Meerkat Deployment

# LightSwitch Deploy

## Files

The Latest light switch files can be found on GIT, they are in the \Meerkat\Development folders.

## Setup

Need to have IIS & SQL Server all setup already. IIS needs to have the two websites setup already, both on their own application pool. Either application pool needs to be setup with a specific user that will be used for the access. The application pool needs to be set to .NET Framework v4.

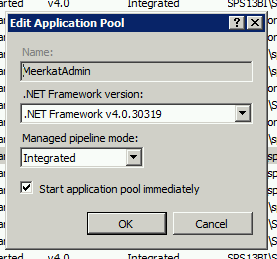


Figure 1: Example of Application Pool settings, important here is the framework version.

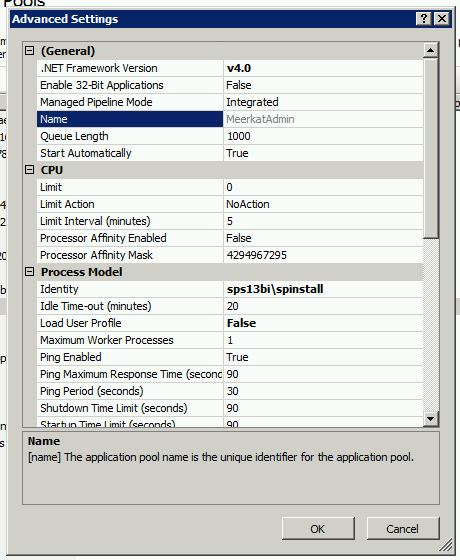


Figure 2: Example of application pool advanced settings. Important here is the identify setting for setting what user will run the process.

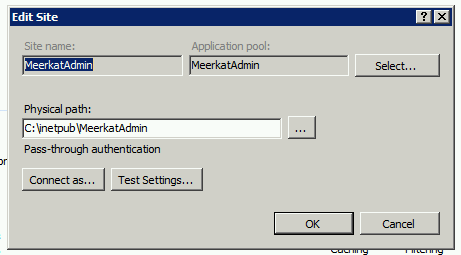


Figure 3: Example of the website for Meerkat Admin with application pool set.

## Configure LightSwitch Apps

For the admin you need to edit: MeerkatAdmin\_2\_1.SetParameters.xml

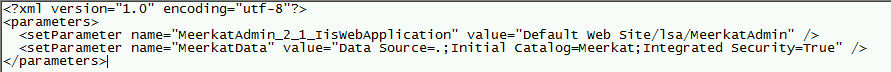
The first line is the website name and the second line is the connection string for the database.

Figure 4: Configuration file before changes.

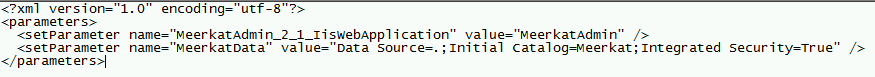


Figure 5: Configuration file after changes, note the website name is change.

For the capture you need to edit the following file, which has the same options as the admin config: Meerkat\_Capture.SetParameters.xml

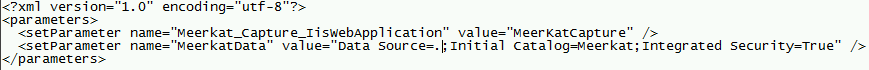


Figure 6: Capture configuration file, after changes.

## Deploy the LightSwitch apps

Deployment of the LightSwitch apps is very straightforward, all you need to do is run the batch file in the folder with the /Y switch. The only important thing to note, is that this must be run with administrator privileges.

For the admin client that is: .\MeerkatAdmin\_2\_1.deploy.cmd /Y

For the capture client, the command is: .\Meerkat\_Capture.deploy.cmd /Y

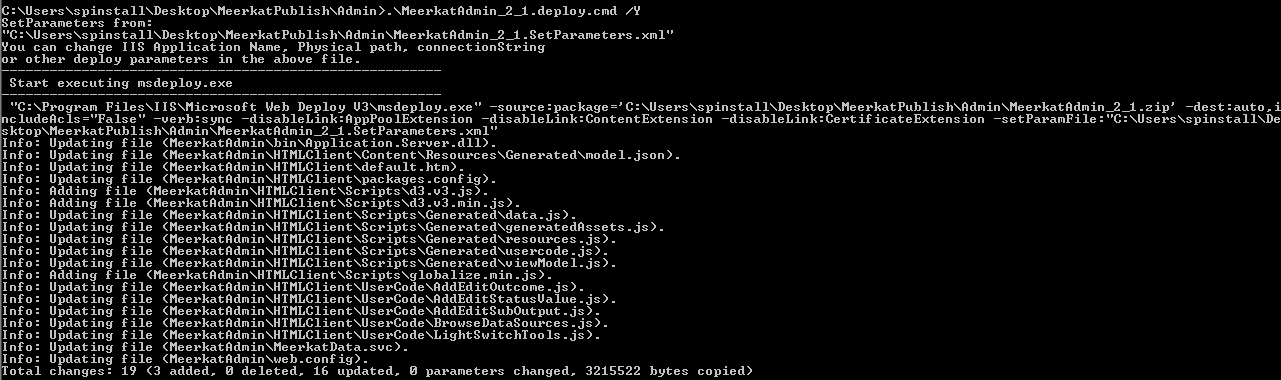


Figure 7: Example of the admin client being run on the server.

# SQL Deploy

## From Visual Studio

From within Visual Studio, right click on the SQL project and select Publish. You may need to allow remote connections and create a SQL login to allow the publish. Publishing from Visual Studio isn’t recommended if we are doing a new

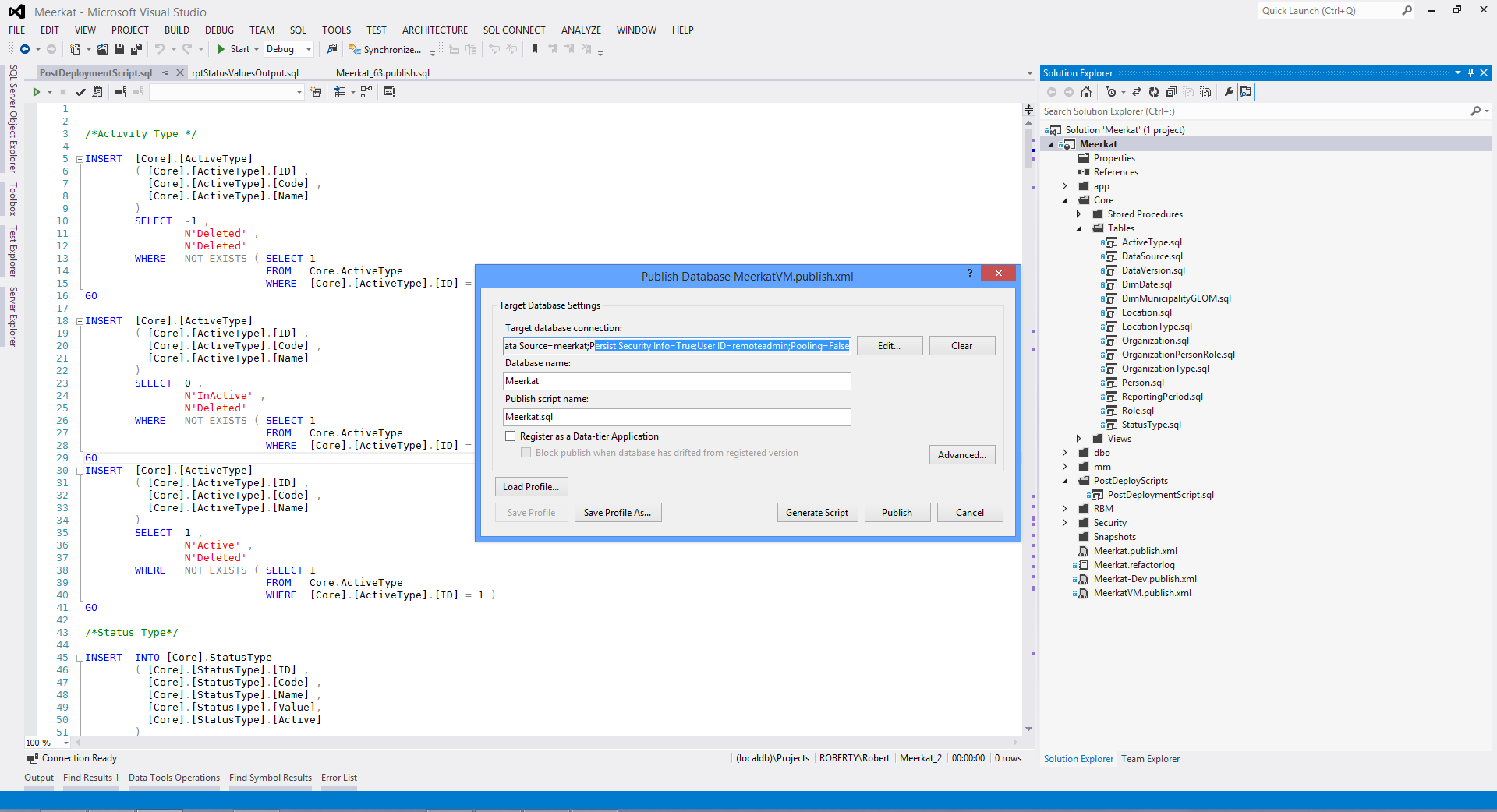


Figure 8: Example of publishing from Visual Studio.

If you are updating an existing database, you want to make sure in the Advanced Publish Settings you select “Always re-create database” and “Back up database before deployment” – this will however kill all the data.

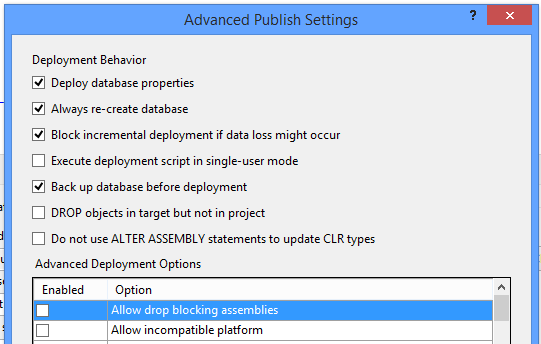


Figure 9: Example of advanced publish settings.

# SQL Reporting Services Deploy

To deploy the reports, you need to open the three SRS projects, namely:

Meerkat\Reports\Admin Reports\MeerkatReports\AdminReports.rptproj

Meerkat\Reports\Status Reports\MeerkatReports\StatusReports.rptproj  
Meerkat\Reports\Value Reports\MeerkatReports\ValueReports.rptproj

For each project you need to first open the project property page and change the target URL to point to the correct server and paths. You should not be changing the query string, just the FQDN portion of the URL.

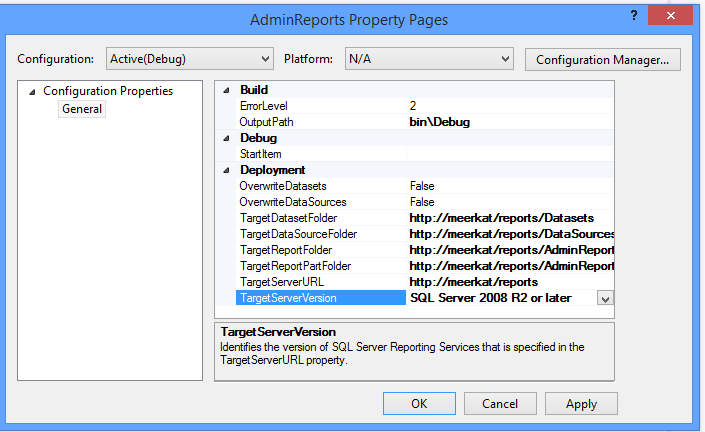


Figure 10: Example of the SRS project property page.

The next step is to open the shared data source in the project and make sure it is pointing to the correct SQL connection string.

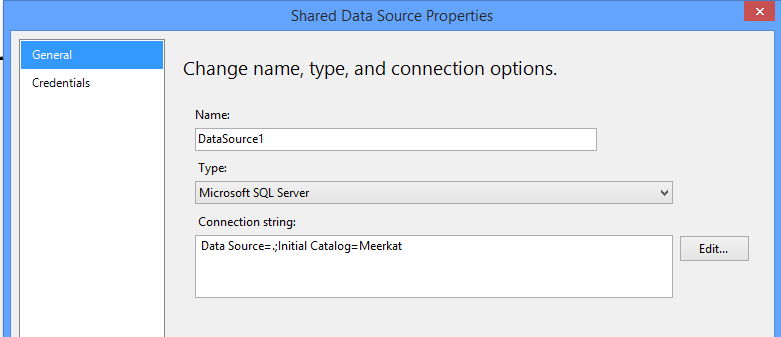


Figure 11: Example of the shared data source properties.

Once you have configured the shared data source, you need to ensure all reports are set to use the shared data source. This involves opening each report and check it’s data source properties.

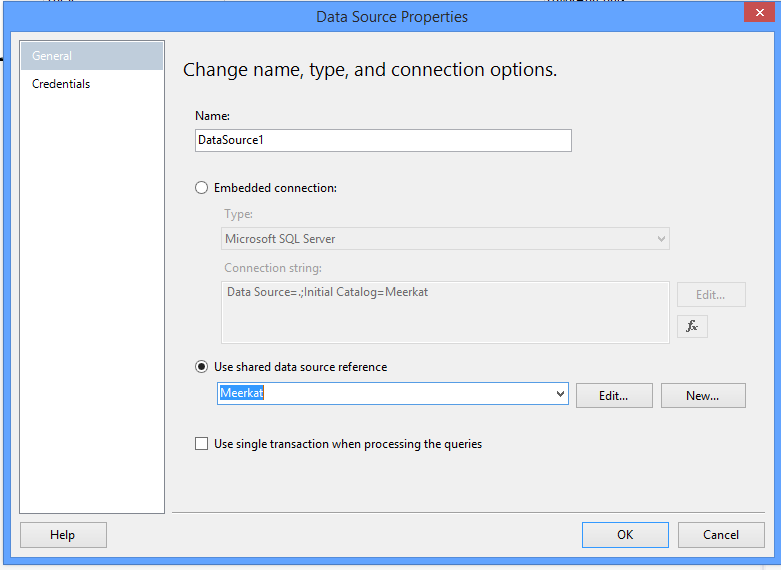


Figure 12: Confirming the shared data source is used in a report.

Finally you can right click on the project and hit deploy. You may get prompted for a login, this needs to be a valid login on the server (not a SQL login).

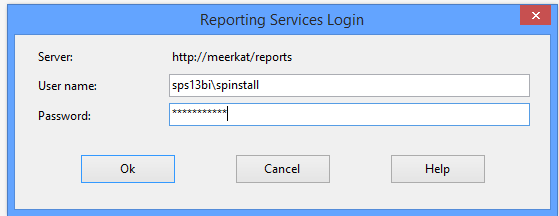


Figure 13: Example of the deploy login dialog.

# SharePoint

Requirements:

SQL Server installed and configured.

SharePoint Foundation to be installed and Configured.  
<http://www.microsoft.com/en-za/download/details.aspx?id=35488>

**Steps to deploy Meerkat**

**SharePoint**

1. Create web application

Create a web application on port 80 with a host header entry of Meerkat  
Ensuring that a descriptive Database name is chosen. Eg. SharePoint\_Content\_Meerkat  
Create a new Application Pool running under its own account.  
Leave the authentication as NTLM.  
All other settings can be left as default.

1. Required Service Applications to be setup are:  
   SQL Reporting Services  
   State Service  
   Usage Service
2. Open up SharePoint PowerShell as administrator  
   Use the restore-SPSite command to import the Root site   
   restore-SPSite –Identity <Http://URLName> –path “Location”
3. Open Central Admin, edit the Site collection Administrator list to include your user account

This site collection will have the Root site and 8 Outcome Sites

**Light Switch**

1. You will need the URL of the two Light Switch application setup in the beginning of the documents
2. The SharePoint Web viewer web parts will need to be updated  
   The pages are located here  
   <http://sitename/Capture>

<http://sitename/Admin>

Edit the page, and edit the web part properties of the web page viewer web part to point to the Light Switch pages as deployed earlier.

**Post Deployment scripts GIT content**

Using the Git tool of your choice – Git Hub for Windows, Git Extensions etc

Pull the latest build.

The reports can be found in \Meerkat\Reports there is a sub folder for each report section.

Namely:

* Meerkat\Reports\Admin Reports
* Meerkat\Reports\Status Reports
* Meerkat\Reports\Value Reports

The SQL Project can be found \Meerkat\SQL Solution, there is a deployment script there to create the DB  
C:\GIT\Meerkat\SQL Solution\Meerkat\bin\Debug\Meerkat.publish.sql

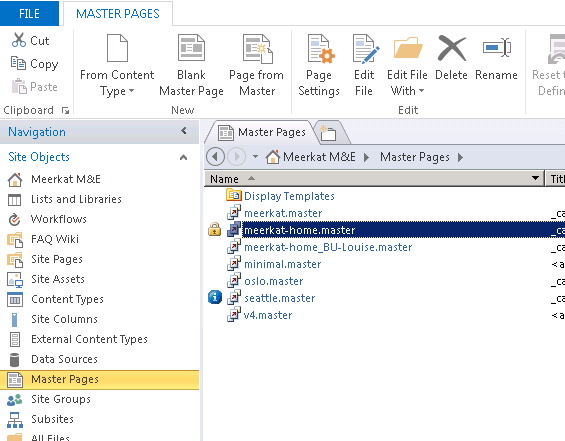
**SharePoint Designer – master page edits**

Using SharePoint Designer open up the master page to make edits if required for the Menu items.

You can download it from <http://www.microsoft.com/en-za/download/details.aspx?id=35491>

There are two master pages, Meerkat-home and Meerkat.master

The Home page is for the Root site, and the Meerkat.master is for the sub sites.



The menu items can be searched for and edited below.

Just search for the text or URL that needs to be edited





**MegaMenu**

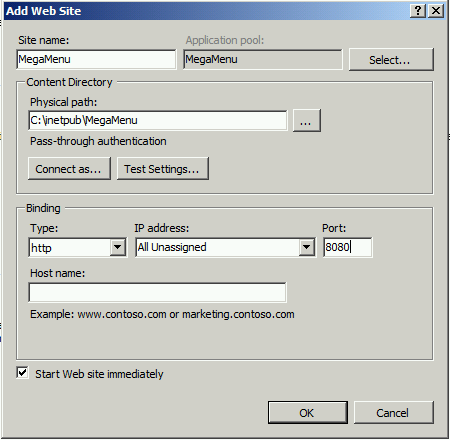
MVC2 is required for the MegaMenu this can be found on the Microsoft site <http://www.microsoft.com/en-za/download/details.aspx?id=22079>

In Git there is a folder for the Megamenu, These files need to be copied to the InetPub Folder and then setup as a web site.

IIS

Create a new Application that runs under a domain account. We used the Installation account.  
Create a new IIS Site Called Megamenu and use the Application pool created above.

Test the settings.



SharePoint

In each Outcome site there is a file called MegaDropDown\_Build.js

The Content of each file is similar but the Outcome information needs to be changed.

$(document).ready(function (e) {

megaDropdown.config.DataSource = "SQL";

megaDropdown.config.CachingEnabled = false;

megaDropdown.config.Logging = false;

megaDropdown.config.SQLRootUrl = "**http://ServerName:8082**";

megaDropdown.config.SQLCategoryTable = "**mm.Outcome1MenuCategory**";

megaDropdown.config.SQLGroupTable = "**mm.Outcome1MenuGroup**";

megaDropdown.config.SQLLinkTable = "**mm.Outcome1MenuLink**";

megaDropdown.config.SQLJSONP = true;

megaDropdown.printConfig("#config");

megaDropdown.build("megadropdown");

});

The URL is the address of the IIS web site we created above in IIS.

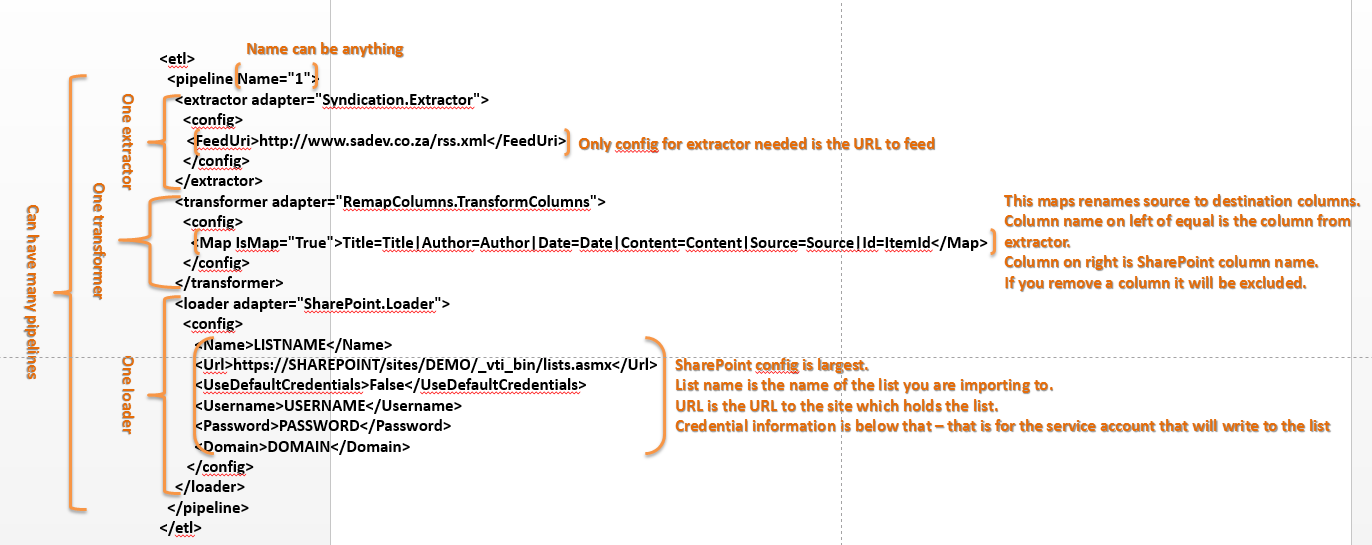
The 3 lines beneath the URL are to be set per Outcome. Just edit the Outcome Number for each outcome.

**News Aggregator**

On Git there is a folder called NewsAgg, within this folder there is an executable called ETL.Runner.Console, this needs to be setup to run on a schedule. A Windows task schedule to run once daily is fine.

There is a config file called ETL.Config, this file is a sequence of repeating nodes telling what information to download and where to go to.

Edit this as required for each news section per outcome



As sample of this is below

<pipeline Name="OutcomeHome">  
 <extractor adapter="Syndication.Extractor">  
 <config> <FeedUri><![CDATA[https://news.google.com/news/feeds?hl=en&gl=za&as\_epq&as\_oq&as\_eq&as\_scoring=r&as\_drrb=q&as\_qdr=a&as\_nsrc&as\_nloc=Kenya&as\_author&as\_occt=any&q=Millennium+development+goals+location:kenya&um=1&ie=UTF-8&output=rss]]></FeedUri>  
 </config>  
 </extractor>  
 <transformer adapter="UniqueRows.TransformAdapter">  
 <config>  
 <UniqueColumnName>UniqueId</UniqueColumnName>  
 <Filename>C:\NewsAgg\syndicationRun0.sqlite</Filename>  
 </config>  
 </transformer>

<transformer adapter="RemapColumns.TransformColumns">  
 <config>  
 <Map IsMap="True">Title=Title|Author=Author|Date=Date|Content=Content|Source=Source|Id=ItemId|UniqueId=UniqueId</Map>  
 </config>  
 </transformer>  
 <loader adapter="SharePoint.Loader">  
 <config>  
 <Name>ExternalNews</Name>

<Url>http://meerkat01/\_vti\_bin/lists.asmx</Url>

<UseDefaultCredentials>True</UseDefaultCredentials>

<Username>UserName</Username>

<Password>Password</Password>

<Domain>Domain</Domain>

<ApplyGoogleImgSrcHackTo>Content</ApplyGoogleImgSrcHackTo>

</config>

</loader>

</pipeline>

**Reports**

The Reports are located on <http://URL/reports>

There are 3 libraries that contain reports, and a Data Source Library.  
Access the Data source library and edit the Meerkat Data source, ensuring that it points to the correct SQL instance and to the Meerkat Database.  
The Data source requires a user name to connect under, the account we have used is called MNEReports.

This account is a domain user, so you will need to create that user, and a password.  
The user will require access to the Meerkat Database, give the user DB\_DataReader access.

The Reports are housed in 3 libraries depending on their area of use.

* Admin Reports
* Status Reports
* Value Reports.

# Sample Data

TBD