תיק פרויוקט- תוכנה לניהול מחסן ומלאי

דרכא רמון | על שם אילן ואסף רמון

נכתב ב UWP

יהונתן וישנה

שנה"ל: תשפ"ג- 2023

תוכן

[עמוד ראשי. 2](#_Toc122702242)

[עמוד אודות: 4](#_Toc122702243)

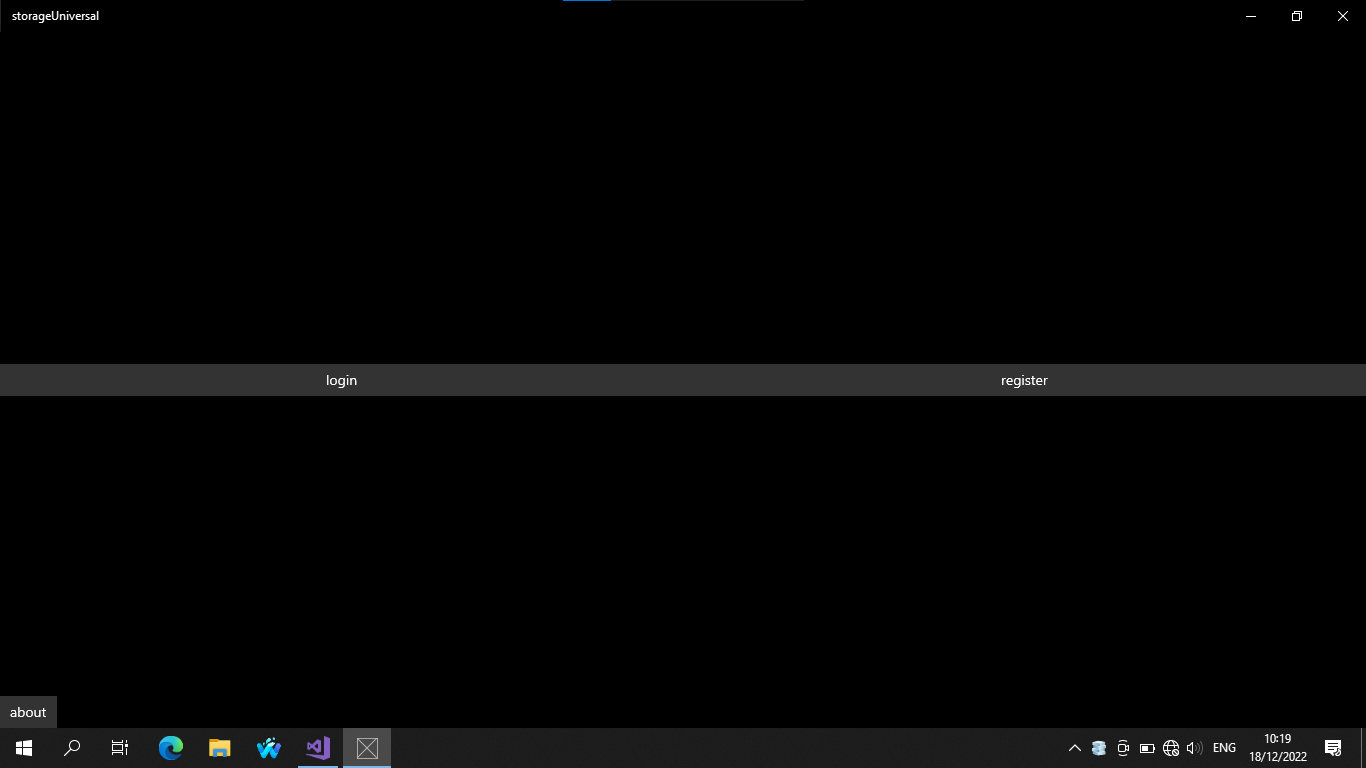
[עמוד הרשמה: 7](#_Toc122702244)

[עמוד כניסה 11](#_Toc122702245)

[עמוד עריכת מלאי 17](#_Toc122702246)

# עמוד ראשי.

מטרת עמוד זה היא להפנות את המשתמש לעמודים המשמעותיים האחרים בפרויקט- כניסה, הרשמה, ועמוד אודות. מצורפת למטה תמונה:



הקוד האחורי לעמוד (הBACKEND):

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;

// 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 MainPage()

        {

            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;

        }

        // 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();

                    }

                }

            }

        }

        //sends user to login

        //שולח את המשתמש לעמוד הכניסה

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

        {

            login.SentFrom = typeof(MainPage);

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

        }

        //sends user to register

        //שולח את המשתמש לעמוד הרשמה

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

        {

            Register.SentFrom = typeof(MainPage);

            Frame.Navigate(typeof(Register));

        }

        //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));

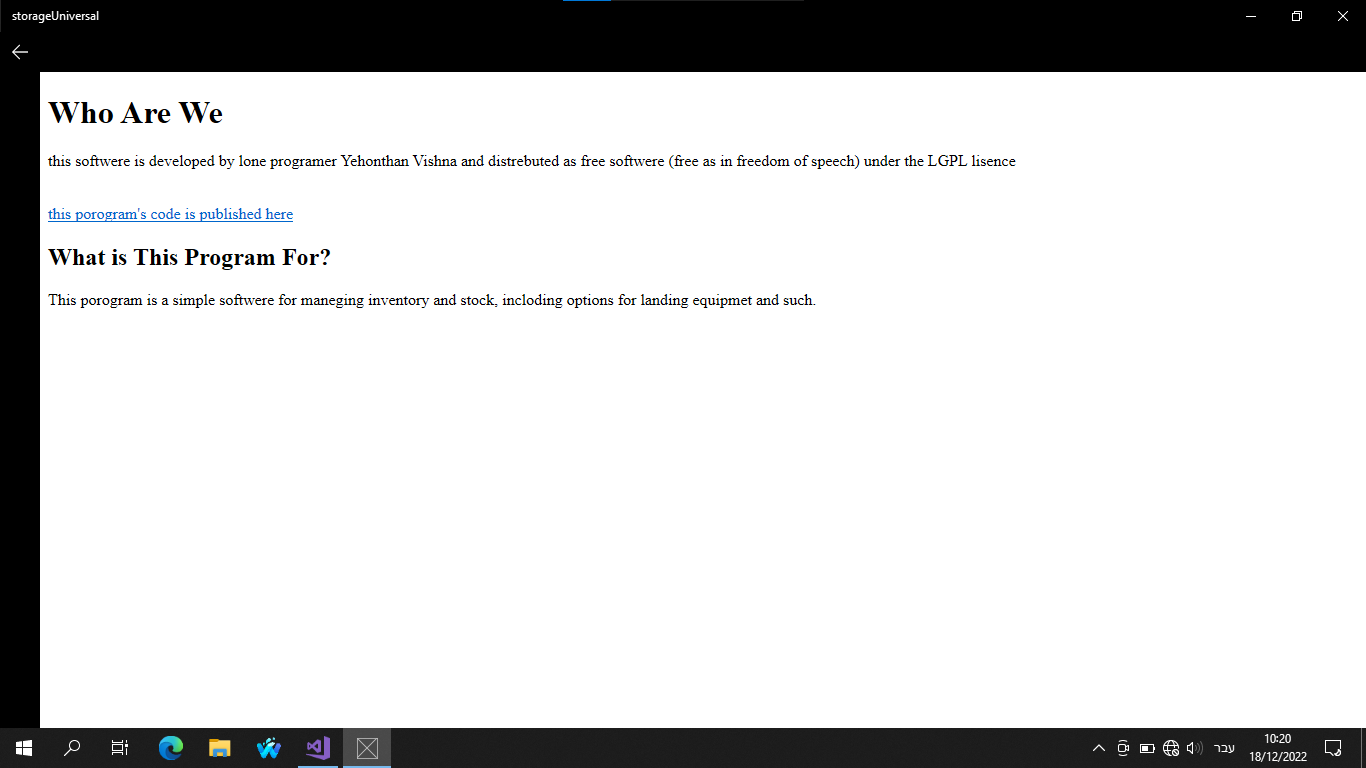
        }

    }

}

# עמוד אודות:

מטעמי נוחות עיצוב, העדפתי לשלב את עמוד האודות על האפליקציה בצורת עמוד HTML והוספתי אותו במסגרת אוביקט WEBVIEW לעמוד באפליקציה. מטרת עמוד זה היא להסביר אודות האפליקציה, ומה מטרתה.



הקוד האחורי לעמוד (BACKEND):

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();

            }

        }

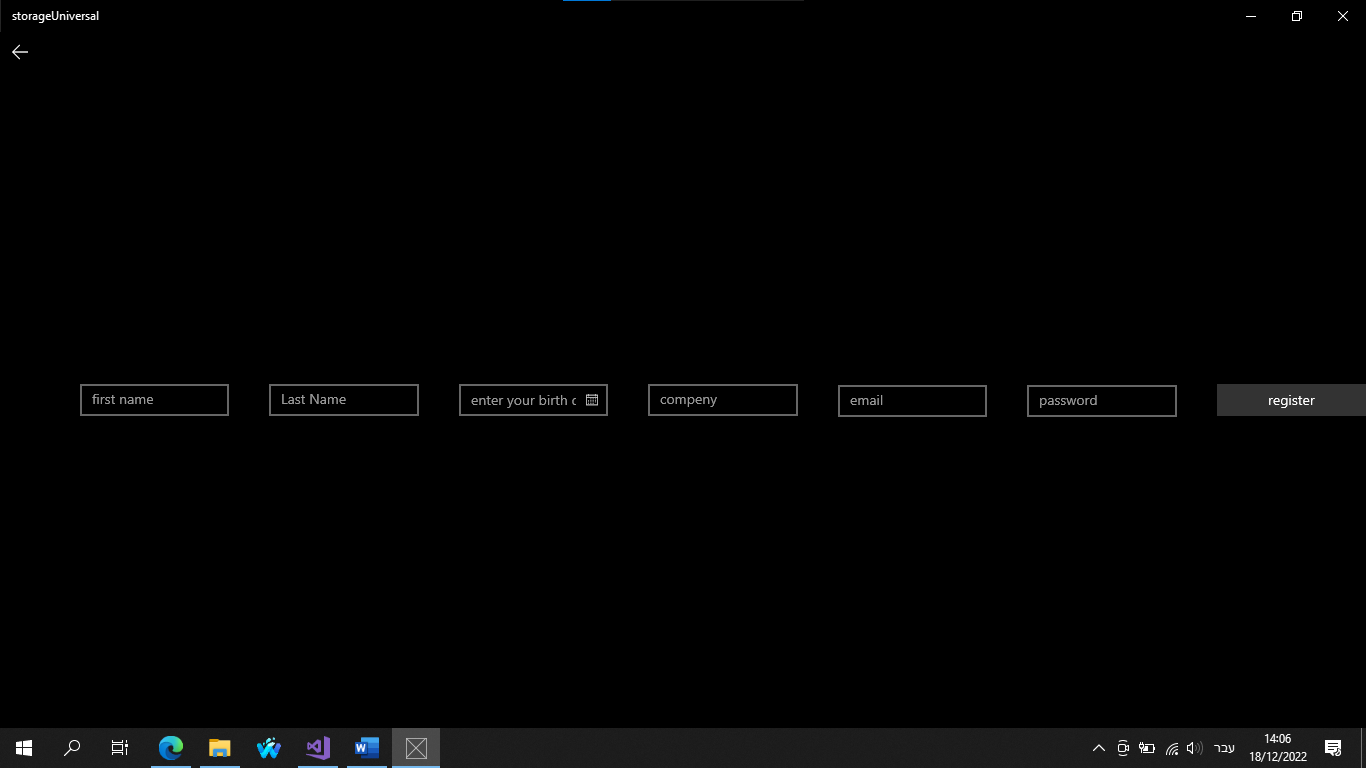
    }

}

# עמוד הרשמה:

מטרת עמוד ההרשמה היא לקבל את פרטי המשתמש ולבדוק אותם (שהשדות חוקיים, ואין כפילויות באימיילים), להעביר אותם לשירות הרשת- כאשר שירות הרשת יוסיף את פרטי המשתמשים למסד הנתונים.

תמונת העמוד:



בדיקות ההרשמה הנוכחיות:

עד כה, לפני ההרשמה, נבדק האם המייל חוקי, ונבדק שהמייל לא קיים כבר במסד נתונים, לאחר ההרשמה (בהנחה שהושלמה בהצלחה), העמוד שולח את המשתמש חזרה לעמוד הראשי.

הקוד האחורי (BACKEND):

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();

            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;

            }

            var date = BDate.Date;

            DateTime time = date.Value.DateTime;

            usr.BDate = time;

            usr.Fname = FN.Text;

            usr.Lname = LN.Text;

            usr.Email = email.Text;

            usr.Password = pass.Text;

            usr.Compeny = compeny.Text;

            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();

            }

        }

        private void Grid\_KeyDown(object sender, KeyRoutedEventArgs e)

        {

        }

    }

}

# עמוד כניסה

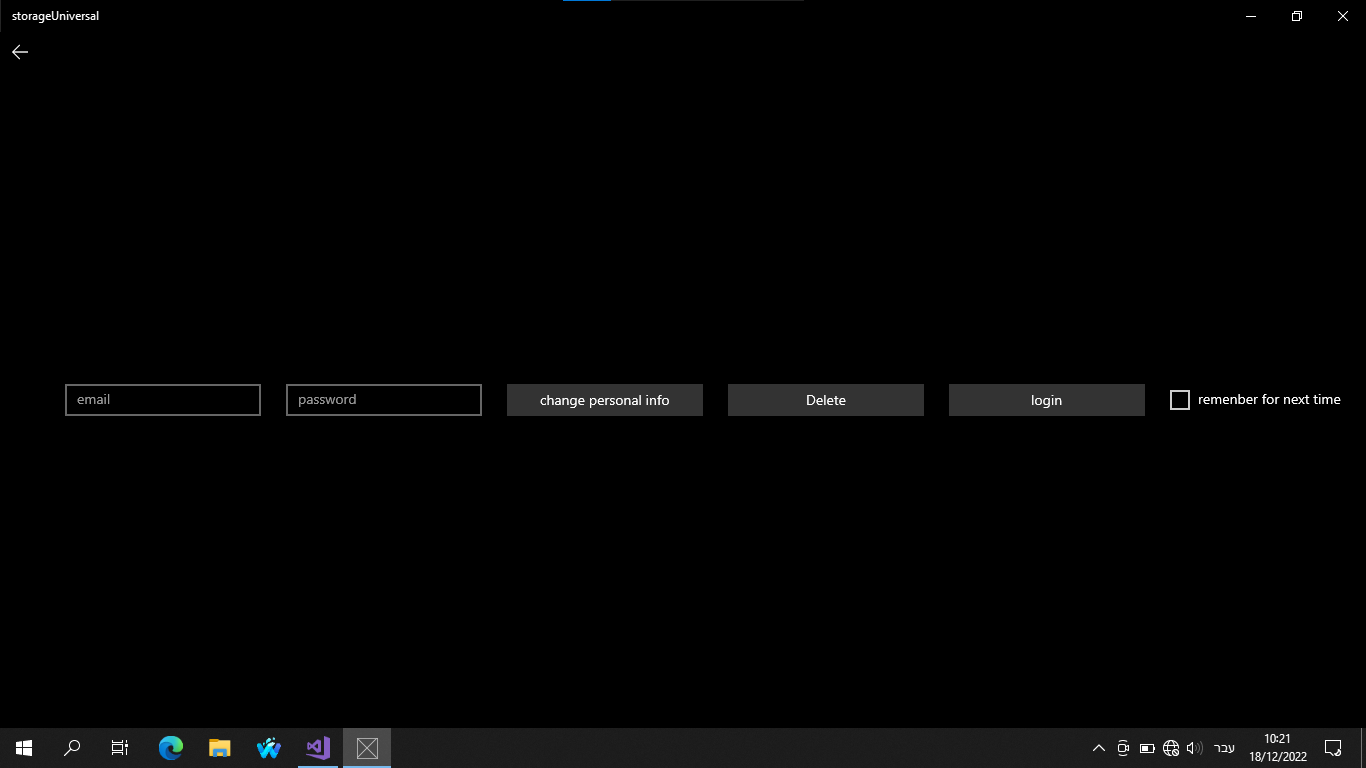
מטרת עמוד הכניסה היא לקחת מהמשתמש את אימייל והססמה שלו, ואז לחבר אותו למערכת ולתת לו לבצע מספר פעולות:

* לערוך את פרטיו האישיים
* למחוק את המשתמש שלו
* להיכנס לעמוד בו יראה את כל הפרטי מלאי, וההשאלות האקטיביות שלו.
* להעביר את מנהל המערכת (בעל האימייל: [admin@administrator.adm](mailto:admin@administrator.adm) והססמה: Admin) לעמוד ניהול משתמשים.

בנוסף, בעמוד זה יישמתי שימוש במסד נתונים מקומי: SQL LITE כדי לשמור את המשתמש האחרון שהתחבר למערכת במסד נתונים מקומי, כדי שיוכל להתחבר אוטומטית בכניסה הבאה (יש לו בחירה האם לשמור את נתוניו מקומית או לא). כשמשתמש ששמורים במחשבו בSQL LITE פרטי ההתחברות שלו נכנס לעמוד הכניסה, עולה לו תפריט POPUP שנותן לו את הבחירה האם להשתמש בנתוני משתמש השמורים או לא.

אם משתמש מכניס אימייל או סיסמה שלא נכונים, עולה לו מודעת POPUP שמודיעה לו שאחד מפרטי ההתחברות שגויים.

תמונה של עמוד ההתחברות:



הקוד האחורי:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

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;

// 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 login : 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;

        public login()

        {

            this.InitializeComponent();

            TryStartAutoLoging();

            // some code to handle mouse back + forward buttons

            // קוד להוספת פונקציית קדימה ואחורה לכפתורי העכבר

            Window.Current.Activate();

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

        }

        // 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(usr);

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

            if (b)

            {

                res.Text = "user exists, data should be showen";

                var TempFullUsr = await UDBS.GetFullUserAsync(usr);

                FullUser = TempFullUsr;

                InventoryView.SentFrom = typeof(login);

                frame.Navigate(typeof(InventoryView));

            }

            else

            {

                res.Text = "email or password are wrong, try again";

            }

        }

        //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 login);

                logAutoCheck(logPg.Frame, logPg.res);

            }

        }

        //logs in user and sends him to change his personal info

        //שולח את המשתמש לשנות את פרטיו האישיים

        private async void SendToDitailUpdate(object sender, RoutedEventArgs e)

        {

            usr.Password = password.Password;

            usr.Email = email.Text;

            usr.BDate = DateTime.Now;

            var a = await  UDBS.IsUserPermittedAsync(usr);

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

            bool isAdmin = await UDBS.IsAdminAsync(usr);

            if (!isAdmin)

            {

                if (b)

                {

                    res.Text = "user exists, data should be showen";

                    var TempFullUsr = await UDBS.GetFullUserAsync(usr);

                    FullUser = TempFullUsr;

                    updateUser.SentFrom = typeof(login);

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

                }

                else

                {

                    res.Text = "email or password are wrong, try again";

                }

            }

            else

            {

                FullUser = usr;

                AdminPanel.SentFrom = typeof(login);

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

            }

        }

        //deletes user from system

        //מוחק את המשתמש

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

        {

            usr.Password = password.Password;

            usr.Email = email.Text;

            var a = await UDBS.DeleteUserAsync(usr);

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

        }

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

        // מחבר את המשתמש, ושולח אותו לרשימת המלאי שלו

        private async void LoginAndSendToInventoryList(object sender, RoutedEventArgs e)

        {

            usr.Password = password.Password;

            usr.Email = email.Text;

            var a = await UDBS.IsUserPermittedAsync(usr);

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

            if (b)

            {

                res.Text = "user exists, data should be showen";

                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});

                }

                InventoryView.SentFrom = typeof(login);

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

            }

            else

            {

                res.Text = "email or password are wrong, try again";

            }

        }

        //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();

            }

        }

    }

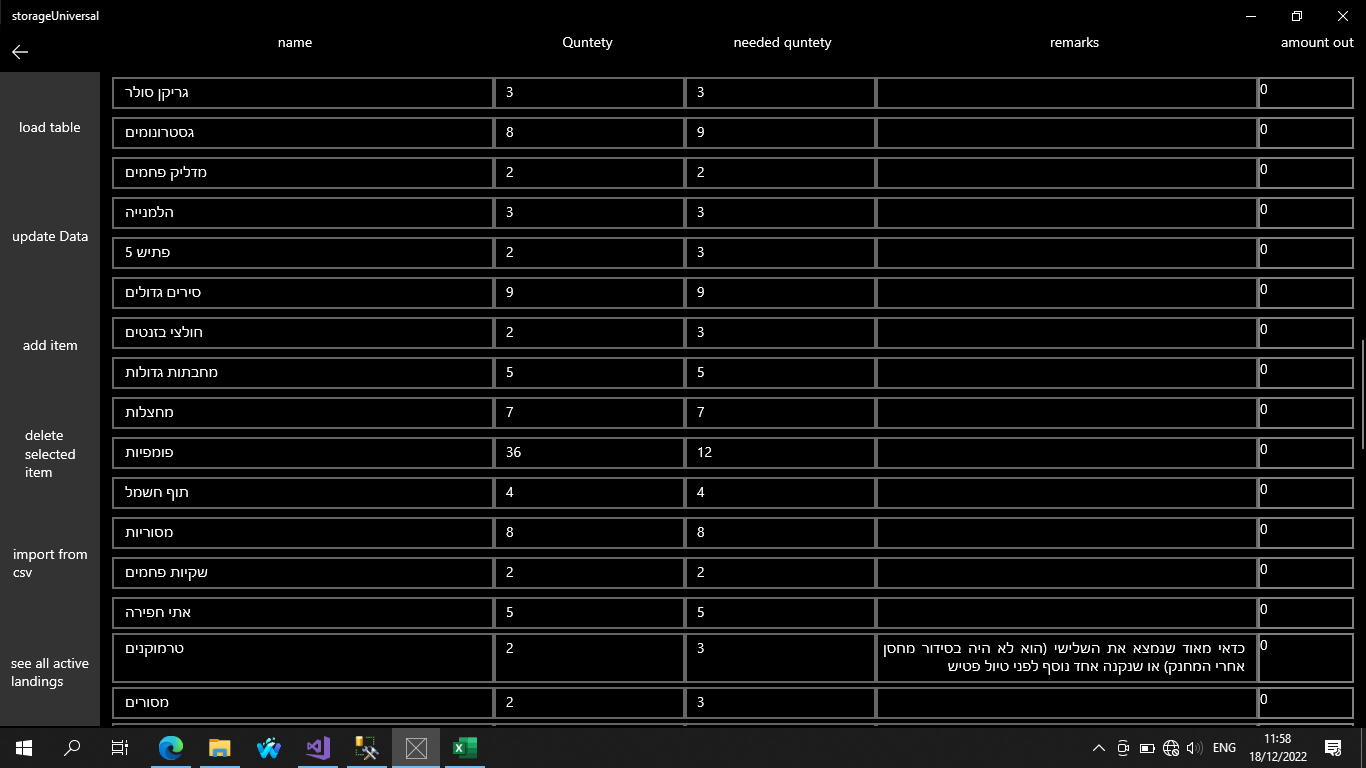
}

# עמוד עריכת מלאי

זהו העמוד המרכזי והמשמעותי ביותר בפרויקט. בעמוד זה, מוצגת עבור המשתמש רשימה של כל המלאי שיש בבעלותו. המשתמש מגיע לעמוד לאחר שהתחבר בעמוד הכניסה, ומוצגת בפניו כל המלאי שלו באובייקט ListView. למשתמש יש מספר דברים שהוא יכול לעשות עם המלאי:

* הוא יכול למחוק פריט
* להוסיף פריט
* לערוך את המידע של הפריט (ישירות בתוך הטבלה)
* לייבא פריטים חדשים מקובץ CSV
* להוסיף השאלה של פריט על ידי קליק ימני על הפריט (יעלה Popup שיבקש ממנו את פרטי ההשאלה, לאחר מכן ההשאלה תישלח לשירות רשת ותתווסף לטבלאת ההשאלות).

תמונה של העמוד:



הקוד האחורי (back end):

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;

// 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 = login.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;

        }

        //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();

                    }

                }

            }

        }

        private void InventoryTbl\_ItemClick(object sender, ItemClickEventArgs e)

        {

        }

        //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);

                UIElement b = sender as UIElement;

                b.ContextFlyout = rightClick;

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

                rightClick.ShowAt(b, point);

            }

        }

        //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));

        }

    }

}