HND Motoring Application.

Development Handbook

Contents

Document structure 3

Development strategy overview 4

Interface design sketches 5

Test Plan Sample 11

Sample UML 12

Sample algorithm flowchart 13

Entities and objects to be implemented 14

Development video list 15

Application Pages in Visual Studio design mode 16

- PageLogin 16
- PageLoan 17
- PageRunning 18

Application Running 20

Sample Test Log 24

Appendix A: XAML Code 25

- PageLogin 25
- PageLoan 28
- PageRunning 32
- PageSummary 37

Appendix B: Application code 48

- Database file 48
- Video 06 Class User 48
- Video 07 Class Loan 49
- Video 08 User Validation 50
- Video 09 Loan Calculator 51
- Video 10 Class RunningCost 53
- Video 13 Class CostSummary 65
- Video 30 Logic PageLogin 66
- Videos 31 and 32 Logic PageLoan 68
- Videos 33 and 34 Logic PageRunning 71
- Videos 35 and 36 Logic PageSummary 75

Document structure

This handbook has sections as follows:

- Strategy overview
- Interface design sketches
- Test Plan design
- UML Diagram
- Flowchart diagram
- Entities list
- Development videos list
- Implemented pages in XAML
- Proof of application running
- Test Log
- Appendix A: XAML code
- Appendix B: C# Code

Development strategy overview

The application will allow the user to:

> Calculate the costs of running a car based on the costs of a loan to buy it and the costs of running it.

The running costs will include the amounts paid for:

- Insurance
- Fuel
- Servicing
- Road Tax

The application will allow the data to be entered as weekly, monthly, or annual costs.

The application will allow the user to view the costs of ownership summarised as weekly, monthly, or annual sums.

The user will be required to login when the application launches.

The user credentials will require a username and a password. The credentials will be stored in an external file.

The application will be developed in Microsoft Visual Studio. The Windows Presentation Foundation (WPF) will be used as the structure.

The pages will be developed using eXtensible Application Markup Language (XAML) to create the page controls and their layouts. The logic will be written in C#, using the .Net 6 framework. The entities of the application will be implemented using Object Oriented Programming (OOP).

The application will run as a desktop application on Microsoft Windows™.

Interface design sketches

01 PageLogin
02 PageLoan
03 PageRunning
04 PageSummary

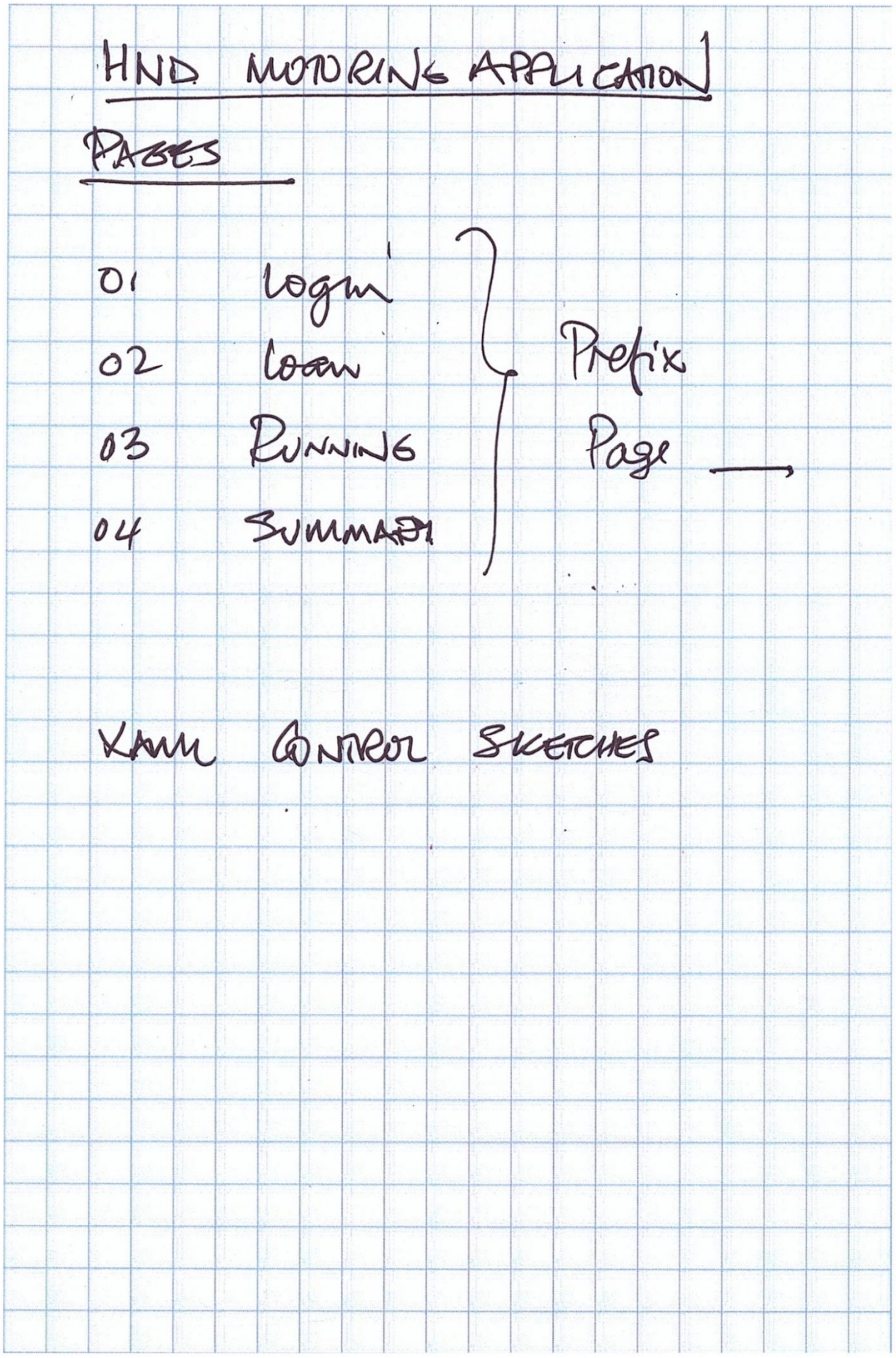

Figure 1 Ideas sketch

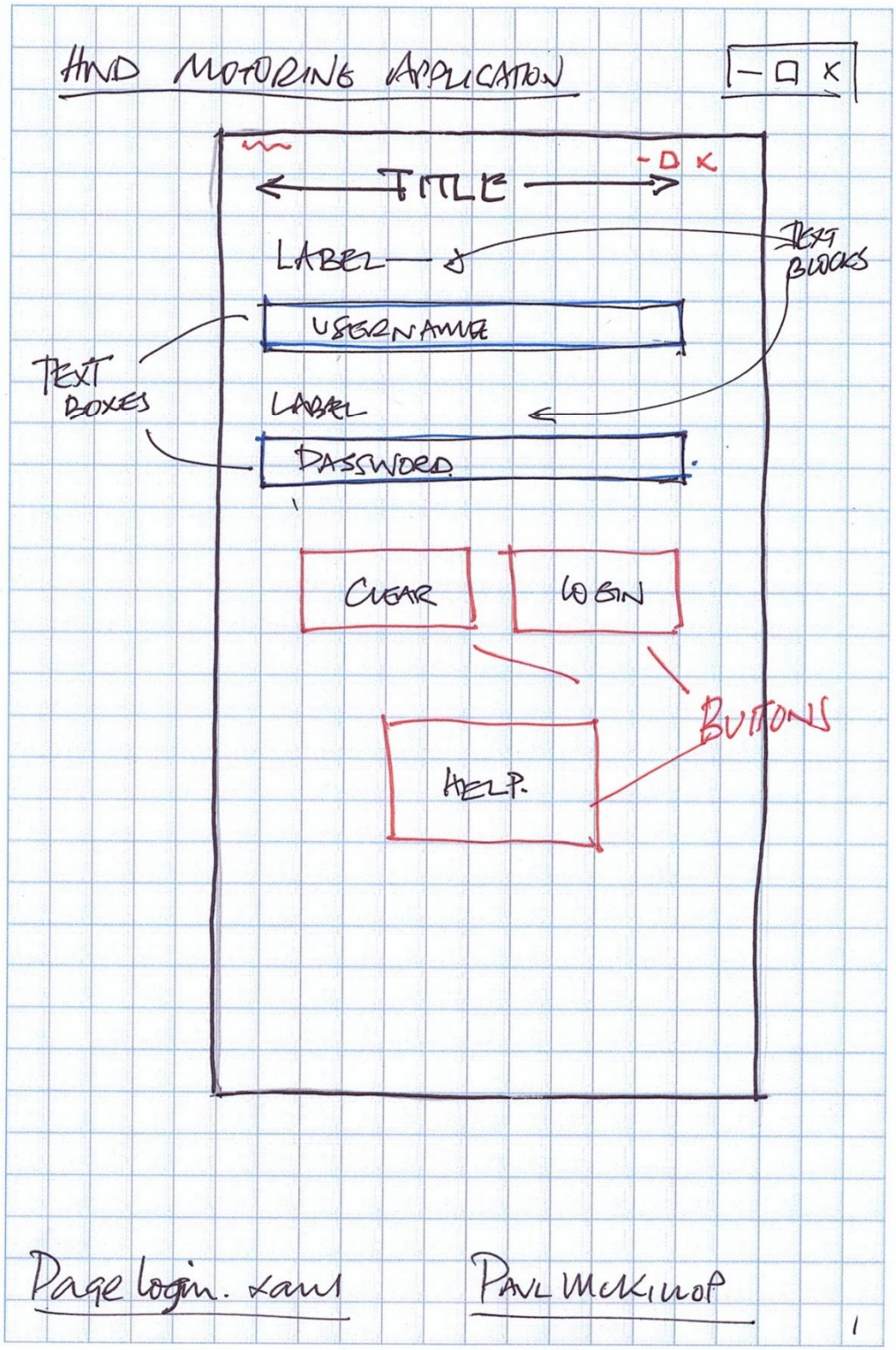

Figure 2 Login sketch

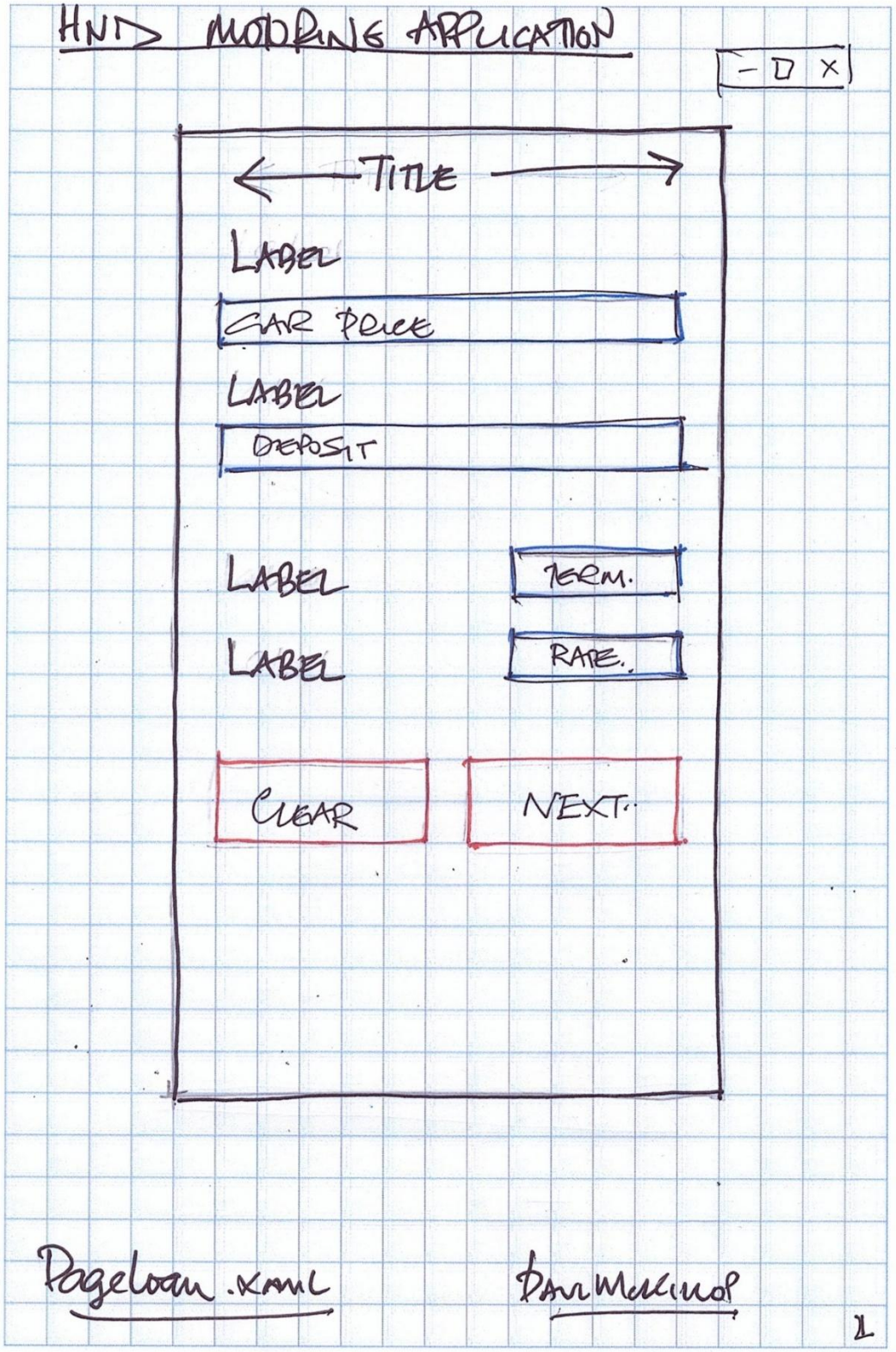

Figure 3 Loan sketch

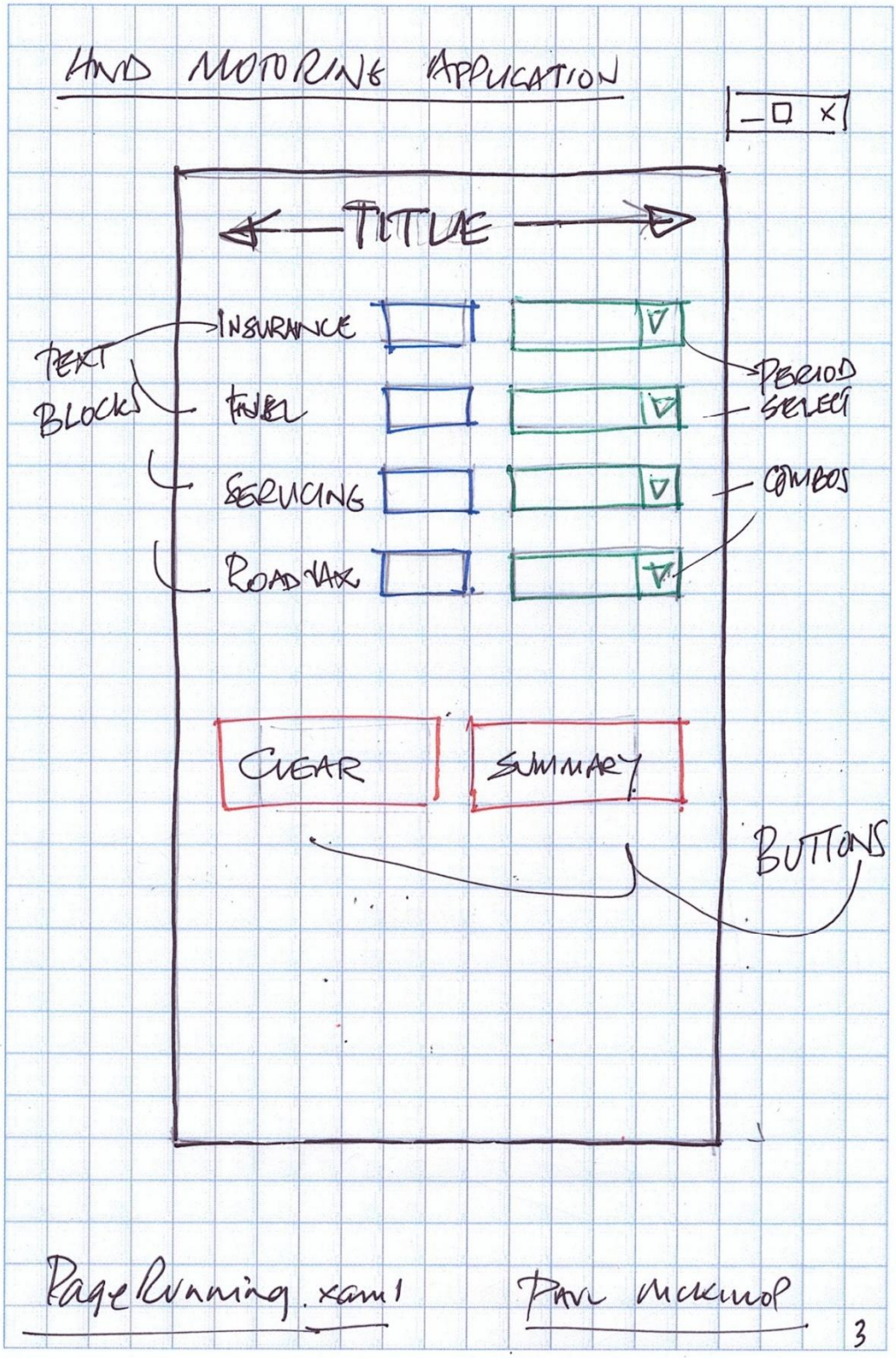

Figure 4 Running costs sketch

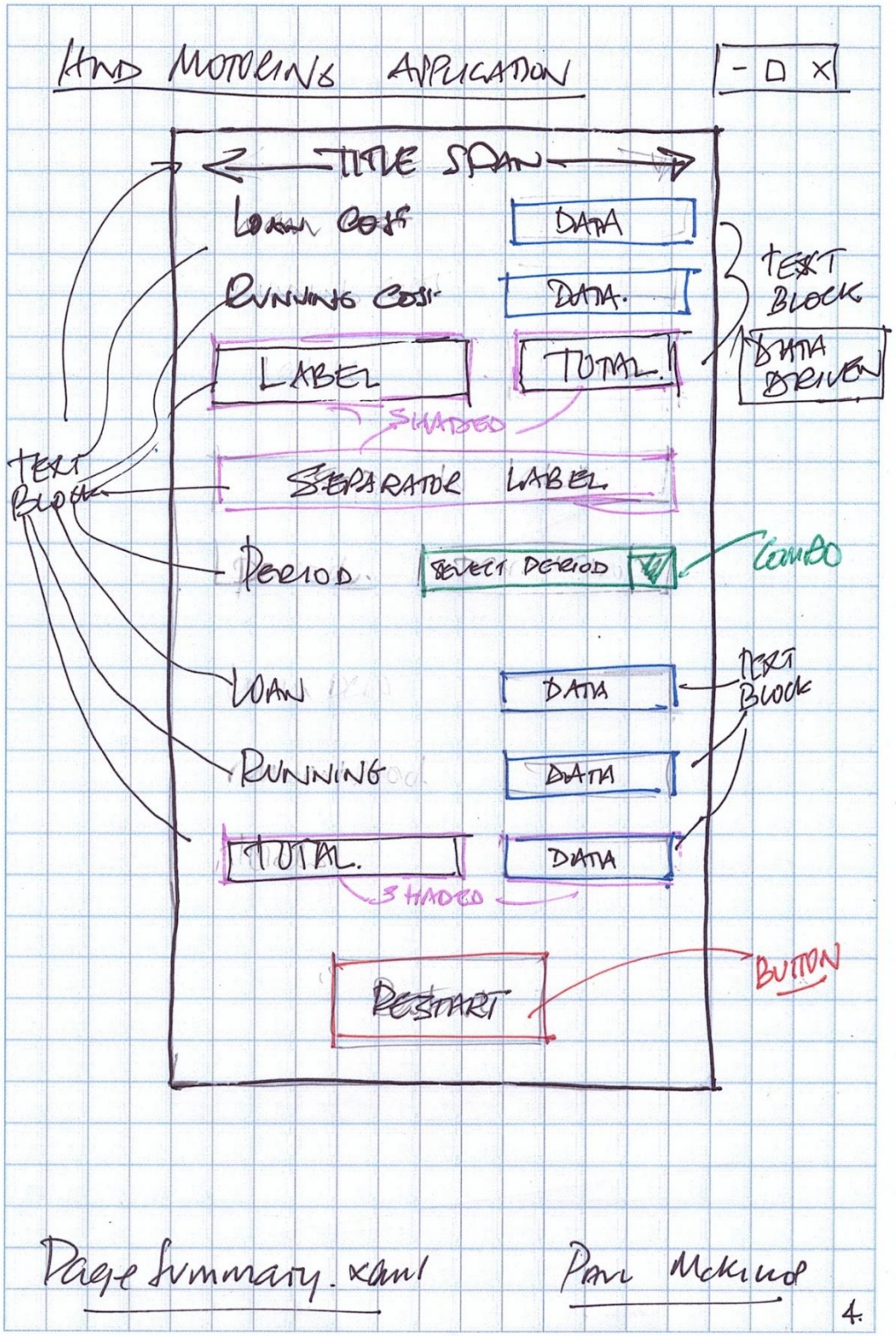

Figure 5 Data Summary sketch

Test Plan Sample

	Object	Item	Test	Method	Expected Outcome
001	PageLogin	Page	Page launches	Launch application	Page displays correctly
002	PageLogin	Form controls	Minimise, Maximise, Close	Use controls after launch	Behave as required
003	PageLogin	TextBoxes	Can enter text and format Okay	Enter text to expected sizes	Text enters to control correctly
004	PageLogin	ButtonClear	Click control clears text entries	Click control with text in boxes	Boxes are cleared
005	PageLogin	ButtonHelp	Click control displays message	Click control	Message shows
006	PageLogin	ButtonLogin	Behaviour both values empty	Click control	Error message shows
007	PageLogin	ButtonLogin	Behaviour Username empty, password OK	Click control	Error message shows
008	PageLogin	ButtonLogin	Behaviour Username OK, password empty	Click control	Error message shows
009	PageLogin	ButtonLogin	Behaviour Username OK, Password OK	Click control	Login success. PageLoan opens
010	PageLoan	Page	Page Launches	Launches from PageLogin	Opens and displays correctly
etc					

Figure 6 Sample Test Plan

Sample UML

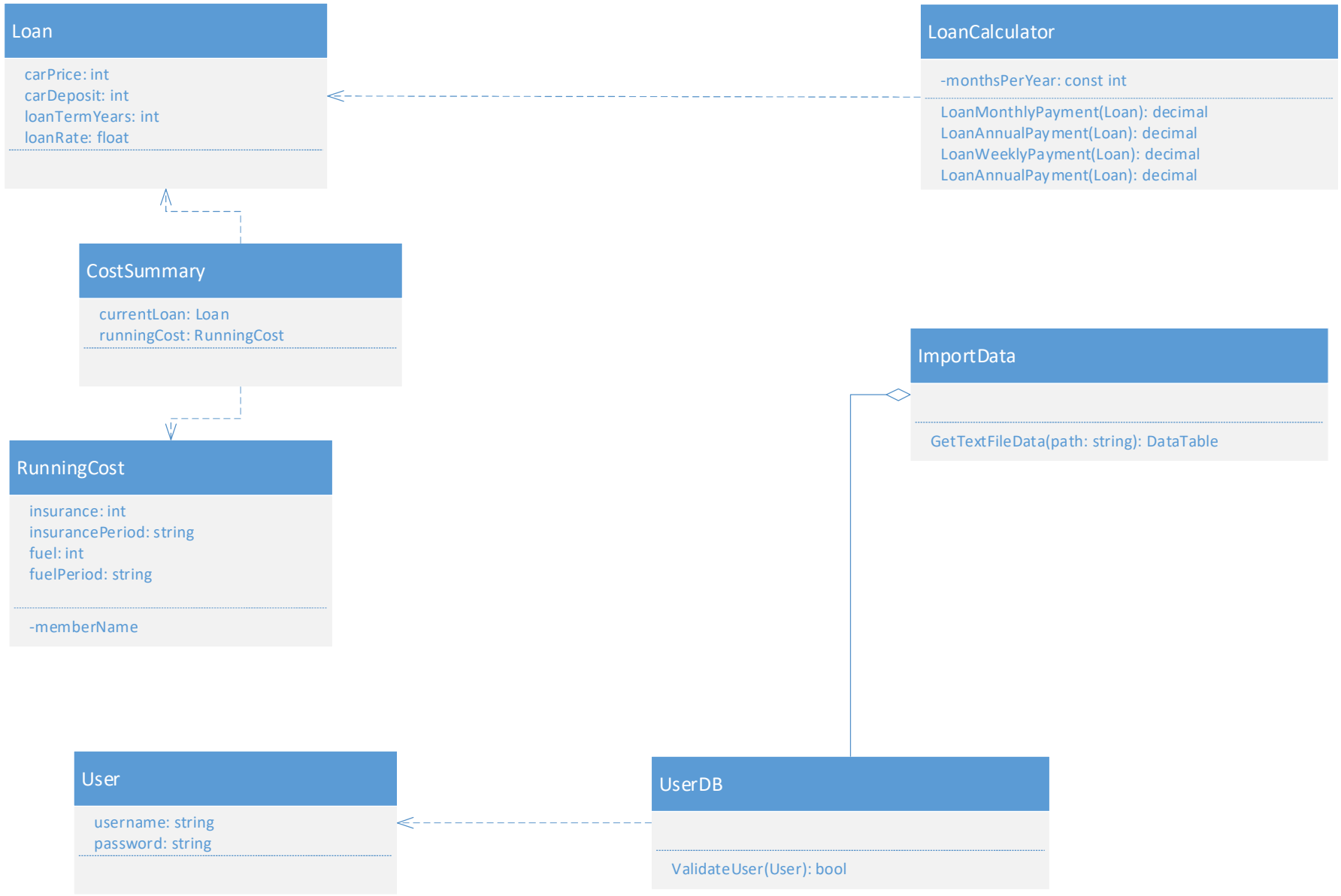

Figure 7 UML Sample

Sample algorithm flowchart

Login Process

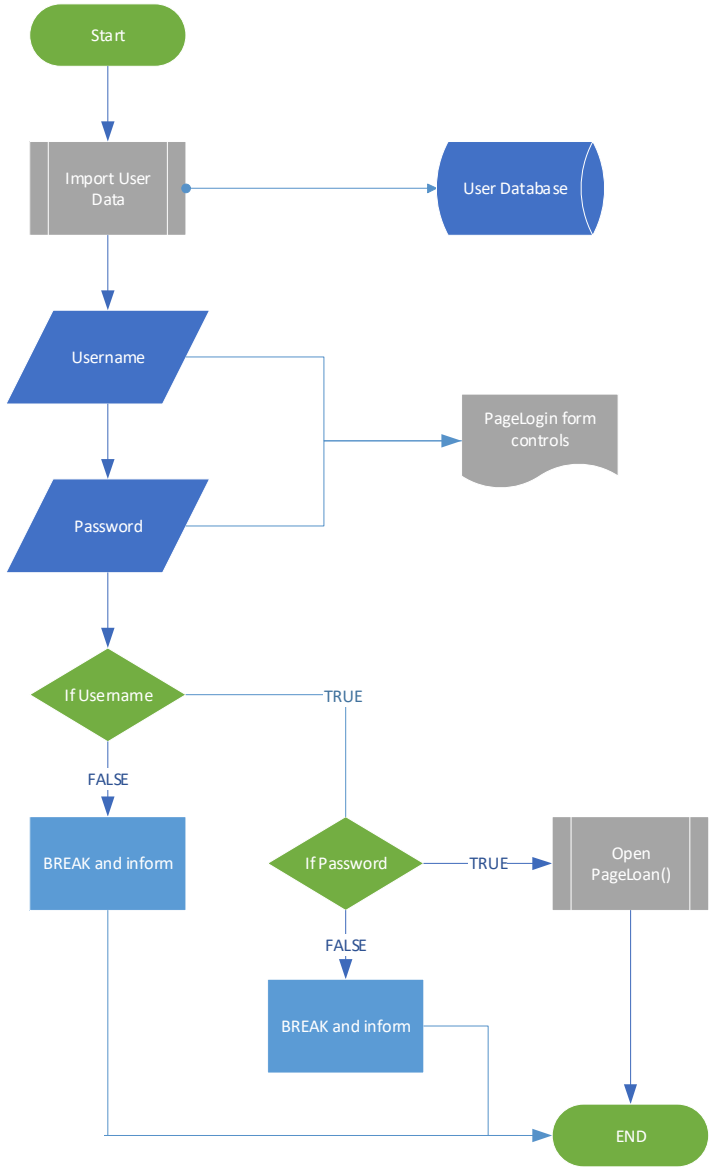

Figure 8 Flowchart of Login Algortihm

Entities and objects to be implemented

Name	Type	Rationale
Graphical User Interface (GUI) layout and logic behind		
MainWindow.xaml	XAML File	Host window for application pages
MainWindow.cs	C# File	Page logic
PageLoan.xaml	XAML File	Page layout for loan data
PageLoan.xaml.cs	C# File	Page logic
PageLogin.xaml	XAML File	Page layout for credential data
PageLogin.xaml.cs	C# File	Page logic
PageRunning.xaml	XAML File	Page layout for running costs data
PageRunning.xaml.cs	C# File	Page logic
PageSummary.xaml	XAML File	Page layout for displaying summary data
PageSummary.xaml.cs	C# File	Page logic
Application Logic		
CostSummary.cs	C# Class	Collate cost data
ImportData.cs	C# Class	Import data from a text file
Loan.cs	C# Class	Data structure for Loan
LoanCalculator.cs	C# Class	Return lost costs
RunningCost.cs	C# Class	Running costs data
RunningCostCalculator.cs	C# Class	Return costs data
User.cs	C# Class	Handle user data
UserDB.cs	C# Class	Validate the user against the external database.

Development video list

The following videos are used to develop the application. They must be completed in the correct order:

Name	Extension	Size
050 01 What we will build.mp4	mp4	24,228,180
050 02 Create the WPF Solution and Application.mp4	mp4	59,030,851
050 03 Application Pages.mp4	mp4	21,325,750
050 04 Text file database.mp4	mp4	8,693,846
050 05 Snippets and Import Data.mp4	mp4	76,215,005
050 06 Class User.mp4	mp4	31,417,450
050 07 Loan Class.mp4	mp4	23,295,005
050 08 User Validation.mp4	mp4	33,005,741
050 09 Loan Calculator.mp4	mp4	40,757,867
050 10 Class RunningCost.mp4	mp4	8,600,799
050 11 Running Costs Calculator Part 1.mp4	mp4	27,715,561
050 12 Running Costs Calculator Part 2.mp4	mp4	24,004,047
050 13 Class CostSummary.mp4	mp4	6,162,380
050 20 WPF GUI Layouts.mp4	mp4	59,802,859
050 21 Style Templates.mp4	mp4	48,132,955
050 22 PageLogin GUI Part 1.mp4	mp4	51,382,371
050 23 PageLogin GUI Part 2.mp4	mp4	62,937,690
050 24 PageLoan GUI.mp4	mp4	28,750,132
050 25 PageRunning GUI Part 1.mp4	mp4	35,973,826
050 26 PageRunning GUI Part 2.mp4	mp4	30,864,086
050 27 PageSummary GUI.mp4	mp4	27,891,515
050 30 Logic PageLogin.mp4	mp4	72,239,121
050 31 Logic PageLoan Part 1.mp4	mp4	41,851,581
050 32 Logic PageLoan Part 2.mp4	mp4	59,482,132
050 33 Logic PageRunning Part 1.mp4	mp4	49,093,324
050 34 Logic PageRunning Part 2.mp4	mp4	70,264,592
050 35 Logic PageSummary Part 1.mp4	mp4	41,834,837
050 36 Logic PageSummary Part 2.mp4	mp4	56,471,730

Figure 9 Video List

Video 01 gives and overview and sight of the completed application

Videos 02 to 05 are setup videos

Videos 06 to 13 are class implementation videos

Videos 20 to 27 are GUI implementation videos

Videos 30 to 36 are Logic implementation videos

An additional video will be published with recommendations for improving the layout and content.

Application Pages in Visual Studio design mode

PageLogin

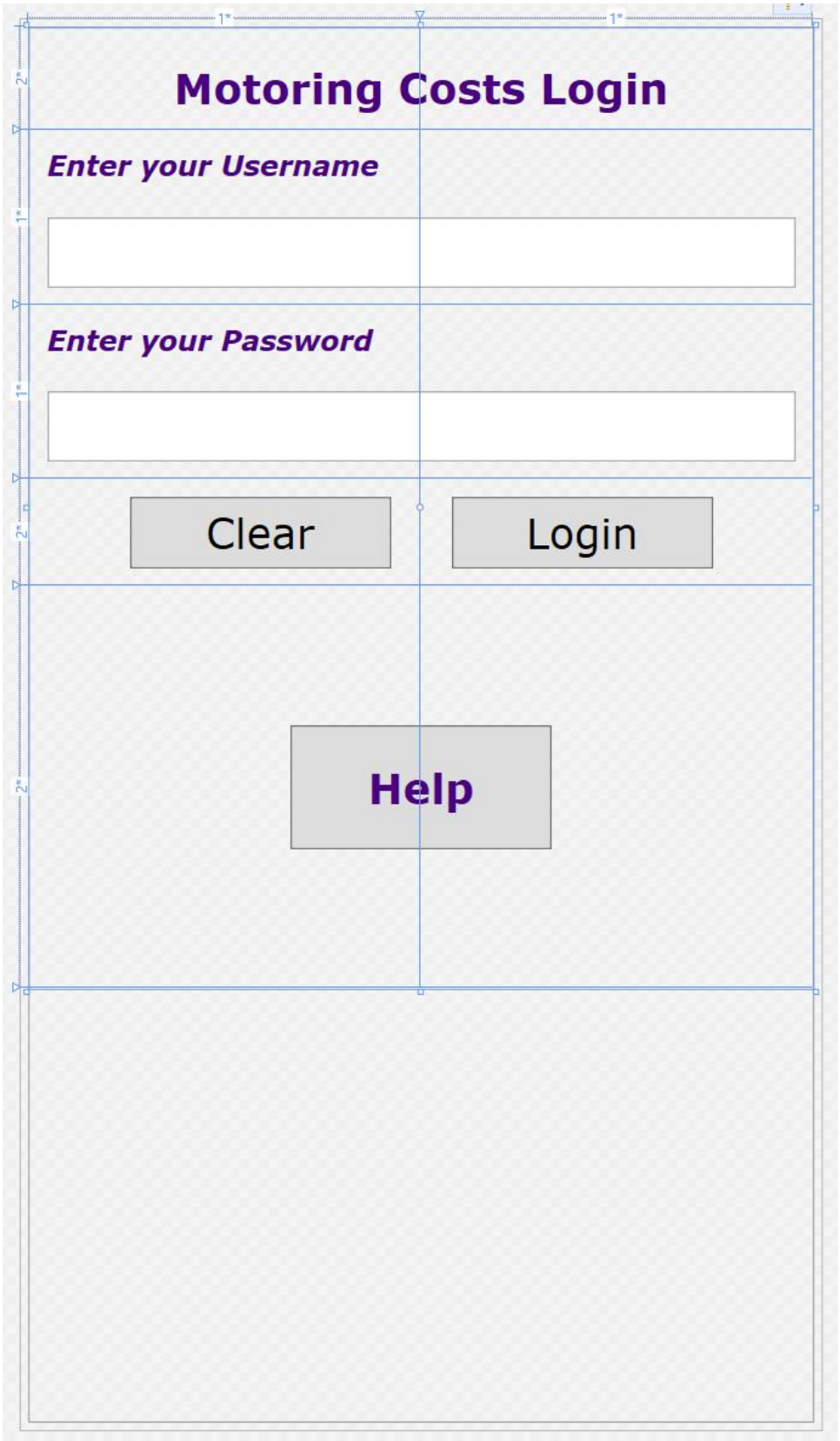

Figure 10 PageLogin Design

PageLoan

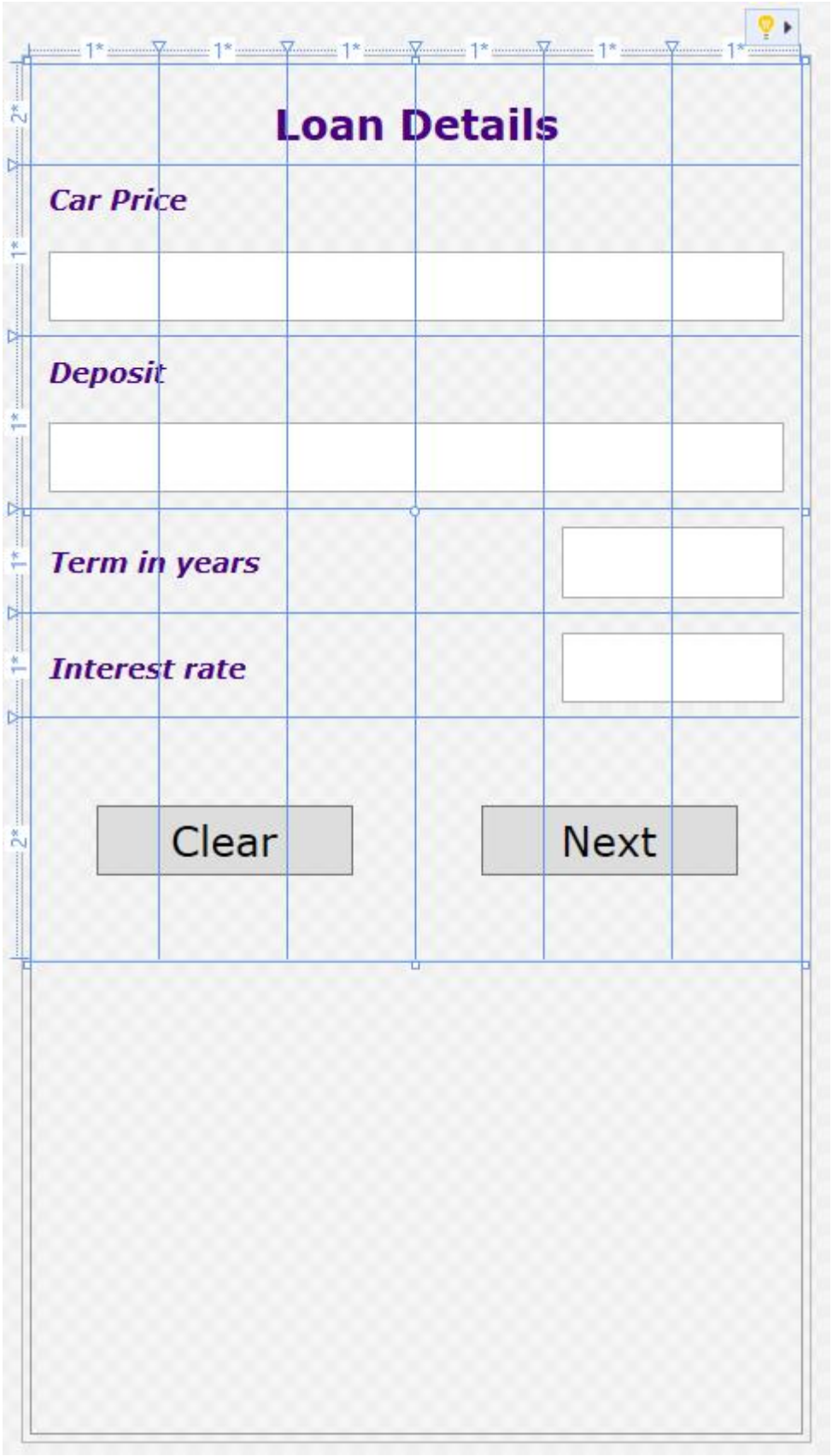

Figure 11 PageLoan Design

PageRunning

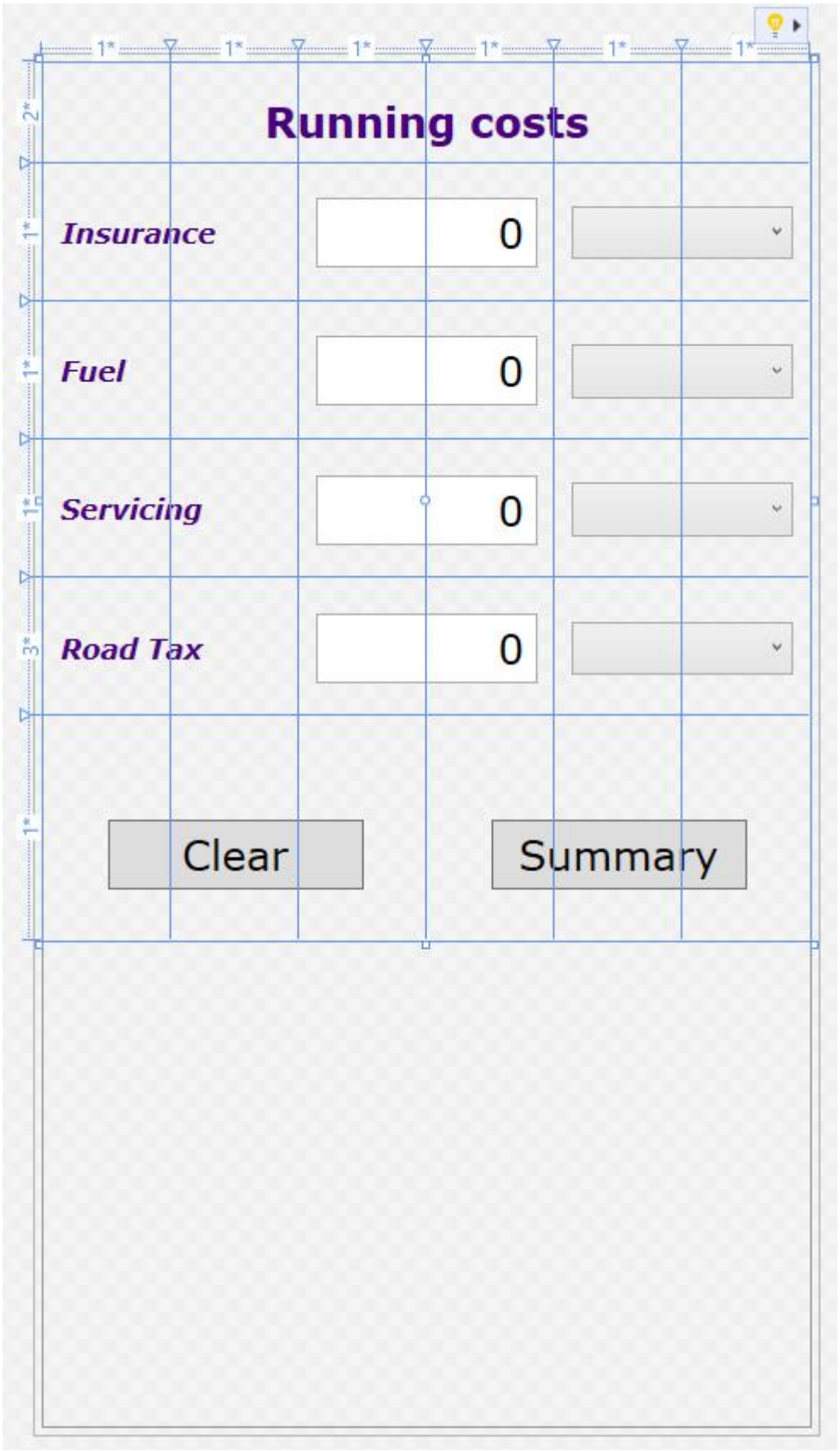

Figure 12 PageRunning Design

PageSummary

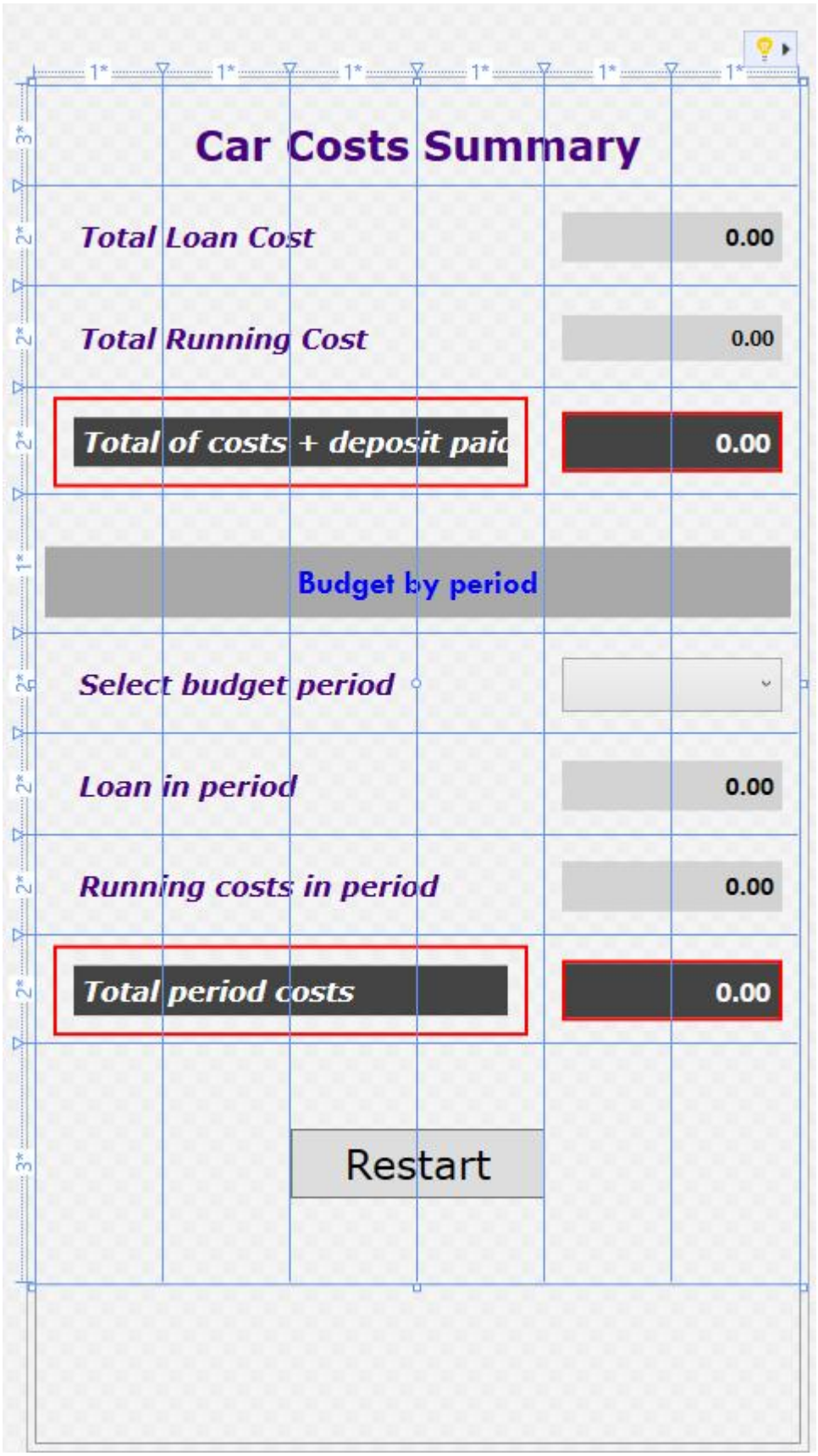

Figure 13 PageSummary Design

Application Running

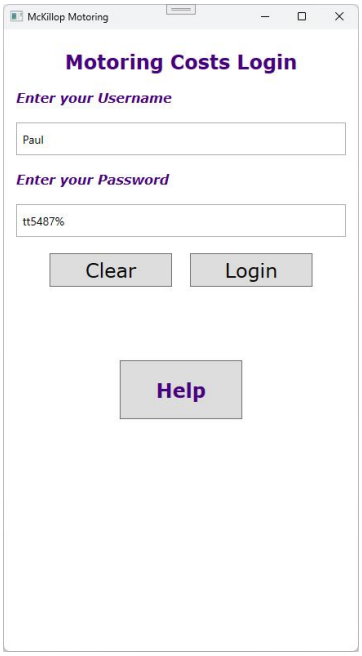

Figure 14 PageLogin

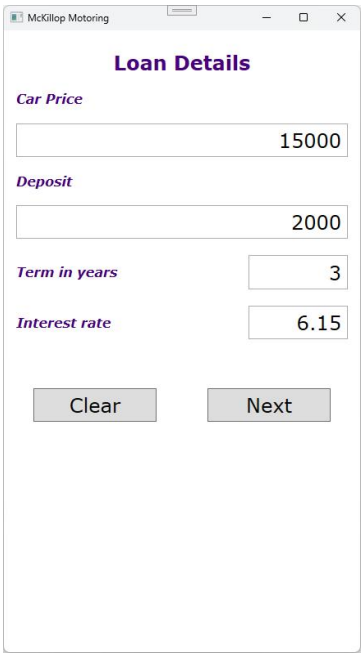

Figure 15 PageLoan

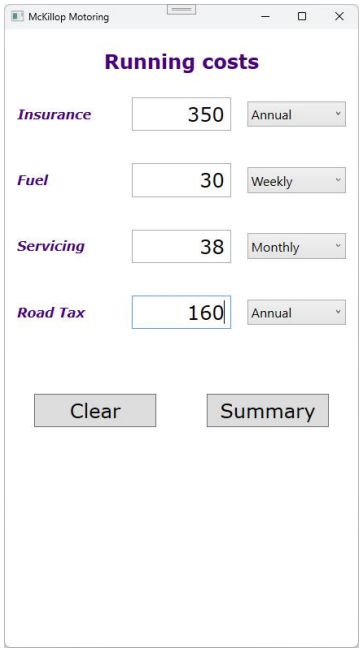

Figure 16 PageRunning

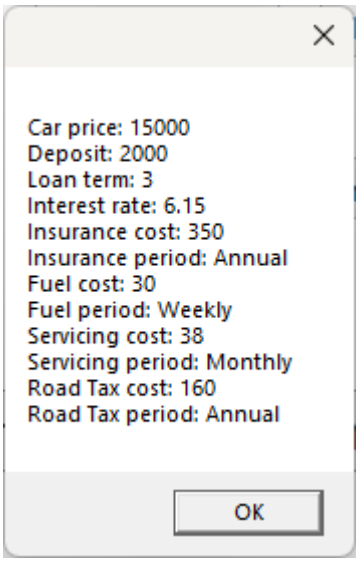

Figure 17 Data Confirmation message

Figure 18 Annual Summary

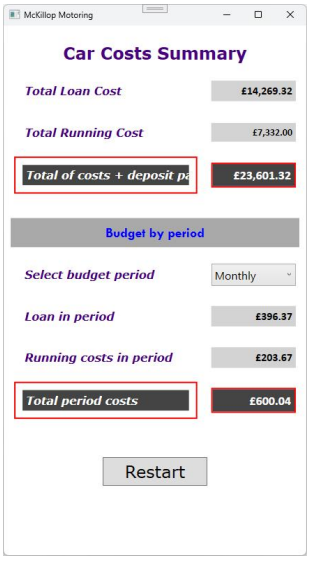

Figure 19 Monthly Summary

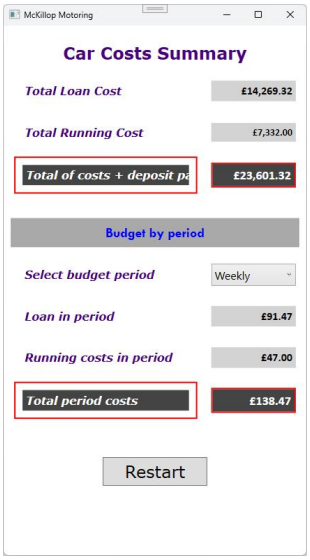

Figure 20 Weekly Summary

Figure 21 PageLoan on Restart button event

Calculation validation

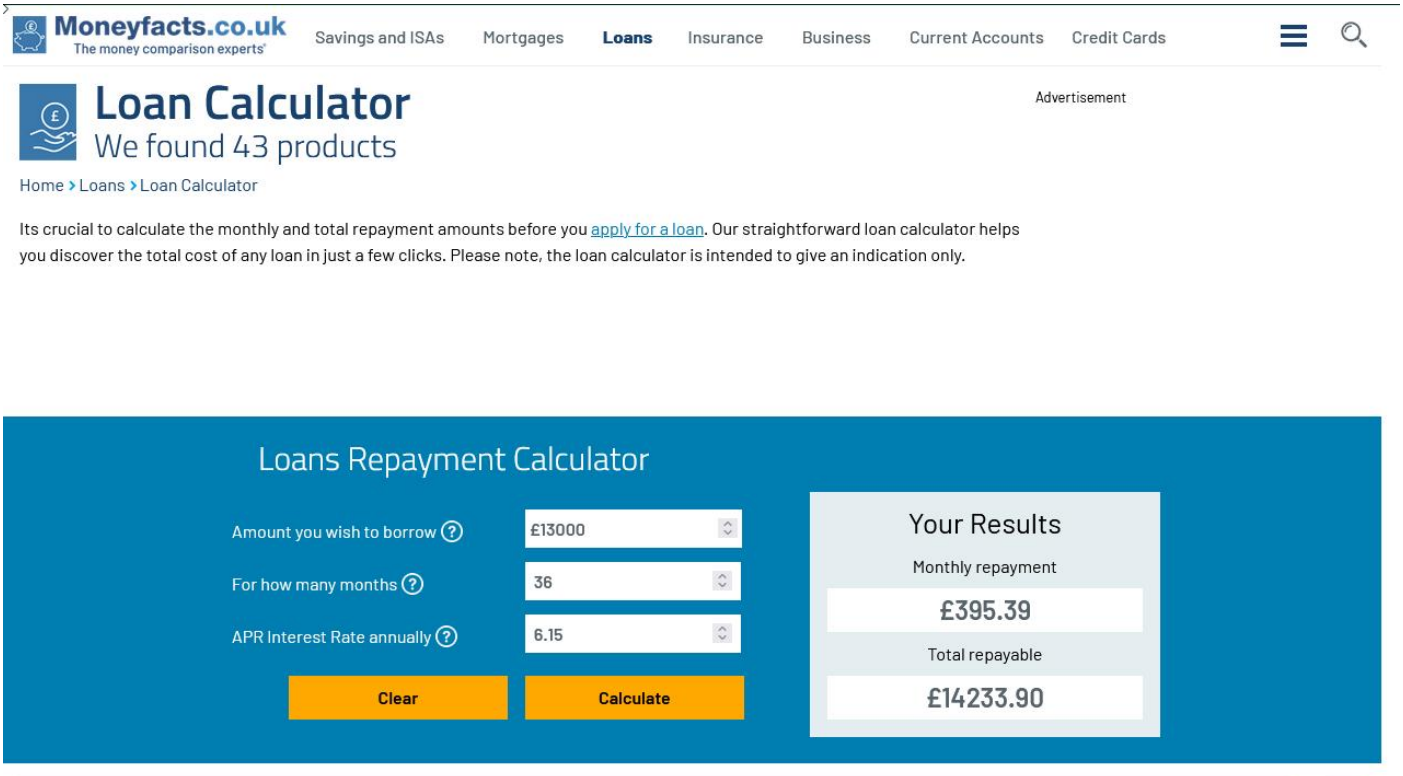

Figure 22 Validation of the calculations

The website (moneyfacts, 2022) produces a result that is very close to the calculated result from the application, £14269.32. The difference is small and acceptable. The error is produced by the rounding of the Math library using the 'POW' code feature.

Sample Test Log

	Object	Item	Test	Method	Expected Outcome	Date Complete	Actual Outcome	Any action taken
001	PageLogin	Page	Page launches	Launch application	Page displays correctly	24/10/2022	Page opened	None
002	PageLogin	Form controls	Minimise, Maximise, Close	Use controls after launch	Behave as required	24/10/2022	All controls working	None
003	PageLogin	TextBoxes	Can enter text and format Okay	Enter text to expected sizes	Text enters to control correctly	24/10/2022	Font is large	Reduce font size by modifying app.xaml style
004	PageLogin	ButtonClear	Click control clears text entries	Click control with text in boxes	Boxes are cleared	24/10/2022	Text cleared	None
005	PageLogin	ButtonHelp	Click control displays message	Click control	Message shows	24/10/2022	Not yet implemented	NA
006	PageLogin	ButtonLogin	Behaviour both values empty	Click control	Error message shows	24/10/2022	Error reported correctly	None
007	PageLogin	ButtonLogin	Behaviour Username empty, password OK	Click control	Error message shows	24/10/2022	Error reported correctly	None
008	PageLogin	ButtonLogin	Behaviour Username OK, password empty	Click control	Error message shows	24/10/2022	Error reported correctly	None
009	PageLogin	ButtonLogin	Behaviour Username OK, Password OK	Click control	Login success. PageLoan opens	24/10/2022	PageLoan opens correctly	None
010	PageLoan	Page	Page Launches	Launches from PageLogin	Opens and displays correctly	26/10/2022	PageLoan opens correctly	None
etc								

Figure 23 Sample Test Log

Appendix A: XAML Code

PageLogin

```
<Page x:Class="McKillopMotoring.PageLogin"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Login"
      Margin="5"
      ShowsNavigationUI="False">

    <!-- PageLogin Code
            Paul McKillop
            October 2022-->

    <Grid>
        <StackPanel>
            <Grid VerticalAlignment="Center">
                <Grid.RowDefinitions>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="*"/>
                    <RowDefinition  Height="*"/>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="2*"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                </Grid.ColumnDefinitions>

                <!-- Row 0 -->
                <TextBlock
                    Grid.Row="0"
                    Grid.Column="0"
                    Grid.ColumnSpan="2"
                    Style="{StaticResource PageHeaderStyle}">
                    Motoring Costs Login
                </TextBlock>
                <!-- Row 1 -->
                <StackPanel
                    Grid.Row="1"
                    Grid.Column="0"
                    Grid.ColumnSpan="2">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Enter your Username
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource ValueTextBox}"
                        x:Name="UsernameTextBox"></TextBox>
                </StackPanel>

                <!-- Row 2 -->
                <StackPanel
                    Grid.Row="2 "
                    Grid.Column="0"
                    Grid.ColumnSpan="2">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Enter your Password
```

```
                </TextBlock>
                <TextBox
                    Style="{StaticResource ValueTextBox}"
                    x:Name="PasswordTextBox"></TextBox>
            </StackPanel>
            <!-- Row 3 -->
            <StackPanel
                Grid.Row="3"
                Grid.Column="0">
                <Button
                    Style="{StaticResource FormButton}"
                    Margin="50 10 10 10"
                    x:Name="ClearButton"
                    Click="ClearButton_Click">Clear</Button>
            </StackPanel>

            <StackPanel
                Grid.Row="3"
                Grid.Column="1">
                <Button
                    Style="{StaticResource FormButton}"
                    Margin="10 10 50 10"
                    x:Name="LoginButton"
                    Click="LoginButton_Click">Login</Button>
            </StackPanel>

            <!-- Row 4 -->
            <StackPanel
                Grid.Row="4"
                Grid.ColumnSpan="2"
                VerticalAlignment="Center"
                HorizontalAlignment="Center"
                Margin="10 50">
                <Button
                    Style="{StaticResource FormButton}"
                    Margin="30"
                    Padding="20"
                    FontWeight="Bold"
                    Foreground="Indigo">
                    Help
                </Button>

            </StackPanel>

        </Grid>
    </StackPanel>
</Grid>
</Page>
```

```xaml
<Page x:Class="McKillopMotoring.PageLogin"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Login"
      Margin="5"
      ShowsNavigationUI="False">

    <!-- PageLogin Code
           Paul McKillop
           October 2022-->

    <Grid>
        <StackPanel>
            <Grid VerticalAlignment="Center">
                <Grid.RowDefinitions>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="*"/>
                    <RowDefinition  Height="*"/>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="2*"/>
                    <RowDefinition  Height="2*"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                </Grid.ColumnDefinitions>

                <!-- Row 0 -->
                <TextBlock
                    Grid.Row="0"
                    Grid.Column="0"
                    Grid.ColumnSpan="2"
                    Style="{StaticResource PageHeaderStyle}">
                    Motoring Costs Login
                </TextBlock>
                <!-- Row 1 -->
                <StackPanel
                    Grid.Row="1"
                    Grid.Column="0"
                    Grid.ColumnSpan="2">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Enter your Username
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource ValueTextBox}"
                        x:Name="UsernameTextBox"></TextBox>
                </StackPanel>

                <!-- Row 2 -->
                <StackPanel
                    Grid.Row="2 "
                    Grid.Column="0"
                    Grid.ColumnSpan="2">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Enter your Password
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource ValueTextBox}"
                        x:Name="PasswordTextBox"></TextBox>
                </StackPanel>
                <!-- Row 3 -->
                <StackPanel
                    Grid.Row="3"
                    Grid.Column="0">
                    <Button
                        Style="{StaticResource FormButton}"
                        Margin="50 10 10 10"
                        x:Name="ClearButton"
                        Click="ClearButton_Click">Clear</Button>
                </StackPanel>

                <StackPanel
                    Grid.Row="3"
                    Grid.Column="1">
                    <Button
                        Style="{StaticResource FormButton}"
                        Margin="10 10 50 10"
                        x:Name="LoginButton"
                        Click="LoginButton_Click">Login</Button>
                </StackPanel>

                <!-- Row 4 -->
                <StackPanel
                    Grid.Row="4"
                    Grid.ColumnSpan="2"
                    VerticalAlignment="Center"
                    HorizontalAlignment="Center"
                    Margin="10 50">
                    <Button
                        Style="{StaticResource FormButton}"
                        Margin="30"
                        Padding="20"
                        FontWeight="Bold"
                        For
                        Help
                    </Button>

                </StackPanel>

            </Grid>
        </StackPanel>
    </Grid>
</Page>
```

Figure 24 PageLogin XAML Code

PageLoan

```
<Page x:Class="McKillopMotoring.PageLoan"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Loan"
      Margin="5">
    <!-- Page Loan Code
            Paul McKillop
            October 2022-->

    <Grid>
        <StackPanel>
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="2*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="2*"/>

                </Grid.RowDefinitions>

                <Grid.ColumnDefinitions>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                </Grid.ColumnDefinitions>

                <!-- Row 0 -->
                <TextBlock
                    Grid.Row="0"
                    Grid.ColumnSpan="6"
                    Style="{StaticResource PageHeaderStyle}">
                    Loan Details
                </TextBlock>

                <!-- Row 1 -->
                <!-- Car price -->
                <StackPanel
                    Grid.Row="1"
                    VerticalAlignment="Center"
                    Grid.ColumnSpan="6">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Car Price
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource NumbersTextBox}"
                        x:Name="CarPriceTextBox">

                    </TextBox>

                </StackPanel>
                <!-- Row 2 -->
                <!-- Deposit -->
                <StackPanel
                    Grid.Row="2"
```

```
        VerticalAlignment="Center"
        Grid.ColumnSpan="6">
        <TextBlock
            Style="{StaticResource ControlLabelStyle}">
            Deposit
        </TextBlock>
        <TextBox
            Style="{StaticResource NumbersTextBox}"
            x:Name="CarDepositTextBox">
        </TextBox>

    </StackPanel>
    <!-- Row 3 -->
    <!-- Term of loan -->
    <StackPanel
        VerticalAlignment="Center"
        Grid.Row="3"
        Grid.Column="0"
        Grid.ColumnSpan="4">
        <TextBlock
            Style="{StaticResource ControlLabelStyle}">
            Term in years
        </TextBlock>
    </StackPanel>

    <StackPanel
        Grid.Row="3"
        Grid.Column="4"
        Grid.ColumnSpan="2">
        <TextBox
            Style="{StaticResource NumbersTextBox}"
            x:Name="LoanTermTextBox">

        </TextBox>
    </StackPanel>
    <!-- Row 4 -->
    <!-- Interest rate -->
    <StackPanel
        VerticalAlignment="Center"
        Grid.Row="4"
        Grid.Column="0"
        Grid.ColumnSpan="4">
        <TextBlock
            Style="{StaticResource ControlLabelStyle}">
            Interest rate
        </TextBlock>
    </StackPanel>

    <StackPanel
        Grid.Row="4"
        Grid.Column="4"
        Grid.ColumnSpan="2">
        <TextBox
            Style="{StaticResource NumbersTextBox}"
            x:Name="InterestRateTextBox">

        </TextBox>
    </StackPanel>
    <!-- Row 5 -->
    <!-- Buttons -->
    <StackPanel
        Grid.Column="0"
        Grid.ColumnSpan="3"
        Grid.Row="5"
        VerticalAlignment="Center"
        Margin="20 40">
        <Button
```

```
                    Style="{StaticResource FormButton}"
                    x:Name="ClearButton"
                    Click="ClearButton_OnClick">
                    Clear
                </Button>

            </StackPanel>

            <StackPanel
                Grid.Column="3"
                Grid.ColumnSpan="3"
                Grid.Row="5"
                VerticalAlignment="Center"
                Margin="20 40">
                <Button
                    x:Name="RunningPageButton"
                    Click="RunningPageButton_OnClick"
                    Style="{StaticResource FormButton}">
                    Next
                </Button>
            </StackPanel>

        </Grid>
    </StackPanel>
</Grid>
</Page>
```

```
<Page x:Class="McKillopMotoring.PageLoan"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Loan"
      Margin="5">
    <!-- Page Loan Code
            Paul McKillop
            October 2022-->

    <Grid>
        <StackPanel>
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="2*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="2*"/>

                </Grid.RowDefinitions>

                <Grid.ColumnDefinitions>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                </Grid.ColumnDefinitions>

                <!-- Row 0 -->
                <TextBlock
                    Grid.Row="0"
                    Grid.ColumnSpan="6"
                    Style="{StaticResource PageHeaderStyle}">
                    Loan Details
                </TextBlock>

                <!-- Row 1 -->
                <!-- Car price -->
                <StackPanel
                    Grid.Row="1"
                    VerticalAlignment="Center"
                    Grid.ColumnSpan="6">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Car Price
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource NumbersTextBox}"
                        x:Name="CarPriceTextBox">

                    </TextBox>

                </StackPanel>
                <!-- Row 2 -->
                <!-- Deposit -->
                <StackPanel
                    Grid.Row="2"
                    VerticalAlignment="Center"
                    Grid.ColumnSpan="6">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Deposit
                    </TextBlock>
                    <TextBox
                        Style="{StaticResource NumbersTextBox}"
                        x:Name="CarDepositTextBox">
                    </TextBox>

                </StackPanel>
                <!-- Row 3 -->
                <!-- Term of loan -->
                <StackPanel
                    VerticalAlignment="Center"
                    Grid.Row="3"
                    Grid.Column="0"
                    Grid.ColumnSpan="4">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Term in years
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="3"
                    Grid.Column="4"
                    Grid.ColumnSpan="2">
                    <TextBox
                        Style="{StaticResource NumbersTextBox}"
                        x:Name="LoanTermTextBox">

                    </TextBox>
                </StackPanel>
                <!-- Row 4 -->
                <!-- Interest rate -->
                <StackPanel
                    VerticalAlignment="Center"
                    Grid.Row="4"
                    Grid.Column="0"
                    Grid.ColumnSpan="4">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Interest rate
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="4"
                    Grid.Column="4"
                    Grid.ColumnSpan="2">
                    <TextBox
                        Style="{StaticResource NumbersTextBox}"
                        x:Name="InterestRateTextBox">

                    </TextBox>
                </StackPanel>
                <!-- Row 5 -->
                <!-- Buttons -->
                <StackPanel
                    Grid.Column="0"
                    Grid.ColumnSpan="3"
                    Grid.Row="5"
                    VerticalAlignment="Center"
                    Margin="20 40">
                    <Button
                        Style="{StaticResource FormButton}"
                        x:Name="ClearButton"
                        Click="ClearButton_OnClick">
                        Clear
                    </Button>

                </StackPanel>

                <StackPanel
                    Grid.Column="3"
                    Grid.ColumnSpan="3"
                    Grid.Row="5"
                    VerticalAlignment="Center"
                    Margin="20 40">
                    <Button
                        x:Name="RunningPageButton"
                        Click="RunningPageButton_OnClick"
                        Style="{StaticResource FormButton}">
                        Next
                    </Button>
                </StackPanel>

            </Grid>
        </StackPanel>
    </Grid>
</Page>

```

Figure 25 PageLoan XAML Code

PageRunning

```
<Page x:Class="McKillopMotoring.PageRunning"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Running Costs"
      Margin="5">

    <!-- PageRunning Code
            Paul McKillop
            October 2022-->

    <Grid>
        <StackPanel>
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="2*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="3*"/>
                    <RowDefinition Height="*"/>
                </Grid.RowDefinitions>

                <Grid.ColumnDefinitions>
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                </Grid.ColumnDefinitions>


                <!-- Row 0 -->
                <TextBlock
                    Grid.ColumnSpan="6"
                    Style="{StaticResource PageHeaderStyle}">
                    Running costs
                </TextBlock>

                <!-- Row 1 -->
                <StackPanel
                    Grid.Row="1"
                    Grid.ColumnSpan="2"
                    Grid.Column="0"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Insurance
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="1"
                    Grid.ColumnSpan="2"
                    Grid.Column="2"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBox
                        x:Name="InsuranceCostTextBox"
```

```
        Style="{StaticResource NumbersTextBox}">0</TextBox>
</StackPanel>

<StackPanel
    Grid.Row="1"
    Grid.ColumnSpan="2"
    Grid.Column="4"
    Margin="0 10"
    VerticalAlignment="Center">
    <ComboBox
        Style="{StaticResource CostPeriodCombo}"
        x:Name="InsurancePeriodCombo"
        Loaded="InsurancePeriodCombo_Loaded"
        SelectionChanged="InsurancePeriodCombo_SelectionChanged"
        ></ComboBox>
</StackPanel>

<!-- Row 2 -->
<StackPanel
    Grid.Row="2"
    Grid.ColumnSpan="2"
    Grid.Column="0"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBlock
        Style="{StaticResource ControlLabelStyle}">
        Fuel
    </TextBlock>
</StackPanel>

<StackPanel
    Grid.Row="2"
    Grid.ColumnSpan="2"
    Grid.Column="2"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBox
        x:Name="FuelCostTextBox"
        Style="{StaticResource NumbersTextBox}">0</TextBox>
</StackPanel>

<StackPanel
    Grid.Row="2"
    Grid.ColumnSpan="2"
    Grid.Column="4"
    Margin="0 10"
    VerticalAlignment="Center">
    <ComboBox
        Style="{StaticResource CostPeriodCombo}"
        x:Name="FuelPeriodCombo"
        Loaded="FuelPeriodCombo_Loaded"
        SelectionChanged="FuelPeriodCombo_SelectionChanged"
        ></ComboBox>
</StackPanel>

<!-- Row 3 -->
<StackPanel
    Grid.Row="3"
    Grid.ColumnSpan="2"
    Grid.Column="0"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBlock
        Style="{StaticResource ControlLabelStyle}">
        Servicing
    </TextBlock>
</StackPanel>
```

```
<StackPanel
    Grid.Row="3"
    Grid.ColumnSpan="2"
    Grid.Column="2"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBox
        x:Name="ServicingCostTextBox"
        Style="{StaticResource NumbersTextBox}">0</TextBox>
</StackPanel>

<StackPanel
    Grid.Row="3"
    Grid.ColumnSpan="2"
    Grid.Column="4"
    Margin="0 10"
    VerticalAlignment="Center">
    <ComboBox
        Style="{StaticResource CostPeriodCombo}"
        x:Name="ServicingPeriodCombo"
        Loaded="ServicingPeriodCombo_Loaded"
        SelectionChanged="ServicingPeriodCombo_SelectionChanged"
        ></ComboBox>
</StackPanel>

<!-- Row 4 -->
<StackPanel
    Grid.Row="4"
    Grid.ColumnSpan="2"
    Grid.Column="0"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBlock
        Style="{StaticResource ControlLabelStyle}">
        Road Tax
    </TextBlock>
</StackPanel>

<StackPanel
    Grid.Row="4"
    Grid.ColumnSpan="2"
    Grid.Column="2"
    Margin="0 10"
    VerticalAlignment="Center">
    <TextBox
        x:Name="RoadTaxCostTextBox"
        Style="{StaticResource NumbersTextBox}">0</TextBox>
</StackPanel>

<StackPanel
    Grid.Row="4"
    Grid.ColumnSpan="2"
    Grid.Column="4"
    Margin="0 10"
    VerticalAlignment="Center">
    <ComboBox
        Style="{StaticResource CostPeriodCombo}"
        x:Name="RoadTaxPeriodCombo"
        Loaded="RoadTaxPeriodCombo_Loaded"
        SelectionChanged="RoadTaxPeriodCombo_SelectionChanged"
        ></ComboBox>
</StackPanel>

<!-- Row 5 -->
<StackPanel
    Grid.Row="5"
```

```
                    Grid.ColumnSpan="3"
                    Grid.Column="0"
                    VerticalAlignment="Center"
                    Margin="10 50 10 20">
                    <Button
                        Style="{StaticResource FormButton}"
                        x:Name="ClearButton"
                        Click="ClearButton_Click">
                        Clear
                    </Button>
                </StackPanel>

                <StackPanel
                    Grid.Row="5"
                    Grid.ColumnSpan="3"
                    Grid.Column="3"
                    VerticalAlignment="Center"
                    Margin="10 50 10 20">
                    <Button
                        Style="{StaticResource FormButton}"
                        x:Name="SummaryPageButton"
                        Click="SummaryPageButton_Click">
                        Summary
                    </Button>
                </StackPanel>
            </Grid>
        </StackPanel>

    </Grid>
</Page>
```

```xaml
<Page x:Class="McKillopMotoring.PageRunning"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:McKillopMotoring"
      mc:Ignorable="d"
      d:DesignHeight="800" d:DesignWidth="450"
      Title="Running Costs"
      Margin="5">

    <!-- PageRunning Code
         Paul McKillop
         October 2022-->

    <Grid>
        <StackPanel>
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="2*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="*"/>
                    <RowDefinition Height="3*"/>
                    <RowDefinition Height="*"/>
                </Grid.RowDefinitions>

                <Grid.ColumnDefinitions>
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                    <ColumnDefinition />
                </Grid.ColumnDefinitions>


                <!-- Row 0 -->
                <TextBlock
                    Grid.ColumnSpan="6"
                    Style="{StaticResource PageHeaderStyle}">
                    Running costs
                </TextBlock>

                <!-- Row 1 -->
                <StackPanel
                    Grid.Row="1"
                    Grid.ColumnSpan="2"
                    Grid.Column="0"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Insurance
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="1"
                    Grid.ColumnSpan="2"
                    Grid.Column="2"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBox
                        x:Name="InsuranceCostTextBox"
                        Style="{StaticResource NumbersTextBox}">0</TextBox>
                </StackPanel>

                <StackPanel
                    Grid.Row="1"
                    Grid.ColumnSpan="2"
                    Grid.Column="4"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <ComboBox
                        Style="{StaticResource CostPeriodCombo}"
                        x:Name="InsurancePeriodCombo"
                        Loaded="InsurancePeriodCombo_Loaded"
                        SelectionChanged="InsurancePeriodCombo_SelectionChanged"
                        ></ComboBox>
                </StackPanel>

                <!-- Row 2 -->
                <StackPanel
                    Grid.Row="2"
                    Grid.ColumnSpan="2"
                    Grid.Column="0"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Fuel
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="2"
                    Grid.ColumnSpan="2"
                    Grid.Column="2"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBox
                        x:Name="FuelCostTextBox"
                        Style="{StaticResource NumbersTextBox}">0</TextBox>
                </StackPanel>

                <StackPanel
                    Grid.Row="2"
                    Grid.ColumnSpan="2"
                    Grid.Column="4"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <ComboBox
                        Style="{StaticResource CostPeriodCombo}"
                        x:Name="FuelPeriodCombo"
                        Loaded="FuelPeriodCombo_Loaded"
                        SelectionChanged="FuelPeriodCombo_SelectionChanged"
                        ></ComboBox>
                </StackPanel>

                <!-- Row 3 -->
                <StackPanel
                    Grid.Row="3"
                    Grid.ColumnSpan="2"
                    Grid.Column="0"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Servicing
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="3"
                    Grid.ColumnSpan="2"
                    Grid.Column="2"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBox
                        x:Name="ServicingCostTextBox"
                        Style="{StaticResource NumbersTextBox}">0</TextBox>
                </StackPanel>

                <StackPanel
                    Grid.Row="3"
                    Grid.ColumnSpan="2"
                    Grid.Column="4"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <ComboBox
                        Style="{StaticResource CostPeriodCombo}"
                        x:Name="ServicingPeriodCombo"
                        Loaded="ServicingPeriodCombo_Loaded"
                        SelectionChanged="ServicingPeriodCombo_SelectionChanged"
                        ></ComboBox>
                </StackPanel>

                <!-- Row 4 -->
                <StackPanel
                    Grid.Row="4"
                    Grid.ColumnSpan="2"
                    Grid.Column="0"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBlock
                        Style="{StaticResource ControlLabelStyle}">
                        Road Tax
                    </TextBlock>
                </StackPanel>

                <StackPanel
                    Grid.Row="4"
                    Grid.ColumnSpan="2"
                    Grid.Column="2"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <TextBox
                        x:Name="RoadTaxCostTextBox"
                        Style="{StaticResource NumbersTextBox}">0</TextBox>
                </StackPanel>

                <StackPanel
                    Grid.Row="4"
                    Grid.ColumnSpan="2"
                    Grid.Column="4"
                    Margin="0 10"
                    VerticalAlignment="Center">
                    <ComboBox
                        Style="{StaticResource CostPeriodCombo}"
                        x:Name="RoadTaxPeriodCombo"
                        Loaded="RoadTaxPeriodCombo_Loaded"
                        SelectionChanged="RoadTaxPeriodCombo_SelectionChanged"
                        ></ComboBox>
                </StackPanel>

                <!-- Row 5 -->
                <StackPanel
                    Grid.Row="5"
                    Grid.ColumnSpan="3"
                    Grid.Column="0"
                    VerticalAlignment="Center"
                    Margin="10 50 10 20">
                    <Button
                        Style="{StaticResource FormButton}"
                        x:Name="ClearButton"
                        Click="ClearButton_Click">
                        Clear
                    </Button>
                </StackPanel>

                <StackPanel
                    Grid.Row="5"
                    Grid.ColumnSpan="3"
                    Grid.Column="3"
                    VerticalAlignment="Center"
                    Margin="10 50 10 20">
                    <Button
                        Style="{StaticResource FormButton}"
                        x:Name="SummaryPageButton"
                        Click="SummaryPageButton_Click">
                        Summary
                    </Button>
                </StackPanel>
            </Grid>
        </StackPanel>

    </Grid>
</Page>
```

Figure 26 PageRunning XAML Code

PageSummary

```
<Page x:Class="McKillopMotoring.PageSummary"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:McKillopMotoring"
    mc:Ignorable="d"
    d:DesignHeight="800" d:DesignWidth="450"
    Title="Summary"
    Margin="5">

  <!--PageSummary Code
        Paul McKillop
        October 2022-->

  <Grid>
    <StackPanel>
      <Grid>
        <Grid.RowDefinitions>
          <RowDefinition Height="3*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="2*"/>
          <RowDefinition Height="3*"/>
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
```

```
    <ColumnDefinition />
    <ColumnDefinition />
    <ColumnDefinition />
    <ColumnDefinition />
    <ColumnDefinition />
    <ColumnDefinition />
</Grid.ColumnDefinitions>

<!-- Row 0 -->
<StackPanel
    HorizontalAlignment="Center"
    Grid.ColumnSpan="6">
    <TextBlock
        Style="{StaticResource PageHeaderStyle}">
        Car Costs Summary
    </TextBlock>
</StackPanel>

<!-- Row 1 -->
<StackPanel
    Grid.Row="1"
    Grid.ColumnSpan="4"
    Margin="10 5 10 5">
    <TextBlock
        Style="{StaticResource ControlLabelStyle}"
        Padding="5">Total Loan Cost</TextBlock>
</StackPanel>

<StackPanel
    Grid.Row="1"
    Grid.Column="4"
    Grid.ColumnSpan="2"
    Margin="10 5 10 5"
```

```
    VerticalAlignment="Center">

    <TextBlock
        x:Name="TotalLoanTextBlock"
        VerticalAlignment="Center"
        TextAlignment="Right"
        Background="LightGray"
        Foreground="Black"
        FontWeight="Bold"
        FontFamily="Calibri"
        FontSize="16"
        Padding="5">0.00</TextBlock>


</StackPanel>


<!-- Row 2 -->

<StackPanel
    Grid.Row="2"
    Grid.ColumnSpan="4"
    Margin="10 5 10 5">
    <TextBlock
      Style="{StaticResource ControlLabelStyle}"
      Padding="5">Total Running Cost</TextBlock>
</StackPanel>

<StackPanel
    Grid.Row="2"
    Grid.Column="4"
    Grid.ColumnSpan="2"
    Margin="10 5 10 5"
```

```
    VerticalAlignment="Center">
    <TextBlock
        x:Name="TotalRunnincCostTextBlock"
        VerticalAlignment="Center"
        TextAlignment="Right"
        Background="LightGray"
        Foreground="Black"
        FontWeight="Bold"
        FontFamily="Calibri"
        FontSize="14"
        Padding="5">0.00</TextBlock>
</StackPanel>

<!-- Row 3 -->

<StackPanel
    Grid.Row="3"
    Grid.ColumnSpan="4"
    Margin="10 5 10 5">
    <Border
        BorderThickness="2"
        BorderBrush="Red">
        <TextBlock
            Style="{StaticResource ControlLabelStyle}"
            Background="#444"
            Foreground="White"
            FontSize="16"
            TextAlignment="Left"
            Padding="5">Total of costs + deposit paid</TextBlock>
    </Border>

</StackPanel>
<StackPanel
```

```
    Grid.Row="3"
    Grid.Column="4"
    Grid.ColumnSpan="2"
    Margin="10 5 10 5"
    VerticalAlignment="Center">
    <Border
      BorderThickness="2"
      BorderBrush="Red">
      <TextBlock
        x:Name="TotalOwenershipTextBlock"
        VerticalAlignment="Center"
        TextAlignment="Right"
        Background="#444"
        Foreground="White"
        FontWeight="Bold"
        FontFamily="Calibri"
        FontSize="18"
        Padding="5">0.00</TextBlock>
    </Border>

</StackPanel>



<!-- Row 4 Separator Block-->

<StackPanel
    Grid.Row="4"
    Grid.ColumnSpan="6"
    VerticalAlignment="Center"
    Margin="5 30 5 10">
    <TextBlock
      HorizontalAlignment="Stretch"
      Foreground="Blue"
```

```
        FontWeight="Bold"
        FontFamily="TW Cen MT"
        TextAlignment="Center"
        FontSize="20"
        Background="DarkGray"
        Padding =" 10">Budget by period</TextBlock>
</StackPanel>


<!-- Row 5 -->

<StackPanel
   Grid.Row="5"
   Grid.ColumnSpan="4"
   Margin="10 5 10 5">
   <TextBlock
      Style="{StaticResource ControlLabelStyle}"
      Padding="5">Select budget period</TextBlock>
</StackPanel>
<StackPanel
   Grid.Row="5"
   Grid.Column="4"
   Grid.ColumnSpan="2"
   Margin="10 5 10 5"
   VerticalAlignment="Center">
   <ComboBox
      x:Name="BudgetPeriodCombo"
      Loaded="BudgetPeriodCombo_OnLoaded"
      SelectionChanged="BudgetPeriodCombo_OnSelectionChanged"
      FontSize="16"
      Padding="5"></ComboBox>
</StackPanel>
```

```
<!-- Row 6 -->

<StackPanel
   Grid.Row="6"
   Grid.ColumnSpan="4"
   Margin="10 5 10 5">
   <TextBlock
      Style="{StaticResource ControlLabelStyle}"
      Padding="5">Loan in period</TextBlock>
</StackPanel>
<StackPanel
   Grid.Row="6"
   Grid.Column="4"
   Grid.ColumnSpan="2"
   Margin="10 5 10 5"
   VerticalAlignment="Center">
   <TextBlock
      x:Name="LoanPeriodTextBlock"
      VerticalAlignment="Center"
      TextAlignment="Right"
      Background="LightGray"
      Foreground="Black"
      FontWeight="Bold"
      FontFamily="Calibri"
      FontSize="16"
      Padding="5">0.00</TextBlock>
</StackPanel>

<!-- Row 7 -->

<StackPanel
   Grid.Row="7"
```

```
    Grid.ColumnSpan="4"
    Margin="10 5 10 5">
    <TextBlock
        Style="{StaticResource ControlLabelStyle}"
        Padding="5">Running costs in period</TextBlock>
</StackPanel>
<StackPanel
    Grid.Row="7"
    Grid.Column="4"
    Grid.ColumnSpan="2"
    Margin="10 5 10 5"
    VerticalAlignment="Center">
    <TextBlock
        x:Name="RunningCostPeriodTextBlock"
        VerticalAlignment="Center"
        TextAlignment="Right"
        Background="LightGray"
        Foreground="Black"
        FontWeight="Bold"
        FontFamily="Calibri"
        FontSize="16"
        Padding="5">0.00</TextBlock>
</StackPanel>

<!-- Row 8 -->

<StackPanel
    Grid.Row="8"
    Grid.ColumnSpan="4"
    Margin="10 5 10 5">
    <Border
        BorderThickness="2"
        BorderBrush="Red">
```

```
      <TextBlock
        Style="{StaticResource ControlLabelStyle}"
        Background="#444"
        Foreground="White"
        FontSize="16"
        TextAlignment="left"
        Padding="5">Total period costs</TextBlock>
    </Border>

</StackPanel>
<StackPanel
    Grid.Row="8"
    Grid.Column="4"
    Grid.ColumnSpan="2"
    Margin="10 5 10 5"
    VerticalAlignment="Center">
    <Border
      BorderThickness="2"
      BorderBrush="Red">
      <TextBlock
        x:Name="TotalPeriodCostsTextBlock"
        VerticalAlignment="Center"
        TextAlignment="Right"
        Background="#444"
        Foreground="White"
        FontWeight="Bold"
        FontFamily="Calibri"
        FontSize="18"
        Padding="5">0.00</TextBlock>
    </Border>

</StackPanel>
```

```
            <!-- Row 9 -->
            <StackPanel
               Grid.Row="9"
               Grid.ColumnSpan="6"
               Grid.Column="0">
               <Button
                  Style="{StaticResource FormButton}"
                  x:Name="LoanPageButton"
                  Click="LoanPageButton_OnClick"
                  Margin="0 50">Restart
               </Button>
            </StackPanel>

         </Grid>
      </StackPanel>

   </Grid>
</Page>
```

Figure 27 PageSummary XAML Code

Appendix B: Application code

Database file

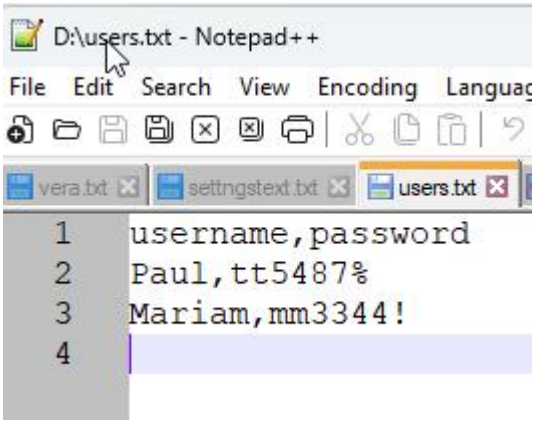

Figure 28 04 Database

Video 06 Class User

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*  TITLE:      User
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    Data class for User
 */

namespace McKillopMotoring
{
    /// <summary>
      /// A data class for holding data about user
      /// </summary>
      public class User
    {
            // -- property encapsulated
            private string username;

            public string Username
            {
                  get { return username; }
                  set { username = value; }
            }

            private string password;

            public string Password
            {
                  get { return password; }
                  set { password = value; }
            }


      }
}
```

Video 07 Class Loan

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace McKillopMotoring
{
    public class Loan
    {
        //-- Price
        private int carPrice;

        public int CarPrice
        {
            get { return carPrice; }
            set { carPrice = value; }
        }

        #region Deposit
        //-- Deposit
        private int carDeposit;

        public int CarDeposit
        {
            get { return carDeposit; }
            set { carDeposit = value; }
        }
        #endregion

        //-- Term
        private byte loanTermYears;

        public byte LoanTermYears
        {
            get { return loanTermYears; }
            set { loanTermYears = value; }
        }

        //-- Rate of interest
        private float loanRate;

        public float LoanRate
        {
            get { return loanRate; }
            set { loanRate = value; }
        }
    }
}
```

Video 08 User Validation

```
using System.Data;
using Motoring;

/*  TITLE:      UserDB
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    To validate a set of user credentials against our database
 */

namespace McKillopMotoring
{
    public class UserDB
    {
        public static bool ValidateUser(User userToValidate)
        {
            //-- tracking value variable
            bool validated = false;

            string userDatabase = @"D:\users.txt";

            DataTable userData = new DataTable();

            userData = ImportData.GetTextFileData(userDatabase);

            //-- validate credentials
            //-- Loop through the DataTable
            foreach (DataRow row in userData.Rows)
            {
                var currentUser = new User
                {
                    Username = row.Field<string>(0),
                    Password = row.Field<string>(1)
                };

                if (currentUser.Username == userToValidate.Username)
                {
                    if (currentUser.Password == userToValidate.Password)
                    {
                        validated = true;
                        break;
                    }
                }
            }

            return validated;
        }
    }
}
```

Video 09 Loan Calculator

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


/*  TITLE:      LoanCalculator
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    To calculate the payment that will be made against a loan
 */


namespace McKillopMotoring
{
    public class LoanCalculator
    {
        //-- create a constant used multiple time in calculations
        private const int MonthsPerYear = 12;

        #region Monthly payments
        /// <summary>
        /// Calculate the monthly costs
        /// </summary>
        /// <param name="myLoan"></param>
        /// <returns>decimal</returns>
        public static decimal LoanMonthlyPayment(Loan myLoan)
        {
            //-- create and initialise the return variable
            double payment = 0;

            //-- create local variables from the argument class
            var loanTermMonths = myLoan.LoanTermYears * MonthsPerYear;
            var interestRate = Convert.ToDouble(myLoan.LoanRate);
            var loanAmount = myLoan.CarPrice - myLoan.CarDeposit;

            if (myLoan.LoanTermYears > 0) //-- Check term years not 0
            {
                if (myLoan.LoanRate != 0)
                {
                    var rate = (double)((interestRate / MonthsPerYear) / 100);
                    var factor = (rate + (rate / (Math.Pow(rate + 1, loanTermMonths) - 1)));
//-- use built-in library for compound interest
                    payment = (loanAmount * factor);
                }
                else
                {
                    payment = (loanAmount / (double)loanTermMonths);
                }
            }

            return Math.Round((decimal)payment, 2);
        }
        #endregion


        #region Annual payments
        /// <summary>
        /// Annual Payments
        /// </summary>
        /// <param name="myLoan"></param>
        /// <returns>decimal</returns>
        public static decimal LoanAnnualPayment(Loan myLoan)
        {
```

```
            return Math.Round(LoanMonthlyPayment(myLoan) * 12, 2);
        }
        #endregion


        #region Weekly payments
        /// <summary>
        /// Weekly payments
        /// </summary>
        /// <param name="myLoan"></param>
        /// <returns></returns>
        public static decimal LoanWeeklyPayment(Loan myLoan)
        {
            return Math.Round(LoanAnnualPayment(myLoan) / 52, 2);
        }

        public static decimal LoanTotalPayment(Loan myLoan)
        {
            return Math.Round(LoanAnnualPayment(myLoan) * myLoan.LoanTermYears, 2);
        }
        #endregion

    }
}
```

Video 10 Class RunningCost

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*  TITLE:      RunningCostsCalculator
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    Calculate the costs other than the Loan costs
 */

namespace McKillopMotoring
{
   public class RunningCostsCalculator
   {

      #region Insurance
      // - Insurance Costs

      // --- Annual costs
      public static double InsuranceCostAnnual(RunningCost myRunningCost)
      {
         //-- return value
         double cost = 0;
         string period = myRunningCost.InsurancePeriod;

         switch (period)
         {
            case "Annual":
               cost = Convert.ToDouble(myRunningCost.Insurance);
               break;
```

```
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Insurance * 12);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Insurance * 52);
            break;
        //-- switch must have a default return value to avoid endless loop or unreachable code
        default:
            cost = 0;
            break;
    }

    // --now return the value from the method based on the switch
    return Math.Round(cost, 2);
}

// --- Monthly costs
public static double InsuranceCostMonthly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.InsurancePeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Insurance / 12);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Insurance);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Insurance * 52 / 12);
            break;
```

```
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}
// --- Weekly costs
public static double InsuranceCostWeekly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.InsurancePeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Insurance / 52);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Insurance * 12 / 52);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Insurance);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}
```

```
#endregion

//-- Fuel
//-- Annual
#region Fuel costs

public static double FuelCostAnnual(RunningCost myRunningCost)
{
   double cost = 0;
   string period = myRunningCost.FuelPeriod;

   switch (period)
   {
      case "Annual":
         cost = Convert.ToDouble(myRunningCost.Fuel);
         break;
      case "Monthly":
         cost = Convert.ToDouble(myRunningCost.Fuel * 12);
         break;
      case "Weekly":
         cost = Convert.ToDouble(myRunningCost.Fuel * 52);
         break;
      default:
         cost = 0;
         break;

   }

   return Math.Round(cost, 2);
}

//-- Monthly
public static double FuelCostMonthly(RunningCost myRunningCost)
```

```
{
    double cost = 0;
    string period = myRunningCost.FuelPeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Fuel / 12);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Fuel);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Fuel * 52 / 12);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}

//-- Weekly
public static double FuelCostWeekly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.FuelPeriod;

    switch (period)
    {
        case "Annual":
```

```
            cost = Convert.ToDouble(myRunningCost.Fuel / 52);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Fuel * 12 / 52);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Fuel);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}
#endregion


//-- Servicing
#region Servicing costs
public static double ServicingCostAnnual(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.ServicingPeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Servicing);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Servicing * 12);
```

```
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Servicing * 52);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}

public static double ServicingCostMonthly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.ServicingPeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Servicing / 12);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Servicing);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Servicing * 52 / 12);
            break;
        default:
            cost = 0;
            break;
```

```
    }

    return Math.Round(cost, 2);
}

public static double ServicingCostWeekly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.ServicingPeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.Servicing / 52);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.Servicing * 12 / 52);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.Servicing);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}
#endregion

//-- RoadTax
#region Road Tax Costs
```

```
public static double RoadTaxAnnual(RunningCost myRunningCost)
{
   double cost = 0;
   string period = myRunningCost.RoadTaxPeriod;

   switch (period)
   {
      case "Annual":
         cost = Convert.ToDouble(myRunningCost.RoadTax);
         break;
      case "Monthly":
         cost = Convert.ToDouble(myRunningCost.RoadTax * 12);
         break;
      case "Weekly":
         cost = Convert.ToDouble(myRunningCost.RoadTax * 52);
         break;
      default:
         cost = 0;
         break;

   }

   return Math.Round(cost, 2);
}

public static double RoadTaxMonthly(RunningCost myRunningCost)
{
   double cost = 0;
   string period = myRunningCost.RoadTaxPeriod;

   switch (period)
   {
      case "Annual":
```

```
            cost = Convert.ToDouble(myRunningCost.RoadTax / 12);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.RoadTax);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.RoadTax * 52 / 12);
            break;
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}

public static double RoadTaxWeekly(RunningCost myRunningCost)
{
    double cost = 0;
    string period = myRunningCost.RoadTaxPeriod;

    switch (period)
    {
        case "Annual":
            cost = Convert.ToDouble(myRunningCost.RoadTax / 52);
            break;
        case "Monthly":
            cost = Convert.ToDouble(myRunningCost.RoadTax * 12 / 52);
            break;
        case "Weekly":
            cost = Convert.ToDouble(myRunningCost.RoadTax);
            break;
```

```
        default:
            cost = 0;
            break;

    }

    return Math.Round(cost, 2);
}
#endregion

//-- Total Weekly Running costs
public static double TotalWeeklyRunningCost(RunningCost myRunningCost)
{
    double totalWeeklyCost = 0;
    totalWeeklyCost += InsuranceCostWeekly(myRunningCost);
    totalWeeklyCost += FuelCostWeekly(myRunningCost);
    totalWeeklyCost += ServicingCostWeekly(myRunningCost);
    totalWeeklyCost += RoadTaxWeekly(myRunningCost);

    return totalWeeklyCost;
}



#region Method Template
//-- Template
public static double CalculatorTemplate(RunningCost myRunningCost)
{
    double cost = 0;
    string period = "Z";

    switch (period)
    {
```

```
            case "Annual":
                cost = Convert.ToDouble(0);
                break;
            case "Monthly":
                cost = Convert.ToDouble(0);
                break;
            case "Weekly":
                cost = Convert.ToDouble(0);
                break;
            default:
                cost = 0;
                break;

        }

        return Math.Round(cost, 2);
    }
    #endregion

  }
}
```

Video 13 Class CostSummary

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*  TITLE:      CostSummary
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    An aggregation class bringing
 *              data for Loan and Running costs together.
 *              Useful for summary display
 */

namespace McKillopMotoring
{
    public class CostSummary
    {
        private Loan currentLoan;

        public Loan CurrentLoan
        {
            get { return currentLoan; }
            set { currentLoan = value; }
        }

        private RunningCost runningCost;

        public RunningCost RunningCost
        {
            get { return runningCost; }
            set { runningCost = value; }
        }
    }
}
```

Video 30 Logic PageLogin

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace McKillopMotoring
{
    /// <summary>
    /// Interaction logic for PageLogin.xaml
    /// </summary>
    public partial class PageLogin : Page
    {
        public PageLogin()
        {
            InitializeComponent();
            // -- Fill the default data in during development
            // -- TODO - Remove on final build
            FillTempCredentials();

        }
        /// <summary>
        /// Clear data in form controls
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void ClearButton_Click(object sender, RoutedEventArgs e)
        {
            this.UsernameTextBox.Text = "";
            this.PasswordTextBox.Text = "";
        }

        /// <summary>
        /// Initialise a User Class object
        /// Code harvests credentials or reports error if no data present in the controls
        /// Assigns credentials to the User object
        /// Uses UserDB Object to validate the assigned user against the text file database
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void LoginButton_Click(object sender, RoutedEventArgs e)
        {
            User myUser = new User();

            // -- Check username exists and assign if there
            // -- Else return a message and break
            if (this.UsernameTextBox.Text != "")
            {
                myUser.Username = this.UsernameTextBox.Text;
            }
            else
            {
                MessageBox.Show("You must enter a username");
                return;
            }
```

```
            // -- Check password present
            if (this.PasswordTextBox.Text != "")
            {
                myUser.Password = this.PasswordTextBox.Text;
            }
            else
            {
                MessageBox.Show("You must enter a password");
                return;
            }

            if (UserDB.ValidateUser(myUser))
            {
                //-- navigate to loan page
                var pageLoan = new PageLoan();
                this.NavigationService.Navigate(pageLoan);
            }
            else
            {
                //-- Show a message informing error
                MessageBox.Show("The credentials are incorrect");
                return;
            }

        }

        /// <summary>
        /// DEBUG remove on final build
        /// </summary>
        private void FillTempCredentials()
        {
            this.UsernameTextBox.Text = "Paul";
            this.PasswordTextBox.Text = "tt5487%";
        }
    }
}
```

Videos 31 and 32 Logic PageLoan

```
using System.Windows;
using System.Windows.Controls;

/*  TITLE:
 *  AUTHOR:     Paul McKillop
 *  DATE:
 *  PURPOSE:
 */

namespace McKillopMotoring
{
    /// <summary>
    /// Interaction logic for PageLoan.xaml
    /// </summary>
    public partial class PageLoan : Page
    {
        public PageLoan()
        {
            InitializeComponent();

            // -- TODO:: Remove at final Build
            FillTestData();
        }

        private void RunningPageButton_OnClick(object sender, RoutedEventArgs e)
        {
            // -- Handling objects for data
            Loan loan  = new Loan();
            CostSummary costSummary = new CostSummary();

            // -- Assign Loan through harvest
            loan = HarvestLoanData();
            // -- Pass loan to summary
            costSummary.CurrentLoan = loan;

            // -- Make a page object for navigator
            var runningCostPage = new PageRunning(costSummary);
            // -- Navigate to page
            this.NavigationService.Navigate(runningCostPage);
        }

        private void ClearButton_OnClick(object sender, RoutedEventArgs e)
        {
            this.CarPriceTextBox.Text = "";
            this.CarDepositTextBox.Text = "";
            this.LoanTermTextBox.Text = "";
            this.InterestRateTextBox.Text = "";
        }

        // -- Harvest Loan Data

        private Loan HarvestLoanData()
        {
            Loan loan = new Loan();
            //-- validate and get data

            if (CarPriceTextBox.Text != "")
            {
                if (int.TryParse(CarPriceTextBox.Text, out int price))
                {
                    loan.CarPrice = price;
                } else
                {
                    //-- Message to the user if the value is not the right type
                    MessageBox.Show("The price entered must be an integer");
```

```
            }
        }
        else
        {
            //-- Message to the user if the TextBox is empty
            MessageBox.Show("You must provide a price for the car");
        }

        // -- Deposit
        if (CarDepositTextBox.Text != "")
        {
            if (int.TryParse(CarDepositTextBox.Text, out int deposit))
            {
                loan.CarDeposit = deposit;
            }
            else
            {
                MessageBox.Show("The deposit value must be an integer");
            }
        }
        else
        {
            MessageBox.Show("Enter an amount for the deposit");
        }

        // -- Term in years
        if (LoanTermTextBox.Text != "")
        {
            if (byte.TryParse(LoanTermTextBox.Text, out byte term))
            {
                loan.LoanTermYears = term;
            }
            else
            {
                MessageBox.Show("The term must be a whole number less than 255");
            }
        }
        else
        {
            MessageBox.Show("You must supply a value for the Loan Term in Years");
        }

        // -- Interest Rate
        if (InterestRateTextBox.Text != "")
        {
            if (float.TryParse(InterestRateTextBox.Text, out float rate))
            {
                loan.LoanRate = rate;
            }
            else
            {
                MessageBox.Show("The rate must be formatted as a number");
            }
        }
        else
        {
            MessageBox.Show("You must supply a value for the interest rate");
        }
        return loan;
    }

    private void FillTestData()
    {
        Loan myLoan = new Loan
        {
            CarPrice = 15000,
            CarDeposit = 2000,
```

```
                LoanTermYears = 3,
                LoanRate = 6.15F
            };

            this.CarPriceTextBox.Text = myLoan.CarPrice.ToString();
            this.CarDepositTextBox.Text = myLoan.CarDeposit.ToString();
            this.LoanTermTextBox.Text = myLoan.LoanTermYears.ToString();
            this.InterestRateTextBox.Text = myLoan.LoanRate.ToString("F");
        }
    }
}
```

Videos 33 and 34 Logic PageRunning

```
using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;

namespace McKillopMotoring
{
    /// <summary>
    /// Interaction logic for PageRunning.xaml
    /// </summary>
    public partial class PageRunning : Page
    {
        //-- variables to hold period values selected in the combos
        //-- Placing them here allows use throughout the module
        //-- This is a different approach to previous used
        private string insurancePeriod = "Annual";     //-- Initialised Annual
        private string fuelPeriod = "Annual";          //-- Ditto
        private string servicingPeriod = "Annual";     //-- Ditto
        private string roadTaxPeriod = "Annual";       //-- Ditto
        //-- Numeric values
        private int insurance = 0;              //-- Initialised to 0
        private int fuel = 0;                   //-- Ditto
        private int servicing = 0;              //-- Ditto
        private int roadTax = 0;                //-- Ditto

        //-- Combo items list of strings
        List<string> costPeriods = new List<string>();

        //-- Object to manage data brought forward
        CostSummary summaryBroughtForward = new CostSummary();

        public PageRunning(CostSummary summaryPassed)
        {
            InitializeComponent();
            // -- Debug message
            // -- TODO:: Remove message on build
            // string price = summaryPassed.CurrentLoan.CarPrice.ToString();
            // MessageBox.Show("The Car Price passed from PageLoan was " + price);

            summaryBroughtForward = summaryPassed;
            costPeriods = Periods();

            //-- Combo data sources (maybe redundant)
            InsurancePeriodCombo.ItemsSource = Periods();
            FuelPeriodCombo.ItemsSource = Periods();
            ServicingPeriodCombo.ItemsSource = Periods();
            RoadTaxPeriodCombo.ItemsSource = Periods();


        }

        private void ClearButton_Click(object sender, RoutedEventArgs e)
        {

        }
        // -- Button Event Methods
        private void SummaryPageButton_Click(object sender, RoutedEventArgs e)
        {
            CostSummary summary = new CostSummary();

            summary = summaryBroughtForward;

            // -- Run harvest form to get values
            HarvestValues();

            RunningCost runningCost = new RunningCost
```

```
            {
                Insurance = insurance,
                InsurancePeriod = insurancePeriod,
                Fuel = fuel,
                FuelPeriod = fuelPeriod,
                Servicing = servicing,
                ServicingPeriod = servicingPeriod,
                RoadTax = roadTax,
                RoadTaxPeriod = roadTaxPeriod
            };

            summary.RunningCost = runningCost;

            var pageSummary = new PageSummary(summary);
            this.NavigationService.Navigate(pageSummary);
        }

        // -- Combo LOAD events methods
        private void InsurancePeriodCombo_Loaded(object sender, RoutedEventArgs e)
        {
            var combo = sender as ComboBox;
            combo.ItemsSource = costPeriods;
            combo.SelectedIndex = 0;
        }

        private void FuelPeriodCombo_Loaded(object sender, RoutedEventArgs e)
        {
            var combo = sender as ComboBox;
            combo.ItemsSource = costPeriods;
            combo.SelectedIndex = 0;
        }

        private void ServicingPeriodCombo_Loaded(object sender, RoutedEventArgs e)
        {
            var combo = sender as ComboBox;
            combo.ItemsSource = costPeriods;
            combo.SelectedIndex = 0;
        }

        private void RoadTaxPeriodCombo_Loaded(object sender, RoutedEventArgs e)
        {
            var combo = sender as ComboBox;
            if (combo != null)
            {
                combo.ItemsSource = costPeriods;
                combo.SelectedIndex = 0;
            }

        }

        // -- Combo SELECTION CHANGED event methods
        private void InsurancePeriodCombo_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            var selectedComboItem = sender as ComboBox;
            insurancePeriod = selectedComboItem.SelectedItem as string;
        }



        private void FuelPeriodCombo_SelectionChanged(object sender, SelectionChangedEventArgs
e)
        {
            var selectedComboItem = sender as ComboBox;
            fuelPeriod = selectedComboItem.SelectedItem as string;
        }
```

```
        private void ServicingPeriodCombo_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            var selectedComboItem = sender as ComboBox;
            servicingPeriod = selectedComboItem.SelectedItem as string;
        }


        private void RoadTaxPeriodCombo_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            var selectedComboItem = sender as ComboBox;
            roadTaxPeriod = selectedComboItem.SelectedItem as string;
        }


        // -- Utility code
        // -- Combo box data population values
        List<string> Periods()
        {
            List<string> myList = new List<string>
            {
                "Annual",
                "Monthly",
                "Weekly"
            };

            return myList;
        }

        // -- Harvest data from the form
        private void HarvestValues()
        {
            //-- Insurance
            if (InsuranceCostTextBox.Text != "")
            {
                if (int.TryParse(InsuranceCostTextBox.Text, out int insuranceValue))
                {
                    insurance = insuranceValue;
                }
                else
                {
                    MessageBox.Show("The insurance cost must be a whole number");
                }
            }
            else
            {
                MessageBox.Show("Enter the insurance cost, even if it's 0");
            }

            //-- Fuel
            if (FuelCostTextBox.Text != "")
            {
                if (int.TryParse(FuelCostTextBox.Text, out int fuelValue))
                {
                    fuel = fuelValue;
                }
                else
                {
                    MessageBox.Show("The fuel cost must be a whole number");
                }
            }
            else
            {
                MessageBox.Show("You must enter a fuel cost, evem if it's 0");
```

```
            }

            //-- Servicing
            if (ServicingCostTextBox.Text != "")
            {
                if (int.TryParse(ServicingCostTextBox.Text, out int servicingValue))
                {
                    servicing = servicingValue;
                }
                else
                {
                    MessageBox.Show("The servicing cost must be a whole number, even if it's
0");
                }
            }
            else
            {
                MessageBox.Show("You must enter a servicing cost, even if it's 0");
            }

            //-- Road tax
            if (RoadTaxCostTextBox.Text != "")
            {
                if (int.TryParse(RoadTaxCostTextBox.Text, out int roadTaxValue))
                {
                    roadTax = roadTaxValue;
                }
                else
                {
                    MessageBox.Show("The road tax amount must be a whole number, even if it's
0");
                }
            }
            else
            {
                MessageBox.Show("you must enter a Road Tax value, even it it's 0");
            }

        }
    }
}
```

Videos 35 and 36 Logic PageSummary

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace McKillopMotoring
{
    /// <summary>
    /// Interaction logic for PageSummary.xaml
    /// </summary>
    public partial class PageSummary : Page
    {
        //-- manage the data module wide
        CostSummary summary = new CostSummary();
        Loan loan = new Loan();
        RunningCost runningCost = new RunningCost();

        //-- Summary value variables
        private decimal totalLoanCost = 0;
        private double totalRunningCost = 0;
        private double totalCostOfOwnership = 0;

        //-- Period list for combo
        //-- Combo items list of strings
        List<string> costPeriods = new List<string>();

        //-- Budget period variables
        private double periodLoan = 0;
        private double periodRunningCost = 0;
        private double periodTotalCost = 0;

        private string periodSelected = "Annual";

        public PageSummary(CostSummary summaryPassed)
        {
            InitializeComponent();
            summary = summaryPassed;
            loan = summary.CurrentLoan;
            runningCost = summary.RunningCost;

            costPeriods = Periods();

            BudgetPeriodCombo.ItemsSource = costPeriods;

            // -- TODO :: Show values held
            MessageBox.Show(DataValuesSummary());

            // -- TODO :: Calculate and show totals
            ShowTotalCosts();
            // -- DEBUG REMOVE AT FINAL BUILD
            //string price = summaryPassed.CurrentLoan.CarPrice.ToString();
            //MessageBox.Show("The Car Price passed from PageLoan was " + price);

            //string insurance = summaryPassed.RunningCost.Insurance.ToString();
            //MessageBox.Show("The Insurance cost passed was " + insurance);
```

```
        }

        private void BudgetPeriodCombo_OnLoaded(object sender, RoutedEventArgs e)
        {
            var combo = sender as ComboBox;
            combo.ItemsSource = costPeriods;
            combo.SelectedIndex = 0;
        }

        private void BudgetPeriodCombo_OnSelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            var selectedComboItem = sender as ComboBox;
            periodSelected = selectedComboItem.SelectedItem as string;

            //-- work out periodLoan value
            switch (periodSelected)
            {
                case "Annual":
                    periodLoan = Convert.ToDouble(LoanCalculator.LoanAnnualPayment(loan));
                    break;
                case "Monthly":
                    periodLoan = Convert.ToDouble(LoanCalculator.LoanMonthlyPayment(loan));
                    break;
                case "Weekly":
                    periodLoan = Convert.ToDouble(LoanCalculator.LoanWeeklyPayment(loan));
                    break;
            }

            //-- Period running costs
            switch (periodSelected)
            {
                case "Annual":
                    periodRunningCost =
RunningCostsCalculator.TotalWeeklyRunningCost(runningCost) * 52;
                    break;
                case "Monthly":
                    periodRunningCost =
RunningCostsCalculator.TotalWeeklyRunningCost(runningCost) * 52 / 12;
                    break;
                case "Weekly":
                    periodRunningCost =
RunningCostsCalculator.TotalWeeklyRunningCost(runningCost);
                    break;
            }

            periodTotalCost = periodLoan + periodRunningCost;

            //-- Put values into the text blocks

            LoanPeriodTextBlock.Text = periodLoan.ToString("C");
            RunningCostPeriodTextBlock.Text = periodRunningCost.ToString("C");

            //-- Total of period costs
            TotalPeriodCostsTextBlock.Text = periodTotalCost.ToString("C");
        }

        private void LoanPageButton_OnClick(object sender, RoutedEventArgs e)
        {
            var pageLoan = new PageLoan();
            this.NavigationService.Navigate(pageLoan);
        }


        // - Utility
        // -- Combo box data population
        List<string> Periods()
```

```
        {
            List<string> myList = new List<string>
            {
                "Annual",
                "Monthly",
                "Weekly"
            };

            return myList;
        }

        private void ShowTotalCosts()
        {
            totalLoanCost = LoanCalculator.LoanTotalPayment(loan);
            TotalLoanTextBlock.Text = totalLoanCost.ToString("C");

            //-- Total running costs
            totalRunningCost = RunningCostsCalculator.TotalWeeklyRunningCost(runningCost) * 52
* loan.LoanTermYears;
            TotalRunnincCostTextBlock.Text = totalRunningCost.ToString("C");

            //-- Total Cost of ownership
            double loanCost = Convert.ToDouble(totalLoanCost);
            totalCostOfOwnership = (loanCost + totalRunningCost + loan.CarDeposit);

            TotalOwenershipTextBlock.Text = totalCostOfOwnership.ToString("C");
        }

        string DataValuesSummary()
        {
            string message = "";

            var valuesHeld = new StringBuilder();

            //-- Loan
            valuesHeld.Append("Car price: ").Append(loan.CarPrice.ToString()).AppendLine();
            valuesHeld.Append("Deposit: ").Append(loan.CarDeposit.ToString()).AppendLine();
            valuesHeld.Append("Loan term:
").Append(loan.LoanTermYears.ToString()).AppendLine();
            valuesHeld.Append("Interest rate:
").Append(loan.LoanRate.ToString()).AppendLine();
            //-- Running Costs
            valuesHeld.Append("Insurance cost:
").Append(runningCost.Insurance.ToString()).AppendLine();
            valuesHeld.Append("Insurance period:
").Append(runningCost.InsurancePeriod).AppendLine();
            valuesHeld.Append("Fuel cost: ").Append(runningCost.Fuel.ToString()).AppendLine();
            valuesHeld.Append("Fuel period: ").Append(runningCost.FuelPeriod).AppendLine();
            valuesHeld.Append("Servicing cost:
").Append(runningCost.Servicing.ToString()).AppendLine();
            valuesHeld.Append("Servicing period:
").Append(runningCost.ServicingPeriod).AppendLine();
            valuesHeld.Append("Road Tax cost:
").Append(runningCost.RoadTax.ToString()).AppendLine();
            valuesHeld.Append("Road Tax period:
").Append(runningCost.RoadTaxPeriod).AppendLine();

            //-- convert StringBuilder to string and return
            message = valuesHeld.ToString();

            return message;

        }


    }
}
```

Files for review location:

https://github.com/pgmckillop/HND-Motoring

Reference:

Moneyfacts

UK Loan Repayment Calculator | moneyfacts.co.uk. (2022). Retrieved 27 October 2022, from https://moneyfacts.co.uk/loans/loan-calculator/