Sage 300 Web Screens SDK

Creating Clear Statistics Screen - Tutorial

May 2020

The MIT License (MIT)

Copyright © 2020 The Sage Group plc or its licensors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

[1. Overview 4](#_Toc40890071)

[2. A/R Clear Statistics 5](#_Toc40890072)

[2.1 Desktop Screen 5](#_Toc40890073)

[2.2 Web Screen 6](#_Toc40890074)

[3. Solution Wizard 7](#_Toc40890075)

[3.1 Solution Information 8](#_Toc40890076)

[3.2 Kendo UI Information 9](#_Toc40890077)

[3.3 Resource Files 10](#_Toc40890078)

[3.4 Generate Solution 11](#_Toc40890079)

[4. Code Generation Wizard 12](#_Toc40890080)

[4.1 Select Code Type and Credentials 13](#_Toc40890081)

[4.2 Add/Edit Process Entities 14](#_Toc40890082)

[4.3 Select Options 15](#_Toc40890083)

[4.4 Generate Code 16](#_Toc40890084)

[4.5 Wizard Completion 17](#_Toc40890085)

[5. Complete the Screen 18](#_Toc40890086)

[5.1 Clear Statistics Entity Interface 18](#_Toc40890087)

[5.2 Clear Statistics Service Interface 19](#_Toc40890088)

[5.3 Clear Statistics Service 20](#_Toc40890089)

[5.4 Clear Statistics UOW Service 21](#_Toc40890090)

[5.5 Clear Statistics Repository 21](#_Toc40890091)

[5.6 Clear Statistics Model 23](#_Toc40890092)

[5.7 Clear Statistics Model Mapper 28](#_Toc40890093)

[5.8 Clear Statistics Fields 31](#_Toc40890094)

[5.9 Clear Statistics View Model 36](#_Toc40890095)

[5.10 Clear Statistics Internal Controller 40](#_Toc40890096)

[5.11 Clear Statistics Controller 46](#_Toc40890097)

[5.12 Clear Statistics Partial Razor View 49](#_Toc40890098)

[5.13 Clear Statistics Repository JavaScript 55](#_Toc40890099)

[5.14 Clear Statistics Knockout Binding JavaScript 56](#_Toc40890100)

[5.15 Clear Statistics Behaviour JavaScript 58](#_Toc40890101)

[5.16 Clear Statistics Worker Role Data Script 77](#_Toc40890102)

[6. Review the Screen 78](#_Toc40890103)

1. Overview

This document is intended to serve as a tutorial for creating the AR Clear Statistics web screen with the Sage 300 Web SDK Wizard.

The wizard creates the solution, the required framework files and generates as much code as possible. However, the generated code specific to the AR Clear Statistics screen is only for the framework components, such as: Process Button, Options Menu, etc. Thus, there is additional manual work required to complete the screen to include the remaining required controls.

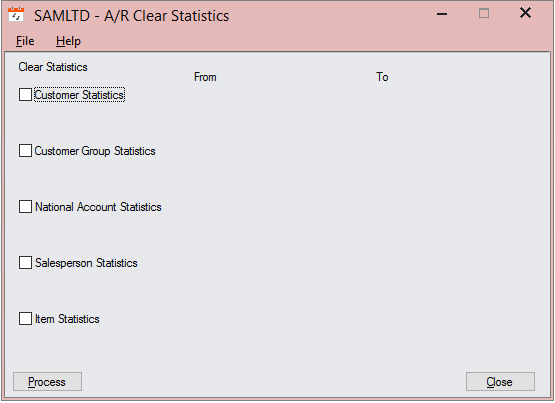
Therefore, this tutorial will:

* Use the Solution Wizard to create the Visual Studio solution for the Valued Partner Company utilizing the TU module
* Use the Code Generation Wizard to create the code files and screen for the AR Clear Statistics Business View (AR0065)
* Use Visual Studio to complete the code required to have a fully functioning AR Clear Statistics screen

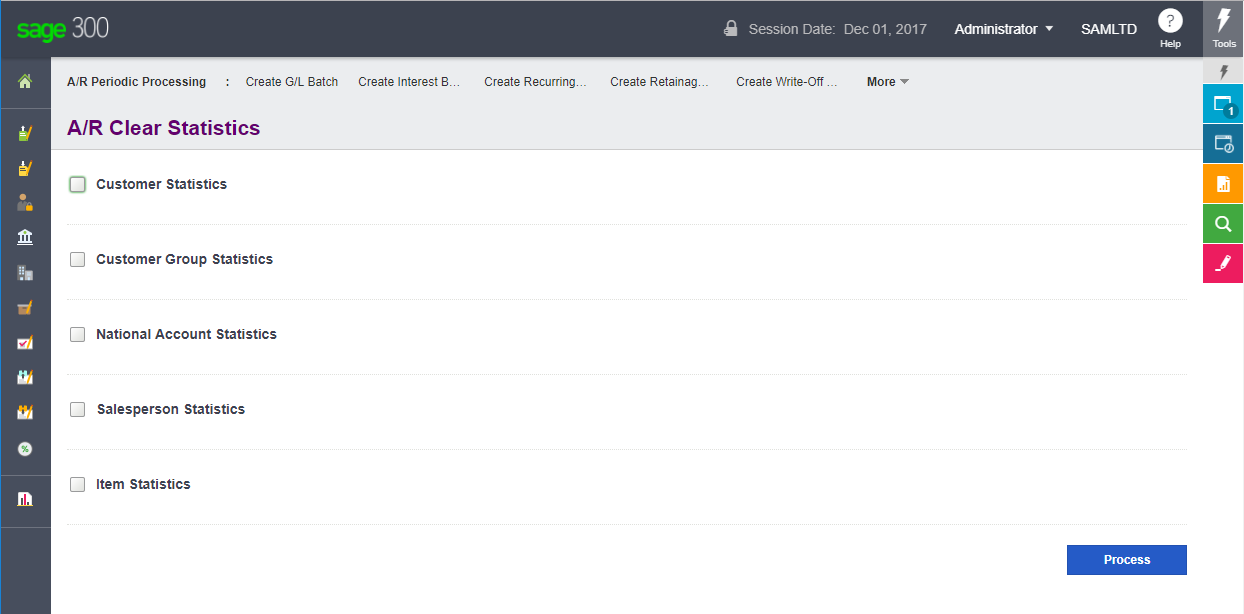
This document’s content is derived from the following documents and these documents should be referred to for further and complete documentation:

* docs\wizards\Sage300SDK\_SolutionWizard.docx
* docs\wizards\Sage300SDK\_CodeGenerationWizard.docx

1. A/R Clear Statistics
   1. Desktop Screen



* 1. Web Screen



1. Solution Wizard

* Launch Visual Studio

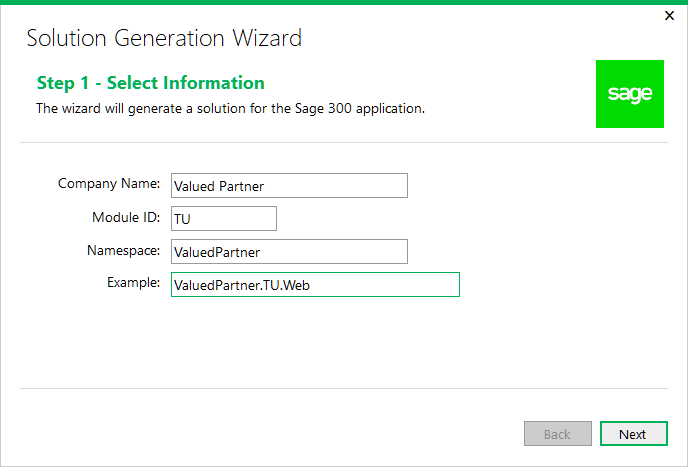
Note: It is recommended to run Visual Studio with administrator privileges

* Select **New/Project** to display the following dialog:



* Enter **ValuedPartner** for the solution name and change the location if needed.
  + Note: The solution created by the AR Payment Codes Tutorial can be used instead of creating a new solution, if already created. In this case, skip the Solution Wizard steps and proceed to the Code Generation Wizard after loading the **ValuedPartner** solution. If the solution does not exist or you wish to place this screen in its own solution, please be sure to specify a name different than **ValuedPartner** for the solution in order to avoid a conflict.
* Press **OK** to create the solution and display the Sage 300 Solution Wizard
  1. Solution Information

After supplying the information required to create a new project (solution), the following dialog box appears:



* Enter Valued Partner for the Company Name field

Used for copyright information and to provide a default value for the Namespace field.

* Enter TU for the Module ID field

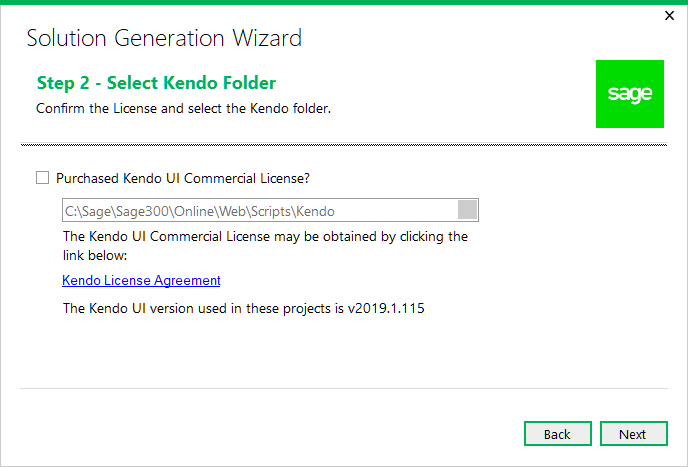
Used as a Module designator that will be used to segregate the projects, and will also become part of the namespace.

* ValuedPartner is defaulted for the Namespace field

Used as the base Namespace value. The namespace will become a composite of this field and the Module ID. This field is defaulted from the Company Name field but may be overridden.

Click Next to proceed.

* 1. Kendo UI Information



Select the Purchased Kendo Commercial License checkbox.

* Purchased Kendo Commercial License

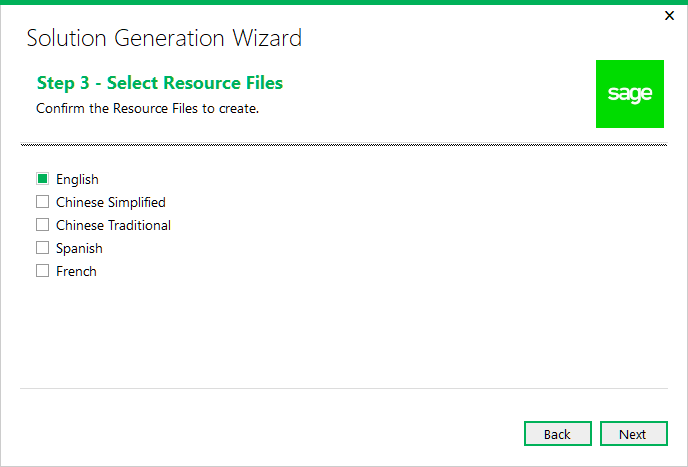
This checkbox must be selected or the Solution Wizard will not proceed. This is the acknowledgment that the user/consumer has purchased a Kendo License since Sage does not distribute the Kendo files for development purposes.

* Kendo Folder

Defaults to the location of the Sage 300 Web installation in order to consume that version of the Kendo file. An alternate folder may be entered, but the Kendo folder must have the version specified as being compatible with Sage 300c Web Screens

Click Next to proceed.

* 1. Resource Files



The following information is required by the wizard:

* English

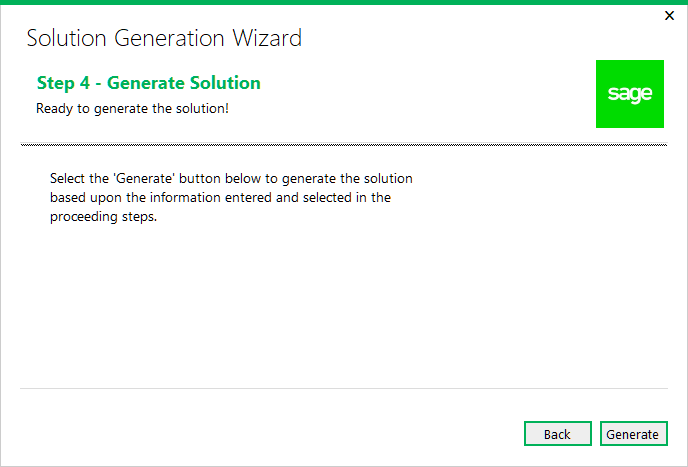
This checkbox is selected by default and cannot be unselected in order to ensure that at least the English Resource Files will be generated in the Resources Project.

* Chinese Simplified, Chinese Traditional, Spanish and French

Select these checkboxes to generate the appropriate language files in the Resources Project. For these optional languages, the keys will be generated with blank values. These blank values must be translated prior to deployment.

Click Next to proceed.

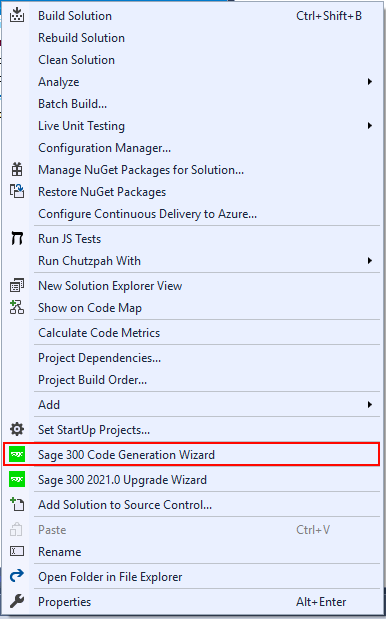
* 1. Generate Solution



Click **Back** to return to the previous step, or click Generate to generate the solution.

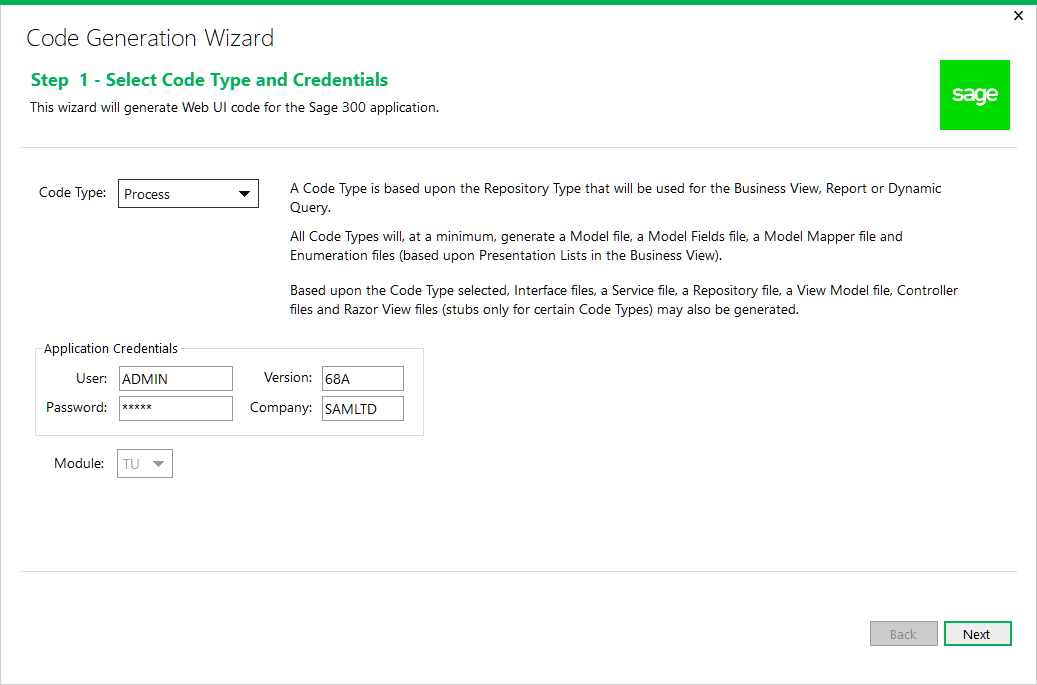
1. Code Generation Wizard

Within Visual Studio, right-click on the solution from the Solution Explorer to display the following context menu:



Select the **Sage 300 Code Generation Wizard** option.

* 1. Select Code Type and Credentials

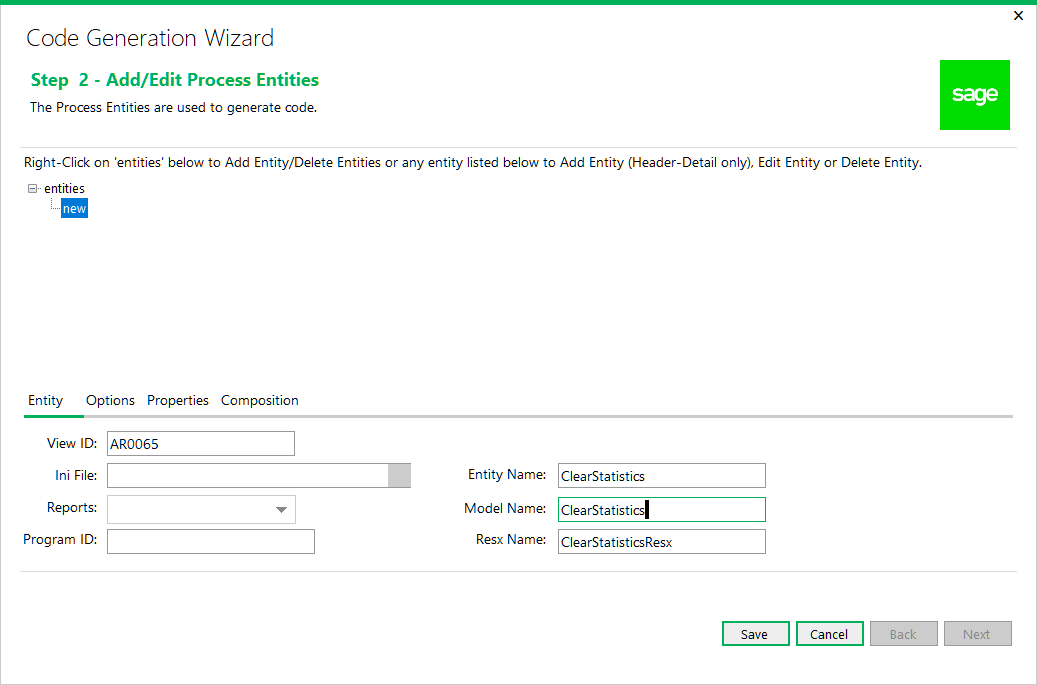


Select Process code type because this is a process screen (super-view and/or long running process).

Ensure the **Application Credentials** and **Module** are correct and change them if necessary.

Click **Next** to proceed.

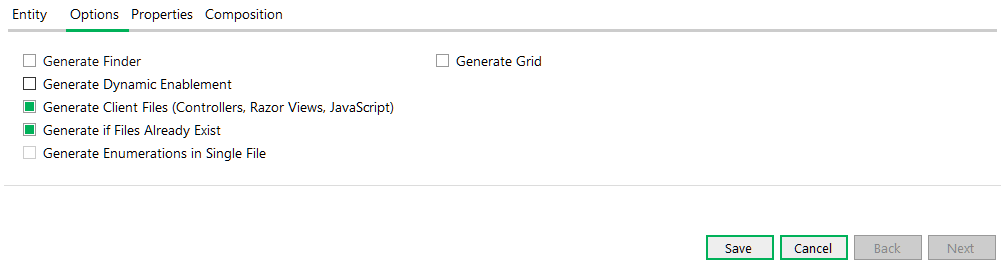
* 1. Add/Edit Process Entities



* Right-click on the ‘**entities**’ tree node and select **Add Entity** to add a new entity.
* Enter **AR0065** for the View ID field.
* After entering the Business View, the Entity Name will default to **ClearStatistic** for the Entity being created, which is generated from the Business View’s description. Change this value to **ClearStatistics**.
* Change the Model Name to **ClearStatistics** as well.
* The Resx Name is defaulted from the Entity Name and can be overridden if needed but for the purposes for this tutorial, keep it as is.
* Click **Save**.

Click on the Options tab.

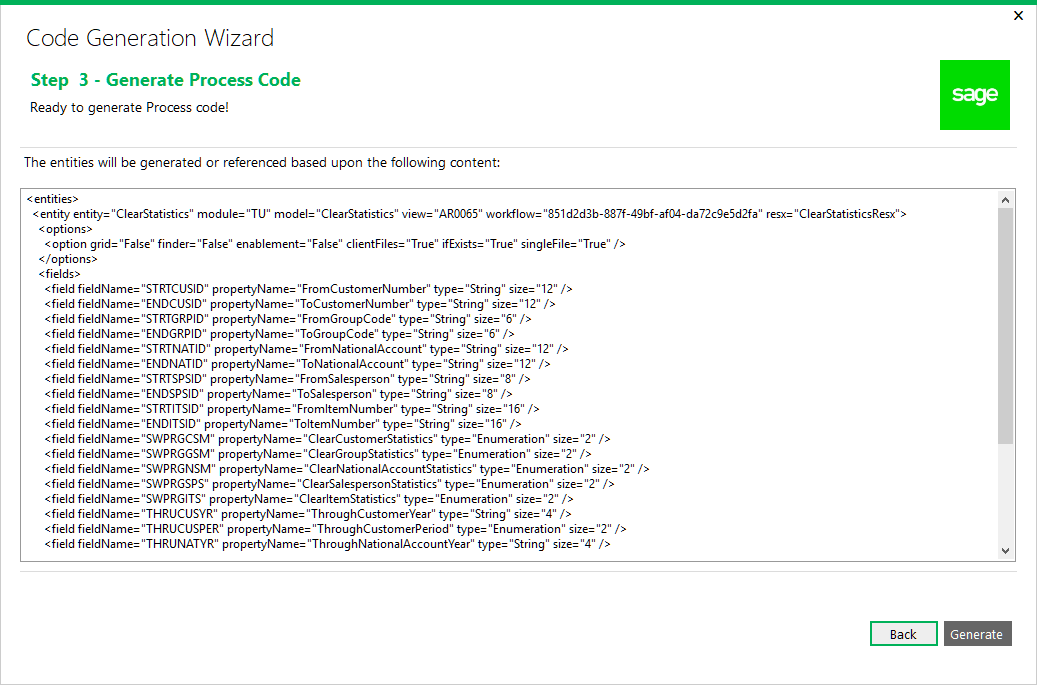
* 1. Select Options



* Note the Generate Finder check box is disabled as a finder is not generated for a Process Type
* The Generate Dynamic Enablement check box is optional and will generate the DynamicAttributes property in the Model Fields class. This provides a reverse mapping of the Server Fields (Business View) to the Model Properties (Business Entity) for those fields in the Business View that have the CheckEditable attribute.
  + This is not applicable for this screen
* The **Generate Client Files** check box, if selected, will generate the client files for a Business View (controllers, Razor Views, and JavaScript files).
* Select the Generate if Files Already Exist check box to allow you to skip files that are generated in case they already exist. An example of this would be if you ran the wizard twice on the same Business View.

Click **Next** to proceed.

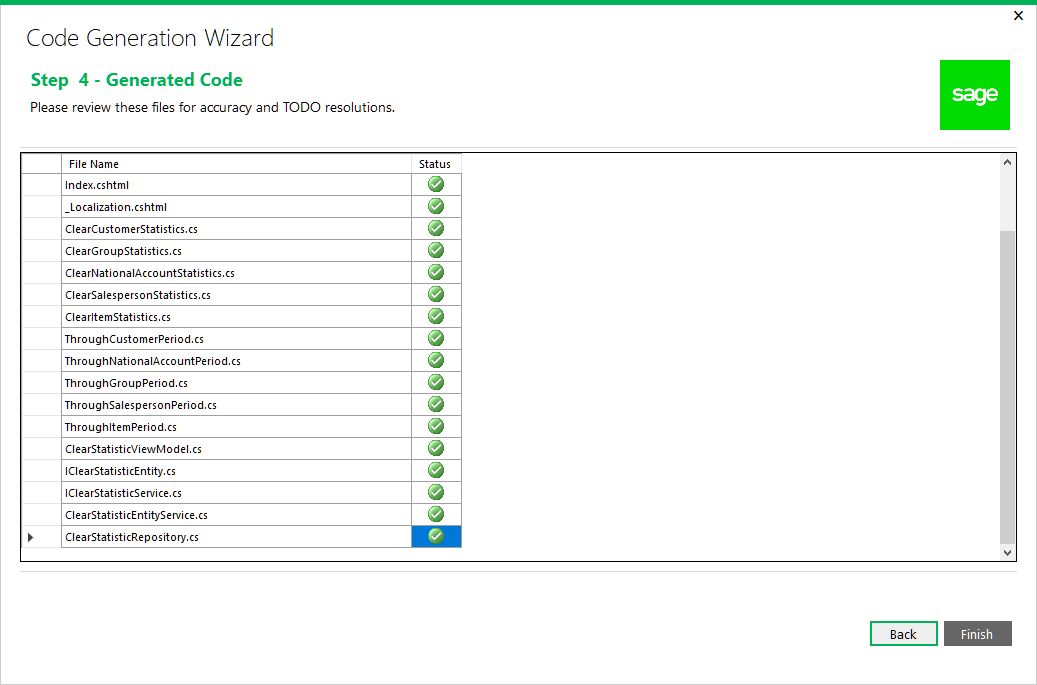
* 1. Generate Code



Verify that the XML shown in the preview is correct.

Click **Next** to proceed.

* 1. Wizard Completion



When the wizard has completed generating the code files based upon the wizard’s inputs, the list of files is displayed. These files, based upon the Generate if Files Already Exist option, have been added to the appropriate projects within the solution.

Note: Nuget Packages may need to be downloaded if your Visual Studio Options are not set to download during build.

Select the **Build\Build Solution** option to compile the solution in order to run the screen.

1. Complete the Screen

The generation activities up to this point have dealt with creating the components required for the A/R Clear Statistics screen and implementing base behaviors and minimal functionality.

The screen is able to be compiled and is functional although processing will not proceed as there is nothing to process without manual implementation.

This section will deal with specific changes to complete the A/R Clear Statistics screen.

* 1. Clear Statistics Entity Interface

Access the **IClearStatisticsEntity.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using ValuedPartner.TU.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Interfaces.Repository;  using System.Collections.Generic;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  #endregion  namespace ValuedPartner.TU.Interfaces.BusinessRepository.Process  {  /// <summary>  /// Interface IClearStatisticsEntity  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public interface IClearStatisticsEntity<T> : IProcessingRepository<T>, ISecurity  where T : ClearStatistics, new()  {  /// <summary>  /// Get Fiscal Years  /// </summary>  /// <returns>Fiscal Years</returns>  List<FiscalPeriod> GetYears();  }  } |

* 1. Clear Statistics Service Interface

Access the **IClearStatisticsService.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using ValuedPartner.TU.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Interfaces.Process;  using System.Collections.Generic;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  #endregion  namespace ValuedPartner.TU.Interfaces.Services.Process  {  /// <summary>  /// Interface IClearStatisticsService  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public interface IClearStatisticsService<T> : IProcessService<T>  where T : ClearStatistics, new()  {  /// <summary>  /// Get Fiscal Years  /// </summary>  /// <returns>Fiscal Years</returns>  List<FiscalPeriod> GetYears();  }  } |

* 1. Clear Statistics Service

Access the **ClearStatisticsService.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System;  using ValuedPartner.TU.Interfaces.BusinessRepository.Process;  using ValuedPartner.TU.Interfaces.Services.Process;  using ValuedPartner.TU.Models;  using ValuedPartner.TU.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Interfaces.Repository.Base;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.Models.Enums.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Services.Process;  using System.Collections.Generic;  #endregion  namespace ValuedPartner.TU.Services.Process  {  /// <summary>  /// Class for ClearStatisticsService  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public class ClearStatisticsService<T> : ProcessService<T, IClearStatisticsEntity<T>>,  IClearStatisticsService<T> where T : ClearStatistics, new()  {  #region Constructor  /// <summary>  /// Constructor for ClearStatistics  /// </summary>  /// <param name="context">Request Context</param>  public ClearStatisticsService(Context context)  : base(context, Guid.Parse("851d2d3b-887f-49bf-af04-da72c9e5d2fa"))  {  }  #endregion  #region Override method  /// <summary>  /// Gets Fiscal Years  /// </summary>  /// <returns>Fiscal Years</returns>  public virtual List<FiscalPeriod> GetYears()  {  using (var repository = Resolve<IClearStatisticsEntity<T>>())  {  return repository.GetYears();  }  }  #endregion  }  } |

* 1. Clear Statistics UOW Service

No changes required in this file.

* 1. Clear Statistics Repository

It is important to mention at this point a concept called Benign Programming or Agnostic Programming. The more generic the class, the less amount of refactoring is to be required if a particular class is copied and pasted into another class.

What does this mean? If the screen has a single business entity, the wizard will name it \_**businessEntity** instead of something specific to the entity. Parameters will be generic unless multiple entities prevents this (such as **id** vs. **paymentCode**)

Access the **ClearStatisticsRepository.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using ValuedPartner.TU.BusinessRepository.Mappers.Process;  using ValuedPartner.TU.Interfaces.BusinessRepository.Process;  using ValuedPartner.TU.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.BusinessRepository;  using Sage.CA.SBS.ERP.Sage300.Common.BusinessRepository.Base;  using Sage.CA.SBS.ERP.Sage300.Common.Interfaces.Entity;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.Models.Enums;  using System.Collections.Generic;  using System;  #endregion  namespace ValuedPartner.TU.BusinessRepository.Process  {  /// <summary>  /// Class ClearStatistics Repository  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public class ClearStatisticsRepository<T> : ProcessingRepository<T>,  IClearStatisticsEntity<T> where T : ClearStatistics, new()  {  #region Business Entity  /// <summary>  /// Business Entity  /// </summary>  private IBusinessEntity \_businessEntity;  #endregion  #region Constructor  /// <summary>  /// Constructor for ClearStatistics  /// </summary>  /// <param name="context">Context</param>  public ClearStatisticsRepository(Context context)  : base(context, new ClearStatisticsMapper<T>(context), ObjectPoolType.Default, BusinessEntitySessionParams.Get(context))  {  }  /// <summary>  /// Constructor for ClearStatistics  /// </summary>  /// <param name="context">Context</param>  /// <param name="session">Business Entity Session</param>  public ClearStatisticsRepository(Context context, IBusinessEntitySession session)  : base(context, new ClearStatisticsMapper<T>(context), session)  {  }  #endregion  #region Abstract methods  /// <summary>  /// Create Business Entity  /// </summary>  /// <returns>IBusinessEntity</returns>  protected override IBusinessEntity CreateBusinessEntities()  {  \_businessEntity = OpenEntity(ClearStatistics.EntityName);  return \_businessEntity;  }  #endregion  #region Public method  /// <summary>  /// To get the fiscal calender years  /// </summary>  /// <returns>fiscal years</returns>  public virtual List<FiscalPeriod> GetYears()  {  var fiscalCalendar = Session.GetFiscalCalendar();  var years = new List<FiscalPeriod>();  if (!string.IsNullOrWhiteSpace(fiscalCalendar.FirstYear.Year))  {  var value = Convert.ToInt32(fiscalCalendar.FirstYear.Year);  while (value <= Convert.ToInt32(fiscalCalendar.LastYear.Year))  {  var year = Session.GetYear(Convert.ToString(value));  fiscalCalendar.FirstYear.Year = Convert.ToString(value);  years.Add(new FiscalPeriod  {  Year = fiscalCalendar.FirstYear.Year,  Period = (short)year.Periods  });  value++;  while (Session.GetYear(Convert.ToString(value)) == null &&  value <= Convert.ToInt32(fiscalCalendar.LastYear.Year))  {  value++;  fiscalCalendar.FirstYear.Year = Convert.ToString(value);  }  }  }  return years;  }  #endregion  }  } |

* 1. Clear Statistics Model

Access the **ClearStatistics.cs** class

* Evaluate the attributes that have been added to the generated properties.
* Evaluate the property names that have been generated by the wizard.
* These generated names are not set in stone.
* These names have been generated from the description of the field/property in the business view. If it does not make sense, is not properly cased or whatever, it is the developer’s responsibility to change it.
* Additional attributes may be added later in the development process.

Add the annotation and change the data types on certain properties

|  |
| --- |
| Add the following reference to the **ValuedPartner.TU.Model** project first:  Sage.CA.SBS.ERP.Sage300.AR.Resources |

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System;  using System.ComponentModel.DataAnnotations;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.Models.Attributes;  using Sage.CA.SBS.ERP.Sage300.Common.Resources;  using ValuedPartner.TU.Models.Enums; // For common enumerations  using ValuedPartner.TU.Models.Enums.Process;  using ValuedPartner.TU.Resources; // For common resources  using ValuedPartner.TU.Resources.Process;  #endregion  namespace ValuedPartner.TU.Models.Process  {  /// <summary>  /// Partial class for ClearStatistics  /// </summary>  public partial class ClearStatistics : ModelBase  {  /// <summary>  /// Gets or sets FromCustomerNumber  /// </summary>  [StringLength(12, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "FromCustomerNumber", ResourceType = typeof (ClearStatisticsResx))]  public string FromCustomerNumber { get; set; }  /// <summary>  /// Gets or sets ToCustomerNumber  /// </summary>  [StringLength(12, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ToCustomerNumber", ResourceType = typeof (ClearStatisticsResx))]  public string ToCustomerNumber { get; set; }  /// <summary>  /// Gets or sets FromGroupCode  /// </summary>  [StringLength(6, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "FromGroupCode", ResourceType = typeof (ClearStatisticsResx))]  public string FromGroupCode { get; set; }  /// <summary>  /// Gets or sets ToGroupCode  /// </summary>  [StringLength(6, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ToGroupCode", ResourceType = typeof (ClearStatisticsResx))]  public string ToGroupCode { get; set; }  /// <summary>  /// Gets or sets FromNationalAccount  /// </summary>  [StringLength(12, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "FromNationalAccount", ResourceType = typeof (ClearStatisticsResx))]  public string FromNationalAccount { get; set; }  /// <summary>  /// Gets or sets ToNationalAccount  /// </summary>  [StringLength(12, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ToNationalAccount", ResourceType = typeof (ClearStatisticsResx))]  public string ToNationalAccount { get; set; }  /// <summary>  /// Gets or sets FromSalesperson  /// </summary>  [StringLength(8, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "FromSalesperson", ResourceType = typeof (ClearStatisticsResx))]  public string FromSalesperson { get; set; }  /// <summary>  /// Gets or sets ToSalesperson  /// </summary>  [StringLength(8, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ToSalesperson", ResourceType = typeof (ClearStatisticsResx))]  public string ToSalesperson { get; set; }  /// <summary>  /// Gets or sets FromItemNumber  /// </summary>  [StringLength(16, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "FromItemNumber", ResourceType = typeof (ClearStatisticsResx))]  public string FromItemNumber { get; set; }  /// <summary>  /// Gets or sets ToItemNumber  /// </summary>  [StringLength(16, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ToItemNumber", ResourceType = typeof (ClearStatisticsResx))]  public string ToItemNumber { get; set; }  /// <summary>  /// Gets or sets ClearCustomerStatistics  /// </summary>  [Display(Name = "ClearCustomerStatistics", ResourceType = typeof (ClearStatisticsResx))]  public ClearCustomerStatistics ClearCustomerStatistics { get; set; }  /// <summary>  /// Gets or sets ClearGroupStatistics  /// </summary>  [Display(Name = "ClearGroupStatistics", ResourceType = typeof (ClearStatisticsResx))]  public ClearGroupStatistics ClearGroupStatistics { get; set; }  /// <summary>  /// Gets or sets ClearNationalAccountStatistics  /// </summary>  [Display(Name = "ClearNationalAccountStatistics", ResourceType = typeof (ClearStatisticsResx))]  public ClearNationalAccountStatistics ClearNationalAccountStatistics { get; set; }  /// <summary>  /// Gets or sets ClearSalespersonStatistics  /// </summary>  [Display(Name = "ClearSalespersonStatistics", ResourceType = typeof (ClearStatisticsResx))]  public ClearSalespersonStatistics ClearSalespersonStatistics { get; set; }  /// <summary>  /// Gets or sets ClearItemStatistics  /// </summary>  [Display(Name = "ClearItemStatistics", ResourceType = typeof (ClearStatisticsResx))]  public ClearItemStatistics ClearItemStatistics { get; set; }  /// <summary>  /// Gets or sets ThroughCustomerYear  /// </summary>  [StringLength(4, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ThroughCustomerYear", ResourceType = typeof (ClearStatisticsResx))]  public string ThroughCustomerYear { get; set; }  /// <summary>  /// Gets or sets ThroughCustomerPeriod  /// </summary>  [Display(Name = "ThroughCustomerPeriod", ResourceType = typeof (ClearStatisticsResx))]  public ThroughCustomerPeriod ThroughCustomerPeriod { get; set; }  /// <summary>  /// Gets or sets ThroughNationalAccountYear  /// </summary>  [StringLength(4, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ThroughNationalAccountYear", ResourceType = typeof (ClearStatisticsResx))]  public string ThroughNationalAccountYear { get; set; }  /// <summary>  /// Gets or sets ThroughNationalAccountPeriod  /// </summary>  [Display(Name = "ThroughNationalAccountPeriod", ResourceType = typeof (ClearStatisticsResx))]  public ThroughNationalAccountPeriod ThroughNationalAccountPeriod { get; set; }  /// <summary>  /// Gets or sets ThroughGroupYear  /// </summary>  [StringLength(4, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ThroughGroupYear", ResourceType = typeof (ClearStatisticsResx))]  public string ThroughGroupYear { get; set; }  /// <summary>  /// Gets or sets ThroughGroupPeriod  /// </summary>  [Display(Name = "ThroughGroupPeriod", ResourceType = typeof (ClearStatisticsResx))]  public ThroughGroupPeriod ThroughGroupPeriod { get; set; }  /// <summary>  /// Gets or sets ThroughSalespersonYear  /// </summary>  [StringLength(4, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ThroughSalespersonYear", ResourceType = typeof (ClearStatisticsResx))]  public string ThroughSalespersonYear { get; set; }  /// <summary>  /// Gets or sets ThroughSalespersonPeriod  /// </summary>  [Display(Name = "ThroughSalespersonPeriod", ResourceType = typeof (ClearStatisticsResx))]  public ThroughSalespersonPeriod ThroughSalespersonPeriod { get; set; }  /// <summary>  /// Gets or sets ThroughItemYear  /// </summary>  [StringLength(4, ErrorMessageResourceName = "MaxLength",ErrorMessageResourceType = typeof(AnnotationsResx))]  [Display(Name = "ThroughItemYear", ResourceType = typeof (ClearStatisticsResx))]  public string ThroughItemYear { get; set; }  /// <summary>  /// Gets or sets ThroughItemPeriod  /// </summary>  [Display(Name = "ThroughItemPeriod", ResourceType = typeof (ClearStatisticsResx))]  public ThroughItemPeriod ThroughItemPeriod { get; set; }  #region UI Strings  /// <summary>  /// Gets ClearCustomerStatistics string value  /// </summary>  public string ClearCustomerStatisticsString  {  get { return EnumUtility.GetStringValue(ClearCustomerStatistics); }  }  /// <summary>  /// Gets ClearGroupStatistics string value  /// </summary>  public string ClearGroupStatisticsString  {  get { return EnumUtility.GetStringValue(ClearGroupStatistics); }  }  /// <summary>  /// Gets ClearNationalAccountStatistics string value  /// </summary>  public string ClearNationalAccountStatisticsString  {  get { return EnumUtility.GetStringValue(ClearNationalAccountStatistics); }  }  /// <summary>  /// Gets ClearSalespersonStatistics string value  /// </summary>  public string ClearSalespersonStatisticsString  {  get { return EnumUtility.GetStringValue(ClearSalespersonStatistics); }  }  /// <summary>  /// Gets ClearItemStatistics string value  /// </summary>  public string ClearItemStatisticsString  {  get { return EnumUtility.GetStringValue(ClearItemStatistics); }  }  /// <summary>  /// Gets ThroughCustomerPeriod string value  /// </summary>  public string ThroughCustomerPeriodString  {  get { return EnumUtility.GetStringValue(ThroughCustomerPeriod); }  }  /// <summary>  /// Gets ThroughNationalAccountPeriod string value  /// </summary>  public string ThroughNationalAccountPeriodString  {  get { return EnumUtility.GetStringValue(ThroughNationalAccountPeriod); }  }  /// <summary>  /// Gets ThroughGroupPeriod string value  /// </summary>  public string ThroughGroupPeriodString  {  get { return EnumUtility.GetStringValue(ThroughGroupPeriod); }  }  /// <summary>  /// Gets ThroughSalespersonPeriod string value  /// </summary>  public string ThroughSalespersonPeriodString  {  get { return EnumUtility.GetStringValue(ThroughSalespersonPeriod); }  }  /// <summary>  /// Gets ThroughItemPeriod string value  /// </summary>  public string ThroughItemPeriodString  {  get { return EnumUtility.GetStringValue(ThroughItemPeriod); }  }  #endregion  }  } |

* 1. Clear Statistics Model Mapper

Access the **ClearStatisticsMapper.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.BusinessRepository;  using Sage.CA.SBS.ERP.Sage300.Common.Interfaces.Entity;  using ValuedPartner.TU.Models.Process;  using ValuedPartner.TU.Models.Enums; // For common enumerations  using ValuedPartner.TU.Models.Enums.Process;  #endregion  namespace ValuedPartner.TU.BusinessRepository.Mappers.Process  {  /// <summary>  /// Class for ClearStatistics mapping  /// </summary>  /// <typeparam name="T">ClearStatistics</typeparam>  public class ClearStatisticsMapper<T> : ModelMapper<T> where T : ClearStatistics, new ()  {  #region Constructor  /// <summary>  /// Constructor to set the Context  /// </summary>  /// <param name="context">Context</param>  public ClearStatisticsMapper(Context context)  : base(context)  {  }  #endregion  #region ModelMapper methods  /// <summary>  /// Get Mapper  /// </summary>  /// <param name="entity">Business Entity</param>  /// <returns>Mapped Model</returns>  public override T Map(IBusinessEntity entity)  {  var model = base.Map(entity);  model.FromCustomerNumber = entity.GetValue<string>(ClearStatistics.Index.FromCustomerNumber);  model.ToCustomerNumber = entity.GetValue<string>(ClearStatistics.Index.ToCustomerNumber);  model.FromGroupCode = entity.GetValue<string>(ClearStatistics.Index.FromGroupCode);  model.ToGroupCode = entity.GetValue<string>(ClearStatistics.Index.ToGroupCode);  model.FromNationalAccount = entity.GetValue<string>(ClearStatistics.Index.FromNationalAccount);  model.ToNationalAccount = entity.GetValue<string>(ClearStatistics.Index.ToNationalAccount);  model.FromSalesperson = entity.GetValue<string>(ClearStatistics.Index.FromSalesperson);  model.ToSalesperson = entity.GetValue<string>(ClearStatistics.Index.ToSalesperson);  model.FromItemNumber = entity.GetValue<string>(ClearStatistics.Index.FromItemNumber);  model.ToItemNumber = entity.GetValue<string>(ClearStatistics.Index.ToItemNumber);  model.ClearCustomerStatistics = (ValuedPartner.TU.Models.Enums.Process.ClearCustomerStatistics)(entity.GetValue<int>(ClearStatistics.Index.ClearCustomerStatistics));  model.ClearGroupStatistics = (ValuedPartner.TU.Models.Enums.Process.ClearGroupStatistics)(entity.GetValue<int>(ClearStatistics.Index.ClearGroupStatistics));  model.ClearNationalAccountStatistics = (ValuedPartner.TU.Models.Enums.Process.ClearNationalAccountStatistics)(entity.GetValue<int>(ClearStatistics.Index.ClearNationalAccountStatistics));  model.ClearSalespersonStatistics = (ValuedPartner.TU.Models.Enums.Process.ClearSalespersonStatistics)(entity.GetValue<int>(ClearStatistics.Index.ClearSalespersonStatistics));  model.ClearItemStatistics = (ValuedPartner.TU.Models.Enums.Process.ClearItemStatistics)(entity.GetValue<int>(ClearStatistics.Index.ClearItemStatistics));  model.ThroughCustomerYear = entity.GetValue<string>(ClearStatistics.Index.ThroughCustomerYear);  model.ThroughCustomerPeriod = (ValuedPartner.TU.Models.Enums.Process.ThroughCustomerPeriod)(entity.GetValue<int>(ClearStatistics.Index.ThroughCustomerPeriod));  model.ThroughNationalAccountYear = entity.GetValue<string>(ClearStatistics.Index.ThroughNationalAccountYear);  model.ThroughNationalAccountPeriod = (ValuedPartner.TU.Models.Enums.Process.ThroughNationalAccountPeriod)(entity.GetValue<int>(ClearStatistics.Index.ThroughNationalAccountPeriod));  model.ThroughGroupYear = entity.GetValue<string>(ClearStatistics.Index.ThroughGroupYear);  model.ThroughGroupPeriod = (ValuedPartner.TU.Models.Enums.Process.ThroughGroupPeriod)(entity.GetValue<int>(ClearStatistics.Index.ThroughGroupPeriod));  model.ThroughSalespersonYear = entity.GetValue<string>(ClearStatistics.Index.ThroughSalespersonYear);  model.ThroughSalespersonPeriod = (ValuedPartner.TU.Models.Enums.Process.ThroughSalespersonPeriod)(entity.GetValue<int>(ClearStatistics.Index.ThroughSalespersonPeriod));  model.ThroughItemYear = entity.GetValue<string>(ClearStatistics.Index.ThroughItemYear);  model.ThroughItemPeriod = (ValuedPartner.TU.Models.Enums.Process.ThroughItemPeriod)(entity.GetValue<int>(ClearStatistics.Index.ThroughItemPeriod));  return model;  }  /// <summary>  /// Set Mapper  /// </summary>  /// <param name="model">Model</param>  /// <param name="entity">Business Entity</param>  public override void Map(T model, IBusinessEntity entity)  {  if (model == null)  {  return;  }  entity.SetValue(ClearStatistics.Index.FromCustomerNumber, model.FromCustomerNumber);  entity.SetValue(ClearStatistics.Index.ToCustomerNumber, model.ToCustomerNumber);  entity.SetValue(ClearStatistics.Index.FromGroupCode, model.FromGroupCode);  entity.SetValue(ClearStatistics.Index.ToGroupCode, model.ToGroupCode);  entity.SetValue(ClearStatistics.Index.FromNationalAccount, model.FromNationalAccount);  entity.SetValue(ClearStatistics.Index.ToNationalAccount, model.ToNationalAccount);  entity.SetValue(ClearStatistics.Index.FromSalesperson, model.FromSalesperson);  entity.SetValue(ClearStatistics.Index.ToSalesperson, model.ToSalesperson);  entity.SetValue(ClearStatistics.Index.FromItemNumber, model.FromItemNumber);  entity.SetValue(ClearStatistics.Index.ToItemNumber, model.ToItemNumber);  entity.SetValue(ClearStatistics.Index.ClearCustomerStatistics, model.ClearCustomerStatistics);  entity.SetValue(ClearStatistics.Index.ClearGroupStatistics, model.ClearGroupStatistics);  entity.SetValue(ClearStatistics.Index.ClearNationalAccountStatistics, model.ClearNationalAccountStatistics);  entity.SetValue(ClearStatistics.Index.ClearSalespersonStatistics, model.ClearSalespersonStatistics);  entity.SetValue(ClearStatistics.Index.ClearItemStatistics, model.ClearItemStatistics);  entity.SetValue(ClearStatistics.Index.ThroughCustomerYear, model.ThroughCustomerYear);  entity.SetValue(ClearStatistics.Index.ThroughCustomerPeriod, model.ThroughCustomerPeriod);  entity.SetValue(ClearStatistics.Index.ThroughNationalAccountYear, model.ThroughNationalAccountYear);  entity.SetValue(ClearStatistics.Index.ThroughNationalAccountPeriod, model.ThroughNationalAccountPeriod);  entity.SetValue(ClearStatistics.Index.ThroughGroupYear, model.ThroughGroupYear);  entity.SetValue(ClearStatistics.Index.ThroughGroupPeriod, model.ThroughGroupPeriod);  entity.SetValue(ClearStatistics.Index.ThroughSalespersonYear, model.ThroughSalespersonYear);  entity.SetValue(ClearStatistics.Index.ThroughSalespersonPeriod, model.ThroughSalespersonPeriod);  entity.SetValue(ClearStatistics.Index.ThroughItemYear, model.ThroughItemYear);  entity.SetValue(ClearStatistics.Index.ThroughItemPeriod, model.ThroughItemPeriod);  }  /// <summary>  /// Map Key  /// </summary>  /// <param name="model">Model</param>  /// <param name="entity">Business Entity</param>  /// <exception cref="NotImplementedException"></exception>  public override void MapKey(T model, IBusinessEntity entity)  {  throw new NotImplementedException();  }  #endregion  }  } |

* 1. Clear Statistics Fields

Access the **ClearStatisticsFields.cs** class

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  #endregion  namespace ValuedPartner.TU.Models.Process  {  /// <summary>  /// Contains list of ClearStatistics Constants  /// </summary>  public partial class ClearStatistics  {  /// <summary>  /// Entity Name  /// </summary>  public const string EntityName = "AR0065";  #region Properties  /// <summary>  /// Contains list of ClearStatistics Field Constants  /// </summary>  public class Fields  {  /// <summary>  /// Property for FromCustomerNumber  /// </summary>  public const string FromCustomerNumber = "STRTCUSID";  /// <summary>  /// Property for ToCustomerNumber  /// </summary>  public const string ToCustomerNumber = "ENDCUSID";  /// <summary>  /// Property for FromGroupCode  /// </summary>  public const string FromGroupCode = "STRTGRPID";  /// <summary>  /// Property for ToGroupCode  /// </summary>  public const string ToGroupCode = "ENDGRPID";  /// <summary>  /// Property for FromNationalAccount  /// </summary>  public const string FromNationalAccount = "STRTNATID";  /// <summary>  /// Property for ToNationalAccount  /// </summary>  public const string ToNationalAccount = "ENDNATID";  /// <summary>  /// Property for FromSalesperson  /// </summary>  public const string FromSalesperson = "STRTSPSID";  /// <summary>  /// Property for ToSalesperson  /// </summary>  public const string ToSalesperson = "ENDSPSID";  /// <summary>  /// Property for FromItemNumber  /// </summary>  public const string FromItemNumber = "STRTITSID";  /// <summary>  /// Property for ToItemNumber  /// </summary>  public const string ToItemNumber = "ENDITSID";  /// <summary>  /// Property for ClearCustomerStatistics  /// </summary>  public const string ClearCustomerStatistics = "SWPRGCSM";  /// <summary>  /// Property for ClearGroupStatistics  /// </summary>  public const string ClearGroupStatistics = "SWPRGGSM";  /// <summary>  /// Property for ClearNationalAccountStatistics  /// </summary>  public const string ClearNationalAccountStatistics = "SWPRGNSM";  /// <summary>  /// Property for ClearSalespersonStatistics  /// </summary>  public const string ClearSalespersonStatistics = "SWPRGSPS";  /// <summary>  /// Property for ClearItemStatistics  /// </summary>  public const string ClearItemStatistics = "SWPRGITS";  /// <summary>  /// Property for ThroughCustomerYear  /// </summary>  public const string ThroughCustomerYear = "THRUCUSYR";  /// <summary>  /// Property for ThroughCustomerPeriod  /// </summary>  public const string ThroughCustomerPeriod = "THRUCUSPER";  /// <summary>  /// Property for ThroughNationalAccountYear  /// </summary>  public const string ThroughNationalAccountYear = "THRUNATYR";  /// <summary>  /// Property for ThroughNationalAccountPeriod  /// </summary>  public const string ThroughNationalAccountPeriod = "THRUNATPER";  /// <summary>  /// Property for ThroughGroupYear  /// </summary>  public const string ThroughGroupYear = "THRUGRPYR";  /// <summary>  /// Property for ThroughGroupPeriod  /// </summary>  public const string ThroughGroupPeriod = "THRUGRPPER";  /// <summary>  /// Property for ThroughSalespersonYear  /// </summary>  public const string ThroughSalespersonYear = "THRUSAPYR";  /// <summary>  /// Property for ThroughSalespersonPeriod  /// </summary>  public const string ThroughSalespersonPeriod = "THRUSAPPER";  /// <summary>  /// Property for ThroughItemYear  /// </summary>  public const string ThroughItemYear = "THRUITSYR";  /// <summary>  /// Property for ThroughItemPeriod  /// </summary>  public const string ThroughItemPeriod = "THRUITSPER";  }  #endregion  #region Properties  /// <summary>  /// Contains list of ClearStatistics Index Constants  /// </summary>  public class Index  {  /// <summary>  /// Property Indexer for FromCustomerNumber  /// </summary>  public const int FromCustomerNumber = 1;  /// <summary>  /// Property Indexer for ToCustomerNumber  /// </summary>  public const int ToCustomerNumber = 2;  /// <summary>  /// Property Indexer for FromGroupCode  /// </summary>  public const int FromGroupCode = 3;  /// <summary>  /// Property Indexer for ToGroupCode  /// </summary>  public const int ToGroupCode = 4;  /// <summary>  /// Property Indexer for FromNationalAccount  /// </summary>  public const int FromNationalAccount = 5;  /// <summary>  /// Property Indexer for ToNationalAccount  /// </summary>  public const int ToNationalAccount = 6;  /// <summary>  /// Property Indexer for FromSalesperson  /// </summary>  public const int FromSalesperson = 7;  /// <summary>  /// Property Indexer for ToSalesperson  /// </summary>  public const int ToSalesperson = 8;  /// <summary>  /// Property Indexer for FromItemNumber  /// </summary>  public const int FromItemNumber = 11;  /// <summary>  /// Property Indexer for ToItemNumber  /// </summary>  public const int ToItemNumber = 12;  /// <summary>  /// Property Indexer for ClearCustomerStatistics  /// </summary>  public const int ClearCustomerStatistics = 13;  /// <summary>  /// Property Indexer for ClearGroupStatistics  /// </summary>  public const int ClearGroupStatistics = 14;  /// <summary>  /// Property Indexer for ClearNationalAccountStatistics  /// </summary>  public const int ClearNationalAccountStatistics = 15;  /// <summary>  /// Property Indexer for ClearSalespersonStatistics  /// </summary>  public const int ClearSalespersonStatistics = 16;  /// <summary>  /// Property Indexer for ClearItemStatistics  /// </summary>  public const int ClearItemStatistics = 17;  /// <summary>  /// Property Indexer for ThroughCustomerYear  /// </summary>  public const int ThroughCustomerYear = 18;  /// <summary>  /// Property Indexer for ThroughCustomerPeriod  /// </summary>  public const int ThroughCustomerPeriod = 19;  /// <summary>  /// Property Indexer for ThroughNationalAccountYear  /// </summary>  public const int ThroughNationalAccountYear = 20;  /// <summary>  /// Property Indexer for ThroughNationalAccountPeriod  /// </summary>  public const int ThroughNationalAccountPeriod = 21;  /// <summary>  /// Property Indexer for ThroughGroupYear  /// </summary>  public const int ThroughGroupYear = 22;  /// <summary>  /// Property Indexer for ThroughGroupPeriod  /// </summary>  public const int ThroughGroupPeriod = 23;  /// <summary>  /// Property Indexer for ThroughSalespersonYear  /// </summary>  public const int ThroughSalespersonYear = 24;  /// <summary>  /// Property Indexer for ThroughSalespersonPeriod  /// </summary>  public const int ThroughSalespersonPeriod = 25;  /// <summary>  /// Property Indexer for ThroughItemYear  /// </summary>  public const int ThroughItemYear = 26;  /// <summary>  /// Property Indexer for ThroughItemPeriod  /// </summary>  public const int ThroughItemPeriod = 27;  }  #endregion  }  } |

* 1. Clear Statistics View Model

Access the **ClearStatisticsViewModel.cs** class and add/modify the properties to complete the form.

|  |
| --- |
| Add the following references to the **ValuedPartner.TU.Web** project first:  Sage.CA.SBS.ERP.Sage300.AR.BusinessRepository  Sage.CA.SBS.ERP.Sage300.AR.Interfaces  Sage.CA.SBS.ERP.Sage300.AR.Models  Sage.CA.SBS.ERP.Sage300.AR.Resources  Sage.CA.SBS.ERP.Sage300.AR.Service  Sage.CA.SBS.ERP.Sage300.AR.Web |

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System.Collections;  using System.Collections.Generic;  using ValuedPartner.TU.Models.Process;  using ValuedPartner.TU.Models.Enums.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.Web.Models.Process;  using Sage.CA.SBS.ERP.Sage300.AR.Models.Enums;  #endregion  namespace ValuedPartner.TU.Web.Areas.TU.Models.Process  {  /// <summary>  /// ClearStatisticsViewModel class  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public class ClearStatisticsViewModel<T> : ProcessViewModel<T>  where T : ClearStatistics, new()  {  /// <summary>  /// ClearCustomerStatistics list  /// </summary>  public IEnumerable<CustomSelectList> ClearCustomerStatistics  {  get { return EnumUtility.GetItemsList<ClearCustomerStatistics>(); }  }  /// <summary>  /// ClearGroupStatistics list  /// </summary>  public IEnumerable<CustomSelectList> ClearGroupStatistics  {  get { return EnumUtility.GetItemsList<ClearGroupStatistics>(); }  }  /// <summary>  /// ClearNationalAccountStatistics list  /// </summary>  public IEnumerable<CustomSelectList> ClearNationalAccountStatistics  {  get { return EnumUtility.GetItemsList<ClearNationalAccountStatistics>(); }  }  /// <summary>  /// ClearSalespersonStatistics list  /// </summary>  public IEnumerable<CustomSelectList> ClearSalespersonStatistics  {  get { return EnumUtility.GetItemsList<ClearSalespersonStatistics>(); }  }  /// <summary>  /// ClearItemStatistics list  /// </summary>  public IEnumerable<CustomSelectList> ClearItemStatistics  {  get { return EnumUtility.GetItemsList<ClearItemStatistics>(); }  }  /// <summary>  /// ThroughCustomerPeriods list  /// </summary>  public IEnumerable<CustomSelectList> ThroughCustomerPeriods  {  get { return EnumUtility.GetItemsList<ThroughCustomerPeriod>(); }  }  /// <summary>  /// ThroughNationalAccountPeriods list  /// </summary>  public IEnumerable<CustomSelectList> ThroughNationalAccountPeriods  {  get { return EnumUtility.GetItemsList<ThroughNationalAccountPeriod>(); }  }  /// <summary>  /// ThroughGroupPeriods list  /// </summary>  public IEnumerable<CustomSelectList> ThroughGroupPeriods  {  get { return EnumUtility.GetItemsList<ThroughGroupPeriod>(); }  }  /// <summary>  /// ThroughSalespersonPeriods list  /// </summary>  public IEnumerable<CustomSelectList> ThroughSalespersonPeriods  {  get { return EnumUtility.GetItemsList<ThroughSalespersonPeriod>(); }  }  /// <summary>  /// ThroughItemPeriods list  /// </summary>  public IEnumerable<CustomSelectList> ThroughItemPeriods  {  get { return EnumUtility.GetItemsList<ThroughItemPeriod>(); }  }  /// <summary>  /// Gets or sets Customer,Customer Group and National Statistics Maximum period  /// </summary>  public string MaximumPeriod { get; set; }  /// <summary>  /// Gets or sets Sales Person Maximum period  /// </summary>  public string SalesPersonMaximumPeriod { get; set; }  /// <summary>  /// Gets or sets Item Maximum period  /// </summary>  public string ItemMaximumPeriod { get; set; }  /// <summary>  /// Gets or sets Customer,Customer Group and National Statistics Minimum period  /// </summary>  public int MinimumPeriod { get; set; }  /// <summary>  /// Gets or sets Salesperson Minimum period  /// </summary>  public int SalesPersonMinimumPeriod { get; set; }  /// <summary>  /// Gets or sets Item Minimum period  /// </summary>  public int ItemMinimumPeriod { get; set; }  /// <summary>  /// Gets or sets Current Period for Customer,Customer Group and National Statistics  /// </summary>  public string CustomerStatisticsCurrentPeriod { get; set; }  /// <summary>  /// Gets or sets Current Period for Item Statistics  /// </summary>  public string ItemStatisticsCurrentPeriod { get; set; }  /// <summary>  /// Gets or sets Current Period for Salesperson Statistics  /// </summary>  public string SalesPersonStatisticsCurrentPeriod { get; set; }  /// <summary>  /// Gets or sets Fiscal Calendars  /// </summary>  public List<FiscalPeriod> FiscalCalendars { get; set; }  /// <summary>  /// Date Range  /// </summary>  public int DateRange { get; set; }  /// <summary>  /// Gets or sets Fiscal Year  /// </summary>  public string FiscalYear { get; set; }  /// <summary>  /// Gets or sets Customer Period  /// </summary>  public StatisticsPeriodType ThroughCustomerPeriod { get; set; }  /// <summary>  /// Gets or sets Group Period  /// </summary>  public StatisticsPeriodType ThroughGroupPeriod { get; set; }  /// <summary>  /// Gets or sets National Acct Period  /// </summary>  public StatisticsPeriodType ThroughNationalAcctPeriod { get; set; }  /// <summary>  /// Gets or sets Salesperson Period  /// </summary>  public StatisticsPeriodType ThroughSalesPersonPeriod { get; set; }  /// <summary>  /// Gets or sets Item Period  /// </summary>  public StatisticsPeriodType ThroughItemPeriod { get; set; }  /// <summary>  /// Checks Customer,  /// Customer Group and  /// National Calendar Year or not  /// </summary>  public bool CalendarYear { get; set; }  /// <summary>  /// Checks Item Calendar Year or not  /// </summary>  public bool ItemCalendarYear { get; set; }  /// <summary>  /// Checks Salesperson Calendar Year or not  /// </summary>  public bool SalesCalendarYear { get; set; }  /// <summary>  /// Checks Customer Statistics Calendar Year or not  /// </summary>  public bool IsCustomerCalendarYear { get; set; }  /// <summary>  /// Checks Customer Group Statistics Calendar Year or not  /// </summary>  public bool IsCustomerGroupCalendarYear { get; set; }  /// <summary>  /// Checks National Acct Calendar Year or not  /// </summary>  public bool IsNationalAcctCalendarYear { get; set; }  /// <summary>  /// Checks Sales Person Calendar Year or not  /// </summary>  public bool IsSalespersonCalendarYear { get; set; }  /// <summary>  /// Checks Item Calendar Year or not  /// </summary>  public bool IsItemCalendarYear { get; set; }  }  } |

* 1. Clear Statistics Internal Controller

Internal methods are only used by the public controller and the complexities are placed here as opposed to the public controller.

Access the **ClearStatisticsControllerInternal.cs** class and add/modify the properties to complete the form

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System;  using System.Globalization;  using System.Linq;  using Microsoft.Practices.Unity;  using ValuedPartner.TU.Interfaces.Services.Process;  using ValuedPartner.TU.Models.Process;  using ValuedPartner.TU.Web.Areas.TU.Models.Process;  using Options = Sage.CA.SBS.ERP.Sage300.AR.Models.Options;  using Sage.CA.SBS.ERP.Sage300.AR.Interfaces.Services;  using Sage.CA.SBS.ERP.Sage300.AR.Models.Enums;  using Sage.CA.SBS.ERP.Sage300.AR.Models.Enums.Reports;  using Sage.CA.SBS.ERP.Sage300.Common.Models;  using Sage.CA.SBS.ERP.Sage300.Common.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Web.Controllers.Process;  using Sage.CA.SBS.ERP.Sage300.CS.Interfaces.Services;  using Sage.CA.SBS.ERP.Sage300.CS.Models;  using Sage.CA.SBS.ERP.Sage300.CS.Models.Enums;  #endregion  namespace ValuedPartner.TU.Web.Areas.TU.Controllers.Process  {  /// <summary>  /// ClearStatistics Internal Controller  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public class ClearStatisticsControllerInternal<T> :  ProcessControllerInternal<T, ClearStatisticsViewModel<T>, IClearStatisticsService<T>>  where T : ClearStatistics, new()  {  #region Private variables  /// <summary>  /// Variable for storing context.  /// </summary>  private readonly Context \_context;  /// <summary>  /// Variable for storing Options.  /// </summary>ClearStatisticsViewModel  private Options \_options;  /// <summary>  /// Constant for Minimum Period  /// </summary>  private const int MinimumPeriod = 1;  /// <summary>  /// Constant for Default Period  /// </summary>  private const int DefaultPeriod = 0;  /// <summary>  /// Constant for To Customer  /// </summary>  private const string ToCustomer = "ZZZZZZZZZZZZ";  /// <summary>  /// Constant for To Group Code  /// </summary>  private const string ToGroupCode = "ZZZZZZ";  /// <summary>  /// Constant for To National Account  /// </summary>  private const string ToNationalAccount = "ZZZZZZZZZZZZ";  /// <summary>  /// Constant for To Sales Person  /// </summary>  private const string ToSalesPerson = "ZZZZZZZZ";  /// <summary>  /// Constant for To Item Number  /// </summary>  private const string ToItemNumber = "ZZZZZZZZZZZZZZZZ";  /// <summary>  /// Defining Total Period Count  /// </summary>  private int \_totalPeriodCount;  #endregion  #region Constructor  /// <summary>  /// Constructor for Clear Statistics Controller Internal  /// </summary>  /// <param name="context">Where context is type of current "User Session".</param>  public ClearStatisticsControllerInternal(Context context)  : base(context)  {  \_context = context;  }  #endregion  #region Public methods  /// <summary>  /// Method to Get the View Model Data.  /// </summary>  /// <returns>Clear Statistics View Model</returns>  public override ClearStatisticsViewModel<T> Get()  {  \_options = GetOptions();  var clearStatistics = Service.Get();  int dateRange = 0;  int maximumPeriod;  int salespersonMaximumPeriod;  int itemMaximumPeriod;  var customerStatisticsCurrentPeriod = GetCurrentPeriod(\_options.CustStatisticsYearType,  \_options.CustStatisticsPeriodType, Context.SessionDate,  out maximumPeriod);  var itemStatisticsCurrentPeriod = GetCurrentPeriod(\_options.ItemStatisticsYearType,  \_options.ItemStatisticsPeriodType, Context.SessionDate,  out itemMaximumPeriod);  var salespersonStatisticsCurrentPeriod = GetCurrentPeriod(\_options.SalesStatisticsYearType,  \_options.SalesStatisticsPeriodType, Context.SessionDate,  out salespersonMaximumPeriod);  clearStatistics.ThroughCustomerYear =  clearStatistics.ThroughNationalAcctYear =  clearStatistics.ThroughGroupYear =  clearStatistics.ThroughSalesPersonYear =  clearStatistics.ThroughItemYear =  Context.SessionDate.Year.ToString(CultureInfo.InvariantCulture);  clearStatistics.ToCustomerNumber = ToCustomer;  clearStatistics.ToGroupCode = ToGroupCode;  clearStatistics.ToNationalAccount = ToNationalAccount;  clearStatistics.ToSalesperson = ToSalesPerson;  clearStatistics.ToItemNumber = ToItemNumber;  var companyProfile = GetCompanyProfile();  if (companyProfile != null)  dateRange = companyProfile.CompanyProfileOptions.WarningDateRange;  var viewModel = new ClearStatisticsViewModel<T>  {  Data = clearStatistics,  UserMessage = new UserMessage(clearStatistics),  ProcessResult = new ProcessResult { ProgressMeter = new ProgressMeter() },  MaximumPeriod = maximumPeriod.ToString(CultureInfo.InvariantCulture),  SalesPersonMaximumPeriod = salespersonMaximumPeriod.ToString(CultureInfo.InvariantCulture),  ItemMaximumPeriod = itemMaximumPeriod.ToString(CultureInfo.InvariantCulture),  MinimumPeriod = maximumPeriod > DefaultPeriod ? MinimumPeriod : DefaultPeriod,  SalesPersonMinimumPeriod = salespersonMaximumPeriod > DefaultPeriod ? MinimumPeriod : DefaultPeriod,  ItemMinimumPeriod = itemMaximumPeriod > DefaultPeriod ? MinimumPeriod : DefaultPeriod,  FiscalYear = Context.SessionDate.Year.ToString(CultureInfo.InvariantCulture),  CustomerStatisticsCurrentPeriod = customerStatisticsCurrentPeriod,  ItemStatisticsCurrentPeriod = itemStatisticsCurrentPeriod,  SalesPersonStatisticsCurrentPeriod = salespersonStatisticsCurrentPeriod,  CalendarYear = \_options.CustStatisticsYearType == StatisticsAccumulateYearType.CalendarYear,  ItemCalendarYear = \_options.ItemStatisticsYearType == StatisticsAccumulateYearType.CalendarYear,  SalesCalendarYear = \_options.SalesStatisticsYearType == StatisticsAccumulateYearType.CalendarYear,  ThroughCustomerPeriod = \_options.CustStatisticsPeriodType,  ThroughGroupPeriod = \_options.CustStatisticsPeriodType,  ThroughNationalAcctPeriod = \_options.CustStatisticsPeriodType,  ThroughSalesPersonPeriod = \_options.SalesStatisticsPeriodType,  ThroughItemPeriod = \_options.ItemStatisticsPeriodType,  FiscalCalendars = Service.GetYears(),  DateRange = dateRange,  };  return viewModel;  }  /// <summary>  /// Method to do Process  /// </summary>  /// <param name="model">Model</param>  /// <returns>Clear Statistics View Model</returns>  public override ClearStatisticsViewModel<T> Process(T model)  {  model.ThroughCustomerPeriod = model.ThroughCustomerPeriod.PadLeft(2, '0');  model.ThroughGroupPeriod = model.ThroughGroupPeriod.PadLeft(2, '0');  model.ThroughNationalAcctPeriod = model.ThroughNationalAcctPeriod.PadLeft(2, '0');  model.ThroughSalesPersonPeriod = model.ThroughSalesPersonPeriod.PadLeft(2, '0');  model.ThroughItemPeriod = model.ThroughItemPeriod.PadLeft(2, '0');  return new ClearStatisticsViewModel<T>  {  WorkflowInstanceId = Service.Process(model),  UserMessage = new UserMessage { IsSuccess = true }  };  }  #endregion  #region private methods  /// <summary>  /// Method to get the option  /// </summary>  /// <returns>option</returns>  private Options GetOptions()  {  var optionsService =  Context.Container.Resolve<IOptionsService<Options>>(new ParameterOverride("context", Context));  return optionsService.Get().Items.FirstOrDefault();  }  /// <summary>  /// Get Company Profile  /// </summary>  /// <returns>Company Profile</returns>  private CompanyProfile GetCompanyProfile()  {  var companyProfileRepository =  Context.Container.Resolve<ICompanyProfileService<CompanyProfile>>(new ParameterOverride("context",  Context));  var companyProfile = companyProfileRepository.Get();  return (companyProfile != null && companyProfile.Items != null && companyProfile.Items.Any()  ? companyProfile.Items.First()  : new CompanyProfile());  }  /// <summary>  /// Method to get current period for Customer,Customer Group,National Account,Sales Person and Item Statistics  /// </summary>  /// <param name="yearType">Year Type</param>  /// <param name="periodType">Period Type</param>  /// <param name="sessionDate">Session Date</param>  /// <param name="totalPeriodCount">Total Period Count</param>  /// <returns>Current Period</returns>  private string GetCurrentPeriod(StatisticsAccumulateYearType yearType, StatisticsPeriodType periodType,  DateTime sessionDate, out int totalPeriodCount)  {  string currentPeriod;  totalPeriodCount = 0;  if ((yearType == StatisticsAccumulateYearType.FiscalYear) &&  (periodType == StatisticsPeriodType.FiscalPeriod))  {  var fiscalCalendarService =  Context.Container.Resolve<ICompanyProfileService<CompanyProfile>>(new ParameterOverride("context",  Context));  var fiscalCalendar = fiscalCalendarService.Get().Items.FirstOrDefault();  if (fiscalCalendar != null)  totalPeriodCount = fiscalCalendar.CompanyProfileOptions.NumberofFiscalPeriods ==  NumberofFiscalPeriods.Num12  ? 12  : 13;  currentPeriod = sessionDate.Month.ToString(CultureInfo.InvariantCulture);  return currentPeriod;  }  var dayOfYear = sessionDate.DayOfYear;  switch (periodType)  {  case StatisticsPeriodType.Weekly:  totalPeriodCount = (int)PeriodCount.Weekly;  DayOfWeek week = (new DateTime(sessionDate.Year, 1, 1)).DayOfWeek;  int extraDays = week - DayOfWeek.Sunday;  currentPeriod =  Math.Ceiling(((double)dayOfYear + extraDays) / 7).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Sevendays:  totalPeriodCount = (int)PeriodCount.Sevendays;  currentPeriod = Math.Ceiling((double)dayOfYear / 7).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Biweekly:  totalPeriodCount = (int)PeriodCount.Biweekly;  currentPeriod = Math.Ceiling((double)dayOfYear / 14).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Fourweeks:  totalPeriodCount = (int)PeriodCount.Fourweeks;  currentPeriod = Math.Ceiling((double)dayOfYear / 28).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Monthly:  totalPeriodCount = (int)PeriodCount.Monthly;  currentPeriod = sessionDate.Month.ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Bimonthly:  totalPeriodCount = (int)PeriodCount.Bimonthly;  currentPeriod =  Math.Ceiling((Convert.ToDecimal(sessionDate.Month) / 2)).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Quarterly:  totalPeriodCount = (int)PeriodCount.Quarterly;  currentPeriod = Math.Ceiling((double)sessionDate.Month / 3).ToString(CultureInfo.InvariantCulture);  break;  case StatisticsPeriodType.Semiannually:  totalPeriodCount = (int)PeriodCount.Semiannually;  currentPeriod = Math.Ceiling((double)sessionDate.Month / 6).ToString(CultureInfo.InvariantCulture);  break;  default:  totalPeriodCount = 0;  currentPeriod = "0";  break;  }  return currentPeriod;  }  #endregion  #region internal method  /// <summary>  /// Gets the fiscal year for Customer,Customer Group,National Account,Sales Person and Item Statistics  /// </summary>  /// <param name="year">Year</param>  /// <param name="type">Type</param>  /// <returns>Maximum period if fiscal year is valid or else 0</returns>  internal int GetMaxPeriodForValidYear(string year, string type)  {  var maxPeriod = 0;  \_totalPeriodCount = 0;  var fiscalCalendarService =  Context.Container.Resolve<IFiscalCalendarService<Sage.CA.SBS.ERP.Sage300.CS.Models.FiscalCalendar>>(  new ParameterOverride("context", Context));  if (fiscalCalendarService.IsValid(year))  {  var options = GetOptions();  if (type == "sales")  {  GetCurrentPeriod(options.SalesStatisticsYearType, options.SalesStatisticsPeriodType,  Context.SessionDate, out \_totalPeriodCount);  }  else if (type == "items")  {  GetCurrentPeriod(options.ItemStatisticsYearType, options.ItemStatisticsPeriodType,  Context.SessionDate, out \_totalPeriodCount);  }  else  {  GetCurrentPeriod(options.CustStatisticsYearType, options.CustStatisticsPeriodType,  Context.SessionDate, out \_totalPeriodCount);  }  maxPeriod = \_totalPeriodCount;  }  return maxPeriod;  }  #endregion  }  } |

* 1. Clear Statistics Controller

Public methods are invoked by the MVC framework and from JavaScript. The complexities of the public methods are hidden in the internal controller’s methods.

Access the **ClearStatisticsController.cs** class to complete the form

|  |
| --- |
| // Copyright (c) 2020 Valued Partner All rights reserved.  #region Namespace  using System.Web.Routing;  using Microsoft.Practices.Unity;  using ValuedPartner.TU.Interfaces.Services.Process;  using ValuedPartner.TU.Models.Process;  using ValuedPartner.TU.Models.Enums.Process;  using ValuedPartner.TU.Web.Areas.TU.Models.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Models.Enums;  using Sage.CA.SBS.ERP.Sage300.Common.Web.Controllers.Process;  using Sage.CA.SBS.ERP.Sage300.Common.Web;  using Sage.CA.SBS.ERP.Sage300.Common.Exceptions;  using Sage.CA.SBS.ERP.Sage300.Common.Resources;  using Sage.CA.SBS.ERP.Sage300.AR.Resources.Forms;  #endregion  namespace ValuedPartner.TU.Web.Areas.TU.Controllers.Process  {  /// <summary>  /// Class ClearStatistics Controller  /// </summary>  /// <typeparam name="T">Where T is type of <see cref="ClearStatistics"/></typeparam>  public class ClearStatisticsController<T> :  ProcessController<T, ClearStatisticsViewModel<T> , ClearStatisticsControllerInternal<T>, IClearStatisticsService<T>>  where T : ClearStatistics, new()  {  #region Private Variables  private const string CustomerType = "customer";  private const string SalesType = "sales";  private const string ItemType = "items";  #endregion  #region Constructor  /// <summary>  /// Constructor for ClearStatistics  /// </summary>  /// <param name="container">Unity container</param>  public ClearStatisticsController(IUnityContainer container)  : base(  container,  (context =>new ClearStatisticsControllerInternal<T>(context)),  "TUClearStatistics")  {  }  #endregion  #region Initialize MultitenantControllerBase  /// <summary>  /// Initializes the specified request context.  /// </summary>  /// <param name="requestContext">The request context.</param>  protected override void Initialize(RequestContext requestContext)  {  base.Initialize(requestContext);  ControllerInternal = new ClearStatisticsControllerInternal<T>(Context);  }  #endregion  #region Action  /// <summary>  /// Getting Customer,Group and National Statistics maximum period for a valid year  /// </summary>  /// <param name="year">Fiscal year.</param>  /// <returns>Maximum period if fiscal year is valid or else 0</returns>  public JsonNetResult GetMaxPeriodForValidYear(string year)  {  try  {  return JsonNet(ControllerInternal.GetMaxPeriodForValidYear(year, CustomerType));  }  catch (BusinessException businessException)  {  return JsonNet(BuildErrorModelBase(CommonResx.GetFailedMessage, businessException, ARCommonResx.Customer));  }  }  /// <summary>  /// Getting Sales Person maximum period for a valid year  /// </summary>  /// <param name="year">Fiscal year.</param>  /// <returns>Maximum period if fiscal year is valid or else 0</returns>  public JsonNetResult GetSalesPersonMaxPeriodForValidYear(string year)  {  try  {  return JsonNet(ControllerInternal.GetMaxPeriodForValidYear(year, SalesType));  }  catch (BusinessException businessException)  {  return JsonNet(BuildErrorModelBase(CommonResx.GetFailedMessage, businessException, ARCommonResx.SalespersonStatistics));  }  }  /// <summary>  /// Getting Item maximum period for a valid year  /// </summary>  /// <param name="year">Fiscal year.</param>  /// <returns>Maximum period if fiscal year is valid or else 0</returns>  public JsonNetResult GetItemMaxPeriodForValidYear(string year)  {  try  {  return JsonNet(ControllerInternal.GetMaxPeriodForValidYear(year, ItemType));  }  catch (BusinessException businessException)  {  return JsonNet(BuildErrorModelBase(CommonResx.GetFailedMessage, businessException, ARCommonResx.ItemStatistics));  }  }  #endregion  }  } |

* 1. Clear Statistics Partial Razor View

Access the \_**ClearStatistics.cshtml** file and add the remaining controls to the screen

Add the following strings to the ClearStatisticsResx file:

* “ThroughYear” string with a value of “Through Year”
* “CustomerStatistics” string with a value of “Customer Statistics”
* “ClearStatistics” string with a value of “Clear Statistics”
* “NationalAccountStatistics” string with a value of “National Account Statistics”

Add the following to the \_Localization.cshtml file:

@using ARCommonResx = Sage.CA.SBS.ERP.Sage300.AR.Resources.Forms.ARCommonResx;

CustomerNumberTitle: '@Html.SageRaw(ARCommonResx.CustomerNumber)',

CustomerGroupFinder: '@Html.SageRaw(ARCommonResx.CustomerGroup)',

FiscalYear: '@Html.SageRaw(ARCommonResx.FiscalYear)',

NationalAccountNumberTitle: '@Html.SageRaw(ARCommonResx.NationalAccount)',

SalesPersonFinderTitle: '@Html.SageRaw(ARCommonResx.Salesperson)',

ItemFinderTitle: '@Html.SageRaw(ARCommonResx.ItemNo)',

|  |
| --- |
| @\* Copyright (c) 2020 Valued Partner All rights reserved. \*@  @using ValuedPartner.TU.Resources.Process;  @using Sage.CA.SBS.ERP.Sage300.Common.Web.HtmlHelperExtension  @using ARCommonResx = Sage.CA.SBS.ERP.Sage300.AR.Resources.Forms.ARCommonResx  @using CommonResx = Sage.CA.SBS.ERP.Sage300.Common.Resources.CommonResx  @model ValuedPartner.TU.Web.Areas.TU.Models.Process.ClearStatisticsViewModel<ValuedPartner.TU.Models.Process.ClearStatistics>  <script type="text/javascript">  @Html.ConvertToJsVariableUsingNewtonSoft("ClearStatisticsViewModel", Model)  @Html.ConvertToJsVariableUsingNewtonSoft("screenName", ClearStatisticsResx.Entity);  </script>  @Html.Partial("~/Areas/Core/Views/Shared/\_ProcessingStatus.cshtml")  @{  bool calendarYear = Model.CalendarYear;  var className = "go-group";  if (calendarYear)  {  className = "input-group";  }  }  @{  bool salespersonCalendarYear = Model.SalesCalendarYear;  var classSalesName = "go-group";  if (salespersonCalendarYear)  {  classSalesName = "input-group";  }  }  @{  bool itemCalendarYear = Model.ItemCalendarYear;  var classItemName = "go-group";  if (itemCalendarYear)  {  classItemName = "input-group";  }  }  <div class="form-screen">  <div id="message"></div>  <header>  <section class="header-group-1">  <div class="header-wrapper">  <div class="header-headline">  @Html.SageHeader1Label("lblClearStatistics", ClearStatisticsResx.Entity)  </div>  </div>  </section>  </header>  <div class="sub-section">  <div class="section-heading">  <div class="form-group">  <div class="ctrl-group">  <div>  @Html.KoSageCheckBox("Data\_ClearCustomerStatistics", false, new { @sagechecked = "Data.bClearCustomerStatistics" }, new { })  @Html.SageLabelFor(model => model.Data.ClearCustomerStatistics, new { @class = "bold" })  </div>  </div>  </div>  </div>  <div class="section-body indented" data-bind="visible:Data.bClearCustomerStatistics">  <div class="form-group">  <div class="go-group ">  @Html.SageLabel("lblClearStatistics", CommonResx.From)  @Html.KoSageTextBoxFor(model => model.Data.FromCustomerNumber, new { @sagevalue = "Data.FromCustomerNumber" }, new { @class = "default txt-upper", @id = "Data\_FromCustomerNumber", @autofocus = "autofocus" })  @Html.KoSageButton("btnFromCustomerFinder", null, new { @id = "btnFromCustomerFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  <div class="go-group ">  @Html.SageLabel("ClearStatistic\_To", CommonResx.To)  @Html.KoSageTextBoxFor(model => model.Data.ToCustomerNumber, new { @sagevalue = "Data.ToCustomerNumber" }, new { @class = "default txt-upper", @id = "Data\_ToCustomerNumber" })  @Html.KoSageButton("btnToCustomerFinder", null, new { @id = "btnToCustomerFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  </div>  <div class="form-group">  <div class="@className">  @Html.SageLabel("ClearStatistic\_ThroughYear", ClearStatisticsResx.ThroughYear)  @Html.KoSageTextBoxFor(model => model.Data.ThroughCustomerYear, new { @sagevalue = "Data.ThroughCustomerYear", @disable = "Data.isCustomerYearBool" }, new { @class = "default", @id = "Data\_ThroughCustomerYear", @formatTextbox = "numeric" })  @if (!calendarYear)  {  @Html.KoSageButton("btnFindCustomerYear", null, new { @id = "btnFindCustomerYear", @class = "icon btn-search", @tabindex = "-1" })  }  </div>  <div class="numeric-group">  @Html.SageLabel("ClearStatistic\_Period", ARCommonResx.Period)  @Html.KoSageTextBoxFor(model => model.Data.ThroughCustomerPeriod, new { @sagevalue = "Data.ThroughCustomerPeriod", @valueUpdate = "'input'" }, new { @id = "Data\_ThroughCustomerPeriod", @class = "default", @maxlength = "2", @formatTextbox = "numeric", @placeholder = 0, @prefixZero = "true" })  </div>  </div>  </div>  </div>  <div class="sub-section">  <div class="section-heading">  <div class="form-group">  <div class="ctrl-group">  <div>  @Html.KoSageCheckBox("Data\_ClearGroupStatistics", false, new { @sagechecked = "Data.bClearGroupStatistics" }, new { })  @Html.SageLabelFor(model => model.Data.ClearGroupStatistics, new { @class = "bold" })  </div>  </div>  </div>  </div>  <div class="section-body indented" data-bind="visible:Data.bClearGroupStatistics">  <div class="form-group">  <div class="go-group ">  @Html.SageLabel("lblClearStatistics", CommonResx.From)  @Html.KoSageTextBoxFor(model => model.Data.FromGroupCode, new { @sagevalue = "Data.FromGroupCode" }, new { @class = "default txt-upper", @id = "Data\_FromGroupCode", @formatTextbox = "alphaNumeric", @autofocus = "autofocus" })  @Html.KoSageButton("btnFromCustomerGroupFinder", null, new { @id = "btnFromCustomerGroupFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  <div class="go-group ">  @Html.SageLabel("ClearStatistic\_To", CommonResx.To)  @Html.KoSageTextBoxFor(model => model.Data.ToGroupCode, new { @sagevalue = "Data.ToGroupCode" }, new { @class = "default txt-upper", @id = "Data\_ToGroupCode", @formatTextbox = "alphaNumeric" })  @Html.KoSageButton("btnToCustomerGroupFinder", null, new { @id = "btnToCustomerGroupFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  </div>  <div class="form-group">  <div class="@className">  @Html.SageLabel("ClearStatistic\_ThroughYear2", ClearStatisticsResx.ThroughYear)  @Html.KoSageTextBoxFor(model => model.Data.ThroughGroupYear, new { @sagevalue = "Data.ThroughGroupYear", @disable = "Data.isCustomerGroupYearBool" }, new { @class = "default", @id = "Data\_ThroughGroupYear", @formatTextbox = "numeric" })  @if (!calendarYear)  {  @Html.KoSageButton("btnFindCustomerGroupYear", null, new { @id = "btnFindCustomerGroupYear", @class = "icon btn-search", @tabindex = "-1" })  }  </div>  <div class="numeric-group">  @Html.SageLabel("ClearStatistic\_Period2", ARCommonResx.Period)  @Html.KoSageTextBoxFor(model => model.Data.ThroughGroupPeriod, new { @sagevalue = "Data.ThroughGroupPeriod", @valueUpdate = "'input'" }, new { @id = "Data\_ThroughGroupPeriod", @class = "default", @maxlength = "2", @formatTextbox = "numeric", @placeholder = 0, @prefixZero = "true" })  </div>  </div>  </div>  </div>  <div class="sub-section">  <div class="section-heading">  <div class="form-group">  <div class="ctrl-group">  <div>  @Html.KoSageCheckBox("Data\_ClearNationalAccountStatistics", false, new { @sagechecked = "Data.bClearNationalAccountStatistics" }, new { })  @Html.SageLabelFor(model => model.Data.ClearNationalAccountStatistics, new { @class = "bold" })  </div>  </div>  </div>  </div>  <div class="section-body indented" data-bind="visible:Data.bClearNationalAccountStatistics">  <div class="form-group">  <div class="go-group ">  @Html.SageLabel("lblClearStatistics", CommonResx.From)  @Html.KoSageTextBoxFor(model => model.Data.FromNationalAccount, new { @sagevalue = "Data.FromNationalAccount" }, new { @class = "default txt-upper", @id = "Data\_FromNationalAccount", @autofocus = "autofocus" })  @Html.KoSageButton("btnFromNationalAcctFinder", null, new { @id = "btnFromNationalAcctFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  <div class="go-group ">  @Html.SageLabel("ClearStatistic\_To2", CommonResx.To)  @Html.KoSageTextBoxFor(model => model.Data.ToNationalAccount, new { @sagevalue = "Data.ToNationalAccount" }, new { @class = "default txt-upper", @id = "Data\_ToNationalAccount" })  @Html.KoSageButton("btnToNationalAcctFinder", null, new { @id = "btnToNationalAcctFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  </div>  <div class="form-group">  <div class="@className">  @Html.SageLabel("ClearStatistic\_ThroughYear3", ClearStatisticsResx.ThroughYear)  @Html.KoSageTextBoxFor(model => model.Data.ThroughNationalAccountYear, new { @sagevalue = "Data.ThroughNationalAccountYear", @disable = "Data.isNationalYearBool" }, new { @class = "default", @id = "Data\_ThroughNationalAccountYear", @formatTextbox = "numeric" })  @if (!calendarYear)  {  @Html.KoSageButton("btnFindNationalAcctYear", null, new { @id = "btnFindNationalAcctYear", @class = "icon btn-search", @tabindex = "-1" })  }  </div>  <div class="numeric-group">  @Html.SageLabel("ClearStatistic\_Period3", ARCommonResx.Period)  @Html.KoSageTextBoxFor(model => model.Data.ThroughNationalAccountPeriod, new { @sagevalue = "Data.ThroughNationalAccountPeriod", @valueUpdate = "'input'" }, new { @id = "Data\_ThroughNationalAccountPeriod", @class = "default", @maxlength = "2", @formatTextbox = "numeric", @placeholder = 0, @prefixZero = "true" })  </div>  </div>  </div>  </div>  <div class="sub-section">  <div class="section-heading">  <div class="form-group">  <div class="ctrl-group">  <div>  @Html.KoSageCheckBox("Data\_ClearSalesPersonStatistics", false, new { @sagechecked = "Data.bClearSalespersonStatistics" }, new { })  @Html.SageLabelFor(model => model.Data.ClearSalesPersonStatistics, new { @class = "bold" })  </div>  </div>  </div>  </div>  <div class="section-body indented" data-bind="visible:Data.bClearSalespersonStatistics">  <div class="form-group">  <div class="go-group ">  Fsessionuti  @Html.SageLabel("lblClearStatistics", CommonResx.From)  @Html.KoSageTextBoxFor(model => model.Data.FromSalesperson, new { @sagevalue = "Data.FromSalesperson" }, new { @class = "default txt-upper", @id = "Data\_FromSalesperson", @formatTextbox = "alphaNumeric" })  @Html.KoSageButton("btnFromSalespersonFinder", null, new { @id = "btnFromSalespersonFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  <div class="go-group ">  @Html.SageLabel("ClearStatistic\_To3", CommonResx.To)  @Html.KoSageTextBoxFor(model => model.Data.ToSalesperson, new { @sagevalue = "Data.ToSalesperson" }, new { @class = "default txt-upper", @id = "Data\_ToSalesperson", @formatTextbox = "alphaNumeric" })  @Html.KoSageButton("btnToSalespersonFinder", null, new { @id = "btnToSalespersonFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  </div>  <div class="form-group">  <div class="@classSalesName">  @Html.SageLabel("ClearStatistic\_ThroughYear4", ClearStatisticsResx.ThroughYear)  @Html.KoSageTextBoxFor(model => model.Data.ThroughSalespersonYear, new { @sagevalue = "Data.ThroughSalespersonYear", @disable = "Data.isSalespersonYearBool" }, new { @class = "default", @id = "Data\_ThroughSalespersonYear", @formatTextbox = "numeric" })  @if (!salespersonCalendarYear)  {  @Html.KoSageButton("btnFindSalespersonYear", null, new { @id = "btnFindSalespersonYear", @class = "icon btn-search", @tabindex = "-1" })  }  </div>  <div class="numeric-group">  @Html.SageLabel("ClearStatistic\_Period4", ARCommonResx.Period)  @Html.KoSageTextBoxFor(model => model.Data.ThroughSalespersonPeriod, new { @sagevalue = "Data.ThroughSalespersonPeriod", @valueUpdate = "'input'" }, new { @id = "Data\_ThroughSalespersonPeriod", @class = "default", @maxlength = "2", @formatTextbox = "numeric", @placeholder = 0, @prefixZero = "true" })  </div>  </div>  </div>  </div>  <div class="sub-section">  <div class="section-heading">  <div class="form-group">  <div class="ctrl-group">  <div>  @Html.KoSageCheckBox("Data\_ClearItemStatistics", false, new { @sagechecked = "Data.bClearItemStatistics" }, new { })  @Html.SageLabelFor(model => model.Data.ClearItemStatistics, new { @class = "bold" })  </div>  </div>  </div>  </div>  <div class="section-body indented" data-bind="visible:Data.bClearItemStatistics">  <div class="form-group">  <div class="go-group ">  @Html.SageLabel("lblClearStatistics", CommonResx.From)  @Html.KoSageTextBoxFor(model => model.Data.FromItemNumber, new { @sagevalue = "Data.FromItemNumber" }, new { @class = "default txt-upper", @id = "Data\_FromItem", @autofocus = "autofocus" })  @Html.KoSageButton("btnFromItemFinder", null, new { @id = "btnFromItemFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  <div class="go-group ">  @Html.SageLabel("ClearStatistic\_To4", CommonResx.To)  @Html.KoSageTextBoxFor(model => model.Data.ToItemNumber, new { @sagevalue = "Data.ToItemNumber" }, new { @class = "default txt-upper", @id = "Data\_ToItem" })  @Html.KoSageButton("btnToItemFinder", null, new { @id = "btnToItemFinder", @class = "icon btn-finder", @tabindex = "-1" })  </div>  </div>  <div class="form-group">  <div class="@classItemName">  @Html.SageLabel("ClearStatistic\_ThroughYear5", ClearStatisticsResx.ThroughYear)  @Html.KoSageTextBoxFor(model => model.Data.ThroughItemYear, new { @sagevalue = "Data.ThroughItemYear", @disable = "Data.isItemYearBool" }, new { @class = "default", @id = "Data\_ThroughItemYear", @formatTextbox = "numeric" })  @if (!itemCalendarYear)  {  @Html.KoSageButton("btnFindItemYear", null, new { @id = "btnFindItemYear", @class = "icon btn-search", @tabindex = "-1" })  }  </div>  <div class="numeric-group">  @Html.SageLabel("ClearStatistic\_Period5", ARCommonResx.Period)  @Html.KoSageTextBoxFor(model => model.Data.ThroughItemPeriod, new { @sagevalue = "Data.ThroughItemPeriod", @valueUpdate = "'input'" }, new { @id = "Data\_ThroughItemPeriod", @class = "default", @maxlength = "2", @formatTextbox = "numeric", @placeholder = 0, @prefixZero = "true" })  </div>  </div>  </div>  </div>  <section class="footer-group-1">  @Html.KoSageButton("btnProcess", null, new { @id = "btnProcess", @class = "btn btn-primary", @value = CommonResx.Process })  </section>  </div> |

* 1. Clear Statistics Repository JavaScript

Access the **…ClearStatisticsRepository.js** file and add code required to complete the screen

|  |
| --- |
| /\* Copyright (c) 2020 Valued Partner All rights reserved. \*/  "use strict";  var clearStatisticsRepository = clearStatisticsRepository || {};  clearStatisticsRepository = {  getCustomerMaxPeriodForValidYear: function (year) {  var data = {  'year': year  };  sg.utls.ajaxPostHtml(sg.utls.url.buildUrl("TU", "ClearStatistics", "GetMaxPeriodForValidYear"), data, clearStatisticsUISuccess.fillCustomerFiscalYear);  },  getGroupMaxPeriodForValidYear: function (year) {  var data = {  'year': year  };  sg.utls.ajaxPostHtml(sg.utls.url.buildUrl("TU", "ClearStatistics", "GetMaxPeriodForValidYear"), data, clearStatisticsUISuccess.fillGroupFiscalYear);  },  getNationalAcctMaxPeriodForValidYear: function (year) {  var data = {  'year': year  };  sg.utls.ajaxPostHtml(sg.utls.url.buildUrl("TU", "ClearStatistics", "GetMaxPeriodForValidYear"), data, clearStatisticsUISuccess.fillNationalAcctFiscalYear);  },  getSalespersonMaxPeriodForValidYear: function (year) {  var data = {  'year': year  };  sg.utls.ajaxPostHtml(sg.utls.url.buildUrl("TU", "ClearStatistics", "GetSalesPersonMaxPeriodForValidYear"), data, clearStatisticsUISuccess.fillSalespersonFiscalYear);  },  getItemMaxPeriodForValidYear: function (year) {  var data = {  'year': year  };  sg.utls.ajaxPostHtml(sg.utls.url.buildUrl("TU", "ClearStatistics", "GetItemMaxPeriodForValidYear"), data, clearStatisticsUISuccess.fillItemFiscalYear);  }  }; |

* 1. Clear Statistics Knockout Binding JavaScript

Access the …**ClearStatisticsKoExtn.js** file and add the new computed properties for **Status**, **InactiveDate**, and **DateLastMaintained** (the code to be added is highlighted in yellow).

|  |
| --- |
| /\* Copyright (c) 2020 Valued Partner All rights reserved. \*/  "use strict";  var tuClearStatisticsKoExtn = tuClearStatisticsKoExtn || {};  tuClearStatisticsKoExtn = {  tuClearStatisticsModelExtension: function (modelData) {  var model = modelData.Data;  model.Year = ko.observable("");  model.IsKoStatisticsDirty = new ko.dirtyFlag(clearStatisticsUI.clearStatisticsModel);  model.bClearCustomerStatistics = ko.computed({  read: function () {  //return original value  return tuClearStatisticsKoExtn.convertToBoolean(clearStatisticsUI.clearStatisticsModel.Data.ClearCustomerStatistics());  },  write: function (newValue) {  //convert boolean to appropriate enum and assign  var parsedValue = newValue ? 1 : 0;  clearStatisticsUI.clearStatisticsModel.Data.ClearCustomerStatistics(parsedValue);  },  owner: this  });  model.bClearGroupStatistics = ko.computed({  read: function () {  //return original value  return tuClearStatisticsKoExtn.convertToBoolean(clearStatisticsUI.clearStatisticsModel.Data.ClearGroupStatistics());  },  write: function (newValue) {  //convert boolean to appropriate enum and assign  var parsedValue = newValue ? 1 : 0;  clearStatisticsUI.clearStatisticsModel.Data.ClearGroupStatistics(parsedValue);  },  owner: this  });  model.bClearNationalAcctStatistics = ko.computed({  read: function () {  //return original value  return tuClearStatisticsKoExtn.convertToBoolean(clearStatisticsUI.clearStatisticsModel.Data.ClearNationalAcctStatistics());  },  write: function (newValue) {  //convert boolean to appropriate enum and assign  var parsedValue = newValue ? 1 : 0;  clearStatisticsUI.clearStatisticsModel.Data.ClearNationalAcctStatistics(parsedValue);  },  owner: this  });  model.bClearSalespersonStatistics = ko.computed({  read: function () {  //return original value  return tuClearStatisticsKoExtn.convertToBoolean(clearStatisticsUI.clearStatisticsModel.Data.ClearSalesPersonStatistics());  },  write: function (newValue) {  //convert boolean to appropriate enum and assign  var parsedValue = newValue ? 1 : 0;  clearStatisticsUI.clearStatisticsModel.Data.ClearSalesPersonStatistics(parsedValue);  },  owner: this  });  model.bClearItemStatistics = ko.computed({  read: function () {  //return original value  return tuClearStatisticsKoExtn.convertToBoolean(clearStatisticsUI.clearStatisticsModel.Data.ClearItemStatistics());  },  write: function (newValue) {  //convert boolean to appropriate enum and assign  var parsedValue = newValue ? 1 : 0;  clearStatisticsUI.clearStatisticsModel.Data.ClearItemStatistics(parsedValue);  },  owner: this  });  },  /\* convert value to bool (True/False) \*/  convertToBoolean: function (value) {  switch (value.toString().toLowerCase()) {  case "true": case "yes": case "1": return true;  case "false": case "no": case "0": case null: return false;  default: return Boolean(value);  }  }  }; |

* 1. Clear Statistics Behaviour JavaScript

Access the **…ClearStatisticsBehaviour.js** file and add the business logic for the new screen controls

|  |
| --- |
| /\* Copyright (c) 2020 Valued Partner All rights reserved. \*/  "use strict";  var clearStatisticsUI = clearStatisticsUI || {}  var minimumFiscalYear = 1900;  clearStatisticsUI = {  clearStatisticsModel: {},  computedProperties: ["bClearCustomerStatistics", "bClearGroupStatistics", "bClearNationalAcctStatistics", "bClearSalespersonStatistics"],  customerFiscalYear: null,  customerGroupFiscalYear: null,  nationalAcctFiscalYear: null,  salespersonFiscalYear: null,  itemFiscalYear: null,  customerFiscalPeriod: null,  customerGroupFiscalPeriod: null,  nationalAcctFiscalPeriod: null,  salespersonFiscalPeriod: null,  itemFiscalPeriod: null,  /\*\*  \* Initialization  \*  \* @method init  \*/  init: function () {  // Initialize the controls and apply kendo bindings  clearStatisticsUI.initKendoBindings();  clearStatisticsUI.initFinder();  clearStatisticsUI.initButtons();  clearStatisticsUI.initTextBox();  clearStatisticsUI.initCheckBox();  clearStatisticsUI.initBlur();  clearStatisticsUI.initProcessUI();  ko.applyBindings(clearStatisticsUI.clearStatisticsModel);  },  /\*\*  \* Initialize the Kendo bindings  \*  \* @method initKendoBindings  \*/  initKendoBindings: function () {  clearStatisticsUI.clearStatisticsModel = ko.mapping.fromJS(ClearStatisticsViewModel);  clearStatisticsUI.customerFiscalYear = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear();  clearStatisticsUI.customerGroupFiscalYear = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear();  clearStatisticsUI.nationalAcctFiscalYear = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear();  clearStatisticsUI.salespersonFiscalYear = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear();  clearStatisticsUI.itemFiscalYear = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear();  clearStatisticsUtilities.customerYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear();  clearStatisticsUtilities.customerGroupYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear();  clearStatisticsUtilities.nationalAcctYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear();  clearStatisticsUtilities.salespersonYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear();  clearStatisticsUtilities.itemYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear();  clearStatisticsUI.customerFiscalPeriod = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerPeriod();  clearStatisticsUI.customerGroupFiscalPeriod = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupPeriod();  clearStatisticsUI.nationalAcctFiscalPeriod = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctPeriod();  clearStatisticsUI.salespersonFiscalPeriod = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonPeriod();  clearStatisticsUI.itemFiscalPeriod = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemPeriod();  clearStatisticsUtilities.customerPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerPeriod();  clearStatisticsUtilities.customerGroupPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupPeriod();  clearStatisticsUtilities.nationalAcctPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctPeriod();  clearStatisticsUtilities.salespersonPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonPeriod();  clearStatisticsUtilities.itemPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemPeriod();  tuClearStatisticsKoExtn.tuClearStatisticsModelExtension(clearStatisticsUI.clearStatisticsModel);  },  /\*\*  \* Initialize the Finder  \*  \* @method initFinder  \*/  initFinder: function () {  // Initializing Customer Statistics Finder  clearStatisticsUI.initCustomerNumberFinder();  // Initializing Customer Group Statistics Finder  clearStatisticsUI.initCustomerGroupFinder();  // Initializing National Account Statistics Finder  clearStatisticsUI.initNationalAcctFinder();  // Initializing Salesperson Statistics Finder  clearStatisticsUI.initSalespersonFinder();  // Initializing Item Statistics Finder  clearStatisticsUI.initItemFinder();  // Initializing Fiscal Year Finder  clearStatisticsUI.initFiscalYearFinder();  },  initCustomerNumberFinder: function () {  var title = jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.CustomerNumberTitle);  //From Customer  sg.finderHelper.setFinder("btnFromCustomerFinder", sg.finder.ARCustomerFinder, onFinderSuccess.FromCustomerFinder, onFinderCancel.FromCustomerFinder, title,  sg.finderHelper.createDefaultFunction("Data\_FromCustomerNumber", "CustomerNumber", sg.finderOperator.StartsWith));  //To Customer  sg.finderHelper.setFinder("btnToCustomerFinder", sg.finder.ARCustomerFinder, onFinderSuccess.ToCustomerFinder, onFinderCancel.ToCustomerFinder, title,  sg.finderHelper.createDefaultFunction("Data\_ToCustomerNumber", "CustomerNumber", sg.finderOperator.StartsWith));  },  initCustomerGroupFinder: function () {  var title = jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.CustomerGroupFinder);  //From Customer Group  sg.finderHelper.setFinder("btnFromCustomerGroupFinder", sg.finder.CustomerGroup, onFinderSuccess.FromCustomerGroupFinder, onFinderCancel.FromCustomerGroupFinder, title,  sg.finderHelper.createDefaultFunction("Data\_FromGroupCode", "GroupCode", sg.finderOperator.StartsWith));  //To Customer  sg.finderHelper.setFinder("btnToCustomerGroupFinder", sg.finder.CustomerGroup, onFinderSuccess.ToCustomerGroupFinder, onFinderCancel.ToCustomerGroupFinder, title,  sg.finderHelper.createDefaultFunction("Data\_ToGroupCode", "GroupCode", sg.finderOperator.StartsWith));  },  initNationalAcctFinder: function () {  var title = jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.NationalAccountNumberTitle);  //From National Account  sg.finderHelper.setFinder("btnFromNationalAcctFinder", sg.finder.NationalAccounts, onFinderSuccess.FromNationalAccountFinder, onFinderCancel.FromNationalAccountFinder, title,  sg.finderHelper.createDefaultFunction("Data\_FromNationalAccount", "NationalAccountNumber", sg.finderOperator.StartsWith));  //To National Account  sg.finderHelper.setFinder("btnToNationalAcctFinder", sg.finder.NationalAccounts, onFinderSuccess.ToNationalAccountFinder, onFinderCancel.ToNationalAccountFinder, title,  sg.finderHelper.createDefaultFunction("Data\_ToNationalAccount", "NationalAccountNumber", sg.finderOperator.StartsWith));  },  initSalespersonFinder: function () {  var title = jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.SalesPersonFinderTitle);  //From Salesperson  sg.finderHelper.setFinder("btnFromSalespersonFinder", sg.finder.SalesPersonFinder, onFinderSuccess.FromSalespersonFinder, onFinderCancel.FromSalespersonFinder, title,  sg.finderHelper.createDefaultFunction("Data\_FromSalesPerson", "SalesPersonCode", sg.finderOperator.StartsWith));  //To Salesperson  sg.finderHelper.setFinder("btnToSalespersonFinder", sg.finder.SalesPersonFinder, onFinderSuccess.ToSalespersonFinder, onFinderCancel.ToSalespersonFinder, title,  sg.finderHelper.createDefaultFunction("Data\_ToSalesperson", "SalesPersonCode", sg.finderOperator.StartsWith));  },  initFiscalYearFinder: function () {  var title = window.jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.FiscalYear);  sg.finderHelper.setFinder("btnFindCustomerYear", sg.finder.FiscalYear, onFinderSuccess.CustomerYearFinder, onFinderCancel.CustomerYearFinder, title, sg.finderHelper.createDefaultFunction("Data\_ThroughCustomerYear", "FiscalYear", sg.finderOperator.StartsWith), null, true);  sg.finderHelper.setFinder("btnFindCustomerGroupYear", sg.finder.FiscalYear, onFinderSuccess.CustomerGroupYearFinder, onFinderCancel.CustomerGroupYearFinder, title, sg.finderHelper.createDefaultFunction("Data\_ThroughGroupYear", "FiscalYear", sg.finderOperator.StartsWith), null, true);  sg.finderHelper.setFinder("btnFindNationalAcctYear", sg.finder.FiscalYear, onFinderSuccess.NationalAcctYearFinder, onFinderCancel.NationalAcctYearFinder, title, sg.finderHelper.createDefaultFunction("Data\_ThroughNationalAcctYear", "FiscalYear", sg.finderOperator.StartsWith), null, true);  sg.finderHelper.setFinder("btnFindSalespersonYear", sg.finder.FiscalYear, onFinderSuccess.SalespersonYearFinder, onFinderCancel.SalespersonYearFinder, title, sg.finderHelper.createDefaultFunction("Data\_ThroughSalesPersonYear", "FiscalYear", sg.finderOperator.StartsWith), null, true);  sg.finderHelper.setFinder("btnFindItemYear", sg.finder.FiscalYear, onFinderSuccess.ItemYearFinder, onFinderCancel.ItemYearFinder, title, sg.finderHelper.createDefaultFunction("Data\_ThroughItemYear", "FiscalYear", sg.finderOperator.StartsWith), null, true);  },  initItemFinder: function () {  var title = jQuery.validator.format(clearStatisticsResources.FinderTitle, clearStatisticsResources.ItemFinderTitle);  //From Item  sg.finderHelper.setFinder("btnFromItemFinder", sg.finder.ARItemNumberFinder, onFinderSuccess.FromItemFinder, onFinderCancel.FromItemFinder, title,  sg.finderHelper.createDefaultFunction("Data\_FromItem", "ItemNumber", sg.finderOperator.StartsWith));  //To Item  sg.finderHelper.setFinder("btnToItemFinder", sg.finder.ARItemNumberFinder, onFinderSuccess.ToItemFinder, onFinderCancel.ToItemFinder, title,  sg.finderHelper.createDefaultFunction("Data\_ToItem", "ItemNumber", sg.finderOperator.StartsWith));  },  /\*\*  \* Initialize the Check Boxes  \*  \* @method initCheckBox  \*/  initCheckBox: function () {  $("#Data\_ClearCustomerStatistics").click(function (e) {  if ($(this).is(':checked')) {  setTimeout(function () {  sg.controls.Focus($("#Data\_FromCustomerNumber"));  });  }  });  $(document).on("change", "#Data\_ClearGroupStatistics", function () {  if ($("#Data\_ClearGroupStatistics").is(":checked")) {  setTimeout(function () {  sg.controls.Focus($("#Data\_FromGroupCode"));  });  }  });  $(document).on("change", "#Data\_ClearNationalAcctStatistics", function () {  if ($("#Data\_ClearNationalAcctStatistics").is(":checked")) {  setTimeout(function () {  sg.controls.Focus($("#Data\_FromNationalAccount"));  });  }  });  $(document).on("change", "#Data\_ClearSalesPersonStatistics", function () {  if ($("#Data\_ClearSalesPersonStatistics").is(":checked")) {  setTimeout(function () {  sg.controls.Focus($("#Data\_FromSalesPerson"));  });  }  });  $(document).on("change", "#Data\_ClearItemStatistics", function () {  if ($("#Data\_ClearItemStatistics").is(":checked")) {  setTimeout(function () {  sg.controls.Focus($("#Data\_FromItem"));  });  }  });  },  /\*\*  \* Initialize the Blur binding  \*  \* @method initBlur  \*/  initBlur: function () {  $("#Data\_ThroughCustomerYear").bind('change', function (e) {  sg.delayOnChange("btnFindCustomerYear", $("Data\_ThroughCustomerYear"), function () {  var year = $("#Data\_ThroughCustomerYear").val();  var oldYear = clearStatisticsUI.customerFiscalYear;  if (clearStatisticsUI.clearStatisticsModel.CalendarYear()) {  if (year < minimumFiscalYear) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear(oldYear);  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear(clearStatisticsUtilities.customerYearBackup);  //clearStatisticsUtilities.revertYearValue();  $('#Data\_ThroughCustomerYear').focus();  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshCustomerStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  } else {  if (!(clearStatisticsUI.fiscalYrExists($("#Data\_ThroughCustomerYear").val()))) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear(clearStatisticsUtilities.customerYearBackup);  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.validateCustomerYear, clearStatisticsUtility.setFocusToFiscalYear);  }  }  });  });  $("#Data\_ThroughGroupYear").bind('change', function (e) {  sg.delayOnChange("btnFindCustomerGroupYear", $("Data\_ThroughGroupYear"), function () {  var year = $("#Data\_ThroughGroupYear").val();  var oldYear = clearStatisticsUI.customerGroupFiscalYear;  if (clearStatisticsUI.clearStatisticsModel.CalendarYear()) {  if (year < minimumFiscalYear) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear(oldYear);  //clearStatisticsUtilities.revertYearValue();  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear(clearStatisticsUtilities.customerGroupYearBackup);  $('#Data\_ThroughGroupYear').focus();  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshGroupStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  }  else {  if (!(clearStatisticsUI.fiscalYrExists($("#Data\_ThroughGroupYear").val()))) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear(clearStatisticsUtilities.customerGroupYearBackup);  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.validateGroupYear, clearStatisticsUtility.setFocusToFiscalYear);  }  }  });  });  $("#Data\_ThroughNationalAcctYear").bind('change', function (e) {  sg.delayOnChange("btnFindNationalAcctYear", $("Data\_ThroughNationalAcctYear"), function () {  var year = $("#Data\_ThroughNationalAcctYear").val();  var oldYear = clearStatisticsUI.nationalAcctFiscalYear;  if (clearStatisticsUI.clearStatisticsModel.CalendarYear()) {  if (year < minimumFiscalYear) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear(oldYear);  //clearStatisticsUtilities.revertYearValue();  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear(clearStatisticsUtilities.nationalAcctYearBackup);  $('#Data\_ThroughNationalAcctYear').focus();  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshNationalAcctStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  }  else {  if (!(clearStatisticsUI.fiscalYrExists($("#Data\_ThroughNationalAcctYear").val()))) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear(clearStatisticsUtilities.nationalAcctYearBackup);  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.validateNationalAcctYear, clearStatisticsUtility.setFocusToFiscalYear);  }  }  });  });  $("#Data\_ThroughSalesPersonYear").bind('change', function (e) {  sg.delayOnChange("btnFindSalespersonYear", $("Data\_ThroughSalesPersonYear"), function () {  var year = $("#Data\_ThroughSalesPersonYear").val();  var oldYear = clearStatisticsUI.salespersonFiscalYear;  if (clearStatisticsUI.clearStatisticsModel.SalesCalendarYear()) {  if (year < minimumFiscalYear) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear(oldYear);  //clearStatisticsUtilities.revertYearValue();  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear(clearStatisticsUtilities.salespersonYearBackup);  $('#Data\_ThroughSalesPersonYear').focus();  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshSalespersonStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  }  else {  if (!(clearStatisticsUI.fiscalYrExists($("#Data\_ThroughSalesPersonYear").val()))) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear(clearStatisticsUtilities.salespersonYearBackup);  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.validateSalespersonYear, clearStatisticsUtility.setFocusToFiscalYear);  }  }  });  });  $("#Data\_ThroughItemYear").bind('change', function (e) {  sg.delayOnChange("btnFindItemYear", $("Data\_ThroughItemYear"), function () {  var year = $("#Data\_ThroughItemYear").val();  var oldYear = clearStatisticsUI.itemFiscalYear;  if (clearStatisticsUI.clearStatisticsModel.ItemCalendarYear()) {  if (year < minimumFiscalYear) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear(oldYear);  //clearStatisticsUtilities.revertYearValue();  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear(clearStatisticsUtilities.itemYearBackup);  $('#Data\_ThroughItemYear').focus();  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshItemStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  }  else {  if (!(clearStatisticsUI.fiscalYrExists($("#Data\_ThroughItemYear").val()))) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear(clearStatisticsUtilities.itemYearBackup);  } else {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.validateItemYear, clearStatisticsUtility.setFocusToFiscalYear);  }  }  });  });  $("#Data\_ThroughCustomerPeriod").bind('change', function (e) {  $("#message").empty();  clearStatisticsUtilities.validateCustomerPeriod();  });  $("#Data\_ThroughGroupPeriod").bind('change', function (e) {  $("#message").empty();  clearStatisticsUtilities.validateGroupPeriod();  });  $("#Data\_ThroughNationalAcctPeriod").bind('change', function (e) {  $("#message").empty();  clearStatisticsUtilities.validateNationalAcctPeriod();  });  $("#Data\_ThroughSalesPersonPeriod").bind('change', function (e) {  $("#message").empty();  // clearStatisticsUtilities.backupSalespersonPeriodValue();  clearStatisticsUtilities.validateSalespersonPeriod();  });  $("#Data\_ThroughItemPeriod").bind('change', function (e) {  $("#message").empty();  clearStatisticsUtilities.validateItemPeriod();  });  },  /\*\*  \* Initialize Textbox bindings  \*  \* @method initTextBox  \*/  initTextBox: function () {  $("#Data\_ThroughCustomerPeriod").kendoNumericTextBox({  format: "00",  spinners: true,  min: clearStatisticsUI.clearStatisticsModel.MinimumPeriod(),  max: clearStatisticsUI.clearStatisticsModel.MaximumPeriod(),  step: "1",  type: "number",  spin: function () {  var value = this.value();  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerPeriod(value);  clearStatisticsUtilities.backupCustomerPeriodValue();  }  });  $("#Data\_ThroughGroupPeriod").kendoNumericTextBox({  format: "00",  spinners: true,  min: clearStatisticsUI.clearStatisticsModel.MinimumPeriod(),  max: clearStatisticsUI.clearStatisticsModel.MaximumPeriod(),  step: "1",  type: "number",  spin: function () {  var value = this.value();  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupPeriod(value);  clearStatisticsUtilities.backupGroupPeriodValue();  }  });  $("#Data\_ThroughNationalAcctPeriod").kendoNumericTextBox({  format: "00",  spinners: true,  min: clearStatisticsUI.clearStatisticsModel.MinimumPeriod(),  max: clearStatisticsUI.clearStatisticsModel.MaximumPeriod(),  step: "1",  type: "number",  spin: function () {  var value = this.value();  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctPeriod(value);  clearStatisticsUtilities.backupNationalAcctPeriodValue();  }  });  $("#Data\_ThroughSalesPersonPeriod").kendoNumericTextBox({  format: "00",  spinners: true,  min: clearStatisticsUI.clearStatisticsModel.SalesPersonMinimumPeriod(),  max: clearStatisticsUI.clearStatisticsModel.SalesPersonMaximumPeriod(),  step: "1",  type: "number",  spin: function () {  var value = this.value();  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonPeriod(value);  clearStatisticsUtilities.backupSalespersonPeriodValue();  }  });  $("#Data\_ThroughItemPeriod").kendoNumericTextBox({  format: "00",  spinners: true,  min: clearStatisticsUI.clearStatisticsModel.ItemMinimumPeriod(),  max: clearStatisticsUI.clearStatisticsModel.ItemMaximumPeriod(),  step: "1",  type: "number",  spin: function () {  var value = this.value();  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemPeriod(value);  clearStatisticsUtilities.backupItemPeriodValue();  }  });  },  /\*\*  \* Initialize the Buttons  \*  \* @method initButtons  \*/  initButtons: function () {  $("#btnProcess").click(function (e) {  debugger;  sg.utls.SyncExecute(clearStatisticsUI.process);  });  },  process: function () {  // debugger;  sg.utls.isProcessRunning = true;  var processUrl = sg.utls.url.buildUrl("TU", "ClearStatistics", "Process");  var isChecked = clearStatisticsUI.clearStatisticsModel.Data.bClearCustomerStatistics()  || clearStatisticsUI.clearStatisticsModel.Data.bClearGroupStatistics()  || clearStatisticsUI.clearStatisticsModel.Data.bClearNationalAcctStatistics()  || clearStatisticsUI.clearStatisticsModel.Data.bClearSalespersonStatistics()  || clearStatisticsUI.clearStatisticsModel.Data.bClearItemStatistics();  if (!isChecked) {  // Do not process  sg.utls.showMessageInfo(sg.utls.msgType.ERROR, clearStatisticsResources.NoProcessingOption);  sg.utls.isProcessRunning = false;  }  // Check if form is valid  if ($("#frmClearStatistics").valid() && sg.utls.isProcessRunning) {  // Check Validations  if (clearStatisticsUI.Validation()) {  $("#message").empty();  sg.utls.clearValidations("frmClearStatistics");  sg.utls.isProcessRunning = true;  var data = { model: ko.mapping.toJS(clearStatisticsUI.clearStatisticsModel, clearStatisticsUI.computedProperties) };  sg.utls.ajaxPost(processUrl, data, onSuccess.process);  }  }  },  /\*\*  \* Initialize the Process UI  \*  \* @method initProcessUI  \*/  initProcessUI: function () {  var progressUrl = sg.utls.url.buildUrl("TU", "ClearStatistics", "Progress");  var cancelUrl = sg.utls.url.buildUrl("TU", "ClearStatistics", "Cancel");  window.progressUI.init(progressUrl, cancelUrl, clearStatisticsUI.clearStatisticsModel, screenName, onSuccess.onProcessComplete);  },  /\*\*  \* Validation  \*  \* @method Validation  \*/  Validation: function () {  var errorRangeMessage = "";  var inputValid = true;  // If FromCustomer is greater than ToCustomer, throw an exception  if (clearStatisticsUI.clearStatisticsModel.Data.ClearCustomerStatistics() && (clearStatisticsUI.clearStatisticsModel.Data.FromCustomerNumber() != null &&  clearStatisticsUI.clearStatisticsModel.Data.FromCustomerNumber().localeCompare(clearStatisticsUI.clearStatisticsModel.Data.ToCustomerNumber())) > 0) {  inputValid = false;  errorRangeMessage = clearStatisticsResources.CustomerNumberTitle;  sg.controls.Focus($("#Data\_FromCustomerNumber"));  }  // If FromCustomerGroup is greater than ToCustomerGroup, throw an exception  else if (clearStatisticsUI.clearStatisticsModel.Data.ClearGroupStatistics() && (clearStatisticsUI.clearStatisticsModel.Data.FromGroupCode() != null &&  clearStatisticsUI.clearStatisticsModel.Data.FromGroupCode().localeCompare(clearStatisticsUI.clearStatisticsModel.Data.ToGroupCode())) > 0) {  inputValid = false;  errorRangeMessage = clearStatisticsResources.CustomerGroupFinder;  sg.controls.Focus($("#Data\_FromGroupCode"));  }  // If FromNationalAccnt is greater than ToNationalAccnt, throw an exception  else if (clearStatisticsUI.clearStatisticsModel.Data.ClearNationalAcctStatistics() && clearStatisticsUI.clearStatisticsModel.Data.FromNationalAccount() != null &&  clearStatisticsUI.clearStatisticsModel.Data.FromNationalAccount().localeCompare(clearStatisticsUI.clearStatisticsModel.Data.ToNationalAccount()) > 0) {  inputValid = false;  errorRangeMessage = clearStatisticsResources.NationalAccountNumberTitle;  sg.controls.Focus($("#Data\_FromNationalAccount"));  }  // If FromSalesPerson is greater than ToSalesPerson, throw an exception  else if (clearStatisticsUI.clearStatisticsModel.Data.ClearSalesPersonStatistics() && clearStatisticsUI.clearStatisticsModel.Data.FromSalesPerson() != null &&  clearStatisticsUI.clearStatisticsModel.Data.FromSalesPerson().localeCompare(clearStatisticsUI.clearStatisticsModel.Data.ToSalesPerson()) > 0) {  inputValid = false;  errorRangeMessage = clearStatisticsResources.SalesPersonFinderTitle;  sg.controls.Focus($("#Data\_FromSalesPerson"));  }  // If FromItem is greater than ToItem, throw an exception  else if (clearStatisticsUI.clearStatisticsModel.Data.ClearItemStatistics() && clearStatisticsUI.clearStatisticsModel.Data.FromItemNumber() != null &&  clearStatisticsUI.clearStatisticsModel.Data.FromItemNumber().localeCompare(clearStatisticsUI.clearStatisticsModel.Data.ToItemNumber()) > 0) {  inputValid = false;  errorRangeMessage = clearStatisticsResources.ItemFinderTitle;  sg.controls.Focus($("#Data\_FromItem"));  }  if (!inputValid) {  if (errorRangeMessage != "") {  sg.utls.showMessageInfo(sg.utls.msgType.ERROR, jQuery.validator.format(clearStatisticsResources.ErrorFromToValueMessage, errorRangeMessage));  } else if (errorMessage != "") {  sg.utls.showMessageInfo(sg.utls.msgType.ERROR, errorMessage);  }  }  return inputValid;  },  fiscalYrExists: function (Year) {  if (clearStatisticsUI.clearStatisticsModel.FiscalCalendars() !== null) {  for (var i = 0; i < clearStatisticsUI.clearStatisticsModel.FiscalCalendars().length; i++) {  if (Year === clearStatisticsUI.clearStatisticsModel.FiscalCalendars()[i].Year()) {  return true;  }  }  }  return false;  },  };  var onSuccess = {  /\*\*  \* Process Success  \*  \* @method process  \*/  process: function (jsonResult) {  if (jsonResult.UserMessage.IsSuccess) {  var model = clearStatisticsUI.clearStatisticsModel;  window.ko.mapping.fromJS(jsonResult.WorkflowInstanceId, {}, model.WorkflowInstanceId);  window.progressUI.progress();  } else {  sg.utls.showMessage(jsonResult);  }  },  /\*\*  \* On Process Complete  \*  \* @method onProcessComplete  \*/  onProcessComplete: function (result) {  if (result.ProcessResult.Results.length <= 0) {  $("#processingResultGrid").hide();  var errorMessage = clearStatisticsResources.ProcessingComplete;  sg.utls.showMessageInfoInCustomDivWithoutClose(sg.utls.msgType.INFO, errorMessage, 'messageDiv');  }  },  };  // Finder success method  var onFinderSuccess = {  // variable for From Customer Number finder.  FromCustomerFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.FromCustomerNumber(data.CustomerNumber);  //setTimeout(function () {  // ($("#Data\_ToCustomerNumber")).siblings('input:visible').focus();  //});  sg.controls.Focus($("#Data\_ToCustomerNumber"));  }  },  // variable for To Customer Number finder.  ToCustomerFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.ToCustomerNumber(data.CustomerNumber);  sg.controls.Focus($("#Data\_ThroughCustomerYear"));  }  },  // variable for From Customer Group finder.  FromCustomerGroupFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.FromGroupCode(data.GroupCode);  //setTimeout(function () {  // ($("#Data\_ToGroupCode")).siblings('input:visible').focus();  //});  sg.controls.Focus($("#Data\_ToGroupCode"));  }  },  // variable for To Customer Group finder.  ToCustomerGroupFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.ToGroupCode(data.GroupCode);  sg.controls.Focus($("#Data\_ThroughGroupYear"));  }  },  // variable for From National Account finder.  FromNationalAccountFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.FromNationalAccount(data.NationalAccountNumber);  //setTimeout(function () {  // ($("#Data\_ToNationalAccount")).siblings('input:visible').focus();  //});  sg.controls.Focus($("#Data\_ToNationalAccount"));  }  },  // variable for To National Account finder.  ToNationalAccountFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.ToNationalAccount(data.NationalAccountNumber);  //setTimeout(function () {  // ($("#Data\_ToNationalAccount")).siblings('input:visible').focus();  //});  sg.controls.Focus($("#Data\_ThroughNationalAcctPeriod"));  }  },  // variable for From Salesperson finder.  FromSalespersonFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.FromSalesPerson(data.SalesPersonCode);  sg.controls.Focus($("#Data\_ToSalesPerson"));  }  },  // variable for To Salesperson finder.  ToSalespersonFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.ToSalesPerson(data.SalesPersonCode);  sg.controls.Focus($("#Data\_ThroughSalesPersonYear"));  }  },  // variable for From Item finder.  FromItemFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.FromItemNumber(data.ItemNumber);  sg.controls.Focus($("#Data\_ToItem"));  }  },  // variable for To Item finder.  ToItemFinder: function (data) {  if (data != null) {  clearStatisticsUI.clearStatisticsModel.Data.ToItemNumber(data.ItemNumber);  sg.controls.Focus($("#Data\_ThroughItemYear"));  }  },  // variable for Customer Year finder.  CustomerYearFinder: function (data) {  if (data) {  var year = data.FiscalYear;  if (sg.controls.GetString(clearStatisticsUI.clearStatisticsModel.Data.Year()) !== sg.controls.GetString(year)) {  clearStatisticsUI.clearStatisticsModel.Data.Year(year);  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear(year);  //Now Validate the method year  clearStatisticsRepository.getCustomerMaxPeriodForValidYear($("#Data\_ThroughCustomerYear").val())  }  }  },  // variable for Customer Group Year finder.  CustomerGroupYearFinder: function (data) {  if (data) {  var year = data.FiscalYear;  if (sg.controls.GetString(clearStatisticsUI.clearStatisticsModel.Data.Year()) !== sg.controls.GetString(year)) {  clearStatisticsUI.clearStatisticsModel.Data.Year(year);  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear(year);  //Now Validate the method year  clearStatisticsRepository.getGroupMaxPeriodForValidYear($("#Data\_ThroughGroupYear").val())  }  }  },  // variable for National Acct Year finder.  NationalAcctYearFinder: function (data) {  if (data) {  var year = data.FiscalYear;  if (sg.controls.GetString(clearStatisticsUI.clearStatisticsModel.Data.Year()) !== sg.controls.GetString(year)) {  clearStatisticsUI.clearStatisticsModel.Data.Year(year);  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear(year);  //Now Validate the method year  clearStatisticsRepository.getNationalAcctMaxPeriodForValidYear($("#Data\_ThroughNationalAcctYear").val())  }  }  },  // variable for Salesperson Year finder.  SalespersonYearFinder: function (data) {  if (data) {  var year = data.FiscalYear;  if (sg.controls.GetString(clearStatisticsUI.clearStatisticsModel.Data.Year()) !== sg.controls.GetString(year)) {  clearStatisticsUI.clearStatisticsModel.Data.Year(year);  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear(year);  //Now Validate the method year  clearStatisticsRepository.getSalespersonMaxPeriodForValidYear($("#Data\_ThroughSalesPersonYear").val())  }  }  },  // variable for Item Year finder.  ItemYearFinder: function (data) {  if (data) {  var year = data.FiscalYear;  if (sg.controls.GetString(clearStatisticsUI.clearStatisticsModel.Data.Year()) !== sg.controls.GetString(year)) {  clearStatisticsUI.clearStatisticsModel.Data.Year(year);  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear(year);  //Now Validate the method year  clearStatisticsRepository.getItemMaxPeriodForValidYear($("#Data\_ThroughItemYear").val())  }  }  }  };  var clearStatisticsUtility = {  checkIsDirty: function (yesFunctionToCall, noFunctionToCall) {  if (clearStatisticsUI.clearStatisticsModel.IsKoStatisticsDirty && clearStatisticsUI.clearStatisticsModel.IsKoStatisticsDirty.isDirty()) {  sg.utls.showKendoConfirmationDialog(  function () { // Yes  yesFunctionToCall.call();  },  function () { // No  noFunctionToCall.call();  },  $.validator.format(globalResource.SaveConfirm));  } else {  yesFunctionToCall.call();  }  },  setFocusToFiscalYear: function () {  sg.utls.focus("Data\_ThroughCustomerYear");  },  };  var clearStatisticsUtilities = {  customerYearBackup: null,  customerGroupYearBackup: null,  nationalAcctYearBackup: null,  salespersonYearBackup: null,  itemYearBackup: null,  customerPeriodBackup: null,  customerGroupPeriodBackup: null,  nationalAcctPeriodBackup: null,  salespersonPeriodBackup: null,  itemPeriodBackup: null,  backupCustomerYearValue: function () {  clearStatisticsUtilities.customerYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear();  },  backupGroupYearValue: function () {  clearStatisticsUtilities.customerGroupYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear();  },  backupNationalAcctYearValue: function () {  clearStatisticsUtilities.nationalAcctYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear();  },  backupSalespersonYearValue: function () {  clearStatisticsUtilities.salespersonYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear();  },  backupItemYearValue: function () {  clearStatisticsUtilities.itemYearBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear();  },  backupCustomerPeriodValue: function () {  clearStatisticsUtilities.customerPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerPeriod();  },  backupGroupPeriodValue: function () {  clearStatisticsUtilities.customerGroupPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupPeriod();  },  backupNationalAcctPeriodValue: function () {  clearStatisticsUtilities.nationalAcctPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctPeriod();  },  backupSalespersonPeriodValue: function () {  clearStatisticsUtilities.salespersonPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonPeriod();  },  backupItemPeriodValue: function () {  clearStatisticsUtilities.itemPeriodBackup = clearStatisticsUI.clearStatisticsModel.Data.ThroughItemPeriod();  },  refreshCustomerStatistic: function () {  clearStatisticsUtilities.backupCustomerPeriodValue();  clearStatisticsUtilities.backupCustomerYearValue();  },  refreshGroupStatistic: function () {  clearStatisticsUtilities.backupGroupPeriodValue();  clearStatisticsUtilities.backupGroupYearValue();  },  refreshNationalAcctStatistic: function () {  clearStatisticsUtilities.backupNationalAcctPeriodValue();  clearStatisticsUtilities.backupNationalAcctYearValue();  },  refreshSalespersonStatistic: function () {  clearStatisticsUtilities.backupSalespersonPeriodValue();  clearStatisticsUtilities.backupSalespersonYearValue();  },  refreshItemStatistic: function () {  clearStatisticsUtilities.backupItemPeriodValue();  clearStatisticsUtilities.backupItemYearValue();  },  validateCustomerYear: function () {  clearStatisticsRepository.getCustomerMaxPeriodForValidYear($("#Data\_ThroughCustomerYear").val());  },  validateGroupYear: function () {  clearStatisticsRepository.getGroupMaxPeriodForValidYear($("#Data\_ThroughGroupYear").val());  },  validateNationalAcctYear: function () {  clearStatisticsRepository.getNationalAcctMaxPeriodForValidYear($("#Data\_ThroughNationalAcctYear").val());  },  validateSalespersonYear: function () {  clearStatisticsRepository.getSalespersonMaxPeriodForValidYear($("#Data\_ThroughSalesPersonYear").val());  },  validateItemYear: function () {  clearStatisticsRepository.getItemMaxPeriodForValidYear($("#Data\_ThroughItemYear").val());  },  validateCustomerPeriod: function () {  sg.utls.clearValidations("frmClearStatistics");  var period = $("#Data\_ThroughCustomerPeriod").val();  var oldPeriod = clearStatisticsUtilities.customerPeriodBackup;  if ((parseInt(clearStatisticsUI.clearStatisticsModel.MaximumPeriod()) < parseInt(period)) || (period == "") || (period == "00") || ((period == "0"))) {  clearStatisticsUI.clearStatisticsModel.CustomerStatisticsCurrentPeriod(clearStatisticsUtilities.customerPeriodBackup);  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerPeriod(clearStatisticsUtilities.customerPeriodBackup);  } else if (oldPeriod != period) {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshCustomerStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  },  validateGroupPeriod: function () {  sg.utls.clearValidations("frmClearStatistics");  var period = $("#Data\_ThroughGroupPeriod").val();  var oldPeriod = clearStatisticsUtilities.customerGroupPeriodBackup;  if ((parseInt(clearStatisticsUI.clearStatisticsModel.MaximumPeriod()) < parseInt(period)) || (period == "") || (period == "00") || ((period == "0"))) {  clearStatisticsUI.clearStatisticsModel.CustomerStatisticsCurrentPeriod(clearStatisticsUtilities.customerGroupPeriodBackup);  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupPeriod(clearStatisticsUtilities.customerGroupPeriodBackup);  } else if (oldPeriod != period) {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshGroupStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  },  validateNationalAcctPeriod: function () {  sg.utls.clearValidations("frmClearStatistics");  var period = $("#Data\_ThroughNationalAcctPeriod").val();  var oldPeriod = clearStatisticsUtilities.nationalAcctPeriodBackup;  if ((parseInt(clearStatisticsUI.clearStatisticsModel.MaximumPeriod()) < parseInt(period)) || (period == "") || (period == "00") || ((period == "0"))) {  clearStatisticsUI.clearStatisticsModel.CustomerStatisticsCurrentPeriod(clearStatisticsUtilities.nationalAcctPeriodBackup);  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctPeriod(clearStatisticsUtilities.nationalAcctPeriodBackup);  } else if (oldPeriod != period) {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshNationalAcctStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  },  validateSalespersonPeriod: function () {  sg.utls.clearValidations("frmClearStatistics");  var period = $("#Data\_ThroughSalesPersonPeriod").val();  var oldPeriod = clearStatisticsUtilities.salespersonPeriodBackup;  if ((parseInt(clearStatisticsUI.clearStatisticsModel.SalesPersonMaximumPeriod()) < parseInt(period)) || (period == "") || (period == "00") || ((period == "0"))) {  clearStatisticsUI.clearStatisticsModel.SalesPersonStatisticsCurrentPeriod(clearStatisticsUtilities.salespersonPeriodBackup);  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonPeriod(clearStatisticsUtilities.salespersonPeriodBackup);  } else if (oldPeriod != period) {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshSalespersonStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  },  validateItemPeriod: function () {  sg.utls.clearValidations("frmClearStatistics");  var period = $("#Data\_ThroughItemPeriod").val();  var oldPeriod = clearStatisticsUtilities.itemPeriodBackup;  if ((parseInt(clearStatisticsUI.clearStatisticsModel.ItemMaximumPeriod()) < parseInt(period)) || (period == "") || (period == "00") || ((period == "0"))) {  clearStatisticsUI.clearStatisticsModel.ItemStatisticsCurrentPeriod(clearStatisticsUtilities.itemPeriodBackup);  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemPeriod(clearStatisticsUtilities.itemPeriodBackup);  } else if (oldPeriod != period) {  clearStatisticsUtility.checkIsDirty(clearStatisticsUtilities.refreshItemStatistic, clearStatisticsUtility.setFocusToFiscalYear);  }  }  };  var clearStatisticsUISuccess = {  fillCustomerFiscalYear: function (result) {  if (result == 0) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughCustomerYear(clearStatisticsUtilities.customerYearBackup);  } else {  clearStatisticsUI.clearStatisticsModel.MaximumPeriod(result);  clearStatisticsUtilities.backupCustomerYearValue();  setTimeout(function () {  ($("#Data\_ThroughCustomerPeriod")).siblings('input:visible').focus();  });  }  clearStatisticsUtilities.refreshCustomerStatistic();  },  fillGroupFiscalYear: function (result) {  if (result == 0) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughGroupYear(clearStatisticsUtilities.customerGroupYearBackup);  } else {  clearStatisticsUI.clearStatisticsModel.MaximumPeriod(result);  clearStatisticsUtilities.backupGroupYearValue();  setTimeout(function () {  ($("#Data\_ThroughGroupPeriod")).siblings('input:visible').focus();  });  }  clearStatisticsUtilities.refreshGroupStatistic();  },  fillNationalAcctFiscalYear: function (result) {  if (result == 0) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughNationalAcctYear(clearStatisticsUtilities.nationalAcctYearBackup);  } else {  clearStatisticsUI.clearStatisticsModel.MaximumPeriod(result);  clearStatisticsUtilities.backupNationalAcctYearValue();  setTimeout(function () {  ($("#Data\_ThroughNationalAcctPeriod")).siblings('input:visible').focus();  });  }  clearStatisticsUtilities.refreshNationalAcctStatistic();  },  fillSalespersonFiscalYear: function (result) {  if (result == 0) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughSalesPersonYear(clearStatisticsUtilities.salespersonYearBackup);  } else {  clearStatisticsUI.clearStatisticsModel.SalesPersonMaximumPeriod(result);  clearStatisticsUtilities.backupSalespersonYearValue();  setTimeout(function () {  ($("#Data\_ThroughSalesPersonPeriod")).siblings('input:visible').focus();  });  }  clearStatisticsUtilities.refreshSalespersonStatistic();  },  fillItemFiscalYear: function (result) {  if (result == 0) {  clearStatisticsUI.clearStatisticsModel.Data.ThroughItemYear(clearStatisticsUtilities.itemYearBackup);  } else {  clearStatisticsUI.clearStatisticsModel.ItemMaximumPeriod(result);  clearStatisticsUtilities.backupItemYearValue();  setTimeout(function () {  ($("#Data\_ThroughItemPeriod")).siblings('input:visible').focus();  });  }  clearStatisticsUtilities.refreshItemStatistic();  }  };  // Finder Cancel method  var onFinderCancel = {  FromCustomerFinder: function () {  sg.controls.Focus($("#Data\_FromCustomerNumber"));  },  ToCustomerFinder: function () {  sg.controls.Focus($("#Data\_ToCustomerNumber"));  },  FromCustomerGroupFinder: function () {  sg.controls.Focus($("#Data\_FromGroupCode"));  },  ToCustomerGroupFinder: function () {  sg.controls.Focus($("#Data\_ToGroupCode"));  },  FromNationalAccountFinder: function () {  sg.controls.Focus($("#Data\_FromNationalAccount"));  },  ToNationalAccountFinder: function () {  sg.controls.Focus($("#Data\_ToNationalAccount"));  },  FromSalespersonFinder: function () {  sg.controls.Focus($("#Data\_FromSalesperson"));  },  ToSalespersonFinder: function () {  sg.controls.Focus($("#Data\_ToSalesperson"));  },  FromItemFinder: function () {  sg.controls.Focus($("#Data\_FromItem"));  },  ToItemFinder: function () {  sg.controls.Focus($("#Data\_ToItem"));  },  CustomerYearFinder: function () {  setTimeout(function () {  sg.controls.Focus($("#Data\_ThroughCustomerYear"));  })  },  CustomerGroupYearFinder: function () {  setTimeout(function () {  sg.controls.Focus($("#Data\_ThroughGroupYear"));  })  },  NationalAcctYearFinder: function () {  setTimeout(function () {  sg.controls.Focus($("#Data\_ThroughNationalAcctYear"));  })  },  SalespersonYearFinder: function () {  sg.controls.Focus($("#Data\_ThroughSalesPersonYear"));  },  ItemYearFinder: function () {  setTimeout(function () {  sg.controls.Focus($("#Data\_ThroughItemYear"));  })  }  };  $(function () {  clearStatisticsUI.init();  }); |

* 1. Clear Statistics Worker Role Data Script

Access the **ClearStatistics\_WorkerRole\_Data.sql** file and replace the contents with

|  |
| --- |
| -- Copyright (c) 2020 Valued Partner All rights reserved.  -- Insert\_WorkerRole\_Data.sql  -- Insert data  SET ANSI\_NULLS ON  SET QUOTED\_IDENTIFIER ON  BEGIN  INSERT INTO [dbo].[WorkflowKind](WorkflowKindId, UniqueName, MaxRetries) VALUES (CONVERT(uniqueidentifier, '0ea99121-5386-46d1-8bed-e2da5811a7bd'), 'TUClearStatistics', 3)  INSERT INTO [dbo].[UnitOfWorkKind](WorkflowKindId, UniqueName, AssemblyName, TypeName, ExecutionOrder, IsAsynchronous)  VALUES(CONVERT(uniqueidentifier, '0ea99121-5386-46d1-8bed-e2da5811a7bd'), 'TU ClearStatistics', 'ValuedPartner.TU.Services', 'ValuedPartner.TU.Services.UnitOfWork.ClearStatisticsUow', 1, 1)  END  GO |

1. Review the Screen

At this point, the screen should be complete and functioning as expected:

