Decision Support System for Formation and Implementing Orders Based on Cross Programming and Cloud Computing

Maksym Fedorov[1], Andrii Berko[1], Yurii Matseliukh[1], Vadim Schuchmann[2], Ihor Budz[1], Olga Garbich-Moshora[3] and Myroslava Mamchyn[1]

[1] *Lviv Polytechnic National University, S. Bandera Street, 12, Lviv, 79013, Ukraine*
[2] *West Ukrainian National University, Lvivska Street, 11, Ternopil, 46004, Ukraine*
[3] *Drohobych Ivan Franko State Pedagogical University, Ivan Franko Street, 24,Drohobych, 82100, Ukraine*

Abstract

As part of the project, a system was developed to support decision-making in orders formation and implementation based on cross-programming and cloud computing. This system is designed for commercial use. The system is designed to automate the delivery and management of goods and the ability to receive goods without leaving home. The object of development is processing information resources in decision support systems based on cross-programming using cloud technology. The purpose of work - the design and implementation of decision support systems based on cross-programming using cloud technology. The central part of the note consists of 5 sections: an analytical review of literary and other sources, system analysis of the object of study, software solutions, practical implementation and economic part. The work result can be used as a ready-made system for the efficient and fast delivery of goods within the city. There are no such systems in Ukraine now. Forecast assumptions for developing the object of development are the expansion of functionality, coverage of more business areas, entering the market of other countries, optimization of algorithms.

Keywords 1

Decision support system, post condition, main alternative flow, main system stream, cloud computing, mobile application, detailed description, shopping cart, alternative stream, cross programming, implementing the order, delivery service, order detail, decision support, order subject customer, active order, courier app, courier service, courier, express delivery, stores and retail chains, logistics services

1. Introduction

With the advent of the internet, new segments of the population began to use the computer. If earlier the internet was primarily used by heads of large organizations, top managers, and influential officials, now the broadest segments of the population have begun to use it: children, pensioners, and the working class. Also, with the development of the internet, more and more sites and applications began to appear, both large companies and small firms offering to use their services in one way or another. Websites, applications, and entire portals started to appear where people from all over the world can enter into business transactions, communicate, and exchange information/services. It is also becoming increasingly common to buy things, real estate and other goods via the internet. Shopping is a practical,

MoMLeT+DS 2021: 3rd International Workshop on Modern Machine Learning Technologies and Data Science, June 5, 2021, Lviv-Shatsk, Ukraine
EMAIL: fedormax98@gmail.com (M. Fedorov); Andrii.Y.Berko@lpnu.ua (A. Berko); indeed.post@gmail.com (Y. Matseliukh); V.shukhmann@st.wunu.edu.ua (V. Schuchmann); ihorbudz@gmail.com (I. Budz); garbich79@gmail.com (O. Garbich-Moshora); mamchynm@ukr.net (M. Mamchyn)
ORCID: 0000-0002-3338-1140 (M. Fedorov); 0000-0003-2892-9519 (A. Berko); 0000-0002-1721-7703 (Y. Matseliukh); 0000-0002-1427-3312 (V. Schuchmann); 0000-0002-5400-0984 (I. Budz); 0000-0002-3172-5499 (O. Garbich-Moshora); 0000-0001-9230-0147 (M. Mamchyn)

exciting activity that took a lot of time, but now these times are a thing of the past because now you can buy absolutely any product in a few minutes, thanks to developing the IT-sphere on the market.

It seems that the offline store has a hall with various products that you can feel. Try on the online store is virtual and there are no such advantages. But the store, as a physical point, has a few negative details that few people know about:

- It requires significant capital investments both for premises, warehouses, etc. – and a significant number of employees;
- Analysis of revenue, expenses, advertising performance, and other indicators;
- Time – you can spend a lot of time searching for a particular outlet, as well as the risk that the right thing may not be found there;
- Geography – as mentioned above, you can lose your precious time, and it may not always be convenient to get to the proper outlet;
- Anonymity – we are all human beings, and everyone can have very piquant desires that we would not like to bring out in public, and it will not always be convenient to discuss our choice with an outsider (seller).

To solve these problems, it is not enough to analyse one specific example – you need to look at the problems as a whole.

First of all, the modern market of online stores in Ukraine and their availability for the end-user were analysed. The analysis showed significant development in this area over the past few years. Still, no one has reached the point of automating the process of search and purchase-delivery of a specific product to the buyer. After all, it is essential to save the maximum amount of time in the modern world and bring even more significant benefits in your field of employment. Some of the world's companies, such as Uber Eats, Glovo, Raketa, understand this problem and occupy a niche in the restaurant sector, thereby giving access to restaurant food to people who may never even visit the restaurant physically.

You should also find an approach to existing physical stores that have already firmly established themselves and will not tolerate competition from newcomers to the industry.

From the above and the current situation in Ukraine, we can conclude that the sphere of delivery of specific things to a certain point within the city should be developed as much as possible. Also, do not forget to use the most modern technologies and approaches to recall and adapt this system to the user quickly. This system provides for:

- The main module that uses current services and cloud computing is a server. Without it, the system cannot work;
- Design module - offers from the store;
- Offers module - analysis of product preferences and demand for the end-user;
- In this case, a module of additional services can attract drivers who will be happy to deliver goods and earn extra money on it. At the same time, drivers are not couriers as such but perform the role of volunteers who go in the same direction to the end customer;
- Display module - responsible for adapting data and displaying offers for the most up-to-date platforms.

As you can see, the system is modular, which is why it is not limited to just one area or technology, taking into account the above – **the relevance of the work** is to develop a cross-platform decision support system [1-6] using cloud computing to automate [7-9], analyse and monitor the process of buying and selling goods in physical stores [10-18], thereby providing the availability of goods for the buyer and an additional opportunity to sell goods for the seller. Based on this, combining the field of fashion in one place, and due to modularity – the ability to attract other areas [19-30].

The work aims to design and implement a decision support system based on cross programming using cloud technologies.

The object of research is processing information resources in decision support systems based on cross programming using cloud technologies.

The research subject is methods and technological means of processing information resources in decision support systems based on cross programming using cloud technologies.

The task of the work is to create a cross-platform decision support system for generating, analysing, and implementing orders using cloud computing.

To complete this task, you need to solve the following tasks:

- Conduct a literature review;
- Perform system analysis of the research object;
- Develop a conceptual model;
- Develop and display algorithms for the operation of system modules;
- Develop a software solution;
- Perform the experimental part;
- Perform an economic analysis of the system under development;
- Conclusions about the completed task.

The scientific novelty of the development lies in implementing a system that can combine and establish logistics and economic ties in various fields of activity and a new way to develop logistics links between food companies.

The practical value of the developed cross-platform system based on cloud computing is as follows:

- The cheapness of this system is that it is hosted on remote servers and does not need to be serviced by a large number of people, and most importantly, it does not need to buy (rent) additional premises or equipment;
- Analysis – the ability to view in detail all financial indicators, data on purchases/sales, advertising effectiveness and much more-all in one place;
- Cost of services - since the system does not require the fee of a large number of special equipment and employees, you can reduce the price of a standard product, thereby increasing product turnover;
- Time - reduces queues in the store. Thereby bandwidth saves the buyer time while making it easier to receive the service. Due to the current situation in the world, it reduces crowding of people in shopping centres;
- Geography – no matter what remote point of the city you are in, you can always use the services of your favourite store;
- Anonymity – ensures complete confidentiality of information about the purchase and its details;
- Cross-platform - the ability to use the system both on computers and on phones or tablets;
- Accessibility - easy to use and requires no training in maintaining or using the system;
- The convenience of payment – all system entities must be financially supported, so algorithms are provided for each user that provides a guarantee for the safety of their costs.

This system is focused on the field of fashion. Still, due to its modularity and flexibility, it can be expanded to be used in other areas such as the restaurant sector, the field of technology, and providing services of various types, and others similar. Thus, without limiting the system owner to a specific area of use. We also implemented tools for analysing available offers, which gives an idea of the current market not only for the seller as a user of the system but also for the buyer, who will choose the most fashionable, expensive or any of the most costly things. The system's modularity allows you to identify specific, highly technical points, such as: forecasting the demand for an item for the seller or recommendations based on preferences for the buyer. All this makes it possible to expand the system in any direction and at the same time re-use the existing algorithms for analysis and Decision Support. Let's consider this system from the other side of Commerce. There are new jobs for couriers, operators of maintaining the quality of service and at the same time removing the need to spend more time on another trip to the other end of the city, thereby reducing traffic on the roads and the risk of morbidity in large crowds.

2. Related works

2.1. Methodological foundations of the study

DSS formation and implementation of orders based on cross-programming and cloud computing designed to automate and optimize the delivery of goods within the city. Processing information resources in DSS formation and implementing orders based on cross-programming and cloud computing. The study subject is methods and technological means of processing information resources in decision support systems based on cross-programming using cloud technology. The purpose of the study - is the design and implementation of decision support systems based on cross-programming using

cloud technology. Now the world of services provided by seemingly completely different market entities is becoming increasingly crowded. Thus, courier companies have long been engaged in full-scale transportation and warehousing, and typical carriers enter the territory recently was "reserved" for courier companies. The key success factors were speed of delivery and electronic information about the cargo. In this area, logistics operators have emerged, offering all complex services, working as an outsourcer. These trends are changing the classical ideas about the express delivery market and actualize the research of the determinants that influence the existing segments and define the directions of the new market segments development. The delivery service services are in demand when it is necessary to transport the goods from one place to another. For entrepreneurs and extensive companies, delivery services are critical. It is due to the need to deliver goods to their consumers and partners in the shortest possible time. Not all services of courier delivery of cargoes across Ukraine can boast of a favourable combination of price and quality. The projected system was created to simplify sending and receiving orders and accelerate their delivery [31]. In 2019, the capacity of the domestic market of logistics services was 20-23 billion dollars, which marks an increase of 50-70% over five years, and compared with the past 2018 by 10-15%. 5 segments represent the domestic market of logistic services: forwarding services; professional warehousing services; express delivery, integrated logistics solutions (contract logistics), of supply chain management. Each of them has its key success factors, formed by the influence and nature of direct competition, depending on the role played by potential competitors, goods-substitutes, consumers and suppliers [32]. A dynamic segment in 2011-2012. There was a segment of express delivery services (an increase of 15-25% according to different estimates [33]), while during the quarantine period, indicators almost doubled. In the literature on logistics, we find few works, and the development of this segment in Ukraine is given.

Today, the national statistical system does not assess logistics at the regional or state level, nor does it consider the indicators that characterized the business activity of economic entities that provide logistics services. And without it, it isn't easy to get an objective and comprehensive picture of logistics, logistics infrastructure and the like. Therefore, as part of the analysis of the express delivery market, we will limit ourselves to the indicators that characterize the total number of legal entities [34].

Compared with the previous years - the number of legal entities engaged in logistics, delivery and similar activities is increasing. The last segment is forecasted to grow further. In 2019, its total capacity was already about 2 million shipments per month [35]. According to its growth, it actively affects the development of distance trade in Ukraine. All brand stores and retail chains have their Internet pages, where customers create their purchase orders, choosing a convenient way to pay and deliver. At the same time, there is a growing number of online stores, which also form the demand for express delivery services. It is evident that with the development of Internet sales, the market for express delivery will also develop. For the created system, the economic analysis and calculation of payback of a product, taking into account economic efficiency, the cost for creation and support of the application and indicators of competitiveness, is carried out. Delivery is the physical movement of goods from one point to another, for example, the movement of goods from a warehouse to a buyer or from one end of the sale to another. The entire delivery process is controlled by a transport or logistics company. Delivery can take several forms, depending on the distance travelled and the need for fast delivery. The most economical method of delivery is land, which is also the slowest. To understand how delivery works, you need to think about the logistics strategy used to receive goods by the end-user. After the first order, the next step is to contact the supplier who will send the goods to the delivery address, but in this case, the system does not take responsibility for all shipments. An effective delivery strategy is essential for any business because you need to fulfil orders on time and build a reputation for your own and suppliers' brands, thereby increasing the frequency of repeated transactions. There is a myth that the work of a courier is considered a new and not fully formed profession in our society. But this is still a myth, and he will have to deviate from the apparent facts. In ancient times, the courier profession appeared when the emperor needed to urgently inform and coordinate specific actions. And the name already came from the Latin word currere – run, move fast. The result is obvious – there were messengers in ancient times. And even today, their number is only growing. But unfortunately, in the pursuit of cheapness, high-quality service is often forgotten. All the variety of delivery services offered can be divided into several groups that will help a person decide on sending a parcel.

- By positioning - they are divided into urban, regional, within the country, and international ones;

- By destination - delivery to the warehouse in the recipient's city and "door-to-door", when the parcel is delivered only to the address;
- According to the delivery plan – the fastest, within a few hours, provides express delivery. There is also "day – today" delivery, and non-urgent-this is when delivery lasts several days.

There are a large number of types and types of deliveries. They differ from each other in a different range of services. People pay attention to when choosing a courier service is what places and regions can deliver the parcel. That is why they are divided into three types:

- Small branches. Such organizations carry out delivery in small areas, usually within the city or district. They all have a relatively small number of customers and are usually all regular customers;
- Delivery within the country. These are already much larger organizations. They have centuries of experience and at least one branch in each of the country's cities;
- International services. As a rule, these are solid companies with many branches or partner organizations around the world.

If we consider the system that is being developed, it is a system that can function at any level of the above. But to scale it, you need to start small.

2.2. State and prospects of the study

Today, the courier service is one of the most popular and promising business areas in the service sector. The idea attracts experienced aspiring entrepreneurs because it requires minimal investment and, at the same time, guarantees a high level of profit. The growth of e-commerce contributes to the development of the delivery service. People get used to shopping from the comfort of their homes. You can order delivery of almost any product, starting from dinner for the whole family and ending with pet food. As the situation developed in 2020 has shown, business on courier services can also be relevant in times of crisis, during the quarantine period, when many people work remotely and are afraid to go shopping once again. Gradually, courier services are developing new directions: for example, a subscription delivery service is growing. Another trend in the service sector is narrow specialization. Consumers perceive small firms as experts in a niche, which means they trust them more. All these trends give aspiring entrepreneurs a great chance to try their hand at the competitive courier services market. Thanks to narrow specialization, you will find your client and stand out from many competitors.

This business idea is suitable for those who have the skills of a manager, can organize work in a coordinated manner, and are ready to search for clients and partners for development actively. For a more accurate understanding of this system, it is essential to evaluate the advantages and disadvantages.

Advantages:

- The minimum amount of capital investment investments;
- Growing demand;
- It does not require renting warehouse or office space;
- There are no qualification requirements for personnel;
- Fast payback;
- Profitability up to 90%;
- Ability to scale.

Disadvantages:

- High level of competition;
- It is difficult to set up all processes at all stages of work;
- Transport is required for operation;
- Staff turnover and problems with service personnel;
- Strong dependence on unpredictable factors – weather conditions, traffic jams, and so on.

As you can see, the delivery business has several advantages, but it can face some difficulties. That is why it is necessary to assess all risks in advance and respond quickly to them.

Several other factors were also taken into account before starting the design of this system:

- The service is relevant and in demand for the local audience;
- The service is aimed at middle and high-class customers who value their time and are willing to pay for a convenient service;

- There are many business and shopping and entertainment centres in the city, respectively, there are both supply and potential consumers;
- The system works with private entrepreneurs and establishes links with trading companies, shops, boutiques, and other corporate structures.

The specifics of the market are such that the business conditions are not easy: price dumping, oversaturation of the market and the struggle for customers. It is difficult for small and newly opened companies to stay afloat. Therefore even at the start, an entrepreneur must determine the direction of their service and competitive advantages.

2.3. Analysis of known sources

Currently, there are few services on the internet that provide delivery services in a short time. Among the existing options, you should pay attention to Uber Eats, Raketa, Glovo, USPS and delivery from the store.

Uber Eats is a food delivery platform created by Uber. Under its current name, the service was launched in 2015 and currently operates in dozens of Europe, Asia, North and South America. To get started with the service, you need to download the app for Android or iOS. If you already have an Uber account, you can log in using the same username and password. On the main screen, you can see a list of available establishments. The user must select the appropriate restaurant and create an order. After that, a map will appear on the screen where courier and the order delivery time will be marked (Fig. 1).

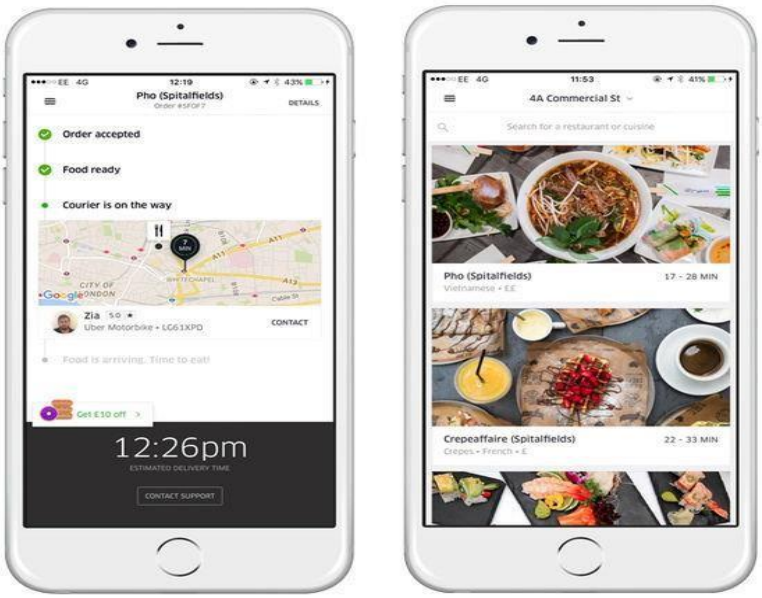


Figure 1: Uber Eats app page

You can add your own preferences to the order in the app – pick up an ingredient and other additional services. To cancel an order, please contact technical support. In some cities, the system does not work throughout the entire territory, but, for example, only in the city centre. Food can be ordered from 11 am to 11 pm in most cities, but in some locations, the platform works longer.

This platform does not have any significant features compared to similar ones. The only thing that should be noted is that the company was founded as providing passenger transportation services. Therefore, it has an advantage in the form of employment of a specific market share and the ability to distribute advertising to its new service without attracting outsiders.

Raketa is a Ukrainian service for delivering food and groceries from supermarkets using a mobile app. It has been operating since 2018. When working with the app, all partner stores, cafes, and restaurants within a certain radius, namely 2.5 km, become available. Then, just like the competitor, you need to choose an institution and dish and make a payment. The main sections of the app can be seen below (Fig. 2-3). As previously reported, the company's original plan was to introduce Ukrainians to free food delivery services for as long as possible. But due to the catastrophic events of 2020, the

company was forced to increase investment volumes and introduced paid deliveries. As of today, the company operates in ten cities of Ukraine and is expanding further.

Figure 2: The main menu of the Raketa app

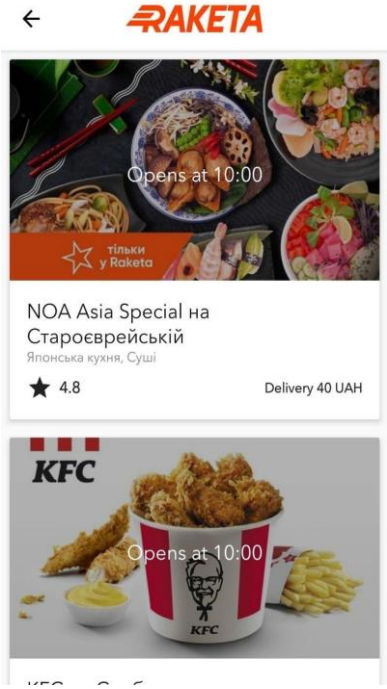


Figure 3: Menu View with restaurants in the Raketa app

Glovo is a marketplace that allows customers to order and ship products. The service delivers food, groceries, pharmaceuticals, and other products. Using a courier, the user can send parcels within the city limits. Delivery, for the most part, takes no more than an hour (Fig. 4).

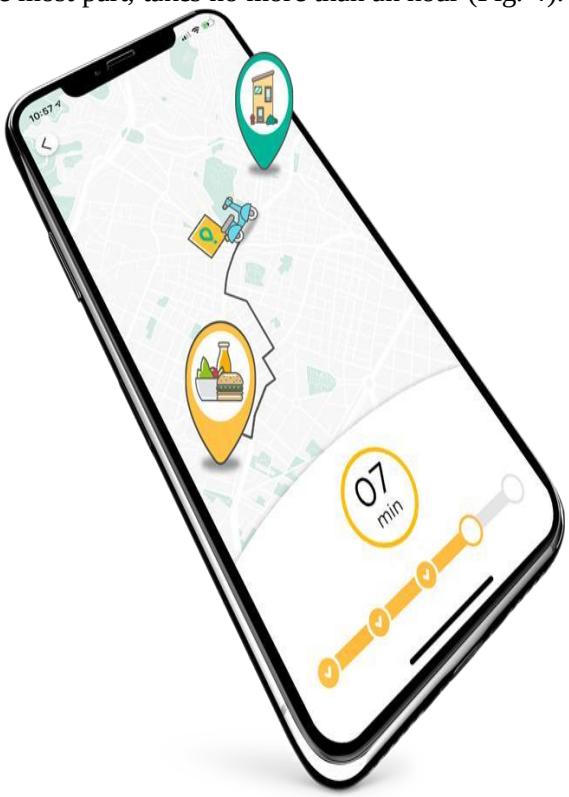


Figure 4: Conditional delivery image in the Glovo app

Relatively recently, Glovo started operating in Ukraine. At the same time, the company notes that Ukrainian users are not yet used to deliver services. Orders are most often made on special occasions, while in Europe, they are ordered constantly and personally for themselves and not for a group of people. Another difference is that Ukrainians prefer cash payments. In Glovo, you should note the availability of additional services in contrast to competitors, such as delivery from pharmacies and delivery of parcels via courier.

USPS - an international online service is becoming more popular every year. It is a state postal service that is part of the universal postal union, like all state postal services such as Ukrposhta. There are three types of delivery: regular, tracked shipment, and express (Fig. 5). This service was chosen as an example of features and versatility, unlike Ukrposhta. As mentioned above, there are three types of delivery. The difference in them is that the USPS can organize delivery directly "under the door", of course, with the help of the support of the express communication enterprise of Ukraine and operates based on the National Postal Service. It should also be noted that each parcel is equipped with a unique label that contains basic information about the property and a tracking number. And already in the official app, you can track each stage of parcel promotion to the end point. Also, this company is ideal for all kinds of stores as planning instant delivery on the delivery side. But unfortunately not in Ukraine.

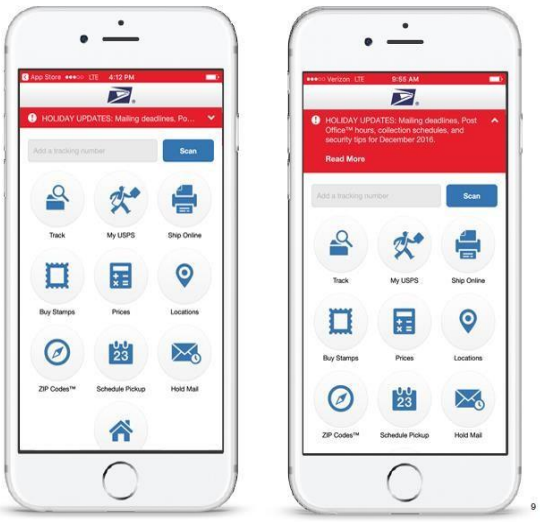


Figure 5: Main menu in the USPS app

Own delivery from the store – all major brands can deliver goods to the apartment itself, but if we talk about restaurants – Ukrainian citizens have long started using it. Still, before the arrival of such courier companies as Glovo or Raketa, this was not often enough. And in the field of fashion and in general, almost do not use and still. On the part of the point of sale, this may not be very convenient, since they need to either hire a courier or cooperate with some delivery service - this can affect the point's profit. That is why today, restaurants have preferred to collaborate with these companies despite their couriers. It is also worth noting that with precise logistics, these companies simply interrupt all hired couriers with a restaurant, which in turn attracts all personnel on the one hand and facilitates work on the other-leaving the restaurant (for example) unnecessarily adjust the logistics of personal couriers.

After analyzing documentation and other information resources, research principles and development prospects, we can conclude that countless companies deliver goods to any corner of the world, a little less that are less specialized and very few companies that specialize in any particular field of activity. Considering the qualitative analysis, the sphere of providing courier services has grown rapidly over the past year. It is in Ukraine that it is still entirely actively developing. That is why the analysis of available software solutions was carried out, and there are not enough of them, especially in Ukraine. Today, there are 3-4 that has captured the entire food delivery market in the city of the functioning systems today. Still, of all the studied ones, there is not a single one that would work with other ambassador areas since there are no systems that specialize in several locations at the same time. In my opinion, we need to expand the existing solutions and supplement them with new ones – non-standard ones.

3. Material and methods

3.1. System analysis of the research object and brief theoretical information

UML modelling is based on the following principles:

- Abstraction – the model includes only elements that are directly related to the performance of their functions by the projected system;
- Multi-model - a system can be described by several representations, each of which depicts a specific structure or behaviour;
- Hierarchy – the system is divided into different levels of abstraction and detail within its limited framework. The first level is the most abstract, and all subsequent ones are detailed up to the physical level.

All language elements can be divided into two groups: Entities and Relationships.

Entities - mostly static and correspond to physical or conceptual elements of the system, there are four types in total: structural (class, interface, cooperation, use case/use case, active class, component, and node), behavioural (interaction and automaton), grouping (package), and note entities (note).

Relationships - also have four types: dependency, association, generalization, and implementation, each of which has its symbol.

Goal tree - a structured set of goals of an organization, built according to a hierarchical principle (distributed by ranking levels) [36]. It is a visual representation of the achievement of a specific goal. The principle that the main goal is achieved through a combination of secondary and additional purposes. Therefore, the plans must be specific, surmountable and achievable, and consistent with each other. The more goals an organization sets for itself, the more complex it is regarding manageability structure [37-40]. A goal is a final state or desired result that the company seeks to achieve. This graphical representation of tasks is because the schematically presented set of goals distributed over levels resembles an inverted tree in appearance. At the top – the general-purpose ("top of the tree"); then – the subordinate sub-goals of the first, second, etc., are equal ("branches of the tree"). Sub-goals must meet five metrics: accuracy, measurability, importance, reach, and a short time frame.

When implementing an Information System, the main goal is to create and implement a high–quality and functional software product. The primary purpose of the work is to implement a decision support system for order generation, which will allow users to conveniently sell their products and receive orders quickly and from different parts of the city. For a detailed visualization of all the application goals, a goal tree was designed, which is shown in Fig. 6.

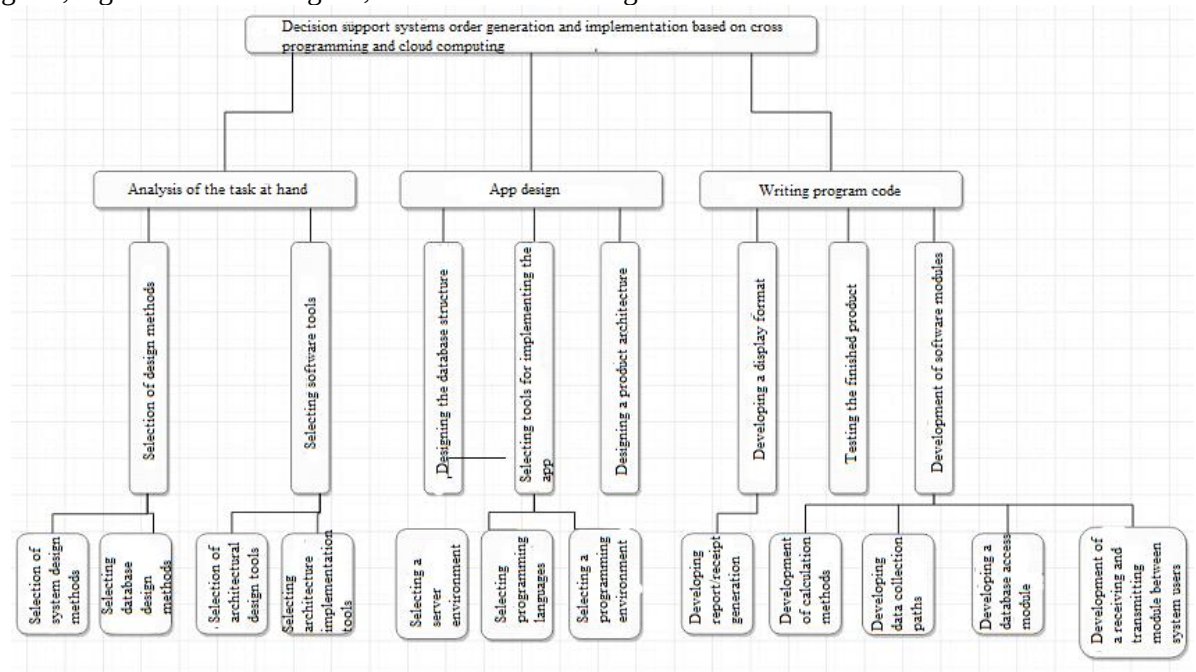


Figure 6: Structure of the app's goal tree

The goal tree for the decision support system for forming and implementing orders at Level 1 is divided into three sublevels, each of which also branches out. The main goal is to create an information system for collecting, analysing, and processing user actions in Windows. To achieve this goal, you need to complete three sub-goals: analyse the task, design the system, and write program code. To achieve the sub-goal " task analysis", you need to select the necessary design methods and software tools that the system will use. To achieve the "Application design" sub-goal, you need to plan the database structure because without it. It will be impossible to process and save input data, select the necessary implementation tools such as the programming language and the environment in which the code will be written, and design the architecture of the source product.

To achieve the sub-goal of "writing program code", you need to think through and develop a data display format, write software modules (the central logical part), and after these conditions are met, test the finished product and return to the previous stage if critical errors are detected.

3.2. Use case diagram

The use case diagram is used to display system usage scenarios and system users using its functions.

The person symbol indicates actors in the use case diagram, and use cases are indicated by an ellipse. Actors and use cases are combined by a directional association – an arrow pointing from the actor to the use case. Actors can also be connected using generalization links.

Fig. 7 shows a use case diagram showing actors and use cases.

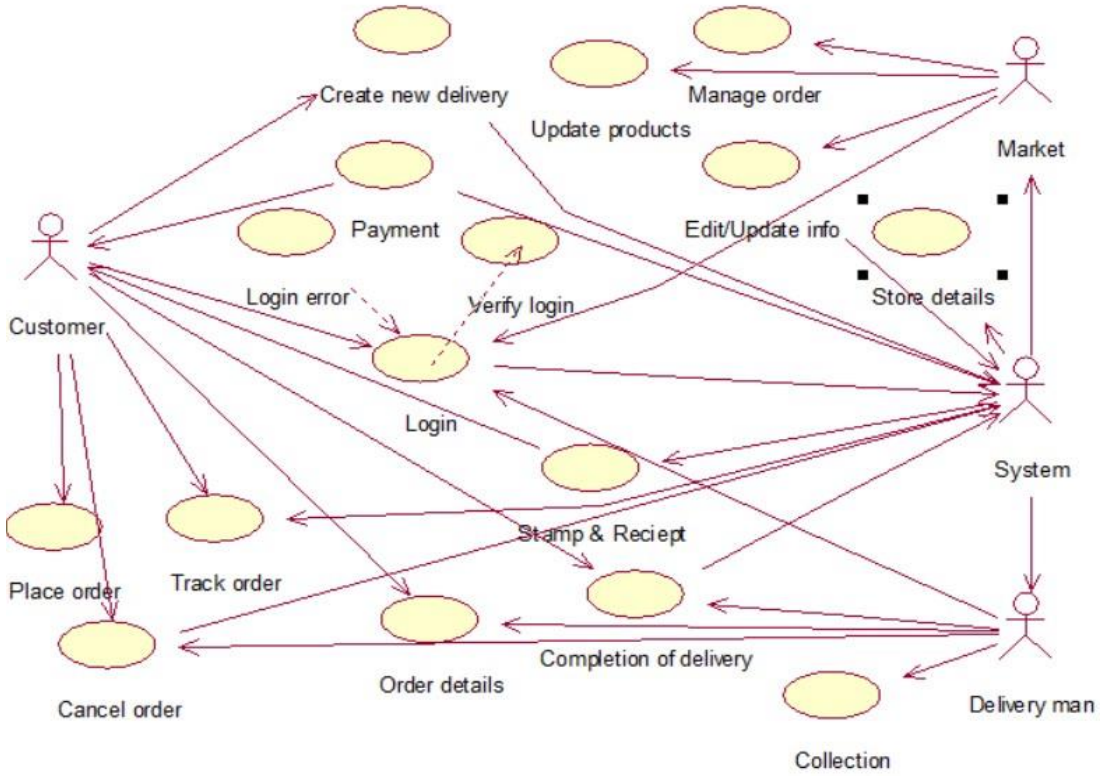


Figure 7: Use case diagram

As you can see from the figure, the system includes four actors. The first one is the customer, who can view the product and place an order. The second one is a store that can view, edit, and add products from its store. The third one is a courier (delivery man), which can view and close orders. The latter is a system. The system operates with data and distributes a specific type of load to the modules.

For more information about actors and their relationships, see the tables below.

Create delivery is used to create an order that the courier can view later. Table 1 provides a detailed description of the order creation precedent, a brief description, subjects, main alternative flows, and post-conditions.

Table 1
Creating an order

Precedent	Creating an order
Brief description	Allows you to create an order
Subjects	Customer, system
Main stream	The user sent a command to place an order
Alternative stream	The system received a command to place an order
Post-conditions	The system has processed the command for placing an order.

Payment is used to pay for the selected order. Table 2 provides a detailed description of the payment use case, a brief description, subjects, main alternative flows, and post-conditions.

Table 2
Payment

Precedent	Payment
Brief description	Allows you to pay for the order
Subjects	Customer, system
Mainstream	The user sent a command to pay for the product
Alternative stream	The system received a payment command
Post-conditions	The system confirmed the payment

Login is used for authorization and authentication in the system. Table 3 provides a detailed description of the login use case, a brief description, subjects, main alternative flows, and post-conditions.

Table 3
Log in

Precedent	Log in to the system
Brief description	Allows authorization and authentication in the system
Subjects	Customer, courier, store, system
Mainstream	The user sent a login request
Alternative stream	The system received and verified the login request
Post-conditions	The system processed the request.

Track order is used to track the movement of a courier with an active order. Table 4 provides a detailed description of the order tracking precedent, a brief description, subjects, main alternative flows, and post-conditions.

Table 4
Order tracking

Precedent	Order tracking
Brief description	Allows you to track order movements on the map in real-time
Subjects	Customer, system
Mainstream	The user opened the tracking page
Alternative stream	The system transmitted geolocation
Post-conditions	The user saw the current order location

A cancel order is used to create a declaration that the courier can view later. Table 5 provides a detailed description of the order cancellation precedent, a brief description, subjects, main alternative flows, and post-conditions.

Table 5
Order cancellation

Precedent	Order cancellation
Brief description	Allows you to cancel an order
Subjects	Customer, courier, system
Mainstream	The courier or customer sent a command to cancel the order
Alternative stream	The system received a command to cancel the order
Post-conditions	The system processed the command to cancel the order

Order details are used to get order details by creating an order. Table 6 provides a detailed description of the order details Use case, a brief description, subjects, main alternative flows, and post-conditions.

Table 6
Order details

Precedent	Order details
Brief description	Allows you to view orders
Subjects	Customer, courier, system
Mainstream	The user sent a request for details
Alternative stream	The system received a request
Post-conditions	The system processed the request for details

An order receipt (Stamp/Receipt) is used to create a receipt for a completed order with details. Table 7 provides a detailed description of the receipt order precedent, a brief description, subjects, main alternative flows, and post-conditions.

Table 7
Receipt

Precedent	Order receipt
Brief description	Allows you to receive an order receipt
Subjects	Customer, courier, system
Mainstream	The courier sent a command about the completed order
Alternative stream	The system received a command about the completed order
Post-conditions	The system processed a command about the completed order and sent it to the customer

Update products are used to update existing products. Table 8 provides a detailed description of the product update precedent, a brief description, subjects, main alternative flows, and post-conditions.

Table 8
Product updates

Precedent	Product updates
Brief description	Allows you to update product details
Subjects	Store, system
Mainstream	The user sent a command to update the product
Alternative stream	The system received a command to update the product
Post-conditions	The system has processed a command to update the product.

Manage order is used to get information about the number of orders, statuses, and other indicators. Table 9 provides a detailed description of the order management use case, a brief description, subjects, main alternative flows, and post-conditions.

Table 9
Order management

Precedent	Order management
Brief description	Allows you to manage all orders
Subjects	Store, system
Mainstream	The user sent a command to get information
Alternative stream	The system received a command to get information
Post-conditions	The system processed the command to get information

Viewing the list of active orders (Collection) is used to get data about existing requests for the product delivery. Table 10 provides a detailed description of the use case view of the list of active orders, a brief description, subjects, main alternative flows, and post-conditions.

Table 10
Viewing the order list

Precedent	View the list of active orders
Brief description	Allows you to receive all existing orders
Subjects	Courier, system
Mainstream	The user sent a request to get a list of orders
Alternative stream	The system received a request to get a list of orders
Post-conditions	The system processed a request to get a list of orders

Class diagram basic diagram for creating application code. The class diagram establishes the system's internal structure and describes classes' inheritance and relative position close to each other. In addition, it represents the logical representation of the system. The most logical thing is that classes are just blanks, based on which physical objects will then be defined. The designed system contains quite a lot of structural units, modules, and abstraction levels, so the architecture of one of the main modules of the system, namely the payment preparation module, was shown for review (Fig. 8).

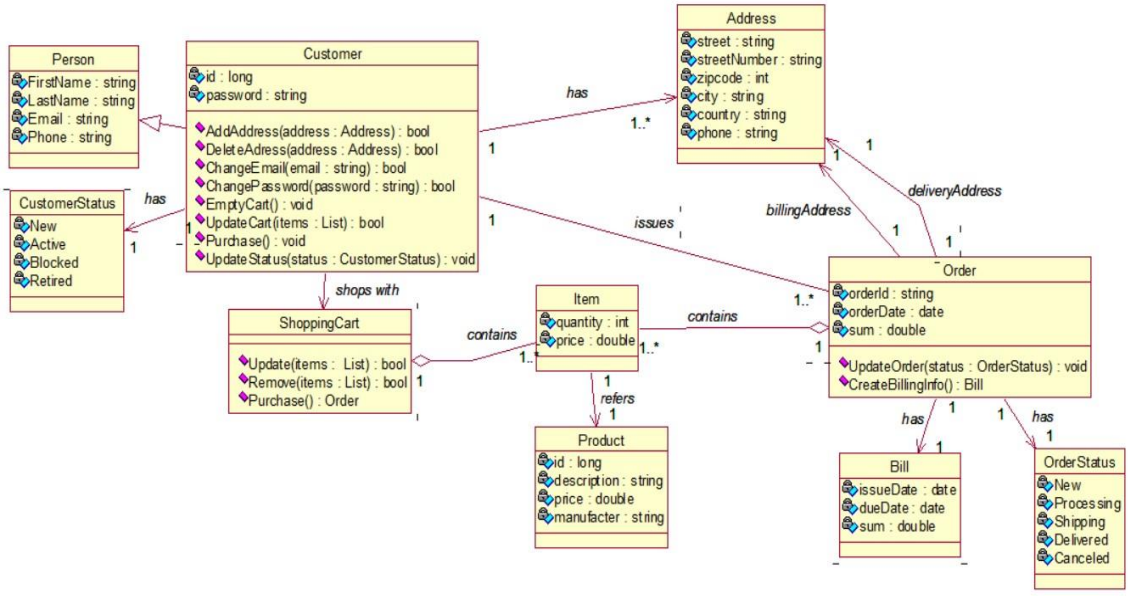


Figure 8: Class Diagram

Figure 8 shows the abstraction of classes that are part of the most powerful and most crucial payment module. The main object is a customer who has private fields such as their id and password. Additionally, the class is expanded with objects for generalizing the customer's entity and its current status in the system with the following fields: first name, last name, email address, phone number. Finally, this class has several methods for operating the data it contains, namely:

- Add address - the method accepts the current geolocation and returns the result whether the process ended correctly and whether the data was saved;
- Delete addresses - this method is used to delete the current address;
- Change email address - this method is used to update the email address;
- Clear cart - this method is used to delete saved items from the order cart;
- Update shopping cart - required for editing or adding items to the shopping cart;
- Order – needed to place an order for all available items in the shopping cart;
- Update status – used to update the client's situation.

The geolocation class mentioned above has the following structure: Street, street number, postal code, city, country. Another module collects data, which automatically fills in all fields at start-up for further processing by the system. *The shopping cart class* contains three reliable methods for adding, deleting an item, and processing existing items for different order creation.

The order object contains three fields: a unique identifier, date, and amount, where the data is filled in by another module that generates the id, sums and analyses the available products in the shopping cart and fills in these fields if filled in correctly. Methods for updating order and creating an invoice for an order are also available in the class. Under certain factors, the system has a *method for updating an order* that displays its current status. When creating an invoice, the method returns *an invoice object* that provides data about the creation and the order amount. Since the designed system is divided as much as possible into modules, each performs certain subtasks – each of them entails several others, so only one of them was demonstrated for a general idea of the modularity principle used.

The sequence diagram shows the interaction of objects in dynamics. The main elements of sequence diagrams are objects, which are logical entities representing individual elements of the system and the messages they exchange. Actors can also act as objects. Notes can be not only abstract actions performed but also class methods created in the class diagram. The messages in the sequence diagram are numbered, i.e. they have a precise sequence. A sequence diagram refers to UML interaction diagrams that describe the behavioural aspects of a system but consider the interaction of objects over time. In other words, the sequence diagram shows the time features of transmitting and receiving messages by objects. In the sequence diagram, the object is displayed as a rectangle on top of a dashed vertical line. This vertical line is called the object's lifeline. It is a fragment of the object's life cycle during the interaction. To represent the chart, the sequence from order selection to payment was selected (Fig. 9). Figure 9 shows the series of performing system actions when the customer initiates a search and then pays for the product. As can be seen from the diagram, the customer first searches for elements that will satisfy them while accessing the program interface. For the interface to make this request, it needs to access the database that contains a list of available products. The next step is to select a specific item from the list. After that, the customer initiates the payment stage that was shown in the diagrams above. After sending a request to start the payment process, the system accesses the payment interface. The payment interface should first receive the selected item by accessing the search and selection interface, which will return the result. Next, you need to get the details of the selected item, its price, tax, and other characteristics to pay. After some work, the payment interface delivers the selected item to the customer and shows all the payment details for this product. The next step is to send a request to the payment object from the system. After the following check for the accuracy of entering and calculating all data, a transaction is created to the payment system that serves the system. And if the payment is successful, the result is sent to the payment interface, which notifies you of a successful transaction. To receive a bank statement, the system also creates a receipt for payment completion and sends the result to the email address registered by the customer.

The state diagram displays a finite automaton in the form of a graph, the vertices of the states of the object whose behaviour is modelled. The transitions are events that move the object under consideration from one state to another. In this case, it is assumed that the time spent by an object in a specific state is much longer than the time required to move from one state to another. That is, transitions between

states are made instantly. The state diagram is used to describe the states of an object and the transition conditions between them. Thus, the state description allows you to accurately describe the behaviour model of an object when receiving various messages and interacting with other objects. Two-state diagrams were shown to provide an example of customer and store behaviour (Fig. 10, 11).

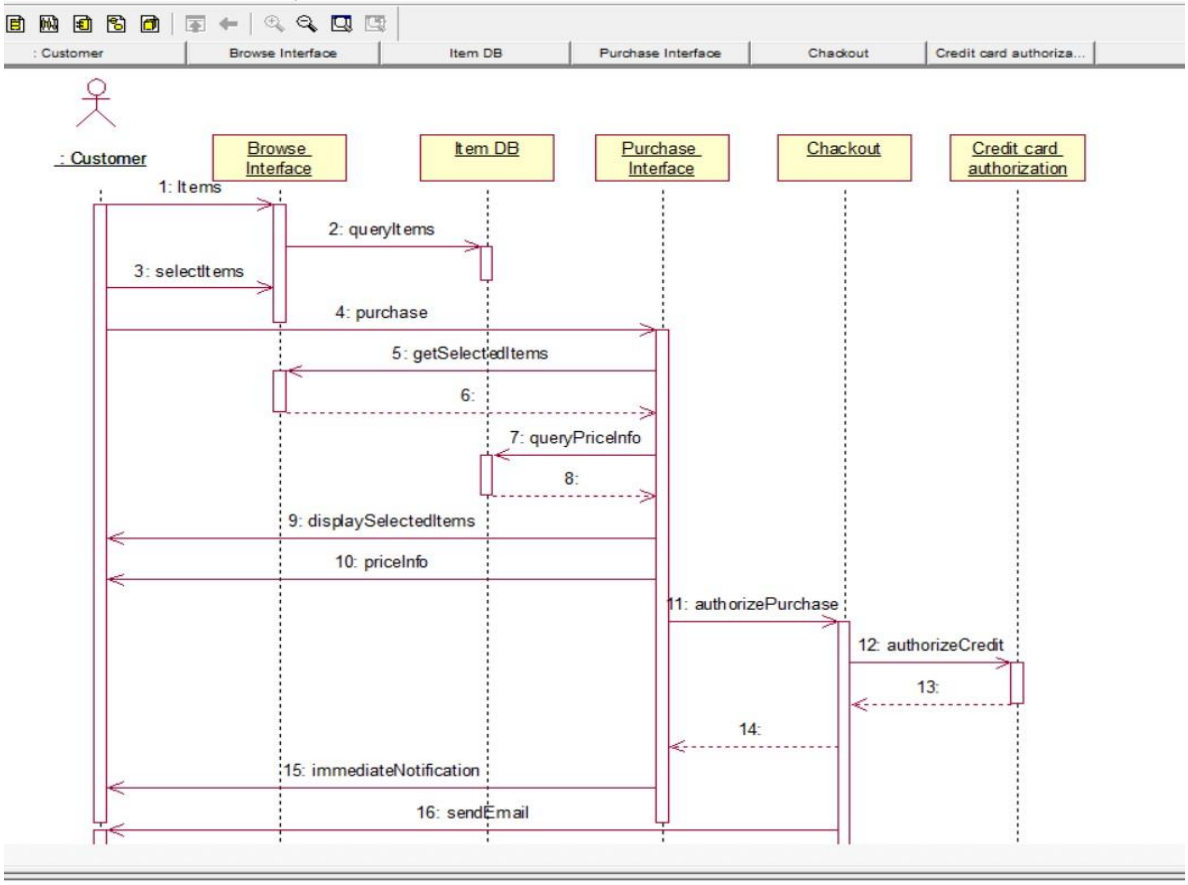


Figure 9: Sequence diagram

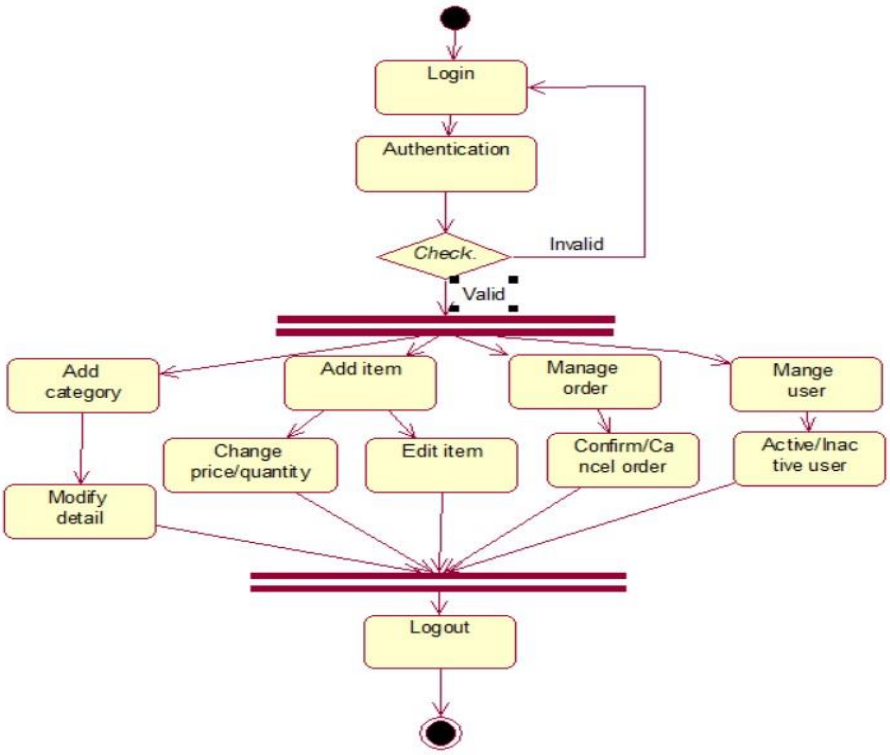


Figure 10: State diagram from the store side

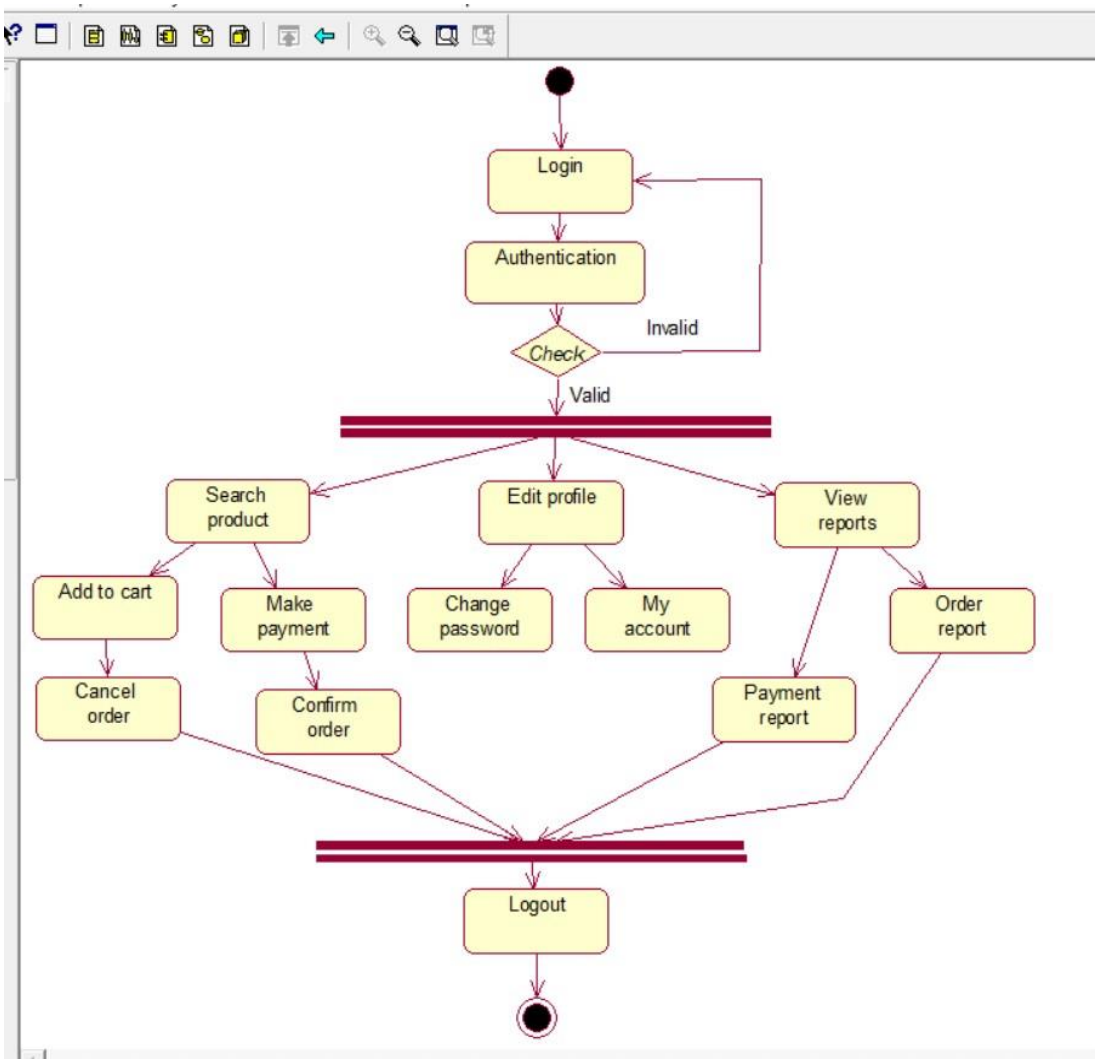


Figure 11: State diagram on the customer's part

As you can see from the diagrams below, both objects start with the authentication process and check for the accuracy of the entered data. If the operation is not entered correctly, you need to repeat the procedure. When entered correctly in an entity, the store is used to create, edit, or add specific product categories and view statistics on products and users of this outlet. The customer can purchase and pay for products, add them to the shopping cart, edit and view their account, and receive receipts and reports on their purchases. All these processes are completed by exiting the application. An activity diagram (activity diagram) allows you to model sequences of business processes or actions implemented by class methods. These sequences can represent alternative branches of the data processing process or branches that can be executed in parallel. Activity diagrams are analogous to a flowchart of any algorithm. Like state and transition diagrams, they are displayed as an oriented graph, which is actions, and the edges are transitions between steps. It is advisable to use activity diagrams for analysis:

- Content of scenarios for applying the projected system;
- Interaction of work flows in different methods;
- Executing scripts in multiprocessor computing environments.

These diagrams are widely used to describe behaviour that involves a large number of parallel processes. Since this system supports authentication from third-party accounts such as Google or Facebook, it is advisable to draw one of them on the activity diagram (Fig. 12).

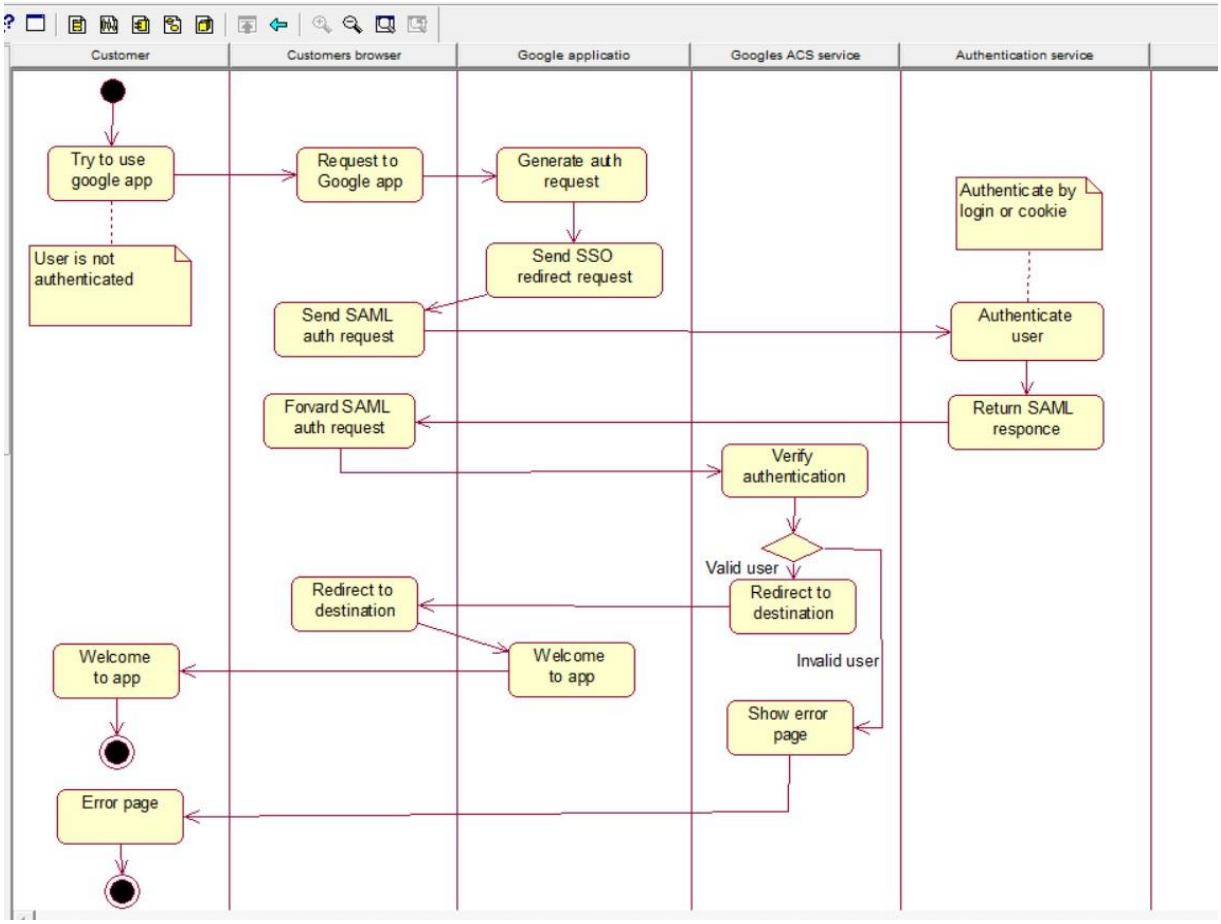


Figure 12: Activity diagram

The first thing you need to do is make sure that you have a Google account. Next, a window is generated to confirm the request for using your data and sent for verification to the Google service. After the first verification, this is usually a password and log in. The user should be notified about this on the app screen. At the same time, a verification response is sent, and if everything is correct, the system receives a token or cookie, which is then sent for secondary processing by the Gul servers. Upon successful operation, a token or cookie signed with a digital label is returned to the system, then sent to the service to check all labels and entered data – this is the last stage of validation. If everything is correct and such a user already exists, a response to the request to the system is returned. Since Google uses its image models, the result is returned as a ready-made screen showing the successful completion of all validation stages in the system under development. And if an error is detected, a window with the code and description of the error is returned.

At first glance, everything looks as simple, fast and affordable as possible for the user. Still, upon a detailed inspection of this process, it turned out that for such a simple operation, you need to pass a relatively large number of checks. It is all done for the security and privacy of each user of this service.

The diagrams that were considered earlier reflected the conceptual and logical aspects of building the system. The peculiarity of a logical representation is that it operates with abstract concepts. And this only reflects an understanding of the static structure of the system or dynamic aspects of behaviour.

To create a physical system, it is necessary to implement all logical representations in certain material entities. In the context of UML, this means a set of related physical commodities, including software and hardware, as well as personnel that are organized to perform specific assigned tasks.

Unlike the diagrams discussed above, the component diagram describes the features of the physical representation of the system. In addition, the component diagram allows you to determine the system's architecture under development by establishing dependencies between software components that can be the source, binary, and executable code. For example, a diagram of the elements of mobile applications of the system under development was presented (Fig. 13).

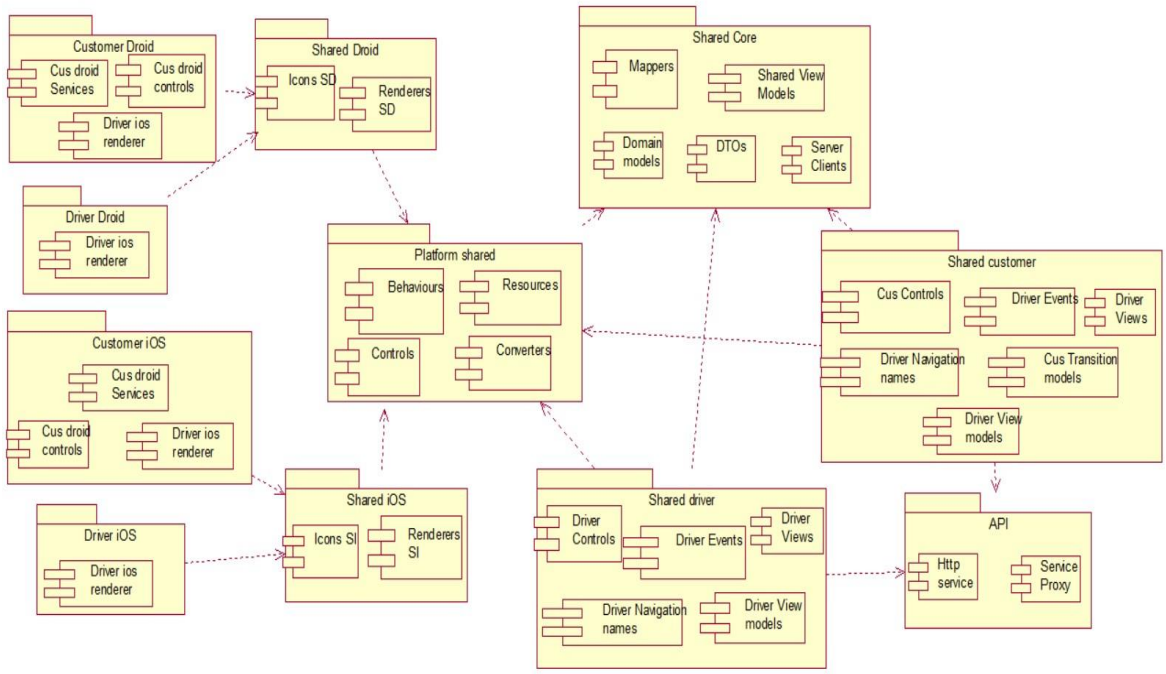


Figure 13: Component diagram

The diagram shows that even mobile applications have a structure that is maximally divided into components. The architecture is designed so that when an application is compiled to an executable file for iOS or Android, all parts are overused. Thus, it allows you to reduce the total amount of code, improve the extension and addition of the application with new functionality, and improve readability.

Now let's take a closer look at each package and the entities in between:

- Customer iOS, customer droid, driver iOS, Driver droid – the main packages for applications for the iOS or Android platform, respectively. These packages contain the native resources themselves, controllers, so-called renderers, and services;
- Shared droid, shared iOS - these packages also include a native implementation but can be re-used in two separate apps. For a general example, you can mention input fields – they are used in both applications, but the platform implementation is different;
- Platform shared - this package contains components that can be used regardless of the platform or project it depends on the shared driver, Shared customer packages that already have elements that are dependent on the project;
- Shared core - contains all project models. you need it for convenient support and search for them;
- Shared driver, Shared customer - because the projects are not identical, so all the visual aspects of the program and classes that relate only to a specific module are in these packages;
- The API package is designed for the communication of components with the server. Since both mobile applications share the server, it was decided to put this logical unit separately.

The deployment diagram represents the overall configuration and topology of a distributed software system and contains an image of the location of components on individual system nodes. In addition, the deployment diagram shows the presence of physical connections – routes for transmitting information between hardware devices involved in the system implementation.

The purpose of the deployment diagram is to visualize the components and elements of the application that exist only at the stage of its execution. In this case, only those components of the program that are executable files or dynamic libraries are represented. Details that are not used at the execution stage will not be shown in the deployment diagram. For example, components with program source texts can only be present on the component diagram.

After presenting the component diagram, it is advisable to build a deployment diagram (Fig. 14).

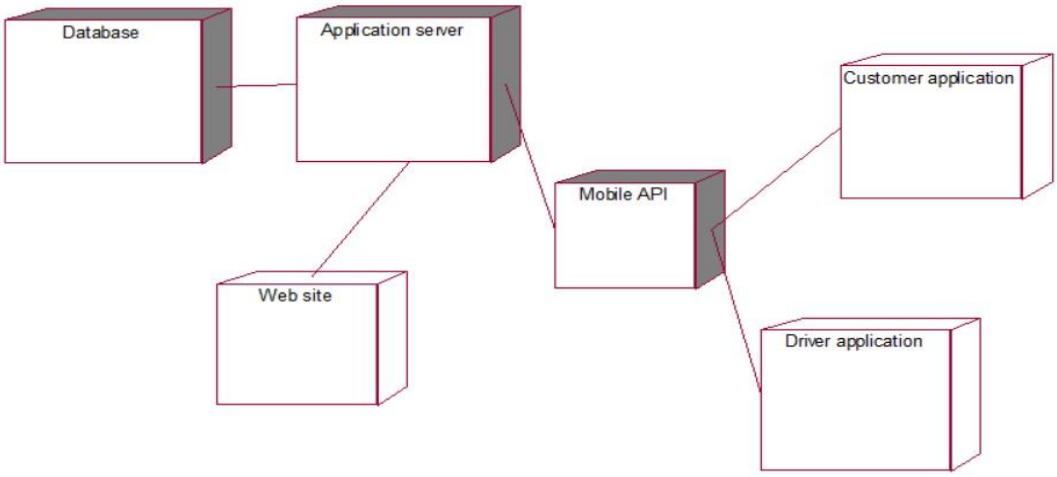


Figure 14: Deployment diagram

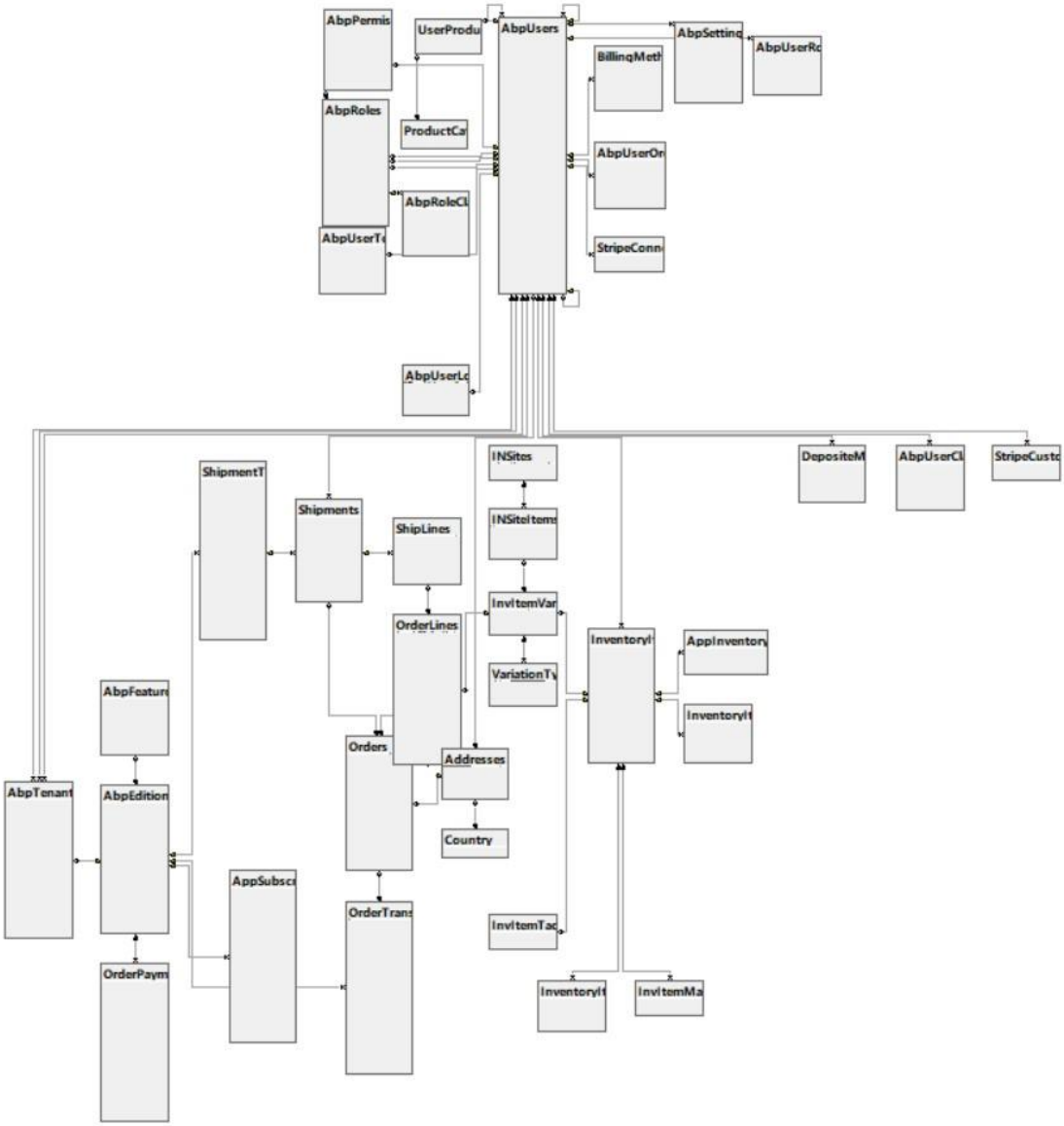


Figure 15: Complete database structure

This diagram shows the main elements of the system, namely:

- Database - which contains all the data that was entered or processed within the system;
- Central logical server - operates data from the website and that comes from the auxiliary server for the mobile application, also reads, updates, and writes all data in the database;
- Mobile server - processes data that comes from mobile devices and sends it to the main server for processing;
- Site – used for administrative purposes and accessed only by sales representatives;
- Mobile client for the driver - provided to an entity that will act as a courier and deliver goods from retail outlets;
- Mobile client for customers - available to anyone who wants to use this system to make their orders.

Today, all software products require a database since it is not a negative part when processing and storing data. However, concluding the analysed approaches to software development – the database is not recommended but is mandatory in a system of any size. A vast database was designed with many dependencies between tables for this system, which is pretty detailed and divided into logical sections (Fig. 15), so a small part will be submitted for analysis (Fig. 16). As you can see from the figure, the database is quite extensive, and this indicates that many different aspects of the system behaviour have been thought out. Various nuances that will be required for regular use of the system have been taken into account.

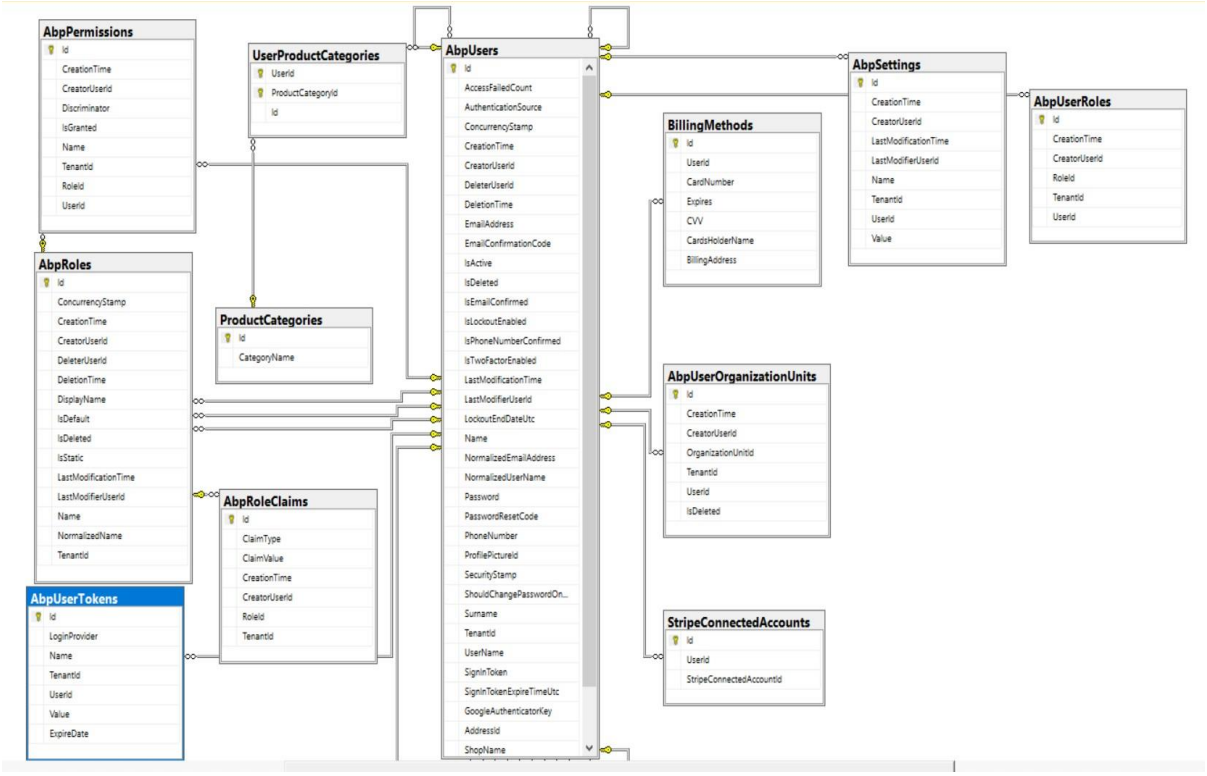


Figure 16: Fragment for database analysis

The AbpUsers table is the largest and contains all the information about the following generalized entity as a system user. The named user means both the driver, buyer, and seller. This table is linked to almost all tables in the database to be considered the main one. The AbpPermissions table contains information about available benefits in accessing the system, the date and time of granting concessions, and communication with the main table for communicating with a specific user. The UserProductCategories table contains a unique identifier for the selected category and is supplemented with the following table called ProductCategories. In turn, ProductCategories includes the category name and also the identifier. The AbpRoles table is linked to the benefits table and contains basic information about accesses and the time of their implementation.

AbpUserTokens table - contains information about the session token. When the token was created, to whom it was granted, the token itself, and the expiration date. AbpSettings table - contains a list of available settings for the user. BillingMethods table - stores data about the user's payment transactions.

AbpUserRoles table – is an extension of the settings table and contains different access rights for each entity. AbpUserOrganizationUnits table - includes data about the organization and is available for users with the "seller" access right. StripeConnectedAccounts table - provides information on accounts that were linked to a payment system called Stripe.

In this section, a systematic analysis of the decision support system for generating and implementing orders based on cross-programming and cloud computing was carried out, namely:

- Built a goal tree;
- A diagram of use cases has been constructed;
- A class diagram is created;
- A sequence diagram is made;
- A state diagram is built;
- An activity diagram is produced;
- Building a component diagram;
- Created a deployment diagram;
- Made a database diagram.

When familiar with the theory of system analysis, it can be argued that it plays a leading role at all stages from the moment of planning to the finished product when developing a software product. These diagrams will significantly simplify the understanding and support of any software product. Based on a large number of articles about building the correct system architecture, we can draw an obvious conclusion: when developing systems of various levels of complexity, designers often omit the moment of constructing diagrams, which in the future, when the system is expanded, results in a massive conflict of "interests". Because no matter how experienced an architect or programmer is, it won't be easy to understand the train of thought of another person who has not left behind proper documentation, diagrams, comments, etc. That is why when organizing and creating a system, you should pay considerable attention to designing different types of diagrams.

3.3. Methods/approaches selection and justification to solving a problem

The choice of methods and approaches is one of the most challenging links when creating any project. Conventionally, the stages can be divided into:

- Identifying the strategy: requirements, evaluating the implementation of requirements, calculating the budget and the possibility of performance;
- Analysis: research of functions identified at the strategy stage, attributes, and relationships;
- Design - collecting all data models;
- Implementation: product development;
- Testing – this stage can and should preferably be started in parallel with the implementation stage. At this stage, the product is tested to see if the product meets the requirements;
- Implementation the product is implemented in small iterations. This facilitates testing and allows you to use an incomplete system that will be gradually updated.
- Usage and technical support after achieving the goal, you should take care of constant maintenance and improvement of the system being developed.

When developing this system, all the stages before implementation were completed. Since the system is very voluminous, it is updated iteratively after testing the developed functionally.

As for the algorithms used, basically, all the algorithms were implemented out of the box. That is, they were already ready for use when installing and configuring the environment. Attention should be paid to one interesting algorithm that was used – GPS triangulation, which Google, namely GoogleMaps, implemented. Although the technology is implemented on the most modern technologies, its principle is as simple as possible. There are three components in any navigation system:

- Satellites: rotate around the Earth and transmit their current time and position in orbit;
- Command centre: transmits orbital data, time corrections, and satellite orbital position;

- GPS receiver: on Earth, it receives all corrected orbital data from as many satellites as possible and calculates its position on Earth (at least four satellites for correct operation).

This technology is used to identify both the customer and the store and track the courier's movement online.

3.4. Description of the selected development software tools

After performing system analysis, you should choose software tools for implementing a decision support system forming and implementing orders based on cross programming and cloud computing. The following IDE was set [41-60]:

SQL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure, from SQL Server to Azure SQL Database. SSMS provides tools for configuring, monitoring, and administering databases. You can also use the program to connect to a remote database and, most importantly, to deploy, monitor, and update all components using scripts and queries. This environment has convenient integration with other Microsoft products and creates an analytical chart of an already built database for a more convenient understanding of the structure. There is also a user-friendly and beautiful interface for creating queries and macros and adaptive input when creating a question, which significantly speeds up writing.

Microsoft Visual Studio is a powerful tool for building engineering applications for any system, starting from small controllers and ending with projects that are used by the whole world. Of course, this is all because the environment supports the compilation of the most common programming languages. Visual Studio includes a code editor that supports IntelliSense, as well as code refactoring. The built-in debugger works both as a debugger at the source level and as a debugger at the machine level. Other built-in tools include a form designer for building GUI applications, a web designer, a class designer, and a database schema designer. It allows you to add plugins that extend functionality at almost every level-including adding support for source code management systems (e.g. Subversion and Visual SourceSafe) to add new toolkits such as editors and visual designers for domain-specific languages or toolkits for other aspects of the software development lifecycle (e.g. Team Foundation Server client: Team Explorer).

ASP .NET Zero is a relatively new tool for developing server applications. It provides reliable and scalable architectural solutions and many additional features, such as user management, which is very suitable for solving the goal of a given system, managing user profiles, localization, configuration, and much more. It also facilitates daily development work by using the complete web application framework. ASP.NET Boilerplate. The solution is built in compliance with the SOLID pattern and uses ASP.NET Core, Entity Framework, Angular and others. What is an equally important feature is that the entire structure code is editable.

Xamarin Forms is a framework for building mobile applications and creating user interfaces. This framework is a cross-platform solution that allows you to develop mobile applications for Android, iOS, and Windows Phone. Xamarin forms are built so that each interface element for each of the platforms is written in the same language called XAML with an attached wrapper class in C#. A unique feature of this framework is that all written code is compiled into native controls and at the same time allows you to customize native elements and adapt them to each of the platforms.

Visual Studio Code is a lightweight yet powerful source code editor that runs on Linux, OS X, and Windows systems. It comes with built-in support for JavaScript, TypeScript, and Node.js and has excellent support for expanding dictionaries of various programming languages.

Angular is a platform and framework for building client applications using HTML and TypeScript. Angular is written in TypeScript and implements the basic concept in a set of libraries that you need to import into your project. Architecture is built on certain fundamental ideas. The main blocks are Angular components, which are organized in NgModules. NgModules, in turn, contains the corresponding functional sets of code.

Microsoft Azure – is a cloud computing service for creating, testing, deploying, and managing applications through data centres. It provides software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) and supports many different programming languages, tools, and frameworks, including both Microsoft-specific and complementary software and systems.

Node.js is a server platform built on Google's JavaScript Engine (V8 Engine)

Chrome. The platform makes it easy to create fast and scalable network applications. Node.js uses an event-oriented non-blocking I/O model, making it simple and efficient. It is ideal for applications that work in real-time and Exchange large data packets on various independent devices. Node.js is an open-source, cross-platform solution for developing server and network applications. It uses the JavaScript writing language and can be compiled and run on OS X, Windows, and Linux operating systems.

After making analytical decisions about the development of the application, the **C# language** was chosen. Since it is pretty flexible to use and has intuitive syntax, both the server and mobile applications are implemented in this language. In the case of a server using ASP .NET Zero, this significantly speeds up application development due to a large amount of functionality implemented. When using it, a significant disadvantage of this tool was revealed, despite all the advantages, the structure is built as fully equipped as possible, and each module depends on each other so much that when removing unnecessary functionality, you need to understand all the dependencies so that the system does not fail. That is why when choosing this tool, you should carefully re-read its capabilities and analyse all its functionality in detail so that in the future, you do not have to do double work. The same programming language was used to develop mobile applications but using the Xamarin forms framework. Thanks to the maximum flexibility in application implementation, when designing, it made it possible to create the architecture to re-use many components and other logical units between platforms and, just as importantly, between projects. Of the disadvantages, we can only note the speed of the application since the shared code is recompiled 2-3 times (depending on the platform) by shells between the shared and native code. To speed up the application, you should choose either a more native approach or optimize existing components and solutions into more native ones, which will slow down development.

MySQL, SQL-Server, SQL Management Studio as database management software, and the database administration and Management Shell were used for database design and development.

Thanks to the convenience of using AWS services, it was possible to quickly deploy the application to the server and connect additional modules that perform cloud computing. For example, you can specify tracking the current location of the courier and transmitting information to the user.

After analysing the above technologies, we can conclude the technologies used, namely Xamarin forms – a convenient cross-platform solution for creating mobile applications with the ability to customize at the native development level. ASP .NET Zero is a fairly powerful tool for developing the server-side, but due to its power and volume, it is not convenient for small projects since it is difficult to customize but easy to complete.

4. Experiments, results and discussion

The decision support system for generating and implementing orders based on cross-programming and cloud computing was developed using such programming languages as C#, JavaScript and using: ASP .NET Zero; Xamarin forms; Angular, NodeJs technologies. For the software product to work correctly, you need a mobile device with Android above version 8 and iOS above version 12.1, respectively. All mobile devices that support these operating systems can efficiently work with the app. The app on a mobile device takes 73MB for the customer and 76MB for the courier. Any modern browser is suitable for the web version. In the structure, the project consists of 3 different design solutions based situational awareness technology [61-70] and multi-agent development [71-78], which in turn contain subprojects for data integration, management and support in e-commerce[79-86].

Mobile project structure. The mobile application project is divided into 11 subprojects, of which six relate directly to the courier program or the client's program. At the same time, the remaining five remain common to both of them (Fig. 17). As you can see from the image above, the project consists of many folders and files, which have objects and methods of written code. And with the shared logic allocation approach, this is still a relatively small number of files.

Server project structure. Since the server is the primary mechanism for data exchange and processing, it is pretty voluminous in design (Fig. 18). The "src" folder contains the main solution designs, and the "test" folder contains unit tests of all the main solution classes. The application project contains 30 folders that contain logically linked classes.

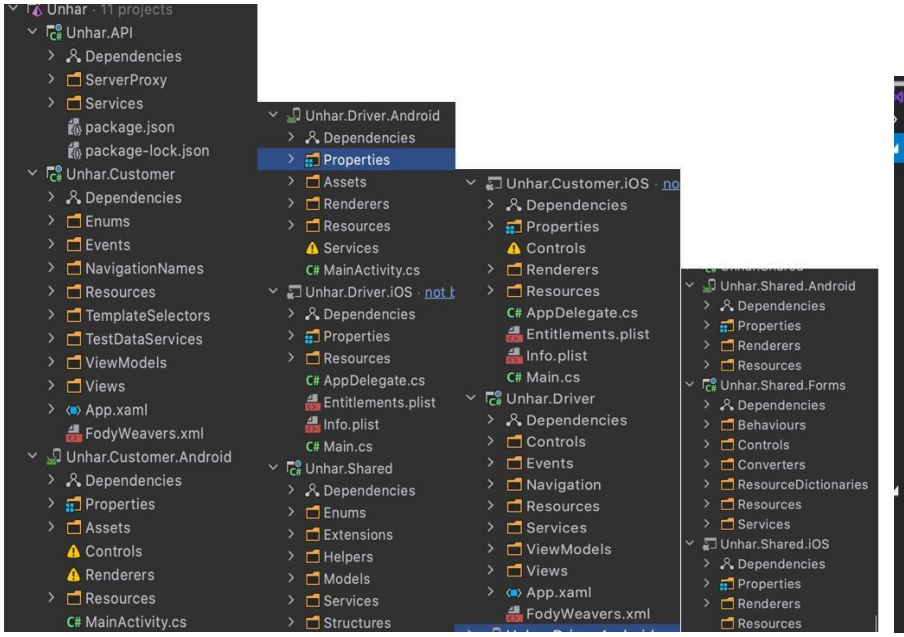


Figure 17: Mobile project structure

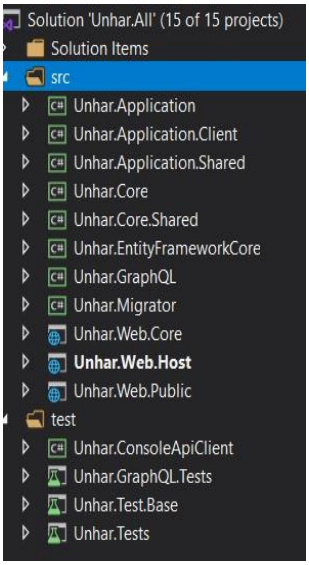


Figure 18: Server project structure

The Application Project. The client contains seven folders that belong to the server client. Application.Shared contains 24 folders, which are mostly repeated with the project application names since it is a project wrapper. Core - has 30 folders, this project is a wrapper for the entire solution, and its classes are used throughout the solution. Core.Shared – this project depends on the previous one and allows you to supplement the key with new functionality – currently contains nine folders. EntityFrameworkCore -contains three folders that collect all information about the database structure. The project is used to save the database configuration and allows you to restore it in case of loss entirely. GraphQL - contains seven folders. You need the project to record all actions with the database easily. Migrator - contains one folder and four files that are used to work with the EntityFrameworkCore project. Web.Core, Web.Host, Web.Public – used for configuring connected Bibliotecas, dependencies, and basic mechanisms for communicating with external entities.

Web project structure. It remains to demonstrate the design of the web project (Fig. 19).

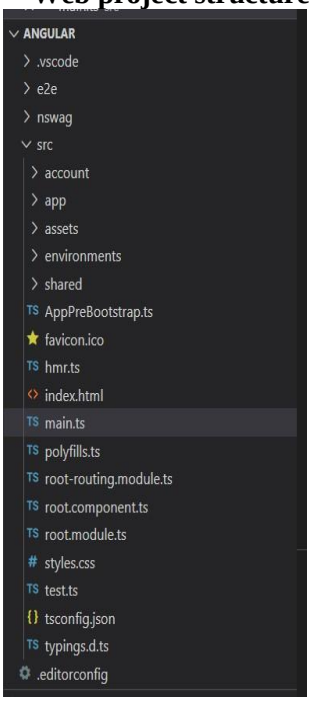


Figure 19: Web project structure

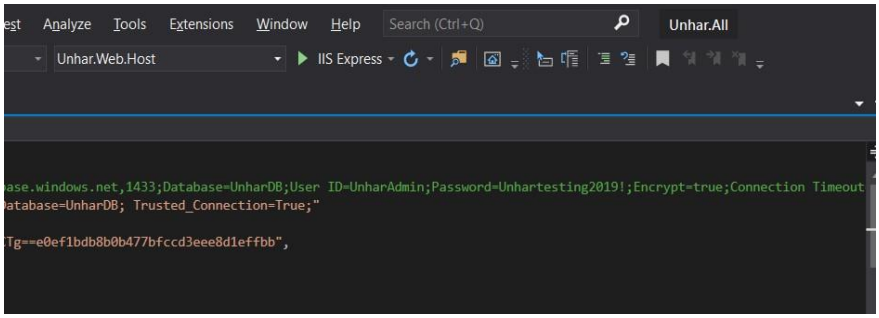


Figure 20: Compile button

The project is not as voluminous as the previous two and contains only a few main directories. The first two are auxiliary directories for deploying the environment. Folder “nswag” - includes a configuration for the visual representation of all controllers that are contained in the project. Folder “src” - contains the main directories used in the development and several project configuration files.

To launch the website, follow the link https://wardrobe.com but at the moment, the server only works locally. To launch the app on your mobile device, go to AppStore or GooglePlay and download the app. Since the free download of the test product in stores is not supported, you can only use the app when compiling the app directly from the development environment. You can download all the project resources here https://github.com/flameci/Wardrobe from the GitHub repository. After downloading all the necessary resources, click the compile button in Visual Studio (Fig. 20). The project compilation button looks like a green triangle. After clicking on the button, you can start launching the web version. The first thing you need to do is check for the npm module to run the web version. To do this, open the console and enter the word npm if you see an output similar to a large amount of text (Fig. 21). It will indicate that the web application is ready to launch. In the worst-case scenario, the system will report that it does not know about such a command, and you will need to follow the link https://nodejs.org/en/ and download this module and install it on your computer. After successful installation, go to the root directory of the downloaded repository and open the console. In the console, enter “npm run start” and wait for the compilation of the web program to complete (Fig. 22). After that, you can navigate to the local host address and port http://localhost:4200/ and see the running site. To compile the app to your phone, you need to connect your phone to your computer and select the project you want to download. After compilation is complete, the app will be downloaded to your device.

```
C:\Users\fedor>npm

Usage: npm <command>

where <command> is one of:
    access, adduser, audit, bin, bugs, c, cache, ci, cit,
    clean-install, clean-install-test, completion, config,
    create, ddp, dedupe, deprecate, dist-tag, docs, doctor,
    edit, explore, fund, get, help, help-search, hook, i, init,
    install, install-ci-test, install-test, it, link, list, ln,
    login, logout, ls, org, outdated, owner, pack, ping, prefix,
    profile, prune, publish, rb, rebuild, repo, restart, root,
    run, run-script, s, se, search, set, shrinkwrap, star,
    stars, start, stop, t, team, test, token, tst, un,
    uninstall, unpublish, unstar, up, update, v, version, view,
    whoami

npm <command> -h  quick help on <command>
npm -l            display full usage info
npm help <term>   search for help on <term>
npm help npm      involved overview

Specify configs in the ini-formatted file:
    C:\Users\fedor\.npmrc
or on the command line via: npm <command> --key value
Config info can be viewed via: npm help config
```

Figure 21: Output of the npm command

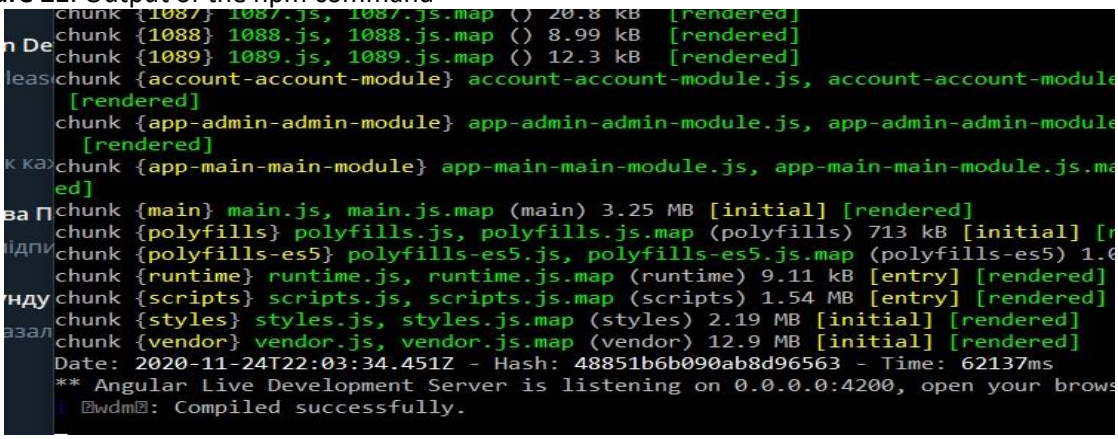


Figure 22: Output of successful compilation.

Input data for the decision support system is data from three resources: a website and two mobile applications. The source data is the database and information used in applications. After installing and configuring all system components, go to the website specified to see the login page in front of you (Fig. 23). As you can see from the figure, you need to enter your email address and password if it was created earlier. After authentication, you will see the main page with the total sales and the navigation bar (Fig. 24). To add a new product to the collection, go to the Listings page in the navigation menu and create a new one (Fig. 25).

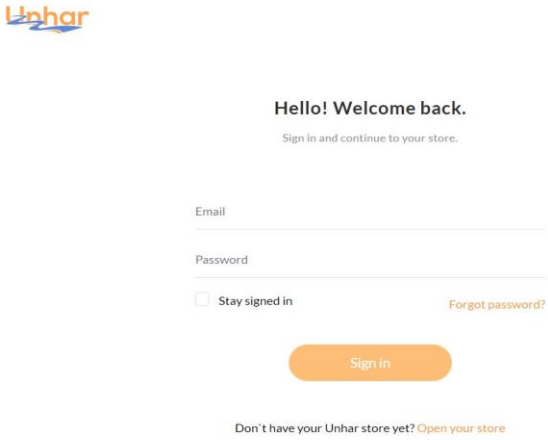


Figure 23: Login page

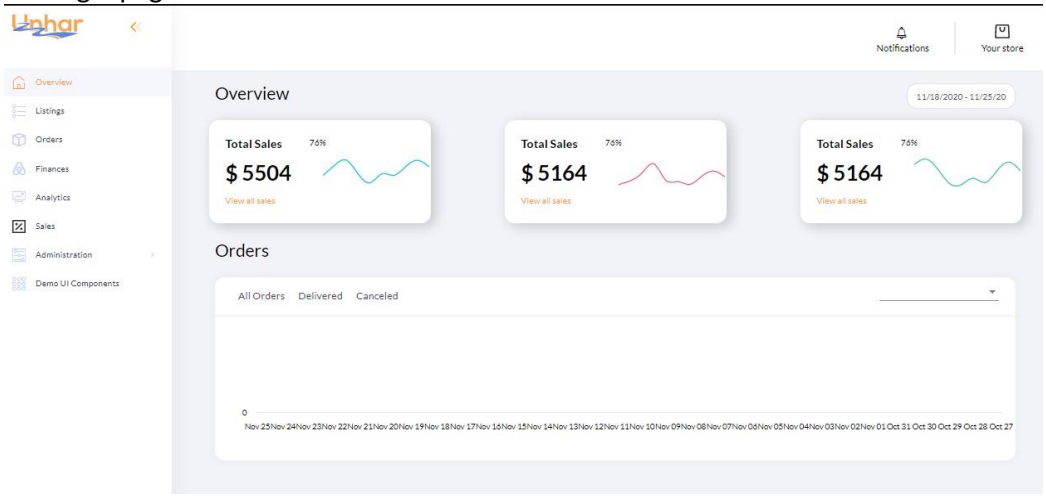


Figure 24: Home page

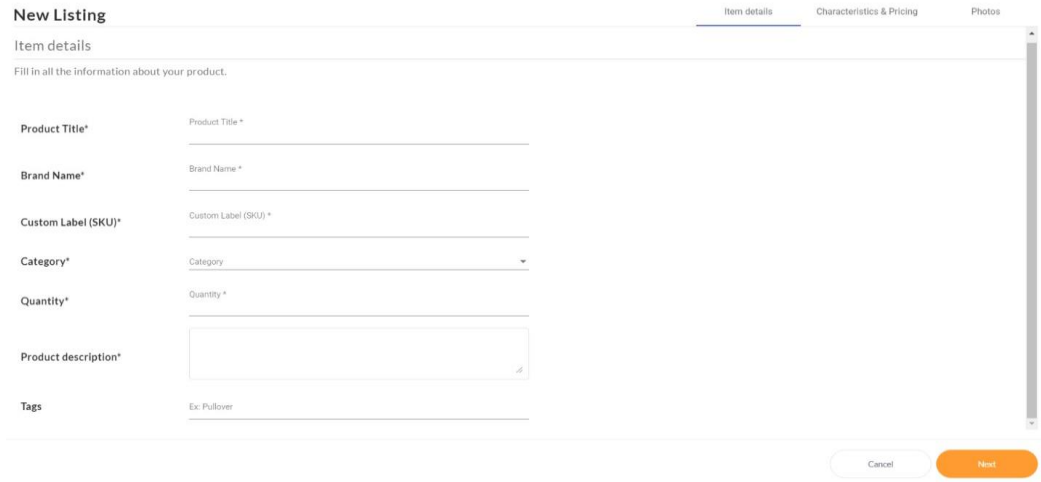


Figure 25: Creating a new product

On this page, enter the name, brand, product category, quantity, and description. After continuing, go to the page with the price and characteristics (Fig. 26). On this page, you will need to enter the product price, weight, dimensions, delivery time, product material, and the ability to return the product. And the last page-uploading product photos (Fig. 27). After completing all the stages of filling in the product details, you will have the product added to your store's collection, and in the future, you will be able to see them in the mobile app for the customer. When you open the app for the customer, you will be taken to the main page, where you will find promotional offers and several top products from each category (Fig. 28). You can also immediately see the offer price, go to the shopping cart, or open the navigation bar (Fig. 29). When you open the side menu, you can see many available options to open the list of available categories

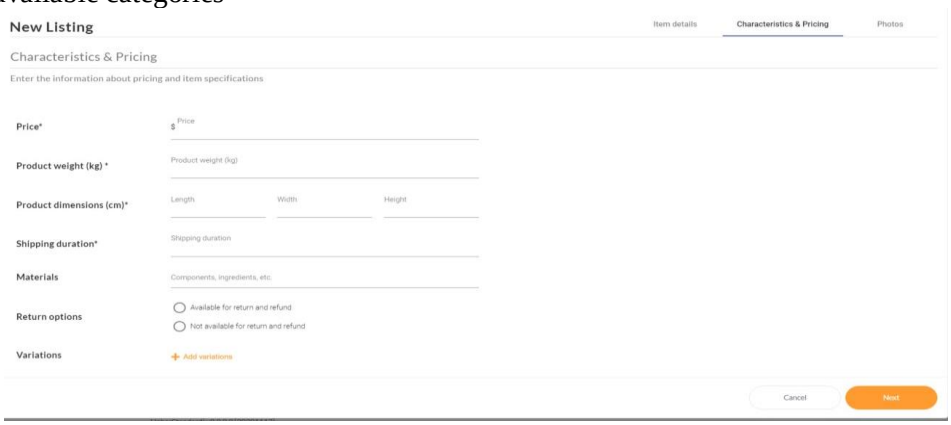


Figure 26: Creating a new product

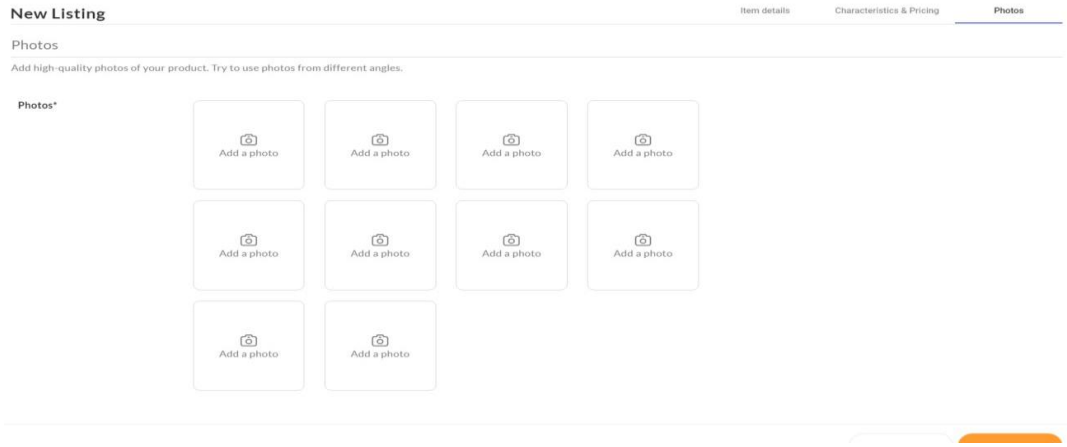


Figure 27: Creating a new product

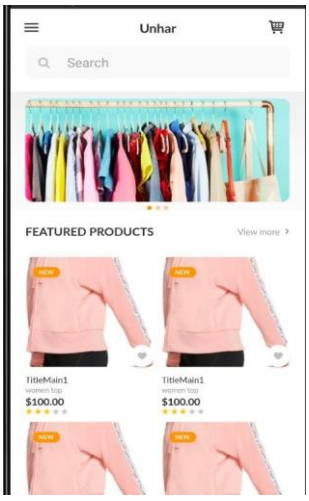


Figure 28: Home page of customer's mobile app

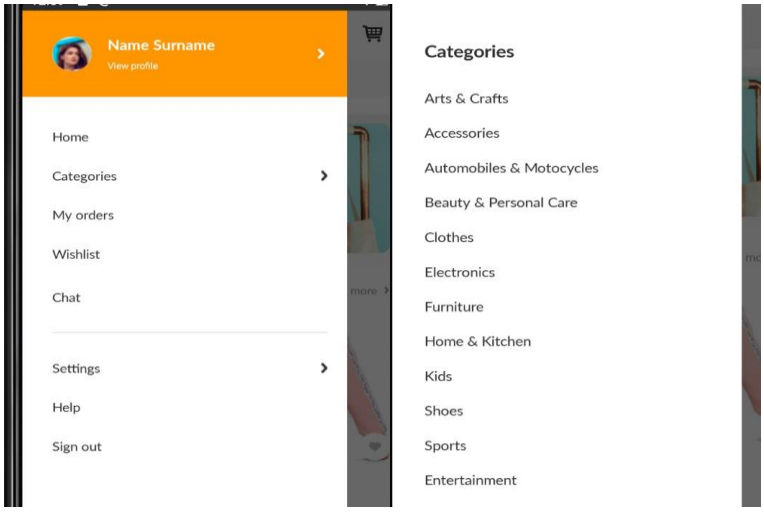


Figure 29: Side app menu for the customer

When you click on one of the categories, a page will open with a list of products of a specific category (Fig. 30). If you want to view the entire list by subcategory, click the “view more” button. When you select a specific item at any stage, you can view the details of the desired product (Fig. 31).

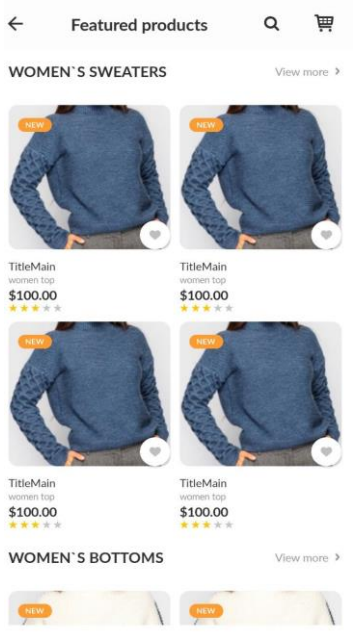


Figure 30: Selected app category

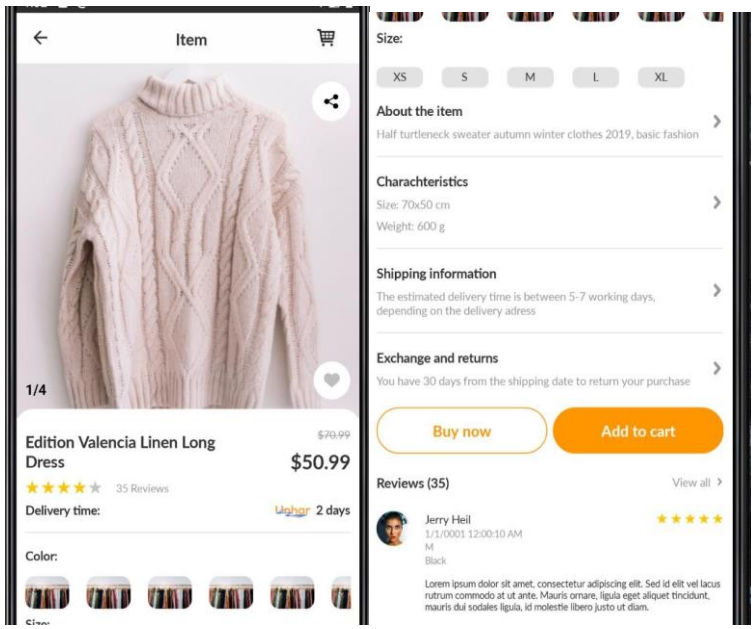


Figure 31: Product details

As you can see from the figure, all the characteristics that were entered from the website are shown in detail, and you can also view comments about the product, buy it, or add it to your shopping cart.

When the customer confirms the payment, it becomes active and appears in the courier app. After authenticating the courier in the app, the first page is a map with active orders (Fig. 32). You can scale the map as you like, and the data will be dynamically loaded. When you click on a point on the map, brief details about the location appear, such as the store name and street. If it is not convenient for the courier to scroll through the map, they can swipe up on the bottom element to expand the list of active products, and all of them are sorted by distance to the current location of the courier (Fig. 33).

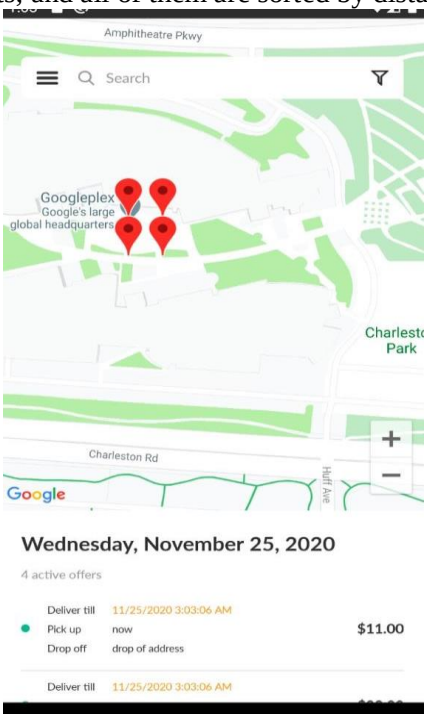


Figure 32: Home page of the courier app

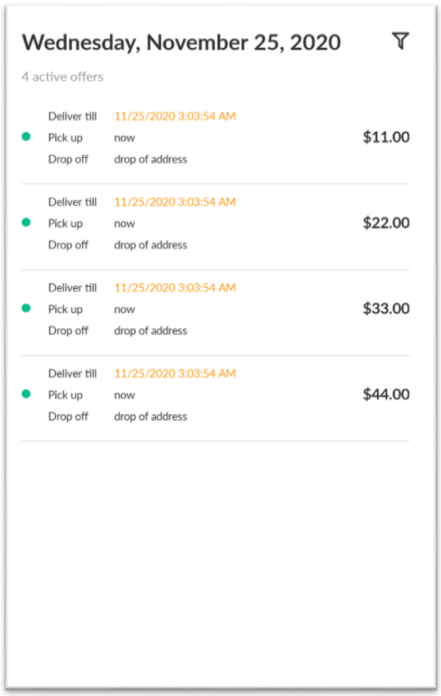
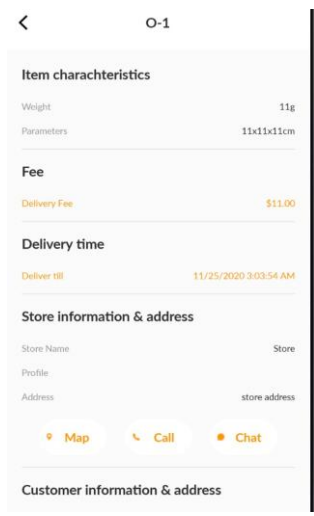
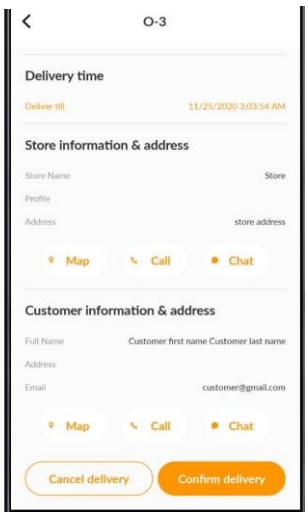
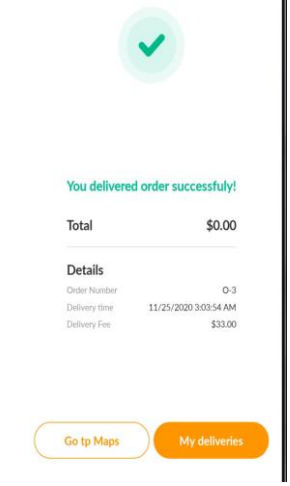


Figure 33: Page with active orders of the courier app

On this page, you can see the number of active deliveries, the time to when you need to deliver, how urgent the order is, delivery addresses, and the price. You can also filter by date and price. When you click on an order that satisfies the courier, a page opens with the details of the selected product (Fig. 34). The person who delivers the product can view how much they will earn for this order. You can also contact the store or create a chat with the supplier (store). The same features are available in connection with the customer of the product. In case of successful delivery, click on the delivery confirmation button. A screen will appear about the successful delivery of the product with the order details and its cost (Fig. 35).







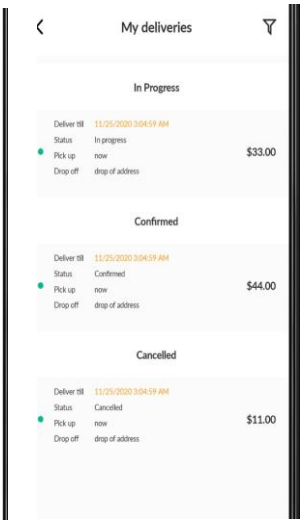


Figure 34: Order details **Figure 35**: Successful order delivery page **Figure 36**: Page with orders

If the courier wants to continue delivering another order, they can return to the map and start the whole process all over again. Otherwise, it can view all its deliveries filtered by status: active, cancelled, and delivered (Fig. 36). From this page, it sees the same abbreviated information as on the map. You can also filter by period, i.e. view orders for a week, month, or year and purely by a specific status.

To perform an example analysis, you need to run all systems on the local network. First, you need to load the server on a local host and make it visible on the local web (Fig. 37).

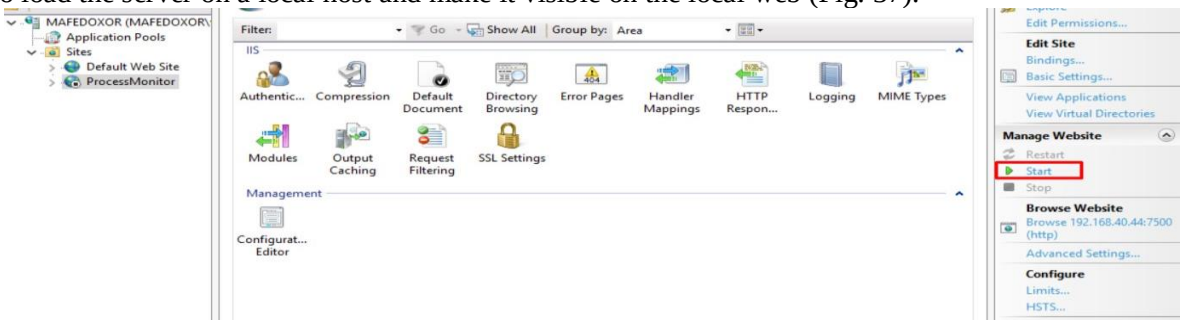


Figure 37: Server host

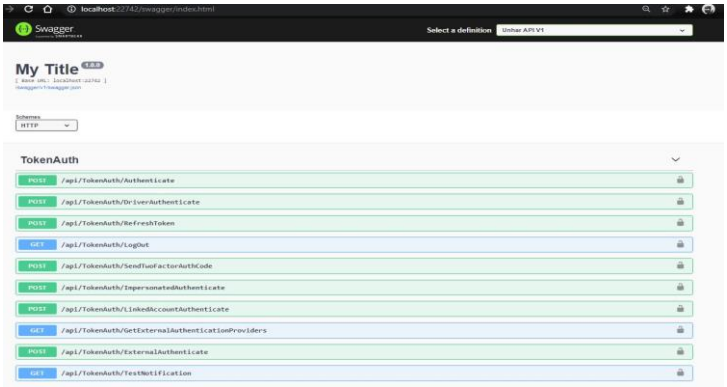


Figure 38: Swagger system

Click on the server start button. Now you can start entering data on the site, namely, creating order and filling in all the necessary fields (Fig. 37). To make sure that the server is working, you can enter the address of the local server and the open port in the browser and see the operation of the already connected monitoring system for existing controllers (Fig. 4.38). The server is working correctly. You can enter your username and password (Fig. 39).

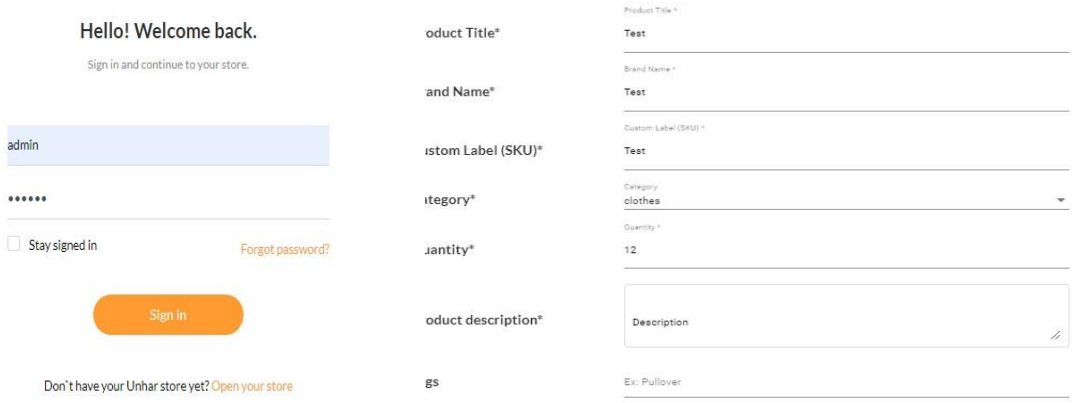
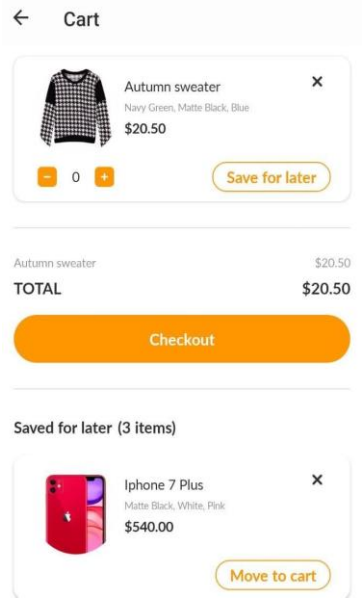
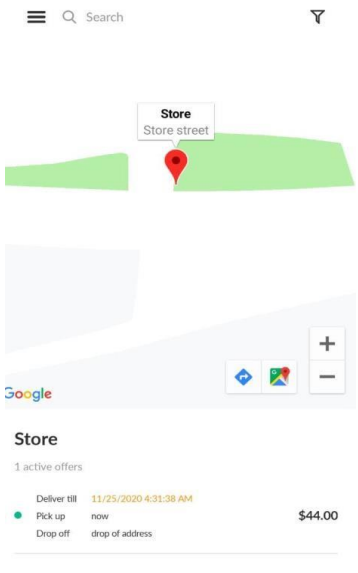


Figure 39: Login on the website and filling in required fields

After creating a product in the category, open the mobile app to purchase it. But, first, add your favourite products to the shopping cart (Fig. 40).





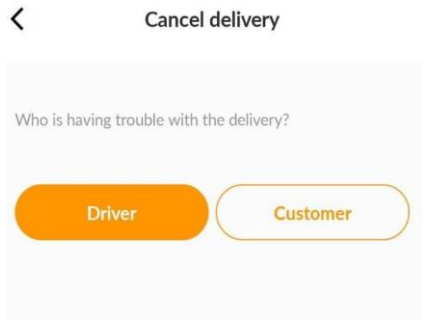


Figure 40: Shopping cart **Figure 41**: Selected store on map **Figure 42**: Selecting the culprit for order

We have selected a sweater. All products that are located below the payment button are not subject to payment. These products are postponed for later. Click the payment button, and since this is a test server, the fee is successful. Next, go to the Courier program and select the test store (Fig. 41).

As you can see from the figure, the name of our store and its test address is available, and just below is the list of orders available in it. Select this order and click cancel the order. The screen displays a selection of the entity that causes the order to be cancelled (Fig. 42). Select the driver as an example and go to the next page for selecting the reason for cancellation (Fig. 43). The figure shows the test data that is stored on the server, so we observe strange reasons. For example, choose the third reason and leave a comment. Click go next, and a screen appears with a warning that if you cancel the order, you will be charged a certain amount (percentage of the order) and if you agree, click pay (Fig. 44). All orders can be viewed in the all deliveries menu. After this step, the whole process is repeated with another order. It should be added that when the order is cancelled, both the customer and the store are

notified. After that, the product must be returned to the store. This system encourages the courier to deliver the order.

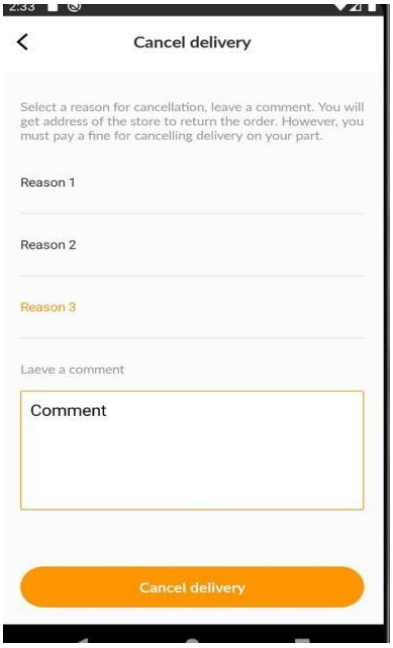


Figure 43: Choosing the reason for cancellation

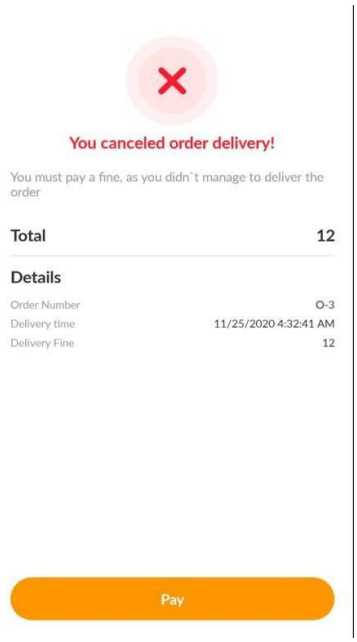


Figure 44: Payment for cancellation

A test example proves that the system performs the specified functionality and works smoothly. Of course, this is only a prototype and not a fully functional ready-made design, so it requires improvements and improvements before releasing a fully finished product to the market. But despite all this, you can see from this section that quite a lot of things are already ready and thought out.

5. Conclusions

This paper analysed, designed, and created a decision support system for generating and implementing orders based on cross-programming and cloud computing. Having carried out an analysis of the Ukrainian market and in the whole world, it can be observed that at the moment, the topic is more relevant than ever about the Ukrainian market – no analogues were found at the date of writing the work. Of course, top stores have their delivery available, but this is not much promoted to the masses. Therefore, this system will allow for additional income for commercial outlets and people who have any vehicle and want a part-time work. When analysing and plotting diagrams, it can be noted that the system is flexible enough to expand and connect tangent spheres to the project. Therefore, there is no need to doubt the prospects and scaling of this system. Also, thanks to the pleasant and intuitive interface, it will be pretty easy for users to get used to the new product.

6. References

[1] M. Korablyov, N. Axak, O. Fomichov, V.Hnidenko, Multi-agent Clinical Decision Support System using Case-Based Reasoning, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1466-1476.

[2] N. Schahovska, Datawarehouse and dataspace information base of decision support system, in: 11th International Conference - The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2011, pp. 170-173.

[3] A. Chiche, Hybrid Decision Support System Framework for Crop Yield Prediction and Recommendation, volume 18(2) of International Journal of Computing, 2019, pp. 181-190.

[4] V. Lytvyn, V. Vysotska, D. Dosyn, O. Lozynska, O. Oborska, Methods of Building Intelligent Decision Support Systems Based on Adaptive Ontology, in: Proceedings of the 2nd International

Conference on Data Stream Mining and Processing, DSMP, 2018, pp. 145-150. DOI: 10.1109/DSMP.2018.8478500
[5] O. Pashchetnyk, V. Lytvyn, V. Zhyvchuk, L. Polishchuk, V. Vysotska, Z. Rybchak, Y. Pukach, The ontological decision support system composition and structure determination for commanders of Land Forces formations and units in Ukrainian Armed Force, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1077-1086.
[6] A. Berko, M. Bublyk, L. Chyrun, Y. Matseliukh, R. Levus, V. Panasyuk, O. Brodyak, L. Dzyubyk, O. Garbich-Moshora, Models and Methods for E-Commerce Systems Designing in the Global Economy Development Conditions Based on Mealy and Moore Machines, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1574-1593.
[7] A. Kazarian, N. Kunanets, V. Pasichnyk, N. Veretennikova, A. Rzheuskyi, A. Leheza, O. Kunanets, Complex Information E-Science System Architecture based on Cloud Computing Model, volume Vol-2362 of CEUR Workshop Proceedings, 2019, pp. 366-377.
[8] A. Kazarian, R. Holoschuk, N. Kunanets, V. Pasichnyk, A. Rzheuskiy, Information Support Of The Virtual Research Community Activities Based On Cloud Computing, in: International Conference on Computer Sciences and Information Technologies, CSIT, CSIT, 2018, pp. 199-202.
[9] L. Wieclaw, V. Pasichnyk, N. Kunanets, O. Duda, O. Matsiuk, P. Falat, Cloud computing technologies in 'smart city' projects, in: Proceedings of the IEEE 9th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS, 2017, pp. 339-342.
[10] V. Vysotska, V. Lytvyn, Y. Burov, A. Gozhyj, S. Makara, The consolidated information web-resource about pharmacy networks in city, volume Vol-2255 of CEUR Workshop Proceedings, 2018, pp. 239-255.
[11] V. Vysotska, L. Chyrun, Analysis features of information resources processing, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2015, pp. 124-128. DOI: 10.1109/STC-CSIT.2015.7325448
[12] V. Vysotska, V.B. Fernandes, M. Emmerich, Web content support method in electronic business systems, volume Vol-2136 of CEUR Workshop Proceedings, 2018, pp. 20-41.
[13] B. Rusyn, V. Lytvyn, V. Vysotska, M. Emmerich, L. Pohreliuk, The Virtual Library System Design and Development, volume 871 of Advances in Intelligent Systems and Computing, 2019, pp. 328-349. DOI: 10.1007/978-3-030-01069-0_24
[14] V. Vysotska, I. Rishnyak, L. Chyrun, Analysis and evaluation of risks in electronic commerce, in: Proceedings of the 9th International Conference on CAD Systems in Microelectronics, 2007, pp. 332-333. DOI: 10.1109/CADSM.2007.4297570
[15] V. Vysotska, L. Chyrun, L. Chyrun, Information Technology of Processing Information Resources in Electronic Content Commerce Systems, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2016, pp. 212-222. DOI: 10.1109/STC-CSIT.2016.7589909
[16] K. Alieksieieva, A. Berko, V. Vysotska, Technology of commercial web-resource processing, in: Proceedings of 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2015.
[17] V. Vysotska, L. Chyrun, Methods of information resources processing in electronic content commerce systems, in: Proceedings of 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2015.
[18] V. Lytvyn, V. Vysotska, V. Shatskykh, I. Kohut, O. Petruchenko, L. Dzyubyk, V. Bobrivetc, V. Panasyuk, S. Sachenko, M. Komar, Design of a recommendation system based on Collaborative Filtering and machine learning considering personal needs of the user, volume 4(2-100) of Eastern-European Journal of Enterprise Technologies, 2019, pp. 6-28. DOI: 10.15587/1729-4061.2019.175507
[19] Y. Kis, L. Chyrun, T. Tsymbaliak, L. Chyrun, Development of System for Managers Relationship Management with Customers, volume 1020 of Advances in Intelligent Systems and Computing, 2020, pp. 405-421. DOI: 10.1007/978-3-030-26474-1_29
[20] O. Cherednichenko, O. Yanholenko, O. Kanishcheva, Developing the Key Attributes for Product Matching Based on the Item's Image Tag Comparison, volume Vol-2631 of CEUR Workshop Proceedings, 2020, pp. 237-247.

[21] O. Piatykop, O. Pronina, Model Selection of the Target Audience in Social Networks in Order to Promote the Product, volume Vol-2604 of CEUR workshop proceedings, 2020, pp. 396-406.
[22] B. Rusyn, L. Pohreliuk, O. Kapshii, J. Varetskyy, A. Demchuk, I. Karpov, A. Gozhyj, V. Gozhyj, I. Kalinina, An Intelligent System for Commercial of Information Products Distribution Based SEO and Sitecore CMS, volume Vol-2604 of CEUR workshop proceedings, 2020, pp. 760-777.
[23] T. Borovska, D. Grishin, I. Kolesnik, V. Severilov, I. Stanislavsky, T. Shestakevych, Research and Development of Models and Program for Optimal Product Line Control, volume 1080 of Advances in Intelligent Systems and Computing IV, Springer Nature Switzerland AG, 2020, pp. 186-201.
[24] T. Shestakevych, V. Pasichnyk, M. Nazaruk, M. Medykovskiy, N. Antonyuk, Web-Products, Actual for Inclusive School Graduates: Evaluating the Accessibility, volume 871 of Advances in Intelligent Systems and Computing, 2019, pp. 350-363.
[25] P. Pukach, K. Shakhovska, The Mathematical Method Development of Decisions Supporting Concerning Products Placement Based on Analysis of Market Basket Content, in: The Experience of Designing and Application of CAD Systems in Microelectronics, 2017, pp. 347-350.
[26] N. Shakhovska, Consolidated processing for differential information products, in: International Conference on Perspective Technologies and Methods in MEMS Design, 2011, pp. 176-177.
[27] N. Shakhovska, O. Vovk, Y. Kryvenchuk, Uncertainty reduction in Big data catalogue for information product quality evaluation, volume 1(2-91) of Eastern-European Journal of Enterprise Technologies, 2018, pp. 12-20.
[28] O. Veres, N.: Shakhovska, Elements of the formal model big date, in: Perspective Technologies and Methods in MEMS Design, MEMSTECH, 2015, pp. 81-83.
[29] V. Ilkiv, Z. Nytrebych, P. Pukach, I. Kohut, B. Pakholok, Order relation on scalar products in real linear spaces, in: 15th International Conference on the Experience of Designing and Application of CAD Systems, CADSM, 2019, pp. 32-35.
[30] L. Chyrun, The E-Commerce Systems Modelling Based on Petri Networks, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1604-1631.
[31] Express delivery. URL: https://www.sat.ua/about/strategy/ekspress-dostavka/.
[32] N.V. Chornopiska, Rynok of logistic services in Ukraine: dynamics and structures, in: the International Conference on Marketing and logistics in the management system, 2012.
[33] Multidirectional vectors of integrated logistics development. Distribution and Logistics, 2(89), 2012, pp. 6-13.
[34] Comprehensive statistical publications. URL: http://www.ukrstat.gov.ua/druk/publicat/kat_u/2020/zb/07/zb_Ukraine%20in%20figures_u.pdf.
[35] Postal Logistics. Transport and Logistics Magazine, 3-4(45-46), 2012, pp. 52-53.
[36] Goal trees. URL: https://pidruchniki.com/18180520/ekonomika/pobudova_dereva_tsiley.
[37] E. Gamm, R. Khelm, R. Dzhonson, D. Vlissides, Priyomy ob"yektno-oriyentirovannogo proyektirovaniya. Patterny proyektirovaniya. Peter, Moscow, 2007.
[38] M. P. Groover, Work systems: the methods, measurement and management of work, Prentice Hall, 2007.
[39] E. Cadbury, Some Principles of Industrial Organization: The Case For and Against Scientific Management, volume 7 of Sociological Review, 1914, pp. 99-125.
[40] M. Krenn, From Scientific Management to Homemaking: Lillian M. Gilbreth's Contributions to the Development of Management Thought, volume 6(2) of Management & Organisational History, 2011, pp. 145-161.
[41] The Complete Guide to Time Tracking. URL: https://www.actitime.com/time-tracking/time-tracking-software-essay.
[42] System.Management Namespace. URL: https://docs.microsoft.com/ruhttps://docs.microsoft.com/ru-ru/dotnet/api/system.management?view=netframework-4.8ru/dotnet/api/system.management?view=netframework-4.8.
[43] AllFusion Process Modeler. URL: http://khpihttp://khpi-iip.mipk.kharkiv.edu/library/technpgm/labs/lab01.htmliip.mipk.kharkiv.edu/library/technpgm/labs/lab01.html.
[44] Full -duplex bi-directional data delivery. URL: https://kaazing.com/kwg/.
[45] Web Application Messaging Protocol. URL: https://wamp-proto.org/intro.html#websocket.
[46] What are Windows Services? URL: https://stackify.com/what-are-windows-services/.
[47] Visual Studio. URL: https://alekseygulynin.ru/visual-studio/.

[48] Import and Export excel in ASP.NET Core 2.0. URL: https://www.talkingdotnet.com/import-export-excel-asp-nethttps://www.talkingdotnet.com/import-export-excel-asp-net-core-2-razor-pages/core-2-razor-pages/.
[49] Time and motion study. URL: https://en.wikipedia.org/wiki/Time_and_motion_study.
[50] The 10 Best Time Tracker Apps in 2019. URL: https://zapier.com/blog/best-time-tracking-apps/.
[51] XAML(WPF). URL: https://msdn.microsoft.com/library/ms752059(v=vs.110).aspx.
[52] JSON. URL: https://ru.wikipedia.org/wiki/JSON.
[53] Angular. URL: https://angular.io/guide/architecture.
[54] ASP.NET ZERO. URL: https://aspnetzero.com/#:~:text=ASP.NET%20ZERO%20is%20an%20intelligent%20and%20mature%20framework.,enables%20faster%20time%20to%20market.
[55] Insales. URL: https://www.insales.ru/blogs/university/oflayn-i-onlayn-magaziny.
[56] Studopedia. URL: https://studopedia.com.ua/1_162872_pobudova-kontekstnoi-diagrami.html.
[57] NodeJs. URL: https://nodejs.org/en/.
[58] NpmJs. URL: https://www.npmjs.com/get-npm.
[59] Telerik. URL: https://www.telerik.com/blogs/what-is-xamarinforms#:~:text=Xamarin's%20website%20is%3A-,Xamarin.,%2C%20iOS%2C%20and%20Windows%20Phone.&text=work%20across%20platforms.-,Xamarin.,starting%20point%20of%20any%20Xamarin.
[60] Microsoft. URL: https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studiossms?view=sql-server-ver15.
[61] V. Lytvyn, V. Vysotska, M. Osypov, O. Slyusarchuk, Y. Slyusarchuk, Development of intellectual system for data de-duplication and distribution in cloud storage, volume 16(2) of Webology, 2019, pp. 1-42.
[62] L. Chyrun, A. Kowalska-Styczen, Y. Burov, A. Berko, A. Vasevych, I. Pelekh, Y. Ryshkovets, Heterogeneous Data with Agreed Content Aggregation System Development, volume Vol-2386 of CEUR Workshop Proceedings, 2019, pp. 35-54.
[63] L. Chyrun, Y. Burov, B. Rusyn, L. Pohreliuk, O. Oleshek, A. Gozhyj, I. Bobyk, Web Resource Changes Monitoring System Development, volume Vol-2386 of CEUR Workshop Proceedings, 2019, pp. 255-273.
[64] K. Mykich, Y. Burov, Algebraic framework for knowledge processing in systems with situational awareness, volume 512 of Advances in Intelligent Systems and Computing, 2017, pp. 217-227.
[65] K. Mykich, Y. Burov, Uncertainty in situational awareness systems, in: Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET, 2016, pp. 729-732.
[66] K. Mykich, Y. Burov, Research of uncertainties in situational awareness systems and methods of their processing, volume 1(4) of Eastern-European Journal of Enterprise Technologies, 2016, pp. 19-27.
[67] K. Mykich, Y. Burov, Algebraic model for knowledge representation in situational awareness systems, in: Computer Sciences and Information Technologies, CSIT, 2016, pp. 165-167.
[68] S. Makara, L. Chyrun, Y. Burov, Z. Rybchak, I. Peleshchak, R. Peleshchak, R. Holoshchuk, S. Kubinska, A. Dmytriv, An Intelligent System for Generating End-User Symptom Recommendations Based on Machine Learning Technology, volyme Vol-2604 of CEUR workshop proceedings, 2020, pp. 844-883.
[69] Y. Burov, K., Mykich, I. Karpov, Building a Versatile Knowledge-Based System Based on Reasoning Services and Ontology Representation Transformations, in: IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT. pp. 255-260.
[70] R. Bekesh, L. Chyrun, P. Kravets, A. Demchuk, Y. Matseliukh, T. Batiuk, I. Peleshchak, R. Bigun, I. Maiba, Structural modeling of technical text analysis and synthesis processes, volume 2604 of CEUR Workshop Proceedings, 2020, pp. 562-589.
[71] P. Kravets, Adaptive method of pursuit game problem solution, in: Modern Problems of Radio Engineering, Telecommunications and Computer Science Proceedings of International Conference, TCSET, 2006, pp. 62-65.

[72] P. Kravets, Game methods of construction of adaptive grid areas. In: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2003, pp. 513-516.
[73] P. Kravets, O. Prodanyuk, Game task of resource allocation. In: Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2009, pp. 437-438.
[74] P. Kravets, Game methods of the stochastic boundary problem solution, in: Perspective Technologies and Methods in MEMS Design, MEMSTECH, 2006, pp. 71-74.
[75] P. Kravets, V. Pasichnyk, N. Kunanets, N. Veretennikova, Game Method of Event Synchronization in Multi-agent Systems, volume 938 of Advances in Intelligent Systems and Computing, 2020, pp. 378-387.
[76] P. Kravets, V. Pasichnyk, N. Kunanets, N. Veretennikova O., Husak, Adaptive Strategies in the Multi-agent "Predator-Prey" Models, volume 1247 of Advances in Intelligent Systems and Computing, 2021, pp. 285-295.
[77] P. Kravets, V. Pasichnyk, O. Artemenko, A. Rzheuskyi, Neuroagent game model of collective decision making in conditions of uncertainty, volume 2631 of CEUR Workshop Proceedings, 2020, pp. 248-267.
[78] L. Chyrun, P. Kravets, O. Garasym, A. Gozhyj, I. Kalinina, Cryptographic information protection algorithm selection optimization for electronic governance IT project management by the analytic hierarchy process based on nonlinear conclusion criteria, volume 2565 of CEUR Workshop Proceedings, 2020, pp. 205-220.
[79] M. Bublyk, O. Rybytska, A. Karpiak, Y. Matseliukh, Structuring the fuzzy knowledge base of the IT industry impact factors, in: Proceed. of the IEEE 13th International Scientific and Technical Conference on Computer sciences and information technologies, CSIT, 2018, pp. 21-24. DOI: https://doi.org/10.1109/STC-CSIT.2018.8526760.
[80] M. Bublyk, V. Mykhailov, Y. Matseliukh, T. Pihniak, A. Selskyi, I. Grybyk, Change Management in R&D-Quality Costs in Challenges of the Global Economy, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1139-1151.
[81] L. Chyrun, I. Turok, I. Dyyak, Information model of the tendering system for large projects, volume 2604 of CEUR Workshop Proceedings, 2020, pp. 1224-1236.
[82] A. Berko, I. Pelekh, L. Chyrun, M. Bublyk, I. Bobyk, Y. Matseliukh, L. Chyrun, Application of ontologies and meta-models for dynamic integration of weakly structured data, in: Proceedings of the IEEE 3rd International Conference on Data Stream Mining and Processing, DSMP, 2020, pp. 432-437. DOI: 10.1109/DSMP47368.2020.9204321
[83] A. Demchuk, B. Rusyn, L. Pohreliuk, A. Gozhyj, I. Kalinina, L. Chyrun, N. Antonyuk, Commercial content distribution system based on neural network and machine learning, volume 2516 of CEUR Workshop Proceedings, 2019, pp. 40-57.
[84] I. Dyyak, L. Chyrun, N. Antonyuk, A. Berko, V. Andrunyk, A. Khudyi, Development of Internet Auction Systems, in: IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT, 2020, pp. 55-61. DOI: 10.1109/CSIT49958.2020.9321844
[85] L. Chyrun, N. Antonyuk, V. Andrunyk, I. Dyyak, A. Berko, K. Mykich, Features of Internet Auction Systems Design, in: IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT, 2020, pp. 134-140. DOI: 10.1109/CSIT49958.2020.9322045
[86] I. Pelekh, A. Berko, V. Andrunyk, L. Chyrun, I. Dyyak, Design of a system for dynamic integration of weakly structured data based on mash-up technology, in: Proceedings of the 2020 IEEE 3rd International Conference on Data Stream Mining and Processing, DSMP, 2020, pp. 420-425. DOI: 10.1109/DSMP47368.2020.9204160