Windows Presentation Foundation

For this project, the team used Windows Presentation Foundation (WPF) to create a desktop application for Chappy. For practical purposes, this program is not needed for the platform, at least not in the scope of what was made as no one would use a service such as this on a desktop computer, but it gives the team a good understanding in dealing with this technology. In addition, it gives the team another application to test the different layers of the platform.

WPF is a system developed by Microsoft, which uses Extensible Application Markup Language (XAML) to build the GUI elements and components. Compared to WinForms, its predecessor, created GUI is much different. Instead of dragging and dropping GUI elements onto a blank canvas, the developer writes the XAML to create them. At first glance, one might think of this as a step backwards, but the way it works, it is actually much easier to create applications, even by typing.

To Display an example, below is the majority of the code in the main login screen for the application.

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="130"></RowDefinition>

<RowDefinition Height="80"></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Image

Source="pack://application:,,,/Resources/chappy\_logo\_draft.png"

Grid.Row="0"

Grid.Column="1"/>

<Label

Content="Email Address:"

Grid.Row="1"

Grid.Column="0"

HorizontalAlignment="Right"

VerticalAlignment="Bottom"

FontSize="20"/>

<Label

Content="Name:"

Grid.Row="1"

Grid.Column="0"

HorizontalAlignment="Right"

VerticalAlignment="Top"

FontSize="20"/>

<TextBox

x:Name="txtEmailAddress"

Grid.Row="1"

Grid.Column="1"

HorizontalAlignment="Left"

Height="30"

VerticalAlignment="Bottom"

Width="170"

FontSize="16"/>

<Label

x:Name="lblError"

Grid.Row="1"

Grid.Column="2"

VerticalAlignment="Bottom"

HorizontalAlignment="Left"

Foreground="Red"

Content="Invalid Email Address"/>

<TextBox

x:Name="txtName"

Grid.Row="1"

Grid.Column="1"

HorizontalAlignment="Left"

Height="30"

VerticalAlignment="Top"

Width="170"

FontSize="16"/>

<StackPanel

Grid.Row="3"

Grid.Column="1"

Orientation="Horizontal">

<Button

x:Name="btnRegister"

Content="Register"

Height="30"

Width="80" Click="btnRegister\_Click"/>

<Button

x:Name="btnLogin"

Content="Login"

Height="30"

Width="80"

Margin="7, 0, 0 0" Click="btnLogin\_Click"/>

</StackPanel>

<WPFSpark:ToggleSwitch

x:Name="toggle"

Grid.Row="2"

Grid.Column="1"

Height="30"

Width="175"

CheckedText="Client"

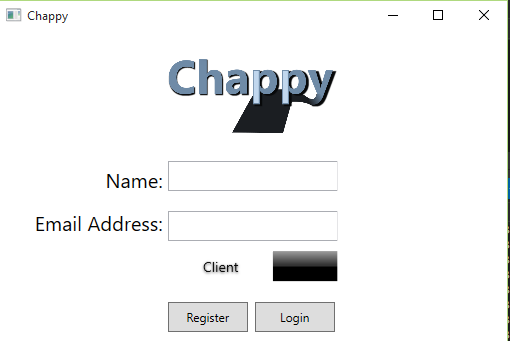
UncheckedText="Chaperone"

IsChecked="True"

FontSize="14px"></WPFSpark:ToggleSwitch>

</Grid>

As you can likely see, each GUI element is tagged by brackets (< and ending with />). Each element contains many different properties, most of which aren’t even type out as there are default values for every control. An additional advantage to using WPF, which unfortunately isn’t demonstrated in this example, is the ability to bind an element with an object. This makes for easy saving and retrieval.



Above, is a picture of the login screen that the XAML example created. There are a few other things needed, for instance the image was also stored as a resource, and the switch control was downloaded as a Nuget called WPFSpark.

The program is fully functional, in that it will call all the necessary methods of our service to update and create records in the database.

For testing purposes, we launched the program on 2 separate laptops. One laptop played the role of a Client, while the other played the role as a Chaperone. The client requested a chaperone, and the clients name popped up on the screen of the chaperone. When the chaperone clicked on the button to accept the client, the clients screen updated and said the chaperone was on the way.