A picture containing laser

Description automatically generated

|  |
| --- |
| BCX  Health & Retail Solutions  Functional & Technical Specification  Clinic & Unisolv Modernization |

**DOCUMENT VERSION 1.0**

**2021-10-12**

**ARCHITECT / CONTRIBUTORS / DEVELOPERS**

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Department** |
| Douw Muller | Architect | BCX |

**TEAM**

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Contact Information** |
|  |  |  |
|  |  |  |

**REVIEWERS**

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Contact Information** |
| Douw Muller |  |  |

**VERSION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **New Functionality** | **Developer** |
| 2021-07-23 | 1.0.0 | Initial Draft | Douw Muller |

**Table of Contents**

[Introduction 5](#_Toc86132793)

[1.1 Terms/Acronyms & Definitions 5](#_Toc86132794)

[1. Data Bureau 6](#_Toc86132795)

[1.1 Overview 6](#_Toc86132796)

[1.2 Contact & Resources 7](#_Toc86132797)

[1.3 Schema 8](#_Toc86132798)

[1.4 Product Price Index 16](#_Toc86132799)

[1.4.1 Overview 16](#_Toc86132800)

[1.4.2 Schema & Data Files 17](#_Toc86132801)

[1.4.3 Functionality 19](#_Toc86132802)

[1.4.4 Backward Compatibility 28](#_Toc86132803)

[1.5 Product Price Detail 28](#_Toc86132804)

[1.5.1 Overview 28](#_Toc86132805)

[1.5.2 Schema & Data Files 29](#_Toc86132806)

[1.5.3 Functionality 30](#_Toc86132807)

[1.6 Product Price Formulary 36](#_Toc86132808)

[1.6.1 Overview 36](#_Toc86132809)

[1.6.2 Schema & Data Files 37](#_Toc86132810)

[1.6.3 Functionality 39](#_Toc86132811)

[1.7 Product Price Equivalence 43](#_Toc86132812)

[1.7.1 Overview 43](#_Toc86132813)

[1.7.2 Schema & Data Files 44](#_Toc86132814)

[1.7.3 Functionality 45](#_Toc86132815)

[1.8 Translation Files – Drug Types 56](#_Toc86132816)

[1.8.1 Overview 56](#_Toc86132817)

[1.8.2 Schema & Data Files 57](#_Toc86132818)

[1.8.3 Functionality 57](#_Toc86132819)

[1.8.4 Backward Compatibility 61](#_Toc86132820)

[1.9 Translation Files – Active Ingredients 61](#_Toc86132821)

[1.9.1 Overview 61](#_Toc86132822)

[1.9.2 Schema & Data Files 62](#_Toc86132823)

[1.9.3 Functionality 63](#_Toc86132824)

[1.9.4 Backward Compatibility 68](#_Toc86132825)

[1.10 Translation Files – Manufacturers 68](#_Toc86132826)

[1.10.1 Overview 68](#_Toc86132827)

[1.10.2 Schema & Data Files 69](#_Toc86132828)

[1.10.3 Functionality 69](#_Toc86132829)

[1.11 Medical Aid Schemes 71](#_Toc86132830)

[1.11.1 Overview 71](#_Toc86132831)

[1.11.2 Schema & Data Files 71](#_Toc86132832)

[1.11.3 Functionality 73](#_Toc86132833)

[1.11.4 Functionality 79](#_Toc86132834)

[1.12 Medical Aid Option Information 82](#_Toc86132835)

[1.12.1 Overview 82](#_Toc86132836)

[1.12.2 Schema & Data Files 82](#_Toc86132837)

[1.12.3 Functionality 83](#_Toc86132838)

[1.13 Claim Admin Office 85](#_Toc86132839)

[1.13.1 Overview 85](#_Toc86132840)

[1.13.2 Schema & Data Files 85](#_Toc86132841)

[1.13.3 Functionality 85](#_Toc86132842)

[1.14 Versions 87](#_Toc86132843)

[1.14.1 Overview 87](#_Toc86132844)

[1.14.2 Schema 87](#_Toc86132845)

[1.14.3 Functionality 87](#_Toc86132846)

[1.15 Uploads 90](#_Toc86132847)

[1.16 Processes 90](#_Toc86132848)

# Introduction

## Terms/Acronyms & Definitions

|  |  |  |
| --- | --- | --- |
| **Term/Acronym** | **Definition** | **Description** |
|  |  |  |
|  |  |  |

# Data Bureau

## Overview

Data Bureau facilitates the import from Unisolv Data. This data will be maintained in the Data Bureau module and any updates will be sync back to Unisolv using backward compatibility files.

Data that will be maintained in Data Bureau:

* Product Price Index
* Product Price Index Details

Datasets will be differentiated by:

* Version

Indicate the version of the dataset. Any committed changes will cause a new version to be created.

* Tags

Tags can be linked to a dataset to indicate the purpose of the datasets. When datasets are sync to clients via Web-Hooks, Tags can be used to filter datasets that will be sync to specified client.

Unisolv files needed for the datasets imports are created with Migration:

Migrations > Tenants > DatabureauCreator.cs

* CreateTranslationDrugTypes
* CreateCommonFiles
  + Create rows in the File Repository that point to the import files.
* CreateTags

## Contact & Resources

|  |  |
| --- | --- |
| Contact | Description |
| [Angelo.Forbes@bcx.co.za](mailto:Angelo.Forbes@bcx.co.za) | IT Support |
| [Weyers.Wium@bcx.co.za](mailto:Weyers.Wium@bcx.co.za) | IT Support |
| BCX IT | 086 122 1102 |
| [Call.Desk@ourservicedesk.com](mailto:Call.Desk@ourservicedesk.com) |
| Contacts | [colin.meintjies@bcx.co.za](mailto:colin.meintjies@bcx.co.za) |
|  |  |

**VPN Access**

You must sign into this VPN in Bloemfontein to be able to access the Data Bureau server:



Your sign in credentials is:

VPN Name: Bloem VPN

Username: douw.muller

Password: Summer@2021

## Schema

**claimadmins**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  |  |

**fileitems**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  |  |
|  | Name |  |  |
|  | MimeType |  |  |
|  | Path |  |  |
|  | FilePath |  |  |
|  | IsFolder |  |  |
|  | FileStatus |  |  |
|  | FileId |  |  |
|  | TenantId |  |  |

**productdetails**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  |  |
|  | ProductIndexId |  | Productindexes.Id |
|  | PackQty |  |  |
|  | EffectDate |  |  |
|  | NappiCodeExtended |  |  |
|  | Price |  |  |
|  | Cost |  |  |
|  | UPDCount |  |  |
|  | DiscontDate |  |  |
|  | ExitPrice |  |  |
|  | Schedule |  |  |
|  | TenantId |  |  |
|  | Tags |  |  |

**productequivalenceatcs**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  | productindexes.ProductEquivalenceATCId  productequivalencegenerics.ProductEquivalenceATCId  versionproductequivalenceatc.ProductEquivalenceATCId |
|  | ATCCode |  |  |
|  | TotalProducts |  |  |
|  | TenantId |  |  |
|  | Tags |  |  |

**productequivalencegenerics**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  | productindexes.ProductEquivalenceGenericId  versionproductequivalencegeneric.ProductEquivalenceGenericId |
|  | GenericCategory |  |  |
|  | GenericSubCategory |  |  |
|  | TotalProducts |  |  |
|  | Strength |  |  |
|  | TranslationDrugTypeId |  | translationdrugtypes.Id |
|  | TenantId |  |  |
|  | Tags |  |  |
|  | ProductEquivalenceATCId |  | productrquivalenceatcs.Id |

**productformularies**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | DiscontDate | datetime(6) | Date when this formulary was discontinued |
|  | EffectDate | datetime(6) | Date when this formulary came into effect |
|  | GenericDrug | tinyint(1) |  |
|  | Group | int |  |
|  | **Id** | bigint |  |
|  | MedicalPrice |  |  |
|  | PackQty |  |  |
|  | Priority |  |  |
|  | ProductFormularyGroupId |  | FK: productformularygroups.Id |
|  | ProductIndexId |  | FK: productindexes.Id |
|  | SubGroup |  |  |
|  | Tags |  |  |
|  | UPD |  |  |

**productformularygroups**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Code |  |  |
|  | **Id** |  | Referenced by:  productformularies.ProductFormularyGroupId |
|  | Tags |  |  |
|  | TotalFormularies |  |  |

**productformularyprices**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | DiscontDate |  | Date when this formulary price was discontinued |
|  | EffectDate |  | Date when this formulary price came into effect |
|  | GenericDrug | tinyint(1) |  |
|  | **Id** | bigint |  |
|  | MedicalPrice |  |  |
|  | Priority | int |  |
|  | ProductFormularyId | bigint | FK: productformularies.Id |

**productindexes**

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  | productformularies.ProductIndexId  productformularygroups.ProductIndexId  versionproductIndex.ProductIndexId |
|  | NappiCode |  |  |
|  | Name |  | Name of Tag |
|  | Strength |  |  |
|  | DosageForm |  |  |
|  | ManufacturerCode |  |  |
|  | TranslationManufacturerId |  |  |
|  | TranslationDrugTypeId |  | translationdrugtypes.Id |
|  | Schedule |  |  |
|  | ScheduleChronic |  |  |
|  | EffectDate |  |  |
|  | DiscontDate |  |  |
|  | NameLong |  |  |
|  | SourceCode |  |  |
|  | Ethical |  |  |
|  | IsChronic |  |  |
|  | IsIndivisablePack |  |  |
|  | Indivisable |  |  |
|  | UPD |  |  |
|  | ProductDetailsCount |  |  |
|  | TenantId |  |  |
|  | Tags |  |  |
|  | ProductEquivalenceATCId |  | productequivalenceatcs.Id |
|  | ProductEquivalenceGenericId |  | productequivalencegenerics.Id |
|  | ProductFormularyCount |  |  |

**tags**

This table contains all tags that can be linked to datasets in the Data Bureau to enable clients to subscribe to specific tags only and therefore only get updates for the required datasets.

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id | char(36) | updatepriceversions.TagId |
|  | Name | longtext | Name of Tag |

**translationactiveingredients**

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Code | longtext |  |
|  | Description | longtext |  |
|  | **Id** | int |  |
|  | Tags | longtext |  |
|  | TenantId | int |  |

**translationdrugtypes**

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Code | longtext |  |
|  | Description | longtext |  |
|  | Direction | longtext |  |
|  | **Id** | int |  |
|  | Tags | longtext |  |
|  | TenantId | int |  |

**translationmanufacturers**

(Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | Id |  |  |
|  | Code | longtext |  |
|  | Description | longtext |  |
|  | Tags | longtext |  |
|  | TenantId | int |  |

**updatepriceversions**

Contains a list of all versions for the Data Bureau. The Data Bureau consists of a set of datasets and if any set is updated and committed that creates a new version. (Note: zero framework columns omitted)

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | **Id** |  |  |
|  | VersionNumber |  |  |
|  | VersionStatus | int | 1:  0: |
|  | CommitDate |  |  |
|  | PublishedDate |  |  |
|  | TenantId |  |  |
|  | TagId |  | Link to tags.Id |
|  | TotalChanges | Int | Indicate the total changes for this version across all datasets |
|  | TotalTranslationDrugTypes | int | Indicate changes for this version for the Translation Drug Types dataset |
|  | TotalTranslationManufacturers | int | Indicate changes for this version for the Translation Manufacturer dataset |
|  | TotalTranslationActiveIngredients | int | Indicate changes for this version for the Translation Active Ingredients dataset |
|  | TotalProductIndex | int | Indicate changes for this version for the Product Index dataset |
|  | TotalProductEquivalenceGeneric | int | Indicate changes for this version for the Product Equivalence Generic dataset |
|  | TotalProductEquivalenceATC | int | Indicate changes for this version for the Product Equivalence ATC dataset |
|  | TotalProductDetails | int | Indicate changes for this version for the Product Details dataset |
|  | TotalProductFormularies | int | Indicate changes for this version for the Product Formularies dataset |
|  | TotalClaimAdmins | Int | Indicate changes for this version for the Claim Admin dataset |
|  | TotalMedicalAidOptions | int | Indicate changes for this version for the Medical Aid Options dataset |
|  | TotalMedicalAidSchemes | int | Indicate changes for this version for the Medical Aid Schemes dataset |

**versionproductdetail**

Link each product index row in table ‘productdetails’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | ProductDetailId |  | productdetails.Id |
|  | UpdatePriceVersionId |  | updatepriceversions.Id |

**versionproductequivalencegeneric**

Link each product index row in table ‘productequivalencegenerics’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | ProductEquivalenceGenericId |  | productequivalencegenerics’.Id |
|  | UpdatePriceVersionId |  | updatepriceversions.Id |

**versionproductformulary**

Link each product formulary row in table ‘productformularies’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | ProductFormularyId |  | FK: productformularies.Id |
|  | UpdatePriceVersionId |  | FK: updatepriceversions.Id |

**versionproductindex**

Link each product index row in table ‘productindexes’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | ProductIndexId |  | productindexes.Id |
|  | UpdatePriceVersionId |  | updatepriceversions.Id |

**versiontranslationactiveingredient**

Link each translation active ingredient row in table ‘translationactiveingredients’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | TranslationActiveIngredientId |  | FK: translationactiveingredients.Id |
|  | UpdatePriceVersionId |  | FK: updatepriceversions.Id |

**versiontranslationdrugtype**

Link each translation drug type row in table ‘translationdrugtypes’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | TranslationDrugTypeId |  | FK: translationdrugtypes.Id |
|  | UpdatePriceVersionId |  | FK: updatepriceversions.Id |

**versiontranslationmanufacturer**

Link each manufacturer row in table ‘translationmanufacturers’ to a specified version stored in table ‘updatepriceversion’

|  |  |  |  |
| --- | --- | --- | --- |
| Ref | Name | Data Type | Description |
|  | TranslationManufacturerId |  | FK: productformularies.Id |
|  | UpdatePriceVersionId |  | FK: updatepriceversions.Id |

## Product Price Index

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Products > Index |

### Schema & Data Files

#### Schema

Timeline

Description automatically generated

dosageforms

sourcecodes

#### Data Files

Web.Mvc\wwwroot\File-Repository\Unisolv\Data\PHMDRG

Web.Mvc\wwwroot\File-Repository\Unisolv\Data\PHMDRG.vix

Web.Mvc\wwwroot\File-Repository\Unisolv\Data\phmdrg.xfd

**PHMDRG**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | DRG\_NAP |  | Nappi Code |
| 1 | DRG\_DRUG | string | Drug Name |
| 2 | DRG\_FUTURE | string | ? Future |
| 3 | DRG\_STK\_CODE | string | <Not Used> |
| 4 | DRG\_SEQ | long | Id |
| 5 | DRG\_STRENGTH | string | Drug Strength |
| 6 | DRG\_DOSAGE | string | Dosage Form |
| 7 | DRG\_UPD\_PACK | string | <Not Used> |
| 8 | DRG\_MANUFACTURER | string | Manufacturer Code |
| 9 | DRG\_SUPPLIER | string | Translation Manufacturer Text |
| 10 | DRG\_DRUG\_TYPE | string | Translation Drug Type Text |
| 11 | DRG\_SCHEDULE | string | Schedule Text |
| 12 | DRG\_SCAN | string | ? |
| 13 | DRG\_CLM\_CODE | string | ? Claim Code |
| 14 | DRG\_USAGE | string | <Not Used> |
| 15 | DRG\_CLASS | string | <Not Used> |
| 16 | DRG\_COUNSEL\_CODE | string | <Not Used> |
| 17 | DRG\_EXCL\_MEDSCHEME | string | <Not Used> |
| 18 | DRG\_EXCL\_MKR | string | <Not Used> |
| 19 | DRG\_EXCL\_MEDISCOR | string | <Not Used> |
| 20 | DRG\_DT\_LAST | DateTime? | Last Modification Time |
| 21 | DRG\_EFFECT\_DATE | DateTime? | Effective Date |
| 22 | DRG\_DISCONT\_DATE | DateTime? | Discontinued Date |
| 23 | DRG\_LONG\_DESC | string | Long Name |
| 24 | DRG\_ATC\_CODE |  |  |
| 25 | DRG\_UPD\_CNT | string | ? UPD |
| 26 | DRG\_SOURCE\_CODE | string | Source Code |
| 27 | DRG\_ETHICAL | string | Ethical |
| 28 | DRG\_CHRONIC | bool? | Is Chronic? |
| 29 | DRG\_INDIVISABLE | bool? | Is Individual Pack? |
| 30 | DRG\_INDIS | string | ? Indivisable |
| 31 | DRG\_SCHEDULE\_2 | Int? | Schedule Chronic |
| 32 | DRG\_VAT\_INDICATOR | bool? | ? VAT |

### Functionality

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Entries added to table [fileitems] for the import data file and the vix file.  8 – PHMDRG file  9 – PHMDRG,vix file |

#### Import Data



|  |  |
| --- | --- |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js   * //File Items Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the [fileitems] that include a .vix file with PHMDRG in the filename. |
| **Import** | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Import Button   The process status of this dataset is stored in table  [abpsettings] as App.ProductPrice.Index  If the status is InProgress (1) the Import button will be disabled.  This setting will prevent CRUD functions to be performed while an import process is busy.  If the import did not complete successfully, you might have to restart the solution, as this status is stored in the zero-framework app settings entity and this will not change if you just change the value in the database. |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_ProductIndexImportModel.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\   * \_ProductIndexImportModel.js   //Import App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportProductIndex   Invoke background job to do the import.  Note: If you have a xfd file you can create a delimited output file during run-time (XFDJob.cs).  If you do not have a xfd file you can use the vtutil process to create an output file (VutilJob.cs). |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Index setting in [abpsettings] indicating that import is running.  This setting will prevent another Import or CRUD functions to be performed while import is  busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime    Application\DataBureau\ImportAppService.cs   * ImportData * case UpdateType.ProductIndex: |

#### Create

NOTE: Removed Schedule select element when adding a new Product, because this caused the value of the schedule a bit higher on the page to be replaced by this value… Need further investigation.

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Create Button |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductIndexModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\   * \_CreateUpdateProductIndexModal.js   //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductIndex |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### List Page (Data)

NOTE: Possible Bug – The product detail quantity box (green one) does not reflect the correct count.

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Get Products Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProducts   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductIndexes   Return product indexes. Also include foreign key relationship data: translationdrugtypes & translationmanufacturers |

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Edit Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductIndexModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductIndexModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\   * \_\_CreateUpdateProductIndexModal.js   //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductIndex |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### List Page (Data) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data) – Discontinue



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Discontinue Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueProductIndex |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs |

#### List Page (Data) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Restore Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueRestoreProductIndex |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs |

#### List Page (Data) – Search

* Search on NAPPI CODE & NAME (LONG NAME)

#### List Page (Data) – Sorting

* Sort on NAPPI CODE & NAME & EFFECTIVE DATE

#### List Page (Data) – Product Detail



The green box indicates the number of product detail rows for this product.

NAPPI CODE is the [ProductIndexId] value in entity [productdetails] + Extended Code supplied under Product Detail when a new Product was added.

NOTE: Not sure if Schedule reflects correctly, because you remove the select element as explained under CREATE.

NOTE: Remove ‘Create’ buttons on Product Detail & Product Formulary list. This could be functionality to be added in the future.

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Sub Details Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductDetails   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then invoke an App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductDetails   Return product details by ProductIndexId |

#### List Page (Data) – Product Detail - Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Edit Detail Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductDetailModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products   * \_\_CreateUpdateProductDetailModal.js   //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * UpdateProductDetails |

#### List Page (Data) – Product Detail - Change History



|  |  |  |
| --- | --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button | |
| **Controller** | | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js | |
| **App Service** | | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | | |
| **Script** | | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data) – Product Detail - Discontinue



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Discontinue Detail Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs |

#### List Page (Data) – Product Detail - Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Restore Detail Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueRestoreProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs |

#### List Page (Data) – Product Detail - Prices

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Sub Prices Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductDetailPrices   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then invoke an App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductDetailPrices   Return product detail prices by product detail. |

#### List Page (Data) – Product Formulary

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Products.cshtml   * //Sub Formularies Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductFormulariesByNappi   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then invoke an App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductFormulariesByNappi   . |

#### List Page (Data) – Product Formulary - Edit



#### List Page (Data) – Product Formulary - Change History



#### List Page (Data) – Product Formulary - Discontinue



#### List Page (Data) – Product Formulary - Restore



#### List Page (Data) – Product Formulary - Prices

#### List Page (Versions)

See Versions > List Page (Versions)

### Backward Compatibility

|  |  |
| --- | --- |
| Operation | Entity Updates |
| Add |  |
| Update Descriptions or Directions |  |
| Delete |  |

## Product Price Detail

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Products > Detail |

### Schema & Data Files

#### Schema

Diagram

Description automatically generated

|  |  |
| --- | --- |
| productindexes | Contains products |
| productdetail | Contains product details |
| productdetailprices | Contains product details prices |
| updatepriceversions | Used for version control of all datasets |
| tags | Contains different tags that can be linked to datasets |
| versionproductdetail | Link table between product detail rows and version they are linked to |
| abpsettings | Contains settings used to disable import and crud functions if import process busy |
| abpbackgroundjobs | Contains information about the background job executing the import process |
| fileitems | Contains import file details |

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMDSP

…\wwwroot\File-Repository\Unisolv\Data\PHMDSP.vix

…\wwwroot\File-Repository\Unisolv\Data\phmdsp.xfd

**PHMDSP**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | DSP\_PAC\_NAP | Long? | Nappi Code used to retrieve row from [productindexes] |
| 1 | DSP\_PAC\_QTY | string | Pack Qty |
| 2 | DSP\_FRM\_DT | DateTime? | Effective Date |
| 3 | DSP\_NAP\_EXT | string | Nappi Code Extended |
| 4 | DSP\_STK\_CODE | string | <Not Used> |
| 5 | DSP\_SEQ | int | Id |
| 6 | DSP\_PRICE | string | Price |
| 7 | DSP\_COST | string | Cost |
| 8 | DSP\_UPD\_PACK | string | <Not Used> |
| 9 | DSP\_USAGE | string | <Not Used> |
| 10 | DSP\_DT\_LAST | string | Last Modification Time |
| 11 | DSP\_DISC\_DATE | DateTime? | Discontinued Date |
| 12 | DSP\_UPD\_CNT | Int? | UPD Count |
| 13 | DSP\_EXIT\_PRICE | string | Exit Price |
| 14 | DSP\_SCHEDULE | Int? | Schedule |
| 15 | DSP\_AUTO\_CREATED | bool? | <Not Used> |

### Functionality

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 16,17 |

#### Import Data



|  |  |
| --- | --- |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_ProductDetailImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\   * \_ProductDetailImportModal.js |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the file repository that include **PHMDSP** in the filename. |
| **Import** | |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportProductDetail |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Index setting indicating that file is processing.  Stored in table [abpsettings]  This setting will prevent crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime    Application\DataBureau\ImportAppService.cs   * ImportData |

#### Create



|  |  |
| --- | --- |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Create Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductDetailModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\\_CreateUpdateProductDetailModal.js   * //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductDetail |

#### Filter Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Filter Apply Button * Show modal embedded in page with id filterModal * getData   This function will reload data table. |
|  |  |

#### List Page (Data)

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductDetails   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then invoke an App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductDetails   Get product details from [productdetails]. Get translation drug type using link to [productindexes] and [translationdrugtypes] |

#### List Page (Data) – Search

* Search on NAPPI CODE

#### List Page (Data) – Sorting

* Sort on NAPPI CODE, COST, SCHEDULE & EFFECTIVE DATA

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Edit Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductDetailModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products   * \_\_CreateUpdateProductDetailModal.js   //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * UpdateProductDetails |

#### List Page (Data) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data) – Discontinue

This will set the [productdetails].DiscontDate to the current date.



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Discontinue Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs   * Execute   Application\DataBureau\ImportAppService.cs   * UpdateProductDetails * Compare |

#### List Page (Data) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Restore Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueRestoreProductDetail |
| **Job** | Application\BackgroundJobs\Imports\ProductDetailUpdateJob.cs   * Execute   Application\DataBureau\ImportAppService.cs   * UpdateProductDetails * Compare |

#### List Page (Data) – Product Detail Price

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml   * //Product Details Prices Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductDetailPrices   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then invoke an App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductDetailPrices   Get product detail prices from [productdetailprices] |

#### List Page (Versions)

See Versions > List Page (Versions)

## Product Price Formulary

### Overview

A formulary is a **continually updated list of medications and related information**, representing the clinical judgment of pharmacists, physicians, and other experts in the diagnosis and/or treatment of disease and promotion of health. A written, unbiased evaluation of a specific medication.

### Schema & Data Files

#### Schema

Diagram

Description automatically generated

#### Formularies is stored in [productformularies] and can have multiple formulary prices [productformularyprices] link to it.

Formularies are also grouped into formulaty groups [productformularygroups]. A formulary is linked to one product index [productindexes].

Additional Tables:

[abpbackgroundjobs] – stores the background job details for the import process.

[abpsettings]

* Stores setting (Abp.ProductPrice.Formulary) to disable import & crud operations while an import process is busy : 0:Idle 1: InProcess 2:Complete
* Stores setting (Abp.ProductPrice.Formulary.Tags) containing tag name used by dataset.

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMMAF

…\wwwroot\File-Repository\Unisolv\Data\PHMMAF.vix

…\wwwroot\File-Repository\Unisolv\Data\phmmaf.xfd

**PHMMAF**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | MAF-FILE | string | Code |
| 1 | MAF-PAC-6 | long | Nappi Code used to get row from [productindexes] |
| 2 | MAF-PAC-EXT |  | <Not used> |
| 3 | MAF-PAC-QTY | decimal | Pack Qty |
| 4 | MAF-FRM-DT | DateTime? | Effective Date |
| 5 | MAF-SEQ | int | Sequence – defaults to 0 if not supplied |
| 6 | MAF-MAIN-GP | int | Group |
| 7 | MAF-SUB-GP | int | Subgroup |
| 8 | MAF-PRI | int | Priority – defaults to 0 if not supplied |
| 9 | MAF-TYP | string | <Not used> |
| 10 | MAF-DESC | string | <Not used> |
| 11 | MAF-MAX-QTY | string | <Not used> |
| 12 | MAF-MAX-VAL | string | <Not used> |
| 13 | MAF-PRC | decimal | Medical Price |
| 14 | MAF-DAYS | string | <Not used> |
| 15 | MAF-MED-CODE | string | <Not used> |
| 16 | MAF-LUSED | DateTime? | Last Modification Time |
| 17 | MAF-DIS-DT | DateTime? | Discontinued Date |
| 18 | MAF-UPD-CNT | string |  |
| 19 | MAF-MAP | string | <Not used> |
| 20 | MAF-PREFER | string | <Not used> |
| 21 | MAF-VERS | string | <Not used> |
| 22 | MAF-GEN-YN | bool | Generic Drug |
| 23 | MAF-PERC | string | <Not used> |

### Functionality

#### File Manager Service

File Manager entries are created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 18,19 |

#### Import Data



This import process will populate tables [productformularygroups], [productformularies] & [productformularyprices]

|  |  |
| --- | --- |
| **Import** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_ProductFormularyImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\  \_ProductFormularyImportModal.js   * //File Select Button |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the file repository that include **PHMMAF** in the filename. |
| **Import** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_ProductFormularyImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\  \_ProductFormularyImportModall.js   * //Import Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportProductFormulary |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Formulary & AppSettings.TenantProcesses.ProductPrice\_Formulary\_Tags settings indicating that import file is processing.  Stored in table [abpsettings]  This setting will prevent import or crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime.    Application\DataBureau\ImportAppService.cs   * ImportData   case UpdateType.ProductFormulary: |

#### Create???



|  |  |
| --- | --- |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml   * //Create Button |

#### Filter Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml   * Show modal embedded in page with id filterModal * //Filter Apply Button * getData   This function will reload data table. |

#### List Page (Data)

This table returns the Formulary Groups.

|  |  |
| --- | --- |
| Formulary Groups Table | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml   * //Formulary Group Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductFormularyGroups   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductFormularyGroups   Returns formulary groups |

#### List Page (Data) – Search

* Search on CODE

#### List Page (Data) – Sorting

* Sort on CODE & TOTAL

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |

#### List Page (Data) – Delete



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |

#### List Page (Data) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductDetails.cshtml |

#### List Page (Data) - Formularies

This sub table for returns formularies for the selected group.

|  |  |
| --- | --- |
| Formularies Sub Table | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml   * //Formularies Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductFormularies   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductFormularies   Returns formularies per group. |

#### List Page (Data) – Formularies - Edit



#### List Page (Data) – Formularies - Change History



#### List Page (Data) – Formularies - Delete



#### List Page (Data) – Formulary Details (Prices)

This sub table returns formulary prices per formulary.

|  |  |
| --- | --- |
| Formulary Price Sub Table | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductFormulary.cshtml   * function subFormularyDetailTableInit (e) { |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductFormularyPrices   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductFormularyPrices   Returns formulary details (prices) |

#### List Page (Versions)

See Versions > List Page (Versions)

## Product Price Equivalence

### Overview

Pharmaceutical Equivalents Drug products in identical dosage forms that contain the same active ingredient (s), ie, the same salt or ester, are of the same dosage form, use the same route of administration, and are identical in strength or concentration can be classified as Product Equivalences.

The WHO assigns **ATC** codes to all active substances contained in **medicines** based on the therapeutic indication for the medicine. Using the **ATC** code, active substances are classified in groups at five different levels according to the organ or system on which they act and their therapeutic, **pharmacological** and chemical properties.

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Products > Equivalence |

### Schema & Data Files

#### Schema

Diagram

Description automatically generated

|  |  |  |
| --- | --- | --- |
| productequivalenceatcs |  |  |
| productequivalencegenerics | Core\DataBureau\Products\ProductEquivalenceGeneric.cs |  |
| productequivalencegenerictranslationactiveingredient | Core\DataBureau\Products\ProductEquivalenceGenericTranslationActiveIngredient.cs |  |

**productequivalenceatc**

This entity contains the active substances contained in the applicable medicine.

**productequivalencegenerics**

This entity contains generic/equivalent products to the products stored in the productindexes entity.

**productequivalencegenerictranslationactiveingredient**

This entity contains active ingredients that is found in a generic/equivalent product.

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMGEN

…\wwwroot\File-Repository\Unisolv\Data\PHMGEN.vix

…\wwwroot\File-Repository\Unisolv\Data\phmgen.xfd

**PHMGEN**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | GEN-NAP | long? | ProductIndexId |
| 1 | GEN-CAT1 | int? | Generic Category |
| 2 | GEN-CAT2 | int? | Generic Sub Category |
| 3 | GEN-PREF | int | Preference |
| 4 | GEN-CAT-DESC | DateTime? | Effective Date |
| 5 | GEN-ATC | string | Active Ingredient Code |
| 6 | GEN-ACT1 | int | Active Ingredient |
| 7 | GEN-ACT2 | int | Active Ingredient |
| 8 | GEN-ACT3 | int | Active Ingredient |
| 9 | GEN-ACT4 | string | Active Ingredient |
| 10 | GEN-ACT5 | string | Active Ingredient |

### Functionality

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 10,11 |

#### Import Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * button id="ImportButton"   The process status of this dataset is stored in table [abpsettings]  If the status is InProgress the Import button will be disabled.  This setting will prevent crud functions to be performed while import process is busy.  You might have to restart the solution, as this status is stored in the zero framework app settings and this will not change if you just change the value in the database. |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Import Button |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_ProductEquivalenceImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\  \_ProductEquivalenceImportModal.js   * //File Lookup |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the file repository that include **PHMGEN** in the filename. |
| **Import** | |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportProductEquivalence |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Index setting indicating that file is processing.  Stored in table [abpsettings]  This setting will prevent import or crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime    Application\DataBureau\ImportAppService.cs   * ImportData |

#### Create Generic

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Create Generic Button |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductEquivalenceGenericModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductEquivalenceGenericModal.js   * //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductEquivalenceGeneric |
| **Job** | If ‘shouldAddToVersion’ true  Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |
| **Job** | If updating existing generic?  Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * UpdateProducts * Compare |

#### Create ATC

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Create ATC Button |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductEquivalenceATCModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductEquivalenceATCModal.js   * //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductEquivalenceATC |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### Filter Data?



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * Filter Button |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Apply Filter Button   Re-populate data tables (Generics Table & ATC Table) |

#### List Page (Data (Generic))

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Generics Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductEquivalenceGenerics   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductEquivalenceGenerics |

#### List Page (Data (Generic)) – Search

* Search on CATEGORY & ACTIVE INGREDIENTS

#### List Page (Data (Generic)) – Sorting

* Sort on CATEGORY, DEFAULTS, ACTIVE INGREDIENTS & TOTAL PRODUCTS

#### List Page (Data (Generic)) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Edit Generic Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductEquivalenceGenericModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductEquivalenceGenericModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductEquivalenceGenericModal.js   * //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductEquivalenceGeneric * CheckATCProducts |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |
| **Job** | Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * UpdateProductDetails |

#### List Page (Data (Generic)) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data (Generic)) – Delete



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Delete Generic Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DeleteProductEquivalenceGeneric |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute   Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs   * Execute * UpdateProducts |

#### List Page (Data (Generic)) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Restore Generic Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * RestoreProductEquivalenceGeneric |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute   Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs   * Execute |

#### List Page (Data (Generic Products))

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Sub Generic Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetGenericProducts   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEquivalenceProducts |

#### List Page (Data (Generic Products)) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Edit Index Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductIndexModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductIndexModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductIndexModal.js   * //Save App Service |

#### List Page (Data (ATC))

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //ATC Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetProductEquivalenceATCs   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetProductEquivalenceATCs |

#### List Page (Data (ATC)) – Search

* Search on ATC

#### List Page (Data (ATC)) – Sorting

* Sort on ATC,GENERICS & TOTAL PRODUCTS

#### List Page (Data (ATC)) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Edit ATC Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductEquivalenceATCModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductEquivalenceATCModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductEquivalenceATCModal.js   * //Save App Service |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveProductEquivalenceATC |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |

#### List Page (Data (ATC)) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data (ATC)) – Delete



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Delete ATC Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DeleteProductEquivalenceATC |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |

#### List Page (Data (ATC)) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Restore ATC Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * RestoreProductEquivalenceATC |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |

#### List Page (Data (ATC Products))

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Sub ATC Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetATCProducts   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEquivalenceProducts |

#### List Page (Data (ATC Products)) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductEquivalences.cshtml   * //Edit Index Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateProductIndexModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateProductIndexModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\  \_CreateUpdateProductIndexModal.js   * //Save App Service |

#### List Page (Versions)

See Versions > List Page (Versions)

## Translation Files – Drug Types

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Translations > Drug Types |

### Schema & Data Files

#### Schema

Diagram

Description automatically generated

|  |  |
| --- | --- |
| translationdrugtypes |  |
| versiontranslationdrugtype |  |

Dependencies

|  |  |
| --- | --- |
| translationdrugtypes.Id | productequivalencegenerics.TranslationDrugTypeId references it  versiontranslationdrugtype.TranslationDrugTypeId references it |

There is no import operation for drug types. Data gets seeded in .EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs > CreateTranslationDrugTypes

Table translationdrugtypes gets populated with the drug type values.

### Functionality

#### Create



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * $('#CreateNewDrugTypeButton').on('click', function (e) { |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * //Create Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateTranslationDrugTypeModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateTranslationDrugTypeModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Translations\  CreateUpdateTranslationDrugTypeModal.js   * //App Service Call |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveTranslationDrugType   Add or update translation drug type.  Utilize background job to update version. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### Filter Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | <button type="button" class="btn btn-secondary pr-1" data-toggle="modal" data-target="#filterModal">  id="filterModal"  function getDrugTypes() { |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTranslationDrugTypes |

#### List Page (Data)

Get all translation drug types from table ‘translationdrugtypes’

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * ajaxFunction: \_dataBureauService.getTranslationDrugTypes, |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTranslationDrugTypes |

#### List Page (Data) – Search

* Search on CODE & DESCRIPTION

#### List Page (Data) - Sorting

* Sort on CODE, DESCRIPTION & DIRECTIONS

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | $('#DrugTypesTable tbody').on('click', '[id\*=btnEdit]', function () { |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateTranslationDrugTypeModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Translations\   * \_CreateUpdateTranslationDrugTypeModal.js   await \_dataBureauService.saveTranslationDrugType(data) |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveTranslationDrugType   Add or update translation drug type.  Utilize background job to update version. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### List Page (Data) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | $('#DrugTypesTable tbody').on('click', '[id\*=btnHistory]', function () {   * The script for this modal view is in the version script file:   Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |

#### List Page (Data) – Delete



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * $('#DrugTypesTable tbody').on('click', '[id\*=btnDelete]', function () { * deleteTranslationDrugType |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DeleteTranslationDrugType   Deleting a drug type requires checking how this will affect the drug type versioning as well as product index and generic category. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs  Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs  Application\BackgroundJobs\Imports\GenericCategoryUpdateJob.cs |

#### List Page (Data) – Restore

Becomes available when drug type has been deleted and ‘Filter Data’, ‘Show Deleted’ enabled.



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * $('#DrugTypesTable tbody').on('click', '[id\*=btnRestore]', function () * restoreTranslationDrugType |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * RestoreTranslationDrugType   Restoring a drug type requires checking how this will affect the drug type versioning as well as product index and generic category. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs  Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs  Application\BackgroundJobs\Imports\GenericCategoryUpdateJob.cs |

#### List Page (Versions)

See Versions > List Page (Versions)

### Backward Compatibility

|  |  |
| --- | --- |
| Operation | Entity Updates |
| Add | translationdrugtypes |
| Update Descriptions or Directions | translationdrugtypes |
| Delete | translationdrugtypes  productindexes  productequivalencegenerics |

## Translation Files – Active Ingredients

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Translations > Active Ingredients |

### Schema & Data Files

#### Schema

Text

Description automatically generated with medium confidence

|  |  |
| --- | --- |
| abpbackgroundjobs | Contains information about the background job executing the import process |
| abpsettings | Stores settings used to disable import and crud functions when import process busy |
| translationactiveingredients | Contains active ingredients |
| productequivalencegenerictranslationactiveingredient |  |
| versiontranslationactiveingredient | Link table between translationactiveingredients and updateversions. This enable every active ingredient to be linked to a specific version |
| Updatepriceversions | Contains version information |

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMACT

…\wwwroot\File-Repository\Unisolv\Data\PHMACT.vix

…\wwwroot\File-Repository\Unisolv\Data\phmact.xfd

**PHMACT**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | ACT-CODE | string | Code |
| 1 | ACT-DESC | string | Description |

### Functionality

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 12,13 |

#### Import Data



|  |  |
| --- | --- |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the file repository that include PHMACT in the filename. |
| **Import** | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\\_TranslationActiveIngredientImportModal.cshtml |
| **Script** | .Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\   * \_TranslationActiveIngredientImportModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportProductIndex |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute   If you have a xfd file you can create a delimited output file during run-time (XFDJob.cs).  If you do not have a xfd file you can use the vtutil process to create an output file (VutilJob.cs).   * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Index setting indicating that file is processing.  Stored in table [abpsettings]  This setting will prevent crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime    Application\DataBureau\ImportAppService.cs   * ImportData |

#### Create



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml   * $('#CreateNewActiveIngredientButton').on('click', function (e) { |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateTranslationActiveIngredientModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Translations\   * \_CreateUpdateTranslationActiveIngredientModal.js   await \_dataBureauService.saveTranslationActiveIngredient(data) |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveTranslationActiveIngredient   Add or update active ingredients.  Utilize background job to update version. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare |

#### Filter Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml |
| **Script** | * <button type="button" class="btn btn-secondary pr-1" data-toggle="modal" data-target="#filterModal"> * id="filterModal" * $('#ApplyNow').on('click', function (e) { * function getData() { |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTranslationActiveIngredients |

#### List Page (Data)

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml   * ajaxFunction: \_dataBureauService.getTranslationActiveIngredients, |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTranslationActiveIngredients |

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml |
| **Script** | $('#DataTable tbody').on('click', '[id\*=btnEdit]', function () { |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateTranslationActiveIngredientModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Translations\   * \_CreateUpdateTranslationActiveIngredientModal.js   await \_dataBureauService. aveTranslationActiveIngredient (data) |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveTranslationActiveIngedient   Add or update translation active ingredient.  Utilize background job to update version. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs   * Execute |
| **App Service** | Application\DataBureau\ImportAppService.cs   * Compare   The arguments for this backg  Input to method: |
| **Job** | Application\DataBureau\ImportAppService.cs   * Compare   The arguments for this background job are of type Tuple, which enables you to sent a multi-dimensional array of different data types as argument.  Input to method:    **List<Tuple<T, U, string, UpdateType>**  T – entity added or updated, represented by Item1.  U – current entity, represented by Item2, Id will be 0 if an entity is being added.  string – contains string of tags, represented by Item3  UpdateType – enum type indicating the entity updated, represented by item4  Split tags (Item3) into an array and iterate through it.  USE cortex;  SELECT  \*  FROM  tags  LEFT OUTER JOIN updatepriceversions ON updatepriceversions.TagId = tags.Id  LEFT OUTER JOIN versiontranslationactiveingredient ON versiontranslationactiveingredient.UpdatePriceVersionId = updatepriceversions.Id  LEFT OUTER JOIN translationactiveingredients ON translationactiveingredients.Id = versiontranslationactiveingredient.TranslationActiveIngredientId;  Get latest version by ordering by LastOrDefault.  Check version status (Draft, Committed, Published or UnPublished)  If version is not a draft, a new version must be created because the current version with this entity details are committed or published.  Create a new version row in table [updatepriceversion] with a new version number 1 more than previous.  This new version will be a draft version by default.  Get versionId of this new version.  Loop through entity list, represented by Item1 |
|  |  |

#### List Page (Data) – Change History



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml |
| **Script** | $('#DrugTypesTable tbody').on('click', '[id\*=btnHistory]', function () {   * The script for this modal view is in the version script file:   Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery |

#### List Page (Data) – Delete



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ActiveIngredients.cshtml   * $('#DrugTypesTable tbody').on('click', '[id\*=btnDelete]', function () { * deleteTranslationDrugType |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DeleteTranslationActiveIngredient   Deleting a drug type requires checking how this will affect the drug type versioning as well as product index and generic category. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs  Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs  Application\BackgroundJobs\Imports\GenericCategoryUpdateJob.cs |

#### List Page (Data) – Restore

Becomes available when drug type has been deleted and ‘Filter Data’, ‘Show Deleted’ enabled.



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\DrugTypes.cshtml   * $('#DrugTypesTable tbody').on('click', '[id\*=btnRestore]', function () * restoreTranslationDrugType |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * RestoreTranslationDrugType   Restoring a drug type requires checking how this will affect the drug type versioning as well as product index and generic category. |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs  Application\BackgroundJobs\Imports\ProductIndexUpdateJob.cs  Application\BackgroundJobs\Imports\GenericCategoryUpdateJob.cs |

#### List Page (Versions)

See Versions > List Page (Versions)

### Backward Compatibility

|  |  |
| --- | --- |
| Operation | Entity Updates |
| Add | PHMACT |
| Update Descriptions only | PHMACT |
| Delete | PHMACT |

## Translation Files – Manufacturers

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Product Price Index > Translations > Manufacturers |

### Schema & Data Files

#### Schema

|  |  |
| --- | --- |
| translationmanufacturer |  |

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMSPL

…\wwwroot\File-Repository\Unisolv\Data\PHMSPL.vix

…\wwwroot\File-Repository\Unisolv\Data\phmspl.xfd

### Functionality

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 14,15 |

#### Import Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\\_TranslationManufacturerImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\   * \_TranslationManufacturerImportModal.js |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems |
| **Import**  Run Unisolv Runtime @ C:\Unisolv\888.bat before you do the import. | |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportTranslationManufacturer |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute   If you have a xfd file you can create a delimited output file during run-time (XFDJob.cs).  If you do not have a xfd file you can use the vtutil process to create an output file (VutilJob.cs).   * UpdateStatus   Update AppSettings.TenantProcesses.ProductPrice\_Index setting indicating that file is processing.  Stored in table [abpsettings]  This setting will prevent crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime    Application\DataBureau\ImportAppService.cs   * ImportData * If the 'TranslationDrugTypeText' have a value, but 'TranslationDrugTypeId' is empty, a new entity must be added to the TranslationDrugType repo. * Also, a comparison needs to be done, to establish if this new added entity will cause the creation of a new version. * If the current version is in 'Draft' mode, this will not cause a new version, but if the version is 'Committed', this will cause the creation of a new version (the new version will default to 'Draft' status, so father updates will not cause a new version again. * If the 'TranslationManufacturerText' have a value, but 'TranslationManufacturerId' is empty, a new entity must be added to the TranslationManufacturer repo. * Also, a comparison needs to be done, to establish if this new added entity will cause the creation of a new version. * If the current version is in 'Draft' mode, this will not cause a new version, but if the version is 'Committed', this will cause the creation of a new version (the new version will default to 'Draft' status, so further updates will not cause a new version again. |

#### List Page (Data)

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Manufacturers.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\Manufacturers.cshtml   * ajaxFunction: \_dataBureauService.getTranslationManufacturers, |
| **Controller** | Application\DataBureau\DataBureauAppService.cs   * GetTranslationManufacturers |

#### List Page (Versions)

See Versions > List Page (Versions)

#### Create/Update

|  |  |
| --- | --- |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateTranslationManufacturerModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Translations\   * \_CreateUpdateTranslationManufacturerModal.js |

## Medical Aid Schemes

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Medical Aids > Schemes |

### Schema & Data Files

#### Schema

Diagram

Description automatically generated

#### Data Files

…\wwwroot\File-Repository\Unisolv\Data\PHMSCH

…\wwwroot\File-Repository\Unisolv\Data\PHMSCH.vix

…\wwwroot\File-Repository\Unisolv\Data\phmsch.xfd

| | | | | | | |

**PHMSCH**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Type | Description |
| 0 | MD-RECORD | string |  |
| 1 | MD-SCH-KEY |  |  |
| 2 | MD-SCH |  | Scheme Code |
| 3 | MD-CODE | string | Option Code |
| 4 | MD-NAME-KEY |  | Option Description |
| 5 | MD-NAME |  |  |
| 6 | MD-NAME-CHR1 |  |  |
| 7 | MD-NAME-CHR2 |  |  |
| 8 | MD-NAME-CHR3 |  |  |
| 9 | MD-NAME-CHR4 |  |  |
| 10 | MD-NAME-CHR5 |  |  |
| 11 | MD-NAME-CHR6 |  |  |
| 12 | MD-NAME-CHR7 |  |  |
| 13 | MD-NAME-CHR8 |  |  |
| 14 | MD-NAME-CHR9 |  |  |
| 15 | MD-NAME-CHR10 |  |  |
| 16 | MD-NAME-CHR11 |  |  |
| 17 | MD-NAME-CHR12 |  |  |
| 18 | MD-NAME-CHR13 |  |  |
| 19 | MD-NAME-CHR14 |  |  |
| 20 | MD-NAME-CHR15 |  |  |
| 21 | MD-NAME-CHR16 |  |  |
| 22 | MD-NAME-CHR17 |  |  |
| 23 | MD-NAME-CHR18 |  |  |
| 24 | MD-NAME-CHR19 |  |  |
| 25 | MD-NAME-CHR20 |  |  |
| 26 | MD-NAME-CHR21 |  |  |
| 27 | MD-NAME-CHR22 |  |  |
| 28 | MD-NAME-CHR23 |  |  |
| 29 | MD-NAME-CHR24 |  |  |
| 30 | MD-NAME-CHR25 |  |  |
| 31 | MD-NAME-CHR26 |  |  |
| 32 | MD-NAME-CHR27 |  |  |
| 33 | MD-NAME-CHR28 |  |  |
| 34 | MD-NAME-CHR29 |  |  |
| 35 | MD-NAME-CHR30 |  |  |
| 36 | MD-SCHEME-KEY |  |  |
| 37 | MD-SCHEME | string | Scheme Description |
| 38 | MD-UPD-YN | bool | Update Y/N |
| 39 | MD-IN-YMD | DateTime? | Effective Date |
| 40 | MD-EX-YMD | DateTime? | Expire Date |
| 41 | MD-FILLER |  |  |
| 42 | MD-DATE |  |  |

### Functionality

#### File Manager Service

File Manager entries are created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 20,21 |

#### Import Data



This import process will populate tables [medicalaidschemes]

|  |  |
| --- | --- |
| **Import** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_MedicalAidSchemeImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\  \_MedicalAidSchemeImportModal.js   * //File Select Button |
| **File Lookup** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems   Returns files stored in the file repository that include **PHMSCH** in the filename. |
| **Import** | |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_MedicalAidSchemeImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\  \_ MedicalAidSchemeImportModal.js   * //Import Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportMedicalAidScheme |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs   * Execute * UpdateStatus   Update AppSettings.TenantProcesses.MedicalAid\_Scheme & AppSettings.TenantProcesses. MedicalAid\_Scheme\_Tags settings indicating that import file is processing.  Stored in table [abpsettings]  This setting will prevent import or crud functions to be performed while import process is busy.   * CreateXFDUnisolvOutput   Set file properties.  Update bat file.  Create 'Data Output' folder (if applicable)  Create output file using Unisolv runtime.    Application\DataBureau\ImportAppService.cs   * ImportData   case UpdateType.MedicalAidScheme: |

#### Create

NOTE: When creating new Medical Aid Scheme row, set Effective Date to the date row was added?



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Create Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateMedicalAidSchemeModal |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateMedicalAidSchemeModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\MedicalAid\  \_CreateUpdateMedicalAidSchemeModal.js   * //Save Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveMedicalAidScheme * Add / Update Medical Aid Scheme |

#### Filter Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * The Filter button toggles a modal view embedded in the page |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Filter Apply Button * Calls a function ‘getData’ to reload the data table with selected search criteria. * Same applies as with List Page (Data) functionality. |
| **Modal View** |  |

#### List Page (Data)

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Medical Aid Scheme Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetMedicalAidSchemes   This method facilitates the functionality (paging, sorting & filtering) of the KT table used and invokes an App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetMedicalSchemes   Returns list of medical schemes from table [medicalaidschemes] |

#### List Page (Data) – Search

* Search on CODE, DESCRIPTION, OPTION CODE & DESCRIPTION

#### List Page (Data) – Sorting

* Sort on CODE, DESCRIPTION, OPTION CODE, DESCRIPTION, EFFECTIVE DATE & EXPIRE DATE

#### List Page (Data) – Edit



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Edit Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * CreateUpdateMedicalAidSchemeModal |
| **Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateMedicalAidSchemeModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\MedicalAid\  \_CreateUpdateMedicalAidSchemeModal.js   * //Save Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * SaveMedicalAidScheme * Add / Update Medical Aid Scheme |

#### List Page (Data) – Change History

Medical Aid Scheme Entity changes does not seem to be captured in table [abpentitychanges].



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //History Button |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * HistoryModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_HistoryModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //History Table |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetEntityChanges   Input to method:    UpdateType will indicate the entity type and EntityId is the table id.   * CreateEntityChangesAndUsersQuery   Get rows from table [abpentitychanges].  Use ‘UpdateType’ value from input to filter on [abpentitychanges].EntityTypeFullName   * ConvertToEntityChangeListDtos   Map results to model ‘HistoryChangeListDto’ for display. |
| Show Entity Change Details | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_HistoryModal.js   * //Entity Change Modal |
| **Controller** | Web.Mvc\Areas\App\Controllers\AuditLogsController.cs   * EntityChangeDetailModal |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\AuditLogs\\_EntityChangeDetailModal.cshtml |

#### List Page (Data) – Discontinue



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Discontinue Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueMedicalAidScheme |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs |

#### List Page (Data) – Restore



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAids.cshtml   * //Restore Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * DiscontinueRestoreMedicalAidScheme |
| **Job** | Application\BackgroundJobs\Imports\CompareUpdateJob.cs |

#### List Page (Versions)

See Versions > List Page (Versions)

## Medical Aid Option Information

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Dev Ops User Story | 1117 |
|  |  |

### Schema & Data Files

#### Schema

|  |  |
| --- | --- |
| medicaloptions |  |
| medicaloptioncosts |  |
| medicaloptionmessages |  |
| medicaloptionschedules |  |
| medicaloptionconfiguration |  |
| versionmedicaloption |  |
| updatepriceversions |  |

#### Data Files

\wwwroot\File-Repository\Unisolv\Data\PHMCEN

\wwwroot\File-Repository\Unisolv\Data\PHMCEN.vix

\wwwroot\File-Repository\Unisolv\Data\phmcen.xfd

|  |  |
| --- | --- |
| Column | Description |
| MD-PAY-OFFICE | claimadmins.Code |
| ME-SCH | medicalaidschemes.SchemeCode |
| ME-CODE | OptionCode |

When importing Medical Aid Options, you should link the medicalaidoptions.MedicalAidSchemeId by using the ME-SCH and ME-CODE columns.

### Functionality

#### List Page (Data)

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\MedicalAidOptions.cshtml |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetMedicalAidOptions |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetMedicalSchemeOptions |
| **Dto** | Application.Shared\DataBureau\Dto\MedicalAid   * MedicalAidOptionsListDto.cs * MedicalAidOptionsLookupRequest.cs |
| **Schema** | medicaloptions  medicalaidscheme |

#### List Page (Versions)

See Versions > List Page (Versions)

#### Import Data



|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\\_MedicalAidOptionImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\\_MedicalAidOptionImportModal.js |
| **File Lookup Modal View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetFileItems |
| Import | |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * ImportMedicalAidOption |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs > Execute  - If you have a xfd file you can create a delimited output file during run-time (XFDJob.cs).  - If you do not have a xfd file you can use the vtutil process to create an output file (VutilJob.cs).  Application\BackgroundJobs\Imports\XFDJob.cs > UpdateStatus  Application\BackgroundJobs\Imports\XFDJob.cs > CreateXFDUnisolvOutput  Application\DataBureau\ImportAppService.cs > ImportData  Application.Shared\DataBureau\Base\Entities\MedicalAidOptionConverterDto.cs |

Note: Run the Unisolv runtime at C:\Unisolv\888.bat

#### Create

|  |  |
| --- | --- |
| **Modal**  **View** |  |
| **Script** |  |

#### Filter Data

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ClaimAdmins.cshtml   * div id="filterModal" |
| **Script** |  |

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 24,25 |

## Claim Admin Office

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Dev Ops User Story | 1262 |

### Schema & Data Files

#### Data Files

\wwwroot\File-Repository\Unisolv\Data\PHMCEN

\wwwroot\File-Repository\Unisolv\Data\PHMCEN.vix

\wwwroot\File-Repository\Unisolv\Data\phmcen.xfd

This is the data files from Unisolv.

### Functionality

#### List Page (Versions)

See Versions > List Page (Versions)

#### Import Data



|  |  |
| --- | --- |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_ClaimAdminImportModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Imports\   * \_ClaimAdminImportModal.js |
| **File Lookup**  **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_FileLookupModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\\_FileLookupModal.js |
| **App Service** | Application\DataBureau\DataBureauAppService.cs > ImportClaimAdmin |
| **Job** | Application\BackgroundJobs\Imports\XFDJob.cs > Execute  - If you have a xfd file you can create a delimited output file during run-time (XFDJob.cs).  - If you do not have a xfd file you can use the vtutil process to create an output file (VutilJob.cs).  Application\BackgroundJobs\Imports\XFDJob.cs > UpdateStatus  Application\BackgroundJobs\Imports\XFDJob.cs > CreateXFDUnisolvOutput  Application\DataBureau\ImportAppService.cs > ImportData |

#### Create

|  |  |
| --- | --- |
| **Modal**  **View** | Web.Mvc\Areas\App\Views\DataBureau\\_CreateUpdateClaimAdminModal.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Admin\  \_CreateUpdateClaimAdminModal.js |

#### Filter Data

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ClaimAdmins.cshtml   * div id="filterModal" |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Admin\  \_CreateUpdateClaimAdminModal.js |

#### List

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ClaimAdmins.cshtml |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetClaimAdministrators |
| **App**  **Service** | Application\DataBureau\DataBureauAppService.cs   * GetClaimAdmins |

#### File Manager Service

File Manager entries will be created during the seed process when solution is started.

|  |  |
| --- | --- |
| **Seed** | EntityFrameworkCore\Migrations\Seed\Tenants\DataBureauCreator.cs   * CreateCommonFiles   Id = 24,25 |

## Versions

### Overview

|  |  |
| --- | --- |
| Solution | Cortex Management Service |
| Navigation | Open the Cortex Management Service   * Login to ‘Default’ tenant as ‘admin’ * Select Data Bureau > Versions |

### Schema

### Functionality

#### List Page

|  |  |
| --- | --- |
| **Tags Table**  This table displays a list of available Tags and the latest version linked to this Tag.  Status:  Draft - changes can be added to this version.  Committed – changes cause a new version to be created.  Published - changes cause a new version to be created. | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml   * //Tags Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersion   This method facilitates functionality (paging, sorting & filtering) of the KT table used and invoke the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTagPriceVersions   This will return all the tags [tags] and linked latest version [updatepriceversion]. |
|  | **Commit**    NOTE: Possible BUG. When version committed existing entity rows are duplicated, but it seems they are still connected to the old version. New entity row id’s, but the same updatepriceversion. |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml   * //Commit Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * CommitUpdatePriceVersion |
|  | **Publish** |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml   * //Publish Button |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * PublishUpdatePriceVersion |
|  | |
| **Tag Versions Table**  This table displays version details for the selected Tag. | |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml   * //Tag Versions Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersionsChanges   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTagPricesVersions |
| **Entity Change Count Table**  This table displays count details of the entities linked to the selected Tag and Version. | |
| **Script** | Web.Mvc\Areas\App\Views\DataBureau\ProductVersions.cshtml   * //Tag Version Entity Count Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersionChanges   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTagPriceVersionChanges |
|  | Core\Webhooks\AppWebhookPublisher.cs   * PublishProductVersionWebhook |
|  | |
| **Entity Change Table**  This table displays details of the changes per entity linked to the selected Tag and Version. | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * //Tag Version Entity Change Table |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceEntityChanges   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetPriceEntityChanges |

#### List Page (Versions)

Datasets are linked to a version and a tag.

Tags are stored in table **tags**. Versions are stored in table **updatepriceversion**

|  |  |
| --- | --- |
| **Tag Version Table**  This table displays the latest version per tag and include details such as Status (Draft or Committed) and Total Changes | |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\<DataSetViewName>.cshtml |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * var tagsDataTable = \_$tagsTable.KTDatatable({ |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersion   This method facilitates functionality (paging, sorting & filtering) of the KT table used and invoke the App Service method. |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTagPriceVersions   This will return all the tags [tags] and linked latest version [updatepriceversion]. |
| **Version Table**  This table displays details of the version, showing number of changes per dataset.  Clicking on a Tag value will expand this table. | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * function subTableInit(e) { |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersionChanges   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetTagPriceVersionChanges   Version Changes:  Entity Changes:  Application\DataBureau\DataBureauAppService.cs > GetPriceEntityChanges |
| **Entity Changes Table**  This table displays details od the changes per dataset entity.  Clicking on an Entity will expand this table. | |
| **Script** | Web.Mvc\wwwroot\view-resources\Areas\App\Views\DataBureau\Products\Versions\Index.js   * function subSubTableInit(e) { |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceEntityChanges   This method facilitates functionality (paging, sorting & filtering) of the KT table used and then calls the App Service method |
| **App Service** | Application\DataBureau\DataBureauAppService.cs   * GetPriceEntityChanges |

## Uploads

## Processes

#### Processes

|  |  |
| --- | --- |
| **View** | Web.Mvc\Areas\App\Views\DataBureau\Processes.cshtml |
| **Controller** | Web.Mvc\Areas\App\Controllers\DataBureauController.cs   * GetPriceVersionChanges |
|  | .Core\Configuration\AppSettings.cs |