KPI PROCESS IMPROVEMENT

Final Report

Anders Ingelsten 20095402

Higher Diploma in Science in Computer Science

1 Contents

Declaration I declare that the work which follows is my own, and that any quotations from any sources (e.g., books, journals, the internet) are clearly identified as such by the use of 'single quotation marks', for shorter excerpt and identified italics for longer quotations. All quotations and paraphrases are accompanied by (date, author) in the text and a fuller citation is the bibliography. I have not submitted the work represented in this report in any other course of study leading to an academic award.

Student…………………………………........................... Date …..…………….............

Workplace Mentor………………………………………… Date ……………………………

2 Summary

Text here

3 Introduction

3.1 Project Type

This is a work-based project, where I will be improving the reporting process within several business areas. And I will be learning new skills in relation to developing a KPI dashboard using the knowledge acquired from the course modules. The internal company mentor will be the Chief Financial Officer. This means that I will be able to spend time on this project as part of my day-to-day work.

3.2 The organisation

Company A is an engineering solutions provider operating in Ireland, the UK and Scandinavia. It provides a wide range of services from the Design & Build of Sub-stations to construction of Airside Aviation Infrastructure to Turn-key Wind & Solar Energy Solutions. The company has a turnover of approx. 30million euro and employees approx. one hundred staff.

3.3 System Background and Project Scope

Company A has several systems that hold business information. Finance, Health and Safety and Operations staff query this data on a regular basis to produce business performance reports and to generate KPI (Key Performance Indicators) metrics for the different departments. The reports are generated on a weekly or monthly basis but there are also ad hoc reports.

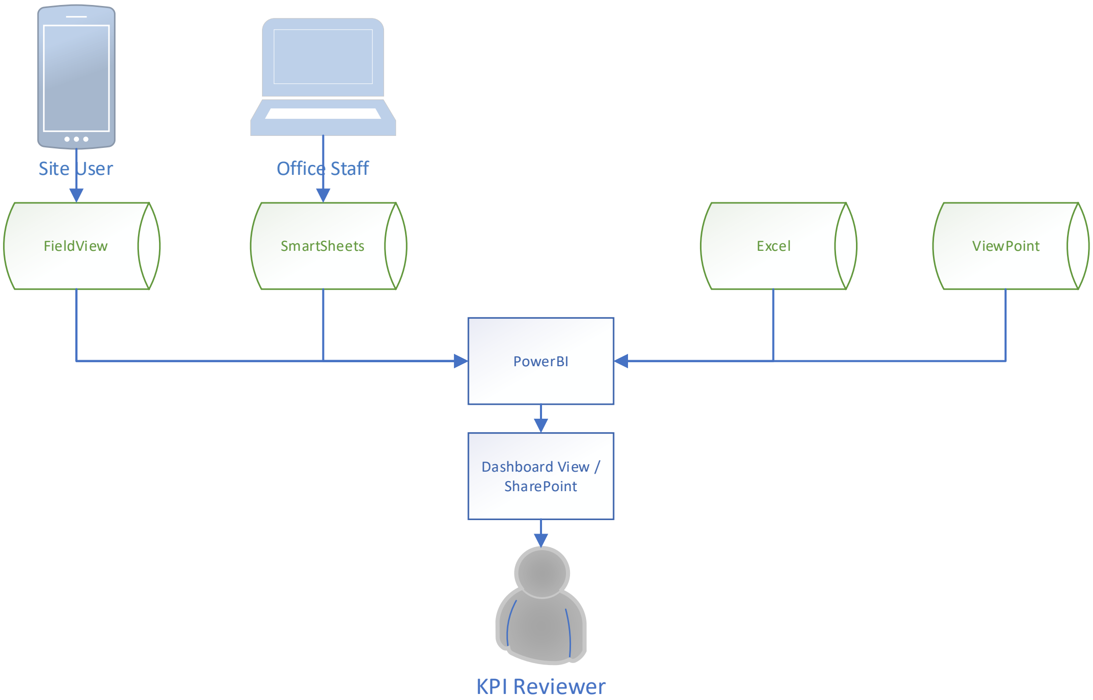


Figure 1: Overview of existing potential systems available to the project

The current systems identified for the initial project scope are:

System	Function	Location
Fieldview	Record QA/QC/Admin forms and Timesheet for site staff	Cloud
Smartsheet	Record timesheets for Salaried Staff	Cloud
Office365	SharePoint, Excel, word etc	MS Cloud

3.4 Fieldview

Fieldview is a third-party cloud-based and off-line mobile solution developed by Trimble (Floor et al., n.d.). It is used in Company A to replace paperwork on site. Users are equipped with a mobile device (phone and tablet), where the app has been installed. The users log in and use the application for snagging tasks, forms & permits, project delivery and handover. When the mobile device is synced – data is pushed to the cloud hosted database.

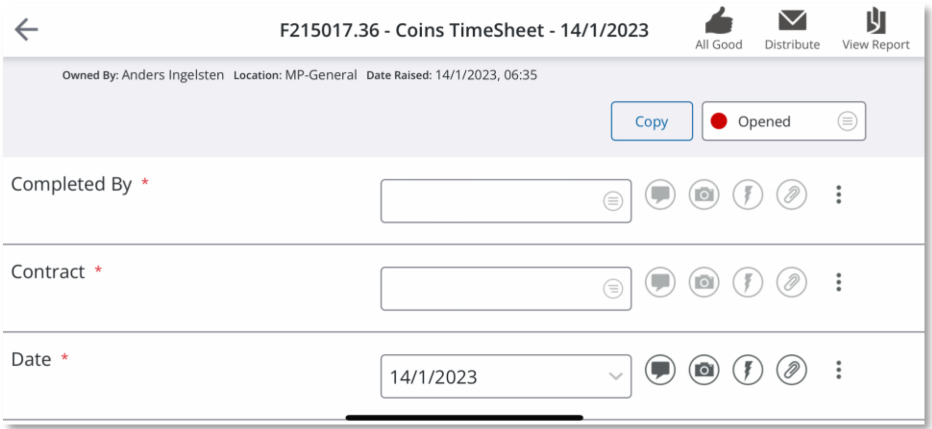


Figure 2: Sample view of Fieldview mobile form

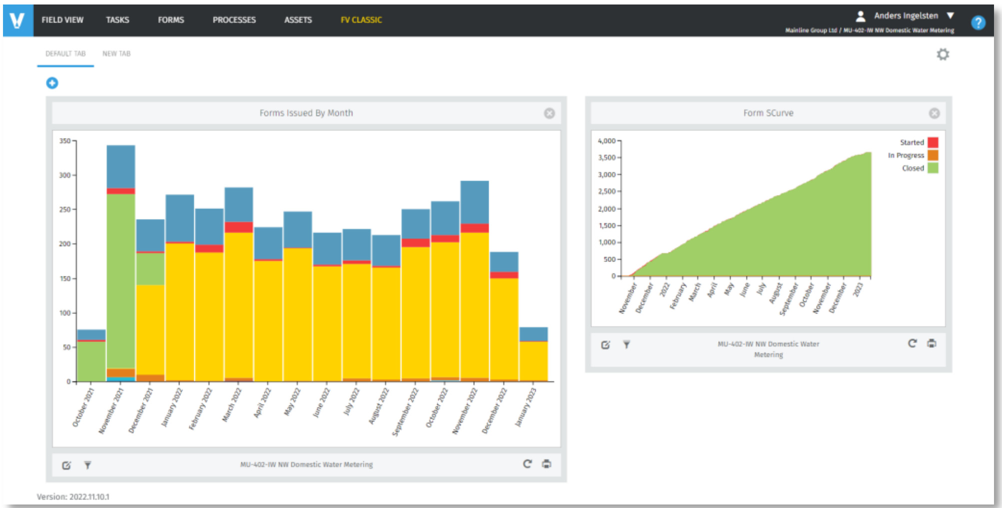


Figure 3: Sample view of the Fieldview widget reports

3.5 Smartsheet

Smartsheet is an online hosted solution that allows organisations to plan, track, automate, and report on work (Smartsheet, 2019). Company A uses the online application to track timesheets for salaried staff.

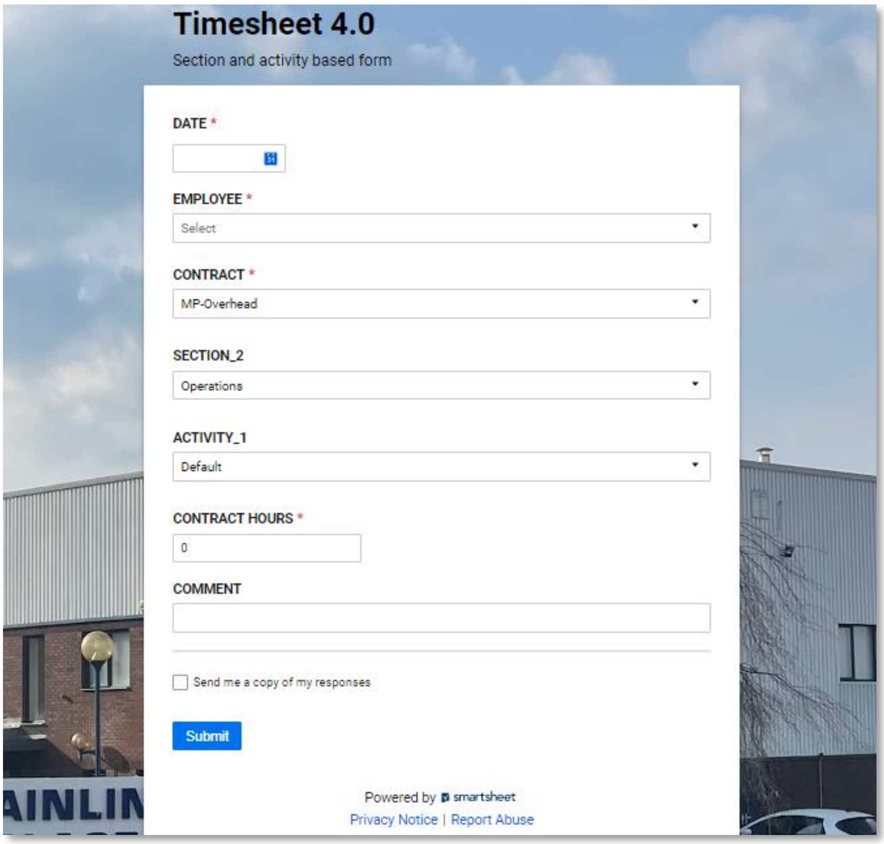


Figure 4: Sample view of Smartsheet form

3.6 Sharepoint/Office365

During the spring of 2022, the organisation has fully migrated its whole IT environment to the Sharepoint Online cloud, i.e., the full file repository of the organisation approx. 2.5 TB of files is hosted online with 24/7 365 access.

SharePoint Online is a cloud-based SAAS - "software as a service" provided by Microsoft, where organisation and users store and share information and use it collaboratively (www.microsoft.com, n.d.). All data and software are hosted on Microsoft own servers. Access, storage, and use of software is done by using a subscription model.

3.7 Current Situation

The company has over the summer of 2022, revised the KPI process and in September rolled out a new KPI process where departmental stakeholders fill in excel spreadsheet reports with various KPI data. For example, hours spent by employee per contract and other data like work site accident frequency etc. The departmental stakeholders access several systems and from there pull-down pdf and excel reports, from where they extract data and then collate it into excel spreadsheets which is then presented to the leadership team. This process is repeated monthly.

The company is aware that this is a time-consuming process and when it's well established is looking to improve the efficiencies in this process. The company has already identified that business intelligence tools like Power BI can be used to display almost real-time dashboard views of the business data. During 2022 there has been internal talks about identifying and bringing in business intelligence consultants/developers to do this work during 2023. Within the project, I will investigate business intelligence products for example the Microsoft Power BI platform. It's worth noting that the company has fully migrated to Office365/SharePoint during 2022 and is using the SharePoint platform as Intranet and repository for company data.

3.8 What are the problems.

The two main problems are:

1. Manual input of data by staff into excel spreadsheets is time consuming and this time can be better spent on other business processes.

2. The time difference between the live situation and the compiled report by staff leads to delays in business understanding, for example if work is profitable or not.

The CFO of Company A is therefore looking for a solution that can display KPI information from systems on the internal company SharePoint internet website.

4 Use Case

The primary reason for developing a dashboard pulling data from sources is the time saving. For example, the Fieldview application is structed in such a way that when a user, for example a project manager, would like to access the online portal, and view and pull-down information it can ONLY be done on a project-by-project basis. The systems do not have functionality in the user interface for a user to pull down for example all vehicle checks across all projects.

This means a user must access one project, access the report view, pull the information down and then access the next project and repeat the process until completed. Currently a user can create widgets that displays summaries but again, this must be done for every user on an individual basis.

A simple calculation will show how a dashboard can provide efficiencies.

Example, a user spends 1.5 hour every week to access every project area and compile a report of the amount of vehicles report submitted by staff. 1.5 hours x 48 working weeks = 72Hours which equates to 9 working days per year.

5 Potential Technologies, Tools, and Languages

As this is a work-based project the preferred direction from the leadership team is to use known technologies that are currently in use by the company. The two main reasons for this are that it creates greater resilience, and any costs are known, for example the company already host a Windows 2019 server in the Azure cloud. List of potential technologies, tools and languages:

- Office365
- SharePoint
- Power BI Desktop
- Power BI Data Gateway
- Power Query Editor
- SOAP UI
- MS Planner
- Html
- PowerShell
- Sharepoint Online Management Shell
- M language and DAX

5.1 Potential Issues

As the project will have access to commercially sensitive and potentially personal data, measures will be put in place to minimise exposure of this data. This may limit the scope of which systems and data will be incorporated in the project. If it's not feasible to limit the exposure, test data will be utilised.

6 Objectives

This project has three main objectives:

- **Ease of access.** By connecting separate data sets, transforming, and cleaning the data into a data model and creating charts or graphs to provide visuals of the data; it will assist users to find insights, within the organisation, of the operational data generated in Fieldview.

- **Real-time information.** To give users in the organisation, an updated almost real-time view of the situation in the company. This should provide the ability to solve problems and identify issues and opportunities.

- **Process improvement.** By streamlining publication and the distribution of the data into dashboards – users interpret published data whenever the underlying dataset is updated. This is instead of compiling reports through a time-consuming process and sharing the data in emails or a shared drive for stakeholders to then review.

7 Methodology

7.1 Dataflow

The data is envisaged to flow in the following way.

1. Users generate data onsite in the Fieldview app.
2. By syncing the device – the data generated in the app is pushed to the Fieldview server environment (Cloud)

3. The PowerShell scripts will then run automated on a server hosted in the Company A Azure environment, pull data from the Fieldview cloud.
4. Data is then saved in CSV files by the PowerShell scripts to Sharepoint Online.
5. PowerBI is then connected to the CSV files and refreshed on a regular basis.

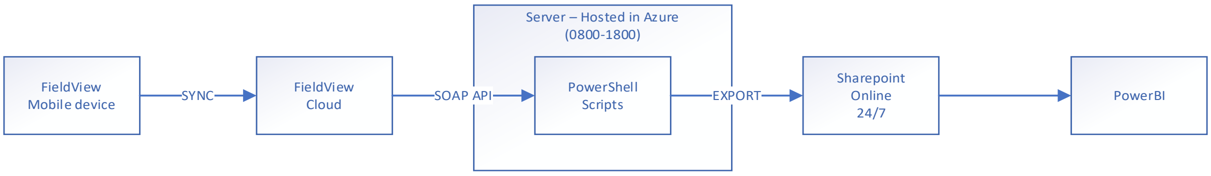


Figure 2: The Flow of Data from Fieldview to PowerBI

7.2 Modelling

Below is a simple representation of modelling of the data in Power BI, it is envisaged that data is presented in several tables. These tables are then joined pending any queries they may relate to. These tables and queries will then be the basis for the visualisation.

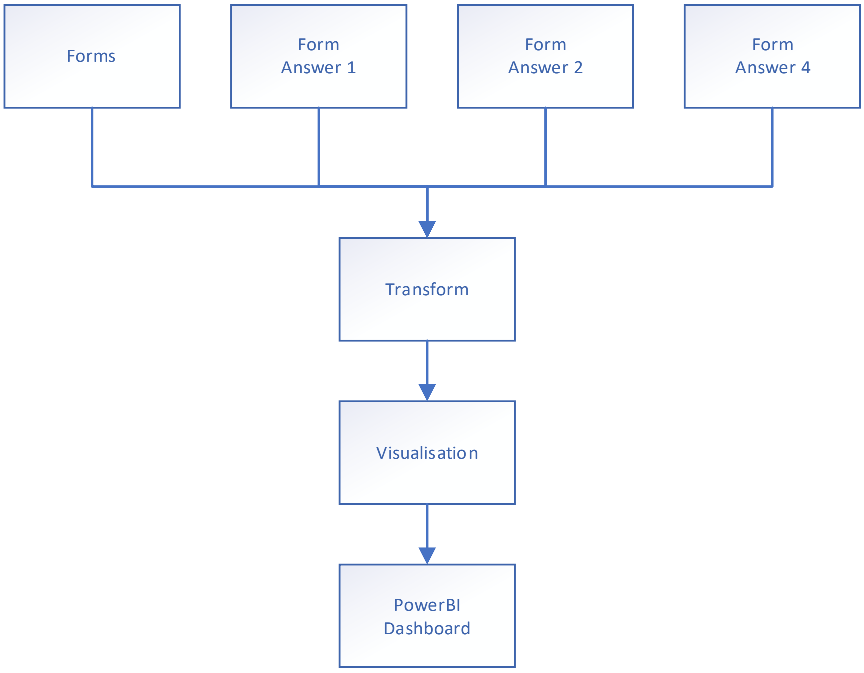


Figure 3:Sample Representation of the data model in PowerBI

7.3 Feedback Loop

The senior leadership team and the supervisor are very keen on the KPI Process improvement, its outputs, results of visualisation the data and the automation of the process.

The company is in a rapid growth spurt as several contracts was won during 2022/2023 and a current ramp up of activities are in progress. The direction was to use a feedback loop during the phased development process. Envisaged steps taken during the feedback loop:

1. Initial requirement meeting with suggestion by the supervisor of the visualisation of data.

2. Analyse, develop and display data as per initial requirement.

3. Follow up meeting to determine if the result has fit the expected requirements.

4. Publish any changes or improvements of any feedback received.

Repeat step 3 and 4 until decision to move to next visualisation feature. As the scope of the project and allocated time is controlled by the organisation supervisor's priorities; it's important to note that I will have to adhere to my supervisor's direction, which is fine and something to be expected when you work in SME environment where priorities and work allocation change on a day-to-day basis.

7.4 Technologies

The technologies used in the project was dictated by several limitation set by several factors. For example, the type of API used by Fieldview, scripting language/framework that can very efficiently and quickly generate data to PowerBI and any other technologies linked, researched or discovered during the project development process. The key drivers for deciding the scripting language were the Fieldview API, SharePoint Online and how quickly and cost effective the solution could be put into productivity. There was no direction from the internal supervisor in relation to choose made of which technology is used.

7.4.1 PowerBI

PowerBI will be the tool to display dashboards of KPI views to the stakeholders of the organisation. Developed by Microsoft, PowerBI is an interactive data visualization software product (Microsoft, 2022). It is a collection of software services, apps, and connectors that work together to turn unrelated sources of data into coherent, visually immersive, and interactive insights. Data can be inputted/connected by reading directly from a database, webpage, or structured files such as spreadsheets, CSV, XML, and JSON.

7.4.2 SOAP API

Fieldview uses SOAP API. SOAP is an acronym for Simple Object Access Protocol (AltexSoft, n.d.), which is a messaging protocol specification for exchanging structured information. I.E., allows for implementation of web services. Normally it uses XML Information Set for its message format, and relies on application layer protocols i.e., Hypertext Transfer Protocol (HTTP). It is worth noting Fieldview uses Hypertext Transfer Protocol Secure (HTTPS) for its API.

SOAP is over two decades old and allows users to pull or push data from a range of operating systems as well as numerous clients to run web services and receive responses over a range of script language and platforms.

7.4.3 PowerShell

During the research process several scripting languages was considered for the project for example Node JS, PowerShell or Python.

PowerShell was selected as it was deemed it would cause the least amount of impact on any existing system or servers and could very easily be transferred between machines in organisations IT environment. It has the potential of running future API requests inside the SharePoint Online Management Shell.

PowerShell is a command-line shell and scripting language (sdwheeler, n.d.) It supports variables, functions, branching (if-then-else), loops (while do, for, and foreach) and structured error/exception handling and closures/lambda expression.

7.4.4 Power Automate

Power Automate is part of Microsoft's Power platform, which is a low-code application environment which allows for data analytics and workflow automation. This is where power automate will be used for the project, automating the flow of getting data from for example the Fieldview API to SharePoint Online.

Power Automate has 2 environments and both will be used in the project.

- The Power Automate Cloud Based service will be used to schedule Power Automate desktop tasks to run on a required basis (powerautomate.microsoft.com, n.d.).

- The Power Automate for desktops application will be installed and used to run PowerShell scripts. The scripting feature allows the user to run blocks of code in a couple of different languages, for example PowerShell, Python, VBScript and JavaScript (georgiostrantzas, n.d.).

7.4.5 DAX

DAX is a formula language used in Power Bi for data analysis. DAX is an acronym for Data Analysis Expressions. DAX allows the user to formulate formulas to perform queries and calculations on data in tables stored in Power BI (Minewiskan, n.d.).

Measures are an important component of Power BI and Dax. A measure is a dynamic calculated formula which results change depending on the content i.e., when the data was last refreshed. Measures are created by using the DAX formula bar in the model designer. (Minewiskan, n.d.).

A formula in a measure can use standard functions such as COUNT or SUM, or you can define your own formula by using the DAX formula bar. It's worth noting that measures can be passed as an argument to other measures (Minewiskan, n.d.).

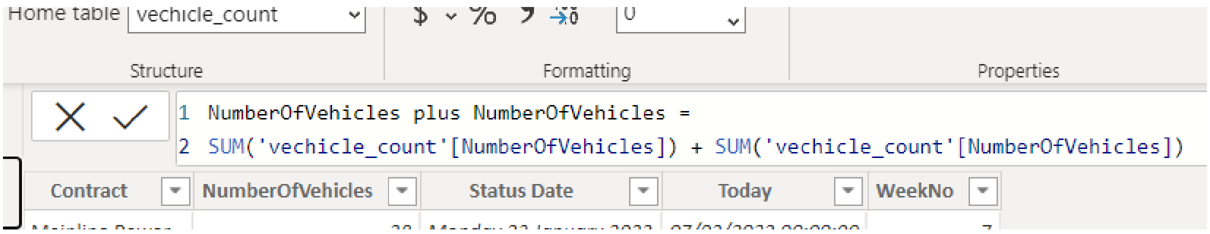


Figure 7: Sample View of DAX language in the formula bar of Power BI

8 Project Plan

The milestones to achieve are outlined below.

Milestone	Description	Due Date
Draft Proposal	Initial project proposal and concept	6th of November
Final Proposal	Articulate project nature and concept	4th of December 2022
Interim Report	Substantial update progress of the project	12th of February 2023
Final Project	Final project submission	2nd of April 2023

The project intends to span over the below phased development activities with preliminary due dates. Note the dates and phases may be subject to change as purpose of phase and allocated time is controlled by the organisation supervisor's priority.

Phase	Preliminary Sprint Due Dates
Research & Training	8th January 2023
Data Connector setup – Smartsheet	15th of January 2023
FV API setup and Data Persistence	29th of January 2023
PowerBI Dashboard – Fieldview Vehicle-check	12th of February 2023
PowerBI Dashboard – Fieldview Form Count	25th of February 2023
PowerBI Dashboard – SmartSheet Timesheet	19th of March 2023
Automation of Data flows and Power BI Refresh	26th of March 2023
Project Wrap Up	2nd April 2023

9 Project Planner

The Microsoft Planner (www.microsoft.com, n.d.) tool was used for planning and execution of the project. In planner the user can divide development phases also knowns as sprints into buckets. Inside the buckets the user would then add tasks into it.

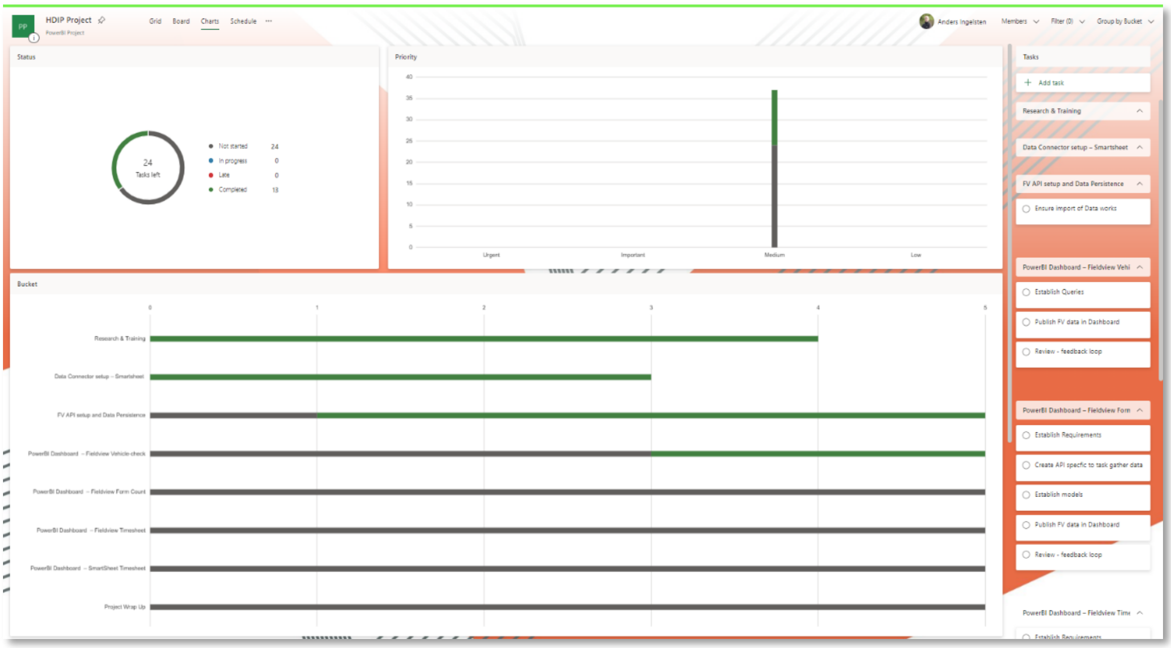


Figure 8: Microsoft Planner - Status of tasks and Buckets

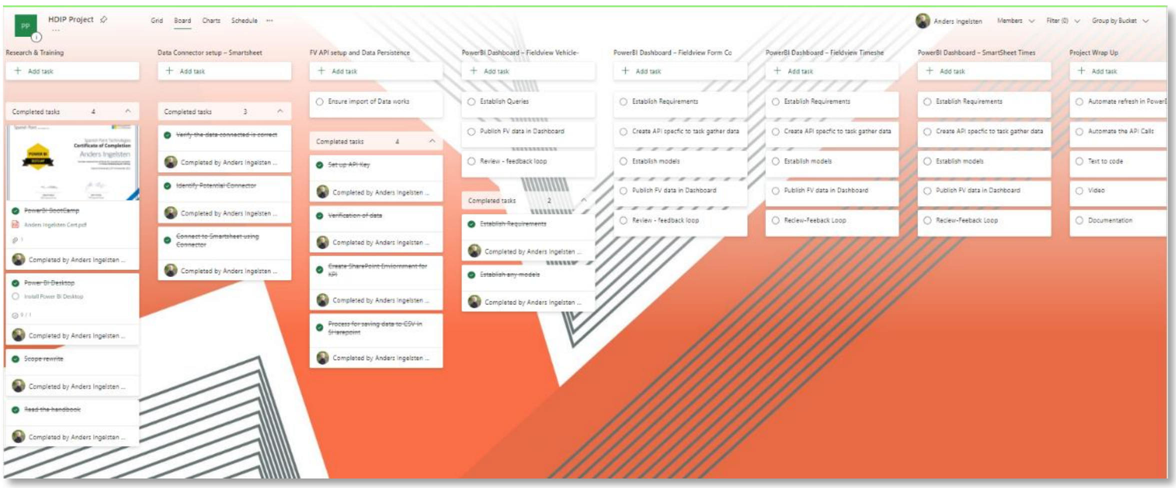


Figure 9: Microsoft Planner - Status of tasks and Buckets

10 Implementation

10.1 Phase 1: Research & Training

The phase of research and training covered three main areas.

The Fieldview API. The API Documentation of in the Viewpoint Help Section was investigated (help.viewpoint.com, n.d.), contact was also made to the support desk of Trimble to get direction of how the API was structured. The research revealed the following.

Fieldview has 3 Data Centre API regions/URLs.

1. UK
2. North America
3. Australia New Zeeland

The organisations data is stored in the UK region data centre.

Every Region has twelve APIs in two set of six different API (help.viewpoint.com, n.d.). Grouped by XML or JSON, the APIs are:

- Configurations Services
- Forms Services
- Tasks Services
- Process Services
- Assets Services
- Project Services

2 APIs was explored for the purposes of the project.

- Configurations Services – this API allowed me to identify and call project ids and associated information. For Example, the projectID is required to get Form Information
- Forms Services – this API allowed me to get form information, and individual answers.

Below follows 3 images of the formsservices API GetQuestionAnswer() command, first the parameters, a view of the SOAP API and then the returned answer information

Parameter	Type	Max Length	Required	Description
apiToken	string	20	Y	Your API security token configured in Field View.
formId	string	20	Y	The unique ID of the form.
questionAlias	string	100	Y	The question alias in the form you wish to retrieve the answer for.

Figure 10: Sample of input Parameters

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /FieldViewWebServices/WebServices/XML/API_FormsServices.asmx HTTP/1.1
Host: www.priority1.uk.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "https://localhost.priority1.uk.net/Priority1WebServices/XML/GetQuestionAnswer"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetQuestionAnswer xmlns="https://localhost.priority1.uk.net/Priority1WebServices/XML">
      <apiToken>string</apiToken>
      <formId>string</formId>
      <questionAlias>string</questionAlias>
    </GetQuestionAnswer>
  </soap:Body>
</soap:Envelope>
```

Figure 11: SOAP request and response

```xml
<FormAnswerResponse>
    <Status>
        SUCCESS
        <Message>Success.</Message>
    </Status>
    <FormAnswerInformation>
        <FormAnswerID></FormAnswerID>
        <FormTemplateID></FormTemplateID>
        <QuestionType></QuestionType>
        <DataType></DataType>
        <Question></Question>
        <Answer></Answer>
        <AnsweredBy></AnsweredBy>
        <AnsweredDateTime></AnsweredDateTime>
        <HasActions></HasActions>
        <HasImages></HasImages>
        <HasComments></HasComments>
        <HasDocuments></HasDocuments>
    </FormAnswerInformation>
</FormdAnswerResponse>
```

Figure 12: Returned information.

Time was spent understanding which API's information had to be pulled, stored, and passed on to other API's. One important feature noted in the research period was the API Call quota. The call quota is the number of points an API token can spend within a minute. An API token in Fieldview has a maximum quota of 120 points allocated. This means you are limited to the number of calls that can be made per minute. For example, the GetProjectFormsList() has a point quota of 10, so this would mean that associated token to the API can only process 12 calls in one minute.

API Token	**Maximum Quota (/min)**	**Remaining Quota**	**Last Reset**
AAAA-BBBB-CCCC-DDDD	120	32	2010-09-21 15:49:43.833

Figure 13: Call quota per API token

The research also showed that the Fieldview API would not easily return any data using the built-in data connectors in PowerBI and that the route of developing scripting of an API had to be taken. This is in line with the purpose of the project i.e., to showcase skills learned from the HDIP course.

It's worth noting that there exist third party apps like ZappySys ODBC Power Pack, who integrate SOAP API with PowerBI. However, this not a free software and annual subscription is approx. $650 per desktop install (ZappySys, 2018).

PowerBI. To get an understanding of what capabilities PowerBI has and its functionality I signed up for Spanish Point Technologies "Dashboard in a day" course. Spanish Point Technologies is a software company and a Gold Certified Microsoft partner. Spanish Point specialises in Azure, Microsoft 365, SharePoint, Dynamics 365, PowerApps & Power Automate, Power BI and SQL Server solutions (Technologies, n.d.). Company A has over the years used Spanish Point for various software solutions.

Dashboard in a day is a free full day workshop covering the capabilities of PowerBI through an instructor led online course.

- The goal of the course is to better understand how to:
- Connect to, import, and transform data from a variety of sources.
- Define business rules and KPIs.
- Explore data with powerful visualization tools.
- Build stunning reports.
- Share dashboards with their team and business partners and publish them to the web.

The power BI Bootcamp and training was completed on the 23rd of November 2022.

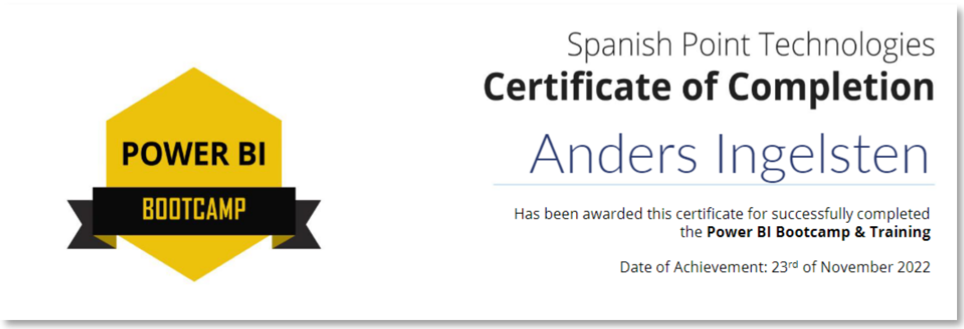


Figure 44: Certificate of Completion

In addition to the above training, I also conducted various research online to find out how to do certain various items, anything from PowerShell Scripting to SharePoint Online solutions and Power BI queries.

10.2 Phase 2: Smartsheet Data Connector setup

Smartsheet provides a Data Connector, which is part of PowerBI built in connectors.

Following steps were completed in this Phase.

1. Set up of user credentials in SmartSheet.
2. Ensure user had access to relevant tables in SmartSheet.
3. Connect Smartsheet to PowerBI using the built-in get Data feature in Power BI
4. Authenticate connection in PowerBI with Smartsheet user credentials.
5. Load and transform the relevant table from Smartsheet into PowerBI

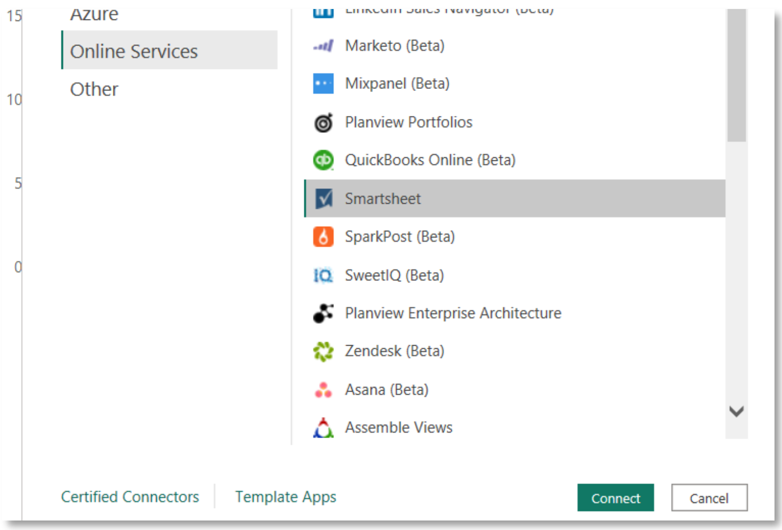


Figure 15: Get Data view in PowerBI of Online Services

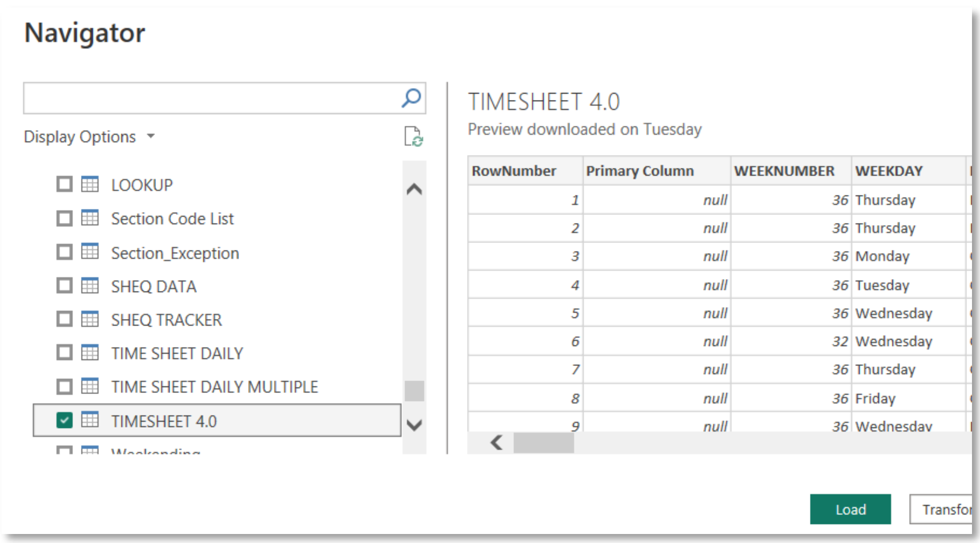


Figure 16: Preview of Connected Table ready to be loaded into PowerBI.

10.3 Phase 3: FV API setup and Data Persistence

The focus of this phase was to be able to reliably pull information from the Fieldview API to the organisation's hosted Data repository i.e., SharePoint Online.

The first thing I did was to generate API keys inside Fieldview, API key can be set on Group Organisation and Company level. A Company is a child of an Organisation. All keys were set up on Group Level.

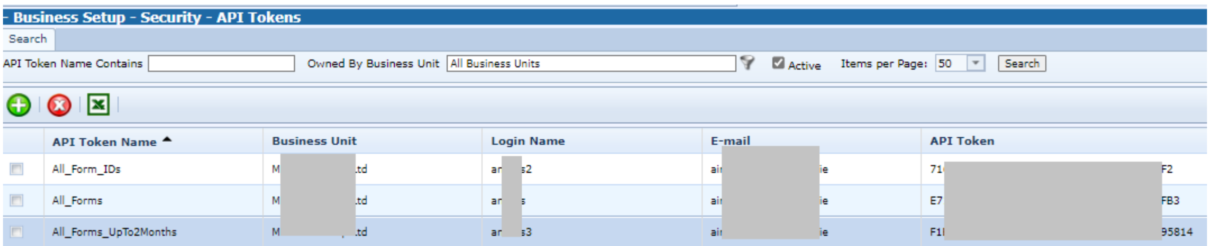
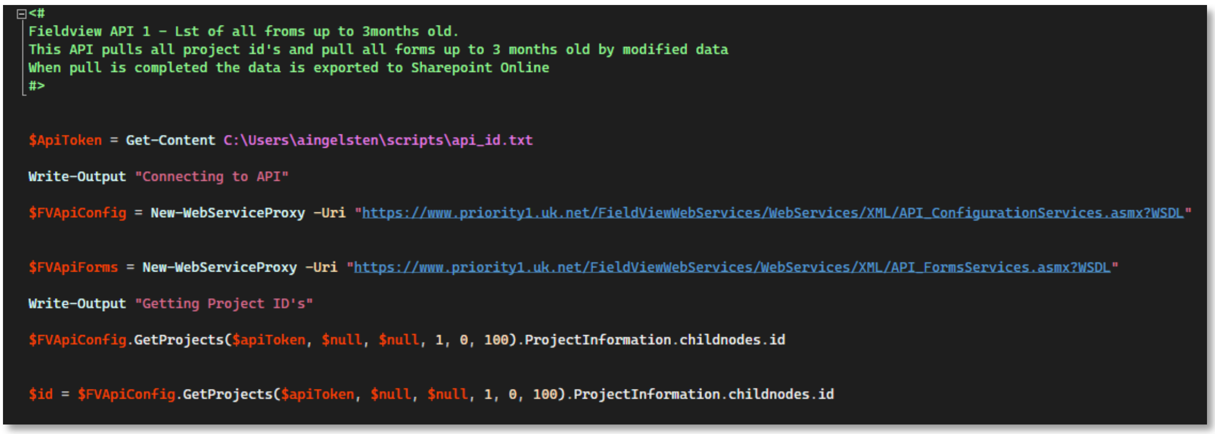


Figure 17: View of Created API Tokens in Fieldview

The SOAP protocol does not have to have any readily available plugins and add-on and the SOAP API relies on discreet calls for every interaction with the host. These calls also need to be structured inside a SOAP API envelope. I needed to be able to use a scripting language compatible with SharePoint online and which has the capabilities to execute general programming tasks, for example for each loop and if statements.

By utilising PowerShell and its module New-WebServiceProxy I managed to achieve this. The PowerShell New-WebServiceProxy will download the API's WSDL and use it to generate types for the proxy's interface, data contracts and headers. PowerShell also allows you to import variables from a file and export variables to csv files. Tasks can then be scheduled to run at certain intervals, but this will be resolved in a later phase.



```
<#
Fieldview API 1 - Lst of all froms up to 3months old.
This API pulls all project id's and pull all forms up to 3 months old by modified data
When pull is completed the data is exported to Sharepoint Online
#>


$ApiToken = Get-Content C:\Users\aingelsten\scripts\api_id.txt

Write-Output "Connecting to API"

$FVApiConfig = New-WebServiceProxy -Uri "https://www.priority1.uk.net/FieldViewWebServices/WebServices/XML/API_ConfigurationServices.asmx?WSDL"


$FVApiForms = New-WebServiceProxy -Uri "https://www.priority1.uk.net/FieldViewWebServices/WebServices/XML/API_FormsServices.asmx?WSDL"

Write-Output "Getting Project ID's"

$FVApiConfig.GetProjects($apiToken, $null, $null, 1, 0, 100).ProjectInformation.childnodes.id


$id = $FVApiConfig.GetProjects($apiToken, $null, $null, 1, 0, 100).ProjectInformation.childnodes.id
```

Figure 18: Code Snippet of the New-WebServiceProxy used to pull data from Fieldview.

The New-WebServiceProxy sets up a proxy object (sdwheeler, n.d.), which allows for interaction with the Fieldview SOAP API and by utilising PowerShell capabilities of running foreach loops, if statements, I could connect and store all the project ids, and then loop them back into the next call which would in this phase retrieve form information. Every API Token has a call quota per minute (help.viewpoint.com, n.d.). Therefor I generated several API keys so I can generate multiple calls. I also added delays in the loop, to avoid exceeding the call quota.

```
foreach ($projectid in $projectids)

{

Write-Output $projectid

$FVApiForms.GetProjectFormsList($apiToken, $projectid, $null, 0, $datefrom, $dateTo, $null, $null, $null, $null, $null, $null).ProjectFormsListInformation.childnodes

$formsList = $FVApiForms.GetProjectFormsList($apiToken, $projectid, $null, 0, $datefrom, $dateTo, $null, $null, $null, $null, $null, $null).ProjectFormsListInformation.childnodes


if ($formsList -eq $null)

{
Write-Output "*****NOTHING FOUND****"
}

else
{
Write-Output  "Adding Projectid"

$formsList | Add-Member -MemberType NoteProperty -Name "ProjectId" -Value $projectid

Write-Output  "Writing to file"

$formsList | Export-Csv -Path c:\Users\aingelsten\scripts\formslist.csv -append -NoTypeInformation

Write-Output  "Data written to file"
}

Start-Sleep -Seconds 10

}
```

Figure 19: Code snippet of foreach loop with an if else statement

```
FormID                  : F268753.52
FormTemplateLinkID      : 14680656
Deleted                 : false
FormType                : Operations
FormName                : MP-194 Site Diary
FormTitle               : 2022-11-25
CreatedDate             : 2022-11-25T12:29:28
OwnedBy                 :
OwnedByOrganisation     : Mainline Power Ltd
IssuedToOrganisation    :
Status                  : Completed and signed off
StatusColour            : #009900
StatusDate              : 2022-11-25T12:48:12
Location                : MP-194 Cork Airport Substation
OpenTasks               : 0
ClosedTasks             : 0
FormExpiryDate          : FormExpiryDate
OverDue                 : false
Complete                : true
Closed                  : true
ParentFormID            :
LastModified            : 2022-11-25T13:07:17
LastModifiedOnServer    : 2022-11-25T13:07:17
ClosedBy                :
FormTemplateID          : 15738621
ParentProcessTaskID     :

Adding Projectid
Writing to file
Data written to file
23754
*****NOTHING FOUND****
23755
```

Figure 21: Screenshot of Sample Data from the GetProjectFormsList( ) call, returning ProjectFormsListInformation.

10.3.1 Data persistence

The main factor that decided which type of method for data persistence was going to be used in the project was availability and cost. The organisation has access to hosted servers but none of these are operational 24 hours a day, 7 days a week.

Increasing availability would lead to increased cost and for the data to be persistent it must write to non-volatile storage. Data persistence was achieved by saving the data to a file, after duplications was removed, it was then exported to Sharepoint Online (Rajack, 2018). By saving a file with the same name to the same location, SharePoint online just saves a new version of the file. A CSV file stored in SharePoint Online allows very easy connection to PowerBI by its built in Web connector.

```
Write-Output "Exporting to SharePoint"

#Configuration of Sharepoint Variables
$SiteURL = "https://typetecmg.sharepoint.com/sites/ITMainline"
$SourceFilePath ="c:\Users\aingelsten\scripts\formslist.csv"
$DestinationPath = "Kpi_Data" #Site Relative Path of the Library


#Connect to PnP Online using Weblogin
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Powershell pnp to upload file to sharepoint online
Add-PnPFile -Path $SourceFilePath -Folder $DestinationPath

Write-Output "Process Completed"
```

Figure 22: Sample Code of writing a file to Sharepoint Online

```
InformationRightsManagementSettings   : Microsoft.SharePoint.Client.InformationRightsManagementFileSettings
IrmEnabled                            : False
Length                                : 0
Level                                 : Published
LinkingUri                            : https://typetecmg.sharepoint.com/sites/ITMainline/Kpi_Data/formslist.csv?d=wfa70bd2094764f24956703ae372bd2a1
LinkingUrl                            : https://typetecmg.sharepoint.com/sites/ITMainline/Kpi_Data/formslist.csv?d=wfa70bd2094764f24956703ae372bd2a1
ListItemAllFields                     : Microsoft.SharePoint.Client.ListItem
LockedByUser                          : Microsoft.SharePoint.Client.User
MajorVersion                          : 1
MinorVersion                          : 0
ModifiedBy                            : Microsoft.SharePoint.Client.User
Name                                  : formslist.csv
PageRenderType                        :
Properties                            : Microsoft.SharePoint.Client.PropertyValues
ServerRedirectedUrl                   :
ServerRelativePath                    :
ServerRelativeUrl                     : /sites/ITMainline/Kpi_Data/formslist.csv
SiteId                                :
TimeCreated                           : 22/01/2023 10:20:18
TimeLastModified                      : 22/01/2023 10:32:45
Title                                 :
UIVersion                             : 512
UIVersionLabel                        : 1.0
UniqueId                              : fa70bd20-9476-4f24-9567-03ae372bd2a1
VersionEvents                         :
```

Figure 53: Screenshot of successfully writing the data to Sharepoint Online in PowerShell

10.4 Phase 4: PowerBI Dashboard – Fieldview Vehicle-check

Phase 4 is the first phase that generated a Power BI dashboard result for the end company user. Employers who utilise company vehicles should have a process in place to conduct regular vehicle checks, to ensure that vehicles are in good order and safe to use (Health and Safety Authority, n.d.).

Company has a fleet of 44 vehicles and every user of the vehicle are required to do a weekly vehicle check. So, the requirement from the Plant Manager was to get a count of completed forms per week and division.

So, for this purpose I utilised the GetProjectFormsList API call, which gets all forms per project and timeframe. As company A have multiple projects in Fieldview I had to loop the API call through all Projects and also do an if statement as not all projects will have Fieldview forms in them during the timeframe requested

```
#Getting data by created date
$FVApiForms.GetProjectFormsList($apiToken, $projectid, $null, 0, $datefrom, $dateTo, $null, $null, $null, $null, $null, $null).ProjectFormsListInformation.childnodes

$formsList = $FVApiForms.GetProjectFormsList($apiToken, $projectid, $null, 0, $datefrom, $dateTo, $null, $null, $null, $null, $null, $null).ProjectFormsListInformation.childnodes

#If statement for if data is null or not
if ($null -eq $formsList)

{
Write-Output "*****NOTHING FOUND****"
}

else
{
Write-Output  "Adding Projectid"

#Adds the project id to Form as an identifier
$formsList | Add-Member -MemberType NoteProperty -Name "ProjectId" -Value $projectid

Write-Output  "Writing to file"

#Write result to file
$formsList | Export-Csv -Path c:\Users\aingelsten\scripts\All_Formslist.csv -append -NoTypeInformation

Write-Output  "Data written to file"
}
```

Data generated via the API call is then appended to a CSV file, which is checked and cleaned for duplication and then exported to SharePoint Online.

```
#EXport to Sharepoint
#Configuration of Sharepoint Variables
$SourceFilePath ="c:\Users\aingelsten\scripts\All_Formslist.csv"
$DestinationPath = "Kpi_Data" #Site Relative Path of the Library
$ClientId = "[redacted]"
$ClientSecret ="T[redacted]g="

#Site collection URL
$SiteURL = "https://t[redacted]sharepoint.com/sites/[redacted]/"

#Connect to SharePoint Online with ClientId and ClientSecret
Connect-PnPOnline -Url $SiteURL -ClientId $ClientId -ClientSecret $ClientSecret

#Powershell pnp to upload file to sharepoint online
Add-PnPFile -Path $SourceFilePath -Folder $DestinationPath
```

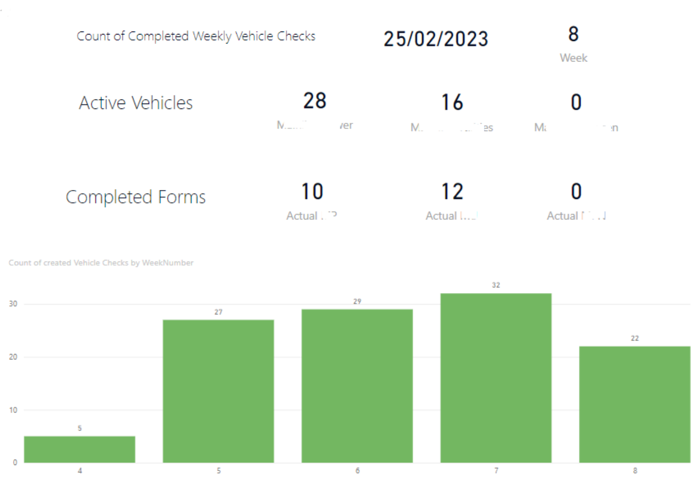
I added the SharePoint data into PowerBI by using the Get Data Web function which takes the SharePoint URL of the CSV file. When the file is updated in SharePoint I can refresh the data in PowerBI by utilising the refresh function.

ProjectId	FormID	FormTemplateLinkID	Deleted	FormType	FormName	FormTitle	CreatedDate
24250	F1.1801134	12252321	FALSE	SHE Forms	S53Site Safety Audit Form		2023-01-06T12:00:1
24250	F1.1809082	16080373	FALSE	Electrical	E20 HV-LV Ducting Cleaning / Proving Report	Yes	2023-01-10T15:00:4
27385	F1.1823210	16854981	FALSE	Administration	207-Coins TimeSheet	001OSM - Prelims	2023-01-16T12:37:5
24250	F1.1826163	16055566	FALSE	Operations	MP-192 Kilroot Site Diary	16/01/2023	2023-01-17T10:18:4
24250	F1.1826236	16055566	FALSE	Operations	MP-192 Kilroot Site Diary	17/01/2023	2023-01-17T10:31:1
24250	F1.1826305	12252321	FALSE	SHE Forms	S53Site Safety Audit Form		2023-01-17T10:45:5
21549	F1.1842734	16750904	FALSE	Assets	Asset Transfer		2023-01-23T14:48:0
24250	F1.1852233	16080373	FALSE	Electrical	E20 HV-LV Ducting Cleaning / Proving Report	Yes	2023-01-26T10:30:1
24250	F1.1852234	16080373	FALSE	Electrical	E20 HV-LV Ducting Cleaning / Proving Report	Yes	2023-01-26T10:30:2
24250	F1.1853322	16080307	FALSE	Electrical	E21 HV Cable Installation Record Sheet v2		2023-01-26T13:53:1
24250	F1.1853332	16079838	FALSE	Electrical	E23 ESB HV Cable Insulation Resistance Test Certificate		2023-01-26T13:54:5
24250	F1.1854386	15866705	FALSE	Operations	Jointing Quality Check		2023-01-26T17:04:4
24250	F1.1871316	16080307	FALSE	Electrical	E21 HV Cable Installation Record Sheet v2		2023-02-02T10:04:5
27385	F1.1885615	16854981	FALSE	Administration	207-Coins TimeSheet	001OSM - Prelims	2023-02-08T08:22:5
26811	F1.1886631	16088592	FALSE	ITP	ITP 010 INSTALLATION OF MV CABLE		2023-02-08T11:42:5
21549	F1.1886871	16088592	FALSE	ITP	ITP 010 INSTALLATION OF MV CABLE		2023-02-08T12:32:1
20698	F1.1887418	12031524	FALSE	Administration	Company Vehicle Check List	Ford Transit Custc	2023-02-08T14:23:3
21206	F1.1887436	12465271	FALSE	Administration	Coins TimeSheet		2023-02-08T14:30:5
21206	F1.1887438	12031524	FALSE	Administration	Company Vehicle Check List	Ford Ranger Wildt	2023-02-08T14:31:0
21549	F1.1899188	14789567	FALSE	Operations	Pre Site Visit Report		2023-02-13T14:54:1

I analysed the imported data, and to create the desired out come I created a Measure in DAX scripting that provides a count per form name, weeknumber, location and status like below. In this case the form called “Company Vehicle Check List”. It is worth noting I had to use the IF(ISBLANK) feature in to avoid the card displaying the value “BLANK” (Rad, 2020).

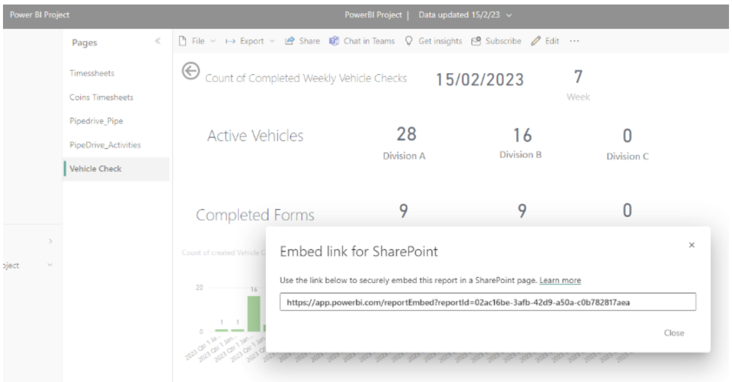
```
Actual MP = IF(ISBLANK(
COUNTX(
    FILTER(
        formslist,
        formslist[FormName] = "Company Vehicle Check List"
            && formslist[WeekNumber] = vechicle_count[Week] && LEFT(formslist[Location], 3) = "MP-" && formslist[Status] = "Completed and signed off"
    ),
    formslist[FormName]
)), 0)
```

I then used the vizulisation tools in PowerBI to create the following Report Dashboard, count of active vehicles and completed vehicle check list forms, and an additional view that shows the count of check lists per week in a table diagram.



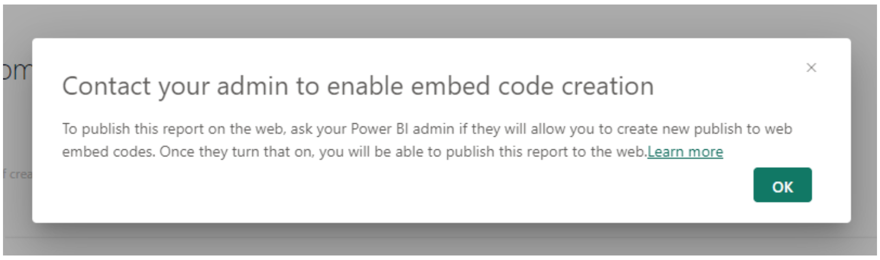
The next task for this phase was to make it visible for the end user through the Company's SharePoint Intranet. I had 2 options.

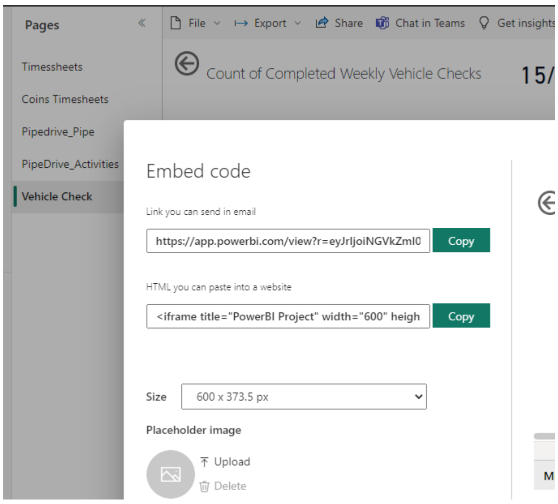
- Publish an Embed Link in SharePoint – this requires that every individual user to have a PowerBI license.



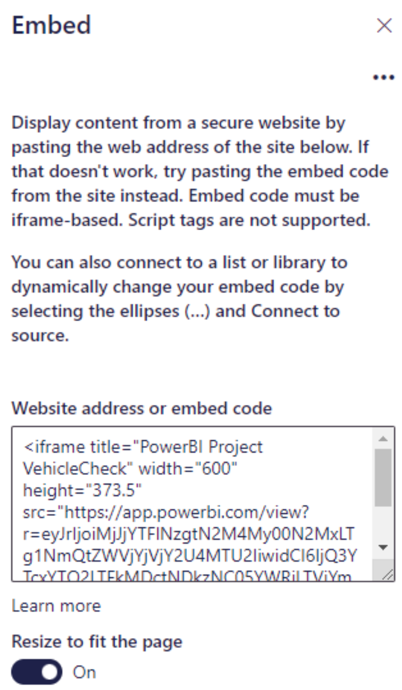
- Or to generate Embedded Html code, so it can be published to multi users.

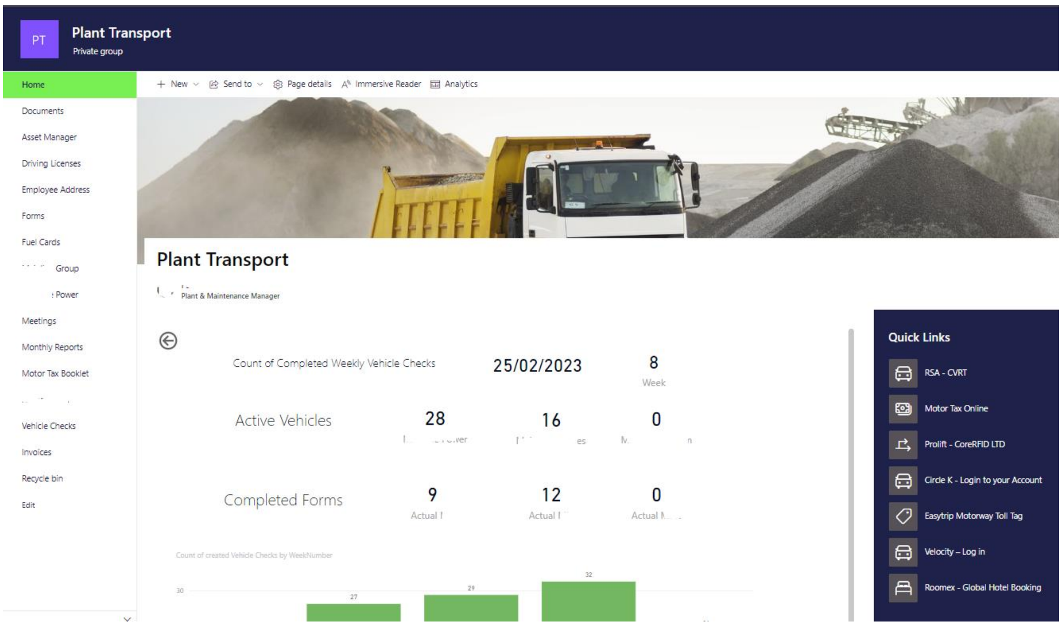
I choose the latter, which required me to purchase PowerBI Pro license and I also had to make sure the default admin settings which allows the user to create embedded code creation.





The embedded code was then published in SharePoint using the Embed code function.





End result of the published view embedded in the Plant Transport page.

10.5 Phase 5: PowerBI Dashboard – Pipedrive

At the project planning phase, I envisaged after the vehicle check phase to develop a dashboard that would display form counts per project. An active project has several forms that are required to be filled in on a regular basis, for example, daily site diary's, weekly project reports, monthly safety audits etc.

However, the commercial team requested that I would instead create a dashboard which would display the current business development pipeline. Displaying a summarised contract value per quarter start period.

Pipedrive is a CRM (Customer Relation Management) system that allows the user to track business development, e.g., potential leads, their value and tracking interactions like phone calls and meetings (Pipedrive Inc / Pipedrive OÜ, 2018).

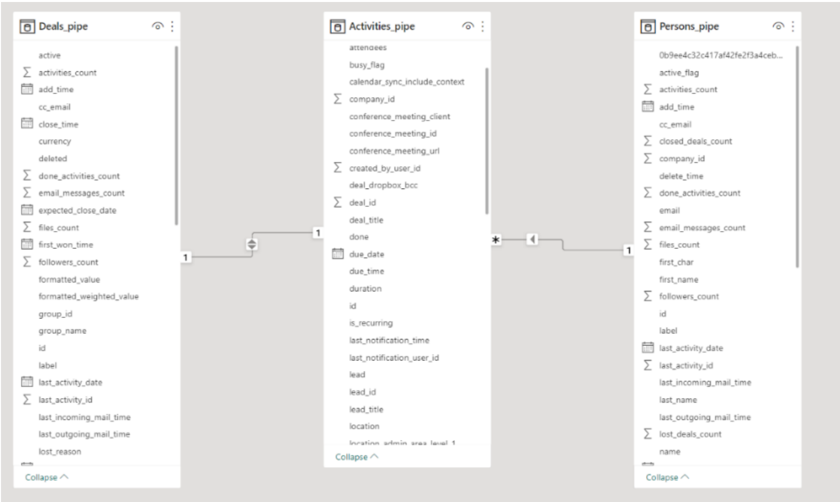
I therefore investigated the Pipedrive application and its API capabilities, I found the Pipedrive API documentation (developers.pipedrive.com, n.d.) and a GIT repository in relation to using PowerShell (Seidlm, 2021), connecting and retrieving data from the Pipedrive API. Effectively using PowerShell's Invoke-RestMethod which sends HTTP and HTTPS requests to Representational State Transfer (REST) web services which then returns data in a rich and structured format (sdwheeler, n.d.).

Together with the Business Development manager we added a custom contract start date and data was entered so the report could be generated in Power BI.

After that I created a PowerShell script that pulled all deals, activities and persons. Due to the data was paginated I had to do a loop to retrieve all data.

```
# Get All Deals
$num = 0
for ($num;; $num+=99)
{
        $Url_Deals=$Pipedrive_BaseURL+"deals?start=$num&api_token=$PipeDriveAPI"

        $Result_Deals=Invoke-RestMethod -uri $URL_Deals -Method GET

        $Result_Deals.data
        Write-Output "Page number" $num

        if ($null -ne $Result_Deals.data)
        {
            $Result_Deals.data | Export-Csv -Path C:\Users\aingelsten\scripts\Deals_pipe.csv -NoTypeInformation -Append
            Write-Output "****EXPORT****"
            Start-Sleep -Seconds 0.5
        }
        else {

        Write-Output "****LOOP ENDS***"
        Break

        }

    }
```

The data was then exported to Sharepoint online, imported to PowerBI and relationship setup between the 3 tables.



From this position we generated a report in Power BI that displays the contract value of open tenders with expected quarterly start dates as per below. Note data has been redacted due to they are commerically sensitive.

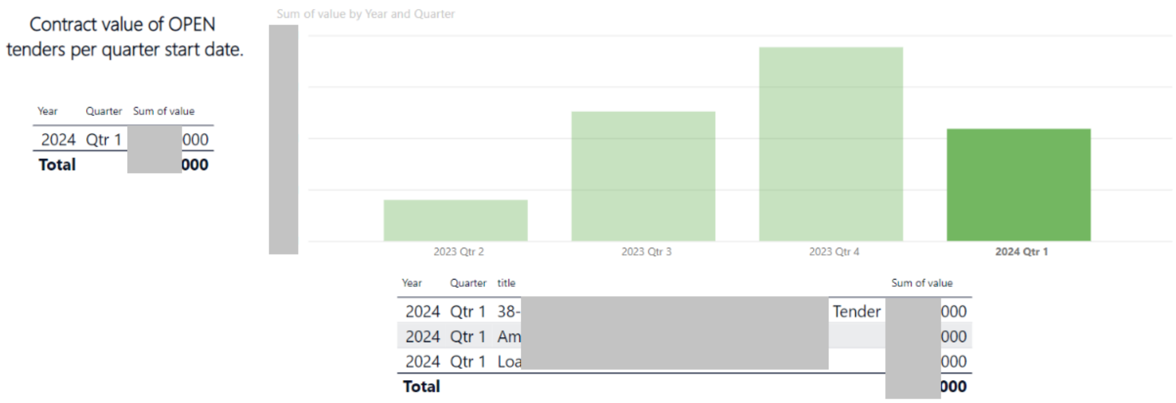


Figure 7: View of Values per Quarter

As an added bonus I also generated a report of the logged activates conducted by the business development manager since the inception of Pipedrive

Figure 6: Screenshot of imported tables in PowerBI

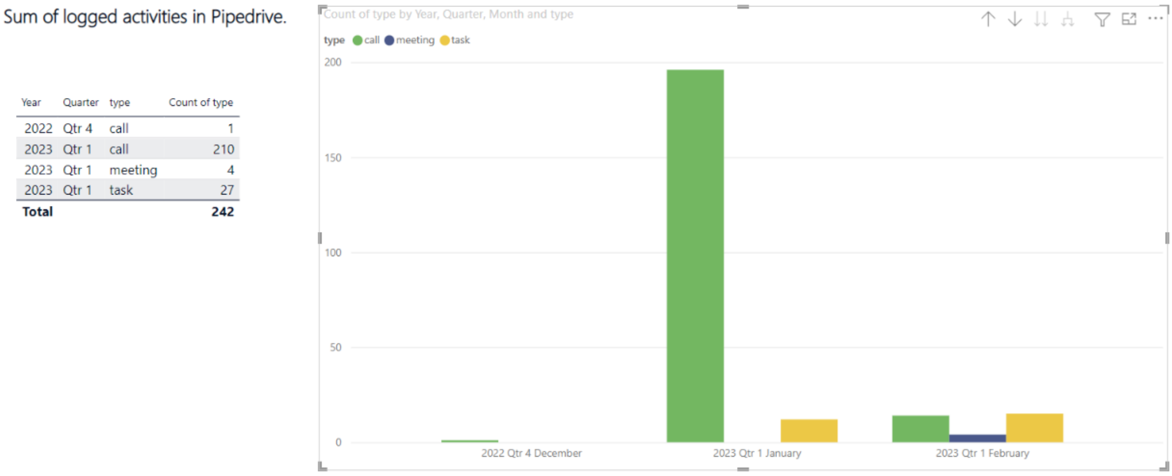


Figure 8: Report view of logged activities

The data for the Pipedrive views must at a minimum be updated monthly as per the KPI dates.

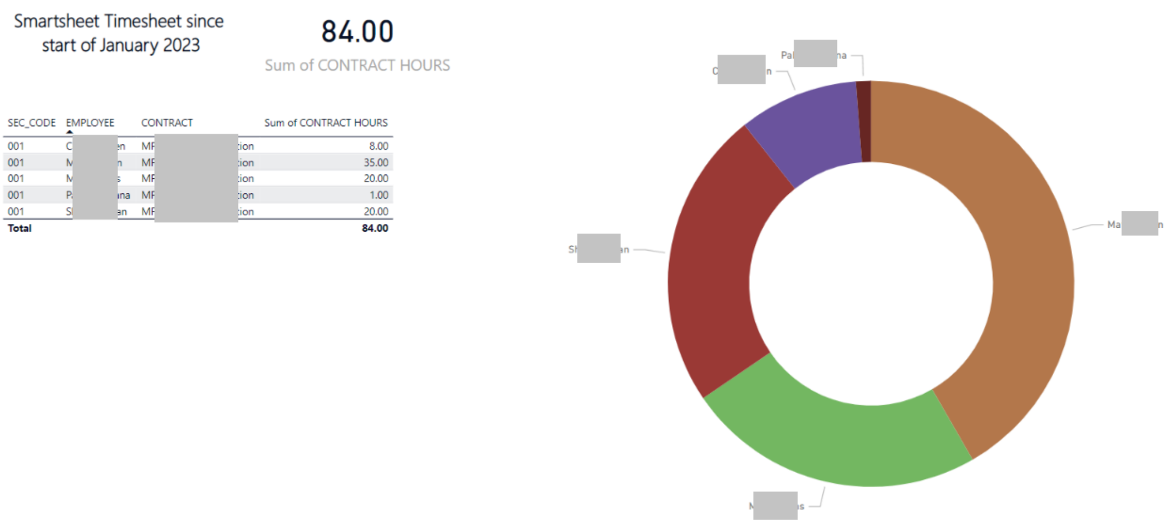
10.6 Phase 6: PowerBI Dashboard – SmartSheet Timesheet

Upon completion of the Pipedrive dashboard, my company supervisor asked me to prioritise a dashboard that displayed a summarised view of hours allocated per contract by user. Utilising the Smartsheet connector as rolled out during Phase 2, I could import the Timesheet table for office staff.

BER	WEEKDAY	EMPLOYEE	EMPLOYED ON PAYROLL	EMPLOYEE NUMBER	COINS COMPANY NUMBER	DATE	CONTRACT	CONTRACT HOURS	EM
2	Tuesday	M[redacted]n	[redacted]	[redacted]4	5	Tuesday 3 January 2023	M[redacted]on	9	L22
2	Tuesday	M[redacted]in	[redacted]	[redacted]1	5	Tuesday 3 January 2023	M[redacted]on	10	L22
2	Wednesday	M[redacted]in	[redacted]	[redacted]1	5	Wednesday 4 January 2023	M[redacted]on	9	L22
2	Tuesday	P[redacted]ana	[redacted]	[redacted]8	5	Tuesday 3 January 2023	M[redacted]	2.5	L36
2	Thursday	C[redacted]e	[redacted]	[redacted]6	5	Thursday 5 January 2023	M[redacted]	8.5	L36
2	Thursday	M[redacted]in	[redacted]	[redacted]1	5	Thursday 5 January 2023	M[redacted]on	9	L22
2	Friday	M[redacted]in	[redacted]	[redacted]1	5	Friday 6 January 2023	M[redacted]on	8.5	L22
2	Wednesday	M[redacted]n	[redacted]	[redacted]4	5	Wednesday 4 January 2023	M[redacted]on	9	L22
2	Thursday	M[redacted]n	[redacted]	[redacted]4	5	Thursday 5 January 2023	M[redacted]on	6	L22
2	Thursday	M[redacted]n	[redacted]	[redacted]4	5	Thursday 5 January 2023	M[redacted]on	4	L22

After that I generated a Donut chart of the summary of hours per contract. I also generated a corresponding table view and a card of the total sum of contracts

As per requirements from the company supervisor, the visual can be drilled down to contract, section code and employee level.



10.7 Phase 7: PowerBI Dashboard – Fieldview Timesheet

Next priority set by the company supervisor was to create a dashboard which displayed a summarized view of hours generated by site staff using the Fieldview application.

For this I had to use a different Fieldview API called GetQuestionAnswer. This API takes in the APItoken, FormID and the QuestionAlias and provides the answer and meta data of the answer.

The minimum required data was employee name, contract, date and hours, so I had to create four get requests and due to the API has an API quota of **10** I quickly realised to run an effective query I had to generate an API Token per API call.

```
$apiToken1 = Get-Content C:\Users\aingelsten\scripts\api_answer1.txt
$apiToken2 = Get-Content C:\Users\aingelsten\scripts\api_answer2.txt
$apiToken3 = Get-Content C:\Users\aingelsten\scripts\api_answer3.txt
$apiToken4 = Get-Content C:\Users\aingelsten\scripts\api_answer4.txt

$FVForms = New-WebServiceProxy -Uri "https://www.priority1.uk.net/FieldViewWebServices/WebServices/XML/API_FormsServices.asmx?WSDL"

$answer = "FVTimeSheet_Completed_By.csv"
$answer1 = "FVTimeSheet_Contract.csv"
$answer2 = "FVTimeSheet_Date.csv"
$answer3 = "FVTimeSheet_Hours.csv"
$questionAlias = "TS_Completed_By"
$questionAlias1 = "TS_Contract"
$questionAlias2 = "TS_Date_Timesheet"
$questionAlias3 = "TS_Hour_Calculation"
```

```
$formsQuestionAnswer = $FVForms.GetQuestionAnswer($apiToken1, $FormID, $questionAlias).FormAnswerInformation.childnodes

$formsQuestionAnswer | Add-Member -MemberType NoteProperty -Name "Form_Id" -Value $FormID

$formsQuestionAnswer | Export-Csv -Path C:\Users\aingelsten\scripts\$answer -Append -NoTypeInformation

Write-Output $formsQuestionAnswer
```

From here I ensured the script generated four CSV files, one per answer and the script exported the files into SharePoint Online.

```
#Configuration of Sharepoint Variables
$SiteURL = "https://[redacted].sharepoint.com/sites/[redacted]"
$SourceFilePath ="c:\Users\aingelsten\scripts\$answer"
$SourceFilePath1 ="c:\Users\aingelsten\scripts\$answer1"
$SourceFilePath2 ="c:\Users\aingelsten\scripts\$answer2"
$SourceFilePath3 ="c:\Users\aingelsten\scripts\$answer3"
$DestinationPath = "Kpi_Data" #Site Relative Path of the Library
$ClientId = "[redacted]"
$ClientSecret ="[redacted]"

#Connect to SharePoint Online with ClientId and ClientSecret
Connect-PnPOnline -Url $SiteURL -ClientId $ClientId -ClientSecret $ClientSecret

#Powershell pnp to upload file to sharepoint online
Add-PnPFile -Path $SourceFilePath -Folder $DestinationPath
Add-PnPFile -Path $SourceFilePath1 -Folder $DestinationPath
Add-PnPFile -Path $SourceFilePath2 -Folder $DestinationPath
Add-PnPFile -Path $SourceFilePath3 -Folder $DestinationPath

Write-Output "Process Answers Completed"
```

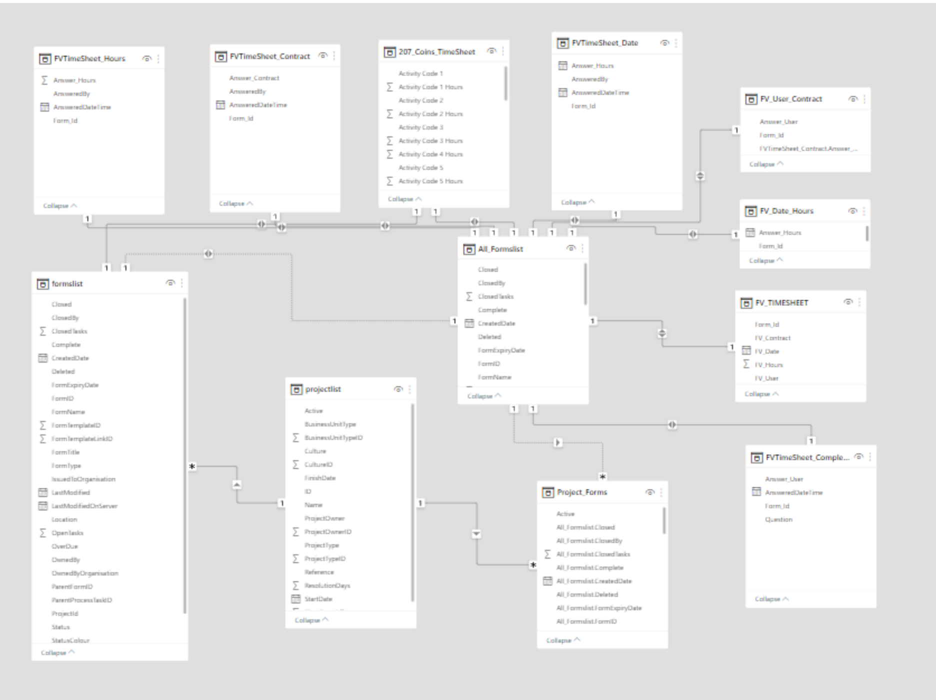
After the 4 answers from Fieldview had been imported into PowerBI, all the 4 tables were joined by their common identifier: the FormID.

After analysing the data and outputs I noticed an issue in relation to amount of generated answer data for the question date. The query generated an answer for every form that has a question alias with the word “date”. Note there are currently over 120 active forms in the Fieldview system. After realising this, I renamed the Timesheet alias in Fieldview to "TS_Date_Timesheet". As a precaution I also updated the aliases for the other three questions. This resolved the issue.

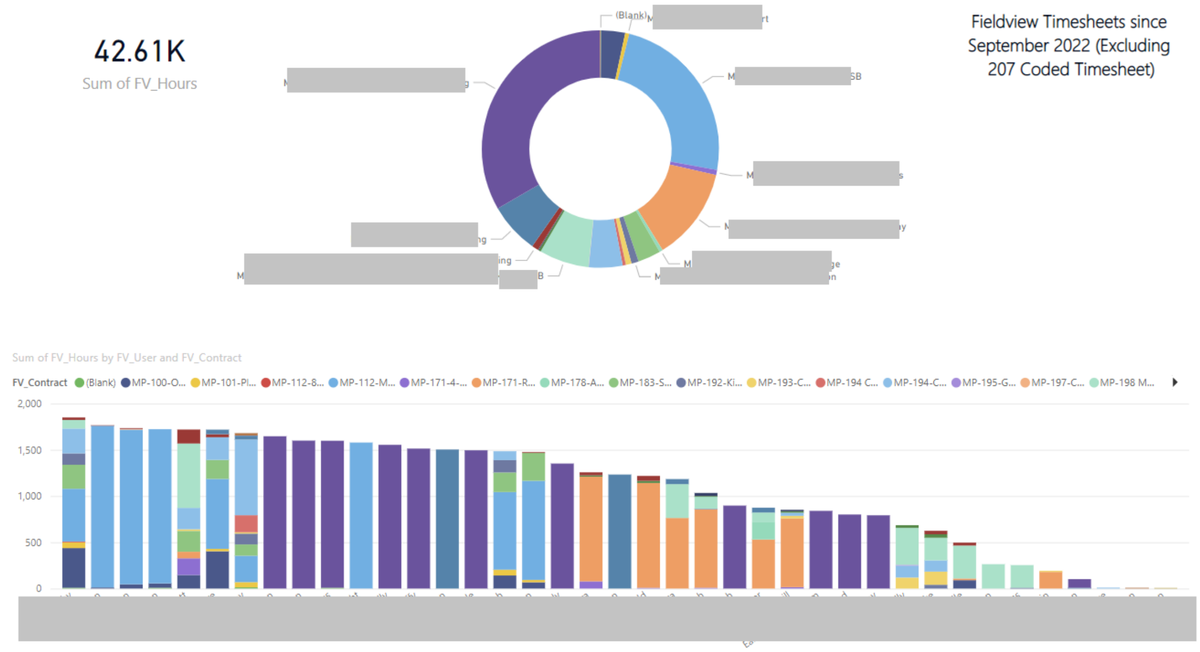
To be able to generate the required visualisations, I proceeded and utilising the PowerBI modelling tool where I created a User/Contract table, a Date/Hours table and from these tables I generated a combined Timesheet table that allowed me to generate the visualisation.

Form_Id	FV_User	FV_Contract	FV_Date	FV_Hours
F207746.157	[redacted]n	M[redacted]er Reading	*Friday 29 April 2022*	*8.5*
F207746.159	[redacted]n	M[redacted]er Reading	*Thursday 5 May 2022*	*8.5*
F207746.160	[redacted]n	M[redacted]er Reading	*Friday 6 May 2022*	*8.5*
F207746.162	[redacted]n	M[redacted]er Reading	*Monday 9 May 2022*	*8.5*
F207746.163	[redacted]n	M[redacted]er Reading	*Tuesday 10 May 2022*	*8.5*
F207746.164	[redacted]n	M[redacted]er Reading	*Wednesday 11 May 2022*	*8.5*
F207746.165	[redacted]n	M[redacted]er Reading	*Thursday 12 May 2022*	*8.5*

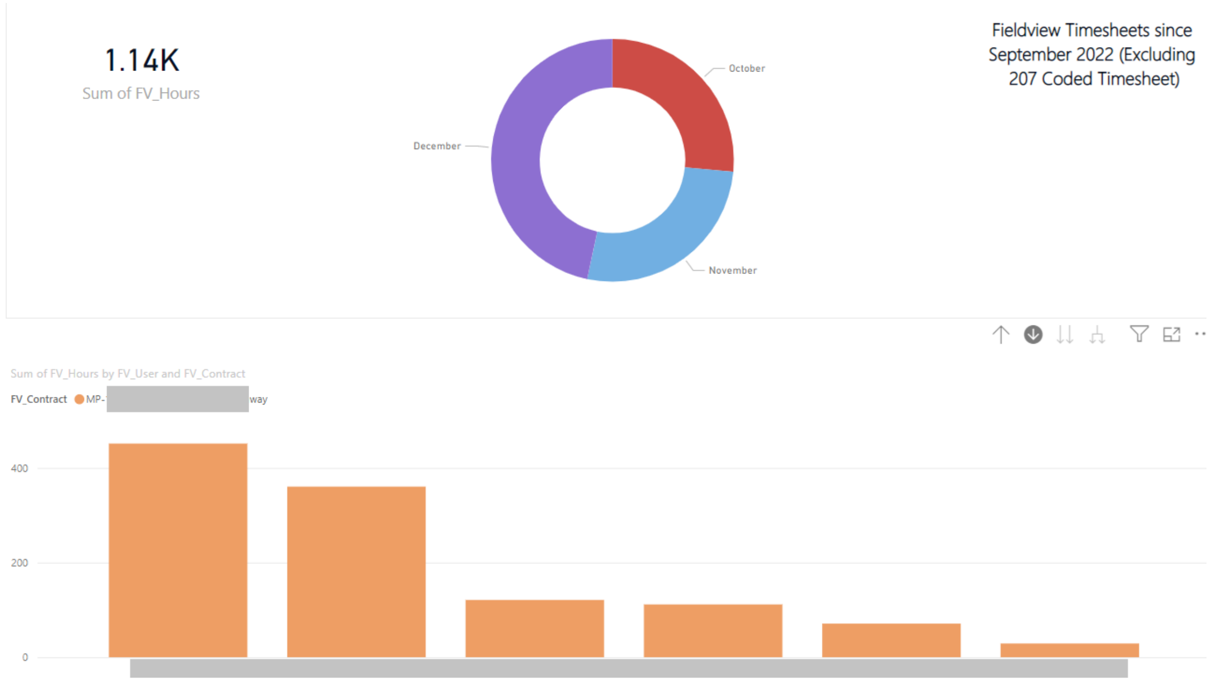
See view below of all the Fieldview table and their relationships in PowerBI.4



The visualisation requested is the following, a card with total summary of all hours, a donut chart of displaying the breakdown of summarised hours per contact and a stacked column chart per employee. The views allow for drill down per person.



Total summary view.



View per contract, month and person.

10.8 Phase 8: PowerBI Dashboard – Asset Manager

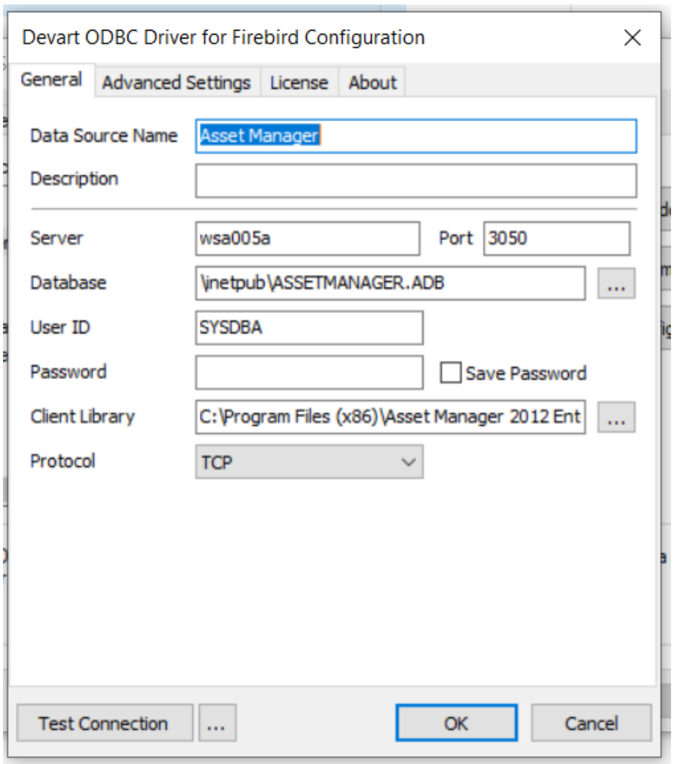
This phase was added as I was asked by the Company's project supervisor to investigate if I can connect Asset Manager and display's KPI report visualistations for the company's plant manager.

Asset Manager is a 3rd party software developed by Kaizen Software Solutions and is utilised by the company to track all fixed and plant assets (www.kzsoftware.com, n.d.). The system holds asset details, value but also service and certification records. And the plant manager needed assistance to much easier see upcoming service and certifications renewal dates as part of his KPI reuirementrs.

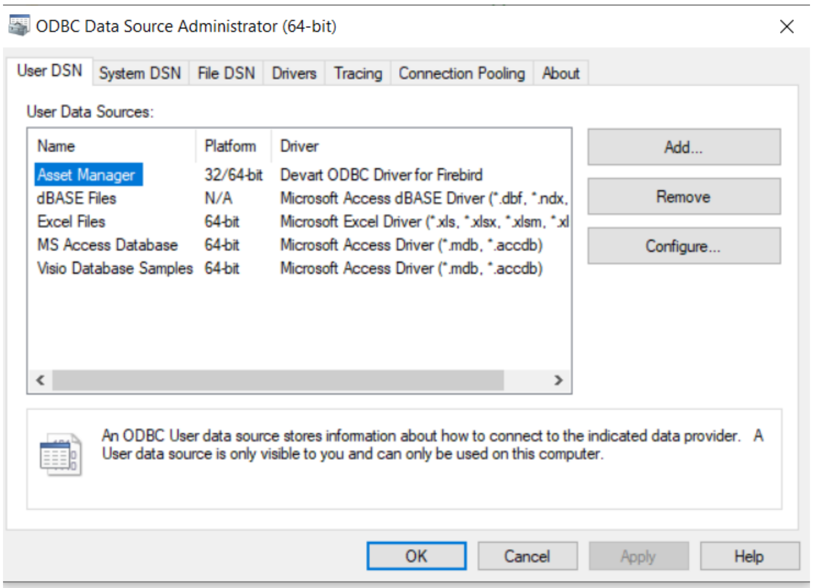
Asset Manager backend database is of type Firebird. Which is an open-source SQL relational database management system that supports Linux, Microsoft Windows, macOS and other Unix platforms Home. (n.d.).

PowerBI has no built in connector to Asset Manager, but after a bit of research I found a software company Devart who provides an ODBC driver that allows Power BI to Firebird via ODBC (docs.devart.com, n.d.).

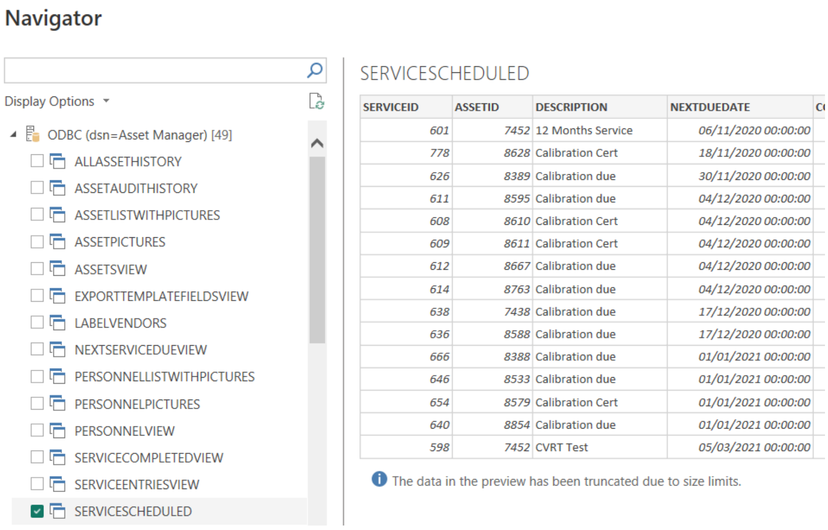
I then progressed and installed the driver, and configured the connection, and used the Power BI ODBC connection to bring in the information, see screenshot samples below.



View of config



View of connected Data Source



View of Tables available from Asset Manager.

From here I brought in the Servicesheduled table, after analysing the data I noticed that the table has every service record done for every plant item. The unique identifier for data is the Service ID. To be able to provide a view of upcoming service dates for the plant manager. I generated a DAX query table of the latest Service ID per vehicles. From here I then did a join with the existing table and could from here then generate the relevant views, see screenshots below.

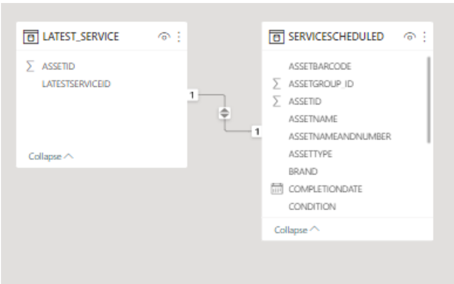
```
LATEST_SERVICEID =
VAR __ASSETID = SERVICESCHEDULED[ASSETID]
VAR __lastSERVICE =
    CALCULATE(
        MAX( SERVICESCHEDULED[SERVICEID] ),
        FILTER( SERVICESCHEDULED, SERVICESCHEDULED[ASSETID] = __ASSETID )
    )
Return __lastSERVICE
```

COMPLETIONDATE	ASSETBARCODE	ASSETNAME	STATUS	ASSETTYPE	LOCATION	MANUFACTURER	MODEL	LATEST_SERVICEID
10/06/2019 14:34:25	AST0004064	B2 HVA 60 PD/TD&VLF Tester	In Storage	Test Equipment	MP Listowel		HVA 60 PD/TD/VLF	544
24/01/2022 11:14:57	AST0004066	Sverker 750 Relay Tester	In Use	Test Equipment	MP Listowel	Megger	750	1222
03/01/2023 12:38:55	AST0004066	Sverker 750 Relay Tester	In Use	Test Equipment	MP Listowel	Megger	750	1222

Query of the latest service ID

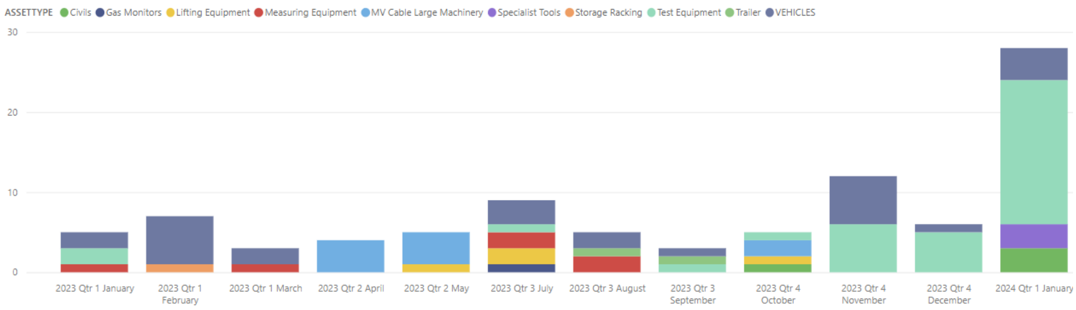
```
LATEST_SERVICE = SUMMARIZE(SERVICESCHEDULED, SERVICESCHEDULED[ASSETID], "LATESTSERVICEID",MAX (SERVICESCHEDULED[SERVICEID]))
```

Table query of the latest service ID



View of the joined table relationship.

The visualization displayed a stacked column table with upcoming plant items, and by utilising the drill down the user can see a list of the Plant items to be serviced per quarter.



ASSETBARCODE	ASSETNAME	ASSETTYPE	CONDITION	DESCRIPTION	LOCATION	MANUFACTURER	MODEL	
AST0008597	Megger 5kV Insulation Tester	Test Equipment	New	Calibration Cert	MP Dublin	Megger	MIT515	In Storage
AST0008422	10 Tonne Jakob Thaler Winch	MV Cable Large Machinery	Good	Calibration Cert	MP Listowel	Jakob Thaler		In Use
AST0007440	4 Tonne OMAC F275.P.40 Winch	MV Cable Large Machinery	Good	12 Months Service	MP Listowel		F275.P.40	In Use
AST0007451	5 Tonne Track Dumper	MV Cable Large Machinery	Good	Calibration Cert	MP Cork		C50R-2	In Use
AST0007749	8 Tonne Drum Carrier	MV Cable Large Machinery	Fair	GA 1 Examination	MP Listowel			In Use
AST0009434	Audi E-Tron 201C9414	VEHICLES	New	Tax	Mainline Place		E-Tron	In Use
AST0007759	C.Scope Cable avoidance tool	Test Equipment		Calibration Cert	MP Listowel		33 Khz	In Use
AST0008663	CAT	Measuring Equipment	New	Calibration due	MP Listowel	C. Scope	CLX4	In Use
AST0008487	CAT 107274	Measuring Equipment	Good	Calibration Cert	MU Cork	C Scope		In Use
AST0007661	CAT 184589	Measuring Equipment	Good	Calibration Cert	MP Listowel			In Use
AST0007745	Clydesdal Stringing Winch	MV Cable Large Machinery	Good	GA 1 Examination	MP Listowel	Clydesdal		In Use
AST0007746	Clydesdal Stringing Winch no. 2	MV Cable Large Machinery	Good	GA 1 Examination	MP Listowel	Clydesdal		In Storage
AST0008629	Crowcon Gas Monitor	Gas Monitors	New	Calibration Cert	MP Dublin	Crowcon		In Use
AST0008775	CXL4 Cable Avoidance Tool CAT	Test Equipment	New	Calibration Cert	MP Listowel	C Scope	CXL4CAT	In Use
AST0008389	Dillon ED Junior Dynamometer Load Limit	Lifting Equipment	Good	Calibration Cert	MP Listowel	Dillon		In Storage
AST0008304	Dillon EDx-10T Dynamometer Weighing Measure	Lifting Equipment	New	Calibration Cert	MP Listowel	Dillon EDXtreme	EDX-10T	In Storage
AST0008398	EA Technology Ultra Tev Plus	Test Equipment	Good	Calibration Cert	MP Listowel	EA Technology	EA Technology Ultra Tev Plus	In Storage

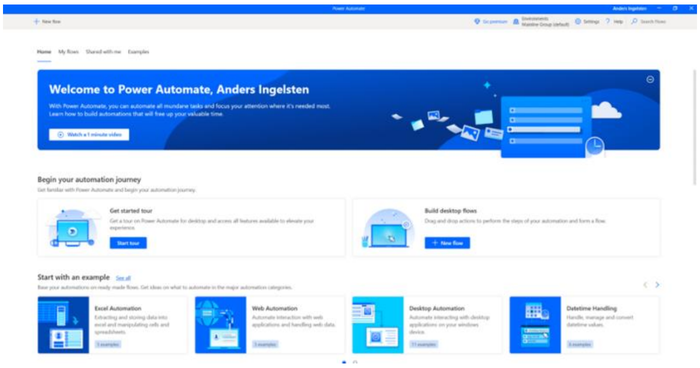
View of assetypes and corresponding table.

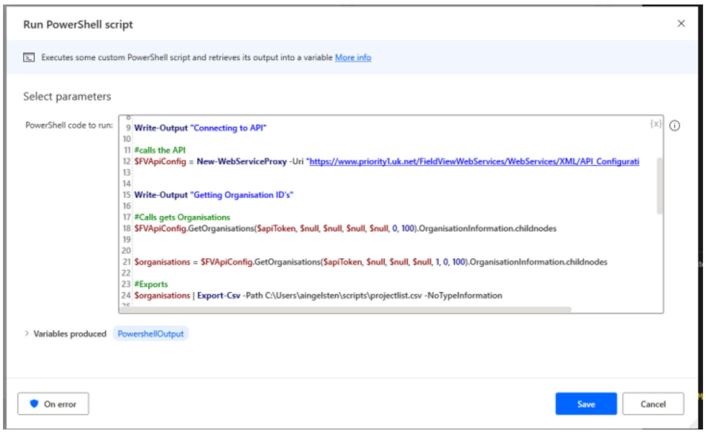
10.9 Phase 9: Automation of Data flows and Power BI Refresh

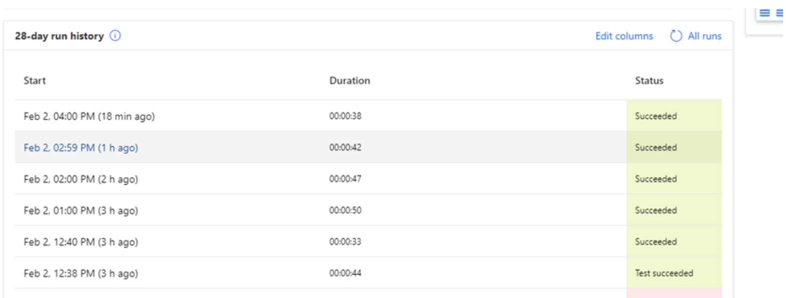
Initial setup and testing of the process was done in February. Automation testing (www.youtube.com, n.d.)

Config file

scripting - How to use a config file (ini, conf,...) with a PowerShell Script? - Server Fault







11 Future Development work

Over the course of the project, I have identified three areas of future development work.

- Further additions are features, in relation to data queries of existing systems or systems that not yet has been logged in to the power BI platform. For example, the ERP system and the HR management system.
- Development in relation to business logic and rules around them and for example just simple changes or updates when it comes to automation and updating periods stop.
- Data persistence saving data as CSV fights SharePoint online might be a solution for the immediate future, but overtime other database solutions might be developed for example Azure SQL Database.

12 Project Evaluation

12.1 Self-Reflection

The first time I showed the first iteration of a published dashboard of real company timesheet data to the company's CFO the reaction was immediate. The eyes lit up, the penny had dropped, the feedback was immensely positive and new requirements was expressed. *"Can we show this per month per person instead, and what about this, can we compare with previous month".* Pandoras box had been opened and there was no stopping now.

From a person standpoint, on several occasions during this project several thoughts kept re-occurring in my mind, these self-reflecting thoughts spanned across various emotions but to highlight the most descriptive ones:

> *"I couldn't have done this 2 years ago".*
> *"It works! - look at all that data coming in".*
> *"Is this right? Is the tool and methods I used, correct and stable?"*

Especially when I'm the sole responsible person for the project, and the company supervisor provides no direction in relation to preferred techniques or methods, one has look at the surrounding conditions, environment, and cost. From this then decide and proceed down a chosen path, deal with problems you encounter and solve them as you go. This process is where I learned the most and where the acquired knowledge from the course assist you to try to decide.

Furthermore, the expression Garbage in, garbage out was very relevant in this project. Both from the perspective of understanding the data and that the reports displayed in Power BI was true, to also seeing samples of uncomplete or "deleted" data and how to apply the appropriate business logic. It's worth noting that in Fieldview you cannot delete data, it is only labelled deleted with a flag of True or False.

But in the end, I did enjoy the whole process of exploration, research, implementation and seeing the result which is benefiting the reporting process of the business.

12.2 Knowledge attained.

12.3 Achievements

12.4 Future Learning

I have identified I need further education and self-development in the following areas.

- The PowerBI platform and its Data Queries
- DAX scripting language
- Sharepoint online management shell
- Microsoft Data verse business layer

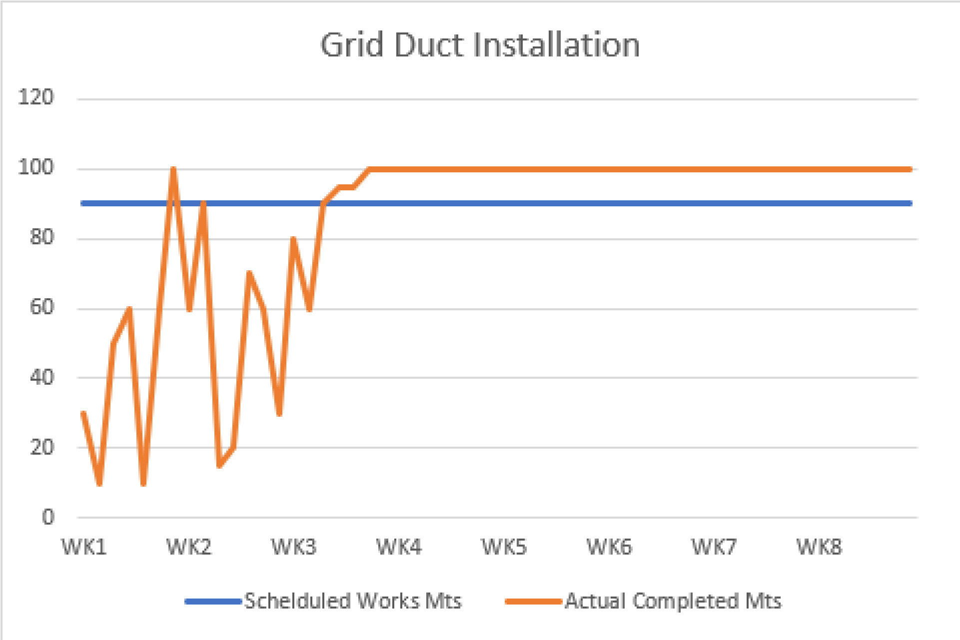
12.5 Future Work

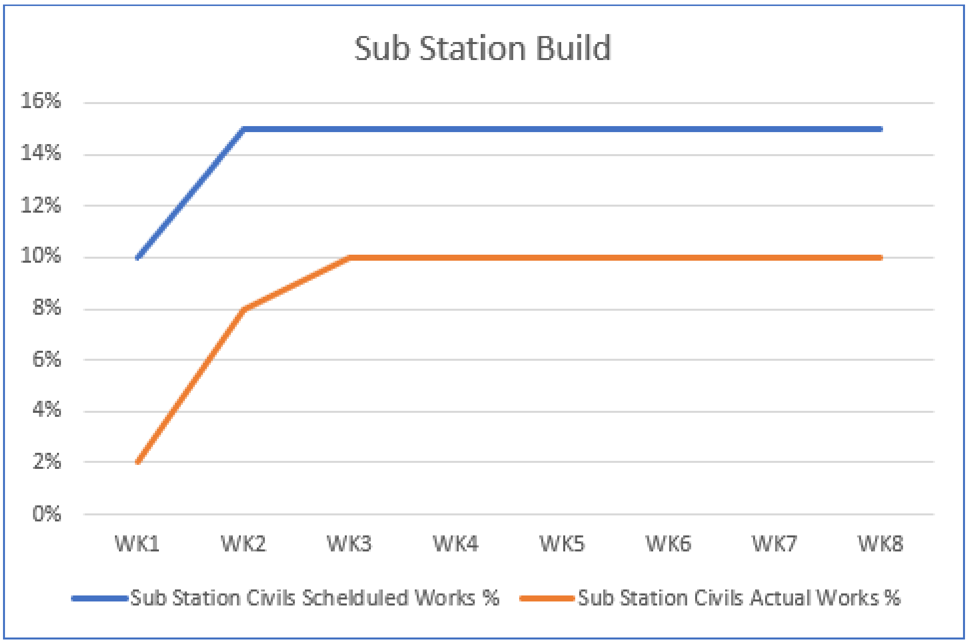
During one of our regular project meetings my supervisor turned to me and said, *"Best thing to do at this moment in time is to shut up and not to tell anyone else about this and remember look after our team first".* The meaning of this was twofold, one was not allowing the project scope to creep, and secondly if I would have allowed the rest of the organisation know of the capabilities, we had in hand that there was a chance I would get inundated with data query request from all parts of the organisation.

Below follows a typical request received during the project development process. Due to Company A is in the utilities/construction business, it would be of great value for a project manager, contract manager and the commercial team to know are the project or a part of project, for example a duct excavation on target as per the project plan.

"Hi Anders,

Not sure if it's possible, maybe it's there, I was wondering if it were possible to create a Table on View Point where a person could just fill in a Tab showing a Weekly Progress Report say on the Meterage of Grid Duct Trenching / Installation, that may run an excel file in the background which would show for instance the agreed amount per week which is a fixed point & the other the actual, Or do we have something else set up, as it would be great to track, say cable installed, joints made off etc"





Chainage control per project

13 References

Smartsheet (2019). Smartsheet: Less Talk, More Action. [online] Smartsheet. Available at: https://www.smartsheet.com/.

Floor, T.V.U.O. 4th, Square, C., Tyne, F.S.N.U. and Kingdom +44.0800.048.8152, N. 3PJ U. (n.d.). *Cloud-Based Field Construction Software*. [online] Trimble Viewpoint. Available at: https://www.viewpoint.com/en-gb/products/viewpoint-field-view.

www.microsoft.com. (n.d.). SharePoint, Team Collaboration Software Tools. [online] Available at: https://www.microsoft.com/en-ie/microsoft-365/sharepoint/collaboration.

Microsoft (2022). Data Visualisation | Microsoft Power BI. [online] powerbi.microsoft.com. Available at: https://powerbi.microsoft.com/en-gb/.

AltexSoft. (n.d.). What is SOAP: Formats, Protocols, Message Structure, and How SOAP is Different from REST. [online] Available at: https://www.altexsoft.com/blog/engineering/what-is-soap-formats-protocols-message-structure-and-how-soap-is-different-from-rest/.

sdwheeler (n.d.). PowerShell Documentation - PowerShell. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/powershell/.

powerautomate.microsoft.com. (n.d.). Power Automate | Microsoft Power Platform. [online] Available at: https://powerautomate.microsoft.com/en-gb/.

georgiostrantzas (n.d.). Scripting actions reference - Power Automate. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/power-automate/desktop-flows/actions-reference/scripting.

Minewiskan (n.d.). Data Analysis Expressions (DAX) Reference - DAX. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/dax/.

www.microsoft.com. (n.d.). Task Management Kanban Solution for Teams | Microsoft Planner. [online] Available at: https://www.microsoft.com/en-ie/microsoft-365/business/task-management-software.

help.viewpoint.com. (n.d.). Viewpoint Help. [online] Available at: https://help.viewpoint.com/en/viewpoint-field-view/field-view/api-documentation/apis.

ZappySys (2018). Calling SOAP API in Power BI (Read XML Web Service data). [online] ZappySys Blog. Available at: https://zappysys.com/blog/call-soap-api-power-bi-read-xml-web-service-data.

Technologies, S.P. (n.d.). Spanish Point Technologies. [online] Available at: https://www.spanishpoint.ie.

Rajack, S. (2018). How to Connect to SharePoint Online using PnP PowerShell? [online] SharePoint Diary. Available at: https://www.sharepointdiary.com/2018/03/connect-to-sharepoint-online-using-pnp-powershell.html.

www.youtube.com. (n.d.). Power Automate Desktop || How to schedule Desktop Flows? [online] Available at: https://www.youtube.com/watch?v=IJ1I2737JWU.

Pipedrive Inc / Pipedrive OÜ (2018). Sales CRM & Pipeline Management Software. [online] Pipedrive. Available at: https://www.pipedrive.com/.

developers.pipedrive.com. (n.d.). Pipedrive API v1 Reference. [online] Available at: https://developers.pipedrive.com/docs/api/v1 [Accessed 19 Feb. 2023].

Seidlm (2021). PIPEDRIVE-PowerShell/1-Basic Example.ps1 at main · Seidlm/PIPEDRIVE-PowerShell. [online] GitHub. Available at: https://github.com/Seidlm/PIPEDRIVE-PowerShell/blob/main/1-Basic%20Example.ps1 [Accessed 19 Feb. 2023].

sdwheeler (n.d.). Invoke-RestMethod (Microsoft.PowerShell.Utility) - PowerShell. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/invoke-restmethod?view=powershell-7.3 [Accessed 19 Feb. 2023].

Home. (n.d.). Firebird: The true open source database for Windows, Linux, Mac OS X and more. [online] Available at: https://firebirdsql.org/en/start/ [Accessed 20 Feb. 2023].

Health and Safety Authority. (n.d.). Managing Driving for Work Information Sheet. [online] Available at: https://www.hsa.ie/eng/vehicles_at_work/driving_for_work/employer_responsibilities_/ [Accessed 25 Feb. 2023].

Rad, R. (2020). Replace BLANK with Zero in Power BI Visuals Such as Card. [online] RADACAD. Available at: https://radacad.com/replace-blank-with-zero-in-power-bi-visuals-such-as-card [Accessed 25 Feb. 2023].

www.kzsoftware.com. (n.d.). Asset Management Software - A Fixed Assets Register. [online] Available at: https://www.kzsoftware.com/products/asset-management-software/ [Accessed 4 Mar. 2023].

Home. (n.d.). Firebird: The true open source database for Windows, Linux, Mac OS X and more. [online] Available at: https://firebirdsql.org/.

docs.devart.com. (n.d.). Connecting Power BI to Firebird via ODBC Driver. [online] Available at: https://docs.devart.com/odbc/firebird/powerbi.htm [Accessed 4 Mar. 2023].