Приложение А.

```
use master
go
create database [KP_Saulin_INCOMSYSTEM];
go
use [KP_Saulin_INCOMSYSTEM]
go
create table Positions(
id tinyint primary key identity(1,1),
[name] nvarchar(75) not null
);
create table UsersDetail(
idUser bigint primary key identity(1,1),
[login] nvarchar(75) not null,
[password] nvarchar(75) not null,
phone bigint null check((len(phone) = 11) OR phone IS NULL),
idPos tinyint not null,
passport bigint not null check(len(passport) = 10) unique,
[address] nvarchar(175) not null,
dateStart datetime not null default(getdate()),
foreign key(idPos) references Positions(id) on delete cascade on update cascade
);
create table Specializations(
id int primary key identity(1,1),
[name] nvarchar(75) not null
);
create table LegalForms(
id int primary key identity(1,1),
[name] nvarchar(75) not null
);
create table Customers(
idUser bigint primary key,
[name] nvarchar(150) not null,
idLegalForm int not null,
foreign key(idUser) references UsersDetail(idUser) on delete cascade on update cascade,
foreign key(idLegalForm) references LegalForms(id) on delete cascade on update cascade
);
create table Employees(
idUser bigint primary key,
[surname] nvarchar(75) not null,
[name] nvarchar(75) not null,
[patronymic] nvarchar(75) null,
foreign key(idUser) references UsersDetail(idUser) on delete cascade on update cascade
);
create table SpecializationsEmployee(
id bigint primary key identity(1,1),
idEmployee bigint not null,
idSpecialization int not null,
foreign key(idEmployee) references Employees(idUser) on delete cascade on update cascade,
foreign key(idSpecialization) references Specializations(id) on delete cascade on update cascade
);
create table Tasks(
id bigint primary key identity(1,1),
[name] nvarchar(75) not null,
idSpecialization int not null,
[description] nvarchar(max) not null,
price decimal(18,2) not null,
discount tinyint null,
approxCompleteTime int not null,
supportPeriod int not null,
attachment varbinary(max) null,
fileExtension nvarchar(10) null,
foreign key(idSpecialization) references Specializations(id) on delete cascade on update cascade
);
create table Statuses(
id tinyint primary key identity(1,1),
[name] nvarchar(75) not null
);
create table Orders(
id bigint primary key identity(1,1),
idCustomer bigint not null,
idExecutor bigint null,
idTask bigint not null,
price decimal(18,2) not null,
difficulty float not null default(1),
dateOrder datetime not null,
planDateStart datetime null,
factDateStart datetime null,
planDateComplete datetime null,
factDateComplete datetime null,
attachment varbinary(max) null,
fileExtension nvarchar(10) null,
idStatus tinyint not null,
foreign key(idCustomer) references Customers(idUser),
foreign key(idExecutor) references Employees(idUser),
foreign key(idTask) references Tasks(id) on delete cascade on update cascade,
foreign key(idStatus) references Statuses(id) on delete cascade on update cascade
);
create table Chats(
idChat bigint primary key,
idCustomer bigint not null,
idManager bigint null,
dateCreate datetime not null,
foreign key(idCustomer) references Customers(idUser),
foreign key(idManager) references Employees(idUser),
foreign key(idChat) references Orders(id) on delete cascade on update cascade
);
create table [Messages](
id bigint primary key identity(1,1),
idChat bigint not null,
idUser bigint not null,
[message] text null,
attachment varbinary(max) null,
fileExtension nvarchar(10) null,
dateSend datetime not null default(getdate()),
foreign key(idChat) references Chats(idChat),
foreign key(idUser) references UsersDetail(idUser),
);
insert into Positions ([name]) VALUES (N'Заказчик');
insert into Positions ([name]) VALUES (N'Исполнитель');
insert into Positions ([name]) VALUES (N'Менеджер');
insert into Statuses ([name]) VALUES (N'Ожидание');
insert into Statuses ([name]) VALUES (N'Обсуждение');
insert into Statuses ([name]) VALUES (N'Выполнение');
insert into Statuses ([name]) VALUES (N'Выполнено');
insert into Statuses ([name]) VALUES (N'Отклонён');
insert into LegalForms ([name]) VALUES (N'Физическое лицо');
insert into LegalForms ([name]) VALUES (N'Юридическое лицо');
```

Приложение В.

AuthWindow.xaml

```
<Window
x:Class="INCOMSYSTEM.Windows.AuthWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d"
AllowsTransparency="True"
WindowStyle="None"
ResizeMode="NoResize"
WindowStartupLocation="CenterScreen"
Height="475"
Width="600"
Background="DimGray"
BorderBrush="Black"
BorderThickness="2">
<WindowChrome.WindowChrome>
<WindowChrome CaptionHeight="0"
GlassFrameThickness="0"
NonClientFrameEdges="None"
ResizeBorderThickness="5"/>
</WindowChrome.WindowChrome>
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
</Grid.RowDefinitions>
<controls:TopPanelControl/>
<Frame x:Name="AFrame"
Grid.Row="1"
Margin="5"
IsTabStop="False"
NavigationUIVisibility="Hidden"/>
</Grid>
</Window>
```

AuthWindow.xaml.cs

```
using System.Windows.Controls;
using System.Windows.Navigation;
using INCOMSYSTEM.Pages;
namespace INCOMSYSTEM.Windows
{
public partial class AuthWindow
{
public AuthWindow()
{
InitializeComponent();
AuthFrame = AFrame;
AuthFrame.Navigated += AuthFrameOnNavigated;
AuthFrame.Navigate(new AuthPage());
}
private void AuthFrameOnNavigated(object sender, NavigationEventArgs e)
{
var page = ((Frame)sender).Content as Page;
this.Title = page?.Title;
}
public static Frame AuthFrame { get; private set; }
}
}
```

MainWindow.xaml

```
<Window
x:Class="INCOMSYSTEM.Windows.MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d"
AllowsTransparency="True"
WindowStyle="None"
WindowStartupLocation="CenterScreen"
Height="600"
MinHeight="600"
Width="950"
MinWidth="950"
WindowState="Maximized"
Background="DimGray"
BorderBrush="Black"
BorderThickness="2">
<WindowChrome.WindowChrome>
<WindowChrome CaptionHeight="0"
GlassFrameThickness="0"
NonClientFrameEdges="None"
ResizeBorderThickness="5"/>
</WindowChrome.WindowChrome>
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="50"/>
<ColumnDefinition/>
</Grid.ColumnDefinitions>
<controls:TopPanelControl
Grid.Column="0" Grid.ColumnSpan="2"/>
<Grid Grid.Row="1" Grid.Column="0"
Grid.ColumnSpan="2">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<StackPanel Orientation="Horizontal"
Margin="55 0">
<Button x:Name="BackBtn"
Content="&lt;-"
Click="BackBtn_Click"/>
<Button x:Name="BackChatBtn"
Content="&lt;-"
Visibility="Collapsed"
IsEnabled="False"
Click="BackChatBtn_Click"/>
</StackPanel>
<StackPanel Grid.Column="2"
Orientation="Horizontal">
<TextBlock x:Name="AuthBlock"
VerticalAlignment="Center"
Style="{StaticResource TextBlockStyle}"/>
<Image Source="../Images/user.png"
Width="32"
Height="32"
Margin="10 0 20 0"/>
</StackPanel>
</Grid>
<Frame x:Name="MFrame"
Grid.Column="1"
Grid.Row="2"
Margin="5"
IsTabStop="False"
NavigationUIVisibility="Hidden"/>
<controls:BorderEx Grid.Column="0"
Grid.ColumnSpan="2"
HorizontalAlignment="Left"
Grid.Row="1"
Grid.RowSpan="2"
BorderBrush="Coral"
BorderThickness="0 0 1 0">
<controls:BorderEx.Style>
```

```
<Style TargetType="controls:BorderEx">
<Setter Property="Width" Value="50"/>
<Setter Property="Panel.ZIndex" Value="1"/>
<Setter Property="Background" Value="#F0696969"/>
<Style.Triggers>
<Trigger Property="IsMouseEnter" Value="True">
<Trigger.EnterActions>
<BeginStoryboard>
<Storyboard>
<DoubleAnimation Storyboard.TargetProperty="Width" To="200" Duration="0:0:0.8">
<DoubleAnimation.EasingFunction>
<SineEase EasingMode="EaseInOut"/>
</DoubleAnimation.EasingFunction>
</DoubleAnimation>
</Storyboard>
</BeginStoryboard>
</Trigger.EnterActions>
<Trigger.ExitActions>
<BeginStoryboard>
<Storyboard>
<DoubleAnimation Storyboard.TargetProperty="Width" To="50" Duration="0:0:0.8">
<DoubleAnimation.EasingFunction>
<SineEase EasingMode="EaseInOut"/>
</DoubleAnimation.EasingFunction>
</DoubleAnimation>
</Storyboard>
</BeginStoryboard>
</Trigger.ExitActions>
</Trigger>
</Style.Triggers>
</Style>
</controls:BorderEx.Style>
<controls:BorderEx.Resources>
<Style TargetType="Image">
<Setter Property="Height" Value="32"/>
<Setter Property="Width" Value="32"/>
</Style>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}">
<Setter Property="VerticalAlignment" Value="Center"/>
<Setter Property="Margin" Value="10 0 0 0"/>
<Setter Property="FontSize" Value="18"/>
</Style>
</controls:BorderEx.Resources>
<Grid Margin="8 0 0 0">
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Button Style="{StaticResource SideBarButton}" Click="MainMenuBtn_Click">
<Button.Content>
<StackPanel Orientation="Horizontal">
<Image Source="../Images/logo.png"/>
<TextBlock Text="INCOMSYSTEM"/>
</StackPanel>
</Button.Content>
</Button>
<StackPanel Grid.Row="1" VerticalAlignment="Center">
<Button Style="{StaticResource SideBarButton}" Click="ReviewBtn_Click">
<Button.Content>
<StackPanel Orientation="Horizontal">
<Image Source="../Images/home.png"/>
<TextBlock Text="Обзор"/>
</StackPanel>
</Button.Content>
</Button>
<Button x:Name="ProfileBtn" Style="{StaticResource SideBarButton}" Margin="0 5" Click="ProfileBtn_Click">
<Button.Content>
<StackPanel Orientation="Horizontal">
<Image Source="../Images/profileSettings.png"/>
<TextBlock Text="Профиль"/>
</StackPanel>
</Button.Content>
</Button>
<Button x:Name="ChatBtn" Style="{StaticResource SideBarButton}" Margin="0 5" Click="ChatBtn_Click">
<Button.Content>
<StackPanel Orientation="Horizontal">
<Image Source="../Images/chat.png"/>
<TextBlock Text="Мессенджер"/>
</StackPanel>
</Button.Content>
</Button>
</StackPanel>
<Button Grid.Row="2" Style="{StaticResource SideBarButton}" Click="QuitBtn_Click" Margin="0 0 0 15">
<Button.Content>
<StackPanel Orientation="Horizontal">
<Image Source="../Images/shutdown.png"/>
<TextBlock Text="Выйти"/>
</StackPanel>
</Button.Content>
</Button>
</Grid>
</controls:BorderEx>
</Grid>
</Window>
```

MainWindow.xaml.cs

```
using System.Windows;
using System.Windows.Controls;
using System.Windows.Navigation;
using System.Windows.Media.Animation;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using INCOMSYSTEM.Context;
using System.Linq;
using INCOMSYSTEM.Pages;
using INCOMSYSTEM.Pages.MainPages;
using System.Windows.Input;
namespace INCOMSYSTEM.Windows
{
public partial class MainWindow : Window
{
public MainWindow()
{
InitializeComponent();
ReviewFrame = new Frame();
ChatFrame = new Frame();
NavigationCommands.BrowseBack.InputGestures.Clear();
NavigationCommands.BrowseForward.InputGestures.Clear();
NavigationCommands.BrowseHome.InputGestures.Clear();
NavigationCommands.BrowseStop.InputGestures.Clear();
_sideBarMenu.Add(MenuItems.Review, null);
_sideBarMenu.Add(MenuItems.Profile, null);
_sideBarMenu.Add(MenuItems.Chat, null);
IsClosed = false;
```

```
ReviewFrame.Navigated += ReviewFrameOnNavigated;
ChatFrame.Navigated += ChatFrameOnNavigated;
if (AuthUser == null)
{
AuthBlock.Text = "Вы вошли как гость!";
ProfileBtn.Visibility = Visibility.Collapsed;
ChatBtn.Visibility = Visibility.Collapsed;
ReviewFrame.Navigate(new ViewTasksPage());
return;
}
SetAuthUserText();
SetStartPage();
}
private void SetAuthUserText()
{
using (var db = new INCOMSYSTEMEntities())
{
if (AuthUser.idPos == 1)
{
var user = db.Customers.First(s => s.idUser == AuthUser.idUser);
AuthBlock.Text = $"{user.name}";
}
else
{
var user = db.Employees.First(s => s.idUser == AuthUser.idUser);
AuthBlock.Text = $"{user}";
}
}
}
private void SetStartPage()
{
if (AuthUser == null)
{
ReviewFrame.Navigate(new ViewTasksPage());
return;
}
switch (AuthUser.idPos)
{
case 1:
ReviewFrame.Navigate(new ViewTasksPage());
break;
case 2:
ReviewFrame.Navigate(new ViewOrdersPage());
ChatBtn.Visibility = Visibility.Collapsed;
break;
case 3:
ReviewFrame.Navigate(new ManagerPage());
break;
default:
MessageBox.Show("Такого окна не существует, ты как вообще сюда попал?");
this.Close();
break;
}
}
public static UsersDetail AuthUser { get; set; }
}
private void ReviewFrameOnNavigated(object sender, NavigationEventArgs e)
{
if (!Equals(MFrame.Content, _sideBarMenu[MenuItems.Review]))
return;
var page = ((Frame)sender).Content as Page;
this.Title = page?.Title;
_sideBarMenu[MenuItems.Review] = page;
Nav(page);
BackBtn.IsEnabled = ReviewFrame.CanGoBack;
}
private async void Nav(Page page)
{
  var anim = new DoubleAnimation
{
EasingFunction = new SineEase { EasingMode = EasingMode.EaseInOut },
To = 0d,
Duration = TimeSpan.FromSeconds(0.35)
};

MFrame.BeginAnimation(OpacityProperty, anim);
  await Task.Delay(350);
  MFrame.Content = page;
  await Task.Delay(25);
  anim.To = 1.0d;
anim.Duration = TimeSpan.FromSeconds(0.35);
MFrame.BeginAnimation(OpacityProperty, anim);
}
private void ChatFrameOnNavigated(object sender, NavigationEventArgs e)
{
if (!Equals(MFrame.Content, _sideBarMenu[MenuItems.Chat])) return;
var page = ((Frame)sender).Content as Page;
this.Title = page?.Title;
_sideBarMenu[MenuItems.Chat] = page;
Nav(page);
BackChatBtn.IsEnabled = !(page is ChatListPage);
}
public static Frame ReviewFrame { get; private set; }
public static Frame ChatFrame { get; private set; }
private void BackBtn_Click(object sender, RoutedEventArgs e)
{
ReviewFrame.GoBack();
}
private void QuitBtn_Click(object sender, RoutedEventArgs e)
{
IsClosed = true;
new AuthWindow().Show();
this.Close();
}
private readonly Dictionary<MenuItems, Page> _sideBarMenu = new Dictionary<MenuItems, Page>();
public static bool IsClosed { get; private set; }
}
private Page ChatListPage { get; set; }
private void ReviewBtn_Click(object sender, RoutedEventArgs e)
{
this.Title = _sideBarMenu[MenuItems.Review].Title;
BackBtn.Visibility = Visibility.Visible;
BackChatBtn.Visibility = Visibility.Collapsed;
MFrame.Content = _sideBarMenu[MenuItems.Review];
}
private void ProfileBtn_Click(object sender, RoutedEventArgs e)
{
if (_sideBarMenu[MenuItems.Profile] == null) _sideBarMenu[MenuItems.Profile] = new ProfilePage(AuthUser);
this.Title = "Профиль";
BackBtn.Visibility = Visibility.Collapsed;
BackChatBtn.Visibility = Visibility.Collapsed;
MFrame.Content =
```

```
_sideBarMenu[MenuItems.Profile];
}
private void ChatBtn_Click(object sender, RoutedEventArgs e)
{
if (_sideBarMenu[MenuItems.Chat] == null)
{
ChatListPage = new ChatListPage();
_sideBarMenu[MenuItems.Chat] = ChatListPage;
}
this.Title = _sideBarMenu[MenuItems.Chat].Title;
BackBtn.Visibility = Visibility.Collapsed;
BackChatBtn.Visibility = Visibility.Visible;
MFrame.Content = _sideBarMenu[MenuItems.Chat];
}
public void GoChat(Chats chat)
{
ChatBtn_Click(null, new RoutedEventArgs());
var chatPage = ChatListPage as ChatListPage;
chatPage.Chat = chat;
chatPage.ChatEnter_LeftBtnUp(null, null);
}
private void MainMenuBtn_Click(object sender, RoutedEventArgs e)
{
ReviewBtn_Click(sender, e);
ReviewFrame = new Frame();
ReviewFrame.Navigated += ReviewFrameOnNavigated;
SetStartPage();
}
private void BackChatBtn_Click(object sender, RoutedEventArgs e)
{
ChatFrame.Navigate(ChatListPage);
}
}
internal enum MenuItems
{
Review = 0,
Profile = 1,
Chat = 2
}
}
```

AuthPage.xaml

```
<Page x:Class="INCOMSYSTEM.Pages.AuthPage"
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:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d"
Title="Окно авторизации">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
<Style TargetType="Label" BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
<ColumnDefinition Width="250"/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Grid Grid.Column="0" Grid.ColumnSpan="2"
Margin="15 0"
HorizontalAlignment="Center"
VerticalAlignment="Center">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="150"/>
</Grid.ColumnDefinitions>
<StackPanel Grid.Row="0" Grid.Column="0">
<TextBlock Text="Введите логин:" Margin="0 15 0 0"/>
<TextBlock Text="Введите пароль:" Margin="0 15 0 0"/>
</StackPanel>
<StackPanel Grid.Row="0" Grid.Column="1"
Margin="0 0 5 0">
<controls:InputTextBox x:Name="LogBox" Margin="0 5" PlaceHolder="example"/>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<controls:InputTextBox x:Name="PassBox" Margin="0 5" PlaceHolder="password"
IsPassword="True" Width="155"/>
<CheckBox Grid.Column="1" IsChecked="{Binding ElementName=PassBox, Path=IsShowed}">
<CheckBox.Style>
<Style TargetType="CheckBox">
<Setter Property="Cursor" Value="Hand"/>
<Setter Property="IsTabStop" Value="False"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="CheckBox">
<Image x:Name="eyeImage" Width="24" Height="24"/>
<ControlTemplate.Triggers>
<Trigger Property="IsChecked" Value="False">
<Setter TargetName="eyeImage" Property="Source" Value="../Images/eyeOpen.png"/>
</Trigger>
<Trigger Property="IsChecked" Value="True">
<Setter TargetName="eyeImage" Property="Source" Value="../Images/eyeClose.png"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</CheckBox.Style>
</CheckBox>
</Grid>
</StackPanel>
<StackPanel Grid.Row="1" Grid.Column="0"
Grid.ColumnSpan="2"
Margin="0 20 0 5"
```

```
Orientation="Horizontal"
HorizontalAlignment="Center">
<Button x:Name="AuthBtn"
Content="Войти"
IsEnabled="False"
Margin="5 0"/>
<Button x:Name="RegBtn"
Content="Зарегистрироваться"
Margin="5 0"/>
</StackPanel>
<Button Grid.Row="2"
Grid.Column="0"
Grid.ColumnSpan="2"
Style="{StaticResource LabelBtn}"
Click="GuestBtn_Click"
Content="Войти как гость"
HorizontalAlignment="Center"/>
<Border Grid.Row="3"
Grid.Column="0"
Grid.ColumnSpan="2"
x:Name="ErrorBorder"
Background="#560b0b"
BorderBrush="Red"
BorderThickness="1"
Visibility="Collapsed"
HorizontalAlignment="Center"
Padding="10 5"
Margin="0 10 0 15">
<TextBlock x:Name="ErrorBlock"
Foreground="White"
VerticalAlignment="Center"
TextAlignment="Center"
HorizontalAlignment="Center"
FontSize="14"/>
</Border>
</Grid>
<Border Grid.Column="2"
BorderBrush="Coral"
BorderThickness="3 0 0 0"
Margin="5 30">
<Grid VerticalAlignment="Center">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition/>
</Grid.RowDefinitions>
<Image Margin="5"
Source="../Images/authImage.png"/>
<RichTextBox IsReadOnly="True"
Grid.Row="1"
HorizontalAlignment="Center">
<FlowDocument
TextAlignment="Center">
<Paragraph>Добро пожаловать в АИС
"ИНКОМСИСТЕМ".</Paragraph>
<Paragraph>Научно-Инженерный Центр
«ИНКОМСИСТЕМ» является
крупнейшим системным интегратором
Российской Федерации и стратегическим
партнёром нефтегазовых и
нефтехимических
компаний.</Paragraph>
</FlowDocument>
</RichTextBox>
</Grid>
</Border>
</Grid>
</Page>
```

AuthPage.xaml.cs

```
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Windows;
using System.Data.Entity;
namespace INCOMSYSTEM.Pages
{
public partial class AuthPage : Page
{
public AuthPage()
{
InitializeComponent();
LogBox.TextChanged += (s, e) =>
CheckEmpty();
PassBox.TextChanged += (s, e) =>
CheckEmpty();
this.KeyDown += (s, e) => { if (e.Key ==
Key.Enter && AuthBtn.IsEnabled)
AuthBtnClick(s, e); };
AuthBtn.Click += AuthBtnClick;
RegBtn.Click += RegBtnClick;
}
private void RegBtnClick(object sender,
RoutedEventArgs e)
{
AuthWindow.AuthFrame.Navigate(new
RegPage());
}
private void AuthBtnClick(object sender,
RoutedEventArgs e)
{
if(CheckEmpty())
{
MessageBox.Show("Поля не могут быть
пустыми");
return;
}
var login = LogBox.Value;
var password = PassBox.Value;
using (var db = new
INCOMSYSTEMEntities())
{
var user = db.UsersDetail.Include(s =>
s.Positions)
.Include(s => s.Employees)
.Include(s => s.Customers)
.Include(s => s.Customers.LegalForms)
.FirstOrDefault(s => s.login == login &&
s.password == password);
if (user != null)
{
MainWindow.AuthUser = user;
var window = new MainWindow();
window.Show();
Application.Current.Windows.OfType<Aut
hWindow>().First().Close();
return;
}
ShowError("Неверный логин или
пароль");
}
}
private bool CheckEmpty()
{
if (LogBox.IsWhiteSpace) return
ShowError("Логин не может быть
пустым");
if (PassBox.IsWhiteSpace) return
ShowError("Пароль не может быть
пустым");
AuthBtn.IsEnabled = true;
ErrorBlock.Text = string.Empty;
ErrorBorder.Visibility =
Visibility.Collapsed;
return false;
}
private bool ShowError(string error)
{
AuthBtn.IsEnabled = false;
ErrorBorder.Visibility = Visibility.Visible;
ErrorBlock.Text = error;
return true;
}
private void GuestBtn_Click(object sender,
RoutedEventArgs e)
{
MainWindow.AuthUser = null;
new MainWindow().Show();
```

```
Application.Current.Windows.OfType<AuthWindow>().First().Close();
}
}
}
```

RegPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.RegPage"
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:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d"
Title="Регистрация">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}">
<Setter Property="Height" Value="30"/>
<Setter Property="FontSize" Value="18"/>
<Setter Property="TextWrapping" Value="Wrap"/>
<Setter Property="Margin" Value="0 5 0 0"/>
</Style>
<Style TargetType="controls:InputTextBox" BasedOn="{StaticResource {x:Type controls:InputTextBox}}">
<Setter Property="Height" Value="30"/>
<Setter Property="Margin" Value="0 5 0 0"/>
</Style>
<Style TargetType="ComboBox" BasedOn="{StaticResource {x:Type ComboBox}}">
<Setter Property="Height" Value="30"/>
<Setter Property="Margin" Value="0 5 0 0"/>
</Style>
</Page.Resources>
<Grid HorizontalAlignment="Center" VerticalAlignment="Center">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="110"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="40"/>
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
<RowDefinition />
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<StackPanel Grid.Column="0" Grid.Row="0">
<TextBlock x:Name="NameBlock" Text="Название: " Margin="0"/>
<TextBlock Text="Правовая форма: " Height="40" Width="110"/>
<TextBlock Text="Телефон: "/>
<TextBlock Text="Паспортные данные: " Height="40" Width="110"/>
<TextBlock Text="Адрес: "/>
<TextBlock Text="Логин: "/>
<TextBlock Text="Пароль: "/>
</StackPanel>
<StackPanel Grid.Column="1" Grid.Row="0" HorizontalAlignment="Left">
<controls:InputTextBox x:Name="InpName" MaxLength="150" Margin="0" PlaceHolder="Название\ФИО"/>
<ComboBox x:Name="CmbLegal" SelectedIndex="0" Margin="0 15 0 0" SelectionChanged="CmbLegal_OnSelectionChanged">
<ComboBox.ItemTemplate>
<DataTemplate>
<TextBlock Text="{Binding name}" Style="{StaticResource ComboBoxTextBlockStyle}"/>
</DataTemplate>
</ComboBox.ItemTemplate>
</ComboBox>
<controls:InputTextBox x:Name="InpPhone" Margin="0 0 0 0" PlaceHolder="+7(900)-000-00-00"/>
<StackPanel Height="40" VerticalAlignment="Center" Margin="0 10 0 0" Orientation="Horizontal">
<StackPanel Orientation="Horizontal">
<TextBlock Text="Серия: " FontSize="16"/>
<controls:InputTextBox MaxLength="4" Width="55" PlaceHolder="Серия" x:Name="InpPassportSerie"/>
</StackPanel>
<StackPanel Orientation="Horizontal" Margin="5 0 0 0">
<TextBlock Text="№: " FontSize="16"/>
<controls:InputTextBox MaxLength="6" Width="65" PlaceHolder="Номер" x:Name="InpPassportNumber"/>
</StackPanel>
</StackPanel>
<controls:InputTextBox x:Name="InpAddress" MaxLength="175" PlaceHolder="Адрес"/>
<controls:InputTextBox x:Name="InpLogin" MaxLength="30" PlaceHolder="Логин"/>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<controls:InputTextBox x:Name="InpPassword" IsPassword="True" MaxLength="50" Width="155" PlaceHolder="Пароль"/>
<CheckBox IsChecked="{Binding ElementName=InpPassword, Path=IsShowed}" Grid.Column="1">
<CheckBox.Style>
<Style TargetType="CheckBox">
<Setter Property="Cursor" Value="Hand"/>
<Setter Property="IsTabStop" Value="False"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="CheckBox">
<Image x:Name="eyeImage" Width="24" Height="24"/>
<ControlTemplate.Triggers>
<Trigger Property="IsChecked" Value="False">
<Setter TargetName="eyeImage" Property="Source" Value="../Images/eyeOpen.png"/>
</Trigger>
<Trigger Property="IsChecked" Value="True">
<Setter TargetName="eyeImage" Property="Source" Value="../Images/eyeClose.png"/>
</Trigger>
</ControlTemplate.Triggers>
```

```
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</CheckBox.Style>
</CheckBox>
</Grid>
</StackPanel>
<Button VerticalAlignment="Bottom" Grid.Column="2" Click="GenerateNewPasswordBtn_Click" Margin="5 0 0 10">
<Button.ToolTip>
<ToolTip ToolTipService.InitialShowDelay="1000" Height="35">
<TextBlock Text="Сгенерировать новый пароль" FontSize="16"/>
</ToolTip>
</Button.ToolTip>
<Button.Content>
<Image Source="../Images/doorKey.png"/>
</Button.Content>
</Button>
<StackPanel Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="2" Margin="0 5 0 5" Orientation="Horizontal" HorizontalAlignment="Center">
<Button x:Name="RegBtn" Content="Зарегистрироваться" IsEnabled="False" Margin="5 0" Click="RegBtn_Click"/>
<Button Content="Назад" Margin="5 0" Click="BackBtn_Click"/>
</StackPanel>
<Border Grid.Row="2" Grid.Column="0" Grid.ColumnSpan="3" x:Name="ErrorBorder" Background="#560b0b" BorderBrush="Red" BorderThickness="1" Visibility="Collapsed" HorizontalAlignment="Center" Padding="10 5" Margin="0 10 0 15">
<TextBlock x:Name="ErrorBlock" Foreground="White" VerticalAlignment="Center" TextAlignment="Center" HorizontalAlignment="Center" FontSize="14"/>
</Border>
</Grid>
</Page>
```

RegPage.xaml.cs

```
using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using INCOMSYSTEM.BehaviorsFiles;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Windows;
namespace INCOMSYSTEM.Pages
{
public partial class RegPage : Page
{
public RegPage()
{
InitializeComponent();
using (var db = new INCOMSYSTEMEntities())
{
CmbLegal.ItemsSource = db.LegalForms.ToList();
}
this.KeyDown += (s, e) => { if (e.Key == Key.Enter && RegBtn.IsEnabled) RegBtn_Click(s, e); };
InpName.TextChanged += (s, e) => CheckEmpty();
InpPhone.TextChanged += (s, e) => CheckEmpty();
InpPassportSerie.TextChanged += (s, e) => CheckEmpty();
InpPassportNumber.TextChanged += (s, e) => CheckEmpty();
InpAddress.TextChanged += (s, e) => CheckEmpty();
InpLogin.TextChanged += (s, e) => CheckEmpty();
InpPassword.TextChanged += (s, e) => CheckEmpty();
}
private void RegBtn_Click(object sender, RoutedEventArgs e)
{
if (CheckEmpty()) return;
if (InpPassword.Value.Length < 6)
{
ShowError("Длина пароля должна достигать минимум 6 символов");
return;
}
if (PasswordDifficulty.CheckDifficultyPassword(InpPassword.Value) == DifficultyPassword.Easy)
{
ShowError("Пароль не может быть слабым");
return;
}
using (var db = new INCOMSYSTEMEntities())
{
long? phone = null;
if (!InpPhone.IsWhiteSpace)
{
try { phone = long.Parse(InpPhone.Value); }
catch { return; }
}
var passportStr = InpPassportSerie.Value + InpPassportNumber.Value;
long.TryParse(passportStr, out var passport);
if (db.UsersDetail.FirstOrDefault(s => s.login.ToLower() == InpLogin.Value.ToLower().Trim()) != null)
{
ShowError("Такой логин уже занят");
return;
}
if (phone != null && db.UsersDetail.FirstOrDefault(s => s.phone == phone) != null)
{
ShowError("Номер телефона уже занят");
return;
}
if (db.UsersDetail.FirstOrDefault(s => s.passport == passport) != null)
{
ShowError("Паспорт уже занят");
return;
}
var userDetail = new UsersDetail
{
login = InpLogin.Value,
password = InpPassword.Value,
phone = phone,
idPos = 1,
passport = passport,
```

```
address = InpAddress.Value,
dateStart = DateTime.Now
};
db.UsersDetail.Add(userDetail);
var customer = new Customers
{
name = InpName.Value,
idLegalForm = ((LegalForms)CmbLegal.SelectedItem).id,
};
db.Customers.Add(customer);
db.SaveChanges();
MessageBox.Show("Вы успешно зарегистрировались");
MainWindow.AuthUser = userDetail;
new MainWindow().Show();
Application.Current.Windows.OfType<AuthWindow>().First().Close();
}
}
private void BackBtn_Click(object sender, RoutedEventArgs e)
{
AuthWindow.AuthFrame.Navigate(new AuthPage());
}
private bool CheckEmpty()
{
if (InpName.IsWhiteSpace) return ShowError("Название/фио не может быть пустым");
if (!InpPhone.IsWhiteSpace && long.TryParse(InpPhone.Value, out _)) return ShowError("Номер телефона не может содержать символы");
if (!InpPhone.IsWhiteSpace && InpPhone.Value.Length != 11) return ShowError("Номер телефона должен содержать 11 цифр");
if (InpPassportSerie.IsWhiteSpace) return ShowError("Серия паспорта не может быть пустым");
if (!long.TryParse(InpPassportSerie.Value, out _)) return ShowError("Серия паспорта не может содержать символы");
if (InpPassportSerie.Value.Length != 4) return ShowError("Серия паспорта должен содержать 4 цифры");
if (InpPassportNumber.IsWhiteSpace) return ShowError("Номер паспорта не может быть пустым");
if (!long.TryParse(InpPassportNumber.Value, out _)) return ShowError("Номер паспорта не может содержать символы");
if (InpPassportNumber.Value.Length != 6) return ShowError("Номер паспорта должен содержать 6 цифры");
if (InpAddress.IsWhiteSpace) return ShowError("Адрес не может быть пустым");
if (InpLogin.IsWhiteSpace) return ShowError("Логин не может быть пустым");
if (InpPassword.IsWhiteSpace) return ShowError("Пароль не может быть пустым");
RegBtn.IsEnabled = true;
ErrorBlock.Text = string.Empty;
ErrorBorder.Visibility = Visibility.Collapsed;
return false;
}
private bool ShowError(string error)
{
RegBtn.IsEnabled = false;
ErrorBorder.Visibility = Visibility.Visible;
ErrorBlock.Text = error;
return true;
}
private void GenerateNewPasswordBtn_Click(object sender, RoutedEventArgs e)
{
InpPassword.IsShowed = true;
InpPassword.Text = PasswordGenerator.GetNewPassword();
}
private void CmbLegal_OnSelectionChanged(object sender, SelectionChangedEventArgs e)
{
NameBlock.Text = ((LegalForms)CmbLegal.SelectedItem).id == 1 ? "ФИО" : "Название";
}
}
}
```

ProfilePage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.ProfilePage"
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:gl="clr-namespace:System.Globalization;assembly=mscorlib"
xmlns:wnd="clr-namespace:INCOMSYSTEM.Windows"
mc:Ignorable="d">
<Page.Resources>
<Style TargetType="{x:Type TextBlock}" BasedOn="{StaticResource TextBlockStyle}" />
</Page.Resources>
<Grid HorizontalAlignment="Center" VerticalAlignment="Center">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto" />
<ColumnDefinition />
</Grid.ColumnDefinitions>
<Border BorderBrush="White" BorderThickness="0 0 1 0" Margin="5 0" />
<StackPanel Margin="10 0">
<Button Style="{StaticResource SideBarButton}">
<StackPanel Orientation="Horizontal">
<Image Width="32" Height="32" Source="../Images/profile.png" />
<TextBlock VerticalAlignment="Center" Text="Основное" Margin="5 0 0 0" />
</StackPanel>
</Button>
</StackPanel>
<Grid Grid.Column="1">
<Grid.RowDefinitions>
<RowDefinition Height="Auto" />
<RowDefinition Height="200" />
</Grid.RowDefinitions>
<Grid Margin="0 0 0 15">
<Grid.ColumnDefinitions>
<ColumnDefinition />
<ColumnDefinition />
</Grid.ColumnDefinitions>
<StackPanel>
<TextBlock x:Name="NameBlock" />
<TextBlock x:Name="PassportBlock" />
<TextBlock x:Name="PositionBlock" />
<TextBlock x:Name="PhoneBlock" />
```

```
<TextBlock
x:Name="CustomerLegalFormBlock" />
<TextBlock x:Name="AddressBlock" />
</StackPanel>
<Border BorderBrush="White"
BorderThickness="0 0 5 0" />
<StackPanel Grid.Column="1"
Margin="5 0">
<TextBlock x:Name="DateStart" />
<TextBlock
Text="{Binding ElementName=OrdersList,
Path=Items.Count, StringFormat='{}Всего
заказов: {0}'}" />
<TextBlock x:Name="PopularOrderBlock"
/>
<TextBlock
x:Name="MostExpensiveBlock" />
</StackPanel>
</Grid>
<ScrollViewer Grid.Row="1"
Grid.ColumnSpan="2"
VerticalScrollBarVisibility="Disabled"
HorizontalScrollBarVisibility="Auto">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto" />
<RowDefinition />
</Grid.RowDefinitions>
<Button Click="RefreshOrdersBtn_Click"
HorizontalAlignment="Left">
<Button.Content>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="32" />
<ColumnDefinition Width="Auto" />
</Grid.ColumnDefinitions>
<Image Source="../Images/refresh.png" />
<TextBlock Text="Обновить"
Grid.Column="1" />
</Grid>
</Button.Content>
</Button>
<Grid Grid.Row="1">
<Grid.RowDefinitions>
<RowDefinition Height="Auto" />
<RowDefinition />
</Grid.RowDefinitions>
<Grid>
<Grid.Resources>
<Style TargetType="TextBlock"
BasedOn="{StaticResource
ComboBoxTextBlockStyle}">
<Setter Property="Foreground"
Value="White" />
<Setter Property="TextAlignment"
Value="Center" />
<Setter Property="VerticalAlignment"
Value="Center" />
<Setter Property="Background"
Value="DimGray" />
<Setter Property="FontWeight"
Value="Bold" />
</Style>
<Style TargetType="Border">
<Setter Property="Background"
Value="DimGray" />
<Setter Property="BorderBrush"
Value="Black" />
<Setter Property="BorderThickness"
Value="1" />
</Style>
</Grid.Resources>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="100" />
<ColumnDefinition Width="200" />
<ColumnDefinition Width="200" />
<ColumnDefinition Width="150" />
<ColumnDefinition Width="100" />
<ColumnDefinition Width="100" />
<ColumnDefinition Width="225" />
<ColumnDefinition Width="100" />
</Grid.ColumnDefinitions>
<Border>
<TextBlock Text="№Заказа" />
</Border>
<Border Grid.Column="1">
<TextBlock>
<TextBlock.Style>
<Style TargetType="TextBlock"
BasedOn="{StaticResource {x:Type
TextBlock}}">
<Setter Property="Text"
Value="Заказчик"/>
<Style.Triggers>
<DataTrigger
Binding="{Binding Source={x:Static
wnd:MainWindow.AuthUser},
Path=idPos}"
Value="1">
<Setter Property="Text"
Value="Исполнитель" />
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
<Border Grid.Column="2">
<TextBlock>
<TextBlock.Style>
<Style TargetType="TextBlock"
BasedOn="{StaticResource {x:Type
TextBlock}}">
<Setter Property="Text"
Value="Менеджер"/>
<Style.Triggers>
<DataTrigger
Binding="{Binding Source={x:Static
wnd:MainWindow.AuthUser},
Path=idPos}"
Value="3">
<Setter Property="Text"
Value="Исполнитель" />
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
<Border Grid.Column="3">
<TextBlock Text="Задача" />
</Border>
<Border Grid.Column="4">
<TextBlock Text="Цена" />
</Border>
<Border Grid.Column="5">
<TextBlock Text="Сложность" />
</Border>
<Border Grid.Column="6">
<TextBlock Text="Дата формирования
заказа" />
</Border>
<Border Grid.Column="7">
<TextBlock Text="Статус" />
</Border>
</Grid>
<ItemsControl x:Name="OrdersList"
Grid.Row="1"
Style="{StaticResource ItemsVisible}">
<ItemsControl.ItemTemplate>
<DataTemplate>
<CheckBox x:Name="OrderGrid"
Margin="0 0 0 10"
Style="{StaticResource
ClearCheckBoxTemplate}">
<CheckBox.Content>
<Grid>
<Grid.Resources>
<Style TargetType="TextBlock"
```

```
BasedOn="{StaticResource ComboBoxTextBlockStyle}">
<Setter Property="Foreground" Value="White" />
<Setter Property="TextAlignment" Value="Center" />
<Setter Property="VerticalAlignment" Value="Center" />
<Setter Property="Background" Value="DimGray" />
</Style>
<Style TargetType="Border">
<Setter Property="Background" Value="DimGray" />
<Setter Property="BorderBrush" Value="Black" />
<Setter Property="BorderThickness" Value="1" />
<Setter Property="Padding" Value="5" />
</Style>
</Grid.Resources>
<Grid.RowDefinitions>
<RowDefinition Height="40" />
<RowDefinition Height="Auto" />
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="100" />
<ColumnDefinition Width="200" />
<ColumnDefinition Width="200" />
<ColumnDefinition Width="150" />
<ColumnDefinition Width="100" />
<ColumnDefinition Width="100" />
<ColumnDefinition Width="225" />
<ColumnDefinition Width="100" />
</Grid.ColumnDefinitions>
<Border>
<TextBlock Text="{Binding id}" />
</Border>
<Border Grid.Column="1">
<TextBlock>
<TextBlock.Style>
<Style TargetType="TextBlock" BasedOn="{StaticResource {x:Type TextBlock}}">
<Setter Property="Text" Value="{Binding Customers.name, TargetNullValue='Отсутствует'}"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static wnd:MainWindow.AuthUser}, Path=idPos}" Value="1">
<Setter Property="Text" Value="{Binding Employees, TargetNullValue='Отсутствует'}" />
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
<Border Grid.Column="2">
<TextBlock>
<TextBlock.Style>
<Style TargetType="TextBlock" BasedOn="{StaticResource {x:Type TextBlock}}">
<Setter Property="Text" Value="{Binding Chats.Employees, TargetNullValue='Отсутствует'}"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static wnd:MainWindow.AuthUser}, Path=idPos}" Value="3">
<Setter Property="Text" Value="{Binding Employees, TargetNullValue='Отсутствует'}" />
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
<Border Grid.Column="3">
<TextBlock Text="{Binding Tasks.name}" />
</Border>
<Border Grid.Column="4">
<TextBlock Text="{Binding price, StringFormat='{}{0:0,0} руб.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}}" />
</Border>
<Border Grid.Column="5">
<TextBlock Text="{Binding difficulty}" />
</Border>
<Border Grid.Column="6">
<TextBlock Text="{Binding dateOrder, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}}" />
</Border>
<Border Grid.Column="7">
<TextBlock Text="{Binding Statuses.name}" />
</Border>
<UniformGrid Grid.Row="1" Grid.ColumnSpan="9" Grid.Column="0" Columns="4" Height="80">
<UniformGrid.Style>
<Style TargetType="UniformGrid">
<Setter Property="Visibility" Value="Collapsed"/>
<Style.Triggers>
<DataTrigger Binding="{Binding ElementName=OrderGrid, Path=IsChecked}" Value="True">
<Setter Property="Visibility" Value="Visible"/>
</DataTrigger>
</Style.Triggers>
</Style>
</UniformGrid.Style>
<Border>
<TextBlock Text="Плановая дата начала" FontWeight="Bold" />
</Border>
<Border>
<TextBlock Text="Фактическая дата начала" FontWeight="Bold" />
</Border>
<Border>
<TextBlock Text="Плановая дата завершения" FontWeight="Bold" />
</Border>
<Border>
<TextBlock Text="Фактическая дата завершения" FontWeight="Bold" />
</Border>
<Border>
<TextBlock Text="{Binding planDateStart, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}" />
</Border>
<Border>
<TextBlock Text="{Binding factDateStart, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}" />
</Border>
```

```
<Border>
<TextBlock Text="{Binding planDateComplete, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}" />
</Border>
<Border>
<TextBlock Text="{Binding factDateComplete, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}" />
</Border>
</UniformGrid>
</Grid>
</CheckBox.Content>
</CheckBox>
</DataTemplate>
</ItemsControl.ItemTemplate>
</ItemsControl>
</Grid>
</Grid>
</ScrollViewer>
</Grid>
</Grid>
</Page>
```

ProfilePage.xaml.cs

```
using System.Collections.ObjectModel;
using System.Data.Entity;
using System.Globalization;
using System.Linq;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using INCOMSYSTEM.Context;
namespace INCOMSYSTEM.Pages
{
public partial class ProfilePage : Page
{
public ProfilePage(UsersDetail usersDetail)
{
_usersDetail = usersDetail;
InitializeComponent();
RefreshOrders();
OrdersList.ItemsSource = OrdersCollection;
NameBlock.Text = usersDetail.idPos == 1 ? $"ФИО: {usersDetail.Customers.name}" : $"ФИО: {usersDetail.Employees}";
PositionBlock.Text = $"Должность: {usersDetail.Positions.name}";
PassportBlock.Text = $"Паспорт: {usersDetail.SeriePassport} {usersDetail.NumberPassport}";
PhoneBlock.Text = $"Номер телефона: {usersDetail.phone}";
PhoneBlock.Text = "Номер телефона: " + (usersDetail.phone != null ? usersDetail.phone.ToString() : "Отсутствует");
if (usersDetail.idPos == 1)
CustomerLegalFormBlock.Text = $"Правовая форма: {usersDetail.Customers.LegalForms.name}";
else CustomerLegalFormBlock.Visibility = Visibility.Collapsed;
AddressBlock.Text = $"Адрес проживания: {usersDetail.address}";
DateStart.Text = "Дата регистрации: " + usersDetail.dateStart.ToString("dd MMMM yyyy", CultureInfo.CurrentCulture);
}
private readonly UsersDetail _usersDetail;
private ObservableCollection<Orders> OrdersCollection { get; set; } = new ObservableCollection<Orders>();
private async void RefreshOrders()
{
var list = await Task.Run(() =>
{
using (var db = new INCOMSYSTEMEntities())
{
return db.Orders
.Include(s => s.Employees)
.Include(s => s.Employees.UsersDetail)
.Include(s => s.Customers.UsersDetail)
.Include(s => s.Statuses)
.Include(s => s.Tasks)
.Include(s => s.Chats)
.Include(s => s.Chats.Employees)
.Where(s => s.idCustomer == _usersDetail.idUser || s.idExecutor == _usersDetail.idUser || s.Chats.idManager == _usersDetail.idUser)
.ToList();
}
});
OrdersCollection = new ObservableCollection<Orders>(list);
OrdersList.ItemsSource = OrdersCollection;
var count = 0;
Orders order = null;
var groupBy = list.GroupBy(s => s.idTask);
foreach (var ordersEnumerable in groupBy)
{
var k = ordersEnumerable.Count();
if (k <= count) continue;
count = k;
order = ordersEnumerable.First();
}
PopularOrderBlock.Text = "Популярный заказ: " + (order != null ? order.Tasks.name : "Отсутствует");
MostExpensiveBlock.Text = "Самый дорогой заказ: " + (list.Count > 0 ? list.Max(s => s.price).ToString("#,#", CultureInfo.CurrentCulture) + " руб." : "Отсутствует");
}
private void RefreshOrdersBtn_Click(object sender, RoutedEventArgs e)
{
RefreshOrders();
}
}
}
```

TopPanelControl.xaml

```
<UserControl
x:Class="INCOMSYSTEM.Controls.TopPanelControl"
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">
<Border
MouseEnter="PnlControlBar_OnMouseEnter"
MouseLeftButtonDown="PnlControlBar_OnMouseLeftButtonDown" Padding="5">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto" />
<RowDefinition />
</Grid.RowDefinitions>
<Grid Grid.Row="0">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
```

```
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<StackPanel Grid.Column="0" Orientation="Horizontal" VerticalAlignment="Center">
<Image Margin="5 0" Source="../Images/logo.png"/>
<Grid>
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition/>
</Grid.RowDefinitions>
<TextBlock Grid.Row="0" FontFamily="/INCOMSYSTEM;component/Fonts/#Roboto" FontSize="16" Foreground="White" Margin="0 5 5 0" Text="ИНКОМСИСТЕМ"/>
<TextBlock Grid.Row="1" FontFamily="/INCOMSYSTEM;component/Fonts/#Roboto" FontSize="12" Foreground="White" Margin="0 0 5 5" Text="Научный Инженерный Центр"/>
</Grid>
</StackPanel>
<TextBlock Grid.Column="1" FontFamily="/INCOMSYSTEM;component/Fonts/#Roboto" TextAlignment="Center" VerticalAlignment="Center" FontSize="18" Foreground="White" Margin="5,5" Text="{Binding Title, RelativeSource={RelativeSource AncestorType=Window}}">
<TextBlock.ToolTip>
<Binding RelativeSource="{RelativeSource Mode=Self}" Path="Text"/>
</TextBlock.ToolTip>
</TextBlock>
<StackPanel Grid.Column="2" Orientation="Horizontal">
<Button Click="MinimizeBtn_Click" Margin="3 0" Width="40">
<TextBlock Text="0" FontFamily="Webdings" TextAlignment="Center" VerticalAlignment="Center" Foreground="White"/>
</Button>
<Button x:Name="SwitchStateBtn" Margin="3 0" Click="SwitchStateBtn_Click" Width="40">
<TextBlock x:Name="WindowCurrentState" Text="1" FontFamily="Webdings" TextAlignment="Center" VerticalAlignment="Center" Foreground="White"/>
</Button>
<Button Click="CloseBtn_Click" Margin="3 0" Width="40">
<TextBlock Text="r" FontFamily="Webdings" TextAlignment="Center" VerticalAlignment="Center" Foreground="White"/>
</Button>
</StackPanel>
</Grid>
<ContentPresenter Grid.Row="1" />
</Grid>
</Border>
</UserControl>
```

TopPanelControl.xaml.cs

```
using System;
using System.Runtime.InteropServices;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Interop;
namespace INCOMSYSTEM.Controls
{
public partial class TopPanelControl : UserControl
{
public TopPanelControl()
{
InitializeComponent();
Loaded += (s, e) =>
{
var window = (this.Parent as Grid).Parent as Window;
window.MaxHeight = SystemParameters.MaximizedPrimaryScreenHeight;
SwitchStateBtn.Visibility = window.ResizeMode != ResizeMode.NoResize ? Visibility.Visible : Visibility.Collapsed;
WindowCurrentState.Text = window.WindowState == WindowState.Maximized ? "2" : "1";
};
}
[DllImport("user32.dll")]
private static extern IntPtr SendMessage(IntPtr hWnd, int wMsg, int wParam, int lParam);
private void PnlControlBar_OnMouseLeftButtonDown( object sender, MouseButtonEventArgs e)
{
if (e.ClickCount == 2)
{
SwitchState();
return;
}
var window = (this.Parent as Grid).Parent as Window;
var helper = new WindowInteropHelper(window);
SendMessage(helper.Handle, 161, 2, 0);
WindowCurrentState.Text = window.WindowState == WindowState.Maximized ? "2" : "1";
}
private void PnlControlBar_OnMouseEnter(object sender, MouseEventArgs e)
{
var window = (this.Parent as Grid).Parent as Window;
window.MaxHeight = SystemParameters.MaximizedPrimaryScreenHeight;
}
private void CloseBtn_Click(object sender, RoutedEventArgs e)
{
var window = (this.Parent as Grid).Parent as Window;
window.Close();
}
private void MinimizeBtn_Click(object sender, RoutedEventArgs e)
{
var window = (this.Parent as Grid).Parent as Window;
window.WindowState = WindowState.Minimized;
}
private void SwitchStateBtn_Click(object
```

```
sender, RoutedEventArgs e)
{
SwitchState();
}
private void SwitchState()
{
var window = (this.Parent as Grid).Parent as
Window;
window.WindowState =
window.WindowState ==
WindowState.Maximized ?
WindowState.Normal :
WindowState.Maximized;
WindowCurrentState.Text =
window.WindowState ==
WindowState.Maximized ? "2" : "1";
}
}
}
```

Приложение С.

AdditionalWindow.xaml

```
<Window
x:Class="INCOMSYSTEM.Windows.Additionalwindow"
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:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d"
AllowsTransparency="True"
WindowStyle="None"
WindowStartupLocation="CenterScreen"
WindowState="Normal"
Background="DimGray"
ResizeMode="NoResize"
Height="575"
Width="600"
BorderBrush="Black"
BorderThickness="2">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<controls:TopPanelControl/>
<Frame x:Name="MFrame"
Grid.Row="1"
Margin="5"
IsTabStop="False"
NavigationUIVisibility="Hidden"/>
<Border Grid.Row="2"
x:Name="ErrorBorder"
Background="#560b0b"
BorderBrush="Red"
BorderThickness="1"
Visibility="Collapsed"
HorizontalAlignment="Center"
Padding="10 5"
Margin="0 0 0 15">
<TextBlock x:Name="ErrorBlock"
Foreground="White"
VerticalAlignment="Center"
TextAlignment="Center"
HorizontalAlignment="Center"
FontSize="14"/>
</Border>
</Grid>
</Window>
```

AdditionalWindow.xaml.cs

```
using System.Windows;
using System.Windows.Controls;
using System.Windows.Navigation;
namespace INCOMSYSTEM.Windows
{
public partial class AdditionalWindow : Window
{
public AdditionalWindow()
{
InitializeComponent();
_errorStaticBorder = ErrorBorder;
_errorStaticBlock = ErrorBlock;
MFrame.Navigated += MainFrameOnNavigated;
}
private static Border _errorStaticBorder;
private static TextBlock _errorStaticBlock;
private void MainFrameOnNavigated(object sender, NavigationEventArgs e)
{
var page = ((Frame)sender).Content as Page;
this.Title = page?.Title;
this.Width = page.Width + 50d;
this.Height = page.Height + 50d;
}
public static void ShowError(string error)
{
_errorStaticBlock.Text = error;
_errorStaticBorder.Visibility = Visibility.Visible;
}
public static void HideError()
{
_errorStaticBlock.Text = string.Empty;
_errorStaticBorder.Visibility = Visibility.Collapsed;
}
}
}
```

Приложение D.

ChatListPageViewModel.cs

```
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Windows;
using System;
using System.Data.Entity;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Data;
using INCOMSYSTEM.BehaviorsFiles;
namespace INCOMSYSTEM.ViewModels
{
public class ChatListPageViewModel : ObservableObject
{
public ChatListPageViewModel()
{
ChatsCollection = new ObservableCollection<ChatMess>();
CollectionViewChat = CollectionViewSource.GetDefaultView(ChatsCollection);
CollectionViewChat.SortDescriptions.Insert(0, new SortDescription("SendDate", ListSortDirection.Descending));
ChatsCollection.CollectionChanged += (s, e) => OnRaiseChanged();
RefreshCollection();
}
public ObservableCollection<ChatMess> ChatsCollection { get; }
public ICollectionView CollectionViewChat { get; }
public async void RefreshCollection()
{
while (!MainWindow.IsClosed)
{
using (var db = await Task.Run(() => new INCOMSYSTEMEntities()))
{
var chats = db.Chats.Include(s => s.Customers)
.Include(s => s.Employees)
.Include(s => s.Employees.UsersDetail)
.Include(s => s.Employees.UsersDetail)
.Include(s => s.Orders)
.Include(s => s.Orders.Tasks)
.Include(s => s.Orders.Statuses)
.Where(s => s.idCustomer == MainWindow.AuthUser.idUser || s.idManager == MainWindow.AuthUser.idUser)
.ToList();
var messages = db.Messages
.Include(s => s.UsersDetail)
.ToList();
var chatColl = ChatsCollection.ToList().Where(s => !chats.Select(k => k.idChat).Contains(s.Id));
if (MainWindow.AuthUser.idPos == 3 && chatColl.Any())
{
foreach (var c in chatColl)
{
ChatsCollection.Remove(c);
}
continue;
}
foreach (var chat in chats)
{
var message = messages.LastOrDefault(s => s.idChat == chat.idChat) ?? new Messages
{
idChat = chat.idChat,
dateSend = chat.dateCreate,
message = $"Подробности заказа \"{chat.Orders.Tasks.name}\"",
idUser = chat.idCustomer,
UsersDetail = chat.Customers.UsersDetail
};
var chatMess = new ChatMess
{
Id = chat.idChat,
Chat = chat,
SendDate = message.dateSend,
Message = message,
Order = chat.Orders,
IsConnected = chat.Employees != null
};

if (MainWindow.AuthUser.idUser == chat.idCustomer) chatMess.Recipient = chat.Employees == null ? "Ожидание подключения менеджера" : $"{chat.Employees}";
else chatMess.Recipient = chat.Customers.name;

var chatMessColl = ChatsCollection.FirstOrDefault(s => s.Id == chatMess.Id);
if (chatMessColl == null) ChatsCollection.Add(chatMess);
else
{
chatMessColl.IsConnected = chatMess.IsConnected;
if (chatMessColl.Chat.Employees != chat.Employees)
{
chatMessColl.Chat.idManager = chat.idManager;
chatMessColl.Chat.Employees = chat.Employees;
chatMessColl.Recipient = chatMess.Recipient;
}

if (chatMessColl.Chat.Employees == null || chatMessColl.Message.id == chatMess.Id) continue;

chatMessColl.SendDate = chatMess.SendDate;
chatMessColl.Message = chatMess.Message;
}
}
}

await Task.Delay(2500);
}
}
}
public class ChatMess : ObservableObject
{
private long _id;
private string _recipient;
private Chats _chat;
private Orders _order;
private Messages _message;
private DateTime? _sendDate;
private bool _isConnected;
public long Id
{
get => _id;
set
{
_id = value;
OnRaiseChanged();
}
}
public string Recipient
{
get => _recipient;
set
```

```
{
_recipient = value;
OnRaiseChanged();
}
}
public Chats Chat
{
get => _chat;
set
{
_chat = value;
OnRaiseChanged();
}
}
public Orders Order
{
get => _order;
set
{
_order = value;
OnRaiseChanged();
}
}
public Messages Message
{
get => _message;
set
{
_message = value;
OnRaiseChanged();
}
}
public DateTime? SendDate
{
get => _sendDate;
set
{
_sendDate = value;
OnRaiseChanged();
}
}
public bool IsConnected
{
get => _isConnected;
set
{
_isConnected = value;
OnRaiseChanged();
}
}
}
}
```

ChatListPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.ChatList
Page"
xmlns="http://schemas.microsoft.com/winf
x/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/wi
nfx/2006/xaml"
xmlns:mc="http://schemas.openxmlformats
.org/markup-compatibility/2006"
xmlns:d="http://schemas.microsoft.com/ex
pression/blend/2008"
xmlns:vm="clr-
namespace:INCOMSYSTEM.ViewModels
"
mc:Ignorable="d"
Title="Мессенджер">
<Page.DataContext>
<vm:ChatListPageViewModel/>
</Page.DataContext>
<Grid>
<ItemsControl BorderThickness="0"
ItemsSource="{Binding
CollectionViewChat,
UpdateSourceTrigger=PropertyChanged}
"
BorderBrush="Transparent"
Margin="75 0">
<ItemsControl.ItemsPanel>
<ItemsPanelTemplate>
<StackPanel Cursor="Hand"/>
</ItemsPanelTemplate>
</ItemsControl.ItemsPanel>
<ItemsControl.Style>
<Style TargetType="ItemsControl"
BasedOn="{StaticResource
ItemsVisible}">
</Style>
</ItemsControl.Style>
<ItemsControl.ItemTemplate>
<DataTemplate>
<Border>
<Grid
Margin="0 3"
MouseLeftButtonUp="ChatEnter_LeftBtnU
p">
<Grid.Style>
<Style TargetType="Grid">
<Style.Triggers>
<Trigger Property="IsMouseOver"
Value="True">
<Setter Property="Background"
Value="#9B9B9B"/>
</Trigger>
</Style.Triggers>
</Style>
</Grid.Style>
<Grid.Resources>
<Style TargetType="TextBlock"
BasedOn="{StaticResource
TextBlockStyle}"/>
</Grid.Resources>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="225"/>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Border Grid.Column="0"
Grid.ColumnSpan="3"
BorderBrush="White"
BorderThickness="0 0 0 2"
x:Name="MainBorder"/>
<Border Grid.Column="0"
BorderBrush="#fff"
BorderThickness="0 0 2 0"/>
<Border Grid.Column="1"
BorderBrush="#fff"
BorderThickness="0 0 2 0"/>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="32"/>
<ColumnDefinition />
</Grid.ColumnDefinitions>
<Image Source="../Images/user.png"
Width="32" Height="32"/>
<StackPanel Grid.Column="1" Margin="5
0">
<TextBlock Text="{Binding
Order.Tasks.name}"
TextWrapping="Wrap"/>
<Border BorderBrush="#fff"
BorderThickness="0 0 0 2"/>
<TextBlock Text="{Binding Recipient}"
TextWrapping="Wrap">
<TextBlock.Style>
<Style TargetType="TextBlock"
BasedOn="{StaticResource
TextBlockStyle}">
<Setter Property="Foreground"
Value="#ccc"/>
<Style.Triggers>
<DataTrigger Binding="{Binding
IsConnected,
UpdateSourceTrigger=PropertyChanged}
" Value="True">
<Setter Property="Foreground"
```

```
Value="#fff"/>
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</StackPanel>
</Grid>
<TextBlock Grid.Column="1" Margin="5 0" TextWrapping="Wrap" VerticalAlignment="Center" Text="{Binding Message.shortMessage}" Foreground="#ccc"/>
<TextBlock Grid.Column="2" Margin="5 0" VerticalAlignment="Top" Text="{Binding SendDate, StringFormat='{}Дата отправки: {0:dd.MM.yy HH:mm}'}"/>
</Grid>
</Border>
</DataTemplate>
</ItemsControl.ItemTemplate>
</ItemsControl>
</Grid>
</Page>
```

ChatListPage.xaml.cs

```
using System.Collections.Generic;
using System.Windows.Controls;
using System.Windows.Input;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.ViewModels;
using INCOMSYSTEM.Windows;
using Page = System.Windows.Controls.Page;
namespace INCOMSYSTEM.Pages
{
public partial class ChatListPage : Page
{
public ChatListPage()
{
InitializeComponent();
}
private Dictionary<long, Page> ChatPages { get; } = new Dictionary<long, Page>();
public Chats Chat { get; set; }
public void ChatEnter_LeftBtnUp(object sender, MouseButtonEventArgs e)
{
ChatMess chatMess;
if (Chat != null)
{
if (ChatPages.ContainsKey(Chat.idChat))
{
MainWindow.ChatFrame.Navigate(ChatPages[Chat.idChat]);
}
else
{
chatMess = new ChatMess
{
Id = Chat.idChat,
IsConnected = true,
Recipient = Chat.Customers.name
};
var chatPage = new ChatPage(chatMess);
ChatPages.Add(Chat.idChat, chatPage);
MainWindow.ChatFrame.Navigate(chatPage);
}
}
else
{
chatMess = (sender as Grid).DataContext as ChatMess;
if (ChatPages.ContainsKey(chatMess.Id))
{
MainWindow.ChatFrame.Navigate(ChatPages[chatMess.Id]);
}
else
{
var chatPage = new ChatPage(chatMess);
ChatPages.Add(chatMess.Id, chatPage);
MainWindow.ChatFrame.Navigate(chatPage);
}
}
Chat = null;
}
}
}
```

ChatPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.ChatPage"
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:controls="clr-namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d">
<Grid Margin="30 10">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="100"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<ItemsControl x:Name="MessagesList" Style="{StaticResource ItemsVisible}" BorderBrush="Wheat" BorderThickness="1" Padding="10 0">
<ItemsControl.ItemTemplate>
<DataTemplate>
<Border Margin="5 3" CornerRadius="15" BorderBrush="Wheat" BorderThickness="1" Width="{Binding RelativeSource={RelativeSource Mode=FindAncestor, AncestorType=ItemsControl}, Path=ActualWidth, Converter={StaticResource WidthConverter}}">
<Border.Style>
<Style TargetType="Border">
<Setter Property="HorizontalAlignment" Value="Left"/>
<Style.Triggers>
<DataTrigger Binding="{Binding ThisUser}" Value="True">
<Setter Property="HorizontalAlignment" Value="Right"/>
</DataTrigger>
</Style.Triggers>
</Style>
</Border.Style>
<Grid Margin="5">
<Grid.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
</Grid.Resources>
<Grid.RowDefinitions>
<RowDefinition Height="30"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition />
<ColumnDefinition Width="Auto"/>
```

```
</Grid.ColumnDefinitions>
<TextBlock Text="{Binding Sender}"/>
<TextBlock Text="{Binding Message}"
Grid.Row="1"
TextWrapping="Wrap"
Margin="0 0 30 0"/>
<TextBlock Text="{Binding SendDate,
StringFormat='{}{0:dd.MM.yy HH:ss}'}"
FontSize="12"
Margin="10 0"
VerticalAlignment="Bottom"
Grid.Row="1"
Grid.Column="1"/>
</Grid>
</Border>
</DataTemplate>
</ItemsControl.ItemTemplate>
</ItemsControl>
<Border BorderBrush="Wheat"
BorderThickness="1"
Grid.Row="1"
Margin="0 10 0 0">
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<controls:InputTextBox
x:Name="InputMessageBox"
PlaceHolder="Текст"
BorderBrush="Transparent"
BorderThickness="0"
MaxLength="255"
TextAlignment="Left"
TextWrapping="Wrap"/>
<StackPanel Grid.Column="1"
VerticalAlignment="Center">
<Button x:Name="SendMessageBtn"
Content="Отправить"
Click="SendMessageBtn_Click"/>
</StackPanel>
</Grid>
</Border>
<Border Grid.Row="2"
x:Name="ErrorBorder"
Background="#560b0b"
BorderBrush="Red"
BorderThickness="1"
Visibility="Collapsed"
HorizontalAlignment="Center"
Padding="10 5"
Margin="0 5 0 15">
<TextBlock x:Name="ErrorBlock"
Foreground="White"
VerticalAlignment="Center"
TextAlignment="Center"
HorizontalAlignment="Center"
FontSize="14"/>
</Border>
</Grid>
</Page>
```

ChatPage.xaml.cs

```
using System;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Data.Entity;
using System.Linq;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Data;
using System.Windows.Input;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.ViewModels;
using INCOMSYSTEM.Windows;
using Page =
System.Windows.Controls.Page;
namespace INCOMSYSTEM.Pages
{
public partial class ChatPage : Page
{
public ChatPage(ChatMess chatMess)
{
_chatMess = chatMess;
InitializeComponent();
Title = chatMess.IsConnected ?
$"Переписка с {chatMess.Recipient}" :
chatMess.Recipient;
CollectionViewMessages.SortDescriptions.
Insert(0, new SortDescription("SendDate",
ListSortDirection.Ascending));
MessagesList.ItemsSource =
CollectionViewMessages;
InputMessageBox.KeyDown += (s, e) =>
{
if (e.Key != Key.Enter) return;
var start = InputMessageBox.SelectionStart;
InputMessageBox.Text =
InputMessageBox.Text.Insert(start++, "\n");
InputMessageBox.SelectionStart = start;
};
Task.Run(RefreshMessages);
}
private ObservableCollection<DialogMess>
MessagesCollection { get; } = new
ObservableCollection<DialogMess>();
private ICollectionView
CollectionViewMessages =>
CollectionViewSource.GetDefaultView(Me
ssagesCollection);
private readonly ChatMess _chatMess;
private async void RefreshMessages()
{
while (!MainWindow.IsClosed)
{
using (var db = await Task.Run(() => new
INCOMSYSTEMEntities()))
{
var messages = db.Messages
.Include(s => s.UsersDetail)
.Include(s => s.UsersDetail.Employees)
.Include(s => s.UsersDetail.Customers)
.Include(s => s.Chats)
.Include(s => s.Chats.Orders)
.Include(s => s.Chats.Orders.Tasks)
.Where(s => s.idChat == _chatMess.Id)
.ToList();
foreach (var message in messages)
{
var dialogMess = new DialogMess
{
Id = message.id,
Message = message.message,
File = message.attachment,
FileExtension = message.fileExtension,
SendDate = message.dateSend,
FileName = message.attachment != null ?
$"{message.Chats.Orders.Tasks.name}.{me
ssage.fileExtension}" : string.Empty
};
if (MainWindow.AuthUser.idUser ==
message.idUser) dialogMess.ThisUser =
true;
switch (message.UsersDetail.idPos)
{
case 1:
dialogMess.Sender =
message.UsersDetail.Customers.name;
break;
case 3:
dialogMess.Sender =
$"{message.UsersDetail.Employees}";
break;
}
if (MessagesCollection.FirstOrDefault(s =>
s.Id == dialogMess.Id) == null)
Application.Current.Dispatcher.Invoke(()
=> MessagesCollection.Add(dialogMess));
}
```

```
await Task.Delay(2500);
}
}
}
private bool _canSend = true;
private void SendMessageBtn_Click(object sender, RoutedEventArgs e)
{
if (InputMessageBox.IsWhiteSpace)
{
ErrorBorder.Visibility = Visibility.Visible;
SetError("Сообщение не может быть пустым");
return;
}
if (InputMessageBox.Text.Length > 255)
{
ErrorBorder.Visibility = Visibility.Visible;
SetError("Сообщение не должно превышать 255 символов");
return;
}
if (!_canSend) { ErrorBorder.Visibility = Visibility.Visible; return; }
Task.Run(async () =>
{
Application.Current.Dispatcher.Invoke(() => SendMessageBtn.IsEnabled = false);
Application.Current.Dispatcher.Invoke(() => _canSend = false);
for (var i = 5; i >= 0; i--)
{
string timerStr;
if (i > 1) timerStr = $"{i} секунды";
else if (i == 1) timerStr = $"{i} секунду";
else timerStr = $"{i} секунд";
Application.Current.Dispatcher.Invoke(() => SetError($"Перед отправкой следующего сообщения подождите {timerStr}"));
await Task.Delay(1000);
}
Application.Current.Dispatcher.Invoke(() => _canSend = true);
Application.Current.Dispatcher.Invoke(() => SendMessageBtn.IsEnabled = true);
HideError();
});
HideError();
SendMessage();
}
private async void SendMessage()
{
using (var db = await Task.Run(() => new INCOMSYSTEMEntities()))
{
var message = new Messages
{
idChat = _chatMess.Id,
idUser = MainWindow.AuthUser.idUser,
message = InputMessageBox.Text,
dateSend = DateTime.Now
};
db.Messages.Add(message);
await db.SaveChangesAsync();
var dialogMess = new DialogMess
{
Id = message.id,
Message = message.message,
SendDate = message.dateSend,
ThisUser = true,
};
switch (MainWindow.AuthUser.idPos)
{
case 1:
dialogMess.Sender = MainWindow.AuthUser.Customers.name;
break;
case 3:
dialogMess.Sender = $"{MainWindow.AuthUser.Employees}";
break;
}
Application.Current.Dispatcher.Invoke(() => MessagesCollection.Add(dialogMess));
InputMessageBox.Clear();
}
if (MessagesCollection.Count >= 2) return;
Application.Current.Dispatcher.Invoke(() =>
{
var style = MessagesCollection.Count < 2 ? MessagesList.Style : null;
MessagesList.Style = null;
MessagesList.Style = style;
});
}
private void SetError(string error)
{
ErrorBlock.Text = error;
}
private void HideError()
{
ErrorBorder.Visibility = Visibility.Collapsed;
ErrorBlock.Text = string.Empty;
}
}
internal class DialogMess
{
public long Id { get; set; }
public string Sender { get; set; }
public string Message { get; set; }
public byte[] File { get; set; }
public string FileExtension { get; set; }
public string FileName { get; set; }
public DateTime SendDate { get; set; }
public bool ThisUser { get; set; }
}
}
```

Приложение Е.

ManagerPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.MainPages.ManagerPage"
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"
Title="Главное окно" Background="DimGray">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
<Style TargetType="Label" BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid HorizontalAlignment="Center" VerticalAlignment="Center">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition/>
<RowDefinition/>
</Grid.RowDefinitions>
<Grid>
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="350"/>
</Grid.ColumnDefinitions>
<Button Margin="5" Content="Задачи" Click="ViewTasksBtn_Click"/>
<TextBlock Grid.Column="1" TextWrapping="Wrap" Text="Перейдя в окно с задачи, появится список задач с возможность их фильтрации, так же можно их добавить, редактировать или удалить"/>
<Rectangle Grid.Row="1" Grid.Column="0" Margin="15 5" Height="3" Fill="Gray" RadiusX="50" RadiusY="50"/>
<Rectangle Grid.Row="1" Grid.Column="1" Margin="15 5" Height="3" Fill="White" RadiusX="50" RadiusY="50"/>
</Grid>
<Grid Grid.Row="1">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="300"/>
</Grid.ColumnDefinitions>
<Button Margin="5" Content="Заказы" Click="ViewOrderBtn_Click"/>
<TextBlock Grid.Column="1" TextWrapping="Wrap" Text="Перейдя в окно с заказами, появится список заказов, где можно просмотреть, изменить, перейти к чату заказа, сформировать договор"/>
<Rectangle Grid.Row="1" Grid.Column="0" Margin="15 5" Height="3" Fill="Gray" RadiusX="50" RadiusY="50"/>
<Rectangle Grid.Row="1" Grid.Column="1" Margin="15 5" Height="3" Fill="White" RadiusX="50" RadiusY="50"/>
</Grid>
<Grid Grid.Row="2">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="300"/>
</Grid.ColumnDefinitions>
<Button Margin="5" Content="Пользователи"/>
<TextBlock Grid.Column="1" TextWrapping="Wrap" Text="Перейдя в окно с пользователями, появится список пользователей с возможность фильтрации поиска, так же можно просмотреть аккаунт пользователя и перейти к чату, если существует"/>
</Grid>
</Grid>
</Page>
```

ManagerPage.xaml.cs

```
using System.Windows;
using System.Windows.Controls;
using INCOMSYSTEM.Windows;
namespace INCOMSYSTEM.Pages.MainPages
{
public partial class ManagerPage : Page
{
public ManagerPage()
{
InitializeComponent();
}
private void ViewTasksBtn_Click(object sender, RoutedEventArgs e)
{
MainWindow.ReviewFrame.Navigate(new ViewTasksPage());
}
private void ViewOrderBtn_Click(object sender, RoutedEventArgs e)
{
MainWindow.ReviewFrame.Navigate(new ViewOrdersPage());
}
}
}
```

ViewOrdersPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.MainPages.ViewOrdersPage"
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:gl="clr-namespace:System.Globalization;assembly=mscorlib"
xmlns:wnd="clr-namespace:INCOMSYSTEM.Windows"
mc:Ignorable="d"
Title="Список заказов">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
<Style TargetType="Label" BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid HorizontalAlignment="Center">
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
</Grid.RowDefinitions>
<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Margin="0 5">
<TextBlock Text="{Binding ElementName=OrdersList, Path=Items.Count}"/>
<TextBlock Text=" из "/>
<TextBlock x:Name="AllItemsCount"/>
</StackPanel>
<ScrollViewer Grid.Row="1" VerticalScrollBarVisibility="Disabled" HorizontalScrollBarVisibility="Auto">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
</Grid.RowDefinitions>
<Grid>
<Grid.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource ComboBoxTextBlockStyle}">
<Setter Property="Foreground" Value="White"/>
<Setter Property="TextAlignment" Value="Center"/>
<Setter Property="VerticalAlignment" Value="Center"/>
<Setter Property="Background" Value="DimGray"/>
<Setter Property="FontWeight" Value="Bold"/>
</Style>
<Style TargetType="Border">
<Setter Property="Background" Value="DimGray"/>
<Setter Property="BorderBrush" Value="Black"/>
<Setter Property="BorderThickness" Value="1"/>
</Style>
</Grid.Resources>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="225"/>
<ColumnDefinition Width="100"/>
</Grid.ColumnDefinitions>
<Border>
<TextBlock Text="№заказа"/>
</Border>
<Border Grid.Column="1">
<TextBlock Text="Заказчик"/>
</Border>
<Border Grid.Column="2" Width="200">
<Border.Style>
<Style TargetType="Border" BasedOn="{StaticResource {x:Type Border}}">
<Setter Property="Visibility" Value="Visible"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static wnd:MainWindow.AuthUser}, Path=idPos}" Value="2">
<Setter Property="Visibility" Value="Collapsed"/>
</DataTrigger>
</Style.Triggers>
</Style>
</Border.Style>
<TextBlock Text="Исполнитель"/>
</Border>
<Border Grid.Column="3">
<TextBlock Text="Менеджер"/>
</Border>
<Border Grid.Column="4">
<TextBlock Text="Задача"/>
</Border>
<Border Grid.Column="5">
<TextBlock Text="Цена"/>
</Border>
<Border Grid.Column="6">
<TextBlock Text="Сложность"/>
</Border>
<Border Grid.Column="7">
<TextBlock Text="Дата формирования заказа"/>
</Border>
<Border Grid.Column="8">
<TextBlock Text="Статус"/>
</Border>
</Grid>
<ItemsControl x:Name="OrdersList" Grid.Row="1" Style="{StaticResource ItemsVisible}">
<ItemsControl.ItemTemplate>
<DataTemplate>
<CheckBox x:Name="OrderGrid" Margin="0 0 0 10" Style="{StaticResource ClearCheckBoxTemplate}">
<CheckBox.ContextMenu>
<ContextMenu>
<MenuItem Header="Просмотреть заказ" Click="ViewOrderMenu_Click" Visibility="{Binding CanViewDetail}" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}"/>
<MenuItem Header="Перейти к чату" Click="OpenChatOrderMenu_Click" Visibility="{Binding CanJoinChat}" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}"/>
<MenuItem Header="Сформировать договор" Click="FormAgreementOrderMenu_Click" Visibility="{Binding CanFormAgreement}" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}"/>
<MenuItem Header="Установить фактическую дату начала" Click="SetFactDateStartMenu_Click" Visibility="{Binding CanSetFactStartDate}" CommandParameter="{Binding RelativeSource={RelativeSource
```

```
Mode=Self},
Path=DataContext}"/>
<MenuItem Header="Установить фактическую дату завершения" Click="SetFactDateCompleteMenu_Click" Visibility="{Binding CanSetFactCompleteDate}" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self},
Path=DataContext}"/>
</ContextMenu>
</CheckBox.ContextMenu>
<CheckBox.Content>
<Grid>
<Grid.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource ComboBoxTextBlockStyle}">
<Setter Property="Foreground" Value="White"/>
<Setter Property="TextAlignment" Value="Center"/>
<Setter Property="VerticalAlignment" Value="Center"/>
<Setter Property="Background" Value="DimGray"/>
</Style>
<Style TargetType="Border">
<Setter Property="Background" Value="DimGray"/>
<Setter Property="BorderBrush" Value="Black"/>
<Setter Property="BorderThickness" Value="1"/>
<Setter Property="Padding" Value="5"/>
</Style>
</Grid.Resources>
<Grid.RowDefinitions>
<RowDefinition Height="40"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="100"/>
<ColumnDefinition Width="225"/>
<ColumnDefinition Width="100"/>
</Grid.ColumnDefinitions>
<Border>
<TextBlock Text="{Binding id}"/>
</Border>
<Border Grid.Column="1">
<TextBlock Text="{Binding Customers.name}"/>
</Border>
<Border Grid.Column="2" Width="200">
<Border.Style>
<Style TargetType="Border" BasedOn="{StaticResource {x:Type Border}}">
<Setter Property="Visibility" Value="Visible"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static wnd:MainWindow.AuthUser}, Path=idPos}" Value="2">
<Setter Property="Visibility" Value="Collapsed"/>
</DataTrigger>
</Style.Triggers>
</Style>
</Border.Style>
<TextBlock Text="{Binding Employees, TargetNullValue='Отсутствует'}"/>
</Border>
<Border Grid.Column="3">
<TextBlock Text="{Binding Chats.Employees, TargetNullValue='Отсутствует'}"/>
</Border>
<Border Grid.Column="4">
<TextBlock Text="{Binding Tasks.name}"/>
</Border>
<Border Grid.Column="5">
<TextBlock Text="{Binding price, StringFormat='{}{0:0,0} руб.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}}"/>
</Border>
<Border Grid.Column="6">
<TextBlock Text="{Binding difficulty}"/>
</Border>
<Border Grid.Column="7">
<TextBlock Text="{Binding dateOrder, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}}"/>
</Border>
<Border Grid.Column="8">
<TextBlock Text="{Binding Statuses.name}"/>
</Border>
<UniformGrid Grid.Row="1" Grid.ColumnSpan="9" Grid.Column="0" Columns="4" Height="80">
<UniformGrid.Style>
<Style TargetType="UniformGrid">
<Setter Property="Visibility" Value="Collapsed"/>
<Style.Triggers>
<DataTrigger Binding="{Binding ElementName=OrderGrid, Path=IsChecked}" Value="True">
<Setter Property="Visibility" Value="Visible"/>
</DataTrigger>
</Style.Triggers>
</Style>
</UniformGrid.Style>
<Border>
<TextBlock Text="Плановая дата начала" FontWeight="Bold"/>
</Border>
<Border>
<TextBlock Text="Фактическая дата начала" FontWeight="Bold"/>
</Border>
<Border>
<TextBlock Text="Плановая дата завершения" FontWeight="Bold"/>
</Border>
<Border>
<TextBlock Text="Фактическая дата завершения" FontWeight="Bold"/>
</Border>
<Border>
<TextBlock Text="{Binding planDateStart, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}"/>
</Border>
<Border>
<TextBlock Text="{Binding factDateStart, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}">
<TextBlock.Style>
```

```
<Style TargetType="TextBlock" BasedOn="{StaticResource {x:Type TextBlock}}">
<Style.Triggers>
<DataTrigger Binding="{Binding IsStartLateYellow}" Value="True">
<Setter Property="Foreground" Value="Yellow"/>
</DataTrigger>
<DataTrigger Binding="{Binding IsStartLateRed}" Value="True">
<Setter Property="Foreground" Value="Red"/>
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
<Border>
<TextBlock Text="{Binding planDateComplete, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}"/>
</Border>
<Border>
<TextBlock Text="{Binding factDateComplete, StringFormat='{}{0:dd MMMM yyyy} г.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue='Отсутствует'}">
<TextBlock.Style>
<Style TargetType="TextBlock" BasedOn="{StaticResource {x:Type TextBlock}}">
<Style.Triggers>
<DataTrigger Binding="{Binding IsCompleteLateYellow}" Value="True">
<Setter Property="Foreground" Value="Yellow"/>
</DataTrigger>
<DataTrigger Binding="{Binding IsCompleteLateRed}" Value="True">
<Setter Property="Foreground" Value="Red"/>
</DataTrigger>
</Style.Triggers>
</Style>
</TextBlock.Style>
</TextBlock>
</Border>
</UniformGrid>
</Grid>
</CheckBox.Content>
</CheckBox>
</DataTemplate>
</ItemsControl.ItemTemplate>
</ItemsControl>
</Grid>
</ScrollViewer>
</Grid>
</Page>
```

ViewOrdersPage.xaml.cs

```
using System;
using System.Data.Entity;
using System.Diagnostics;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using INCOMSYSTEM.BehaviorsFiles;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Pages.MainPages.Views;
using INCOMSYSTEM.Windows;
using Microsoft.Win32;
using Word = Microsoft.Office.Interop.Word;
namespace INCOMSYSTEM.Pages.MainPages
{
public partial class ViewOrdersPage : Page
{
public ViewOrdersPage()
{
InitializeComponent();
using (var db = new INCOMSYSTEMEntities())
{
AllItemsCount.Text = db.Orders
.Count(s => MainWindow.AuthUser.idPos != 2 || s.idExecutor == MainWindow.AuthUser.idUser)
.ToString();
}
ApplyFilter();
}
private void ApplyFilter()
{
using (var db = new INCOMSYSTEMEntities())
{
OrdersList.ItemsSource = db.Orders
.Include(s => s.Customers)
.Include(s => s.Employees)
.Include(s => s.Employees.UsersDetail)
.Include(s => s.Customers.UsersDetail)
.Include(s => s.Statuses)
.Include(s => s.Tasks)
.Include(s => s.Chats)
.Include(s => s.Chats.Employees)
.Where(s => MainWindow.AuthUser.idPos != 2 || s.idExecutor == MainWindow.AuthUser.idUser)
.ToList();
}
}
private void ViewOrderMenu_Click(object sender, RoutedEventArgs e)
{
var order = ((MenuItem)sender).CommandParameter as Orders;
var addWindow = new AdditionalWindow();
addWindow.MFrame.Navigate(new ViewDetailOrderPage(order));
if (addWindow.ShowDialog() != true) return;
ApplyFilter();
}
private void OpenChatOrderMenu_Click(object sender, RoutedEventArgs e)
{
var order = ((MenuItem)sender).CommandParameter as Orders;
if (order.Chats.idManager == null && MessageBox.Show("Перейти к чату?", "Подтверждение", MessageBoxButton.YesNo) != MessageBoxResult.Yes) return;
using (var db = new INCOMSYSTEMEntities())
{
var chat = db.Chats.Include(s => s.Customers).First(s => s.idChat == order.id);
var orderDb = db.Orders.First(s => s.id == order.id);
if(orderDb.idStatus < 2) orderDb.idStatus = 2;
chat.idManager =
```

```
MainWindow.AuthUser.idUser;
db.SaveChanges();
Application.Current.Windows.OfType<MainWindow>().First().GoChat(chat);
}
ApplyFilter();
}
private void FormAgreementOrderMenu_Click(object sender, RoutedEventArgs e)
{
var order = ((MenuItem)sender).CommandParameter as Orders;
var saveFileDialog = new SaveFileDialog
{
FileName = $"Договор №{order.id}",
Filter = "Documents | *.docx, *.doc",
DefaultExt = ".docx",
OverwritePrompt = true
};
if (saveFileDialog.ShowDialog() != true) return;
if (!FormAgreement(saveFileDialog.FileName, order)) return;
if (MessageBox.Show("Договор успешно сохранён. Открыть?", "Успешное сохранение", MessageBoxButton.YesNo, MessageBoxImage.Question) != MessageBoxResult.Yes) return;
Process.Start(saveFileDialog.FileName);
}
private static bool FormAgreement(string path, Orders order)
{
var wApp = new Word.Application();
var doc = wApp.Documents.Open(Directory.GetCurrentDirectory() + @"\Resources\AgreementTemplate.docx");
while (true)
{
try
{
doc.ReplaceWordText("{dayOrder}", order.dateOrder.ToString("dd"));
doc.ReplaceWordText("{monthOrder}", order.dateOrder.ToString("MMMM", CultureInfo.CurrentCulture));
doc.ReplaceWordText("{yearOrder}", order.dateOrder.ToString("yy"));
doc.ReplaceWordText("{customerName}", order.Customers.name);
var employeePatronymic = order.Employees.patronymic != null ? $" {order.Employees.patronymic}" : string.Empty;
doc.ReplaceWordText("{executorName}", $"{order.Employees.surname} {order.Employees.name}{employeePatronymic}");
doc.ReplaceWordText("{passportCustomer}", order.Customers.UsersDetail.passport);
doc.ReplaceWordText("{passportExecutor}", order.Employees.UsersDetail.passport);
doc.ReplaceWordText("{taskName}", order.Tasks.name);
doc.ReplaceWordText("{planDayStart}", order.planDateStart.Value.ToString("dd"));
doc.ReplaceWordText("{planMonthStart}", order.planDateStart.Value.ToString("MMMM", CultureInfo.CurrentCulture));
doc.ReplaceWordText("{planYearStart}", order.planDateStart.Value.ToString("yy"));
doc.ReplaceWordText("{planDayComplete}", order.planDateComplete.Value.ToString("dd"));
doc.ReplaceWordText("{planMonthComplete}", order.planDateComplete.Value.ToString("MMMM", CultureInfo.CurrentCulture));
doc.ReplaceWordText("{planYearComplete}", order.planDateComplete.Value.ToString("yy"));
doc.ReplaceWordText("{supportPeriod}", order.Tasks.supportPeriod);
doc.ReplaceWordText("{taskPrice}", (int)order.Tasks.newPrice);
doc.ReplaceWordText("{taskPriceInWords}", ((int)order.Tasks.newPrice).GetNumberInWords().Trim());
var taskDifficultyPrice = order.Tasks.newPrice * (decimal)order.difficulty;
doc.ReplaceWordText("{taskDifficultyPrice}", (int)taskDifficultyPrice);
doc.ReplaceWordText("{taskDifficultyPriceInWords}", ((int)taskDifficultyPrice).GetNumberInWords().Trim());
doc.ReplaceWordText("{orderPrice}", (int)order.price);
doc.ReplaceWordText("{orderPriceInWords}", ((int)order.price).GetNumberInWords().Trim());
doc.ReplaceWordText("{customerPhone}", order.Customers.UsersDetail.phone != null ? order.Customers.UsersDetail.phone.Value.ToString() : "Отсутствует");
doc.ReplaceWordText("{executorPhone}", order.Employees.UsersDetail.phone != null ? order.Employees.UsersDetail.phone.Value.ToString() : "Отсутствует");
doc.ReplaceWordText("{customerSeriePassport}", order.Customers.UsersDetail.SeriePassport);
doc.ReplaceWordText("{customerNumberPassport}", order.Customers.UsersDetail.NumberPassport);
doc.ReplaceWordText("{executorSeriePassport}", order.Employees.UsersDetail.SeriePassport);
doc.ReplaceWordText("{executorNumberPassport}", order.Employees.UsersDetail.NumberPassport);
doc.ReplaceWordText("{customerAddress}", order.Customers.UsersDetail.address);
doc.ReplaceWordText("{executorAddress}", order.Employees.UsersDetail.address);
break;
}
catch
{
var result = MessageBox.Show("Для сохранения записи, необходимо закрыть Word документ", "Ошибка", MessageBoxButton.YesNo, MessageBoxImage.Error);
if (result != MessageBoxResult.No) continue;
doc.Close();
wApp.Quit();
return false;
}
}
doc.SaveAs(path);
doc.Close();
```

```
wApp.Quit();
return true;
}
private void SetFactDateStartMenu_Click(object sender, RoutedEventArgs e)
{
if (MessageBox.Show("Установить текущую дату начала?", "Подтверждение", MessageBoxButton.YesNo) != MessageBoxResult.Yes) return;
var order = ((MenuItem)sender).CommandParameter as Orders;
using (var db = new INCOMSYSTEMEntities())
{
var orderDb = db.Orders.First(s => s.id == order.id);
orderDb.factDateStart = DateTime.Now;
orderDb.idStatus = 3;
db.SaveChanges();
}
ApplyFilter();
}
private void SetFactDateCompleteMenu_Click(object sender, RoutedEventArgs e)
{
if (MessageBox.Show("Установить текущую дату завершения?", "Подтверждение", MessageBoxButton.YesNo) != MessageBoxResult.Yes) return;
var order = ((MenuItem)sender).CommandParameter as Orders;
using (var db = new INCOMSYSTEMEntities())
{
var orderDb = db.Orders.First(s => s.id == order.id);
orderDb.factDateComplete = DateTime.Now;
orderDb.idStatus = 4;
db.SaveChanges();
}
ApplyFilter();
}
}
}
```

ViewTasksPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.MainPages.ViewTasksPage"
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:windows="clr-namespace:INCOMSYSTEM.Windows"
xmlns:controls="clr-namespace:INCOMSYSTEM.Controls"
xmlns:gl="clr-namespace:System.Globalization;assembly=mscorlib"
mc:Ignorable="d"
Title="Список задач">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
<Style TargetType="Label" BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<StackPanel Orientation="Horizontal">
<TextBlock Text="{Binding ElementName=TasksList, Path=Items.Count}"/>
<TextBlock Text=" из "/>
<TextBlock x:Name="AllCountTask"/>
</StackPanel>
<ItemsControl x:Name="TasksList" Grid.Row="1">
<ItemsControl.ContextMenu>
<ContextMenu>
<ContextMenu.Style>
<Style TargetType="ContextMenu" BasedOn="{StaticResource {x:Type ContextMenu}}">
<Setter Property="Visibility" Value="Collapsed"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static windows:MainWindow.AuthUser}, Path=idPos}" Value="3">
<Setter Property="Visibility" Value="Visible"/>
</DataTrigger>
</Style.Triggers>
</Style>
</ContextMenu.Style>
<MenuItem Header="Добавить" Click="AddTaskMenu_Click"/>
</ContextMenu>
</ItemsControl.ContextMenu>
<ItemsControl.Resources>
<Style TargetType="TextBlock" x:Key="disc" BasedOn="{StaticResource {x:Type TextBlock}}">
<Setter Property="Visibility" Value="Visible"/>
<Style.Triggers>
<DataTrigger Binding="{Binding discoutStyle}" Value="True">
<Setter Property="Foreground" Value="Yellow"/>
</DataTrigger>
<DataTrigger Binding="{Binding discountVisible}" Value="False">
<Setter Property="Visibility" Value="Collapsed"/>
</DataTrigger>
</Style.Triggers>
</Style>
<Style TargetType="TextBlock" x:Key="oldPrice" BasedOn="{StaticResource {x:Type TextBlock}}">
<Style.Triggers>
<DataTrigger Binding="{Binding discountVisible}" Value="True">
<Setter Property="Foreground" Value="#ccc"/>
<Setter Property="TextDecorations" Value="Strikethrough"/>
</DataTrigger>
</Style.Triggers>
</Style>
<Style TargetType="Button" x:Key="customerBtn" BasedOn="{StaticResource {x:Type Button}}">
```

```
<Setter Property="Visibility" Value="Collapsed"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static windows:MainWindow.AuthUser}, Path=idPos}" Value="1">
<Setter Property="Visibility" Value="Visible"/>
</DataTrigger>
</Style.Triggers>
</Style>
<Style TargetType="Button" x:Key="managerBtn" BasedOn="{StaticResource {x:Type Button}}">
<Setter Property="Visibility" Value="Collapsed"/>
<Style.Triggers>
<DataTrigger Binding="{Binding Source={x:Static windows:MainWindow.AuthUser}, Path=idPos}" Value="3">
<Setter Property="Visibility" Value="Visible"/>
</DataTrigger>
</Style.Triggers>
</Style>
</ItemsControl.Resources>
<ItemsControl.ItemsPanel>
<ItemsPanelTemplate>
<WrapPanel Orientation="Horizontal"/>
</ItemsPanelTemplate>
</ItemsControl.ItemsPanel>
<ItemsControl.ItemTemplate>
<DataTemplate>
<Border Width="400" Height="160" BorderThickness="1" BorderBrush="White" CornerRadius="10" Padding="5" Margin="5">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition />
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="100"/>
</Grid.ColumnDefinitions>
<TextBlock Text="{Binding name}" Grid.Row="0" Style="{StaticResource TextBlockStyle}" FontSize="18" FontWeight="Bold"/>
<TextBlock Text="{Binding shortDescription}" TextWrapping="Wrap" Style="{StaticResource TextBlockStyle}" Grid.Row="1">
<TextBlock.ToolTip>
<TextBlock Text="{Binding description}"/>
</TextBlock.ToolTip>
</TextBlock>
<StackPanel Grid.Row="2" Margin="0 5">
<TextBlock Text="{Binding Specializations.name, StringFormat='{}Область: {0}'}" TextWrapping="Wrap" Style="{StaticResource TextBlockStyle}"/>
<TextBlock Text="{Binding approxString, StringFormat='{}Примерные дни выполнения: {0}'}" Style="{StaticResource TextBlockStyle}"/>
</StackPanel>
<Border Grid.Column="1" Grid.Row="0" Grid.RowSpan="4" BorderThickness="1 0 0 0" BorderBrush="White" Margin="5 0 0 0">
<StackPanel HorizontalAlignment="Center" VerticalAlignment="Center">
<TextBlock Text="{Binding price, StringFormat='{}{0:0,0} руб.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}}" TextWrapping="Wrap" Style="{StaticResource oldPrice}"/>
<TextBlock Text="{Binding discount, StringFormat='{}{0}%', TargetNullValue={x:Null}}" TextWrapping="Wrap" Style="{StaticResource disc}" Margin="0 5"/>
<TextBlock Text="{Binding newPrice, StringFormat='{}{0:0,0} руб.', ConverterCulture={x:Static gl:CultureInfo.CurrentCulture}, TargetNullValue={x:Null}}" Style="{StaticResource disc}" TextWrapping="Wrap"/>
</StackPanel>
</Border>
<StackPanel Grid.Row="3" HorizontalAlignment="Left" Margin="0 5 0 0" Orientation="Horizontal">
<Button Content="Подробнее" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}" Click="SelectBtn_Click" Style="{StaticResource customerBtn}"/>
<Button Content="Редактировать" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}" Click="EditBtn_Click" Style="{StaticResource managerBtn}"/>
<Button Content="Удалить" Margin="5 0" CommandParameter="{Binding RelativeSource={RelativeSource Mode=Self}, Path=DataContext}" Click="DeleteBtn_Click" Style="{StaticResource managerBtn}"/>
</StackPanel>
</Grid>
</Border>
</DataTemplate>
</ItemsControl.ItemTemplate>
<ItemsControl.Template>
<ControlTemplate>
<ScrollViewer VerticalScrollBarVisibility="Auto" HorizontalScrollBarVisibility="Disabled">
<ItemsPresenter/>
</ScrollViewer>
</ControlTemplate>
</ItemsControl.Template>
</ItemsControl>
<Grid Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="0" Grid.RowSpan="2" HorizontalAlignment="Right">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
</Grid.ColumnDefinitions>
<Button VerticalAlignment="Top" Click="SettingsFilterBtn_Click">
```

```
<Button.Content>
<Image Source="../../Images/gear.png"/>
</Button.Content>
</Button>
<Border BorderBrush="Coral" BorderThickness="1 0 0 0" Grid.Column="1"
Width="0"
x:Name="SideBarFilter">
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
</Grid.ColumnDefinitions>
<Border Background="#F0696969" Grid.RowSpan="5"
Grid.ColumnSpan="5"/>
<StackPanel>
<TextBlock Height="30" Margin="0 5" Text="Фильтр по названию: "/>
<TextBlock Height="30" Margin="0 5" Text="Сортировка по области: "/>
<TextBlock Height="30" Margin="0 5" Text="Сортировка по цене: "/>
<TextBlock Height="30" Margin="0 5" Text="Диапазон цены: "/>
</StackPanel>
<StackPanel Grid.Column="1" Margin="50">
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<controls:InputTextBox
x:Name="FilterText" Height="30" Width="150" PlaceHolder="Введите текст"
TextChanged="FilterText_TextChanged"/>
<Button x:Name="ClearBtn" Grid.Column="1" Width="25"
VerticalAlignment="Center"
Click="ClearBtn_Click">
<Button.Style>
<Style TargetType="Button" BasedOn="{StaticResource {x:Type Button}}">
<Setter Property="IsEnabled" Value="True"/>
<Style.Triggers>
<DataTrigger Binding="{Binding ElementName=FilterText,
Path=IsWhiteSpace}" Value="True">
<Setter Property="IsEnabled" Value="False"/>
</DataTrigger>
</Style.Triggers>
</Style>
</Button.Style>
<TextBlock Text="r" FontFamily="Webdings"/>
</Button>
</Grid>
<ComboBox x:Name="SpecBox" Height="30"
Margin="0 5" SelectedIndex="0"
SelectionChanged="FilterBox_SelectionChanged">
<ComboBox.ItemTemplate>
<DataTemplate>
<TextBlock Text="{Binding name}" Style="{StaticResource
ComboBoxTextBlockStyle}"/>
</DataTemplate>
</ComboBox.ItemTemplate>
</ComboBox>
<ComboBox x:Name="PriceBox" Height="30"
Margin="0 5" SelectedIndex="0"
SelectionChanged="FilterBox_SelectionChanged">
<TextBlock Text="По возрастанию"/>
<TextBlock Text="По убыванию"/>
</ComboBox>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition/>
</Grid.ColumnDefinitions>
<StackPanel>
<TextBlock Text="От" TextAlignment="Center"/>
<Slider
ValueChanged="Slider_ValueChanged"
Minimum="0"
x:Name="LeftSlider"
Maximum="{Binding ElementName=RightSlider, Path=Value, Mode=TwoWay,
UpdateSourceTrigger=PropertyChanged}
"/>
<TextBox Text="{Binding ElementName=LeftSlider, Path=Value,
StringFormat='{}{0:0}', Mode=TwoWay,
UpdateSourceTrigger=PropertyChanged}
"
TextAlignment="Center"
Style="{StaticResource TextBoxStyle}"/>
</StackPanel>
<StackPanel Grid.Column="1">
<TextBlock Text="До" TextAlignment="Center"/>
<Slider
ValueChanged="Slider_ValueChanged"
Minimum="{Binding ElementName=LeftSlider, Path=Value, Mode=TwoWay,
UpdateSourceTrigger=PropertyChanged}
"
x:Name="RightSlider"/>
<TextBox Text="{Binding ElementName=RightSlider, Path=Value,
StringFormat='{}{0:0}', Mode=TwoWay,
UpdateSourceTrigger=PropertyChanged}
"
TextAlignment="Center"
Style="{StaticResource TextBoxStyle}"/>
</StackPanel>
</Grid>
</StackPanel>
</Grid>
</Border>
</Grid>
</Grid>
</Page>
```

ViewTasksPage.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media.Animation;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Pages.Details;
using
INCOMSYSTEM.Pages.MainPages.Views;
using INCOMSYSTEM.Windows;
namespace
INCOMSYSTEM.Pages.MainPages
{
public partial class ViewTasksPage
{
public ViewTasksPage()
{
InitializeComponent();
```

```
using (var db = new INCOMSYSTEMEntities())
{
AllCountTask.Text = db.Tasks.Count().ToString();
var max = db.Tasks.Max(s => s.price);
RightSlider.Maximum = (long)max;
LeftSlider.Value = 0;
RightSlider.Value = (long)max;
TasksList.ItemsSource = db.Tasks.Include(s => s.Specializations).ToList();
var list = new List<Specializations> { new Specializations { name = "Очистить" } };
list.AddRange(db.Specializations.ToList());
SpecBox.ItemsSource = list;
}
}
private void SelectBtn_Click(object sender, RoutedEventArgs e)
{
var task = (Tasks)((Button)sender).CommandParameter;
MainWindow.ReviewFrame.Navigate(new ViewDetailTaskPage(task));
}
private void FilterBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
ApplyFilter();
}
private void Slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
{
ApplyFilter();
}
private void SettingsFilterBtn_Click(object sender, RoutedEventArgs e)
{
var anim = new DoubleAnimation
{
Duration = TimeSpan.FromSeconds(0.7d),
From = SideBarFilter.ActualWidth,
EasingFunction = new SineEase { EasingMode = EasingMode.EaseInOut },
To = _isShowed ? 0 : 350
};
_isShowed = !_isShowed;
SideBarFilter.BeginAnimation(WidthProperty, anim);
}
private void ClearBtn_Click(object sender, RoutedEventArgs e)
{
FilterText.Clear();
}
private bool _isShowed;
private void ApplyFilter()
{
if (PriceBox == null || FilterText == null || TasksList == null || SpecBox == null || LeftSlider == null || RightSlider == null) return;
using (var db = new INCOMSYSTEMEntities())
{
IEnumerable<Tasks> list = db.Tasks.Include(s => s.Specializations);
if(!FilterText.IsWhiteSpace) list = list.Where(s => s.name.ToLower().Trim().Contains(FilterText.Text.ToLower().Trim())).ToList();
if (SpecBox.SelectedIndex > 0) list = list.Where(s => s.idSpecialization == ((Specializations)SpecBox.SelectedItem).id).ToList();
switch (PriceBox.SelectedIndex)
{
case 0:
list = list.OrderBy(s => s.discount != null && s.discount > 0 ? s.newPrice : s.price);
break;
case 1:
list = list.OrderByDescending(s => s.discount != null && s.discount > 0 ? s.newPrice : s.price);
break;
}
var min = (decimal)LeftSlider.Value;
var max = (decimal)RightSlider.Value;
list = list.Where(s => (s.discount != null && s.discount > 0 ? s.newPrice : s.price) >= min && (s.discount != null && s.discount > 0 ? s.newPrice : s.price) <= max);
TasksList.ItemsSource = list.ToList();
}
}
private void FilterText_TextChanged(object sender, TextChangedEventArgs e)
{
if (ClearBtn == null || FilterText == null) return;
ApplyFilter();
}
private void EditBtn_Click(object sender, RoutedEventArgs e)
{
var task = (Tasks)((Button)sender).CommandParameter;
var addWindow = new AdditionalWindow();
addWindow.MFrame.Navigate(new TaskDetailPage(task));
if (addWindow.ShowDialog() != true) return;
using (var db = new INCOMSYSTEMEntities())
{
var max = db.Tasks.Max(s => s.price);
RightSlider.Maximum = (long)max;
}
ApplyFilter();
}
private void DeleteBtn_Click(object sender, RoutedEventArgs e)
{
if (MessageBox.Show("Вы действительно хотите удалить эту задачу?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Warning) != MessageBoxResult.Yes) return;
var task = (Tasks)((Button)sender).CommandParameter;
using (var db = new INCOMSYSTEMEntities())
{
db.Tasks.Remove(db.Tasks.First(s => s.id == task.id));
db.SaveChanges();
var max = db.Tasks.Max(s => s.price);
RightSlider.Maximum = (long)max;
}
ApplyFilter();
}
private void AddTaskMenu_Click(object sender, RoutedEventArgs e)
{
var addWindow = new AdditionalWindow();
addWindow.MFrame.Navigate(new TaskDetailPage());
if (addWindow.ShowDialog() != true) return;
```

```
using (var db = new INCOMSYSTEMEntities())
{
var max = db.Tasks.Max(s => s.price);
RightSlider.Maximum = (long)max;
}
ApplyFilter();
}
}
}
```

Приложение F.

ViewDetailTaskPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.MainPages.Views.ViewDetailTaskPage"
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"
Title="Просмотр задачи">
<Page.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}"/>
<Style TargetType="Label" BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid VerticalAlignment="Center" HorizontalAlignment="Center" Width="400">
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<TextBlock FontSize="20" x:Name="TitleTask" TextAlignment="Center"/>
<Border BorderBrush="Coral" BorderThickness="1" Padding="5" Grid.Row="1">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<CheckBox Visibility="Collapsed" x:Name="discountStyle"/>
<CheckBox Visibility="Collapsed" x:Name="discountVisible"/>
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Border BorderBrush="White" BorderThickness="0 0 0 1" Grid.ColumnSpan="3"/>
<StackPanel Grid.Column="2">
<TextBlock Text="Область направления:" TextAlignment="Center"/>
<TextBlock x:Name="SpecTask"/>
</StackPanel>
</Grid>
<Border BorderBrush="White" BorderThickness="0 0 0 1" Grid.ColumnSpan="3" Grid.Row="1"/>
<TextBlock x:Name="DescriptionTask" TextWrapping="Wrap" Margin="0 10" Grid.Row="1" Text="{Binding Description}"/>
<Grid Grid.Row="2" Margin="0 5">
<Grid.Resources>
<Style TargetType="TextBlock" x:Key="disc" BasedOn="{StaticResource {x:Type TextBlock}}">
<Setter Property="Visibility" Value="Visible"/>
<Style.Triggers>
<DataTrigger Binding="{Binding ElementName=discountStyle, Path=IsChecked}" Value="True">
<Setter Property="Foreground" Value="Yellow"/>
</DataTrigger>
<DataTrigger Binding="{Binding ElementName=discountVisible, Path=IsChecked}" Value="False">
<Setter Property="Visibility" Value="Collapsed"/>
</DataTrigger>
</Style.Triggers>
</Style>
<Style TargetType="TextBlock" x:Key="oldPrice" BasedOn="{StaticResource {x:Type TextBlock}}">
<Style.Triggers>
<DataTrigger Binding="{Binding ElementName=discountStyle, Path=IsChecked}" Value="True">
<Setter Property="Foreground" Value="#ccc"/>
<Setter Property="TextDecorations" Value="Strikethrough"/>
</DataTrigger>
</Style.Triggers>
</Style>
</Grid.Resources>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
<ColumnDefinition Width="100"/>
</Grid.ColumnDefinitions>
<StackPanel>
<TextBlock x:Name="PriceTask" Style="{StaticResource oldPrice}"/>
<TextBlock x:Name="DiscountTask" Style="{StaticResource disc}"/>
<TextBlock x:Name="NewPriceTask" Style="{StaticResource disc}"/>
<TextBlock x:Name="SupportPeriodTask"/>
</StackPanel>
<StackPanel Grid.Column="2" x:Name="AttachmentBlock" VerticalAlignment="Center">
<TextBlock Text="Вложение" TextAlignment="Center"/>
<Button Content="Скачать" Margin="5 2" Click="DownloadAttachmentBtn_Click"/>
</StackPanel>
</Grid>
<TextBlock x:Name="ApproxCompleteTime" Grid.Row="3"/>
</Grid>
</Border>
<StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Right">
<Button Content="Заказать" Margin="5 10" Click="MakeOrderBtn_Click"/>
</StackPanel>
</Grid>
</Page>
```

ViewDetailTaskPage.xaml.cs

```
using INCOMSYSTEM.Context;
using Microsoft.Win32;
using System;
using System.Globalization;
using System.IO;
using System.Windows;
using System.Windows.Controls;
using INCOMSYSTEM.BehaviorsFiles;
```

```
using INCOMSYSTEM.Windows;
namespace INCOMSYSTEM.Pages.MainPages.Views
{
public partial class ViewDetailTaskPage : Page
{
public ViewDetailTaskPage(Tasks task)
{
InitializeComponent();
discountStyle.IsChecked = task.discoutStyle;
discountVisible.IsChecked = task.discountVisible;
TitleTask.Text = task.name;
SpecTask.Text = $"{task.Specializations.name}";
DescriptionTask.Text = task.description;
PriceTask.Text = $"Цена: {task.price.ToString("#,#", CultureInfo.CurrentCulture)} руб.";
DiscountTask.Text = $"Скидка: {task.discount}%";
NewPriceTask.Text = $"Новая цена: {task.newPrice:0.00}";
SupportPeriodTask.Text = $"Период поддержки: {task.supportPeriod.ConvertDay()}";
ApproxCompleteTime.Text = $"Примерное время выполнения {task.approxCompleteTime.ConvertDay()}";
AttachmentBlock.Visibility = task.attachment == null ? Visibility.Collapsed : Visibility.Visible;
_task = task;
}
private readonly Tasks _task;
private void DownloadAttachmentBtn_Click(object sender, RoutedEventArgs e)
{
var saveFileDialog = new SaveFileDialog
{
Title = "Скачивание файла",
CreatePrompt = true,
FileName = $"{_task.name}.{_task.fileExtension}",
Filter = $"File | *.{_task.fileExtension}"
};
if (saveFileDialog.ShowDialog() != true) return;
using(var file = new FileStream(saveFileDialog.FileName, FileMode.OpenOrCreate, FileAccess.Write))
{
file.Write(_task.attachment, 0, _task.attachment.Length);
file.Dispose();
}
}
private void MakeOrderBtn_Click(object sender, RoutedEventArgs e)
{
if (MessageBox.Show("Вы уверены с заказом?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Information) != MessageBoxResult.Yes) return;
var order = new Orders
{
idCustomer = MainWindow.AuthUser.idUser,
idTask = _task.id,
difficulty = 1f,
dateOrder = DateTime.Now,
idStatus = 1
};
order.price = _task.price * (decimal)order.difficulty;
var chat = new Chats
{
idChat = order.id,
idCustomer = MainWindow.AuthUser.idUser,
dateCreate = DateTime.Now
};
using (var db = new INCOMSYSTEMEntities())
{
db.Orders.Add(order);
db.Chats.Add(chat);
db.SaveChanges();
}
MessageBox.Show("Заказ успешно сформирован!");
MainWindow.ReviewFrame.GoBack();
}
}
}
```

ViewDetailOrderPage.xaml

```
<Page x:Class="INCOMSYSTEM.Pages.MainPages.Views.ViewDetailOrderPage"
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:controls="clr-namespace:INCOMSYSTEM.Controls"
xmlns:sys="clr-namespace:System;assembly=mscorlib"
mc:Ignorable="d">
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<StackPanel HorizontalAlignment="Right">
<StackPanel Orientation="Horizontal" HorizontalAlignment="Right">
<TextBlock Text="Дата формирования заказа: " Style="{StaticResource TextBlockStyle}"/>
<TextBlock x:Name="DateOrderBlock" Style="{StaticResource TextBlockStyle}"/>
</StackPanel>
<StackPanel Orientation="Horizontal" HorizontalAlignment="Right">
<TextBlock Text="Статус заказа: " Style="{StaticResource TextBlockStyle}"/>
<TextBlock x:Name="StatusOrderBlock" Style="{StaticResource TextBlockStyle}"/>
</StackPanel>
</StackPanel>
<Grid Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition/>
</Grid.ColumnDefinitions>
<StackPanel>
<StackPanel.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource TextBlockStyle}">
<Setter Property="Height" Value="30"/>
<Setter Property="TextWrapping" Value="Wrap"/>
<Setter Property="Margin" Value="0 5 0
```

```
0"/>
</Style>
</StackPanel.Resources>
<TextBlock Text="Заказчик"/>
<TextBlock Text="Исполнитель" Margin="0 5 0 0"/>
<TextBlock Text="Менеджер"/>
<TextBlock Text="Цена"/>
<TextBlock Text="Сложность"/>
<TextBlock Text="Плановая дата начала"/>
<TextBlock Text="Плановая дата выполнения"/>
<TextBlock Text="Вложение" Height="85"/>
</StackPanel>
<StackPanel Grid.Column="1" Width="175" Margin="5 0">
<StackPanel.Resources>
<Style TargetType="controls:InputTextBox" BasedOn="{StaticResource {x:Type controls:InputTextBox}}">
<Setter Property="Height" Value="30"/>
<Setter Property="Margin" Value="0 5 0 0"/>
</Style>
<Style TargetType="ComboBox" BasedOn="{StaticResource {x:Type ComboBox}}">
<Setter Property="Height" Value="30"/>
<Setter Property="Margin" Value="0 5 0 0"/>
</Style>
<Style TargetType="controls:DatePickerEx" BasedOn="{StaticResource {x:Type controls:DatePickerEx}}">
<Setter Property="BorderBrush" Value="White"/>
<Style.Triggers>
<DataTrigger Binding="{Binding RelativeSource={RelativeSource Mode=Self}, Path=IsWrong}" Value="True">
<Setter Property="BorderBrush" Value="Red"/>
</DataTrigger>
</Style.Triggers>
</Style>
</StackPanel.Resources>
<controls:InputTextBox x:Name="CustomerBlock" IsReadOnly="True"/>
<ComboBox x:Name="ExecutorBox">
<ComboBox.ItemTemplate>
<DataTemplate>
<TextBlock Style="{StaticResource ComboBoxTextBlockStyle}">
<TextBlock.Text>
<MultiBinding StringFormat="{}{0} {1} {2}">
<Binding Path="Employees.surname"/>
<Binding Path="Employees.name"/>
<Binding Path="Employees.patronymic"/>
</MultiBinding>
</TextBlock.Text>
</TextBlock>
</DataTemplate>
</ComboBox.ItemTemplate>
</ComboBox>
<controls:InputTextBox x:Name="ExecutorBlock" Visibility="Collapsed" IsReadOnly="True"/>
<controls:InputTextBox x:Name="ManagerBlock" IsReadOnly="True"/>
<controls:InputTextBox x:Name="PriceBox" IsReadOnly="True"/>
<controls:InputTextBox x:Name="DifficultyBox" TextChanged="DifficultyBox_TextChanged"/>
<controls:DatePickerEx x:Name="PlanDateStartBox" DisplayDateStart="{x:Static sys:DateTime.Today}" SelectedDateChanged="DateBox_SelectedDateChanged"/>
<controls:DatePickerEx x:Name="PlanDateCompleteBox" DisplayDateStart="{x:Static sys:DateTime.Today}" SelectedDateChanged="DateBox_SelectedDateChanged"/>
<Grid Height="100" Drop="FileDownload_OnDrop" AllowDrop="True" x:Name="FileDownloadGrid">
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Button Style="{StaticResource LabelBtn}" x:Name="FileDownload" Click="FileDownloadBtn_Click" VerticalAlignment="Center" HorizontalContentAlignment="Center" IsEnabled="False"/>
<StackPanel Grid.Column="1">
<Button Content="Загрузить" Click="UploadBtn_Click" x:Name="UploadBtn"/>
<Button Content="Очистить" IsEnabled="False" Click="ClearBtn_Click" x:Name="ClearBtn"/>
<Button Content="Вернуть" IsEnabled="False" Click="ReturnBtn_Click" x:Name="ReturnBtn"/>
</StackPanel>
</Grid>
</StackPanel>
</Grid>
<StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Center" Margin="0 5">
<Button x:Name="SaveBtn" Content="Сохранить" Click="SaveBtn_Click"/>
<Button x:Name="CancelBtn" Content="Отменить" Margin="5 0" Click="CancelBtn_Click"/>
</StackPanel>
</Grid>
</Page>
```

ViewDetailOrderPage.xaml.cs

```
using System.Linq;
using System.Windows.Controls;
using INCOMSYSTEM.Context;
using System.Data.Entity;
using System.Globalization;
using System.Windows;
using Microsoft.Win32;
using System.IO;
using System;
using INCOMSYSTEM.Windows;
namespace INCOMSYSTEM.Pages.MainPages.Views
{
public partial class ViewDetailOrderPage : Page
{
public ViewDetailOrderPage(Orders order)
{
```

```
_order = order;
InitializeComponent();
Title = $"Подробности заказа №{order.id}";
_fileOrder = order.attachment;
_fileOrderExtension = order.fileExtension;
SetInputBoxValues(order);
SetFileValues(order);
if (order.idStatus >= 4 || order.factDateStart != null || MainWindow.AuthUser.idPos == 2)
{
ChangeExecutorBox();
DisableBoxes();
return;
}
using (var db = new INCOMSYSTEMEntities())
{
ExecutorBox.ItemsSource = db.Employees.SelectMany(s => s.SpecializationsEmployee)
.Include(s => s.Employees.UsersDetail)
.Where(s => s.Employees.UsersDetail.idPos == 2 && s.idSpecialization == order.Tasks.idSpecialization).ToList();
if (order.idExecutor != null)
ExecutorBox.SelectedItem = ExecutorBox.ItemsSource.Cast<SpecializationsEmployee>()
.First(s => s.Employees.idUser == order.idExecutor);
}
}
private void ChangeExecutorBox()
{
ExecutorBox.Visibility = Visibility.Collapsed;
ExecutorBlock.Visibility = Visibility.Visible;
}
private void DisableBoxes()
{
ExecutorBox.Visibility = Visibility.Collapsed;
ExecutorBlock.Visibility = Visibility.Visible;
DifficultyBox.IsReadOnly = true;
FileDownloadGrid.AllowDrop = false;
UploadBtn.IsEnabled = false;
ClearBtn.IsEnabled = false;
SaveBtn.Visibility = Visibility.Collapsed;
CancelBtn.Content = "Закрыть";
}
private void SetInputBoxValues(Orders order)
{
CustomerBlock.Text = $"{order.Customers.name}";
PriceBox.Text = $"{order.price.ToString("#,#", new CultureInfo("ru-RU"))} руб.";
DifficultyBox.Text = order.difficulty.ToString(CultureInfo.InvariantCulture);
if (order.Chats.Employees != null)
{
ManagerBlock.Text = $"{order.Chats.Employees.surname} {order.Chats.Employees.name}";
ManagerBlock.Text += order.Chats.Employees.patronymic != null ? $" {order.Chats.Employees.patronymic}" : "";
}
if (order.idStatus >= 3)
{
ExecutorBlock.Text = $"{order.Employees.surname} {order.Employees.name}";
ExecutorBlock.Text += order.Employees.patronymic != null ? $" {order.Employees.patronymic}" : "";
}
if(order.factDateStart != null)
{
PlanDateStartBox.IsHitTestVisible = false;
PlanDateCompleteBox.IsHitTestVisible = false;
}
PlanDateStartBox.SelectedDate = order.planDateStart;
PlanDateCompleteBox.SelectedDate = order.planDateComplete;
DateOrderBlock.Text = order.dateOrder.ToString("dd MMMM yyyy", CultureInfo.CurrentCulture);
StatusOrderBlock.Text = order.Statuses.name;
}
private void SetFileValues(Orders order)
{
if (_fileOrder == null)
{
FileDownload.IsEnabled = false;
ReturnBtn.Visibility = Visibility.Collapsed;
}
else
{
FileDownload.Content = $"Дополнение к договору.{order.fileExtension}";
FileDownload.IsEnabled = true;
TempFile = _fileOrder;
TempFileExtension = _fileOrderExtension;
ClearBtn.IsEnabled = true;
}
}
private readonly Orders _order;
private string _fileName = "Дополнение к договору";
private string FileName
{
get => _fileName;
set
{
FileDownload.IsEnabled = !string.IsNullOrWhiteSpace(value);
_fileName = value;
FileDownload.Content = value;
}
}
private readonly byte[] _fileOrder;
private readonly string _fileOrderExtension;
private byte[] TempFile { get; set; }
private string TempFileExtension { get; set; }
private void UploadBtn_Click(object sender, RoutedEventArgs e)
{
var openFile = new OpenFileDialog
{
Title = "Загрузка файла",
Filter = "Document | *.docx; *.doc | Portable Document | *.pdf",
DefaultExt = ".docx"
};
if (openFile.ShowDialog() != true) return;
TempFile = File.ReadAllBytes(openFile.FileName);
TempFileExtension = openFile.SafeFileName.Split('.').Last();
FileName = $"Дополнение к договору.{TempFileExtension}";
ClearBtn.IsEnabled = true;
ReturnBtn.IsEnabled = true;
}
private void ClearBtn_Click(object sender,
```

```
RoutedEventArgs e)
{
TempFile = null;
TempFileExtension = null;
FileName = string.Empty;
ClearBtn.IsEnabled = false;
ReturnBtn.IsEnabled = true;
}
private void ReturnBtn_Click(object sender, RoutedEventArgs e)
{
TempFile = _fileOrder;
TempFileExtension = _fileOrderExtension;
FileName = $"Дополнение к договору.{TempFileExtension}";
ReturnBtn.IsEnabled = false;
ClearBtn.IsEnabled = true;
}
private void FileDownloadBtn_Click(object sender, RoutedEventArgs e)
{
if (TempFile == null) return;
var saveFileDialog = new SaveFileDialog
{
Title = "Скачивание файла",
FileName = $"{FileName}",
Filter = $"File | * {TempFileExtension}",
DefaultExt = $".{TempFileExtension}"
};
if (saveFileDialog.ShowDialog() != true) return;
using (var file = new FileStream(saveFileDialog.FileName, FileMode.OpenOrCreate, FileAccess.Write))
{
file.Write(TempFile, 0, TempFile.Length);
}
}
private void FileDownload_OnDrop(object sender, DragEventArgs e)
{
if (!e.Data.GetDataPresent(DataFormats.FileDrop)) return;
var file = ((string[])e.Data.GetData(DataFormats.FileDrop))?[0];
if (file == null) return;
var fileExtension = file.Split('.').Last();
if (fileExtension != "docx" && fileExtension != "doc" && fileExtension != "pdf")
{
AdditionalWindow.ShowError("Не верный формат файла");
return;
}
TempFile = File.ReadAllBytes(file);
TempFileExtension = fileExtension;
ClearBtn.IsEnabled = true;
ReturnBtn.IsEnabled = true;
FileName = $"Дополнение к договору.{TempFileExtension}";
AdditionalWindow.HideError();
}
private void DifficultyBox_TextChanged(object sender, TextChangedEventArgs e)
{
if (!decimal.TryParse(DifficultyBox.Text.Replace(',', '.'), NumberStyles.AllowDecimalPoint, CultureInfo.InvariantCulture, out var difficulty))
{
AdditionalWindow.ShowError("Не удалось преобразовать коэффициент сложности в строку");
return;
}
var price = _order.Tasks.newPrice * difficulty;
PriceBox.Text = Math.Round(price).ToString("#,#", new CultureInfo("ru-RU")) + " руб.";
AdditionalWindow.HideError();
}
private void SaveBtn_Click(object sender, RoutedEventArgs e)
{
if (!decimal.TryParse(DifficultyBox.Text.Replace(',', '.'), NumberStyles.AllowDecimalPoint, CultureInfo.InvariantCulture, out var difficulty))
{
AdditionalWindow.ShowError("Не удалось преобразовать коэффициент сложности в строку");
return;
}
if (!IsEquals())
{
byte? status = null;
if (_order.factDateStart != null && _order.factDateComplete != null) status = 4;
else if ((_order.factDateStart != null && _order.factDateComplete == null) || (PlanDateStartBox.SelectedDate != null && PlanDateCompleteBox.SelectedDate != null)) status = 3;
using (var db = new INCOMSYSTEMEntities())
{
var order = db.Orders.First(s => s.id == _order.id);
order.idExecutor = ((SpecializationsEmployee)ExecutorBox.SelectedItem)?.idEmployee;
order.price = Math.Round(_order.Tasks.newPrice * difficulty);
order.difficulty = (double)difficulty;
order.planDateStart = PlanDateStartBox.SelectedDate;
order.planDateComplete = PlanDateCompleteBox.SelectedDate;
if (status != null) order.idStatus = status.Value;
order.attachment = TempFile;
order.fileExtension = TempFileExtension;
db.SaveChanges();
}
}
AdditionalWindow.HideError();
var addWindow = Application.Current.Windows.OfType<AdditionalWindow>().First();
addWindow.DialogResult = true;
addWindow.Close();
}
private void CancelBtn_Click(object sender, RoutedEventArgs e)
{
if (!IsEquals() && MessageBox.Show("Отменить внесённые изменения?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.No) return;
var addWindow = Application.Current.Windows.OfType<AdditionalWindow>().First();
addWindow.DialogResult = false;
addWindow.Close();
```

```
}
private bool IsEquals()
{
if (_order.idStatus >= 3) return true;
return
((SpecializationsEmployee)ExecutorBox.SelectedItem)?.idEmployee ==
_order.idExecutor
&& DifficultyBox.Text ==
_order.difficulty.ToString(CultureInfo.InvariantCulture)
&& PlanDateStartBox.SelectedDate ==
_order.planDateStart
&& PlanDateCompleteBox.SelectedDate ==
_order.planDateComplete
&& (_fileOrder == TempFile &&
_fileOrderExtension ==
TempFileExtension);
}
private void
DateBox_SelectedDateChanged(object
sender, SelectionChangedEventArgs e)
{
PlanDateStartBox.IsWrong =
PlanDateStartBox.SelectedDate != null &&
PlanDateCompleteBox.SelectedDate != null
&& PlanDateStartBox.SelectedDate.Value
>
PlanDateCompleteBox.SelectedDate.Value;
if(PlanDateStartBox.IsWrong)
AdditionalWindow.ShowError("Плановая
дата начала не может быть больше
плановой даты завершения");
else AdditionalWindow.HideError();
}
}
}
```

TaskDetailPage.xaml

```
<Page
x:Class="INCOMSYSTEM.Pages.Details.
TaskDetailPage"
xmlns="http://schemas.microsoft.com/winf
x/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/wi
nfx/2006/xaml"
xmlns:mc="http://schemas.openxmlformats
.org/markup-compatibility/2006"
xmlns:d="http://schemas.microsoft.com/ex
pression/blend/2008"
xmlns:controls="clr-
namespace:INCOMSYSTEM.Controls"
mc:Ignorable="d">
<Page.Resources>
<Style TargetType="TextBlock"
BasedOn="{StaticResource
TextBlockStyle}"/>
<Style TargetType="Label"
BasedOn="{StaticResource LabelStyle}"/>
</Page.Resources>
<Grid HorizontalAlignment="Center"
VerticalAlignment="Center">
<Grid.RowDefinitions>
<RowDefinition/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="150"/>
<ColumnDefinition Width="200"/>
<ColumnDefinition Width="30"/>
</Grid.ColumnDefinitions>
<StackPanel>
<StackPanel.Resources>
<Style TargetType="TextBlock"
BasedOn="{StaticResource
TextBlockStyle}">
<Setter Property="Height" Value="30"/>
<Setter Property="TextWrapping"
Value="Wrap"/>
</Style>
</StackPanel.Resources>
<TextBlock Text="Введите название"/>
<TextBlock Text="Выберите
специализацию" Height="45"/>
<TextBlock Text="Введите описание"/>
<TextBlock Text="Введите цену"/>
<TextBlock Text="Введите скидку"/>
<TextBlock Text="Введите преиод
поддержки (дни)" Height="40"/>
<TextBlock Text="Введите примерное
время выполнения (дни)" Height="70"/>
<TextBlock Text="Добавьте вложение"
Height="100"/>
</StackPanel>
<StackPanel Grid.Column="1" Margin="5
0">
<StackPanel.Resources>
<Style
TargetType="controls:InputTextBox"
BasedOn="{StaticResource {x:Type
controls:InputTextBox}}">
<Setter Property="Height" Value="30"/>
</Style>
<Style TargetType="ComboBox"
BasedOn="{StaticResource {x:Type
ComboBox}}">
<Setter Property="Height" Value="30"/>
<Setter Property="Margin" Value="0 5 0
0"/>
</Style>
</StackPanel.Resources>
<controls:InputTextBox
PlaceHolder="Название"
x:Name="NameBox"/>
<ComboBox x:Name="SpecBox"
Height="30" Margin="0 10 0 0"
SelectedIndex="0">
<ComboBox.ItemTemplate>
<DataTemplate>
<TextBlock Style="{StaticResource
ComboBoxTextBlockStyle}"
Text="{Binding name}"/>
</DataTemplate>
</ComboBox.ItemTemplate>
</ComboBox>
<controls:InputTextBox
PlaceHolder="Описание"
x:Name="DescriptionBox"/>
<controls:InputTextBox
PlaceHolder="Цена" x:Name="PriceBox"
TextChanged="PriceBox_OnTextChanged"
/>
<controls:InputTextBox
PlaceHolder="Скидка"
x:Name="DiscountBox"/>
<controls:InputTextBox
PlaceHolder="Период поддержки"
x:Name="SupportPeriod" Margin="0 10 0
0"/>
<controls:InputTextBox
PlaceHolder="Примерное время
выполнения" x:Name="ApproxTimeBox"
Margin="0 15 0 15"/>
<Grid Height="100"
Drop="FileDownload_OnDrop"
AllowDrop="True">
<Grid.ColumnDefinitions>
<ColumnDefinition/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Button Style="{StaticResource
LabelBtn}"
x:Name="FileDownload"
Click="FileDownloadBtn_Click"
VerticalAlignment="Center"
HorizontalContentAlignment="Center"
IsEnabled="False"/>
<StackPanel Grid.Column="1">
<Button Content="Загрузить"
```

```
Click="UploadBtn_Click"/>
<Button Content="Очистить" IsEnabled="False" Click="ClearBtn_Click" x:Name="ClearBtn"/>
<Button Content="Вернуть" IsEnabled="False" Click="ReturnBtn_Click" x:Name="ReturnBtn"/>
</StackPanel>
</Grid>
</StackPanel>
<StackPanel Grid.Column="2" Margin="50">
<StackPanel.Resources>
<Style TargetType="TextBlock" BasedOn="{StaticResource {x:Type TextBlock}}">
<Setter Property="Height" Value="30"/>
<Setter Property="TextWrapping" Value="Wrap"/>
</Style>
</StackPanel.Resources>
<TextBlock Text="*"/>
<TextBlock Text="*" Margin="0 15 0 5"/>
<TextBlock Text="*"/>
<TextBlock Text="*"/>
<TextBlock Text=""/>
<TextBlock Text="*" Margin="0 10 0 5"/>
<TextBlock Text="*" Margin="0 15 0 0"/>
<TextBlock Text=""/>
</StackPanel>
<StackPanel Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="3" Orientation="Horizontal" HorizontalAlignment="Center" Margin="0 5">
<Button Content="Сохранить" Click="SaveBtn_Click" x:Name="SaveBtn"/>
<Button Content="Отменить" Margin="50" Click="CancelBtn_Click"/>
</StackPanel>
</Grid>
</Page>
```

TaskDetailPage.xaml.cs

```
using System.Globalization;
using System.IO;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using INCOMSYSTEM.Context;
using INCOMSYSTEM.Windows;
using Microsoft.Win32;
namespace INCOMSYSTEM.Pages.Details
{
public partial class TaskDetailPage : Page
{
public TaskDetailPage(Tasks task)
{
InitializeComponent();
Title = "Редактирование задачи";
_isEdit = true;
SaveBtn.Content = "Сохранить";
_task = task;
_fileTask = task.attachment;
_fileTaskExtension = task.fileExtension;
SetFileValues(task);
_oldName = task.name;
SetInputBoxValues(task);
using (var db = new INCOMSYSTEMEntities())
{
SpecBox.ItemsSource = db.Specializations.ToList();
SpecBox.SelectedItem = SpecBox.ItemsSource.Cast<Specializations>()
.First(s => s.id == task.idSpecialization);
}
}
private void SetInputBoxValues(Tasks task)
{
NameBox.Text = task.name;
DescriptionBox.Text = task.description;
PriceBox.Text = $"{task.price.ToString("#,#", CultureInfo.CurrentCulture)}";
if (task.discount != null) DiscountBox.Text = $"{task.discount}";
ApproxTimeBox.Text = $"{task.approxCompleteTime}";
SupportPeriod.Text = task.supportPeriod.ToString();
}
private void SetFileValues(Tasks task)
{
if (_fileTask == null)
{
FileDownload.IsEnabled = false;
ReturnBtn.Visibility = Visibility.Collapsed;
}
else
{
FileDownload.Content = $"{task.name}.{task.fileExtension}";
FileDownload.IsEnabled = true;
TempFile = _fileTask;
TempFileExtension = _fileTaskExtension;
ClearBtn.IsEnabled = true;
}
}
public TaskDetailPage()
{
InitializeComponent();
Title = "Добавление задачи";
_isEdit = false;
SaveBtn.Content = "Добавить";
FileDownload.IsEnabled = false;
ReturnBtn.Visibility = Visibility.Collapsed;
using (var db = new INCOMSYSTEMEntities())
{
SpecBox.ItemsSource = db.Specializations.ToList();
}
}
private readonly Tasks _task;
private string _fileName;
private string FileName
{
get => _fileName;
set
{
FileDownload.IsEnabled = !string.IsNullOrWhiteSpace(value);
_fileName = value;
FileDownload.Content = value;
}
}
private readonly bool _isEdit;
private readonly string _oldName;
private readonly byte[] _fileTask;
private readonly string _fileTaskExtension;
private byte[] TempFile { get; set; }
private string TempFileExtension { get; set; }
private void UploadBtn_Click(object sender, RoutedEventArgs e)
{
var openFile = new OpenFileDialog
{
Title = "Загрузка файла",
Filter = "Document | *.docx; *.doc | Portable Document | *.pdf",
DefaultExt = ".docx"
};
if (openFile.ShowDialog() != true) return;
```

```
TempFile = File.ReadAllBytes(openFile.FileName);
TempFileExtension = openFile.SafeFileName.Split('.').Last();
FileName = openFile.SafeFileName;
ClearBtn.IsEnabled = true;
ReturnBtn.IsEnabled = true;
}
private void ClearBtn_Click(object sender, RoutedEventArgs e)
{
TempFile = null;
TempFileExtension = null;
FileName = string.Empty;
ClearBtn.IsEnabled = false;
ReturnBtn.IsEnabled = true;
}
private void ReturnBtn_Click(object sender, RoutedEventArgs e)
{
TempFile = _fileTask;
TempFileExtension = _fileTaskExtension;
FileName = $"{_oldName}.{_fileTaskExtension}";
ReturnBtn.IsEnabled = false;
ClearBtn.IsEnabled = true;
}
private void FileDownloadBtn_Click(object sender, RoutedEventArgs e)
{
if (TempFile == null) return;
var saveFileDialog = new SaveFileDialog
{
Title = "Скачивание файла",
FileName = $"{NameBox.Value}",
Filter = $"File | * {TempFileExtension}",
DefaultExt = $".{TempFileExtension}"
};
if (saveFileDialog.ShowDialog() != true) return;
using (var file = new FileStream(saveFileDialog.FileName, FileMode.OpenOrCreate, FileAccess.Write))
{
file.Write(TempFile, 0, TempFile.Length);
}
}
private void FileDownload_OnDrop(object sender, DragEventArgs e)
{
if (!e.Data.GetDataPresent(DataFormats.FileDrop)) return;
var file = ((string[])e.Data.GetData(DataFormats.FileDrop))?[0];
if (file == null) return;
var fileExtension = file.Split('.').Last();
if (fileExtension != "docx" && fileExtension != "doc" && fileExtension != "pdf")
{
AdditionalWindow.ShowError("Не верный формат файла");
return;
}
TempFile = File.ReadAllBytes(file);
TempFileExtension = fileExtension;
ClearBtn.IsEnabled = true;
ReturnBtn.IsEnabled = true;
FileName = file.Split('\\').Last();
AdditionalWindow.HideError();
}
private void SaveBtn_Click(object sender, RoutedEventArgs e)
{
switch (_isEdit)
{
case true when !SaveTask():
case false when !AddTask():
return;
}
AdditionalWindow.HideError();
var addWindow = Application.Current.Windows.OfType<AdditionalWindow>().First();
addWindow.DialogResult = true;
addWindow.Close();
}
private bool SaveTask()
{
if (IsNullOrWhiteSpace())
{
AdditionalWindow.ShowError("Поля отмеченные звёздочкой не могут быть пустыми");
return false;
}
if (!GetValues()) return false;
if (IsEqualsOrNull()) return true;
using (var db = new INCOMSYSTEMEntities())
{
var task = db.Tasks.First(s => s.id == _task.id);
task.name = _name;
task.description = _desc;
task.price = _price;
if (_disc > 0) task.discount = _disc;
else task.discount = null;
task.approxCompleteTime = _approx;
task.idSpecialization = _spec.id;
task.supportPeriod = _support;
task.attachment = TempFile;
task.fileExtension = TempFileExtension;
db.SaveChanges();
}
return true;
}
private bool AddTask()
{
if (IsNullOrWhiteSpace())
{
AdditionalWindow.ShowError("Поля не могут быть пустыми");
return false;
}
if (!GetValues()) return false;
using (var db = new INCOMSYSTEMEntities())
{
var task = new Tasks
{
name = _name,
description = _desc,
price = _price,
approxCompleteTime = _approx,
idSpecialization = _spec.id,
supportPeriod = _support,
attachment = TempFile,
fileExtension = TempFileExtension
};
if (_disc > 0) task.discount = _disc;
else task.discount = null;
db.Tasks.Add(task);
db.SaveChanges();
}
return true;
}
private string _name;
private Specializations _spec;
private string _desc;
private decimal _price;
public int _support;
private byte _disc;
private int _approx;
private bool GetValues()
{
```

```
if (IsNullOrWhiteSpace())
{
AdditionalWindow.ShowError("Поля не могут быть пустыми");
return false;
}
_name = NameBox.Value;
_spec = SpecBox.SelectedItem as Specializations;
_desc = DescriptionBox.Value;
if (!decimal.TryParse(PriceBox.Value, out _price))
{
AdditionalWindow.ShowError("Не удалось преобразовать строку в число");
return false;
}
if(!int.TryParse(SupportPeriod.Value, out _support))
{
AdditionalWindow.ShowError("Не удалось преобразовать период поддержки в число");
return false;
}
if (!DiscountBox.IsWhiteSpace && !byte.TryParse(DiscountBox.Value, out _disc))
{
AdditionalWindow.ShowError("Не удалось преобразовать скидку в число");
return false;
}
if (int.TryParse(ApproxTimeBox.Value, out _approx)) return true;
return false;
}
private bool IsNullOrWhiteSpace()
{
return NameBox.IsWhiteSpace || DescriptionBox.IsWhiteSpace || PriceBox.IsWhiteSpace || ApproxTimeBox.IsWhiteSpace || SupportPeriod.IsWhiteSpace;
}
private void CancelBtn_Click(object sender, RoutedEventArgs e)
{
if (!IsEqualsOrNull() && MessageBox.Show("Отменить внесённые изменения?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.No) return;
var addWindow = Application.Current.Windows.OfType<AdditionalWindow>().First();
addWindow.DialogResult = false;
addWindow.Close();
}
private bool IsEqualsOrNull()
{
if (_isEdit) return NameBox.Value == _task.name
&& ((Specializations)SpecBox.SelectedItem).id == _task.idSpecialization
&& DescriptionBox.Value == _task.description
&& PriceBox.Value == _task.price.ToString("#,#", CultureInfo.CurrentCulture)
&& SupportPeriod.Value == _task.supportPeriod.ToString()
&& DiscountBox.Value == _task.discount.ToString()
&& ApproxTimeBox.Value == _task.approxCompleteTime.ToString()
&& (_fileTask == TempFile && _fileTaskExtension == TempFileExtension);
return NameBox.IsWhiteSpace
&& DescriptionBox.IsWhiteSpace
&& PriceBox.IsWhiteSpace
&& DiscountBox.IsWhiteSpace
&& SupportPeriod.IsWhiteSpace
&& ApproxTimeBox.IsWhiteSpace
&& string.IsNullOrWhiteSpace(FileName);
}
private void PriceBox_OnTextChanged(object sender, TextChangedEventArgs e)
{
if (PriceBox.IsWhiteSpace) return;
if (!decimal.TryParse(PriceBox.Value, out _price))
{
AdditionalWindow.ShowError("Не удалось преобразовать строку в число");
return;
}
AdditionalWindow.HideError();
var selStart = PriceBox.SelectionStart;
var selLength = PriceBox.SelectionLength;
var length = PriceBox.Text.Length;
PriceBox.Text = _price.ToString("#,#", CultureInfo.CurrentCulture);
if (length == PriceBox.Text.Length - 1) selStart++;
PriceBox.SelectionStart = selStart;
PriceBox.SelectionLength = selLength;
}
}
}
```

Приложение G.

ActualWidthConverter.cs

```
using System;
using System.Globalization;
using System.Windows;
using System.Windows.Data;
namespace INCOMSYSTEM.BehaviorsFiles
{
public class ActualWidthConverter : IValueConverter
{
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
{
return (double)value - 125d;
}
public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
{
return DependencyProperty.UnsetValue;
}
}
}
```

AnyItemsToVisibilityConverter.cs

```
using System;
using System.Collections;
using System.Globalization;
using System.Linq;
using System.Windows;
using System.Windows.Data;
namespace INCOMSYSTEM.BehaviorsFiles
{
public class AnyItemsToVisibilityConverter : IValueConverter
{
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
{
if (!(value is IEnumerable collection)) return Visibility.Collapsed;
return collection.OfType<object>().Any() ? Visibility.Collapsed : Visibility.Visible;
}
public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
{
return DependencyProperty.UnsetValue;
}
}
}
```

Extension.cs

```
using Word = Microsoft.Office.Interop.Word;
namespace INCOMSYSTEM.BehaviorsFiles
{
public static class Extension
{
public static bool ReplaceWordText(this Word.Document document, string findText, object replaceText, bool replaceAll = true)
{
var range = document.Content;
range.Find.ClearFormatting();
return range.Find.Execute(FindText: findText, ReplaceWith: replaceText, Forward: true, Replace: replaceAll ? Word.WdReplace.wdReplaceAll : Word.WdReplace.wdReplaceOne);
}
public static string ConvertDay(this int value)
{
var valStr = value.ToString();
string str;
var newVal = value;
if (valStr.Length > 1) newVal = int.Parse(valStr[valStr.Length - 1].ToString());
if (value == 14) str = $"{value} дней";
else if (newVal == 1) str = $"{value} день";
else if (newVal > 0 && newVal < 5) str = $"{value} дня";
else str = $"{value} дней";
return str;
}
}
}
```

NumberInWords.cs

```
using System;
using System.Text;
namespace INCOMSYSTEM.BehaviorsFiles
{
public static class NumberInWords
{
//Наименования сотен
private static readonly string[] Hunds =
{
"", "сто ", "двести ", "триста ", "четыреста ",
"пятьсот ", "шестьсот ", "семьсот ", "восемьсот ", "девятьсот "
};
//Наименования десятков
private static readonly string[] Tens =
{
"", "десять ", "двадцать ", "тридцать ", "сорок ", "пятьдесят ", "шестьдесят ", "семьдесят ", "восемьдесят ", "девяносто "
};
/// <summary>
/// Перевод в строку числа с учётом падежного окончания относящегося к числу существительного
/// </summary>
/// <param name="val">Число</param>
/// <param name="male">Род существительного, которое относится к числу</param>
/// <param name="one">Форма существительного в единственном числе</param>
/// <param name="two">Форма существительного от двух до четырёх</param>
/// <param name="five">Форма существительного от пяти и больше</param>
/// <returns></returns>
private static string Str(int val, bool male, string one, string two, string five)
{
string[] frac20 =
{
"", "один ", "два ", "три ", "четыре ", "пять ", "шесть ", "семь ", "восемь ", "девять ", "десять ", "одиннадцать ", "двенадцать ", "тринадцать ", "четырнадцать ", "пятнадцать ", "шестнадцать ", "семнадцать ", "восемнадцать ", "девятнадцать "
};
var num = val % 1000;
if(0 == num) return "";
if(num < 0) throw new ArgumentOutOfRangeException("val",
```

```
"Параметр не может быть отрицательным");
if(!male)
{
frac20[1] = "одна ";
frac20[2] = "две ";
}
var r = new StringBuilder(Hunds[num / 100]);

if(num % 100 < 20)
{
r.Append(frac20[num % 100]);
}
else
{
r.Append(Tens[num % 100 / 10]);
r.Append(frac20[num % 10]);
}
r.Append(Case(num, one, two, five));
if(r.Length != 0) r.Append(" ");
return r.ToString();
}
/// <summary>
/// Выбор правильного падежного окончания сущесвительного
/// </summary>
/// <param name="val">Число</param>
/// <param name="one">Форма существительного в единственном числе</param>
/// <param name="two">Форма существительного от двух до четырёх</param>
/// <param name="five">Форма существительного от пяти и больше</param>
/// <returns>Возвращает существительное с падежным окончанием, которое соответсвует числу</returns>
private static string Case(int val, string one, string two, string five)
{
var t=(val % 100 > 20) ? val % 10 : val % 20;
switch (t)
{
case 1: return one;
case 2: case 3: case 4: return two;
default: return five;
}
}
/// <summary>
/// Перевод целого числа в строку
/// </summary>
/// <param name="val">Число</param>
/// <returns>Возвращает строковую запись числа</returns>
public static string GetNumberInWords(this int val) {
var minus = false;
if (val < 0) { val = -val; minus = true; }
var n = (int)val;
var r = new StringBuilder();
if (0 == n) r.Append("0 ");
if (n % 1000 != 0)
r.Append(Str(n, true, "", "", ""));
n /= 1000;
r.Insert(0, Str(n, false, "тысяча", "тысячи", "тысяч"));
n /= 1000;
r.Insert(0, Str(n, true, "миллион", "миллиона", "миллионов"));
n /= 1000;
r.Insert(0, Str(n, true, "миллиард", "миллиарда", "миллиардов"));
n /= 1000;
r.Insert(0, Str(n, true, "триллион", "триллиона", "триллионов"));
n /= 1000;
r.Insert(0, Str(n, true, "триллиард", "триллиарда", "триллиардов"));
if (minus) r.Insert(0, "минус ");
//Делаем первую букву заглавной
r[0] = char.ToUpper(r[0]);
return r.ToString();
}
}
}
```

ObservableObject.cs

```
using System.ComponentModel;
using System.Runtime.CompilerServices;
namespace INCOMSYSTEM.BehaviorsFiles
{
public class ObservableObject : INotifyPropertyChanged
{
public event PropertyChangedEventHandler PropertyChanged;
protected void OnRaiseChanged([CallerMemberName]
string prop = "")
{
PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(prop));
}
}
}
```

PasswordGenerator.cs

```
using System;
namespace INCOMSYSTEM.BehaviorsFiles
{
public static class PasswordGenerator
{
public static string GetNewPassword()
{
const string text = "abcdefghijklmnopqrstuvwxyz" + "ABCDEFGHIJKLMNOPQRSTUVWXYZ" + "0123456789" + "!@#$%&";
var random = new Random();
var newPassword = "";
while(string.IsNullOrWhiteSpace(newPassword)
||
PasswordDifficulty.CheckDifficultyPassword(newPassword) == DifficultyPassword.Easy)
for (var i = 0; i < random.Next(5, 15); i++)
newPassword += text[random.Next(0, text.Length)];
return newPassword;
}
}
}
```

PasswordDifficulty.cs

```
namespace INCOMSYSTEM.BehaviorsFiles
{
public static class PasswordDifficulty
{
public static DifficultyPassword CheckDifficultyPassword(string password)
{
return DifficultyPassword.Hard;
}
}
public enum DifficultyPassword
{
Easy = 0,
Medium = 1,
```

```
Hard = 2
}
}
```

BorderEx.cs

```
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Input;
using System.Windows.Threading;
namespace INCOMSYSTEM.Controls
{
public class BorderEx : Border
{
public BorderEx()
{
MouseEnter += OnMouseEnter;
MouseLeave += OnMouseLeave;
timer.Interval = TimeSpan.FromSeconds(0.5d);
timer.Tick += TimerOnTick;
}
private void TimerOnTick(object sender, EventArgs e)
{
IsMouseEnter = _isMouseOver;
timer.Stop();
}
private DispatcherTimer timer = new DispatcherTimer();
private bool _isMouseOver = false;
public bool IsMouseEnter
{
get => (bool)GetValue(IsMouseEnterProperty);
set => SetValue(IsMouseEnterProperty, value);
}
public static readonly DependencyProperty IsMouseEnterProperty = DependencyProperty.Register(nameof(IsMouseEnter), typeof(bool), typeof(BorderEx), new FrameworkPropertyMetadata(default(bool), FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
null, null, false, UpdateSourceTrigger.PropertyChanged));
private void OnMouseEnter(object sender, MouseEventArgs e)
{
_isMouseOver = true;
timer.Start();
}
private void OnMouseLeave(object sender, MouseEventArgs e)
{
IsMouseEnter = false;
timer.Stop();
}
}
}
```

DatePicker.cs

```
using System.Windows;
using System.Windows.Controls;
using System.Windows.Controls.Primitives;
using System.Windows.Data;
namespace INCOMSYSTEM.Controls
{
public class DatePickerEx : DatePicker
{
static DatePickerEx()
{
DefaultStyleKeyProperty.OverrideMetadata(typeof(DatePickerEx), new FrameworkPropertyMetadata(typeof(DatePickerEx)));
}
public override void OnApplyTemplate()
{
base.OnApplyTemplate();
var popup = GetTemplateChild("PART_Popup") as Popup;
if (popup != null) ApplyCustomTemplate(popup);
}
public bool IsWrong
{
get => (bool)GetValue(IsWrongProperty);
set => SetValue(IsWrongProperty, value);
}
public static readonly DependencyProperty IsWrongProperty = DependencyProperty.Register(nameof(IsWrong), typeof(bool), typeof(DatePickerEx), new FrameworkPropertyMetadata(default(bool), FrameworkPropertyMetadataOptions.BindsTwoWayByDefault, null, null, false, UpdateSourceTrigger.PropertyChanged));
void ApplyCustomTemplate(Popup popup)
{
var calendar = popup.Child as Calendar;
if (calendar == null) return;
calendar.SetResourceReference(Calendar.StyleProperty,
"DatePickerEx_CustomPopup");
}
}
}
```

InputTextBox.cs

```
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Input;
namespace INCOMSYSTEM.Controls
{
class InputTextBox : TextBox
{
public bool IsNull
{
get => (bool)GetValue(IsNullProperty);
set => SetValue(IsNullProperty, value);
}
public static readonly DependencyProperty IsNullProperty = DependencyProperty.Register(nameof(IsNull), typeof(bool), typeof(InputTextBox), new PropertyMetadata(true));
public bool IsWhiteSpace
{
get => (bool)GetValue(IsWhiteSpaceProperty);
set => SetValue(IsWhiteSpaceProperty, value);
}
public static readonly DependencyProperty IsWhiteSpaceProperty = DependencyProperty.Register(nameof(IsWhiteSpace), typeof(bool), typeof(InputTextBox),
new FrameworkPropertyMetadata(true, FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
null, null, false, UpdateSourceTrigger.PropertyChanged));
public string Value
{
get => (string)GetValue(ValueProperty);
set => SetValue(ValueProperty, value);
}
public static readonly DependencyProperty ValueProperty =
```

```
DependencyProperty.Register(nameof(Value), typeof(string), typeof(InputTextBox), new PropertyMetadata(""));
public bool IsPlaceHolder
{
get => (bool)GetValue(IsPlaceHolderProperty);
set => SetValue(IsPlaceHolderProperty, value);
}
public static readonly DependencyProperty IsPlaceHolderProperty = DependencyProperty.Register(nameof(IsPlaceHolder), typeof(bool), typeof(InputTextBox),
new FrameworkPropertyMetadata(true, FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
null, null, false, UpdateSourceTrigger.PropertyChanged));
public bool IsPassword
{
get => (bool)GetValue(IsPasswordProperty);
set => SetValue(IsPasswordProperty, value);
}
public static readonly DependencyProperty IsPasswordProperty = DependencyProperty.Register(nameof(IsPassword), typeof(bool), typeof(InputTextBox),
new FrameworkPropertyMetadata(false, FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
null, null, false, UpdateSourceTrigger.PropertyChanged));
public bool IsShowed
{
get => (bool)GetValue(IsShowedProperty);
set => SetValue(IsShowedProperty, value);
}
public static readonly DependencyProperty IsShowedProperty = DependencyProperty.Register(nameof(IsShowed), typeof(bool), typeof(InputTextBox),
new FrameworkPropertyMetadata(false, FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
IsShowedPropertyChanged, null, false, UpdateSourceTrigger.PropertyChanged));
private static void IsShowedPropertyChanged(DependencyObject d, DependencyPropertyChangedEventArgs e)
{
if (!(d is InputTextBox inputTextBox)) return;
if (inputTextBox.IsPlaceHolder) return;
inputTextBox._isChanged = false;
if ((bool)e.NewValue)
{
var start = inputTextBox.SelectionStart;
inputTextBox.Text = inputTextBox.Value;
inputTextBox.SelectionStart = start;
}
else
{
if (string.IsNullOrWhiteSpace(inputTextBox.Value)) return;
var start = inputTextBox.SelectionStart;
var text = inputTextBox.Text;
foreach (var c in text)
{
text = text.Replace(c, '●');
}
inputTextBox.Text = text;
inputTextBox.SelectionStart = start;
}
inputTextBox._isChanged = true;
}
public string PlaceHolder
{
get => (string)GetValue(PlaceHolderProperty);
set => SetValue(PlaceHolderProperty, value);
}
public static readonly DependencyProperty PlaceHolderProperty = DependencyProperty.Register(nameof(PlaceHolder), typeof(string), typeof(InputTextBox),
new FrameworkPropertyMetadata("", FrameworkPropertyMetadataOptions.BindsTwoWayByDefault,
PlaceHolderPropertyChanged, null, false, UpdateSourceTrigger.PropertyChanged));
private static void PlaceHolderPropertyChanged(DependencyObject d, DependencyPropertyChangedEventArgs e)
{
if (!(d is InputTextBox inputTextBox)) return;
inputTextBox.IsPlaceHolder = !string.IsNullOrWhiteSpace((string)e.NewValue);
inputTextBox.Text = (string)e.NewValue;
}
public InputTextBox()
{
GotFocus += GotFocusText;
LostFocus += LostFocusText;
TextChanged += OnTextChanged;
PreviewKeyDown += OnPreviewKeyDown;
}
private void OnPreviewKeyDown(object sender, KeyEventArgs e)
{
switch (e.Key)
{
case Key.Delete:
case Key.Back:
if (SelectionLength > 0)
{
_isRemove = true;
RemoveFromSecureString(SelectionStart, SelectionLength);
}
else switch (e.Key)
{
case Key.Delete when SelectionStart < Text.Length:
_isRemove = true;
RemoveFromSecureString(SelectionStart, 1);
break;
case Key.Back when SelectionStart > 0:
_isRemove = true;
var caretIndex = SelectionStart;
if (SelectionStart > 0 && SelectionStart < Text.Length)
caretIndex -= 1;
RemoveFromSecureString(SelectionStart - 1, 1);
SelectionStart = caretIndex;
break;
default:
_isRemove = false;
break;
}
e.Handled = true;
break;
default:
_isChanged = true;
_isRemove = false;
```

```
break;
}
}
private bool _isRemove = false;
private bool _isChanged = false;
private readonly object _lock = new object();
private void OnTextChanged(object sender,
TextChangedEventArgs e)
{
if (Text != PlaceHolder || !IsPlaceHolder)
{
IsPlaceHolder = false;
IsNull = false;
}
else
{
IsPlaceHolder = true;
IsNull = true;
}
lock (Value)
{
IsWhiteSpace = IsPlaceHolder ||
string.IsNullOrWhiteSpace(Text);
if (IsPlaceHolder) return;
if (IsShowed)
{
Value = Text;
return;
}
if (IsWhiteSpace)
{
Value = string.Empty;
return;
}
if (!IsPassword) Value = Text;
else
{
if (_isRemove || !_isChanged) return;
_isChanged = false;
var start = SelectionStart;
var r = Text[start - 1];
Value = !Text.Contains("●") ? Text :
Value.Insert(start - 1, r.ToString());
if (!Text.Contains("●"))
{
var str = Text.Where(s => s != '●');
foreach (var c in str)
{
Text = Text.Replace(c, '●');
}
}
else Text = Text.Replace(r, '●');
SelectionStart = start;
}
}
}
private void RemoveFromSecureString(int
startIndex, int trimLength)
{
if (string.IsNullOrWhiteSpace(Value))
return;
_isChanged = false;
_isRemove = true;
var caretIndex = SelectionStart;
Value = Value.Remove(startIndex,
trimLength);
Text = Text.Remove(startIndex,
trimLength);
SelectionStart = caretIndex;
}
private void LostFocusText(object sender,
RoutedEventArgs e)
{
if (string.IsNullOrWhiteSpace(Text))
{
_isChanged = false;
Value = string.Empty;
IsPlaceHolder = true;
Text = PlaceHolder;
IsNull = true;
}
else IsNull = false;
}
private void GotFocusText(object sender,
RoutedEventArgs e)
{
if (IsPlaceHolder)
{
_isChanged = false;
IsPlaceHolder = false;
Text = string.Empty;
}
IsNull =
string.IsNullOrWhiteSpace(base.Text);
}
public new void Clear()
{
base.Text = string.Empty;
LostFocusText(new object(), new
RoutedEventArgs());
}
}
}
```

Приложение Н.

ButtonStyle.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
<Style TargetType="{x:Type Button}" BasedOn="{StaticResource MainStyle}">
<Setter Property="Background" Value="#4C37B1D4"/>
<Setter Property="Height" Value="30" />
<Setter Property="FontSize" Value="18" />
<Setter Property="Margin" Value="3 0" />
<Setter Property="Padding" Value="5 2" />
<Setter Property="Foreground" Value="Black" />
<Setter Property="Cursor" Value="Hand" />
<Setter Property="HorizontalContentAlignment" Value="Center" />
<Setter Property="VerticalContentAlignment" Value="Center" />
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type Button}">
<Border Background="{TemplateBinding Background}"
CornerRadius="5"
VerticalAlignment="{TemplateBinding VerticalAlignment}"
HorizontalAlignment="{TemplateBinding HorizontalAlignment}">
<Label Content="{TemplateBinding Content}"
Foreground="White"
Style="{StaticResource LabelStyle}"
FontFamily="{TemplateBinding FontFamily}"
VerticalAlignment="{TemplateBinding VerticalContentAlignment}"
HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"/>
</Border>
</ControlTemplate>
</Setter.Value>
</Setter>
<Style.Triggers>
<Trigger Property="IsMouseOver" Value="True">
<Setter Property="Background" Value="#4C29FEFF" />
</Trigger>
<Trigger Property="IsPressed" Value="True">
<Setter Property="Background" Value="#4C03090B" />
</Trigger>
<Trigger Property="IsEnabled" Value="False">
<Setter Property="Background" Value="#4D4D4D"/>
</Trigger>
</Style.Triggers>
</Style>
<Style TargetType="Button" x:Key="LabelBtn" BasedOn="{StaticResource MainStyle}">
<Setter Property="Cursor" Value="Hand" />
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="Button">
<Border BorderBrush="White" BorderThickness="0 0 0 2" VerticalAlignment="{TemplateBinding VerticalAlignment}"
HorizontalAlignment="{TemplateBinding HorizontalAlignment}">
<TextBlock Text="{TemplateBinding Content}"
Foreground="White"
TextWrapping="Wrap"
TextAlignment="{TemplateBinding HorizontalContentAlignment}"/>
</Border>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</ResourceDictionary>
```

ComboBoxStyle.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
<ControlTemplate x:Key="ComboBoxToggleButton" TargetType="{x:Type ToggleButton}">
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition />
<ColumnDefinition Width="20" />
</Grid.ColumnDefinitions>
<Border
x:Name="Border"
Grid.ColumnSpan="2"
CornerRadius="0"
Background="#FF3F3F3F"
BorderBrush="#FF97A0A5"
BorderThickness="1" />
<Path
x:Name="Arrow"
Grid.Column="1"
Fill="White"
HorizontalAlignment="Center"
VerticalAlignment="Center">
<Path.Style>
<Style TargetType="Path">
<Style.Triggers>
<DataTrigger Binding="{Binding RelativeSource={RelativeSource AncestorType=ToggleButton}, Path=IsChecked}" Value="True">
<Setter Property="Data" Value="M0,8 L0,6 L4,2 L8,6 L8,8 L4,4 z"/>
</DataTrigger>
</Style.Triggers>
<Style.Setters>
<Setter Property="Data" Value="M0,0 L0,2 L4,6 L8,2 L8,0 L4,4 z"/>
</Style.Setters>
</Style>
</Path.Style>
</Path>
</Grid>
</ControlTemplate>
<ControlTemplate
x:Key="ComboBoxTextBox"
TargetType="{x:Type TextBox}">
<Border x:Name="PART_ContentHost"
Focusable="False"
CornerRadius="5"
BorderBrush="#0AD2CF"
BorderThickness="1"
Background="{TemplateBinding Background}" />
</ControlTemplate>
<Style TargetType="{x:Type ComboBox}">
<Setter
Property="HorizontalContentAlignment"
Value="Left"/>
<Setter Property="SnapsToDevicePixels"
```

```
Value="True"/>
<Setter Property="MinHeight" Value="30"/>
<Setter Property="Foreground" Value="White"/>
<Setter Property="OverridesDefaultStyle" Value="False"/>
<Setter Property="Padding" Value="5 5 20 0"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="ComboBox">
<Grid>
<ToggleButton Name="ToggleButton" Template="{StaticResource ComboBoxToggleButton}" Grid.Column="2" Focusable="false" IsChecked="{Binding Path=IsDropDownOpen,Mode=TwoWay,RelativeSource={RelativeSource TemplatedParent}}" ClickMode="Press">
</ToggleButton>
<ContentPresenter x:Name="ContentSite" IsHitTestVisible="False" Content="{TemplateBinding SelectionBoxItem}" ContentTemplate="{TemplateBinding SelectionBoxItemTemplate}" ContentTemplateSelector="{TemplateBinding ItemTemplateSelector}" VerticalAlignment="{TemplateBinding VerticalContentAlignment}" HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}" Margin="{TemplateBinding Padding}"/>
<TextBox x:Name="PART_EditableTextBox" Style="{x:Null}" Template="{StaticResource ComboBoxTextBox}" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="3,3,23,3" Focusable="True" Background="#FF3F3F3F" Foreground="Green" Visibility="Hidden" IsReadOnly="{TemplateBinding IsReadOnly}"/>
<Popup x:Name="Popup" Placement="Bottom" HorizontalAlignment="Center" IsOpen="{TemplateBinding IsDropDownOpen}" AllowsTransparency="True" Focusable="False" PopupAnimation="Slide">
<Grid x:Name="DropDown" SnapsToDevicePixels="True" MinWidth="{TemplateBinding ActualWidth}" MaxHeight="{TemplateBinding MaxDropDownHeight}">
<Border x:Name="DropDownBorder" Background="#FF3F3F3F" BorderThickness="1" BorderBrush="#888888"/>
<ScrollViewer Margin="4,6,4,6" Width="{Binding RelativeSource={RelativeSource AncestorType=ComboBox}, Path=ActualWidth}" HorizontalScrollBarVisibility="Disabled" SnapsToDevicePixels="True">
<StackPanel IsItemsHost="True" KeyboardNavigation.DirectionalNavigation="Contained" />
</ScrollViewer>
</Grid>
</Popup>
</Grid>
<ControlTemplate.Triggers>
<Trigger Property="HasItems" Value="false">
<Setter TargetName="DropDownBorder" Property="MinHeight" Value="30"/>
</Trigger>
<Trigger Property="IsEnabled" Value="false">
<Setter Property="Foreground" Value="#888888"/>
</Trigger>
<Trigger Property="IsGrouping" Value="true">
<Setter Property="ScrollViewer.CanContentScroll" Value="false"/>
</Trigger>
<Trigger SourceName="Popup" Property="Popup.AllowsTransparency" Value="true">
<Setter TargetName="DropDownBorder" Property="CornerRadius" Value="0"/>
<Setter TargetName="DropDownBorder" Property="Margin" Value="0,2,0,0"/>
</Trigger>
<Trigger Property="IsEditable" Value="true">
<Setter Property="IsTabStop" Value="false"/>
<Setter TargetName="PART_EditableTextBox" Property="Visibility" Value="Visible"/>
<Setter TargetName="ContentSite" Property="Visibility" Value="Hidden"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style x:Key="ComboBoxTextBlockStyle" TargetType="TextBlock">
<Setter Property="FontSize" Value="15"/>
<Setter Property="ToolTip">
<Setter.Value>
<Binding RelativeSource="{RelativeSource Mode=Self}" Path="Text"/>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="ToolTip">
<Setter Property="Background" Value="Gray"/>
<Setter Property="Foreground" Value="White"/>
</Style>
</ResourceDictionary>
```

DatePickerExStyle.xaml

```
<ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:local="clr-namespace:INCOMSYSTEM.Controls">
<ControlTemplate x:Key="CalendarItemTemplate" TargetType="{x:Type CalendarItem}">
<ControlTemplate.Resources>
<DataTemplate x:Key="{x:Static
```

```
CalendarItem.DayTitleTemplateResourceKey}">
<TextBlock
Foreground="#FFFFFF"
FontFamily="Trebuchet MS"
FontWeight="Bold"
FontSize="9.5"
HorizontalAlignment="Center"
Margin="0,6,0,6"
Text="{Binding}"
VerticalAlignment="Center"/>
</DataTemplate>
</ControlTemplate.Resources>
<Grid x:Name="PART_Root">
<Grid.Resources>
<SolidColorBrush x:Key="DisabledColor"
Color="#A5FFFFFF"/>
</Grid.Resources>
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualState x:Name="Normal"/>
<VisualState x:Name="Disabled">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="PART_Disabled
Visual" To="1"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<Border Background="{TemplateBinding
Background}"
BorderBrush="{TemplateBinding
BorderBrush}"
BorderThickness="{TemplateBinding
BorderThickness}"
CornerRadius="15">
<Border BorderBrush="#FFFFFFFF"
BorderThickness="0" CornerRadius="1">
<Grid>
<Grid.Resources>
<ControlTemplate
x:Key="PreviousButtonTemplate"
TargetType="{x:Type Button}">
<Border CornerRadius="20"
Cursor="Hand"
Background="#FD413C">
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualState x:Name="Normal"/>
<VisualState x:Name="MouseOver">
<Storyboard>
<ColorAnimation Duration="0"
Storyboard.TargetName="path"
To="#FF73A9D8"
Storyboard.TargetProperty="(Shape.Fill).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Disabled">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="path" To=".5"
Storyboard.TargetProperty="(Shape.Fill).(Brush.Opacity)"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<Grid>
<Path x:Name="path"
Data="M19.03125 4.28125 L8.03125
15.28125 L7.34375 16 L8.03125
16.71875 L19.03125 27.71875 L20.46875
26.28125 L10.1875 16 L20.46875
5.71875Z"
Fill="#FFFFFF"
HorizontalAlignment="Center"
Height="8"
Stretch="Uniform"
VerticalAlignment="Center"
Width="6"/>
</Grid>
</Border>
</ControlTemplate>
<ControlTemplate
x:Key="NextButtonTemplate"
TargetType="{x:Type Button}">
<Border CornerRadius="20"
Cursor="Hand"
Background="#FD413C">
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualState x:Name="Normal"/>
<VisualState x:Name="MouseOver">
<Storyboard>
<ColorAnimation Duration="0"
Storyboard.TargetName="path"
To="#FF73A9D8"
Storyboard.TargetProperty="(Shape.Fill).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Disabled">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="path" To=".5"
Storyboard.TargetProperty="(Shape.Fill).(Brush.Opacity)"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<Grid>
<Path x:Name="path"
Data="M12.96875 4.28125 L11.53125
5.71875 L21.8125 16 L11.5125
26.28125 L12.96875 27.71875 L23.96875
16.71875 L24.65625 16 L23.96875
15.28125Z"
Fill="#FFFFFF"
HorizontalAlignment="Center"
Height="10"
Stretch="Uniform"
VerticalAlignment="Center"
Width="6"/>
</Grid>
</Border>
</ControlTemplate>
<ControlTemplate
x:Key="HeaderButtonTemplate"
TargetType="{x:Type Button}">
<Grid Cursor="Hand">
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualState x:Name="Normal"/>
<VisualState x:Name="MouseOver">
<Storyboard>
<ColorAnimation
Duration="0"
Storyboard.TargetName="buttonContent"
To="#606467"
Storyboard.TargetProperty="(TextElement.Foreground).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Disabled">
<Storyboard>
<DoubleAnimation
Duration="0"
Storyboard.TargetName="buttonContent"
To=".5"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
```

```
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<ContentPresenter
x:Name="buttonContent"
ContentTemplate="{TemplateBinding
ContentTemplate}"
Content="{TemplateBinding Content}"
TextElement.Foreground="#FFFFFF"
Margin="8 4 1 9"
HorizontalAlignment="{TemplateBinding
HorizontalContentAlignment}"
VerticalAlignment="{TemplateBinding
VerticalContentAlignment}"/>
</Grid>
</ControlTemplate>
</Grid.Resources>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"
MinWidth="79"/>
<ColumnDefinition Width="*"/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
<RowDefinition Height="40"/>
<RowDefinition Height="*"/>
</Grid.RowDefinitions>
<Button x:Name="PART_HeaderButton"
Grid.Column="0"
Focusable="False"
FontWeight="Bold"
FontSize="10.5"
HorizontalAlignment="Center"
Grid.Row="0"
Template="{StaticResource
HeaderButtonTemplate}"
VerticalAlignment="Center"/>
<StackPanel Orientation="Horizontal"
Grid.Column="1"
Grid.Row="0"
Margin="0 13 10 5"
HorizontalAlignment="Right">
<Button x:Name="PART_PreviousButton"
Focusable="False"
Height="17"
Margin="0 0 10 0"
Template="{StaticResource
PreviousButtonTemplate}"
Width="17"/>
<Button x:Name="PART_NextButton"
Focusable="False"
Height="17"
Template="{StaticResource
NextButtonTemplate}"
Width="17"/>
</StackPanel>
<Grid x:Name="PART_MonthView"
Grid.ColumnSpan="3"
HorizontalAlignment="Center"
Margin="6"
Grid.Row="1"
Visibility="Visible">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
<ColumnDefinition Width="30"/>
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
</Grid>
<Grid x:Name="PART_YearView"
Grid.ColumnSpan="3"
HorizontalAlignment="Center"
Margin="6" Grid.Row="1"
Visibility="Hidden">
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Grid.RowDefinitions>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
<RowDefinition Height="Auto"/>
</Grid.RowDefinitions>
</Grid>
</Grid>
</Border>
</Border>
<Rectangle
x:Name="PART_DisabledVisual"
Fill="{StaticResource DisabledColor}"
Opacity="0" RadiusX="2" RadiusY="2"
Stroke="{StaticResource DisabledColor}"
Stretch="Fill" StrokeThickness="1"
Visibility="Collapsed"/>
</Grid>
<ControlTemplate.Triggers>
<Trigger Property="IsEnabled"
Value="False">
<Setter Property="Visibility"
TargetName="PART_DisabledVisual"
Value="Visible"/>
</Trigger>
<DataTrigger Binding="{Binding
DisplayMode,
RelativeSource={RelativeSource
FindAncestor, AncestorType={x:Type
Calendar}}}" Value="Year">
<Setter Property="Visibility"
TargetName="PART_MonthView"
Value="Hidden"/>
<Setter Property="Visibility"
TargetName="PART_YearView"
Value="Visible"/>
</DataTrigger>
<DataTrigger Binding="{Binding
DisplayMode,
RelativeSource={RelativeSource
FindAncestor, AncestorType={x:Type
Calendar}}}" Value="Decade">
<Setter Property="Visibility"
TargetName="PART_MonthView"
Value="Hidden"/>
<Setter Property="Visibility"
TargetName="PART_YearView"
Value="Visible"/>
</DataTrigger>
</ControlTemplate.Triggers>
</ControlTemplate>
<Style x:Key="CalendarStyle"
TargetType="{x:Type Calendar}">
<Setter Property="Foreground"
Value="#FFFFFF"/>
<Setter Property="Background"
Value="#282F33"/>
<Setter Property="BorderBrush"
Value="Transparent"/>
<Setter Property="BorderThickness"
Value="0"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type
Calendar}">
<StackPanel x:Name="PART_Root"
HorizontalAlignment="Center">
<CalendarItem
```

```
Template="{DynamicResource
CalendarItemTemplate}"
x:Name="PART_CalendarItem"
Background="{TemplateBinding
Background}"
BorderBrush="{TemplateBinding
BorderBrush}"
BorderThickness="{TemplateBinding
BorderThickness}"
Style="{TemplateBinding
CalendarItemStyle}"/>
</StackPanel>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style
x:Key="CalendarCalendarDayButtonStyle"
TargetType="{x:Type
CalendarDayButton}">
<Setter Property="Width" Value="25"/>
<Setter Property="Height" Value="25"/>
<Setter Property="FontSize" Value="10"/>
<Setter Property="Cursor" Value="Hand"/>
<Setter Property="Background"
Value="Transparent"/>
<Setter
Property="HorizontalContentAlignment"
Value="Center"/>
<Setter
Property="VerticalContentAlignment"
Value="Center"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type
CalendarDayButton}">
<Grid>
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0:0:0.1"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Normal"/>
<VisualState x:Name="MouseOver">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="HighlightBackgr
ound" To="0.5"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Pressed">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="HighlightBackgr
ound" To="0.5"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Disabled">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="HighlightBackgr
ound" To="0"
Storyboard.TargetProperty="Opacity"/>
<DoubleAnimation Duration="0"
Storyboard.TargetName="NormalText"
To=".35"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="SelectionStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Unselected"/>
<VisualState x:Name="Selected">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="SelectedBackgro
und" To=".75"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="CalendarButtonFocusStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState
x:Name="CalendarButtonFocused">
<Storyboard>
<ObjectAnimationUsingKeyFrames
Duration="0"
Storyboard.TargetName="DayButtonFocus
Visual"
Storyboard.TargetProperty="Visibility">
<DiscreteObjectKeyFrame KeyTime="0">
<DiscreteObjectKeyFrame.Value>
<Visibility>Visible</Visibility>
</DiscreteObjectKeyFrame.Value>
</DiscreteObjectKeyFrame>
</ObjectAnimationUsingKeyFrames>
</Storyboard>
</VisualState>
<VisualState
x:Name="CalendarButtonUnfocused">
<Storyboard>
<ObjectAnimationUsingKeyFrames
Duration="0"
Storyboard.TargetName="DayButtonFocus
Visual"
Storyboard.TargetProperty="Visibility">
<DiscreteObjectKeyFrame KeyTime="0">
<DiscreteObjectKeyFrame.Value>
<Visibility>Collapsed</Visibility>
</DiscreteObjectKeyFrame.Value>
</DiscreteObjectKeyFrame>
</ObjectAnimationUsingKeyFrames>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="ActiveStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Active">
<Storyboard>
<ColorAnimation
Duration="0"
Storyboard.TargetName="NormalText"
To="#FFFFFF"
Storyboard.TargetProperty="(TextElement.
Foreground).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Inactive">
<Storyboard>
<ColorAnimation
Duration="0"
Storyboard.TargetName="NormalText"
To="#606467"
Storyboard.TargetProperty="(TextElement.
Foreground).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup x:Name="DayStates">
<VisualStateGroup.Transitions>
<VisualTransition
```

```
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="RegularDay"/>
<VisualState x:Name="Today">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="TodayBackgroun
d" To="1"
Storyboard.TargetProperty="Opacity"/>
<ColorAnimation Duration="0"
Storyboard.TargetName="NormalText"
To="#FFFFFFFF"
Storyboard.TargetProperty="(TextElement.
Foreground).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="BlackoutDayStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="NormalDay"/>
<VisualState x:Name="BlackoutDay">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="Blackout"
To=".2"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<Rectangle x:Name="TodayBackground"
Fill="#FD413C"
Opacity="0"
RadiusX="20"
RadiusY="20"/>
<Rectangle x:Name="SelectedBackground"
Fill="#FD413C"
Opacity="0"
RadiusX="20"
RadiusY="20"/>
<Border Background="{TemplateBinding
Background}"
BorderBrush="{TemplateBinding
BorderBrush}"
BorderThickness="{TemplateBinding
BorderThickness}"/>
<Rectangle
x:Name="HighlightBackground"
Fill="#FD413C"
Opacity="0"
RadiusX="20"
RadiusY="20"/>
<ContentPresenter x:Name="NormalText"
TextElement.Foreground="#FF333333"
HorizontalAlignment="{TemplateBinding
HorizontalContentAlignment}"
Margin="5,1,5,1"
VerticalAlignment="{TemplateBinding
VerticalContentAlignment}"/>
<Path x:Name="Blackout"
Data="M8.1772461,11.029181
L10.433105,11.029181
L11.700684,12.801641
L12.973633,11.029181
L15.191895,11.029181
L12.844727,13.999395
L15.21875,17.060919
L12.962891,17.060919
L11.673828,15.256231
L10.352539,17.060919
L8.1396484,17.060919
L10.519043,14.042364 z"
Fill="#FFFFFF"
HorizontalAlignment="Stretch"
Margin="3"
Opacity="0"
RenderTransformOrigin="0.5,0.5"
Stretch="Fill"
VerticalAlignment="Stretch"/>
<Rectangle
x:Name="DayButtonFocusVisual"
IsHitTestVisible="false"
RadiusX="20"
RadiusY="20"
Stroke="Transparent"
Visibility="Collapsed"/>
</Grid>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style
x:Key="CalendarCalendarButtonStyle"
TargetType="{x:Type CalendarButton}">
<Setter Property="Background"
Value="#FD413C"/>
<Setter Property="Foreground"
Value="#FFFFFF"/>
<Setter Property="MinWidth"
Value="40"/>
<Setter Property="MinHeight"
Value="42"/>
<Setter Property="FontSize" Value="10"/>
<Setter
Property="HorizontalContentAlignment"
Value="Center"/>
<Setter
Property="VerticalContentAlignment"
Value="Center"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type
CalendarButton}">
<Grid>
<VisualStateManager.VisualStateGroups>
<VisualStateGroup
x:Name="CommonStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0:0:0.1"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Normal"/>
<VisualState x:Name="MouseOver">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="Background"
To=".5"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
<VisualState x:Name="Pressed">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="Background"
To=".5"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="SelectionStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Unselected"/>
<VisualState x:Name="Selected">
<Storyboard>
<DoubleAnimation Duration="0"
Storyboard.TargetName="SelectedBackgro
und" To=".75"
Storyboard.TargetProperty="Opacity"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
```

```
<VisualStateGroup
x:Name="ActiveStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState x:Name="Active"/>
<VisualState x:Name="Inactive">
<Storyboard>
<ColorAnimation Duration="0"
Storyboard.TargetName="NormalText"
To="#FF777777"
Storyboard.TargetProperty="(TextElement.
Foreground).(SolidColorBrush.Color)"/>
</Storyboard>
</VisualState>
</VisualStateGroup>
<VisualStateGroup
x:Name="CalendarButtonFocusStates">
<VisualStateGroup.Transitions>
<VisualTransition
GeneratedDuration="0"/>
</VisualStateGroup.Transitions>
<VisualState
x:Name="CalendarButtonFocused">
<Storyboard>
<ObjectAnimationUsingKeyFrames
Duration="0"
Storyboard.TargetName="CalendarButtonF
ocusVisual"
Storyboard.TargetProperty="Visibility">
<DiscreteObjectKeyFrame KeyTime="0">
<DiscreteObjectKeyFrame.Value>
<Visibility>Visible</Visibility>
</DiscreteObjectKeyFrame.Value>
</DiscreteObjectKeyFrame>
</ObjectAnimationUsingKeyFrames>
</Storyboard>
</VisualState>
<VisualState
x:Name="CalendarButtonUnfocused">
<Storyboard>
<ObjectAnimationUsingKeyFrames
Duration="0"
Storyboard.TargetName="CalendarButtonF
ocusVisual"
Storyboard.TargetProperty="Visibility">
<DiscreteObjectKeyFrame KeyTime="0">
<DiscreteObjectKeyFrame.Value>
<Visibility>Collapsed</Visibility>
</DiscreteObjectKeyFrame.Value>
</DiscreteObjectKeyFrame>
</ObjectAnimationUsingKeyFrames>
</Storyboard>
</VisualState>
</VisualStateGroup>
</VisualStateManager.VisualStateGroups>
<Rectangle x:Name="SelectedBackground"
Fill="{TemplateBinding Background}"
Opacity="0" RadiusX="20"
RadiusY="20"/>
<Rectangle x:Name="Background"
Fill="{TemplateBinding Background}"
Opacity="0" RadiusX="20"
RadiusY="20"/>
<ContentPresenter x:Name="NormalText"
TextElement.Foreground="#FFFFFF"
HorizontalAlignment="{TemplateBinding
HorizontalContentAlignment}"
Margin="1,0,1,1"
VerticalAlignment="{TemplateBinding
VerticalContentAlignment}"/>
<Rectangle
x:Name="CalendarButtonFocusVisual"
IsHitTestVisible="false" RadiusX="20"
RadiusY="20" Stroke="#FD413C"
Visibility="Collapsed"/>
</Grid>
<ControlTemplate.Triggers>
<Trigger Property="IsFocused"
Value="True">
<Setter Property="Visibility"
TargetName="CalendarButtonFocusVisual"
Value="Visible"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="{x:Type
local:DatePickerEx}"
BasedOn="{StaticResource
DatePickerStyle}">
<Style.Resources>
<Style TargetType="{x:Type Calendar}"
x:Key="DatePickerEx_CustomPopup">
<Setter Property="CalendarButtonStyle"
Value="{DynamicResource
CalendarCalendarButtonStyle}"/>
<Setter
Property="CalendarDayButtonStyle"
Value="{DynamicResource
CalendarCalendarDayButtonStyle}"/>
<Setter Property="FontFamily"
Value="Trebuchet MS"/>
<Setter Property="Foreground"
Value="#FFFFFF"/>
<Setter Property="Background"
Value="#282F33"/>
<Setter
Property="VerticalContentAlignment"
Value="Center"/>
<Setter Property="BorderBrush"
Value="Transparent"/>
<Setter Property="BorderThickness"
Value="0"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type
Calendar}">
<StackPanel x:Name="PART_Root"
HorizontalAlignment="Center">
<CalendarItem
Template="{DynamicResource
CalendarItemTemplate}"
x:Name="PART_CalendarItem"
Background="{TemplateBinding
Background}"
BorderBrush="{TemplateBinding
BorderBrush}"
BorderThickness="{TemplateBinding
BorderThickness}"
Style="{TemplateBinding
CalendarItemStyle}"/>
</StackPanel>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</Style.Resources>
</Style>
</ResourceDictionary>
```

ItemsControlNulTarget.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winf
x/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/wi
nfx/2006/xaml"
xmlns:bf="clr-
namespace:INCOMSYSTEM.BehaviorsFil
es">
<bf:AnyItemsToVisibilityConverter
x:Key="AnyItemsToVisibilityConverter"/>
<Style TargetType="{x:Type
ItemsControl}" x:Key="ItemsVisible">
<Setter Property="Template">
<Setter.Value>
```

```
<ControlTemplate TargetType="{x:Type ItemsControl}">
<ScrollViewer VerticalScrollBarVisibility="Auto" HorizontalScrollBarVisibility="Disabled">
<Grid>
<TextBlock Margin="30 5 0 0" Text="Список пуст" VerticalAlignment="Top" Style="{StaticResource TextBlockStyle}" Visibility="{Binding Path=Items, RelativeSource={RelativeSource TemplatedParent}, UpdateSourceTrigger=PropertyChanged, Converter={StaticResource AnyItemsToVisibilityConverter}}"/>
<ItemsPresenter/>
</Grid>
</ScrollViewer>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</ResourceDictionary>
```

MainStyle.xaml

```
<ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:controls="clr-namespace:INCOMSYSTEM.Controls">
<Style x:Key="MainStyle">
<Setter Property="TextBlock.FontFamily" Value="/INCOMSYSTEM;component/Fonts/#Roboto"/>
<Setter Property="TextBlock.FontSize" Value="16"/>
<Setter Property="TextBlock.Foreground" Value="White"/>
<Setter Property="TextBox.IsTabStop" Value="False"/>
</Style>
<Style x:Key="TextPassBoxStyle" BasedOn="{StaticResource MainStyle}">
<Setter Property="TextBox.BorderBrush" Value="White"/>
<Setter Property="TextBox.BorderThickness" Value="0 0 0 1"/>
<Setter Property="TextBox.Padding" Value="5 2"/>
<Setter Property="TextBox.IsTabStop" Value="True"/>
<Setter Property="TextBox.Margin" Value="0"/>
<Setter Property="TextBox.CaretBrush" Value="White"/>
<Setter Property="TextBox.Background" Value="Transparent"/>
<Setter Property="TextBox.Cursor" Value="IBeam"/>
<Setter Property="TextBox.Template">
<Setter.Value>
<ControlTemplate>
<Grid Background="{TemplateBinding TextBox.Background}" Width="{TemplateBinding TextBox.Width}" x:Name="PART_Grid">
<Border BorderBrush="White" BorderThickness="0 0 0 1" Margin="5 0" x:Name="PART_Border"/>
<ScrollViewer Margin="{TemplateBinding TextBox.Padding}" Foreground="{TemplateBinding TextBox.Foreground}" x:Name="PART_ContentHost"/>
</Grid>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style x:Key="TextBlockStyle" TargetType="TextBlock" BasedOn="{StaticResource MainStyle}"></Style>
<Style x:Key="TextBoxStyle" TargetType="TextBox" BasedOn="{StaticResource TextPassBoxStyle}"></Style>
<Style TargetType="controls:InputTextBox" BasedOn="{StaticResource MainStyle}">
<Setter Property="BorderBrush" Value="White"/>
<Setter Property="BorderThickness" Value="0 0 0 1"/>
<Setter Property="Padding" Value="3 2"/>
<Setter Property="IsTabStop" Value="True"/>
<Setter Property="Margin" Value="0"/>
<Setter Property="CaretBrush" Value="White"/>
<Setter Property="Background" Value="Transparent"/>
<Setter Property="Cursor" Value="IBeam"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="controls:InputTextBox">
<Grid Background="{TemplateBinding Background}" Width="{TemplateBinding Width}" x:Name="PART_Grid">
<Border BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThickness}" CornerRadius="5" Width="{TemplateBinding ActualWidth}" HorizontalAlignment="Left" VerticalAlignment="Bottom" x:Name="PART_Border"/>
<ScrollViewer Margin="{TemplateBinding Padding}" Foreground="{TemplateBinding Foreground}" x:Name="PART_ContentHost"/>
</Grid>
<ControlTemplate.Triggers>
<Trigger Property="IsPlaceHolder" Value="True">
<Setter TargetName="PART_ContentHost" Property="Opacity" Value="0.7"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="PasswordBox" BasedOn="{StaticResource TextPassBoxStyle}"></Style>
<Style x:Key="LabelStyle" TargetType="Label" BasedOn="{StaticResource MainStyle}"></Style>
<Style TargetType="RichTextBox" BasedOn="{StaticResource MainStyle}">
<Setter Property="Background" Value="Transparent"/>
<Setter Property="BorderBrush" Value="Transparent"/>
<Setter Property="BorderThickness"
```

```
Value="0"/>
<Setter Property="IsReadOnly" Value="True"/>
</Style>
<Style TargetType="ContextMenu">
<Setter Property="Background" Value="Gray"/>
<Setter Property="Foreground" Value="White"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="ContextMenu">
<Border x:Name="Border" Background="{TemplateBinding Background}" Padding="3">
<StackPanel IsItemsHost="True" KeyboardNavigation.DirectionalNavigation="Cycle"/>
</Border>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style x:Key="DatePickerTextBoxStyle" TargetType="{x:Type DatePickerTextBox}" BasedOn="{StaticResource MainStyle}">
<Setter Property="CaretBrush" Value="White"/>
<Setter Property="BorderBrush" Value="White"/>
<Setter Property="Margin" Value="0 4"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type DatePickerTextBox}">
<Border CornerRadius="5" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="0 0 0 1">
<ScrollViewer x:Name="PART_ContentHost"/>
</Border>
<ControlTemplate.Triggers>
<Trigger Property="IsEnabled" Value="False">
<Setter Property="Background" Value="{DynamicResource {x:Static SystemColors.ControlBrushKey}}"/>
<Setter Property="Foreground" Value="{DynamicResource {x:Static SystemColors.GrayTextBrushKey}}"/>
</Trigger>
<Trigger Property="Width" Value="Auto">
<Setter Property="MinWidth" Value="100"/>
</Trigger>
<Trigger Property="Height" Value="Auto">
<Setter Property="MinHeight" Value="20"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="{x:Type Button}" x:Key="BtnDatePickerStyle">
<Setter Property="Cursor" Value="Hand" />
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type Button}">
<ContentPresenter />
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style x:Key="DatePickerStyle" TargetType="{x:Type DatePicker}">
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type DatePicker}">
<Grid x:Name="PART_Root">
<Grid.ColumnDefinitions>
<ColumnDefinition />
<ColumnDefinition Width="Auto" />
</Grid.ColumnDefinitions>
<DatePickerTextBox x:Name="PART_TextBox" IsReadOnly="True" BorderBrush="{TemplateBinding BorderBrush}" Style="{StaticResource DatePickerTextBoxStyle}"/>
<Button Background="Transparent" Grid.Column="1" Style="{StaticResource BtnDatePickerStyle}" x:Name="PART_Button">
<Path Height="30" HorizontalAlignment="Center" Margin="4,3,4,3" Stretch="Fill" Stroke="Wheat" Fill="#FFFFFF" StrokeThickness="2" VerticalAlignment="Center" Width="20">
<Path.Data>
M5,5 h50 c0,0 40,30 80,0 c0,0 -40,-30 -80,0
m80,0 h75 c0,0 40,30 80,0 c0,0 -40,-30 -80,0
m80,0 h75 c0,0 40,30 80,0 c0,0 -40,-30 -80,0
m80,0 h75 c0,0 40,30 80,0 c0,0 -40,-30 -80,0
m80,0 h50 v50 H5 V5 h1
M5,55 v130 h645 V55 z
M5,195 h640 c0,0 2,5 0,1 h-640 c0,0 -2,-5 0,-1
M5,205 h640 c0,0 2,5 0,1 h-640 c0,0 -2,-5 0,-1
M40,70 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M160,70 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M280,70 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M420,70 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M560,70 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M40,100 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M160,100 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M280,100 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M420,100 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M560,100 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M40,130 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M160,130 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M280,130 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M420,130 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M560,130 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M40,160 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-
```

```
5
M160,160 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M280,160 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M420,160 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
M560,160 h50 c0,0 5,5 0,5 h-50 c0,0 -5,-5 0,-5
</Path.Data>
</Path>
</Button>
<Grid
Grid.Column="0"
Grid.ColumnSpan="2"
IsHitTestVisible="False"
Opacity="0"
x:Name="PART_DisabledVisual">
<Popup
AllowsTransparency="True"
Placement="Bottom"
PlacementTarget="{Binding
ElementName=PART_TextBox}"
StaysOpen="False"
x:Name="PART_Popup" />
</Grid>
</Grid>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="MenuItem">
<Setter Property="Template">
<Setter.Value>
<ControlTemplate
TargetType="MenuItem">
<Border MaxWidth="175" MinWidth="30"
Background="{TemplateBinding
Background}" Padding="5">
<Grid>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="Auto"/>
<ColumnDefinition Width="Auto"/>
</Grid.ColumnDefinitions>
<Border x:Name="BorderIcon"
BorderBrush="White"
BorderThickness="0 0 1 0"
Padding="0 0 5 0"
Margin="0 0 5 0">
<Label x:Name="IconMenu"
Content="{TemplateBinding Icon}"
Padding="0"
Margin="0"
Style="{x:Null}"
Width="20"/>
</Border>
<TextBlock Text="{TemplateBinding
Header}" Grid.Column="1"
FontSize="15">
<TextBlock.ToolTip>
<Binding RelativeSource="{RelativeSource
Mode=Self}" Path="Text"/>
</TextBlock.ToolTip>
</TextBlock>
</Grid>
</Border>
<ControlTemplate.Triggers>
<DataTrigger Binding="{Binding
ElementName=IconMenu, Path=Content}"
Value="{x:Null}">
<Setter TargetName="BorderIcon"
Property="Visibility" Value="Collapsed"/>
</DataTrigger>
<Trigger Property="IsMouseOver"
Value="True">
<Setter Property="Background"
Value="#5d514b"/>
</Trigger>
</ControlTemplate.Triggers>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style TargetType="CheckBox"
x:Key="ClearCheckBoxTemplate">
<Setter Property="Cursor" Value="Hand"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate
TargetType="CheckBox">
<Label Content="{TemplateBinding
Content}" Padding="0" Style="{x:Null}"/>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
<Style x:Key="SideBarButton"
TargetType="Button">
<Setter Property="Cursor" Value="Hand"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="Button">
<ContentPresenter/>
</ControlTemplate>
</Setter.Value>
</Setter>
</Style>
</ResourceDictionary>
```

WindowStyle.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
<Style TargetType="{x:Type Button}"
x:Key="MainButton"
BasedOn="{StaticResource {x:Type
Button}}">
<Setter Property="Background"
Value="#2FAA87"/>
<Setter Property="Foreground"
Value="White"/>
<Setter Property="BorderBrush"
Value="Coral"/>
<Setter Property="BorderThickness"
Value="2"/>
<Setter Property="Padding" Value="15 5"/>
<Setter Property="Template">
<Setter.Value>
<ControlTemplate TargetType="{x:Type
Button}">
<Border BorderBrush="{TemplateBinding
BorderBrush}"
BorderThickness="{TemplateBinding
BorderThickness}"
Padding="{TemplateBinding Padding}"
Background="{TemplateBinding
Background}"
Cursor="{TemplateBinding Cursor}"
CornerRadius="5">
<TextBlock Text="{TemplateBinding
Content}" Foreground="{TemplateBinding
Foreground}" Style="{StaticResource
TextBlockStyle}" />
</Border>
</ControlTemplate>
</Setter.Value>
</Setter>
<Style.Triggers>
<Trigger Property="IsEnabled"
Value="False">
<Setter Property="Background"
Value="Gray"/>
<Setter Property="Foreground"
Value="#FFAE5F"/>
</Trigger>
<Trigger Property="IsMouseOver"
Value="True">
```

```
<Setter Property="Background" Value="#3946AA"/>
</Trigger>
<Trigger Property="IsPressed" Value="True">
<Setter Property="Background" Value="#171717"/>
</Trigger>
</Style.Triggers>
</Style>
</ResourceDictionary>
```

App.xaml

```
<Application
x:Class="INCOMSYSTEM.App"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:bf="clr-namespace:INCOMSYSTEM.BehaviorsFiles"
StartupUri="Windows/AuthWindow.xaml"
>
<Application.Resources>
<ResourceDictionary>
<bf:ActualWidthConverter x:Key="WidthConverter"/>
<ResourceDictionary.MergedDictionaries>
<ResourceDictionary Source="Resources/MainStyle.xaml"/>
<ResourceDictionary Source="Resources/ButtonStyle.xaml"/>
<ResourceDictionary Source="Resources/ComboBoxStyle.xaml" />
<ResourceDictionary Source="Resources/DatePickerExStyle.xaml"/>
<ResourceDictionary Source="Resources/ItemsControlNullTarget.xaml"/>
</ResourceDictionary.MergedDictionaries>
</ResourceDictionary>
</Application.Resources>
</Application>
```

Orders.cs

```
namespace INCOMSYSTEM.Context
{
using System;
using System.Windows;
using System.Collections.Generic;
using INCOMSYSTEM.Windows;
public partial class Orders
{
public long id { get; set; }
public long idCustomer { get; set; }
public Nullable<long> idExecutor { get; set; }
public long idTask { get; set; }
public decimal price { get; set; }
public double difficulty { get; set; }
public System.DateTime dateOrder { get; set; }
public Nullable<System.DateTime> planDateStart { get; set; }
public Nullable<System.DateTime> factDateStart { get; set; }
public Nullable<System.DateTime> planDateComplete { get; set; }
public Nullable<System.DateTime> factDateComplete { get; set; }
public byte[] attachment { get; set; }
public string fileExtension { get; set; }
public byte idStatus { get; set; }
public virtual Chats Chats { get; set; }
public virtual Customers Customers { get; set; }
public virtual Employees Employees { get; set; }
public virtual Statuses Statuses { get; set; }
public virtual Tasks Tasks { get; set; }
public Visibility CanViewDetail => Chats.idManager != null ? Visibility.Visible : Visibility.Collapsed;
public Visibility CanJoinChat => (Chats.idManager == null || MainWindow.AuthUser?.idUser == Chats.idManager) && MainWindow.AuthUser?.idPos == 3 ? Visibility.Visible : Visibility.Collapsed;
public Visibility CanFormAgreement => (idExecutor != null && planDateStart != null && planDateComplete != null) && MainWindow.AuthUser?.idPos == 3 ? Visibility.Visible : Visibility.Collapsed;
public Visibility CanSetFactStartDate => MainWindow.AuthUser?.idPos == 2 && factDateStart == null && planDateStart != null && planDateComplete != null ? Visibility.Visible : Visibility.Collapsed;
public Visibility CanSetFactCompleteDate => MainWindow.AuthUser?.idPos == 2 && factDateStart != null && factDateComplete == null ? Visibility.Visible : Visibility.Collapsed;
public bool IsStartLateYellow => factDateStart > planDateStart?.AddDays(5);
public bool IsStartLateRed => factDateStart > planDateComplete;
public bool IsCompleteLateYellow => factDateComplete > planDateComplete?.AddDays(5);
public bool IsCompleteLateRed => factDateComplete > planDateComplete;
}
}
```

Tasks.cs

```
using INCOMSYSTEM.BehaviorsFiles;
namespace INCOMSYSTEM.Context
{
using System;
using System.Collections.Generic;
public partial class Tasks
{
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
public Tasks()
{
this.Orders = new HashSet<Orders>();
}
public long id { get; set; }
public string name { get; set; }
public int idSpecialization { get; set; }
public string description { get; set; }
public decimal price { get; set; }
public Nullable<byte> discount { get; set; }
public int approxCompleteTime { get; set; }
public int supportPeriod { get; set; }
public byte[] attachment { get; set; }
public string fileExtension { get; set; }
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
public virtual ICollection<Orders> Orders { get; set; }
public virtual Specializations Specializations { get; set; }
```

```
public bool discoutStyle => discount != null && discount > 10;
public decimal newPrice => discount != null ? (decimal)(price - (price * (discount / 100m))) : price;
public bool discountVisible => discount != null && discount > 0;
public string shortDescription => description.Length > 60 ? description.Substring(0, 60) + "..." : description;
public string approxString => approxCompleteTime.ConvertDay();
}
}
```

Employee.cs

```
namespace INCOMSYSTEM.Context
{
using System;
using System.Collections.Generic;
public partial class Employees
{
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
public Employees()
{
this.Chats = new HashSet<Chats>();
this.Orders = new HashSet<Orders>();
this.SpecializationsEmployee = new HashSet<SpecializationsEmployee>();
}
public override string ToString()
{
return $"{surname} {name} {patronymic}".Trim();
}
public long idUser { get; set; }
public string surname { get; set; }
public string name { get; set; }
public string patronymic { get; set; }
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
public virtual ICollection<Chats> Chats { get; set; }
public virtual UsersDetail UsersDetail { get; set; }
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
public virtual ICollection<Orders> Orders { get; set; }
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
public virtual ICollection<SpecializationsEmployee> SpecializationsEmployee { get; set; }
}
}
```

UsersDetail.cs

```
namespace INCOMSYSTEM.Context
{
using System;
using System.Collections.Generic;
public partial class UsersDetail
{
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
public UsersDetail()
{
this.Messages = new HashSet<Messages>();
}
public long idUser { get; set; }
public string login { get; set; }
public string password { get; set; }
public Nullable<long> phone { get; set; }
public byte idPos { get; set; }
public long passport { get; set; }
public string address { get; set; }
public System.DateTime dateStart { get; set; }
public virtual Customers Customers { get; set; }
public virtual Employees Employees { get; set; }
[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
public virtual ICollection<Messages> Messages { get; set; }
public virtual Positions Positions { get; set; }
public string SeriePassport => passport.ToString().Substring(0, 4);
public string NumberPassport => passport.ToString().Substring(4, 6);
}
}
```

Messages.cs

```
namespace INCOMSYSTEM.Context
{
using System;
using System.Collections.Generic;
public partial class Messages
{
public long id { get; set; }
public long idChat { get; set; }
public long idUser { get; set; }
public string message { get; set; }
public byte[] attachment { get; set; }
public string fileExtension { get; set; }
public System.DateTime dateSend { get; set; }
public virtual Chats Chats { get; set; }
public virtual UsersDetail UsersDetail { get; set; }
public string shortMessage => message.Length > 125 ? message.Substring(0, 125) + "..." : message;
}
}
```