Proposed Power BI set up

# Overview

This proposed set up assumes the ‘app owns data’ approach to embedding Power BI would be taken as:

* the data set(s) will be embedded within the Analytics Portal (the ‘portal’)
* the users are not required to have their own Power BI application (or their own Power BI licenses)
* the portal will handle authentication and access
* users may not be AD users i.e. client users

# Requirements

To achieve set up, the following elements/assets are required:

* Azure AD tenant
* Power BI Pro account
* Power BI dashboard, tile or report
* Power BI workspace
* Power BI embedded capacity (for testing and for production)
* The application within which the Power BI content will be embedded i.e. the portal

# Power BI licensing and costs

## Licensing requirements for Power BI Embedded

[source: Microsoft Power BI Embedded Analytics Pricing Guidance WP - Oct 2017.pdf]

*Two elements are required to take embedded analytics to production. First, Power BI Pro licenses are required for administration, content publishing, and development. Then, Power BI Embedded is required for testing and deployment.*

### *Development*

*Development requires a Power BI Pro license for each developer on the team. Anyone using the Power BI APIs is required to have a Power BI Pro license. Power BI Pro licenses contain all APIs and can be used to develop a full proof of concept. Pre-production and production For testing and deployment, Power BI Embedded must be acquired. Power BI Embedded enables full testing of the solution with embed tokens that allow multi-user access to the embedded Power BI reports and dashboards. When deploying to a production environment, options include using the same Power BI Embedded capacity used for preproduction or a dedicated capacity for pre-production and another for production.*

### *Calculating capacity*

*Calculating how much capacity is needed is based on multiple parameters like the specific data models used, the number and complexity of queries, the hourly distribution of the usage of your SaaS application, the data refresh rates, and other usage patterns that are hard to predict.*

|  |  |  |  |
| --- | --- | --- | --- |
| Asset/Item | Pricing per unit | Number of units | Annual cost |
| Power BI Pro Account | £7.50 per user per month (excl. existing corporate discount) | 2 x software development  3 x insights | £450 |
| Power BI Embedded Capacity | TBC | TBC | TBC |
| Power BI Workspace |  |  |  |

## Example capacity calculation

In a peak hour, the portal will handle 20 users.

Each user will access a product and interact with it 25 times during the peak hour.

A typical product will contain 8 reports.

The total page renders for that hour are 20 x 8 x 25 = 4,000 page renders.

Per the capacity planning guidelines published by Microsoft, this would necessitate choosing Power BI Embedded A5.

# Implementation

[Source: <https://datasavvy.me/2018/04/25/thoughts-and-lessons-learned-from-a-power-bi-embedded-poc/>]

* Assign a Power BI Pro license to the AD account that will be used by the portal application. This should be a generic account and not a named individual e.g. [analyticsportaladmin@psyon.co.uk](mailto:analyticsportaladmin@psyon.co.uk). This is the ‘service account’.
* Create an app workspace in PowerBI.com; set to private.
* Set a report owner as the workspace admin i.e. this would be an internal analyst – it should be a person within Anna Spender’s team but would recommend this is another generic account e.g. [insightsadmin@psyon.co.uk](mailto:insightsadmin@psyon.co.uk) . This is the ‘analyst account’.
* Set the service account as an additional workspace admin within the app workspace.
* Create a report in Power BI Desktop.
* Build row-level security roles and filters.
* Publish the report to the app workspace.
* Register the application that will show the report in Azure AD.
* Add code to the application to get the AD access token when required (service account to access Power BI)
* Add .js to the application to get the content, create the embed token and load the content
* Provision the appropriate Power BI embedded capacity in Azure and assign the app workspace containing the report to the embedded capacity.

# Security

Our clients would be non-Power BI users so the embed token would be configured for a user and role.

Portal authentication and product entitlement is controlled within a SQL database (Elysium or another).

Power BI row-level security is used to define roles and related rules. [This document](https://docs.microsoft.com/en-us/power-bi/developer/embedded-row-level-security) summarises the concepts of users, roles and rules and some further context is in the below table.

|  |  |  |  |
| --- | --- | --- | --- |
| Concept | Description (from Microsoft documentation) | Psyon examples (under development) | Security data model comments |
| **Users** | End users viewing the artifact (dashboard, tile, report, or dataset). In Power BI Embedded, users are identified by the username property in an embed token | * End users within a corporate client | Users are not held in Power BI, users are Portal users |
| **Roles** | Users belong to roles. A role is a container for rules and can be named something like Sales Manager or Sales Rep. You create roles within Power BI Desktop | * Client [Client ID-Name] Key Contact * Client [Client ID-Name] End User | Roles are mastered within Power BI (also where reports are developed); our own client user role relationships need to be stored at application level to enable the creation of the embed token |
| **Rules** | Roles have rules, and those rules are the actual filters that are going to be applied to the data. The rules could be as simple as “Country = USA” or something much more dynamic. | For the key contact – they can access everything related to that client ID so the rule might be Client ID = [client ID]  For an end user within the same client company, they might only be able to access data related to employees within | Rules are configured within Power BI |