

Документация за извършените промени по тема

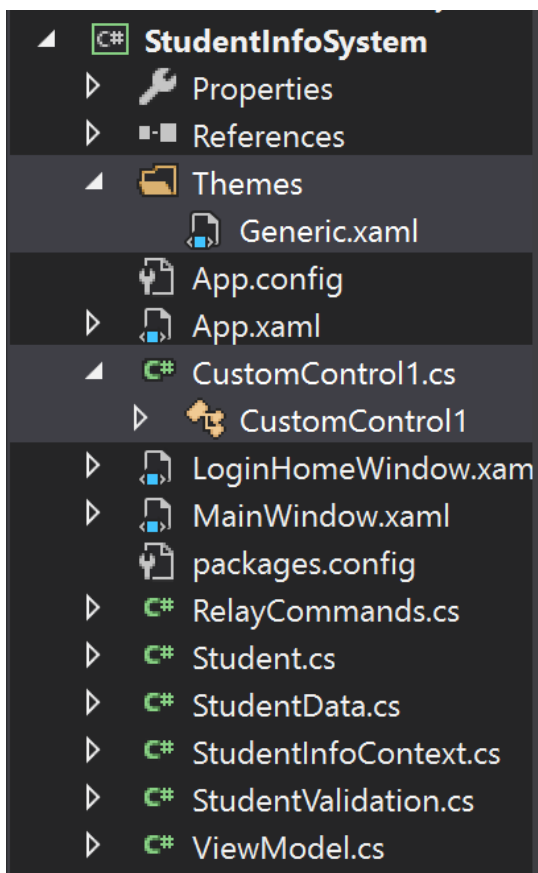
*“Реализация и използване на
собствена контрола чрез
наследяване на клас на контрола
от WPF”*

Разработено от Христо Николаев Илиев,
38 група, ФN°121217091

1. Създаване на собствена контрола

Създаване собствената Custom Control от десен бутон върху името на проекта -> Add new item -> Custom Control (WPF)

По този начин се създават два файла Generic.xaml и .cs файл



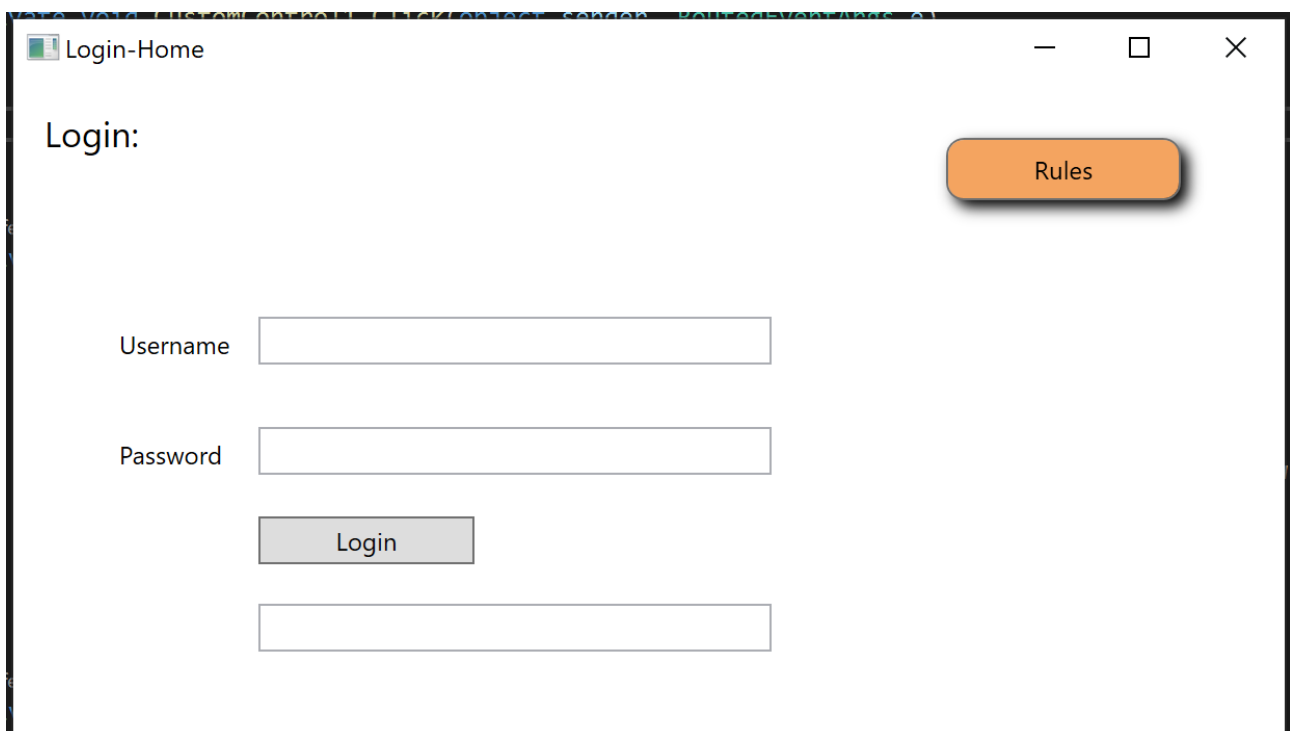
2. Реализация на собствената контрола. Идея и код

Създадената от мен контрола представлява бутон, с различен вид анимация от вградените, който изпълнява няколко функции. Една от функциите е, когато курсорът на мишката попадне върху бутона, без да го натиска, се визуализира информация. В случая информацията е списък от правила, които дават яснота как трябва да бъдат изписани потребителското име и парола за вход в системата. Втората функционалност е, когато курсорът на мишката напусне очертанията на бутона, информацията, визуализирана на екрана, се скрива. По този начин добавяме още една функционалност на бутона без той да бъде натиснат, което е ни дава предимството, когато натиснем бутона той да изведе друга информация. Имено това е и следващата му функционалност - при натискане се визуализира съобщение, че бутонът е натиснат, което съобщение може да бъде заменено с

каквато и да е друга информация. Например информация за това кой е разработчикът на приложението, какво точно представлява то, за какво служи.

В следващите снимки ще покажа нагледно как точно работи бутонът и след това програмният код.

При стартиране на програмата, началният прозорец изглежда така:



Бутонът Rules е нашият Custom Control Button.

При попадане на курсора на мишката в очертанията на бутона се визуализира информацията.

Login-Home

Login:

Rules

Username

Password

Login

For successful login:

1. Username must start with uppercase
1. Username must be 5 characters at least
2. Password must be 5 characters at least

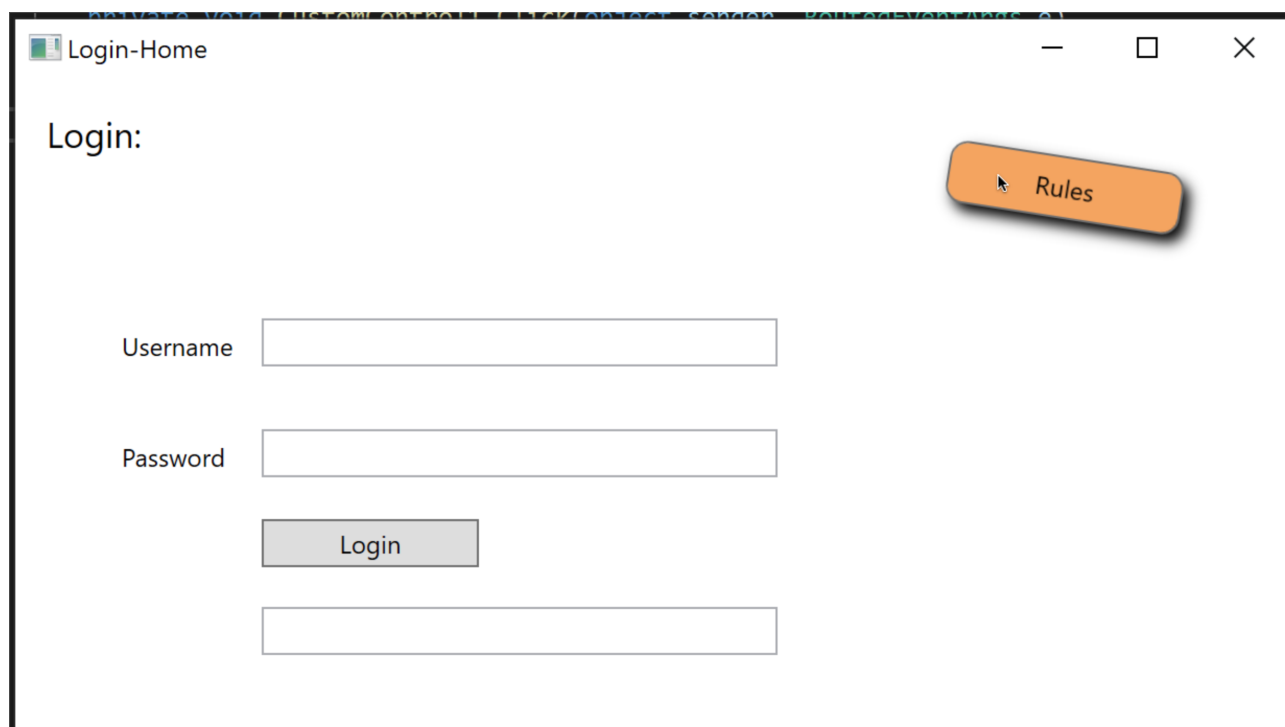
Valid user data:

Username: Hristo

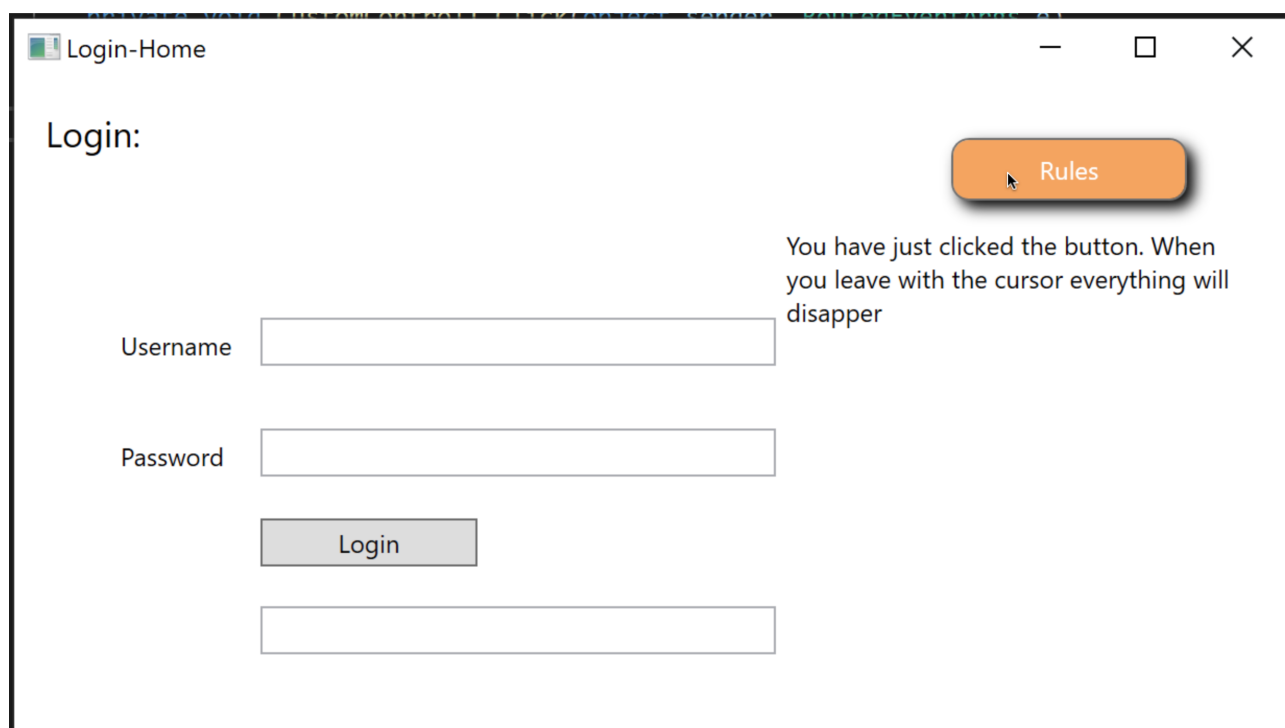
Password : 12345678

Виждаме как курсорът е върху бутона(без да го натиска) и на екрана са изписани правилата за вход в системата както и примерни потребителско име и парола за успешен вход. Ако курсорът на мишката излезе от очертанията на бутона информацията ще се скрие веднага и екранът ще изглежда по същия начин както в началото(при стартиране на програмата). Виждаме също, че когато курсорът на мишката е върху бутона цветът на надписа Rules се променя от черен на бял.

Натискане на бутона



При натискане на бутона се задейства анимация, която кара десния му край да падне под ъгъл с определени градуси за определено време, които се задават в `Generic.xaml` кода. След което бутонът се връща в изходната си позиция и се визуализира информацията, зададена при тази функционалност.



Информацията, която се изписва на екрана казва, че бутонът е натиснат и когато курсорът напусне очертанията му, информацията ще изчезне и екранът отново ще изглежда по същия начин както в началото(при стартиране на програмата).

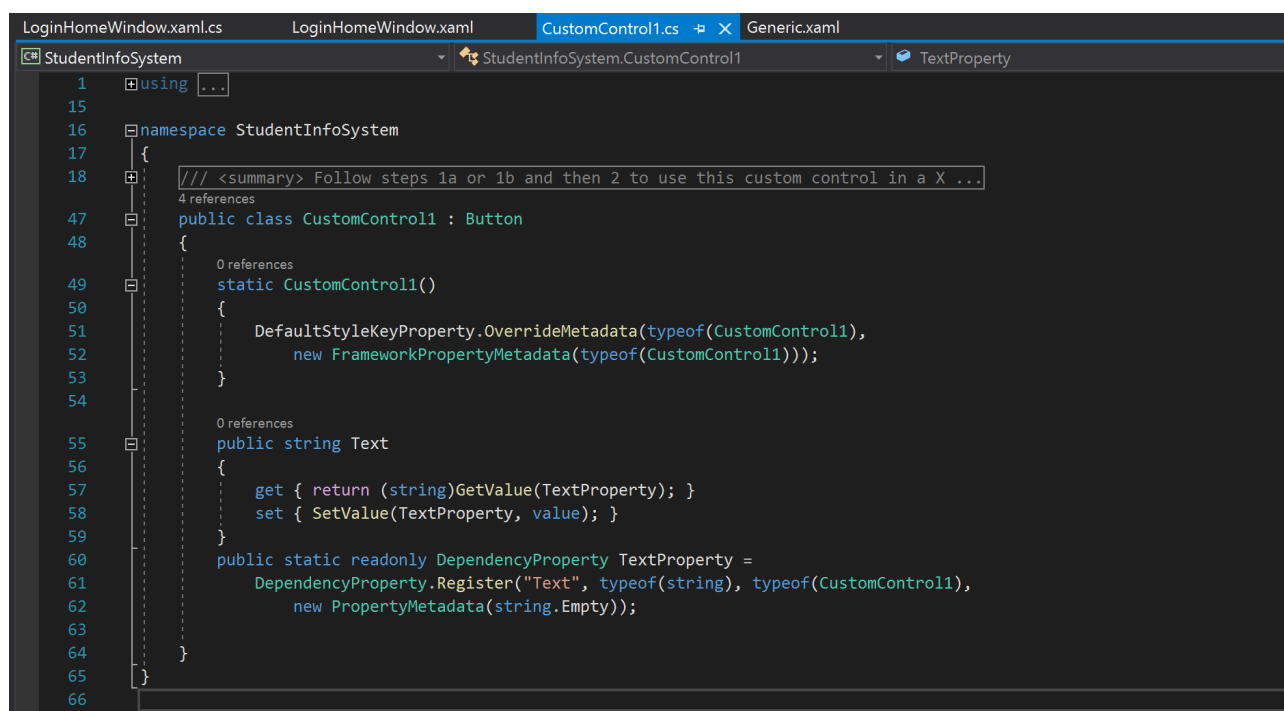
3. Програмен код

Generic.xaml

```
LoginHomeWindow.xaml.cs LoginHomeWindow.xaml CustomControl1.cs Generic.xaml packages.config
ResourceDictionary
1  ResourceDictionary
2  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4  xmlns:local="clr-namespace:StudentInfoSystem"
5  xmlns:MyNamespace="clr-namespace:StudentInfoSystem"
6
7  <Style TargetType="{x:Type MyNamespace:CustomControl1}" BasedOn="{StaticResource {x:Type Button}}">
8
9      <Setter Property="Template">
10         <Setter.Value>
11             <ControlTemplate TargetType="{x:Type MyNamespace:CustomControl1}">
12                 <Border Background="SandyBrown"
13                     BorderBrush="{TemplateBinding BorderBrush}"
14                     BorderThickness="{TemplateBinding BorderThickness}"
15                     CornerRadius="8"
16                     Cursor="Help">
17                     <Border.Effect>
18                         <DropShadowEffect BlurRadius="10" />
19                     </Border.Effect>
20
21                     <ContentPresenter>
22                         <ContentPresenter.Content>
23                             <TextBlock x:Name="Text"
24                                 Text="{TemplateBinding Text}"
25                                 VerticalAlignment="Center"
26                                 HorizontalAlignment="Center"/>
27                         </ContentPresenter.Content>
28                     </ContentPresenter>
29
30                     <Border.RenderTransform>
31                         <RotateTransform x:Name="RTR" />
32                     </Border.RenderTransform>
33                 </Border>
34
35                 <ControlTemplate.Triggers>
36                     <Trigger Property="IsMouseOver" Value="true">
37                         <Setter TargetName="Text" Property="Foreground" Value="white"/>
38                     </Trigger>
39
40                     <EventTrigger RoutedEvent="Button.Click">
41                         <BeginStoryboard>
42                             <Storyboard FillBehavior="HoldEnd">
43                                 <DoubleAnimation Storyboard.TargetName="RTR"
44                                     Storyboard.TargetProperty="Angle"
45                                     From="0"
46                                     To="10"
47                                     Duration="0:0:0.5"
48                                     AutoReverse="True">
49
50                                 </DoubleAnimation>
51                             </Storyboard>
52                         </BeginStoryboard>
53                     </EventTrigger>
54                 </ControlTemplate.Triggers>
55             </ControlTemplate>
56         </Setter.Value>
57     </Setter>
58 </Style>
59 </ResourceDictionary>
60
```

Generic.xaml представлява xaml код, който задава стилът на контролата. Можем да му зададем Background, BorderBrush, BorderThickness, CornerRadius, Cursor, ефект на сянка на самия бутон ShadowEffect. След това задаваме, че в бутонът ще има TextBlock, който ще визуализира зададен от нас текст в него, както и настройки как точно да бъде позициониран текстът чрез Vertical и HorizontalAlignment. Следващите редове от кода задават property trigger и event trigger. Първият вид “наблюдава” кога зададеното property(в случая IsMouseOver) ще приеме стойност true и променя цветът на текста от черен на бял. Вторият вид(event trigger) следи за настъпило събитие, в случая когато бутонът е натиснат, и задава анимацията, карайки бутонът да се отклонява на определени градуси, за определено време и след това да се връща.

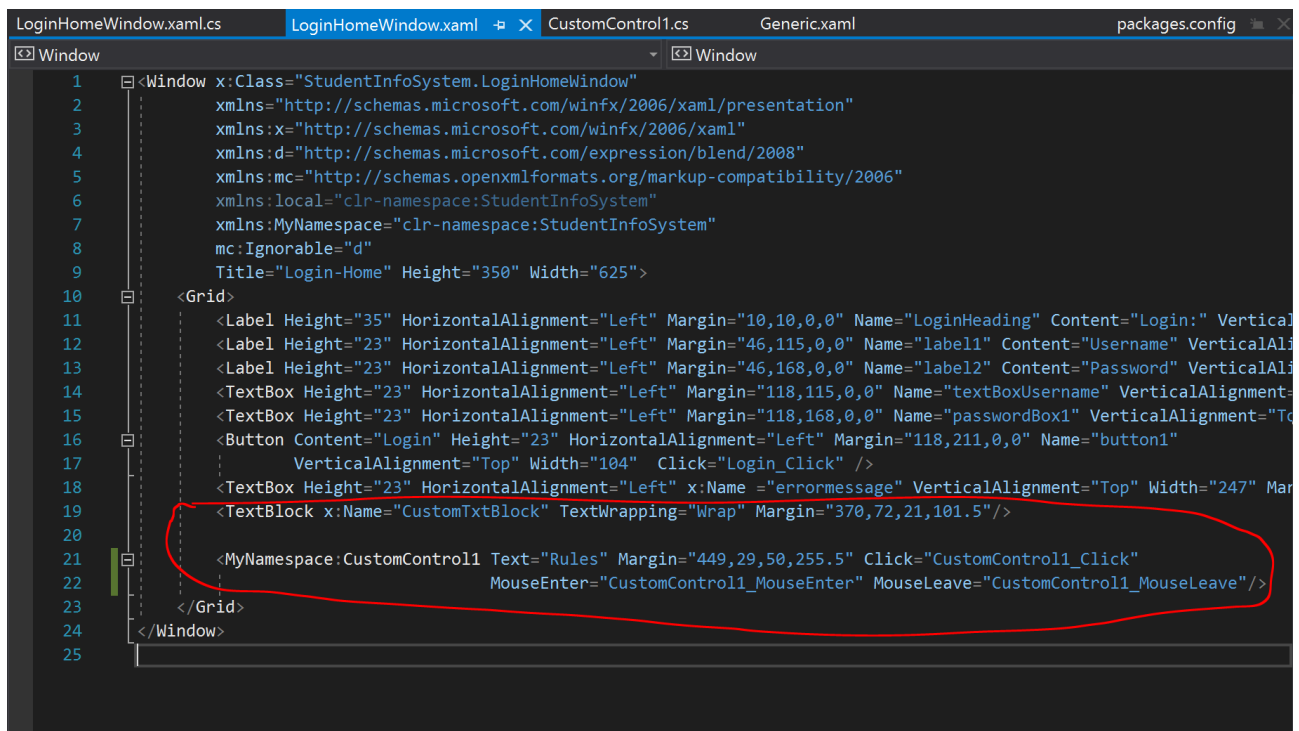
CustomControl1.cs



```
1  using ...
15
16  namespace StudentInfoSystem
17  {
18      /// <summary> Follow steps 1a or 1b and then 2 to use this custom control in a X ...
19      4 references
20      public class CustomControl1 : Button
21      {
22          0 references
23          static CustomControl1()
24          {
25              DefaultStyleKeyProperty.OverrideMetadata(typeof(CustomControl1),
26                  new FrameworkPropertyMetadata(typeof(CustomControl1)));
27          }
28
29          0 references
30          public string Text
31          {
32              get { return (string)GetValue(TextProperty); }
33              set { SetValue(TextProperty, value); }
34          }
35
36          public static readonly DependencyProperty TextProperty =
37              DependencyProperty.Register("Text", typeof(string), typeof(CustomControl1),
38                  new PropertyMetadata(string.Empty));
39      }
40  }
```

CustomControl1.cs представлява C# кодът на собствената контрола, който наследява базовият клас Button и съответно в конструктура overrides the metadata. Направил съм и едно property Text както и Custom Dependency Property, за да може да се зададе текстът на собствената контрола.

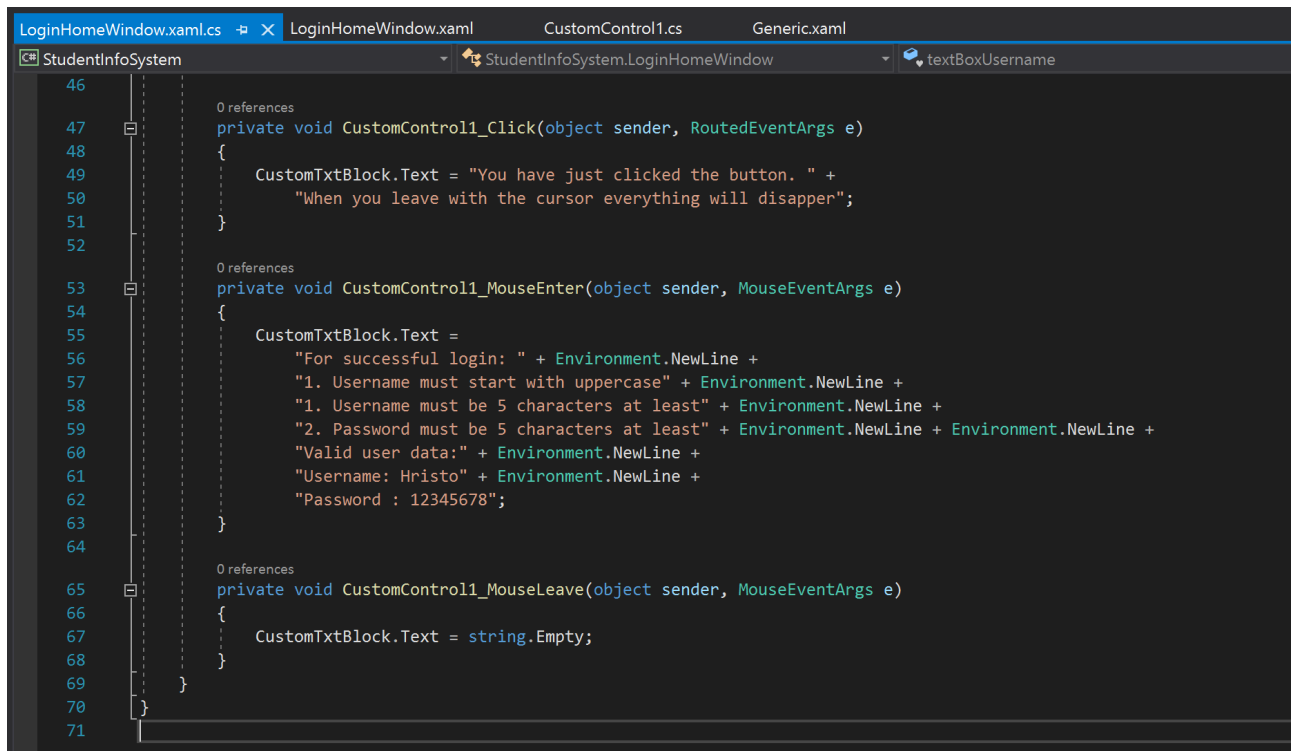
LoginHome.xaml



```
1 <Window x:Class="StudentInfoSystem.LoginHomewindow"
2       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4       xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5       xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6       xmlns:local="clr-namespace:StudentInfoSystem"
7       xmlns:MyNamespace="clr-namespace:StudentInfoSystem"
8       mc:Ignorable="d"
9       Title="Login-Home" Height="350" Width="625">
10
11     <Grid>
12         <Label Height="35" HorizontalAlignment="Left" Margin="10,10,0,0" Name="LoginHeading" Content="Login:" VerticalAl
13         <Label Height="23" HorizontalAlignment="Left" Margin="46,115,0,0" Name="label1" Content="Username" VerticalAl
14         <Label Height="23" HorizontalAlignment="Left" Margin="46,168,0,0" Name="label2" Content="Password" VerticalAl
15         <TextBox Height="23" HorizontalAlignment="Left" Margin="118,115,0,0" Name="textBoxUsername" VerticalAlignment=
16         <TextBox Height="23" HorizontalAlignment="Left" Margin="118,168,0,0" Name="passwordBox1" VerticalAlignment="To
17         <Button Content="Login" Height="23" HorizontalAlignment="Left" Margin="118,211,0,0" Name="button1"
18             VerticalAlignment="Top" Width="104" Click="Login_Click" />
19         <TextBox Height="23" HorizontalAlignment="Left" x:Name="errorMessage" VerticalAlignment="Top" Width="247" Mar
20         <TextBlock x:Name="CustomTxtBlock" TextWrapping="Wrap" Margin="370,72,21,101.5"/>
21
22         <MyNamespace:CustomControl1 Text="Rules" Margin="449,29,50,255.5" Click="CustomControl1_Click"
23             MouseEnter="CustomControl1_MouseEnter" MouseLeave="CustomControl1_MouseLeave"/>
24     </Grid>
25 </Window>
```

В LoginHome.xaml файла създаваме въпросната собствена контрола както и един TextBlock, в който да се визуализира информацията от бутона. Тук задаваме какъв ще бъде текстът на CustomControl1, както и Click, MouseEnter и MouseLeave events, които ще бъдат описани в code-behind.

LoginHome.cs



```
46
47 0 references
48 private void CustomControl1_Click(object sender, RoutedEventArgs e)
49 {
50     CustomTxtBlock.Text = "You have just clicked the button. " +
51         "When you leave with the cursor everything will disapper";
52 }
53
54 0 references
55 private void CustomControl1_MouseEnter(object sender, MouseEventArgs e)
56 {
57     CustomTxtBlock.Text =
58         "For successful login: " + Environment.NewLine +
59         "1. Username must start with uppercase" + Environment.NewLine +
60         "1. Username must be 5 characters at least" + Environment.NewLine +
61         "2. Password must be 5 characters at least" + Environment.NewLine + Environment.NewLine +
62         "Valid user data:" + Environment.NewLine +
63         "Username: Hristo" + Environment.NewLine +
64         "Password : 12345678";
65 }
66
67 0 references
68 private void CustomControl1_MouseLeave(object sender, MouseEventArgs e)
69 {
70     CustomTxtBlock.Text = string.Empty;
71 }
```

Първият метод е click методът на бутона, който задава текстът при натискането му. Вторият е, когато курсорът на мишката попадне в очертанията на бутона. Тогава се задават на TextBlock правилата за вход в системата. И третият е, когато курсорът излезе от очертанията на контролата. Тогава се задава string.Empty.