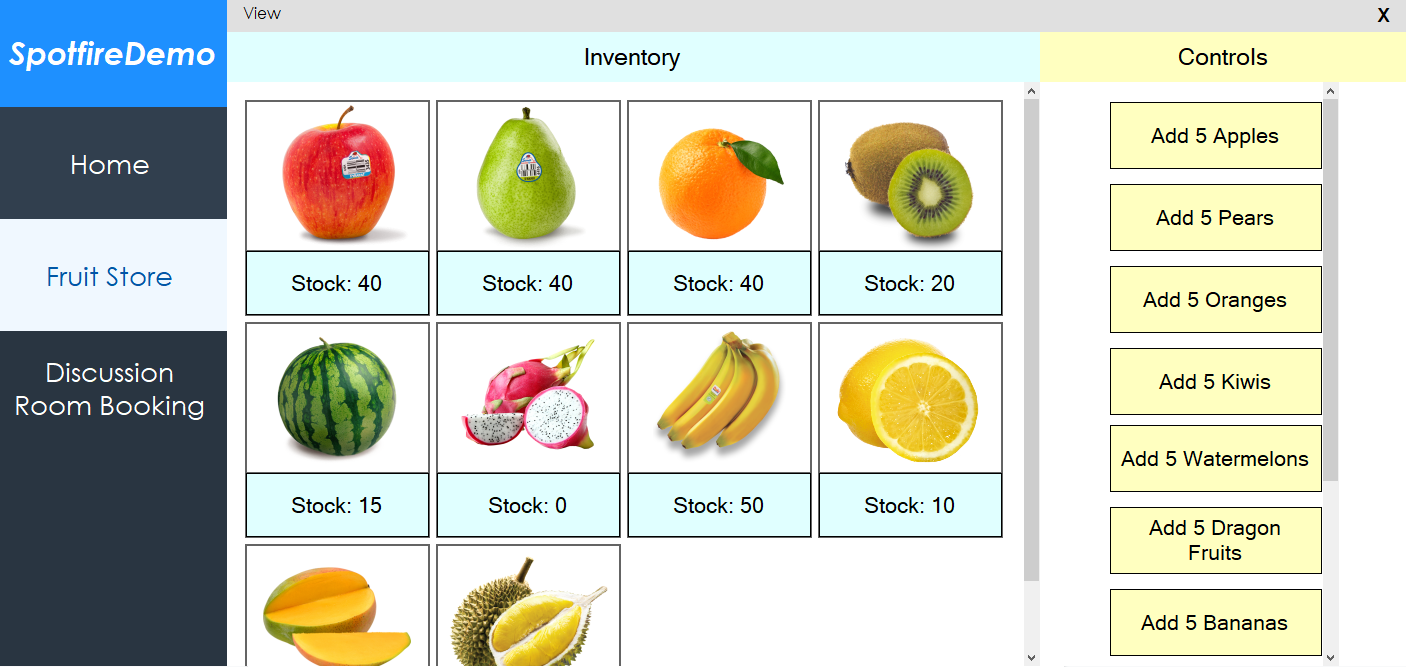
Next, ensure that you have the necessary prerequisites already set up for the IoT project. Ensure that you have:

1. Visual Studio 2015 Updates 3 Installed
2. Windows 10 operating system, developer mode enabled
3. IoT hardware

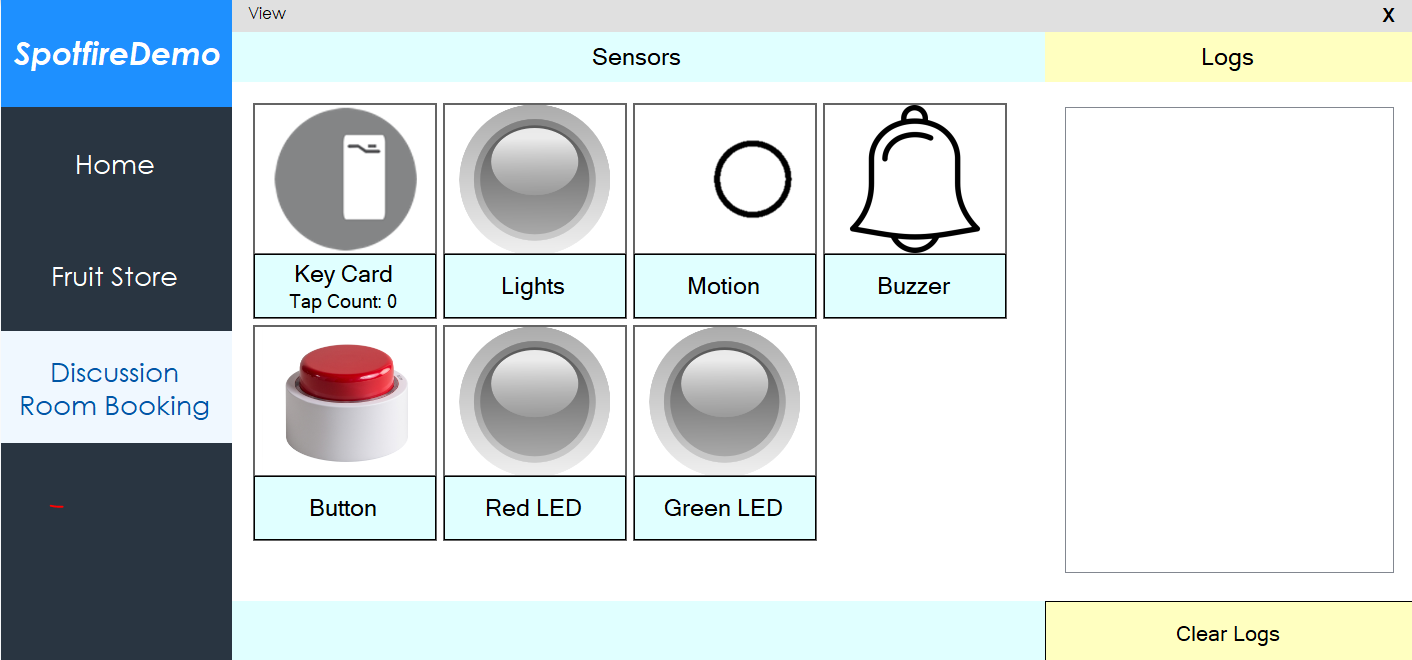
Then ensure that your IoT hardware connection is ready. Once all done, start the SpotfireDemo App (WinForms). 

**5.1 Fruit Store Demo**

The Fruit Store Demo can be run independently. By clicking on any of the right panels, you can add 5 stock to the corresponding fruit. To view the full potential of this demo, open your TIBCO Spotfire Desktop Application and open the linked analysis and visualisation set. Changes made in your application will be reflected in your visualisations after it reloads its data.



**5.2 Discussion Room Booking Demo**

The Discussion Room Booking Demo must run with the IoT Background Application. First, run the Windows Forms application.

Next, start the IoT Background Application. When the IoT Background Application is loaded, the logs will display a message, and the Red LED icon will turn red. From there, you may proceed to test the application with TIBCO Spotfire visualisations.

How the Discussion Room Booking Demo works is that, after starting the application, the Red LED will turn on. To start recording of a “Booking” (A Simulation), tap with the correct key card ID. The Simulation will then start, where all the sensors will start running at the same time.

**Simulation Actions:**

The Key Card will record the number of key card taps.

Lights will gather the light value and give the status of the lights turned on/off.

Motion will inform of any movement in the room.

Buzzer will be used as an alarm after no motion is detected after a specified time period.

Button will be used to end the simulation.

Red LED will remain off during and after the simulation.

Green LED will turn on only during the simulation.

**6. References**

**6.1 General**

**6.1.1 Prerequisites**

TIBCO Spotfire: <https://cloud.tibco.com/>

Spotfire JavaScript API: <https://community.tibco.com/wiki/tibco-spotfire-javascript-api-overview>

Extending Spotfire: <https://community.tibco.com/wiki/extending-tibco-spotfire#toc-9>

Denormalization:

<https://techdifferences.com/difference-between-normalization-and-denormalization.html>

**6.1.2 Utilities**

Comment Shortcut: <https://community.dynamics.com/gp/b/gpdynland/archive/2017/06/10/changing-visual-studio-keyboard-shortcut-for-comment-and-uncomment>

Free Icons: <http://www.iconarchive.com/>

Database Formats: <https://fileinfo.com/filetypes/database>

**6.1.3 Potential Bugs**

Database:

<https://visualstudiomagazine.com/blogs/tool-tracker/2012/05/dealing-with-local-databases-or-why-your-updates-dont-stick.aspx>

<https://blogs.msdn.microsoft.com/smartclientdata/2005/08/26/working-with-local-databases/>

<https://stackoverflow.com/questions/2575907/cannot-open-database-test-requested-by-the-login-the-login-failed-login-fail>

Running App:

<https://stackoverflow.com/questions/18102859/visual-studio-could-not-copy-during-build>

Multithreading:

<https://techdifferences.com/difference-between-normalization-and-denormalization.html>

**6.2 C# Coding**

**6.2.1 TIBCO Spotfire JavaScript API (Never used)**

Overview: <https://community.tibco.com/wiki/tibco-spotfire-javascript-api-overview#toc-3>

**6.2.2 Database**

Connection:

<https://www.youtube.com/watch?v=V9r-Gp3uNCE>

<https://social.msdn.microsoft.com/Forums/en-US/de641c58-dd70-4a4f-a96a-c47463c5baf9/relative-path-for-sql-database-file-connection?forum=csharpgeneral>

CRUD Commands:

<https://stackoverflow.com/questions/15246182/sql-update-statement-in-c-sharp>

<http://lamahashim.blogspot.com/2010/04/c-read-insert-update-delete-from-sql.html>

Commit:

<https://www.c-sharpcorner.com/article/two-phase-commit-protocol-in-c-sharp/>

<https://visualstudiomagazine.com/blogs/tool-tracker/2012/05/dealing-with-local-databases-or-why-your-updates-dont-stick.aspx>

Datatype Bit:

<https://stackoverflow.com/questions/43996465/bit-type-default-to-0-instead-of-null>

Datetime:

<https://stackoverflow.com/questions/4888277/add-default-value-of-datetime-field-in-sql-server-to-a-timestamp>

<https://social.msdn.microsoft.com/Forums/sqlserver/en-US/e91a7c8e-7bf0-4506-836e-db3650cafe00/set-default-value-for-datetime-column?forum=transactsql>

Column names from database:

<https://forums.asp.net/t/1943541.aspx?Best+way+of+extracting+column+names+from+a+datatable>

**6.2.3 General**

Random Number:

<https://www.c-sharpcorner.com/article/generating-random-number-and-string-in-C-Sharp/>

Do-While:

<https://www.tutorialsteacher.com/csharp/csharp-do-while-loop>

Array:

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/arrays/>

Reverse Array:

<http://www.vbforums.com/showthread.php?569858-RESOLVED-Reverse-Items-in-a-listbox>

**6.3 C# Windows Forms**

**6.3.1 Sections**

UI Ideas:

<https://www.youtube.com/watch?v=tgqKd7l7_s8>

<https://www.youtube.com/watch?v=K9Ps66GoD-k>

<https://www.youtube.com/watch?v=r8s35leUR6A>

Full-Screen App:

<https://www.youtube.com/watch?v=LPeMsjvLoco>

Waiting Function:

<https://stackoverflow.com/questions/10458118/wait-one-second-in-running-program>

**6.3.2 Controls:**

WebBrowser:

<https://stackoverflow.com/questions/5362591/how-to-display-the-string-html-contents-into-webbrowser-control>

Button:

<https://stackoverflow.com/questions/17796151/close-form-button-event>

<https://stackoverflow.com/questions/48490184/visual-studio-windows-forms-button-background-color-selected-when-click>

MenuStrip:

<https://www.youtube.com/watch?v=_MwlrBouHWs>

<https://stackoverflow.com/questions/15380730/foreach-every-subitem-in-a-menustrip>

Form:

<https://stackoverflow.com/questions/2891686/set-form-backcolor-to-custom-color>

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.form.size?view=netframework-4.8>

<https://stackoverflow.com/questions/4601827/how-do-i-center-a-window-onscreen-in-c>

UserControl:

<https://www.codeproject.com/Questions/156146/How-to-load-UserControl-into-panel>

<https://stackoverflow.com/questions/10871565/how-to-make-winforms-usercontrol-fill-the-size-of-its-container>

LinkLabel:

<https://www.youtube.com/watch?v=bzFPaxVCE48>

FlowLayoutPanel:

<https://www.youtube.com/watch?v=DJX6WgvOgk0>

**6.3.3 Programmatic Controls:**

<https://support.microsoft.com/en-sg/help/319266/how-to-programmatically-add-controls-to-windows-forms-at-run-time-by-u>

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.form.size?view=netframework-4.8>

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.control.dock?view=netframework-4.8>

<https://stackoverflow.com/questions/2891686/set-form-backcolor-to-custom-color>

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.control.font?view=netframework-4.8>

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.control.controlcollection.clear?view=netframework-4.8>

<https://stackoverflow.com/questions/4116743/picturebox-backgroundimage-property-c-sharp>

<https://stackoverflow.com/questions/24764353/load-picturebox-image-in-c-sharp-from-file-in-relative-path>

<https://stackoverflow.com/questions/674479/how-do-i-get-the-directory-from-a-files-full-path>

<https://stackoverflow.com/questions/21959490/picturebox-backgroundimagelayout-change-fails>