

**COT 6931 Server Side Design Document**



Food Giant Sales Flyer Generator



Table of Contents

[Revision History 2](#_Toc478153465)

[Section 1 Introduction 3](#_Toc478153466)

[Section 1.1 Project Purpose 3](#_Toc478153467)

[Section 1.2 Server Side Summary 3](#_Toc478153468)

[Section 1.3 Requirements Satisfied 3](#_Toc478153469)

[Section 2 Server Side Design Structure 3](#_Toc478153470)

[Section 2.1 Design Overview 3](#_Toc478153471)

[Section 2.1 Server Side Classes 4](#_Toc478153472)

[Section 2. Additional Components 4](#_Toc478153473)

[Section 3 Detailed Design 5](#_Toc478153474)

[Section 3.1 Flyer Creator Classes 5](#_Toc478153475)

[Section 3.1.1 Overview of Flyer Creator Class 5](#_Toc478153476)

[Section 3.1.2 View 6](#_Toc478153477)

[Section 3.1.3 ViewModel 6](#_Toc478153478)

[Section 3.2 Created Flyer Page Classes 6](#_Toc478153479)

[Section 3.2.1 Overview of Created Flyer Page Class 6](#_Toc478153480)

[Section 3.2.2 View 6](#_Toc478153481)

[Section 3.2.3 ViewModel 6](#_Toc478153482)

[Section 3.3 Database Structure 6](#_Toc478153483)

[Section 3.4 Database Maintainer Classes 7](#_Toc478153484)

[Section 3.4.1 Overview of Database Maintainer Class 7](#_Toc478153485)

[Section 3.4.2 View 8](#_Toc478153486)

[Section 3.4.3 ViewModel 8](#_Toc478153487)

[Section 4 Terms of Reference 8](#_Toc478153488)

# Revision History

|  |  |  |
| --- | --- | --- |
| Revision # | Date | Changed Items |
| 1 | 02/26/2017 | Initial Document Creation |
| 2 | 03/04/2017 | Added Information on Design Strategies |

# Section 1 Introduction

This class is designed to retrieve data from a database and show these selections to a user in order to populate a page with images of the selected items. Also, this will allow a user to set a price for each item.

## Section 1.1 Project Purpose

The Food Giant Flyer Creation program is designed to allow Food Giant store managers more control over selling their products. This program will contain an easy to use interface that allows them to create custom flyers that contain items and pricing of their choosing, after approved by district managers.

## Section 1.2 Server Side Summary

These Server Side Projects are designed to retrieve data from a database and show these selections to a user in order to populate a page with images of the selected items. Also, this will allow a user to set a price for each item.

## Section 1.3 Requirements Satisfied

The Server Side Program applies to and satisfies the following requirements:

* SR 1.2
* SR 1.2.1
* SR 1.3
* SR 1.3.1
* SR 1.3.2
* SR 1.4
* SR 1.4.1
* SR 1.4.2
* SR 2.1
* SR 2.2
* SR 2.3
* SR 2.4
* SR 3.1
* SR 3.2

# Section 2 Server Side Design Structure

## Section 2.1 Design Overview

The overall program will follow a Model View ViewModel (MVVM) design. This means that each visual element (View) will contain as little code as possible, except for the visual components. The ViewModel classes will primarily drive the logic in the code and instantiate the View classes for the user to see. Any events take by the user (button click, item select, etc.) will inform the View Model of the action and allow it to handle the logic behind what to do on each of these actions.

The Model class will be a “.cs” file that will contain data on what needs to be entered to fulfill a criterion for an object.

To accomplish this effectively, we will be using the Caliburn Micro framework, a completely free Application Program Interface (API) that allows us to very quickly and effectively bind ViewModels, Objects, and Events together with very little redundant code. To use Caliburn, we will enforce class naming for all classes created in the Flyer Generator program. This naming convention is defined as:

1. View – *ClassName*View
2. ViewModel - *ClassName*ViewModel
3. Model - *ClassName*Model

This allows a new team to quickly identify what the class’s overall purpose is just from its name, making maintenance easier.

Variable naming conventions will be enforced and will be defined in a short “Coding Standards” document. This is again to enforce a consistent design and assist with maintainability.

## Section 2.1 Server Side Classes

The Server Side will consist of a single project, FoodGiantFlyer. The FoodGiantFlyer project will consist of the following classes:

Views:

* FlyerCreatorView
* DatabaseMaintainerView
* BasicFlyerTemplateView

ViewModels:

* FlyerCreatorViewModel
* DatabaseMaintainerViewModel
* BasicFlyerTemplateViewModel

Models:

* FlyerDataModel

Others:

* AppBootstrapper
* DatabaseInterface

## Section 2. Additional Components

In addition to the classes listed in the previous section, the Server Side program will contain the following components:

* Database – FoodGiantItemSQLDatabase.mdf. This database will contain all data for the Food Giant Items entered in by an admin. This is an integral part of the FlyerCreator classes.
* Images Folder – Located In subfolder under exe folder. This folder contains all added images by an admin. This is an integral part of the FlyerCreator classes.

These components will be discussed further in the detailed design section where applicable to the classes that require their usage.

# Section 3 Detailed Design

This section will detail the purpose of each class, the SRS requirements it affects and a description of the classes’ components.

## Section 3.1 Flyer Creator Classes

### Section 3.1.1 Overview of Flyer Creator Classes

The Flyer Creator classes handle displaying Food Giant Inventory Items, retrieved from the database, to the user. The user will select the flyer template they wish to use, change the price to what they wish for the sale price to be, then select images they wish to use for the flyer.

After the manager has selected the items they want to use for the flyer, this class will pass the selected items to the select flyer template page to generate a printable flyer.

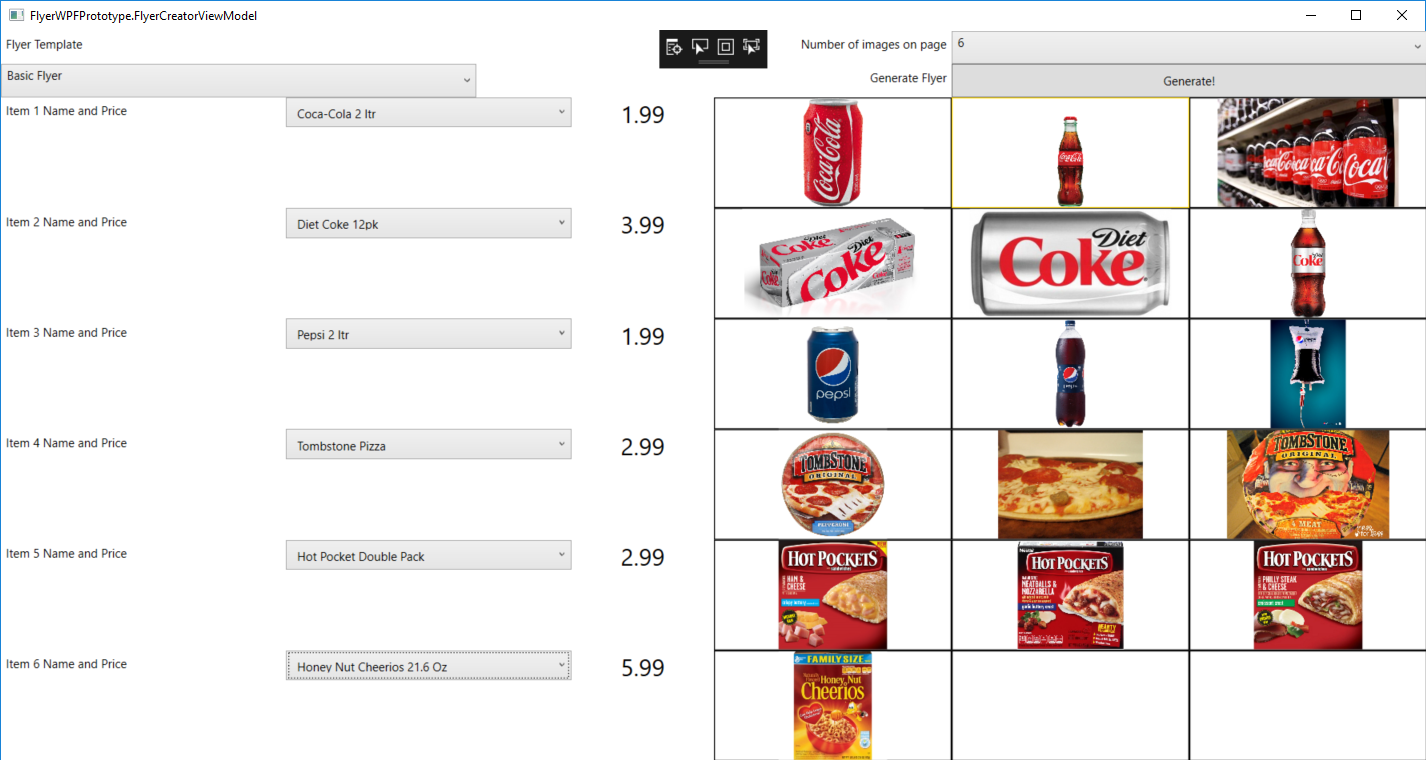


Image 1.1 Flyer Creator Prototype Picture

These classes satisfy requirements

* SR 1.2
* SR 1.2.1
* SR 3.1
* SR 3.2

### Section 3.1.2 View

The FlyerCreatorView will contain all the Visible Components that the user can interact with. All logic handling the user events will reside in the FlyerCreatorViewModel, keeping in line with the MVVM standard.

The View will contain multiple drop down menus where the user can select the number of items they wish to display on the flyer, the flyer type they want, and the items they wish to use for the flyer. Each selected item will populate a text box containing the item’s default price, and one to three images that the user can select from to use for the flyer. A user will select the item they want for the flyer by simply clicking the image, which will highlight the border around it Gold. If there is only one image, that image will be selected by default.

### Section 3.1.3 ViewModel

The FlyerCreatorViewModel will be the primary C# class used for the Food Giant Program.

The FlyerCreatorViewModel will contain all the actions and logic for the FlyerCreatorView. It will also instantiate the View and populate all the data for the visible controls, keeping in line with the MVVM standard.

This class will also use the DatabaseInterface class to retrieve data from the SQL database to populate in the drop down boxes.

When a user selects the number of images to display on the page, this class will hide the extra selections.

## Section 3.2 Basic Flyer Template Classes

### Section 3.2.1 Overview of Basic Flyer Template Classes

These classes satisfy requirements:

* SR 1.4
* SR 1.4.1

### Section 3.2.2 View

### Section 3.2.3 ViewModel

## Section 3.3 Database Structure

The SQL database satisfies requirements:

* SR 2.1
* SR 2.2

The SQL Database will contain the following tables:

* FlyerHistory
* ItemList

### Section 3.3.1 Flyer History Table

The FlyerHistory Table’s purpose is to store all parameters entered by a Store Manager when they generate a flyer. This is to keep a company record of flyers created and to allow a district manager the ability to see what flyers have been created.

These values will be read out by the program to generate the exact flyer the Manager used instead of saving the entire flyer image onto the hard drive, saving hard drive storage space and keeping the data better protected.

Each FlyerHistory Table entry will contain the following fields:

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Allow Nulls |
| ID (Primary Key, Unique) | Int | No |
| Manager Name | Text | No |
| Template Name | Text | No |
| Item 1 Name | Text | No |
| Item 1 Price | Int | No |
| Item 1 Image | Text | No |
| Item 2 Name | Text | Yes |
| Item 2 Price | Int | Yes |
| Item 2 Image | Text | Yes |
| Item 3 Name | Text | Yes |
| Item 3 Price | Int | Yes |
| Item 3 Image | Text | Yes |
| Item 4 Name | Text | Yes |
| Item 4 Price | Int | Yes |
| Item 4 Image | Text | Yes |
| Item 5 Name | Text | Yes |
| Item 5 Price | Int | Yes |
| Item 5 Image | Text | Yes |
| Item 6 Name | Text | Yes |
| Item 6 Price | Int | Yes |
| Item 6 Image | Text | Yes |

* ID (Primary Key, Unique) – Unique Identifier for each flyer created
* Manager Name – This is the Name and ID number of the manager that created the flyer
* Template Name – This is the template the Manager used to create the flyer. This is needed for the Flyer Creator program to find the correct template type.
* Item Name – These fields will be used to populate the item name in the recreated flyer
* Item Image – These fields will be used to populate the item image in the recreated flyer
* Item Price – These fields will be used to populate the item price in the recreated flyer

### Section 3.3.2 Item List Table

The ItemList Table’s purpose is to store the vast amount of products Food Giant sells. This data will be pulled out of the database to allow a manager the ability to select the item they want, and display the images associated with the item.

The ItemList Table will contain the following fields:

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Allow Nulls |
| ID (Primary Key, Unique) | Int | No |
| Item Name (Unique) | Text | No |
| Item Category | Varchar(50) | No |
| Item Price | Int | No |
| Image Name 1 | Text | No |
| Image Name 2 | Text | Yes |
| Image Name 3 | Text | Yes |

ID – Unique Identifier for each Food Giant Inventory database item

Item Name – Unique Identifier for each Food Giant Inventory database item

Item Category – Category to group each item in. Will be used to quickly view number of items for each category, which is useful for data gathering

Item Price – This is the standard price of the item.

Image Name - These fields handle the images associated for each item. The second and third item can be null, since there may only be one required image for the item entered.

## Section 3.4 Database Maintainer Classes

### Section 3.4.1 Overview of Database Maintainer Classes

The Database Maintainer classes purpose is to allow a manager or manager’s assistant to add new items to the SQL Database. Since neither person will be expected to have any database experience, we will need to create these classes to simplify the process and add in error handling to reduce the chance of a bad database entry.

This class will also back-up the databases when a manager or assistant finishes making database updates. This will be another safety net to further reduce the chance of data loss.

These classes satisfy requirements

* SR 2.3
* SR 2.4

### Section 3.4.2 View

The Database Maintainer’s View will contain a very similar layout to the Item Name, Price and Image section of the Flyer Creator View. The difference being that the DM View will only allow one entry at a time, and have a log below the entry fields allowing the user to see what actions have been done. This is important if the user is expecting to enter in a significant amount of data entries.

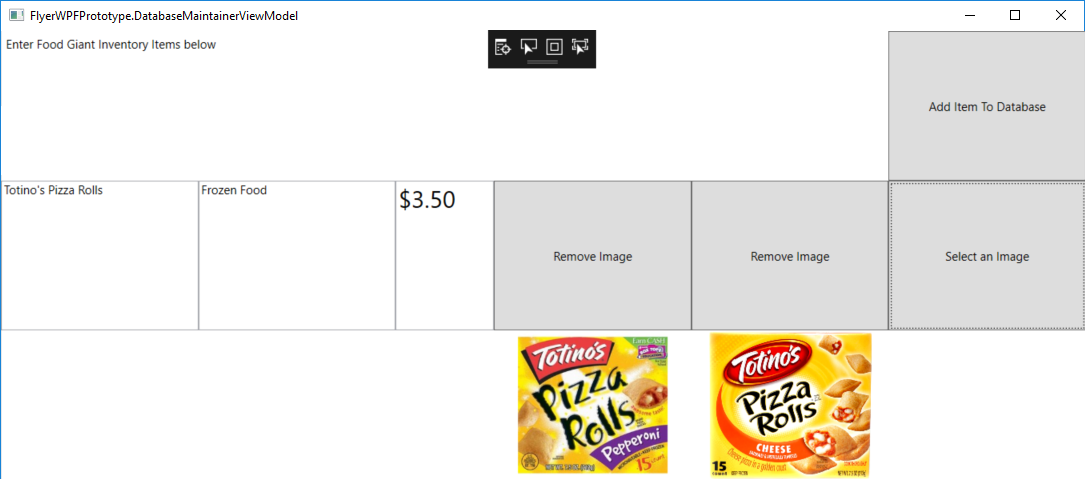


Image 1.1 Database Maintainer Prototype Picture

### Section 3.4.3 ViewModel

## Section Flyer History Classes

These classes satisfy requirements

* SR 1.3
* SR 1.3.1
* SR 1.3.2

## Section 3.5 Database Interface

This class will contain all the database queries used by the application. Database Interface will be the **only** class that contains queries to connect database. This is to enforce a secure, central location to retrieve data from the database.

This class will contain the following methods:

* SQLCommand CreateConnectionString
* SQLCommand GetInventoryItems
* SQLCommand EnterNewInventoryItem
* SQLCommand UpdateInventoryItem
* SQLCommand GetFlyerTemplateHistory
* Void ConnectToDatabase
* Void RunSQLCommand

All connects to the database will go through the Connect to Database method, which will contain all error handling and Query Validation. This is to have a single protected entry point into the database reducing the chance of damaging the database from a malformed query.

## Section 3.6 FlyerDataModel

This model will contain the valriables needed to create a new Food Giant Inventory item. This model will be used in the Database Maintainer Class for when the user adds new items to the database to validate the items before entered in the database query, and to allow the Flyer Creator classes to pull out the individual values to populate the data fields.

This class will consist of the following values:

* public string itemName
* public string itemCategory
* public string itemPrice
* public string imageName1
* public string imageName2
* public string imageName3

# Section 4 Terms of Reference

|  |  |
| --- | --- |
| Term | Definition |
| ASP.NET | Active Server Pages |
| GUI | Graphical User Interface |
| MVVM | Model View View-Model |
| QA | Quality Assurance |
| SDD | Software Design Description |
| SPMP | Software Project Management Plan |
| SQL | Structured Query Language |
| SRS | Software Requirements Specifications |
| STD | Software Test Document |