



Zadání diplomové práce

Název:	Frontend administrace e-shopu
Student:	Bc. Iuliia Evseenko
Vedoucí:	Ing. Jiří Hunka
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Webové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Cílem této práce je důkladný návrh a následná prototypová realizace frontendu pro vznikající novou administraci e-shopů skupiny Stylka.cz. Vzhledem k velkému rozsahu funkcí daného e-shopu je třeba zaměřit se především na všechny procesy týkající se objednávek a produktů. Pokrytí funkcionalit musí respektovat funkcionalitu současné staré administrace.

Postupujte v těchto krocích:

1. Analyzujte současné řešení e-shopové administrace.
2. Proveďte analýzu možných řešení, navrhnete vhodný papírový model.
3. Při návrhu přihlídněte k budoucímu rozšiřování administrace.
4. Konzultujte návrh alespoň s jedním uživatelem současné administrace.
5. Implementujte výsledný návrh.
6. Při realizaci nezapomeňte na důkladné testování.
7. Zhodnoťte výsledné řešení, navrhnete úpravy do budoucna.



**FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE**

Master's thesis

E-shop administration frontend

Bc. Iuliia Evseenko

Department of Software Engineering

Supervisor: Ing. Jiří Hunka

May 6, 2021

Acknowledgements

I would like to thank the supervisor of this master thesis Ing. Jiří Hunka for consultations and for helping me in writing this thesis. Thanks to the e-shop employees who consulted me and devoted their time to clarify all the details. I would also like to thank my family and friends for their priceless support in writing this work.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No.121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on May 6, 2021

.....

Czech Technical University in Prague

Faculty of Information Technology

© 2021 Iuliia Evseenko. All rights reserved.

This thesis is school work as defined by Copyright Act of the Czech Republic. It has been submitted at Czech Technical University in Prague, Faculty of Information Technology. The thesis is protected by the Copyright Act and its usage without author's permission is prohibited (with exceptions defined by the Copyright Act).

Citation of this thesis

Evseenko, Iuliia. *E-shop administration frontend*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2021.

Abstrakt

Cílem této práce je analýza stávající aplikace pro správu e-shopů, pochopení business domény a procesů. Dalším cílem je návrh low-fidelity prototypu a jeho následná konzultace se zaměstnanci e-shopů. Dále následuje implementace prototypu aplikace s použitím knihovny React. Výstupem této diplomové práce je funkční a důkladně otestovaný prototyp, splňující požadavky, jež vznikly během analýzy.

Klíčová slova GUI, UX, UI, React, e-shop, OpenCart, Balsamiq, analýza, návrh, webová aplikace, JavaScript, TypeScript

Abstract

The aim of this master thesis is to analyze the existing application for e-shops management, understanding the business domain and processes. Another goal is to design a low-fidelity prototype and its subsequent consultation with e-shop employees. Then follows the implementation of a prototype application using the React library. The output of this master thesis is a functional and thoroughly tested prototype, meeting the requirements that arose during the analysis.

Keywords GUI, UX, UI, React, e-shop, OpenCart, Balsamiq, analysis, design, web application, JavaScript, TypeScript

Contents

Introduction	1
1 Analysis	3
1.1 UX design	3
1.2 Domain analysis	4
1.2.1 Orders page	5
1.2.1.1 Orders	5
1.2.1.2 Order detail	8
1.2.2 Products page	14
1.2.2.1 Products	14
1.2.2.2 Product detail	15
1.3 Prototyping tools	22
1.3.1 Analysis of the wireframing tools	22
1.3.1.1 Balsamiq	23
1.3.1.2 Moqups	24
1.3.1.3 Wireframe.cc	24
1.3.1.4 Mockitt	25
1.3.1.5 Conclusion	26
2 Design	27
2.1 Prototyping	27
2.1.1 Wireframes	28
2.1.2 The lo-fi prototype	28
2.1.2.1 Orders page	30
2.1.2.2 Order detail page	33
2.2 Technologies	38
2.2.1 Potential problems	41
2.3 Communication with the server	41
2.4 Security	42

2.4.1	CORS	42
2.4.2	Web attacks	42
2.4.3	Authentication and authorization	44
3	Realization	47
3.1	React libraries stack	47
3.2	Architecture of the app	48
3.3	Model	48
3.4	Services and transformers	49
3.5	Internationalization	52
3.6	Implemented prototype	53
3.7	Integration with the back-end application	55
3.7.1	Authentication and authorization	55
3.7.1.1	CORS	55
3.7.1.2	Sessions	56
3.7.1.3	Authentication	57
3.7.2	Problems	57
3.7.2.1	Documentation	58
3.7.2.2	Shipping methods	59
3.7.2.3	Order status	61
3.7.2.4	Payment methods	63
3.7.2.5	Performance	63
3.7.2.6	Suggestions for improvement	66
3.8	Saving user state	67
4	Testing	69
4.1	Development testing	69
4.2	User acceptance testing	69
4.3	Usability testing	70
4.3.1	Test plan	70
4.3.2	Scenarios	72
4.3.2.1	Orders page	72
4.3.2.2	Order detail page	74
4.3.3	Persons and test results	75
4.3.4	Further improvements	87
4.4	Conclusion	88
	Conclusion	89
	Bibliography	91
	A Acronyms	95
	B Contents of enclosed CD	97

List of Figures

1.1	Orders page	6
1.2	Order detail page	9
1.3	Products table with shipping and payment fee row	11
1.4	Products page	14
1.5	Product detail - General tab	16
1.6	Product detail - Data tab	18
1.7	Product detail - Special offer tab	19
1.8	Product detail - Images tab	20
1.9	Product detail - Domain settings tab	20
1.10	Balsamiq user interface	23
1.11	Moqups user interface	24
1.12	Wireframe.cc user interface	25
1.13	Mockitt user interface	26
2.1	Sketch of the orders page	29
2.2	Sketch of the order detail page	29
2.3	Lo-fi prototype of the orders page	30
2.4	The “Was paid” setting dialog	32
2.5	Hide column functionality	32
2.6	The lo-fi prototype of the order detail page	33
2.7	Order detail editing	34
2.8	The products panel in the order detail	35
2.9	New product	35
2.10	New shipping or payment method	36
2.11	Packages panel in the order detail	36
2.12	New package	37
2.13	History panel in the order detail	38
2.14	Update an order	39
2.15	The “Select invoice number” dialog	40
2.16	Preflight request mechanism	43

3.1	Login page	53
3.2	Orders page	54
3.3	Order detail page	54
3.4	Documentation page of the endpoints	58
4.1	Custom columns setting of the first participant	76
4.2	Custom detail setting of the first participant	78
4.3	Custom columns setting of the second participant	79
4.4	Custom detail setting of the second participant	81
4.5	Custom columns setting of the fifth participant	86
4.6	Custom detail setting of the fifth participant	87

Introduction

Nowadays it is hard to imagine a world without e-shops. You can find and buy almost everything on the internet. There are a huge amount of e-shops and everyone tries to do its graphical user interface as attractive as possible. The graphical user interface is very important in modern software applications because the more “user-friendly” it is, the more people want to use this application. Every e-shop tries to attract more users doing their graphical interface good-looking, stylish, and comfortable to use. But every e-shop requires an administration application to manage orders, products, shipments, etc. How quickly the employee will navigate in the system and fulfill their responsibilities, related to e-shop operation, depends on how the administration graphical interface good is. Thus the administration graphical interface is equally important as the graphical interface of the e-shop.

The goal of this thesis is to design a new GUI for the administration application of www.stylka.cz e-shops group. My motivation for choosing this master’s thesis was the potentiality to improve my UX knowledge and skills. The other motivation was a benefit for e-shop employees, who could use more user-friendly application, what will be created within this thesis. Another big motivation for choosing this thesis was an opportunity to improve my front-end development skills. The application is written using modern languages and frameworks.

This work is a follow-up to the bachelor’s thesis of colleague Bc. Radomír Koudela. His part was the creation of the back-end application, which will serve a REST API.

The goals of this master’s thesis are analyzing the current version of the e-shop administration application, preparing a draft of the new application design respecting the user experience principles, and creating a prototype of the new web application.

The application will respect all the principles of proper software design. In the first version, there will be only pages related to orders, but the application will be easily expandable, thanks to its modular architecture.

This thesis is divided into chapters. In chapter Analysis, I will analyze the current version of the application, find out the problems within the e-shop administration application, explore the requirements of the application users. After then I'll suggest a way, how to solve discovered problems.

In chapter Design, I will demonstrate my design proposal for a new application's GUI. Then I'll describe the technologies are used within the application realization. The last part of this chapter will be the description of the security design, that will be implemented in this thesis.

In chapter Realization, I will describe the process of web application creation. There will be technologies and components, application architecture, integration with the back-end application, and problems related to it.

In chapter Testing, I will perform testing of the created prototype. One of the performed tests will be a usability testing with the real users of the current version of the e-shop administration application.

In the last chapter Conclusion, I will summarize the work, describe what requirements from the analysis were done, which intentions were achieved, and which were not.

Analysis

This chapter consists of three parts: 1.1 UX design, 1.2 Domain analysis, and 1.3 Prototyping tools. The 1.1 UX design section describes what user experience design is, why is it so important and which principles should be followed to create a great user experience. In the 1.2 Domain analysis section, I analyzed the business domain, walk through each field on analyzed pages explaining their meaning. Then I summarize all the user needs and problems related to each page. In the last 1.3 Prototyping tools sections, I analyzed prototyping tools to choose which one I will to crate a low-fidelity prototype.

1.1 UX design

User experience (next UX) design is the process of product designing that provides meaningful and relevant experiences to users. [1]

The main requirement of a good user experience is to meet the particular user's needs, without fussing the user. UX designer should understand in which context, for which purpose, and how user would use the designed product.

It is usually tended to mistake the user experience and the user interface (UI). The UI is an extremely important part of the design, but it is important to distinguish UX and UI. The UX focused on how the user solves a problem, the UI focused on how the product looks and functions.

We should also distinguish UX and usability: According to the definition of usability, it is a quality attribute of the UI, covering whether the system is easy to learn, efficient to use, pleasant, etc. Again, this is very important, but the total user experience is an even broader concept. [2]

Usability is a quality attribute that measures how easy user interfaces are to use and how well a specific user can achieve a defined goal effectively, efficiently and satisfactorily.

Usability is defined by 5 quality attributes:

Learnability - How easy for users to execute basic tasks the first time they face the design?

Efficiency - Once users have learned the design, how quickly can they perform tasks?

Memorability - When users return to the design after a period of not using it, how easily can they restore skill?

Errors - How many errors do users make, how critical are these errors, and how easily can they fix them?

Satisfaction : How pleasant is it to use the design?

There is an other important quality attribute - utility. Utility refers to the design's functionality: Does it do what users need? Usability and utility are equally important and together determine whether something is useful. [3]

The most basic and useful method for studying usability is usability testing that has 3 components:

- Representative users
- Representative tasks
- Observation what the users do, where they succeed, and where they have difficulties with the user interface.

1.2 Domain analysis

This section describes the current state of the administration application and user needs assessment. Each subsection consists of the page description, the meaning of the important fields within the page from the business perspective, and a summary of the changes, which should be done or could be done in the further version of the web application.

I have analyzed the **Orders** and **Products** pages. For the **Products** page the analysis and the user's requirements collection are done, see 1.2.2 section. Whereas for the **Orders** page I have created a mockup, which will be described in the section 2.1.

It is worth saying a few words about the business context. Every e-shop has an administration page, where the whole management is done for a smooth operation of the e-shop. From the user's perspective, they visit the e-shop page, choose the products they want, fill all fields related to the delivery, select the payment method, and place the order. From now on, they wait for their purchase.

But from the e-shop perspective, it is not such a simple process. Before the e-shop is able to operate, an administrator should upload all the required

data and options. For each product that a user can see on the e-shop page, the administrator has to write descriptions and upload images, specify possible sizes and colors, set prices and so on. Products have to be sorted by categories for easier user navigation on the web. The administrator has to link all related products.

After the e-shop receives an order, it has to be processed by the administrator. The administrator has to check if the products are in stock, if the order was paid, if there are comments from the customer. Then, if there are no special requirements from the customer, the administrator or an e-shop employee should complete all necessary information related to the delivery and packaging. Next, the products must be taken from a warehouse. Finally, the order is packaged and sent via a shipping method user selected.

I consulted e-shop employees on all application requirements. They described their work process with the current web application, which fields are important for them to increase the efficiency of their work, what certain fields from the business point of view mean. Using their knowledge in the field I made a list of all suggestions for change, that you can see in the sections below. The e-shop employees, especially my supervisor Ing. Jiří Hunka, gave me invaluable knowledge about the business domain, which helped me to create a new design of the administration web application.

1.2.1 Orders page

Orders page consist of two pages: **Orders** and **Order detail**. **Orders** contains an overview of all orders, that came from the e-shop. There should be main pieces of information about orders for employees' easy navigation. The **Order detail** page contains more specific information about an order.

1.2.1.1 Orders

In Figure 1.1 is shown the current version of the orders page in the administration application.

There are some important fields within the table:

- **Order ID** - an ID of the order in the system
- **Invoice number** - is a unique, sequential code that is systematically assigned to invoices. The value in the column is empty if the invoice for an order hasn't been created
- **Domain** - URL of an e-shop an order came from
- **Customer name** - the name of a customer, who made an order
- **Order status** - current status of an order. There are some important statuses:

1. ANALYSIS

ID obj.	Č. Faktury	Doména	Jméno zákazníka	Stav	Doprava	Datum vytvoření	Celkem	Uhrazeno	Zisk	Příšel z	Souhlas	Akce
35692	20210001	styka.cz	Jan Veverka	Odesláno dopravcem	Zásilkovna	06. 01. 2021 22:14:05	3 000,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35674	20210001	styka.cz	Jan Veverka	Odesláno dopravcem	Zásilkovna	06. 01. 2021 18:07:41	-214,0 Kč	ano	47,1 Kč	Google	Ano	[Upravit]
35672	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 18:00:26	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35670	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 17:58:17	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35668	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 17:37:41	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35666	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 17:31:54	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35663	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 17:22:39	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35661	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 17:11:07	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35659	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 16:27:48	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]
35657	20210001	moje-mediana.cz	Jan Veverka	Odesláno dopravcem		06. 01. 2021 16:19:45	1 089,0 Kč	ano	420,0 Kč	Google	Ano	[Upravit]

Figure 1.1: Orders page

Waiting for payment - an order is not paid yet

To the expedition - order has been physically prepared for shipping

Prepared for a personal pickup - order is waiting for a customer to be personally picked up

Was sent by a carrier - an order was sent by a shipping company, which the user has selected

Customer took goods - a customer received an order or picked up it from a store

Shipment only - flag to identify a specific type of orders. There are situations when an employee has to create a fake order, to register in the system only shipping events. For example, when some product should be sent to a supplier or some product should be sent to a winner of a competition

- **Shipping method** - the way the order will be sent
- **Added** - date, when the product was added
- **Total** - total price with VAT
- **Was paid** - if an order already has been paid
- **Profit** - profitability of a single order
- **Referrer** - from what site a customer came to an e-shop
- **Agreement** - if a customer agrees to receive a satisfaction questionnaire

On the top of the table is a selectbox where the user can choose what they want to print. There is a list of different product filter options. For example, users can print a list of all products, which are within the orders with the “For expedition” status. We have an agreement, that this feature will migrate to the warehouse system of e-shops.

Features, that are done in the prototype:

- Full-text search within the table
- A pagination within the table, fixed table height, a user would be able to select how many rows they want to see
- A user will be able to go to an order detail page after double clicking a row in the orders table
- An opportunity to set order was paid without going to the order’s detail page
- A user can hide some columns, the application will remember the user’s layout
- **Profit** column could be colored based on the profit value
- A complex date filter for the **Added** column, an opportunity to filter data within an interval
- Add new columns:
 - **Viewed** - if an order was viewed by an e-shop employee
 - **Comment** - if there is a comment from a customer
 - **Shipping label** - if a shipping label was already created
 - **Payment method** - a payment method of an order
 - **Slovakia** - if an order came from the Slovak Republic
 - **In stock** - if all products within an order are in a warehouse

Features, that could be implemented in the further version:

- Double clicking the field in the order table - add the value to the filtering
- A configurable color palette for statuses, profit, and similar information

1.2.1.2 Order detail

This section describes **Order detail** page, which pieces of information it contains and how the page is organized. After consulting e-shop employees, I wrote up changes users need to see.

In Figure 1.2, there is an example how a current version of the order detail page looks like. This page is organized into sections. Next, the sections explanation follows.

The upper panel

There are four buttons: **VAT on the Slovak Republic**, **Invoice**, **Set was paid**, and **Back**

VAT on the Slovak Republic button removes VAT value from the total price because the customer is not in the Czech Republic.

Invoice button allows users to print the invoice for a certain order.

Set was paid button allows a user to change the order status to “Was paid”.

Back button brings a user back to the **Orders** page.

Order detail

Order detail section contains general information about an order. There are **Order ID**, **Invoice number**, **Payment method**, **Shipping method**, **Added**, and **Domain** fields. Almost all these fields are described in the 1.2.1.1 section.

There are all payment methods:

- **Cash on delivery** - a customer will pay after order receiving
- **By card** - an order was paid by card
- **By cash** - a customer will pay while personally picking up
- **Via Bank transfer** - an order will be paid via a bank transfer. A payment instruction will be sent to a contact email

Contact information

This section contains customer contact information. There are **E-mail**, **Telephone**, and **Fax**. Nowadays, the fax is not used, so in the new version the fax field is not required.

1.2. Domain analysis

OpenCart | ADMINISTRATION Přihášení admin

Náhledy Katalog Rozšíření **Prodej** System Záznamy Zásilký nápověda E-shop Odtlačení

Úvod - Objednávka DPH na Slovensku Faktura Nastavit zaplacení Zpět

Objednávka

Podrobnosti objednávky

ID Objednávky	Datum vytvoření	Způsob úhrady	Způsob dopravy
95642	13. 12. 2020 23:40:20	Hotově, zaplacení ano	Geis
Faktura č.: 20200001		Doména: www.moje-medisana.cz	

Kontaktní údaje

E-Mail	Telefon	Fax
o@gmail.com	+42055	5ad5fa

Adresy

Fakturační adresa	Adresa pro doručení
Hyundai Na poště 7 Jeseník 75501 Česká republika IČ: 15611554 DIČ: 15611554	Hyundai Pátek 703/14 Lipová - Lázně 755 01 Česká republika

Produkty

	Produkt	Model	Záruka	Interní	Externí	Množství	Cena / kus	Cena / kus	Celkem	
<input type="checkbox"/>	lamazaf [otolat.ecod]	Holci strojek Thovt	88396	24	-4	-2	6	100.0 Kč	121.0 Kč	600.0 Kč
<input type="checkbox"/>	lamazaf [otolat.ecod]	Dárek k 88396 Holci strojek Thovt	88396	24		-6	6	100.0 Kč	121.0 Kč	600.0 Kč
<input type="checkbox"/>	lamazaf [otolat.ecod]	Holci strojek Thovt	88396	24		-6	6	100.0 Kč	121.0 Kč	600.0 Kč
									Cena celkem bez DPH	1800 Kč
									DPH 21%	378 Kč
									Cena celkem s DPH	2178 Kč
									Vybráno:	Oubrosit

Komentář k objednávce

Objednávka zaplacená.

Historie objednávky

Datum vytvoření	Stav	Zákazník informován
13. 12. 2020 23:40:56	Nevytvářeno	Ano
Komentáře		
Objednávka zaplacená.		
Datum vytvoření	Stav	Zákazník informován
13. 12. 2020 23:43:29	Odesláno dopravcem	Ano
Komentáře		
Objednávka zaplacená.		

Zásilký (Česká pošta) sbalit/rozbalit

Zásilký (Geis) sbalit/rozbalit

Nejsou evidovány žádné zásilký

Zásilký (Zásilkovna) sbalit/rozbalit

Aktualizovat objednávku

Stav: Informovat zákazníka:

Komentáře

S pozdravem
Kateřina Harková, DIS
Moje-Medisana.cz

[Uložit](#) [Storno](#)

OpenCart © 2009 All Rights Reserved.

Figure 1.2: Order detail page

Addresses

This section contains customer addresses. There are two addresses - payment address and shipping address. One main requirement from e-shop employees was to warn a web application user when these addresses are not the same. When the user sees that addresses are different, it allows him to orient faster in the detail.

Products

This section describes a table with products within the order. Every product row contains the following information

- **Name** - a product name
- **Model** - a product model
- **Guarantee** - warranty period, mostly 24 months
- **External warehouse quantity** - how many pieces from the total quantity should be ordered to the internal warehouse
- **Internal warehouse quantity** - how many pieces from the total quantity are from the internal warehouse
- **Quantity** - ordered quantity
- **Price/Piece without VAT** - price of one piece of the product without VAT
- **Price/Piece with VAT** - price of one piece of the product with VAT
- **Total** - price of the product item without VAT ($\text{Quantity} \times \text{Price/Piece without VAT}$)

This table also contains information about the shipping price and the payment fee. In Figure 1.3 a package table is illustrated, where the last row shows payment and shipping methods details with their total price. Typically, only the **Cash on delivery** payment method has a fee, while other methods do not.

Finally, a button at the bottom of the section is “Create a credit note”. A credit note is effectively a negative invoice - it is a way of showing a customer that they do not have to pay the full amount of an invoice. A credit note might either cancel an invoice out completely if it is for the same amount as the invoice, or it might be for less than the invoice. [4] This functionality is very important in the e-commerce domain, so it will remain in the new version of the administration application.

Produkty										
	Produkt	Model	Záruka	Interní	Externí	Množství	Cena / kus	Cena / kus	Celkem	
<input type="checkbox"/>	Remington Kulma na vlasy Multistyle S8670	S8670	36		-1	1	990.0 Kč	1 199.0 Kč	990.0 Kč	
<input type="checkbox"/>	Základní + Bankovní převod		24			1	0.0 Kč	0.0 Kč	0.0 Kč	
									Cena celkem bez DPH	990.0 Kč
									DPH 21%	208.1 Kč
									Cena celkem s DPH	1 199.0 Kč
										Vybrání <input type="button" value="Doplnit"/>

Figure 1.3: Products table with shipping and payment fee row

The main problem is that products are not connected with the warehouse system. That is the name, model, and other product data a user has to fill manually. In the new version of the administration web application, there should be a connection with the warehouse system, to load a list of products with all possible properties, which are used within the table. The same problem is with payment methods and shipping methods. In the new application, the name of the methods and their prices should be loaded from a back-end application.

Comment

This section contains a customer comment from the order.

History

The **History** section contains historical data about an order. Every update event is contained in this section. This section consists of list of history events, every event contains four fields: **Creation date**, **Status**, **Notify customer**, **Comments**. **Creation date** is a date when the update was done. Remaining fields are described in 1.2.1.2 section.

Packages

The **Packages** section tracks information about created packages for an order. In the current version of the application there are three types of packages based on a carrier. Each type of a package has different fields a user has to fill.

The list of carriers with their specific required fields are below:

- **Czech Post**

Package ID - unique package ID

Created - creation date of a package

Cash on delivery - price that a customer should pay on delivery

Type of a package - three options of delivery:

Delivery To Hand

Delivery To Post Office

Balíkovna - parcel pickup outlets

Status - tracking status of a package

To - addressee name and surname

Weight - weight in kilograms

- **GEIS**

Package ID - unique package ID

Created - creation date of a package

Cash on delivery - price that a customer should pay on delivery

Message to a courier - text field with a message

Message to a recipient - text field with a message

Status - tracking status of a package

To - an addressee name and surname

- **Zásilkovna**

Branch ID - destination branch identification number

Created - creation date of a package

Cash on delivery - price, which a customer should pay on delivery

Weight - weight in kilograms

When this order detail section was being analyzed, there was a persuasion, that if the user has created a package they could not remove it from the system. But then, a link has been noticed, which looks like a text within the package detail. The text of the link is “Remove a package” and after a click, the system displays a modal dialog, if the user really wants to remove a package. This behavior is misleading and it will be also changed in the new application.

Another functionality which relates with the packages is printing a shipping label. Shipping labels display the key information for a carrier to transport a package from their start destination (the warehouse) to its end destination (the customer’s address). The key information includes: postal code, country, tracking number, date, package quantity as well as the weight, address, validation, and ship street, city and state (area).

Update an order

E-shop employees have to update the order status from time to time. For instance, the employees check an order, get all products within the order from a warehouse and send them by a carrier. The order status should be changes from “For expedition” to “Was sent by a carrier”. Usually a customer

should be noticed about this event. After a user submits an update, the new information will appear in the **History** section.

This section contains three fields:

- **Status** - a new status of an order
- **Notify customer** - checkbox to determine if the customer will receive an information e-mail
- **Comments** - text, which will be sent to a customer if the **Notify customer** checkbox is checked

There was a request to prefill the **Comments** text area with the text corresponding to the changed order status.

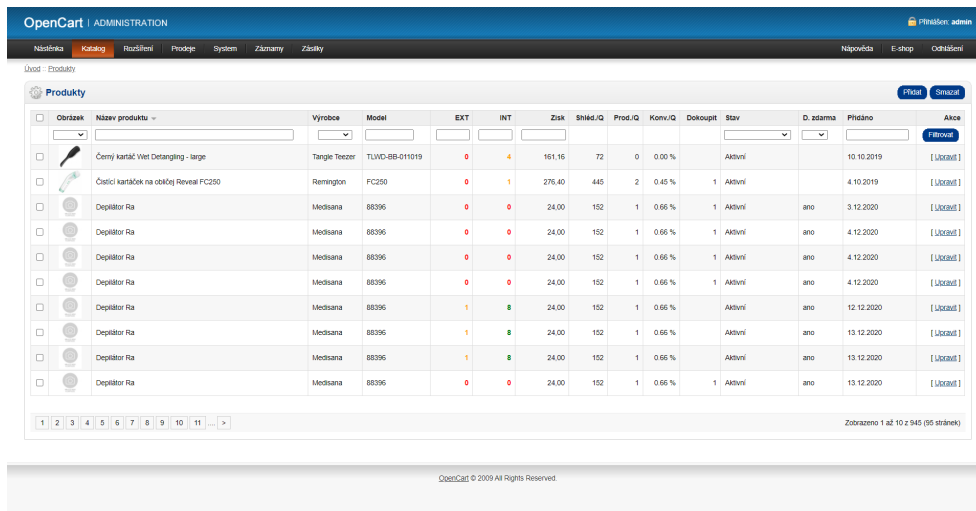
Features, that are done in the prototype:

- Warn about the different payment and shipping addresses
- Choosing the invoice number in some time period
- Every panel in the order detail is an autonomic unit, so it can be edited independently
- Adding the new shipping package is implemented within the order detail page
- Hide some details about the package, for example, the comment for the carrier
- Add changes to history, after every change
- Delete the package when the shipping row is deleted
- Delete the Fax field from a contact information
- Well visible button to remove a package
- Well visible button to print a shipping label
- Connecting products section with the warehouse system

Features, that could be implemented in the further version:

- Warn if the shipping method differs from the carrier in the packages panel
- Prefill the **Comments** text area within the **Update an order**

1. ANALYSIS



The screenshot shows the OpenCart Administration interface. The top navigation bar includes 'OpenCart | ADMINISTRATION' and a user profile 'Přihlášen: admin'. Below the navigation bar, there are tabs for 'Náhledy', 'Katalog', 'Rozšíření', 'Prodeje', 'System', 'Zálohny', and 'Zásilký'. The main content area is titled 'Produkty' and contains a table of products. The table has columns for 'Obrázek', 'Název produktu', 'Výrobce', 'Model', 'EXT', 'INT', 'Zisk', 'SHled./Q', 'Prod./Q', 'Kom./Q', 'Dokoupit', 'Stav', 'D. zdarma', 'Přidáno', and 'Akce'. The table lists 10 products, including 'Černý kartáč Wet Detangling - large', 'Čistič kartáčů na oběhové Reveál FC250', and several 'Depilátor Ra' models. The table also includes a pagination bar at the bottom showing 'Zobrazeno 1 až 10 z 940 (95 stránek)'.

Obrázek	Název produktu	Výrobce	Model	EXT	INT	Zisk	SHled./Q	Prod./Q	Kom./Q	Dokoupit	Stav	D. zdarma	Přidáno	Akce
	Černý kartáč Wet Detangling - large	Tangle Teezer	TLVLD-BB-011019	0	4	161,16	72	0	0,00 %		Aktivní		10.10.2019	[Upravit]
	Čistič kartáčů na oběhové Reveál FC250	Remington	FC250	0	1	276,40	445	2	0,45 %	1	Aktivní		4.10.2019	[Upravit]
	Depilátor Ra	Medisana	88396	0	0	24,00	152	1	0,66 %	1	Aktivní	ano	3.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	0	0	24,00	152	1	0,66 %	1	Aktivní	ano	4.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	0	0	24,00	152	1	0,66 %	1	Aktivní	ano	4.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	0	0	24,00	152	1	0,66 %	1	Aktivní	ano	4.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	1	8	24,00	152	1	0,66 %		Aktivní		12.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	1	8	24,00	152	1	0,66 %		Aktivní		13.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	1	8	24,00	152	1	0,66 %		Aktivní		13.12.2020	[Upravit]
	Depilátor Ra	Medisana	88396	0	0	24,00	152	1	0,66 %	1	Aktivní	ano	13.12.2020	[Upravit]

Figure 1.4: Products page

1.2.2 Products page

As mentioned in the introduction to this section, **Products** page has been analyzed, but there is no mockup designed and the products pages are not implemented in the prototype.

1.2.2.1 Products

In Figure 1.4 the current version of the products page within the administration application is shown. This page contains an overview about all products, which are available in e-shops.

There are some important fields within the table:

- **Image** - a main image of a product, has a big importance for users of the application
- **Producer** - a producer of a product
- **Model** - a product model
- **External warehouse quantity** - how many pieces are in the external warehouse
- **Internal warehouse quantity** - how many pieces are in the internal warehouse
- **Profit** - profitability of a single product
- **Viewed/Quarter** - how often a product has been viewed in a quarter

- **Bought/Quarter** - how often a product has been bought in a quarter
- **Conversion/Quarter** - the ratio of the sold pieces to the pieces viewed
- **Buy** - how many pieces of product should be bought
- **State** - state of a product, has a big importance for users of the application. There are three states:
 - Active** - product is on offer in e-shops
 - Non-Active** - product is not already on offer in e-shops
 - Active/Finished** - product is still in offer in e-shops, but it will be deactivated on a specific date
- **Free delivery** - when the profit of the product is greater than 300 Kč, there is a free delivery
- **Added** - date, when the product was added

Features, that could be implemented in the further version:

- The checkboxes within rows could be deleted, there is no deletion of the products yet
- User could hide some columns, application will remember the user's layout
- Add new product state - **Not in stock**
- Add new columns:
 - Changed** - date, when the product was changed
 - External** - link to the product page in the external warehouse system
 - Internal** - link to the product page in the internal warehouse system
 - Transfers** - link to the product's transfers within the warehouses

1.2.2.2 Product detail

This section describes the information available on the Product detail page, how it is organized, and what changes users want to see on this page to increase the efficiency of their work while using it. Let's go through all tabs on the page.

There are **General**, **Data**, **Parameters**, **Choices**, **Packaging**, **Discount**, **Special offers**, **Images**, **Domain settings**, and **Order** tabs.

General tab

1. ANALYSIS

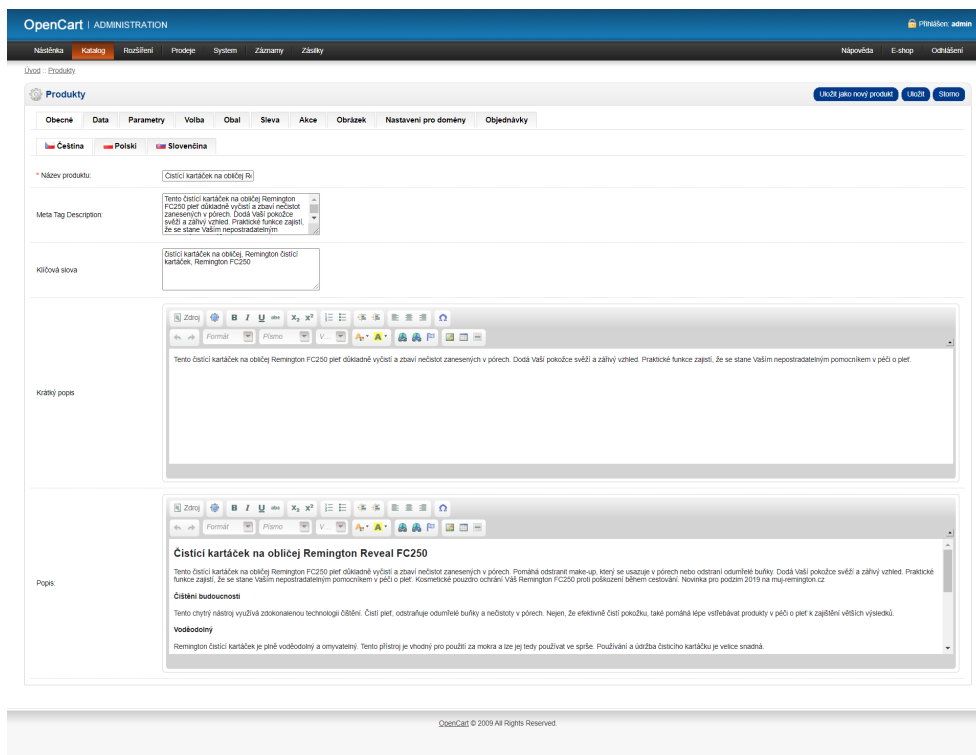


Figure 1.5: Product detail - General tab

Figure 1.5 shows the **General** tab of the product detail. There is general information about the product. There are fields such as the product name, various descriptions, and keywords. The main requirement from the users is to have the language setting in one place.

Data tab

Figure 1.6 shows the **Data** tab of the product details. The most important information about the product is shown here.

There are some important fields:

- **Model** - a product model
- **SKU** - a barcode
- **SEO** - keywords for the SEO
- **Location** - the location in the warehouse
- **Image** - the main image of the product
- **Producer** - a producer of the product

- **Availability date** - date, from when the product will be displayed in the e-shop
- **External warehouse quantity** - how many pieces there are in the external warehouse
- **Internal warehouse quantity** - how many pieces there are in the internal warehouse
- **Price** - there are three price fields: price without VAT, price with VAT, and purchase price
- **Free delivery** - when the profit of the product is greater than 300 Kč, delivery is offered for free
- **Delivery by Zásilkovna** - some of the products could not be sent by Zásilkovna because of their dimensions
- **Main category** - the main category, where the product will be placed within the e-shop. All categories are specific for each e-shop
- **Secondary category** - additional categories, which relate to the product
- **Related products** - all products that are related to the product and that the user on the e-shop web should see as offers. There is also a checkbox here used when the related product should map the product in the same way as a related product
- **Color variants** - mapping to the same products in different colors
- **Heureka fields** - overwriting some properties for the Heureka service

Parameters tab

The **Parameters** tab contains additional specific properties, which are related to the category of the product. So, for product parameters loading to be loaded properly, its main category must be selected.

Choices tab

This tab contains settings for additional properties. For example, available sizes.

Packaging tab

Current users of the application stated, that they usually don't use this tab, so this tab can be hidden.

1. ANALYSIS

The screenshot displays the 'Data' tab of the OpenCart Administration interface for a product. The top navigation bar includes 'Nástěnka', 'Katalog', 'Rozšíření', 'Prodejce', 'System', 'Základní', and 'Zásady'. The user is logged in as 'Příhlášen: admin'. The page title is 'Produkt' and the breadcrumb trail is 'Úvod > Produkty'. The product details are as follows:

- Model:** FC250
- SKU:** 5030061101249
- Lokace:** (empty)
- SEO klíčové slovo:** (880) kartáček na obličej remington fc250
- Obrázek:** (Image of the Remington FC250 facial brush)
- Výrobce:** Remington
- Výžatek doručení:** Ano (selected)
- Datum dostupnosti:** 2019-10-04
- Množství:** 0
- entry_internal_quantity:** 1
- Stav, když není skladem:** Do 14 dnů
- Stav:** Aktivní
- Daňová třída:** Zboží s DPH 21%
- Záruka (měsíců, -1 = neomezená):** 24
- Cena:** 660.3306
- entry_price_vat:** 799.0000599999999
- entry_purchase_price:** 430.0000
- Doprava zdarma:** (checkbox)
- Poslat Zákazkovou:** (checkbox)
- Rozměry (D x Š x V):** 0.00 | 0.00 | 0.00
- Rozměr:** Centimetry
- Váha:** 0.00
- Váhová třída:** Kilograms
- Hlavní kategorie:** [www.muj-remington.cz] Dámy > Péče o pleť a tělo > Čistič kartáčky a kartáče
- Styka kategorie:** [www.styka.cz www.styka.sk www.stykatop.pl] Kůže > Péče o pleť > Čistič kartáčky

The 'Kategorie' section lists various categories with checkboxes, including 'Kůže', 'Kůže - kosmetická zrcátka', 'Masážní a pedikury', 'Obličejové sauny', 'Proti celulitidě', 'Trvalá epilace', 'Péče o tělo', 'Masáž', 'Perflová koupel', 'Masáž v Kósa', 'Masáž těla', and 'Masáž nohou'. The 'Související produkty' section shows related products like 'MAN1000, Set na manikuru a pedikuru Reveal MAN1000'. The 'Barvé varianty' section shows color variants for the product. The 'Heureka' section includes 'Heureka productionname', 'Heureka product', 'Heureka categorytext', and 'Barva' (Dílá, Ilyková, neuvodena).

Figure 1.6: Product detail - Data tab

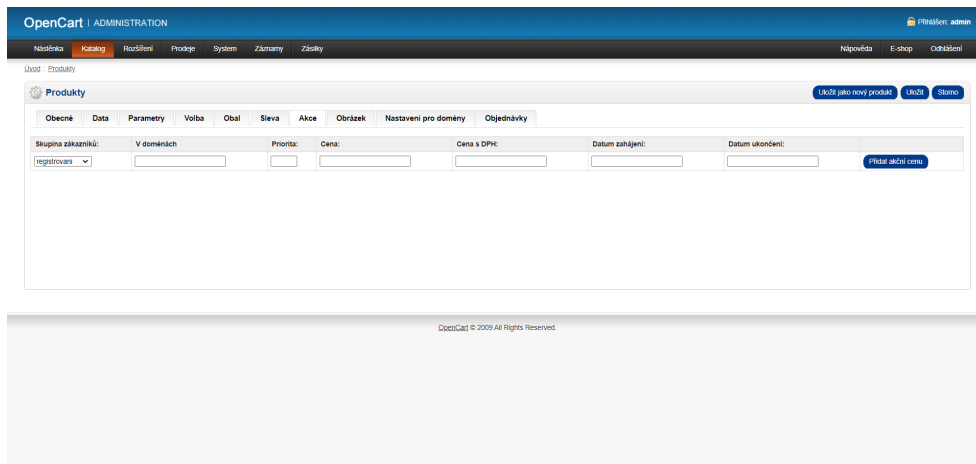


Figure 1.7: Product detail - Special offer tab

Discount tab

This tab also is not used by the users all the sales are entered using the Special offers tab. Thus this tab can also be deleted.

Special offers tab

Figure 1.7 shows the **Special offers** tab of the product detail. There, users could define special offers for the current product.

Images tab

Figure 1.8 shows the **Images** tab of the product detail, where are all images of the product. As we can see in Figure 1.6, there is already one main image of the product was uploaded. For the user's effective work with the application, it could be done in one single place within the product detail page. Ideally, it should be implemented using modern uploading components, which allow a user to drag and drop the images into the browser.

Domain settings tab

Figure 1.9 shows the **Domain settings** tab of the product detail. There are settings for texts for each domain and language. It is used for properties overwriting if a user wants to display different data for different domains. Users could change such properties as name, keywords, description, price, or even SEO properties (search engine optimization). Current users of the administration application complain about the way the properties are edited. There is the requirement to analyze, how all this data could be entered and edited for each domain and language in one place.

1. ANALYSIS

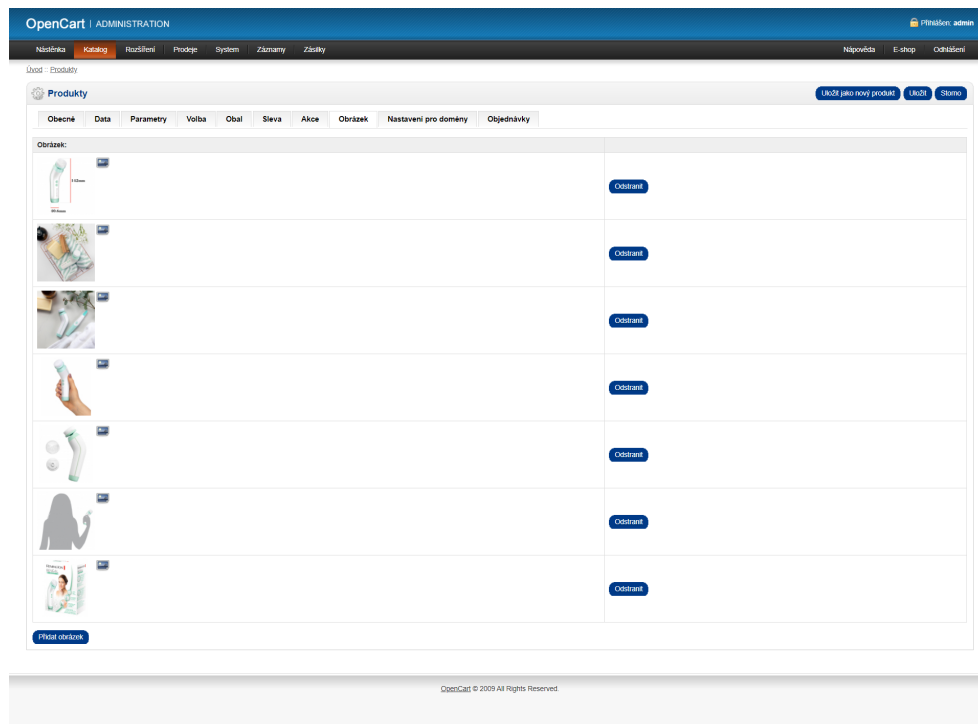


Figure 1.8: Product detail - Images tab

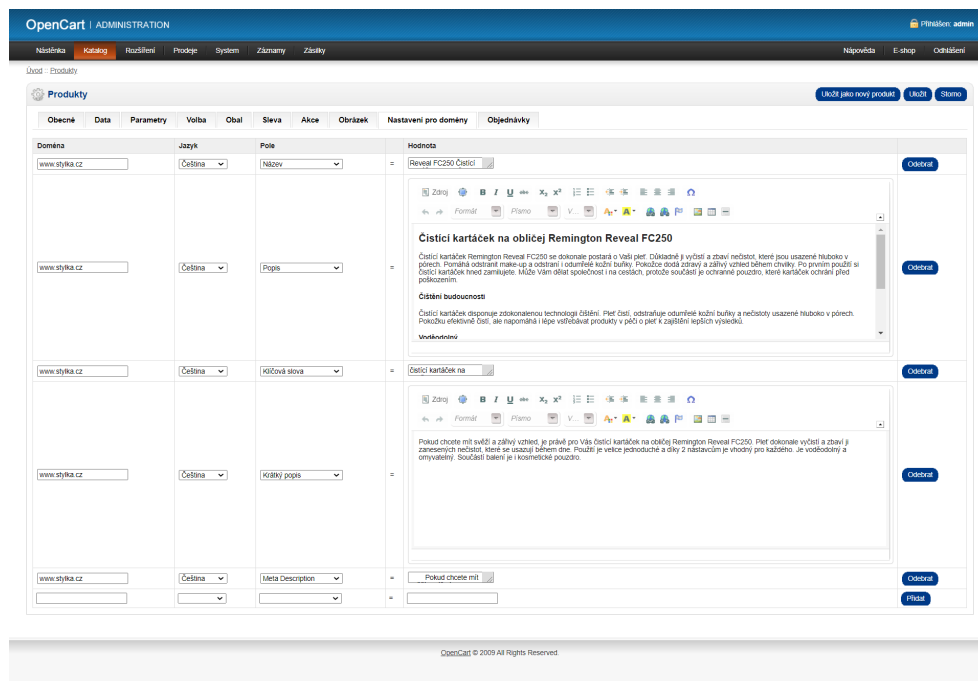


Figure 1.9: Product detail - Domain settings tab

Orders tab

On the **Orders** tab there is a table with all orders, where the product is contained. There is basically nothing to change here.

Summarize features that could be implemented in the further version:

- Language settings for each product could be in one place
- Images uploading could be in one place, using a drag-and-drop component for this
- Hide or delete **Packaging** and **Discount** tabs, which are not used by employees
- Entering and editing data for each domain and language in the one place, not to have the particular tab for properties overwriting
- A copy capability from the Meta Tag Description to a short description of the product
- There is the component that is used for the formatted text editing. But the component has many settings, such as, for example, fonts, which mostly are not used, so in the further version simpler component could be used
- Changes in the **Data** tab:
 - **Location** - add the link to the warehouse with the pre-filled product id
 - **Image** - all images should be managed in one place. Working with images should be consistent
 - **Requires delivery** - not used, it could be hidden in the default layout
 - **Availability date** - not used, it could be hidden in the default layout
 - **Internal warehouse quantity** - add the link to the status page with the pre-filled product id
 - **Free delivery** - this option should be selected automatically, based on the profit value
 - **Sizes** - size and weight properties are not used so often, it could be hidden in the default layout
 - **Categories** - A well-arranged way to display categories in the application

Related products - use the drag and drop component for selecting related products. Display images of products to increase efficiency. Implement a full-text search, search within the same category the product belongs

Attachments - not used, it could be hidden in the default layout

Downloading - not used, it could be hidden in the default layout

Color variants - same as for the **Related products**, using of some more user-friendly component

Heureka fields - making them consistent with the rest of the product information

1.3 Prototyping tools

Let's start with the terms. Wireframes, mockups, and prototypes are terms that are often used interchangeably, and they are actually quite confusing. Wireframes, mockups, and prototypes represent the different steps of a design flow. [5]

A **wireframe** is a low-fidelity way of showing a design. It is the graphical representation of an application or a website containing the most essential elements and the content.

A few characteristic features of a wireframe are the following:

- It shows the main chunks of content
- It draws the outline and the layout structure
- It depicts the most basic UI

A **mockup** is a visual way of representing a product. While a wireframe mostly represents a product's structure, a mockup shows how the product is going to look like. Unlike a wireframe, a mockup looks more like a finished product or prototype, but it is not interactive and not clickable.

A **prototype** is already very close to the finished product. Unlike the previous two, a prototype is clickable and thus allows the user to experience content and interactions in the interface. In fact, a prototype is very much like the final product itself. [6]

1.3.1 Analysis of the wireframing tools

In this section I will analyze existing web wireframing tools and choose the most suitable one for the mockup creation.

1.3. Prototyping tools

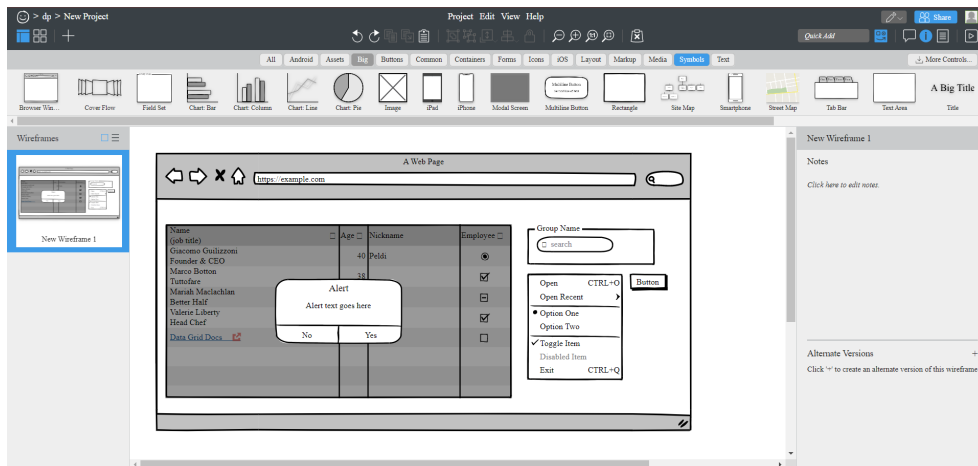


Figure 1.10: Balsamiq user interface

1.3.1.1 Balsamiq

Balsamiq Wireframes is a rapid low-fidelity UI wireframing tool that reproduces the experience of sketching on a notepad or whiteboard, but using a computer. Balsamiq has a free trial and premium version. Available on: <https://balsamiq.com/wireframes/>.

Pros

- There are all the components, which will be used in the design
- Free export to PNG or PDF without the watermarks
- In the free version there are multiple-page wireframes
- The ability to define actions

Cons

- Menu interface for adding design elements can be confusing to a newcomer
- The icon library in Balsamiq is quite limited
- Adding data to a grid with many columns can also be somewhat frustrating

On the 1.10 is shown the user interface of the Balsamiq web application.

1. ANALYSIS

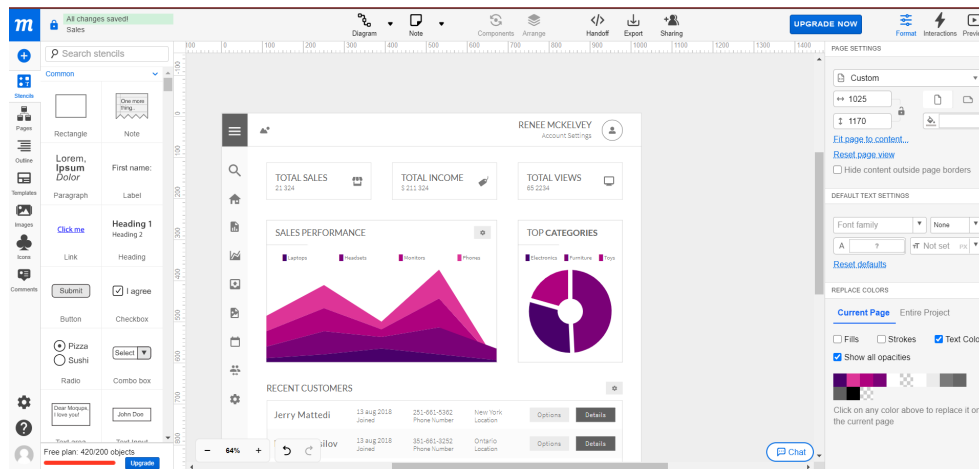


Figure 1.11: Moqups user interface

1.3.1.2 Moqups

A streamlined web app that helps you create and collaborate in real-time on wireframes, mockups, diagrams and prototypes.

Pros

- An opportunity of doing the sketch or the styled prototype
- Many various components you can use
- An ability to define actions

Cons

- There is no export feature in the free plan
- Limited number of the components you can use in the free version.

In Figure 1.11 is shown the user interface of the Moqups web application.

1.3.1.3 Wireframe.cc

A minimal wireframing tool, which has a free trial and premium version. Good for sketches, but isn't suitable for more complex designs with styles and components. Available on: <https://wireframe.cc/>.

Pros

- Very intuitive design

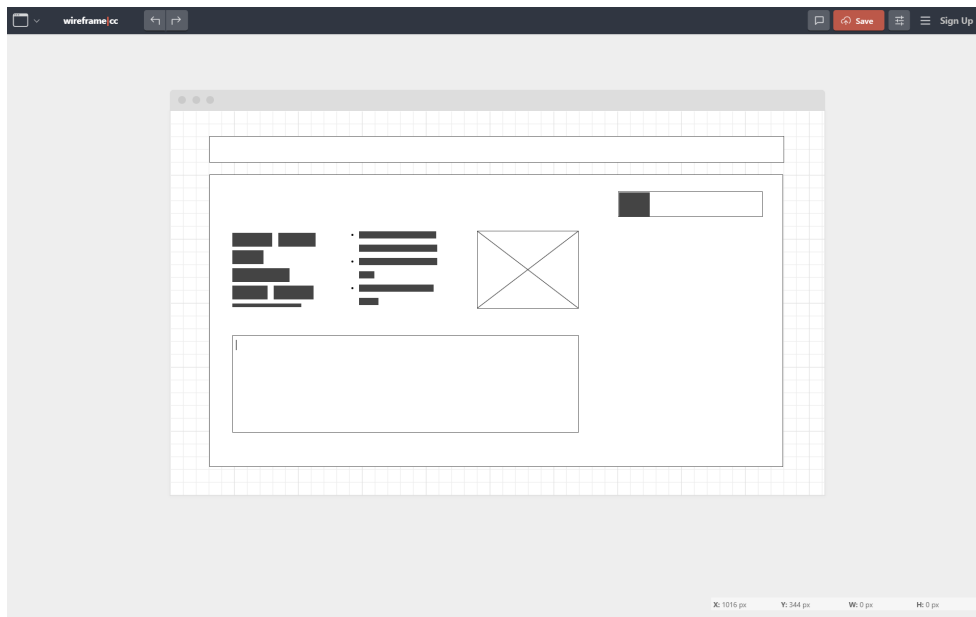


Figure 1.12: Wireframe.cc user interface

- Good for fast sketching

Cons

- Very minimal, there are no components, which will be used in the design
- In free version only single-page wireframes
- There is no opportunity to define actions

On the 1.12 is shown the user interface of the Wireframe.cc web application.

1.3.1.4 Mockitt

Mockitt is the web tool for creating wireframes/interactive prototypes with built-in widgets and templates, by simply drag-and-drop. [7] Mockitt also has a free trial and premium version. Available on: <https://mockittapp.wondershare.com/>.

Pros

- Intuitive design
- The free version has multiple-page wireframes
- An ability to define actions

1. ANALYSIS

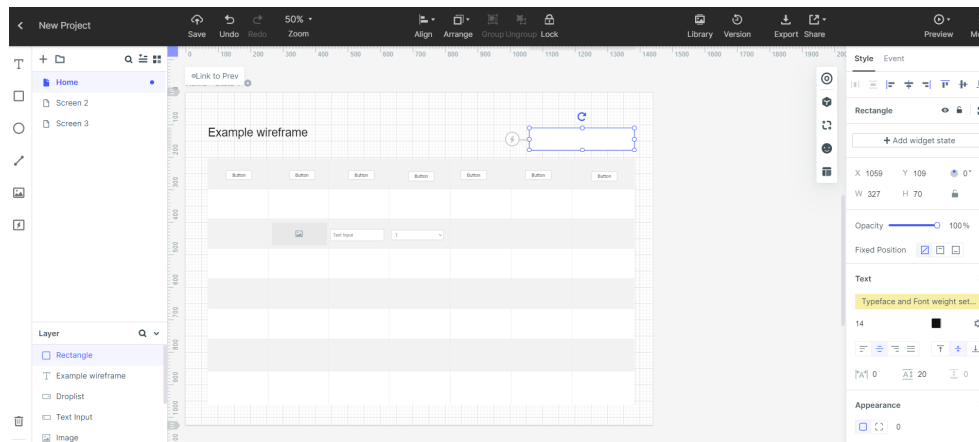


Figure 1.13: Mockitt user interface

Cons

- There are watermarks in the exported prototype in the free version

In Figure 1.13 is shown the user interface of the Mockitt web application.

1.3.1.5 Conclusion

After analyzing the wireframing tools I chose the **Balsamiq**, because of its easy-to-use interface, the ability to define interactions, and almost unlimited features in the free plan.

Design

This chapter describes the process of prototype creation and what steps to create the final version of the new application wireframes were done. Then I will present technologies, that I decided to use for implementation. Also in this chapter, I explain how communication with a server works, what technologies are used, and what is happening under the hood. Next, general information about application security follows. In this section, I am describing how CORS policies work, what web attacks exist, and how authentication and authorization works.

2.1 Prototyping

This section presents the process of prototyping the UI design of the front-end administration application. Created mockups consider user requirements and take into account user experience (next UX) design principles.

Why is prototyping required? The primary goal of building a prototype is to test designs (and product ideas) before creating real products. Product's success is directly related to whether it is tested or not.

The following are two cases that require a prototype:

- **Ensure the design concept works as expected.** In most cases, it is relatively easy to test a concept with real users. Once an interactive version of a product design is in the hands of real users, a product team will be able to see how a target audience wants to use the product. Based on this feedback, it is possible to adjust an initial concept.
- **Determine if people are able to use a product.** Prototyping is essential for finding and resolving usability issues before launch. Testing exposes areas that need improvement. That is why so many product teams create prototypes, have users test them, and iterate the design until it is good enough. [8]

The process of prototype creating was iterative and consisted of key design steps:

- Needs assessment
- Prototyping
- Evaluation
- Re-design

2.1.1 Wireframes

The first wireframes were created were paper sketches. In Figures 2.1 and 2.2 you can see the result of the first phase of prototyping. Here are visible basic concepts, which I wanted to realize in the final application prototype.

One of them, for example, is using colored label components for faster user orientation in the orders page. Another difference from the current version of the application you can see in Figure 2.1 is the absence of a “Filter” button in the filter area of the table. In the current version of the application a user must click on the “Filter” button to apply filters they have selected and it could reduce the efficiency of interaction with the system. Especially, if the user expects filtering on the fly like it is done in most modern web applications. So, in the new application, the filtering on the orders page will be done on the fly.

In Figure 2.2 you can see the first sketch version of the Order detail page. In this version, I decided to organize order data using tabs. But after evaluation and consultations with my supervisor and e-shop employees, it was figured out, that the data I wanted to separate using the tabs are logically connected and a user usually wants to see them together. For instance, when the user updates the order, they probably want to see previous historical events of the order. So, the concept of tabs was rejected.

2.1.2 The lo-fi prototype

Low-fidelity (lo-fi) prototyping is a quick and easy way to translate high-level design concepts into tangible and testable artifacts. The first and most important role of lo-fi prototypes is to check and test functionality rather than the visual appearance of the product.

Clickable wireframes are the simplest form of an interactive prototype — created by linking static wireframes together. [8]

The following sections introduce a lo-fi prototype of the order pages. The entire clickable prototype in PDF format you can find at the end of the thesis in chapter C and also on the enclosed media.

Generally, it was decided to use colored label tags, where it is possible. This decision is justified by the fact that the user could build the associations

2.1. Prototyping

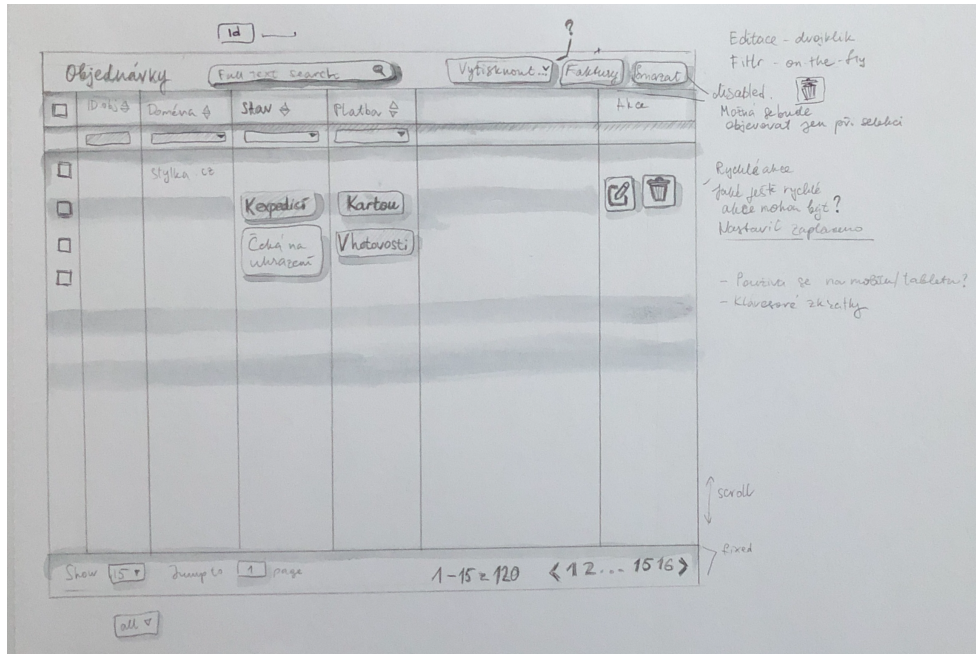


Figure 2.1: Sketch of the orders page

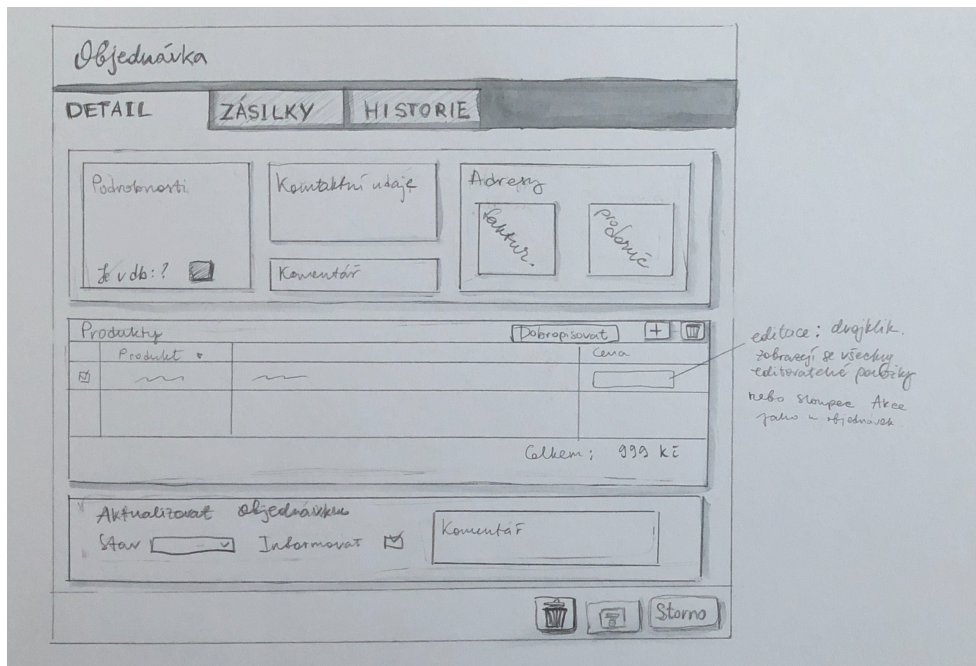


Figure 2.2: Sketch of the order detail page

2. DESIGN

ID	prohlázeno	ID objednávky	Kód faktury	Doména	Meno zákazník	Komentár	Stav	Doprava	MAG	Datum vyhotovení	Celkem	Zálohová platba	Uhraneno	Základ	DPH	Měsíční	Přítel z	Souhlas	Akce
		1111	25678	stbba.cz	Jul. Juli		OK	Kespeďo	OK	02.02.2020	1000 Kč	Kartou	OK	1000 Kč	OK		https://m.beurba.cz	OK	
		1112	5678	stbba.cz	Jul. Juli		OK	Čeká na uhradení	Zobrazit	02.02.2020	100 Kč	V hotovost	OK	100 Kč	OK		https://m.beurba.cz	OK	
		1113		stbba.cz	Jul. Juli		OK	Ve zpracování	OK	02.02.2020	1500 Kč	Pravidelně	OK	1500 Kč	OK		https://m.beurba.cz	OK	

Figure 2.3: Lo-fi prototype of the orders page

between type values and colors, and it could maximize the efficiency of interaction with the administration app. In this section presented sketches, how the coloring could look like, the final color palette will be ready at the realization phase. The maximum number of colors that can be used for color-coding is something between 7 - 9. For maximum efficiency and learnability, only 3-5 colors should be used. It should also be taken into consideration that approximately 4% of the population is color blind, so additionally icons should be used.

Also, a lot of icons are used to replace text-labeled actions in the current application. When icons are used correctly, they reflect the core idea and intent of a product or action, and they bring a lot of nice benefits to user interfaces, such as enhancing the aesthetic appearance. Chosen icons are well-known and universal for easier recognition. Every icon in the prototype has a tooltip with the icon explanation or with the action name, in case a user is uncertain what the icon does.

2.1.2.1 Orders page

In Figure 2.3 is Orders page prototype. There are new columns, which were described in 1.2.1.1 section.

New columns

In the prototype, you can see the newly added columns.

- **Viewed** - if an order was viewed by an e-shop employee. Here an **Eye** icon was chosen and it is crossed out if the order have not been seen by any e-shop employee yet.
- **Comment** - if there is a comment from a customer. In the prototype, this flag is emphasized by using colors to indicate if there is a comment or not.
- **Shipping label** - if a shipping label has been already created. Same as **Comment**, using red and green colors to represent if the label was printed.
- **Payment method** - a payment method of an order. Colored tags are added for each payment method.
- **Slovakia** - if an order came from the Slovak Republic. Also, **true/false** column emphasized by using colors.
- **In stock** - if all products in the order are in a warehouse. Another **true/false** column emphasized by using red and green colors.

Search bar

In the 2.3 prototype in the upper panel of the table, a search bar is present. The search bar allows full-text search within orders.

A “Was paid” setting dialog

The next feature that was added in the prototype is setting the “Was paid” flag for an order. After a user clicks on **Banknote** icon a modal dialog appears (see Figure 2.4). In this dialog, the user can set a payment date and if the system should send an informative e-mail to a customer.

Hide columns

During the analysis of the Orders page, the requirement to hide columns if a current user does not use them arose. This functionality is presented in Figure 2.5. The **Status** column is already hidden, so this column absents in the table behind. The application remembers the hidden columns, so the user is not forced to hide columns every time they open the application.

Profit column

During the analysis in the 1.2.1.1 section, the requirement to mark profit value with the different colors arose. As you can see in Figure 2.3, the **Profit** column also use colored labeled tags. We have agreed with the supervisor that it makes sense to have three categories of profit:

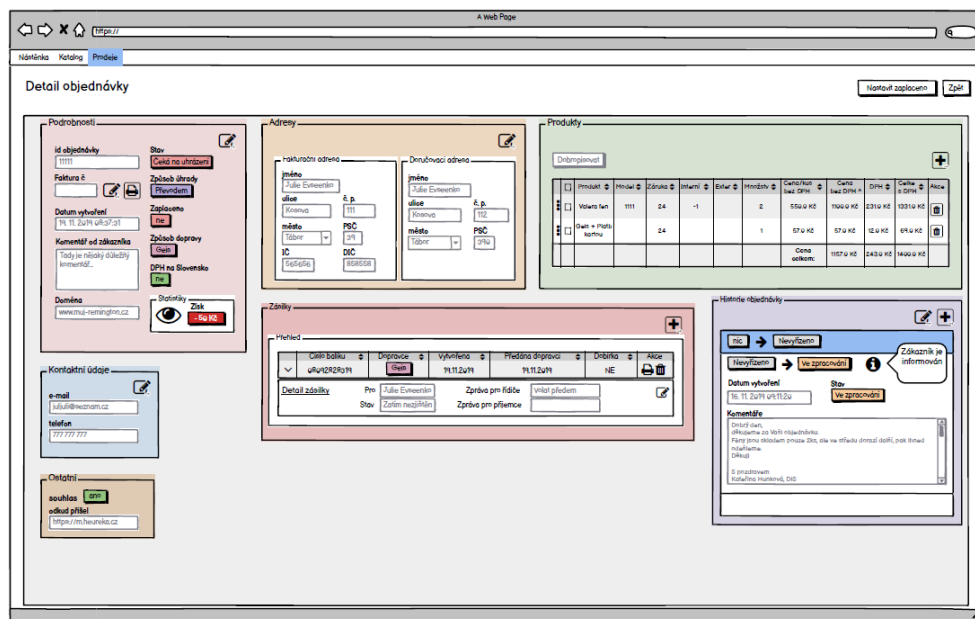


Figure 2.6: The lo-fi prototype of the order detail page

- Under 0 Kč
- Between 0 Kč and 100 Kč
- More than 100 Kč

2.1.2.2 Order detail page

In Figure 2.6, Order detail page prototype is presented. The prototype is based on the analysis was done in the 1.2.1.2 section.

The whole layout is constructed using draggable panels. Each panel is resizable, draggable, and contains a logically connected piece of information. The application remembers the user's layout, so the user does not need to set their own layout every time they open the application.

Detail editing

In Figure 2.7, you can see the Order detail page in edit mode. For simplicity, all the cards are in edit mode on one page. But in the created application, the user will edit each panel independently from the others. As you can realize, when edit mode is activated, some of the labeled tags become select boxes, where is a limited predefined number of options. For instance, the shipping method tag becomes a select box, where the user can select one of the options described in the 1.2.1.2 section.

2. DESIGN

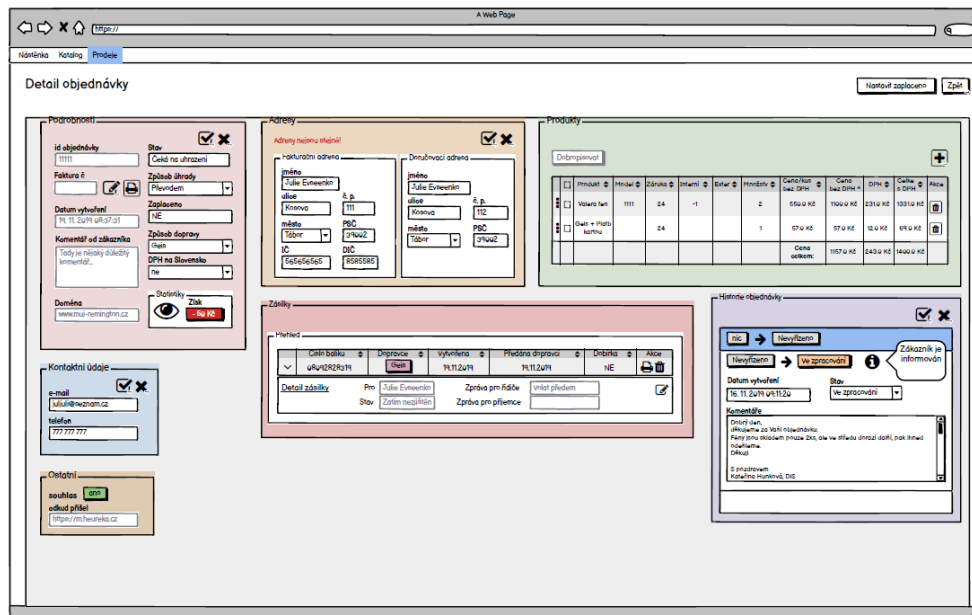


Figure 2.7: Order detail editing

Some panels are not editable in the same way as others. There are **Products** and **Packages** panel, that are edited in a different way. **Products** table has inline editing of products. **Packages** has a package detail component with the built-in edit mode.

During the analysis, the requirement to warn about the different payment and shipping addresses arose. The created prototype fulfills this requirement.

Products

In Figure 2.8, the **Products** card is displayed. The **Products** panel looks like the **Products** panel in the current application. In the prototype, icons for action indication are used. Rows in the products table are draggable so a user can change the order of rows using the drag-and-drop feature.

After clicking on the **Plus** icon the “Add a new product” modal dialog appears. This dialog is presented in Figures 2.9 and 2.10. The dialog has two tabs: **Product** and **Shipping and payment methods**.

In the **Product** tab, a user can add a new product to the order. The new application will be connected with the warehouse system, so data in this dialog will not be filled manually by the user. There are select boxes for product **Name** and **Model** fields. After the user selects the product name and model, the **Price/Piece** field will be filled by the application based on the data from the back-end application. The user should fill the **Quantity** field for total price calculation.

Produkty

Dobropisovat +

<input type="checkbox"/>	Produkt	Model	Záruka	Interní	Exter	Poč	Cena/kus bez DPH	Cena bez DPH	DPH	Celkem s DPH	Akce
<input type="checkbox"/>	Valera fen	1111	24	-1		2	550.0 Kč	1100.0 Kč	231.0 Kč	1331.0 Kč	
<input type="checkbox"/>	Gels + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
<input type="checkbox"/>	Fen Obvykly	123	24	-1		1	500.0 Kč	500.0 Kč	105.0 Kč	605.0 Kč	
							Cena celkem:	1657.0 Kč	348.0 Kč	2005.0 Kč	

Figure 2.8: The products panel in the order detail

Nový produkt

Produkt Doprava a způsob platby

Vyberte prosím produkt Cena za kus

Cena bez DPH

Vyberte prosím model * DPH

Množství Celkem s DPH

* Nejdřív si zvolte produkt

Stav Zpráva pro příjemce

Figure 2.9: New product

In the **Shipping and payment methods** tab, a user could add a new row with shipping and payment methods to calculate the final price of an order. This tab also gets data from the back-end application, so after selecting a method all prices recalculate based on prices were sent from the back-end.

The table supports inline editing.

Packages

The packages panel is shown in Figure 2.11.

Between the requirements on the packages section were:

2. DESIGN

Nový produkt

Produkt Doprava a způsob platby

Vyberte prosím dopravce GEIS
GEIS
Česka Pošta
Zasilkovna

Vyberte prosím způsob platby Kartou
Převodem
Na dobírku

Cena za kus 100.0 Kč
Cena bez DPH 100.0 Kč
DPH 21.0 Kč
Celkem s DPH 121.0 Kč

* Nejdřív si zvolte produkt

Figure 2.10: New shipping or payment method

Zásilky

Přehled

Císlo balíku	Dopravce	Vytvořena	Předána dopravci	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Detail zásilky

Pro Julie Evseenko Zpráva pro řidiče Volat předem

Stav Zatím nezjištěn Zpráva pro příjemce

Figure 2.11: Packages panel in the order detail

- Well visible button to remove a package
- Well visible button to print a shipping label

Both of these two buttons are represented as icons in the prototype. The **Printer** icon represents the “Print the shipping label” action. After a user clicks on the **Delete** icon, the system displays a modal dialog for confirmation if they really want to delete a package.

In the left part of the row, there is a **Detail** icon, which helps a user to show or hide details of the package. As we discussed in the 1.2.1.2 analysis section, not all the package properties user wants to see in the package panel and it is nice to have an opportunity to hide some unused fields. In the table is general important information, such as package number, carrier, when the package was created, when it was handed over to the carrier, and the “Cash

The screenshot shows a web application window titled 'Zásilky'. It contains two main sections:

Přehled (Overview): A table with columns: Císlo balíku, Dopravce, Vytvořena, Předána dopravci, Dobírka, and Akce. The first row contains: >, 08092828319, Geis, 19.11.2019, 19.11.2019, NE, and icons for print and delete.

Vytvoření zásilky (Create shipment): A form with the following fields:

- Zvolte dopravce: A dropdown menu with 'Geis' selected.
- Počet balíků: An empty text input field.
- Poznámka pro řidiče: An empty text input field.
- Dobírka: An empty text input field.
- Poznámka pro příjemce: An empty text input field.

 At the bottom right of the form are a checkmark icon and an 'X' icon.

Figure 2.12: New package

on delivery” price if the order has a self-titled payment method. The detail of a package contains carrier-specific data were described in the 1.2.1.2 section. The package detail is editable.

After a user clicks on the **Plus** icon the “Create a new package” modal dialog will appear, see Figure 2.12.

History

There is new design for the **History** panel in Figure 2.13. Two sections from the current version of the application were combined: **History** and **Update order** section because they are tightly connected. The expansion panel, also known as an accordion, is suitable for this purpose. The accordion provides an expandable details-summary view. The card header shows a change of an order status within the historical event and acts as the control for expanding and collapsing. The **Info** icon indicates if a customer was informed about an order update. There is a tooltip for this icon, so a user is always aware of the meaning of each icon.

After clicking on the **Plus** icon the **Update an order** panel within the **History** card appears, see Figure 2.12. Fields within this panel have been already described in the 1.2.1.2 section. After submitting the changes, a new historical event panel shows up.

Invoice

There was the requirement to choose the invoice number in a time period. In Figure 2.6, you can see the **Edit** icon button. After a user clicks on this button the “Select invoice number” modal dialog, which is shown in Figure 2.15, appears. In this modal dialog, a user can choose the time period for

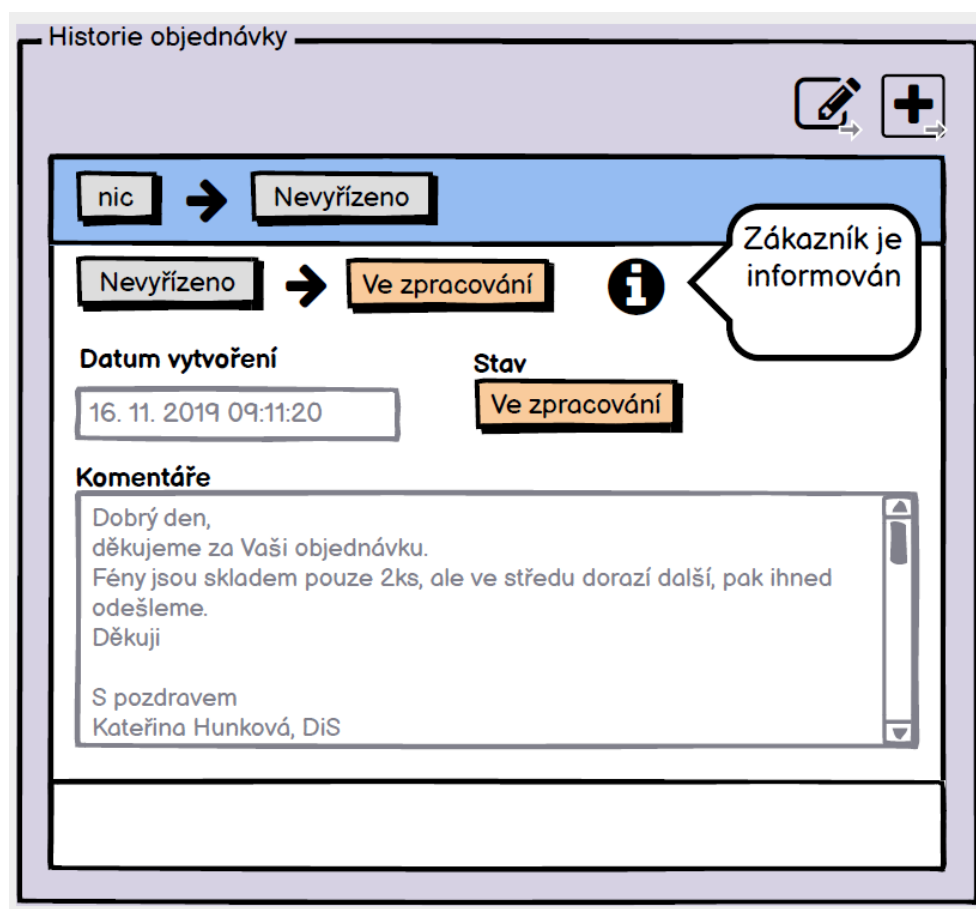


Figure 2.13: History panel in the order detail

unused invoice numbers searching and write them out. The other functionality related to invoices is resetting the invoice number. Reset means, that even though an order had an assigned invoice number the user can reset it, in other words, the invoice number becomes free to use in other orders.

2.2 Technologies

This section describes technologies that were used for the new administration application implementation.

TypeScript is an open-source language which builds on JavaScript, by adding static type definitions. TypeScript checks a program for errors before execution, and does so based on the kinds of values, it is a static type checker. One of the big benefits is to enable IDEs to provide a richer environment for spotting common errors in the code.

Historie objednávky

nic → Nevyřízeno

Nevyřízeno → Ve zpracování ⓘ

Aktualizovat objednávku

Stav
Ve zpracování ▾ Informovat zákazníka

Komentáře

Dobrý den,
děkujeme za Vaši objednávku.
Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak
ihned odešleme.
Děkuji

S pozdravem
Kateřina Hunková, DiS
www.muj-remington.cz

✕

Figure 2.14: Update an order

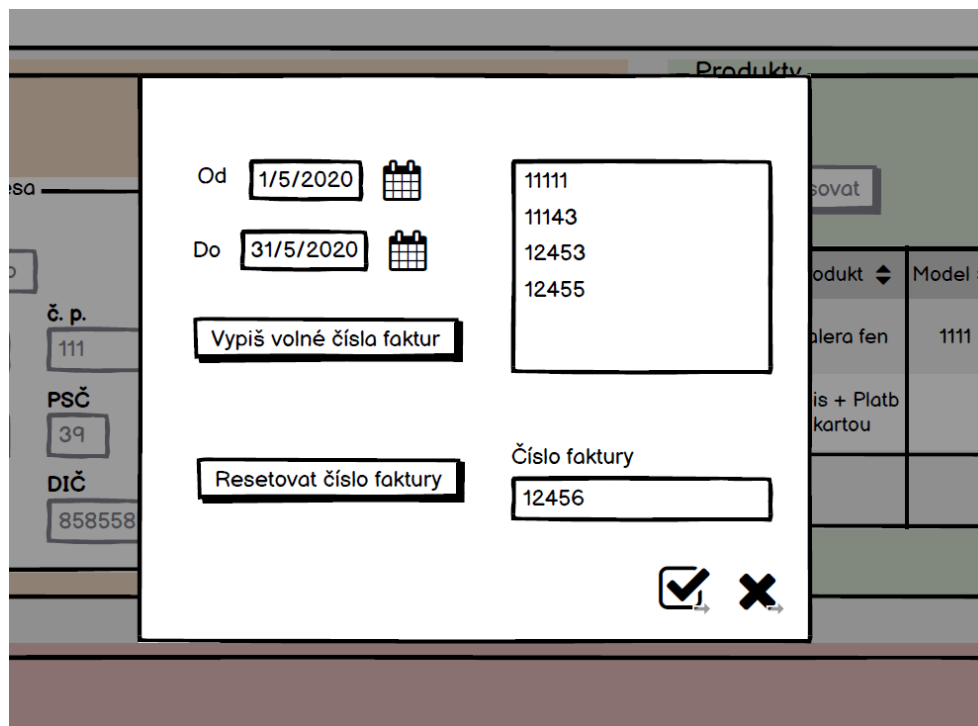


Figure 2.15: The “Select invoice number” dialog

Types provide a way to describe the shape of an object, providing better documentation, and allowing TypeScript to validate that the code is working correctly.

TypeScript allows for the use of many of the latest ECMAScript features and translates them to older ECMAScript targets of your choosing. This means that new features, like modules, lambda functions, classes, the spread operator and destructuring could be safely used, while remaining backwards compatible with older browsers and JavaScript runtimes. [9]

npm is the world’s largest software registry and it is the package manager for Node.js. It was created to help JavaScript developers easily share packaged modules of code. [10]

React is a declarative, component-based JavaScript library for building user interfaces.

React allows to build encapsulated components that manage their own state, then compose them to make complex user interfaces. [11]

ESLint is a static code analysis tool for identifying problematic patterns found in JavaScript code. The main ESLint goal is making code more consistent and avoiding bugs. [12]

2.2.1 Potential problems

This section describes the potential problems that may arise in the final version of the E-shop administration application. This thesis does not solve these problems, but only notes their presence and suggests potential solutions.

In the production environment of the current application exists an enormous amount of orders, over 30,000 orders are currently stored in the production database. Hence, the final version of the application should use server-side pagination, filtering, and search. The current prototype handles these functionalities but in the next versions, mentioned functions should be handled by the back-end application.

2.3 Communication with the server

A client is whatever you are using to interact with the internet. It is, for example, a web browser. The web browser on your computer is one client, the web browser on your phone is another client. A server is a computer that responds to requests by serving responses. A client and a server communicate via HTTP protocol.

The Hypertext Transfer Protocol (HTTP) is an application-level protocol. Basically, HTTP is a TCP/IP based communication protocol, that is used to deliver data. It provides a standardized way for computers to communicate with each other. HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests. [13]

The communication is based on the AJAX (Asynchronous JavaScript and XML) model. With AJAX, web applications can send and retrieve data from a server asynchronously (in the background), which allows changing content dynamically without the need to reload the entire page. Although X in AJAX stands for XML, JSON is used more than XML nowadays because of its many advantages such as being lighter and a part of JavaScript. [14]

For the communication between the front-end and back-end applications, Axios library is used. Axios is a simple promise-based HTTP client for the browser and node.js. [15] The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value. On the 2.1 is the example, how to create and send a GET request from the client to the server.

```
1 import axios from "axios";
2
3 axios.get('/users')
4   .then(res => {
5     console.log(res.data);
6   })
```

7)

Listing 2.1: Axios GET request example

2.4 Security

The first section of this chapter describes what CORS mechanism is and how it is connected with a client-server architecture. After that, we will have a look at the popular web attacks and the ways how to prevent them. In the last section, we describe what authorization and authentication are, and then details about how it is implemented in the e-shop administration application are given.

2.4.1 CORS

Let's with a small theory. The front-end part of the e-shop administration communicates with the back-end application via HTTP protocol. Front-end sends HTTP requests, back-end sends HTTP responses back. But there is a problem with the fact, that the back-end runs and listening on the other domain and port than the front-end, which sends the requests on the API. For example, while a local development back-end server runs on the `http://127.0.0.1:8000` address and the front-end application runs on the `http://127.0.0.1:3000` address, these are the different origins, because of the port. At this moment we should set CORS policies, that will allow our back-end and front-end applications to communicate with each other.

Cross-Origin Resource Sharing (CORS) is an HTTP-header based mechanism that allows a server to indicate any other origins (domain, scheme, or port) than its own from which a browser should permit loading of resources. [16]

The browser's "preflight" requests mechanism is also related to CORS. In the preflight request, the browser finds out if the application could request the server. The preflight request contains information about who wants to request the server, what HTTP method it wants to use, and other information. The server in its turn should send information about who could access the resources, which HTTP methods and headers are allowed, and so on. When the browser makes sure, that server allows to do the required request, it sends the real request with the properties from the preflight request.

Figure 2.16 shows how the communication between the client and the server is implemented.

2.4.2 Web attacks

Cross-Site Request Forgery (CSRF) is a type of attack that occurs when a malicious website, email, blog, instant message, or program causes a user's

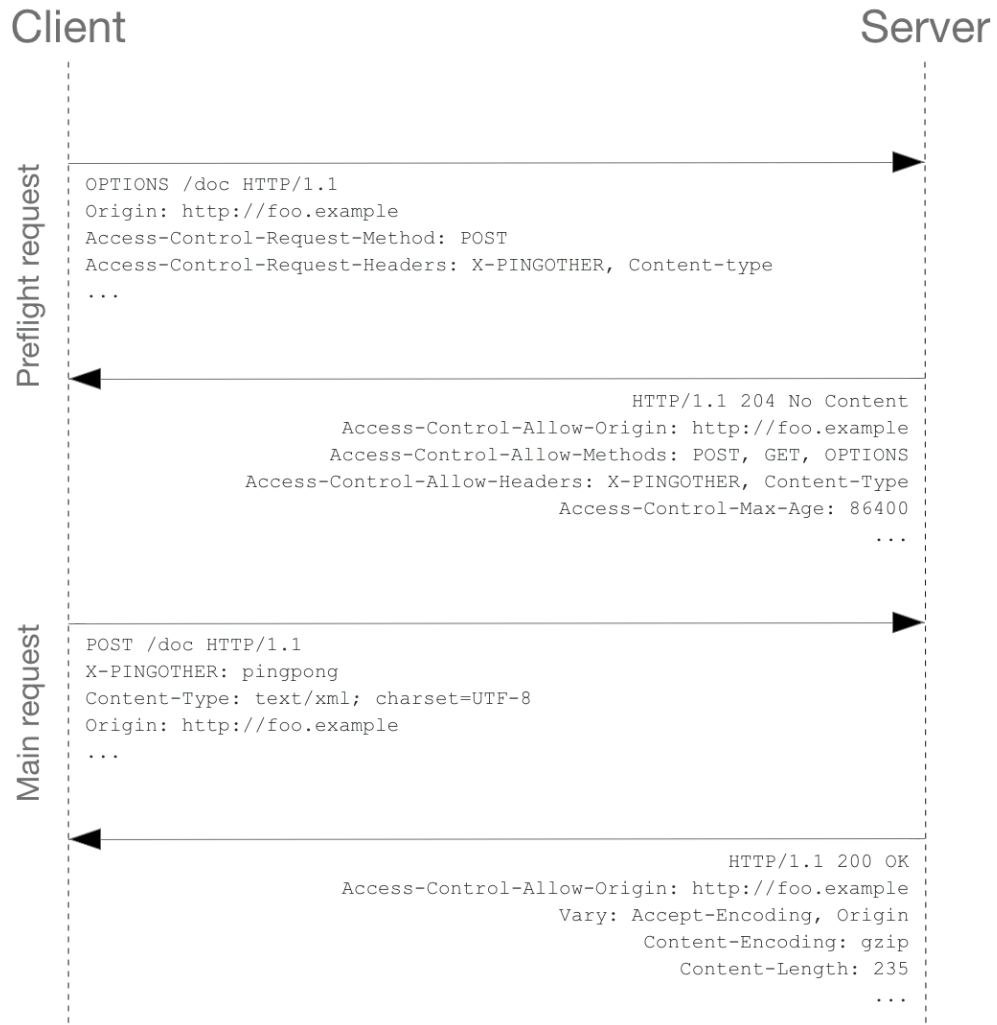


Figure 2.16: Preflight request mechanism [17]

web browser to perform an unwanted action on a trusted site when the user is authenticated. A CSRF attack works because browser requests automatically include all cookies including session cookies. Therefore, if the user is authenticated to the site, the site cannot distinguish between legitimate requests and forged requests. [18]

Here are described only one prevention method against the CSRF attack called **Double Submit Cookie**. There are other methods such as **Synchronizer Tokens** or **Encrypted Token**. We use **Double Submit Cookie** in the e-shop administration application because it is easy to implement and it is stateless.

In this technique, we send a random value in both a cookie and as a request parameter, with the server verifying if the cookie value and request value match. If both of them match on the server's side, the server accepts it as a legitimate request and if they don't, it would reject the request. [18]

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser's side script, to a different end-user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it. [19]

There are methods to prevent XSS attack:

Escaping means taking the data an application has received and ensuring it's secure before rendering it for the end user.

Validating Input is the process of ensuring an application is rendering the correct data and preventing malicious data from doing harm to the site, database, and users.

Sanitizing is a cybersecurity measure of checking, cleaning, and filtering data inputs from users, APIs, and web services of any unwanted characters and strings to prevent the injection of harmful codes into the system. [20]

2.4.3 Authentication and authorization

Let's define terms. The **authentication** is a process of validating that users are who they claim to be. The **authorization** is the process of giving the user permission to access a specific resource or function. [21]

Laravel Fortify authentication back-end implementation for Laravel is used in the BE part of the e-shop administration application.

In the back-end part, a session-based authentication is implemented. It means that the server has to get from the client information about the user's username and password to authenticate a user for the first time. If the user

enters the correct data, the system assumes the identity is valid and grants access. After the server authenticates the user for the first time, it generates the session and stores it in the database. The `SESSION_ID`, which is an identification of the session, is sent to the user via HTTP cookies. After the user receives the `SESSION_ID`, they have to send it back in each subsequent request via cookies to authenticate themselves.

The authorization process is performed using the built-in authorization services provided by Laravel using gates and policies.

Realization

This chapter describes the prototype implementation process. What steps were done to create the prototype, what problems arose while the realization, and how they were solved.

3.1 React libraries stack

material-table is a simple and powerful Datatable for React based on Material-UI Table with some additional features. [22] The needed features supporting were: filtering, sorting, actions supporting, the opportunity to hide columns, pagination, supporting of remote data, and custom column rendering.

react-grid-layout is a grid layout system for React. The needed features supporting were: it is responsive and supports breakpoints, the layout can be serialized and restored, bounds checking for dragging and resizing.

Material-UI provides React components for faster and easier web development. [23] Material-UI helps to have a unified view of the application and a unified React component programming approach.

Moment.js is a JavaScript date library for parsing, validating, manipulating, and formatting dates. [24]

react-intl is a JavaScript library that helps to internationalize React apps. This library provides React components and an API to format dates, numbers, and strings.

react-router is a collection of navigational components that compose declaratively with the application. [31]

3.2 Architecture of the app

Application architecture has a direct impact on the sustainability and extensibility of an application. Great emphasis was placed on the extensibility of the prototype, so now it is prepared for further improvements. Also, the prototype application uses reusable and easy sustainable components, thus, how was mentioned before, it prepared for the next features realization.

The administration application is a single-page application (next SPA). A single-page application is an app that works inside a browser and does not require page reloading during use. SPA is just one web page that you visit which then loads all other content using JavaScript. [25] The network communication is taking place behind the scenes, all required data are loaded before or while rendering the web page. The application was implemented using the **React** library with **React Hooks**.

The application architecture consists of three layers: **Service**, **Service**, and **View** layers.

The **Data** layer is responsible for storing the data model of the application. Every object which the application received from the backend is transformed into a model object for further use inside the application.

The **Service** layer is responsible for communication with the back-end application. It sends requests to the backend and receives the JSON objects back. After the JSON object comes, transformers within the service layer convert them into model objects.

The **View** layer contains all components which are used in the application. All the “rendering” logic, such as routing, navigation, each page look, and functionality, is here.

3.3 Model

Models in the application are classes that describe a single object used in it. Models are tightly connected with the business domain. Every object which the application receives from the back-end application should be converted into a model. Transferring of objects is implemented using JSON format, so some data types could be changed. For instance, a date format is transferred in the string format, but in the application, is needed to use a **Date** format for correct working with it. Moreover, model objects are used for data validation, if an object from the backend contains all fields it should have and if they could be converted on the correct types for further usage.

```
1 export default class Order {
2   constructor(
3     readonly id: number,
4     readonly domain: string,
5     readonly invoiceId: string | null,
6     readonly customer: Customer,
```



```

7     readonly shippingMethod: string,
8     readonly paymentMethod: string,
9     readonly paymentStatus: boolean,
10    readonly comment: string,
11    readonly total: number,
12    readonly orderId: number,
13    readonly dateAdded: Date,
14    readonly referrer: string,
15    readonly profit: number,
16    readonly agreeGdpr: boolean,
17    readonly label: boolean,
18    readonly instock: boolean,
19    readonly slovakia: boolean,
20    readonly viewed: boolean
21  ) {
22  }
23 }

```

Listing 3.1: Order.ts

As you can see on 3.1 listing every field has a type declaration, so the application always has control over data types.

3.4 Services and transformers

Services

Service classes are responsible for the communication with the back-end application. How the communication takes place in detail you could find in the 2.3 section.

On the 3.2 listing you can see easy extensible list of the back-end endpoints, URLs for serving the data.

```

1  const DEV_SERVER_URL = process.env.REACT_APP_API_SERVER_URL as string;
2
3  const ORDERS_URL = `${DEV_SERVER_URL}/orders`;
4
5  export const ApiEndpoints = {
6    orders: {
7      all: ORDERS_URL,
8      detail: (id: string) => `${ORDERS_URL}/${id}`,
9      productsById: (id: string) => `${ORDERS_URL}/${id}/products`,
10     historyById: (id: string) => `${ORDERS_URL}/${id}/history`,
11     addressesById: (id: string) => `${ORDERS_URL}/${id}/addresses`,
12     packagesById: (id: string) => `${ORDERS_URL}/${id}/packages`,
13     /* ... */
14   },
15   user: {
16     login: `${DEV_SERVER_URL}/login`,

```

3. REALIZATION

```
17     logout: `${DEV_SERVER_URL}/logout`
18   }
19 };
```

Listing 3.2: ApiEndpoints.ts

On the 3.3 listing you can see an instance of the service class. Here is `OrderService.ts`, which is responsible for getting all data related with orders. In this example the method `getOrderHistoryById()` gets the history of the order with the specific id. On the listing is illustrated, that the application expects to receive an array of objects of `HistoryApiObj` type. Then if there are no errors, received array is passed to a transformer.

```
1  /* ... */
2  // Description of the JSON object, which application receives from the BE
3  export interface HistoryApiObj {
4    order_history_id: number;
5    order_id: number;
6    comment: string;
7    order_status_id: number;
8    previous_osid: number;
9    date_added: string;
10   notify: number;
11 }
12
13 export default class OrderService {
14
15   /* ... */
16
17   public static getOrderHistoryById(id: string): Promise<Array<HistoryDetail>> {
18     return axios.get(ApiEndpoints.orders.historyById(id), {withCredentials: true})
19       .then((response: AxiosResponse<Array<HistoryApiObj>>) => {
20         return HistoryTransformer.transformHistory(response.data);
21       })
22       .catch(e => {
23         error("Failure: getting order history.", e);
24         throw e;
25       })
26   }
27
28   /* ... */
29 }
30
```

Listing 3.3: OrderService.ts

Transformers

Transformers are responsible for the conversion between objects received from the back-end application and model objects.

The example illustrated on the 3.4 shows how the transformer looks like. In this example, the application transforms the array of the HistoryApiObj to the array of the HistoryDetail model object.

```

1
2 export default class HistoryTransformer {
3
4   static transformHistory(history: Array<HistoryApiObj>): Array<HistoryDetail> {
5     return history.map(historyApiObj =>
6       HistoryTransformer.transformHistoryDetail(historyApiObj));
7   }
8
9   static transformHistoryDetail(historyApiObj: HistoryApiObj): HistoryDetail {
10    return new HistoryDetail(
11      historyApiObj.order_history_id,
12      historyApiObj.order_id,
13      historyApiObj.comment,
14      historyApiObj.order_status_id,
15      historyApiObj.previous_osid,
16      CommonTransformer.transformDate(historyApiObj.date_added),
17      CommonTransformer.transformNumberToBoolean(historyApiObj.notify)
18    );
19  }
20
21  /* ... */
22 }

```

Listing 3.4: HistoryTransformer.ts

Using services from the components

The general pattern within the application is executing data loading using the `useEffect` React hook. Before the component mounted it had called the service method to load the data. Here on the 3.5 listing is illustrated, how it looks in the application. Every component has under control what to do with the data if the Promise is fulfilled or if there is an error and the Promise is rejected.

```

1   /* ... */
2
3   useEffect(() => {
4     if (!loaded) {
5       OrderService.getOrderHistoryById(orderId.toString())
6         .then(data => {
7           /* ... */
8         })
9         .catch(e => {
10          /* ... */
11            throw e;

```

3. REALIZATION

```
12         });
13     }
14     }, [loaded]);
15
16     /* ... */
17 }
```

Listing 3.5: HistoryCard.tsx

3.5 Internationalization

The prototype supports two languages: Czech and English. The internationalization in the application using the `react-intl` library.

```
1
2 /**
3  * Localization keys used within the application.
4  * The translations themselves are located in cs.json and en.json
5  * files under the given ID.
6  */
7 export const messages = {
8   app: {
9     home: {id: "app.home"} as MessageDescriptor,
10    loading: {id: "app.loading"} as MessageDescriptor,
11    yes: {id: "app.yes"} as MessageDescriptor,
12    /* ... */
13    orderDetail: {
14      title: {id: "app.orderDetail.title"} as MessageDescriptor,
15      /* ... */
16    }
17  }
18 }
```

Listing 3.6: messages.ts

```
1 <FormattedMessage {...messages.app.orderDetail.statistics} />
```

Listing 3.7: Example of using `react-intl` components

```
1 const intl = useIntl();
2 intl.formatMessage(messages.app.orderDetail.detail.printInvoice)
```

Listing 3.8: Example of using `react-intl` hook

On the 3.6 listing you can see how looks the `messages` constant helps to organize the application text labels into the hierarchy. On the 3.7 and 3.8 listings are shown examples of using this library in the application.

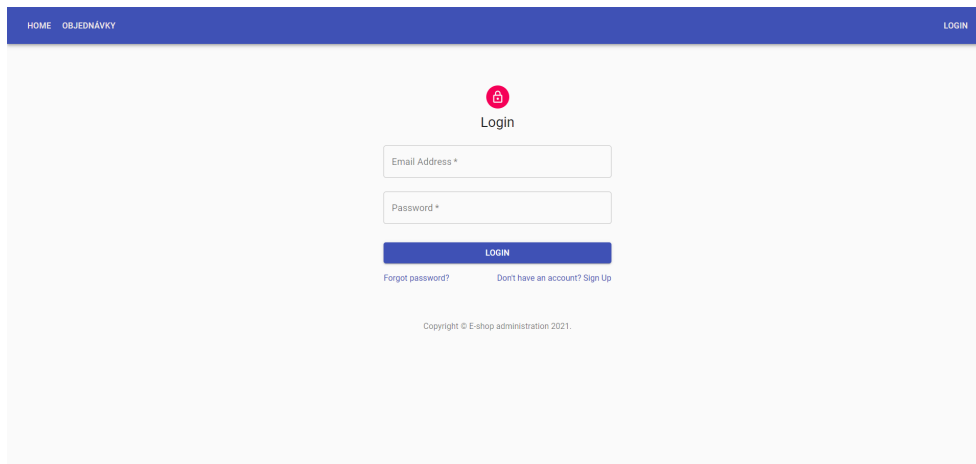


Figure 3.1: Login page

3.6 Implemented prototype

Currently, in the prototype are implemented three pages: the Login page, the Orders page, and the Order detail page. As mentioned above, the application is prepared for further improvements and extensions.

How the Login page looks is presented in Figure 3.1. After the user logged in, they find themselves on the Orders page. The Orders page is shown in Figure 3.2. The Orders page contains all required columns described in the 2.1.2 section. By default, all the columns are shown. The order of columns could be changed and each column could be hidden.

There is a search bar that allows to search data within the table. Near the search bar is “Hide columns” icon. After clicking on the icon, a selection menu appears. In this menu, a user can choose which columns will be visible in the table. Then there is the filter row where a user can filter data within each column. For some columns, a user can filter values in a selected range.

After double-clicking on a row or after clicking on the “Edit” icon the user goes on the Order detail page that is presented in ref fig: currentPrototypeDetail Figure. In Figure ref fig: currentPrototypeDetail is shown the default layout of the page. A user can change the size and position of each panel. Each panel could be edited independently to the other. The current version of the prototype supports presenting data received from the back-end application.

After clicking on each icon or button in the system a user sees pop-up window with a message if an action was succeeded or not. It helps a user to know at every moment in which state the application is during the interaction. The navigation bar and breadcrumbs help a user to understand in which part of the application they currently are.

3. REALIZATION

HOME OBJEDNÁVKY LOGOUT

Home / Objednávky

Objednávky

Search

VYMAZAT FILTRY

<input type="checkbox"/>	Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Způsob dopravy	Má štetek	Datum vytvoření	Celkem	Způsob úhrady	Uhrázeno	Zisk	DPH na Slovensko	Skladem
<input type="checkbox"/>		35001	20193890	www.kamen.cz	Ja Maršálková		Zákazník převzal zboží	Zásilkovna		13. 11. 2019	1 139,00 Kč	Na dobítku		169,23 Kč		
<input type="checkbox"/>		35002	20193891	www.mjg-mediana.cz	Františka Míšková		Zákazník převzal zboží	Česká pošta (Balk Do ruky)		13. 11. 2019	818,00 Kč	Na dobítku		169,55 Kč		
<input type="checkbox"/>		35003	20193892	www.muj-remington.cz	Petr Křážík		Zákazník převzal zboží	Geis		13. 11. 2019	1 339,00 Kč	Na dobítku		143,28 Kč		
<input type="checkbox"/>		35004	20193893	www.styka.cz	Ivana Koptíková		Zákazník převzal zboží	Zásilkovna		14. 11. 2019	739,00 Kč	Na dobítku		81,44 Kč		
<input type="checkbox"/>		35005	20193878	www.muj-russephobbs.cz	Jindřiška Šibánková		Zákazník převzal zboží	Zásilkovna		14. 11. 2019	1 049,00 Kč	Na dobítku		61,89 Kč		
<input type="checkbox"/>		35006	20193894	www.styka.cz	Marcela Káňáková		Zákazník převzal zboží	Zásilkovna		14. 11. 2019	3 690,00 Kč	Na dobítku		568,43 Kč		
<input type="checkbox"/>		35007	20193899	www.styka.cz	Petra Voborníková		Zákazník převzal zboží	Zásilkovna		14. 11. 2019	589,00 Kč	Bankovní převod		62,07 Kč		
<input type="checkbox"/>		35008	20193895	www.styka.cz	Jiřka Voráčková		Zákazník převzal zboží	Zásilkovna		14. 11. 2019	1 149,00 Kč	Na dobítku		56,24 Kč		
				www.muj-	Monika											

10 rows | 1-10 of 510

Figure 3.2: Orders page

HOME OBJEDNÁVKY LOGOUT

Home / Objednávky / Detail objednávky

Podrobnosti

ID objednávky: 35001

Stav: **Zákazník převzal zboží**

Číslo faktury: 20193890

Způsob úhrady: **Na dobítku**

Datum vytvoření: 13. 11. 2019

Uhrázeno: **Ano**

Komentář od zákazníka:

Způsob dopravy: **Zásilkovna**

DPH na Slovensko: **Ne**

Státníky: **169,23 Kč**

Adresy

Fakturační adresa

Jméno: Jana Příjmení: Maršálková

Město: Praha PSČ: 120 00

Číslo: 123456789

Česká republika

Doručovací adresa

Jméno: Jana Příjmení: Maršálková

Město: Praha PSČ: 120 00

Číslo: 123456789

Česká republika

Produkty

Produkt	Model	Záruka	INT	EXT	Počet	Cena/kus bez DPH	Celkem bez DPH	DPH	Celkem s DPH	Actions
Remington Zastřihovač vosů MB4128 Boss Manchester United	MB4128	24	1	0	1	908,26 Kč	908,26 Kč	190,74 Kč	1 099,00 Kč	
Na dobítku + Zásilkovna					1	24,79 Kč	24,79 Kč	5,21 Kč	30,00 Kč	
Celkem						933,06 Kč	933,06 Kč	195,94 Kč	1 129,00 Kč	

Kontaktní údaje

E-mail: J789

Telefon: 777347812

Ostatní

Souhlas: **Ne**

Soubor: http://m.facebook.com/

Zásilky

Číslo balíku	Dopravce	Vyvořeno	Předána dopravci	Na dobítku	Actions
2779088408	Zásilkovna	14. 11. 2019	15. 11. 2019	1 139,00 Kč	

Detail zásilky

Pro: Jana Maršálková

Hmotnost: 0,532 kg

Stav: 7

Historie

- Neúspěšně → Nevyřizováno
- Neúspěšně → Expedice
- Expedice → Odesláno dopravcem
- Odesláno dopravcem → Zákazník převzal zboží

Průběh

15. 11. 2019

Stav: **Zákazník převzal zboží**

Komentář: Vážený zákazníku, děkujeme Vám za převzetí Vaší zásilky. S pozdravem Kateřina Hurková, DIS

Figure 3.3: Order detail page

3.7 Integration with the back-end application

As mentioned in the Introduction chapter, the back-end application was created by colleague Bc. Radomír Koudela in his bachelor thesis. This section describes what has been done to make front-end and back-end applications communicate to each other.

3.7.1 Authentication and authorization

3.7.1.1 CORS

The first encountered problem was setting CORS policies. As mentioned in the 2.4.1 section about CORS, the front-end part could not communicate with the back-end part without additional settings, because CORS is disabled by default. Unfortunately, there were not any CORS configurations on the back end, so it was needed to set.

For inspecting and filtering HTTP requests entering the back end, Laravel provides a convenient mechanism called middleware. [26]

```
1
2 <?php
3
4 namespace App\Http\Middleware;
5
6 use Closure;
7
8 class CorsMiddleware
9 {
10     /**
11      * Handle an incoming request.
12      *
13      * @param \Illuminate\Http\Request $request
14      * @param \Closure $next
15      * @return mixed
16      */
17     public function handle($request, Closure $next)
18     {
19         $response = $next($request);
20         $response->header('Access-Control-Allow-Origin',
21             'http://localhost:3000');
22
23         $response->header('Access-Control-Allow-Methods', '*');
24
25         $response->header('Access-Control-Allow-Credentials',
26             'true');
27
28         $response->header('Access-Control-Allow-Headers',
29             'x-xsrf-token, content-type');
30     }
31 }
```

3. REALIZATION

```
31     return $response;
32   }
33 }
```

Listing 3.9: CorsMiddleware.php

As you can see in the 3.9 listing, sending the requests from the `http://localhost:3000` using `Access-Control-Allow-Origin` header was enabled. The `http://localhost:3000` is the default URL of the front-end application. Then sending requests with all HTTP methods was allowed. The `Access-Control-Allow-Credentials` response header tells browsers whether to expose the response to frontend JavaScript code when the request's credentials mode is included. The `credentials` read-only property of the Request interface indicates whether the user agent should send cookies from the other domain in the case of cross-origin requests. As demonstrated on the 3.10 listing, the `withCredentials` parameter in Axios request was set, to send the cookies for each request to the server.

```
1 axios.get(ApiEndpoints.orders.all, {withCredentials: true})
```

Listing 3.10: Example of sending AJAX request to the server

The last `Access-Control-Allow-Headers` header in 3.9 tells the browser, which headers could be sent in the request on the server. You can see here `x-xsrf-token` header, I wrote about in the 2.4.2 section.

When the middleware was added to the application, it was made possible to send HTTP requests to the server from the front-end application. Moreover, sending the authentication cookies for each request was been possible.

3.7.1.2 Sessions

On every request the back-end application sends `Set-cookie` header, where is the information about `xcsrf-token` and `SESSION_ID` values.

```
1 Set-Cookie: laravel_session=eyJpdiI6InVBZWJzRElCUGZ3ST
2 dJelNpOUtxTXc9PSIsInZhbnVlIjoIn1lqOFJMMGw1bldqdEFaVjRL
3 MUN3d3F3YSTwU2h5ajlaMWxLYmhBOC9uelkrRmZVcFZTTVJtWERpV;
4 expires=Sun, 04-Apr-2021 21:15:22 GMT; Max-Age=7200;
5 path=/; httponly
```

Listing 3.11: Set-cookie header

Here is the example of `Set-cookie` header, received from the server. As demonstrated in 3.11 header is set `SESSION_ID` value, `expires`, `Max-age`, `path` and `httponly` properties. Although, here are listed both of `expires` and `Max-age` properties, `Max-age` has priority over the `expires` property.

The `httpOnly` parameter ensures that cookies could not be read by JavaScript applications, neither my front-end application.

3.7.1.3 Authentication

There were troubles with the authentication using the username and password. The server responded with a 422 Unprocessable Entity error, although the user credentials were in the database. Hence, the authentication process, provided by Fortify package was customized.

```
1 class FortifyServiceProvider extends ServiceProvider
2 {
3     /*...*/
4
5     /**
6      * Bootstrap any application services.
7      *
8      * @return void
9      */
10    public function boot()
11    {
12        Fortify::authenticateUsing(function (Request $request) {
13            $user = Customer::where('email', $request->email)
14                ->first();
15
16            if ($user && Hash::check($request->password,
17                $user->password)) {
18                return $user;
19            }
20        });
21    /*...*/
22    }
23 }
```

Listing 3.12: FortifyServiceProvider.php

After the changes from the 3.12 listing, authentication started to work. It is useful to note, that even after an unsuccessful login attempt, the server still returns Set-cookies with the SESSION_ID. In the future version, it should be fixed. The browser sends cookies with session identification for the authentication in each request. The front-end part of the administration application stores only the boolean flag, if a user were logged in the localStorage, described in the 3.8 section.

3.7.2 Problems

This section describes problems that were faced during the front-end and back-end applications integration. The back-end application has to serve REST API, which the front-end application uses to get data.

We have been in contact with the colleague, so some of the problems, which were described here have been already fixed.

3. REALIZATION

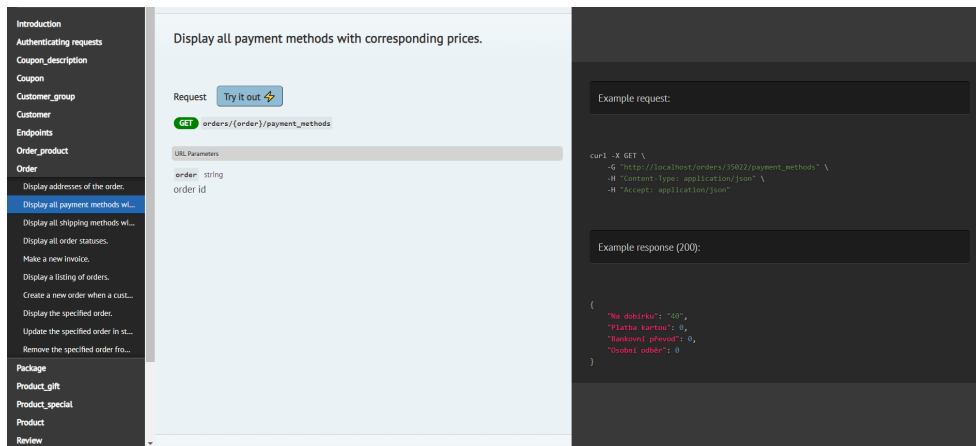


Figure 3.4: Documentation page of the endpoints

3.7.2.1 Documentation

There is poor documentation of the back-end application. Because there was no installation guide in the source code repository, a lot of time running the back-end application locally was spent.

The endpoints documentation are represented as HTML page you can see in Figure 3.4. The correct ordering is missing for some of the endpoints. For instance, the documentation for the `orders/{order}/history` endpoint could not be found until endpoints in a section **Endpoints** were listed. After it was found, there was still a problem to understand, what this endpoint does, because there was an error in the documentation, see the 3.13 listing. So, back-end application code should have been read to understand which data are sent. Another way to found out, what is returned from the endpoint was to send HTTP request directly on this endpoint.

```
1 {
2   "message": "No query results for model [App\\Models\\
3     Order] qui",
4   "exception": "Symfony\\Component\\HttpKernel\\Exception\\
5     NotFoundHttpException",
6   /* ... */
7 }
```

Listing 3.13: Error message in the endpoint documentation

Another thing that has complicated the integration with the back-end application is the inconsistency of the data types in the documentation and in the real application.

On the 3.14 listing is documentation of an `orders/{order}/products`, this endpoint returns all products within the order. On the 3.15 listing is which data the endpoint actually returns. This example illustrates that price values

have different types, a price is a number in the documentation, in the back-end application, it is a string. Those kinds of inconsistencies slowed down integration with the back-end application.

```

1  { [
2  "order_product_id":112,
3  "order_id":50,
4  "product_id":183,
5  "name":"Náhradní hřebeny SP
   -HC5000",
6  "model":"4008496717552",
7  "price":207.5000,
8  "total":207.5000,
9  "tax":21.0000,
10 "quantity":1,
11 "sort_order":0,
12 "is_transfer":0,
13 "is_action":0,
14 "purchase_price":131.8200,
15 "warranty":24,
16 "gift":0
17 ] }

```

Listing 3.14: Products within an order in documentation

```

1  [ {
2  "order_product_id": 74807,
3  "order_id": 35002,
4  "product_id": 2295,
5  "name": "Náhradní hřebeny
   SP-HC5000",
6  "model": "41019",
7  "price": "577.6860",
8  "total": "577.6860",
9  "tax": "21.0000",
10 "quantity": 1,
11 "sort_order": 1,
12 "is_transfer": 0,
13 "is_action": 0,
14 "purchase_price":"380.000",
15 "warranty": 24,
16 "gift": 0
17 } ]

```

Listing 3.15: Products within an order in real application

3.7.2.2 Shipping methods

There is an endpoint `orders/{order}/shipping_methods` with the information about all available shipping methods for an order.

On the 3.16 and 3.17 listings you could see the responses from the server for two different orders, one has shipment to the Slovak Republic, the other one to the Czech Republic. These objects contain information about a shipping company name and price, which customers should pay for the shipping using this company. As you can see, the format of these two responses is not the same, even though the business context is similar. The price for a shipping method in one case is sent as a number, in the second one as a string. This kind of inconsistency leads to a more complicated type conversion within the front-end application.

```

1  {
2  "Slovenská pošta": 129.34078965273682,
3  "Geis Slovensko": 181.18078550553716,
4  "Zásielkovňa": 77.50079379993649,
5  "GLS": 129.34078965273682
6  }

```

Listing 3.16: Shipping methods available in the Slovak Republic

3. REALIZATION

```
1 {
2   "Česká pošta (Balík Do balíkovny)": "0",
3   "Česká pošta (Balík Do ruky)": "89",
4   "Česká pošta (Balík Na poštu)": "79",
5   "Geis": "69",
6   "Zásilkovna": "0",
7   "DPD": "99"
8 }
```

Listing 3.17: Shipping methods available in the Czech Republic

Enumerations are used in the front-end application to store information about the text and color of a particular shipping company label. So the data coming from the back-end application are checked, validated, and an error are thrown if there is unexpected data. During transforming the orders data an error was thrown, that there was an unexpected shipping method “Česká pošta”, which is not in the list above, see 3.17.

```
1 /**
2  * Display all shipping methods with corresponding prices.
3  * @urlParam order required order id Example: 35022
4  *
5  * @param Order $order
6  * @return Response
7  */
8 public function getShipping_methods(Order $order)
9 {
10     if ($order->shipping_country === 'Česká republika')
11         return response()->json(
12             [
13                 "Česká pošta (Balík Do balíkovny)" =>
14                     OrderService::getTransitOrCODPrice(
15                         $order,
16                         "base",
17                         "Česká pošta (Balík Do balíkovny)"
18                     ),
19                 "Česká pošta (Balík Do ruky)" =>
20                     OrderService::getTransitOrCODPrice(
21                         $order,
22                         "base",
23                         "Česká pošta (Balík Do ruky)"
24                     ),
25                 "Česká pošta (Balík Na poštu)" =>
26                     OrderService::getTransitOrCODPrice(/* ... */),
27                 "Geis" =>
28                     OrderService::getTransitOrCODPrice(/* ... */),
29                 "Zásilkovna" =>
30                     OrderService::getTransitOrCODPrice(/* ... */),
31                 "DPD" =>
```

```

32         OrderService::getTransitOrCODPrice(/* ... */)
33     ];
34
35     else if ($order->shipping_country === 'Slovensko')
36         return response()->json(
37             [
38                 "Slovenská pošta" =>
39                     OrderService::getTransitOrCODPrice(
40                         $order,
41                         "base",
42                         "Slovenská pošta"
43                     ),
44                 "Geis Slovensko" =>
45                     OrderService::getTransitOrCODPrice(/* ... */),
46                 "Zásielkovňa" =>
47                     OrderService::getTransitOrCODPrice(/* ... */),
48                 "GLS" =>
49                     OrderService::getTransitOrCODPrice(/* ... */)
50             ]);
51     }

```

Listing 3.18: OrderController.php

On the 3.18 listing is the method, which returns hardcoded methods with their price. There are two methods that exist in a database, but do not exist in the listing above: “Česká pošta” and “Osobní odběr”.

As an improvement for a further version, using one format for the price is suggested, a number is a better choice for this field. The second suggestion is using enumerations not to send the whole name of a shipping company. Using enumerations will improve interoperability. The front-end application will be responsible for the particular texts and names. The last suggestion is using constants and functions for repeated pieces of code which will decrease the probability of errors.

3.7.2.3 Order status

The next problem was mapping order statuses. Order object which is received from the back-end application contains **Order status** ID that is a foreign key of a `oc_order_status` table which is a code list of all order statuses. So in the front-end application, status IDs with their texts are mapped. For this purpose, there is `order_statuses` endpoint. As a response after sending GET request to the endpoint a list of order status labels without IDs is received, so they could not be mapped. On the 3.19 listing is code from the controller that should return all order statuses. Here was a list of hardcoded values. I asked my colleague to change this to fetching data from the `oc_order_status` table. This problem was fixed and now the response contains IDs, see 3.20 listing.

3. REALIZATION

```
1 public function getOrder_statuses()
2 {
3     return
4     [
5         "Nevyřízeno",
6         "Ve zpracování",
7         "Odesláno dopravcem",
8         "Zákazník převzal zboží",
9         "Zrušeno obchodem",
10        /* ... */
11    ]
12 }
```

Listing 3.19: OrderController.php

The new response now contains IDs as keys of the object in a string format. As mentioned before, the models are used to validate data from the back-end application. During converting orders, an error that there is unexpected order status was thrown. There was `order_status_id = 0`, which is not exist in the response. After current application and database analysis, it was found out, that this status was “Chybějící objednávky”.

```
1 {
2     "1": "Nevyřízeno",
3     "2": "Ve zpracování",
4     "3": "Odesláno dopravcem",
5     "5": "Zákazník převzal zboží",
6     "7": "Zrušeno obchodem",
7     "10": "Nepřevzal zásilku",
8     "11": "Dobropisováno",
9     "14": "Připraveno k osobnímu odběru",
10    "15": "Čeká na uhrazení",
11    "16": "Uhrazeno - k expedici",
12    "17": "K expedici",
13    "18": "Vráceno ve 14ti dnech",
14    "19": "Zrušeno zákazníkem",
15    "20": "Zrušeno pro nezaplacení",
16    "21": "Dobropis",
17    "22": "Pouze zásilka"
18 }
```

Listing 3.20: Order statuses response

Using enumerations with the same keys on the back-end and front-end applications for easier data mapping is proposed. It could allow to send such objects, for example, `{"FOR_EXPEDITION": {"id": 17, "msg": "K expedici"}}`. This way will keep data semantics and simplify data processing.

3.7.2.4 Payment methods

On the 3.21 listing is a code from the controller which returns payment methods with their prices. As in the cases described above, there are hardcoded values instead of getting values from the database. It should have been found all missed values, which were in the database but were not in the list. The missed values were: “Běžný bankovní převod”, “Platební karta”, “Hotově”, “Poštovní spořitelna / Era”, “Na dobierku”.

```

1      /**
2       * Display all payment methods with corresponding prices.
3       * @urlParam order required order id Example: 35022
4       *
5       * @param Order $order
6       * @return Response
7       */
8      public function getPayment_methods(Order $order)
9      {
10         return response()->json(
11             [
12                 "Na dobírku" => OrderService::getTransitOrCODPrice
13                 ($order, "cod", $order->shipping_method),
14                 "Platba kartou" => 0,
15                 "Bankovní převod" => 0,
16                 "Osobní odběr" => 0
17             ]);
18     }

```

Listing 3.21: OrderController.php

Again, there would be helpful enumerations to sent some key instead of the payment method name. Unfortunately, the database has mixed data for the relatively same data, for example “Na dobírku” / “Na dobierku”. The task of the back-end application was to not just resend data from the database but fix data inconsistency before sending data to the front-end application.

3.7.2.5 Performance

When the implementation of the service layer of the front end was started, performance problems arose. It was found out that a request for getting the order detail lasted about 20 seconds, which is a large amount of time from the UI prospect. Worth mentioning, that we have access only to testing MySQL database, which doesn't contain so much data as the production database. You can imagine a user waiting for 20 seconds for the detail page rendering, all this time watching the loading screen. The application becomes unusable, employees are not able to do their work effectively. But it was one of the main objectives of my master thesis, so the problem researching was started.

```

1      public function show(Order $order)

```

3. REALIZATION

```
2     {
3         $this->authorize('accessByAdminOrCustomer', $order);
4         $id = $order->order_id;
5         return Cache::remember('order' . $id, 5,
6             function () use ($id) {
7                 return Order::getInformation()
8                     ->groupBy('oc_order.order_id')
9                     ->having('oc_order.order_id', $id)
10                    ->first();
11            });
12    }
```

Listing 3.22: OrderController.php

On the 3.22 listing you can see the method, that returns one order detail with `order_id = $id`. At first sight it was found strange that here are used `groupBy()` and `having()` clauses. Thus, exploration how `getInformation()` method works, was continued.

```
1 /**
2  * Scope a query to get order information.
3  *
4  * @param Builder $query
5  * @return Builder
6  */
7 public function scopeGetInformation($query)
8 {
9     return $query
10
11     ->leftjoin('oc_order_status',
12         'oc_order.order_status_id', '=',
13         'oc_order_status.order_status_id')
14
15     // other joins
16     /*...*/
17
18     ->leftjoin('oc_order_product_move',
19         'oc_order.order_id', '=',
20         'oc_order_product_move.order_id')
21
22     ->select(
23         /*...*/
24         DB::raw('IF((geis_package.package_order IS NOT NULL)
25             OR (postcz_package.package_order IS NOT NULL)
26             OR (zasilkovna_package.creation_time IS NOT NULL),1,0) as label'),
27         'oc_order.date_added',
28         'oc_order.total',
29         'oc_order.payment_status',
30         DB::raw('(SUM((oc_order_product.price -
31             oc_order_product.purchase_price) * oc_order_product.quantity)) /
32             (sqrt(count(oc_order_product_move.order_product_move_id))) as profit'),
```


Table 3.1: Extract from the EXPLAIN statement execution

id	select_type	table	type
1	PRIMARY	oc_order	ALL
1	PRIMARY	oc_order_status	ref
1	PRIMARY	oc_order_product	ref
1	PRIMARY	geis_package	ref
1	PRIMARY	postcz_package	ref
1	PRIMARY	zasilkovna_package	ref
1	PRIMARY	oc_order_product_move	ALL
2	SUBQUERY	oc_order_product_move	ALL

```

33     DB::raw('(IF(oc_order.shipping_country = "Slovensko",1,0)) as slovakia'),
34     DB::raw('(IF((SELECT SUM(quantity_ext)
35     FROM oc_order_product_move) = 0,1,0)) as instock'),
36     /*...*/
37     );
38 }

```

Listing 3.23: Order.php

The `scopeGetInformation()` method returns the query builder and after the returned result `groupBy()` clause is chained. The demonstrated query didn't seem so time-consuming as it was, so it was decided to explore the generated SQL query and run it directly from the database console to find out if the problem was not in the framework used for the database access.

```

1  DB::enableQueryLog(); // Enable query log
2
3  // My Eloquent query executed by using get()
4
5  dd(DB::getQueryLog()); // Show results of log

```

Listing 3.24: Query log

Here in the 3.24 listing you could see the `DB::getQueryLog()` statement, which returns the generated SQL query, for the query mentioned above. After the SQL query was gotten, it was ran from the SQL console, average duration of the query execution was about 20 seconds, so the conclusion was that operations `->groupBy('oc_order.order_id')->having('oc_order.order_id', $id)` in the getting order detail were relatively cheap. The next step was to get the execution plan to understand, why the query lasts so long. The EXPLAIN statement provides information about how MySQL executes statements.

In the table 3.1 you could see the execution plan of the SQL query. The `type` column contains the information if the join using the index - a data structure that improves the speed of data retrieval operations. You could see the table `oc_order_product_move` does not have an index on the `order_id`

3. REALIZATION

column, so the index was created. Creation of the index was done by using the 3.25 statement.

```
1 create index oc_order_product_move_order_id_index
2     on oc_order_product_move (order_id);
```

Listing 3.25: Index creation

After the index creation, the query execution lasts about 500 ms, which is a dramatic performance improvement. This way, the performance problem while retrieving the order's data was solved. The `scopeGetInformation()` were used by retrieving all orders and getting the order detail operations, so by the index generation, the speed has been increased for them both.

3.7.2.6 Suggestions for improvement

As was demonstrated in the 3.22 listing, the `groupBy()` statement was used directly in the controller. This is disappointing because usually, it is not expected to see database statements in the controller. Moreover, this code breaks the **Separation of Concern** principle, which tells that code has to be separated into layers and components that each have distinct functionality with as little overlap as possible. [27] It means that the controllers should take care only of data serving and use the service layer methods, not finishing the query creation.

Another thing, that would promote code improvement the code is the clearly separated service layer. The 3.23 listing takes place in the `Order.php` class, which is the model object. On the one hand, such queries could be placed in the model classes, because they are narrowly tied with the data layer. On the other hand, the code becomes quite hard to read, when this method is used in more than one place, but for different purposes. Right here it would be great to have a separated layer, which will take care of working with the data layer and sending data to the controllers. Also it would increase the readability of the code, because here it could be separated on the one-purpose methods such as for example `getAllOrders()` and `getOneOrderById()`.

It was expected that the back-end application will fix data inconsistency in the database, but it just resends data from the database. In the further versions of the back-end application, it would be nice to have data sanitizing and compliance with the data semantics. For example, the "Price" field is expected to be a number, not a string.

Some of the required changes from the analysis have been implemented only partially, which had an impact on the front-end part of the application. For instance, the "Was viewed" flag, which has to indicate if an order was viewed by an e-shop employee, was added but without taking business logic into account. So now every order by default has set this flag on the `false` value even though the order status is "Customer took goods".

The last recommendation for the further version of the back-end application is using enumerations for getting control over data stored in the database. Then using constants and functions to decrease the possibility of an error while copying the same parts of code.

3.8 Saving user state

As mentioned in the 2.1.2 section, there is a feature, that application remembers a user state, which columns the user has hidden, and what is the panel layout within the order detail page.

For this purpose, storing the data in local storage was chosen. Local storage is web storage, where web applications can store data locally within the user's browser.

Before HTML5, application data had to be stored in cookies, included in every server request. Web storage is more secure, and large amounts of data can be stored locally, without affecting website performance. Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

Web storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data. [28]

An array of the hidden column keys and a definition of the layout for the `react-grid-layout` library used within the detail page are stored in the local storage. The application first looks into the local storage if there are some user's preferences. In case there are not, the application will use default definitions for each component.

Testing

This chapter describes the process of testing the application prototype. The application was tested using several approaches. The application was thoroughly tested during the development. The acceptance test has been performed after the development phase. The usability test was held as a last part of the program control. The participants were the current application users and people with no experience with e-commerce applications. Usability testing helped to identify problems and their severity in the application design.

4.1 Development testing

The source code was evaluated several times for code quality, maintainability, and readability, using various tools. The main tool is ESLint - static analyzer of a code. Static means that the quality is being evaluated without actually running the code itself. For the application implementation TypeScript was used, simply said JavaScript with static type definitions. Types allow TypeScript to validate that the code is working correctly.

```
1 > eslint . --ext .ts --ext .tsx
2
3
4 Process finished with exit code 0
```

Listing 4.1: The ESLint output

To verify an application's correctness, user tests were performed after completing the implementation of each needed feature.

4.2 User acceptance testing

Acceptance testing is formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satis-

fies the acceptance criteria and to enable users, customers, or other authorized entity to determine whether or not to accept the system. [29]

Acceptance testing is a level of software testing where a system is tested for acceptability. The prototype acceptance testing with the thesis supervisor has not identified any substantial issues. The prototype meets the critical requirements raised during the analysis, therefore the application was accepted.

4.3 Usability testing

The main purpose of usability testing is to evaluate how easy an application design is to use by testing it with a group of representative users. Typically, during a test, participants will try to complete typical tasks while observers watch, listen, and take notes. The goal is to identify any usability problems, collect qualitative and quantitative data and determine the participant's satisfaction with the product. [30]

The usability testing helps to identify design flaws that otherwise would be overlooked. Usability test allows to:

- Learn if participants are able to complete specified tasks successfully.
- Identify how long it takes to complete specified tasks.
- Find out if participants are satisfied with the application.
- Identify changes required to improve user performance and satisfaction.
- Identify how well the application was designed.

Participants were asked to think out loud during usability testing to understand the participant thought process and motivation.

For the usability testing has been chosen remote-moderated method. Remote-moderated usability tests work very similarly to in-person studies. Despite the fact that a participant and a facilitator are physically separated from each other, a facilitator still interacts with a participant and asks them to perform tasks. [31]

The participant was performing tasks within the application prototype through the remote desktop connection.

4.3.1 Test plan

The goal of the plan is to document what will be tested, what metrics will be captured, who will participate in testing, and how the application will be tested. [32]

Scope - indication what will be tested.

- Web prototype of the e-shop administration application.
- Tested pages: Orders page, Order detail page.

Participants - the number and types of participants.

- 3 e-shop employees, active users of the current version of the administration application.
- 2 users without experience with e-commerce web applications.

Equipment - the type of equipment you will be using in the test.

- Desktop
- AnyDesk - application to remote access the prototype

Metrics - questions the participants will be asked.

- Satisfaction questions about the task, difficulty of the task.

Qualitative metrics - focuses on collecting insights, findings how people use the application.

- Which components are used often?
- Which data are the most important for the user?
- How comfortable is the interface? (Color scheme, chosen icons)
- Do used components help a user to increase a work efficiency in the application.

Quantitative metrics - focuses on collecting metrics that describe the user experience.

- Time spent on a task.
- Successful completion rates.

Questions after each task

- Was the task hard to understand/complete?
- What would you like to change/improve to increase the efficiency of completing the task?
- What feelings do you have about completing this task via the application?

Questions at the end

- How would you describe your overall experience with the application?

- What did you like the most about using this application?
- What did you like the least?
- What, if anything, surprised you about the experience?
- What, if anything, caused you frustration?

4.3.2 Scenarios

This section contains a list of testing scenarios which had been executed by the participants.

4.3.2.1 Orders page

Broad view

- Do you like what you see on the screen?
- Which columns do you use mostly? (If a participant is an e-shop employee)
- What information do you find the most valuable? (If a participant is an e-shop employee)
- What are the most important tasks you perform within the page? (If a participant is an e-shop employee)
- Could colors help you to navigate within the page?

Tasks

Task 1

- Scroll the table, horizontal scroll – SHIFT + mouse scroll.
- Hide next columns: **Was viewed, Domain, Customer name, VAT in the Slovakia, In stock, Referrer, Agreement.**
- Filter **Total price** column, where price should be greater than 1000 Kč.
- Filter **Profit** column, where price should be less than 100 Kč.
- Delete all filters.
- Unhide columns, which you used to using in the current administration application. (If a participant is an e-shop employee)

Task 2

- Search for the order with a number **ID** 35090.
- Tell a **shipping** and **payment** method for this order.
- Tell if the **profit** is greater than 100 Kč.
- Delete all filters / search history.

Task 3

- Filter all orders with **status** Nevyřízeno.
- Tell which orders have a **comment** from customers.
- Filter orders with printed shipping label.
- Delete all filters / search history.

Task 4

- Unhide **Domain** column.
- Search orders contains string “muj-” within the domain.
- Filter orders within 15.11.2019 – 18.11.2019 interval.
- How many orders have been already paid?
- Delete all filters / search history.

Task 5

- Filter all orders with **shipping** method: Geis, DPD, GLS
- Filter orders with **payment** method: Bankovní převod
- How many orders have **profit** greater than 150 Kč?
- Delete all filters / search history.

Task 6

- Search for the order with a number **ID** 35040.
- Go to the order detail. You can use double click on the row or edit icon.

4.3.2.2 Order detail page

Broad view

- Do you like what do you see on the screen?
- You can resize and reorder panels. Is the panel order comfortable for your usual tasks? (If a participant is an e-shop employee)
- What information do you find the most valuable? (If a participant is an e-shop employee)
- What are the most important tasks you perform within the page? (If a participant is an e-shop employee)
- Could colors help you to navigate within the page?

Tasks

Task 1

- Set the **payment method** for the order on Na dobírku.
- Fill with some text **comment** from a customer.
- Save the changes.

Task 2

- Change the **shipping address** street on Ruská 192
- Save the changes.

Task 3

- Change the order of the **products**.
- Attempt to add a new product.

Task 4

- Find out the **status** of a package.
- Attempt to print a **shipping label** for the package.
- Attempt to add a new package.

Task 5

- Change the status of the last **history event** on Zrušeno zákazníkem.

- Change the event date on 31.12.2019.
- Save the changes.

Task 6

- Try to reorganize the layout in a way you would comfortably work with a detail page. (If a participant is an e-shop employee)

4.3.3 Persons and test results

Participant number one

Veronika, e-shop employee.

Summary from the answers to questions from the 4.3.2.1 Orders broad view:

- An eye icon is a nice functionality to indicate if somebody has already viewed an order. But it is nice to have an opportunity to reset the **Was viewed** flag because sometimes you can click on the order detail by mistake. Perfectly, having a confirming dialog before resetting the flag.
- For the payment method **Cash on delivery** is unexpected to see the **Was paid** flag with the true value (green icon) because in the current version of the application this flag is not set even after a customer took their order.
- Mostly used columns: **Order ID, Invoice number, Customer name, Shipping method, Total price**. New columns, which potentially will be used: **Was viewed, In stock**.
- Important tasks:
 - The shipping method filtration.
 - Determine which orders should be processed, for which orders should be invoice and shipping label printed, and which shipments could be sent.
- Surely, the green and red icons could help, because a user could say if the value is **true** or **false** at first glance. Generally, colors could help for better navigation, but the participant does not need to have some columns colored, such as **Shipping method** or **Order status**.

Summary of all notes that were made while the tasks 4.3.2.1, for the Orders page, were being executed:

- The participant searched the “Columns” button for a long time, she started to search near the column names and inside a filter row.

4. TESTING

The screenshot shows a web application interface for managing orders. The header includes 'HOME OBJEDNÁVKY' and 'LOGOUT'. Below the header, there is a search bar and a 'VYMAZAT FILTRY' button. The main content is a table with columns: 'Je prohlíženo', 'ID objednávky', 'Číslo faktury', 'Jméno zákazníka', 'Stav', 'Způsob dopravy', 'Mě sítěk', 'Datum vytvoření', 'Celkem', 'Uhráženo', and 'Actions'. The table contains 10 rows of order data. The first row is highlighted. The 'Stav' column contains status labels like 'Zákazník převzal zboží' (green), 'Zásilkovna' (red), 'Česka pošta (Balík Do ruky)' (blue), and 'Chybějící objednávky' (red). The 'Datum vytvoření' column shows dates from 13. 11. 2019 to 14. 11. 2019. The 'Celkem' column shows values like 1 139,00 Kč, 818,00 Kč, 1 339,00 Kč, 739,00 Kč, 1 049,00 Kč, 3 690,00 Kč, 589,00 Kč, 1 149,00 Kč, 968,00 Kč, and -2 799,00 Kč. The 'Uhráženo' column shows green and red checkmarks. The 'Actions' column contains icons for edit, delete, and print. At the bottom right, there is a pagination control showing '10 rows' and '1-10 of 510'.

Figure 4.1: Custom columns setting of the first participant

- For the participant is important to have an opportunity to search by an invoice number.
- The participant would welcome the new column or different way of determining if an invoice has already been printed. Because in the current version of the application they learn, that the invoice has been printed by the **Invoice number** column. If there is an invoice number, then the invoice has been already printed. But in case, that an order has a “By card” payment method, the invoice number has been generated because of EET but the invoice has not been printed.
- Nice to have functionality is to print a shipping label immediately after printing an invoice if the carrier does not require information about a package weight.
- The participant does not usually use the filtering by the **Total price** and **Profit** columns.
- The participant used to search data within the table using the filters, not the search bar.
- The choice of the colors and icons are not a suitable for the **Comment** column. The participant was confused by the cross mark with the green colors for an order which does not have a comment from a customer. After discussing, we came to an opinion, that is better to display icon only for orders which have a comment and the color could be green or neutral, for instance - gray.

Summary from the answers to questions from the 4.3.2.2 Order details broad view:

- Currently, the prototype shows that shipping and payment addresses are not the same only when an **Address** card is in edit mode. The participant would like to see an alert either in a display mode.
- Important information within the page:
 - List of products.
 - If all products are in a warehouse.
 - Addresses.
 - If shipping label was printed.
 - Detailed information within the **Summary** card.
- Important tasks:
 - Print the invoice.
 - Changing addresses, e-mail address, and telephone number.
 - Print the shipping label.
 - Package tracking.
- Colors are not so important on the page. Maybe only “Order status” and “Agreement” could be colored.
- The participant would like to have a more visible icon or button for invoice printing.
- This page looked a little chaotic from the beginning but after panels reordering it was suitable for work.
- A comment from a customer could be highlighted to easier finding within a detail.

Summary of all notes that were made while the tasks 4.3.2.2, for the Order details page, were being executed:

- It would be nice to have a connection with “Zasilkovna” API. When you change a shipping method on the “Zasilkovna” there will appear the new window, where you can choose a dispensing point.
- The participant would prefer not to have two small panels: **Contacts** and **Other**. The information could be distributed inside other panels.
- It was hard to find a status of a package.

4. TESTING

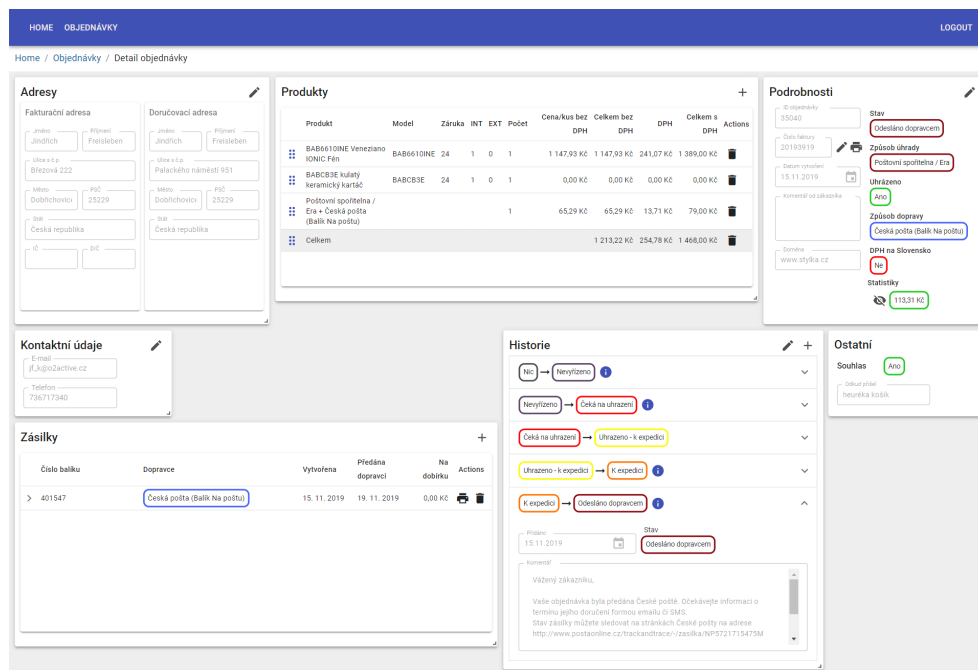


Figure 4.2: Custom detail setting of the first participant

Summary from the answers to questions at the end of testing:

- The participant likes about an application that it is customizable.
- Change a placement and highlighting of an “Print an invoice” icon is critical.
- A way of hiding columns was frustrating for the participant.

Participant number two

Libor, e-shop employee.

Summary from the answers to questions from the 4.3.2.1 Orders broad view:

- In general, the application looks good but a horizontal scroll bar is small.
- Mostly used columns: **Order ID, Invoice number, Domain, Customer name, Order status, Shipping method, Creation date, Total price, Was paid, Referrer, Agreement**. New columns, which potentially will be used: **Profit, Payment method, In stock**.
- Items in the selection menu within the filtration could be alphabetically ordered for easier navigation.

4.3. Usability testing

ID objednávky	Číslo faktury	Doména	Jméno zakazníka	Komentář	Stav	Způsob dopravy	MA	Datum vytvoření	Celkem	Způsob úhrady	Uhrázeno	Zisk	Skladem	Odžad přístav	Soutřas	Actions
35001	2019390	www.kamen.cz	Ja Mařáková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	13. 11. 2019	1 139,00 Kč	na dobrou	✓	149,23 Kč	✗	http://m.facebook.com/v	✗	✗
35002	2019391	www.mjg-mediana.cz	Františka Měřáková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	13. 11. 2019	818,00 Kč	na dobrou	✓	189,55 Kč	✗	https://www.google.com/v	✗	✗
35003	2019392	www.mjg-remington.cz	Petr Kráží	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	13. 11. 2019	1 339,00 Kč	na dobrou	✓	141,28 Kč	✗	https://www.google.com/v	✗	✗
35004	2019393	www.styka.cz	Ivana Kocetková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	14. 11. 2019	739,00 Kč	na dobrou	✓	81,64 Kč	✗	https://www.google.com/v	✗	✗
35005	2019373	www.mjg-rusellhobby.cz	Jiřina Štěpánková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	14. 11. 2019	1 049,00 Kč	na dobrou	✓	61,89 Kč	✗	https://search.saznam.cz/v	✗	✗
35006	2019394	www.styka.cz	Marek Kralčík	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	14. 11. 2019	3 690,00 Kč	na dobrou	✓	56,43 Kč	✗	https://www.dobry.cz/v	✗	✗
35007	2019399	www.styka.cz	Petra Vozžáková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	14. 11. 2019	589,00 Kč	bankovní převod	✓	62,07 Kč	✗	https://www.google.com/v	✗	✗
35008	2019395	www.styka.cz	Jiřina Vozžáková	Zákazník převez zboží	Zákazníka	Česká pošta (Balík Do ruky)	✓	14. 11. 2019	1 149,00 Kč	na dobrou	✓	56,24 Kč	✗	https://www.google.com/v	✗	✗
35009		www.mjg-remington.cz	Monika Wěřáková	Chybějící objednávky	Chybějící	Česká pošta (Balík Do ruky)	✗	14. 11. 2019	968,00 Kč	na dobrou	✗		✗	https://www.google.cz/v	✗	✗
35010	02019106	www.mjg-rusellhobby.cz	Daniel Dostál	Dobropis	Dobropis	Česká pošta (Balík Do ruky)	✗	14. 11. 2019	-2 799,00 Kč	bankovní převod	✗		✗		✗	✗

Figure 4.3: Custom columns setting of the second participant

- Important tasks:
 - The shipping method filtration.
 - The order statuses filtration.
- The participant would like to have a color configurator to select which data will be colored. Shipping methods do not need to be colored. Only some of the payment methods could be colored.

Summary of all notes that were made while the tasks 4.3.2.1, for the Orders page, were being executed:

- The participant searched for the “Hide” button in the column name area. The “Hide columns” icon is not visible and it would be better to have a button with the label.
- It would be nice to have an opportunity to set colors for **Profit** column. Because there are domains for which profit over 100 Kč is normal, and there exist domains for which the same profit is small. So, if the user could set their own preferences in values and colors would be a great feature.
- The participant would like to have a column **Weight** where will be the weight of a package.
- The participant was confused about the **Comment** column. The best choice would be displaying the icon only for orders which have a comment from a customer.

4. TESTING

Summary from the answers to questions from the 4.3.2.2 Order detail broad view:

- Important tasks:
 - Check the quantity of products and if they are in stock.
 - Check if the order was paid.
 - If **VAT on the Slovak Republic** flag is true and **Taxpayer Identification Number** is present, then control if the VAT value is zero.
 - Printing shipping labels.
 - Changing an addresses and contact information.
- The application has a small font size.

Summary of all notes that were made while the tasks 4.3.2.2, for the Order details page, were being executed:

- The participant would like to have a new column within the **Products** card - **VAT rate**.
- The participant expected that field editing starts after double click.
- The participant would like to have a cross sign at the end of text fields to delete the whole text within.
- The participant would like to see the package status as a new column within the **Packages** card.
- In the package detail must be link on the Track & Trace system for tracking the order. It would be nice to have a button to send the tracking information directly to the customer's email address with pre-filled text.
- It would be nice to have a bigger text field for a **Comment from a customer** because a longer comment would be displayed with a scroll bar.
- **Company Identification Number** and **Taxpayer Identification Number** could be validated.
- The application could show information if a customer orders in the e-shop often.
- The **VAT on the Slovak Republic** field could be hidden for orders within the Czech Republic.

4.3. Usability testing

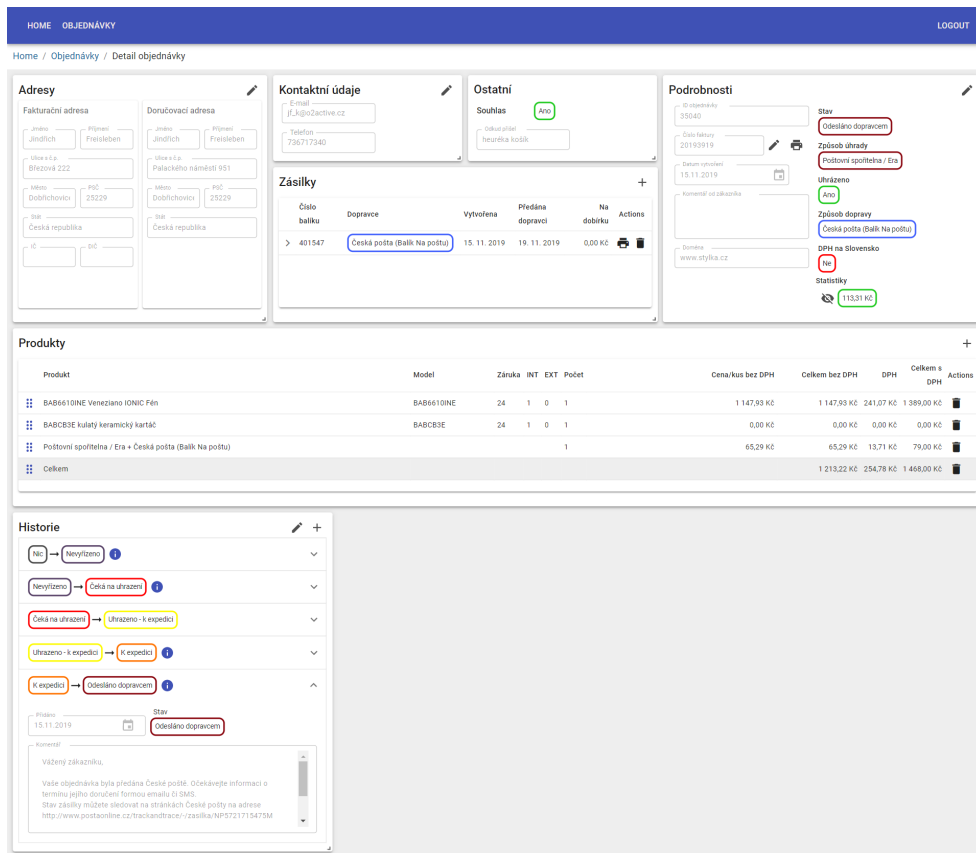


Figure 4.4: Custom detail setting of the second participant

- The participant would like to see the history event creation date and time in the history accordion header. This change would increase the efficiency of a work.

Summary from the answers to questions at the end of testing:

- The participant likes about an application that it is customizable.
- A way of hiding columns and unnecessarily coloring were frustrating for the participant.
- The application looks modern.
- The participant would like to have an opportunity of a deep search within all order properties, not only in the visible columns data.

Participant number three

4. TESTING

Alena, an economics student, has no experience with e-commerce applications.

Summary from the answers to questions from the 4.3.2.1 Orders broad view:

- The participant likes the application appearance.
- The participant thinks, that colors could help to navigate within the application.

Summary of all notes which that made while the tasks 4.3.2.1, for the Orders page, were being executed:

- The participant searched for the “Hide” button in the column name area. The “Hide columns” icon is not visible and it would be better to have a button with the label. Another option offered by the participant is having an additional menu panel where will be “Hide column” option and which will be opened by a right mouse click on a column header.
- The participant was confused by clicking outside the filter selection, she expects that the selected filter would be applied immediately after selecting and she would not do another click outside the selection menu.

Summary from the answers to questions from the 4.3.2.2 Order detail broad view:

- The participant likes the colors in the **Summary** card because they are corresponding with the colors within the Orders page. The **History** card could be less colored, the participant does not understand the meaning of colors in this card.

Summary of all notes that were made while the tasks 4.3.2.2, for the Order details page, were being executed:

- The participant expected that field editing would start after double click on the field. She would like to save the changes just by Enter pressing, not by clicking on the Save icon.
- Icon for showing the package detail is not visible, it was hard to find.

Summary from the answers to questions at the end of testing:

- In general, colors helped at work rather than disturb.
- The participant likes that all required information for work is visible.
- The participant likes about an application that it is customizable.

- The most frustrating were columns hiding and the manner of details editing. The participant wants to have the opportunity to edit fields by double clicking and to save the changes by Enter pressing.

Participant number four

Kyrylo, a software developer, has no experience with e-commerce applications.

Summary from the answers to questions from the 4.3.2.1 Orders broad view:

- The participant noted that there is a lot of information and colors on the page.
- The participant liked, that there was a breadcrumb and he knew where in which part of the system he was.
- The participant noted, that a lot of colors bothered him. But they would have an opportunity to be helpful if the participant understood their meaning.

Summary of all notes which were made while the tasks 4.3.2.1, for the Orders page, were being executed:

- The participant searched for the “Hide” button in the column name area.
- It would be nice to have the functionality to add the value to filters by clicking on the value within the table.
- In would be nice to have an opportunity to choose a year and a month in the calendar component.

Summary from the answers to questions from the 4.3.2.2 Order detail broad view:

- This page looked a bit chaotic from the beginning because the participant was used to seeing the data displayed in the list, not panels and cards.
- It would be nice to have an icon for Drag’n’Drop functionality, so the user can drag the panel only when they hold the icon. The dragging using the whole panel surface could cause inconvenience when the user would replace the panel by mistake.
- It would be nice to have an opportunity to close and restore a panel.

4. TESTING

- The participant likes the colors in the particular cards because they are corresponding with the colors within the Orders page. The **History** card could be less colored, the participant does not understand the meaning of colors in this card.

Summary of all notes that were made while the tasks 4.3.2.2, for the Order details page, were being executed:

- All tasks were executed without problems.
- The participant noted that it needs to consider if all the color boxes are required.

Summary from the answers to questions at the end of testing:

- The participant liked the application customizability.
- The participant liked that the application remembers the user preferences.
- The horizontal scroll within the Orders table would be bigger to be user-friendly.
- The participant noted that it would be nice to have a color configurator. For instance. change colors for different values of the profit.

Participant number five

Jiří, e-shop employee.

Summary from the answers to questions from the 4.3.2.1 Orders broad view:

- The participant noted that there are a lot of colors on the page. Also, there are a lot of columns and information. At first sight, it is unclear and it is hard to orient within the table.
- Green color is considered as an indicator if the order is in a correct state, red color vice versa means that order is in an incorrect state. This corresponds with expectations and with traditional meaning.
- Mostly used columns: **Order ID, Invoice number, Domain, Customer name, Order status, Creation date, Was paid, Profit, In stock, Agreement**. New columns, which potentially will be used: **Was viewed, Comment**.

- Important tasks:
 - The participant primarily solves issues which are arising in the e-shop operation. For example, why an order was not expedited when all products are in stock.
- The participant would like to change the background of an order row if all properties are in the correct state.
- Use icons and text instead of colored boxes for each state. The same for the shipping method and payment method.
- The participant likes that the **Profit** column is colored but in his opinion, the colored text would be enough, and there no need to use the colored boxes.

Summary of all notes that were made while the tasks 4.3.2.1, for the Orders page, were being executed:

- The participant found the “Hide columns” button relatively fast because he had an experience with the systems which had the same way of design this functionality.
- The participant noted, that the filter row could be also fixed as column headers for easier finding it when a user has already scrolled the table down.
- “Delete all filters” button could be in the filter row, above the actions.
- The colors of the **Comment** column are correctly used responding to the business meaning. But the icon choice is not suitable for this purpose, because users are accustomed to the color-icon combination: a green check mark and a red cross.
- The application does not obviously indicates clickable areas. So, the user never notices that he can do double click on the order row, instead of it the user always will go to the detail page using the detail icon, which is not always handy.

Summary from the answers to questions from the 4.3.2.2 Order detail broad view:

- The participant likes that the colors are corresponding with the colors within the Orders page.
- The participant expected the true/false flags would be displayed the same way as within the Orders table. For instance, **Agreement** will be represented as a green icon, not as colored text.

4. TESTING

Je prohlázeno	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Datum vytvoření	Celkem	Způsob úhrady	Uhrázeno	Zisk	Skladem	Odkud přišel	Souhlas	Actions
<input type="checkbox"/>	35001	20193900	www.kamen.cz	Ja Maršáková		Zákazník převzal zboží	13. 11. 2019	1 139,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	169,23 Kč	<input checked="" type="checkbox"/>	http://m.facebook.com/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35002	20193901	www.moj-medicea.cz	Františka Měsíková		Zákazník převzal zboží	13. 11. 2019	818,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	199,55 Kč	<input checked="" type="checkbox"/>	https://www.google.com/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35003	20193892	www.maj-remington.cz	Petr Krážíl		Zákazník převzal zboží	13. 11. 2019	1 339,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	143,28 Kč	<input checked="" type="checkbox"/>	https://www.google.com/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35004	20193893	www.stylka.cz	hana Kocianová		Zákazník převzal zboží	14. 11. 2019	739,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	81,44 Kč	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35005	20193878	www.maj-russehhobbs.cz	Jindřiška Šilhánková		Zákazník převzal zboží	14. 11. 2019	1 049,00 Kč	Platba kartou	<input checked="" type="checkbox"/>	61,89 Kč	<input checked="" type="checkbox"/>	https://search.seznam.cz/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35006	20193894	www.stylka.cz	Mareta Kadáková		Zákazník převzal zboží	14. 11. 2019	3 690,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	508,43 Kč	<input checked="" type="checkbox"/>	https://www.zbozi.cz/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35007	20193899	www.stylka.cz	Petra Vrbomířková		Zákazník převzal zboží	14. 11. 2019	589,00 Kč	Bankovní převod	<input checked="" type="checkbox"/>	62,07 Kč	<input checked="" type="checkbox"/>	https://www.google.com/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35008	20193905	www.stylka.cz	Jiřka Vozáčková		Zákazník převzal zboží	14. 11. 2019	1 149,00 Kč	Na dobrou	<input checked="" type="checkbox"/>	56,24 Kč	<input checked="" type="checkbox"/>	https://www.google.com/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35009		www.maj-remington.cz	Monika Wilsnerová		Chybějící objednávky	14. 11. 2019	968,00 Kč	Platba kartou	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	https://www.google.ca/	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	35010	02019106	www.maj-russehhobbs.cz	Daniel Dostál		Dobrosr	14. 11. 2019	-2 799,00 Kč	Bankovní převod	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	

Figure 4.5: Custom columns setting of the fifth participant

- The participant would welcome to start editing a field after double click on it.
- The participant would like to not have a **Contact** and **Other** cards, but distribute the information within other cards.
- Important tasks:
 - Checking a history on an order and update the status of the order.
 - Checking and editing of addresses and contact information.
 - Checking products within the order.
- The participant would like to highlight the product quantity in the **Products** table. The green color would indicate that the required product is in the internal stock, the red color would indicate that there are products that should be ordered.

Summary of all notes that were made while the tasks 4.3.2.2, for the Order details page, were being executed:

- The participant would like to see the package status as a new column within the **Packages** table.
- The participant expected getting to Track & Trace system after clicking on a package number.
- It would be nice to have a big button within the **Packages** table to create a package when the order has no packages yet.

4.3. Usability testing

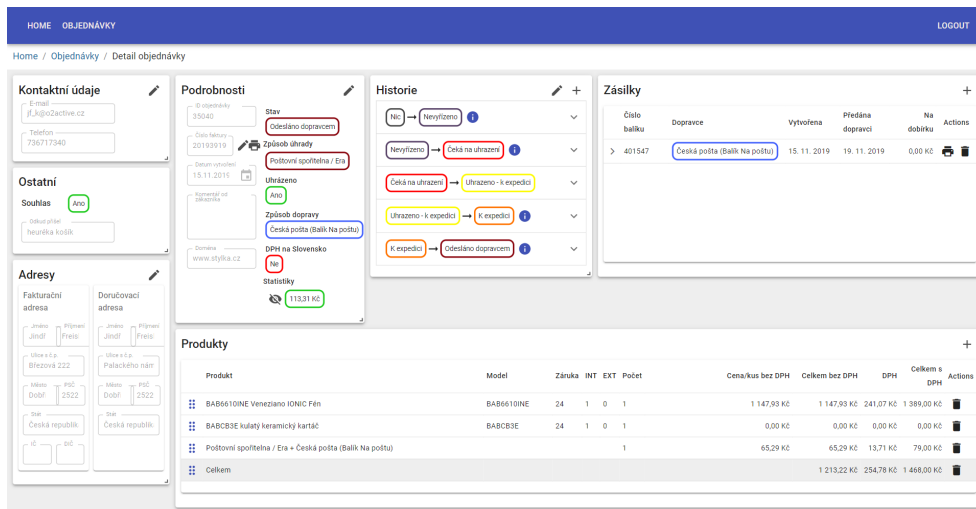


Figure 4.6: Custom detail setting of the fifth participant

- Changing history is not usual action, typically the new history event is created.

Summary from the answers to questions at the end of testing:

- The participant likes that there is an opportunity to filter all columns within the table.
- The participant expected that all detail cards would be fully responsible.
- The participant does not expect that there would a functionality of history event editing.
- The participant noticed the search bar within the Orders page at the end of the testing, the placement is not typical. The participant expected, that it would be centered or stretched over the length of the top panel.
- The participant would like to have an opportunity of a deep search within all order properties, not only in the visible columns data.
- In the further versions of the application is nice to have an opportunity for an admin to set which columns will be visible for a user.

4.3.4 Further improvements

This section summarizes the main requests for changes or improvements that were arisen during testing.

- The hiding columns functionality could be designed in another way than it is designed now using the “Hide columns” icon in the upper space of the Orders table.
- The application should have an opportunity to search within all order properties, not only in the visible columns data.
- The “Print an invoice” icon could be more highlighted for easier found on the Order detail page.
- The application could have a color configurator to select which data will be colored.
- The way of displaying if an order has the comment from the customer should be changed because it was disappointing for the users.
- The warning that shipping and payment addresses are not the same could be displayed all the time, not only in the Edit mode.
- The information about the product quantity in the internal stock is important for users, therefore it should be highlighted.
- The application should have a link to the Track & Trace system within the **Packages** table.
- The font size used within the application could be increased.

4.4 Conclusion

In the chapter Testing the prototype testing process was described, what types of tests were done, how tests were being performed, and the test results. The code quality tool did not discover any problems with the code. The application passed through the acceptance testing and was accepted. Usability testing discovered that all crucial requirements on the application were realized within the prototype. All usability tests were succeeded. The further improvements that will improve the usability of the application are summarized in section 4.3.4.

Conclusion

The objectives of this master thesis were to analyze the current version of the E-shop administration application, understand the business domain and processes that take place in the E-shop operations, collect user needs and summarize all the requirements on the new administration system. These were required for designing a low-fidelity prototype of the new emerging application. The final goal was to implement the prototype application and integrate it with the back-end application that serves the data. In the end, the created prototype was properly tested.

All the mentioned objectives were accomplished. The current application was analyzed, all user needs were assessed. The wireframes and low-fidelity prototype was designed and accepted. The application prototype was implemented, integration with the back-end part of the administration system was done and the prototype was thoroughly tested using various types of tests. Great attention was paid to usability testing with the users of the current version of the administration application.

A significant part of this thesis was the integration with the back-end application. The integration was challenging since there was not sufficient documentation and there were concepts and functionalities required for the correct communication of the front-end and the back-end application that had not been implemented. We could have been built the prototype on the mocks, this could have accelerated the development and could have had an impact on the implemented features in the prototype.

The implemented prototype will be extended to other pages, such a Products. The main benefit of this thesis is that the skeleton of the final application is ready and the designed architecture is extensible and scalable.

Bibliography

- [1] *What is User Experience (UX) Design?* [online] Interaction Design Foundation. ©2021. [Cit. 04.05.2021]. Available on: <https://www.interaction-design.org/literature/topics/ux-design>
- [2] Don Norman and Jakob Nielsen. The Definition of User Experience (UX). In: *Nielsen Norman Group* [online] Nielsen Norman Group. ©1998-2021. [Cit. 04.05.2021]. Available on: <https://www.nngroup.com/articles/definition-user-experience/>
- [3] Jakob Nielsen. Usability 101: Introduction to Usability. In: *Nielsen Norman Group* [online] Nielsen Norman Group. 03.01.2012. [Cit. 04.05.2021]. Available on: <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>
- [4] *What is a credit note?* [online]. FreeAgent. ©2021. [Cit. 20.04.2021]. Available on: <https://www.freeagent.com/glossary/credit-note/>
- [5] Vincent Xia. What are differences between wireframes, mockups and prototypes? In: *Medium* [online]. Medium. 7.7.2017. [Cit. 14.04.2021]. Available on: <https://uiux.blog/wireframe-vs-mockup-vs-prototype-selection-of-prototyping-tools-d803c4fd031d>
- [6] Rafayel Mkrtchyan. Wireframe, Mockup, Prototype: What is What? In: *UX Planet* [online]. Medium. 26.7.2018. [Cit. 14.04.2021]. Available on: <https://uxplanet.org/wireframe-mockup-prototype-what-is-what-8cf2966e5a8b>
- [7] <https://mockitt.wondershare.com/> [online]. Wondershare. ©2021. [Cit. 14.04.2021]. Available on: <https://mockitt.wondershare.com/>
- [8] Nick Babich. Prototyping 101: The Difference between Low-Fidelity and High-Fidelity Prototypes and When to Use Each.

BIBLIOGRAPHY

- In: *Adobe blog* [online]. Adobe. 29.11.2017. [Cit. 20.04.2021]. Available on: <https://blog.adobe.com/en/publish/2017/11/29/prototyping-difference-low-fidelity-high-fidelity-prototypes-use.html>
- [9] *TypeScript for JavaScript Programmers* [online] Microsoft. ©2012-2021. [Cit. 05.05.2021]. Available on: <https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html>
- [10] *About npm* [online] npm, Inc. [Cit. 05.05.2021]. Available on: <https://www.npmjs.com/about>
- [11] *React* [online] Facebook Inc. ©2021. [Cit. 05.05.2021]. Available on: <https://reactjs.org/>
- [12] *ESLint* [online] OpenJS Foundation. ©2021. [Cit. 05.05.2021]. Available on: <https://eslint.org/>
- [13] *HTTP - Overview* [online] ©2021. [Cit. 28.04.2021]. Available on: https://www.tutorialspoint.com/http/http_overview.htm
- [14] *Ajax* [online] Mozilla. ©2005-2021. [Cit. 28.04.2021]. Available on: <https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX>
- [15] *Axios* [online] The Axios Project. ©2020-2021. [Cit. 28.04.2021]. Available on: <https://axios-http.com>
- [16] *Cross-Origin Resource Sharing (CORS)* [online]. ©2005-2021. [Cit. 03.04.2021]. Available on: <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>
- [17] `preflight_correct`. In: *Cross-Origin Resource Sharing (CORS)* [online]. ©2005-2021. [Cit. 03.04.2021]. Available from: <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>
- [18] *Cross-Site Request Forgery Prevention Cheat Sheet* [online]. ©2021. [Cit. 03.04.2021]. Available on: https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html
- [19] KirstenS. *Cross Site Scripting (XSS)* [online]. ©2021. [Cit. 03.04.2021]. Available on: <https://owasp.org/www-community/attacks/xss/>
- [20] Sarah Vonnegut. 3 Ways to Prevent XSS. In: *Checkmarx* [online]. Checkmarx Ltd., 09.10.2017. [Cit. 04.04.2021]. Available on: <https://www.checkmarx.com/blog/3-ways-prevent-xss/>
- [21] *Authentication vs. Authorization* [online]. Okta. ©2021. [Cit. 04.04.2021]. Available on: <https://www.okta.com/identity-101/authentication-vs-authorization/>

-
- [22] *Get Started* [online] [Cit. 05.05.2021]. Available on: <https://material-table.com/#/docs/get-started>
- [23] *MATERIAL-UI* [online] Material-UI. ©2021. [Cit. 05.05.2021]. Available on: <https://material-ui.com/>
- [24] *Moment.js* [online] [Cit. 05.05.2021]. Available on: <https://www.npmjs.com/package/moment>
- [31] *REACT ROUTER* [online] React Training. ©2021. [Cit. 05.05.2021]. Available on: <https://reactrouter.com/>
- [25] *Single-page application vs. multiple-page application* [online]. Neoteric. 02.12.2016. [Cit. 25.04.2021]. Available on: <https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58>
- [26] *Middleware* [online]. Laravel LLC. ©2011-2021. [Cit. 04.04.2021]. Available on: <https://laravel.com/docs/8.x/middleware>
- [27] John Spacey. What is Separation of Concerns? In: *Siplicable* [online]. 27.12.2016. [Cit. 08.04.2021]. Available on: <https://simplicable.com/new/separation-of-concerns>
- [28] *HTML Web Storage API* [online]. Refsnes Data. ©1999-2021. [Cit. 28.04.2021]. Available on: https://www.w3schools.com/html/html5_webstorage.asp
- [29] *Acceptance Testing* [online] Software Testing Fundamentals. 13.09.2020. [Cit. 04.05.2021]. Available on: <https://softwaretestingfundamentals.com/acceptance-testing/>
- [30] *What is Usability Testing?* [online] Nielsen Norman Group. ©1998-2021. [Cit. 04.05.2021]. Available on: <https://www.interaction-design.org/literature/topics/usability-testing>
- [31] *Usability Testing 101* [online] Interaction Design Foundation. ©2021. [Cit. 04.05.2021]. Available on: <https://www.nngroup.com/articles/usability-testing-101/>
- [32] *Planning a Usability Test* [online] The Axios Project. ©2021. [Cit. 02.05.2021]. Available on: <https://www.usability.gov/how-to-and-tools/methods/planning-usability-testing.html>

Acronyms

AJAX Asynchronous JavaScript and XML

API Application Programming Interface

BE Back end

EET Electronic records of sales

FE Front end

GUI Graphical user interface

HTTP Hypertext Transfer Protocol

IDE Integrated Development Environment

JSON JavaScript Object Notation

PDF Portable Document Format

SEO Search Engine Optimization

SPA Single Page Application

SQL Structured Query Language

UI User interface

URL Uniform Resource Locator

UX User experience

Contents of enclosed CD

readme.txt	the file with CD contents description
src	the directory of source codes
├ administration	implementation sources
├ thesis	the directory of \LaTeX source codes of the thesis
text	the thesis text directory
├ thesis.pdf	the thesis text in PDF format
└ administration-prototype.pdf	the clickable lo-fi prototype

Attachments

Q search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Způsob platby	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	11111	25678	sty/ka.cz	Ju1 Jul1	ano	K expedici	Geis	ano	02.02.2020	1000 Kč	Kartou	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
<input type="checkbox"/>	11112	5678	sty/ka.cz	Ju1 Jul1	ne	Čeká na uhrázení	Zásilkovna	ne	02.02.2020	100 Kč	V hotovosti	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
<input type="checkbox"/>	11113		sty/ka.cz	Ju1 Jul1	ano	Ve zpracování	GLS	ne	02.02.2020	1500 Kč	Převodem	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Q search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Způsob platby	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	11111	25678	sty/ka.cz	Jul Juli	ano	K expedici	Geis	ano	2/2/2	1000 Kč	Kartou	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
<input type="checkbox"/>	11112	5678	sty/ka.cz	Jul Juli	ne	Čeká na uhrázení	Zásilkovna	ne	Equals Between	100 Kč	V hotovosti	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
<input type="checkbox"/>	11113		sty/ka.cz	Jul Juli	ano	Ve zpracování	GLS	ne	02.02.2020	1500 Kč	Převodem	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Q search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Způsob platby	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	<input type="text" value="11111"/>	<input type="text" value="25678"/>	<input type="text" value="sty/ka.cz"/>	<input type="text" value="Jul Juli"/>	<input type="text" value="ano"/>	<input type="text" value="K expedici"/>	<input type="text" value="Geis"/>	<input type="text" value="ano"/>	<input type="text" value="1/1/20 to 2/2/20"/>	<input type="text" value="100 Kč"/>	<input type="text" value="Kartou"/>	<input type="text" value="ano"/>	<input type="text" value="150 Kč"/>	<input type="text" value="ne"/>	<input type="text" value="ano"/>	<input type="text" value="https://m.heureka.cz"/>	<input type="text" value="ne"/>	<input type="text" value=""/>
<input type="checkbox"/>	<input type="text" value="11112"/>	<input type="text" value="5678"/>	<input type="text" value="sty/ka.cz"/>	<input type="text" value="Jul Juli"/>	<input type="text" value="ne"/>	<input type="text" value="Čeká na uhrázení"/>	<input type="text" value="Zásilkovna"/>	<input type="text" value="ne"/>	<input type="text" value="02.02.2020"/>	<input type="text" value="100 Kč"/>	<input type="text" value="V hotovosti"/>	<input type="text" value="ne"/>	<input type="text" value="80 Kč"/>	<input type="text" value="ne"/>	<input type="text" value="ano"/>	<input type="text" value="https://m.heureka.cz"/>	<input type="text" value="ano"/>	<input type="text" value=""/>
<input type="checkbox"/>	<input type="text" value="11113"/>	<input type="text" value=""/>	<input type="text" value="sty/ka.cz"/>	<input type="text" value="Jul Juli"/>	<input type="text" value="ano"/>	<input type="text" value="Ve zpracování"/>	<input type="text" value="GLS"/>	<input type="text" value="ne"/>	<input type="text" value=""/>	<input type="text" value=""/>	<input type="text" value="Převodem"/>	<input type="text" value="ano"/>	<input type="text" value="-50 Kč"/>	<input type="text" value="ano"/>	<input type="text" value="ne"/>	<input type="text" value="https://m.heureka.cz"/>	<input type="text" value="ano"/>	<input type="text" value=""/>

OK icon is only for demonstration. In the app filter will be applied on blur.

Q search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Komentář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Způsob platby	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	11111	25678	sty/ka.cz	Jul Juli	ano	K expedici	Geis	ano	02.02.2020	1000 Kč	Kartou	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
<input type="checkbox"/>	11112	5678	sty/ka.cz	Jul Juli	ne	Čeká na uhrázení	Zásilkovna	ne	02.02.2020	100 Kč	V hotovosti	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
<input type="checkbox"/>	11113		sty/ka.cz	Jul Juli	ano	Ve zpracování	GLS	ne	02.02.2020	1500 Kč	Převodem	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Číslo řady	Doména	Jméno zákazníka	Komentář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
25678	stylka.cz	Juli Juli		K expedici	Geis	ano	02.02.2020	1000 Kč	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
5678	stylka.cz	Juli Juli		Čeká na uhrázení	Zásilkovna	ne	02.02.2020	100 Kč	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
	stylka.cz	Juli Juli		Ve zpracování	GLS	ne	02.02.2020	1500 Kč	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Je prohlíženo
 ID objednávky
 Číslo faktury
 Doména
 Jméno zákazníka
 Komentář
 Stav
 Doprava
 Má štítek
 Datum vytvoření
 Celkem
 Způsob platby
 Uhrazeno
 Zisk
 DPH na Slovensko
 Skladem
 Příšel z
 Souhlas

Q search...

- Je prohlíženo
- ID objednávky
- Číslo faktury
- Doména
- Jméno zákazník
- Komentář
- Stav
- Doprava
- Má štítek
- Datum vytvoření
- Celkem
- Způsob platby
- Uhrazeno
- Zisk
- DPH na Slovensko
- Skladem
- Přišel z
- Souhlas

Číslo faktury	Doména	Jméno zákazník	Konemidř	Doprava	Má štítek	Datum vytvoření	Celkem	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Přišel z	Souhlas	Akce
25678	stylka.cz	Jul Juli	ano	Geis	ano	02.02.2020	1000 Kč	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
5678	stylka.cz	Jul Juli	ne	Zásilkovna	ne	02.02.2020	100 Kč	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
	stylka.cz	Jul Juli	ano	GLS	ne	02.02.2020	1500 Kč	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Q search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Konemtář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	1111	\$ 25678	stylka.cz	Jul Jüli	ano	K expedici	Geis	ano	02.02.2020	1000 Kč	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
<input type="checkbox"/>	1112	\$ 5678	stylka.cz	Jul Jüli	ne	Čeká na uhrázení	Zásilkovna	ne	02.02.2020	100 Kč	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
<input type="checkbox"/>	1113	\$	stylka.cz	Jul Jüli	ano	Ve zpracování	GLS	ne	02.02.2020	1500 Kč	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Nastavit zaplacení

Datum zaplacení:

Informovat zákazníka

search...

Je prohlíženo	ID objednávky	Číslo faktury	Doména	Jméno zákazníka	Konemtář	Stav	Doprava	Má štítek	Datum vytvoření	Celkem	Uhrazeno	Zisk	DPH na Slovensko	Skladem	Příšel z	Souhlas	Akce
<input type="checkbox"/>	1111	\$ 25678	stylka.cz	Jul Jüli	ano	K expedici	Geis	ano	02.02.2020	1000 Kč	ano	150 Kč	ne	ano	https://m.heureka.cz	ne	
<input type="checkbox"/>	1112	\$ 5678	stylka.cz	Jul Jüli	ne	Čeká na uhrázení	Zásilkovna	ne	02.02.2020	100 Kč	ne	80 Kč	ne	ano	https://m.heureka.cz	ano	
<input type="checkbox"/>	1113	\$	stylka.cz	Jul Jüli	ano	Ve zpracování	GLS	ne	02.02.2020	1500 Kč	ano	-50 Kč	ano	ne	https://m.heureka.cz	ano	

Opravdu chcete smazat objednávku?

Informovat zákazníka

Ano Ne

Detail objednávky

Nastavit zaplacení

Zpět

Podrobnosti

id objednávky 11111

Faktura č

Datum vytvoření 19. 11. 2019 08:37:31

Komentář od zákazníka
Tady je nějaký důležitý komentář...

Doména www.muji-remington.cz

Statistiky Zisk - 50 Kč

Stav Čeká na uhrázení

Způsob úhrady Převodem

Zaplaceno ne

Způsob dopravy Geis

DPH na Slovensko ne

Adresy

Fakturační adresa

Jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 111

PSČ 39

DIČ 858558

IČ 565656

Doručovací adresa

Jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 112

PSČ 390

Produkty

Dobropisovat

Produkt	Model	Záruka	Interní	Exter	Množstv	Cena/kus bez DPH	Cena bez DPH	DPH	Celke s DPH	Akce
<input type="checkbox"/> Valera řen	1111	24	-1		2	550.0 Kč	1100.0 Kč	2310 Kč	13310 Kč	
<input type="checkbox"/> Geis + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
						Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Zásilky

Přehled

Číslo balíku	Dopravce	Vyvořena	Předná doprava	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Detail zásilky

Pro Julie Evseenko

Stav Zatím nezjištěn

Zpráva pro řidiče Volat předem

Zpráva pro příjemce

Kontaktní údaje

e-mail jujulie@seznam.cz

telefon 777 777 777

Ostatní

souhlas odkud přišel ano

<https://m.heureka.cz>

Historie objednávky

Stav Ve zpracování

Datum vytvoření 16. 11. 2019 09:11:20

Komentáře

Dobry den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme.
Děkuji
S pozdravem
Kateřina Hunková, DJS

Zákazník je informován

Nevyřizeno

Ve zpracování

Nevyřizeno

nic

Detail objednávky

Nastavit zaplacení Zpět

Podrobnosti

id objednávky 11111

Faktura č

Datum vytvoření 19.11.2019 08:37:31

Komentář od zákazníka
Tady je nějaký důležitý komentář...

Doména www.mu-j-remington.cz

Statistiky Zisk - 50 Kč

Stav Čeká na uhrázení

Způsob úhrady Převodem

Zaplaceno ne

Způsob dopravy Geis

DPH na Slovensko ne

Adresy

Fakturační adresa

jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 111

PSČ 39

DIČ 858558

Doručovací adresa

jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 112

PSČ 390

Produkty

Dobropisovat

Produkt	Model	Záruka	Interní	Exter	Množstv	Cena/kus bez DPH	Cena bez DPH	DPH	Celke s DPH	Akce
Valera Ien	1111	24	-1		2	550.0 Kč	1100.0 Kč	2310 Kč	13310 Kč	
Geis + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
						Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Zásilký

Přehled

Číslo balíku	Dopravce	Vytvořena	Předána dopravci	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Detail zásilký

Pro Julie Evseenko

Stav Zatím nezjištěn

Zpráva pro řidiče Volat předem

Zpráva pro příjemce

Historie objednávky

nic **Nevyřizeno** **Ve zpracování**

Aktualizovat objednávku

Stav Ve zpracování **Informovat zákazníka**

Komentáře

Dobry den,
děkujeme za Vaši objednávku.
Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme.
Děkuji

S pozdravem
Kateřina Hunková, DIS
www.mu-j-remington.cz

Kontaktní údaje

e-mail jujujuli@seznam.cz

telefon 777 777 777

Ostatní

souhlas odkud přišel ano

https://m.heureka.cz

Detail objednávky

Podrobnosti

id objednávky 11111

Faktura č

Datum vytvoření 19.11.2019 08:37:31

Komentář od zákazníka
Tady je nějaký důležitý komentář...

Doména www.muji-remington.cz

Statistiky Zisk - 50 Kč

Stav Čeká na uhrázení

Způsob úhrady Převodem

Zaplaceno ne

Způsob dopravy Gels

DPH na Slovensko ne

Adresy

Fakturační adresa
jméno Julie Eveseniková
ulice
Kč
měs
Tá
IČ

Doručovací adresa
jméno
Dobropisovat

Produkty

Par	Poče	Cena/ku bez DP	Cena bez DP	DPH	Celkem s DPH	Alce
	2	550.0 Kč	1100.0 Kč	231.0 Kč	1331.0 Kč	
	1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
			Cena celkem:	243.0 Kč	1400.0 Kč	

Nový produkt

Produkt Doprava a způsob platby

Vyberte prosím produkt

Fen. Obvykly	500.0 Kč
Fen Obvykly	500.0 Kč
Fen Neobykly	105.0 Kč
Fen Valera	605.0 Kč

Vyberte prosím model *

Množství

* Nejdřív si zvolte produkt

Kontaktní údaje

e-mail juljuli@seznam.cz

telefon 777 777 777

Ostatní

souhlas odkud přišel ano

<https://m.heureka.cz>

Zásilkový

Přehled

Stav

Zpráva pro příjemce

Detail objednávky

Ve zprávu

Ve zprávu

Informovat zákazníka

Komentáře

Dobrý den,
děkujeme za Vaši objednávku.
Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme.
Děkují

S pozdravem
Kateřina Hunková, DIS
www.muji-remington.cz

Nastavit zaplacení

Zpět

Detail objednávky

Podrobnosti

id objednávky 11111 **Stav** Čeká na uhrázení

Faktura č **Faktura** **Způsob úhrady** Převodem

Datum vytvoření 19.11.2019 08:37:31 **Zaplaceno** ne

Komentář od zákazníka Tady je nějaký důležitý komentář... **Způsob dopravy** Geis

Doména www.muji-remington.cz **DPH na Slovensko** ne

Statistiky **Zisk** - 50 Kč

Kontaktní údaje

e-mail juljuli@seznam.cz

telefon 777 777 777

Ostatní

souhlas odkud přišel ano <https://m.heureka.cz>

Adresa

Fakturační adresa **Doručovací adresa**

jméno Julie Evesenik **jméno**

ulice **číslo** **město** **řada** **IC**

Produkty

Dobropisovat

Nastavit zaplacení

Zpět

Nový produkt

Produkt **Doprava a způsob platby**

Vyberte prosím dopravce

- GEIS
- GEIS
- Ceska Posta
- Zasilkovna

Vyberte prosím způsob platby

- Kartou
- Převodem
- Na dobírku

Cena za kus 100.0 Kč

Cena bez DPH 100.0 Kč

DPH 21.0 Kč

Celkem s DPH 121.0 Kč

* Nejdřív si zvolte produkt

Par	Poče	Cena/ku bez DP	Cena bez DP	DPH	Celkem s DPH	Alice
2		550.0 Kč	1100.0 Kč	231.0 Kč	1331.0 Kč	
1		57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
		Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Objednávka

Nový řádek

veřejněno **Ve zpracování**

Ktualizovat objednávku

Stav **Ve zpracování** **Informovat zákazníka**

Komentáře

Dobrý den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme. Děkuji

S pozdravem
Kateřina Hunková, DIS
www.muji-remington.cz

Detail objednávky

Nastavit zaplacení Zpět

Podrobnosti

id objednávky: 11111

Faktura č:

Datum vytvoření: 19.11.2019 08:37:31

Komentář od zákazníka: Tady je nějaký důležitý komentář...

Doména: www.muji-remington.cz

Stav: Čeká na uhrázení

Způsob úhrady: Převodem

Zaplaceno: ne

Způsob dopravy: Geis

DPH na Slovensko: ne

Statistiky Zisk: -50 Kč

Adresa

Fakturační adresa:

Jméno: Julie Evseenko
ulice: Kosova
město: Tábor
IČ: 565656

č. p.: 111
PSČ: 39
DIČ: 858558

Doručovací adresa:

Jméno: Julie Evseenko
ulice: Kosova
město: Tábor
PSČ: 390

č. p.: 112

Produkty

Dobropisovat

Produkt	Model	Záruka	Jmení	Exter	Poč	Cena/kus bez DPH	Cena bez DPH	DPH	Cellkem s DPH	Akce
Valera fen	1111	24	-1		2	550.0 Kč	1100.0 Kč	231.0 Kč	1331.0 Kč	
Geis + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
Fen Obvykly	123	24	-1		1	500.0 Kč	500.0 Kč	105.0 Kč	605.0 Kč	
Cena celkem:									348.0 Kč	2005.0 Kč

Zásilky

Přehled

Číslo balíku	Dopravce	Vytvořena	Předána dopravci	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Detail zásilky

Pro: Julie Evseenko
Stav: Zatím nezjištěn

Zpráva pro řidiče: Volat předem
Zpráva pro příjemce: Zpráva pro příjemce

Historie objednávky

nic

Novyřizeno

Ve zpracování

Aktualizovat objednávku

Stav: Ve zpracování Informovat zákazníka

Komentáře

Dobry den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme.
Děkuji

S pozdravem
Kateřina Hunková, DIS
www.muji-remington.cz

Ostatní

souhlas odkud přišel: ano

https://m.heureka.cz

Kontaktní údaje

e-mail: jujujul@seznam.cz

telefon: 777 777 777

Detail objednávky

Nastavit zaplacení Zpět

Podrobnosti

id objednávky: 11111

Faktura č:

Datum vytvoření: 19. 11. 2019 08:37:31

Komentář od zákazníka: Tady je nějaký důležitý komentář...

Doména: www.muji-remington.cz

Stav: Čeká na uhrázení

Způsob úhrady: Převodem

Zaplaceno: ne

Způsob dopravy: Geis

DPH na Slovensko: ne

Statistiky: Zisk - 50 Kč

Adresy

Fakturační adresa:

Jméno: Julie Evseenko
 ulice: Kosova
 město: Tábor
 IČ: 565656

č. p.: 111
 PSČ: 39
 DIČ: 858558

Doručovací adresa:

Jméno: Julie Evseenko
 ulice: Kosova
 město: Tábor
 IČ: 565656

č. p.: 112
 PSČ: 390

Produkty

Dobropisovat

Produkt	Model	Záruka	Interní	Exter	Množstv	Cena/kus bez DPH	Cena bez DPH	DPH	Celke s DPH	Akce
Valera Ien	1111	24	-1		2	550.0 Kč	1100.0 Kč	2310 Kč	13310 Kč	
Geis + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
						Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Zásilký

Přehled

Císlo balíku	Dopravce	Vytvořena	Předná dopravci	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Vytvoření zásilký

Zvolte dopravce: Geis

Počet balíků:

Dobírka:

Poznámka pro řidiče:

Poznámka pro příjemce:

Historie objednávký

nic

Nevyřizeno

Nevyřizeno

Datum vytvoření: 16. 11. 2019 09:11:20

Stav: Ve zpracování

Zákazník je informován

Komentáře: Dobry den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme. Děkuji S pozdravem Kateřina Hunková, DJS

Kontaktní údaje

e-mail: jujuji@seznam.cz

telefon: 777 777 777

Ostatní

souhlas odkud přišel: ano

https://m.heureka.cz

Detail objednávky

Nastavit zaplacení

Zpět

Podrobnosti

id objednávky 11111

Faktura č

Datum vytvoření 19. 11. 2019 08:37:31

Komentář od zákazníka
Tady je nějaký důležitý komentář...

Doména www.muj-remington.cz

Statistiky **Zisk** -50 Kč

Stav Čeká na uhrazení

Způsob úhrady Převodem

Zaplaceno NE

Způsob dopravy Geis

DPH na Slovensko ne

Adresy

Adresy nejsou stejné!

Fakturační adresa

jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 111

PSČ 39002

DIČ 8585585

IČ 565656565

Doručovací adresa

jméno Julie Evseenko

ulice Kosova

město Tábor

č. p. 112

PSČ 39002

Produkty

Dobropisovat

Produkt	Model	Záruka	Interní	Exter	Množstv	Cena/kus bez DPH	Cena bez DPH	DPH	Celke s DPH	Akce
<input type="checkbox"/>	Valera řen	1111	-1		2	550.0 Kč	1100.0 Kč	2310 Kč	13310 Kč	
<input type="checkbox"/>	Geis + Platb kartou	24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
						Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Zásilký

Přehled

Číslo balíku	Dopravce	Vytvořena	Předána dopravci	Dobírka	Akce
08092828319	Geis	19.11.2019	19.11.2019	NE	

Detail zásilký

Pro Julie Evseenko

Stav Zatím nezjištěn

Zpráva pro řidiče Volat předem

Zpráva pro příjemce Zpráva pro příjemce

Historie objednávký

nic

Nevyřizeno

Nevyřizeno

Ve zpracování

Datum vytvoření 16. 11. 2019 09:11:20

Stav Ve zpracování

Komentáře

Zákazník je informován

Dobry den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme. Děkuji

S pozdravem
Kateřina Hunková, DIS

Kontaktní údaje

e-mail

julijuli@seznam.cz

telefon

777 777 777

Ostatní

souhlas

ano

odkud přišel

https://m.heureka.cz

Detail objednávky

Nastavit zaplacení Zpět

Podrobnosti

id objednávky 11111

Faktura č

Datum vytvoření 19. 11. 2019 08:37:31

Komentář od zákazníka
Tady je nějaký důležitý komentář...

Doména www.mu-j-remington.cz

Statistiky Zisk - 50 Kč

Stav Čeká na uhrázení

Způsob úhrady Převodem

Zaplaceno ne

Způsob dopravy Gels

DPH na Slovensko ne

Adresa

Fakturační adresa

Jméno Julie Evseeniko

ulice Kosova

město Tábor

IČ 565656

č. p. 111

PSČ 39

DIČ 858558

Produkt

Model 1111

Záruka 24

Interní -1

Exter

Množstv 2

Cena/kus bez DPH 550.0 Kč

Cena bez DPH 1100.0 Kč

DPH 2310 Kč

Celke s DPH 13310 Kč

Akce

Číslo faktury 12456

Číslo faktury 12455

Číslo faktury 12453

Číslo faktury 11111

Od 1/5/2020

Do 31/5/2020

Vypiš volné číslo faktury

Resetovat číslo faktury

Produkt	Model	Záruka	Interní	Exter	Množstv	Cena/kus bez DPH	Cena bez DPH	DPH	Celke s DPH	Akce
plera řen	1111	24	-1		2	550.0 Kč	1100.0 Kč	2310 Kč	13310 Kč	
is + Platb kartou		24			1	57.0 Kč	57.0 Kč	12.0 Kč	69.0 Kč	
						Cena celkem:	1157.0 Kč	243.0 Kč	1400.0 Kč	

Zásilky

Přehled

Číslo balíku	Dopravce	Vyvořena	Předána dopravci	Dobřítka	Akce
08092828319	Gels	19.11.2019	19.11.2019	NE	

Detaily zásilky

Pro Julie Evseeniko

Stav Zatím nezjištěn

Zpráva pro řidiče Volat předem

Zpráva pro příjemce Zpráva pro příjemce

Historie objednávek

nic **Nevyřizeno** **Ve zpracování** **Stav** **Ve zpracování**

Datum vyvoření 16. 11. 2019 09:11:20

Komentáře

Zákazník je informován

Dobry den, děkujeme za Vaši objednávku. Fény jsou skladem pouze 2ks, ale ve středu dorazí další, pak ihned odešleme. Děkuji

S pozdravem
Kateřina Hunková, DJS

Kontaktní údaje

e-mail juljuli@seznam.cz

telefon 777 777 777

Ostatní

souhlas odkud přišel ano

<https://m.heureka.cz>