‏19/5/2020

מנחה: עוז ארוטז מקיף י"א אשדוד

|  |  |
| --- | --- |
| עומר בניסטטי | מערכת לניהול מספרה |

תעודת זהות: 209265628

# **תוכן**

[**נושא הפרויקט:** 3](#_Toc41749774)

[**רקע כללי על המספרה:** 3](#_Toc41749775)

[**מהות התוכנה:** 3](#_Toc41749776)

[**תהליכים מרכזיים:** 4](#_Toc41749777)

[**מבנה נתונים:** 5](#_Toc41749778)

[קישור בין טבלאות 5](#_Toc41749779)

[הסבר על מבנה הנתונים 7](#_Toc41749780)

[**מדריך למשתמש:** 11](#_Toc41749781)

[הפעלת התוכנה: 11](#_Toc41749782)

# **נושא הפרויקט**

הקמה של מערכת ממחושבת לניהול מספרה.

# **רקע כללי על המספרה**

המספרה הכי ותיקה באשדוד "מספרת גבי" שעובדת יותר מ 15 שנה, מספרה זו שנים מספרת את תושבי אשדוד הותיקים וגם הצעירים, בעיקר מתמצא בתספורות גברים אך ישנה גם אפשרות לתספורות נשים.

# **מהות התוכנה**

כיום המספרה עדיין משתמשת בשיטות ניהול הישנות שהשתמשה כשנפתחה, רישום התורים, העובדים, הלקוחות וספירת מלאי כולם נעשים ידני על ידי כתיבה ביומנים ושמירת מסמכים, שיטה זו היא ישנה ומוסרבלת ובנוסף שמירת כל הנתונים שנאספו במיוחד במשך 15 שנה לוקח מקום פיזי רב.

תוכנה זו נועדה לטפל בבעיות אלה, היא תאפשר מעקב נוח ופשוט יותר אחר הלקוחות, העובדים, והמלאי, תשמור את כל המידע הנאסף במבנה נתונים ותאפשר הצגה שלו באופן נוחה למשתמש כך אנו פותרים את הבעיות הנלוות בשימוש שיטות ניהול ישנות כמו שהעסק הנ"ל משתמש בהם.

# **תהליכים מרכזיים**

דף ראשי:

* דוחות
* ניהול תורים
* ניהול עובדים
* ניהול מלאי
* ניהול לקוחות

דוחות:

* הכנסות הוצאות
* תורים
* הזמנות וחומרים
* עובדים

ניהול לקוחות:

* הוספת לקוח
* עריכת לקוח
* היסטורית לקוח

ניהול תורים:

* הוספת תור
* ביטול תור
* רשימת תורים

ניהול עובדים:

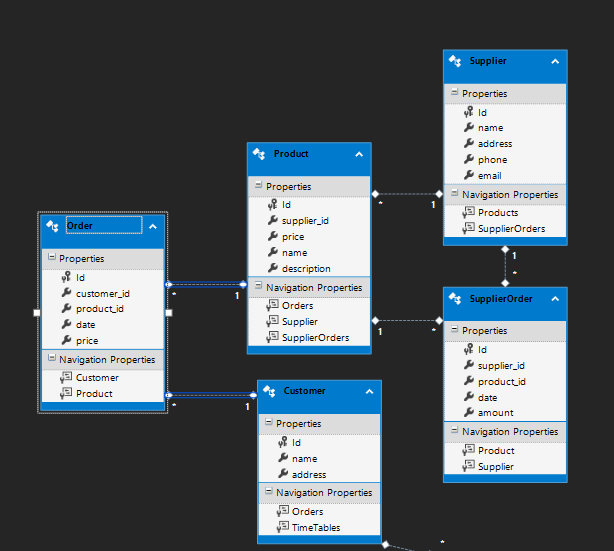
* הוספת עובד
* עריכת פרטים
* רישום שעות עבודה

ניהול מלאי:

* הוספת מוצרים
* עריכת פרטי מוצר
* הוספת ספקים
* עריכת פרטי ספק
* רשימת הזמנות

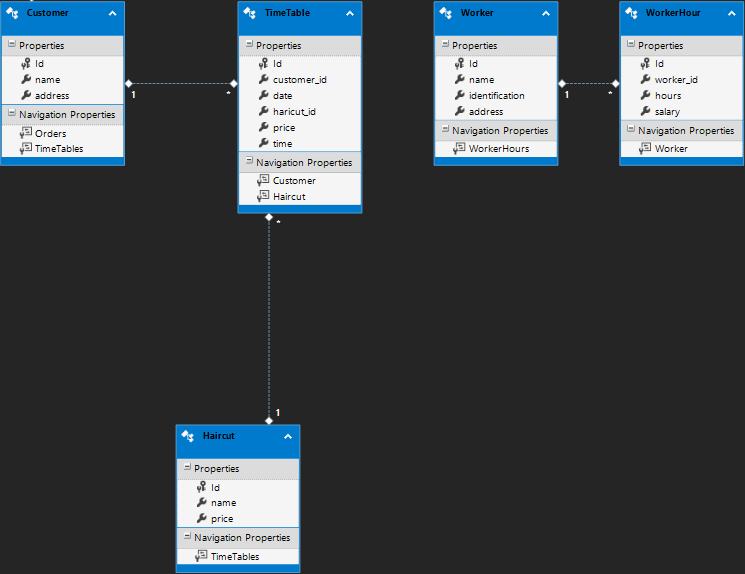
# **מבנה נתונים**

# קישור בין טבלאות



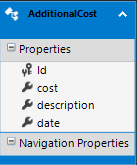
קישורים:

* השדה customer\_id בטבלה Order קשור לשדה Id בטבלה Customer, לכל הזמנה מקשרים את הלקוח שהזמין להזמנה שלו.
* השדה product\_id בטבלה Order קשור לשדה Id בטבלה Product, לכל הזמנה מקשרים את המוצר שהוזמן .
* השדה supplier\_id בטבלה Product קשור לשדה Id ב Supplier, לכל מוצר קשור הספק שמספק את המוצר.
* השדה supplier\_id בטבלה SupplierOrder קשור לדשה Id ב Supplier, לכל הזמנה למלאי מספק קשור הספק שממנו הוזמן המלאי.
* השדה product\_id בטבלה SupplierOrder קשור לשדה Id ב Product, לכל הזמנה למלאי קשור המוצר שאותו הזמנו.



* השדה customer\_id בטבלה TimeTable קשור לשדה Id בטבלה Customer, לכל תור במספרה קשור הלקוח שהזמין את הטור.
* השדה haircut\_id בטבלה TimeTable קשור לשדה Id בטבלה Haircut, לכל קביעת תור מצוין סוג התספורת שהלקוח ביקש.
* השדה worker\_id בטבלה WorkerHour קשור לשדה Id בטבלה Worker, שמירת שעות העבודה של העובדים מקושר לכל עובד.

# הסבר על מבנה הנתונים

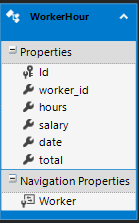
**AdditionalCost**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Cost** – מחיר ההוצאה – float

**Description** – תיאור ההוצאה - nvarchar(150)

**Date** – תאריך שבו התבצעה ההוצאה – date

**WorkerHour**

**Id** – המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

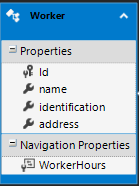
**Worker\_id** – מספר הסידורי של העובד שנוסף לו שעות - int

**Hours** – מספר השעות שהעובד עבד – int

**Salary** – משכורת לשעה של העובד בשעות עבודה אלו – float

**Date** – תאריך המשמרת – Date

**Total** – משכורת לשעה כפול שעות העבודה - float

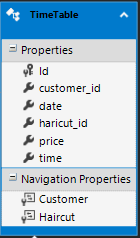
**Worker**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Name** – שם העובד - nvarchar(150)

**Identification** – תעודת זהות העובד - nvarchar(150)

**Address** – כתובת העובד - nvarchar(150)

**TimeTable**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

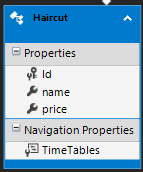
**Customer\_id** – מספר הסידורי של הלקוח שקבע תור - int

**Date** – תאריך שאליו הוזמן התור – date

**Haircut\_id** – מספר הסידורי של סוג התספורת – int

**Price** – מחיר התספורת – float

**Time** – זמן התור – Time

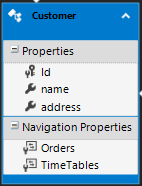


**Haircut**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Name** – שם התספורת - nvarchar(50)

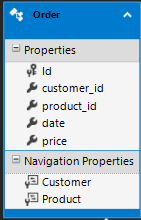
**Price** – מחיר הקבוע של התספורת – float

**Customer**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Name** – שם הלקוח - nvarchar(50)

**Address** – כתובת הלקוח - nvarchar(50)

**Order**

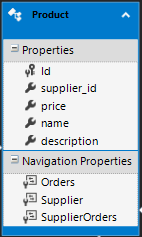
**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Customer\_id** – המספר הסידורי של הלקוח שהזמין מוצר – int

**Product\_id** – המספר הסידורי של המוצר שהלקוח הזמין – int

**Date** – תאריך שבו בוצעה ההזמנה – date

**Price** – מחיר ההזמנה – float



**Product**

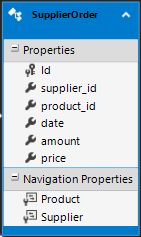
**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Supplier\_id** – המספר הסידורי של הספק המספק מוצר זה – Int

**Price** – מחיר המוצר – float

**Name** – שם המוצר - nvarchar(50)

**Description** – תיאור המוצר - nvarchar(150)

**SuppliersOrders**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

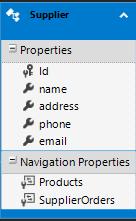
**Supplier\_id** – המספר הסידורי של הספק שמזמינים ממנו מלאי – int

**Product\_id** – המספר הסידורי של המוצר שמוזמן – int

**Date** – תאריך ההזמנה מהספק – date

**Amount** – כמות המוצר שהוזמן – int

**Price** – מחיר כולל של ההזמנה - float

**Supplier**

**Id** - המספר הסידורי, מספר שרירותי שנקבע אוטומתית ע"י המערכת – Int

**Name** – שם הספק - nvarchar(50)

**Address** – כתובת הספק - nvarchar(50)

**Phone** – טלפון הספק - nvarchar(50)

**Email** – אימייל הספק - nvarchar(50)

# **מדריך למשתמש**

# הפעלת התוכנה

כדי להפעיל את התוכנה על המשתמש ללחוץ על קיצור הדרך בשולחן העבודה בשם Gabi’s Barber Shop אשר הותקן על המחשב על ידי המפתח.

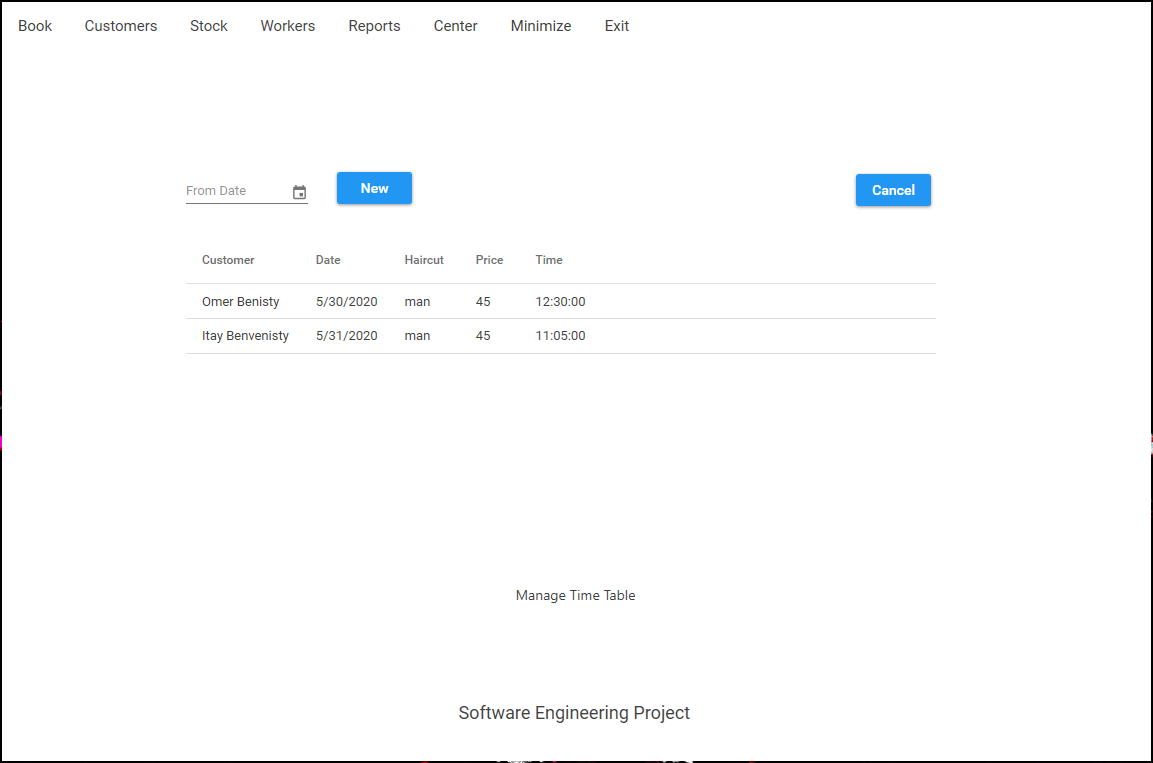
# בית



זהו דף הבית של התוכנה.

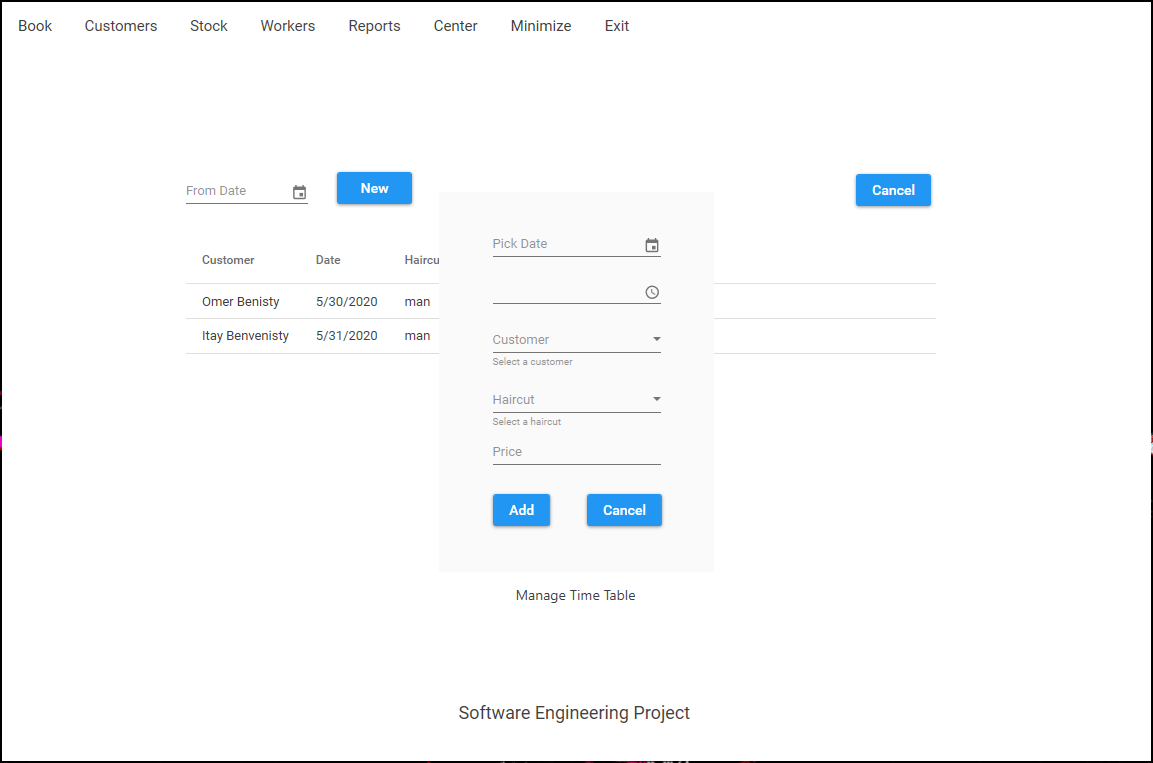
בעזרת הסרגל בראשית הדף המשתמש יוכל לגשת לכל אפשרויות הניהול האפשריות, כאשר נלחץ אחד מאפשרויות הסרגל נפתחות אפשרויות נוספות בהתאם לקטגורייה הנלחצת.

# ניהול תורים ולוח זמנים



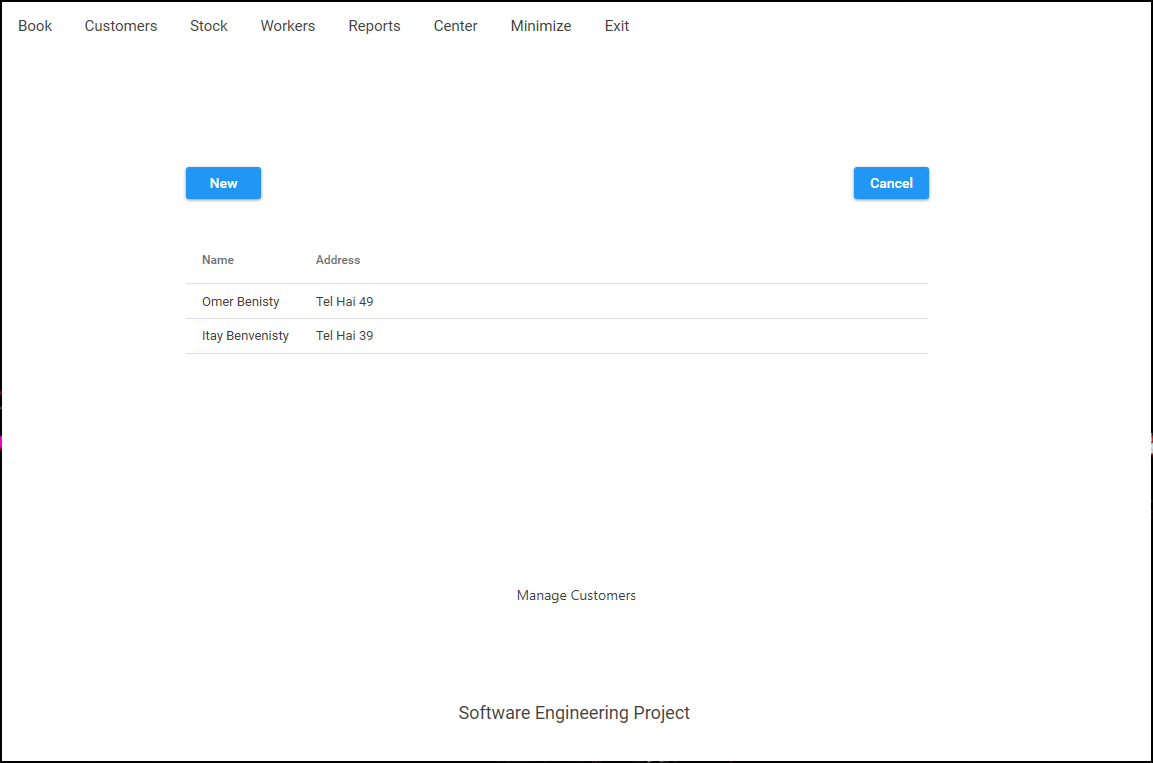
זהו חלון ניהול התורים, כאן ניתן לראות את התורים הקיימים במערכת, אפשר גם לסנן אותם מתאריך ספציפי עד התאריך הנוכחי. למחיקת תור או עריכת תור יש ללחוץ מקש ימני בעכבר על אחת התורים.

# קביעת תור



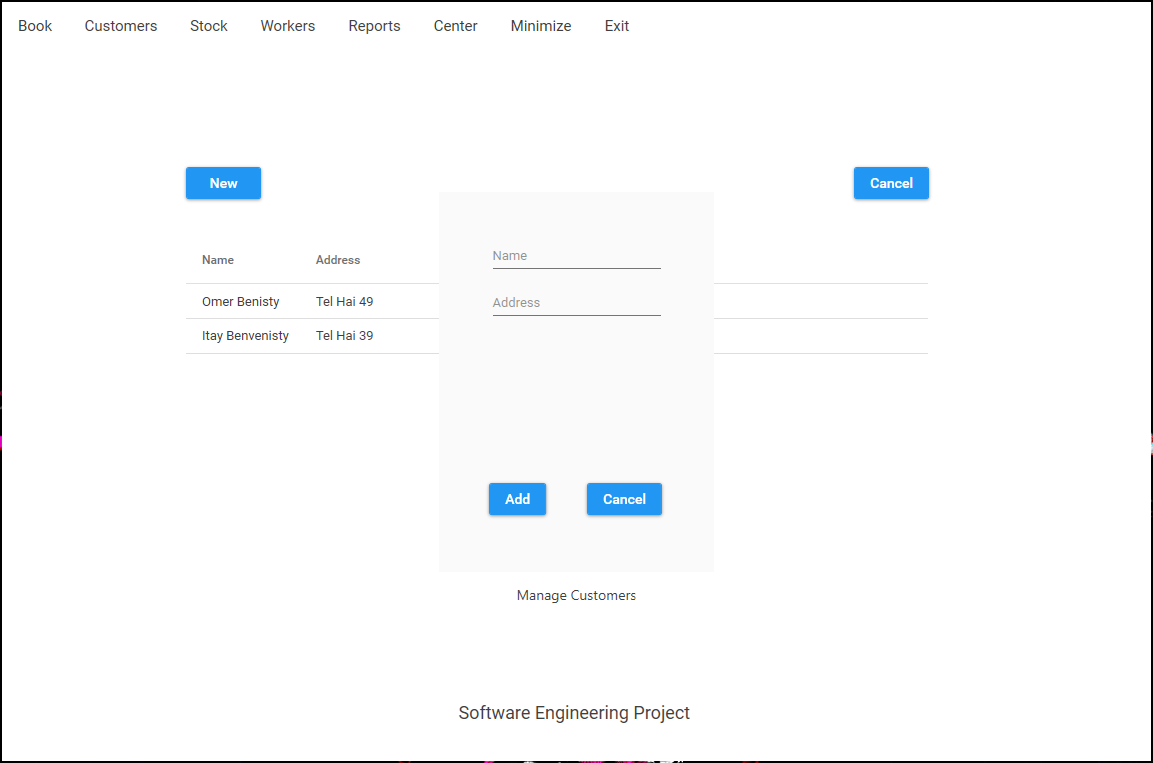
לאחר לחיצה על הכפתור new ניתן ליצור תור חדש, כאן נבחר את תאריך התור, זמן התור, הלקוח שהזמין את השירות, סוג התספורת ומחיר התספורת.

# ניהול לקוחות



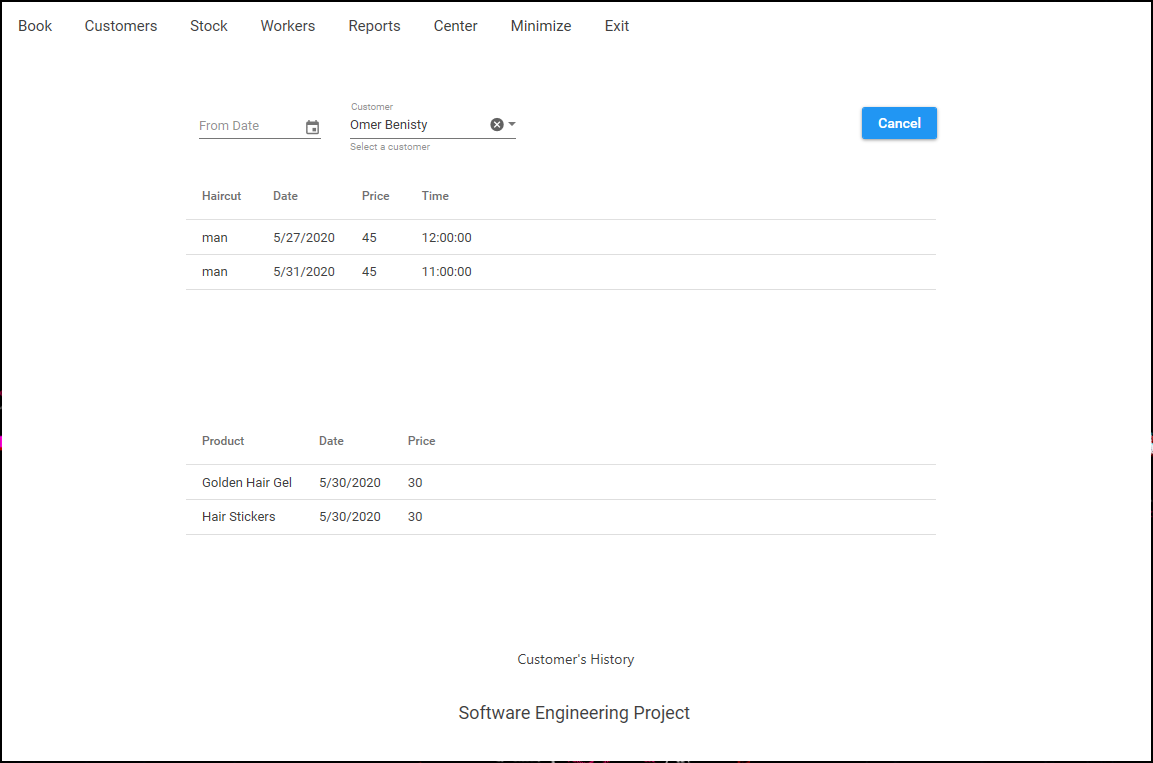
זהו חלון ניהול הלקוחות, כאן ניתן לנהל את פרטי הלקוחות שהשתמשו בשירות העסק, למחיקת לקוח או עריכת לקוח יש ללחוץ מקש ימני בעכבר על אחד הלקוחות.

# לקוח חדש



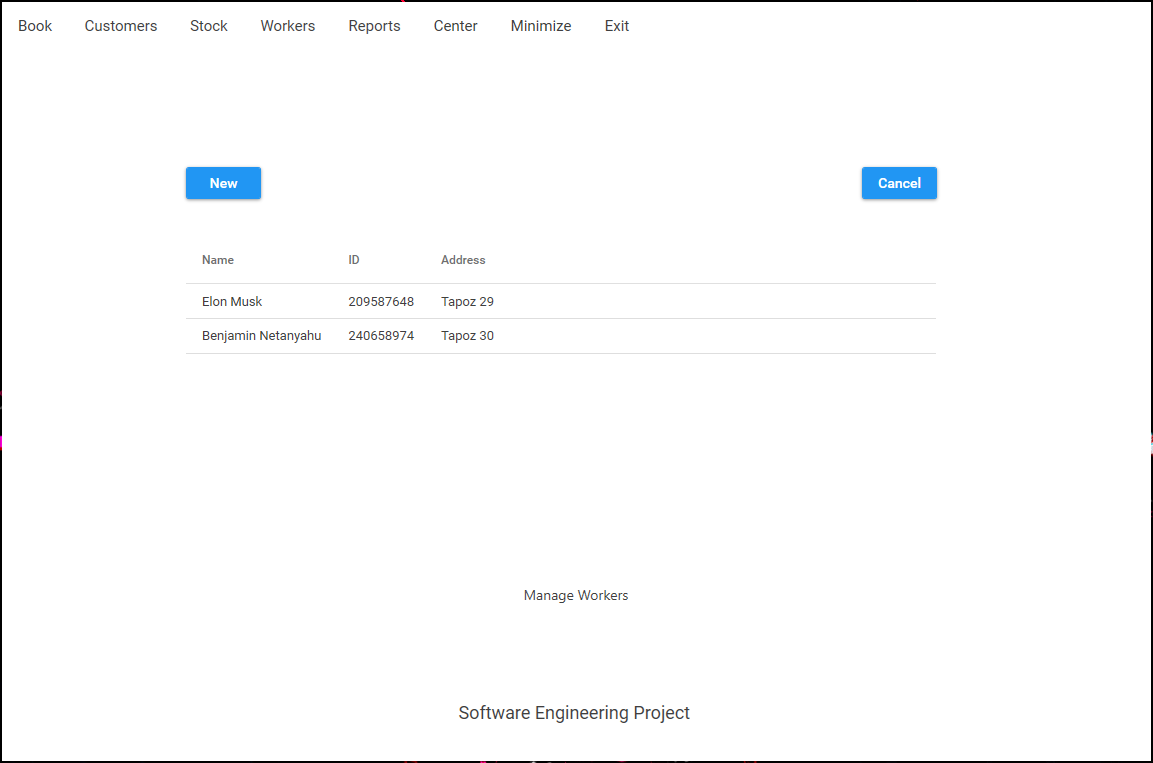
לאחר לחיצה על הפתור new ניתן ליצור לקוח חדש, כאן נבחר את שם הלקוח וכתובת הלקוח.

# היסטוריית לקוח



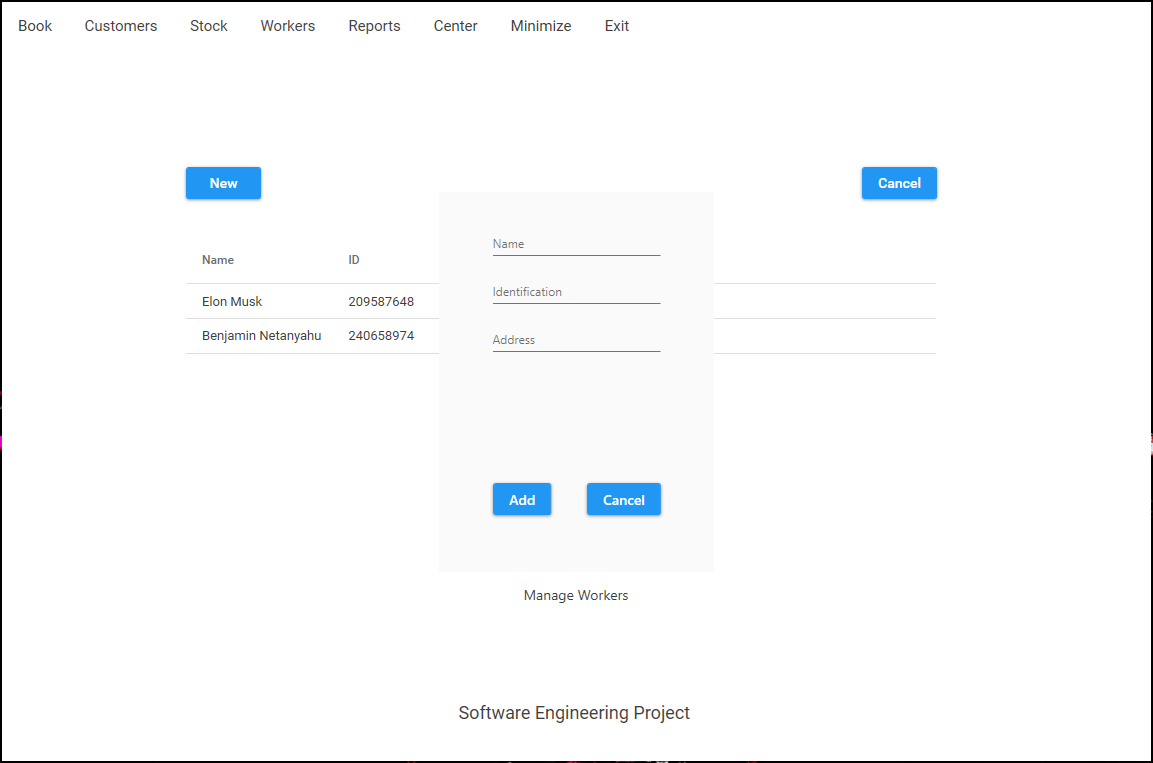
זהו חלון היסטוריית לקוח, כאן ניתן לצפות בהיסטוריית הלקוח הנבחר, יוצגו המוצרים שקנה הלקוח והתורים שקבע, אפשר בנוסף לסנן מתאריך ספציפי

# ניהול עובדים



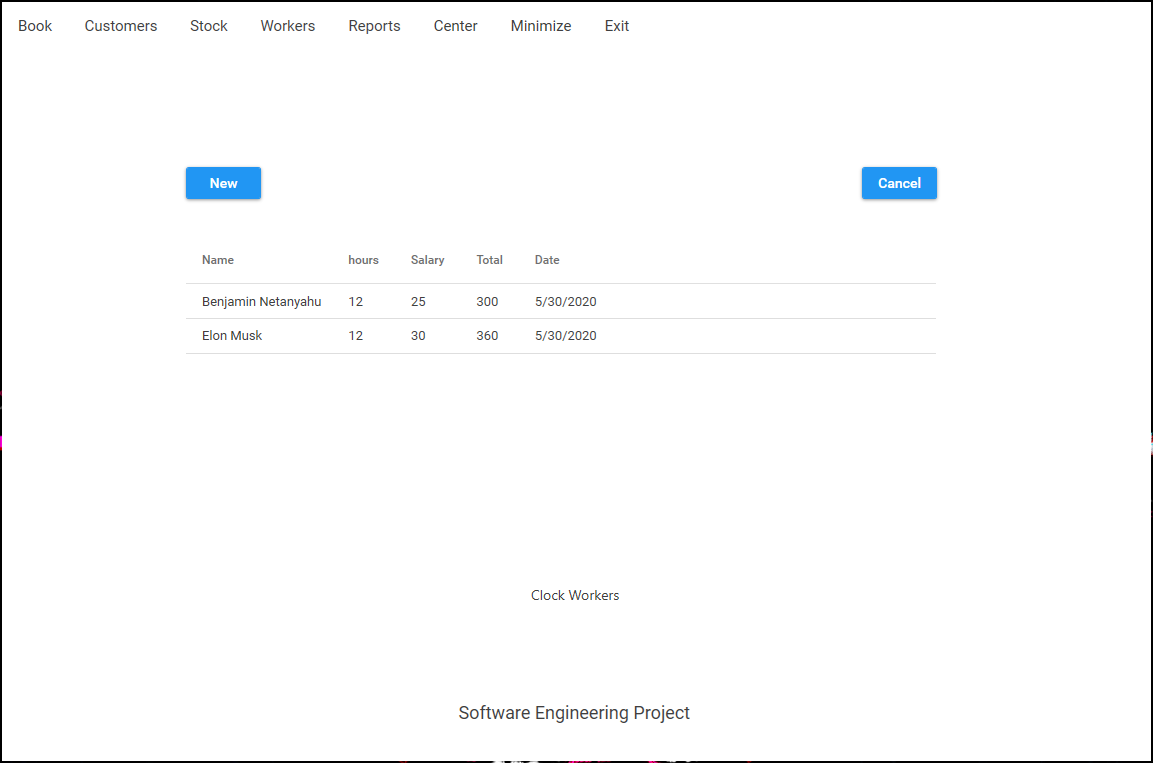
זהו חלון ניהול העובדים, כאן ניתן לראות את העובדים הכוללים בעסק, ניתן לערוך או מלחוק עובד על ידי לחיצת מקש ימני בעכבר על העובד ברשימה.

# יצירת עובד חדש



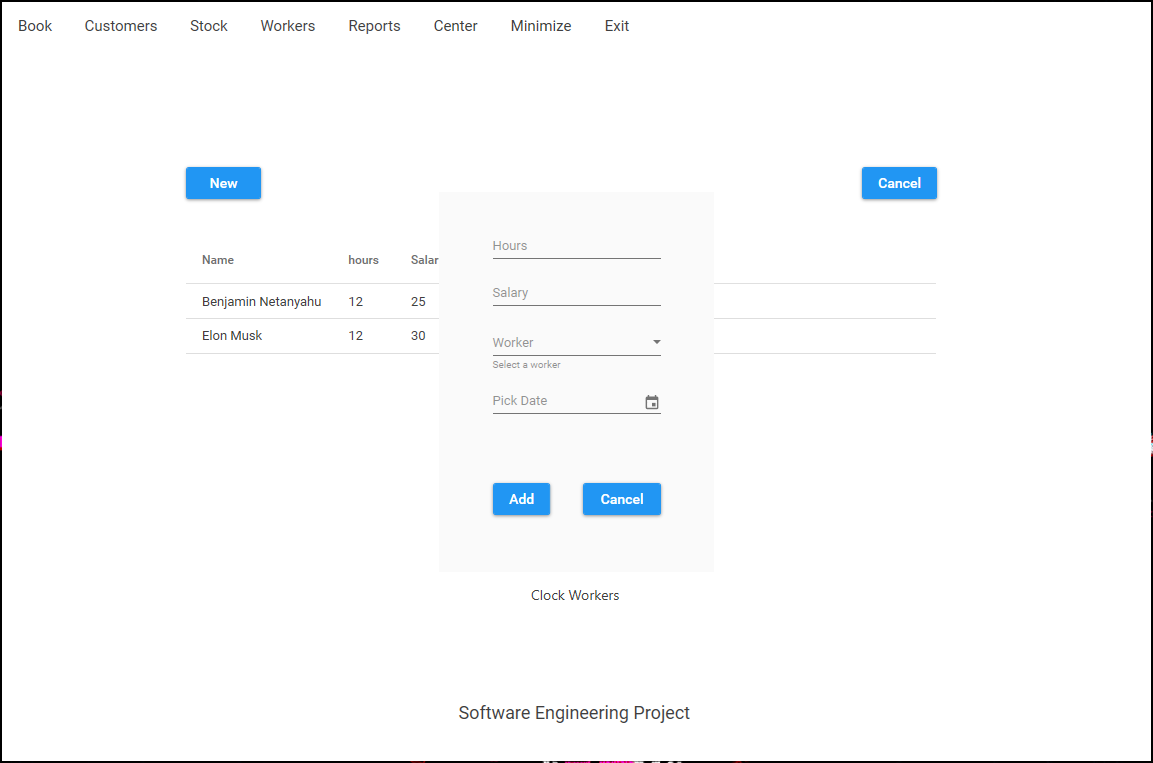
על ידי לחיצה על הכפתור New ניתן ליצור עובד חדש, כאן ניתן להזין את שם העובד, תעודת זהות העובד וכתובת המגורים של העובד.

# שעות העבודה של עובד



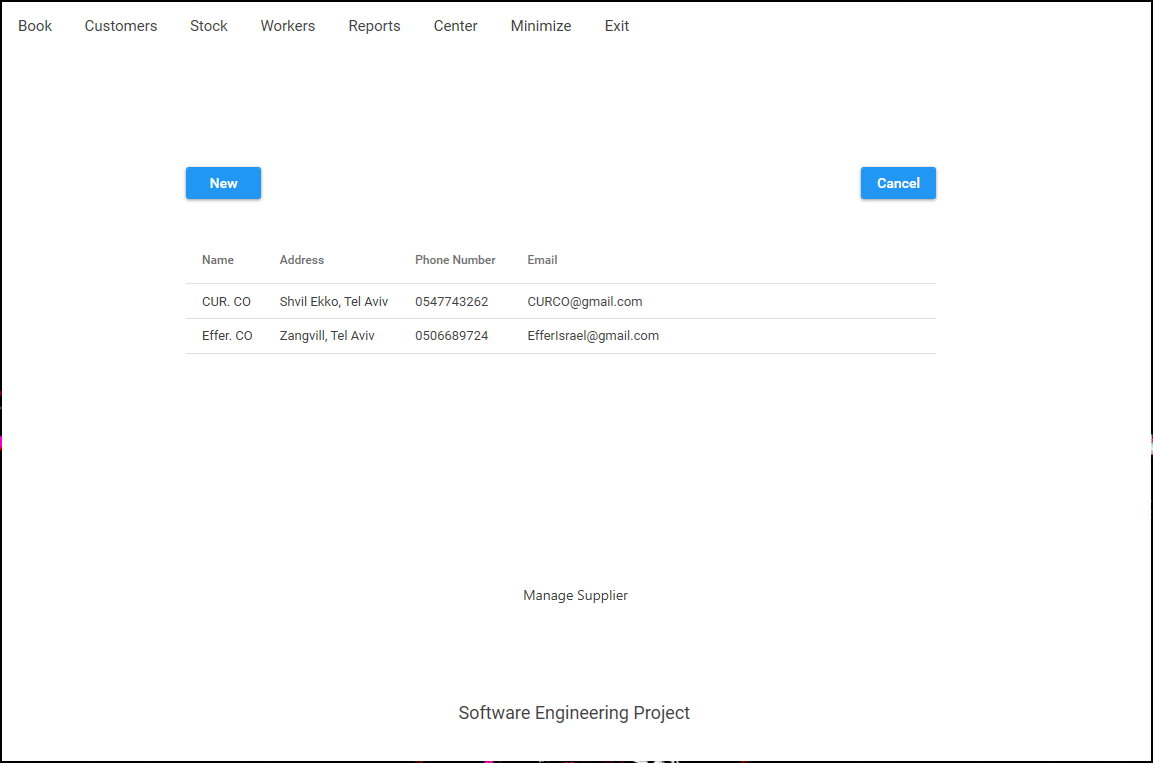
בחלון זה ניתן לראות את דיווח העובדים על שעות העבודה שלהם, כאן מצויין את סכום הכסף שהרוויחו בסך שעות העבודה שהכניסו במשמרת מסויימת בתאריך מסוים, ניתן לערוך את פרטי המשמרת על ידי לחיצת מקש ימני בעכבר על המשמרת הרצויה

# דיווח משמרת חדשה



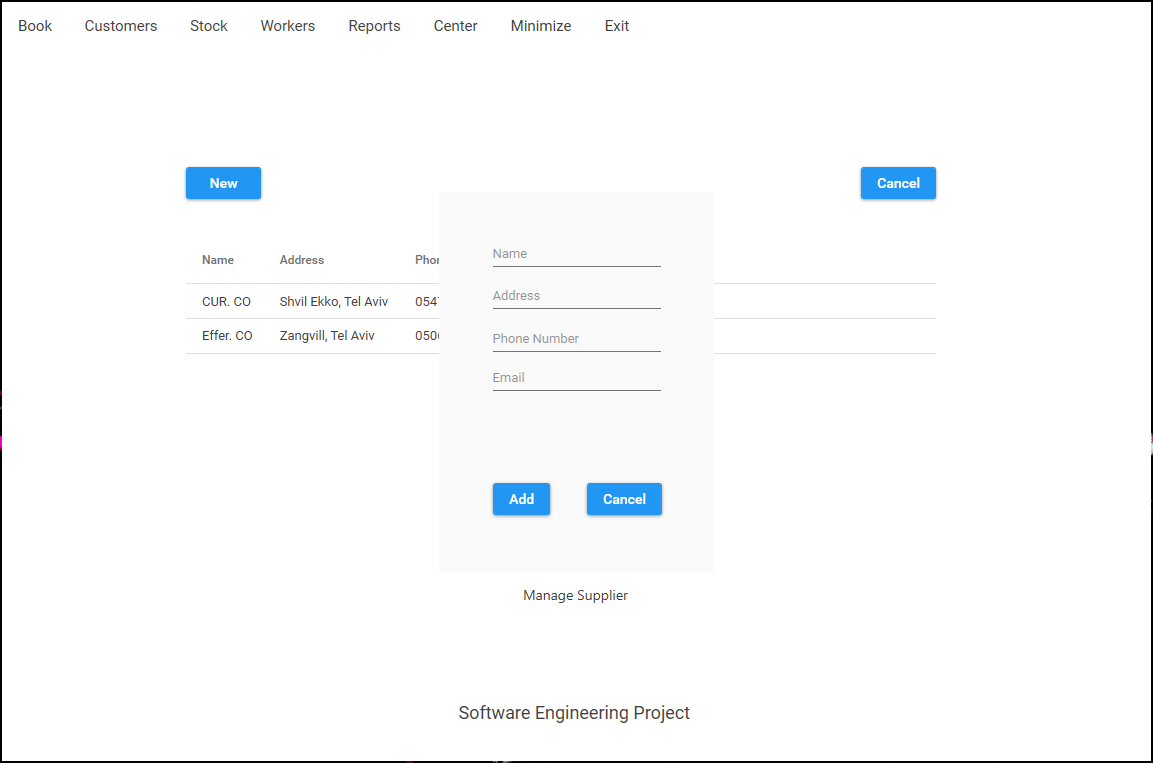
על ידי לחיצה על הכפתור New ניתן לדווח על משמרת חדשה, כאן ניתן להזין את כמות השעות שהעובד עבד, משכורתו לשעה, העובד המצוין ותאריך המשמרת, לאחר מכן יתחשב הסכום הכולל של המשמרת.

# ניהול ספקים



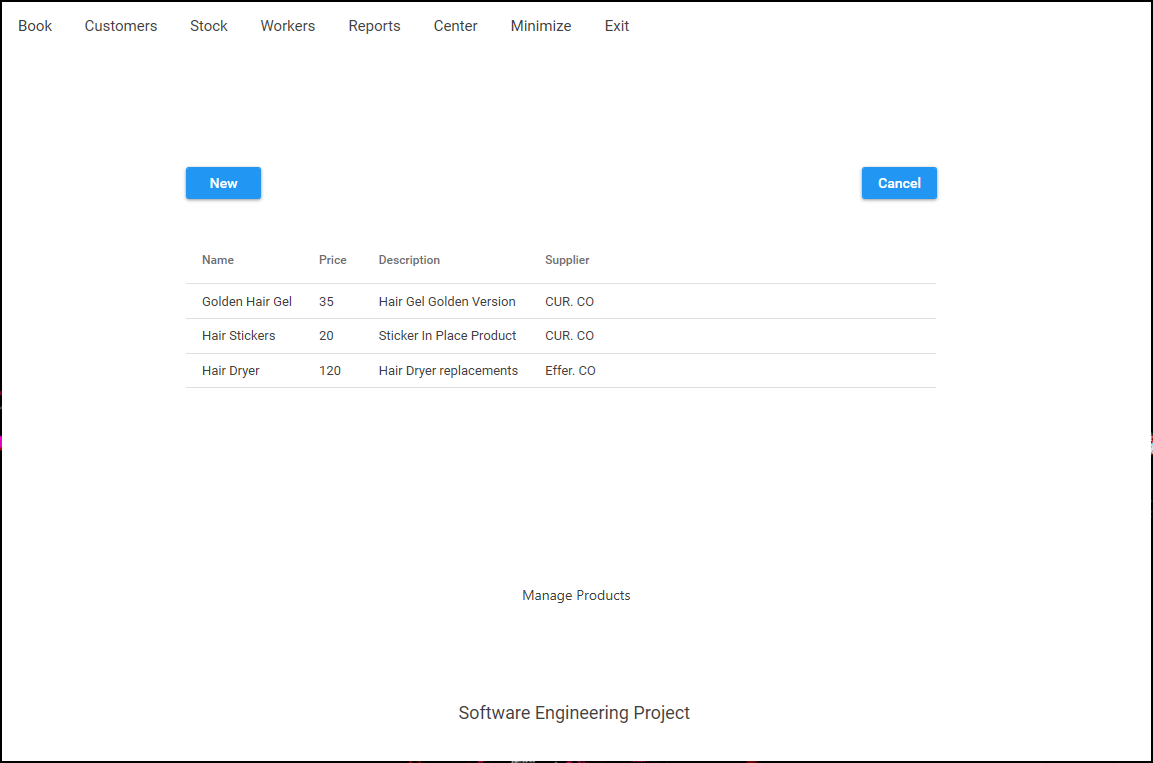
זהו חלון ניהול הספקים, כאן ניתן לנהל את פרטי הספקים הקשורים לעסק, ניתן לערוך או למחוק ספק על ידי לחיצת מקש ימני בעכבר על הספק המבוקש.

# יצירת ספק חדש



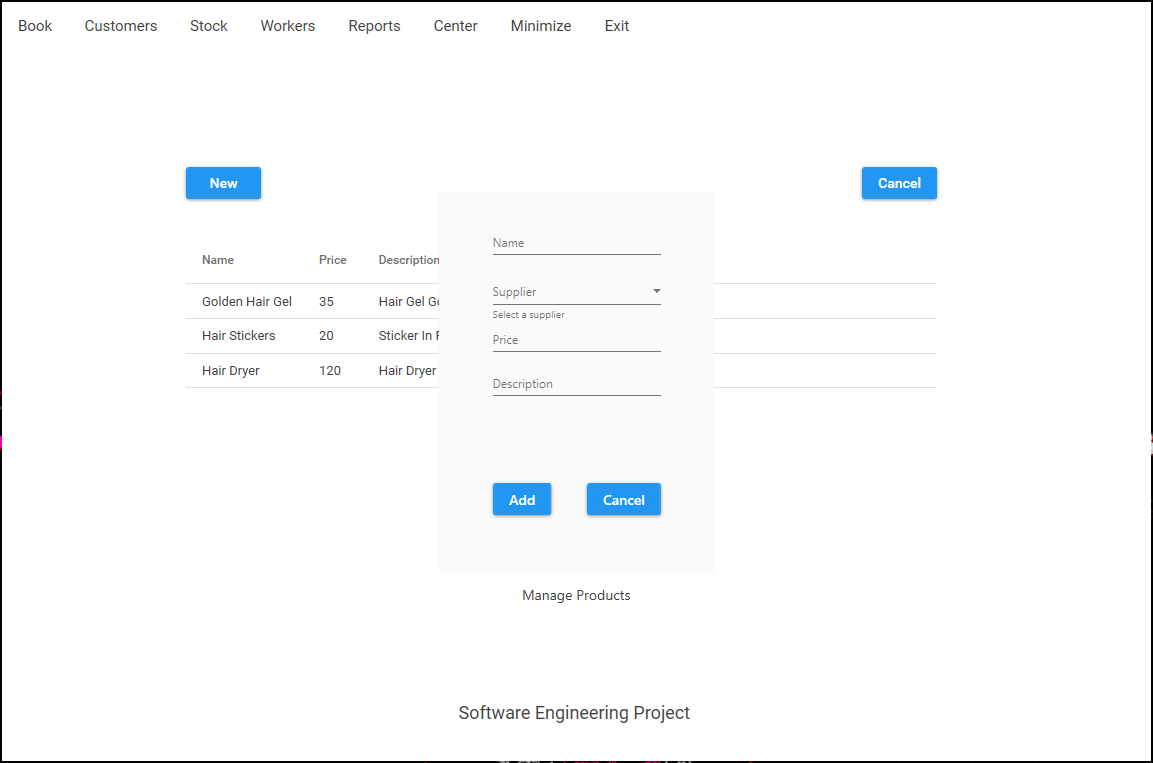
על ידי לחיצה על המקש New ניתן ליצור ספק חדש, כאן ניתן להכניס את שם הספק, כתובת הספק, מספר טלפון הספק ואימייל הספק.

# עריכת רשימת המוצרים



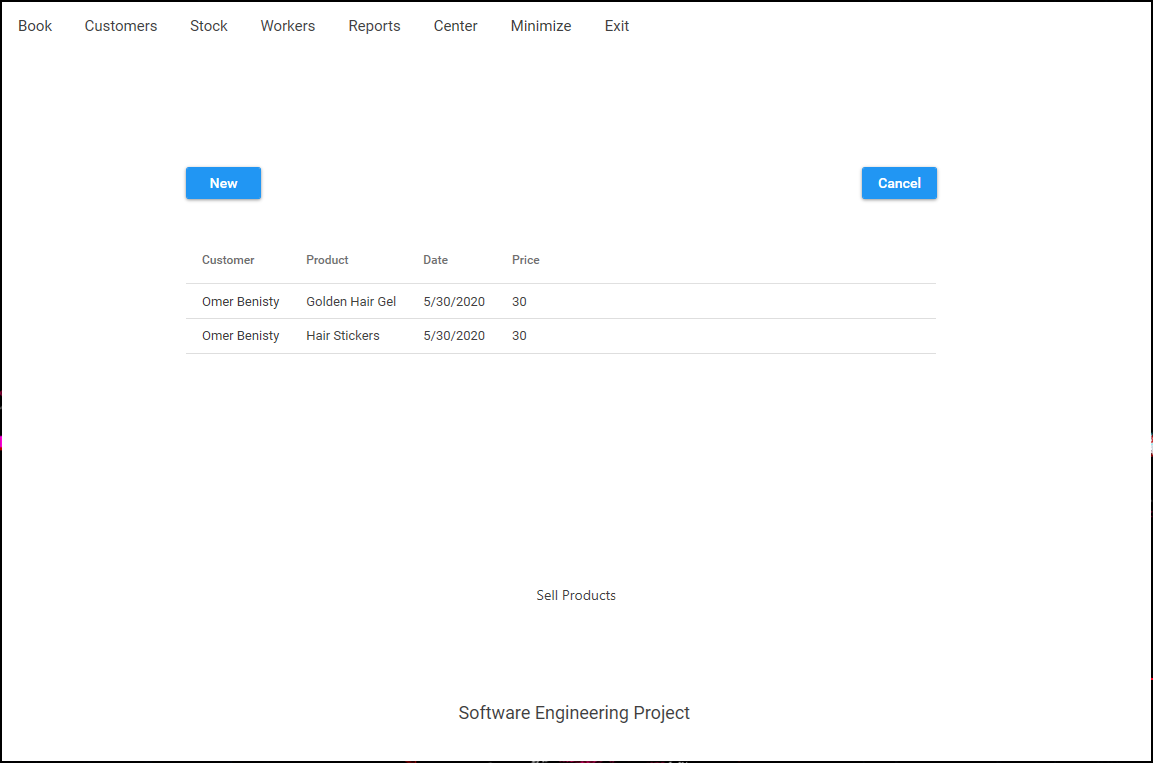
זהו חלון ניהול רשימת המוצרים, כאן ניתן לראות את כל סוגי המוצרים שהעסק מזמין, את הספק המספק אותם ואת מחירם שהעסק מוחר, ניתן לערוך או למחוק פרטי מוצר על ידי לחיצת מקש ימני על המוצר המבוקש.

# יצירת מוצר חדש



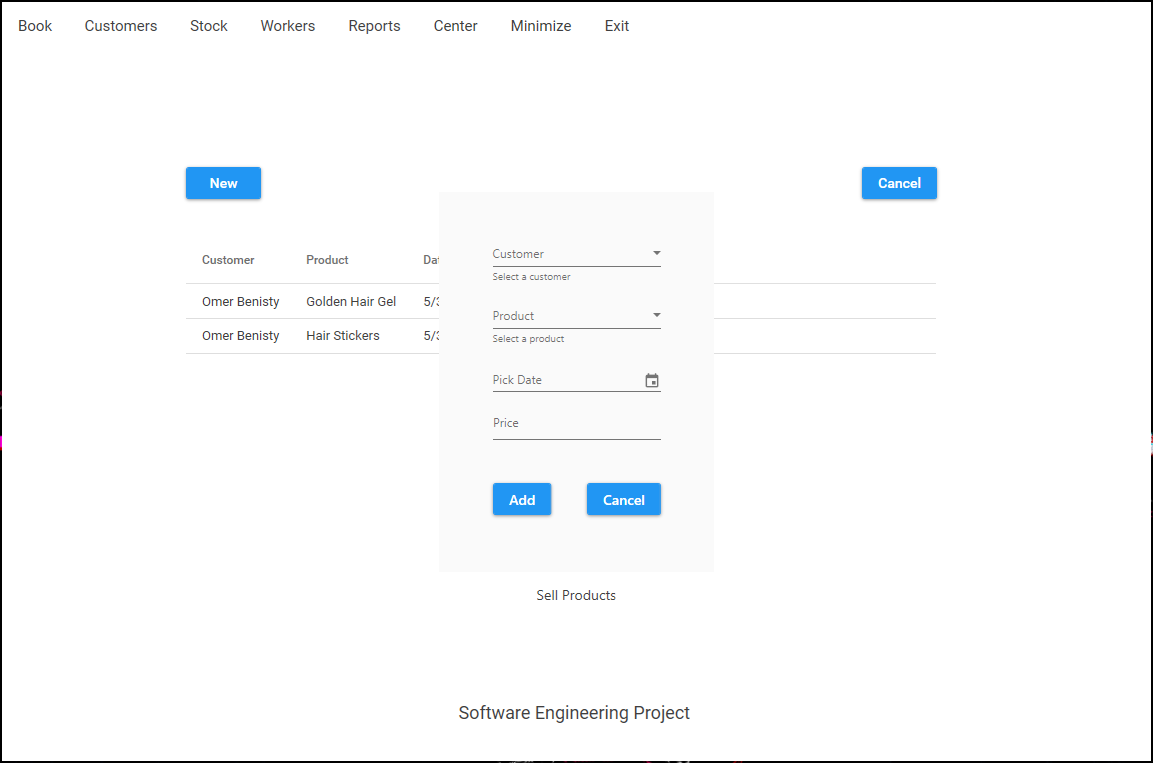
על ידי לחיצה על המקש New ניתן ליצור מוצר חדש, כאן ניתן להכניס את שם המוצר, הספק המספק את המוצר, מחיר המוצר ותיאור המוצר.

# מכירת מוצרים ללקוח



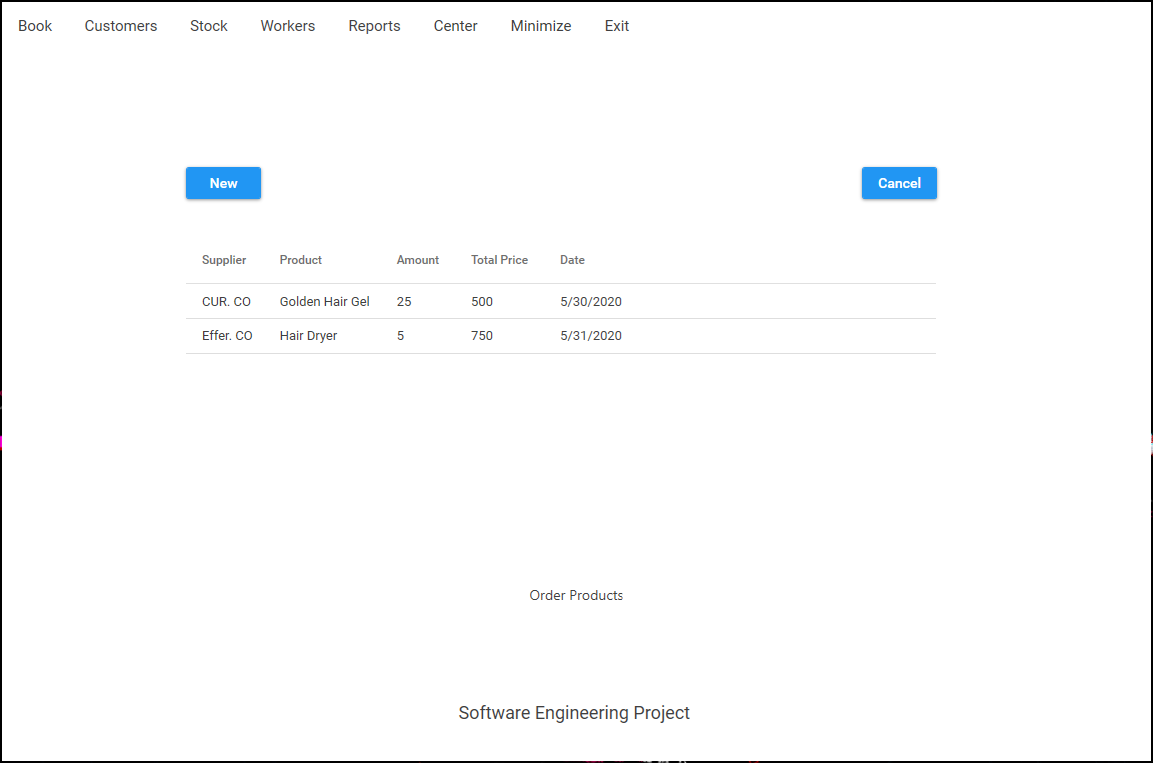
כאן ניתן לראות את המוצרים הנמכרו בעסק, מצוין הלקוח שקנה את המוצר, המוצר שהוא קנה, תאריך הקנייה ומכיר הקנייה, ניתן לערוך או למחוק פרטי מכירה על ידי לחיצת מקש ימני אל המכירה הרצויה.

# מכירה חדשה



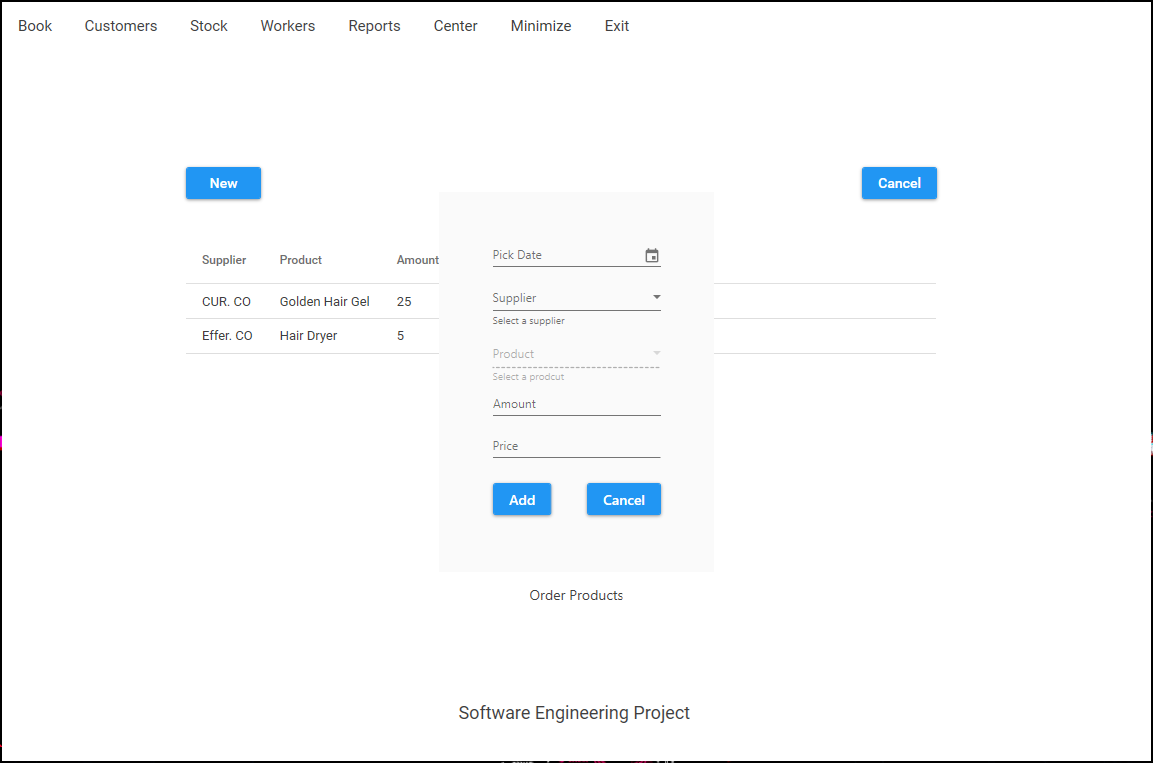
על ידי לחיצה על הכפתור New ניתן לבצע מכירה חדשה, כאן ניתן לציין את הלקוח שקנה את המוצר, המוצר שקנה, תאריך הקנייה וסכום הקנייה

# הזמנת מוצרים לעסק



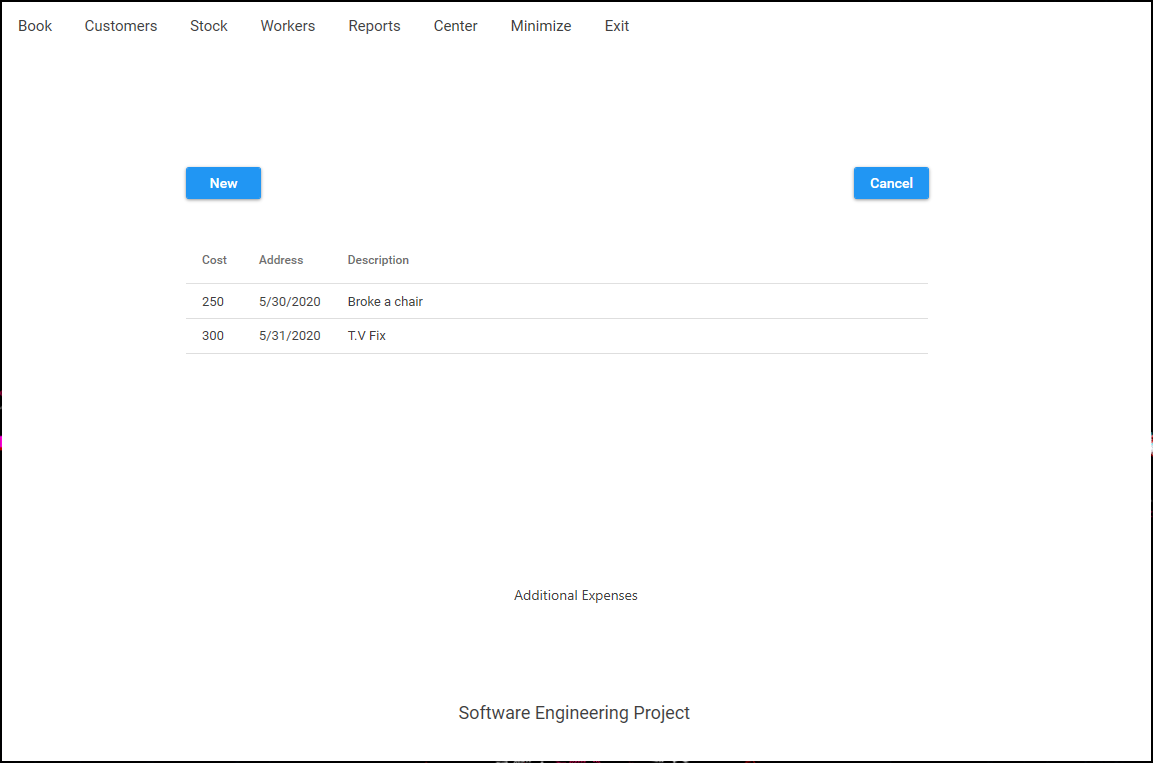
כאן ניתן להראות את ההזמנות ההספקה לעסק, מצוין הספק שסיפק את המוצר, כמות המוצרים שנקנו, סוג המוצר שנקנה, תאריך הקנייה וסכום הכולל של ההזמנה, ניתן לערוך או למחוק הזמנה על ידי לחיצת מקש ימני על פרטי הזמנה.

# הזמנה חדשה



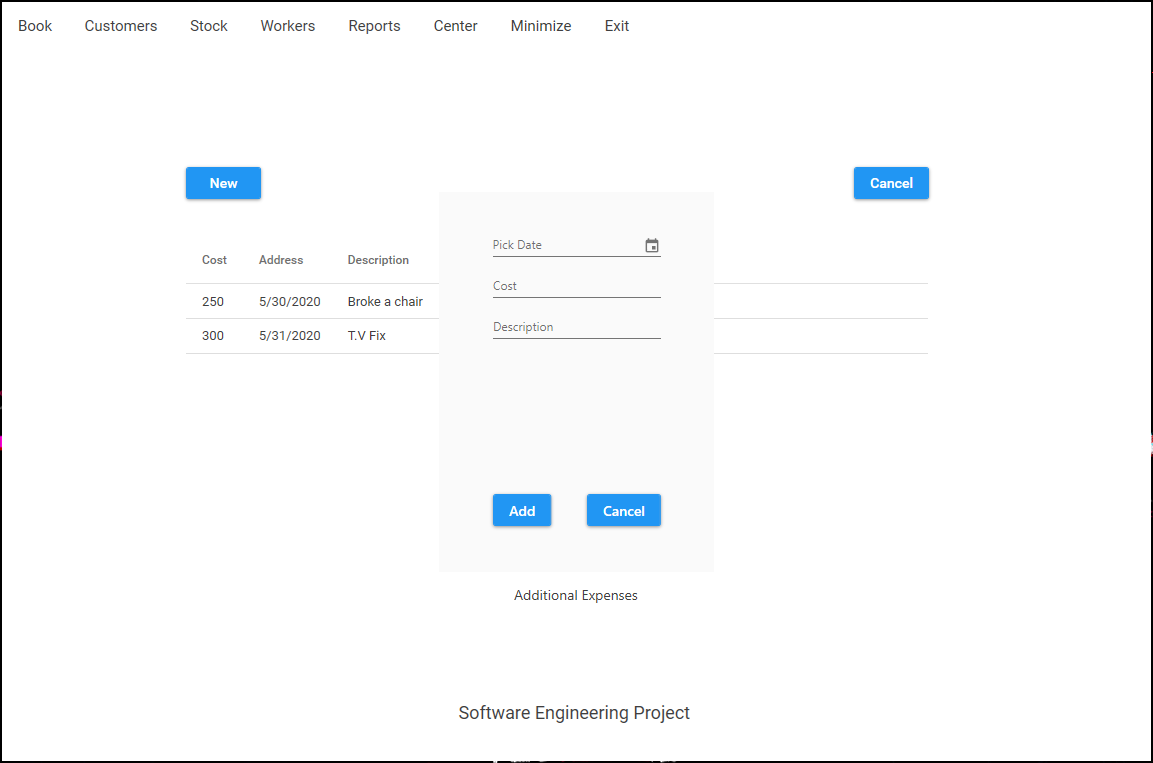
על ידי לחיצה על הכפתור New ניתן לבצע הזמנה חדשה של מוצרים לעסק, כאן ניתן למלות את תאריך ההזמנה, הספק שממנו נקנה ההזמנה, המוצר שהוזמן, כמות המוצרים ומחיר כל ההזמנה

# הוצאות נוספות



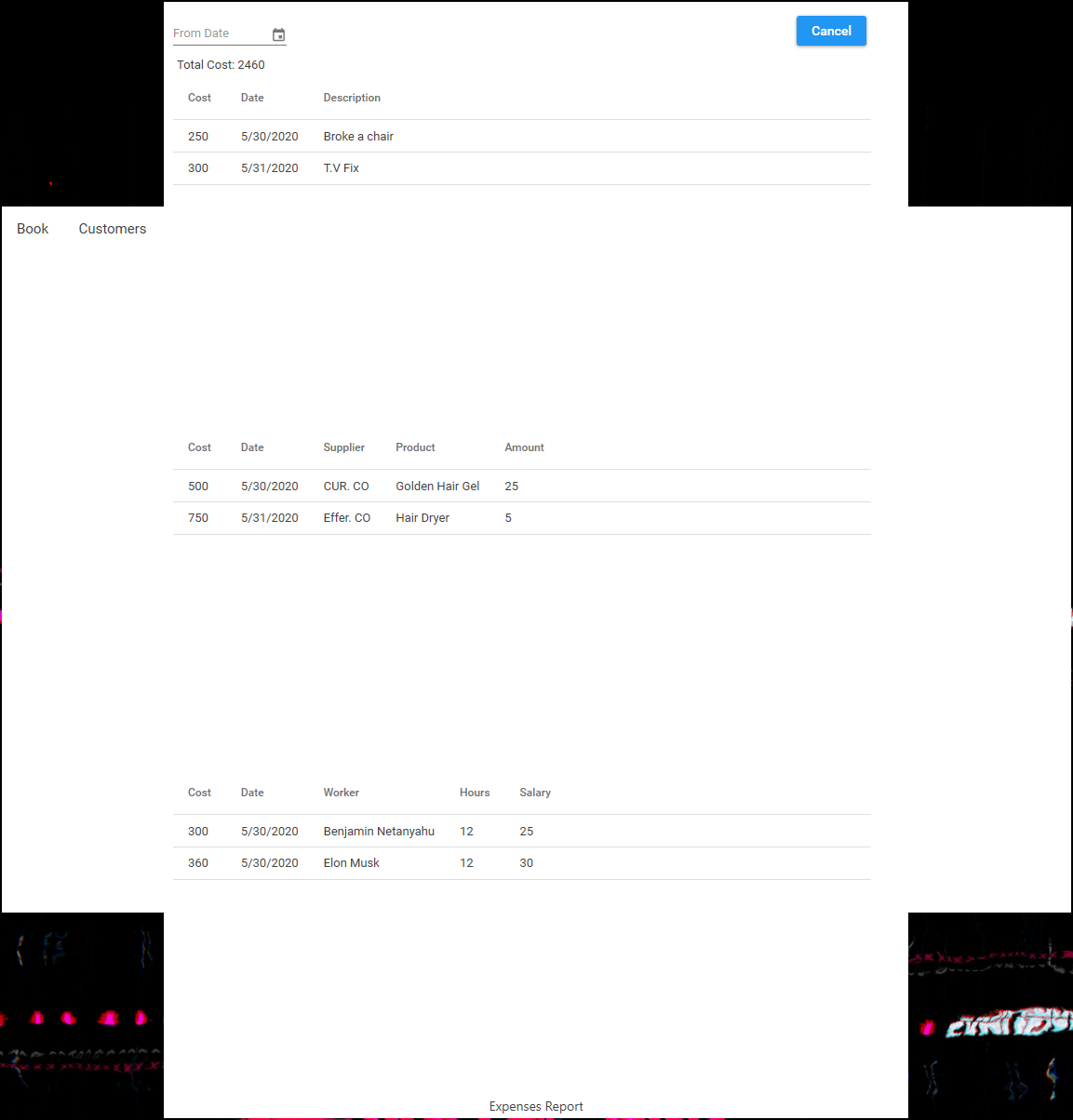
זהו חלון דיווח הוצאות נוספות, כאן ניתן לראות את מחיר ההוצאה, תאריך ההוצאה ותיאור ההוצאה, על מנת לערוך או למחוק הוצאה יש ללחוץ מקש ימני בעכבר על הדיווח הרצוי.

# הוצאה לא צפויה חדשה



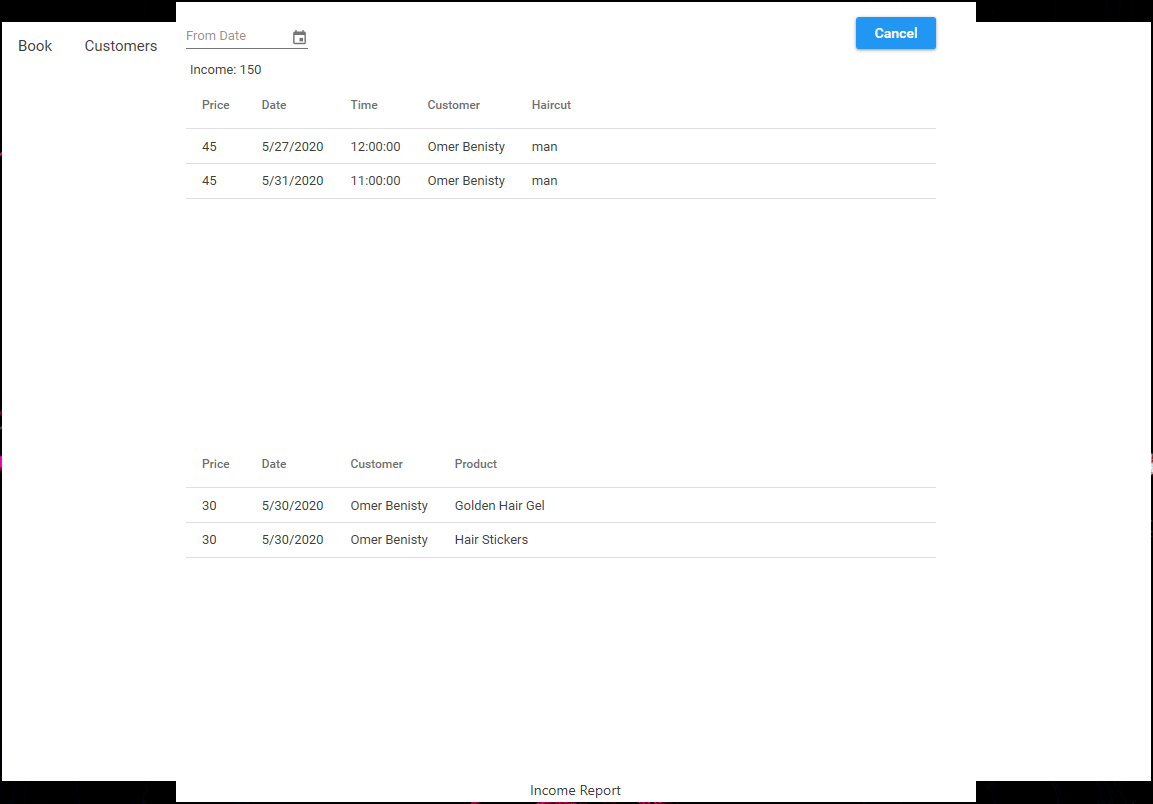
על ידי לחיצה על הכפתור New ניתן לדווח על הוצאה חדשה, כאן ניתן להכניס את תאריך ההוצאה, מחיר ההוצאה ותיאור ההוצאה.

# דוח הוצאות



כאן ניתן לראות את כל ההוצאות שהעסק קיים מהתאריך המבוקש, דוח זה כולל הוצאות מהוצאות לא צפויות, קניית מוצרים ומשכורת לעובדים.

# דוח הכנסות



כאן ניתן לראות את כל ההכנסות שהעסק קיבל, דוח זה מציג את כמות הכסף שנכנסה לעסק מתורים וקניית מוצרים.

# **מדריך למפתח**

# Controller.cs

using System;

using System.CodeDom;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows;

namespace BarberShop

{

    class Controller

    {

        DataModel model = new DataModel();

        Regex r = new Regex(@"^[A-Z0-9a-z. ,@]{1,50}$");

        public void ValidateCustomer(string name, string address)

        {

            if (!r.IsMatch(name))

                throw new System.ArgumentException("Name is too long or invalid");

            if (!r.IsMatch(address))

                throw new System.ArgumentException("Address is too long or invalid");

        }

        public void ValidateSupplier(string name, string address, string phone, string email)

        {

            if (!r.IsMatch(name))

                throw new System.ArgumentException("Name is too long or invalid");

            if (!r.IsMatch(address))

                throw new System.ArgumentException("Address is too long or invalid");

            if (!r.IsMatch(phone))

                throw new System.ArgumentException("Phone is too long or invalid");

            if (!r.IsMatch(email))

                throw new System.ArgumentException("Email is too long or invalid");

        }

        public void ValidateWorker(string name, string id, string address)

        {

            if (!r.IsMatch(name))

                throw new System.ArgumentException("Name is too long or invalid");

            if (!r.IsMatch(id))

                throw new System.ArgumentException("ID is too long or invalid");

            if (!r.IsMatch(address))

                throw new System.ArgumentException("Address is too long or invalid");

        }

        public void ValidateBook(double money)

        {

            if (money < 0 || money > double.MaxValue)

                throw new System.ArgumentException("Money number is too large or negetive");

        }

        public void ValidateProduct(string name, double price, string description)

        {

            if (!r.IsMatch(name))

                throw new System.ArgumentException("Name is too long or invalid");

            if (price < 0 || price > double.MaxValue)

                throw new System.ArgumentException("Price number is too large or negetive");

            if (!r.IsMatch(description))

                throw new System.ArgumentException("Description is too long or invalid");

        }

        public void ValidateWorkerHours(double hours, double salary, double total)

        {

            if (hours < 0 || hours > double.MaxValue)

                throw new System.ArgumentException("Salary number is too large or negetive");

            if (salary < 0 || salary > double.MaxValue)

                throw new System.ArgumentException("Salary number is too large or negetive");

            if (total < 0 || total > double.MaxValue)

                throw new System.ArgumentException("Total Salary number is too large or negetive");

        }

        public void ValidateSupplierOrder(int amount, double price)

        {

            if (amount < 0 || amount > int.MaxValue)

                throw new System.ArgumentException("Amount number is too large or negetive");

            if (price < 0 || price > double.MaxValue)

                throw new System.ArgumentException("Price number is too large or negetive");

        }

        public void ValidateOrder(double price)

        {

            if (price < 0 || price > double.MaxValue)

                throw new System.ArgumentException("Price number is too large or negetive");

        }

        public void ValidateAdditionalCost(double cost, string description)

        {

            if (cost < 0 || cost > double.MaxValue)

                throw new System.ArgumentException("Cost number is too large or negetive");

            if (!r.IsMatch(description))

                throw new System.ArgumentException("Description is too long or invalid");

        }

        public void AddCustomer(string name, string address)

        {

            ValidateCustomer(name, address);

            Customer c = new Customer();

            c.name = name;

            c.address = address;

            model.AddCustomer(c);

        }

        public void AddSupplier(string name, string address, string phone, string email)

        {

            ValidateSupplier(name, address, phone, email);

            Supplier s = new Supplier();

            s.name = name;

            s.address = address;

            s.phone = phone;

            s.email = email;

            model.AddSupplier(s);

        }

        public void AddBook(double money, DateTime date, int type, int customer, TimeSpan time)

        {

            ValidateBook(money);

            TimeTable t = new TimeTable();

            t.customer\_id = customer;

            t.date = date;

            t.haricut\_id = type;

            t.price = money;

            t.time = time;

            model.AddBook(t);

        }

        public void AddWorker(string name, string id, string address)

        {

            ValidateWorker(name, id, address);

            Worker w = new Worker();

            w.name = name;

            w.identification = id;

            w.address = address;

            model.AddWorker(w);

        }

        public void AddProduct(string name, double price, string description, int supplier)

        {

            ValidateProduct(name, price, description);

            Product p = new Product();

            p.name = name;

            p.price = price;

            p.description = description;

            p.supplier\_id = supplier;

            model.AddProduct(p);

        }

        public void AddWorkerHour(int worker, double hours, double salary, DateTime date, double total)

        {

            ValidateWorkerHours(hours, salary, total);

            WorkerHour w = new WorkerHour();

            w.worker\_id = worker;

            w.hours = hours;

            w.salary = salary;

            w.date = date;

            w.total = total;

            model.AddWorkerHour(w);

        }

        public void AddSupplierOrder(int supplier, int product, DateTime date, int amount, double price)

        {

            ValidateSupplierOrder(amount, price);

            SupplierOrder s = new SupplierOrder();

            s.supplier\_id = supplier;

            s.product\_id = product;

            s.date = date;

            s.amount = amount;

            s.price = price;

            model.AddSupplierOrder(s);

        }

        public void AddOrder(int customer, int product, DateTime date, double price)

        {

            ValidateOrder(price);

            Order o = new Order();

            o.customer\_id = customer;

            o.product\_id = product;

            o.date = date;

            o.price = price;

            model.AddOrder(o);

        }

        public void AddAdditionalCost(double cost, string description, DateTime date)

        {

            ValidateAdditionalCost(cost, description);

            AdditionalCost a = new AdditionalCost();

            a.cost = cost;

            a.description = description;

            a.date = date;

            model.AddAdditionalCost(a);

        }

        public void EditBook(TimeTable t)

        {

            ValidateBook(t.price);

            model.EditBook(t);

        }

        public void EditSupplier(Supplier s)

        {

            ValidateSupplier(s.name, s.address, s.phone, s.email);

            model.EditSupplier(s);

        }

        public void EditWorker(Worker w)

        {

            ValidateWorker(w.name, w.identification, w.address);

            model.EditWorker(w);

        }

        public void EditCustomer(Customer c)

        {

            ValidateCustomer(c.name, c.address);

            model.EditCustomer(c);

        }

        public void EditProduct(Product p)

        {

            ValidateProduct(p.name, p.price, p.description);

            model.EditProduct(p);

        }

        public void EditWorkerHour(WorkerHour w)

        {

            ValidateWorkerHours(w.hours, w.salary, w.total);

            model.EditWorkerHour(w);

        }

        public void EditSupplierOrder(SupplierOrder s)

        {

            ValidateSupplierOrder(s.amount, s.price);

            model.EditSupplierOrder(s);

        }

        public void EditOrder(Order o)

        {

            ValidateOrder(o.price);

            model.EditOrder(o);

        }

        public void EditAdditionalCost(AdditionalCost a)

        {

            ValidateAdditionalCost(a.cost, a.description);

            model.EditAdditionalCost(a);

        }

        public List<Customer> GetCustomers()

        {

            return model.GetCustomers();

        }

        public List<TimeTable> GetTimeTable()

        {

            return model.GetTimeTable();

        }

        public List<Haircut> GetHaircuts()

        {

            return model.GetHaircuts();

        }

        public List<SupplierOrder> GetSupplierOrders()

        {

            return model.GetSupplierOrders();

        }

        public List<Supplier> GetSuppliers()

        {

            return model.GetSuppliers();

        }

        public List<Worker> GetWorkers()

        {

            return model.GetWorkers();

        }

        public List<Product> GetProducts()

        {

            return model.GetProducts();

        }

        public List<WorkerHour> GetWorkerHours()

        {

            return model.GetWorkerHours();

        }

        public List<Order> GetOrders()

        {

            return model.GetOrders();

        }

        public List<AdditionalCost> GetAdditionalCosts()

        {

            return model.GetAdditionalCosts();

        }

        public void DeleteCustomer(Customer c)

        {

            model.DeleteCustomer(c);

        }

        public void DeleteTimeTable(TimeTable t)

        {

            model.DeleteTimeTable(t);

        }

        public void DeleteSupplier(Supplier s)

        {

            model.DeleteSupplier(s);

        }

        public void DeleteWorker(Worker w)

        {

            model.DeleteWorker(w);

        }

        public void DeleteProduct(Product p)

        {

            model.DeleteProduct(p);

        }

        public void DeleteWorkerHour(WorkerHour w)

        {

            model.DeleteWorkerHour(w);

        }

        public void DeleteSupplierOrder(SupplierOrder s)

        {

            model.DeleteSupplierOrder(s);

        }

        public void DeleteOrder(Order o)

        {

            model.DeleteOrder(o);

        }

        public void DeleteAdditionalCosts(AdditionalCost a)

        {

            model.DeleteAdditionalCost(a);

        }

    }

}

# DataModel.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BarberShop

{

    class DataModel

    {

        DatabaseEntities db = new DatabaseEntities();

        public void EditBook(TimeTable t)

        {

            var update = db.TimeTables.Where(o => o.Id == t.Id).FirstOrDefault();

            if (update != null)

            {

                update.price = t.price;

                update.date = t.date;

                update.haricut\_id = t.haricut\_id;

                update.customer\_id = t.customer\_id;

                update.time = t.time;

                update.Customer = t.Customer;

                update.Haircut = t.Haircut;

            }

            db.SaveChanges();

        }

        public void EditSupplier(Supplier s)

        {

            var update = db.Suppliers.Where(o => o.Id == s.Id).FirstOrDefault();

            if (update != null)

            {

                update.name = s.name;

                update.address = s.address;

                update.phone = s.phone;

                update.email = s.email;

            }

            db.SaveChanges();

        }

        public void EditWorker(Worker w)

        {

            var update = db.Workers.Where(o => o.Id == w.Id).FirstOrDefault();

            if (update != null)

            {

                update.name = w.name;

                update.identification = w.identification;

                update.address = w.address;

            }

            db.SaveChanges();

        }

        public void EditCustomer(Customer c)

        {

            var update = db.Customers.Where(o => o.Id == c.Id).FirstOrDefault();

            if (update != null)

            {

                update.name = c.name;

                update.address = c.address;

            }

            db.SaveChanges();

        }

        public void EditProduct(Product p)

        {

            var update = db.Products.Where(o => o.Id == p.Id).FirstOrDefault();

            if (update != null)

            {

                update.supplier\_id = p.supplier\_id;

                update.Supplier = p.Supplier;

                update.price = p.price;

                update.name = p.name;

                update.description = p.description;

            }

            db.SaveChanges();

        }

        public void EditWorkerHour(WorkerHour w)

        {

            var update = db.WorkerHours.Where(o => o.Id == w.Id).FirstOrDefault();

            if (update != null)

            {

                update.worker\_id = w.worker\_id;

                update.Worker = w.Worker;

                update.hours = w.hours;

                update.salary = w.salary;

                update.date = w.date;

                update.total = w.total;

            }

            db.SaveChanges();

        }

        public void EditSupplierOrder(SupplierOrder s)

        {

            var update = db.SupplierOrders.Where(o => o.Id == s.Id).FirstOrDefault();

            if (update != null)

            {

                update.supplier\_id = s.supplier\_id;

                update.product\_id = s.product\_id;

                update.date = s.date;

                update.amount = s.amount;

                update.price = s.price;

                update.Supplier = s.Supplier;

                update.Product = s.Product;

            }

            db.SaveChanges();

        }

        public void EditOrder(Order r)

        {

            var update = db.Orders.Where(o => o.Id == r.Id).FirstOrDefault();

            if (update != null)

            {

                update.customer\_id = r.customer\_id;

                update.product\_id = r.product\_id;

                update.Customer = r.Customer;

                update.Product = r.Product;

                update.date = r.date;

                update.price = r.price;

            }

            db.SaveChanges();

        }

        public void EditAdditionalCost(AdditionalCost a)

        {

            var update = db.AdditionalCosts.Where(o => o.Id == a.Id).FirstOrDefault();

            if (update != null)

            {

                update.cost = a.cost;

                update.description = a.description;

                update.date = a.date;

            }

            db.SaveChanges();

        }

        public void AddBook(TimeTable t)

        {

            db.TimeTables.Add(t);

            db.SaveChanges();

        }

        public void AddCustomer(Customer c)

        {

            db.Customers.Add(c);

            db.SaveChanges();

        }

        public void AddSupplier(Supplier s)

        {

            db.Suppliers.Add(s);

            db.SaveChanges();

        }

        public void AddWorker(Worker w)

        {

            db.Workers.Add(w);

            db.SaveChanges();

        }

        public void AddProduct(Product p)

        {

            db.Products.Add(p);

            db.SaveChanges();

        }

        public void AddWorkerHour(WorkerHour w)

        {

            db.WorkerHours.Add(w);

            db.SaveChanges();

        }

        public void AddSupplierOrder(SupplierOrder s)

        {

            db.SupplierOrders.Add(s);

            db.SaveChanges();

        }

        public void AddOrder(Order o)

        {

            db.Orders.Add(o);

            db.SaveChanges();

        }

        public void AddAdditionalCost(AdditionalCost a)

        {

            db.AdditionalCosts.Add(a);

            db.SaveChanges();

        }

        #region Get DB Tables as list

        public List<Customer> GetCustomers()

        {

            return db.Customers.ToList();

        }

        public List<TimeTable> GetTimeTable()

        {

            return db.TimeTables.ToList();

        }

        public List<Haircut> GetHaircuts()

        {

            return db.Haircuts.ToList();

        }

        public List<Supplier> GetSuppliers()

        {

            return db.Suppliers.ToList();

        }

        public List<SupplierOrder> GetSupplierOrders()

        {

            return db.SupplierOrders.ToList();

        }

        public List<Worker> GetWorkers()

        {

            return db.Workers.ToList();

        }

        public List<Product> GetProducts()

        {

            return db.Products.ToList();

        }

        public List<WorkerHour> GetWorkerHours()

        {

            return db.WorkerHours.ToList();

        }

        public List<Order> GetOrders()

        {

            return db.Orders.ToList();

        }

        public List<AdditionalCost> GetAdditionalCosts()

        {

            return db.AdditionalCosts.ToList();

        }

        #endregion

        public void DeleteCustomer(Customer c)

        {

            db.Customers.Remove(c);

            db.SaveChanges();

        }

        public void DeleteTimeTable(TimeTable t)

        {

            db.TimeTables.Remove(t);

            db.SaveChanges();

        }

        public void DeleteSupplier(Supplier s)

        {

            db.Suppliers.Remove(s);

            db.SaveChanges();

        }

        public void DeleteWorker(Worker w)

        {

            db.Workers.Remove(w);

            db.SaveChanges();

        }

        public void DeleteProduct(Product p)

        {

            db.Products.Remove(p);

            db.SaveChanges();

        }

        public void DeleteWorkerHour(WorkerHour w)

        {

            db.WorkerHours.Remove(w);

            db.SaveChanges();

        }

        public void DeleteSupplierOrder(SupplierOrder s)

        {

            db.SupplierOrders.Remove(s);

            db.SaveChanges();

        }

        public void DeleteOrder(Order o)

        {

            db.Orders.Remove(o);

            db.SaveChanges();

        }

        public void DeleteAdditionalCost(AdditionalCost a)

        {

            db.AdditionalCosts.Remove(a);

            db.SaveChanges();

        }

    }

# דף הבית

using BarberShop.Workers\_Windows;

using MaterialDesignThemes.Wpf;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for MainWindow.xaml

    /// </summary>

    public partial class MainWindow : Window

    {

        public MainWindow()

        {

            ManageTimeTable window = new ManageTimeTable();

            window.Close();

            InitializeComponent();

        }

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

        {

            ManageTimeTable window = new ManageTimeTable();

            window.ShowDialog();

        }

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

        {

            ManageSuppliers window = new ManageSuppliers();

            window.ShowDialog();

        }

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

        {

            ManageWorkers window = new ManageWorkers();

            window.ShowDialog();

        }

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

        {

            System.Windows.Application.Current.Shutdown();

        }

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

        {

            ManageCustomers window = new ManageCustomers();

            window.ShowDialog();

        }

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

        {

            ManageProductsList window = new ManageProductsList();

            window.ShowDialog();

        }

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

        {

            ClockWorkers window = new ClockWorkers();

            window.ShowDialog();

        }

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

        {

            OrderProducts window = new OrderProducts();

            window.ShowDialog();

        }

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

        {

            SellProduct window = new SellProduct();

            window.ShowDialog();

        }

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

        {

            CustomerHistory window = new CustomerHistory();

            window.ShowDialog();

        }

        private void ReportAdditionalExpenses(object sender, RoutedEventArgs e)

        {

            AdditionalExpenses window = new AdditionalExpenses();

            window.ShowDialog();

        }

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

        {

            ExpensesReport window = new ExpensesReport();

            window.ShowDialog();

        }

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

        {

            IncomesReport window = new IncomesReport();

            window.ShowDialog();

        }

        private void DialogHost\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            Rect workArea = System.Windows.SystemParameters.WorkArea;

            this.Left = (workArea.Width - this.Width) / 2 + workArea.Left;

            this.Top = (workArea.Height - this.Height) / 2 + workArea.Top;

        }

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

        {

            WindowState = WindowState.Minimized;

        }

    }

}

# ניהול תורים

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ManageTimeTable.xaml

    /// </summary>

    public partial class ManageTimeTable : Window

    {

        Controller controller = new Controller();

        public ManageTimeTable()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            controller = null;

            controller = new Controller();

            TimeTableDataGrid.ItemsSource = null;

            TimeTableDataGrid.ItemsSource = controller.GetTimeTable();

        }

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

        {

            AddBook window = new AddBook();

            window.ShowDialog();

            UpdateDataGrid();

            DatePicker.SelectedDate = null;

        }

        private void DatePicker\_CalendarClosed(object sender, RoutedEventArgs e)

        {

            List<TimeTable> l = controller.GetTimeTable();

            DateTime date = (DateTime)DatePicker.SelectedDate;

            for(int i =0; i < l.Count;)

            {

                if (DateTime.Compare(date, l[i].date) > 0)

                {

                    l.RemoveAt(i);

                }

                else

                {

                    i++;

                }

            }

            TimeTableDataGrid.ItemsSource = l;

        }

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

        {

            try

            {

                controller.DeleteTimeTable((TimeTable)TimeTableDataGrid.SelectedItem);

            }

            catch(Exception ex)

            {

                MessageBox.Show(ex.ToString(), "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            UpdateDataGrid();

        }

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

        {

            TimeTable t = (TimeTable)TimeTableDataGrid.SelectedItem;

            AddBook window = new AddBook(t);

            window.ShowDialog();

            UpdateDataGrid();

            DatePicker.SelectedDate = null;

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# הוספת תור

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Animation;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddBook.xaml

    /// </summary>

    public partial class AddBook : Window

    {

        Controller controller = new Controller();

        TimeTable t;

        bool isEdit;

        public AddBook()

        {

            InitializeComponent();

            SetComboBox();

            isEdit = false;

        }

        public AddBook(TimeTable t)

        {

            InitializeComponent();

            SetComboBox();

            isEdit = true;

            this.t = t;

            UpdateBookInfo();

        }

        private void UpdateBookInfo()

        {

            List<Customer> l1 = (List<Customer>)CustomerPicker.ItemsSource;

            List<Haircut> l2 = (List<Haircut>)HaircutPicker.ItemsSource;

            //MessageBox.Show(l1.FindIndex(x => x == t.Customer).ToString(), "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            DatePicker.SelectedDate = t.date;

            TimePicker.Text = t.time.ToString();

            CustomerPicker.SelectedIndex = l1.FindIndex(x => x.Id == t.Customer.Id);

            HaircutPicker.SelectedIndex = l2.FindIndex(x => x.Id == t.Haircut.Id);

            PriceTextBox.Text = t.price.ToString();

        }

        private void SetComboBox()

        {

            CustomerPicker.ItemsSource = controller.GetCustomers();

            CustomerPicker.DisplayMemberPath = "name";

            HaircutPicker.ItemsSource = controller.GetHaircuts();

            HaircutPicker.DisplayMemberPath = "name";

        }

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

        {

            try

            {

                double price = double.Parse(PriceTextBox.Text);

                TimeSpan time = TimeSpan.Parse(TimePicker.Text);

                DateTime date = (DateTime)DatePicker.SelectedDate;

                int type = (HaircutPicker.SelectedItem as Haircut).Id;

                int customer = (CustomerPicker.SelectedItem as Customer).Id;

                if (!isEdit)

                    controller.AddBook(price, date, type, customer, time);

                else

                {

                    t.price = price;

                    t.time = time;

                    t.date = date;

                    t.haricut\_id = type;

                    t.customer\_id = customer;

                    t.Customer = controller.GetCustomers().Find(x => x.Id == t.customer\_id);

                    t.Haircut = controller.GetHaircuts().Find(x => x.Id == t.haricut\_id);

                    controller.EditBook(t);

                }

                this.Close();

            }

            catch(ArgumentNullException ex)

            {

                MessageBox.Show("One of the fields is empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch(ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch(FormatException ex)

            {

                MessageBox.Show("Ensure that only numbers is present in number fields", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (NullReferenceException ex)

            {

                MessageBox.Show("Ensure that the selection boxes are not empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void HaircutPickerChanged(object sender, SelectionChangedEventArgs e)

        {

            Haircut h = (Haircut)HaircutPicker.SelectedItem;

            PriceTextBox.Text = h.price.ToString();

        }

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

        {

            this.Close();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# ניהול לקוחות

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ManageCustomers.xaml

    /// </summary>

    public partial class ManageCustomers : Window

    {

        Controller controller = new Controller();

        public ManageCustomers()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        public void UpdateDataGrid()

        {

            CustomersDataGrid.ItemsSource = controller.GetCustomers();

        }

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

        {

            AddCustomer window = new AddCustomer();

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            Customer c = (Customer)CustomersDataGrid.SelectedItem;

            AddCustomer window = new AddCustomer(c);

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            try

            {

                controller.DeleteCustomer((Customer)CustomersDataGrid.SelectedItem);

            }

            catch

            {

                MessageBox.Show("Unexpected error, please try again", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# לקוח חדש

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddCustomer.xaml

    /// </summary>

    public partial class AddCustomer : Window

    {

        Controller controller = new Controller();

        Customer c;

        bool isEdit;

        public AddCustomer()

        {

            InitializeComponent();

            isEdit = false;

        }

        public AddCustomer(Customer c)

        {

            InitializeComponent();

            this.c = c;

            isEdit = true;

            UpdateCustomerInfo();

        }

        public void UpdateCustomerInfo()

        {

            NameTextBox.Text = c.name;

            AddressTextBox.Text = c.address;

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                string name = NameTextBox.Text;

                string address = AddressTextBox.Text;

                if (!isEdit)

                    controller.AddCustomer(name, address);

                else

                {

                    c.name = name;

                    c.address = address;

                    controller.EditCustomer(c);

                }

                this.Close();

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# היסטוריית לקוח

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for CustomerHistory.xaml

    /// </summary>

    public partial class CustomerHistory : Window

    {

        Controller controller = new Controller();

        public CustomerHistory()

        {

            InitializeComponent();

            UpdateComboBox();

        }

        private void UpdateDataGrids()

        {

            List<TimeTable> l1 = controller.GetTimeTable();

            List<Order> l2 = controller.GetOrders();

            if (DatePicker.SelectedDate != null)

            {

                l1 = l1.FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

                l2 = l2.FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

            }

            HaircutDataGrid.ItemsSource = l1.FindAll(x => x.customer\_id == ((Customer)CustomerPicker.SelectedItem).Id);

            OrderDataGrid.ItemsSource = l2.FindAll(x => x.customer\_id == ((Customer)CustomerPicker.SelectedItem).Id);

        }

        private void UpdateComboBox()

        {

            CustomerPicker.ItemsSource = controller.GetCustomers();

            CustomerPicker.DisplayMemberPath = "name";

        }

        private void CustomerPicker\_SelectionChanged(object sender, SelectionChangedEventArgs e)

        {

            if (CustomerPicker.SelectedIndex > -1)

            {

                UpdateDataGrids();

                DatePicker.IsEnabled = true;

            }

            else

            {

                OrderDataGrid.ItemsSource = null;

                HaircutDataGrid.ItemsSource = null;

                DatePicker.IsEnabled = false;

            }

        }

        private void DatePicker\_SelectedDateChanged(object sender, SelectionChangedEventArgs e)

        {

            UpdateDataGrids();

        }

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

        {

            this.Close();

        }

    }

}

# ניהול עובדים

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop.Workers\_Windows

{

    /// <summary>

    /// Interaction logic for ManageWorkers.xaml

    /// </summary>

    public partial class ManageWorkers : Window

    {

        Controller controller = new Controller();

        public ManageWorkers()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            WorkersDataGrid.ItemsSource = controller.GetWorkers();

        }

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

        {

            try

            {

                controller.DeleteWorker((Worker)WorkersDataGrid.SelectedItem);

            }

            catch

            {

                MessageBox.Show("Unexpected error, please try again", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            UpdateDataGrid();

        }

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

        {

            Worker w = (Worker)WorkersDataGrid.SelectedItem;

            AddWorker window = new AddWorker(w);

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            AddWorker window = new AddWorker();

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# עובד חדש

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop.Workers\_Windows

{

    /// <summary>

    /// Interaction logic for AddWorker.xaml

    /// </summary>

    public partial class AddWorker : Window

    {

        Controller controller = new Controller();

        Worker w;

        bool isEdit;

        public AddWorker()

        {

            InitializeComponent();

            isEdit = false;

        }

        public AddWorker(Worker w)

        {

            InitializeComponent();

            isEdit = true;

            this.w = w;

            UpdateWorkerInfo();

        }

        private void UpdateWorkerInfo()

        {

            NameTextBox.Text = w.name;

            IdTextBox.Text = w.identification;

            AddressTextBox.Text = w.address;

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                string name = NameTextBox.Text;

                string id = IdTextBox.Text;

                string address = AddressTextBox.Text;

                if (!isEdit)

                    controller.AddWorker(name, id, address);

                else

                {

                    w.name = name;

                    w.identification = id;

                    w.address = address;

                    controller.EditWorker(w);

                }

                this.Close();

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# ניהול שעות עבודה של עובד

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ClockWorkers.xaml

    /// </summary>

    public partial class ClockWorkers : Window

    {

        Controller controller = new Controller();

        public ClockWorkers()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            controller = null;

            controller = new Controller();

            ClockWorkersDataGrid.ItemsSource = controller.GetWorkerHours();

        }

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

        {

            AddClockHours window = new AddClockHours();

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            WorkerHour w = (WorkerHour)ClockWorkersDataGrid.SelectedItem;

            controller.DeleteWorkerHour(w);

            UpdateDataGrid();

        }

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

        {

            WorkerHour w = (WorkerHour)ClockWorkersDataGrid.SelectedItem;

            AddClockHours window = new AddClockHours(w);

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# משמרת חדשה

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddClockHours.xaml

    /// </summary>

    public partial class AddClockHours : Window

    {

        Controller controller = new Controller();

        bool isEdit;

        WorkerHour w;

        public AddClockHours()

        {

            InitializeComponent();

            isEdit = false;

            UpdateComboBox();

        }

        public AddClockHours(WorkerHour w)

        {

            InitializeComponent();

            this.w = w;

            isEdit = true;

            UpdateComboBox();

            UpdateClockHourInfo();

        }

        private void UpdateClockHourInfo()

        {

            List<Worker> l = (List<Worker>)WorkerPicker.ItemsSource;

            WorkerPicker.SelectedIndex = l.FindIndex(x => x.Id == w.Worker.Id);

            HoursTextBox.Text = w.hours.ToString();

            SalaryTextBox.Text = w.salary.ToString();

            DatePicker.SelectedDate = w.date;

        }

        private void UpdateComboBox()

        {

            WorkerPicker.ItemsSource = controller.GetWorkers();

            WorkerPicker.DisplayMemberPath = "name";

        }

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

        {

            try

            {

                int hours = int.Parse(HoursTextBox.Text);

                double salary = double.Parse(SalaryTextBox.Text);

                DateTime date = (DateTime)DatePicker.SelectedDate;

                int worker = (WorkerPicker.SelectedItem as Worker).Id;

                if (!isEdit)

                    controller.AddWorkerHour(worker, hours, salary, date, salary \* hours);

                else

                {

                    w.hours = hours;

                    w.salary = salary;

                    w.date = date;

                    w.worker\_id = worker;

                    w.total = w.salary \* w.hours;

                    w.Worker = controller.GetWorkers().Find(x => x.Id == w.worker\_id);

                    controller.EditWorkerHour(w);

                }

                this.Close();

            }

            catch (ArgumentNullException ex)

            {

                MessageBox.Show("One of the fields is empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (FormatException ex)

            {

                MessageBox.Show("Ensure that only numbers is present in number fields", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (NullReferenceException ex)

            {

                MessageBox.Show("Ensure that the selection boxes are not empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

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

        {

            this.Close();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

        }

    }

}

# ניהול ספקים

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ManageSuplliers.xaml

    /// </summary>

    public partial class ManageSuppliers : Window

    {

        Controller controller = new Controller();

        public ManageSuppliers()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            SuppliersDataGrid.ItemsSource = controller.GetSuppliers();

        }

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

        {

            AddSupplier window = new AddSupplier();

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            try

            {

                controller.DeleteSupplier((Supplier)SuppliersDataGrid.SelectedItem);

            }

            catch

            {

                MessageBox.Show("Unexpected error, please try again", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            UpdateDataGrid();

        }

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

        {

            Supplier s = (Supplier)SuppliersDataGrid.SelectedItem;

            AddSupplier window = new AddSupplier(s);

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# יצירת ספק חדש

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddSupplier.xaml

    /// </summary>

    public partial class AddSupplier : Window

    {

        Controller controller = new Controller();

        Supplier s;

        bool isEdit;

        public AddSupplier()

        {

            InitializeComponent();

            isEdit = false;

        }

        public AddSupplier(Supplier s)

        {

            InitializeComponent();

            isEdit = true;

            this.s = s;

            UpdateSupplierInfo();

        }

        private void UpdateSupplierInfo()

        {

            NameTextBox.Text = s.name;

            AddressTextBox.Text = s.address;

            PhoneTextBox.Text = s.phone;

            EmailTextBox.Text = s.email;

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                string name = NameTextBox.Text;

                string address = AddressTextBox.Text;

                string phone = PhoneTextBox.Text;

                string email = EmailTextBox.Text;

                if (!isEdit)

                    controller.AddSupplier(name, address, phone, email);

                else

                {

                    s.name = name;

                    s.address = address;

                    s.phone = phone;

                    s.email = email;

                    controller.EditSupplier(s);

                }

                this.Close();

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# רשימת מוצרים

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ManageProductsList.xaml

    /// </summary>

    public partial class ManageProductsList : Window

    {

        Controller controller = new Controller();

        public ManageProductsList()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            controller = null;

            controller = new Controller();

            ProductsDataGrid.ItemsSource = controller.GetProducts();

        }

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

        {

            AddProduct window = new AddProduct();

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            Product p = (Product)ProductsDataGrid.SelectedItem;

            controller.DeleteProduct(p);

            UpdateDataGrid();

        }

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

        {

            Product p = (Product)ProductsDataGrid.SelectedItem;

            AddProduct window = new AddProduct(p);

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# מוצר חדש

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddProduct.xaml

    /// </summary>

    public partial class AddProduct : Window

    {

        Controller controller = new Controller();

        bool isEdit;

        Product p;

        public AddProduct()

        {

            InitializeComponent();

            UpdateComboBox();

            isEdit = false;

        }

        public AddProduct(Product p)

        {

            InitializeComponent();

            UpdateComboBox();

            this.p = p;

            isEdit = true;

            UpdateProductInfo();

        }

        private void UpdateProductInfo()

        {

            List<Supplier> s = (List<Supplier>)SupplierPicker.ItemsSource;

            NameTextBox.Text = p.name;

            PriceTextBox.Text = p.price.ToString();

            DescriptionTextBox.Text = p.description;

            SupplierPicker.SelectedIndex = s.FindIndex(x => x.Id == p.Supplier.Id);

        }

        private void UpdateComboBox()

        {

            SupplierPicker.ItemsSource = controller.GetSuppliers();

            SupplierPicker.DisplayMemberPath = "name";

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                string name = NameTextBox.Text;

                double price = double.Parse(PriceTextBox.Text);

                string description = DescriptionTextBox.Text;

                int supplier = (SupplierPicker.SelectedItem as Supplier).Id;

                if (!isEdit)

                    controller.AddProduct(name, price, description, supplier);

                else

                {

                    p.name = name;

                    p.price = price;

                    p.description = description;

                    p.supplier\_id = supplier;

                    p.Supplier = controller.GetSuppliers().Find(x => x.Id == p.supplier\_id);

                    controller.EditProduct(p);

                }

                this.Close();

            }

            catch (ArgumentNullException ex)

            {

                MessageBox.Show("One of the fields is empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (FormatException ex)

            {

                MessageBox.Show("Ensure that only numbers is present in number fields", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (NullReferenceException ex)

            {

                MessageBox.Show("Ensure that the selection boxes are not empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# מכירות מוצרים

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for SellProduct.xaml

    /// </summary>

    public partial class SellProduct : Window

    {

        Controller controller = new Controller();

        public SellProduct()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            controller = null;

            controller = new Controller();

            OrdersDataGrid.ItemsSource = null;

            OrdersDataGrid.ItemsSource = controller.GetOrders();

        }

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

        {

            Order o = (Order)OrdersDataGrid.SelectedItem;

            controller.DeleteOrder(o);

            UpdateDataGrid();

        }

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

        {

            Order o = (Order)OrdersDataGrid.SelectedItem;

            AddOrderCustomer window = new AddOrderCustomer(o);

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            AddOrderCustomer window = new AddOrderCustomer();

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# מכירה חדשה

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddOrderCustomer.xaml

    /// </summary>

    public partial class AddOrderCustomer : Window

    {

        Controller controller = new Controller();

        bool isEdit;

        Order o;

        public AddOrderCustomer()

        {

            InitializeComponent();

            UpdateComboBox();

            isEdit = false;

        }

        public AddOrderCustomer(Order o)

        {

            InitializeComponent();

            UpdateComboBox();

            isEdit = true;

            this.o = o;

            UpdateOrderInfo();

        }

        private void UpdateOrderInfo()

        {

            List<Customer> l1 = (List<Customer>)CustomerPicker.ItemsSource;

            List<Product> l2 = (List<Product>)ProductPicker.ItemsSource;

            CustomerPicker.SelectedIndex = l1.FindIndex(x => x.Id == o.Customer.Id);

            ProductPicker.SelectedIndex = l2.FindIndex(x => x.Id == o.Product.Id);

            DatePicker.SelectedDate = o.date;

            PriceTextBox.Text = o.price.ToString();

        }

        private void UpdateComboBox()

        {

            CustomerPicker.ItemsSource = controller.GetCustomers();

            CustomerPicker.DisplayMemberPath = "name";

            ProductPicker.ItemsSource = controller.GetProducts();

            ProductPicker.DisplayMemberPath = "name";

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                int customer = (CustomerPicker.SelectedItem as Customer).Id;

                int product = (ProductPicker.SelectedItem as Product).Id;

                DateTime date = (DateTime)DatePicker.SelectedDate;

                double price = double.Parse(PriceTextBox.Text);

                if (!isEdit)

                    controller.AddOrder(customer, product, date, price);

                else

                {

                    o.customer\_id = customer;

                    o.product\_id = product;

                    o.date = date;

                    o.price = price;

                    o.Customer = controller.GetCustomers().Find(x => x.Id == o.customer\_id);

                    o.Product = controller.GetProducts().Find(x => x.Id == o.product\_id);

                    controller.EditOrder(o);

                }

                this.Close();

            }

            catch (ArgumentNullException ex)

            {

                MessageBox.Show("One of the fields is empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (FormatException ex)

            {

                MessageBox.Show("Ensure that only numbers is present in number fields", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (NullReferenceException ex)

            {

                MessageBox.Show("Ensure that the selection boxes are not empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

    }

}

# הזמנת מוצרים לעסק

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for OrderProducts.xaml

    /// </summary>

    public partial class OrderProducts : Window

    {

        Controller controller = new Controller();

        public OrderProducts()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            controller = null;

            controller = new Controller();

            OrdersDataGrid.ItemsSource = null;

            OrdersDataGrid.ItemsSource = controller.GetSupplierOrders();

        }

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

        {

            SupplierOrder s = (SupplierOrder)OrdersDataGrid.SelectedItem;

            controller.DeleteSupplierOrder(s);

            UpdateDataGrid();

        }

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

        {

            SupplierOrder s = (SupplierOrder)OrdersDataGrid.SelectedItem;

            AddOrder window = new AddOrder(s);

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            AddOrder window = new AddOrder();

            window.ShowDialog();

            UpdateDataGrid();

        }

        private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

        {

            if (e.ChangedButton == MouseButton.Left)

                this.DragMove();

        }

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

        {

            this.Close();

        }

    }

}

# הוצאות נוספות

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AdditionalExpenses.xaml

    /// </summary>

    public partial class AdditionalExpenses : Window

    {

        Controller controller = new Controller();

        public AdditionalExpenses()

        {

            InitializeComponent();

            UpdateDataGrid();

        }

        private void UpdateDataGrid()

        {

            ExpensesDataGrid.ItemsSource = controller.GetAdditionalCosts();

        }

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

        {

            AddAdditionalCost window = new AddAdditionalCost();

            window.ShowDialog();

            UpdateDataGrid();

        }

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

        {

            this.Close();

        }

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

        {

            AdditionalCost a = (AdditionalCost)ExpensesDataGrid.SelectedItem;

            controller.DeleteAdditionalCosts(a);

            UpdateDataGrid();

        }

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

        {

            AdditionalCost a = (AdditionalCost)ExpensesDataGrid.SelectedItem;

            AddAdditionalCost window = new AddAdditionalCost(a);

            window.ShowDialog();

            UpdateDataGrid();

        }

    }

}

# הוצאה לא צפויה חדשה

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for AddAdditionalCost.xaml

    /// </summary>

    public partial class AddAdditionalCost : Window

    {

        Controller controller = new Controller();

        AdditionalCost a;

        bool isEdit;

        public AddAdditionalCost()

        {

            InitializeComponent();

            isEdit = false;

        }

        public AddAdditionalCost(AdditionalCost a)

        {

            InitializeComponent();

            this.a = a;

            isEdit = true;

            UpdateAdditionalCostInfo();

        }

        private void UpdateAdditionalCostInfo()

        {

            CostTextBox.Text = a.cost.ToString();

            DescriptionTextBox.Text = a.description;

            DatePicker.SelectedDate = a.date;

        }

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

        {

            this.Close();

        }

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

        {

            try

            {

                double cost = double.Parse(CostTextBox.Text);

                string description = DescriptionTextBox.Text;

                DateTime date = (DateTime)DatePicker.SelectedDate;

                if (!isEdit)

                    controller.AddAdditionalCost(cost, description, date);

                else

                {

                    a.cost = cost;

                    a.description = description;

                    a.date = date;

                    controller.EditAdditionalCost(a);

                }

                this.Close();

            }

            catch (ArgumentNullException ex)

            {

                MessageBox.Show("One of the fields is empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (ArgumentException ex)

            {

                MessageBox.Show(ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (FormatException ex)

            {

                MessageBox.Show("Ensure that only numbers is present in number fields", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

            catch (NullReferenceException ex)

            {

                MessageBox.Show("Ensure that the selection boxes are not empty", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

            }

        }

    }

}

# דוח הוצאות

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for ExpensesReport.xaml

    /// </summary>

    public partial class ExpensesReport : Window

    {

        Controller controller = new Controller();

        public ExpensesReport()

        {

            InitializeComponent();

            UpdateDataGrid(null, null);

        }

        private void UpdateDataGrid(object sender, SelectionChangedEventArgs e)

        {

            if (DatePicker.SelectedDate == null)

            {

                OrderDataGrid.ItemsSource = controller.GetSupplierOrders();

                AdditionalExpenseDataGrid.ItemsSource = controller.GetAdditionalCosts();

                WorkerHoursDataGrid.ItemsSource = controller.GetWorkerHours();

            }

            else

            {

                OrderDataGrid.ItemsSource = controller.GetSupplierOrders().FindAll(x=> DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

                AdditionalExpenseDataGrid.ItemsSource = controller.GetAdditionalCosts().FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

                WorkerHoursDataGrid.ItemsSource = controller.GetWorkerHours().FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

            }

            List<WorkerHour> l1 = (List<WorkerHour>)WorkerHoursDataGrid.ItemsSource;

            List<AdditionalCost> l2 = (List<AdditionalCost>)AdditionalExpenseDataGrid.ItemsSource;

            List<SupplierOrder> l3 = (List<SupplierOrder>)OrderDataGrid.ItemsSource;

            TotalCost.Content = "Total Cost: " + (l1.Sum(x => x.total) + l2.Sum(x => x.cost) + l3.Sum(x => x.price)).ToString();

        }

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

        {

            this.Close();

        }

    }

}

# דוח הכנסות

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace BarberShop

{

    /// <summary>

    /// Interaction logic for IncomesReport.xaml

    /// </summary>

    public partial class IncomesReport : Window

    {

        Controller controller = new Controller();

        public IncomesReport()

        {

            InitializeComponent();

            UpdateDataGrid(null, null);

        }

        private void UpdateDataGrid(object sender, SelectionChangedEventArgs e)

        {

            if (DatePicker.SelectedDate == null)

            {

                BookDataGrid.ItemsSource = controller.GetTimeTable();

                OrderDataGrid.ItemsSource = controller.GetOrders();

            }

            else

            {

                BookDataGrid.ItemsSource = controller.GetTimeTable().FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

                OrderDataGrid.ItemsSource = controller.GetOrders().FindAll(x => DateTime.Compare(x.date, (DateTime)DatePicker.SelectedDate) >= 0);

            }

            TotalCost.Content = "Income: " + (((List<TimeTable>)BookDataGrid.ItemsSource).Sum(x => x.price) + ((List<Order>)OrderDataGrid.ItemsSource).Sum(x => x.price)).ToString();

        }

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

        {

            this.Close();

        }

    }

# סיכום אישי

לאחר הגשת כמה וכמה פרויקטים בנושא מדעי המחשב רכשתי מיומנויות וידע אשר עזרו לי בפרויקט זה, ראשית כבר עבדתי עם מבנה נתונים בעבר ולכן הנושא היה מובן לי כראוי ויכולתי להתקדם מבלי תלות העברת החומר על ידי המורה, שנית חילוק הזמן יחד עם הלימודים הוא דבר שאני מכיר והצלחתי לנהל אותו כראוי ולבסוף כתיבת הספר דומה לפרויקטים שעשיתי בעבר לכן כתיבת הספר לא דרשה תשומת לב גדולה מהמורה המלמד.

אך למרות הניסיון הזעיר שיש לי עדיין למדתי רבות מפרויקט זה, השתמשתי בספריית עיצוב של Material Design שקישטה את הפרויקט כראוי, לא השתמשתי בספרייה זה בעבר ולכן הייתי צריך ללמוד איך להשתמש בה, ניהול טבלאות בצורה גרפית היא נושא חדש עבורי ולכן חקרתי אותו היטב על מנת שאוכל לבצע אותו כאן.

אני מאמין שחקרתי את נושא הפרויקט כמו שצריך, הבנתי את החומר הנלמד ואף אוכל לראות מצבים אשר אשתמש בו בעתיד, דבר אחד אשר הייתי משנה היא ניהול הזמן עם הבגרויות וחזרות מסיבת הסיום, מפני שאני בכיתה יב ומצב הקורונה עשו לנו מגוון חזרות בתקופת זמן קצר כנל גם הבגרויות ודבר זה השפיע על ניהול הזמן עם הפרויקט, אדע לפעם הבאה לסיים משימות מוקדם יותר.

אני מאוד גאה בתוכנה שבניתי, תוכנה זו מבוססת על עסק אמיתי אשר אני מכיר אישית את הבעלים ואני יכול לתאר את בעל העסק משתמש בתוכנה זו, הכרתי מגוון כלים אשר עזרו לי במהלך העבודה מהמורה ומחיפוש באינטרנט, אני ללא ספק מרגיש שמימשתי את עצמי בתוכנית זו של 5 יחידות מדעי המחשב.