Democracy APP

LoginPage.xaml 1

Utilities.cs 3

LoginRequest.cs 3

Response.cs 3

User.cs 3

UserPassword.cs 4

UserPasswordConfirmation.cs 4

LoginPage.xaml.cs 4

HomePage.xaml 6

HomePage.xaml.cs 7

MySettingsPage.xaml 8

MySettingsPage.xaml.cs 10

RegisterPage.xaml 12

RegisterPage.xaml.cs 14

ChangePasswordRequest.cs 17

ChangePasswordPage.xaml 17

ChangePasswordPage.xaml.cs 18

State.cs 20

Candidate.cs 20

Voting.cs 20

MyVotingsPage.xaml 21

MyVotingsPage.xaml.cs 22

MyVotingsCell.cs 23

VotingPage.xaml 24

VotingPage.xaml.cs 25

VotingCell.cs 26

ResultsPage.xaml 27

ResultsPage.xaml.cs 28

ResultCell.cs 29

ResultDetailsPage.xaml 30

ResultDetailsPage.xaml.cs 31

CandidateCell.cs 32

# LoginPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.LoginPage"  
             Title="Login">  
    <ContentPage.Content>  
        <ScrollView>  
            <StackLayout Padding="10, 20, 10, 10" Spacing="20">  
                <Label Text="Democracy"   
                       FontSize="30"   
                       FontAttributes="Bold"  
                       HorizontalOptions="Center"  
                       VerticalOptions="Center"/>  
                <Label Text="Enter your email and password and press 'Login' button"   
                       HorizontalOptions="Center"  
                       VerticalOptions="Center"/>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="E-Mail"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                    <Entry x:Name="emailEntry"   
                           Placeholder="Enter your email"   
                           Keyboard="Email"  
                           Text="debora@gmail.com"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Password"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                    <Entry x:Name="passwordEntry"   
                           Placeholder="Enter your password"   
                           Text="debora@gmail.com"  
                           IsPassword="true"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Rememberme"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                    <Switch x:Name="remembermeSwitch"   
                            IsToggled="true"  
                            HorizontalOptions="Start"  
                            VerticalOptions="Center"/>  
                </StackLayout>  
                <ActivityIndicator x:Name="waitActivityIndicator"  
                                   HorizontalOptions="CenterAndExpand"  
                                   VerticalOptions="CenterAndExpand"/>  
                <Button x:Name="loginButton"  
                        Text="Login"  
                        BackgroundColor="Blue"  
                        TextColor="White"  
                        HorizontalOptions="FillAndExpand"  
                        VerticalOptions="Center"/>  
                <Button x:Name="registerButton"  
                        Text="Register"  
                        BackgroundColor="Purple"  
                        TextColor="White"  
                        HorizontalOptions="FillAndExpand"  
                        VerticalOptions="Center"/>  
            </StackLayout>  
        </ScrollView>  
    </ContentPage.Content>  
</ContentPage>

# Utilities.cs

using System.Text.RegularExpressions;  
  
namespace PreDemocracy  
{  
    public class Utilities  
    {  
        public static bool IsValidEmail(string email)  
        {  
            return Regex.Match(email, @"^([\w\.\-]+)@([\w\-]+)((\.(\w){2,3})+)$").Success;  
        }  
  
        public static bool IsValidPassword(string password)  
        {  
            return Regex.Match(password, @"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[$@$!%\*?&])[A-Za-z\d$@$!%\*?&]{8,}").Success;  
        }  
    }  
}

# LoginRequest.cs

using System;  
  
namespace PreDemocracy  
{  
    public class LoginRequest  
    {  
        public string Email { get; set; }  
  
        public string Password { get; set; }  
    }  
}

# Response.cs

using System;  
  
namespace PreDemocracy  
{  
    public class Response  
    {  
        public string Message { get; set; }  
    }  
}

# User.cs

using System;  
  
namespace PreDemocracy  
{  
    public class User  
    {  
        public int UserId { get; set; }  
  
        public string UserName { get; set; }  
  
        public string FirstName { get; set; }  
  
        public string LastName { get; set; }  
  
        public string FullName { get { return string.Format("{0} {1}", this.FirstName, this.LastName); } }  
  
        public string Phone { get; set; }  
  
        public string Address { get; set; }  
  
        public string Grade { get; set; }  
  
        public string Group { get; set; }  
  
        public string Photo { get; set; }  
    }  
}

# UserPassword.cs

using System;  
  
namespace PreDemocracy  
{  
    public class UserPassword : User  
    {  
        public string CurrentPassword { get; set; }  
    }  
}

# UserPasswordConfirmation.cs

using System;  
  
namespace PreDemocracy  
{  
    public class UserPasswordConfirmation : User  
    {  
        public string Password { get; set; }  
  
        public string ConfirmPassword { get; set; }  
    }  
}

# LoginPage.xaml.cs

using System;  
using System.Collections.Generic;  
using System.Net.Http;  
using Newtonsoft.Json;  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public partial class LoginPage : ContentPage  
    {  
        public LoginPage ()  
        {  
            InitializeComponent ();  
  
            loginButton.Clicked += this.loginButton\_Clicked;  
            registerButton.Clicked += RegisterButton\_Clicked;  
        }  
  
        private async void RegisterButton\_Clicked (object sender, EventArgs e)  
        {  
            await Navigation.PushAsync (new RegisterPage ());  
        }  
  
        private async void loginButton\_Clicked (object sender, EventArgs e)  
        {  
            if (string.IsNullOrEmpty (emailEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter an email", "Acept");  
                emailEntry.Focus ();  
                return;  
            }  
  
            if (!Utilities.IsValidEmail (emailEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a valid email", "Acept");  
                emailEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (passwordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a password", "Acept");  
                passwordEntry.Focus ();  
                return;  
            }  
  
            this.Login ();  
        }  
  
        private async void Login ()  
        {  
            waitActivityIndicator.IsRunning = true;  
  
            var loginRequest = new LoginRequest {  
                Email = emailEntry.Text,  
                Password = passwordEntry.Text,  
            };  
  
            var result = string.Empty;  
  
            try {  
                var jsonRequest = JsonConvert.SerializeObject (loginRequest);  
                var httpContent = new StringContent (jsonRequest, System.Text.Encoding.UTF8, "application/json");  
                var client = new HttpClient ();  
                client.BaseAddress = new Uri ("http://zulu-software.com");  
                var url = "/Democracy/api/Users/Login";  
  
                var response = await client.PostAsync (url, httpContent);  
                if (!response.IsSuccessStatusCode) {  
                    waitActivityIndicator.IsRunning = false;  
                    await DisplayAlert ("Error", "Wrong user or password ", "Acept");  
                    passwordEntry.Text = string.Empty;  
                    passwordEntry.Focus();  
                    return;  
                }  
                result = await response.Content.ReadAsStringAsync ();  
  
            } catch (Exception ex) {  
                await DisplayAlert ("Error", ex.Message, "Acept");  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = false;  
            var user = JsonConvert.DeserializeObject<User> (result);  
            var userPassword = new UserPassword {  
                CurrentPassword = passwordEntry.Text,  
                Address = user.Address,  
                FirstName = user.FirstName,  
                Grade = user.Grade,  
                Group = user.Group,  
                LastName = user.LastName,  
                Phone = user.Phone,  
                Photo = user.Photo,  
                UserId = user.UserId,  
                UserName = user.UserName,  
            };  
  
            await Navigation.PushAsync (new HomePage (userPassword));  
        }  
    }  
}

# HomePage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.HomePage"  
             Title="Home Page">  
    <ContentPage.Content>  
        <ScrollView>  
            <StackLayout Padding="10, 20, 10, 10" Spacing="10">  
                <Image x:Name="photoImage"  
                       HorizontalOptions="Center"   
                       VerticalOptions="Center" />  
                <Label Text="Welcome"   
                       FontSize="20"   
                       FontAttributes="Bold"   
                       HorizontalOptions="Center"   
                       VerticalOptions="Center" />  
                <Label x:Name="userNameLabel"  
                       FontSize="30"   
                       FontAttributes="Bold"   
                       HorizontalOptions="Center"   
                       VerticalOptions="Center" />  
                <StackLayout Orientation="Horizontal">  
                    <Button x:Name="mySettingsButton"  
                            Text="My Settings"  
                            BackgroundColor="Purple"  
                            TextColor="White"  
                            HorizontalOptions="FillAndExpand"  
                            VerticalOptions="Center"/>  
                    <Button x:Name="myVotingsButton"  
                            Text="My Votings"  
                            BackgroundColor="Lime"  
                            TextColor="Black"  
                            HorizontalOptions="FillAndExpand"  
                            VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Button x:Name="resultsButton"  
                            Text="Results"  
                            BackgroundColor="Yellow"  
                            TextColor="Blue"  
                            HorizontalOptions="FillAndExpand"  
                            VerticalOptions="Center"/>  
                    <Button x:Name="logOutButton"  
                            Text="Log Out"  
                            BackgroundColor="Red"  
                            TextColor="White"  
                            HorizontalOptions="FillAndExpand"  
                            VerticalOptions="Center"/>  
                </StackLayout>  
            </StackLayout>  
        </ScrollView>  
    </ContentPage.Content>  
</ContentPage>

# HomePage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public partial class HomePage : ContentPage  
    {  
        private UserPassword user;  
  
        public HomePage (UserPassword user)  
        {  
            InitializeComponent ();  
  
            this.user = user;  
            userNameLabel.Text = user.FullName;  
            if (!string.IsNullOrEmpty (user.Photo)) {  
                photoImage.Source = string.Format ("http://www.zulu-software.com/Democracy{0}", user.Photo.Substring (1));  
            }  
            photoImage.HeightRequest = 280;  
            photoImage.WidthRequest = 280;  
  
            mySettingsButton.Clicked += MySettingsButton\_Clicked;  
        }  
  
        private async void MySettingsButton\_Clicked (object sender, EventArgs e)  
        {  
            await Navigation.PushAsync (new MySettingsPage (this.user));  
        }  
    }  
}

# MySettingsPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.MySettingsPage"  
             Title="My Settings">  
    <ContentPage.Content>  
        <ScrollView>  
            <StackLayout Padding="10, 20, 10, 10" Spacing="10">  
                <Image x:Name="photoImage"  
                       HorizontalOptions="Center"   
                       VerticalOptions="Center" />  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="E-Mail \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="userNameEntry"  
                           Keyboard="Email"  
                           Placeholder="E-Mail"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="First name \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="firstNameEntry"  
                           Placeholder="First name"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Last name \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="lastNameEntry"  
                           Placeholder="Last name"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Telephone \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="phoneEntry"  
                           Keyboard="Telephone"  
                           Placeholder="Phone"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Address \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="addressEntry"  
                           Placeholder="Address"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Grade:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="gradeEntry"  
                           Placeholder="Grade"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Group:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="groupEntry"  
                           Placeholder="Grade"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <ActivityIndicator x:Name="waitActivityIndicator"  
                                   HorizontalOptions="Center"  
                                   VerticalOptions="Center"/>  
                <StackLayout Orientation="Horizontal">  
                    <Button x:Name="changePhotoButton"  
                             Text="Change photo"  
                             BackgroundColor="Navy"  
                             TextColor="White"  
                             HorizontalOptions="FillAndExpand"  
                             VerticalOptions="Center"/>  
                    <Button x:Name="changePasswordButton"  
                             Text="Change password"  
                             BackgroundColor="Yellow"  
                             TextColor="Black"  
                             HorizontalOptions="FillAndExpand"  
                             VerticalOptions="Center"/>  
                </StackLayout>  
                <Button x:Name="saveButton"  
                         Text="Save"  
                         BackgroundColor="Lime"  
                         TextColor="Black"  
                         HorizontalOptions="FillAndExpand"  
                         VerticalOptions="Center"/>  
            </StackLayout>  
        </ScrollView>  
    </ContentPage.Content>  
</ContentPage>

# MySettingsPage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
using Newtonsoft.Json;  
using System.Net.Http;  
  
namespace PreDemocracy  
{  
    public partial class MySettingsPage : ContentPage  
    {  
        private UserPassword user;  
  
        public MySettingsPage (UserPassword user)  
        {  
            InitializeComponent ();  
  
            this.user = user;  
  
            if (!string.IsNullOrEmpty (user.Photo)) {  
                photoImage.Source = string.Format ("http://www.zulu-software.com/Democracy{0}", user.Photo.Substring (1));  
            }  
  
            photoImage.HeightRequest = 280;  
            photoImage.WidthRequest = 280;  
            userNameEntry.Text = user.UserName;  
            firstNameEntry.Text = user.FirstName;  
            lastNameEntry.Text = user.LastName;  
            phoneEntry.Text = user.Phone;  
            addressEntry.Text = user.Address;  
            gradeEntry.Text = user.Grade;  
            groupEntry.Text = user.Group;  
  
            saveButton.Clicked += SaveButton\_Clicked;  
        }  
  
        private async void SaveButton\_Clicked (object sender, EventArgs e)  
        {  
            if (string.IsNullOrEmpty (userNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter an e-mail", "Acept");  
                userNameEntry.Focus ();  
                return;  
            }  
  
            if (!Utilities.IsValidEmail (userNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a valid email", "Acept");  
                userNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (firstNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a first name", "Acept");  
                firstNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (lastNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a last name", "Acept");  
                lastNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (phoneEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a phone", "Acept");  
                phoneEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (addressEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter an address", "Acept");  
                addressEntry.Focus ();  
                return;  
            }  
  
            this.UpdateUser ();  
        }  
  
        private async void UpdateUser ()  
        {  
            waitActivityIndicator.IsRunning = true;  
  
            user.Address = addressEntry.Text;  
            user.FirstName = firstNameEntry.Text;  
            user.Grade = gradeEntry.Text;  
            user.Group = groupEntry.Text;  
            user.LastName = lastNameEntry.Text;  
            user.Phone = phoneEntry.Text;  
            user.UserName = userNameEntry.Text;  
  
            try {  
                var jsonRequest = JsonConvert.SerializeObject (user);  
                var httpContent = new StringContent (jsonRequest, System.Text.Encoding.UTF8, "application/json");  
                var client = new HttpClient ();  
                client.BaseAddress = new Uri ("http://zulu-software.com");  
                var url = string.Format("/Democracy/api/Users/{0}", user.UserId);  
  
                var response = await client.PutAsync(url, httpContent);  
                if (!response.IsSuccessStatusCode) {  
                    waitActivityIndicator.IsRunning = false;  
                    await DisplayAlert ("Error", "Error updating user", "Acept");  
                    return;  
                }  
  
            } catch (Exception ex) {  
                await DisplayAlert ("Error", ex.Message, "Acept");  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = false;  
            await DisplayAlert("Ok", "User updated ok", "Acept");  
            await Navigation.PopAsync ();  
        }  
    }  
}

# RegisterPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.RegisterPage"  
             Title="Register new user">  
    <ContentPage.Content>  
        <ScrollView>  
            <StackLayout Padding="10, 20, 10, 10" Spacing="10">  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="E-Mail \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="userNameEntry"  
                           Keyboard="Email"  
                           Placeholder="E-Mail"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="First name \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="firstNameEntry"  
                           Placeholder="First name"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Last name \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="lastNameEntry"  
                           Placeholder="Last name"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Telephone \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="phoneEntry"  
                           Keyboard="Telephone"  
                           Placeholder="Phone"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Address \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="addressEntry"  
                           Placeholder="Address"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Grade:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="gradeEntry"  
                           Placeholder="Grade"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Group:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="groupEntry"  
                           Placeholder="Grade"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Password \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="passwordEntry"  
                           IsPassword="true"  
                           Placeholder="Password"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Confirm password \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="confirmPasswordEntry"  
                           IsPassword="true"  
                           Placeholder="Confirm password"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <ActivityIndicator x:Name="waitActivityIndicator"  
                                   HorizontalOptions="Center"  
                                   VerticalOptions="Center"/>  
                <Button x:Name="saveButton"  
                         Text="Save"  
                         BackgroundColor="Lime"  
                         TextColor="Black"  
                         HorizontalOptions="FillAndExpand"  
                         VerticalOptions="Center"/>  
            </StackLayout>  
        </ScrollView>  
    </ContentPage.Content>  
</ContentPage>

# RegisterPage.xaml.cs

using System;  
using System.Collections.Generic;  
using System.Net.Http;  
using Newtonsoft.Json;  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public partial class RegisterPage : ContentPage  
    {  
        public RegisterPage ()  
        {  
            InitializeComponent ();  
            saveButton.Clicked += SaveButton\_Clicked;  
        }  
  
        private async void SaveButton\_Clicked (object sender, EventArgs e)  
        {  
            if (string.IsNullOrEmpty (userNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter an e-mail", "Acept");  
                userNameEntry.Focus ();  
                return;  
            }  
  
            if (!Utilities.IsValidEmail (userNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a valid email", "Acept");  
                userNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (firstNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a first name", "Acept");  
                firstNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (lastNameEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a last name", "Acept");  
                lastNameEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (phoneEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a phone", "Acept");  
                phoneEntry.Focus ();  
                return;  
            }  
  
            if (phoneEntry.Text.Length < 7 || phoneEntry.Text.Length > 20) {  
                await DisplayAlert ("Error", "The phone must be between 7 and 20 characters", "Acept");  
                phoneEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (addressEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter an address", "Acept");  
                addressEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (passwordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a password", "Acept");  
                passwordEntry.Focus ();  
                return;  
            }  
  
            if (!Utilities.IsValidPassword(passwordEntry.Text)) {  
                await DisplayAlert ("Error", "The password must be contain minimum 8 " +  
                    "characters at least 1 uppercase alphabet, 1 lowercase Alphabet, " +  
                    "1 number and 1 special character", "Acept");  
                passwordEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (confirmPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a confirm password", "Acept");  
                confirmPasswordEntry.Focus ();  
                return;  
            }  
  
            if (!passwordEntry.Text.Equals(confirmPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "The password and confirm does not match", "Acept");  
                confirmPasswordEntry.Focus ();  
                return;  
            }  
  
            this.RegisterUser ();  
        }  
  
        private async void RegisterUser ()  
        {  
            waitActivityIndicator.IsRunning = true;  
  
            var userPasswordConfirmation = new UserPasswordConfirmation {  
                ConfirmPassword = confirmPasswordEntry.Text,  
                Password = passwordEntry.Text,  
                Address = addressEntry.Text,  
                FirstName = firstNameEntry.Text,  
                Grade = gradeEntry.Text,  
                Group = groupEntry.Text,  
                LastName = lastNameEntry.Text,  
                Phone = phoneEntry.Text,  
                UserName = userNameEntry.Text,  
            };  
  
            var result = string.Empty;  
  
            try {  
                var jsonRequest = JsonConvert.SerializeObject (userPasswordConfirmation);  
                var httpContent = new StringContent (jsonRequest, System.Text.Encoding.UTF8, "application/json");  
                var client = new HttpClient ();  
                client.BaseAddress = new Uri ("http://zulu-software.com");  
                var url = "/Democracy/api/Users";  
  
                var response = await client.PostAsync (url, httpContent);  
                result = await response.Content.ReadAsStringAsync ();  
  
                if (!response.IsSuccessStatusCode) {  
                    waitActivityIndicator.IsRunning = false;  
                    await DisplayAlert ("Error", "Error saving the user", "Acept");  
                    return;  
                }  
  
            } catch (Exception ex) {  
                await DisplayAlert ("Error", ex.Message, "Acept");  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = false;  
            var user = JsonConvert.DeserializeObject<User> (result);  
            var userPassword = new UserPassword {  
                CurrentPassword = passwordEntry.Text,  
                Address = user.Address,  
                FirstName = user.FirstName,  
                Grade = user.Grade,  
                Group = user.Group,  
                LastName = user.LastName,  
                Phone = user.Phone,  
                Photo = user.Photo,  
                UserId = user.UserId,  
                UserName = user.UserName,  
            };  
  
            await Navigation.PushAsync (new HomePage (userPassword));  
        }  
    }  
}

# ChangePasswordRequest.cs

using System;  
  
namespace PreDemocracy  
{  
    public class ChangePasswordRequest  
    {  
        public string OldPassword { get; set; }  
  
        public string NewPassword { get; set; }  
    }  
}

# ChangePasswordPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.ChangePasswordPage"  
             Title="Change password">  
    <ContentPage.Content>  
        <ScrollView>  
            <StackLayout Padding="10, 20, 10, 10" Spacing="10">  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Current password \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="currentPasswordEntry"  
                           IsPassword="true"  
                           Placeholder="Current password"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="New password \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="newPasswordEntry"  
                           IsPassword="true"  
                           Placeholder="Password"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Horizontal">  
                    <Label Text="Confirm password \*:"  
                           HorizontalOptions="Start"   
                           VerticalOptions="Center" />  
                    <Entry x:Name="confirmPasswordEntry"  
                           IsPassword="true"  
                           Placeholder="Confirm password"  
                           HorizontalOptions="FillAndExpand"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <ActivityIndicator x:Name="waitActivityIndicator"  
                                   HorizontalOptions="CenterAndExpand"  
                                   VerticalOptions="CenterAndExpand"/>  
                <Button x:Name="saveButton"  
                         Text="Save"  
                         BackgroundColor="Lime"  
                         TextColor="Black"  
                         HorizontalOptions="FillAndExpand"  
                         VerticalOptions="Center"/>  
            </StackLayout>  
        </ScrollView>  
    </ContentPage.Content>  
</ContentPage>

# ChangePasswordPage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
using Newtonsoft.Json;  
using System.Net.Http;  
  
namespace PreDemocracy  
{  
    public partial class ChangePasswordPage : ContentPage  
    {  
        private UserPassword user;  
  
        public ChangePasswordPage (UserPassword user)  
        {  
            InitializeComponent ();  
            this.user = user;  
  
            saveButton.Clicked += SaveButton\_Clicked;  
        }  
  
        private async void SaveButton\_Clicked (object sender, EventArgs e)  
        {  
            if (string.IsNullOrEmpty (currentPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a current password", "Acept");  
                currentPasswordEntry.Focus ();  
                return;  
            }  
  
            if (!currentPasswordEntry.Text.Equals(this.user.CurrentPassword)) {  
                await DisplayAlert ("Error", "The currect password is wrong", "Acept");  
                confirmPasswordEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (newPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a new current password", "Acept");  
                newPasswordEntry.Focus ();  
                return;  
            }  
  
            if (!Utilities.IsValidPassword(newPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "The password must be contain minimum 8 " +  
                    "characters at least 1 uppercase alphabet, 1 lowercase Alphabet, " +  
                    "1 number and 1 special character", "Acept");  
                newPasswordEntry.Focus ();  
                return;  
            }  
  
            if (string.IsNullOrEmpty (confirmPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "You must enter a confirm password", "Acept");  
                confirmPasswordEntry.Focus ();  
                return;  
            }  
  
            if (!newPasswordEntry.Text.Equals(confirmPasswordEntry.Text)) {  
                await DisplayAlert ("Error", "The password and confirm does not match", "Acept");  
                confirmPasswordEntry.Focus ();  
                return;  
            }  
  
            this.ChangePassword ();  
        }  
  
        private async void ChangePassword ()  
        {  
            waitActivityIndicator.IsRunning = true;  
  
            var changePasswordPage = new ChangePasswordRequest {  
                OldPassword = user.CurrentPassword,  
                NewPassword = newPasswordEntry.Text,  
            };  
  
            try {  
                var jsonRequest = JsonConvert.SerializeObject (changePasswordPage);  
                var httpContent = new StringContent (jsonRequest, System.Text.Encoding.UTF8, "application/json");  
                var client = new HttpClient ();  
                client.BaseAddress = new Uri ("http://zulu-software.com");  
                var url = string.Format("/Democracy/api/Users/ChangePassword/{0}", user.UserId);  
  
                var response = await client.PutAsync(url, httpContent);  
                if (!response.IsSuccessStatusCode) {  
                    waitActivityIndicator.IsRunning = false;  
                    await DisplayAlert ("Error", "Error changing password", "Acept");  
                    return;  
                }  
  
            } catch (Exception ex) {  
                await DisplayAlert ("Error", ex.Message, "Acept");  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = false;  
            await DisplayAlert("Ok", "Password changed ok", "Acept");  
            await Navigation.PopAsync ();  
        }  
    }  
}

# State.cs

public class State  
{  
    public int StateId { get; set; }  
  
    public string Description { get; set; }  
}

# Candidate.cs

public class Candidate  
{  
    public int CandidateId { get; set; }  
  
    public int QuantityVotes { get; set; }  
  
    public User User { get; set; }  
  
    public override string ToString ()  
    {  
        return string.Format ("[Candidate: CandidateId={0}, " +  
            "QuantityVotes={1}, User={2}]",   
            CandidateId, QuantityVotes, User);  
    }  
}

# Voting.cs

public class Voting  
{  
    public int VotingId { get; set; }  
  
    public string Description { get; set; }  
  
    public string Remarks { get; set; }  
  
    public DateTime DateTimeStart { get; set; }  
  
    public DateTime DateTimeEnd { get; set; }  
  
    public bool IsForAllUsers { get; set; }  
  
    public bool IsEnabledBlankVote { get; set; }  
  
    public int QuantityVotes { get; set; }  
  
    public int QuantityBlankVotes { get; set; }  
  
    public User Winner { get; set; }  
  
    public State State { get; set; }  
  
    public List<Candidate> Candidates { get; set; }  
  
    public string DateTimeStartEdited {   
        get {  
            return string.Format("Starts: {0:g}", DateTimeStart);  
        }  
    }  
  
    public string DateTimeEndEdited {   
        get {  
            return string.Format("Ends: {0:g}", DateTimeEnd);  
        }  
    }  
  
    public string QuantityVotesEdited {   
        get {  
            return string.Format("Votes: {0:N0}", QuantityVotes);  
        }  
    }  
  
  
    public override string ToString ()  
    {  
        return string.Format ("[Voting: VotingId={0}, Description={1}, Remarks={2}, DateTimeStart={3}, " +  
            "DateTimeEnd={4}, IsForAllUsers={5}, IsEnabledBlankVote={6}, QuantityVotes={7}, " +  
            "QuantityBlankVotes={8}, Winner={9}, State={10}, Candidates={11}]",   
            VotingId, Description, Remarks, DateTimeStart, DateTimeEnd, IsForAllUsers,   
            IsEnabledBlankVote, QuantityVotes, QuantityBlankVotes, Winner, State, Candidates);  
    }  
}

# MyVotingsPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.MyVotingsPage"  
             Title="My Votings">  
    <ContentPage.Content>  
        <StackLayout Spacing="10">  
            <Label Text="My Votings"   
                   FontSize="30"   
                   FontAttributes="Bold"   
                   HorizontalOptions="Center"   
                   VerticalOptions="Center" />  
            <ActivityIndicator x:Name="waitActivityIndicator"  
                               HorizontalOptions="Center"  
                               VerticalOptions="Center"/>  
            <ListView x:Name="myVotingsListView" />  
        </StackLayout>  
    </ContentPage.Content>  
</ContentPage>

# MyVotingsPage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
using System.Net.Http;  
using Newtonsoft.Json;  
  
namespace PreDemocracy  
{  
    public partial class MyVotingsPage : ContentPage  
    {  
        private UserPassword user;  
  
        public MyVotingsPage (UserPassword user)  
        {  
            InitializeComponent ();  
  
            this.user = user;  
  
            this.Padding = Device.OnPlatform (  
                new Thickness (10, 20, 10, 10),  
                new Thickness (10),  
                new Thickness (10));  
  
            myVotingsListView.ItemTemplate = new DataTemplate(typeof(MyVotingsCell));  
            myVotingsListView.RowHeight = 80;  
            myVotingsListView.ItemSelected += MyVotingsListView\_ItemSelected;  
        }  
  
        protected override void OnAppearing ()  
        {  
            base.OnAppearing ();  
            this.LoadVotings ();  
        }  
      
        private async void MyVotingsListView\_ItemSelected (object sender, SelectedItemChangedEventArgs e)  
        {  
            await Navigation.PushAsync (new VotingPage ((Voting)e.SelectedItem, user));  
        }  
  
        private async void LoadVotings ()  
        {  
            waitActivityIndicator.IsRunning = true;  
            var client = new HttpClient();  
            client.BaseAddress = new Uri ("http://zulu-software.com");  
            var url = string.Format("/Democracy/api/Votings/{0}", user.UserId);  
            var response = await client.GetAsync(url);  
            var result = response.Content.ReadAsStringAsync().Result;  
            waitActivityIndicator.IsRunning = false;  
  
            if (string.IsNullOrEmpty (result)) {  
                waitActivityIndicator.IsRunning = false;  
                await DisplayAlert("Error", "No response form server", "Acept");  
                return;  
            }  
  
            var myVotings = JsonConvert.DeserializeObject<List<Voting>> (result);  
            myVotingsListView.ItemsSource = myVotings;  
            waitActivityIndicator.IsRunning = false;  
        }  
    }  
}

# MyVotingsCell.cs

using System;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public class MyVotingsCell : ViewCell  
    {  
        public MyVotingsCell()  
        {  
            var descriptionLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
                FontSize = 20,  
                FontAttributes = FontAttributes.Bold,  
            };  
  
            descriptionLabel.SetBinding(Label.TextProperty, new Binding("Description"));  
  
            var remarksLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            remarksLabel.SetBinding(Label.TextProperty, new Binding("Remarks"));  
  
            var dateTimeStartLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            dateTimeStartLabel.SetBinding(Label.TextProperty, new Binding("DateTimeStartEdited"));  
  
            var dateTimeEndLabel = new Label {  
                HorizontalOptions = LayoutOptions.EndAndExpand  
            };  
  
            dateTimeEndLabel.SetBinding(Label.TextProperty, new Binding("DateTimeEndEdited"));  
  
            var line1 = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                Children = {  
                    descriptionLabel,   
                },  
            };  
  
            var line2 = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                Children = {  
                    remarksLabel,    
                },  
            };  
  
            var line3 = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                Children = {  
                    dateTimeStartLabel, dateTimeEndLabel,  
                },  
            };  
  
            View = new StackLayout {  
                Orientation = StackOrientation.Vertical,  
                Children = {   
                    line1, line2, line3,  
                },  
            };  
        }  
    }  
}

# VotingPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.VotingPage"  
             Title="Voting">  
    <ContentPage.Content>  
        <StackLayout Spacing="10">  
            <Label Text="Voting"   
                   FontSize="30"   
                   FontAttributes="Bold"   
                   HorizontalOptions="Center"   
                   VerticalOptions="Center" />  
            <ActivityIndicator x:Name="waitActivityIndicator"  
                               HorizontalOptions="Center"  
                               VerticalOptions="Center"/>  
            <ListView x:Name="candidatesListView" />  
        </StackLayout>  
    </ContentPage.Content>  
</ContentPage>

# VotingPage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
using System.Net.Http;  
  
namespace PreDemocracy  
{  
    public partial class VotingPage : ContentPage  
    {  
        private Voting voting;  
        private UserPassword user;  
  
        public VotingPage (Voting voting, UserPassword user)  
        {  
  
            InitializeComponent ();  
  
            this.voting = voting;  
            this.user = user;  
  
            this.Padding = Device.OnPlatform (  
                new Thickness (10, 20, 10, 10),  
                new Thickness (10),  
                new Thickness (10));  
  
            candidatesListView.ItemTemplate = new DataTemplate(typeof(VotingCell));  
            candidatesListView.RowHeight = 160;  
            candidatesListView.ItemsSource = voting.Candidates;  
            candidatesListView.ItemSelected += CandidatesListView\_ItemSelected;  
        }  
  
        private async void CandidatesListView\_ItemSelected (object sender, SelectedItemChangedEventArgs e)  
        {  
            var candidate = (Candidate)e.SelectedItem;  
            var answer = await DisplayAlert (  
                "Confirm",   
                string.Format ("Are you sure to vote for: {0}", candidate.User.FullName),   
                "Yes",   
                "No");  
  
            if (!answer) {  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = true;  
  
            try {  
                var client = new HttpClient ();  
                client.BaseAddress = new Uri ("http://zulu-software.com");  
                var url = string.Format(  
                    "/Democracy/api/Votings/{0}/{1}/{2}",   
                    this.user.UserId,   
                    candidate.CandidateId,   
                    this.voting.VotingId);  
  
                var response = await client.PostAsync(url, null);  
                if (!response.IsSuccessStatusCode) {  
                    waitActivityIndicator.IsRunning = false;  
                    await DisplayAlert ("Error", "Error trying to vote", "Acept");  
                    return;  
                }  
  
            } catch (Exception ex) {  
                waitActivityIndicator.IsRunning = false;  
                await DisplayAlert ("Error", ex.Message, "Acept");  
                return;  
            }  
  
            waitActivityIndicator.IsRunning = false;  
            await DisplayAlert ("Confirm", "You vote was taken successfully", "Acept");  
            await Navigation.PopAsync ();  
        }  
    }  
}

# VotingCell.cs

using System;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public class VotingCell : ViewCell  
    {  
        public VotingCell()  
        {  
            var photoImage = new Image {  
                HeightRequest = 150,  
                WidthRequest = 150,  
            };  
  
            photoImage.SetBinding(Image.SourceProperty, new Binding("User.PhotoFullPath"));  
  
            var fullNameLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
                FontSize = 20,  
                FontAttributes = FontAttributes.Bold,  
            };  
  
            fullNameLabel.SetBinding(Label.TextProperty, new Binding("User.FullName"));  
  
            var gradeLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            gradeLabel.SetBinding(Label.TextProperty, new Binding("User.GradeEdited"));  
  
            var groupLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            groupLabel.SetBinding(Label.TextProperty, new Binding("User.GroupEdited"));  
  
            var left = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                Children = {  
                    photoImage,   
                },  
            };  
  
            var right = new StackLayout {  
                Orientation = StackOrientation.Vertical,  
                VerticalOptions = LayoutOptions.Center,  
                Children = {  
                    fullNameLabel, gradeLabel, groupLabel,  
                },  
            };  
  
            View = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                VerticalOptions = LayoutOptions.Center,  
                Children = {   
                    left, right,  
                },  
            };  
        }  
    }  
}

# ResultsPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.ResultsPage"  
             Title="Results">  
    <ContentPage.Content>  
        <StackLayout Spacing="10">  
            <Label Text="Results"   
                   FontSize="30"   
                   FontAttributes="Bold"   
                   HorizontalOptions="Center"   
                   VerticalOptions="Center" />  
            <ActivityIndicator x:Name="waitActivityIndicator"  
                               HorizontalOptions="Center"  
                               VerticalOptions="Center"/>  
            <ListView x:Name="resultsListView" />  
        </StackLayout>  
    </ContentPage.Content>  
</ContentPage>

# ResultsPage.xaml.cs

using System;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
using System.Net.Http;  
using Newtonsoft.Json;  
  
namespace PreDemocracy  
{  
    public partial class ResultsPage : ContentPage  
    {  
        public ResultsPage ()  
        {  
            InitializeComponent ();  
  
            this.Padding = Device.OnPlatform (  
                new Thickness (10, 20, 10, 10),  
                new Thickness (10),  
                new Thickness (10));  
  
            resultsListView.ItemTemplate = new DataTemplate(typeof(ResultCell));  
            resultsListView.RowHeight = 160;  
            resultsListView.ItemSelected += ResultsListView\_ItemSelected;  
        }  
  
        private async void ResultsListView\_ItemSelected (object sender, SelectedItemChangedEventArgs e)  
        {  
            await Navigation.PushAsync (new ResultDetailsPage ((Voting)e.SelectedItem));  
        }  
  
        protected override void OnAppearing ()  
        {  
            base.OnAppearing ();  
            this.LoadResults ();  
        }  
  
        private async void LoadResults ()  
        {  
            waitActivityIndicator.IsRunning = true;  
            var client = new HttpClient();  
            client.BaseAddress = new Uri ("http://zulu-software.com");  
            var url = "/Democracy/api/Votings";  
            var response = await client.GetAsync(url);  
            var result = response.Content.ReadAsStringAsync().Result;  
            waitActivityIndicator.IsRunning = false;  
  
            if (string.IsNullOrEmpty (result)) {  
                waitActivityIndicator.IsRunning = false;  
                await DisplayAlert("Error", "No response form server", "Acept");  
                return;  
            }  
  
            var results = JsonConvert.DeserializeObject<List<Voting>> (result);  
            resultsListView.ItemsSource = results;  
            waitActivityIndicator.IsRunning = false;  
        }  
    }  
}

# ResultCell.cs

using System;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public class ResultCell : ViewCell  
    {  
        public ResultCell()  
        {  
            var photoImage = new Image {  
                HeightRequest = 150,  
                WidthRequest = 150,  
            };  
  
            photoImage.SetBinding(Image.SourceProperty, new Binding("Winner.PhotoFullPath"));  
  
            var descriptionLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
                FontSize = 20,  
                FontAttributes = FontAttributes.Bold,  
            };  
  
            descriptionLabel.SetBinding(Label.TextProperty, new Binding("Description"));  
  
            var remarksLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
            };  
  
            remarksLabel.SetBinding(Label.TextProperty, new Binding("Remarks"));  
  
            var winnerLabel = new Label {  
                Text = "Winner",  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
            };  
  
            var fullNameLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
                FontSize = 20,  
                FontAttributes = FontAttributes.Bold,  
            };  
  
            fullNameLabel.SetBinding(Label.TextProperty, new Binding("Winner.FullName"));  
  
            var gradeLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            gradeLabel.SetBinding(Label.TextProperty, new Binding("Winner.GradeEdited"));  
  
            var groupLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand  
            };  
  
            groupLabel.SetBinding(Label.TextProperty, new Binding("Winner.GroupEdited"));  
  
            var left = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                Children = {  
                    photoImage,   
                },  
            };  
  
            var right = new StackLayout {  
                Orientation = StackOrientation.Vertical,  
                VerticalOptions = LayoutOptions.Center,  
                Children = {  
                    descriptionLabel, remarksLabel, winnerLabel, fullNameLabel, gradeLabel, groupLabel,  
                },  
            };  
  
            View = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                VerticalOptions = LayoutOptions.Center,  
                Children = {   
                    left, right,  
                },  
            };  
        }  
    }  
}

# ResultDetailsPage.xaml

<?xml version="1.0" encoding="UTF-8"?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"   
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"   
             x:Class="PreDemocracy.ResultDetailsPage"  
             Title="Result Details">  
    <ContentPage.Content>  
        <StackLayout Spacing="10">  
            <Label Text="Results Details"   
                   FontSize="30"   
                   FontAttributes="Bold"   
                   HorizontalOptions="Center"   
                   VerticalOptions="Center" />  
            <StackLayout Orientation="Horizontal">  
                <StackLayout Orientation="Vertical">  
                    <Label x:Name="descriptionLabel"  
                           FontSize="20"  
                           FontAttributes="Bold"  
                           HorizontalOptions="StartAndExpand"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="remarksLabel"  
                           HorizontalOptions="StartAndExpand"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="dateTimeStartLabel"  
                           HorizontalOptions="StartAndExpand"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="dateTimeEndLabel"  
                           HorizontalOptions="StartAndExpand"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="quantityVotesLabel"  
                           HorizontalOptions="StartAndExpand"  
                           VerticalOptions="Center"/>  
                    <Label Text="Winner"  
                           FontAttributes="Bold"  
                           HorizontalOptions="Center"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="winnerNameLabel"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="winnerGradeLabel"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                    <Label x:Name="winnerGroupLabel"  
                           HorizontalOptions="Start"  
                           VerticalOptions="Center"/>  
                </StackLayout>  
                <StackLayout Orientation="Vertical">  
                    <Image x:Name="winnerPhotoImage"  
                           HorizontalOptions="End"/>  
                </StackLayout>  
            </StackLayout>  
            <Label Text="Candidates"  
                   FontAttributes="Bold"  
                   HorizontalOptions="Center"  
                   VerticalOptions="Center"/>  
            <ListView x:Name="candidatesListView" />  
        </StackLayout>  
    </ContentPage.Content>  
</ContentPage>

# ResultDetailsPage.xaml.cs

using System;  
using System.Linq;  
using System.Collections.Generic;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public partial class ResultDetailsPage : ContentPage  
    {  
        public ResultDetailsPage (Voting voting)  
        {  
            InitializeComponent ();  
  
            this.Padding = Device.OnPlatform (  
                new Thickness (10, 20, 10, 10),  
                new Thickness (10),  
                new Thickness (10));  
  
            candidatesListView.ItemTemplate = new DataTemplate(typeof(CandidateCell));  
            candidatesListView.RowHeight = 30;  
  
            winnerPhotoImage.Source = voting.Winner.PhotoFullPath;  
            winnerPhotoImage.HeightRequest = 180;  
            winnerPhotoImage.WidthRequest = 180;  
            winnerNameLabel.Text = voting.Winner.FullName;  
            winnerGradeLabel.Text = voting.Winner.GradeEdited;  
            winnerGroupLabel.Text = voting.Winner.GroupEdited;  
            descriptionLabel.Text = voting.Description;  
            remarksLabel.Text = voting.Remarks;  
            dateTimeStartLabel.Text = voting.DateTimeStartEdited;  
            dateTimeEndLabel.Text = voting.DateTimeEndEdited;  
            quantityVotesLabel.Text = voting.QuantityVotesEdited;  
            candidatesListView.ItemsSource = voting.Candidates.OrderByDescending(c => c.QuantityVotes).ToList();  
        }  
    }  
}

# CandidateCell.cs

using System;  
  
using Xamarin.Forms;  
  
namespace PreDemocracy  
{  
    public class CandidateCell  : ViewCell  
    {  
        public CandidateCell()  
        {  
            var candidateNameLabel = new Label {  
                HorizontalOptions = LayoutOptions.StartAndExpand,  
            };  
  
            candidateNameLabel.SetBinding(Label.TextProperty, new Binding("User.FullName"));  
  
            var candidateVotesLabel = new Label {  
                HorizontalOptions = LayoutOptions.End  
            };  
  
            candidateVotesLabel.SetBinding(Label.TextProperty, new Binding("QuantityVotes"));  
  
            View = new StackLayout {  
                Orientation = StackOrientation.Horizontal,  
                VerticalOptions = LayoutOptions.Center,  
                Children = {   
                    candidateNameLabel, candidateVotesLabel,  
                },  
            };  
        }  
    }  
}