**Spencer Gifts**

**PO and Product Sample Review (POPSamples) Implementation**

**High Level Design Document**



Prepared by : Nagarjun Tirunagari

Reviewed by : Sarvani Josyam

Date Submitted : 7/20/2020

Version : 1.0

Last revised date : 7/20/2020

This document is being submitted to **Spencer Gifts** by **WinWire Technologies, Inc**. This document and any supporting appendices submitted with it are confidential and intended solely for the use of the business users and IT team working at **Spencer Gifts**.

**Revision History**

**Change Record**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Author** | **Version** | **Change reference** |
| 7/20/2020 | Krishna vardhineedi | 1.0 | Initial draft. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Reviewers**

|  |  |  |
| --- | --- | --- |
| **Name** | **Version approved** | **Date** |
| Sarvani Josyam | 1.0 | 7/20/2020 |
|  |  |  |
|  |  |  |

Table of Contents

[1. Introduction 4](#_Toc47342181)

[2. Purpose/Audience 5](#_Toc47342182)

[3. Goals and Objectives 5](#_Toc47342183)

[4. Scope and Assumptions 6](#_Toc47342184)

[4.1 In Scope 6](#_Toc47342185)

[4.2 Out of scope 6](#_Toc47342186)

[4.3 Assumptions 6](#_Toc47342187)

[5. Solution Overview 7](#_Toc47342188)

[5.1 Worksheet Page 7](#_Toc47342189)

[5.1.1 UI Changes 7](#_Toc47342190)

[5.1.2 API Changes 7](#_Toc47342191)

[5.1.3 Backend Changes 8](#_Toc47342192)

[5.2 PO Page 8](#_Toc47342193)

[5.2.1 UI Changes 8](#_Toc47342194)

[5.2.2 API Changes 9](#_Toc47342195)

[5.2.3 Backend Changes 9](#_Toc47342196)

[5.3 PO Cost Adjustment Page 9](#_Toc47342197)

[5.4 Admin Page 9](#_Toc47342198)

[5.4.1 Vendor Term Codes 9](#_Toc47342199)

[5.5 Settings Page 10](#_Toc47342200)

[5.5.1 User Roles List 10](#_Toc47342201)

[6. High Level Design/Application Architecture 10](#_Toc47342202)

[6.1 Development Environment Requirements 11](#_Toc47342203)

[6.2 Layered Architecture Design Details: 11](#_Toc47342204)

[6.2.1 Model Layer 11](#_Toc47342205)

[6.2.2 View Layer 11](#_Toc47342206)

[6.2.3 Controller Layer 11](#_Toc47342207)

[6.2.4 Exception Handling Layer 12](#_Toc47342208)

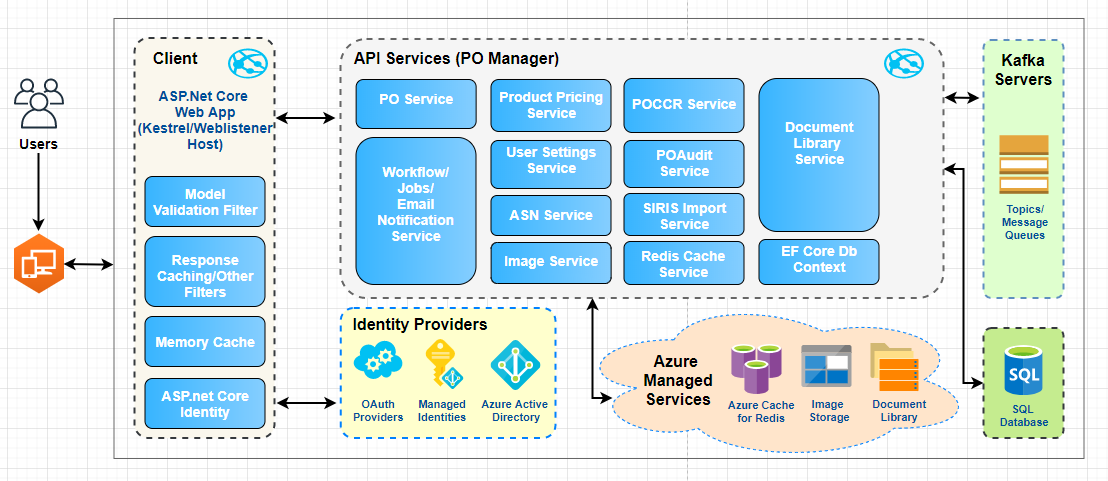
[7. Terms and definitions 12](#_Toc47342209)

[8. References 12](#_Toc47342210)

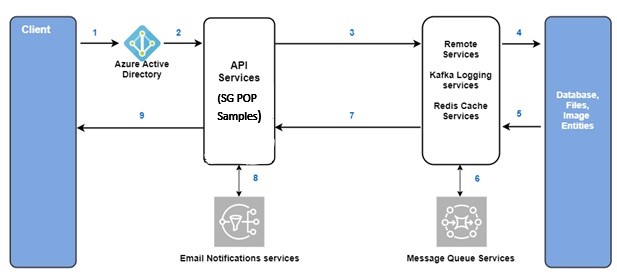
# Introduction

Spencer's Spirit Holdings, Inc. wishes to implement a flexible and configurable purchase order and product Review system.

Following is a high-level system architecture overview of POPSample Spencer's Spirit Holdings application at presentation level.



Following is the Purchase Order workflow approval process flow.



1. User create TOP Review, Contractual ,CartonScan.
2. Azure active directory active token will be verified and pushes the request to next level.
3. API services processes the request by interacting with necessary remote services.
4. Remote services processes request and pushes the request to next Database level to update the records.
5. Database saves the records and returns the status of the process.
6. Once upon receiving the success flag status, it will send the logs messages in Kafka queue systems.
7. Remote services push the success response back to API services.
8. API services will push email notifications to the respective users review status.
9. API services will send response back to UI and page will be refreshed with latest review workflow processed status information.

# Purpose/Audience

The purpose of this document is to serve as the governing and specification document to add POPSamples functionality to the admins to better serve the customers and provide them with a better ease of maintenance. This functionality will allow the company to maintain purchase order information at a lower complexity while providing the different application users with more options with flexibility.

# Goals and Objectives

The primary objective of the POPSamples project is to provide the admin/application user faster ease of maintaining TOP, Contractual reviews:

* Provide user to view and maintain information such as Shipping dates, Days since scan, Vendor/Sub-Vendor information, review information.
* Provide user to view/maintain reviews by Buyer, Creative, QA/QC, Product Development.
* Ability to search Reviews by product code all TOP, Contractual, CartonScan.
* Ability to maintain edit and Complete the review.
* Cancelling the reviews.
* Ability to complete the review and notify to the vendor completed or rejected.
* Track all the user permissions available in POPSamples user info page as part of the POPSamples implementation. This is used for analytics/maintains purpose.

# Scope and Assumptions

## In Scope

1. Add TOP, Contractual ,CartonScan review to the systems using the Create pages.
2. UI changes to the Create/List pages, settings pages.
3. UI should be responsive (Tablet devices as well as Desktop, Desktop big screens range from 17inchs to 24 inches monitors).
4. Application should support Chrome browser.

## Out of scope

To be added

## Assumptions

1. Buyer accesses a page/form that shows all Sample Reviews for a PO in their brand/LOB that they have not completed.
2. Buyer accesses a page/form that shows all Licensor Reviews for a PO in their brand/LOB that they have not been completed by Licensing.
3. Any member of PD (Product Development) Team accesses a page/form that shows all Sample Reviews that have not been completed by the buyer for which the associated SKU is flagged as "PD Merch".
4. Member of the PD group accesses list of open sample reviews
5. Authorized users (Spencer Identity server users) for accessing the application will be maintained by spencer.
6. Core API SG POPSamples services and other integration services will be provided/maintained by spencer team.
7. Back-end Stored procedures logics and tables creation will be maintained by spencer.
8. Logging and Kafka server configurations will be maintained by spencer team.
9. Redis cache connections services, Custom Spencer NuGet packages, Image library services, Document library services will be provided and maintained by spencer team.

# Solution Overview

The proposed system summary is to develop POPSamples features to better serve the admin users and provide them with a better maintenance. This functionality will allow the company to offer faster review the products at a less fault tolerance while providing the admin user with more options.

## Create Review Page

### UI Changes

1. Add the menu item “TOP”,” Contractual”,”CartonScan” in the header section of the page. On click of the menu item. Open the list of the items. In the items Create TOP Review, Completed Reviews, Search Completed review for the Sample room coordinator (SRC).
2. Clicking create review, it’s open to the create review page.
3. SRC either types a SKU and searches for matching product data.
4. Product information for the scanned SKU is displayed:
   1. Master SKU
   2. Department/Class
   3. Buyer
   4. Sample Prep Date (today's date)
   5. Vendor
   6. SKU
   7. Vendor Style Number
   8. Short Ticket Description - NOTE: not sure if this is currently available as part of query service.  Currently available in mms\_product\_label\_code table in mms\_integration
   9. D/F - whether or not the vendor is Foreign (F) or Domestic (D)
   10. PO - the PO number of the open PO containing the scanned/entered SKU. If multiple are found, use one for which the SKU is included, and not canceled, with the earliest ship date.  If none found, display "No open PO found"
   11. Ship Date - should be the earliest ship date for the sku on the matching PO.  If no matching PO is found, nothing shows
   12. Full Description
   13. Ticket Retail - if there is a different Canada retail price, show both US and Canada ($10.99/C$12.99)
   14. PD Merch (Y/N)
5. SRC indicates that a review process should be created. The Sample Prep Date of new review will be the current date
6. Page/form resets so that another SKU can be searched, and a new review created for it
7. Completed Reviews list should be able to filter based on
8. All the fields,
9. Worksheets grid should be searchable.
10. Grid should show 25rows, rows count information and filtered rows count information in the page.
11. User should be able to go to individual review page on row click in the grid.
12. Page should display TOP Review information as per below sections.
13. **Summary** Panel – Should have information for fields – Department/Class, Ship Date, Sample Prep Date, Master SKU Code, Sku#, Vendor Style#, Vendor, Ticket Retail, Buyer and other important fields. Summary should be read-only.
    1. **SKUs** Panel – Should display SKU’s information in grid format.
    2. **PrePack SKUs** Panel – Should display Prepack SKU’s information in grid format
    3. **Dropship Locations** Panel – Should display Dropship SKU’s information in grid format.
    4. **Sub Vendor-A/P Vendor** Panel – Should display vendor related information.
    5. **Terms** Panel– Should display discounts and allowances information.
    6. **Logistics** Panel – Should display routing and payment information.
14. User should be able to Edit/Delete Worksheets.
15. User should be able to jump to worksheet without navigating to grid

### API Changes

API services should be created -

1. To load data in the grid based on Role, Login User or All Users.
2. To load the individual review pages.
3. To Create/Update/Cancel review data.
4. Create a new Controller which responds to the UI events.
5. Create action methods to Save and Complete actions.
6. Handle the calls to api from a service layer.
7. Create needed mapping classes using Auto mapper.

### Backend Changes

Stored procedures should be created -

1. To load data in the grid based on Role, Login User or All Users.
2. To load/Update/Cancel the individual review pages.
3. Create new action methods in Query service to get all the cost adjustment data based on the Review Id.
4. Handle the calls to api from a service layer.
5. Create needed mapping classes using Auto mapper.
6. Generate Entities using EF core.

## Admin Page

### User Info

This screen should provide information about the user roles across the application.

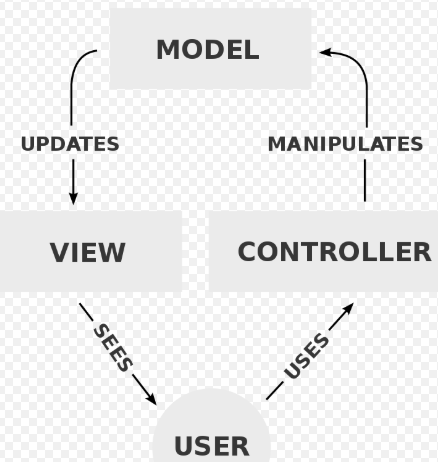
1. Add the menu “Admin” in the header section of the page. Provide “User Info” menu item option in Admin menu.
2. On click of the menu item , should open a popup box with user roles information.
3. Should show Top review roles.
4. Should show Contractual review roles.
5. Should show Carton Scan review roles.
6. Necessary API services should be created to load the roles information.

### Configuration Settings

This screen should provide information about how the application behaves with key and value pairs.

1. Add the menu Admin in the header section of the page. Provide “Configuration Settings” menu item option in Admin menu.
2. On click of menu item, should open a popup with Configuration Settings list in the grid.
3. Should provide options to Add New/ Edit Configuration Settings data.
4. On click of Add New, it should display Key, Description fields to provide and should save the data to Db.
5. User should be able to select and edit the value in the grid.
6. User should be able to cancel the changes in between during editing. It should warn user if any pending changes are cancelling without saving to db.
7. Necessary API, repository methods should be created to save the records to database tables.
8. Validations like mandatory fields , max length should be implemented as per DB design.
9. Pop up should close on click of ‘X’ button or ‘Escape’ key.

# High Level Design/Application Architecture



## Development Environment Requirements

Windows/Mac machine with the following software:

|  |  |
| --- | --- |
| **S.No** | **Software Name** |
|  | .Net core 2.1 |
|  | SQL Server |
|  | EF Core 2.0 |
|  | Fluent Validations 8.4 |
|  | BuildBundlerMinifier 3.2 |
|  | Git Source Code Repository |
|  | Java Script |
|  | CSS |
|  | jQuery 3.4.1 |
|  | AutoMapper 6.2.2 |
|  | KellermanSoftware CompareNetObjects |
|  | Bootstrap 4.3.1 |
|  | Kafka |
|  | Log4J |
|  | ASP.Net MVC |
|  | Swagger |
|  | Jquery datatables |
|  | StackExchange.Redis |

## Layered Architecture Design Details:

The application consisting of following Layers which explained in section 1 in this document along with Diagram.

### 6.2.1 Model Layer

The central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic, and rules of the application

### 6.2.2 View Layer

Any representation of information such as a chart, diagram, or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.

### 6.2.3 Controller Layer

Accepts input and converts it to commands for the model or view. In addition to dividing the application into these components, the model–view–controller design defines the interactions between them.

1. The model is responsible for managing the data of the application. It receives user input from the controller.
2. The view means presentation of the model in a format.
3. The controller responds to the user input and performs interactions on the data model objects.
4. The controller receives the input, optionally validates it and then passes the input to the model.

### 6.2.4 Exception Handling Layer

The Exception handling layer handling all Database and user defined exceptions.

# Terms and definitions

|  |  |
| --- | --- |
| **Acronyms and Abbreviations** | **Definitions** |
| PO | Purchase Order |
| SKU | Stock Keeping Unit |
| DB | Database |
| MVC | Model View and Controller |
| LIC | Contractual |
| PD | Product Development |

# References

1. User stories defined by client
2. Spencer Gifts - PO Cost Adjustment Implementation - HLD - July 20 2020.docx

**\*\*\***