## 2D Shape Drag and Drop in WPF using MVVM, PRISM and SQLite

## Introduction

## In this article, the goal is to show how to implement drag-and-drop of 2D shapes using MVVM pattern with PRISM and SQLite database. 2D Shape Drag and Drop in WPF using MVVM, PRISM, and SQLite" refers to a software development project that involves building a graphical user interface (GUI) for a desktop application in Windows Presentation Foundation (WPF). The goal of this project is to implement a drag-and-drop feature for 2D shapes, which is a common requirement in many applications. The application will be built using the MVVM (Model-View-ViewModel) design pattern, which is a widely-used design pattern in WPF and Silverlight applications. MVVM separates the presentation logic from the business logic, making the code easier to maintain and test. The project will also make use of Microsoft Prism, which is a set of libraries that help developers build WPF and Silverlight applications that are loosely-coupled, maintainable, and testable. Prism provides an implementation of MVVM and provides additional services such as event aggregation and module management. Finally, the application will use SQLite as the database technology for storing data. SQLite is a lightweight and fast database that is suitable for use in desktop applications and is easy to integrate with .NET applications using ADO.NET.

## 2D Shape Drag and Drop in WPF

## The drag-and-drop facilities provided by WPF are designed to be highly flexible and customizable to support a wide variety of drag-and-drop scenarios. Drag-and-drop supports manipulating objects within a single application, or between different applications. Dragging-and-dropping between WPF applications and other Windows applications is also fully supported. The requirement for 2D Shape Drag and Drop in WPF stems from the need for a user-friendly interface for manipulating and arranging shapes in the application. This functionality will enable users to easily arrange shapes according to their needs, improving the overall usability and user experience of the application.

## MVVM design pattern

## MVVM is a design pattern that separates the presentation logic from the business logic, improving the maintainability and testability of the code. By using MVVM, the application will have a clean and well-structured architecture, making it easier to understand, modify, and extend in the future. This pattern ensures the UI of the application, and the underlying presentation and business logic is separated into three separate classes: the view, which encapsulates the UI and UI logic; the view model, which encapsulates presentation logic and state; and the model, which encapsulates the application's business logic and data.

## Microsoft Prism

## Microsoft Prism is a framework that provides a set of libraries and guidance for building WPF applications using MVVM. It helps to simplify the implementation of complex MVVM scenarios and provides a set of common components and services that can be reused throughout the application.

## Implementation of Microsoft Prism in the application

## In the 2D Shape Drag and Drop application, Microsoft Prism will be used to implement the MVVM design pattern, handle navigation, and manage the dependency injection. This will improve the overall structure and maintainability of the code.

## SQLite Database

## SQLite is an open-source, lightweight, self-contained and zero-configuration relational database management system. It is a popular choice for embedded systems, mobile devices, and web applications because of its small size, low memory footprint, and high performance. SQLite supports the SQL standard, including transactions, ACID compliance, and multiple concurrent connections. It can be used to store, organize, and retrieve data in a structured manner, making it a powerful tool for managing data in applications. The Sqlite database can be used in C#.NET applications with the help of ADO.NET, LINQ, and Entity Framework. In the context of the project "2D Shape Drag and Drop in WPF using MVVM, Microsoft Prism, and Sqlite," the Sqlite database would be used to store and manage the data related to the shapes being dragged and dropped on the WPF canvas. It is ideal for use in desktop applications and can be used to store and manage the data for the 2D Shape Drag and Drop application.

## Implementation of Sqlite in the application

## In the 2D Shape Drag and Drop application, Sqlite will be used to store and manage the data related to the shapes and their arrangements. The SQLite database can be managed using ADO.NET, which enables developers to interact with the database using C# code.

## Drag and Drop functionality

## The drag and drop functionality will allow users to select a shape, drag it to a desired location in the user interface, and drop it to complete the arrangement. This functionality will be implemented in the WPF application using the MVVM design pattern and Microsoft Prism.

## Implementation of the drag and drop functionality

## The drag and drop functionality will be implemented using event handlers in the ViewModel and event triggers in the View. The ViewModel will handle the logic for the dragging and dropping of shapes, while the View will display the shapes and handle the user interactions. The data related to the shapes and their arrangements will be stored in the Sqlite database and managed using Entity Framework Core.

## Database Configuration and Development Tools

Database Configuration

This section considers the Database Management System Used in the development process.

* Sqlite Database

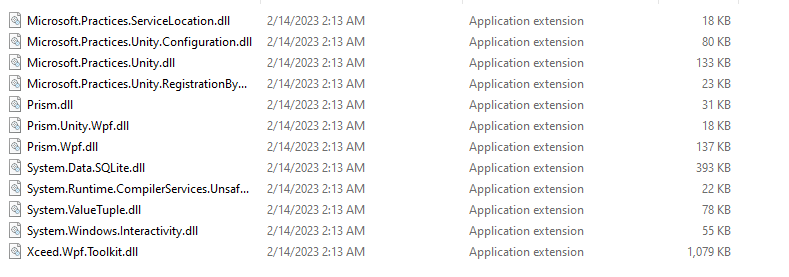
Name : Shape\_db

Development Tools

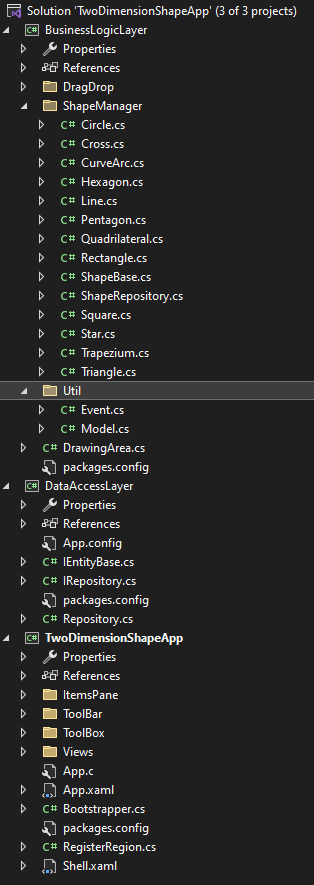
* Visual Studio 2022
* Microsoft Windows
* C#.Net Programming language

## Dependencies/Libraries

* Below PRISM packages are for creating loosely coupled application, and adding the color picker to the toolbar



## PROJECT LAYOUT IN VISUAL STUDIO



## DESIGN AND DEVELOPMENT APPROACH

## Design Patterns

## The following Software development patterns were adopted for the development of this test project

* Structural Pattern: Adapter Pattern, Decorator Pattern
* Behavioral Patterns: Strategy Pattern, Command Pattern, Observer Pattern etc.
* Creational Patterns: Singleton Pattern

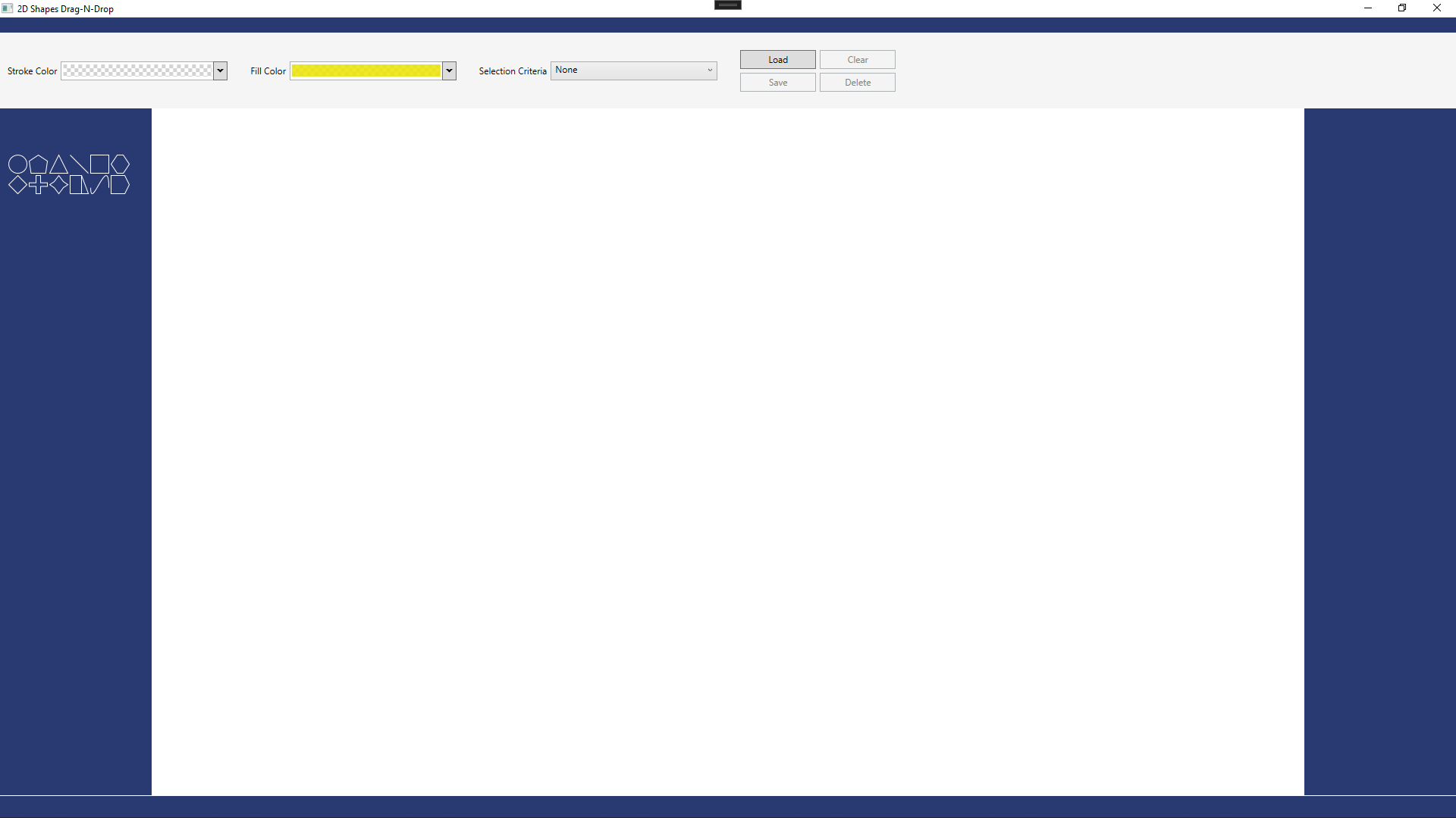
## Development Approach

Object Oriented Development Approach was adopted to keep the implementation process of the test projectclean, easily maintainable and readable.

## GUI Descriptions

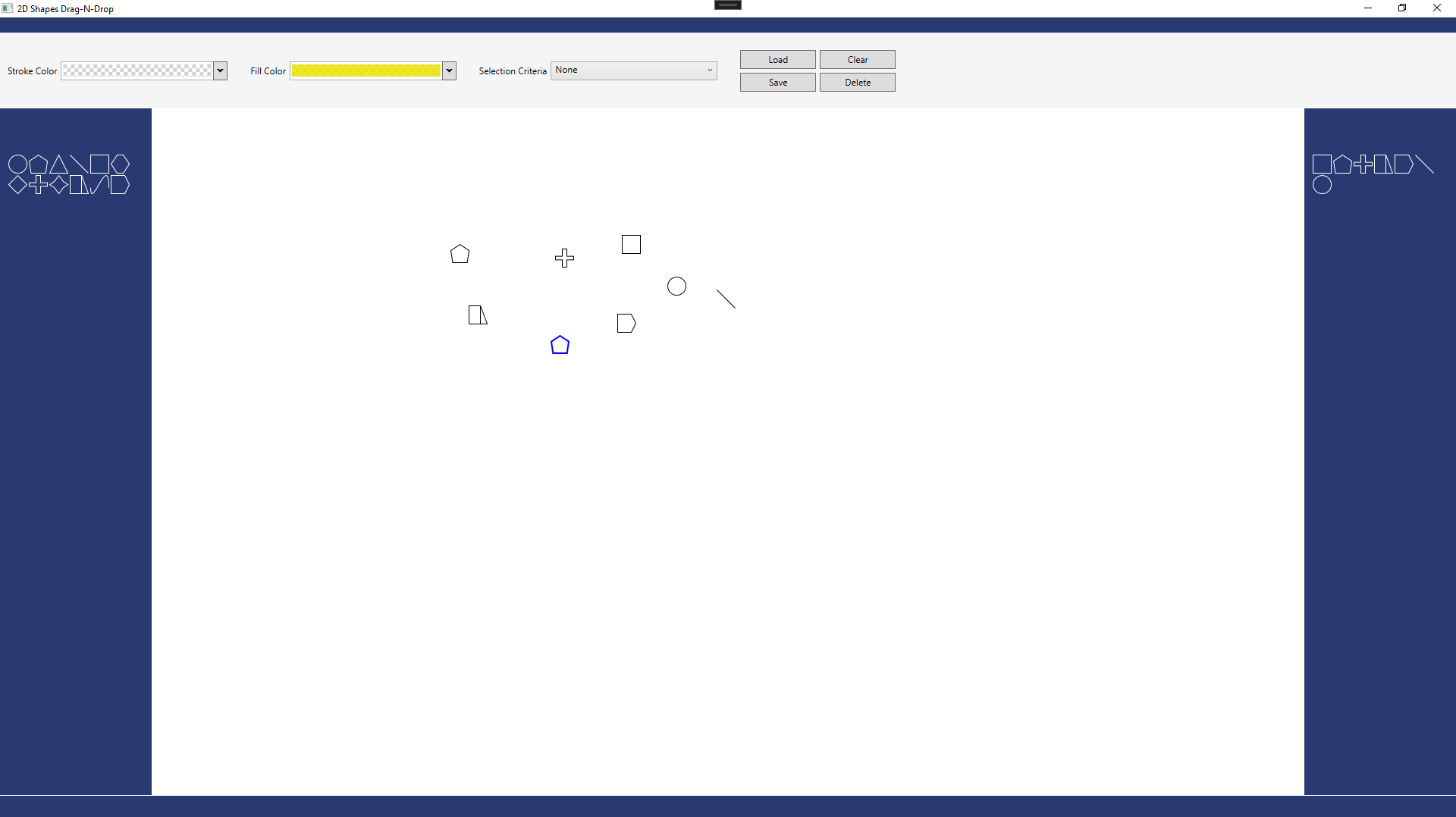
Start Up Window

* Application start up Window contains toolbar, Toolbox, drawing Canvas and Items Pane and
* The toolbox contains the shown WPF controls



*Figure 1: Application Launch Window*

DRAGGING SHAPES TO CANVAS

The user should be able to drag an item from the toolbox and place it on the drawing canvas. The selected item should be placed on the user’s drop location  *Figure 2: Dragging Shapes to Drawing Canvas*

BLUE COLOR HIGHLIGHTING ON A SELECTED SHAPE

The enables the user to select any item on the canvas by clicking it, then a blue rectangle will appear around any selected item(s) and selected item(s) can also be dragged around the canvas.

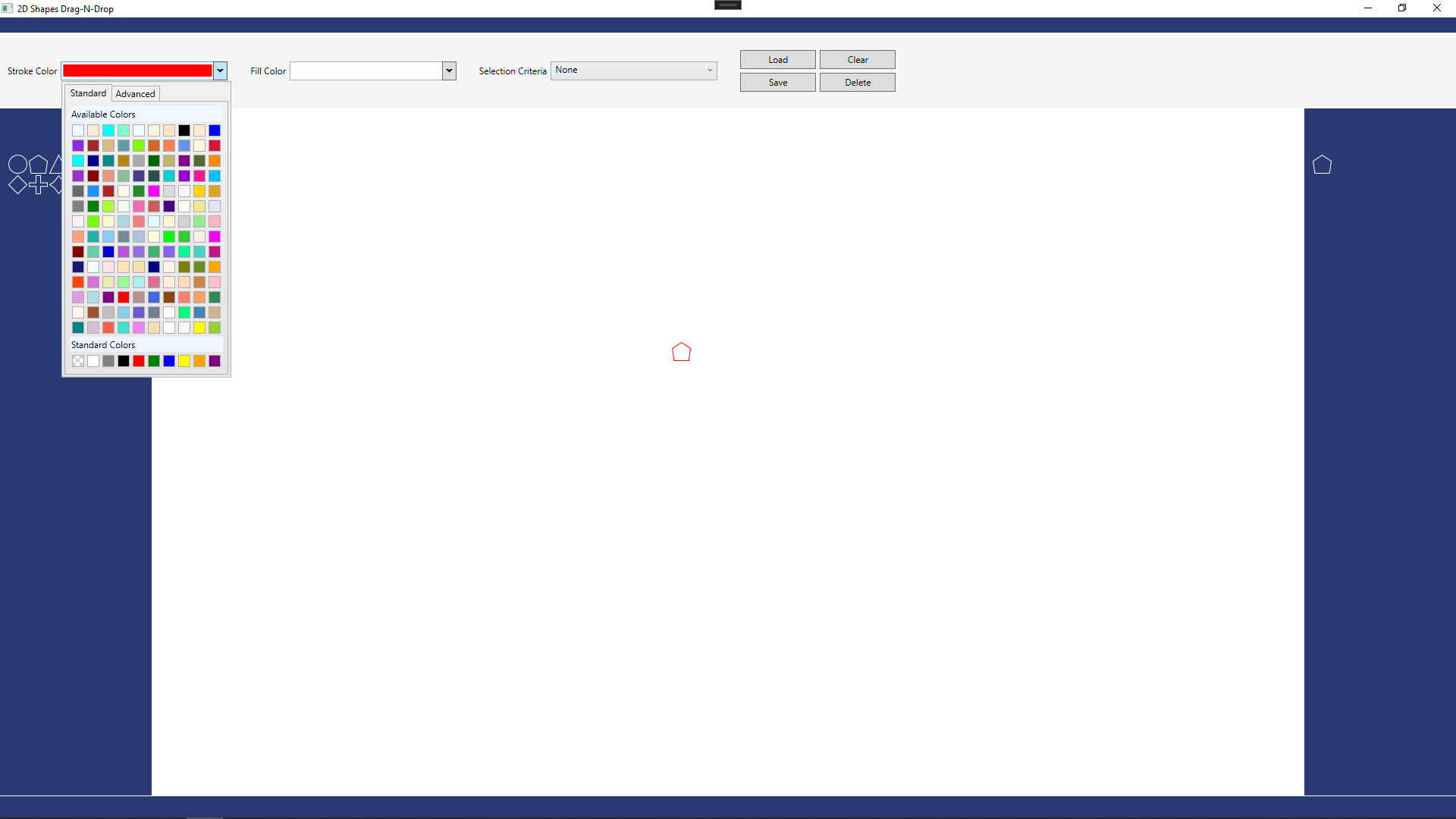
Graphical user interface, application

Description automatically generated

*Figure 3: Color Highlighting a Selected Shape(s)*

STROKE COLOR

The *Stroke Color* tool on the toolbar was used to change the border color of the selected item on the drawing canvas.



*Figure 4: Stroke Color Palette*

FILL COLOR

The *Fill Color* tool on the toolbar as shown below was used to change the fill color of the selected item(s) on the drawing canvas.

Graphical user interface, application

Description automatically generated *Figure 5: Fill Color Palette*

ITEMS PANE

The *Items Pane* displays a list of shapes. A shape is only shown if it is present on the drawing canvas. Each shape was only displayed once on the *Items Pane* irrespective of how many times it appears on the drawing canvas.

Graphical user interface

Description automatically generated *Figure 6: Distinct Shapes on the Drawing Canvas*

SELECT FIRST ITEM

A user can select several items on the drawing canvas by selecting the appropriate shape on the *Items Pane* and selecting an option from the *Selection Criteria* tool on the toolbar as shown in Figure below. Select first item: When this option is active, the first item of the selected shape that was dropped on the drawing canvas is selected.

Graphical user interface, application, Word

Description automatically generated

*Figure 7: Select First Item Criterion*

SELECT LAST ITEM

When this option is active, the last item of the selected shape that was dropped on the drawing canvas is selected

Graphical user interface, application, Word

Description automatically generated *Figure 8: Select Last Item Criterion*

MIDDLE ITEM

When this option is active, the middle item of the selected shape on the drawing canvas is selected

If the total number of the selected shape is even, two shapes should be selected. If the total number of the selected shape is odd, one shape will be selected

Graphical user interface, application, Word

Description automatically generated

*Figure 9: Select Middle Item Criterion*

SELECT ALL ITEMS

When this option is active, all items of the selected shape on the drawing canvas are selected

Graphical user interface, application, Word

Description automatically generated

*Figure 10: Select All Items Criterion*

SAVE ITEMS

A user can save the drawing by clicking the ‘Save’ button. When this happens, the current drawing is saved to a sqlite database.

Graphical user interface

Description automatically generated

*Figure 11: Save items on the Drawing Canvas to Sqlite Database*

LOAD

A user can load a saved drawing by clicking the “Load”button in order to populate the drawing Area with all the saved shapes in the SQLite Database.

Chart, scatter chart

Description automatically generated

*Figure 12: loaded items on the Drawing Canvas from Sqlite Database*

NB: This is the link to the codebase <https://github.com/worinium/DragAndDropTwoDimensionalShapes>

In conclusion, the use of MVVM, PRISM, and SQLite in this project provides a well-structured and maintainable codebase, making it easier to implement the drag-and-drop feature for 2D shapes in WPF.