



书写人：陈志扬

## 一、项目分工

学号	名字	角色	班级	职责	贡献
15331046	陈志扬	组长	晚上班	负责创意设计，界面设计，个人信息、天气、手机号，项目报告、演示视频等	40%
15331351	杨涵	组员	晚上班	负责整个项目的主要设计（难点攻克），包括记账本的后台逻辑代码，就是我们的大佬攻城狮	50%
15331044	陈再兴	组员	晚上班	负责记账本后台数据	5%
14307004	蔡冠文	组员	晚上班	端茶倒水...	5%

## 二、开发环境

操作系统：Windows 10

开发工具：Visual Studio 2015

开发语言：C#、Xaml

## 三、项目阐述

项目名称：**记账本 Cashbook**

项目简介：为已注册登录的用户提供记账本、天气查询、手机号码归属地查询等功能，其中记账本是核心功能，天气查询、手机号码归属地查询、播放器属于附加小工具小功能。

项目功能：

- ① 用户可注册登录
- ② 每个用户可使用记账本功能为其消费收入做记录
- ③ 可根据城市查询该城市未来两天的天气
- ④ 可根据手机号码查询该手机号的归属地、运营商
- ⑤ 可选择媒体文件进行播放



项目亮点：界面简洁，体现出极简主义；功能完善，体现出完美主义。  
记账本功能基本实现，而且细节也都完美展现。

## 四、项目展示

① 程序运行后初始界面：默认为登录界面，上方登录按钮呈现形式如下：

为天蓝色，下方有一横杠，接下来是两个输入框分别为邮箱和密码，  
以及登录按钮

The screenshot shows a login interface. At the top, the word "Green" is displayed in a light blue font. Below it, the text "Change your world!" is centered. Underneath, there are two links: "注册" (Register) and "登录" (Login), with "登录" being underlined. Below the links are two input fields: the first is labeled "邮箱" (Email) and the second is labeled "密码" (Password). At the bottom, there is a large blue button labeled "登录" (Login).

② 若点击上方注册按钮，界面显示如下：为天蓝色，下方有一横杠，  
下来是三个输入框分别为昵称、邮箱、密码，以及注册按钮



Green

Change your world!

[注册](#) 登录

昵称

邮箱格式:xx@xx.com

密码至少为6位！

注册

③ 先注册一个账户，下面的截图是注册账户输入不合法的处理形式：



Green

Change your world!

注册

登录

昵称

昵称不能为空！

邮箱格式:xx@xx.com

密码至少为6位！

注册

Green

Change your world!

注册

登录

Young

123



格式错误！

密码至少为6位！

注册

Green

Change your world!

注册

登录

Young

123@qq.com

...



密码至少为6位！

注册

- ④ 按照上面所输的昵称和邮箱注册一个账户，邮箱不重复，则可注册成功，跳转到如下的个人主页界面



≡


👤


✎

🌐

🌐

📺



  
select

邮箱  
123@qq.com

昵称  
Young

修改密码  
修改密码

确认密码  
确认密码

确认修改退出登录

数据库中信息如下：

rowid	id	nickname	mail	password	imageUriString
(empty)	(empty)	(empty)	(empty)	(empty)	(empty)
1	320d211b-3ae2-4e2f-b274-ecf0e9578b83	Young	123@qq.com	123456	ms-appx:///Assets/background.jpg

⑤ 利用上面已注册的邮箱登录，也可看到如上界面

⑥ 如果修改昵称或者密码，可看到数据库更新，显示界面如下：

rowid	id	nickname	mail	password	imageUriString
(empty)	(empty)	(empty)	(empty)	(empty)	(empty)
1	320d211b-3ae2-4e2f-b274-ecf0e9578b83	Y	123@qq.com	111111	ms-appx:///Assets/background.jpg



≡


👤

✎

🌐

🌐

📺



select

修改密码成功！

关闭(C)

修改密码

.....

确认密码

.....

确认修改

退出登录

⑦ 现在假设注册邮箱重复(已被注册过), 弹出窗口告知用户该邮箱已被注册



Green

Change your world!

该邮箱已注册！请选择登录！

关闭(C)

123@qq.com

已被注册过的邮箱

.....

注册

⑧ 现在利用所注册的账户登录，输错邮箱或者密码，显示情况如下：



Green

Change your world!

注册

登录

1@qq.com



•••••

登录

账号或密码错误！请重新登录！

关闭(C)

⑨ 点击记账本，显示界面如下：





←

Cashbook

—

□

×

≡

记账本

2017年5月

0

支出 (元)

0

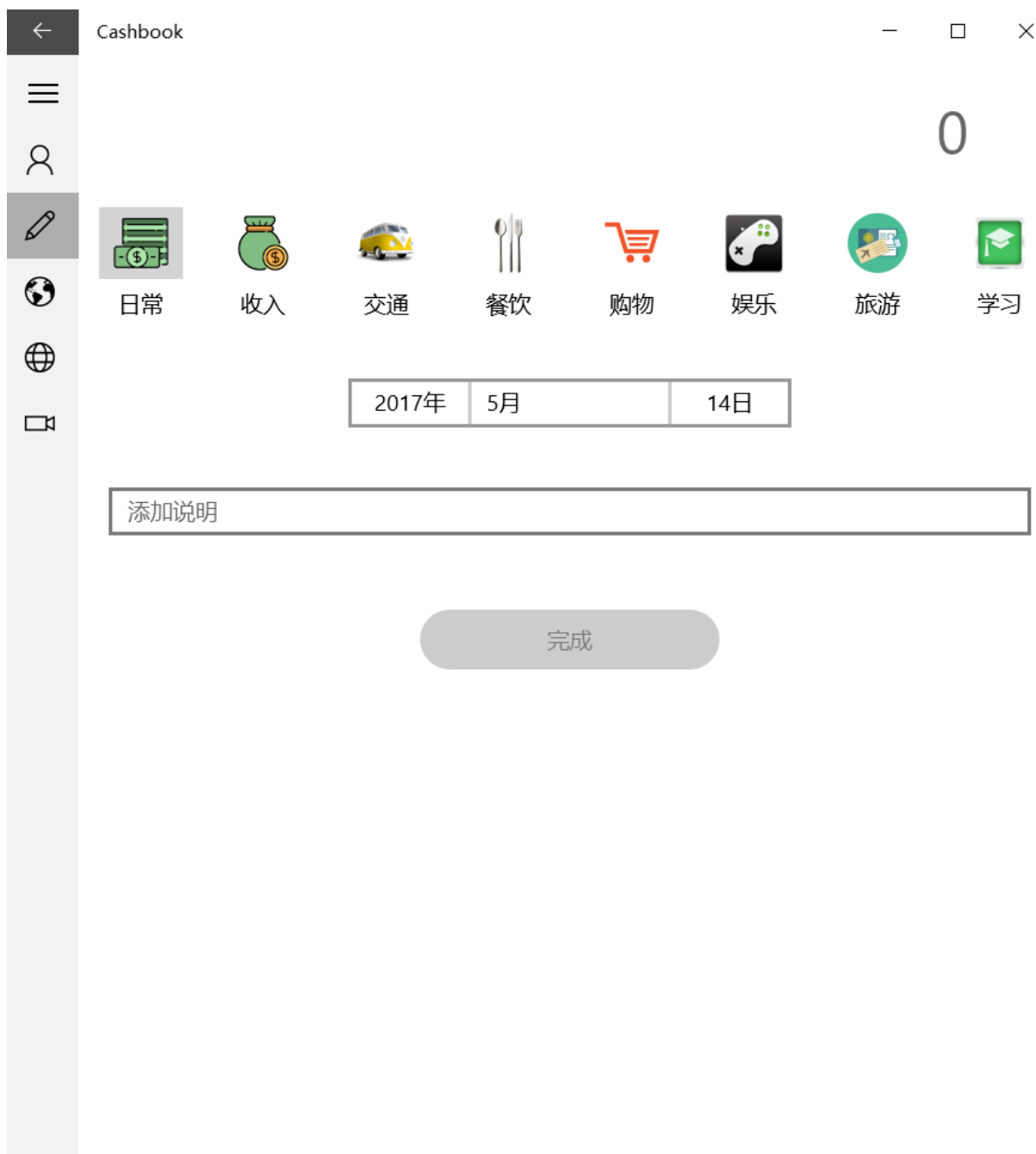
收入 (元)

0

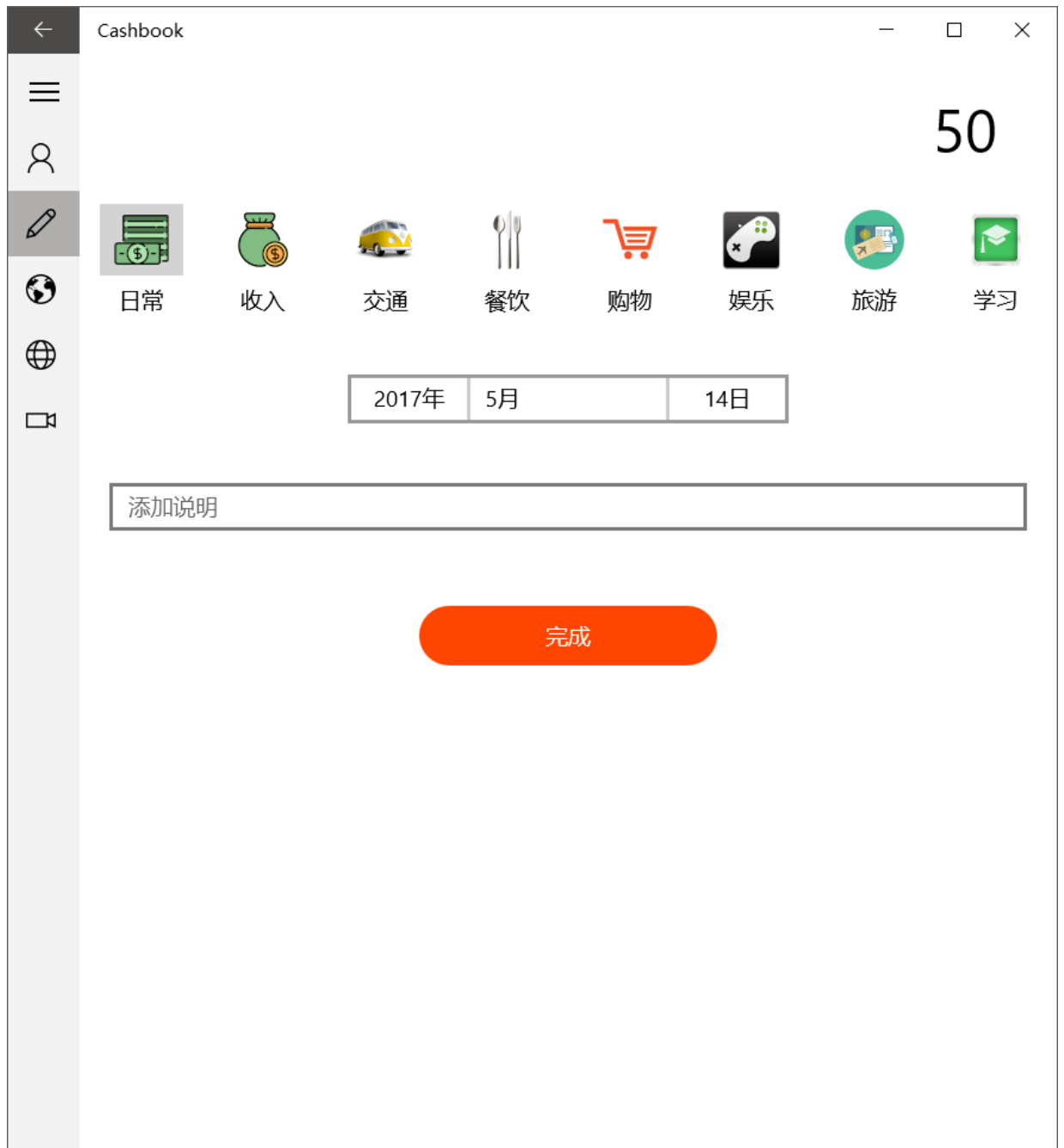
结余 (元)

记一笔

- ⑩ 点击记一笔，跳转到新建页面：默认选中消费类型“日常”，注意此时上方金额输入窗口为 0，下方完成按钮为灰色：



- ⑪ 在上方金额输入窗口输入一些数字，如 50，可见下方完成按钮为橘红色



- ⑫若我们更改消费类型，例如改为“交通”，此时“交通”按钮被选中，显示灰色，原来“日常”按钮灰色去掉，换为“白色”



≡

👤

✎

🌐

🌐

📺

50

💰

日常

💰

收入

🚗

交通

🍴

餐饮

🛒

购物

🎮

娱乐

🏠

旅游

🎓

学习

2017年

5月

14日

添加说明

完成

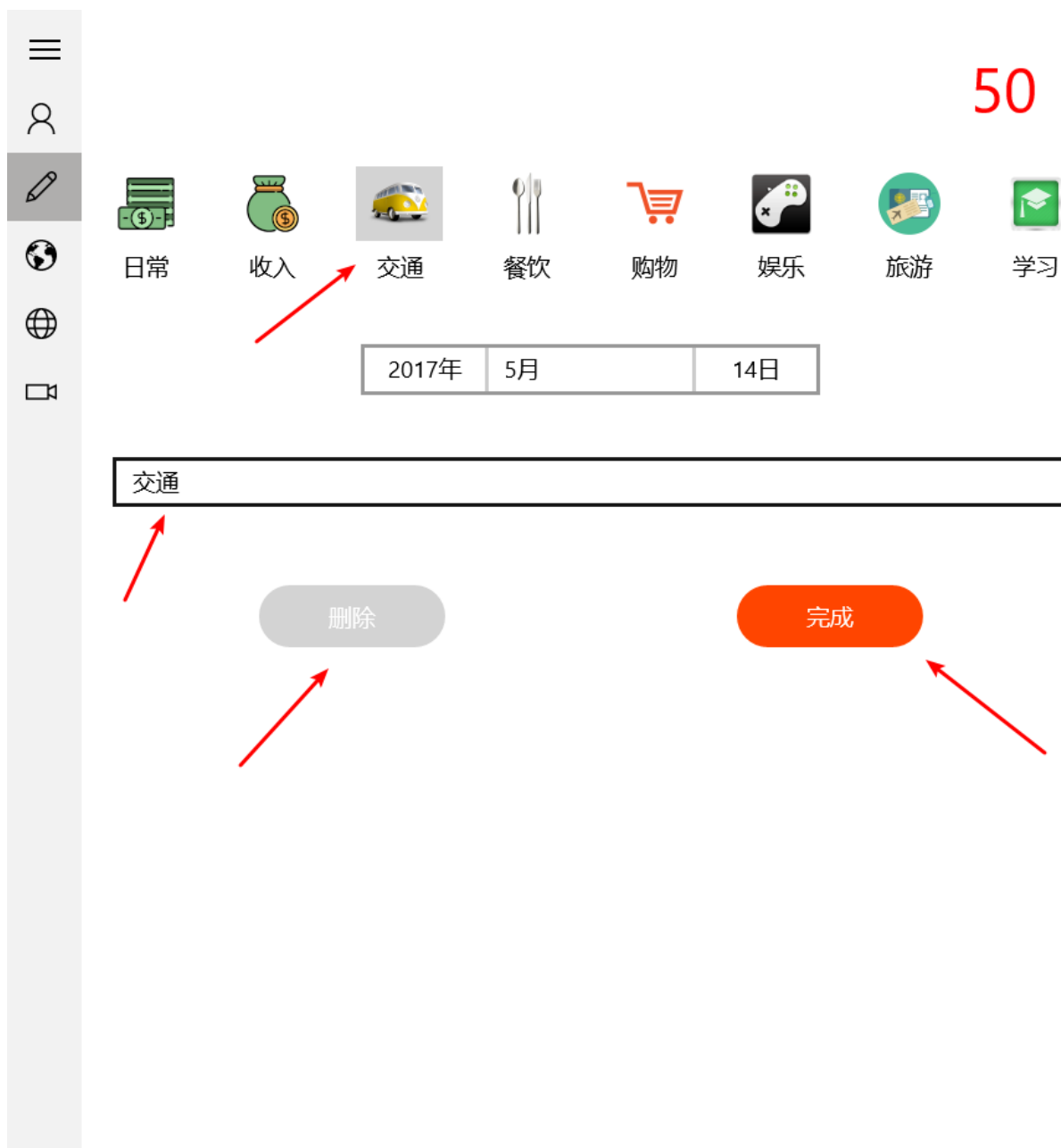
- ⑬按照上图点击完成按钮，跳转回记录页面，显示界面如下：注意到上方“支出”、“收入”、“结余”对应准确，记录日期也准确。



- ⑭再创建一个“收入”记录，例如收入 30 块，此时显示界面如下：可看到收入一栏显示为 30，结余一栏显示为-20，准确。



- ⑮ 接下来，点击“交通”这一记录进行修改，跳转到修改页面，显示如下：交通按钮为“灰色”，备注信息默认为消费类型，下方有两个按钮分别为删除和完成。



⑩对上图做修改，修改备注信息、日期、金额、消费类型，跳转回记录界面，显示如下：



- ⑪ 删除全部记录，跳转回记录界面，显示如下：支出、收入、结余都为 0.





⑱ 自适应 UI：小于 600 时如左，大于 600 时如右

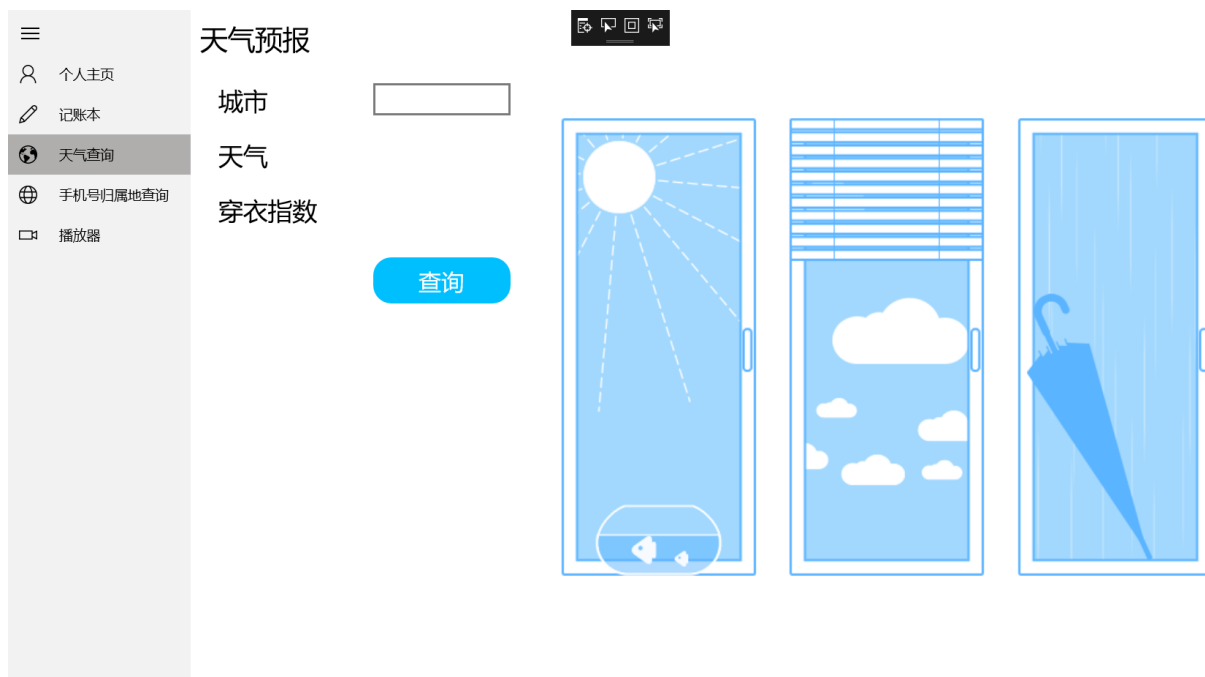


⑪ 点击左上角按钮可以展开 SplitView 的 Pane，展开如下：





⑳ 点击天气查询，如下：



21 左边三张图其实是动图，可以根据天气来选择显示哪张图，例如查询广州天气，如下图：



22. 点击手机号查询，可进行手机号码归属地查询。



☰

👤

✎

🌐

🌐

📺

手机号码归属地查询

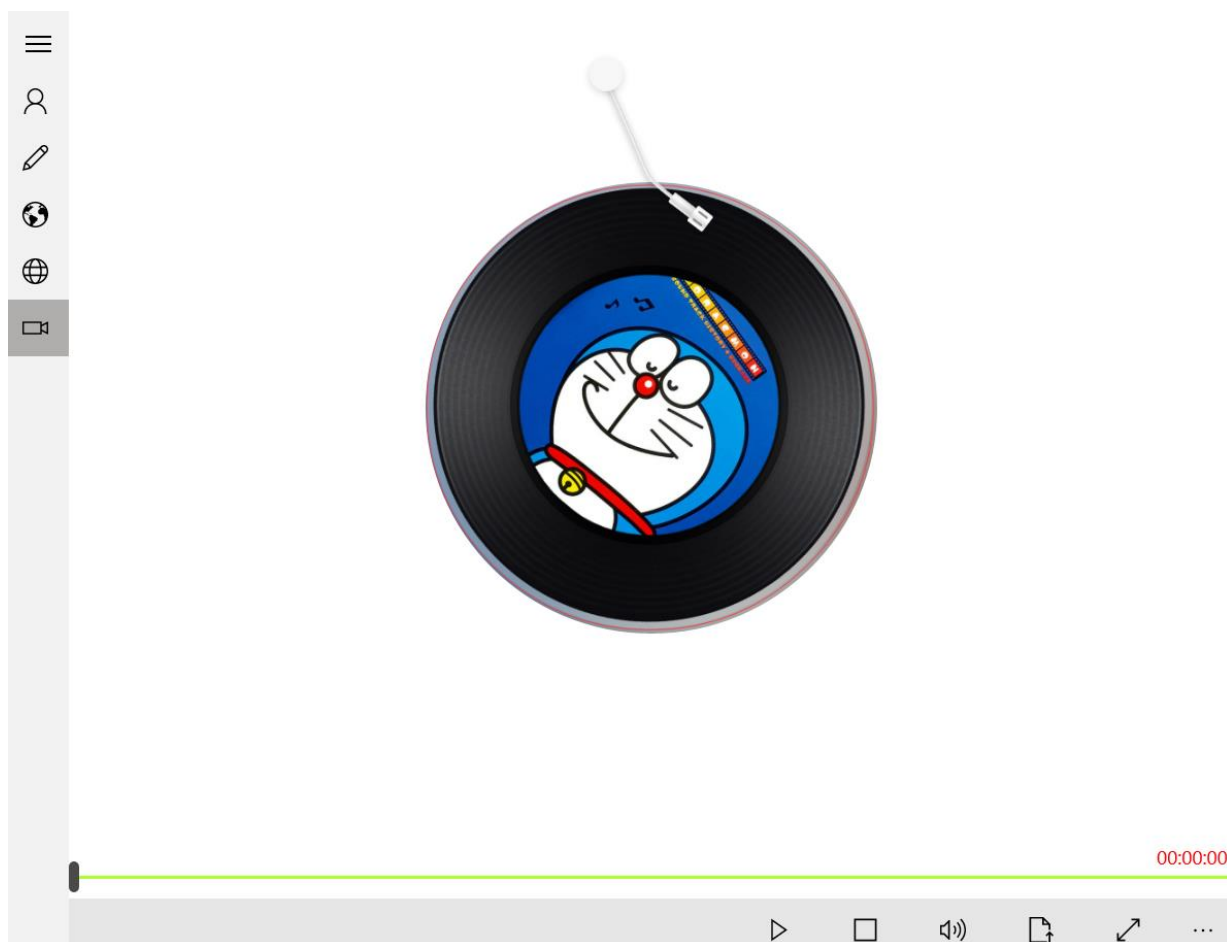
手机号码

归属地陕西宝鸡市

运营商中国陕西电信

查询

23. 点击播放器，可选择媒体文件播放。



24. 动态磁贴更新：两条记录



17 打工  
Wednesda

+40

15 食堂  
Monday

-20



## 五、项目难点及解决方案

- ① 汉堡菜单 SplitView 的实现。一开始就知道用 SplitView 可以实现类似的功能，以为微软应该会提供这方面的控件来方便开发者实现展开 Pane 的功能，没想到连那个点击三杠都要自己来设置样式才能实现，查找了很多资料，最后终于实现了，下面是关键代码（详细代码较多，这里不截出来）：

```
<Page.Resources>
    <local:NullableBooleanToBooleanConverter x:Key="converter"/>
    <local:FrameToIndexConverter x:Key="FrameToIndex"/>
    <!--设置SplitView样式-->
    <Style x:Key="SplitViewTogglePaneButtonStyle" TargetType="ToggleButton">
        <Setter Property="FontSize" Value="20" />
        <Setter Property="FontFamily" Value="{ThemeResource SymbolThemeFontFamily}" />
        <Setter Property="MinHeight" Value="48" />
        <Setter Property="MinWidth" Value="48" />
        <Setter Property="Margin" Value="0" />
        <Setter Property="Padding" Value="0" />
        <Setter Property="HorizontalAlignment" Value="Left" />
        <Setter Property="VerticalAlignment" Value="Top" />
        <Setter Property="HorizontalContentAlignment" Value="Center" />
        <Setter Property="VerticalContentAlignment" Value="Center" />
        <Setter Property="Background" Value="Transparent" />
    </Style>
</Page.Resources>
```



```
<SplitView x:Name="splitview" x:FieldModifier="Public"
    IsPaneOpen="{x:Bind list.IsChecked, Mode=TwoWay, Converter={StaticResource converter}}}"
    DisplayMode="CompactInline"
    CompactPanelLength="48"
    OpenPanelLength="200">
    <SplitView.Pane>
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="48" />
                <RowDefinition Height="*" />
                <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <!--顶端按钮，点击可展开Pane-->
            <ToggleButton x:Name="list" Width="48" Height="48" Padding="0, 0, 0, 0" BorderThickness="0" St
```

对应的转换代码：Converter.cs

```
// SplitView点击顶部展开Pane
class NullableBooleanToBooleanConverter : IValueConverter
{
    public object Convert(object value, Type targetType, object parameter, string language)
    {
        return (Boolean)value;
    }

    public object ConvertBack(object value, Type targetType, object parameter, string language)
    {
        if ((Boolean)value == true)
        {
            return true;
        } else
        {
            return false;
        }
    }
}
```

- ② 接下来是汉堡菜单内页面跳转导航的问题，在各个子页面间跳转时汉堡菜单不会随着跳转。可以通过绑定汉堡菜单（ListView），为每个ListView 设置一个 value 值，通过转换可以实现，关键代码如下：

Xaml 代码：

```
<!--设置该页为公共Public，并且绑定ListView，返回跳转时可对应各个listview-->
<Frame x:Name="frame" x:FieldModifier="Public" SourcePageType="{Binding ElementName=listview,
    Path=SelectedIndex, Converter={StaticResource FrameToIndex}, Mode=TwoWay}"></Frame>
```

对应的转换代码：Converter.cs



// 点击返回按钮ListView与返回到的界面相对应

```
class FrameToIndexConverter : IValueConverter
{
    public object Convert(object value, Type targetType, object parameter, string language)
    {
        if ((int)value == 0)
        {
            return typeof(ContactInfo);
        }
        else if ((int)value == 1)
        {
            return typeof(cashbook);
        }
        else if ((int)value == 2)
        {
            return typeof(weather);
        }
        else if ((int)value == 3)
        {
            return typeof(phonenumber);
        }
        else if ((int)value == 4)
        {
            return typeof(mediaplayer);
        }
        else
        {
            throw new NotImplementedException();
        }
    }

    public object ConvertBack(object value, Type targetType, object parameter, string language)
    {
        if ((Type)value == typeof(ContactInfo))
        {
            return 0;
        }
        else if ((Type)value == typeof(cashbook))
        {
            return 1;
        }
        else if ((Type)value == typeof(weather))
        {
            return 2;
        }
        else if ((Type)value == typeof(phonenumber))
        {
            return 3;
        }
        else if ((Type)value == typeof(mediaplayer))
        {
            return 4;
        }
        else
        {
            return 1;
        }
    }
}
```

③ 主界面 MainPage 点击上方登录注册按钮时下方的显示问题。因为我们



想做出知乎网站登录注册的效果，一开始只能把昵称先隐藏掉，待用户点击注册时才显示出来，但是这样做起来体验效果非常差，所以我们发现可以改每个 TextBox 的 Grid.Row 行号，那这样就很好办了，直接修改 TextBox 的行号，就能得到想要的效果。关键代码如下：

Xaml 代码：

```
</RelativePanel>
<TextBox Grid.Row="3" Visibility="Collapsed" x:Name="Nickname" Width="250" HorizontalAlignment="Center" VerticalAlignment="Center"
<TextBlock Grid.Row="4" Visibility="Collapsed" x:Name="check_Nickname" Width="250" Foreground="OrangeRed"/>
<TextBox Grid.Row="3" x:Name="Mail" Width="250" HorizontalAlignment="Center" VerticalAlignment="Center" Foreground="Gray" PlaceholderText="邮箱格式:xx@xx.com"
<TextBlock Grid.Row="4" x:Name="check_Mail" Width="250" Foreground="OrangeRed"/>
<PasswordBox Grid.Row="5" x:Name="Password" PlaceholderText="密码" Width="250" HorizontalAlignment="Center" VerticalAlignment="Center"
<TextBlock Grid.Row="6" x:Name="check_Password" Width="250" Foreground="OrangeRed"/>
<Button Grid.Row="7" x:Name="Sign" Content="登录" HorizontalAlignment="Center" Width="250" Background="DeepSkyBlue" Foreground="White"
rid>
```

MainPage.xaml.cs 代码：

```
// 点击上方注册按钮，控制显示界面
private void SignUp_Click(object sender, RoutedEventArgs e)
{
    Nickname.Text = "";
    Mail.Text = "";
    Password.Password = "";
    check_Nickname.Text = "";
    check_Mail.Text = "";
    check_Password.Text = "";
    Nickname.Visibility = Visibility.Visible;
    check_Nickname.Visibility = Visibility.Visible;
    Grid.SetRow(Mail, 5);
    Grid.SetRow(check_Mail, 6);
    Grid.SetRow(Password, 7);
    Grid.SetRow(check_Password, 8);
    Grid.SetRow(Sign, 9);
    SignUp.BorderThickness = new Thickness(0, 0, 0, 2);
    SignIn.BorderThickness = new Thickness(0);
    SignUp.Foreground = new SolidColorBrush(Colors.DeepSkyBlue);
    SignIn.Foreground = new SolidColorBrush(Colors.Gray);
    Sign.Content = "注册";
    Mail.PlaceholderText = "邮箱格式:xx@xx.com";
    Password.PlaceholderText = "密码至少为6位! ";
}
```





// 点击上方登录按钮，控制显示界面

```
private void SignIn_Click(object sender, RoutedEventArgs e)
{
    Nickname.Text = "";
    Mail.Text = "";
    Password.Password = "";
    check_Nickname.Text = "";
    check_Mail.Text = "";
    check_Password.Text = "";
    Nickname.Visibility = Visibility.Collapsed;
    check_Nickname.Visibility = Visibility.Collapsed;
    Grid.SetRow(Mail, 3);
    Grid.SetRow(check_Mail, 4);
    Grid.SetRow(Password, 5);
    Grid.SetRow(check_Password, 6);
    Grid.SetRow(Sign, 7);
    SignIn.BorderThickness = new Thickness(0, 0, 0, 2);
    SignUp.BorderThickness = new Thickness(0);
    SignUp.Foreground = new SolidColorBrush(Colors.Gray);
    SignIn.Foreground = new SolidColorBrush(Colors.DeepSkyBlue);
    Sign.Content = "登录";
    Mail.PlaceholderText = "邮箱";
    Password.PlaceholderText = "密码";
}
```

- ④ 登录成功后进入个人主页时无法显示出对应的信息，比如昵称、邮箱等。解决办法：应传入 MainPage 用于登录的邮箱，然后通过数据库查询邮箱对应的昵称，关键代码如下：

```
if (e.Parameter.GetType() == typeof(string))
{
    mail = (string)(e.Parameter);
    Model = ViewModel.queryMail(mail);
    Mail.Text = Model.mail;
    Nickname.Text = Model.nickname;
    BitmapImage bi = new BitmapImage();
    bi.UriSource = Model.imageUri;
    Picture.ImageSource = bi;
}
```



```

if (ApplicationData.Current.LocalSettings.Values.ContainsKey("TheWorkInProgress"))
{
    var composite = ApplicationData.Current.LocalSettings.Values["TheWorkInProgress"] as ApplicationDataCompositeValue;
    Nickname.Text = (string)composite["nickname"];
    Mail.Text = (string)composite["mail"];
    string s = (string)composite["imageUri"];
    BitmapImage bi = new BitmapImage();
    bi.UriSource = new Uri(s);
    Picture.ImageSource = bi;
    // We're done with it, so remove it
    ApplicationData.Current.LocalSettings.Values.Remove("TheWorkInProgress");
}
}

protected override void OnNavigatingFrom(NavigatingCancelEventArgs e)
{
    var composite = new ApplicationDataCompositeValue();
    composite["nickname"] = Nickname.Text;
    composite["mail"] = Mail.Text;
    composite["imageUri"] = Model.imageUri.ToString();
    //var i = new MessageDialog(Model.imageUri.ToString()).ShowAsync();
    ApplicationData.Current.LocalSettings.Values["TheWorkInProgress"] = composite;
}

```

- ⑤ 记账本显示日期和星期的问题。需要绑定后台数据 dayOfMouth、dayOfWeek 来实现

```

<StackPanel Grid.Column="0" Margin="0, 0, 0, 0">
    <TextBlock Text="{x:Bind dayOfMouth}" FontSize="15" FontWeight="Bold" HorizontalAlignment="Center"/>
    <TextBlock Text="{x:Bind dayOfWeek}" FontSize="10" Foreground="Gray" HorizontalAlignment="Center"/>
</StackPanel>

```

```

public System.DateTimeOffset day; //记录的日期
public string dayOfMouth; // 记录当天是几号
public string dayOfWeek; // 记录当天星期几

this.REMARKS = REMARKS,
this.day = day;
this.dayOfMouth = day.Day.ToString(); // 获取记录当天是几号
this.dayOfWeek = day.DayOfWeek.ToString(); // 获取记录当天星期几

```

- ⑥ 记账本记录界面上方如何显示支出、收入、结余的问题。解决办法：  
通过读取数据库中每条记录的金额进行计算，然后显示出来。关键代码如下：

cashbook.xaml.cs 中：



```
// 从数据库中获取每条消费收入记录，并计算出支出、收入、结余显示在页面上
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    if (e.Parameter != null)
    {
        if (e.Parameter.GetType() == typeof(ViewModels.DocumentViewModel))
        {
            this.ViewModel = (ViewModels.DocumentViewModel)(e.Parameter);
        }
    }
    payout.Text = ViewModel.payout.ToString();
    income.Text = ViewModel.income.ToString();
    result.Text = ViewModel.result.ToString();
}
```

DocumentViewModel.cs 中：

```
// 从数据库读取消费收入记录并计算支出、收入、结余
if (document.consumptionType == "收入")
{
    income += document.amountOfMoney;
} else
{
    payout -= document.amountOfMoney;
}
this.allDocuments.Add(document);
```

⑦ 点击一个记录跳转到详细信息界面，处理点击消费类型按钮时的视觉效果，因为我们的想法是实现点击一个记录跳转后选中其消费类型，然后若点击其他按钮更改消费类型时，原来选中的取消选中（也就是背景变成白色），当前选中的变为灰色，在这个看似简单的逻辑中我们尝试了很久才搞定，具体的解决办法是在 cashbookInfo.xaml.cs 文件中创建一个 Button 类的实例来记录最近一次点击的按钮，具体代码如下：

```
Button last; // 用来记录用户最近一次点击的消费类型
```



// 接下来八个Click事件都是类似的  
// 在该页面下，用户随机点击消费类型的每个按钮，记录下最近一次的点击，并更新消费类型和备注信息  
// 难点是处理点击按钮前后，上一次点击的按钮呈白色，新点击的按钮呈浅灰色

```
private void payout_Click(object sender, RoutedEventArgs e)
{
    last.Background = new SolidColorBrush(Colors.White);
    updateConsumption = "日常";
    info.Text = "日常";
    payout.Background = new SolidColorBrush(Colors.LightGray);
    last = (Button)e.OriginalSource;
}
```

```
private void income_Click(object sender, RoutedEventArgs e)
{
    last.Background = new SolidColorBrush(Colors.White);
    updateConsumption = "收入";
    info.Text = "收入";
    income.Background = new SolidColorBrush(Colors.LightGray);
    last = (Button)e.OriginalSource;
}
```

- ⑧ 根据用户是否在金额输入框中是否输入信息，来决定完成按钮是否可起作用

```
// 根据TextChanged可以更改“完成”按钮的颜色，形成视觉效果，且点击无效
private void money_TextChanged(object sender, TextChangedEventArgs e)
{
    if (money.Text == "") complete.IsEnabled = false;
    else complete.IsEnabled = true;
}
```

- ⑨ 接下来是播放器全屏后仍然显示 SplitView 的问题，这里可以设置 SplitView 为 Public，然后在 MediaPlayer.xaml.cs 中修改其 CompactPaneLength，关键代码如下：

```
<SplitView x:Name="splitview" x:FieldModifier="Public"
    IsPaneOpen="{x:Bind list.IsChecked, Mode=TwoWay, Converter={StaticResource converter}}"
    DisplayMode="CompactInline">
```



```
private void FullScreen_Click(object sender, RoutedEventArgs e)
{
    ApplicationView view = ApplicationView.GetForCurrentView();

    bool isInFullScreenMode = view.IsFullScreenMode;

    if (isInFullScreenMode)
    {
        if (!myMediaElement.IsAudioOnly) Grid.SetRowSpan(myMediaElement, 1);
        view.ExitFullScreenMode();
        Information.current.splitview.CompactPaneLength = 48;
    }
    else
    {
        if (!myMediaElement.IsAudioOnly) Grid.SetRowSpan(myMediaElement, 2);
        view.TryEnterFullScreenMode();
        Information.current.splitview.CompactPaneLength = 0;
    }
}
```

- ⑩ 另外，就是两个数据库的处理逻辑了，记账本的记录信息主要是需要判断消费类型来显示支出、收入、结余，账户登录的数据库主要涉及查询、修改等。
- ⑪ 整个项目涉及到很多数据绑定，这里不再一一赘述；尽量完善界面的美观性也是一件很费时的事，同样不再赘述。

## 六、项目总结

先说说我们整个项目从创意的提出到实现的流程吧。

当时 TA 在第三四周时刚提到期中项目时，我的脑海里就蹦出做“记账本”的 Idea，因为我感觉做记账本挺有实用性，在当今我们大学生的消费观念如此“奔放”的情况下，有必要记录一下平时消费记录或者收入记录。

和小组组员再三确认好我们的想法后，已经来到第九周了，需要交一份项目策划书，当时又处于实训中，所以项目设计策划书也只能先把做“记账本”的 Idea 大概体现在策划书中，具体细节也没法多透漏。

很快，时间来到十二周了，而我们的项目才刚刚开始。做完现操的游戏





作业后，大概是从十二周周二下午，我们启动项目的代码编写。

首先确认一下分工，我负责注册登录的界面和登录之后的个人主页显示界面，杨涵做汉堡菜单界面；与此同时，根据之前讨论过的记账本的几种数据类型：消费类型（string）、消费金额（double）、消费备注记录（string）、消费记录日期（DateTimeOffset），数据部分和消费记录存数据库部分交由再兴同学编写，但是后期我们使用数据库时发现很多地方需要更改添加。

接着，在我移植天气查询和手机号码归属地查询这两部分的 Xaml 和 C# 代码后，杨涵同学也差不多做好了汉堡单菜单界面。

然后，根据事先设计好的记账本 UI，杨涵同学先编写每条消费记录的显示信息以及支出、收入、结余三个 TextBlock 显示界面；我负责点击每条消费记录跳转到消费记录的详细信息显示界面以及新建一条消费记录的显示界面。

接下来我负责用户注册登录的后台逻辑代码（包括个人信息数据库），杨涵同学负责记账本的后台逻辑代码，包括数据库的增删查改，前端界面显示的更新，以及各种细节操作的体现。

最后，就是缝缝补补了。主要是界面的自适应、动态磁贴的显示、（页面跳转）返回按钮对应汉堡菜单，以及修复各种 bug 等。

**接下来说说我们做完整个项目的心得体会。**

首先，做完整个项目当然是极其开心，“爽”字来形容最恰当不过了，但其中的艰难困苦真的是难以描述。

整个过程中遇到的问题非常多，我觉得最主要的原因是“追求极致”，每个方面都要求严格，每个细节都要求完美，比如 xaml 中边框的距离的调整，点击消费类型时的视觉效果，以及每个按钮的颜色和点击视觉效果。在逻辑



功能实现方面，又追求尽量少 bug，存取数据库也都要实现，用户注册登录逻辑判断要完善等。

再者，小组合作的问题其实很严重，各个组员的时间基本无法统一，而缺少对方代码时自己的代码又很难实现，而且沟通交流也很不方便；另外一个原因，是有的组员积极性不是很高，整个项目下来基本没做什么工作。还有，代码移植在我们小组合作时很不方便，首先要统一文件命名、变量命名，而且有时候对方修改了代码导致自己的代码运行报错，总之，小组合作的项目最后基本成了我和杨涵的“二人转”表演。

最近我也读了很多关于产品设计思考的书，深谙极简主义、完美主义之道，在本项目中，我对此非常执着，界面尽量简洁，各个按钮视觉效果要完美体现，功能尽量完善，做到少 bug。

本来以为“记账本”这个 Idea 的实现不是很难，但通过整个项目的实现，才知道里面的东西很复杂，具体包括 UI 界面（汉堡菜单、自适应、页面间返回跳转），“记账本”信息的显示，“记账本”消费记录的创建、修改、删除的视觉效果，“记账本”数据库相关处理，用户注册登录 UI 和逻辑代码，“用户个人信息”数据库相关处理，而且也基本囊括了我们所学内容知识：Adaptive UI、Data Binding、DataBase、App to app communication、Network accessing、File management、Live Tiles 这七部分的知识。

看似简单的“记账本”，如果只是实现比较简陋的逻辑功能，那确实简单了一些，但考虑到界面的美观性，功能的强大性，一切就非同寻常了，背后隐藏了很多知识，需要我们学习很多来实现每一个功能，完善每一个细节。这也深深地体现出程序员工程师的智慧和辛劳，在这里向广大程序员表示敬意！