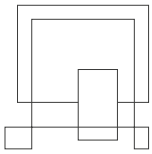


ShoeInc

Bachelor Project

Florin – Leonard Bordei – 280593

Jaume Lopez Topping – 282231

Supervisor: Jørn Martin Hajek

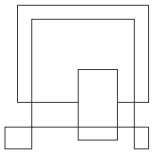
VIA University College

57931 characters

Software Technology Engineering

7th Semester

17-December-2021



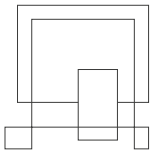
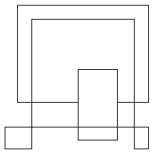
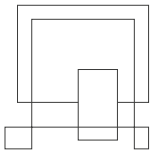


Table of content





Abstract

Author: Florin Bordei

The goal of the project is to assist a local shoes retailer from Horsens that would like to meet their business needs with new technological means so that their business operations and management would improve.

The business needs of the company are to improve their sales by adding to their operations a functioning online shop where customers are able to register, login, visit the web shop and make purchases. By fulfilling this needs the company is able to increase their market reach.

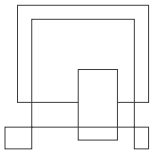
Secondly, for the company managers has been of a great importance to be able to accurately manage their portfolio of products and to streamline the process where they could add, modify or remove products from their collections. Also, to understand the current sales trends is of great importance as they could have a better understanding of the market and shopping trends of their customers.

Thirdly, for the warehouse managers it is critical to have control over their stocks where they could be notified of their current stocks and if there would be any products that could become out of stock. Meeting their market demand with an appropriate supply would not disrupt any processes.

The main technologies that are being used in this project are Github for code version control, Rider and a cloud deployed Azure Database. All these elements have been utilized to produce an end-result that would satisfy the company needs.

The Web Application Blazor is based on ASP.NET Core, the code language utilized is C# and the database used for the system is implemented through SQL dialect.

The result of the project is to bring a new shopping experience for the customers of ShoeInc and a new way of managing business for the company and warehouse managers.



1 Introduction

ShoeInc is a local retailer located in the city of Horsens, Denmark. Their main market where they operate is sales of shareware and through their excellent customer services they have managed to get notoriety in the market. Besides their customer services also their portfolio of products has been of great interest for their customers as the shop manages to cover any market need.

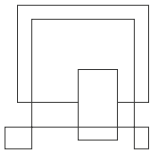
With the price of notoriety and success, a desire of expansion has emerged and the company managed to be funded by private equity funds to expand their line of physical stores but also to enter the online market. With the experience from their shop in Horsens, the company has been able to expand their physical shops but the process to enter the online market has been delayed as they did not possess any technological solutions for online purchases, deliveries and stock management.

The warehouse staff is under equipped with the right tools as they have inappropriate stock management tools and their ordering mechanism still relies on a classical phone call to the suppliers. The staff is not able to foresee or to prevent any out of stock situations and they are sometimes unable to supply accordingly.

The sales management is not able at moment to clearly understand how their sales performance indicators are performing or to have a strong overview of their top sold products. In such circumstances, the staff is not able to fully understand the market and how they could see any market trends.

The company management is unable to have a strong overview over their portfolio of products and by entering the online market they would need a strong representation of their products and an understanding of their customers.

Some of the delimitations of this project will have are that the system will not be able to automatically predict or recommend any product purchases to the company’s customers.



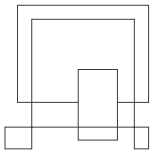
As the reader is understanding the issues that ShoeInc is facing, the next chapters will introduce the reader to the sections of Analysis and Design that would explain how the requirements have been formulated, the actors involved, diagrams that would help understand the behaviour or actions and the overall structure of the system designed.

Choices of technology will be fully described as they are required in order to understand the Implementation section where the reader will be able to have a glimpse into the system code implementation.

Testing will help the reader to understand how the system behaves in its end result form.

Results and Discussion section will present the final outcome for the reader an what results have been achieved in the presented project.

Lastly, Conclusions section will review the whole report so it can conclude its end.



2 Analysis

The main focus of the Analysis is problem domain and the solutions that could bring a solution to it. In this part of the project, the team works closely with the client and the main actors involved. In such a way, the functional and non-function requirements can be formulated.

With the finalized requirements, the next step is to depict the system behaviour or interaction with the main actors and functionalities of the system through use case diagrams and use case descriptions.

Next step is the creation of the System Sequence Diagrams would bring a better understanding of the interactions between the actors and the system as in what events can be generated by the actions of the actors.

Lastly, the Domain Model is created so it can present the type of relationships between the domain concepts and they influence each other.

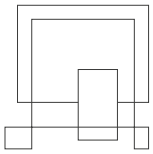
2.1 Requirements

After great lengthy conversations with the involved actors, the formulation of requirements can be performed. Prior to formulation, aspects such as functional and non-functional attributes must be taken in consideration.

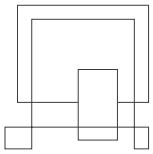
Implication of SMART principles is crucial as the requirements should be Specific, Measurable, Attainable, Relevant and Timely.

2.1.1 Functional Requirements

The following functional requirements have been grouped according to their designated actors and each requirement is assigned a priority value: high, medium or low.

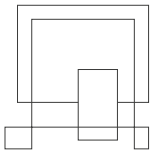


Actor: User(Customer)	
Priority	**Requirement**
High	As a customer, I would like to be able to create an account so I can log in to my account.
High	As a customer, I would like to be able to log in to my account so I can order products.
High	As a customer, I would like to be able to have all products displayed with picture, description, and price so I can make my research.
High	As a customer, I would like to be able to add or remove products in my shopping cart so I can have a final list of products that I would like to purchase.
High	As a customer, I would like to check out my products from the shopping cart so I can finalize my order.
Medium	As a customer, I would like to be able to add or modify my shipping address so I can have my correct shipping address stored into my account.
Medium	As a customer, I would like to be able to modify my email address so I can have my correct email address stored into my account.
Low	As a customer, I would like to be able to delete my account so my personal details are not in risk.

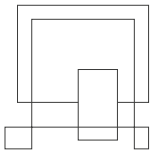


Actor: Company Manager	
Priority	**Requirement**
High	As a company manager, I would like to be able to add new products to my online shop so I can offer them for sale to my customers.
High	As a company manager, I would like to be able to edit/delete my products so I can have an up-to-date offer of products.
High	As a company manager, I would like to be able to apply or remove discount on my products so that I am able to boost my sales.
Medium	As a company manager, I would like to be able to check all my customer details so I can have an accurate overview of my customers.
Medium	As a company manager, I would like to be able to check all my products so I can have an accurate overview of my products.
Low	As a company manager, I would like to be able to check on the completed an uncompleted order so that I am able to keep track of my deliveries.

Actor: Warehouse Manager	
Priority	**Requirement**
High	As a warehouse manager, I would like to be able to check on my stock information so that I would be able to have an overview over my current stocks
High	As a warehouse manager, I would like to be able to add or delete products in my stocks so I can have an accurate list of products in my warehouse.
High	As a warehouse manager, I would like to be notified if products are about to go out of stock so I can be able to make orders from suppliers



Actor: Sales Manager	
Priority	**Requirement**
Medium	As a sales manager, I would like to be able to view all my sales data from my online shop so I am able to have an overview on my online sales.
Medium	As a sales manager, I would like to be able to view all my sales data from my shops, so I am able to have an overview on my physical sales.
Medium	As a sales manager, I would like to have a way of viewing my monthly and yearly sales from my online shop so that I am able to see the sales performance.
Medium	As a sales manager, I would like to have a way of viewing my monthly and yearly sales from my physical shop so that I am able to see the sales performance.
Medium	As a sales manager, I would like to have a way of comparing sales from different shops so that I am able to detect my top performant shop.
Medium	As a sales manager, I would like to have a way of viewing my top sold products so that I am able to understand which product performs better.
Medium	As a sales manager, I would like to have a way of viewing my top customers based on region, age and gender so that I am able to understand how market acts.
Medium	As a sales manager, I would like to have a way of viewing my top employee sales performance monthly and yearly so that I am able to detect my top performant employee.
Medium	As a sales manager, I would like to have a way of viewing the financial efficiency of a discount campaign so that I am able to understand if it has been profitable or not.
Medium	As a sales manager, I would like to have a way of viewing my overall business profitability so that I am able to report further on to my CEO.



2.1.2 Non-Functional Requirements

The application must be developed using programming language C#.

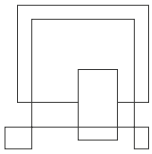
The application must be under version control over the entire development process.

The application must be developed using the official ASP.NET Core framework.

The database development must be done using SQL.

The database must be hosted online, on the cloud.

The User Interface must be in English language.



2.2 Actor Description

The actors that are defined for the system are: customer(user), company manager, warehouse manager and sales manager.

The customer, in some circumstances named user, should be able to create an account with required data such as email, password and other personal details. The customer must have the ability to login to the system and to browse the web shop where a list of products can be studied and, if decided, can make purchases where products are added in shopping cart. Final step is to finalize the order and to record it.

The warehouse manager should be able to see the list of current products that are located in the warehouse. Details such as: product name, product quantity etc. will be included in the list of products. Also, it should be able to see if there is any situation where products could become out of stock and, to prevent such situations, should be also able to order more products. Also, this actor can add or remove products from the list of products.

The company manager actor should be able to view lists of products and customers, to add or remove products and to edit products where is needed. The actor can also apply or remove discount on the products.

2.3 Use Case Diagram

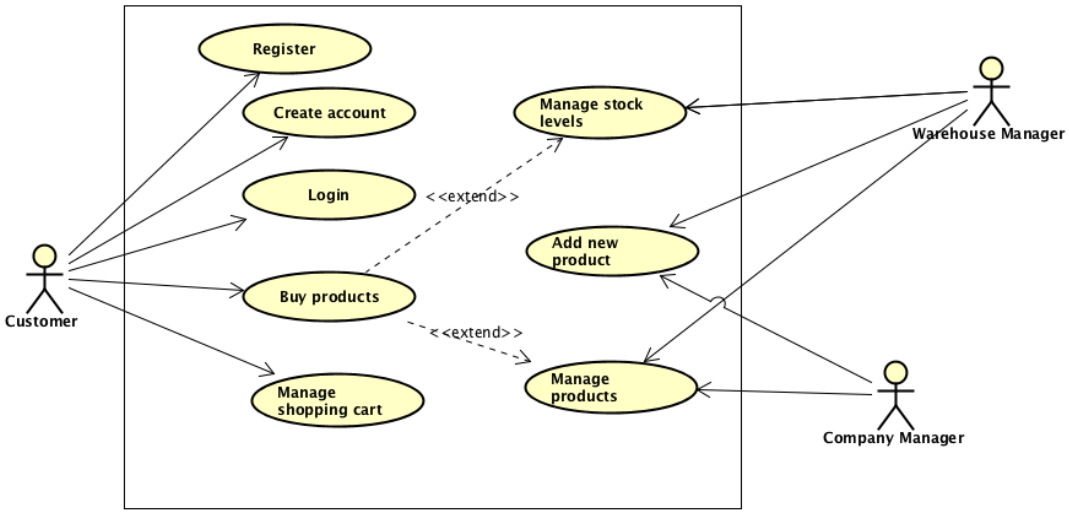
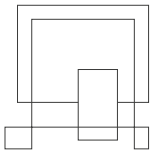


Figure 1 - Use Case Diagram

The Use Case Diagram shown in Figure 1 describes the ways in which the actors are interacting with the system. The actors are associated with the Use Case and the simplified version of the above Use Case Diagram describes the responsibilities of each actor.

The Customer is able to login, create account and buy products where the managers they are either managing the warehouse, the Warehouse Manager, or they manage the products that are currently sold on the web shop, the Company Manager.

As they are responsible for their activities they are also separated, as in the Customer is not able to manage or add new products.



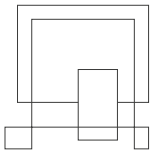
2.4 Use Case Description

A Use Case description is a textual description of a specific use. It contains information about the actor and the intended action, post and pre-conditions that are required before or an outcome of the interaction and a detailed step by step of the interaction between the actor and the system.

Figure 2 shows a Use Case for the interaction between the company manager and the system when a product is being added.

Item	Value	
Use Case	**Add Product**	
Summary	Actor adds a product	
Actor	Company Manager	
Precondition	User must be logged in so it can access to the view	
Postcondition	Product is created and displayed	
Base Sequence (Main Scenario)	**Actor**	**System**
	1. Selects menu option Products 2. Selects menu option Add Product 3. User fills in new Product details 4. User selects menu option Save	1.1 Returns list of products with menu 2.1 Returns form to add product 3.1 Validates details of new Product 4.1 Creates the new Product
Branch Sequence 1 (Alternate Scenarios)	2a. The product already exists based on its modelCode	
	Actor	System
	1. User fills in the details of the new Product 2. User selects menu option Save	1.1 Validates details of new Product 2.1 Creates the new Product
	Exception Sequence for the Branch Sequence 1 - At any time during steps (2a.1.), (2a.2.) Actor can cancel the process/ Use case ends	
Exception Sequence – Base Sequence	At any step, the actor aborts the process. The use case ends.	
Notes	The details filled in by the actor are: Brand, Model, ModelCode, Description, StockLevel, Price, Discount, Image	

Figure 2 - Use Case Description



2.5 Activity Diagrams

The activity diagrams describe the flow from one action to another in between the interaction between the actor and the system.

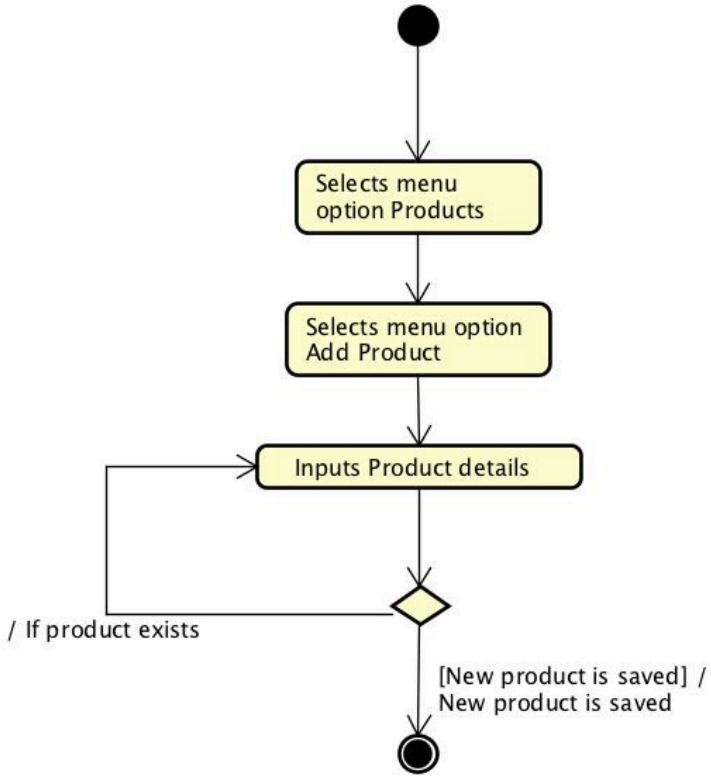
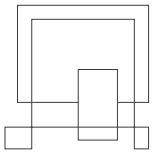


Figure 3 - Activity Diagram Add Product

As it can be seen in Figure 3 the actor follows specified steps towards his end result but also decision markers that can create a reaction from the system.

The actor selects the menu option Products followed by the menu option Add Product. The next step is where the actor will input the information about the new product that will go through a decision marker where the system will return the action in case the product exists or will end the action with the end result of creating and adding a new product.



The next case of an activity diagram is where the actor, company manager, applies or removes a discount. As shown in Figure 4 the actor selects the menu option Products, followed by selecting a specific product. In this situation, the actor has two alternative actions where a discount is applied or removed. If applied, the actor selects menu option Edit, fills in the discount value and selects menu option Ok. The action finishes with the product updated. If removed, the actor selects the menu option Renew and the action is terminated with the price updated.

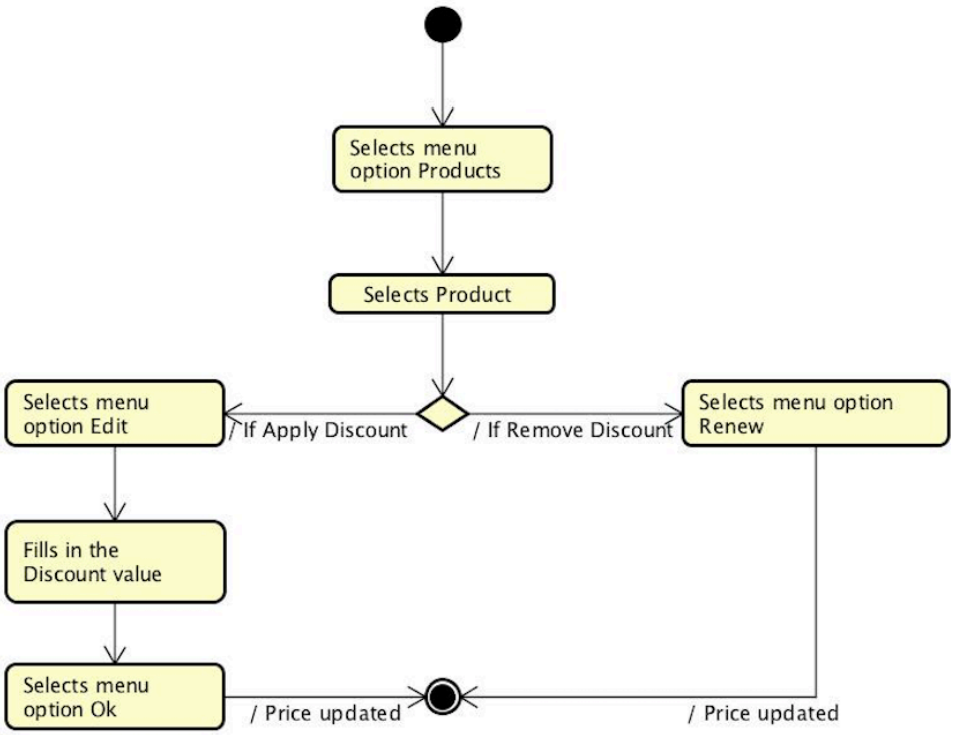
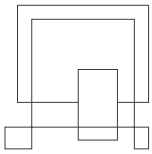


Figure 4 - Activity Diagram Apply/Remove Discount



2.6 System Sequence Diagram

The system sequence diagram displays the objects interactions that are following a certain time sequence. The messages or the objects interchanged are carrying out the functionality of the scenario. For the scenarios explained, the diagram shows the generated events by the actors and the certain order of the action [3*] (Sequence diagram – Wikipedia).*

The system sequence diagram in Figure 5 shows how the company manager would add a new product. The actions start with the company manager clicking the Products section and the system will respond with displaying a list of Products. The actor will proceed with clicking to AddProduct button and the system will display a form so that the company manager will fill in the details of the new product. If the actor decides to go with the action, button Save will be clicked and the product will be created and added. Otherwise, the actor will click Cancel product and the system will return the list of products.

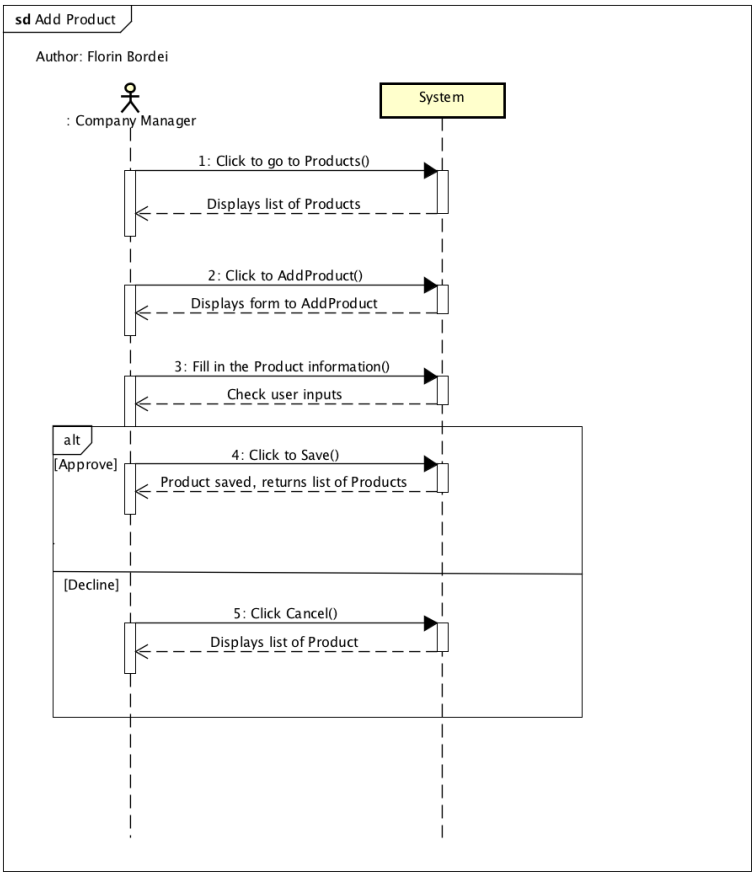
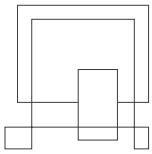


Figure 5 - System Sequence Diagram Add Product



The system sequence diagram in Figure 6 represents the flow of interactions between the actor, company manager, and the system when a discount is applied to a product or the discount is being removed. The company manager clicks on Products that will generate an action from the system to display the list of products. The actor is facing two alternative scenarios: apply or remove the discount. If applied, the company manager will click on Edit button, system will return the forms to fill in the values, the actor will click on Ok button and the price will be updated and listed. If removed, the actor will click on Renew button where the system will update the price and display it.

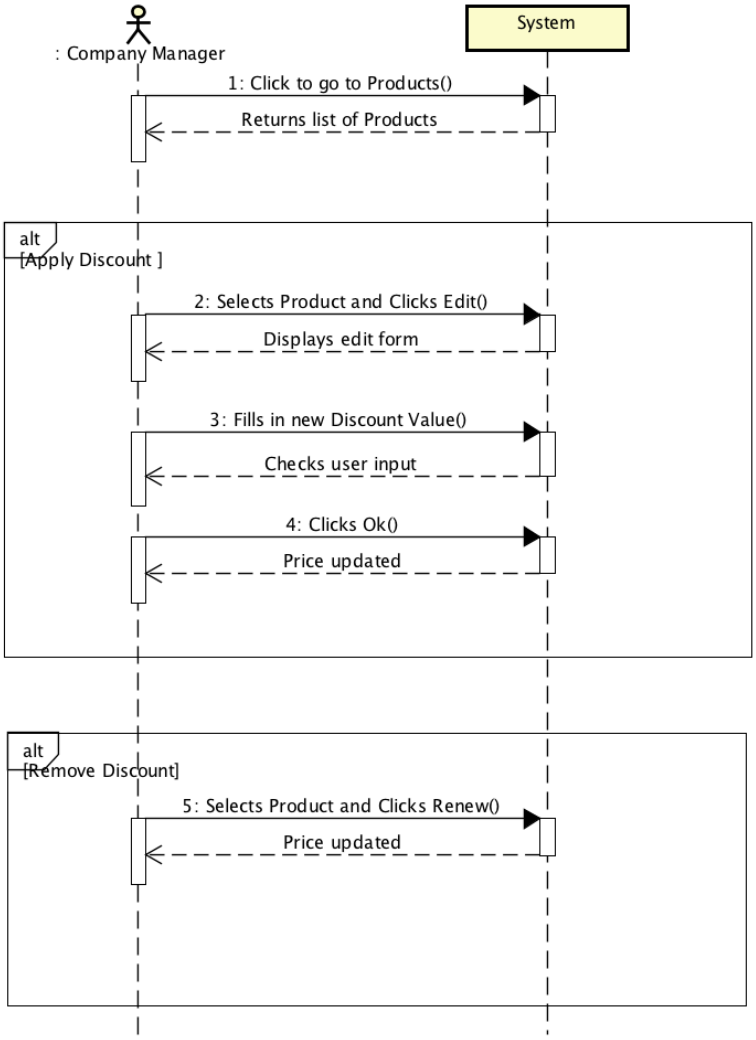
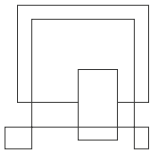


Figure 6 - System Sequence Diagram Apply/Remove Discount



2.7 Domain Model

After carefully analysing the previous steps of the analysis, it has been concluded the entities and the roles or interactions that are in between.

The warehouse manager has the role of managing products. In the term of managing, the warehouse manager can order, remove or add products.

The company manager has the roles of managing the products, view the list of customers and orders. In terms of managing products, the company manager can add, remove or edit products.

The customer is able to view the list of products from which shopping carts can be created where later orders and orderliness containing the products are generated.

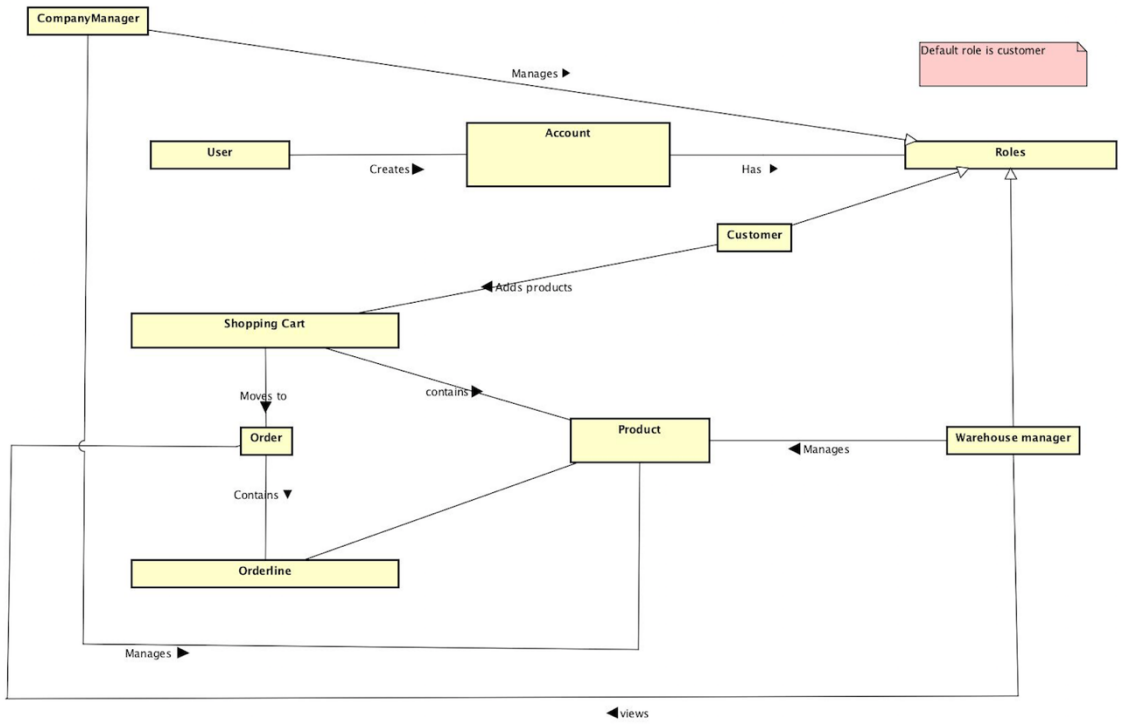
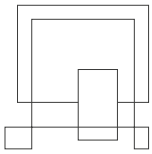


Figure 7 - Domain Model



3 Design

3.1 Class Diagram

This is the main class diagram the system is organized in 5 main packages:

Page: The page class is responsible for the UI design Controller: Each page has its own controller that contains all the necessary methods.

Data: Each entity in the app has its own data model class.

Repositories: All access to the database is through the repository classes.

Services: The service classes are called from the controllers to access API and repositories.

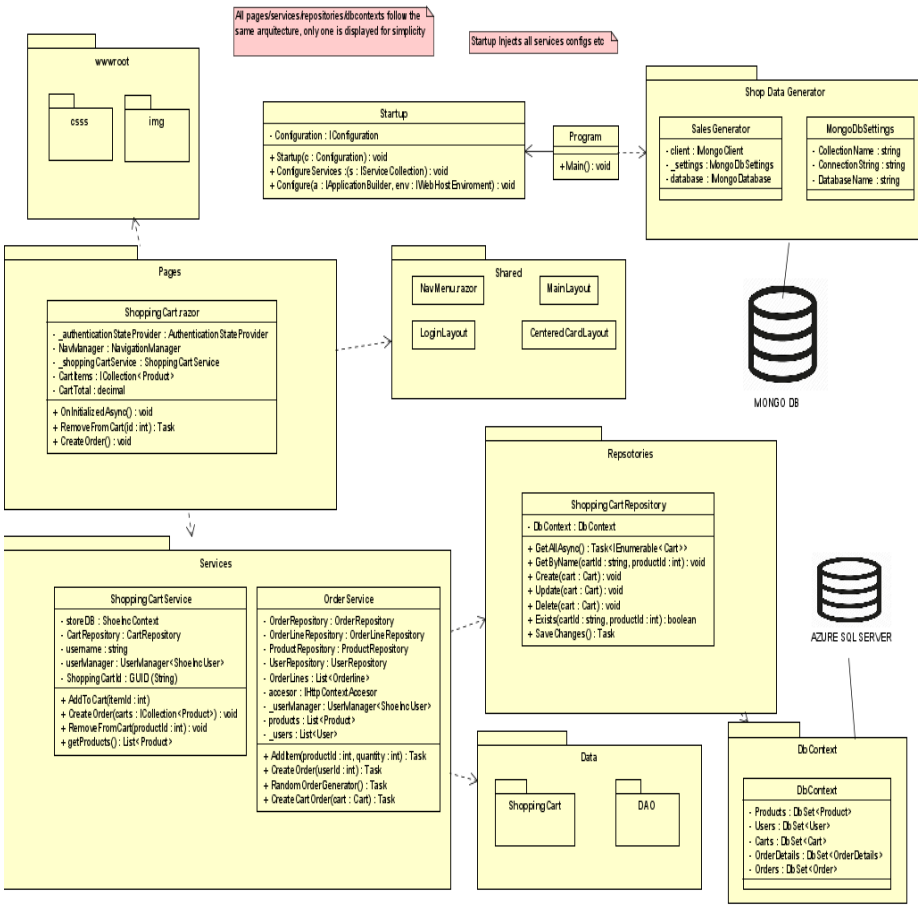
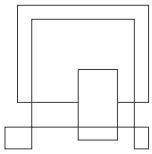


Figure 8 - Class Diagram



3.2 Design patterns and libraries

3.2.1 Design patterns and libraries

Blazor Server is NET Core framework. It removes much of the ceremony of ASP.NET MVC by adopting a file-based routing approach. Each Blazor Pages file found under the Pages directory equates to an endpoint. Blazor Pages can have associated service class or actual code in the page, which holds each page's behaviour. Blazor Server integrates the front end and back end in one application, there is no need to add external web services.

3.2.2 Repository pattern

All operations that access the database is accessed through the repository classes, to abstract the data sources from the rest of the app.

3.2.3 Data storage

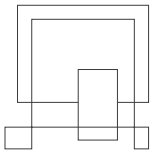
All the data is hosted in Azure Cloud, in an Azure SQL Server.

3.2.4 User management and Login features

For managing the users and login, the application uses ASP.NET Core Identity package:

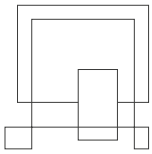
- It is an API that supports user interface (UI) login functionality.
- Manages users, passwords, profile data, roles, claims, tokens, email confirmation, and more.

The package has been scaffolded to be able to modify custom functionality and views.



3.2.5 Authorized views

Views can be enabled/disabled depending on if the user is logged in or has a specific role. ASP.NET Core Identity package is used for this purpose.



4 Implementation

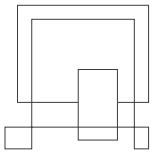
4.1 ASP.NET Core Identity package

4.1.1 Authorized views

In ASP.NET Core Identity System you can create any number of Roles and assign users to these roles. With the roles created it is possible enable/disable views in Blazor pages.

```
<AuthorizeView Roles="CompanyManager">
<li class="nav-item px-3">
    <NavLink class="nav-link" href="viewuserradzen">
        <span class="oi oi-list-rich" aria-hidden="true"></span> ProductsManager
    </NavLink>
</li>


<li class="nav-item px-3">
    <NavLink class="nav-link" href="xlistproductsmanagerradzen">
        <span class="oi oi-list-rich" aria-hidden="true"></span> Radzen PView
    </NavLink>
</li>
 </AuthorizeView>
<AuthorizeView Roles="Customer">
    <li class="nav-item px-3">
        <NavLink class="nav-link" href="MyShoppingCart">
            <span class="oi oi-list-rich" aria-hidden="true"></span> Shopping Cart
        </NavLink>
    </li>



    <li class="nav-item px-3">
        <NavLink class="nav-link" href="Shop">
            <span class="oi oi-list-rich" aria-hidden="true"></span> Shop
        </NavLink>
    </li>
</AuthorizeView>
```

Figure 9 - Authorized views



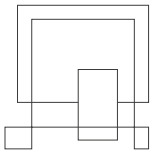
4.1.3 Authenticated views

In the Blazor pages any component can be wrapped with <Authorized>/<Not Authorized> tag and it will be displayed or hidden depending if the user is logged in or not.

```
<AuthorizeView>
    <Authorized>
        <a href="Identity/Account/Manage">Hello, @context.User.Identity?.Name!</a>

        <form method="post" action="Identity/Account/LogOut">
            <button type="submit" class="nav-link btn btn-link">Log out</button>
        </form>
    </Authorized>
    <NotAuthorized>
        <a href="Identity/Account/Register">Register</a>
        <a href="Identity/Account/Login">Log in</a>
    </NotAuthorized>
</AuthorizeView>
```

Figure 10 - Authenticated views

Authorized tags are used for displaying the user details if logged in or the login button if the user is not logged in.



4.2 Products management – Company manager

4.2.1 View list of products

For the user interface to view the list of products it has been used Radzen Blazor components which belongs to a library of user interfaces with the purpose to make interactive ASP.NET Core web applications [3] *(Radzen Blazor Components)*

The Blazor component to view the list of products has references to the classes that are being used in the methods but also dependency injections. The @page property is responsible for the routing of the system.

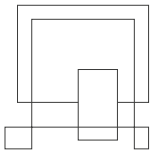
```
@page "/imageproduct"
@using ShoeInc.Data
@using ShoeInc.Repositories.Products
@inject ProductRepository ProductRepository
@inject NavigationManager NavigationManager
@inject DialogService DialogService
@implements IDisposable
```

Figure 11 - Resources, Routing and Dependency Injection

The RadzenDataGrid has a reference to a _productsGrid that controls the listing of the products and updates whenever there are changes. The data displayed comes from a List<Product> source _products and the columns have binding values for each property of a Product object that is displayed or modified on the _productsGrid.

```
<RadzenButton Click="@OnAddProduct" Text="Add product" Icon="check_circle" ButtonStyle="ButtonStyle.Success" />

<RadzenDataGrid @ref="_productsGrid" AllowFiltering="true" AllowPaging="true" PageSize="14" AllowSorting="true"
                EditMode="DataGridEditMode.Single" Data="@_products" TItem="Product" RowUpdate="@OnUpdateRow" >
        <Columns>
            <RadzenDataGridColumn TItem="Product" Property="Product.Brand" Title="Brand" Width="100px" >
                <EditTemplate Context="product">
                    <RadzenTextBox style="..." Name="Brand" @bind-Value="product.Brand" />

                </EditTemplate>
            </RadzenDataGridColumn>

            <RadzenDataGridColumn TItem="Product" Property="Product.Model" Title="Model" Width="100px" >
                <EditTemplate Context="product">
                    <RadzenTextBox style="..." Name="Model" @bind-Value="product.Model" />
                </EditTemplate>
            </RadzenDataGridColumn>

            <RadzenDataGridColumn TItem="Product" Property="Product.ModelCode" Title="ModelCode" Width="125px" >
                <EditTemplate Context="product">
                    <RadzenTextBox style="..." Name="ModelCode" @bind-Value="product.ModelCode" />
                </EditTemplate>
            </RadzenDataGridColumn>
```

Figure 12 - RadzenDataGrid



When the Blazor page of Products is first loaded, the method OnInitializeAsync() is executed, part of the lifecycly methods of Blazor [3] (*What are the application lifecycle methods in Blazor?)*.The method is running asynchronous and the _productRepository is being instantiated where is being given the Context for its constructer. The next instantiation is of the _products that receives the list of products from the ProductRepository. As the method is completed, the list of products is listed in the component.

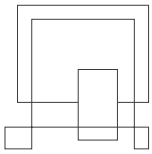
```
RadzenDataGrid<Product> _productsGrid;
private IList<Product> _products;
private ProductRepository _productRepository;

protected override async Task OnInitializedAsync()
{
     _productRepository = new ProductRepository(Context);
    _products =  (IList<Product>) await _productRepository.GetAllProductsAsync();

}
```

Figure 13 - OnInitializedAsync() method

The repository ProductsRepository has the role to access the objects from the database and to separate the layers of responsibility. As the method to obtain all products is awaited in the Blazor component, the method inside the repository is also running asynchronously. The GetAllProductsAsync() method is utilizing Linq to query the data from the database, order it according to its Model name and to return it as a List.

```
// returns all Product objects stored in the db
7 usages   Florin Leonard Bordei +1
public async Task<IEnumerable<Product>> GetAllProductsAsync()
{
    |
    return await FindAll().OrderBy(prod:Product => prod.Model).ToListAsync();
}
```

Figure 14 - Product Repository, GetAllProductsAsync()



4.2.2 View list of products

For the company manager to be able to add a new product, the AddProduct button must be click that is located on the Products page. The button contains a click action that will run the OnAddProduct method which will redirect the user to the Add Product page by using the NavigationManager which handles the routing.

```
<RadzenButton Click="@OnAddProduct" Text="Add product" Icon="check_circle" ButtonStyle="ButtonStyle.Success" />
```

Figure 15 - OnAddProduct button action

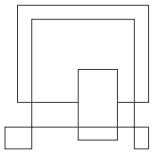
```
private void OnAddProduct()
{
    NavigationManager.NavigateTo( uri: "/imageproduct");
}
```

Figure 16 - OnAddProduct() method

The Blazor component that adds a new product is also making use of Radzen components, RadzenTemplateForm. The form binds the values of the new product added and is using the new product as a parameter in the Submit method.

```
<RadzenTemplateForm Data="@Product" Submit="@((Product product) => { Submit(product); })">
        <div class="row">
        <div class="col-md-6">
            <RadzenFieldset Text="Add a new product ">

                <div class="row">
                    <div class="col-md-4 align-items-center d-flex">
                        <RadzenLabel Text="Brand" />
                    </div>
                    <div class="col-md-8">
                        <RadzenTextBox style="..." Name="Brand" @bind-Value="Product.Brand" />
                    </div>
                </div>

                 <div class="row">
                                <div class="col-md-4 align-items-center d-flex">
                                    <RadzenLabel Text="Model" />
                                </div>
                                <div class="col-md-8">
                                    <RadzenTextBox style="..." Name="Model" @bind-Value="Product.Model" />
                                </div>
                            </div>
```

Figure 17 - RadzenTemplateForm



Besides text or numerical properties of the new Product, the form is also taking an image as a property.

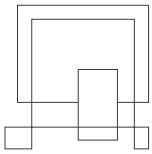
```
<div class="row">
    <div class="col-md-4 align-items-center d-flex">
        <label>Image:</label>
        </div>
    <div class="col-md-3">
        <InputFile OnChange="@OnFileSelection" />
        </div>
    </div>
```

Figure 18 - Image upload HTML

When a new image is being uploaded the OnFileSelection method will run where the file will be obtainged from InputFileChangeEventArgs, turnedinto a buffer byte and read through a stream. The byte content of the file will be stored in a string imagetype and the imageUrl will be converted to a string format also.

```
public async Task OnFileSelection(InputFileChangeEventArgs e)
 {
    // Obtaining the uploaded file
     IBrowserFile img = e.File;
    // The file is turned into a new byte
     var buffers = new byte[img.Size];
     await img.OpenReadStream().ReadAsync( buffers);
     string imagetype = img.ContentType;
    // The image lik is obtained
     _imgSrc = $"data:{imagetype};base64,{Convert.ToBase64String(buffers)}";

    Product.Image = buffers;
    Product.ImageUrl = _imgSrc;

 }
```

Figure 19 - OnFileSelection method



As this previous step is complete, the Submit method will make a call to the ProductRepository's method CreateProduct where it will pass the new product and the user will be redirected to the ListOfProducts page with a call to NavigationManager.

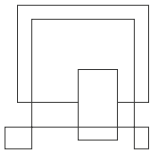
```
private void Submit(Product product)
{
    // New product is saved in the db
    ProductRepository.CreateProduct(product);
    NavigationManager.NavigateTo( uri: "/ListOfProducts");
}
```

Figure 20 - Submit method

In the ProductRepository the passed object will go through validation to check if the product already exists. If not, the new object will be created and stored in the databased.

```
public void CreateProduct(Product product)
{
    if (!ProductExists(product.ModelCode))
    {
        Create(product);
        _context.SaveChanges();
    }
    else
    {
        Console.WriteLine("Product already exists in our warehouse");
    }
}
```

Figure 21 - ProductRepository, CreateProduct method



4.3 Database implementation

For the implementation of the database it has been used SQL queries and the creation of the tables has been done according to the ER diagram from the Design phase. To facilitate the implementation process, a cloud provider has been selected to work with: Azure. Connection to the database has been established in two different ways. First one is through Rider's interface of databases so that data can be visualized in a fast manner.

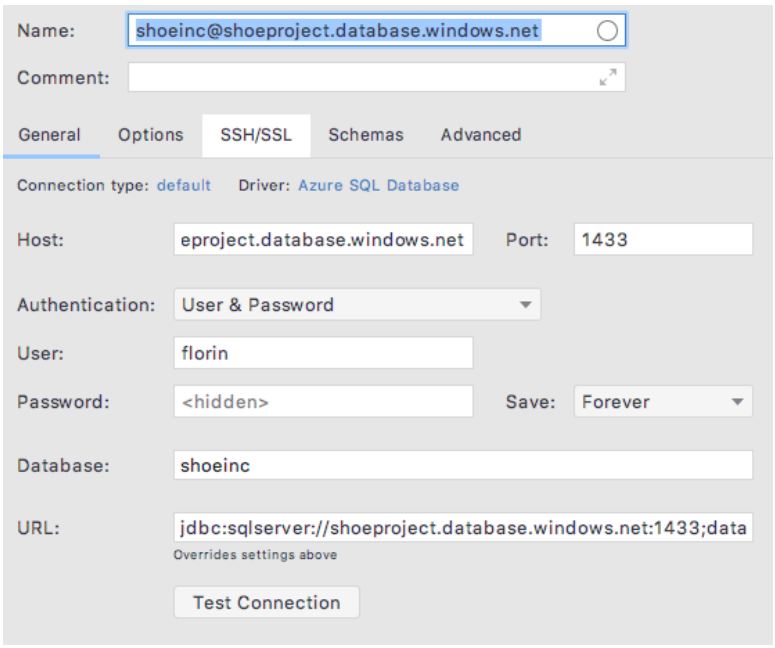


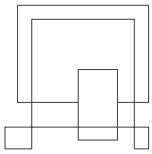
Figure 22 - Rider database connection

Second connection has been in the code by adding the connection string in the file appsettings.json.

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=tcp:shoeinc.database.windows.net;Database=shoeinc;User Id=shoeinc;
```

Figure 23 - ConnectionString to database

After establishing all connections, the database is created and the tables have been created.



4.3.1 Create Product table

The table contains 10 attributes from which Id is the primary key that has been used as a foreign key in other tables. The primary key has the restriction of not being null and as it is defined as IDENTITY, the Id will auto increment whenever a new product is being added in the table.

```
CREATE TABLE Products (
                        Id int NOT NULL IDENTITY PRIMARY KEY ,
                        Brand varchar(255),
                        Model varchar(255),
                        ModelCode varchar(255),
                        Price double precision ,
                        Discount double precision,
                        Description varchar(255),
                        Image varbinary(MAX),
                        ImageUrl varchar(MAX),
                        Stocklevel int,
);
```

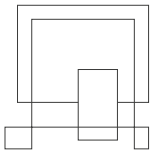
Figure 24 - SQL Create Product table

4.3.2 Create Orders table

The Orders table it is required for whenever a new order is created. The table has a primary key that has the restriction of not being null and it will auto increment as it is defined as IDENTITY.

The OrderNumber is constrained as UNIQUE as it is used as a foreign key in the Orderlines table.

The table also contains a foreign key that will make a reference to a user, which is the customer that created the order.



```
CREATE TABLE Orders (
                        OrderId int NOT NULL IDENTITY PRIMARY KEY ,
                        OrderNumber varchar(255) UNIQUE,
                        UserId int,
                        Date DATE,
                        TotalPrice decimal ,
                        FOREIGN KEY  (UserId) REFERENCES [User](ID) ON DELETE CASCADE
);
```

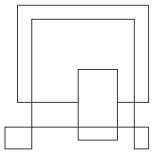
Figure 25 - SQL Create Orders table

4.3.3 Create Orderlines table

The Orderlines table has 5 attributes from which OrderlineId is the primary key. The table contains two foreign keys. The OrderNumber makes a reference to the order to which the Orderline objects belong. The ProductId makes a reference to a Product object that it is contained inside of the Orderline object.

```
CREATE TABLE Orderlines (
                            OrderlineId int NOT NULL IDENTITY PRIMARY KEY ,
                            OrderNumber varchar(255),
                            ProductId int,
                            Quantity int,
                            Price decimal,
                            FOREIGN KEY  (OrderNumber) REFERENCES [Orders](OrderNumber),
                            FOREIGN KEY  (ProductId) REFERENCES [Products](Id) ON DELETE CASCADE
);
```

Figure 26 - SQL Create Orderlines table



5 Test

Testing has been implemented once the implementation phase ended. The purpose of the testing is to verify that all functionalities are according to the requirements. The next sections will describe the testing methodologies applied.

5.1 Integration testing

To test the components of the system it has been used Integration testing to verify that the functionalities are functioning as expected. Every requirement is tested in a test case and the test cases contains prerequisites, steps, expect result and pass or not pass.

As it can be observed the test case starts with the requirement. The prerequisite in this case is that the company manager must be logged in. The steps is to click on User on left side menu and the expected result is to display the list of user information. The results is that the test has passed.

1. Requirement

The company manager must be able to view the list of current registered users.

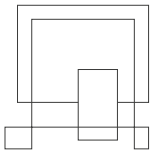
Prerequisites:
Company manager is logged in

Steps:
1.Click Users on left side menu

Expected Result:
List is displayed with user information

Pass or Not: Passed

Figure 27 - Integration testing #1



Another case that has been tested is if the company manager can edit products. The requirement and the prerequisites are defined. The steps are described and the expected results. The test in this particular case has passed.

4. Requirement

The company manager must be able to edit products.

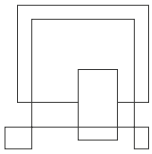
Prerequisites:
Company manager is logged in

Steps:
1.Click Products on left side menu
2.Click Edit button
3.Fill in the new product information
4.Click Ok button

Expected Result:
Product updated, saved and displayed

Pass or Not: Passed

Figure 28 - Integration testing #2



5.2 Acceptance testing

Accepting testing is utilized to verify that all requirements have been fully implemented. The acceptance testing is performed by the user where it is described the expected results and the actual results. At the end it is assed if the test is passed or not.

1. The company manager must be able to view the list of current registered users.

   Expected outcome: Company manager can see the list of users

   Does outcome match expected: YES

2. The company manager must be able to view the list of current products.

   Expected outcome: Company manager can see the list of products

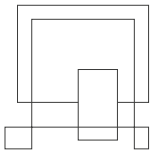
   Does outcome match expected: YES

3. The company manager must be able to add products.

   Expected outcome: Company manager can add a product

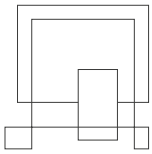
   Does outcome match expected: YES

Figure 29 - Acceptance testing



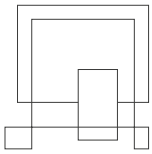
6 Results and Discussion

From all the requirements 16 have been implemented and 12 have not been implemented.



ID	User Story	Status
1	As a customer I would like to be able to create an account so I can log in to my account.	IMPLEMENTED
2	As a customer I would like to be able to add or modify my shipping address so I can have my correct shipping address stored into my account.	IMPLEMENTED
3	As a customer I would like to be able to log in to my account so I can order products.	IMPLEMENTED
4	As a customer I would like to be able to have all products displayed with picture, description, and price so I can make my research.	IMPLEMENTED
5	As a customer I would like to be able to add or remove products in my shopping cart so I can have a final list of products that I would like to purchase.	IMPLEMENTED
6	As a customer I would like to check out my products from the shopping cart so I can finalize my order.	IMPLEMENTED
7	As a customer I would like to be able to modify my email address so I can have my correct email address stored into my account.	IMPLEMENTED
8	As a customer I would like to be able to delete my account so my personal details are not in risk.	IMPLEMENTED
9	As a company manager I would like to be able to check all my customer details so I can have an accurate overview of my customers.	IMPLEMENTED
10	As a company manager I would like to be able to check all my products so I can have an accurate overview of my products.	IMPLEMENTED
11	As a company manager I would like to be able to add new products to my online shop so I can offer them for sale to my customers.	IMPLEMENTED
12	As a company manager I would like to be able to edit/delete my products so I can have an up-to-date offer of products.	IMPLEMENTED
13	As a company manager I would like to be able to apply or remove discount on my products so that I am able to boost my sales.	IMPLEMENTED
14	As a company manager I would like to be able to check on the completed an uncompleted orders so that I am able to keep track of my deliveries.	NOT IMPLEMENTED
15	As a sales manager I would like to be able to view all my sales data from my online shop so I am able to have an overview on my online sales.	NOT IMPLEMENTED
16	As a sales manager I would like to be able to view all my sales data from my shops, so I am able to have an overview on my physical sales.	NOT IMPLEMENTED
17	As a sales manager I would like to have a way of viewing my monthly and yearly sales from my online shop so that I am able to see the sales performance.	NOT IMPLEMENTED
18	As a sales manager I would like to have a way of viewing my monthly and yearly sales from my physical shop so that I am able to see the sales performance.	NOT IMPLEMENTED
19	As a sales manager I would like to have a way of comparing sales from different shops so that I am able to detect my top performant shop.	NOT IMPLEMENTED
20	As a sales manager I would like to have a way of comparing sales from different shops so that I am able to detect my top performant shop.	NOT IMPLEMENTED
21	As a sales manager I would like to have a way of viewing my top sold products so that I am able to understand which product performs better.	NOT IMPLEMENTED
22	As a sales manager I would like to have a way of viewing my top customers based on region, age and gender so that I am able to understand how market acts.	NOT IMPLEMENTED
23	As a sales manager I would like to have a way of viewing my top employee sales performance monthly and yearly so that I am able to detect my top performant employee.	NOT IMPLEMENTED
24	As a sales manager I would like to have a way of viewing the financial efficiency of a discount campaign so that I am able to understand if it has been profitable or not.	NOT IMPLEMENTED
25	As a sales manager I would like to have a way of viewing my overall business profitability so that I am able to report further on to my CEO.	NOT IMPLEMENTED
26	As a warehouse manager I would like to be able to check on my stock information so that I would be able to have an overview over my current stocks	IMPLEMENTED
27	As a warehouse manager I would like to be able to add or delete products in my stocks so I can have an accurate list of products in my warehouse.	IMPLEMENTED
28	As a warehouse manager I would like to be notified if products are about to go out of stock so I can be able to make orders from suppliers	IMPLEMENTED

Figure 30 - Results requirements



7 Conclusions

ShoeInc project has successfully delivered to the customer an end product that satisfies the majority of the functional and non-functional requirements. Exceptions have been in the case of Sales Manager but the foundation has implemented. Each actor of the company that has been described in this project can make use of their assigned use cases. The warehouse manager can have control over the stocks and make changes where it is needed, the company manager can have an overview over its customer and products with changes if needed and the customer can register, login, view the products and make purchases.

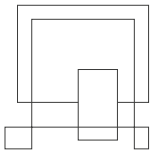
The project has been executed following its natural path where the first phase has been the analysis on the background problem. The requirements of the project have been established according to what the customer and its actor's needs. Further, the use case descriptions and activity diagrams have been drawn for each actor and case, system sequence diagrams to have a representation of the interaction and the domain model that provides a picture of the entities and their roles.

The next phase has been the design where all class diagrams have been created and the sequence diagrams so we can understand how the objects or different attributes and methods affects the system and its classes. Both of these elements, class diagrams and sequence diagrams, are critical so that the implementation phase can commence.

The implementation followed closely the chosen design of the system and it had been implemented accordingly. The system benefited of a web application Blazor that uses the coding language of C# and the database has been created and implemented with the help of SQL.

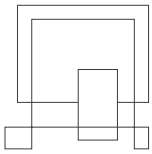
Followed after implementation, the testing section describes all performed tests on the system and the achieved results. As described previously, the system has been tested using Integration and Acceptance testing.

Lastly, the system can demonstrate how the initial planned requirements and functionalities have been delivered to the customer. The system has been successfully implemented and reached the goals of the project. All the involved actors are able to make use of the functionalities of the system.



8 Project future

As required for Shoe Inc, the application can be used to browse and shop products. However, the project manager would like to add some business intelligence uses to the app uses like, sales reports, data cleansing and an additional database to keep all the data organized and safe. With this implemented the company will be able study the possibility of expanding the company around the world.



9 Sources of information

[1] YourCoach. 2021. SMART goals - YourCoach. [ONLINE] Available at: https://www.yourcoach.be/en/coaching-tools/smart-goals/. [Accessed 16 December 2021].

[2] TutorialsPoint. 2021. UML - Activity Diagrams. [ONLINE] Available at: https://www.tutorialspoint.com/uml/uml_activity_diagram.htm. [Accessed 14 December 2021].

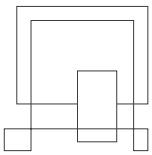
3] Radzen Blazor Components . 2021. Radzen Blazor Components . [ONLINE] Available at: https://blazor.radzen.com/docs/. [Accessed 15 December 2021].

[4] Syncfusion. 2021. What are the application lifecycle methods in Blazor? | Blazor FAQ | Syncfusion. [ONLINE] Available at: https://www.syncfusion.com/faq/blazor/lifecycle/what-are-the-application-lifecycle-methods-in-blazor. [Accessed 16 December 2021].

[5] Code with Shadman. 2021. Repository Pattern C# - Code with Shadman. [ONLINE] Available at: https://codewithshadman.com/repository-pattern-csharp/#:~:text=Repository%20pattern%20C%23%20is%20a,view%20of%20the%20persiste. [Accessed 16 December 2021].

[6] Per-Erik Bergman. 2021. Repository Design Pattern. The repository pattern is one of the… | by Per-Erik Bergman | Medium. [ONLINE] Available at: https://medium.com/@pererikbergman/repository-design-pattern-e28c0f3e4a30. [Accessed 16 December 2021].

[7] What is Integration Testing (Tutorial with Integration Testing Example). 2021. What is Integration Testing (Tutorial with Integration Testing Example). [ONLINE] Available at: https://www.softwaretestinghelp.com/what-is-integration-testing/. [Accessed 15 December 2021].



10 Appendices

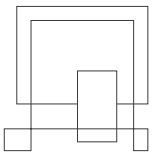
The purpose of your appendices is to provide extra information to the expert reader.

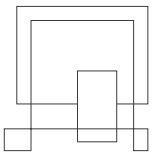
List the appendices in order of mention.

Examples of appendices

- Appendix A – Analysis
- Appendix B – Design
- Appendix C – Testing
- Appendix D – User Guide
- Appendix E – Project Description
- Appendix F – System Source Code

List of Figures:





Appendix Project Description

Project Description
ShoeInc

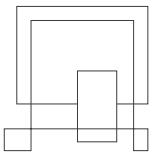
Florin-Leonard Bordei 280593

Jaume Lopez Topping 282231

Software Technology Engineering

7th Semester

17-December-2021



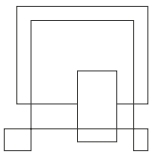
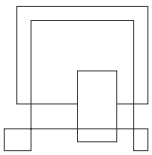


Table of content



1. Background Description

ShoeInc is a local shoes retailer that emerged out of Horsens as an upcoming business that took the market by surprise. As they started with a small local shop in the city center of Horsens, they gained notoriety for their excellent customer service and great quality products sold at bargain prices. Their line of products is representative for any age group, gender, social occasion, sports, or outdoor interests and this helped them to succeed as they could serve any demands of their customers.

In the last 5 years they have been funded by private equity funds to expand with their network of shops and later to expand into the online market also. Therefor, 3 more other shops have been opened in key cities of Denmark: Aarhus, Odense and Copenhagen and one new warehouse facility in Fredericia to serve all their shops and online deliveries. As for the online market, they needed to delay this process as they have been understaffed in the warehouse facility and they have not been able to find a good software solution for the online market and deliveries.

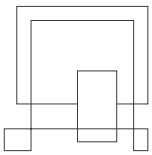
Such a rapid expansion came with a price to be paid in their internal operations of managing current sales and sales predictions, stock management and order fulfilment from their warehouse to their shops.

The ordering system from the shops to the warehouse is still done by the phone and in high season period orders are being misplaced or delayed at a big period where the shops are being not stocked with enough merchandise.

The sales managers from their shops are not being able to record accurately the sales data and there have been situations where they have sent to their HQ errored data, which has been to no use to provide any insights in the purchase tendencies or consumer behavior. Therefore, the company is unable to do any sales predictions or business planning on long term.

As for the online software solution to be able to sell products online, they have not been able to find a good product that fulfilled all their requirements. The available products were contractually bounded with many years of maintenance and high costs.

For these reasons, ShoeInc tries to restructure their current daily operations and would like to acquire new products that would help with their current issues. They are looking for simple solutions but effective and trustworthy.



2. Problem Statement

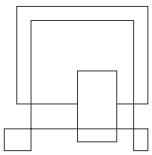
It is observable that ShoeInc is a growing company with internal errors that needs to be addressed for the company to grow even larger. Therefore, the following areas have been identified with problems that needs to be addressed: sales, warehouse, and e-commerce platform.

Main Problem

ShoeInc are losing their potential customers not being able to make online sales, inefficient warehouse operations and not being able to correctly record the sales data.

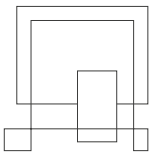
The following sub-questions are formulated to get a better understanding of the main problem:

1. How to enable ShoeInc to make online sales?
2. What kind of data is necessary to improve their sales?
3. What kind of sales reports must be created for different end users?
4. What kind of data analyses must be performed?
5. What is needed to manage stocks effectively?



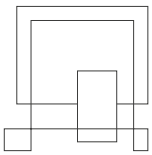
3. Definition of purpose

The purpose is to help ShoeInc with an integrated solution so that the company would be able to manage their sales, improve their warehouse operations and making available their products for the online market.



4. Delimitation

For the moment, the team has been unable to find any relevant delimitations.



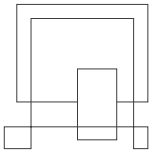
5. Methodology

The selected software development process for this project is SCRUM, with few alterations from the original format.

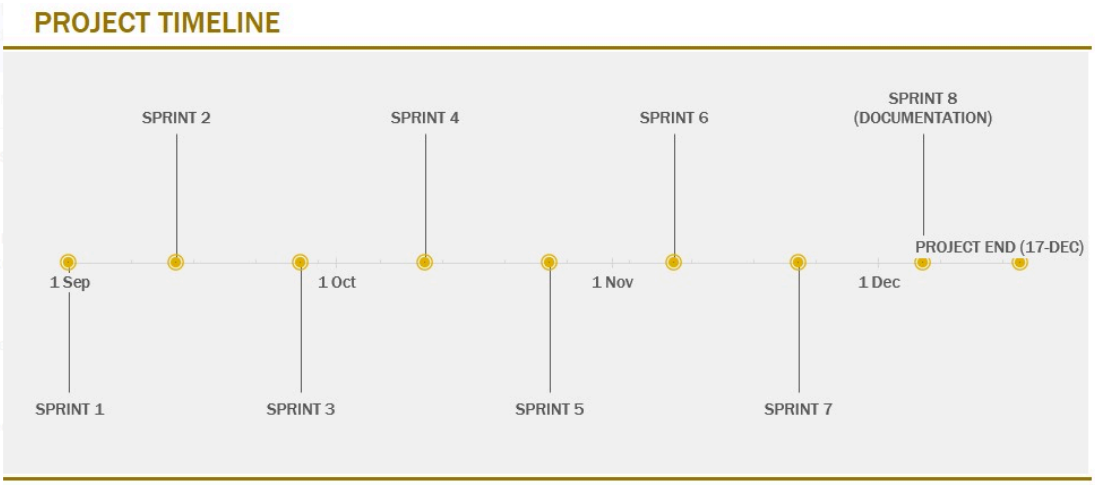
The team members will assume all roles, therefore there will be no specific roles such as Scrum Master, Product Owner or Developer.

The SCRUM activities and artifacts that will be used are:

- Product Backlog
- Sprints with their assigned Sprint Backlog
- Sprint Planning

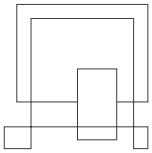


6. Time schedule



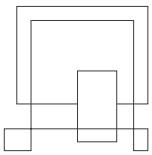
PROJECT DETAILS

DATE	MILESTONE	POSITION
1-Sep	Sprint 1	-10
13-Sep	Sprint 2	10
27-Sep	Sprint 3	-10
11-Oct	Sprint 4	10
25-Oct	Sprint 5	-10
8-Nov	Sprint 6	10
22-Nov	Sprint 7	-10
6-Dec	Sprint 8 (Documentation)	10
17-Dec	Project End (17-Dec)	0



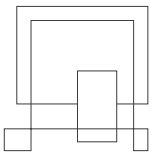
7. Risk assessment

Risks	Likelihood Scale: 1-5 5 = high risk	Severity Scale: 1-5 5 = high risk	Product of likelihood and severity	Risk mitigation e.g. Preventive- & Responsive actions	Identifiers	Responsible
Increasing the complexity of the project	4	5	20	Follow the requirements	Implementing functionality that is not a requirement	Jaume
COVID-19 pandemic	5	5	25	Notify the team about the health condition	Temperature, flu symptoms	Florin



8. Sources of Information

Scrumguides.org. 2021. [online] Available at: <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf> [Accessed 3 May 2021].



Appendices

Group Contract

Group number 10 Date: **03-05-2021**

These are the terms of group conduct and cooperation that we agree on as a team.

Participation:

We agree to equally share the workload and respect de delivery deadlines set by the team. Each task will have an estimated working hour and the workload will be divided based on that.

Individual work assignments will be given to each group member to fulfil until the next meeting. To ensure that all members are at the same level, the group will have a small talk at the beginning of the meeting about the current topics.

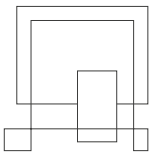
If any assignments will be missed, the individual will be warned about his performance an if continues, his lack of contribution will be highlighted in the group documentation.

Communication:

Primary written communication channel will be through Whatsapp group chat and online meetings will be done via Microsoft Teams. The group agreed for a daily communication, if there are any problems, within the timeframe of 09:00 – 19:00. A maximum 12-hour response time is expected.

Cancellations from any participation should be announced at least 24 hours prior to the meeting.

Meetings:



It is expected that all meetings will be physical but with one week prior it is possible to change for a remotely format. For the remote meetings, the group will use Microsoft Teams.

The meetings will be every Wednesday from 08:00 – 16:00 and, if necessary, every Monday the group will have a short meeting to see the status of the individual assignments and what things are to be worked on for the Wednesday meeting.

The group will keep minutes of each meeting and at the end of the day, the group will plan tasks for the next upcoming meeting.

If one member is not able to participate in a meeting, the group will try to reschedule for another day to meet.

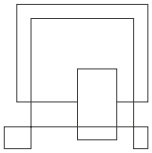
Conduct:

We expect each member to respect every other member's preferences and opinions on the status of the project and direction of it. We expect that we will have a free platform to express and debate our ideas or our concerns.

Conflict:

The conflicts should be tried to be solved in a peaceful manner, both group members being responsible for resolving the potential conflicts.

Main responsible as a conflict mediation will be Florin Bordei. In case of major conflict, the group members should stop the meeting and reschedule it for another day to approach it more calmly.



Deadlines:

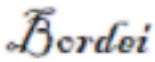
We agree to have all hand ins/ tasks/ assignments completed with 24 hours before the deadline, so we can review the work.

Responsible for ensuring the deadlines is Jaume Lopez who must send a reminder message with 48 hours before the deadline.

Consequences of missing deadlines by one of the team members is a fine, a pizza for all the group.

Other Issues:

Any limitation of a participant should be addressed to the group as they arise.

Group member's name	Student number	Signature
Jaume Lopez	282231	
Florin Leonard Bordei	280593