**Задание 1. Разработка пользовательских историй и сценариев использования**

Таблица 1

|  |  |
| --- | --- |
| **Пользователь Админ** | **Система** |
| 1. Входит в систему | Проверяет учетные данные, предоставляет доступ или сообщает об ошибке |
| 1. Нажимает на кнопку регистрации нового пользователя. 2. Заполняет персональные данные пользователя | Добавляет пользователя в базу данных, сообщает об успешной регистрации или ошибке |
| 1. Выбор необходимо пользователя. 2. Нажатие на checkbox «Уволен» | Обновляет статус пользователя в базе данных |
| 1. Выбор требуемой смены 2. Выбор необходимого пользователя. | Обновляет информацию о сменах в базе данных |
| 1. Прохождение авторизации 2. Открытие страницы заказов | Отображает список всех заказов |
| **Пользователь Переводчик** | **Система** |
| 1. Входит в систему | Проверяет учетные данные, предоставляет доступ или сообщает об ошибке |
| 1. Прохождение авторизации 2. Открытие страницы заказов | Отображает список заказов, назначенных данному переводчику |
| 1. Открыть заказ 2. Просмотреть текст/ прописать путь к аудио файлу для проигрывания | Отображает/ воспроизводит информацию из заказа |
| 1. Перейти в окно заказов 2. Открыть заказ 3. Изменить статус заказа | Обновляет статус заказа в базе данных |
| **Пользователь Менеджер** | **Система** |
| 1. Входит в систему | Проверяет учетные данные, предоставляет доступ или сообщает об ошибке |
| 1. Перейти в окно заказов | Вывод перечня заказов |
| 1. Перейти в окно заказов 2. Нажать на кнопку «Создать заказ» 3. заполнить данными 4. Сохранить | Добавляет заказ в базу данных, сообщает об успешном создании или ошибке |
| 1. Перейти в окно заказов 2. Открыть заказ 3. Изменить статус | Обновляет статус заказа в базе данных |

Таблица 2

|  |  |  |
| --- | --- | --- |
| Роль | Действие | Цель |
| **Администратор** | Авторизоваться в системе | Вход в систему Администратора |
| 1. Нажать на кнопку регистрации нового пользователя. 2. Заполняет персональные данные пользователя | Добавить в систему нового пользователя |
| 1. Выбор необходимо пользователя. 2. Нажать на checkbox «Уволен» | Уволить сотрудника |
| 1. Выбор требуемой смены 2. Выбор необходимого пользователя. | Назначение пользователей на смены. |
| 1. Прохождение авторизации 2. Открытие страницы заказов | Вывод заказов клиентов |
| **Переводчик** | Авторизоваться в системе | Вход в систему Переводчика |
| Открыть страницу заказов | Просмотр заказов клиента |
| 1. Открыть страницу заказа 2. Выбрать заказ 3. Просмотреть текст/ прописать путь к аудио файлу для проигрывания | Просмотреть текст/аудио, принятый от клиента |
| 1. Открыть страницу заказа 2. Выбрать заказ 3. Изменить статус заказа | Изменить статуса заказа (готовится, готов). |
| **Менеджер** | Авторизоваться в системе | Вход в систему Менеджера |
| Открыть страницу заказов | Просмотр заказов клиента |
| 1. Открыть страницу заказов. Нажать на кнопку «Создать заказ» 2. Заполнить данными 3. Сохранить | Добавить новый заказа клиента |
| 1. Открыть страницу заказа 2. Выбрать заказ 3. Изменить статус заказа | Изменить статусы заказа (принят, оплачен). |

**Задание 2. Постановка задач по разработке информационной системы**

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Входные параметры | Выходные параметры | Описание функции |
| Как функция называется? | Какие параметры функция принимает? | Какие параметры функция возвращает? | Что делает функция? |
| RegisterUser | string username, string password, int role, bool status | bool success, string errorMessage | Регистрация нового пользователя в системе |
| FireUser | int userId | bool success, string errorMessage | Перевод пользователя в статус "уволен" |
| AssignShift | int userId, int shiftId | bool success, string errorMessage | Назначение менеджеров и переводчиков на смены |
| ViewAllOrders | None | List<Order> orders | Просмотр всех заказов |
| ViewOrdersByTranslator | int translatorId | List<Order> orders | Просмотр заказов, принятых от клиентов (для переводчика) |
| ChangeOrderStatusTranslator | int orderId, string newStatus | bool success, string errorMessage | Изменение статуса заказа (готовится, готов) |
| CreateOrder | string clientId, string language, int volume, string type | int orderId, bool success, string errorMessage | Создание нового заказа |
| ChangeOrderStatusManager | int orderId, string newStatus | bool success, string errorMessage | Изменение статуса заказа (принят, оплачен) |
| ViewAllEmployees | None | List<User> employees | Просмотр списка всех сотрудников |
| AddEmployee | string username, string password, string role | bool success, string errorMessage | Добавление нового сотрудника |
| ViewAllShifts | None | List<Shift> shifts | Просмотр всех смен |
| CreateShift | DateTime startTime, DateTime endTime | int shiftId, bool success, string errorMessage | Формирование новой смены |
| ViewOrdersByPeriodManager | int managerId, DateTime periodStart, DateTime periodEnd | List<Order> orders | Просмотр заказов, принятых за период активной смены (для менеджера) |

**Задание 3. Разработка прототипов пользовательских интерфейсов системы**

**1.1. Окно авторизации**

xml

Copy code

<Window x:Class="AdminLoginWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Admin Login" Height="200" Width="300">

<Grid>

<StackPanel>

<TextBox Name="Username" PlaceholderText="Username" Margin="10"/>

<PasswordBox Name="Password" PlaceholderText="Password" Margin="10"/>

<Button Content="Login" Click="Login\_Click" Margin="10"/>

</StackPanel>

</Grid>

</Window>

**Mainwindow**

Xml

Title="{Binding Title}" Height="450" Width="800"

DataContext="{Binding RelativeSource={RelativeSource Self}}">

<Grid>

<Frame x:Name="MyFrame" NavigationUIVisibility="Hidden"/>

</Grid>

Cs

public MainWindow()

{

Title = string.Empty;

InitializeComponent();

App.MainWindowInstance = this;

MyFrame.Navigate(new AuthorizationPage());

}

**App**

public partial class App : Application

{

public static DemoEntities DataContext = new DemoEntities();

public static MainWindow MainWindowInstance;

public static AdministratorPage AdministratorPageInstance;

}

**Authpage**

Xml

Title="AuthorizationPage">

<StackPanel HorizontalAlignment="Center" VerticalAlignment="Center" MinWidth="300">

<StackPanel>

<TextBlock Text="Логин"/>

<TextBox x:Name="LoginTB"/>

</StackPanel>

<StackPanel>

<TextBlock Text="Пароль"/>

<PasswordBox x:Name="PasswordTB"/>

</StackPanel>

<Button x:Name="AuthBTN" Content="Войти"/>

</StackPanel>

Cs

public partial class AuthorizationPage : Page

{

public AuthorizationPage()

{

App.MainWindowInstance.Title = "Авторизация";

InitializeComponent();

AuthBTN.Click += (sender, e) =>

{

var user = AuthorizeUser(LoginTB.Text, PasswordTB.Password);

if (user is null) return;

switch (user.IdRole)

{

case 1:

App.MainWindowInstance.MyFrame.Navigate(new AdministratorPage());

break;

default:

MessageBox.Show("У пользователя не задана роль", "Не задана роль", MessageBoxButton.OK, MessageBoxImage.Error);

break;

}

};

}

public User AuthorizeUser(string login, string password)

{

if(string.IsNullOrEmpty(login.Trim()) || string.IsNullOrEmpty(password.Trim()))

{

MessageBox.Show("Поля логина и пароля пустые", "Поля пустые", MessageBoxButton.OK, MessageBoxImage.Error);

return null;

}

var user = App.DataContext.User.FirstOrDefault(u => u.Login.ToLower() == login.Trim().ToLower()

&& u.Password == password.Trim());

if(user is null)

{

MessageBox.Show("Пользователя с таким логином и паролем не существует", "Пользователя не существует", MessageBoxButton.OK, MessageBoxImage.Error);

}

return user;

}

}

**AdminUserPage**

Xml

Title="AdministratorPage"

DataContext="{Binding RelativeSource={RelativeSource Self}}">

<Grid>

<Button x:Name="AddUserBTN" Content="Добавить пользователя" VerticalAlignment="Top" HorizontalAlignment="Right" MinWidth="100" Margin="5"/>

<Button x:Name="AddShiftBTN" Content="Добавить смену" VerticalAlignment="Top" HorizontalAlignment="Left" MinWidth="100" Margin="5"/>

<Button x:Name="ShowOrdersBTN" Content="Заказы" VerticalAlignment="Top" HorizontalAlignment="Left" MinWidth="100" Margin="120 5 5 5"/>

<ListView x:Name="userListView" ItemsSource="{Binding Users}" SelectionMode="Single" Margin="0 30 0 0">

<ListView.ItemContainerStyle>

<Style>

<Setter Property="Control.HorizontalContentAlignment" Value="Stretch"/>

<Setter Property="Control.VerticalContentAlignment" Value="Stretch"/>

</Style>

</ListView.ItemContainerStyle>

<ListView.View>

<GridView>

<GridViewColumn x:Name="idColumn" Header="Индификатор" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="6,-1,-6,-1" Text="{Binding Id}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="surNameColumn" Header="Фамилия" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding SurName}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="nameColumn" Header="Имя" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding Name}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="lastNameColumn" Header="Отчество" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding LastName}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Роль" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding Role.Name}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="loginColumn" Header="Логин" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding Login}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="passwordColumn" Header="Пароль" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Margin="-6,-1,-6,-1" Text="{Binding Password}"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn x:Name="statusColumn" Header="Статус сотрудника" Width="80">

<GridViewColumn.CellTemplate>

<DataTemplate>

<CheckBox IsChecked="{Binding Status}" Checked="CheckBox\_Checked" Unchecked="CheckBox\_Checked" Margin="-6,-1,-6,-1"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

</GridView>

</ListView.View>

</ListView>

</Grid>

Cs

public partial class AdministratorPage : Page

{

public ObservableCollection<User> Users

{

get { return (ObservableCollection<User>)GetValue(UsersProperty); }

set { SetValue(UsersProperty, value); }

}

public static readonly DependencyProperty UsersProperty =

DependencyProperty.Register("Users", typeof(ObservableCollection<User>), typeof(AdministratorPage));

public AdministratorPage()

{

App.DataContext.User.Load();

Users = App.DataContext.User.Local;

InitializeComponent();

App.MainWindowInstance.Title = "Администратор";

AddUserBTN.Click += (sender, e) =>

{

new AddUserWindow().Show();

};

AddShiftBTN.Click += (sender, e) =>

{

App.MainWindowInstance.MyFrame.Navigate(new ShiftUsersPage());

};

ShowOrdersBTN.Click += (sender, e) =>

{

App.MainWindowInstance.MyFrame.Navigate(new OrderPage());

};

}

private void CheckBox\_Checked(object sender, RoutedEventArgs e)

{

App.DataContext.SaveChanges();

}

}

**addUserPage**

xml

Title="Добавление нового пользователя" Height="350" Width="400"

WindowStartupLocation="CenterScreen"

DataContext="{Binding RelativeSource={RelativeSource Self}}"

x:Name="MyWindow">

<Grid>

<Grid DataContext="{Binding User}" HorizontalAlignment="Center" VerticalAlignment="Center">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="Auto"/>

</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>

<Label Grid.Column="0" Content="Фамилия:" HorizontalAlignment="Left" Margin="3" Grid.Row="1" VerticalAlignment="Center"/>

<TextBox x:Name="surNameTextBox" Grid.Column="1" HorizontalAlignment="Left" Height="NaN" Margin="3" Grid.Row="1" Text="{Binding SurName, Mode=TwoWay, NotifyOnValidationError=true, ValidatesOnExceptions=true}" VerticalAlignment="Center" Width="120"/>

<Label Grid.Column="0" Content="Имя:" HorizontalAlignment="Left" Margin="3" Grid.Row="2" VerticalAlignment="Center"/>

<TextBox x:Name="nameTextBox" Grid.Column="1" HorizontalAlignment="Left" Height="NaN" Margin="3" Grid.Row="2" Text="{Binding Name, Mode=TwoWay, NotifyOnValidationError=true, ValidatesOnExceptions=true}" VerticalAlignment="Center" Width="120"/>

<Label Grid.Column="0" Content="Отчество:" HorizontalAlignment="Left" Margin="3" Grid.Row="3" VerticalAlignment="Center"/>

<TextBox x:Name="lastNameTextBox" Grid.Column="1" HorizontalAlignment="Left" Height="NaN" Margin="3" Grid.Row="3" Text="{Binding LastName, Mode=TwoWay, NotifyOnValidationError=true, ValidatesOnExceptions=true}" VerticalAlignment="Center" Width="120"/>

<Label Grid.Column="0" Content="Логин:" HorizontalAlignment="Left" Margin="3" Grid.Row="4" VerticalAlignment="Center"/>

<TextBox x:Name="loginTextBox" Grid.Column="1" HorizontalAlignment="Left" Height="NaN" Margin="3" Grid.Row="4" Text="{Binding Login, Mode=TwoWay, NotifyOnValidationError=true, ValidatesOnExceptions=true}" VerticalAlignment="Center" Width="120"/>

<Label Grid.Column="0" Content="Пароль:" HorizontalAlignment="Left" Margin="3" Grid.Row="5" VerticalAlignment="Center"/>

<TextBox x:Name="passwordTextBox" Grid.Column="1" HorizontalAlignment="Left" Height="NaN" Margin="3" Grid.Row="5" Text="{Binding Password, Mode=TwoWay, NotifyOnValidationError=true, ValidatesOnExceptions=true}" VerticalAlignment="Center" Width="120"/>

<Label Grid.Column="0" Content="Статус:" HorizontalAlignment="Left" Margin="3" Grid.Row="6" VerticalAlignment="Center"/>

<ComboBox Grid.Column="1" ItemsSource="{Binding RolesList, ElementName=MyWindow}" DisplayMemberPath="Name" SelectedItem="{Binding Role}" HorizontalAlignment="Left" Margin="3" Grid.Row="6" VerticalAlignment="Center"/>

</Grid>

<Button x:Name="AddNewUserBTN" Content="Добавить" VerticalAlignment="Bottom" HorizontalAlignment="Center" MinWidth="180" Margin="0 0 0 20"/>

</Grid>

Cs

public partial class AddUserWindow : Window

{

public User User

{

get { return (User)GetValue(UserProperty); }

set { SetValue(UserProperty, value); }

}

public static readonly DependencyProperty UserProperty =

DependencyProperty.Register("User", typeof(User), typeof(AddUserWindow));

public List<Role> RolesList

{

get { return (List<Role>)GetValue(RolesListProperty); }

set { SetValue(RolesListProperty, value); }

}

public static readonly DependencyProperty RolesListProperty =

DependencyProperty.Register("RolesList", typeof(List<Role>), typeof(AddUserWindow));

public AddUserWindow()

{

User = new User();

RolesList = App.DataContext.Role.ToList();

InitializeComponent();

AddNewUserBTN.Click += (sender, e) =>

{

App.DataContext.User.Add(User);

App.DataContext.SaveChanges();

};

}

}

**В бд юсер**

public User()

{

this.Order = new HashSet<Order>();

this.UserShift = new HashSet<UserShift>();

}