# רפלקציה

במהלך תהליך בניית הפרויקט אני מרגיש שלמדתי רבות אודות נושאים שלא הייתי נחשף אליהם. לפני תחילת העבודה על הפרויקט, לא ידעתי דבר או חצי דבר על נושאים רחבים כמו: קשירת מידע, פונקציות אסינכרוניות ושימוש פרקטי בתכנות מונחה עצמים. אני מרגיש שתהליך בניית הפרוייקט תרם לי מאוד בהרחבת הידע שלי בנושאים אלו, ואילץ אותי ללמוד תכניקות תכנות מתקדמות.

אני מרגיש שהדבר הכי חשוב שלמדתי מתהליך כתיבת הפרויקט הוא היכולת ללמוד בעצמי נושאים. היו מקומות רבים בפרויקט שבהם לא ידעתי לפתור בעיה שבה נתקלתי, בזמנים אלו, אני הייתי חייב לחקור את הנושא בעצמי בעזרת שימוש במשאבים בנושא ברשת- במהלך העבודה על הפרויקט למדתי איך ללמוד לבדי נושאים מגוונים בצורה יעילה.

בהסתכלות לאחור, אני יכול לראות בבירור דברים בפרויקט שהייתי עושה אחרת לחלוטין. אם הייתי בונה עכשיו את הפרויקט מהתחלה עד הסוף- עם כל הניסיון שצברתי מאז, הייתי משתמש באפציות ממשק הרבה יותר מתקדמות כדי לבנות ממשת משתמש יותר נוח ואינטואיטיבי. הייתי עושה שימוש נרחב יותר בקשירת מידע (databinding), בפקדים מתקדמים מאינטרנט שמציגים מידע בצורה נוחה יותר (כמו datagrid של uwp community toolkit) ובטכניקות שונות שהיות חוסכות לי עבודה רבה, ולמדתי רק לאחר כתיבת חלק גדול מהעבודה.

חוץ מזה, העבודה על הפרויקט לימדה אותי שעלי לשקול מספר אופציות לאיך ניתן להשיג מטרה מסויימת, ולבחור באופציה המתאימה ביותר. לדוגמא, כאשר יצרתי את המערכת ששומרת מקומית את פרטי ההתחברות של המשתמשים העליונים, שקלתי רק אופציה אחת- מסד נתונים sqllite מקומי, לקח לי זמן רב ללמוד איך ליצור מסד נתונים שכזה ואיך להשתמש בו. כאשר הגעתי לאותה בעיה במערכת ההתחברות של המשתמשים התחתונים, מצאתי דרך הרבה יותר פשוטה לעשות בעזרת שמירת הגדרות מקומית במערכת windows.

# נספחים:

## הקוד לצד לקוח:

### פקד תיבת סימון עם תיבת סיסמה:

הקוד הויזואלי:

<UserControl

    x:Class="storageUniversal.PasswordWithCheck"

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

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

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    d:DesignHeight="300"

    d:DesignWidth="400">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="\*"/>

            <ColumnDefinition Width="0.2\*"/>

        </Grid.ColumnDefinitions>

        <PasswordBox x:Name="pass" Grid.Column="0" PlaceholderText="{x:Bind placeHolder}" Password="{x:Bind Text, Mode=TwoWay}" HorizontalAlignment="Stretch" VerticalAlignment="Stretch"/>

        <CheckBox Grid.Column="1" Checked="CheckBox\_Checked" Unchecked="CheckBox\_Unchecked" ToolTipService.ToolTip="see your password"></CheckBox>

    </Grid>

</UserControl>

הקוד האחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

// The User Control item template is documented at https://go.microsoft.com/fwlink/?LinkId=234236

namespace storageUniversal

{

    public sealed partial class PasswordWithCheck : UserControl

    {

        public PasswordWithCheck()

        {

            this.InitializeComponent();

        }

        public string Text

        {

            get => (string)GetValue(TextProperty);

            set => SetValue(TextProperty, value);

        }

        public string placeHolder

        {

            get => (string)GetValue(PlaceHolderProp);

            set => SetValue(PlaceHolderProp, value);

        }

        public static readonly DependencyProperty TextProperty =

          DependencyProperty.Register(nameof(Text), typeof(string),

            typeof(PasswordWithCheck), new PropertyMetadata(string.Empty));

        public static readonly DependencyProperty PlaceHolderProp =

  DependencyProperty.Register(nameof(placeHolder), typeof(string),

    typeof(PasswordWithCheck), new PropertyMetadata(string.Empty));

        private void Button\_Click(object sender, RoutedEventArgs e)

        {

            pass.PasswordRevealMode = PasswordRevealMode.Visible;

        }

        private void Pass\_GotFocus(object sender, RoutedEventArgs e)

        {

            pass.PasswordRevealMode = PasswordRevealMode.Visible;

        }

        private void Pass\_LostFocus(object sender, RoutedEventArgs e)

        {

            pass.PasswordRevealMode = PasswordRevealMode.Hidden;

        }

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

        {

            pass.PasswordRevealMode = PasswordRevealMode.Visible;

        }

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

        {

            pass.PasswordRevealMode = PasswordRevealMode.Hidden;

        }

    }

}

### העמוד הראשי:

הקוד הויזואלי:

<Page

    x:Class="storageUniversal.MainPage"

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

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

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Button Grid.Column="0" x:Name="UpUser" Click="UpUser\_Click" VerticalAlignment="Center" HorizontalAlignment="Stretch">I am a Inventory admin</Button>

        <Button Grid.Column="1" x:Name="LowerUser" Click="LowerUser\_Click" VerticalAlignment="Center" HorizontalAlignment="Stretch">I am a user</Button>

        <Button Content="about" Name="about" Click="About\_Click"  HorizontalAlignment="Left" VerticalAlignment="Bottom"/>

    </Grid>

</Page>

הקוד האחורי:

using System;

using System.IO;

using System.Text.RegularExpressions;

using Windows.ApplicationModel.Core;

using Windows.UI.Core;

using Windows.UI.ViewManagement;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using System.Net;

using storageUniversal.xamls;

using Windows.UI.Xaml.Navigation;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=402352&clcid=0x409

namespace storageUniversal

{

    /// <summary>

    /// A basic main page that sends user to login or register

    /// </summary>

    public sealed partial class MainPage : Page

    {

        public static Frame frame;

        public MainPage()

        {

            InitializeComponent();

        }

        override

        protected void OnNavigatedTo(NavigationEventArgs e)

        {

            Frame.IsNavigationStackEnabled = true;

        }

        //sends user to about page

        //שולח את המשתמש לעמוד אודות

        private void About\_Click(object sender, RoutedEventArgs e)

        {

            About.SentBy = typeof(MainPage);

            Frame.Navigate(typeof(About));

        }

        private void UpUser\_Click(object sender, RoutedEventArgs e)

        {

            Frame.Navigate(typeof(UpperLogin));

        }

        private void LowerUser\_Click(object sender, RoutedEventArgs e)

        {

            frame = Frame;

            Frame.Navigate(typeof(LowerLogin));

        }

    }

}

### עמוד אודות

קוד ויזואלי:

<Page

    x:Class="storageUniversal.xamls.About"

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

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

    xmlns:local="using:storageUniversal.xamls"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Name="back" Click="Back\_Click" VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <!--for desine flexability, I used HTML + WebView-->

        <WebView Grid.Column="1" Grid.Row="1" Name="webview1" Source="ms-appx-web:///about.html" HorizontalAlignment="Stretch" VerticalAlignment="Stretch"/>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using storageUniversal;

using Windows.Storage;

using Windows.UI.Core;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal.xamls

{

    /// <summary>

    /// An empty page that can be used on its own or navigated to within a Frame.

    /// </summary>

    public sealed partial class About : Page

    {

        public static Type SentBy;

        public About()

        {

            this.InitializeComponent();

            //sets so links would be opened in browser instad of localy in the app

            // מגדיר את ההצגה כך שקישורים חיצוניים יפתחו בדפדפן ולא בתוך הפקד

            webview1.NavigationStarting += async (webViewSender, args) =>

            {

                // Cancel the navigation

                args.Cancel = true;

                // Get the URI of the link that was clicked

                var uri = args.Uri;

                // Open the link in the default external browser

                var success = await Windows.System.Launcher.LaunchUriAsync(uri);

                if (success)

                {

                    // The link was opened successfully

                }

                else

                {

                    // An error occurred, the link could not be opened

                }

            };

            // some code to handle mouse back + forward buttons

            // קוד להוספת פונקציית קדימה ואחורה לכפתורי העכבר

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        // קוד להוספת פונקציית קדימה ואחורה לכפתורי העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //Back button

        //קוד לכפתור אחורה- שולח את המשתמש לעמוד הקודם

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

    }

}

קוד html לעמוד אודות

כיוון שעמוד זה נכתב על ידי תוכנת Word, הקוד שלו כולל מעל 800 שורות, ולכן לא אצרף אותו.

### עמוד התחברות: משתמש עליון

קוד ויזואלי

<Page

    x:Class="storageUniversal.UpperLogin"

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

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

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <!--made by yehonatan vishna-->

    <Grid Name="g1" ColumnSpacing="25">

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Grid.Row="0" Grid.Column="0" Name="back" Click="Back\_Click" VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <Button Grid.Row="1" Grid.Column="0" x:Name="RegBut" Click="RegBut\_Click" VerticalAlignment="Bottom" HorizontalAlignment="Left" ToolTipService.ToolTip="register">

            <SymbolIcon Symbol="OtherUser"></SymbolIcon></Button>

        <Grid Grid.Column="0" Grid.Row="1" Grid.ColumnSpan="2" ColumnSpacing="20">

            <Grid.ColumnDefinitions>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

            </Grid.ColumnDefinitions>

            <Grid Grid.Column="0">

                <Grid.RowDefinitions>

                    <RowDefinition></RowDefinition>

                    <RowDefinition></RowDefinition>

                    <RowDefinition></RowDefinition>

                </Grid.RowDefinitions>

                <TextBox Name="email" Grid.Row="1" TextWrapping="Wrap" HorizontalAlignment="Stretch"  PlaceholderText="email" VerticalAlignment="Center" />

                <TextBlock Name="IsEmailValidBlock" Grid.Row="0" TextWrapping="Wrap" HorizontalAlignment="Stretch" VerticalAlignment="Bottom" IsColorFontEnabled="True" Foreground="Red" Text=""/>

            </Grid>

            <PasswordBox  Name="password" Grid.Column="1" HorizontalAlignment="Stretch" VerticalAlignment="Center"  PlaceholderText="password" />

            <TextBlock Visibility="Collapsed" Name="res"  Text="" TextWrapping="Wrap" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

            <Button  Grid.Column="2" Name="sendToInventoryTbl" ToolTipService.ToolTip="this button sends you to see your inventory table" Click="LoginAndSendToHome" HorizontalAlignment="Stretch" VerticalAlignment="Center">

                <FontIcon FontFamily="{StaticResource SymbolThemeFontFamily}" Glyph="&#xE8AD;" />

            </Button>

            <CheckBox Grid.Column="3" Name="rememberBox" Content="remenber for next time" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

        </Grid>

        <Grid Grid.Column="1" Grid.Row="1" HorizontalAlignment="Right" VerticalAlignment="Bottom" >

            <Grid.ColumnDefinitions>

                <ColumnDefinition Width="4\*"/>

                <ColumnDefinition Width="0.25\*"/>

            </Grid.ColumnDefinitions>

            <Button Grid.Column="0" x:Name="logout" Click="Logout\_Click">

                log out

            </Button>

            <CheckBox Grid.Column="1" x:Name="forget\_saved\_users" ToolTipService.ToolTip="forget all saved users"></CheckBox>

        </Grid>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Net.Mail;

using System.Runtime.InteropServices.WindowsRuntime;

using System.Windows.Input;

using Windows.ApplicationModel.Core;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Core;

using Windows.UI.ViewManagement;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using storageUniversal.codes;

using System.Threading.Tasks;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal

{

    /// <summary>

    /// The page gets the users email and password and signs him in

    /// </summary>

    public sealed partial class UpperLogin : Page

    {

        public static UserDBServ.User usr = new UserDBServ.User();

        public static UserDBServ.User FullUser;

        private UserDBServ.UserDBServSoapClient UDBS = new UserDBServ.UserDBServSoapClient();

        public static Type SentFrom;

        //שולח את המשתמש לעמוד הבית אם הוא כבר התחבר

        public async void gotoHome()

        {

            await Task.Delay(1);

            Frame.Navigate(typeof(xamls.UpperUserHomePage));

        }

        public UpperLogin()

        {

            this.InitializeComponent();

            // some code to handle mouse back + forward buttons

            // קוד להוספת פונקציית קדימה ואחורה לכפתורי העכבר

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

            var frame = Window.Current.Content as Frame;

            var sender = frame.SourcePageType;

            if(sender == typeof(MainPage))

            {

                if(FullUser != null)

                {

                    gotoHome();

                }

                else

                {

                    TryStartAutoLoging();

                }

            }

        }

        // some code to handle mouse back + forward buttons

        // קוד להוספת פונקציית קדימה ואחורה לכפתורי העכבר

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //checking if the user has previusly signed in, if he did, offering him to use previus login info

        //בודק האם המשתמש כבר נכנס בעבר ממחשב זה,אם כן מציע לו להשתמש בחיבור לפי הפרטים שנשמרו

        private async void TryStartAutoLoging()

        {

            var db = new UsersDatabase();

            var a = await db.GetItemsAsync();

            if (a.Count > 0)

            {

                //in case there is alredy content dialog up, using try to avoid erro

                try

                {

                    //הודעת ההצעה

                    ContentDialog getLandingDits = new ContentDialog()

                    {

                        Title = "do you wish to use saved user?",

                        SecondaryButtonText = "yes",

                        SecondaryButtonCommand = new logSavedUser(),

                        SecondaryButtonCommandParameter = this,

                        CloseButtonText = "no"

                    };

                    await getLandingDits.ShowAsync();

                }

                catch { }

            }

        }

        //tryes to log user in

        // מנסה לחבר את המשתמש אוטומטית, כך שאם לא מצליח לא מעלה שגיאה

        public static async void logAutoCheck(Frame frame, TextBlock res)

        {

            var db = new UsersDatabase();

            var a = await db.GetItemsAsync();

            try

            {

                    autoLogin(a[0], frame, res);

            }

            catch { }

        }

        //logs in user autologin

        // מכניס את המשתמש בכניסה אוטומטית לפי הפרטים השמורים

        public static async void autoLogin(User user, Frame frame, TextBlock res)

        {

            var UDBS = new UserDBServ.UserDBServSoapClient();

            usr.Password = user.Password;

            usr.Email = user.Email;

            var a = await UDBS.IsUserPermittedAsync(new UserDBServ.User() {Password= usr.Password, Email = usr.Email });

            bool b = bool.Parse(a.ToString());

            if (b)

            {

                var TempFullUsr = await UDBS.GetFullUserAsync(usr);

                FullUser = TempFullUsr;

                frame.Navigate(typeof(xamls.UpperUserHomePage));

            }

            else

            {

                var fail = new ContentDialog() { Title = "email or password are wrong, try again", CloseButtonText = "ok" };

                await fail.ShowAsync();

            }

        }

        //the command that passes user data to next page from auto login menu

        //הקומנד שיפעל ברגע שהמשתמש יבחר שהוא רוצה להתחבר עם הפרטים השמורים

        public class logSavedUser : ICommand

        {

            public event EventHandler CanExecuteChanged;

            public bool CanExecute(object parameter)

            {

                throw new NotImplementedException();

            }

            public async void Execute(object parameter)

            {

                var logPg = (parameter as UpperLogin);

                logAutoCheck(logPg.Frame, logPg.res);

            }

        }

        //logs in user and sends him to his home page

        // מחבר את המשתמש, ושולח אותו לדף הבית

        private async void LoginAndSendToHome(object sender, RoutedEventArgs e)

        {

            usr.Password = password.Password;

            usr.Email = email.Text;

            if (!isEmailValid(usr.Email))

            {

                IsEmailValidBlock.Text = "email address isn't valid";

                return;

            }

            var a = await UDBS.IsUserPermittedAsync(usr);

            if(await UDBS.IsAdminAsync(usr))

            {

                FullUser = new UserDBServ.User() { Email = usr.Email, Password = usr.Password };

                Frame.Navigate(typeof(AdminPanel));

                return;

            }

            var isval = isEmailValid(usr.Email);

            bool b = bool.Parse(a.ToString());

            if (b)

            {

                //raises sucsess popup

                //מעלה הודעת הצלחה

                var sucsess = new ContentDialog() { Title = "user exists, data should be showen", CloseButtonText="ok" };

                await sucsess.ShowAsync();

                var TempFullUsr = await UDBS.GetFullUserAsync(usr);

                FullUser = TempFullUsr;

                if (rememberBox.IsChecked.Value)

                {

                    var db = new UsersDatabase();

                    await db.DeleteAll();

                    await db.InsertItemAsync(new User() { ID = FullUser.ID, Email = FullUser.Email, Password = FullUser.Password , BDate = FullUser.BDate.Value, Compeny = FullUser.Compeny, Fname = FullUser.Fname, Lname = FullUser.Lname});

                }

                this.Frame.Navigate(typeof(xamls.UpperUserHomePage));

            }

            else

            {

                //raises failiur popup

                //מעלה הודעת שגיאה

                var fail = new ContentDialog() { Title = "email or password are wrong, try again", CloseButtonText = "ok" };

                await fail.ShowAsync();

            }

        }

        //בודק האם האימייל שהתקבל תקין בתחביר שלו

        //checks whether the recived email is valid

        public bool isEmailValid(String emailaddress)

        {

            try

            {

                MailAddress m = new MailAddress(emailaddress);

                foreach(char a in emailaddress.ToList())

                {

                    if (a < 32 || a>122)

                    {

                        return false;

                    }

                }

                return true;

            }

            catch

            {

                return false;

            }

        }

        //send user back to previus page

        //כפתור אחורה- שולח את המשתמש לעמוד הקודם

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

        private void RegBut\_Click(object sender, RoutedEventArgs e)

        {

            Frame.Navigate(typeof(Register));

        }

        //מנתק את המשתמש על ידי איפוס משנה המשתמש

        private void Logout\_Click(object sender, RoutedEventArgs e)

        {

            FullUser = null;

            if (forget\_saved\_users.IsChecked.GetValueOrDefault())

            {

                var db = new UsersDatabase();

                db.DeleteAll();

            }

        }

    }

}

### עמוד הרשמה- משתמש עליון

קוד ויזואלי:

<Page

    x:Class="storageUniversal.Register"

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

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

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid ColumnSpacing="40">

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Name="back" Grid.Column="0" Grid.Row="0" Click="Back\_Click" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <TextBox Name="FN" Grid.Column="1" Grid.Row="1" TextWrapping="Wrap" HorizontalAlignment="Stretch" PlaceholderText="first name" VerticalAlignment="Center" Width="Auto"/>

        <TextBox Name="LN" Grid.Column="2" Grid.Row="1" TextWrapping="Wrap" HorizontalAlignment="Stretch" PlaceholderText="Last Name" VerticalAlignment="Center"/>

        <CalendarDatePicker Name="BDate" Grid.Column="3" Grid.Row="1"  PlaceholderText="enter your birth date"  HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

        <TextBox Name="compeny" Grid.Column="4" Grid.Row="1" TextWrapping="Wrap" HorizontalAlignment="Stretch" PlaceholderText="compeny" VerticalAlignment="Center"/>

        <Grid Grid.Column="5" Grid.Row="1">

            <Grid.RowDefinitions>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

            </Grid.RowDefinitions>

            <TextBox Name="email" Grid.Row="1" TextWrapping="Wrap" HorizontalAlignment="Stretch" VerticalAlignment="Center" PlaceholderText="email"/>

            <!--a place for invalid password warning-->

            <TextBlock Name="IsEmailValidBlock" Grid.Row="0" TextWrapping="Wrap" HorizontalAlignment="Stretch" VerticalAlignment="Bottom" IsColorFontEnabled="True" Foreground="Red" Text=""/>

        </Grid>

        <Grid Grid.Column="6" Grid.Row="1">

            <Grid.RowDefinitions>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

            </Grid.RowDefinitions>

            <PasswordBox Name="pass" Grid.Row="1" PlaceholderText="password" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

            <PasswordBox Name="passAgain" Grid.Row="2" PlaceholderText="enter password again" HorizontalAlignment="Stretch" VerticalAlignment="Top"/>

            <!--a place for invalid password warning-->

            <TextBlock Name="IsPasswordValidBlock" Grid.Row="0" TextWrapping="Wrap" HorizontalAlignment="Stretch" VerticalAlignment="Bottom" IsColorFontEnabled="True" Foreground="Red" Text=""/>

        </Grid>

        <Button Name="regBot" Grid.Column="7" Grid.Row="1" Click="RegBot\_Click" Content="register" HorizontalAlignment="Stretch" VerticalAlignment="Center" />

        <TextBlock Visibility="Collapsed" Name ="isDone"  HorizontalAlignment="Left" Margin="263,341,0,0" Text="" VerticalAlignment="Top"/>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using System.Threading;

using System.Threading.Tasks;

using Windows.UI.Core;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal

{

    /// <summary>

    /// this page allows the user to register (add his ditails db)

    /// </summary>

    public sealed partial class Register : Page

    {

        public static Type SentFrom;

        public Register()

        {

            this.InitializeComponent();

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            // some code to handle mouse back + forward buttons

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //בודק האם האימייל שהתקבל תקין בתחביר שלו

        //checks whether the recived email is valid

        public bool isEmailValid(String email)

        {

            var trimmedEmail = email.Trim();

            if (trimmedEmail.EndsWith("."))

            {

                return false;

            }

            try

            {

                var addr = new System.Net.Mail.MailAddress(email);

                return addr.Address == trimmedEmail;

            }

            catch

            {

                return false;

            }

        }

        // בודק האם האימייל שהתקבל כבר בשימוש במסד הנתונים

        //checks whether the email alredy axists in db

        public async Task<bool> DoesEmailExistAlready(string email)

        {

            var UDBS = new UserDBServ.UserDBServSoapClient();

            return await UDBS.DoesEmailExistAsync(email);

        }

        //מגיב ללחיצה על כפתור ההרשמה, רושם את המשתמש לשירות

        //responds to click on register button, registers user

        private async void RegBot\_Click(object sender, RoutedEventArgs e)

        {

            UserDBServ.UserDBServSoapClient UDBS = new UserDBServ.UserDBServSoapClient();

            UserDBServ.User usr = new UserDBServ.User();

            //בודק האם האימיל והססמה תקינים ולא קיימים במערכת

            if (!isEmailValid(email.Text)) {

                IsEmailValidBlock.Text = "email is not valid";

                return;

            }

            bool exists = await DoesEmailExistAlready(email.Text);

            if (exists)

            {

                IsEmailValidBlock.Text = "user with this email already exists, use another email adress"; return;

            }

            if (!pass.Password.Equals(passAgain.Password))

            {

                IsPasswordValidBlock.Text = "passwords don't match, try again"; return;

            }

            var date = BDate.Date;

            if (date != null)

            {

                DateTime time = date.Value.DateTime;

                usr.BDate = time;

                //בודק שהמשתמש לא צעיר מידי

                var age = DateTime.Now - time;

                if(age.TotalDays < ((int)(365.25 \* 10)))

                {

                    ContentDialog AgeDialog = new ContentDialog()

                    {

                        Title = "you must be 10 or older to use this service",

                        CloseButtonText = "ok"

                    };

                    await AgeDialog.ShowAsync();

                    return;

                }

            }

            usr.Fname = FN.Text;

            usr.Lname = LN.Text;

            usr.Email = email.Text;

            usr.Password = pass.Password;

            usr.Compeny = compeny.Text;

            //בודק האם כל השדות מלאים

            if(!date.HasValue || usr.Fname == ""|| usr.Lname == ""|| usr.Email == ""|| usr.Password == ""|| usr.Compeny == "") {

                ContentDialog fail = new ContentDialog()

                {

                    Title = "Make sure you fill out all the fileds",

                    CloseButtonText = "ok"

                };

                await fail.ShowAsync();

                return; }

            var a = await UDBS.regAsync(usr);

            bool IsSuccess = bool.Parse(a.ToString());

            //מעלה למשתמש הודעה- האם ההרשמה הסתיימה בהצלחה או לא

            // raises a popup for the user- tells him wether the regestation was completed sucsessfully

            if (IsSuccess)

            {

                ContentDialog sucsessDialog = new ContentDialog()

                {

                    Title = "regestration completed sucssesfully",

                    CloseButtonText = "ok"

                };

                await sucsessDialog.ShowAsync();

                Frame.Navigate(typeof(MainPage));

            }

            else

            {

                ContentDialog FailDialog = new ContentDialog()

                {

                    Title = "regestration failed, please try again",

                    CloseButtonText = "ok"

                };

                await FailDialog.ShowAsync();

            }

        }

        // מגיב ללחיצה על כפתור החזרה, מחזיר את המשתמש למסך הקודם

        //go back to previus page

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

    }

}

### עמוד בית- משתמש עליון:

קוד ויזואלי:

<Page

    x:Class="storageUniversal.xamls.UpperUserHomePage"

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

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

    xmlns:local="using:storageUniversal.xamls"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid ColumnSpacing="50">

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button x:Name="back" Grid.Column="0" Grid.Row="0" Click="Back\_Click" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <TextBlock x:Name="helloMsg" Grid.ColumnSpan="2" Grid.Column="1" Grid.Row="0" Text="{x:Bind UseName, Mode=TwoWay}"></TextBlock>

        <Button  Grid.Column="1" Grid.Row="1" Name="sendToInventoryTbl" ToolTipService.ToolTip="this button sends you to see your inventory table" Click="LoginAndSendToInventoryList" HorizontalAlignment="Stretch" VerticalAlignment="Center">

            <SymbolIcon Symbol="List"/>

        </Button>

        <Button x:Name="ManegeSubUsers" Grid.Column="2" Grid.Row="1" Click="ManegeSubUsers\_Click" HorizontalAlignment="Stretch">manege you users</Button>

        <Button x:Name="GoToChangeDitailes" Grid.Column="3" Grid.Row="1" Click="GoToChangeDitailes\_Click" HorizontalAlignment="Stretch">change your personal ditailes</Button>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Core;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal.xamls

{

    /// <summary>

    /// An empty page that can be used on its own or navigated to within a Frame.

    /// </summary>

    public sealed partial class UpperUserHomePage : Page

    {

        private string UseName = "Hello "+ UpperLogin.FullUser.Fname + UpperLogin.FullUser.Lname;

        public UpperUserHomePage()

        {

            this.InitializeComponent();

            // some code to handle mouse back + forward buttons

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //logs in user and sends him to his invntory list

        // מחבר את המשתמש, ושולח אותו לרשימת המלאי שלו

        private void LoginAndSendToInventoryList(object sender, RoutedEventArgs e)

        {

            var usr = UpperLogin.FullUser;

            this.Frame.Navigate(typeof(InventoryView));

        }

        //the button sends the user to manege his subusers

        //הכפטור שולח את המשתמש לנהל את המשתמשים שמשוייכים אליו

        private void ManegeSubUsers\_Click(object sender, RoutedEventArgs e)

        {

            Frame.Navigate(typeof(SubUsersManegment));

        }

        //the button sends the user to edit his ditailes

        //הכפטור שולח את המשתמש לערוך את פרטיו האישיים

        private void GoToChangeDitailes\_Click(object sender, RoutedEventArgs e)

        {

            Frame.Navigate(typeof(updateUser));

        }

        //מחזיר את המשתמש לעמוד הקודם

        //goes back to previus page

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

    }

}

### עמוד שינוי פרטים- משתמש עליון:

קוד ויזואלי:

<Page

    x:Class="storageUniversal.updateUser"

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

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

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Name="back" Grid.Column="0" Grid.Row="0" Click="Back\_Click" VerticalAlignment="Stretch" HorizontalAlignment="Stretch" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <Grid Grid.Column="0" Grid.Row="1" Grid.ColumnSpan="2">

            <Grid.ColumnDefinitions>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

                <ColumnDefinition></ColumnDefinition>

            </Grid.ColumnDefinitions>

            <TextBox Name="FN" Grid.Column="0" PlaceholderText="first name" VerticalAlignment="Center" HorizontalAlignment="Stretch" />

            <TextBox Name="LN" Grid.Column="1" HorizontalAlignment="Stretch" VerticalAlignment="Center"  PlaceholderText="Last Name"/>

            <CalendarDatePicker Name="BDate" Grid.Column="2" PlaceholderText="this is your birthday" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

            <TextBox Name="compeny" Grid.Column="3" PlaceholderText="compeny" VerticalAlignment="Center" HorizontalAlignment="Stretch"/>

            <TextBox Name="newemail" Grid.Column="4" PlaceholderText="email" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

            <local:PasswordWithCheck x:Name="pass" Grid.Column="5" placeHolder="password" HorizontalAlignment="Stretch" VerticalAlignment="Center"/>

            <Button Name="UpdateUsr" Grid.Column="6" Click="Update\_Click"  HorizontalAlignment="Stretch" VerticalAlignment="Center">

                <SymbolIcon Symbol="Accept" HorizontalAlignment="Stretch"></SymbolIcon>

            </Button>

            <Button Grid.Column="7" Name="DelBot" ToolTipService.ToolTip="this button would delete your user from this service" Click="DelBot\_Click"  HorizontalAlignment="Stretch" VerticalAlignment="Center">

                <SymbolIcon Symbol="Delete"></SymbolIcon>

            </Button>

        </Grid>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using storageUniversal;

using System.Threading;

using Windows.UI.Core;

//made by yehonatan vishna

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal

{

    /// <summary>

    /// a page that allows user to update his personal details

    /// </summary>

    public sealed partial class updateUser : Page

    {

        private UserDBServ.UserDBServSoapClient UDBS = new UserDBServ.UserDBServSoapClient();

        public static UserDBServ.User FullUser = UpperLogin.FullUser;

        public static Type SentFrom;

        public updateUser()

        {

            this.InitializeComponent();

            //מכניס לשדות את הנתונים הנוחכיים של המשתמש

            //fills in user current data

            FN.Text = FullUser.Fname;

            LN.Text = FullUser.Lname;

            newemail.Text = FullUser.Email;

            DateTime time = FullUser.BDate.Value;

            BDate.Date = time.Date;

            compeny.Text = FullUser.Compeny;

            pass.Text = FullUser.Password;

            // some code to handle mouse back + forward buttons

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //שולח לשירות הרשת את הנתונים העדכניים של המשתמש

        //sends motified data to web service

        private async void Update\_Click(object sender, RoutedEventArgs e)

        {

            UserDBServ.User NewUser = new UserDBServ.User();

            var date = BDate.Date;

            DateTime time = date.Value.DateTime;

            NewUser.BDate = time;

            NewUser.Fname = FN.Text;

            NewUser.Lname = LN.Text;

            NewUser.Email = newemail.Text;

            NewUser.Password = pass.Text;

            NewUser.Compeny = compeny.Text;

            var resTemp = await UDBS.updateUserAsync(FullUser, NewUser);

            bool res = resTemp;

            if (res)

            {

                var secsussPop = new ContentDialog()

                {

                    Title = "Detailes Update Have Been Sucsussfully Completed", CloseButtonText="ok"

                };

                await secsussPop.ShowAsync();

                FullUser = NewUser;

                Frame.Navigate(typeof(UpperLogin));

            }

            else

            {

                var failPop = new ContentDialog()

                {

                    Title = "Detailes Update Have Been Unsecsussfull", Content = "Please try again.",

                    CloseButtonText = "ok"

                };

                await failPop.ShowAsync();

            }

            }

        //deletes user from system

        //מוחק את המשתמש

        private async void DelBot\_Click(object sender, RoutedEventArgs e)

        {

            var usr = UpperLogin.FullUser;

            var a = await UDBS.DeleteUserAsync(usr);

            bool b = bool.Parse(a.ToString());

        }

        //מחזיר את המשתמש לעמוד הקודם

        //goes back to previus page

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

    }

    }

### עמוד הוספת משתמשים תחתונים:

קוד ויזואלי:

<Page

    x:Class="storageUniversal.xamls.SubUsersManegment"

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

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

    xmlns:local="using:storageUniversal.xamls"

    xmlns:data="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"/>

            <ColumnDefinition/>

        </Grid.ColumnDefinitions>

        <StackPanel Orientation="Vertical">

            <Button  Name="back" Click="Back\_Click"  VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

            <Button Name="addUser" Click="AddUser\_Click" VerticalAlignment="Top" ToolTipService.ToolTip="add user">

                <SymbolIcon Symbol="Add"/>

            </Button>

            <Button  Name="refresh" Click="Refresh\_Click" VerticalAlignment="Top" ToolTipService.ToolTip="refresh">

                <SymbolIcon Symbol="Refresh"></SymbolIcon>

            </Button>

            <Button Name="sync" Click="Sync\_Click" VerticalAlignment="Top" ToolTipService.ToolTip="sync">

                <SymbolIcon Symbol="Sync"></SymbolIcon>

            </Button>

            <Button Name="delete" Click="Delete\_Click" VerticalAlignment="Top" ToolTipService.ToolTip="delete selected row">

                <SymbolIcon Symbol="Delete"></SymbolIcon>

            </Button>

        </StackPanel>

        <ListView Grid.Column="1" Grid.Row="1" Name="SubUsersTbl" HorizontalContentAlignment="Stretch" HorizontalAlignment="Stretch" SelectionMode="Extended">

            <ListView.Header>

                <Grid HorizontalAlignment="Stretch" RightTapped="Grid\_RightTapped">

                    <Grid.ColumnDefinitions>

                        <ColumnDefinition/>

                        <ColumnDefinition/>

                        <ColumnDefinition/>

                        <ColumnDefinition/>

                        <ColumnDefinition/>

                        <ColumnDefinition/>

                    </Grid.ColumnDefinitions>

                    <TextBlock Grid.Column="0" HorizontalTextAlignment="Center">User First Name</TextBlock>

                    <TextBlock Grid.Column="1" HorizontalTextAlignment="Center">User Last Name</TextBlock>

                    <TextBlock Grid.Column="2" HorizontalTextAlignment="Center">Role</TextBlock>

                    <TextBlock Grid.Column="3" HorizontalTextAlignment="Center">Email</TextBlock>

                    <TextBlock Grid.Column="4" HorizontalTextAlignment="Center">UserName</TextBlock>

                    <TextBlock Grid.Column="5" HorizontalTextAlignment="Center">Password</TextBlock>

                </Grid>

            </ListView.Header>

            <ListView.ItemTemplate>

                <DataTemplate  x:Name="dataTemp" x:DataType="data:SubUser" >

                    <Grid HorizontalAlignment="Stretch" RightTapped="Grid\_RightTapped">

                        <Grid.ColumnDefinitions>

                            <ColumnDefinition/>

                            <ColumnDefinition/>

                            <ColumnDefinition/>

                            <ColumnDefinition/>

                            <ColumnDefinition/>

                            <ColumnDefinition/>

                        </Grid.ColumnDefinitions>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="0" Name="FN" Text="{Binding FName, Mode=TwoWay}"></TextBox>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="1" Name="LN" Text="{Binding LName, Mode=TwoWay}"></TextBox>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="2" Name="Role" Text="{Binding Role, Mode=TwoWay}"></TextBox>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="3" Name="Email" Text="{Binding Email, Mode=TwoWay}"></TextBox>

                        <!--<PasswordBox HorizontalAlignment="Stretch" Grid.Column="4" Name="Password" Password="{Binding Role, Mode=TwoWay}" PasswordRevealMode ="Peek" IsPasswordRevealButtonEnabled="True"/>-->

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="4" Name="UserName" Text="{Binding UserName, Mode=TwoWay}"></TextBox>

                        <data:PasswordWithCheck Grid.Column="5" Text="{Binding Password, Mode=TwoWay}"></data:PasswordWithCheck>

                        <TextBlock Visibility="Collapsed" Name="ID" Text="{Binding Id}"></TextBlock>

                    </Grid>

                </DataTemplate>

            </ListView.ItemTemplate>

            <ListView.ItemContainerStyle>

                <Style TargetType="ListViewItem">

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

                    <Setter Property="HorizontalAlignment"  Value="Stretch"></Setter>

                </Style>

            </ListView.ItemContainerStyle>

        </ListView>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Data;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Core;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal.xamls

{

    /// <summary>

    /// An empty page that can be used on its own or navigated to within a Frame.

    /// </summary>

    public sealed partial class SubUsersManegment : Page

    {

        //רשימה המכילה את כל המשתמשים כפי שהיו אחרי העדכון האחרון

        //user list as it is in the last refresh

        private List<SubUser> UsersOriginal;

        //רשימה של כל המשתמשים כפי שהם כעת בטבלה

        //users as they are after the changes the user made

        private ObservableCollection<SubUser> BinedUsersInTbl = new ObservableCollection<SubUser>();

        public SubUsersManegment()

        {

            this.InitializeComponent();

            loudTbl();

        }

        public async void LoadTbl()

        {

            loudTbl();

        }

        //מכניס את כל המשתמשים השייכים למשתמש זה לטבלה

        //loads all users's users from db and inserts them to the table (ListView)

        public async void loudTbl()

        {

            var s = new SubUserServ.SubUsersServSoapClient();

            var TmpUsr = UpperLogin.FullUser;

            var usr = new SubUserServ.User() { ID = TmpUsr.ID, BDate = TmpUsr.BDate, Compeny = TmpUsr.Compeny, Email = TmpUsr.Email, Fname = TmpUsr.Fname, Lname = TmpUsr.Lname, Password = TmpUsr.Password };

            var r = await s.getYourSubUsersAsync(usr);

            List<SubUser> Users = new List<SubUser>();

            foreach (DataRow a in r.Rows)

            {

                SubUser row = new SubUser();

                row.Id = int.Parse(a["ID"].ToString());

                row.FName = a["FName"].ToString();

                row.LName = a["LName"].ToString();

                row.Email = a["Email"].ToString();

                row.BelongsToUpperUser = int.Parse(a["BelongsToUpperUser"].ToString());

                row.Role = a["Role"].ToString();

                row.Password = a["Password"].ToString();

                row.UserName = a["UserName"].ToString();

                Users.Add(row);

            }

            if (BinedUsersInTbl.ToList().Count > 0)

            {

                BinedUsersInTbl.Clear();

            }

            foreach (SubUser row in Users)

            {

                BinedUsersInTbl.Add(row);

            }

            SubUsersTbl.ItemsSource = BinedUsersInTbl;

            UsersOriginal = new List<SubUser>();

            foreach (SubUser Row in Users)

            {

                UsersOriginal.Add(Row.Copy());

            }

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            // some code to handle mouse back + forward buttons

            var win = new codes.backMouse();

        }

        //send user back to previus page

        //כפתור אחורה- שולח את המשתמש לעמוד הקודם

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

        private void AddUser\_Click(object sender, RoutedEventArgs e)

        {

            addUserF();

        }

        //פונקציה שמוסיפה משתמש ריק למסד הנתונים ולטבלה

        private async void addUserF()

        {

            var sub = new SubUserServ.SubUsersServSoapClient();

            var usr = new SubUserServ.SubUser() { BelongsToUpperUser = UpperLogin.FullUser.ID };

            var getNewDetails = new StackPanel() { Orientation = Orientation.Vertical, DataContext=usr};

            //מוסיף שדה שם פרטי להודעה

            Binding FirstNameBinding = new Binding();

            var FNBox = new TextBox() { PlaceholderText = "The new user's first name." };

            FirstNameBinding.Source = usr.FName;

            FirstNameBinding.Path = new PropertyPath("FName");

            FirstNameBinding.Mode = BindingMode.TwoWay;

            FirstNameBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(FNBox, TextBox.TextProperty, FirstNameBinding);

            getNewDetails.Children.Add(FNBox);

            //מוסיף שדה שם פרטי להודעה

            Binding LastNameBinding = new Binding();

            var LNBox = new TextBox() { PlaceholderText = "The new user's last name." };

            LastNameBinding.Source = usr.LName;

            LastNameBinding.Path = new PropertyPath("LName");

            LastNameBinding.Mode = BindingMode.TwoWay;

            LastNameBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(LNBox, TextBox.TextProperty, LastNameBinding);

            getNewDetails.Children.Add(LNBox);

            //מוסיף שדה תפקיד להודעה

            Binding RoleBinding = new Binding();

            var RoleBox = new TextBox() { PlaceholderText = "The new user's role." };

            RoleBinding.Source = usr.Role;

            RoleBinding.Path = new PropertyPath("Role");

            RoleBinding.Mode = BindingMode.TwoWay;

            RoleBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(RoleBox, TextBox.TextProperty, RoleBinding);

            getNewDetails.Children.Add(RoleBox);

            //מוסיף שדה שם משתמש להודעה

            Binding UserNameBinding = new Binding();

            var UserName = new TextBox() {PlaceholderText="The new user's username."};

            UserNameBinding.Source = usr.UserName;

            UserNameBinding.Path = new PropertyPath("UserName");

            UserNameBinding.Mode = BindingMode.TwoWay;

            UserNameBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(UserName, TextBox.TextProperty, UserNameBinding);

            getNewDetails.Children.Add(UserName);

            //מוסיף שדה ססמה להודעה

            Binding PasswordBinding = new Binding();

            var PassBox = new PasswordBox() { PlaceholderText = "The new user's password." };

            PasswordBinding.Source = usr.Password;

            PasswordBinding.Path = new PropertyPath("Password");

            PasswordBinding.Mode = BindingMode.TwoWay;

            PasswordBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(PassBox, PasswordBox.PasswordProperty, PasswordBinding);

            getNewDetails.Children.Add(PassBox);

            //מוסיף שדה דואל להודעה

            Binding EmailBinding = new Binding();

            var EmailBox = new TextBox() { PlaceholderText = "The new user's email." };

            EmailBinding.Source = usr.Email;

            EmailBinding.Path = new PropertyPath("Email");

            EmailBinding.Mode = BindingMode.TwoWay;

            EmailBinding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;

            BindingOperations.SetBinding(EmailBox, TextBox.TextProperty, EmailBinding);

            getNewDetails.Children.Add(EmailBox);

            //מציג את ההודעה עם הבקשה לנתינת הפרטים

            var ask = new ContentDialog() { Title = "write the new user name", Content=getNewDetails, CloseButtonText = "ok" };

            await ask.ShowAsync();

            var id = await sub.createSubUserAsync(usr, new SubUserServ.User() { Email = UpperLogin.FullUser.Email, Password = UpperLogin.FullUser.Password });

            var localUsr = new SubUser() { BelongsToUpperUser = UpperLogin.FullUser.ID, Id = id, UserName=usr.UserName, Email=usr.Email, FName = usr.FName, LName = usr.LName, Password=usr.Password, Role= usr.Role };

            BinedUsersInTbl.Add(localUsr);

            UsersOriginal.Add(localUsr.Copy());

        }

        private void Refresh\_Click(object sender, RoutedEventArgs e)

        {

            loudTbl();

        }

        //מתבצע כאשר המשתמש מסנכרן את המשתמשים בטבלה עם שרות הרשת

        private async void Sync\_Click(object sender, RoutedEventArgs e)

        {

            var s = new SubUserServ.SubUsersServSoapClient();

            for(int i=0; i< BinedUsersInTbl.Count; i++)

            {

                if (!BinedUsersInTbl.ToList()[i].IsSame(UsersOriginal[i]))

                {

                    try

                    {

                        bool worked = await s.updateSubAsync(conv(BinedUsersInTbl[i]), new SubUserServ.User() { Password = UpperLogin.FullUser.Password, Email = UpperLogin.FullUser.Email, ID = UpperLogin.FullUser.ID });

                    }

                    catch(Exception exeption)

                    {

                        string msg = exeption.Message;

                        if (exeption.Message.Contains("the user"))

                        {

                            msg = msg.Remove(0, 121);

                            int count = 215;

                            msg = msg.Remove(msg.Length-count, count);

                        }

                        var TellAboutError = new ContentDialog() { Title = "the operation couldn't complete sucsussfully", Content = msg, CloseButtonText = "ok" };

                        await TellAboutError.ShowAsync();

                        loudTbl();

                    }

                 }

            }

        }

        //ממיר טיפוס משתמש תחתון מקומי לטיפוס של שירות רשת

        private SubUserServ.SubUser conv(SubUser a)

        {

            return new SubUserServ.SubUser() { BelongsToUpperUser = a.BelongsToUpperUser, Email = a.Email, FName = a.FName, Id = a.Id, LName = a.LName, Password = a.Password, Role = a.Role, UserName = a.UserName};

        }

        //מעלה אפשרויות טפריט לחיצה ימנית

        private List<SubUser> selectedUsers = new List<SubUser>();

        private void Grid\_RightTapped(object sender, RightTappedRoutedEventArgs e)

        {

            selectedUsers.Clear();

            foreach (ItemIndexRange range in SubUsersTbl.SelectedRanges.ToList())

            {

                for (int i = range.FirstIndex; i <= range.LastIndex; i++)

                {

                    selectedUsers.Add(SubUsersTbl.Items[i] as SubUser);

                }

            }

            if (selectedUsers.Count <= 0)

            {

                SubUser clickedItem = new SubUser();

                int Id = int.Parse(((sender as Grid).Children.Last() as TextBlock).Text);

                foreach (SubUser a in SubUsersTbl.Items)

                {

                    if (a.Id == Id)

                    {

                        clickedItem = a;

                    }

                }

                SubUsersTbl.SelectedItem = clickedItem;

            }

            MenuFlyout rightClick = new MenuFlyout();

            var icon = new SymbolIcon() { Symbol = Symbol.Delete };

            MenuFlyoutItem deleteOption = new MenuFlyoutItem { Icon = icon, Text = "delete", FontFamily = new FontFamily("Segoe MDL2 Assets") };

            deleteOption.Click += DeleteOption\_Click;

            rightClick.Items.Add(deleteOption);

            UIElement b = sender as UIElement;

            b.ContextFlyout = rightClick;

            Point point = new Point(e.GetPosition(b).X, e.GetPosition(b).Y);

            rightClick.ShowAt(b, point);

        }

        private async void DeleteOption\_Click(object sender, RoutedEventArgs e)

        {

            deleteSelected();

        }

        //מוחק את כל הפריטים ברשימת הנבחרים

        private async void deleteSelected()

        {

            Window.Current.CoreWindow.PointerCursor = new Windows.UI.Core.CoreCursor(Windows.UI.Core.CoreCursorType.Wait, 1);

            var UsrServ = new SubUserServ.SubUsersServSoapClient();

            foreach (SubUser br in selectedUsers)

            {

                var bs = conv(br);

                var isok = await UsrServ.DeleteSubUserAsync(new SubUserServ.User() { ID = UpperLogin.FullUser.ID, Email = UpperLogin.FullUser.Email, Password = UpperLogin.FullUser.Password }, bs.Id);

                if (isok)

                {

                    BinedUsersInTbl.Remove(br);

                }

                else

                {

                    var erro = new ContentDialog() { Title = "something went wrong", CloseButtonText = "ok" };

                    await erro.ShowAsync();

                }

            }

            Window.Current.CoreWindow.PointerCursor = new CoreCursor(CoreCursorType.Arrow, 1);

        }

        //לחיצה על כפטור המחיקה תעלה את זה

        //שונה מלחיצה על המחיקה בטפריט הימני

        private void Delete\_Click(object sender, RoutedEventArgs e)

        {

            selectedUsers.Clear();

            foreach (ItemIndexRange range in SubUsersTbl.SelectedRanges.ToList())

            {

                for (int i = range.FirstIndex; i <= range.LastIndex; i++)

                {

                    selectedUsers.Add(SubUsersTbl.Items[i] as SubUser);

                }

            }

            if (selectedUsers.Count <= 0)

            {

                SubUser clickedItem = new SubUser();

                int Id = int.Parse(((sender as Grid).Children.Last() as TextBlock).Text);

                foreach (SubUser a in SubUsersTbl.Items)

                {

                    if (a.Id == Id)

                    {

                        clickedItem = a;

                    }

                }

                SubUsersTbl.SelectedItem = clickedItem;

            }

            deleteSelected();

        }

    }

}

### עמוד ניהול רשימת מלאי

קוד ויזואלי:

<Page

    x:Class="storageUniversal.InventoryView"

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

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

    xmlns:data="using:storageUniversal"

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    xmlns:Windows10FallCreatorsUpdate="http://schemas.microsoft.com/winfx/2006/xaml/presentation?IsApiContractPresent(Windows.Foundation.UniversalApiContract, 5)"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <!--made by yehonatan vishna-->

    <!--a page where user can menege is current inventory-->

    <Grid HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

        <Grid.ColumnDefinitions >

            <ColumnDefinition Width="100"></ColumnDefinition>

            <ColumnDefinition ></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

            <RowDefinition Height="20"></RowDefinition>

        </Grid.RowDefinitions>

        <Button Grid.Column="0" Grid.Row="0" Name="back" Click="Back\_Click" VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <Grid Grid.Column="1" Grid.Row="0" Name="headers" Width="auto">

            <Grid.ColumnDefinitions>

                <ColumnDefinition  Width="2\*"></ColumnDefinition>

                <ColumnDefinition MaxWidth="2000" Width="1\*"></ColumnDefinition>

                <ColumnDefinition MaxWidth="2000" Width="1\*"></ColumnDefinition>

                <ColumnDefinition Width="2\*"></ColumnDefinition>

                <ColumnDefinition MaxWidth="2000" Width="0.5\*"></ColumnDefinition>

            </Grid.ColumnDefinitions>

            <TextBlock Grid.Column="0" HorizontalTextAlignment="Center">name</TextBlock>

            <TextBlock Grid.Column="1" HorizontalTextAlignment="Center">Quntety</TextBlock>

            <TextBlock Grid.Column="2" HorizontalTextAlignment="Center">needed quntety</TextBlock>

            <TextBlock Grid.Column="3" HorizontalTextAlignment="Center">remarks</TextBlock>

            <TextBlock Grid.Column="4" HorizontalTextAlignment="Center">amount out</TextBlock>

        </Grid>

        <ListView Grid.Column="1" Grid.Row="1" Name="InventoryTbl" CanDragItems="True" RightTapped="Gridy\_RightTapped" HorizontalContentAlignment="Stretch" HorizontalAlignment="Stretch" VerticalAlignment="Top" >

            <ListView.ItemTemplate>

                <DataTemplate  x:Name="dataTemp" x:DataType="data:InventoryRow">

                    <Grid   Name="gridy" VerticalAlignment="Stretch" HorizontalAlignment="Stretch">

                        <Grid.ColumnDefinitions>

                            <ColumnDefinition MinWidth="50"  Width="2\*"></ColumnDefinition>

                            <ColumnDefinition MinWidth="50" MaxWidth="2000" Width="1\*"></ColumnDefinition>

                            <ColumnDefinition MinWidth="50" MaxWidth="2000" Width="1\*"></ColumnDefinition>

                            <ColumnDefinition MinWidth="50" Width="2\*"></ColumnDefinition>

                            <ColumnDefinition MinWidth="50" MaxWidth="2000" Width="0.5\*"></ColumnDefinition>

                        </Grid.ColumnDefinitions>

                        <Grid.RowDefinitions>

                            <RowDefinition></RowDefinition>

                        </Grid.RowDefinitions>

                        <TextBox Grid.Row="0" TextWrapping="Wrap" FlowDirection="LeftToRight" TextAlignment="Left" HorizontalAlignment="Stretch" Grid.Column="0" Name="NameBlock" Text= "{x:Bind Name, Mode=TwoWay}"/>

                        <TextBox Grid.Row="0" TextWrapping="Wrap" HorizontalAlignment="Stretch" Grid.Column="1" Name="QuantityBlock" Text= "{x:Bind Quantity, Mode=TwoWay}"/>

                        <TextBox Grid.Row="0" TextWrapping="Wrap" HorizontalAlignment="Stretch" Grid.Column="2" Name="NeededQuantityBlock" Text= "{x:Bind NeededQuantity, Mode=TwoWay}"/>

                        <Frame Grid.Row="0" Grid.Column="4" HorizontalAlignment="Stretch" BorderThickness="2" BorderBrush="Gray">

                            <TextBlock  TextWrapping="Wrap" HorizontalAlignment="Stretch" Name="AmountOutBox" Text="{x:Bind AmountOut, Mode=TwoWay}"/>

                        </Frame>

                        <TextBox  Grid.Row="0" TextWrapping="Wrap" HorizontalContentAlignment="Right"  Grid.Column="3" FlowDirection="RightToLeft" TextAlignment="Justify" Name="Remarkes" Text="{x:Bind Remarkes, Mode=TwoWay}"/>

                    </Grid>

                </DataTemplate>

            </ListView.ItemTemplate>

            <ListView.ItemContainerStyle>

                <Style TargetType="ListViewItem">

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

                    <Setter Property="HorizontalAlignment"  Value="Stretch"></Setter>

                </Style>

            </ListView.ItemContainerStyle>

        </ListView>

        <Grid Grid.Column="0" Grid.Row="1">

            <Grid.RowDefinitions>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

                <RowDefinition></RowDefinition>

            </Grid.RowDefinitions>

            <Button Grid.Row="0" Name="LoadTbl" Click="LoadTbl\_Click" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

                <!--<TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="load table"></TextBlock>-->

                <SymbolIcon Symbol="Refresh"></SymbolIcon>

            </Button>

            <Button Grid.Row="1" Name="updateDataFromTbl" Click="UpdateDataFromTbl\_Click" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

                <!--<TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="update Data"></TextBlock>-->

                <SymbolIcon Symbol="Save"></SymbolIcon>

            </Button>

            <Button Grid.Row="2" Name="addItem" Click="AddItem\_Click" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

                <!--<TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="add item"></TextBlock>-->

                <SymbolIcon Symbol="Add"></SymbolIcon>

            </Button>

            <Button Grid.Row="3" Name="Delete" Click="Delete\_Click" VerticalAlignment="Stretch" HorizontalAlignment="Stretch">

                <!--<TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="delete selected item"></TextBlock>-->

                <SymbolIcon Symbol="Delete"></SymbolIcon>

            </Button>

            <Button Grid.Row="4" Name="CsvImport" Click="CsvImport\_Click"  HorizontalAlignment="Stretch" VerticalAlignment="Stretch" ToolTipService.ToolTip="import your existing inventory from csv file">

                <SymbolIcon Symbol="Import"></SymbolIcon>

                <!--<TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="import from csv"></TextBlock>-->

            </Button>

            <Button Grid.Row="5" Name="seeLandings" Click="SeeLandings\_Click"  HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

                <TextBlock HorizontalAlignment="Stretch" VerticalAlignment="Stretch" TextWrapping="WrapWholeWords" Text="see all active landings"></TextBlock>

            </Button>

        </Grid>

    </Grid>

</Page>

קוד אחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using System.Data;

using System.Xml;

using storageUniversal;

using System.Text.RegularExpressions;

using System.Windows.Input;

using System.Collections.ObjectModel;

using Windows.UI.Core;

using Google.Apis.Services;

using Google.Apis.Discovery.v1;

using Google.Apis.Discovery.v1.Data;

using Google.Apis.Services;

using System.Threading.Tasks;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal

{

    /// <summary>

    /// a page where user can menege is current inventory

    /// </summary>

    public sealed partial class InventoryView : Page

    {

        public List<InventoryRow> InventoryBeforeChange;// original inventory

        public ObservableCollection<InventoryRow> InventoryRowesBindedToUser = new ObservableCollection<InventoryRow>();

        public static UserDBServ.User FullUser = UpperLogin.FullUser;//the user that activly uses the page;

        public static Type SentFrom;//the page from which the user were sent

        public static InventoryRow LantItem;

        public InventoryView()

        {

            this.InitializeComponent();

            LoadTblFunc();

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            // some code to handle mouse back + forward buttons

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //מבצע חיפוש בגוגל במנוע חיפוש מותאם אישית עם מגוון אתרי קמעונעות

        private async Task<Google.Apis.CustomSearchAPI.v1.Data.Search> googleApi(string qury)

        {

            string apiKey = "AIzaSyAs4wufxBxLid6quJCN2ynqihPieMxP3qY";

            string cx = "f427caefac6b24f26";

            var svc = new Google.Apis.CustomSearchAPI.v1.CustomSearchAPIService(new BaseClientService.Initializer { ApiKey = apiKey });

            var listRequest = svc.Cse.List();

            listRequest.Q = qury;

            listRequest.Gl = "il";

            listRequest.Cx = cx;

            var search = await listRequest.ExecuteAsync();

            return search;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //מרענן את הטבלה בלחיצה על הכפתור

        //loades the table at a click on the button

        private void LoadTbl\_Click(object sender, RoutedEventArgs e)

        {

            LoadTblFunc();

        }

        //פונקציה שמכניסה לטבלה את המידע העדכני המופיע במסד הנתונים

        //a function that inserts into the table the current data in the db

        private async void LoadTblFunc()

        {

            InventoryServ.InventoryFuncsSoapClient s = new InventoryServ.InventoryFuncsSoapClient();

            var broww = new BorowwDb.BorowwingsDBSoapClient();

            var hasCalcWorked = await broww.UpdateUserAmountOutAsync(FullUser.ID);

            var r = await s.GetInventoryUserDataTableAsync(FullUser.ID, FullUser.Email, FullUser.Password);

            List<InventoryRow> inventoryRows = new List<InventoryRow>();

            foreach (DataRow dr in r.Rows) {

                InventoryRow row = new InventoryRow();

                if (dr["Name"].ToString() != "")

                    row.Name = dr["Name"].ToString();

                row.ID = int.Parse(dr["ID"].ToString());

                if(dr["NeededQuantity"].ToString() != "")

                    row.NeededQuantity = float.Parse(dr["NeededQuantity"].ToString());

                row.OwnerUserId = int.Parse(dr["OwnerUserId"].ToString());

                if (dr["Quantity"].ToString() != "")

                    row.Quantity = float.Parse(dr["Quantity"].ToString());

                if (dr["Remarkes"].ToString() != "")

                    row.Remarkes = dr["Remarkes"].ToString();

                if (dr["AmountOut"].ToString() != "")

                    row.AmountOut = float.Parse(dr["AmountOut"].ToString());

                inventoryRows.Add(row);

            }

            InventoryRowesBindedToUser.Clear();

            foreach (InventoryRow Row in inventoryRows)

            {

                InventoryRowesBindedToUser.Add(Row);

            }

            InventoryTbl.ItemsSource = InventoryRowesBindedToUser;

            InventoryBeforeChange = new List<InventoryRow>();

            foreach (InventoryRow Row in inventoryRows)

            {

                InventoryBeforeChange.Add(Row.copy());

            }

        }

        //מפנה לפונקציה שמעדכנת את השינויים שנעשו בידי המשתמש במסד הנתונים. מופע בלחיצה על כפתור העידכון

        //updates the data in db according to the changes made by the user (directs to a diffrent function). Treiggered at the click on the update button.

        private void UpdateDataFromTbl\_Click(object sender, RoutedEventArgs e)

        {

            UpdateDataToDB();

        }

        // הפונקציה בודקת אלו שינויים נעשו בידי המשתמש בטבלה, ומעדכנת את שינויים אלו בטבלה

        //checks what have been changed in the table by the user and updates the changes to db.

        private async void UpdateDataToDB()

        {

            InventoryRow sel = (InventoryRow)InventoryTbl.SelectedItem;

            ItemCollection ic = InventoryTbl.Items;

            List<InventoryRow> inventoryRows = new List<InventoryRow>();

            int len = 0;

            foreach (InventoryRow a in ic)

            {

                len++;

                InventoryRow ro = new InventoryRow();

                ro.Name = a.Name;

                ro.Quantity = a.Quantity;

                ro.NeededQuantity = a.NeededQuantity;

                ro.ID = a.ID;

                ro.Remarkes = a.Remarkes;

                inventoryRows.Add(ro);

            }

            foreach (InventoryRow a in inventoryRows)

            {

                InventoryServ.InventoryFuncsSoapClient s = new InventoryServ.InventoryFuncsSoapClient();

                foreach (InventoryRow original in InventoryBeforeChange)

                {

                    if (a.ID == original.ID)

                    {

                        if (a.Equal(original))

                        {

                            break;

                        }

                        else

                        {

                            InventoryServ.InventoryRow row = new InventoryServ.InventoryRow();

                            row.ID = a.ID;

                            row.Name = a.Name;

                            row.Quantity = a.Quantity;

                            row.NeededQuantity = a.NeededQuantity;

                            row.Remarkes = a.Remarkes;

                            bool isOk = await s.changeInventoryRowAsync(row);

                        }

                    }

                }

            }

        }

        //מוחק את הרשומה המסומנת בלחיצה על כפתור המחיקה

        //deletes the selected item when delete button is clicked

        private async void Delete\_Click(object sender, RoutedEventArgs e)

        {

            InventoryServ.InventoryFuncsSoapClient s = new InventoryServ.InventoryFuncsSoapClient();

            int index = InventoryTbl.SelectedIndex;

            int id = ((InventoryRow)InventoryTbl.Items[index]).ID;

            var HadWorked = await s.DeleteInventoryRowAsync(id, FullUser.Email, FullUser.Password);

            InventoryRowesBindedToUser.Remove(InventoryRowesBindedToUser[InventoryTbl.SelectedIndex]);

        }

        //מגיב ללחיצה על כתור ההוספה בהוספת רשומה ריקה לטבלה ולמסד הנתונים

        //responds to click by adding item to ListView

        private async void AddItem\_Click(object sender, RoutedEventArgs e)

        {

            InventoryServ.InventoryFuncsSoapClient s = new InventoryServ.InventoryFuncsSoapClient();

            InventoryRow NewRow = new InventoryRow();

            var ItemId = await s.getNewItemIdAsync(FullUser.ID, FullUser.Email, FullUser.Password);

            NewRow.ID = int.Parse(ItemId.ToString());

            NewRow.OwnerUserId = FullUser.ID;

            InventoryRowesBindedToUser.Add(NewRow);

            InventoryBeforeChange.Add(NewRow.copy());

        }

        //הפונקציה מקבלת עצם, ומוסיפה אותו למסד הנתונים תחת המשתמש הנוכחי

        //rwcives a new InventoryRow and add new item to the ListView, and to db

        private async void addItemFunc(InventoryRow a)

        {

            InventoryServ.InventoryFuncsSoapClient s = new InventoryServ.InventoryFuncsSoapClient();

            InventoryRow NewRow = a;

            var ItemId = await s.getNewItemIdAsync(FullUser.ID, FullUser.Email, FullUser.Password);

            NewRow.ID = int.Parse(ItemId.ToString());

            NewRow.OwnerUserId = FullUser.ID;

            var IsOk = await s.changeInventoryRowAsync(new InventoryServ.InventoryRow() { OwnerUserId = NewRow.OwnerUserId, ID = NewRow.ID, AmountOut= NewRow.AmountOut, Name = NewRow.Name, NeededQuantity = NewRow.NeededQuantity, Quantity = NewRow.Quantity, Remarkes = NewRow.Remarkes });

            InventoryRowesBindedToUser.Add(NewRow);

        }

        //מייבא רשימת מלאי של פריטים מקובץ סי.אס.וי

        //import inventory list from csv file

        private async void CsvImport\_Click(object sender, RoutedEventArgs e)

        {

            try

            {

                var picker = new Windows.Storage.Pickers.FileOpenPicker();

                picker.ViewMode = Windows.Storage.Pickers.PickerViewMode.Thumbnail;

                picker.SuggestedStartLocation = Windows.Storage.Pickers.PickerLocationId.PicturesLibrary;

                picker.FileTypeFilter.Add(".csv");

                Windows.Storage.StorageFile file = await picker.PickSingleFileAsync();

                var randomAccessStream = await file.OpenReadAsync();

                Stream stream = randomAccessStream.AsStreamForRead();

                StreamReader reader = new StreamReader(stream);

                List<InventoryRow> inventoryRows = new List<InventoryRow>();

                using (reader)

                {

                    string line;

                    while ((line = reader.ReadLine()) != null)

                    {

                        //define pattern

                        Regex csvparser = new Regex(",(?=(?:[^\"]\*\"[^\"]\*\")\*(?![^\"]\*\"))");

                        //separating columns to array

                        string[] x = csvparser.Split(line);

                        /\* do something with x \*/

                        InventoryRow row = new InventoryRow();

                        row.Name = x[0];

                        row.Quantity = float.Parse(x[1]);

                        row.NeededQuantity = float.Parse(x[2]);

                        row.Remarkes = x[3];

                        row.OwnerUserId = FullUser.ID;

                        inventoryRows.Add(row);

                    }

                }

                foreach (InventoryRow row in inventoryRows)

                {

                    addItemFunc(row);

                }

            }

            catch { }

        }

        // שולח את המשתמש לעמוד האחרון בו היה בלחיצה על כפתור החזרה

        //triggered at a click on the back button. Sendes the user back to the previus page.

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

        //מעלה תפריט פופאפ למילוא פרטי השאלה בידי המשתמש בלחיצה על כפתור הוספת ההשאלה בתפריט הקליק הימני של הפריט

        //reviles add landing menu after clicking add landing button

        private async void LandButton\_Click(object sender, RoutedEventArgs e)

        {

            Grid grid = new Grid();

            var rowDefinitions = grid.RowDefinitions;

            int i = 2;

            for(int j=0; j<i; j++)

            {

                rowDefinitions.Add(new RowDefinition());

            }

            TextBox amount = new TextBox { PlaceholderText = "amount" };

            Grid.SetRow(amount, 0); grid.Children.Add(amount);

            TextBox lentTo = new TextBox { PlaceholderText = "Lent to" };

            Grid.SetRow(lentTo, 1); grid.Children.Add(lentTo);

            ContentDialog getLandingDits = new ContentDialog()

            {

                Title = "please fill out the landing ditails",

                Content = grid,

                SecondaryButtonText = "save",

                SecondaryButtonCommandParameter= grid, SecondaryButtonCommand=new saveBtnCmd(),

                CloseButtonText = "cancel"

            };

            await getLandingDits.ShowAsync();

            LoadTblFunc();

        }

        //מעלה כפטור הוספת השאלה בחיצה על פריט בטבלה

        //reviles add landing button after right clicking an item

        private void Gridy\_RightTapped(object sender, RightTappedRoutedEventArgs e)

        {

            if (InventoryTbl.SelectedItem != null)

            {

                LantItem = InventoryTbl.Items[InventoryTbl.SelectedIndex] as InventoryRow;

                MenuFlyout rightClick = new MenuFlyout();

                MenuFlyoutItem firstItem = new MenuFlyoutItem { Text = "land out" };

                firstItem.Click += LandButton\_Click;

                rightClick.Items.Add(firstItem);

                var InventoryItemSelected = InventoryTbl.SelectedItem as InventoryRow;

                MenuFlyoutItem GoogleSearchItem = new MenuFlyoutItem { Text = "find where to buy more of this" };

                GoogleSearchItem.DataContext = InventoryItemSelected;

                GoogleSearchItem.Click += GoogleSearchItem\_Click ;

                rightClick.Items.Add(GoogleSearchItem);

                UIElement b = sender as UIElement;

                b.ContextFlyout = rightClick;

                Point point = new Point(e.GetPosition(b).X, e.GetPosition(b).Y);

                rightClick.ShowAt(b, point);

            }

        }

        //הפעולה שמתבצעת כאשר המשתמש לוחץ על האופציה של מציאת עוד מאותו מוצר

        //פעולה זו שולחת אותו ללינק הראשון ממגוון אתרים קמעוניים שיש להם עמוד לאותו מוצר

        //משתמש בשירות רשת של גוגל

        private async void GoogleSearchItem\_Click(object sender, RoutedEventArgs e)

        {

            var button = sender as MenuFlyoutItem;

            InventoryRow row = button.DataContext as InventoryRow;

            var s = (await googleApi(row.Name)).Items.First().Link;

            var success = await Windows.System.Launcher.LaunchUriAsync( new Uri(s ));

        }

        //הפקודה להוספת השאלה למסד הנתונים

        //the command for adding a landing

        class saveBtnCmd : ICommand

        {

            public event EventHandler CanExecuteChanged;

            public bool CanExecute(object parameter)

            {

                throw new NotImplementedException();

            }

            public async void Execute(object parameter)

            {

                UIElement content = parameter as UIElement;

                float amount = float.Parse(((content as Grid).Children[0] as TextBox).Text);

                string lentTo = ((content as Grid).Children[1] as TextBox).Text;

                var s = new BorowwDb.BorowwingsDBSoapClient();

                var id = await s.AddLendingAsync(LantItem.ID, lentTo, DateTime.Now, amount, FullUser.ID);

                var b = s.UpdateAmountOutAsync(LantItem.ID);

            }

        }

        //לוקח את המשתמש לעמוד בו הוא יכול לראות את כל ההשאלות הנוכחיות מן הרשימה שלו

        //takes the user to a page where he can see all of his active landings

        private void SeeLandings\_Click(object sender, RoutedEventArgs e)

        {

            BrowwingsAndDistractions.user = FullUser;

            BrowwingsAndDistractions.senderPage = typeof(InventoryView);

            Frame.Navigate(typeof(BrowwingsAndDistractions));

        }

    }

}

### עמוד ניהול השאלות- משתמש עליון

קוד ויזואלי

<Page

    x:Class="storageUniversal.BrowwingsAndDistractions"

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

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

    xmlns:data="using:storageUniversal.codes"

    xmlns:local="using:storageUniversal"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    xmlns:muxc="using:Microsoft.UI.Xaml.Controls"

    xmlns:Windows10FallCreatorsUpdate="http://schemas.microsoft.com/winfx/2006/xaml/presentation?IsApiContractPresent(Windows.Foundation.UniversalApiContract, 5)"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

<!--a page for adding and removing borrowings-->

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Grid.Column="0" Grid.Row="0" Name="back" Click="Back\_Click" VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <StackPanel Orientation="Horizontal" Grid.Column="1" Grid.Row="0">

            <Button Name="refresh" Background="Black" BorderBrush="DarkGray" BorderThickness="0" Click="Refresh\_Click" Height="40" Width="40" ToolTipService.ToolTip="refresh the items in the table">

                <SymbolIcon Symbol="Refresh"/>

            </Button>

            <Button Name="syncButton" Background="Black" BorderBrush="DarkGray" BorderThickness="0" Click="SyncButton\_Click" ToolTipService.ToolTip="sync changes to database" Height="40" Width="40">

                <SymbolIcon Symbol="Sync"></SymbolIcon>

            </Button>

        </StackPanel>

        <Button Grid.Column="0" Grid.Row="1" VerticalAlignment="Bottom" ToolTipService.ToolTip="see all pendind order requsts" x:Name="pendingOrders" Click="PendingOrders\_Click">

            <SymbolIcon Symbol="MailReply"></SymbolIcon>

        </Button>

        <ListView Grid.Column="1" Grid.Row="1" Name="LandsTbl" HorizontalContentAlignment="Stretch" HorizontalAlignment="Stretch" IsMultiSelectCheckBoxEnabled="False" SelectionMode="Extended">

            <ListView.Header>

                <Grid HorizontalAlignment="Stretch">

                    <Grid.ColumnDefinitions>

                        <ColumnDefinition Width="\*"></ColumnDefinition>

                        <ColumnDefinition Width="\*"></ColumnDefinition>

                        <ColumnDefinition Width="\*"></ColumnDefinition>

                        <ColumnDefinition Width="0.5\*"></ColumnDefinition>

                    </Grid.ColumnDefinitions>

                    <TextBlock Grid.Column="0" HorizontalAlignment="Center">item name</TextBlock>

                    <TextBlock Grid.Column="1" HorizontalAlignment="Center">amount lent</TextBlock>

                    <TextBlock Grid.Column="2" HorizontalAlignment="Center">taken by</TextBlock>

                    <TextBlock Grid.Column="3" HorizontalAlignment="Center">borrowed in</TextBlock>

                </Grid>

            </ListView.Header>

            <ListView.ItemTemplate>

                <DataTemplate  x:Name="dataTemp" x:DataType="data:Borrow" >

                    <Grid HorizontalAlignment="Stretch" RightTapped="Grid\_RightTapped">

                        <Grid.ColumnDefinitions>

                            <ColumnDefinition Width="\*"></ColumnDefinition>

                            <ColumnDefinition Width="\*"></ColumnDefinition>

                            <ColumnDefinition Width="\*"></ColumnDefinition>

                            <ColumnDefinition Width="0.5\*"></ColumnDefinition>

                        </Grid.ColumnDefinitions>

                        <TextBlock HorizontalAlignment="Stretch" Grid.Column="0" Name="itemName" Text="{Binding Name, Mode=TwoWay}" TextAlignment="DetectFromContent"></TextBlock>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="1" Name="amount" Text="{Binding Quantity, Mode=TwoWay}"></TextBox>

                        <TextBox HorizontalAlignment="Stretch" Grid.Column="2" Name="takenBy" Text="{Binding BorrowedBy, Mode=TwoWay}"></TextBox>

                        <TextBlock HorizontalAlignment="Center" Name="WhenBorrowed" Grid.Column="3" Text="{Binding When}"></TextBlock>

                        <TextBlock Visibility="Collapsed" Name="ID" Text="{Binding BorrowingId}"></TextBlock>

                    </Grid>

                </DataTemplate>

            </ListView.ItemTemplate>

            <ListView.ItemContainerStyle>

                <Style TargetType="ListViewItem">

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

                    <Setter Property="HorizontalAlignment"  Value="Stretch"></Setter>

                </Style>

            </ListView.ItemContainerStyle>

        </ListView>

    </Grid>

</Page>

קוד אחורי

using System;

using System.Data;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using Windows.UI.Core;

using System.Collections.ObjectModel;

using storageUniversal.xamls.UpperUser;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal

{

    /// <summary>

    /// This page allowes the user to menege his active landings

    /// </summary>

    public sealed partial class BrowwingsAndDistractions : Page

    {

        public static UserDBServ.User user;

        public static Type senderPage;

        //a collection of all the borrows binded in the table

        //אוסף של כל העצמים המוצגים בטבלה

        public static ObservableCollection<codes.Borrow> borrows = new ObservableCollection<codes.Borrow>();

        private List<codes.Borrow> borroesUnTouched = new List<codes.Borrow>();

        public BrowwingsAndDistractions()

        {

            this.InitializeComponent();

            loadTbl();

            //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

            // some code to handle mouse back + forward buttons

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

        //מעלה את כל ההשאלות הפעילות של המשתמש ומכניס אותם לטבלה

        //loads all the user's active borrowings and inserts them into ListView

        private async void loadTbl()

        {

            var BDB = new BorowwDb.BorowwingsDBSoapClient();

            var DataTbl = await BDB.GetLandingsAsync(user.ID);

            borrows.Clear();

            foreach(DataRow dr in DataTbl.Rows)

            {

                var bro = new codes.Borrow();

                bro.ItemId = int.Parse(dr["ItemId"].ToString());

                bro.BorrowedBy = dr["BorrowedBy"].ToString();

                bro.When = DateTime.Parse(dr["When"].ToString());

                bro.Quantity = float.Parse(dr["Quantity"].ToString());

                bro.UserId = int.Parse(dr["UserId"].ToString());

                bro.BorrowingId = int.Parse(dr["BorrowingId"].ToString());

                await bro.SetName(bro.ItemId);

                borrows.Add(bro);

            }

            foreach(codes.Borrow a in borrows)

            {

                borroesUnTouched.Add(a.copy());

            }

            LandsTbl.ItemsSource = borrows;

        }

        //מחזיר את המשתמש לעמוד הקודם

        //takes you back to previos page

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

        private List<codes.Borrow> selectedItems = new List<codes.Borrow>();

        private void Grid\_RightTapped(object sender, RightTappedRoutedEventArgs e)

        {

            selectedItems.Clear();

            foreach (ItemIndexRange range in LandsTbl.SelectedRanges.ToList())

            {

                for(int i=range.FirstIndex; i<=range.LastIndex; i++)

                {

                    selectedItems.Add(LandsTbl.Items[i] as codes.Borrow);

                }

            }

            if (selectedItems.Count <= 0)

            {

                codes.Borrow clickedItem = new codes.Borrow();

                int borrowid = int.Parse(((sender as Grid).Children.Last() as TextBlock).Text);

                foreach (codes.Borrow a in LandsTbl.Items)

                {

                    if (a.BorrowingId == borrowid)

                    {

                        clickedItem = a;

                    }

                }

                LandsTbl.SelectedItem = clickedItem;

            }

                //LandsTbl.SelectRange(new ItemIndexRange(clickedIndex, 1));

                //clickedItem = LandsTbl.Items[LandsTbl.item] as InventoryRow;

                MenuFlyout rightClick = new MenuFlyout();

                var icon = new SymbolIcon() { Symbol = Symbol.Delete };

                MenuFlyoutItem deleteOption = new MenuFlyoutItem { Icon = icon, Text = "delete", FontFamily = new FontFamily("Segoe MDL2 Assets") };

                deleteOption.Click += DeleteOption\_Click;

                rightClick.Items.Add(deleteOption);

                UIElement b = sender as UIElement;

                b.ContextFlyout = rightClick;

                Point point = new Point(e.GetPosition(b).X, e.GetPosition(b).Y);

                rightClick.ShowAt(b, point);

        }

        //מתבצע כאשר המשתמש לוחץ על אפשרות המחיקה

        //מוחק את המשתמש מהטבלה וממסד הנתונים

        private async void DeleteOption\_Click(object sender, RoutedEventArgs e)

        {

            Window.Current.CoreWindow.PointerCursor = new Windows.UI.Core.CoreCursor(Windows.UI.Core.CoreCursorType.Wait, 1);

            var BorServ = new BorowwDb.BorowwingsDBSoapClient();

            foreach(codes.Borrow br in selectedItems)

            {

                var bs = new BorowwDb.Borrow() { BorrowingId = br.BorrowingId, BorrowedBy= br.BorrowedBy, ItemId = br.ItemId, Quantity = br.Quantity, UserId = br.UserId, When = br.When };

                var isok =await BorServ.DeleteLandingAsync(bs, user.Email, user.Password);

            }

            loadTbl();

            Window.Current.CoreWindow.PointerCursor = new CoreCursor(CoreCursorType.Arrow, 1);

        }

        //מרענן את הטבלה

        private void Refresh\_Click(object sender, RoutedEventArgs e)

        {

            loadTbl();

        }

        //מסנכרן את הטבלה עם מסד הנתונים

        private async void SyncButton\_Click(object sender, RoutedEventArgs e)

        {

            syncToDataBase();

        }

        //ממיר השאלה מקומית להשאלה מהטיפוס של שרות הרשת

        public BorowwDb.Borrow convert(codes.Borrow borrow)

        {

            return new BorowwDb.Borrow() { BorrowedBy = borrow.BorrowedBy, BorrowingId = borrow.BorrowingId, ItemId = borrow.ItemId, Quantity = borrow.Quantity, UserId = borrow.UserId, When = borrow.When};

        }

        //מסנכרן את הטבלה עם מסד הנתונים

        public async void syncToDataBase()

        {

            Window.Current.CoreWindow.PointerCursor = new Windows.UI.Core.CoreCursor(Windows.UI.Core.CoreCursorType.Wait, 1);

            var bdb = new BorowwDb.BorowwingsDBSoapClient();

            for(int i=0; i<borrows.Count; i++)

            {

                var old = borroesUnTouched[i];

                await bdb.updateBorrowAsync(convert(old), convert(LandsTbl.Items[i] as codes.Borrow), user.Email, user.Password);

            }

            Window.Current.CoreWindow.PointerCursor = new CoreCursor(CoreCursorType.Arrow, 1);

        }

        //לוקח את המשתמש לעמוד המכיל את כל ההשאלות הממתינות לו

        private void PendingOrders\_Click(object sender, RoutedEventArgs e)

        {

            Frame.Navigate(typeof(UpperSeeOrders));

        }

    }

}

### עמוד ניהול הזמנות

קוד ויזואלי

<Page

    x:Class="storageUniversal.xamls.UpperUser.UpperSeeOrders"

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

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

    xmlns:local="using:storageUniversal.xamls.UpperUser"

    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

    xmlns:controls ="using:Microsoft.Toolkit.Uwp.UI.Controls"

    mc:Ignorable="d"

    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="40"></ColumnDefinition>

            <ColumnDefinition></ColumnDefinition>

        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions>

            <RowDefinition Height="40"></RowDefinition>

            <RowDefinition></RowDefinition>

            <RowDefinition></RowDefinition>

        </Grid.RowDefinitions>

        <Button Grid.Row="0" Grid.Column="0" Name="back" Click="Back\_Click" VerticalAlignment="Top" HorizontalAlignment="Left" Style="{StaticResource NavigationBackButtonNormalStyle}"/>

        <controls:DataGrid Grid.Column="1" Grid.Row="1" x:Name="UnAprovedOrders"

    AutoGenerateColumns="False" SelectionMode="Single" IsReadOnly="False" GridLinesVisibility="All" AlternatingRowBackground="Gray"

    ItemsSource="{x:Bind local:UpperSeeOrders.UnAprovedBindedOrders}" >

            <controls:DataGrid.Columns>

                <controls:DataGridTextColumn Header="item name" Binding="{Binding ItemName}" IsReadOnly="True"/>

                <controls:DataGridTextColumn Header="Amount" Binding="{Binding Amount, Mode=TwoWay}"/>

                <controls:DataGridTextColumn Header="Remarkes" Binding="{Binding Remarkes, Mode=TwoWay}"/>

                <controls:DataGridTextColumn Header="date" Binding="{Binding OrderDate}" IsReadOnly="True"/>

                <controls:DataGridTemplateColumn >

                    <controls:DataGridTemplateColumn.CellTemplate>

                        <DataTemplate>

                            <Button x:Name="AcceptOrder" Click="AcceptOrder\_Click">

                                <TextBlock>Accept order</TextBlock>

                            </Button>

                        </DataTemplate>

                    </controls:DataGridTemplateColumn.CellTemplate>

                </controls:DataGridTemplateColumn>

                <controls:DataGridTemplateColumn >

                    <controls:DataGridTemplateColumn.CellTemplate>

                        <DataTemplate>

                            <Button x:Name="RejectOrder" Click="RejectOrder\_Click" ToolTipService.ToolTip="reject order">

                                <SymbolIcon Symbol="Delete"></SymbolIcon>

                            </Button>

                        </DataTemplate>

                    </controls:DataGridTemplateColumn.CellTemplate>

                </controls:DataGridTemplateColumn>

            </controls:DataGrid.Columns>

        </controls:DataGrid>

        <StackPanel Orientation="Vertical" Grid.Column="1" Grid.Row="2" >

            <TextBlock TextDecorations="Underline">Aproved Orders:</TextBlock>

            <controls:DataGrid x:Name="AprovedOrders"

        AutoGenerateColumns="False" SelectionMode="Single" IsReadOnly="False" GridLinesVisibility="All" AlternatingRowBackground="Gray"

        ItemsSource="{x:Bind local:UpperSeeOrders.AprovedBindedOrders}" >

                <controls:DataGrid.Columns>

                    <controls:DataGridTextColumn Header="item name" Binding="{Binding ItemName}" IsReadOnly="True"/>

                    <controls:DataGridTextColumn Header="Amount" Binding="{Binding Amount, Mode=TwoWay}"/>

                    <controls:DataGridTextColumn Header="Remarkes" Binding="{Binding Remarkes, Mode=TwoWay}"/>

                    <controls:DataGridTextColumn Header="date" Binding="{Binding OrderDate}" IsReadOnly="True"/>

                    <!--<controls:DataGridTemplateColumn >

                        <controls:DataGridTemplateColumn.CellTemplate>

                            <DataTemplate>

                                <Button x:Name="AcceptOrder" Click="AcceptOrder\_Click">

                                    <TextBlock>Accept order</TextBlock>

                                </Button>

                            </DataTemplate>

                        </controls:DataGridTemplateColumn.CellTemplate>

                    </controls:DataGridTemplateColumn>

                    <controls:DataGridTemplateColumn >

                        <controls:DataGridTemplateColumn.CellTemplate>

                            <DataTemplate>

                                <Button x:Name="RejectOrder" Click="RejectOrder\_Click" ToolTipService.ToolTip="reject order">

                                    <SymbolIcon Symbol="Delete"></SymbolIcon>

                                </Button>

                            </DataTemplate>

                        </controls:DataGridTemplateColumn.CellTemplate>

                    </controls:DataGridTemplateColumn>-->

                </controls:DataGrid.Columns>

            </controls:DataGrid>

            <Button x:Name="addAllAprovedToBorrowings" Click="AddAllAprovedToBorrowings\_Click" ToolTipService.ToolTip="add all the aproved orders to active borowwings">

                <SymbolIcon Symbol="Send"></SymbolIcon>

            </Button>

        </StackPanel>

    </Grid>

</Page>

קוד אחורי

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using System.Collections.ObjectModel;

using System.Data;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=234238

namespace storageUniversal.xamls.UpperUser

{

    /// <summary>

    /// An empty page that can be used on its own or navigated to within a Frame.

    /// </summary>

    public sealed partial class UpperSeeOrders : Page

    {

        //רשימות של ההזמנות שאושרו ושלא אושרו

        public static ObservableCollection<SubUserServ.Order> UnAprovedBindedOrders = new ObservableCollection<SubUserServ.Order>();

        public static ObservableCollection<SubUserServ.Order> AprovedBindedOrders = new ObservableCollection<SubUserServ.Order>();

        private List<SubUserServ.Order> orders = new List<SubUserServ.Order>();

        public UpperSeeOrders()

        {

            this.InitializeComponent();

            LoadTables();

        }

        //ממיר את המשתמש מעמוד הכניסה למשתמש שמתאים לשירות הרשת של התת משתמשים

        public SubUserServ.User FullUpperUserFromSubServ

        {

            get

            {

                return new SubUserServ.User() { ID = UpperLogin.FullUser.ID, BDate = UpperLogin.FullUser.BDate, Compeny = UpperLogin.FullUser.Compeny, Email = UpperLogin.FullUser.Email, Fname = UpperLogin.FullUser.Fname, Lname = UpperLogin.FullUser.Lname, Password = UpperLogin.FullUser.Password };

            }

        }

        //מעלה את הטבלאות לפי המידע משירות הרשת

        public async void LoadTables()

        {

            var serv = new UserDBServ.UserDBServSoapClient();

            var tbldata = await serv.getUpperOrdersAsync(UpperLogin.FullUser);

            orders.Clear();

            foreach (DataRow row in tbldata.Rows)

            {

                var order = new SubUserServ.Order();

                order.ItemName = row["ItemName"].ToString();

                order.Amount = float.Parse(row["Amount"].ToString());

                order.ID = int.Parse(row["ID"].ToString());

                order.ItemId = int.Parse(row["ItemId"].ToString());

                order.BySubUser = int.Parse(row["BySubUser"].ToString());

                order.ToUpperUser = int.Parse(row["ToUpperUser"].ToString());

                order.Aproved = bool.Parse(row["Aproved"].ToString());

                order.Rejected = bool.Parse(row["Rejected"].ToString());

                order.Remarkes = row["Remarks"].ToString();

                if (DateTime.TryParse(row["OrderDate"].ToString(), out DateTime OrderDate))

                {

                    order.OrderDate = OrderDate;

                }

                orders.Add(order);

            }

            AprovedBindedOrders.Clear();

            UnAprovedBindedOrders.Clear();

            foreach(var order in orders)

            {

                if (!order.Rejected) {

                    if (order.Aproved)

                    {

                        AprovedBindedOrders.Add(order);

                    }

                    else

                    {

                        UnAprovedBindedOrders.Add(order);

                    }

                }

            }

        }

        //send user back to previus page

        //כפתור אחורה- שולח את המשתמש לעמוד הקודם

        private void Back\_Click(object sender, RoutedEventArgs e)

        {

            Frame frame = Window.Current.Content as Frame;

            if (frame.CanGoBack)

            {

                frame.GoBack();

            }

        }

        //לחיצה מאשרת את ההזמנה

        private async void AcceptOrder\_Click(object sender, RoutedEventArgs e)

        {

            Button btn = sender as Button;

            var order = btn.DataContext as SubUserServ.Order;

            order.Aproved = true;

            var serv = new SubUserServ.SubUsersServSoapClient();

            bool a = await serv.UpdateOrderByUpperUserAsync(FullUpperUserFromSubServ, order);

            UnAprovedBindedOrders.Remove(order);

            AprovedBindedOrders.Add(order);

        }

        //לחיצה דוחה את ההזמנה

        private async void RejectOrder\_Click(object sender, RoutedEventArgs e)

        {

            Button btn = sender as Button;

            var order = btn.DataContext as SubUserServ.Order;

            order.Rejected = true;

            var serv = new SubUserServ.SubUsersServSoapClient();

            bool a = await serv.UpdateOrderByUpperUserAsync(FullUpperUserFromSubServ, order);

            UnAprovedBindedOrders.Remove(order);

        }

        //פעולה שמוסיפה את ההשאלות שאאושרו לרשימת ההשאלות

        private async void AddAllAprovedToBorrowings\_Click(object sender, RoutedEventArgs e)

        {

            var borowwServ = new BorowwDb.BorowwingsDBSoapClient();

            var SubServ = new SubUserServ.SubUsersServSoapClient();

            foreach(var order in AprovedBindedOrders)

            {

                var borrower = await SubServ.GetYourSubUserAsync(FullUpperUserFromSubServ, order.BySubUser);

                int borrowing\_ID = await borowwServ.AddLendingAsync(order.ItemId,borrower.FName + " " + borrower.LName, DateTime.Now, order.Amount, UpperLogin.FullUser.ID);

                borowwServ.UpdateAmountOutAsync(order.ItemId);

            }

            AprovedBindedOrders.Clear();

        }

    }

}

### קוד למחלקה User

using System;

namespace storageUniversal

{

    public class User

    {

        //תכונות בסיסיות של המשתמש

        //defining basic properties

        private int id;

        private string fname;

        private string lname;

        private DateTime bdate;

        private string compeny;

        private string email;

        private string password;

        //בנאי ריק

        // empty constractor

        public User()

        {

        }

        //גטרים וסטרים לכל התכונות

        //getters + setters for properties

        public int ID

        {

            get

            {

                return this.id;

            }

            set

            {

                this.id = value;

            }

        }

        public string Fname

        {

            get { return this.fname; }

            set { this.fname = value; }

        }

        public string Lname

        {

            get { return this.lname; }

            set { this.lname = value; }

        }

        public DateTime BDate

        {

            get { return this.bdate; }

            set { this.bdate = value; }

        }

        //על מנת להציג את תכונת תאריך הלידה בפקד סטנדרטי יש צורך להוציא את המידע בצורת סוג דייטטיימאופסט עם יכולת להיות תכונה ריקה

        // מטרת תכונה זו היא להמיר את תאריך הלידה לסוג זה ולהיתממשק עם פקד בחירת התאריך

        //In order to display BDate in standard CalendarDatePicker I had to write a property that would return BDate as a nullable DateTimeOffset

        public DateTimeOffset? BDateTimeOffset

        {

            get { try { return DateTimeOffset.Parse(this.BDate.ToString()); }

                catch { return null; }

                }

            set { this.bdate = DateTime.Parse(value.ToString()); }

        }

        public string Compeny

        {

            get { return this.compeny; }

            set { this.compeny = value; }

        }

        public string Email

        {

            get { return this.email; }

            set { this.email = value; }

        }

        public string Password

        {

            get { return this.password; }

            set { this.password = value; }

        }

        //פעולה שיוצרת עצם חדש עם אותם תכונות כמו עצם זה

        // a fution that creatates another User obj identical to this one

        public User copy() {

            User a = new User();

            a.ID = this.id;

            a.Fname = this.Fname;

            a.Lname = this.Lname;

            a.BDate = this.BDate;

            a.compeny = this.compeny;

            a.email = this.email;

            a.password = this.password;

            return a;

        }

        //תכונה הבודקת האם משתמש אחר זהה בכל פרטיו למשתמש זה

        // a function that returns wether another User object is identical to this one

        public bool IsSame(User user)

        {

            try

            {

                bool idS = this.ID == user.ID;

                bool FnameS = this.Fname.Equals(user.Fname);

                bool LnameS = this.Lname.Equals(user.Lname);

                bool DateS = this.BDate.Equals(user.BDate);

                bool CompS = this.compeny.Equals(user.Compeny);

                bool EmailS = this.email.Equals(user.Email);

                bool PassS = this.password.Equals(user.Password);

                return idS && FnameS && LnameS && DateS && CompS && EmailS && PassS;

            }

            catch {

                return false;

            }

            }

        //made by yehonatan vishna

    }

}

### קוד למחלקת InventoryRow

using System;

using System.Collections.Generic;

using System.Linq;

namespace storageUniversal

{

    public class InventoryRow

    {

        //מגדיר את התכונות של העצם

        //defines some properies

        private int id;

        private string name;

        private float quantity;

        private float neededQuantity;

        private int ownerUserId;

        private float amountOut;

        private string remarks;

        //פעולות בונות

        //constractores

        public InventoryRow() { }

        public InventoryRow(int id, string name, int q, int nq, int oui)

        {

            this.id = id;

            this.name = name;

            this.quantity = q;

            this.neededQuantity = nq;

            this.ownerUserId = oui;

        }

        //גטרים וסטרים לכל התכונות

        //getters and setters for all the properties listed above

        public int ID

        {

            get { return this.id; }

            set { this.id = value; }

        }

        public string Name

        {

            get { return this.name; }

            set { this.name = value; }

        }

        public float Quantity

        {

            get { return this.quantity; }

            set { this.quantity = value; }

        }

        public float NeededQuantity

        {

            get { return this.neededQuantity; }

            set { this.neededQuantity = value; }

        }

        public int OwnerUserId

        {

            get { return this.ownerUserId; }

            set { this.ownerUserId = value; }

        }

        public float AmountOut

        {

            get { return this.amountOut; }

            set { this.amountOut = value; }

        }

        public String Remarkes

        {

            get { return this.remarks; }

            set { this.remarks = value; }

        }

        //פונקציה שיוצרת עצם שורת מלאי זהה עם אותם תכונות

        // a fution that creatates another InventoryRow obj identical to this one

        public InventoryRow copy()

        {

            InventoryRow a = new InventoryRow();

            a.ID = this.id;

            a.Quantity = this.quantity;

            a.OwnerUserId = this.ownerUserId;

            a.NeededQuantity = this.neededQuantity;

            a.Name = this.name;

            a.Remarkes = this.remarks;

            return a;

        }

        //פונקציה שבודקת האם עצם רשימת מלאי אחר זהה לעצם זה בכל תכונותיו

        // a function that returns wether another InventoryRow object is identical to this one

        public bool Equal(InventoryRow row)

        {

            bool IdSame = this.id == row.ID;

            bool QSame = this.quantity == row.Quantity;

            bool OUIdSame = this.ownerUserId == row.OwnerUserId;

            bool NQSame = this.neededQuantity == row.NeededQuantity;

            bool NameSame;

            if ((row.Name == null && this.name != null) || (row.Name != null && this.name == null))

            {

                NameSame = false;

            }

            else

            {

                if ((row.Name == null && this.name == null))

                {

                    NameSame = true;

                }

                else

                {

                    NameSame = this.name.Equals(row.Name.ToString());

                }

            }

            bool RSame;

            if ((row.Remarkes == null && this.remarks != null) || (row.Remarkes != null && this.remarks == null))

            {

                RSame = false;

            }

            else

            {

                if ((row.Remarkes == null && this.remarks == null))

                {

                    RSame = true;

                }

                else

                {

                    RSame = this.remarks.Equals(row.Remarkes.ToString());

                }

            }

            return IdSame && QSame && OUIdSame && NQSame && NameSame && RSame;

        }

    }

}

### קוד למחלקת SubUser

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace storageUniversal

{

    public class SubUser

    {

        //מגדיר תכונות בסיסיות

        //defining reqired properties

        private int id;

        private int belongsToUpperUser;

        private string fName;

        private string lName;

        private string role;

        private string email;

        private string password;

        private string userName;

        public SubUser()

        {

        }

        public int Id { get => id; set => id = value; }

        public int BelongsToUpperUser { get => belongsToUpperUser; set => belongsToUpperUser = value; }

        public string FName { get => fName; set => fName = value; }

        public string LName { get => lName; set => lName = value; }

        public string Role { get => role; set => role = value; }

        public string Email { get => email; set => email = value; }

        public string Password { get => password; set => password = value; }

        public string UserName { get => userName; set => userName = value; }

        public SubUser Copy()

        {

            return new SubUser() { Id = this.id, BelongsToUpperUser = this.BelongsToUpperUser, FName = this.FName, LName = this.LName, Role = this.Role, UserName = this.userName };

        }

        public bool IsSame(SubUser user)

        {

            try

            {

                bool idS = this.Id == user.Id;

                bool FnameS = this.FName.Equals(user.FName);

                bool LnameS = this.LName.Equals(user.LName);

                bool EmailS = this.email.Equals(user.Email);

                bool PassS = this.password.Equals(user.Password);

                bool belS = this.BelongsToUpperUser.Equals(user.BelongsToUpperUser);

                bool roleS = this.Role.Equals(user.Role);

                bool Usr = this.UserName.Equals(user.UserName);

                return idS && FnameS && LnameS && EmailS && PassS && belS && roleS && Usr;

            }

            catch

            {

                return false;

            }

        }

    }

}

### קוד למחלקת UsersDatabase

הקוד הזה בשימוש ליצירת מסד נתונים מקומי ב sqllite

using Microsoft.Data.Sqlite;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Threading.Tasks;

namespace storageUniversal

{

    public class UsersDatabase

    {

        SqliteConnection Database;

        public UsersDatabase()

        {

        }

        //מוודא שמסד הנתונים מאותחל אם לא מאתחל אותו

        //makes shure the db is initilized

        async Task Init()

        {

            // Return if the database has already been initialized

            if (Database != null)

                return;

            // Create a new Sqlite database connection using the specified database file path

            Database = new SqliteConnection($"Filename={Constants.DatabasePath}");

            // Open the database connection

            Database.Open();

            // Create the table command string with the specified columns and data types

            //שאילתה שיוצרת את מסד הנתונים אם הוא לא כבר קיים

            String tableCommand = "CREATE TABLE IF NOT " +

        "EXISTS Users ( [ID][int] IDENTITY NOT NULL," +

        "[FName] [nchar] (20) NULL," +

        "[LName] [nchar] (20) NULL," +

        "[BDate] [datetime] NULL," +

        "[compeny] [nchar] (20) NULL," +

        "[email] [nchar] (30) NULL," +

        "[password] [nchar] (20) NULL)";

            // Create a new Sqlite command using the table command string and the database connection

            SqliteCommand createTable = new SqliteCommand(tableCommand, Database);

            // Execute the command to create the table

            createTable.ExecuteReader();

        }

        //מחזיר רשימה של כל המשתמשים במסד הנתונים המקומי

        //returns a User list of all users in db

        public async Task<List<User>> GetItemsAsync()

        {

            // Initialize the database

            await Init();

            // Create a query string to select all rows from the Users table

            var qury = "select \* from Users";

            // Create a new Sqlite command using the query string and the database connection

            var comd = new Microsoft.Data.Sqlite.SqliteCommand(qury, Database);

            // Execute the command to retrieve the rows from the table

            var reader = comd.ExecuteReader();

            // Create a new DataTable to hold the rows

            var tbl = new DataTable();

            // Load the rows into the DataTable

            tbl.Load(reader);

            // Create a list to hold the user objects

            var users = new List<User>();

            // Loop through each row in the DataTable

            foreach (DataRow dr in tbl.Rows)

            {

                // Create a new User object and populate its properties with the data from the DataRow

                var user = new User();

                user.ID = int.Parse(dr["ID"].ToString());

                user.Password = dr["password"].ToString();

                user.Email = dr["email"].ToString();

                user.Fname = dr["FName"].ToString();

                user.Lname = dr["LName"].ToString();

                user.Compeny = dr["compeny"].ToString();

                // Add the user to the list of users

                users.Add(user);

            }

            // Return the list of users

            return users;

        }

        //מקבל עצם של משתמש ומכניס את המידע המוכל בעצם למסד הנתונים

        //takes a user object and inserts it into the db

        public async Task<bool> InsertItemAsync(User item)

        {

            // Initialize the database

            await Init();

            // Create a string containing the query to insert a new row into the Users table with the data from the provided User object

            string nonQury = "Insert into Users (ID, email, password, FName, LName, BDate, compeny) values (" + item.ID.ToString() + ",'" + item.Email + "', '" + item.Password + "','" + item.Fname.ToString() + "', '" + item.Lname + "', " + "CAST('" + item.BDate.ToShortDateString() + "' AS DateTime)" + ", '" + item.Compeny + "');";

            // Create a new Sqlite command using the query string and the database connection

            var comd = new SqliteCommand(nonQury, Database);

            // Execute the command to insert the new row

            var a = comd.ExecuteNonQuery();

            // Dispose of the command object

            comd.Dispose();

            // Return a value indicating whether the insertion was successful

            return a > 0;

        }

        //מוחק את כל המשתמשים במסד הנתונים

        //deletes all users in db

        public async Task<bool> DeleteAll()

        {

            // Initialize the database

            await Init();

            // Create a string containing the query to delete all rows from the Users table

            string nonQury = "Delete from Users";

            // Create a new Sqlite command using the query string and the database connection

            var comd = new SqliteCommand(nonQury, Database);

            // Execute the command to delete the rows

            var a = comd.ExecuteNonQuery();

            // Dispose of the command object

            comd.Dispose();

            // Return a value indicating whether the deletion was successful

            return a > 0;

        }

    }

}

### הקוד למחלקת backMouse

הקוד בשימוש להוספה מהירה של מעבר עמוד אחורה עם כפתורי העכבר

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Windows.UI.Core;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

namespace storageUniversal.codes

{

    public class backMouse

    {

        public backMouse()

        {

            Window.Current.Activate();

            Window.Current.CoreWindow.PointerPressed += CoreWindow\_PointerPressed;

        }

        //קוד לניווט אחורה וקדימה עם כפטורי הניווט של העכבר

        // some code to handle mouse back + forward buttons

        private void CoreWindow\_PointerPressed(CoreWindow sender, PointerEventArgs args)

        {

            if (args.CurrentPoint.Properties.IsXButton1Pressed)

            {

                Frame frame = Window.Current.Content as Frame;

                if (frame.CanGoBack)

                {

                    frame.GoBack();

                }

            }

            else

            {

                if (args.CurrentPoint.Properties.IsXButton2Pressed)

                {

                    Frame frame = Window.Current.Content as Frame;

                    if (frame.CanGoForward)

                    {

                        frame.GoForward();

                    }

                }

            }

        }

    }

}

## הקוד לצד שרת:

הקוד למחלקות צד שרת ולשירותי הרשת כבר צורף בחלק המתאים בעבודה.