



KPI PROCESS IMPROVEMENT

Interim Report

Anders Ingelsten 20095402

Higher Diploma in Science in Computer Science

Contents

Summary	2
Introduction	4
Project Type	4
The organisation	4
System Background and Project Scope.....	4
Smartsheet.....	5
Fieldview	6
Use Case	7
Current Situation.....	7
What is the problem.	7
Potential Technologies, Tools, and Languages	8
Potential Issues.....	8
Objectives	8
Methodology.....	9
Dataflow	9
Modelling	9
Feedback Loop	9
Technologies	10
Sharepoint/Office365	10
PowerBI	10
SOAP API	10
PowerShell	11
Power Automate	11
Potential Disadvantages.....	11
Project Plan	11
Project Planner.....	12
Implementation	13
Phase 1: Research & Training.....	13
Phase 2: Smartsheet Data Connector setup	15
Phase 3: FV API setup and Data Persistence.....	17
Data persistence	19
Phase 4: PowerBI Dashboard – Fieldview Vehicle-check.....	20

Phase 5: PowerBI Dashboard – Fieldview Form Count.....	20
Phase 6: PowerBI Dashboard – Fieldview Timesheet.....	20
Phase 7: PowerBI Dashboard – SmartSheet Timesheet	20
Project Evaluation	20
Conclusion.....	20
Reflection	20
Challenges	20
Future Work	20
References	20

Summary

Last Text to add

Introduction

Project Type

This is a work-based project, where 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 as part of my day-to-day work.

The organisation

The company 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.

System Background and Project Scope

Mainline Group 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.

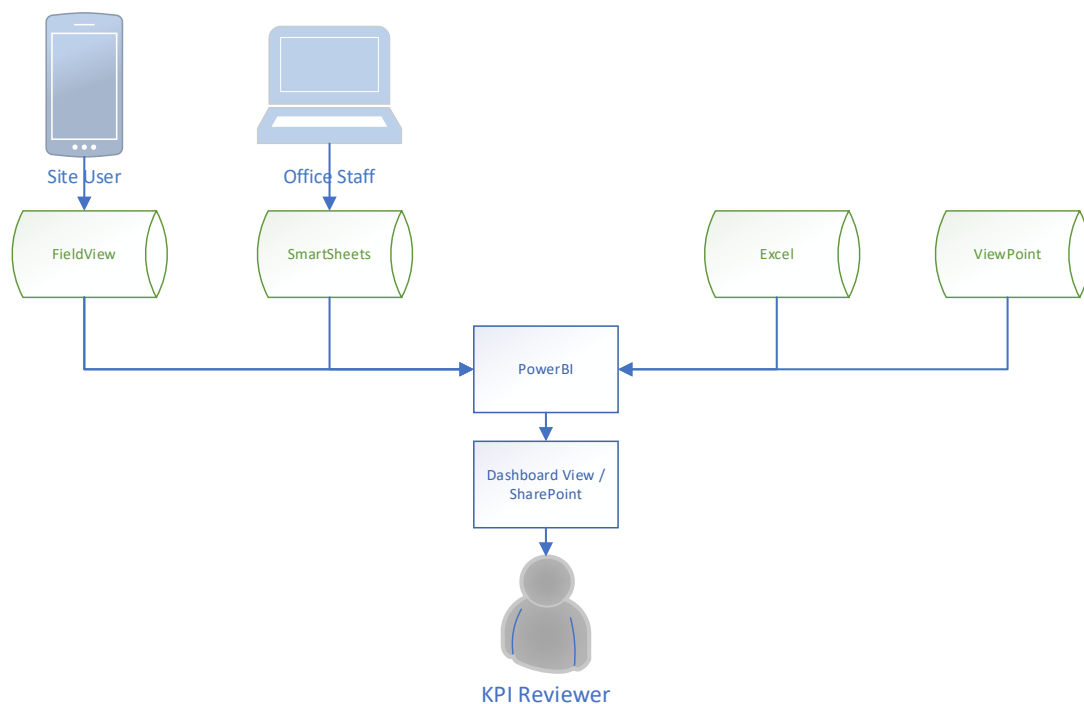


Figure 1: Overview of existing potential systems available to the project

The current systems identified for the initial project scope are:

System	Function	Location	Has API	Users
Fieldview	Record QA/QC/Admin forms and Timesheet for site staff	Cloud	Yes - JXML	On-Site Staff / Operations
Smartsheet	Record timesheets for Salaried Staff	Cloud	Yes	All management & admin staff
Office365	SharePoint, Excel, word etc	MS Cloud	Yes	All -as per assigned rights

Reworks and Expand a bit more in these
Smartsheet

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

Timesheet 4.0
Section and activity based form

DATE *

EMPLOYEE *

CONTRACT *

SECTION_2

ACTIVITY_1

CONTRACT HOURS *

COMMENT

☐ Send me a copy of my responses

Submit

Powered by smartsheet
[Privacy Notice](#) | [Report Abuse](#)

Figure 2: Sample view of Smartsheet form

Fieldview

Fieldview is a third-party cloud-based and off-line mobile solution developed by Trimble. It is used in Mainline 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.

← F215017.36 - Coins TimeSheet - 14/1/2023 All Good Distribute View Report

Owned By: Anders Ingelsten Location: MP-General Date Raised: 14/1/2023, 06:35

Copy Opened

Completed By *

Contract *

Date *

Figure 3: Sample view of Fieldview mobile form

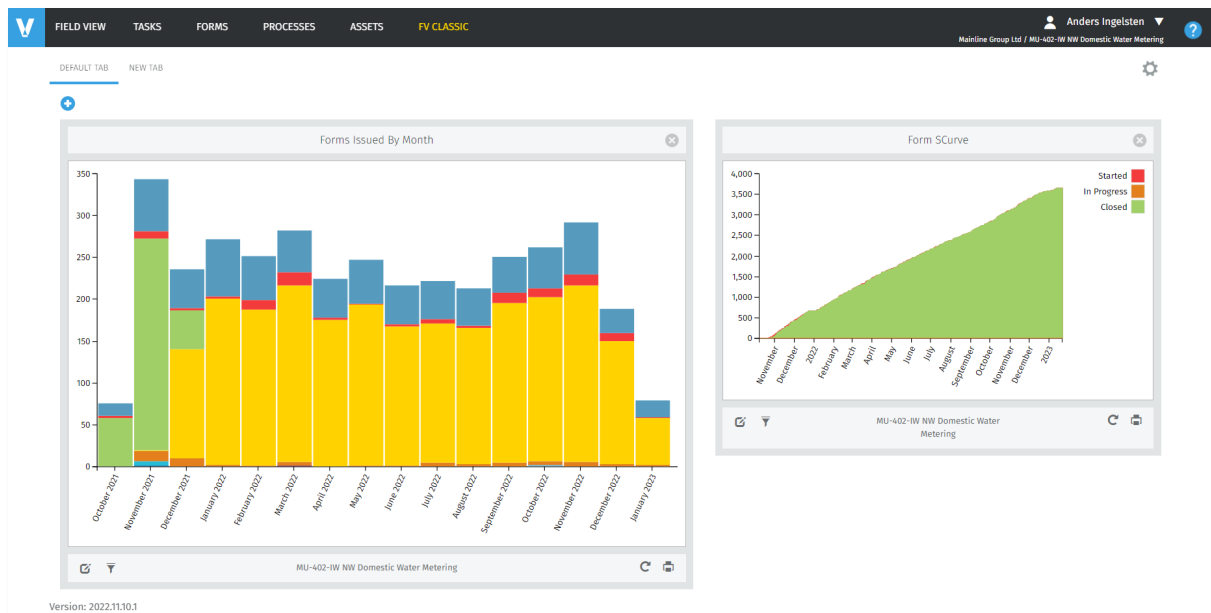


Figure 4: Sample view of the Fieldview widget reports

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.

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 Mainline Group is therefore looking for a solution that can display KPI information from systems on the internal company SharePoint internet website.

Use Case

The primary reason for developing a dashboard pulling data from sources is the time saving. For example, the Fieldview application is structured 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. $1.5 \text{ hours} \times 48 \text{ working weeks} = 72 \text{ Hours}$ which equates to 9 working days per year.

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

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.

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.

Methodology

Dataflow

Text about data flow and why

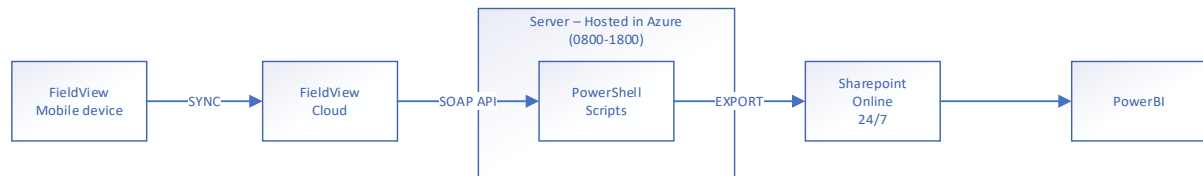


Figure 5: The Flow of Data from Fieldview to PowerBI

Modelling

Text about the data model

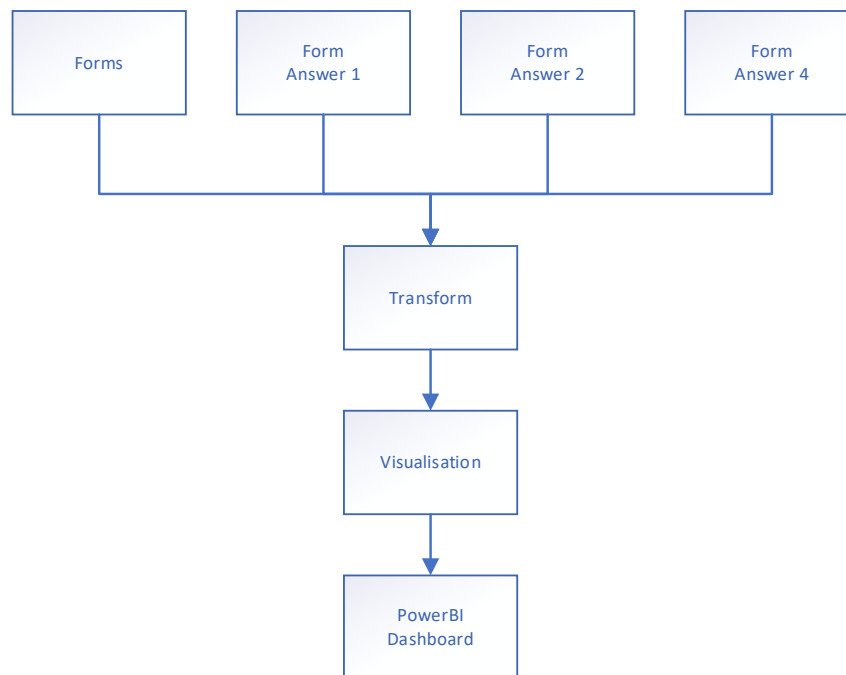


Figure 6: Sample Representation of the data model in PowerBI

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.

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 choice made of which technology is used.

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. All data and software are hosted on Microsoft own servers. Access, storage, and use of software is done by using a subscription model.

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. 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.

SOAP API

Fieldview uses SOAP API. SOAP is an acronym for Simple Object Access Protocol, 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.

PowerShell

During the research process several scripting languages was considers 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. It supports variables, functions, branching (if-then-else), loops (while do, for, and foreach) and structured error/exception handling and closures/lambda expression.

Power Automate

[Text here.](#)

Potential Disadvantages

[Text here.](#)

Project Plan

The milestones to achieve are outlined below.

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

The project will span over the following development activities with preliminary dates

Phase	Preliminary Sprint Due Dates
Research & Training	8 th January 2023
Data Connector setup – Smartsheet	15 th of January 2023
FV API setup and Data Persistence	29 th of January 2023
PowerBI Dashboard – Fieldview Vehicle-check	12 th of February 2023
PowerBI Dashboard – Fieldview Form Count	25 th of February 2023
PowerBI Dashboard – Fieldview Timesheet	19 th of March 2023

PowerBI Dashboard – SmartSheet Timesheet	26 th of March 2023
Project Wrap Up	2 nd April 2023

Project Planner

The Microsoft Planner tool was used for planning and execution of the project. In planner the user can divide development phases also known as sprints into buckets. Inside the buckets the user would then add tasks into it.

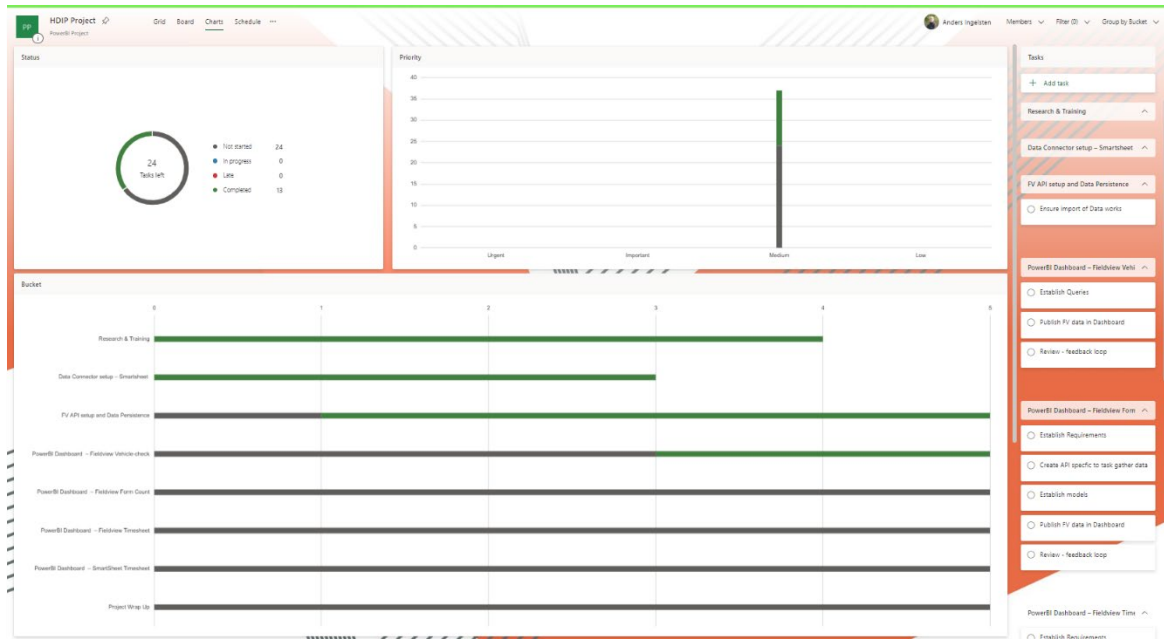


Figure 7: Microsoft Planner - Status of tasks and Buckets

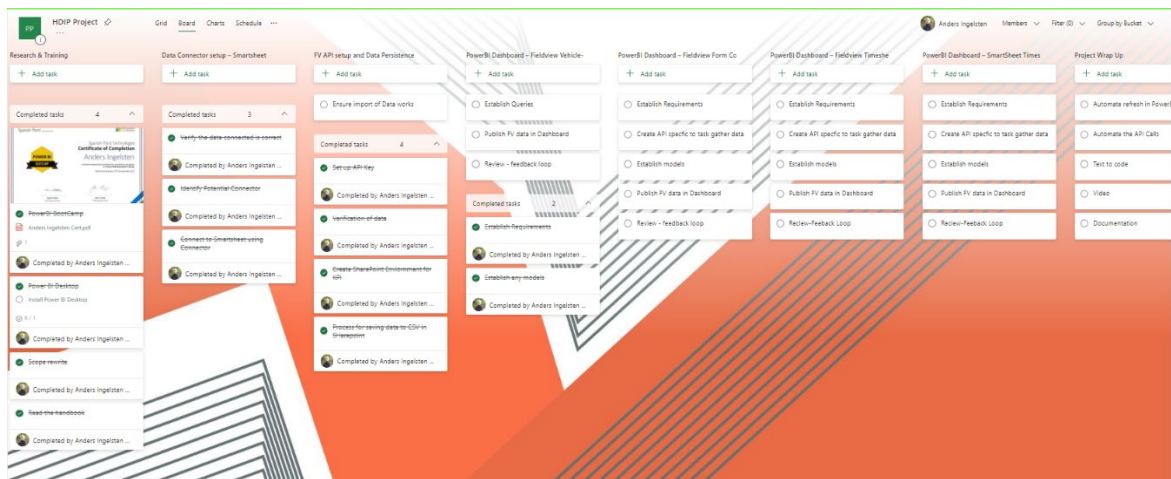


Figure 8: Microsoft Planner - Status of tasks and Buckets

Implementation

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, 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 Zealand

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

Every Region has twelve APIs in two set of six different API. 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 9: 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 10: SOAP request and response

```
<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>
</FormAnswerResponse>
```

Figure 11: 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 12: 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. Source [Calling SOAP API in Power BI \(Read XML Web Service data\) | ZappySys Blog](#)

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. Mainline 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

Add source. [Spanish Point Technologies Ltd.](#)

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

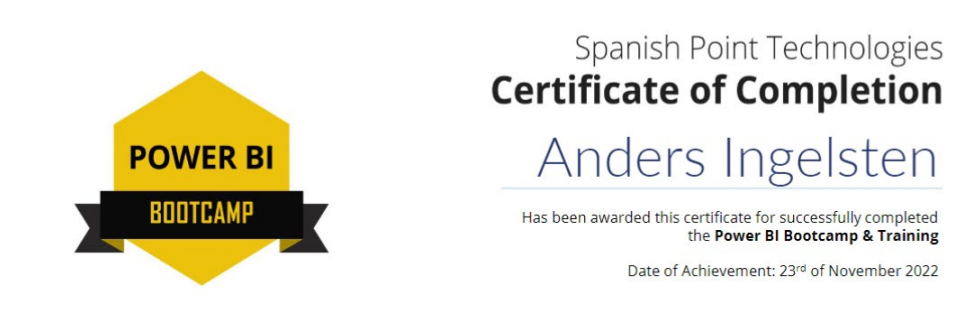


Figure 13: Certificate of Completion

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.

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

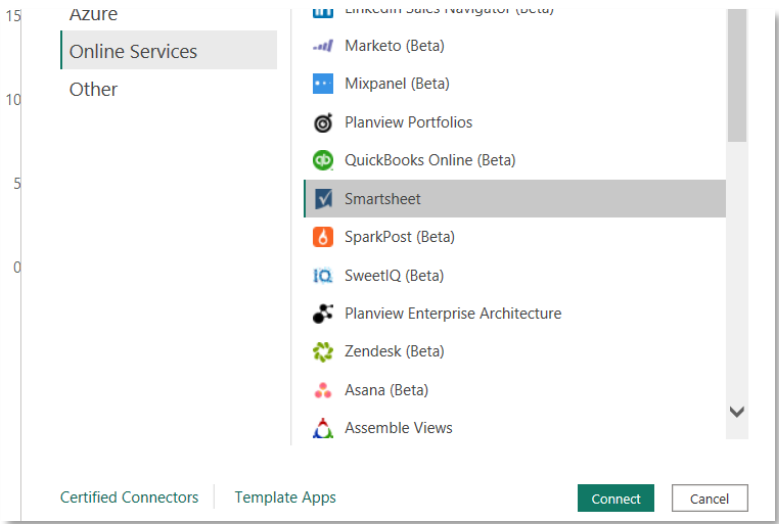


Figure 14: Get Data view in PowerBI of Online Services

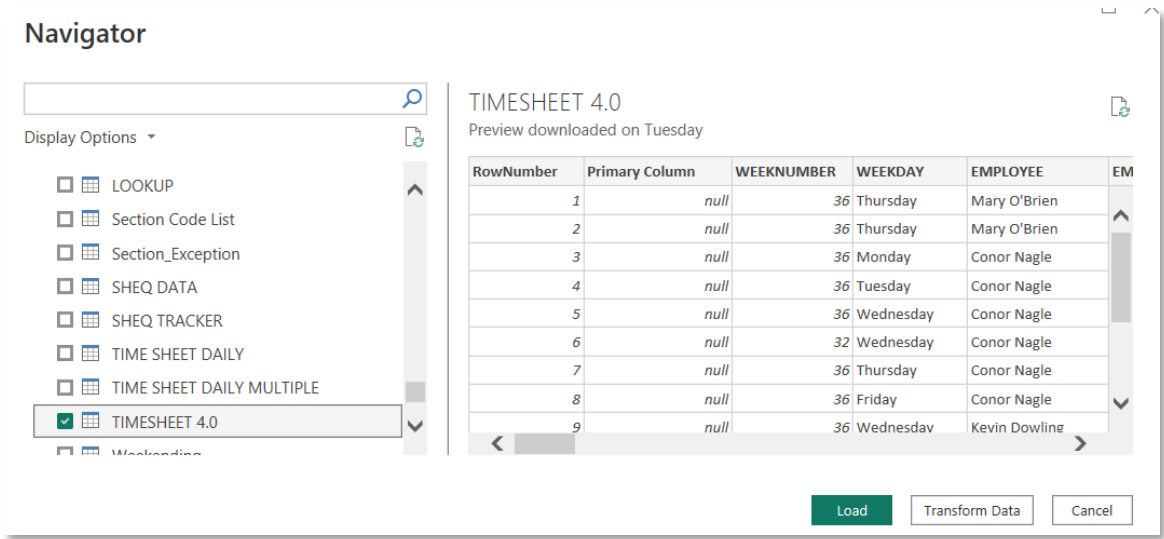
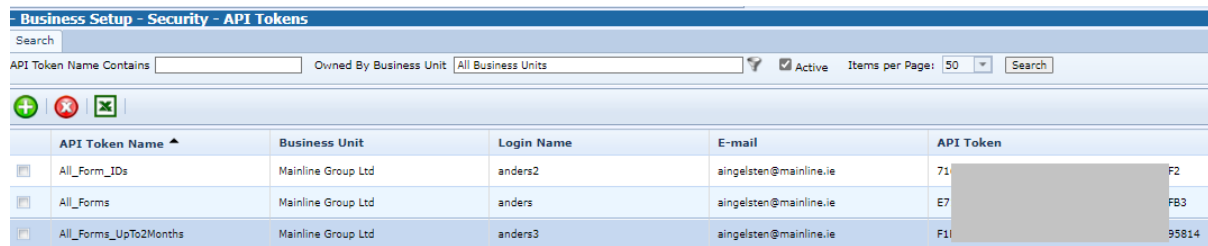


Figure 15: Preview of Connected Table ready to be loaded into PowerBI

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.



API Token Name	Business Unit	Login Name	E-mail	API Token
All_Form_IDs	Mainline Group Ltd	anders2	aingelsten@mainline.ie	71 F2
All_Forms	Mainline Group Ltd	anders	aingelsten@mainline.ie	E7 FB3
All_Forms_UpTo2Months	Mainline Group Ltd	anders3	aingelsten@mainline.ie	F11 95814

Figure 16: 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. Source [New-WebServiceProxy \(Microsoft.PowerShell.Management\) - PowerShell | Microsoft Learn](#)

```
<#
Fieldview API 1 - Lst of all forms 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.chiltnodes.id
$id = $FVApiConfig.GetProjects($ApiToken, $null, $null, 1, 0, 100).ProjectInformation.chiltnodes.id
```

Figure 17: Code Snippet of the New-WebServiceProxy used to pull data from Fieldview

The New-WebServiceProxy sets up a proxy object, 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. Therefor I generated a number of API keys so I can concurrent 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 18: 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 19: Screenshot of Sample Data from the GetProjectFormsList() call, returning ProjectFormsListInformation.

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. 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.

Source: [How to Connect to SharePoint Online using PnP PowerShell? - SharePoint Diary](#)

```
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 20: Sample Code of writing a file to Sharepoint Online

```
InformationRightsManagementSettings : Microsoft.SharePoint.Client.InformationRightsManagementFileSettings
ImEnabled                          : False
Length                             : 0
Level                              : Published
LinkingUrl                         : 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                    : Microsoft.SharePoint.Client.PropertyValues
Properties                         :
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 21: Screenshot of successfully writing the data to Sharepoint Online in PowerShell

Phase 4: PowerBI Dashboard – Fieldview Vehicle-check

Text here

Phase 5: PowerBI Dashboard – Fieldview Form Count

Text here

Phase 6: PowerBI Dashboard – Fieldview Timesheet

Text here

Phase 7: PowerBI Dashboard – SmartSheet Timesheet

Text here

Project Evaluation

Conclusion

Reflection

	A	B
1	Length	
2	10	
3		
4		

Challenges

Future Work

SQL?

Chainage control per project

PDF files and images

References

Mainline

[Mainline Group - Projects that Matter](#)

Fieldview

[Field View \(priority1.uk.net\)](#)