Tutorial: Buat aplikasi WPF pertama Anda di Visual Studio 2019

Materi ini menunjukkan kepada Anda bagaimana mengembangkan aplikasi desktop Windows Presentation Foundation (WPF) yang mencakup elemen-elemen yang umum untuk sebagian besar aplikasi WPF: Extensible Application Markup Language (XAML), source kode, definisi aplikasi, kontrol, tata letak, *binding* data , dan *style*. Untuk mengembangkan aplikasi, Anda akan menggunakan Visual Studio.

Dalam materi ini,akan dibahas:

- Buat proyek WPF.
- Gunakan XAML untuk merancang tampilan antarmuka pengguna aplikasi (UI).
- Tulis kode untuk membangun perilaku aplikasi.
- Buat definisi aplikasi untuk mengelola aplikasi.
- Tambahkan kontrol dan buat tata letak untuk menyusun UI aplikasi.
- Buat gaya untuk penampilan yang konsisten di seluruh UI aplikasi.
- Bind UI untuk data, baik untuk mengisi UI dari data dan untuk menjaga data dan UI disinkronkan.

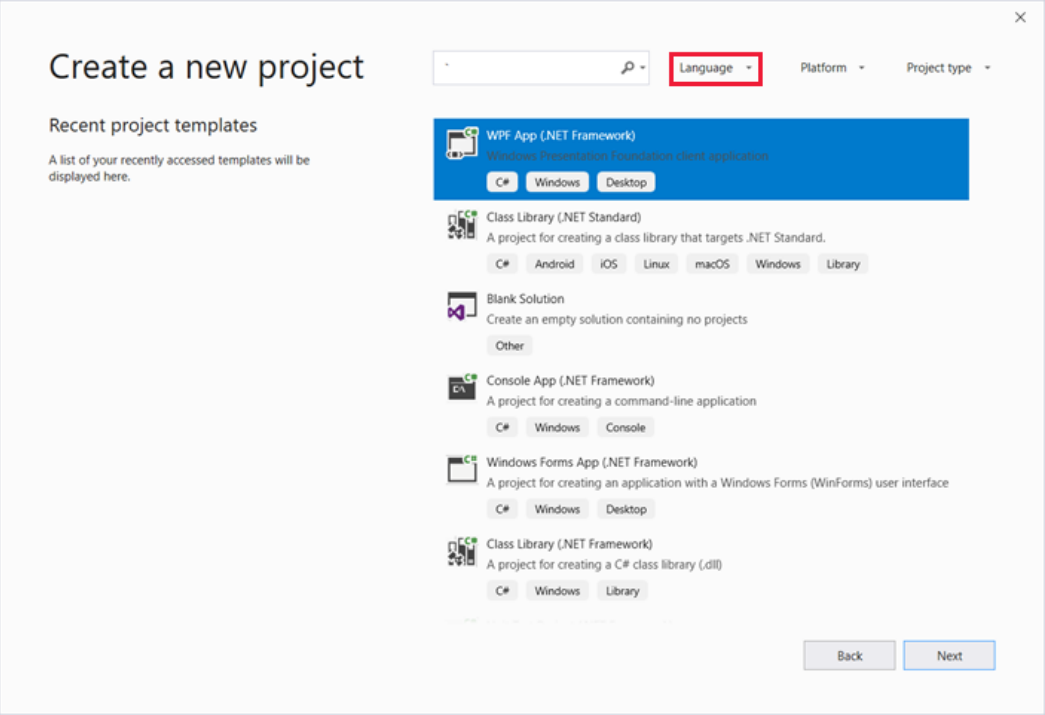
Pada akhir materi ini, Anda akan membangun aplikasi Windows mandiri yang memungkinkan pengguna untuk melihat laporan pengeluaran untuk orang-orang tertentu. Aplikasi ini terdiri dari beberapa halaman WPF yang di-host di jendela bergaya browser.

Prerequisites

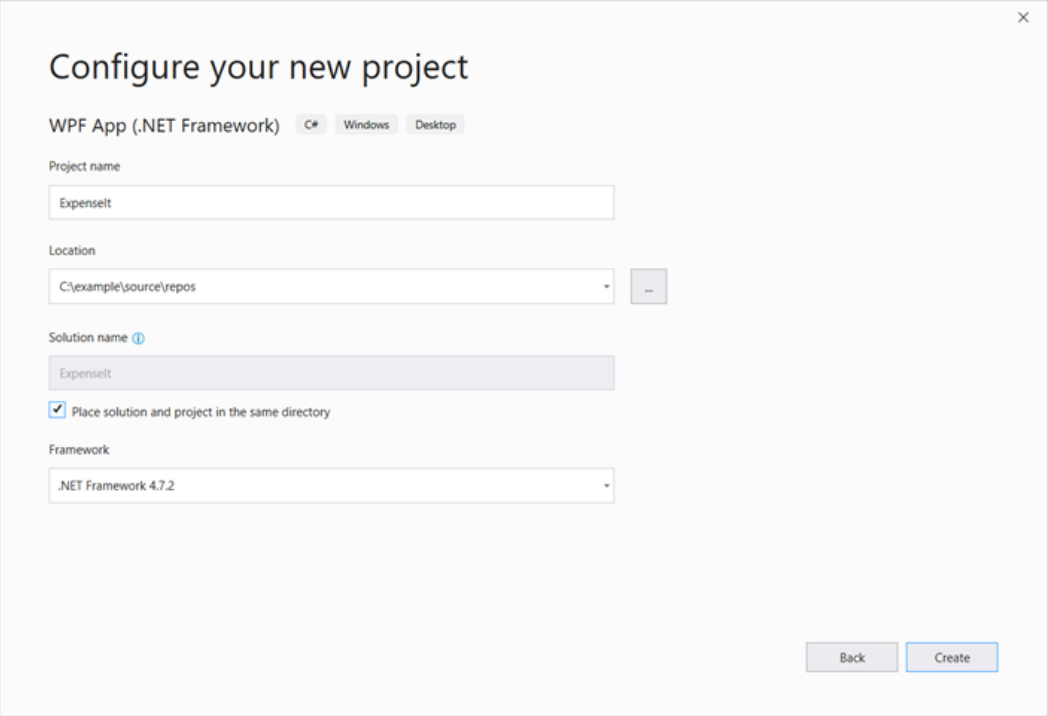
- Visual Studio 2019 with the **.NET desktop development** workload installed.
  For more information about installing the latest version of Visual Studio, see Install Visual Studio.

Membuat Project

1. Create a new WPF Application project in Visual Basic or Visual C# named `ExpenseIt`:
    a. Buka IDE visual studio → **Create a new project** (Pada jendela **Create a new project**).
    b. Pilihan **Language** pilih **C#**.
    c. Pilih template **WPF App (.NET Framework)** → **Next**.



    Dialoh **Configure your new project**.
    d. Berikan nama Project yang baru dibuat `ExpenseIt` → **Create**.



Visual Studio membuat proyek dan membuka tab desain untuk aplikasi default bernama **MainWindow.xaml**.

2. Open *App.xaml* (C#).
   File XAML ini mendefinisikan aplikasi WPF dan sumber daya (resource) yang dibutuhkan aplikasi. Anda juga menggunakan file ini untuk menentukan UI, dalam hal ini MainWindow.xaml, yang secara otomatis ditampilkan ketika aplikasi dimulai.
   Berikut tampilan file App.xaml pada C #:

```
<Application x:Class="ExpenseIt.App"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
     StartupUri="MainWindow.xaml">
    <Application.Resources>

    </Application.Resources>
</Application>
```

3. Buka *MainWindow.xaml*.
   File XAML ini adalah jendela utama aplikasi Anda dan menampilkan konten yang dibuat. Kelas Window mendefinisikan properti jendela, seperti judul, ukuran, atau ikonnya, dan menangani *event*, seperti menutup(*close*) atau menyembunyikan(*hide*).

4. Lakukan modifikasi mada MainWindow.xaml ubah Window element menjadi NavigationWindow, seperti source XAML berikut ini:

```
<NavigationWindow x:Class="ExpenseIt.MainWindow"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
     ...
</NavigationWindow>
```

   Aplikasi ini menavigasi ke konten yang berbeda tergantung pada input pengguna. Inilah sebabnya mengapa Jendela utama perlu diubah menjadi NavigationWindow. NavigationWindow mewarisi semua properti Window. Elemen NavigationWindow dalam file XAML membuat turunan dari kelas NavigationWindow. Untuk informasi lebih lanjut, lihat Ikhtisar navigasi.

5. Hapus Grid elements pada tag NavigationWindow.
6. Ubah beberapa property berikut ini pada kode XAML untuk elemen NavigationWindow :
   - Set properti Title menjadi "`ExpenseIt`".
   - Set properti Height menjadi 350 pixels.
   - Set properti Width menjadi 500 pixels.

   Hasil kode XAMP menjadi seperti berikut ini :

```
<NavigationWindow x:Class="ExpenseIt.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
```

```
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="ExpenseIt" Height="350" Width="500">

    </NavigationWindow>
```

2. Buka *MainWindow.xaml.cs*.
   File ini adalah file kode yang berisi kode untuk menangani event yang dideklarasikan di MainWindow.xaml. File ini berisi kelas parsial untuk jendela yang didefinisikan dalam XAML.
3. Ubah kelas MainWindow untuk diturunkan dari NavigationWindow. Kode C #
   Anda sekarang akan terlihat seperti ini::

```
    using System;
    using System.Collections.Generic;
    using System.Linq;
    using System.Text;
    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 ExpenseIt
    {
        /// <summary>
        /// Interaction logic for MainWindow.xaml
        /// </summary>
        public partial class MainWindow : NavigationWindow
        {
            public MainWindow()
            {
                InitializeComponent();
```

```
            }
        }
    }
```

Tambahkan files pada application/project

Di bagian ini, Anda akan menambahkan dua halaman dan gambar ke aplikasi.

1. Tambahkan halaman baru ke proyek, dan beri nama ExpenseItHome.xaml:
    a. Di **Solution Explorer**, klik kanan pada project ExpenseIt dan pilih **Add → Page**.
    b. Dalam dialog **Add New Item**, pilih template Halaman (WPF). Masukkan nama ExpenseItHome, lalu pilih **Add**.

Halaman ini adalah halaman pertama yang ditampilkan ketika aplikasi diluncurkan. Ini akan menampilkan daftar orang untuk dipilih, untuk menunjukkan laporan pengeluaran.

2. Buka *ExpenseItHome.xaml*.

3. Set properti Title menjadi "`ExpenseIt - Home`".
4. Set properti `DesignHeight` menjadi 350 pixels dan `DesignWidth` menjadi 500 pixels.

Berikut source-code dalam C#:

```
    <Page x:Class="ExpenseIt.ExpenseItHome"
          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"
          mc:Ignorable="d"
          d:DesignHeight="350" d:DesignWidth="500"
        Title="ExpenseIt - Home">

        <Grid>

        </Grid>
    </Page>
```

5. Buka *MainWindow.xaml*.
6. Tambahkan Source property pada NavigationWindow element dan set menjadi "`ExpenseItHome.xaml`".

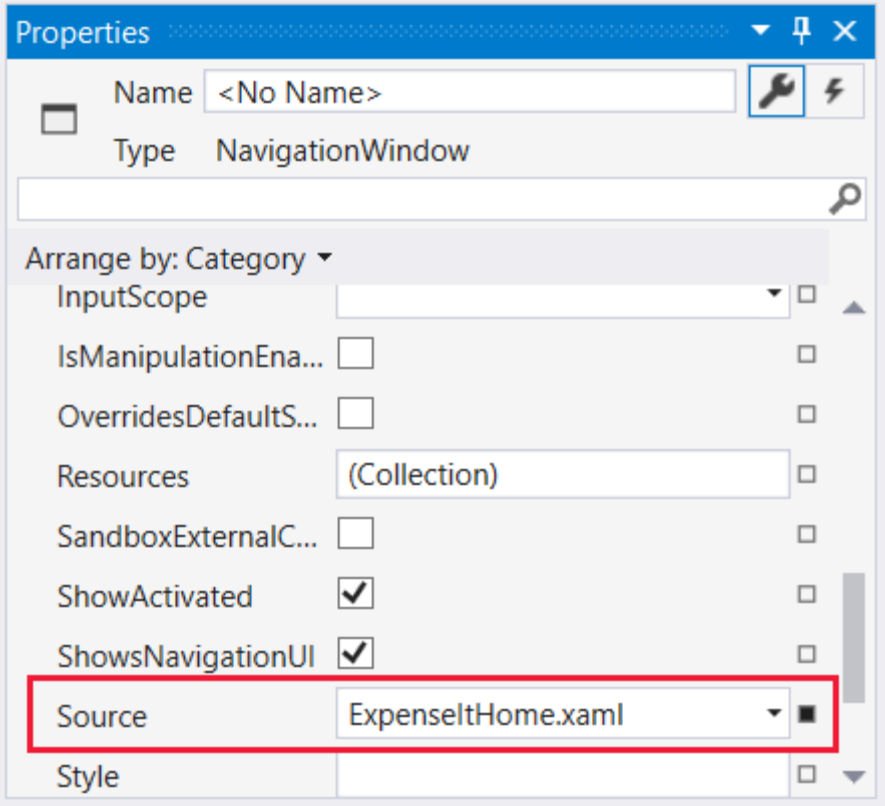
   (ini menjadikan *`ExpenseItHome.xaml`* halaman pertama yang dibuka ketika aplikasi dijalankan)

   Berikut source-code dalam C#:

```
<NavigationWindow x:Class="ExpenseIt.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="ExpenseIt" Height="350" Width="500" Source="ExpenseItHome.xaml">

</NavigationWindow>
```

Tip

Anda juga dapat mengatur properti pada jendela Properti.



7. Ulangi proses/langkah pada poin 6 untuk menambahkan halaman(page) WPF lagi pada project, dengan nama *ExpenseReportPage.xaml*::
8. Buka *ExpenseReportPage.xaml*.
9. Set properti Title to "`ExpenseIt - View Expense`".
10. Set properti `DesignHeight` menjadi 350 pixels dan `DesignWidth` menjadi 500 pixels. Berikut source-code dalam C#:

```
<Page x:Class="ExpenseIt.ExpenseReportPage"
      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"
      mc:Ignorable="d"
      d:DesignHeight="350" d:DesignWidth="500"
    Title="ExpenseIt - View Expense">

    <Grid>

    </Grid>
</Page>
```

11. Buka *ExpenseItHome.xaml.cs* dan *ExpenseReportPage.xaml.cs*.
    Saat Anda membuat file halaman baru, Visual Studio secara otomatis membuat file kode-nya. File kode ini menangani logika untuk merespons input pengguna. Kode Anda akan terlihat seperti berikut untuk ExpenseItHome:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
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 ExpenseIt
{
    /// <summary>
    /// Interaction logic for ExpenseItHome.xaml
    /// </summary>
    public partial class ExpenseItHome : Page
    {
        public ExpenseItHome()
        {
            InitializeComponent();
        }
    }
}
```

Dan untuk **ExpenseReportPage**:

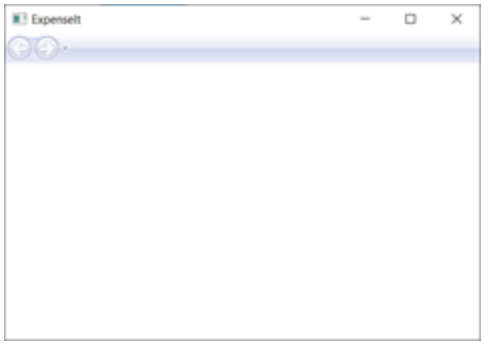
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;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

namespace ExpenseIt
{
```

```
        /// <summary>
        /// Interaction logic for ExpenseReportPage.xaml
        /// </summary>
        public partial class ExpenseReportPage : Page
        {
            public ExpenseReportPage()
            {
                InitializeComponent();
            }
        }
    }
```

12. Tambahkan gambar bernama watermark.png ke proyek. Anda dapat membuat gambar Anda sendiri.
    1. Klik kanan pada project **Add** > **Existing Item**, atau menggunakan shortcut **Shift**+**Alt**+**A**.
    2. Pada dialog **Add Existing Item** , set file filter menjadi **All Files** atau **Image Files**, browse ke direktori file image yang akan digunakan lalu klik **Add**.

Build dan menjalankan applikasi

1. To build and run the application, press **F5** or select **Start Debugging** from the **Debug** menu.
   The following illustration shows the application with the NavigationWindow buttons:



2. Close the application to return to Visual Studio.

Membuat Tata Letak(layout)

Layout pada WPF menyediakan cara untuk mengurutkan dan menempatkan elemen UI, dan juga mengatur ukuran dan posisi elemen tersebut ketika UI diubah ukurannya. Anda biasa membuat tata letak dengan salah satu kontrol tata letak berikut:

- **Canvas** - Menentukan area di mana Anda dapat secara eksplisit memposisikan elemen anak dengan menggunakan koordinat yang relatif terhadap area Kanvas.
- **DockPanel** - Menentukan area di mana Anda dapat mengatur elemen anak baik secara horizontal atau vertikal, relatif satu sama lain.
- **Grid** - Menentukan area grid yang fleksibel yang terdiri dari kolom dan baris.
- **StackPanel** - Mengatur elemen anak menjadi satu baris yang dapat diorientasikan secara horizontal atau vertikal.
- **VirtualizingStackPanel** - Mengatur dan memvirtualisasikan konten pada satu baris yang berorientasi baik secara horizontal maupun vertikal.
- **WrapPanel** - Memposisikan elemen anak dalam posisi berurutan dari kiri ke kanan, memecah konten ke baris berikutnya di tepi container. Urutan selanjutnya terjadi secara berurutan dari atas ke bawah atau dari kanan ke kiri, tergantung pada nilai properti Orientation.

Setiap kontrol tata letak ini mendukung tipe tata letak tertentu untuk elemen turunannya. Halaman ExpenseIt dapat diubah ukurannya, dan setiap halaman memiliki elemen yang disusun secara horizontal dan vertikal bersama elemen lainnya. Dalam contoh ini, Kotak digunakan sebagai elemen tata letak untuk aplikasi.

Tip

Untuk informasi lebih lanjut tentang elemen Panel, lihat Ikhtisar panel. Untuk informasi lebih lanjut tentang tata letak, lihat Tata Letak.

Di bagian ini, Anda membuat tabel kolom tunggal dengan tiga baris dan margin 10-pixel dengan menambahkan definisi kolom dan baris ke Grid di ExpenseItHome.xaml

1. Pada *ExpenseItHome.xaml*, set property Margin pada Grid element menjadi "10,0,10,10", yang sesuai dengan margin kiri, atas, kanan dan bawah:

```
<Grid Margin="10,0,10,10">
```

Tip

Anda juga dapat mengatur nilai Margin di jendela Properties, di bawah kategori Layout:

1. Taambahkan kode XAML berikut antara tag Grid untuk membuat devisnisi baris dan kolom:

```
<Grid.ColumnDefinitions>
    <ColumnDefinition />
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
    <RowDefinition Height="Auto"/>
    <RowDefinition />
    <RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
```

Ketinggian dua baris diatur ke Otomatis, yang berarti bahwa baris-baris tersebut diukur berdasarkan konten dalam baris. Tinggi standar adalah Ukuran bintang, yang berarti bahwa tinggi baris adalah proporsi tertimbang dari ruang yang tersedia. Misalnya, jika dua baris masing-masing memiliki Ketinggian "*", masing-masing memiliki ketinggian setengah dari ruang yang tersedia.

Grid Anda sekarang harus berisi XAML berikut:

```
<Grid Margin="10,0,10,10">
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto"/>
        <RowDefinition />
```

```
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>
    </Grid>
```

Menambahkan komponen controls

Di bagian ini, Anda akan memperbarui UI beranda(home) untuk menampilkan daftar orang, di mana Anda memilih satu orang untuk menampilkan laporan pengeluaran mereka. Kontrol adalah objek UI yang memungkinkan pengguna berinteraksi dengan aplikasi.

Untuk membuat UI ini, Anda akan menambahkan elemen berikut ke ExpenseItHome.xaml:

- ListBox (untuk daftar orang).
- Label (untuk header daftar).
- Button (untuk mengklik untuk melihat laporan pengeluaran untuk orang yang dipilih dalam daftar).

Setiap kontrol ditempatkan di baris Grid dengan mengatur Grid. Properti terlampir. Untuk informasi lebih lanjut tentang properti terlampir, lihat Ikhtisar Properti Terlampir.

1. In *ExpenseItHome.xaml*, add the following XAML somewhere between the Grid tags:

```
    <!-- People list -->
    <Border Grid.Column="0" Grid.Row="0" Height="35" Padding="5"
    Background="#4E87D4">
        <Label VerticalAlignment="Center" Foreground="White">Names</Label>
    </Border>
    <ListBox Name="peopleListBox" Grid.Column="0" Grid.Row="1">
        <ListBoxItem>Mike</ListBoxItem>
        <ListBoxItem>Lisa</ListBoxItem>
        <ListBoxItem>John</ListBoxItem>
        <ListBoxItem>Mary</ListBoxItem>
    </ListBox>
```

```
<!-- View report button -->
<Button Grid.Column="0" Grid.Row="2" Margin="0,10,0,0" Width="125" Height="25"
HorizontalAlignment="Right">View</Button>
```

Tip

Anda juga dapat membuat kontrol dengan menyeretnya dari jendela Toolbox ke jendela desain, dan kemudian mengatur propertinya di jendela Properties..

2. Build dan Jalankan Aplikasi.

Menambahkan Gambar dan title

Di bagian ini, Anda akan memperbarui UI beranda dengan gambar dan judul halaman.

1. Pada *`ExpenseItHome.xaml`*, tambahkan kolom pada ColumnDefinitions dengan fixed Width 230 pixels:

```
<Page x:Class="ExpenseIt.ExpenseItHome"
  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"
  mc:Ignorable="d"
  d:DesignHeight="350" d:DesignWidth="500"
Title="ExpenseIt - Home">

<Grid Margin="10,0,10,10">

    <Grid.Resources>

        <!-- Expense Report Data -->
        <XmlDataProvider x:Key="ExpenseDataSource" XPath="Expenses">
            <x:XData>
                <Expenses xmlns="">
                    <Person Name="Mike" Department="Legal">
```

```
                        <Expense ExpenseType="Lunch" ExpenseAmount="50" />
                        <Expense ExpenseType="Transportation" ExpenseAmount="50"
/>
                    </Person>
                    <Person Name="Lisa" Department="Marketing">
                        <Expense ExpenseType="Document printing"
                 ExpenseAmount="50"/>
                        <Expense ExpenseType="Gift" ExpenseAmount="125" />
                    </Person>
                    <Person Name="John" Department="Engineering">
                        <Expense ExpenseType="Magazine subscription"
                ExpenseAmount="50"/>
                        <Expense ExpenseType="New machine" ExpenseAmount="600" />
                        <Expense ExpenseType="Software" ExpenseAmount="500" />
                    </Person>
                    <Person Name="Mary" Department="Finance">
                        <Expense ExpenseType="Dinner" ExpenseAmount="100" />
                    </Person>
                  </Expenses>
              </x:XData>
          </XmlDataProvider>

          <!-- Name item template -->
          <DataTemplate x:Key="nameItemTemplate">
              <Label Content="{Binding XPath=@Name}"/>
          </DataTemplate>

      </Grid.Resources>

      <Grid.Background>
          <ImageBrush ImageSource="watermark.png"  />
      </Grid.Background>

      <Grid.ColumnDefinitions>
          <ColumnDefinition Width="230" />
          <ColumnDefinition />
      </Grid.ColumnDefinitions>
```

```
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition Height="Auto"/>
            <RowDefinition />
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- People list -->

        <Label Grid.Column="1" Style="{StaticResource headerTextStyle}" >
            View Expense Report
        </Label>

        <Border Grid.Column="1" Grid.Row="1" Style="{StaticResource
listHeaderStyle}">
            <Label Style="{StaticResource listHeaderTextStyle}">Names</Label>
        </Border>

        <ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2"
                 ItemsSource="{Binding Source={StaticResource ExpenseDataSource},
XPath=Person}"
                 ItemTemplate="{StaticResource nameItemTemplate}">
        </ListBox>

        <!-- View report button -->
        <Button Grid.Column="1" Grid.Row="3" Click="Button_Click"
Style="{StaticResource buttonStyle}">View</Button>


    </Grid>

</Page>
```

2. Tambahkan baris RowDefinitions, total ada 4 baris

```
<Page x:Class="ExpenseIt.ExpenseItHome"
  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"
  mc:Ignorable="d"
  d:DesignHeight="350" d:DesignWidth="500"
Title="ExpenseIt - Home">

<Grid Margin="10,0,10,10">

    <Grid.Resources>

        <!-- Expense Report Data -->
        <XmlDataProvider x:Key="ExpenseDataSource" XPath="Expenses">
            <x:XData>
                <Expenses xmlns="">
                    <Person Name="Mike" Department="Legal">
                      <Expense ExpenseType="Lunch" ExpenseAmount="50" />
                      <Expense ExpenseType="Transportation" ExpenseAmount="50"/>
                    </Person>
                    <Person Name="Lisa" Department="Marketing">
                        <Expense ExpenseType="Document printing"
             ExpenseAmount="50"/>
                        <Expense ExpenseType="Gift" ExpenseAmount="125" />
                    </Person>
                    <Person Name="John" Department="Engineering">
                        <Expense ExpenseType="Magazine subscription"
            ExpenseAmount="50"/>
                       <Expense ExpenseType="New machine" ExpenseAmount="600" />
                       <Expense ExpenseType="Software" ExpenseAmount="500" />
                    </Person>
                    <Person Name="Mary" Department="Finance">
                        <Expense ExpenseType="Dinner" ExpenseAmount="100" />
                    </Person>
                </Expenses>
```

```
            </x:XData>
        </XmlDataProvider>

        <!-- Name item template -->
        <DataTemplate x:Key="nameItemTemplate">
            <Label Content="{Binding XPath=@Name}"/>
        </DataTemplate>

    </Grid.Resources>

    <Grid.Background>
        <ImageBrush ImageSource="watermark.png"  />
    </Grid.Background>

    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="230" />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>

    <Grid.RowDefinitions>
        <RowDefinition/>
        <RowDefinition Height="Auto"/>
        <RowDefinition />
        <RowDefinition Height="Auto"/>
    </Grid.RowDefinitions>

    <!-- People list -->

    <Label Grid.Column="1" Style="{StaticResource headerTextStyle}" >
        View Expense Report
    </Label>

  <Border Grid.Column="1" Grid.Row="1" Style="{StaticResource listHeaderStyle}">
        <Label Style="{StaticResource listHeaderTextStyle}">Names</Label>
    </Border>

    <ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2"
```

```
            ItemsSource="{Binding Source={StaticResource ExpenseDataSource},
            XPath=Person}"
              ItemTemplate="{StaticResource nameItemTemplate}">
        </ListBox>

        <!-- View report button -->
        <Button Grid.Column="1" Grid.Row="3" Click="Button_Click"
        Style="{StaticResource buttonStyle}">View</Button>


    </Grid>

</Page>
```

3. Pindahkan kontrol ke kolom kedua dengan mengatur properti Grid. Column ke 1 di masing-masing dari tiga kontrol (Border, ListBox, dan Tombol).
4. Pindahkan setiap kontrol ke bawah satu baris dengan menambah grid. Nilai nilainya sebesar 1 untuk masing-masing dari tiga kontrol (Border, ListBox, dan Tombol) dan untuk elemen Border.

XAML untuk tiga kontrol sekarang terlihat seperti berikut:

```
    <Border Grid.Column="1" Grid.Row="1" Height="35" Padding="5"
    Background="#4E87D4">
          <Label VerticalAlignment="Center" Foreground="White">Names</Label>
      </Border>
      <ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2">
          <ListBoxItem>Mike</ListBoxItem>
          <ListBoxItem>Lisa</ListBoxItem>
          <ListBoxItem>John</ListBoxItem>
          <ListBoxItem>Mary</ListBoxItem>
      </ListBox>

      <!-- View report button -->
      <Button Grid.Column="1" Grid.Row="3" Margin="0,10,0,0" Width="125"
```

```
    Height="25" HorizontalAlignment="Right">View</Button>
```

5. Set properti **Panel.Background** ke file gambar watermark.png, dengan menambahkan XAML berikut di mana saja di antara tag <Grid> dan </Grid>:

```
    <Grid.Background>
        <ImageBrush ImageSource="watermark.png"/>
    </Grid.Background>
```

6. Sebelum elemen **border**, tambahkan **Label** dengan konten "Lihat Laporan Biaya". Label ini adalah judul halaman.

```
    <Label Grid.Column="1" VerticalAlignment="Center" FontFamily="Trebuchet MS"
           FontWeight="Bold" FontSize="18" Foreground="#0066cc">
        View Expense Report
    </Label>
```

7. Build dan Jalankan applikasi.

Ilustrasi berikut menunjukkan hasil dari apa yang baru saja Anda tambahkan:

Menambahkan code untuk menangani events

1. Pada *ExpenseItHome.xaml*, tambahan event Click untuk elemen Button.

```
      <!-- View report button -->
      <Button Grid.Column="1" Grid.Row="3" Margin="0,10,0,0" Width="125"
    Height="25" HorizontalAlignment="Right" Click="Button_Click">View</Button>
```

2. Buka *ExpenseItHome.xaml.cs*.
3. Tambahkan kode berikut ke kelas ExpenseItHome untuk menambahkan event klik pada tombol. Event ini akan membuka halaman ExpenseReportPage.

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    // View Expense Report
    ExpenseReportPage expenseReportPage = new ExpenseReportPage();
    this.NavigationService.Navigate(expenseReportPage);
}
```

Membuuat UI untuk ExpenseReportPage

ExpenseReportPage.xaml menampilkan laporan pengeluaran untuk orang yang dipilih pada halaman **ExpenseItHome**. Di bagian ini, Anda akan membuat UI untuk **ExpenseReportPage**. Anda juga akan menambahkan latar belakang dan mengisi warna ke berbagai elemen UI

1. Buka *ExpenseReportPage.xaml*.
2. Tambahkan code XAML berikut antara tag Grid:

```
<Grid.Background>
     <ImageBrush ImageSource="watermark.png" />
 </Grid.Background>
 <Grid.ColumnDefinitions>
     <ColumnDefinition Width="230" />
     <ColumnDefinition />
 </Grid.ColumnDefinitions>
 <Grid.RowDefinitions>
     <RowDefinition Height="Auto" />
     <RowDefinition />
 </Grid.RowDefinitions>


 <Label Grid.Column="1" VerticalAlignment="Center" FontFamily="Trebuchet MS"
 FontWeight="Bold" FontSize="18" Foreground="#0066cc">
     Expense Report For:
 </Label>
 <Grid Margin="10" Grid.Column="1" Grid.Row="1">
```

```
        <Grid.ColumnDefinitions>
            <ColumnDefinition />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition />
        </Grid.RowDefinitions>

        <!-- Name -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="0"
    Orientation="Horizontal">
            <Label Margin="0,0,0,5" FontWeight="Bold">Name:</Label>
            <Label Margin="0,0,0,5" FontWeight="Bold"></Label>
        </StackPanel>

        <!-- Department -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="1"
    Orientation="Horizontal">
            <Label Margin="0,0,0,5" FontWeight="Bold">Department:</Label>
            <Label Margin="0,0,0,5" FontWeight="Bold"></Label>
        </StackPanel>

        <Grid Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="2"
    VerticalAlignment="Top"
              HorizontalAlignment="Left">
            <!-- Expense type and Amount table -->
            <DataGrid  AutoGenerateColumns="False" RowHeaderWidth="0" >
                <DataGrid.ColumnHeaderStyle>
                    <Style TargetType="{x:Type DataGridColumnHeader}">
                        <Setter Property="Height" Value="35" />
                        <Setter Property="Padding" Value="5" />
                        <Setter Property="Background" Value="#4E87D4" />
                        <Setter Property="Foreground" Value="White" />
                    </Style>
```

```
                </DataGrid.ColumnHeaderStyle>
                <DataGrid.Columns>
                    <DataGridTextColumn Header="ExpenseType" />
                    <DataGridTextColumn Header="Amount"  />
                </DataGrid.Columns>
            </DataGrid>
        </Grid>
    </Grid>
```

UI ini mirip dengan ExpenseItHome.xaml, kecuali data laporan ditampilkan dalam DataGrid.

3. Build dan jalankan aplikasi.
4. Pilih tombol **View**.

Halaman laporan pengeluaran muncul. Juga perhatikan bahwa tombol navigasi belakang diaktifkan.

Ilustrasi berikut menunjukkan elemen UI yang ditambahkan pada xpenseReportPage.xaml.

Pengaturan Style

Tampilan berbagai elemen seringkali sama untuk semua elemen dengan tipe yang sama di UI. UI menggunakan style untuk membuat tampilan yang dapat digunakan kembali di berbagai elemen. Penggunaan kembali style membantu menyederhanakan pembuatan dan pengelolaan XAML. Bagian ini menggantikan atribut per elemen yang didefinisikan pada langkah sebelumnya dengan style.

1. Buka *App.xaml*.
2. Tambahkan code XAML berikut antara tag Application.Resources :

```
    <!-- Header text style -->
    <Style x:Key="headerTextStyle">
        <Setter Property="Label.VerticalAlignment" Value="Center"></Setter>
```

```
        <Setter Property="Label.FontFamily" Value="Trebuchet MS"></Setter>
        <Setter Property="Label.FontWeight" Value="Bold"></Setter>
        <Setter Property="Label.FontSize" Value="18"></Setter>
        <Setter Property="Label.Foreground" Value="#0066cc"></Setter>
    </Style>

    <!-- Label style -->
    <Style x:Key="labelStyle" TargetType="{x:Type Label}">
        <Setter Property="VerticalAlignment" Value="Top" />
        <Setter Property="HorizontalAlignment" Value="Left" />
        <Setter Property="FontWeight" Value="Bold" />
        <Setter Property="Margin" Value="0,0,0,5" />
    </Style>

    <!-- DataGrid header style -->
    <Style x:Key="columnHeaderStyle" TargetType="{x:Type DataGridColumnHeader}">
        <Setter Property="Height" Value="35" />
        <Setter Property="Padding" Value="5" />
        <Setter Property="Background" Value="#4E87D4" />
        <Setter Property="Foreground" Value="White" />
    </Style>

    <!-- List header style -->
    <Style x:Key="listHeaderStyle" TargetType="{x:Type Border}">
        <Setter Property="Height" Value="35" />
        <Setter Property="Padding" Value="5" />
        <Setter Property="Background" Value="#4E87D4" />
    </Style>

    <!-- List header text style -->
    <Style x:Key="listHeaderTextStyle" TargetType="{x:Type Label}">
        <Setter Property="Foreground" Value="White" />
        <Setter Property="VerticalAlignment" Value="Center" />
        <Setter Property="HorizontalAlignment" Value="Left" />
    </Style>

    <!-- Button style -->
```

```
<Style x:Key="buttonStyle" TargetType="{x:Type Button}">
    <Setter Property="Width" Value="125" />
    <Setter Property="Height" Value="25" />
    <Setter Property="Margin" Value="0,10,0,0" />
    <Setter Property="HorizontalAlignment" Value="Right" />
</Style>
```

XAML menambahkan style berikut ini:

- `headerTextStyle`: untuk menambahkan style pada **title.**
- `labelStyle`: untuk menambahkan style pada komponen control Label.
- `columnHeaderStyle`: untuk menambahkan style pada DataGridColumnHeader.
- `listHeaderStyle`: untuk menambahkan style pada list header Border.
- `listHeaderTextStyle`: untuk menambahkan style pada list header Label.
- `buttonStyle`: untuk menambahkan style pada Button pada `ExpenseItHome.xaml`.

Perhatikan bahwa style adalah resource dan turunan dari elemen properti Application.Resources. Di lokasi ini, style diterapkan ke semua elemen dalam aplikasi.

3. Pada *`ExpenseItHome.xaml`*, ganti semua code antara elemen Grid dengan code XAML berikut:

```
<Grid.Background>
    <ImageBrush ImageSource="watermark.png"  />
</Grid.Background>

<Grid.ColumnDefinitions>
    <ColumnDefinition Width="230" />
    <ColumnDefinition />
</Grid.ColumnDefinitions>

<Grid.RowDefinitions>
    <RowDefinition/>
    <RowDefinition Height="Auto"/>
    <RowDefinition />
    <RowDefinition Height="Auto"/>
```

```
    </Grid.RowDefinitions>

    <!-- People list -->

    <Label Grid.Column="1" Style="{StaticResource headerTextStyle}" >
        View Expense Report
    </Label>

    <Border Grid.Column="1" Grid.Row="1" Style="{StaticResource listHeaderStyle}">
        <Label Style="{StaticResource listHeaderTextStyle}">Names</Label>
    </Border>
    <ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2">
        <ListBoxItem>Mike</ListBoxItem>
        <ListBoxItem>Lisa</ListBoxItem>
        <ListBoxItem>John</ListBoxItem>
        <ListBoxItem>Mary</ListBoxItem>
    </ListBox>

    <!-- View report button -->
    <Button Grid.Column="1" Grid.Row="3" Click="Button_Click"
Style="{StaticResource buttonStyle}">View</Button>
```

Properti seperti VerticalAlignment dan FontFamily yang menentukan tampilan setiap kontrol dihapus dan diganti dengan menerapkan style. Misalnya, headerTextStyle diterapkan ke label "Lihat Laporan Biaya".

4. Buka *ExpenseReportPage.xaml*.
5. ganti semua code antara elemen Grid dengan code XAML berikut:

```
    <Grid.Background>
        <ImageBrush ImageSource="watermark.png" />
    </Grid.Background>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="230" />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto" />
        <RowDefinition />
```

```
    </Grid.RowDefinitions>


    <Label Grid.Column="1" Style="{StaticResource headerTextStyle}">
        Expense Report For:
    </Label>
    <Grid Margin="10" Grid.Column="1" Grid.Row="1">

        <Grid.ColumnDefinitions>
            <ColumnDefinition />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition />
        </Grid.RowDefinitions>

        <!-- Name -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="0"
Orientation="Horizontal">
            <Label Style="{StaticResource labelStyle}">Name:</Label>
            <Label Style="{StaticResource labelStyle}"></Label>
        </StackPanel>

        <!-- Department -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="1"
  Orientation="Horizontal">
            <Label Style="{StaticResource labelStyle}">Department:</Label>
            <Label Style="{StaticResource labelStyle}"></Label>
        </StackPanel>

        <Grid Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="2"
VerticalAlignment="Top"
              HorizontalAlignment="Left">
            <!-- Expense type and Amount table -->
            <DataGrid ColumnHeaderStyle="{StaticResource columnHeaderStyle}"
```

```
                    AutoGenerateColumns="False" RowHeaderWidth="0" >
                <DataGrid.Columns>
                    <DataGridTextColumn Header="ExpenseType" />
                    <DataGridTextColumn Header="Amount"  />
                </DataGrid.Columns>
            </DataGrid>
        </Grid>
    </Grid>
```

XAML ini menambahkan style ke elemen Label dan border.

6. Build dan jalankan aplikasi.

Binding data ke dalam komponen control

Di bagian ini, Anda akan membuat data XML yang di binding ke berbagai komponen control.

1. Pada *ExpenseItHome.xaml*, setelah elemen/tag pembuka Grid , tambahkan code XAML berikut untuk membuat XmlDataProvider yang terdiri dari data untuk masing-masing orang:

```
    <Grid.Resources>
        <!-- Expense Report Data -->
        <XmlDataProvider x:Key="ExpenseDataSource" XPath="Expenses">
            <x:XData>
                <Expenses xmlns="">
                    <Person Name="Mike" Department="Legal">
                        <Expense ExpenseType="Lunch" ExpenseAmount="50" />
                        <Expense ExpenseType="Transportation" ExpenseAmount="50" />
                    </Person>
                    <Person Name="Lisa" Department="Marketing">
                        <Expense ExpenseType="Document printing"
          ExpenseAmount="50"/>
                        <Expense ExpenseType="Gift" ExpenseAmount="125" />
                    </Person>
                    <Person Name="John" Department="Engineering">
```

```
                    <Expense ExpenseType="Magazine subscription"
          ExpenseAmount="50"/>
                    <Expense ExpenseType="New machine" ExpenseAmount="600" />
                    <Expense ExpenseType="Software" ExpenseAmount="500" />
                </Person>
                <Person Name="Mary" Department="Finance">
                    <Expense ExpenseType="Dinner" ExpenseAmount="100" />
                </Person>
            </Expenses>
        </x:XData>
    </XmlDataProvider>
</Grid.Resources>
```

Data dibuat sebagai resource Grid. Biasanya data ini akan dimuat sebagai file, tetapi pada matri ini data ditambahkan secara inline.

2. Tambahkan diantara elemen `<Grid.Resources>` , elemen code berikut `<xref:System.Windows.DataTemplate>` , yang mendefinisikan bagaimana menampilkan data dalam ListBox, setelah elemen `<XmlDataProvider>` :

```
<Grid.Resources>
    <!-- Name item template -->
    <DataTemplate x:Key="nameItemTemplate">
        <Label Content="{Binding XPath=@Name}"/>
    </DataTemplate>
</Grid.Resources>
```

3. Ganti ListBox dengan code XAML berikut ini:

```
<ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2"
         ItemsSource="{Binding Source={StaticResource ExpenseDataSource},
XPath=Person}"
         ItemTemplate="{StaticResource nameItemTemplate}">
</ListBox>
```

XAML ini mengikat properti ItemSource ListBox ke sumber data dan menerapkan templat data sebagai ItemTemplate.

Menghubungkan data ke komponen controls

Selanjutnya, akan dibahan menambahkan kode untuk mengambil nama yang dipilih pada halaman ExpenseItHome dan meneruskannya ke konstruktor ExpenseReportPage. ExpenseReportPage menetapkan konteks datanya dengan item yang diteruskan, yang mana yang didefinisikan oleh kontrol di ExpenseReportPage.xaml.

1. Buka *ExpenseReportPage.xaml.cs*.
2. Tambahkan konstruktor yang mengambil objek sehingga Anda dapat meneruskan data laporan pengeluaran orang yang dipilih.

```
public partial class ExpenseReportPage : Page
{
    public ExpenseReportPage()
    {
        InitializeComponent();
    }

    // Custom constructor to pass expense report data
    public ExpenseReportPage(object data):this()
    {
        // Bind to expense report data.
        this.DataContext = data;
    }
}
```

3. Buka *ExpenseItHome.xaml.cs*.
4. Ubah event handling untuk klik untuk memanggil constructor yang baru dan melakukan passing data expense report dari orang yang dipilih.

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    // View Expense Report
    ExpenseReportPage expenseReportPage = new
ExpenseReportPage(this.peopleListBox.SelectedItem);
    this.NavigationService.Navigate(expenseReportPage);
}
```

Style data with data templates

Di bagian ini, UI akan diperbarui untuk setiap item dalam daftar terikat data dengan menggunakan templat data.

1. Bula *ExpenseReportPage.xaml*.
2. Binding content "Name" dan "Department" elemen Label pada data source yang bersesuaian.

```
<!-- Name -->
<StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="0"
Orientation="Horizontal">
    <Label Style="{StaticResource labelStyle}">Name:</Label>
    <Label Style="{StaticResource labelStyle}" Content="{Binding
XPath=@Name}"></Label>
</StackPanel>

<!-- Department -->
<StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="1"
Orientation="Horizontal">
    <Label Style="{StaticResource labelStyle}">Department:</Label>
    <Label Style="{StaticResource labelStyle}" Content="{Binding
XPath=@Department}"></Label>
</StackPanel>
```

3. Setelah element/tag pembuka Grid, tambahkan data template berikut ini, data template mendefinisikan bagaimana menampilkan report:

```
<!--Templates to display expense report data-->
<Grid.Resources>
    <!-- Reason item template -->
    <DataTemplate x:Key="typeItemTemplate">
        <Label Content="{Binding XPath=@ExpenseType}"/>
    </DataTemplate>
    <!-- Amount item template -->
    <DataTemplate x:Key="amountItemTemplate">
        <Label Content="{Binding XPath=@ExpenseAmount}"/>
    </DataTemplate>
</Grid.Resources>
```

4. Replace elemen DataGridTextColumn dengan DataGridTemplateColumn dibawah elemen DataGrid dan terapkan template.

```
<!-- Expense type and Amount table -->
<DataGrid ItemsSource="{Binding XPath=Expense}" ColumnHeaderStyle="{StaticResource
columnHeaderStyle}" AutoGenerateColumns="False" RowHeaderWidth="0" >

    <DataGrid.Columns>
        <DataGridTemplateColumn Header="ExpenseType" CellTemplate="{StaticResource
        typeItemTemplate}" />
        <DataGridTemplateColumn Header="Amount" CellTemplate="{StaticResource
        amountItemTemplate}" />
    </DataGrid.Columns>

</DataGrid>
```

5. Build dan jalankan aplikasi