Вінницький національний технічний університет
(повне найменування вищого навчального закладу)
Факультет комп'ютерних систем та автоматики
(повне найменування інституту, назва факультету (відділення))
Кафедра автоматизації та інтелектуальних інформаційних технологій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до магістерської кваліфікаційної роботи
на тему «Розробка інформаційної системи «Особистий кабінет мешканця міста Вінниця». Частина 3. Підсистема серверів та інструментів»

Виконав: студент 2 курсу,
групи 3АКІТ – 18м
спеціальності
151 – «Автоматизація та комп'ютерно–інтегровані технології»
(шифр і назва напряму підготовки, спеціальності)

Плисак В. О.
(прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи:
к.т.н., доц. Ковалюк О. О.
(прізвище та ініціали)

Рецензент: к.т.н., доц. Кривогубченко С. Г.
(прізвище та ініціали)

Вінниця ВНТУ – 2019

Vinnytsia National Technical University
(full name of university)
Faculty of Computer Systems and Automation
(full name of institution, name of Faculty (Department))
Department of Computer Systems
(full name of the department (subject, framing Commission))

Thesis

To Master's qualifying paper
on the topic «Development of the information system «Personal account of a resident of Vinnytsia». Part 3. Server and tools of subsystem»

Performed by: student of 2nd course,
3ACIT – 18m group
specialty
151 – «Automation and Computer Integrated Technologies»
(code name and direction of training, specialty)

Plysak V. O.
(surname and initials)

Supervisor: Ph D., associate professor
Kovaliuk O. O.
(surname and initials)

Reviewer: Ph D., associate professor
Kryvohubchenko S.G.
(surname and initials)

Vinnytsia VNTU – 2019

Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет комп'ютерних систем та автоматики
Кафедра автоматизації та інтелектуальних інформаційних технологій
Освітньо-кваліфікаційний рівень: магістр
Спеціальність 151 – «Автоматизація та комп'ютерно-інтегровані технології»
(шифр і назва)
Освітньо-професійна програма: Інформаційні системи і Інтернет речей

ЗАТВЕРДЖУЮ
Завідувач кафедри АІВТ,
д.т.н., проф. Р. Н. Квєтний
_____
«____» _____ 201__ року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Плисаку Владиславу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема магістерської кваліфікаційної роботи «Розробка інформаційної системи «Особистий кабінет мешканця міста Вінниця». Частина 3. Підсистема серверів та інструментів»,керівник магістерської кваліфікаційної роботи:
Ковалюк Олег Олександрович к. т. н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "" 201 року № .

2. Строк подання студентом МКР 07.12.2019 р.

3. Вихідні дані до магістерської кваліфікаційної роботи підтримка ОС – кросплатформенна, наявність серверної та клієнтської частини – так; максимальний час завантаження системи – 2 с; авторизація користувачів – так; мова графічного інтерфейсу – українська, графічна візуалізація даних – так.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) вступ, огляд аналогів, призначення та основних компонентів інформаційної системи "Особистий кабінет жителя Вінниці", огляд методів та завдань інформаційної системи, розвиток інформаційної системи, розробка UML діаграм інформаційної системи, опис інформаційної системи функції, огляд технологій, що використовуються для розробки інформаційних систем.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень), структурна схема системи, UML-діаграма послідовності, UML-діаграми.

6. Консультанти розділів магістерської кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3,4	к.т.н. , доцент Ковалюк О. О.		

7. Дата видачі завдання “3” 10 2019 року

Календарний план:

№ з/п	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1.	Огляд технологій, моделей та методів створення інформаційної системи	12.10.2019 р.	
2.	Проектування та реалізація інформаційної системи особистого кабінету мешканця м. Вінниці	22.11.2019 р.	
3.	Апробація результатів дослідження	22.11.2019 р.	
4.	Публікації		
5.	Оформлення пояснювальної записки, графічного матеріалу і презентації	05.12.2019 р.	
6.	Захист МКР	12.12.2019 р.	

Студент _____ Плисак В. О.
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ Ковалюк О.О.
(підпис) (прізвище та ініціали)

Vinnytsia National Technical University

(Full name of university)

Faculty of Computer Systems and Automation

(Full name of institution, name of Faculty (Department))

Department of Automation and Intelligent Information Technology

(full name of the department (subject, framing Commission))

Faculty of Computer Systems and Automation
Department of Computer Systems
Education and qualification level: Master
Specialty: 151 – «Automation and computer integrated technologies»
(code name and direction of training, specialty)
Educational-professional program: Information Systems and Internet of Things

APPROVED
Head of the AIIT department
D. Sc., Prof. R. Kvetny
_____
«____» _____ 201__

TASK
AT THE MASTER'S QUALIFICATION WORK STUDENT

Vladyslav Plysak
(firstname, surname)

1. Theme of the master's qualification work «Development of the information system "Personal office of the resident of Vinnitsa". Part 3. Server and tools of subsystem» supervisor:
Kovaliuk Oleh, Ph D., associate professor _____
(surname, name, academic degree, academic title

approved by the order of the higher educational institution from “ ” 201 № .

2. The term of submission by the student of master's qualification work 18. 01. 2019

3. Initial data for master's qualification work OS support - cross-platform; the presence of the server and client part - yes; maximum load time of the system - 2 s; user authorization - yes; language of the graphical interface-Ukrainian; graphical data visualization-yes.

4. Contents of the explanatory note (list of questions to be developed) introduction, review of analogues, purpose and main components of information system "Personal office of the resident of Vinnitsa", overview of methods and tasks of information system, development of information system, development of UML - diagrams of information system, description of information system functions, review of technologies used to develop information systems.

5. List of graphic material (with exact indication of mandatory drawings) system structure diagram, UML diagrams.

6. Consultants of chapters of master's qualification work

Section	Surname, Lastname	Signature, date	
		task issued	task accepted
1,2,3,4	Phd., associate professor Kovaliuk O. O.		

7. Date of issue of the assignment «3» 10 2019

Schedule

№	The name of the stages works	The duration of the stages of work	Note
1	Overview of technologies, models and methods of creating an information system	12.10.2019 p.	
2	Design and implementation of the information system of the personal cabinet of the resident of Vinnitsa	22.11.2019 p.	
3	Approval of research results	22.11.2019 p.	
4	Publications		
5	Designing an explanatory note, graphic material and presentation	05.12.2019 p.	
6	Protection of MQW	12.12.2019 p.	

Student _____ V. Plysak
(signature) (firstname, surname)

Supervisor _____ O. Kovaliuk
(signature) (firstname, surname)

АНОТАЦІЯ

При розробці інформаційної системи «Особистий кабінет мешканця міста Вінниця» та виконанні магістерської роботи було проаналізовано існуючі аналоги та їх застосування, визначено особливості та роль обробки даних у таких інформаційних системах. Проаналізовано основний функціонал інформаційних систем, та розроблено програмне забезпечення. Було розроблено інформаційну систему «Особистий кабінет мешканця міста Вінниця». Було представлено підсистему, яка дозволяє оплачувати комунальні рахунки, додаткові рахунки, створювати електронні петиції, створювати голосування на нововведення у об'єднаннях співвласників багатоквартирних будинків користувачів, перегляд статистики, форум, придбання квитків на культурні заходи міста Вінниці. Інформаційна система «Особистий кабінет мешканця міста Вінниця» має зручний інтерфейс для користувачів. Виконано тестування розробленого програмного забезпечення.

Розроблено структурну схему системи, UML-діаграми.

ABSTRACT

During the development of the information system "Personal cabinet of the resident of the city of Vinnitsa" and the implementation of the master's work, the existing analogues and their application were analyzed, the features and role of data processing in such information systems were determined. The basic functionality of information systems is analyzed and software is developed. The information system "Personal office of the resident of Vinnitsa" was developed. There was presented a subsystem that allows to pay utility bills, additional bills, create electronic petitions, create votes for innovations in associations of owners of apartment buildings of users, view statistics, forum, purchase tickets for cultural events in the city of Vinnitsa. Vinnytsia resident's personal information system has a user-friendly interface. Testing of the developed software is completed.

The structural diagram of the system, UML diagrams is developed.

CONTENT

INTRODUCTION

Topicality. The rapid development of information systems has seen a sharp increase in demand for systems that would make it easier to manage people's finances. Such systems require proper control, transmission and retrieval of data. Therefore, software such as subsystems for utility billing, cost monitoring, and local government monitoring are essential for information systems in the oblast.

Information systems such as "Personal cabinets" are very popular. All over the world, including in Ukraine, such systems are widely used by residents of major cities. It allows from anywhere to pay for all kinds of services and take an active part in the life of the city.

Development features. The developed information system contains: pay utility bills, additional services (garbage collection, housecleaning, pay kindergarten), buy tickets for cultural events.

Innovation. A new system for residents of the city of Vinnytsia is proposed, which differs from the existing analogues, with functionality that allows to manage expenses in full. Such information systems are very useful for saving time. Because they allow all users to do everything through their smartphones or computers. Without such systems, you would have to go to a bank or visit specialized institutions.

The results of the work. As a result of the work done, a information system was created which allow users to manage their payments, manage work of their associations of co-owners of apartments buildings and communicate with government.

The purpose and objectives of study. Establish an information system that would reduce the time spent on managing money for utilities and other expenses associated with cultural activities. Provide a complete set of tools to improve the interaction of residents with other institutions.

Object of study. In this work, the object of study is the processes of processing, receiving and transmitting user data in the information system and between other information systems.

The practical significance of the results obtained. The developed information system can be used for paying utility bills or additional services (garbage collection, housecleaning, pay kindergarten), buying tickets for cultural events.

Approval of the results of the work. The main provisions and results of the research were prepared and provided to the Ministry for Development of Economy, Trade and Agriculture of Ukraine to get a copyright certificate

1 ANALYSIS OF THE PROBLEM OF RESEARCH «PERSONAL CABINET OF RESIDENT OF VINNITSA CITY» INFORMATION SYSTEM

1.1 Application of «Personal cabinet of the resident of Vinnitsa» information system

The development of information technology is steadily increasing. Some technologies have made it possible to improve the interaction of people with public and private institutions or businesses. For example, e-petitions, bank user accounts, and etc.

When designing an automated control system (ACS) which it would not be level (state, regional, production or organizations), it is necessary in any case to divide into separate parts, to facilitate the development of ACS [1]. All the typical patterns, rules of creation are given in the general methodological materials for the development of automatic control systems of enterprises and industrial and non-industrial associations [2].

Information system "Personal cabinet of the resident of Vinnitsa" enables all users to participate in the improvement of the city, to buy tickets for various concerts, performances, and movie theaters.

Users of such systems have the full set of tools needed to facilitate community and MP engagement. Users are provided with up-to-date information on the status and future plans of the City. In European countries, such systems have been time-tested and widely used.

When creating the information system "Personal Cabinet of the resident of Vinnitsa" you need to pay attention to the interaction of the system with the systems. Since the information system must have information provided by other systems (cinema sites, philharmonic sites, public sites, banks) [3–8].

1.2 Analysis of existing analogues of the information system of "Personal cabinets of cities"

There are a small number of such systems in Ukraine. The most famous: «КИЇВ ID», «Особистий кабінет львів'янина», «Персональний кабінет мешканця Івано-Франківська», «Особистий кабінет мешканця Дніпра», «Кабінет ужгородця».

«КИЇВ ID» it is a service aimed at promptly informing residents about changes in traffic, road closures, shutdowns of utilities, emergencies and emergency situations in the city of Kyiv via email or SMS. The service was developed as part of the Kyiv Smart City initiative. The «KYIV ID» information system is shown in figure 1.1

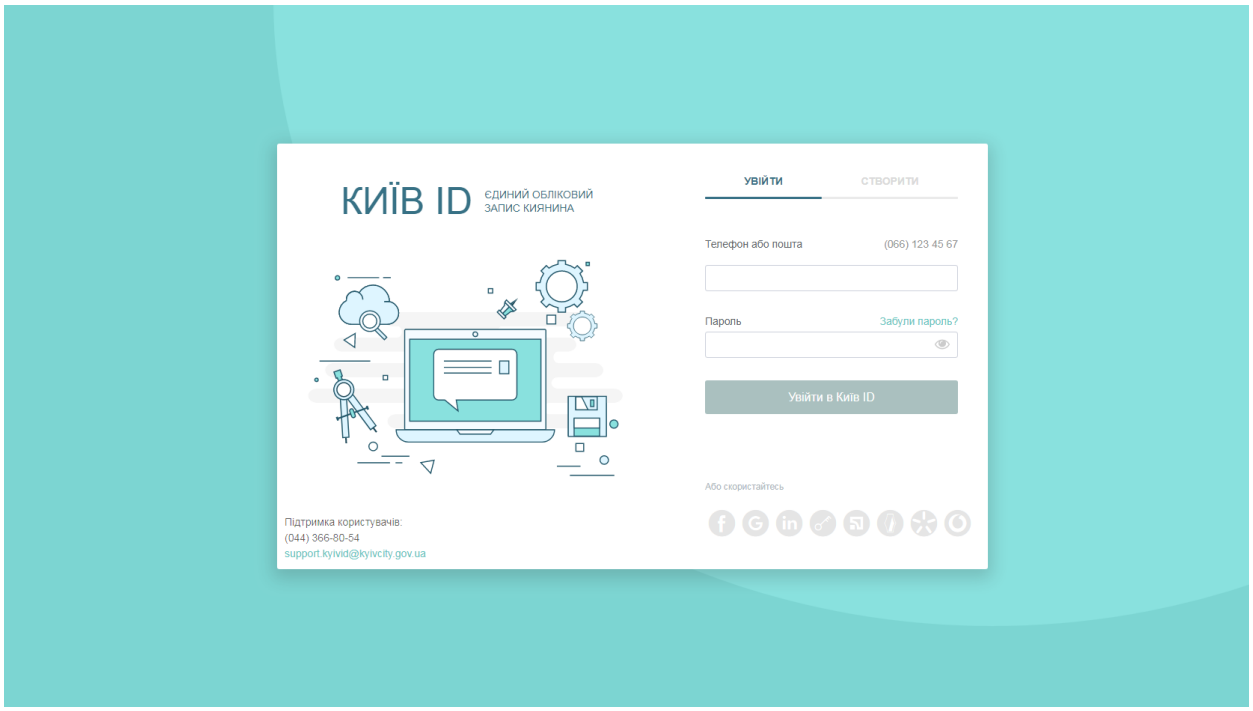


Figure 1.1 – View of the system «КИЇВ ID»

«КИЇВ ID» provides access to many useful functions online:

- starting with a doctor's appointment;
- paying for public transport.

- maintaining housekeeping records with the residents' accounting function;
- ordering various services online;
- editable news feeds and announcements;
- survey and internal form;
- chat and mailing;
- a guide for residents;
- payment for utilities.

The «KYIV ID» list of functions is shown in figure 1.2

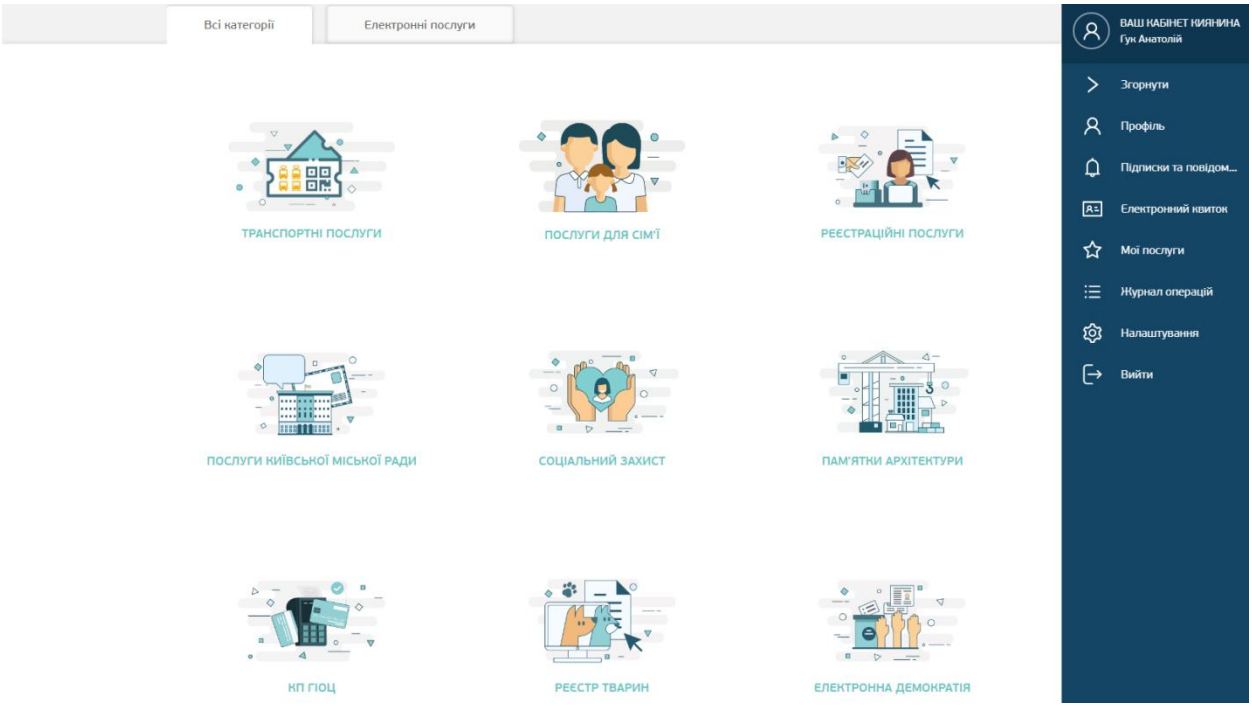


Figure 1.2 – View of «КИЇВ ID» functions

In «КИЇВ ID» information system has simple authorization with mail confirmation. The system is still in progress. «My Services» menu doesn't work. We can see only message that informs about page status. «My Services» page is shown in figure 1.2.

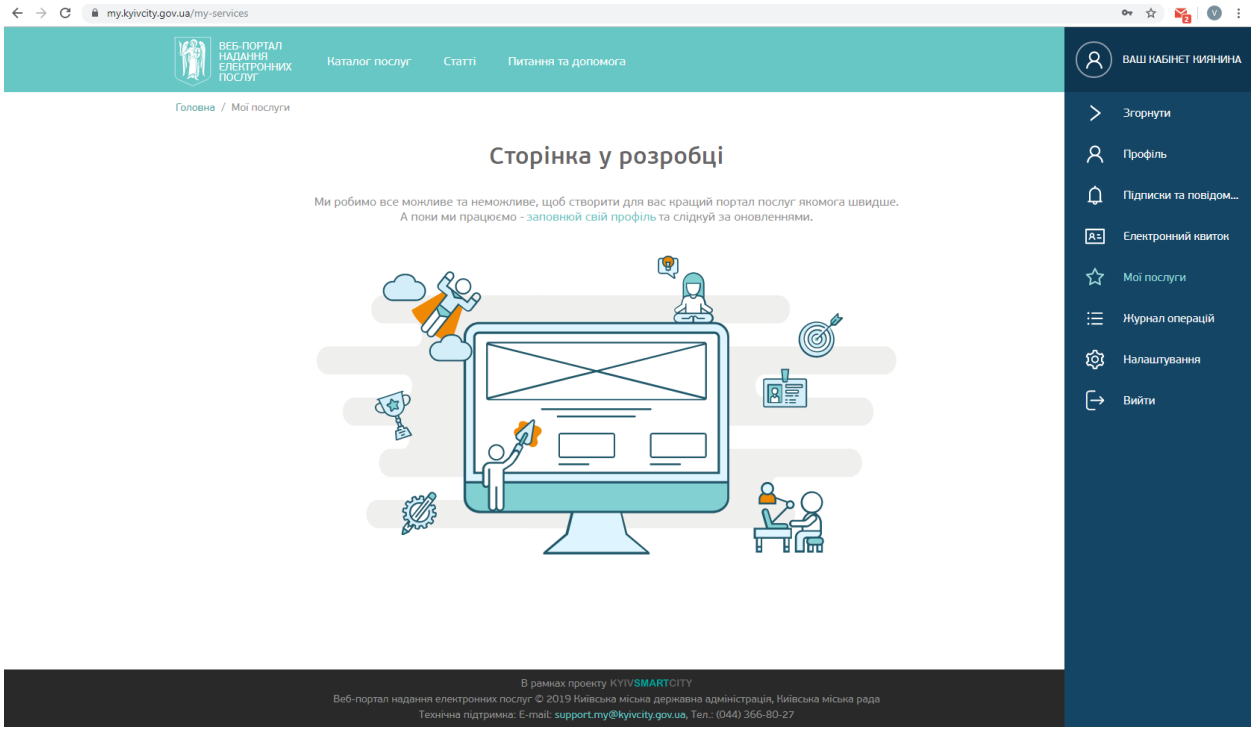


Figure 1.2 – View of the system «КИЇВ ID»

All services required contact information (e-mail, mobile phone, first name, last name etc.). After confirmation, user will get email, with number confirmation.

Confirmation message is shown on figure 1.3.

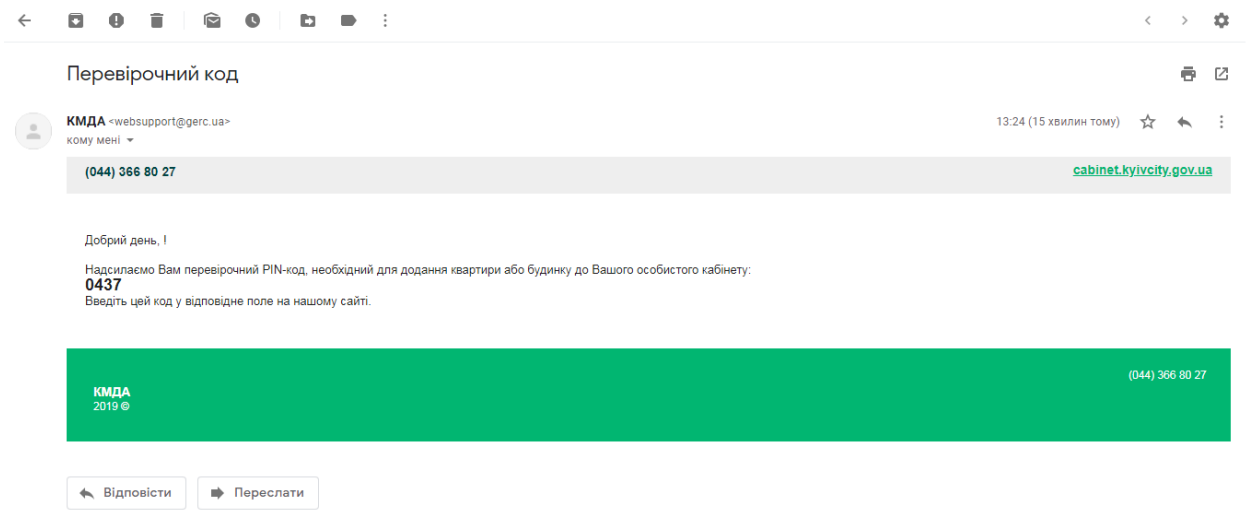


Figure 1.3 – View of confirmation message

Some functionality represented by other information system («Services for families» and «Culture, sports, leisure»). All systems are using Single Sign-On. Single Sign-On (SSO) technology is a technology that allows the user to switch from one system to another without being re-authenticated.

In large private networks, there are often several independent subsystems. SSO provides access to all subsystems without re-entering the login / password. To do this, it is enough for the user to enter the login / password only once for one of the subsystems and he will have access to all others [9]

The advantages of this system include: a user-friendly and intuitive interface, a large number of basic and additional features, a directory of contacts of all the necessary services and services [10].

The disadvantages include: lack of functionality to regulate the work of the association of co-owners of apartment buildings and statistics [11].

«Особистий кабінет львів'янина» was created in 2015 and is one of the latest management systems. The system has the ability to interact with local authorities, buy tickets to local cultural events, and pay for utilities [12].

The advantages of the system can be attributed:

- City hotline;
- E-mail;
- Petitions online;
- Request for public information;
- Payment of taxes;
- Services for residents and guests of Lviv;
- Statistics on Lviv;
- Card of the lion cub;
- Instructions for electronic services.

When you are registering into a system, information system working very slowly and sometimes requests fails. Work of system is shown in figures 1.4 – 1.6.

Figure 1.4 – Work of «Особистий кабінет львів'янина» information system

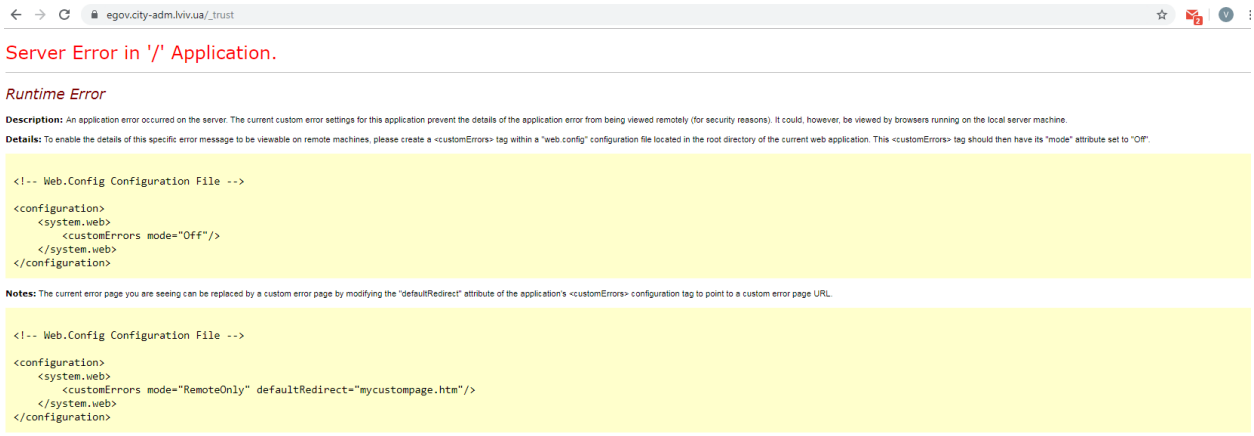


Figure 1.5 – Error page of «Особистий кабінет львів'янина» information system

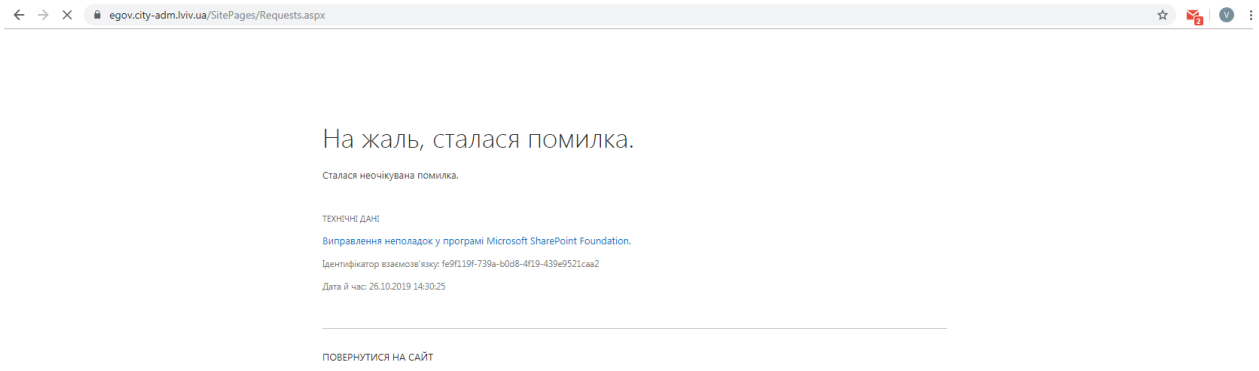


Figure 1.6 – Error page of «Особистий кабінет львів'янина» information system

From the figure 1.6 we can see, that «Особистий кабінет львів'янина» information system is bases on SharePoint.

Microsoft SharePoint Products and Technologies is a collection of software products and components from Microsoft that includes:

- A set of web applications for collaboration;
- Functionality for creating web portals;
- The module of information search in documents and information systems;
- Workflow management functionality and enterprise-wide content management system;
- Form creation module for entering information;
- Functionality for business analysis.

SharePoint can be used to create sites that allow users to collaborate. Sites created on the SharePoint platform can be used as repositories of information, knowledge and documents, and can be used to run web applications such as wikis and blogs that facilitate interaction. Users can manage and interact with information in lists and document libraries using controls called Web Parts (SharePoint WebParts) [13].

Services in this information system represented by another «ІнфоЛьвів» information system [14]. is shown on figure 1.7.

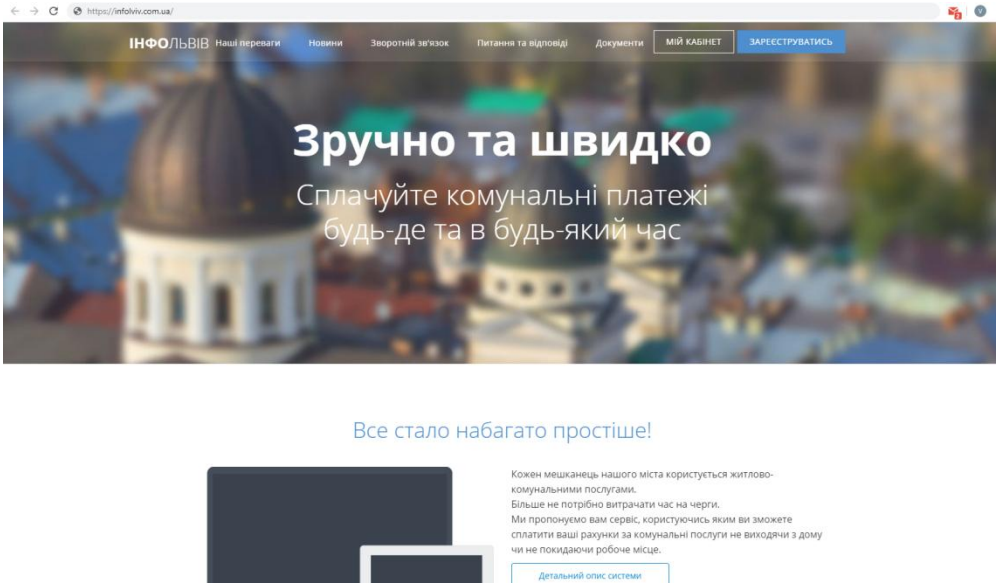


Figure 1.7 – «ІнфоЛьвів» main page

These information system do not use SSO. Each use should register again. And use another credentials. «ІнфоЛьвів» information system has e-mail confirmation for registration.

The layout of the «Особистий кабінет львів'янина» automated system is shown on figures 1.8 and 1.9

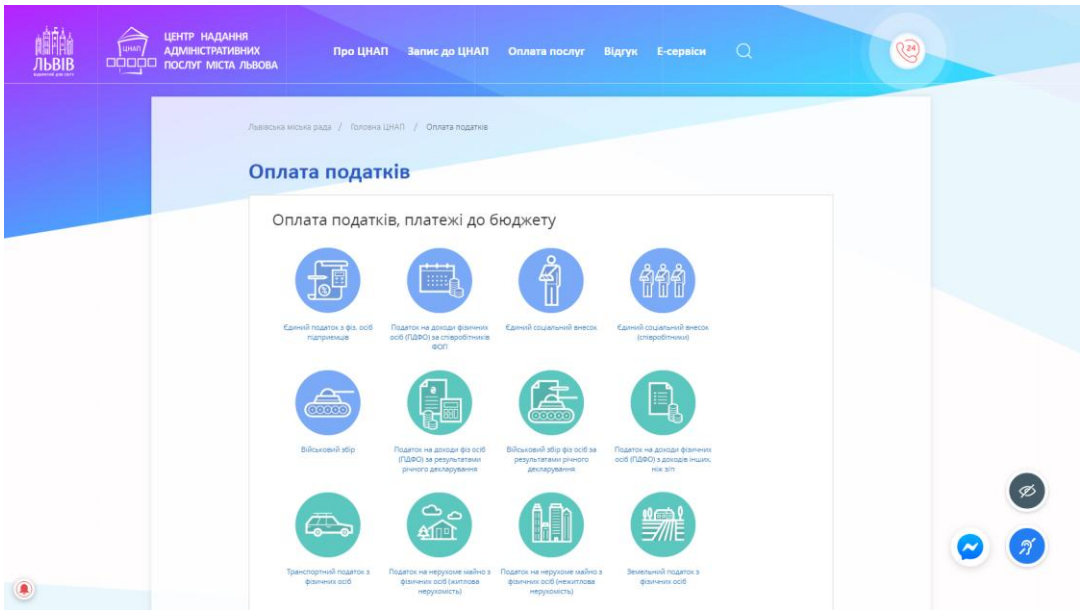


Figure 1.8 – View of the system «Особистий кабінет львів'янина»

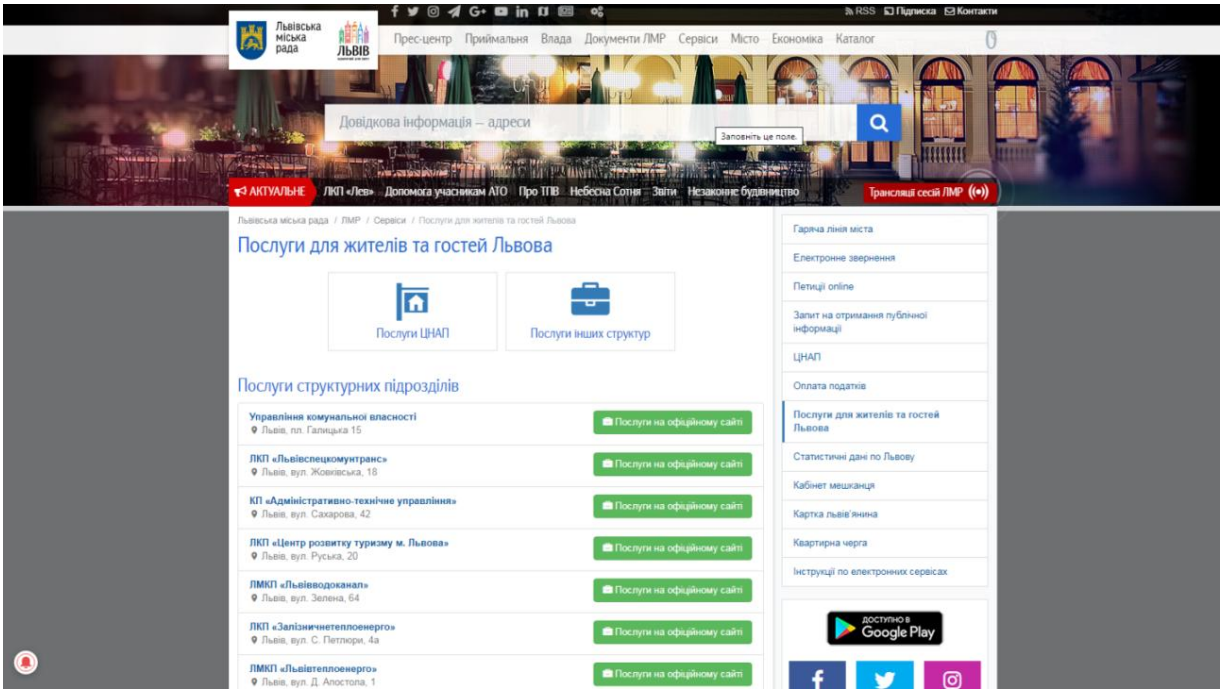


Figure 1.9 – View of the system «Особистий кабінет львів'янина»

Disadvantages of the system: site is working very slowly; the system is still evolving; the need for an ACMB accountant; no SSO, no e-petitions, no forums or chats, no statistics.

System «Персональний кабінет мешканця Івано-Франківська» has been operating since 2014, developed by Ivano-Frankivsk City Council. The system provides functionality for managing the association of co-owners of an apartment building, appointing an "elder" to publish various information regarding the condominium, discussing the problem and introducing new ideas, the ability to pay for utility services online and order a variety of references and documents [15].

As a resident, the user can control the costs of maintenance and repairs of the home check the reasonableness of costs, analyze the history of payments for utilities, view payments and payments on the home. Track the status of fundraising for individual projects, the ability to receive notifications of all events, vote for projects, offer their own ideas, pay for utilities [16]. Electronic personal cabinet provides a

unique opportunity for the user to use the following services through electronic identification:

- “pre-appointment” - the opportunity to apply for an appointment with an administrator, state administrator or state registrar on a convenient day and time with a choice of days two weeks in advance;
- "feedback code" - each visitor receives an accompanying card with a feedback code upon receipt of an administrative or permitting service and can check the status of the administrative or permitting service in real time on the CNAP website;
- "on-line consultation" - service of the official site of the Center for administrative services, through which residents of the city in real time can consult on the provision of administrative or permitting services in chat mode, without visiting the Center. Good-quality counseling is an important component of further administrative or permitting services;
- check the status of the queue at the CNAP
- download the results of administrative and permitting services;
- providing administrative services online with Bank ID technology.

The advantages of the system include: user-friendly and intuitive interface, a large number of basic and advanced features, a directory of contacts of all the necessary services and services [13].

The disadvantages include: no notification of new events, news.

The view of the system «Персональний кабінет мешканця Івано-Франківська» is shown in figure 1.4, figure 1.5 and figure 1.6.

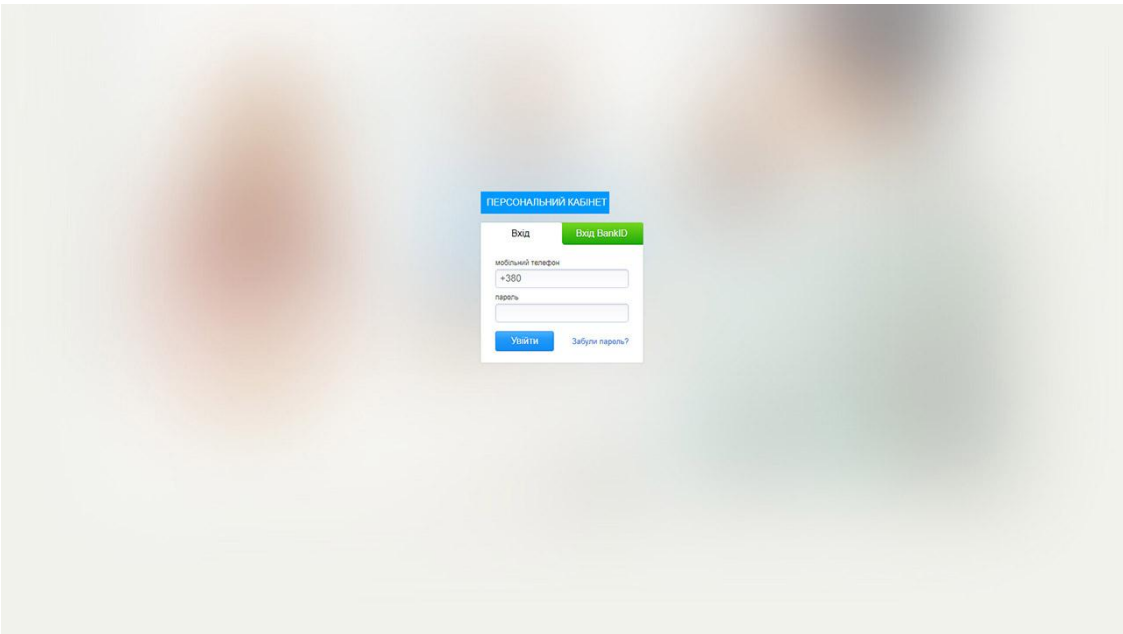


Figure 1.4 – View of the authorization on system «Персональний кабінет мешканця Івано-Франківська»

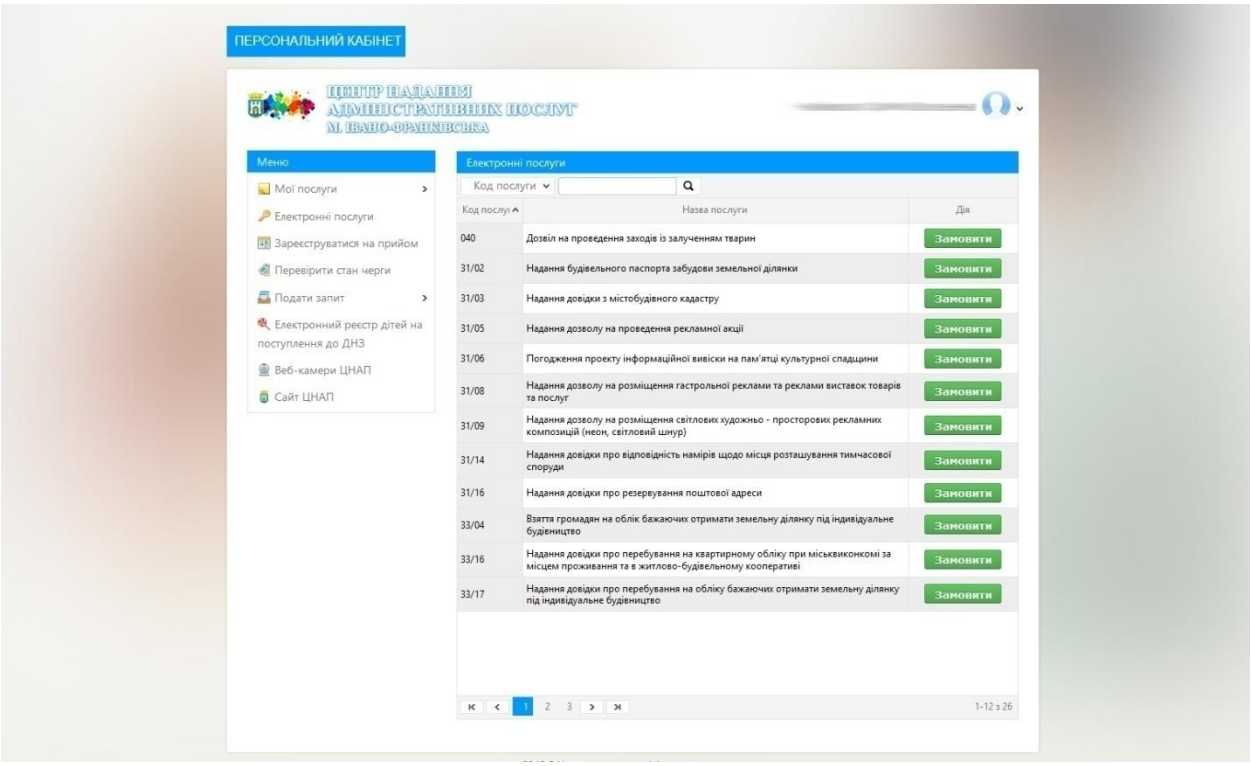


Figure 1.5 – View of the personal cabinet on system «Персональний кабінет мешканця Івано-Франківська»

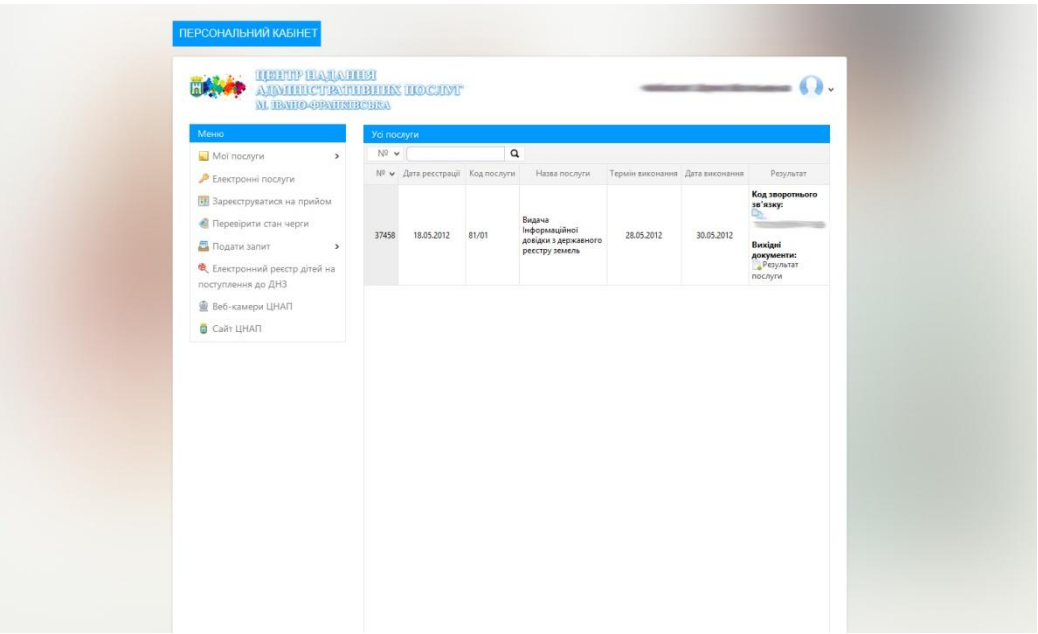


Figure 1.6 – View of the personal cabinet on system «Персональний кабінет мешканця Івано-Франківська»

1.3 Features of an automated control system

Automated Control System (ACS) is a system that is designed for a wide range of hardware and software applications using established economic and mathematical methods to meet the objectives of management [15].

Automated control system is a set of components that allow you to manage the technical, organizational processes of production, enterprises with the participation of the operator (person).

Any ACS consists of:

- object of management;
- control part;
- human operator;

The human operator in the system sets the parameters, restrictions on characteristics [16]. A general diagram of the automated control system is shown in figure 1.12.

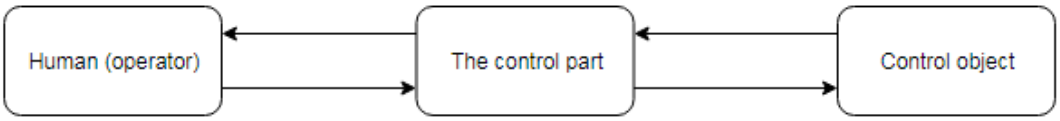


Figure 1.12 – Generalized scheme of automated control system

Information system by subject area of application is a system of economic and organizational class. This system consists of ACS containing the following automated systems [17]:

- intersectoral (automated system of scheduled payments, automated system of information management, automated system of monetary calculations);
- production (automated management systems by organizations);

The functional component of an information system consists of methods that must properly address the planning, organization, and decision-making problems.

When developing a functional component, it is necessary to organize the work of all technical means so that the time for their implementation does not exceed the established limits, even at high loads on these technical means and computers [18].

The ACS technical or hardware is a set of technical means, which are intended to perform the functions properly and to ensure the interaction of the personnel with the technical processes of the ACS [19].

The technical support of the information system «Personal cabinet of the resident of Vinnitsa» consists of:

- Servers where user information is stored;
- Servers on which information concerning personal accounts of users is stored;
- Servers that store information relating to users' bank accounts;
- Internet connection cables;
- personal computers of users and administrators.

DSTU 2938-94 software (software) is a program or set of programs that serve to process information [20].

The definition of software quality is defined in ISO / IEC25010: 2011. For which software quality is a set of properties that characterize the ability of software to meet all the requirements and needs that have been established in its development [21].

The information system software «Personal cabinet of the resident of Vinnitsa» can be designed both on a two-tier client-server architecture, and on a three-tier client-server architecture which is one of the leading concepts for creating distributed web applications.

A two-tier client-server architecture consists of a set of client servers and networks for data transmission between servers and clients [22].

The architecture of such software consists of a graphical layer, a logical layer, and a database layer, where most of the business logic and server level are concentrated [23].

The software of the «Personal cabinet of the resident of Vinnitsa»consists of:

- client application;
- application server;
- a database server that is implemented by means.

1.4 The use of data processing tools in «Personal cabinet of the resident of Vinnitsa» information system

A set of operations such as collecting, recording, converting, storing, deleting, converting are a process of processing information that is performed by software through dedicated data channels [23]. Operation information can be automated in whole or in part known as a "machine" or "automated information processing" system [25], [26].

Automated information gathering and processing not only processes and accumulates data, but also provides the result of the processed information, without duplication according to the role (user, administrator) and the level of access provided [27]. The information processed can also be used for accounting and statistical reporting [28]. All this makes it easier to work with admin data.

Among the information processing tools available to the average user is the organization of a database for querying, searching for the desired information, sorting and filtering information [29]. Comparative analysis of automated information systems is shown in Table 1.

Well-organized database, will allow users of the automatic control system to receive quickly and qualitatively, to find all necessary information, and also to optimize the operation of the automatic control system.

For database management, Microsoft SQL Server 2018 is one of the leading database management systems (DBMS), which will save on the cost of creating new services and applications [30]. This DBMS is also powerful, secure, well-scalable and cross-platform, ie it can work with Windows, Linux and Docker containers. To work with the database, Microsoft SQL Server 2018 uses a procedural SQL language extension - Transact-SQL (T-SQL) [31].

1.5 Clarification of the problem statement for system development

During the analysis of the subject area it was found that the main task to be solved by the information system "Personal office of the resident of Vinnitsa" is to provide all users with the necessary tools to improve living conditions, control the home improvement and quickly solve problems. Each system has a different set of databases.

All considered information systems have practically the same functionality and a set of services. In some systems there are more functions for interaction with public institutions, in others more services for interaction with private institutions.

Considering the characteristics, advantages and disadvantages of similar management information systems, a comparative table can be created. A comparative analysis of information systems is given in Table 1.

Table 1 - Comparative analysis of automated information systems.

Characteristic	Information systems		
	КИЇВ ID	Особистий кабінет львів'янина	Персональний кабінет мешканця Івано-Франківська
Software architecture	client-servera	client-server	client-server
User interface	Web	Web/Adroid	Web
Database model	Relational/Cloud/non-relational	Relational/Cloud/non-relational	Relational
Notifications	E-mail	Absent	E-mail
Status	In progress	In progress	Done
Buying tickets	Present	Present	Absent

Considering all the drawbacks of analogues and the latest technology, it was decided to use the following technology stack:

- - HTML5, CSS3, Meterial Desight Kite, TypeScript, Chart.JS, ng2-charts for the visual part.
- - .Net Framework (.NET Core 2.2, ADO.NET, Entity Framework), MSSQL Server for server side.

2 DEVELOPMENT OF «PERSONAL CABINET OF RESIDENT OF VINNITSA CITY» INFORMATION SYSTEM

2.1 Analysis of the functions of the «Personal cabinet of resident Vinnytsia city» information system

Information system "Personal cabinet of the resident of Vinnitsa" will include the following functionality:

- the possibility of adding several apartments;
- payment for utilities;
- forum for communication;
- the ability to submit petitions and votes to introduce new ideas;
- mail news notification;
- payment of tickets for cultural events;
- statistics;
- news.

This information system has three actors. An administrator who oversees the work of the subsystem, which can edit user data as needed. The administrator can also notify all users of changes to the information system. The headmaster is the head of the condominium, which, in addition to the possibilities of the ordinary user (resident), has additional powers such as: adding news, editing condominium data, confirming applications for the addition of apartments. A simple resident can pay for utilities, edit their own data. Be actively involved in discussing innovations. Buy tickets to cultural events and create online petitions.

2.2 Modeling of «Personal cabinet of resident Vinnytsia city» information system

When modeling an information system, it should be borne in mind that users should only receive data that relates to them and data that is provided depending on the access rights. The data is processed in controller functions using LINQ to Entities technology and the ADO.NET provider for SQL Server [32, 33]. The block diagram of the subsystem is shown in Figure 2.1.

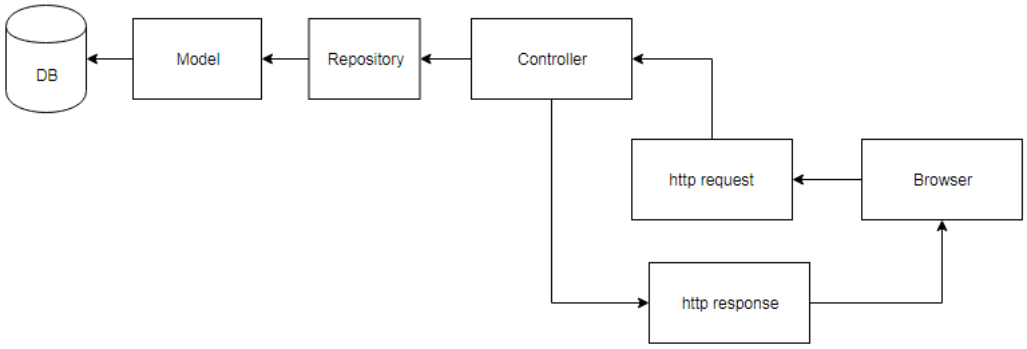


Figure 2.1 – Structural diagram of user interaction with the information system

To optimize development, they use UML (Unified Modeling Language), a language used in object-based software modeling, system planning, or organizational structures. To display the UML precedent diagram - displays the relationship between actors and precedents [34]. To date, there are the following types of UML diagrams: use case (use case diagram, scenario diagram, class (class diagram), cooperation (collaboration diagram, collaboration diagram), sequence (sequence diagram), states (activity diagram), activity (activity diagram), components (component diagram), deployment (deployment diagram) [35].

The UML activity diagram is presented in figure 2.2.

The UML sequence diagram is given in figure 2.3.

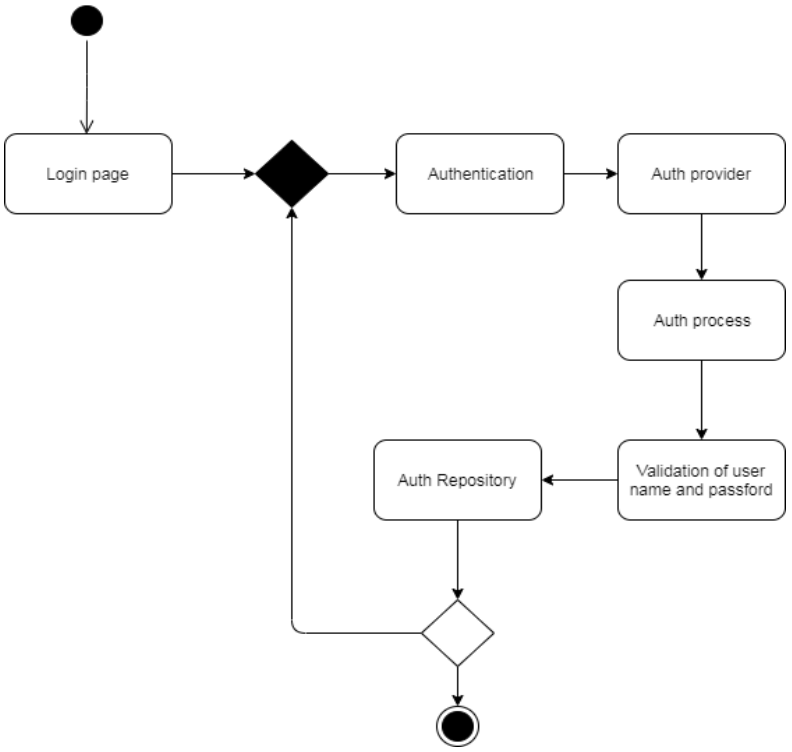


Figure 2.2 – UML activity diagram

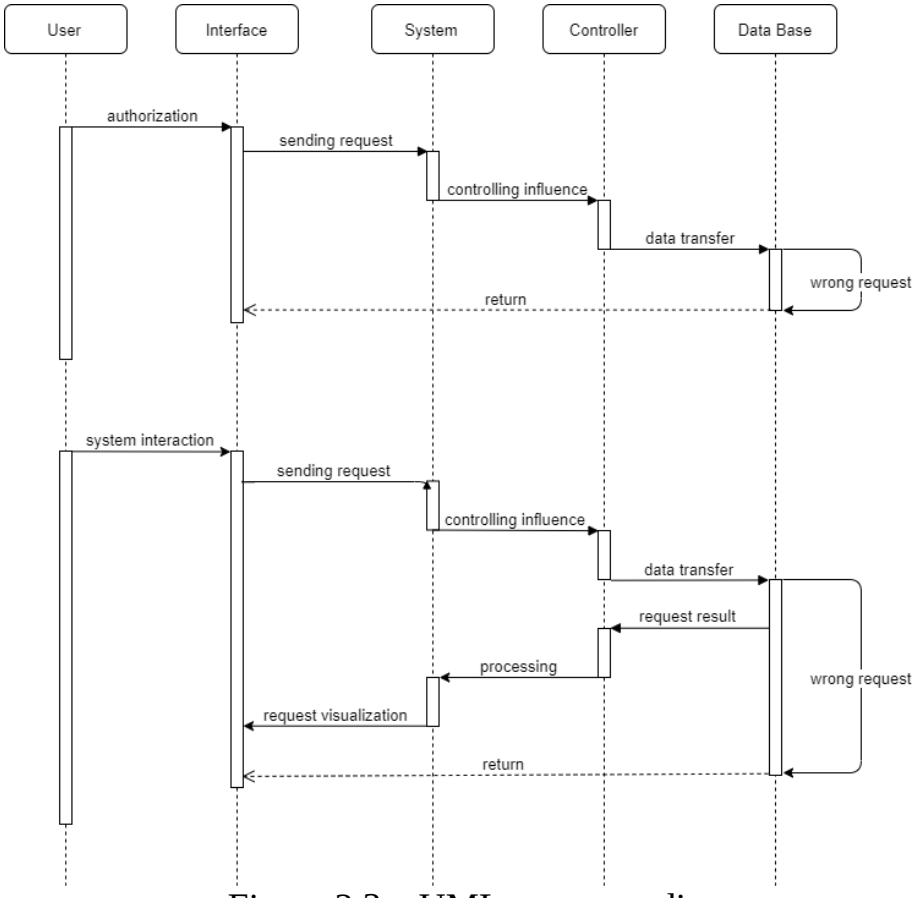


Figure 2.3 – UML sequence diagram

2.3 Development of software "Personal cabinet of resident Vinnytsia city" information system

To develop such a subsystem, you must pay attention to the tasks that must be performed in the development. First of all, the system should be well protected, easy to scale, cross-platform and fast to use. Based on these requirements, the C # programming language was chosen to build the .NET Framework, a technology used to create web applications and web services [36]. This framework includes .NET Core 2.2 technology [37].

The advantages of using this pattern include:

- huge opportunities in software development;
- cross-platform;
- scalability of the software [37].

TypeScript programming language and Chart.JS library, Material Design Lite were used to work with the visual part.

TypeScript is one of the leading programming languages on the Internet that allows you to test and debug scripts and has many libraries and frameworks [38].

The Chart.JS library is specially designed for data visualization. The advantages of this library include:

- the opportunity to apply on a free basis;
- open source;
- a large selection of schedules;
- is supported on all browsers;
- good scalability [39].

Material Design Lite is a free open source library for rendering HTML pages.

Material Design Lite library – includes usage documentation, free CSS libraries, scripts, fonts that allow you to create custom websites with responsive design [40].

The result of using these technologies is shown in figures 2.5 and 2.6.

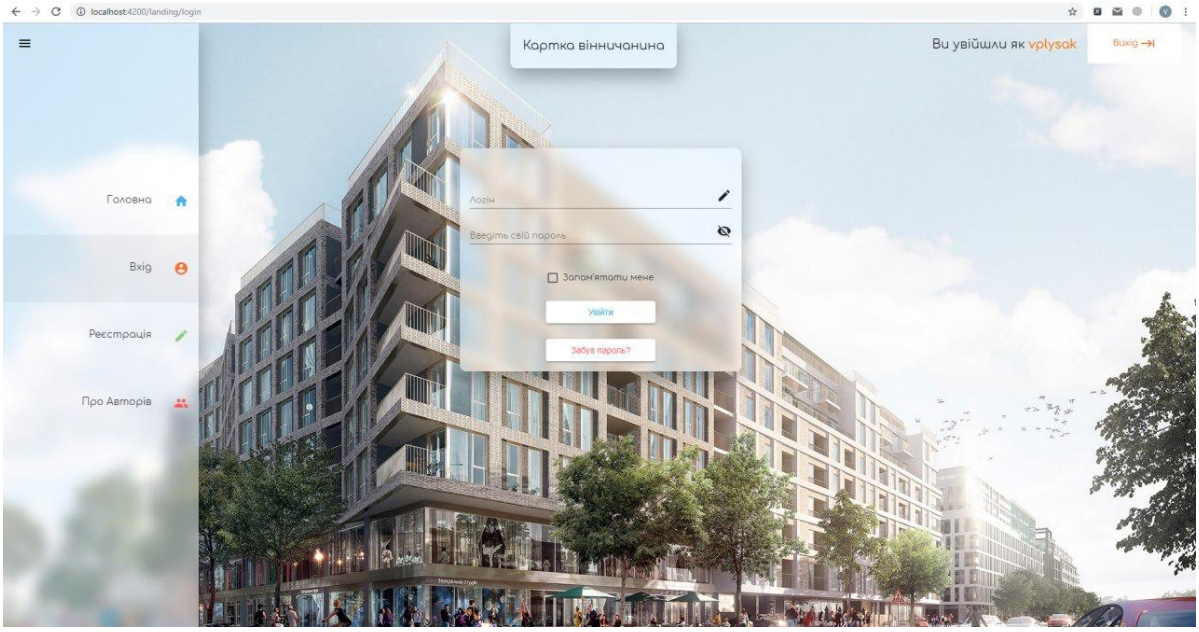


Figure 2.5 – Authorization page

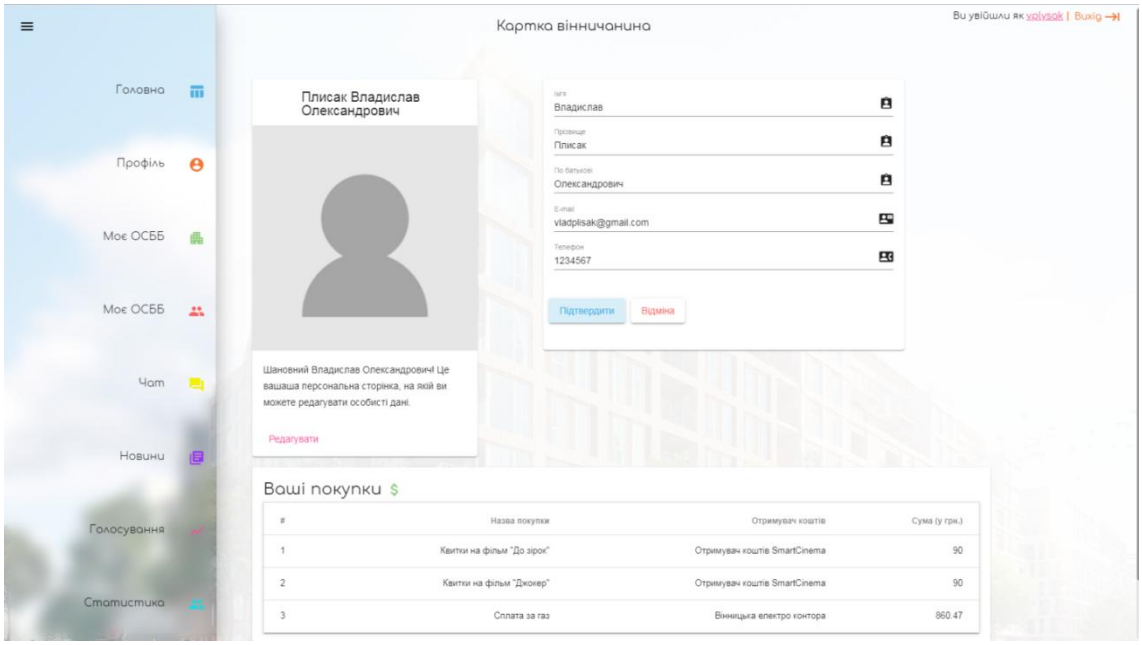


Figure 2.6 – Profile page

2.4 Developing models

The model provides data and methods for working with them: queries to the database, validation. The model does not depend on the presentation (does not know how to visualize data) and the controller (does not have user interaction points) simply providing access to data and its management.

The model is constructed in such a way as to respond to requests, changing its state, and notification of «observers» can be built in.

The model, due to independence from the visual representation, can have several different representations for one «model».

For optimizing data transfers in our information system, we must create some specific models, which will be useful for paginating data. Transfer all data per request is a very bad approach. This type of model calls as «data transfer object» [41]. DTO is a class representing some data with no logic in it. DTO's are usually used for transferring data between different applications or different layers within a single application. You can look at them as dumb bags of information the sole purpose of which is to just get this information to a recipient.

In following recommendations for using DTOs (Data Transfer Objects) for remote services, the use of clean, untainted POCOs to define a well-defined contract with that should keep in a large implementation and dependency-free .dll. The benefits of this allow you to be able to re-use typed DTOs used to define your services with, as-is, in your C#/.NET Clients - providing an end-to-end typed API without the use of any code-gen or other artificial machinery.

These objects are using in data transfers between subsystems. Our information system contains a lot of data related to users. And for a better presentation of information, special models which contains a number of page and page size will help us for solving this issue. Class diagram of «AdditionalBillDto», «ForumMessageDto» and «PaymentDto» classes that were created for this purpose are shown in Figure 2.7.

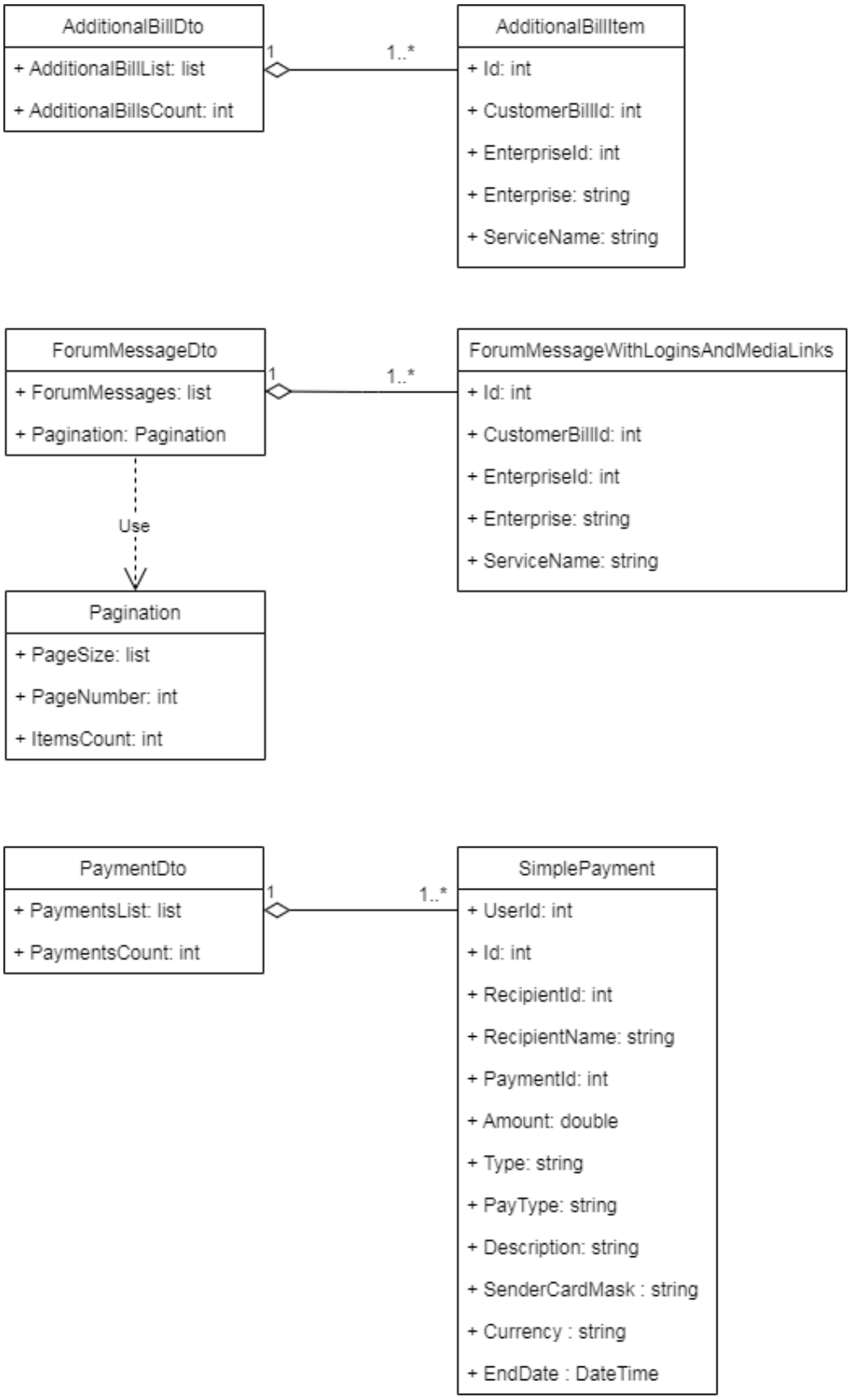


Figure 2.7 – Class diagram of «AdditionalBillDto», «ForumMessageDto» and «PaymentDto» classes

2.5 Developing payment service

Main part of the information system «Personal account of a resident of Vinnytsia». «Stripe» payment platform provide all necessary tools for developing payment services. Stripe is a US technology company that develops solutions for accepting and processing electronic payments. The company was founded on September 29, 2011 by natives of Ireland brothers John and Patrick Collison. The company's headquarters are located in San Francisco, California. The company provides technical and banking infrastructure for online payment systems [42].

First of all, payment must be secure. That is why, «Sripe» payment system generate specific token for each transaction. For more information about developed payment service see appendix B.

For creating connection to Stripe platform, our information system uses next code:

```
loadStripe() {
   if (!window.document.getElementById('stripe-custom-form-script')) {
     const s = window.document.createElement('script');
     s.id = 'stripe-custom-form-script';
     s.type = 'text/javascript';
     s.src = 'https://js.stripe.com/v2/';
     s.onload = () => {
       window['Stripe'].setPublishableKey('');
     };
     window.document.body.appendChild(s);
   }
 }
```

UML sequence diagram of payment is shown on Figure 2.8.

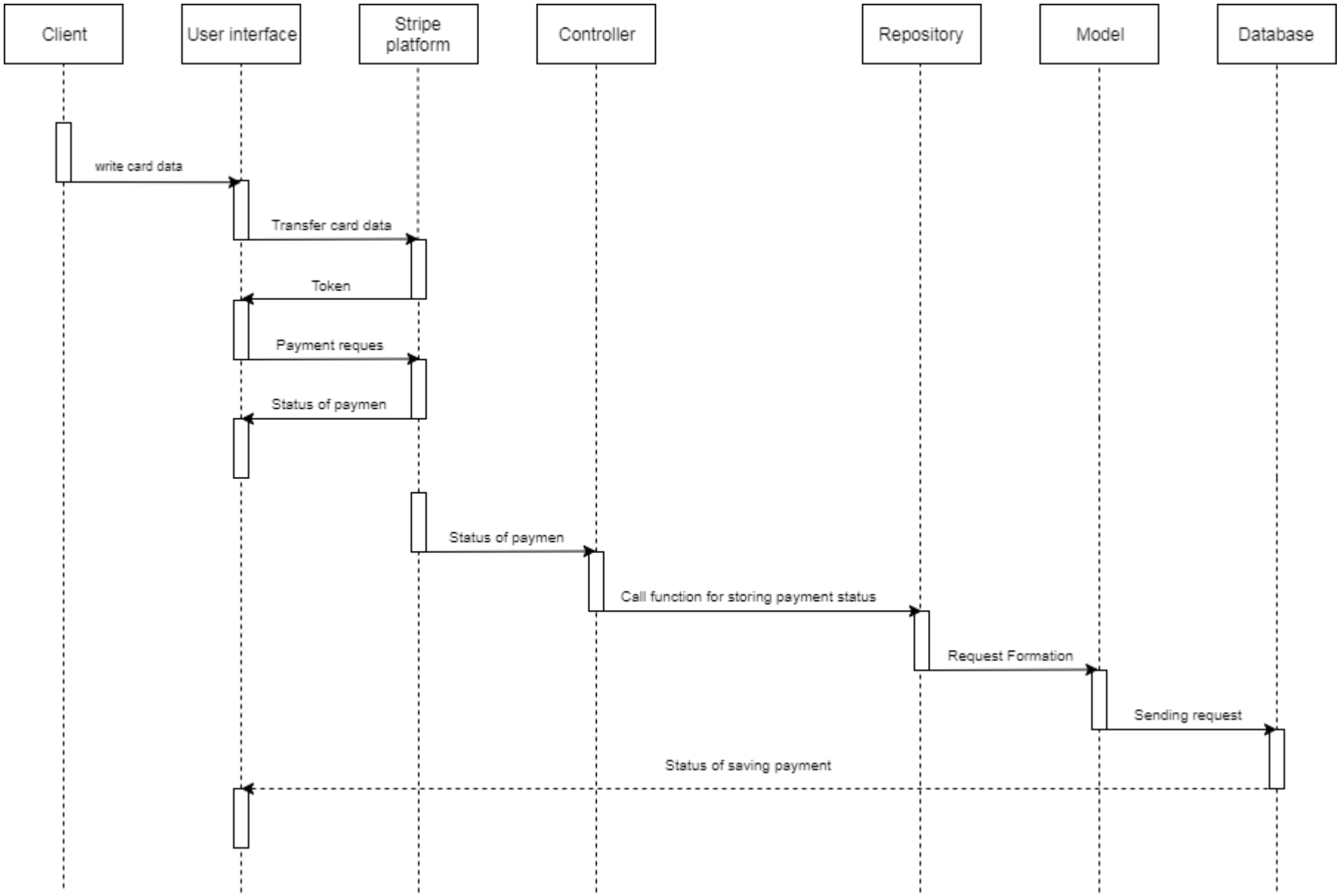


Figure 2.8 – UML sequence diagram of payment is show

2.6 Developing service for getting news

Service for getting news requires a lot of agreements with magazines and newspapers. That is why «News API» is a good choice, for our information system. This platform is gathering all news by period and keyword. Using this API, we can separate information into section. For example: section which contains information about last events in Vinnytsia. The advantages of this platform are that it provides the necessary functionality for a small fee. As it is not necessary to harmonize the conditions of use of information portals in the developed information system. News about last events in Vinnytsia is shown on Figure 2.9.

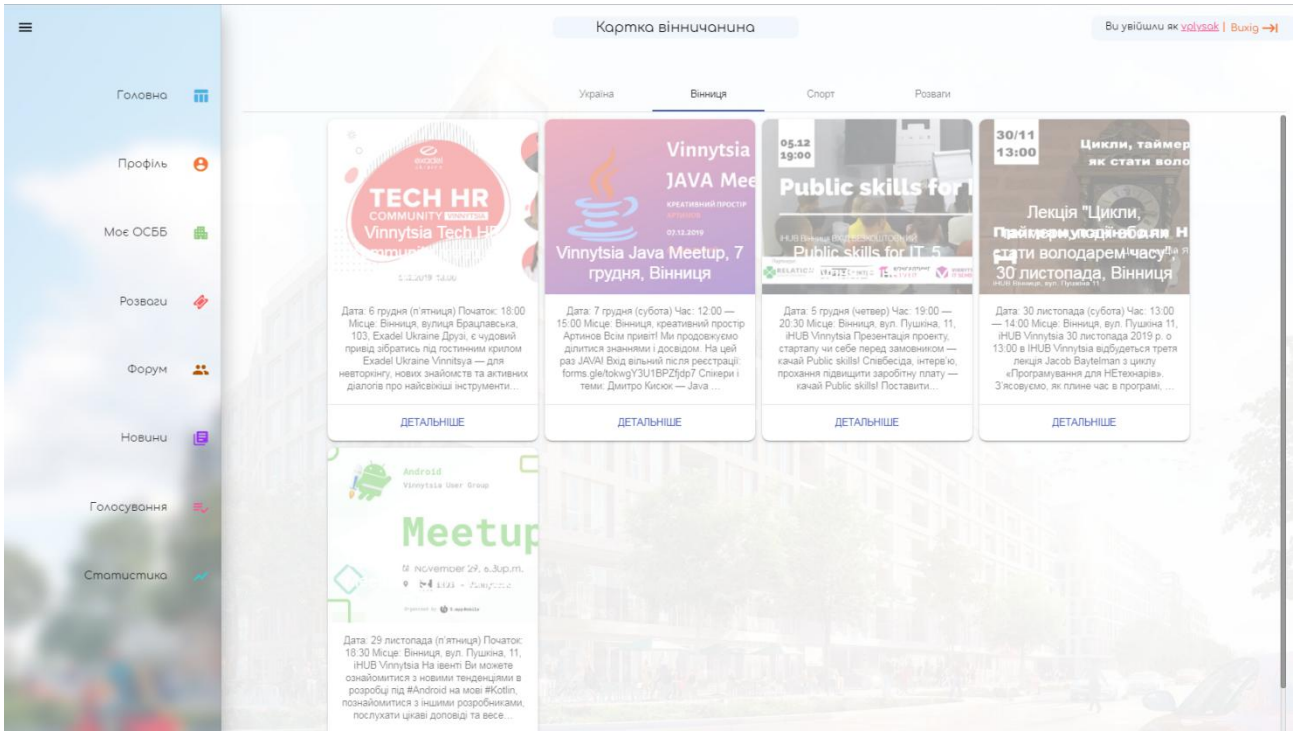


Figure 2.9 – Page with news of Vinnysia city

UML sequence diagram of getting news is shown on Figure 2.10.

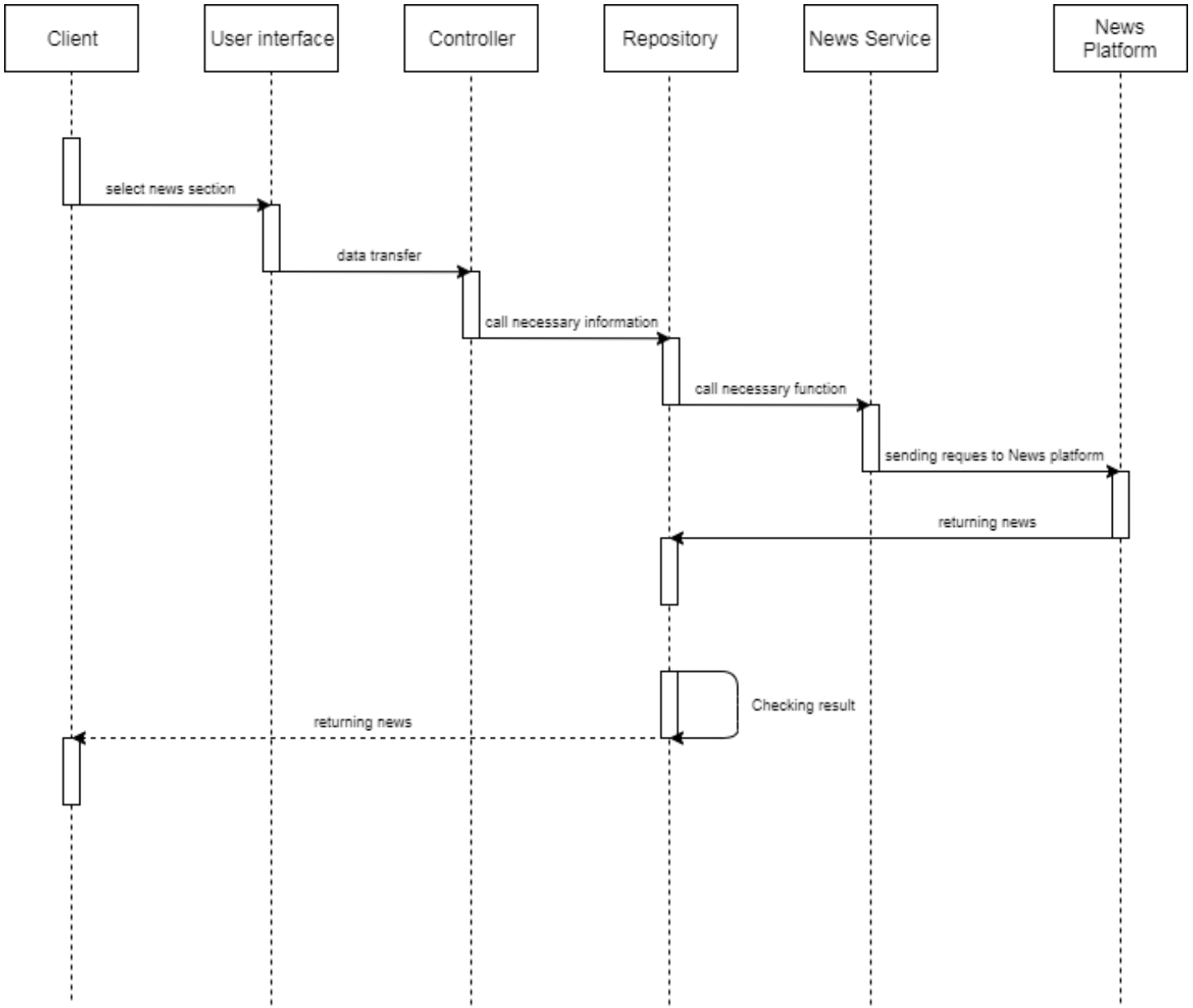


Figure 3.9 – UML sequence diagram of getting news is shown on.

2.7 Conclusion

During the undergraduate practice, the main problems of this subject area were considered in accordance with the topic of the master's thesis. Advantages and disadvantages of existing information systems for urban residents are analyzed and identified. The role of data processing tools in such information systems is defined. The problem statement was made.

Also for this information system the functions and services that will be implemented have been analyzed. This analysis was used to build a UML precedent diagram that reflects the basic functions that users can perform. The structural diagram of this subsystem is developed, as well as the UML diagrams of activity and sequence are designed and developed.

3 DEVELOPMENT OF THE SOFTWARE

3.1 Analysis of system functions

The main purpose of the information system is a set of services that would enable the interaction with state institutions and enterprises of the city.

The following functions should be implemented in the program:

1) Payment for utilities.

2) Payment for additional services.

3) Forum.

4) Signing electronic petitions.

5) Vote for the introduction of new ideas for the improvement of ACMH users.

6) Connection of several apartments.

7) News

8) View statistics on utility costs.

9) View event statistics, cost of spending on the improvement of the city of Vinnitsa.

3.2 Choice of programming language and technology

The C # and TypeScript programming language was selected to create the "Vinnitsa Personal Cabinet" information system. Microsoft .NET Core 2.2 technologies, Entity Framework Core, Dapper, MS SQL database, Angular 8 (Angular material), News Api and the Visual Studio Code development environment were used.

.NET Core is an open source modular software development platform. Compatible with operating systems such as Windows, Linux and macOS. It was

released by Microsoft. The platform has its own community on GitHub. Supports the following programming languages: C #, Visual Basic .NET (partially) and F#.

NET Core is based on the .NET Framework. The .NET Core platform differs from it in modularity, cross-platform, the ability to use cloud technologies, and in that there was a separation between the CoreFX library and the CoreCLR runtime.

.NET Core is a modular platform. Each of its components is updated through the NuGet package manager, which means that you can update its modules separately, while the .NET Framework is updated as a whole. Each application can work with different modules and does not depend on a single platform update. CoreFX is a library integrated into .NET Core. Among its components: System.Collections, System.IO, System.Xml.

Similar to how the .NET Framework implements the Common Language Infrastructure (CLI) via the Common Language Runtime (CLR) and the Framework Class Library (FCL), the two main components of .NET Core are CoreCLR and CoreFX, respectively.

As a CLI implementation of Virtual Execution System (VES), CoreCLR is a complete runtime and virtual machine for managed execution of .NET programs and includes a just-in-time compiler called RyuJIT. .NET Core also contains CoreRT, the .NET Native runtime optimized to be integrated into AOT compiled native binaries [43].

The .NET Core command-line interface offers an execution entry point for operating systems and provides developer services like compilation and package management [44]. Architecture of .Net Core is shown on Figure 3.1

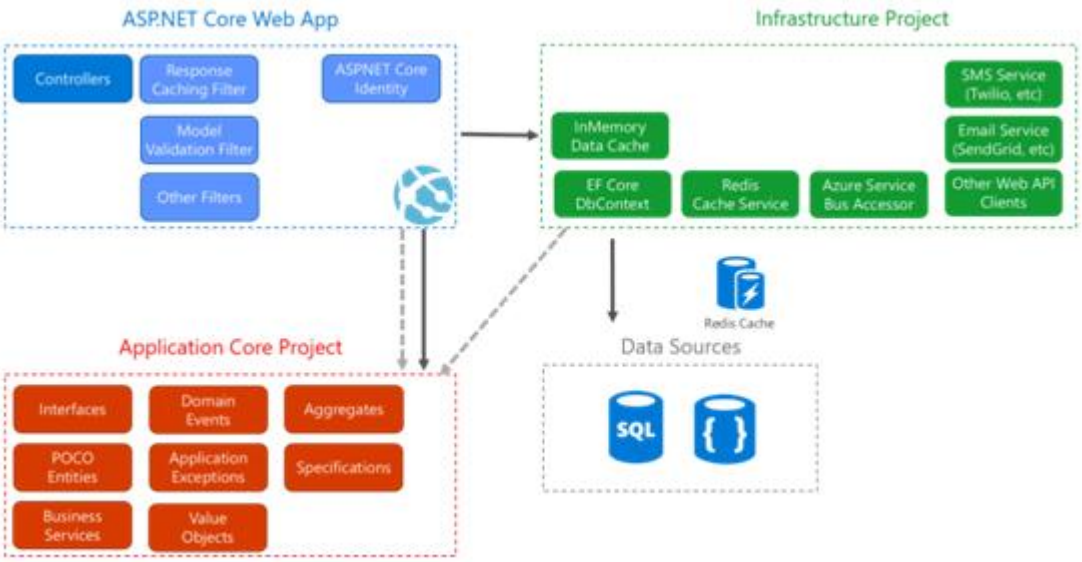


Figure 3.1 – Architecture of .Net Core

Microsoft SQL Server is a commercial database management system distributed by Microsoft. The language used for queries is Transact-SQL, created jointly by Microsoft and Sybase. Transact-SQL is an implementation of the ANSI / ISO standard for structured SQL query language with extensions. Used for both small and medium sized databases and large enterprise-wide databases. It has been successfully competing with other database management systems for many years [45].

SQL Server supports redundant data duplication in three scenarios:

- Snapshot: The database snapshot that the server sends to the recipients is running.
- Change History: All changes to the database are continuously transmitted to users.
- Synchronization with other servers: Databases of multiple servers are synchronized with each other. Changes to all databases occur independently on each server, and data synchronization occurs during synchronization. Duplication of this type provides for the possibility of resolving contradictions between databases.

Entity Framework (EF) Core is a simple, cross-platform and scalable version of the popular open source data access technology Entity Framework.

EF Core can be used as an object relational mapping module (O / RM), allowing .NET developers to work with the database using .NET objects and eliminating the need to write much of the code required to access the data [46].

The Entity Framework is a set of technologies in ADO.NET that support the development of data-oriented software applications. Architects and developers of data-oriented applications have typically struggled with the need to achieve two very different objectives. They must model the entities, relationships, and logic of the business problems they are solving, and they must also work with the data engines used to store and retrieve the data. The data may span multiple storage systems, each with its own protocols; even applications that work with a single storage system must balance the requirements of the storage system against the requirements of writing efficient and maintainable application code [47].

The architecture of the ADO.NET Entity Framework, from the bottom up, consists of the following:

- Data source specific providers, which abstract the ADO.NET interfaces to connect to the database when programming against the conceptual schema.
- Map provider, a database-specific provider that translates the Entity SQL command tree into a query in the native SQL flavor of the database. It includes the Store-specific bridge, which is the component responsible for translating the generic command tree into a store-specific command tree.
- EDM parser and view mapping, which takes the SDL specification of the data model and how it maps onto the underlying relational model and enables programming against the conceptual model. From the relational schema, it creates views of the data corresponding to the conceptual model. It aggregates information from multiple tables in order to aggregate them into an entity, and

splits an update to an entity into multiple updates to whichever table(s) contributed to that entity.

- Query and update pipeline, processes queries, filters and updates requests to convert them into canonical command trees which are then converted into store-specific queries by the map provider.
- Metadata services, which handle all metadata related to entities, relationships and mappings.

Initially, from the very first version, Entity Framework supported the Database First approach, which allowed the edmx model to be generated on a ready-made database [45].

TypeScript - a programming language introduced by Microsoft in the fall of 2012; is positioned as a web application development tool that enhances JavaScript.

The developer of TypeScript is Anders Hejlsberg, who previously created C #, Turbo Pascal and Delphi.

TypeScript is backwards compatible with JavaScript. In fact, after compilation, the TypeScript application can be run in any modern browser or shared with the Node.js server platform.

Advantages over JavaScript:

- possibility of explicit definition of types (static typing),
- support for the use of full-fledged classes (as in traditional object-oriented languages),
- module connection support.

By design, these innovations should increase the speed of development, readability, refactoring and code reuse, search for bugs at the stage of development and compilation, as well as the speed of programs [48].

Angular (commonly referred to as Angular 2 or Angular 2+ frameworks, ie higher versions) is an TypeScript front-end, open source framework developed by Google's Angular Team and a community of private developers and corporations.

Angular is an AngularJS that has been rethought and completely rewritten by the same development team [49].

Material Design (codenamed Quantum Paper) is a design language that Google developed in 2014. Expanding on the "card" motifs that debuted in Google Now, Material Design uses more grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows [50].

The News API is a simple API that allows you to search more than 30,000 news outlets from around the world. For example, you can use the API to know the trending stories in the New York Times, new articles published about a product, or a company reviewed by a blog recently.

API features: The News API lets you search for published articles using keywords or phrases, languages, names of the publication sources, publication dates, and domain names of the publication sources. You can also sort the results according to the date published, the popularity of the publication source, or relevancy of the search keyword.

Price: For non-commercial purposes, the News API is offered for free. However, it comes with some restrictions such as access to delayed new articles, snippet article content, and 1,000 requests per day. For commercial purposes, which has fewer restrictions, the pricing starts from $449 per month.

Ease of use: The API is easy-to-use. Furthermore, you can use its documentation to assist you to get started implementing the API within minutes [51].

Part of .Net Core Web Api project is shown on Figure 3.2. Part of Angular 8 project is shown on Figure 3.3.

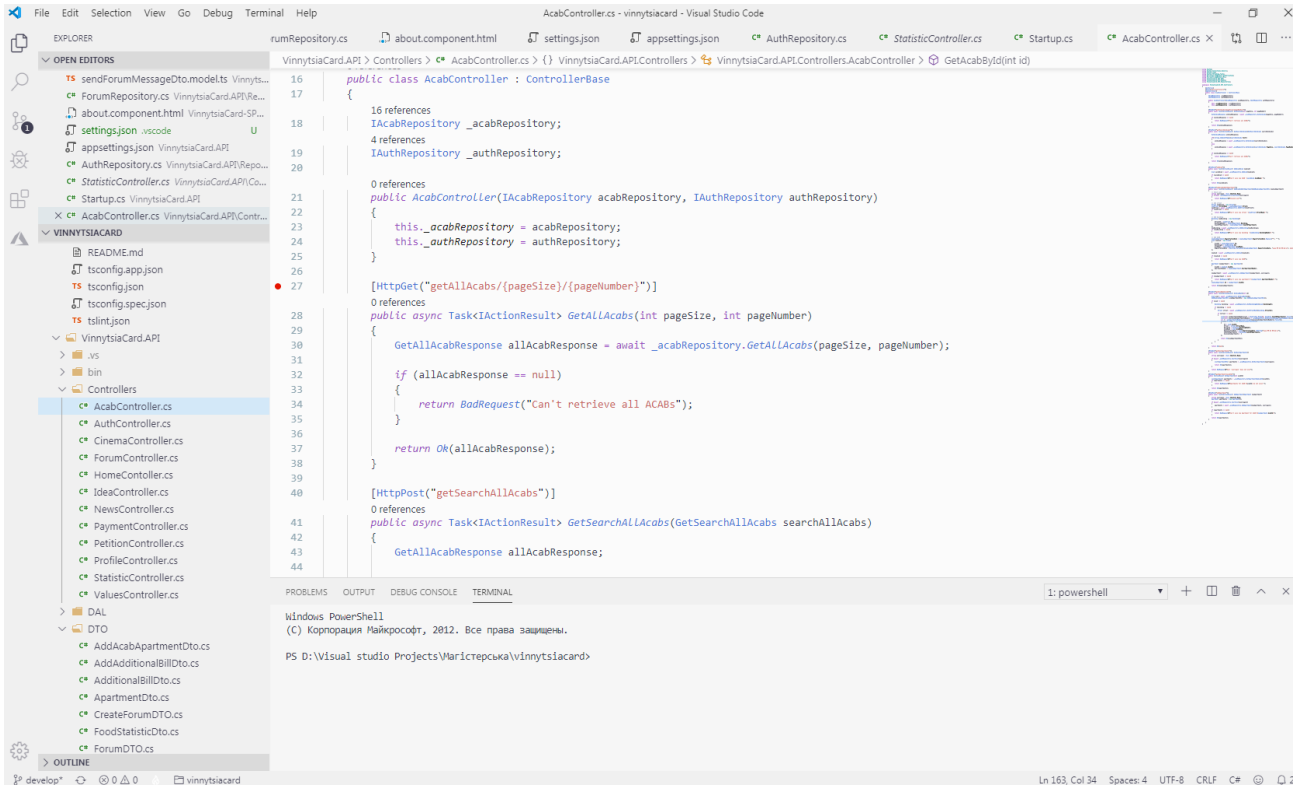


Figure 3.2 – Part of .Net Core Web Api project

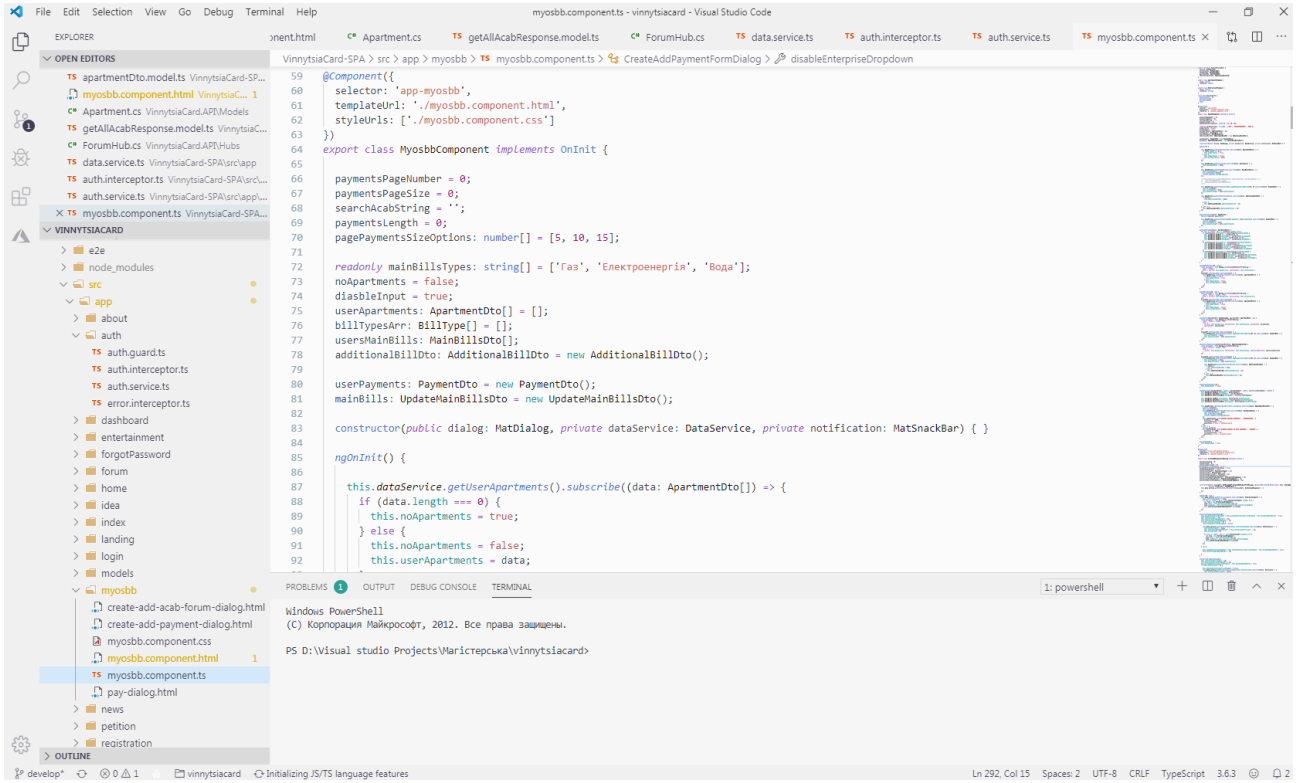


Figure 3.3 – Part of Angular 8 project

3.3 Choosing an Architectural Software Model

REST means sending out the status of a representative, and API means an application. REST is a style of software architecture that defines the set of rules that will be used to create web services. REST-style web services are known as "Vacationers" web services. This allows you to request access systems and manage web resources using a single and predefined set of rules. Interaction with REST-based systems is via Hypertext Transfer Protocol (HTTP).

The rest system consists of:

- a client requesting resources
- a server that has resources.

It's important to create standard REST APIs that simplify development and increase customer experience.

Architectural Constraints of the RESTful API:

Unified interface

- Stateless
- Cached
- Client-server
- Multilayer system
- Code on Demand

The only optional limitation of the REST architecture is the on-demand code. If the service violates any other restrictions, it cannot be clearly called RESTful.

Unified interface: This is a key limitation that distinguishes between the REST API and the non-REST API. This means that there should be a single way to interact with this server, regardless of the device or type of application (website, mobile application).

There are four principles to a single interface:

- Resource-based: Individual resources are defined in requests. For example, APIs/users.

- Manipulation of resources through views: The client has a view of the resource, and it contains enough information to change or delete the resource on the server, provided that it has permission to do so. Example: Usually, a user receives a user ID when a user requests a list of users, and then uses it to delete or change that specific user.

Self-described messages: Each message contains enough information to describe how to process the message so that the server can easily analyze the request.

Hypermedia as a request status engine (HATEOAS): it needs to include links for each response so that the client can easily find other resources.

Stateless: This means that the status required to process the request is contained in the request itself and the server will not store anything related to the session. In REST, the client must include all server information to execute the request as part of the parameters, headers, or request URIs. Stateless provides more accessibility because the server does not need to maintain, update, and report session status. The downside is that the client needs to send too much data to the server to reduce network optimization and require more bandwidth.

Cacheability: Each answer should contain, managed response or not, and how long the response can be cached on the client side. The client will return the data from its cache for any subsequent request and will not need to resubmit the request to the server. Well-managed caching partially or completely eliminates some client-server interactions, further enhancing availability and performance. But sometimes it is likely that the user can get the latest data.

Client-server: The REST application must have a client-server architecture. The client is the one who queries the resources and does not care about storing data that remains internal to each server, and the server is the one who owns the resources and does not relate to the user interface or status of the user. They can develop

independently. The client does not need to know anything about business logic, and the server does not need to know anything about business interface.

Layered system: An application architecture needs to be composed of multiple layers. Each layer doesn't know any thing about any layer other than that of immediate layer and there can be lot of intermediate servers between client and the end server. Intermediary servers may improve system availability by enabling load-balancing and by providing shared caches.

Code on demand: It is an optional feature. According to this, servers can also provide executable code to the client. The examples of code on demand may include the compiled components such as Java applets and client-side scripts such as JavaScript.

REST (Representative State Transfer) is an approach to the architecture of network protocols that provide access to information resources. It was described and publicized in 2000 by Roy Fielding, one of the creators of HTTP. REST is based on the principles of how the World Wide Web works, and in particular HTTP capabilities. Fielding developed REST in parallel with HTTP 1.1 based on the previous HTTP 1.0 protocol.

Data must be transmitted in a small number of standard formats (eg HTML, XML, JSON). Any REST protocol (including HTTP) should support caching, should not depend on the network layer, should not store status information between request-response pairs. It is argued that this approach ensures the scalability of the system and allows it to evolve with new requirements.

The REST antipode is a Remote Procedure Call (RPC) approach. The RPC approach allows the use of a small number of network resources with many methods and complex protocol. With the REST approach, the number of methods and the complexity of the protocol are severely limited, leading to a large number of individual resources. Typical Rest architecture is shown on figure 3.4.

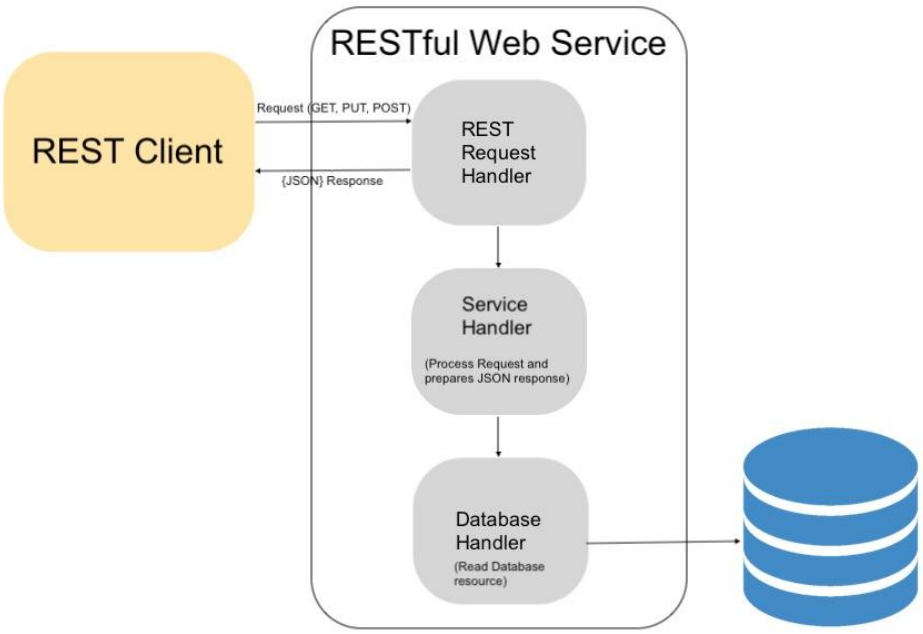


Figure 3.4 – Typical Rest architecture

3.5 Developing database

Windows Azure SQL Databases (first SQL Server Data Services, later SQL Services, and later Windows Azure SQL Databases) is a cloud-based service from Microsoft that enables you to store and process relational data, as well as generate reports. Introduces functionality for different data synchronization scenarios (local infrastructure <=> cloud, cloud <=> cloud). It is part of Windows Azure.

Microsoft Azure was created by Microsoft to build, test, deploy and manage applications and services though their data centres. It was released in 2010.

Microsoft Azure not only offers database as a service but also platform as a service, software as a service and infrastructure as a service. With Azure, clients can use the services purely on the cloud or it can be combined with any existing applications, data centre or infrastructure you may already have in place. Azure's SQL database has the familiar look and feel of Microsoft. It has a strong SQL engine

compatibility and machine learning. The service offers all SQL tools and applications needed for creating a database. It's easy to use.

Microsoft Azure SQL database has an online portal with access to everything you need. Setup is quick and painless but users need to have a Microsoft account to get started.

Subscribers can use the ‘Connection Libraries’ to choose which operating drivers they want to connect. From here you can also choose your preferred language settings, database name, identify a source and a price tier.

Windows Azure SQL Databases are written on Microsoft SQL Server but represent only a subset of data types. Main types are supported: exact and approximate numbers, character strings (including Unicode), date and time, spatial, binary, and other types of data. XML is written, a format for data transmission. Like Microsoft SQL Server, Windows Azure SQL Databases uses T-SQL as query languages. Tabular Data Stream (TDS) is used as a protocol for accessing the service over the Internet. HTTP REST access is not granted. Microsoft recommends using ADO.NET Data Services to transmit data and create services. The user can send Transact SQL queries via TDS to Windows Azure SQL Databases, and this allows applications to use Windows Azure SQL Databases just as they use local SQL Server. Because Windows Azure SQL Databases is a service, its administration has its own features. Unlike local SQL Server administration, Windows Azure SQL Databases separates the logical and physical aspects of administration. The client continues to administer the database, manage logins, users and roles, but the hardware is taken care of by Microsoft. As a result, Windows Azure SQL Databases provides a scalable, multi-user database service with high availability, scalability, security, and self-recovery [52].

Database structure of information system of «Personal account of a resident of Vinnytsia» is shown on Figure 3.5

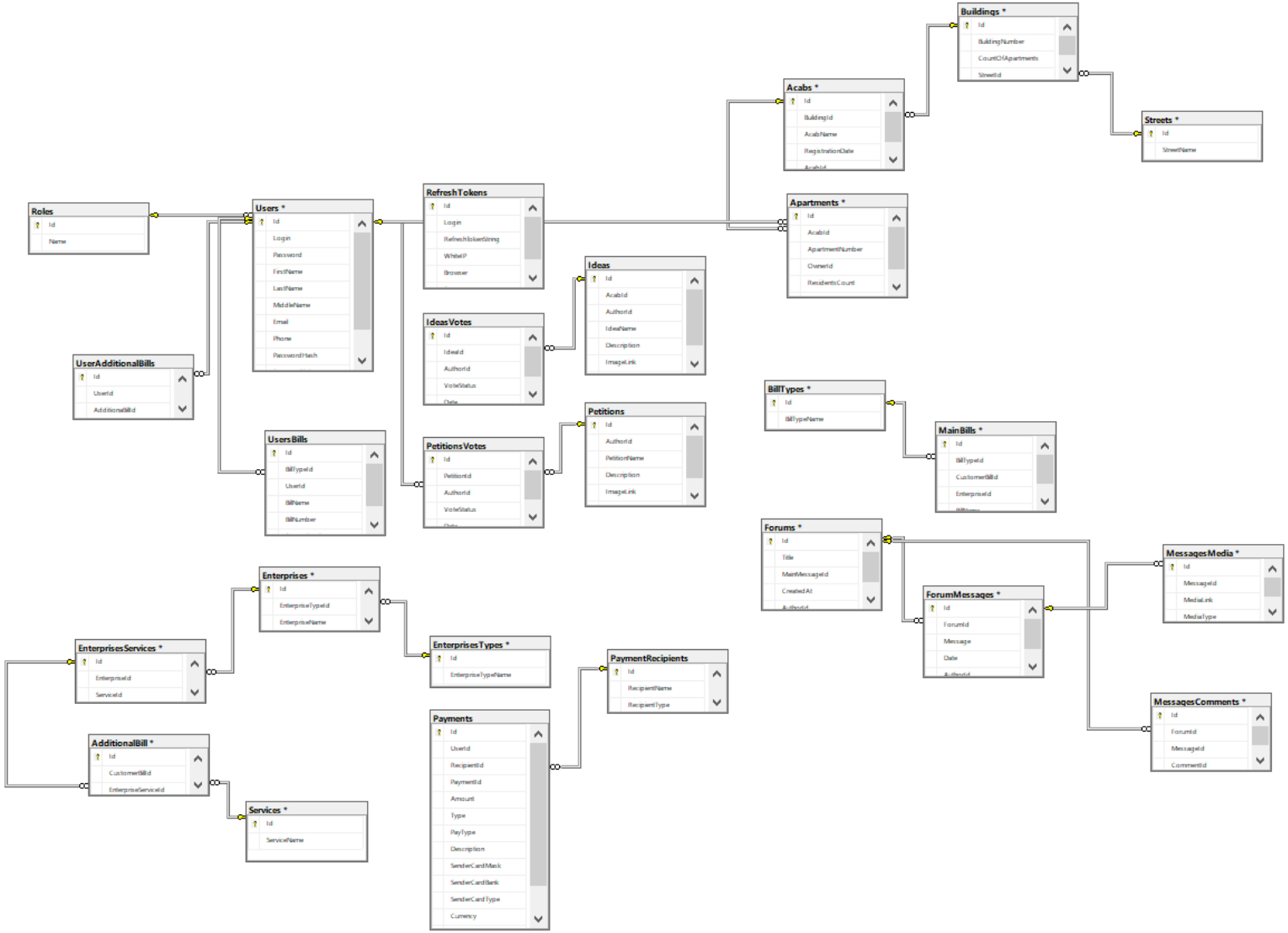


Figure 3.5 – Database structure of information system of «Personalaccountof a residentofVinnytsia»

3.6 Developing forum

A web forum or just a forum is an online resource, a popular form of communication on the Internet. The forum creates topics for communication that makes it better than chat. Anyone interested in certain information can easily and quickly browse it in the forum. The forum has administrators (forum owners) and moderators (maintenance staff who follow the rules and order). Forums can be dedicated to software, cars, football teams and more.

The essence of the work of the forum is to create users (visitors to the forum) their topics with their further discussion, by posting the posting inside these topics. A separate topic, in essence, is a thematic guest book. Users can comment on the stated topic, ask questions and receive answers, as well as answer the questions of other users of the forum and give them advice. Polls (voting) can also be arranged inside the topic if the mover allows it. The questions and answers are stored in the forum's database, and can be useful to participants of the forum, as well as any Internet users who can access the forum, knowing the address of the site, or receiving it from search engines when searching for information.

The topics of the forums can be varied, covering all spheres of life, and either determined by the owners of the forum or its administration, or depending on the contingent of participants. In this case, the forums can be specialized, for example, dedicated to music or any music genre, as well as highly specialized, dedicated to any music group or artist, and without specialization, in this case, users choose the topics of interest to them.

Forums can exist both offline, without being tied to any site, or be part of web portals.

The forum is one of the main information system services. That is why, this service must be optimized. For selecting paginated messages, stored procedure is the most suitable approach for this task [53].

T-SQL query for selecting forum messages:

```
USE [VinnytsiaCardDb2]
GO
/****** Object:  StoredProcedure [dbo].[GetForumMessages]    Script Date:
10.12.2019 18:46:07 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[GetForumMessages] @ForumId int = 0, @PageSize
int = 0, @PageNumber int = 0
AS
WITH [ForumMessagesCRE] ([Id], [ForumId], [Message], [Date], [AuthorId]) AS
(
   SELECT [Id],[ForumId],[Message],[Date],[AuthorId] FROM
[VinnytsiaCardDb2].[dbo].[ForumMessages] WHERE [ForumId] = @ForumId
        ORDER BY [Date]
        OFFSET     @PageNumber * @PageSize ROWS
        FETCH NEXT @PageSize ROWS ONLY
),
[MessagesCommentsCTE] ([Id], [ForumId], [MessageId], [CommentId]) AS
(
   SELECT [Id],[ForumId],[MessageId],[CommentId] FROM
[VinnytsiaCardDb2].[dbo].[MessagesComments]
),
[UsersMessageCTE]
([Id],[Login],[Password],[FirstName],[LastName],[MiddleName],[Email],[Phone],[P
asswordHash]
```

```
    ,[PasswordSalt],[RoleId]) AS
(
SELECT
[Id],[Login],[Password],[FirstName],[LastName],[MiddleName],[Email],[Phone],[Pa
sswordHash],[PasswordSalt],[RoleId]
FROM [VinnytsiaCardDb2].[dbo].[Users]
),
[UsersCommentsCTE]
([Id],[Login],[Password],[FirstName],[LastName],[MiddleName],[Email],[Phone],[P
asswordHash]
    ,[PasswordSalt],[RoleId]) AS
(
SELECT
[Id],[Login],[Password],[FirstName],[LastName],[MiddleName],[Email],[Phone],[Pa
sswordHash],[PasswordSalt],[RoleId]
FROM [VinnytsiaCardDb2].[dbo].[Users]
),
[UsersMessagesCTE]
([Id],[ForumId],[AuthorMessageId],[AuthorMessageLogin],[Message],[Date],[Quote
Id]) AS
(
SELECT FM.Id, FM.ForumId as ForumId, UM.Id as AuthorMessageId, UM.Login as
AuthorMessageLogin, FM.Message, FM.Date, MC.CommentId as QuoteId
FROM [ForumMessagesCRE] as FM, [MessagesCommentsCTE] as MC,
[UsersMessageCTE] as UM
WHERE MC.MessageId = FM.Id AND UM.Id = FM.AuthorId
)
```

```
SELECT UM.*, FM.AuthorId as AuthorQuoteId, U.Login as AuthorQuoteLogin,
FM.Message as Quote  FROM [UsersMessagesCTE] as UM
LEFT JOIN [ForumMessages] FM on UM.QuoteId > 0 AND UM.QuoteId = FM.Id
LEFT JOIN [USERS] U on FM.AuthorId = U.Id
```

Data processing time using this procedure is – 00: 00: 00.1650094. This result is excellent, allowing many users to retrieve data without any delay.

3.7 ACAB service

ACAB service is core service, which allow users to pay utility bills and track payments. For adding apartments information system must know about them. Information system takes this information from opendata.gov.ua web portal [54].

Information system gets data in JSON data format. JSON (pronounced Jason Object JavaScript) is a text format for communicating between computers. JSON is text based, can be read by a person. The format allows you to describe objects and other data structures. This format is mainly used to transmit structured information over the network (thanks to a process called serialization) [55]. Response with existing ACABs is shown on Figure 3.6.

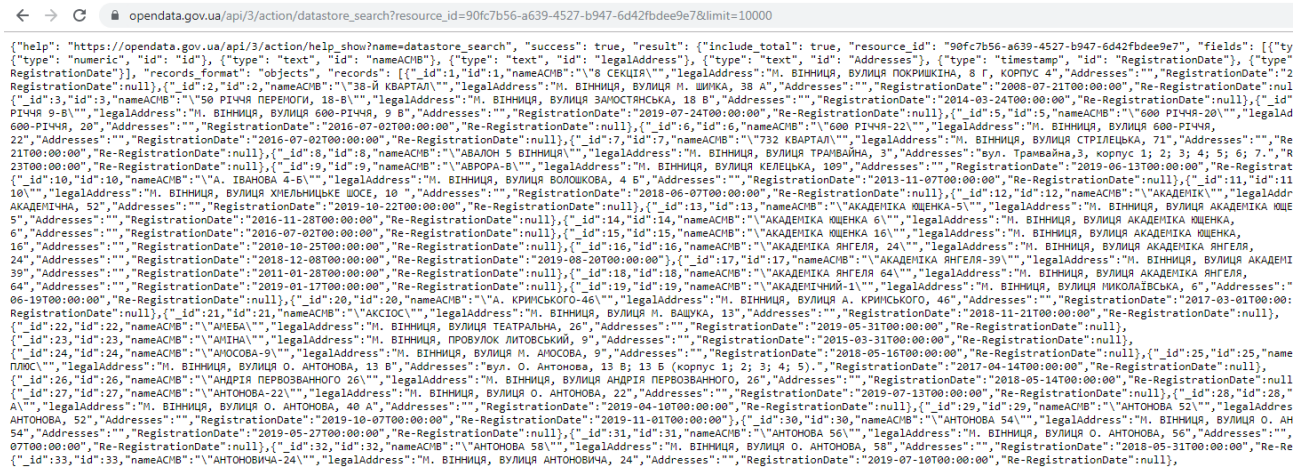


Figure 3.6 – Response with existing ACABs

For getting this data, AcabRepository must use HttpClient, which created by IHttpClientFactory.CreateClient() method.

Code that handling selecting data of ACAB’s:

```
public async Task<GetAllAcabResponse> GetAllAcabs(int pageSize, int pageNumber)
{
    string json = "";
    StringContent content = new StringContent(json, Encoding.UTF8, "application/json");
    string requestUri =
        "https://opendata.gov.ua/api/3/action/datastore_search?resource_id=90fc7b56-a639-4527-b947-6d42fbdee9e7&limit=10000";

    var client = _httpClientFactory.CreateClient();

    var moviesResponse = await client.GetAsync(requestUri);
    GetAllAcabResponse response;

    try
    {
        response = JObject.Parse(await
            moviesResponse.Content.ReadAsStringAsync()).ToObject<GetAllAcabResponse>();
    }
    catch (Exception ex)
    {
```

```
                throw new Exception($"Can't parse response.Content to GetAllAcabResponse object: { ex.Message }");
            }

            try
            {
            response.Result.Records = response.Result.Records.Skip(pageSize * pageNumber).Take(pageSize).ToList();
            }
            catch (Exception ex)
            {
                throw new Exception($"Can't pagianate GetAllAcabResponse response.Result.Records (Page size: {pageSize}; page number {pageNumber}): { ex.Message }");
            }

            return response;
        }
```

After getting necessary count of paginated ACABs data, user will be able to add to select ACAB and apartment on UI. Code of form, for connecting apartment from angular file myosbb.component.ts:

```
        saveAcabApartment() {
          this.addAcabDto.userApartmentNumber = Number.parseInt(this.apartmentUser);
```

```
this.data.service.addNewAcabAndUserApartment(this.addAcabDto).subscribe((data: AddAcabWithApartmentDto) => {
        console.log('Apartment was successfully added.');
        console.log(data);
        this.dialogRef.close();
        this.data.notification.open('Ви підключились до ОСББ ' + data.acabName + '!', 'Повідомлення', {
          duration: 5 * 1000,
          verticalPosition: 'top',
          panelClass: ['blue', 'popupmessage']
        });
      }, err => {
        console.log(err);
        this.data.notification.open('Ви не підключились до ОСББ' + this.addAcabDto.acabName + '!', 'Помилка', {
          duration: 5 * 1000,
          verticalPosition: 'top',
          panelClass: ['red', 'popupmessage']
        });
      });
    }
```

3.8 Algorithm of the system

The developed subsystems serve to improve the communication of the inhabitants with each other and to facilitate the management of the association. You first need to sign up. Then apply to add yourself and your apartment or flats to an

already registered association. If the association is not registered on the system, it must be registered by the headman. When a user is added to an association, they are given the opportunity to:

a) communicate with other residents;

b) pay for utilities and ancillary services provided by specific associations of residents in which the user is registered;

c) create voting proposals;

d) participate in the voting on the submitted proposals;

e) discuss all news and events presented to residents;

e) change their personal data;

f) withdraw from the merger if the apartment was sold by the user or for any other reason he lost the ownership of the apartment;

g) access to statistics on utility costs to compare their costs to the average per unit of residents;

When logged out or deleted, the user loses access to the system.

4 TESTING THE PROGRAM AND THE RESULTS OF THE WORK OF «PERSONAL CABINET OF RESIDENT OF VINNITSA CITY» INFORMATION SYSTEM

4.1 Description how system is working

After opening web site of information system, the user goes to the «Вхід» page of the system (user authorization form), as shown in Figure 4.1.

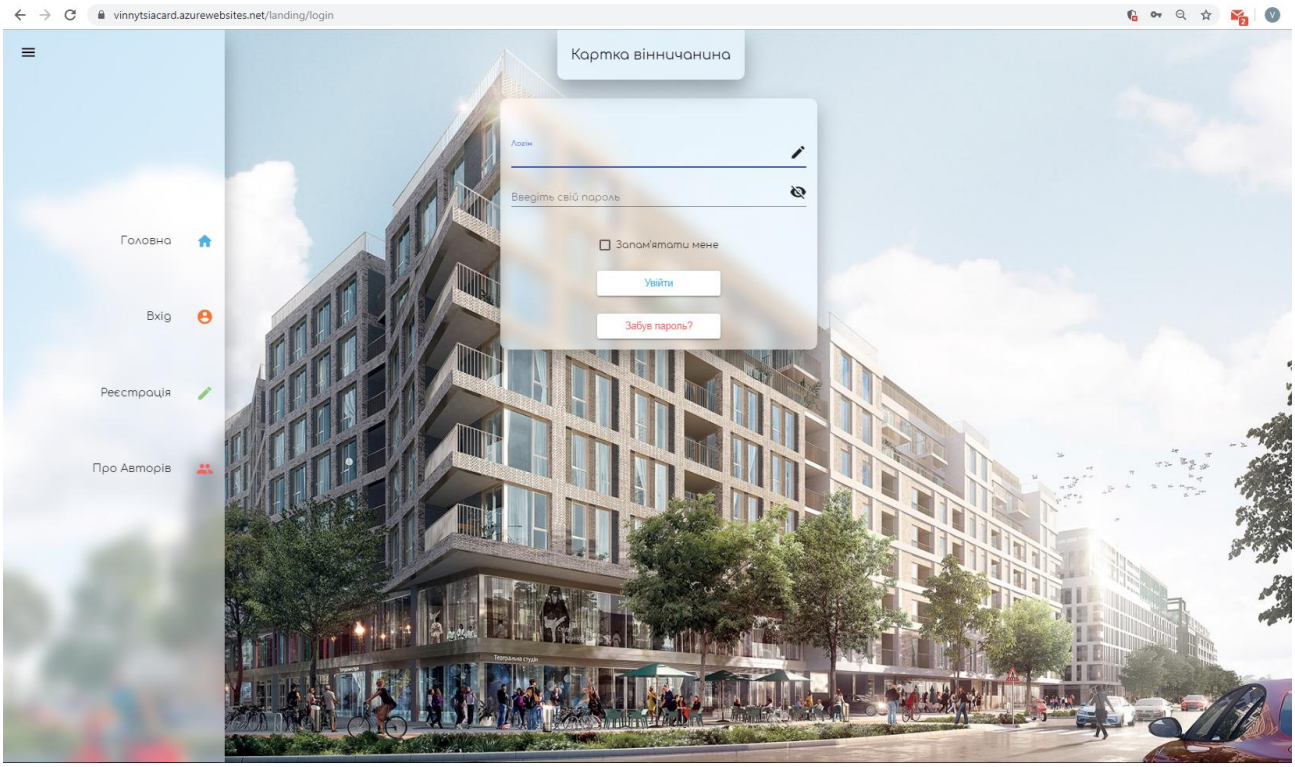


Figure 4.1 – Web page «Вхід»

But since every new user of this information system has to register, you first need to click on the "Registration" button and go to the «Registration» window, which is shown in Figure 4.2

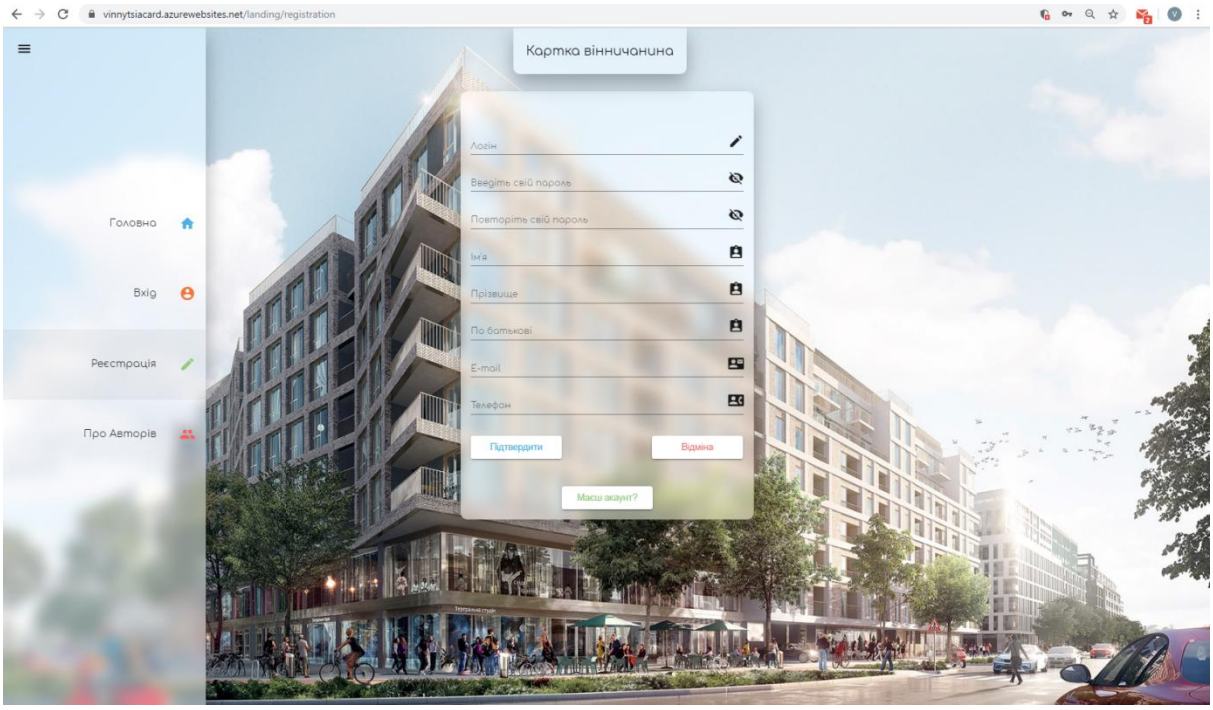


Figure 4.2 – Web page «Реєстрація»

In «Вхід» web page, the user must complete all fields provided and click on the «Увійти» button. If user wrote wrong credentials, he will see error message. A window with an example of this function is depicted in Figure 4.3.

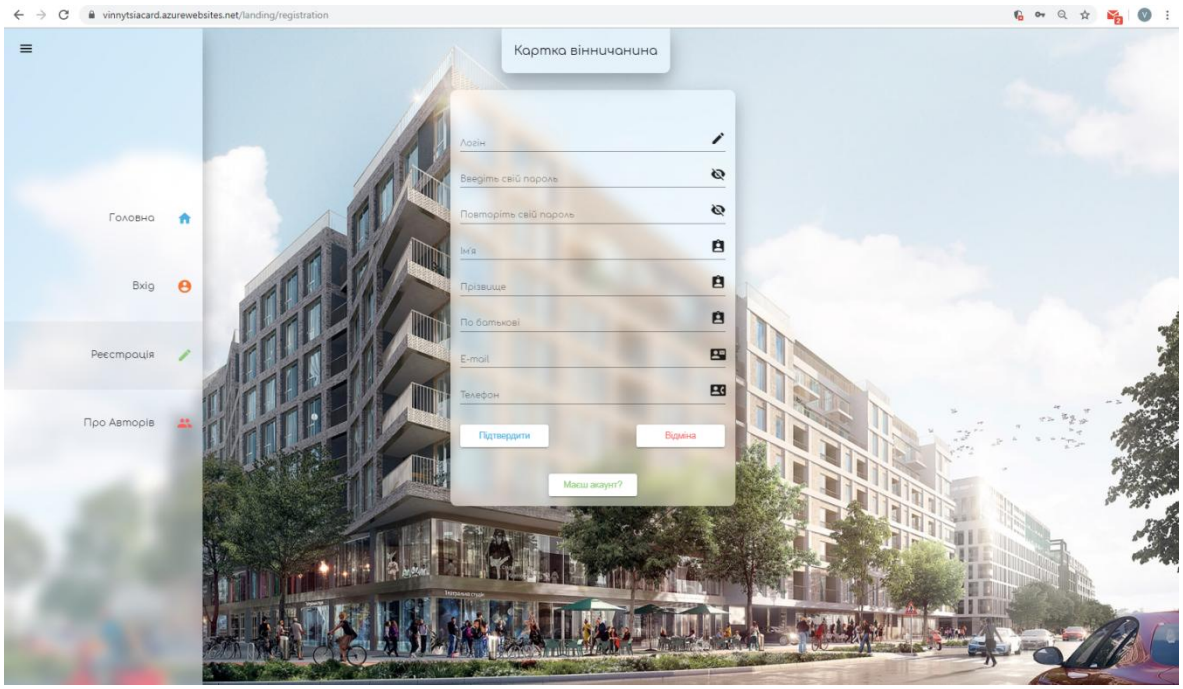


Figure 4.3. – Notification when user data is not correct

If the user data is correct, then by clicking the «Увійти» button, the user will be redirected to «Головна» page, which is shown in Figure 4.4.

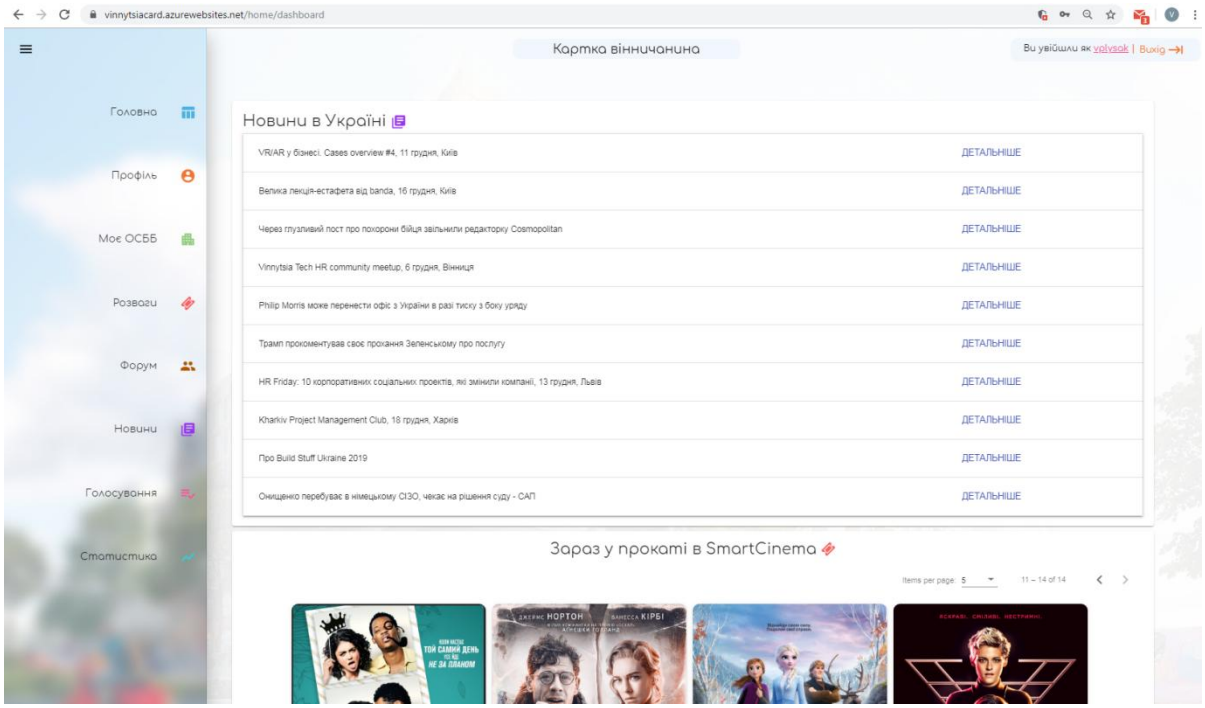


Figure 4.4 – The main page of information system

User page contains user data and information about logins. User page is shown on Figure 4.5.

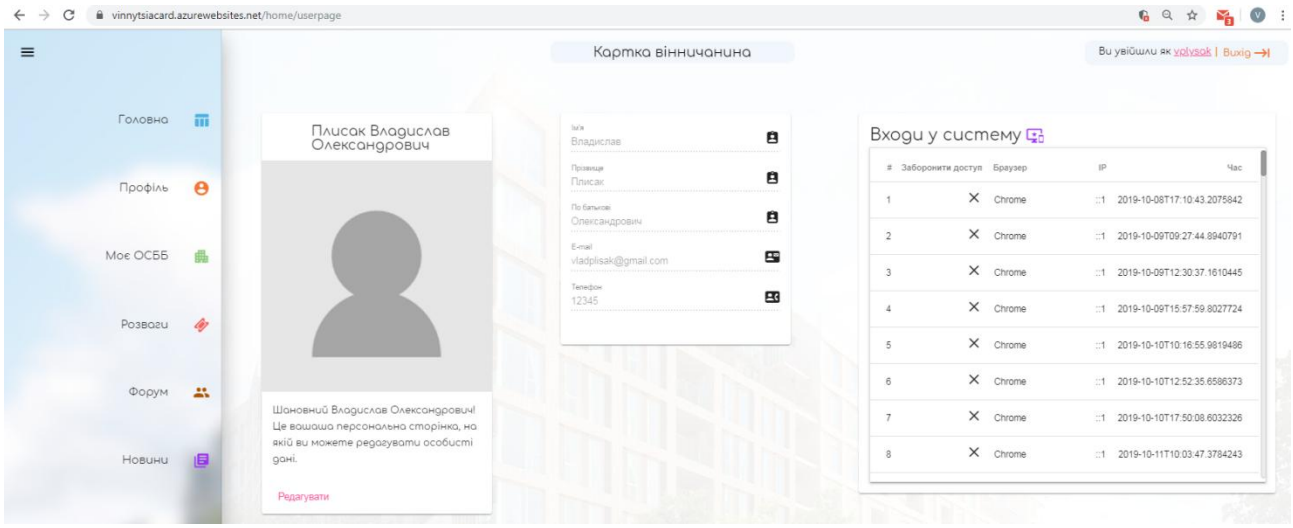


Figure 4.5 – User page of information system

For changing users data, user should click on «Редагувати» button. Then form with user's data will be editable. Editable form is shown on Figure 4.6.

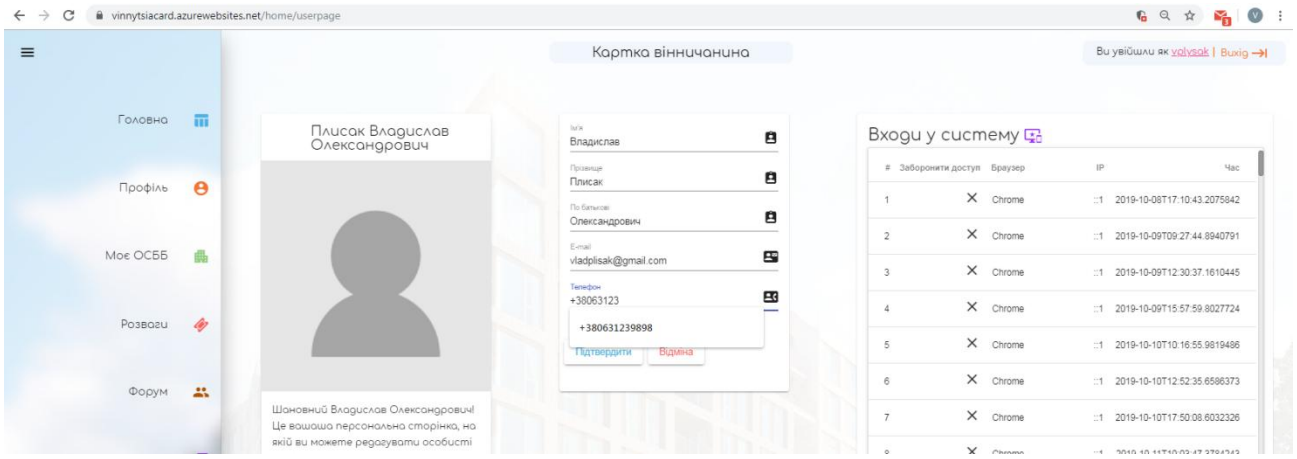


Figure 4.6 – Updating users data

After updating user's data, user will popup message, about updating. Result of updated data is shown on Figure 4.7.

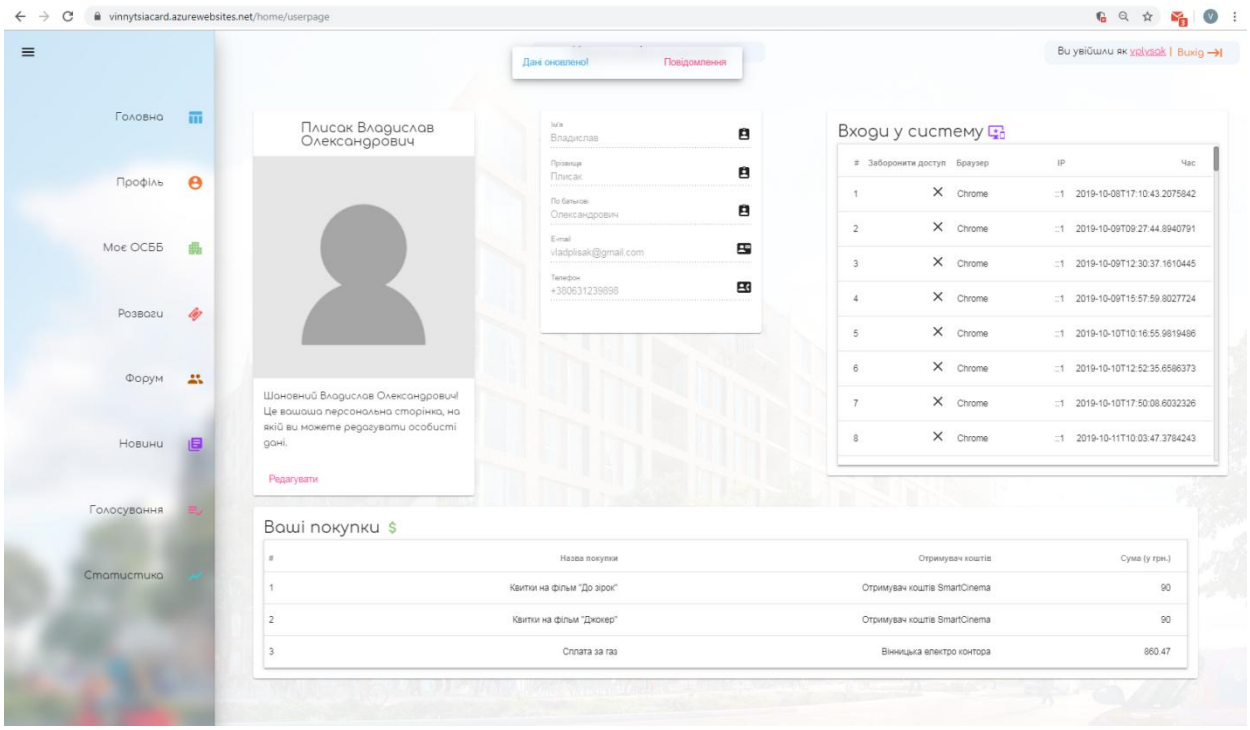


Figure 4.7 –Result of updating users data

For paying bills or additional payments, user should go to «Моє ОСББ». Web page of ACAB of user is shown on Figure 4.8.

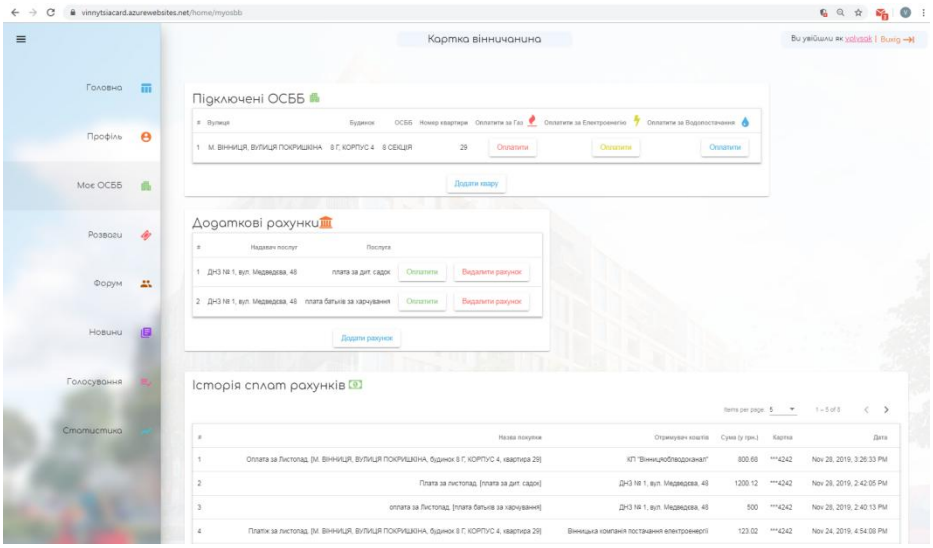


Figure 4.8 – Web page of ACAB of user

For adding apartments, user must click on «Додати квартиру». After clicking on button, user will see model form with existing ACABs. Modal form for adding apartment is shown on Figure 4.9.

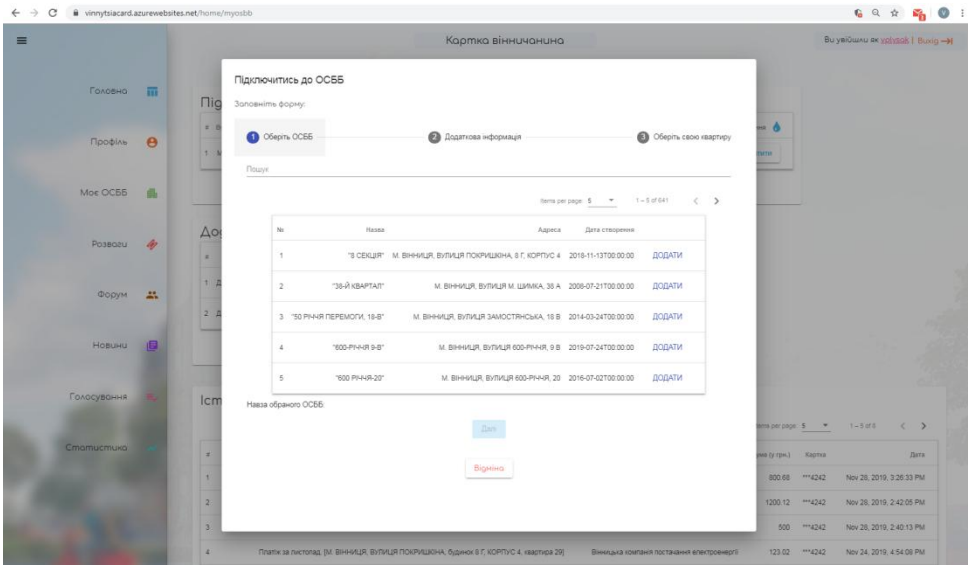


Figure 4.9 – Modal form for adding apartment

If no one have added ACAB into the information system, modal form will require to set count of apartments. Second step of modal form for adding apartment is shown on Figure 4.10.

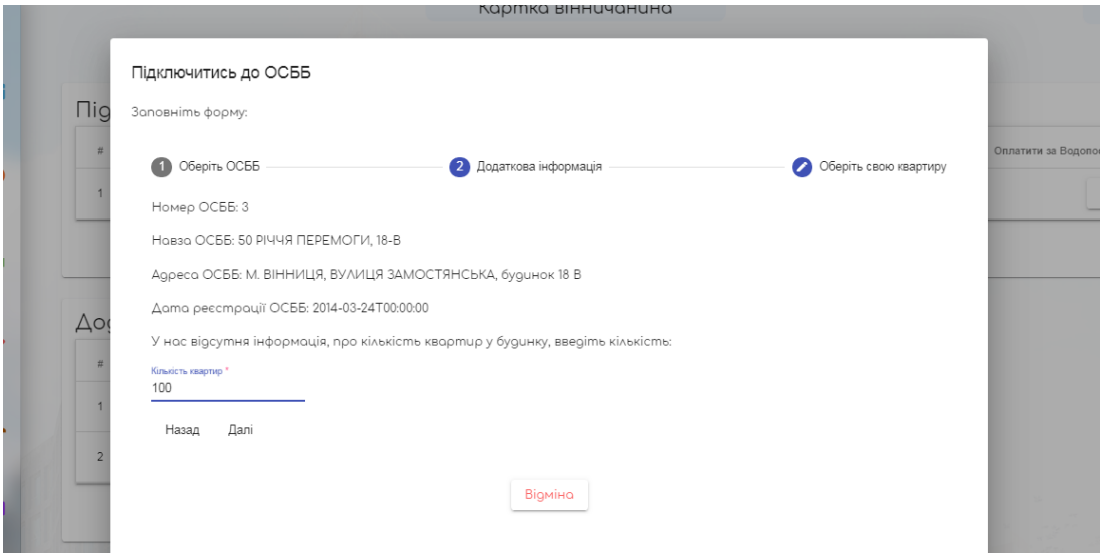


Figure 4.10 – Second step of modal form for adding apartment

Third step requires selecting users number of apartment. Selecting users number of apartment is shown on Figure 4.11.

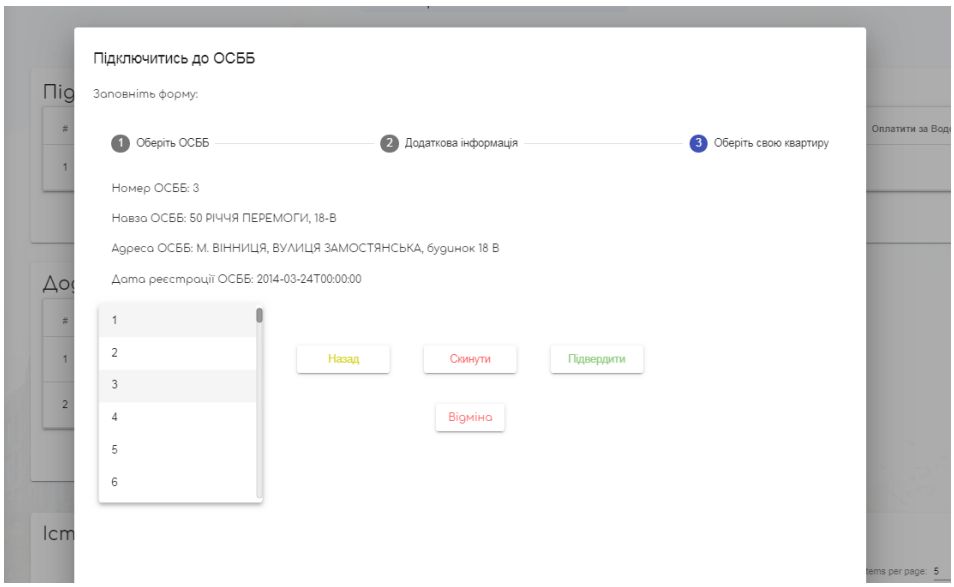


Figure 4.11 – Third step of modal form for adding apartment

If ACAB already has been added to information system, modal form contains only two steps. Modal form with two steps is shown on Figure 4.12.

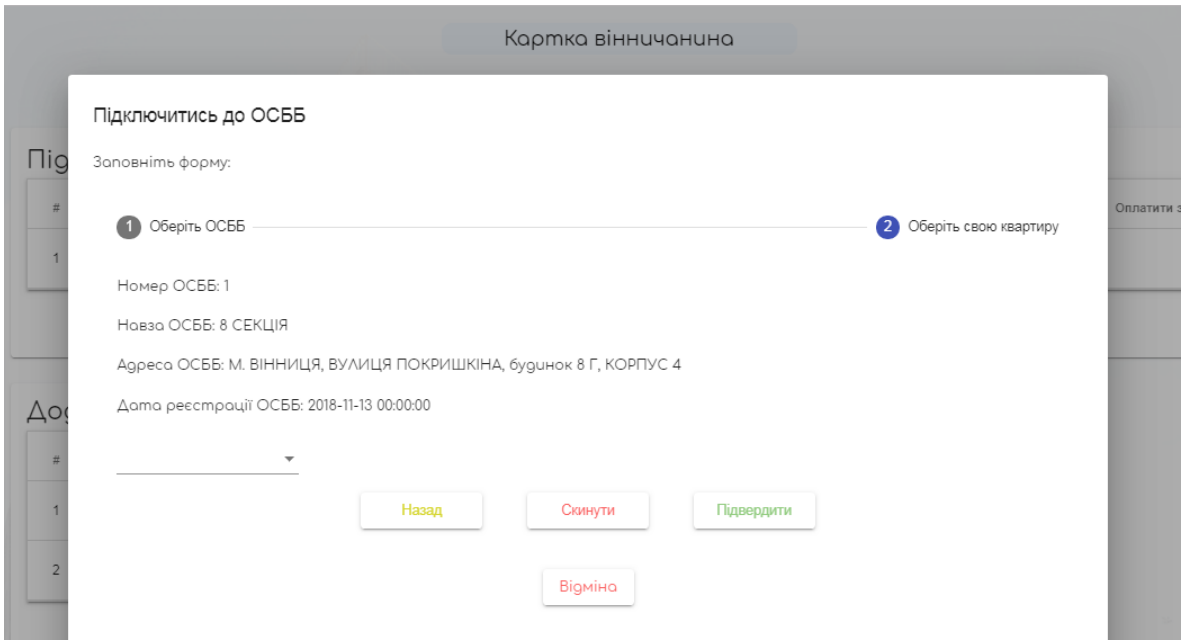


Figure 4.12 – Last step of modal form for adding apartment of existing ACAB

For adding additional bills, user must click on «Додати рахунок» button. It will open modal form for adding additional bills. Modal form for adding additional bills is shown on Figure 4.13.

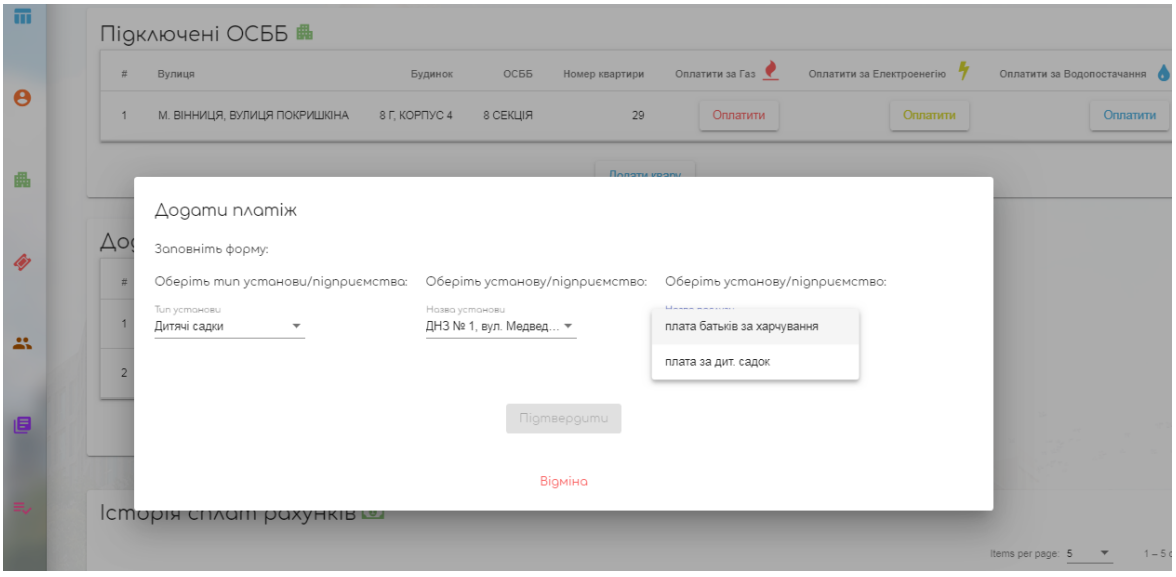


Figure 4.13 – Modal form for adding additional bills

For paying bill, user must click on «Оплатити» button near bill in table for main bills or additional bills. Modal form for paying bills is shown on Figure 4.14 and 4.15.

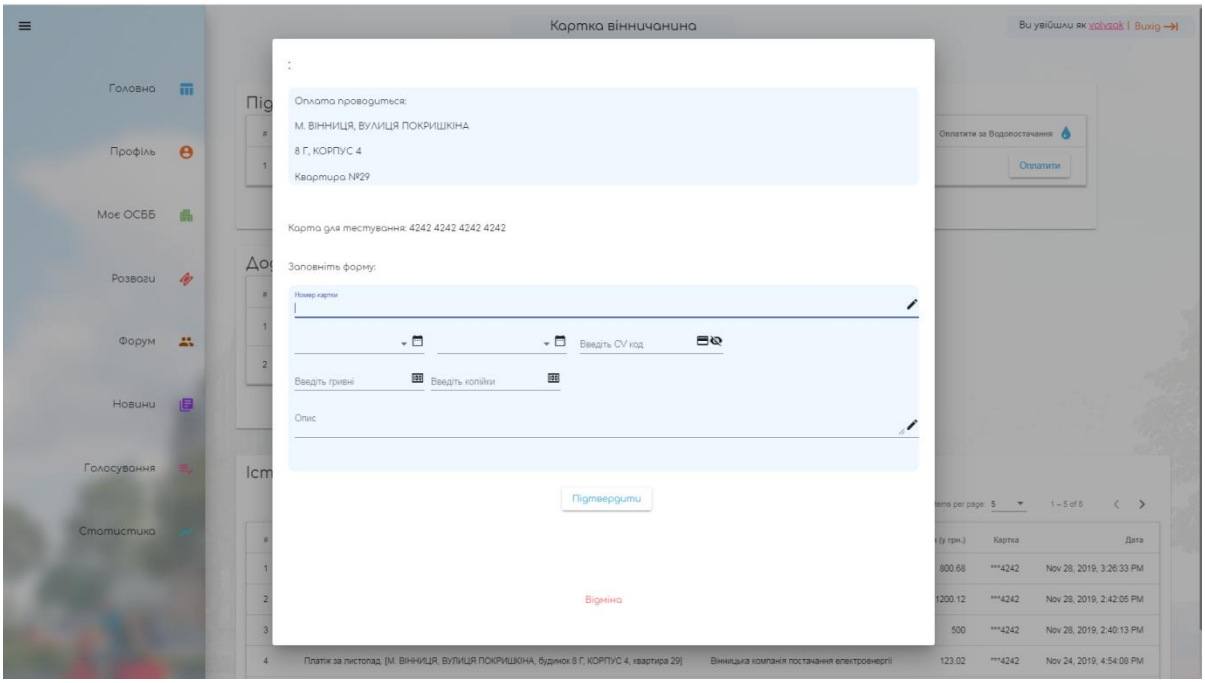


Figure 4.14 – Modal form of paying main bills

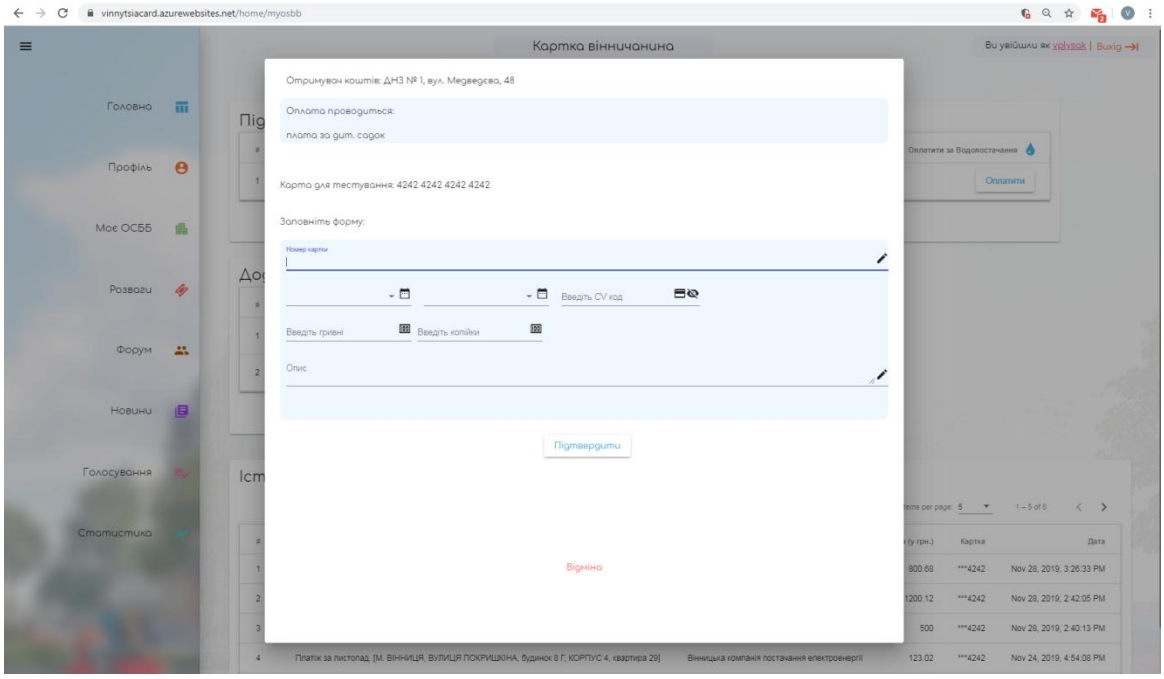


Figure 4.15 – Modal form of paying additional bills

If user will write correct data, information system will see his payment in table of history of payments. Successful pays are shown on Figure 4.16.

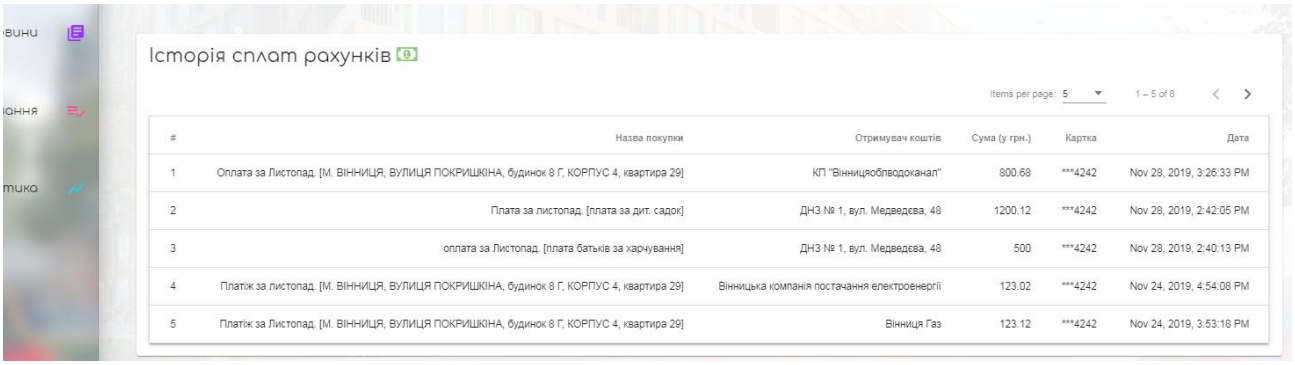


Figure 4.16 – Table of history of payments

If user click on «Розваги» menu item, he will see two widgets with films. First widget shows only available films, second widget shows only films that will be available soon. Widgets with films are shown on Figure 4.17 and 4.18.

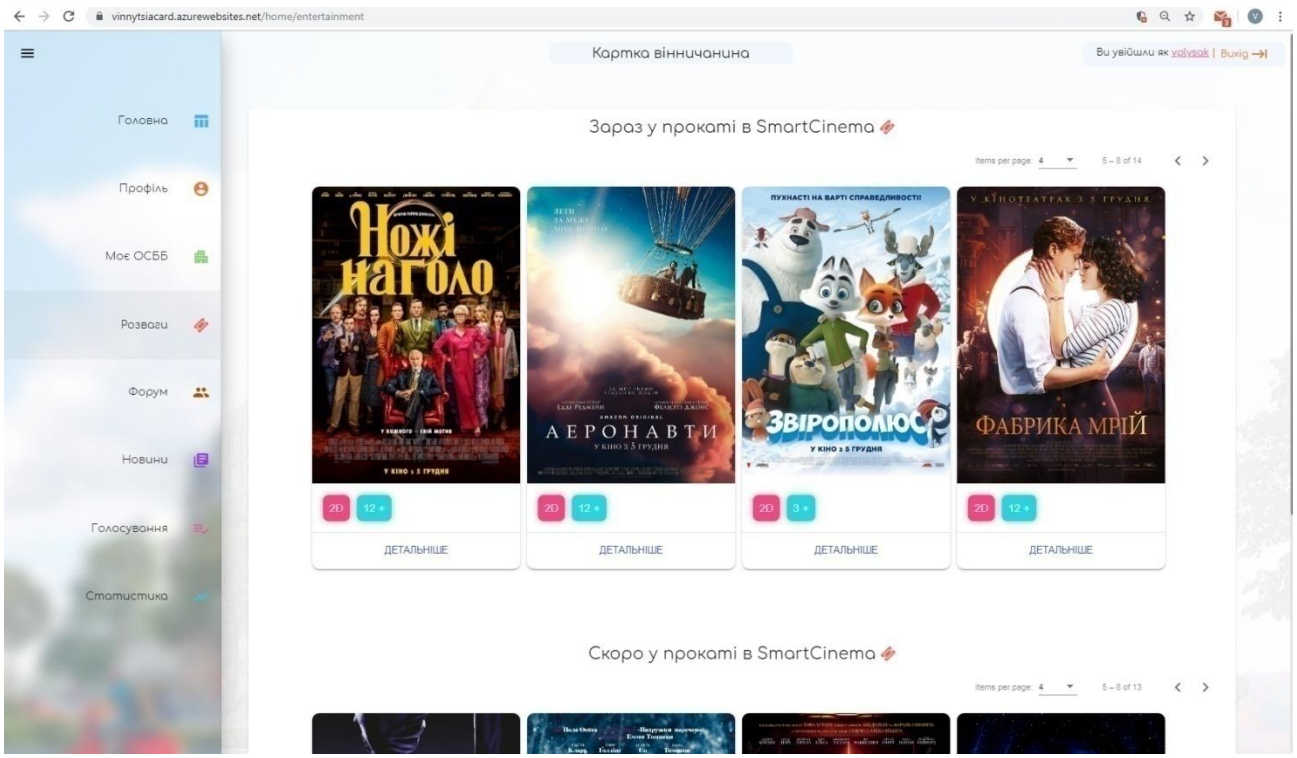


Figure 4.17 – Widget with available films

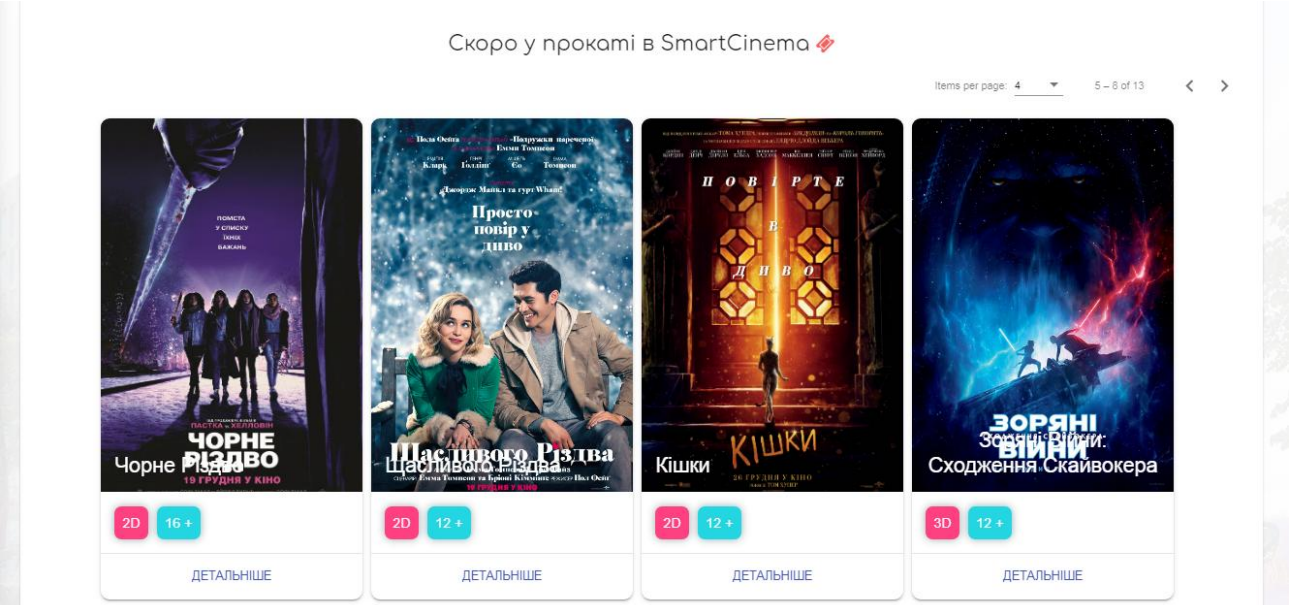


Figure 4.18 – Widget with films that will be available soon

If user clicks on «Детальніше» button, he will be redirected to web page of selected movie. Result of clicking on «Детальніше» button is shown on Figure 4.19.

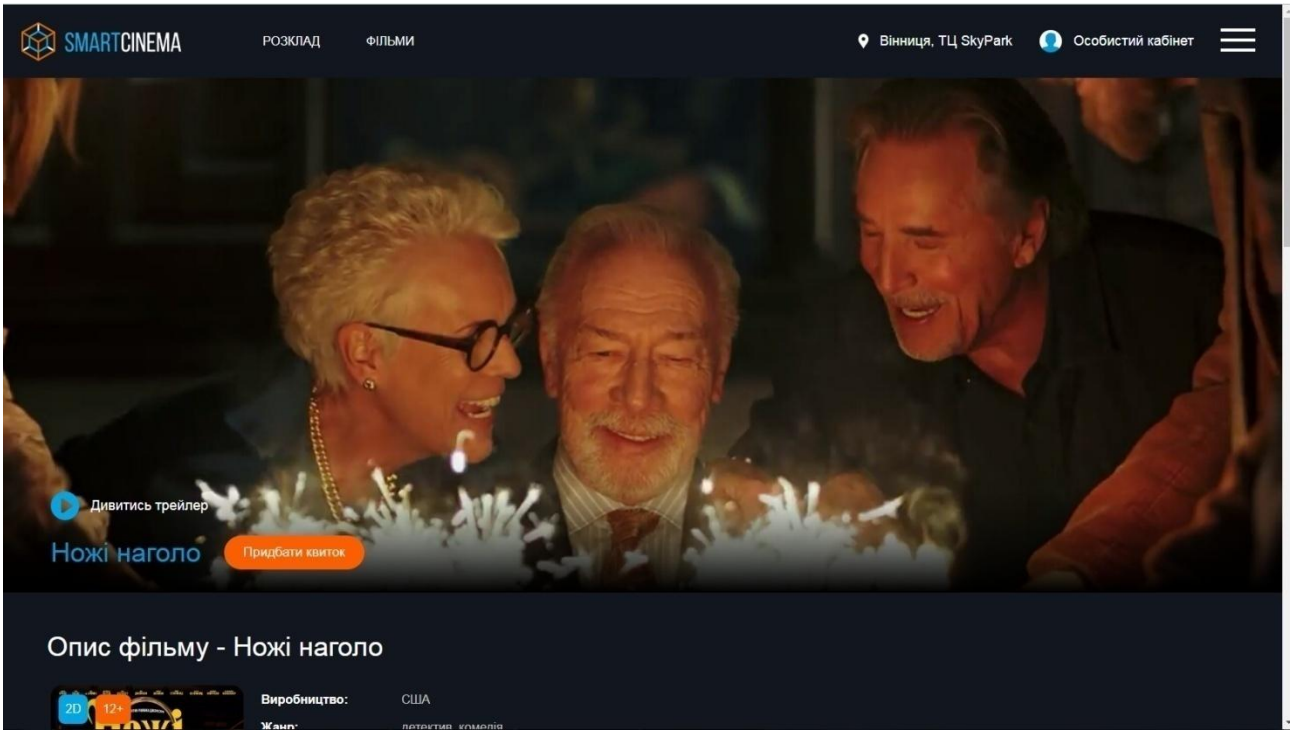


Figure 4.19 – Result of clicking on «Детальніше» button

If user clicks on «Новини» menu item, he will see news about Ukraine. News web page is shown on Figure 4.20 – 4.22.

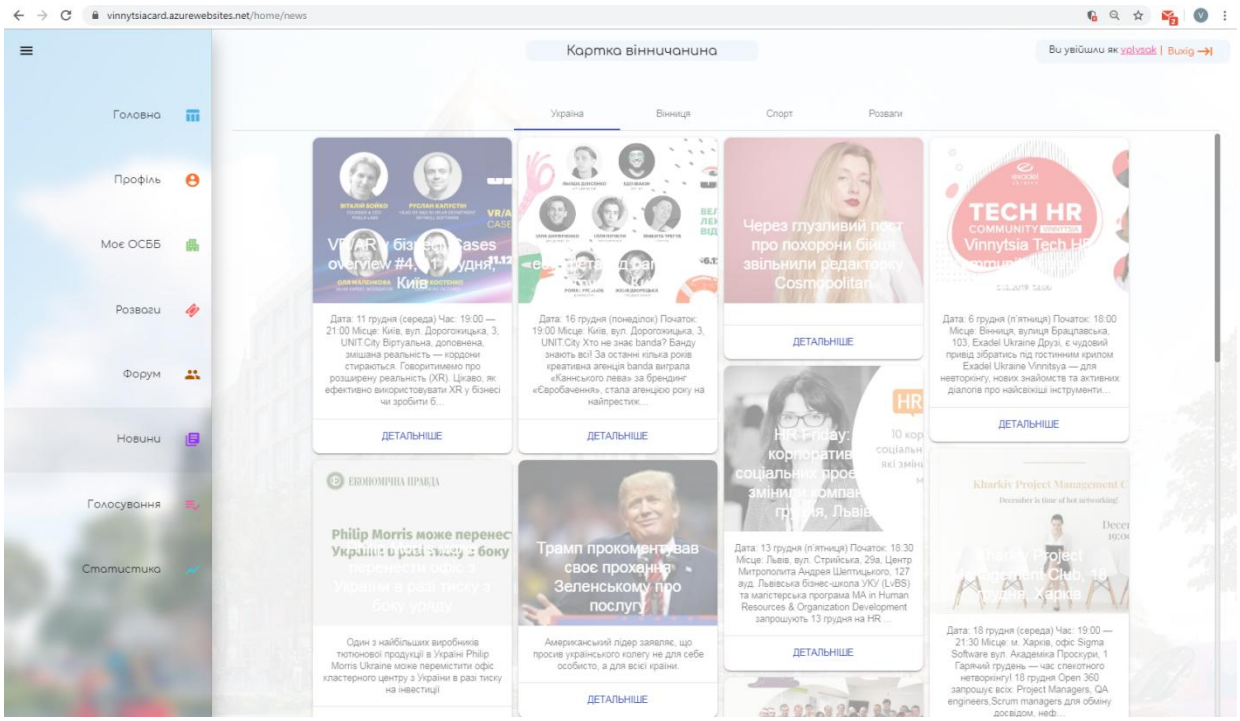


Figure 4.20 – News about Ukraine

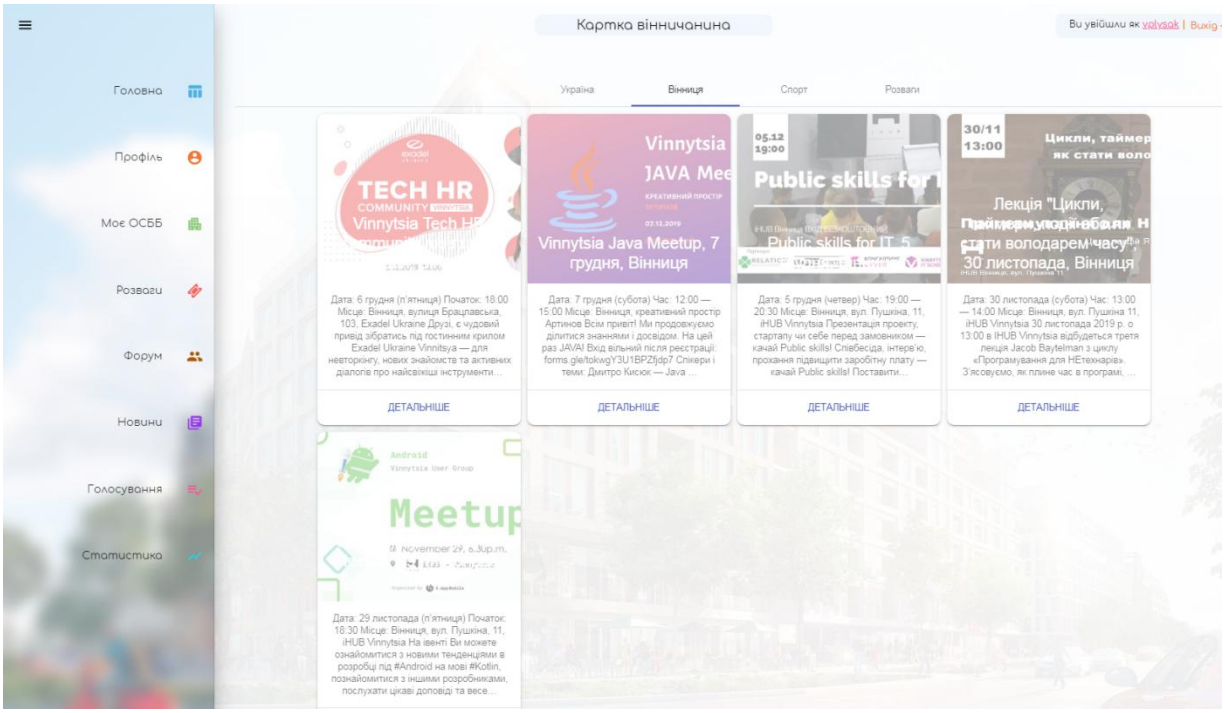


Figure 4.21 – News about Vinnytsia

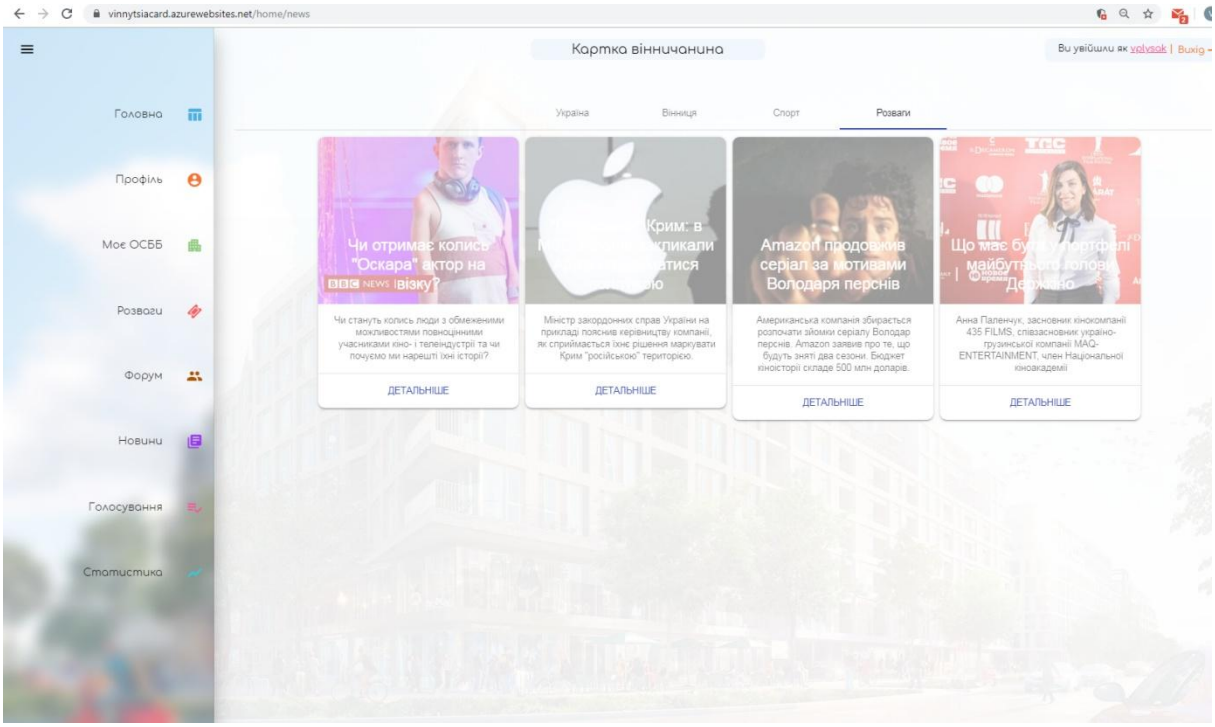


Figure 4.22 – News about Entertainment

If user clicks on «Голосування» menu item, he will see ideas of users ACABs and petitions. News web page is shown on Figure 4.23.

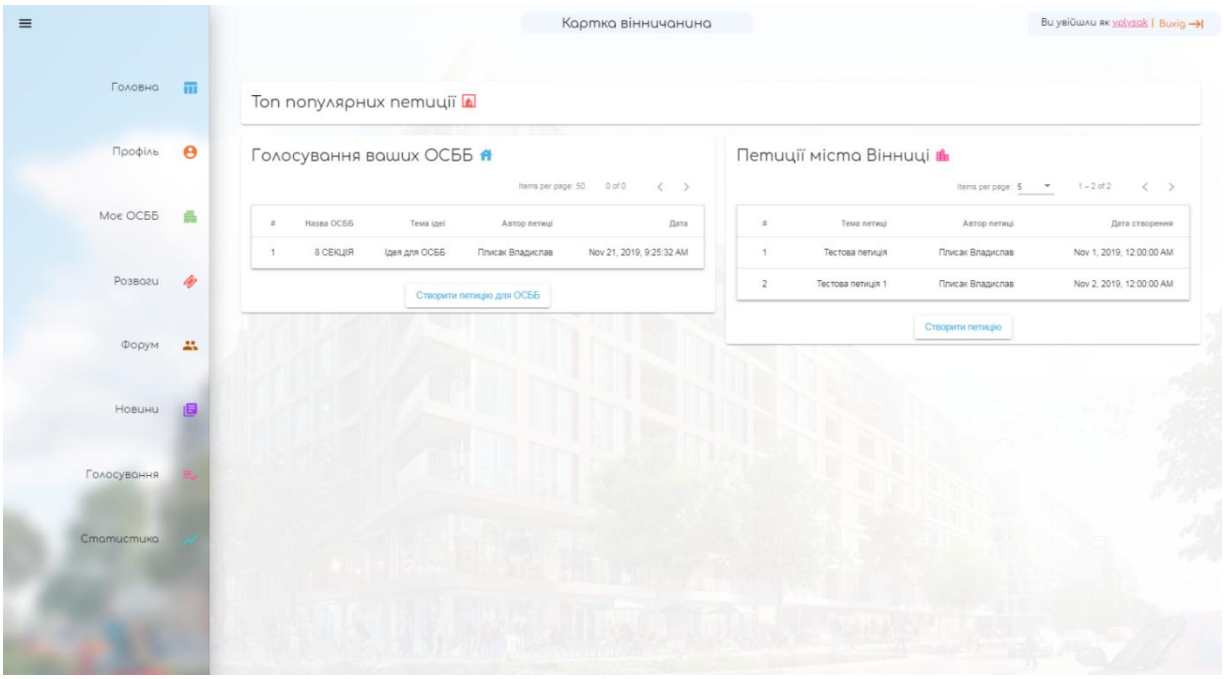


Figure 4.23 – Web page of petitions and ideas.

If user clicks on some idea, he will be redirected to web page of selected idea. Web page of idea is shown on Figure 4.24.

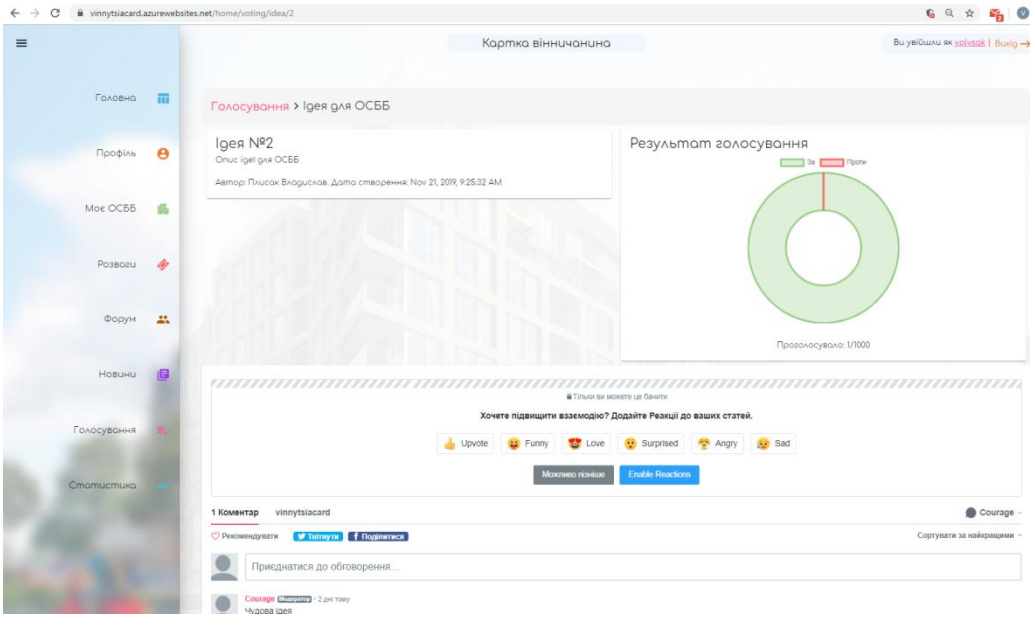


Figure 4.24 – Web page of idea

If user clicks on some petition, he will be redirected to web page of selected idea. Web page of idea is shown on Figure 4.25.

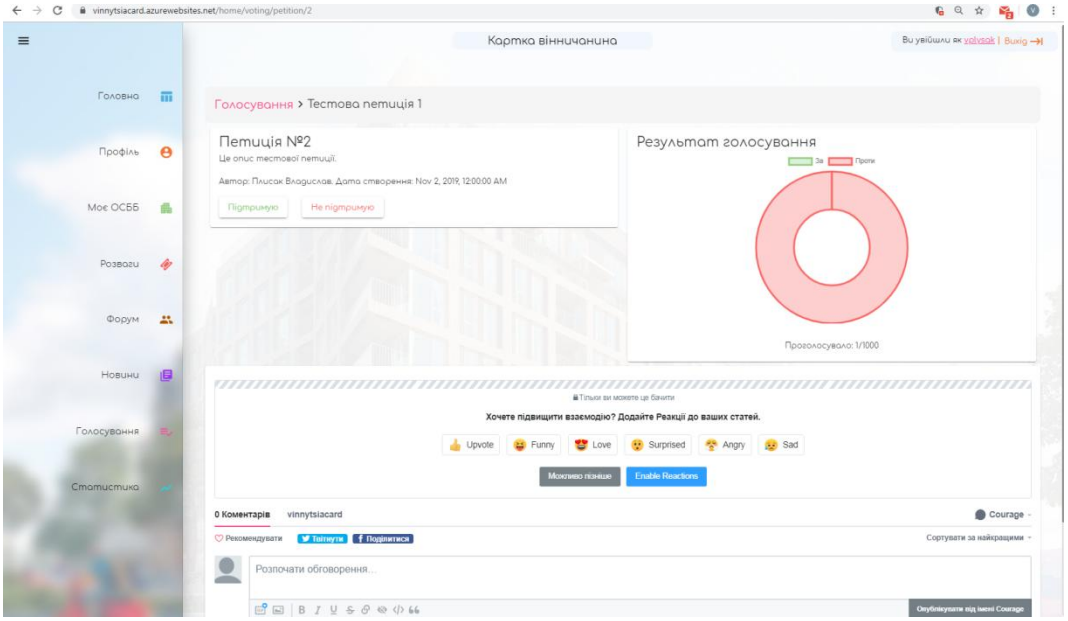


Figure 4.25 – Web page of petition

User can write some comment, below petition data. Comment is shown on Figure 4.26.

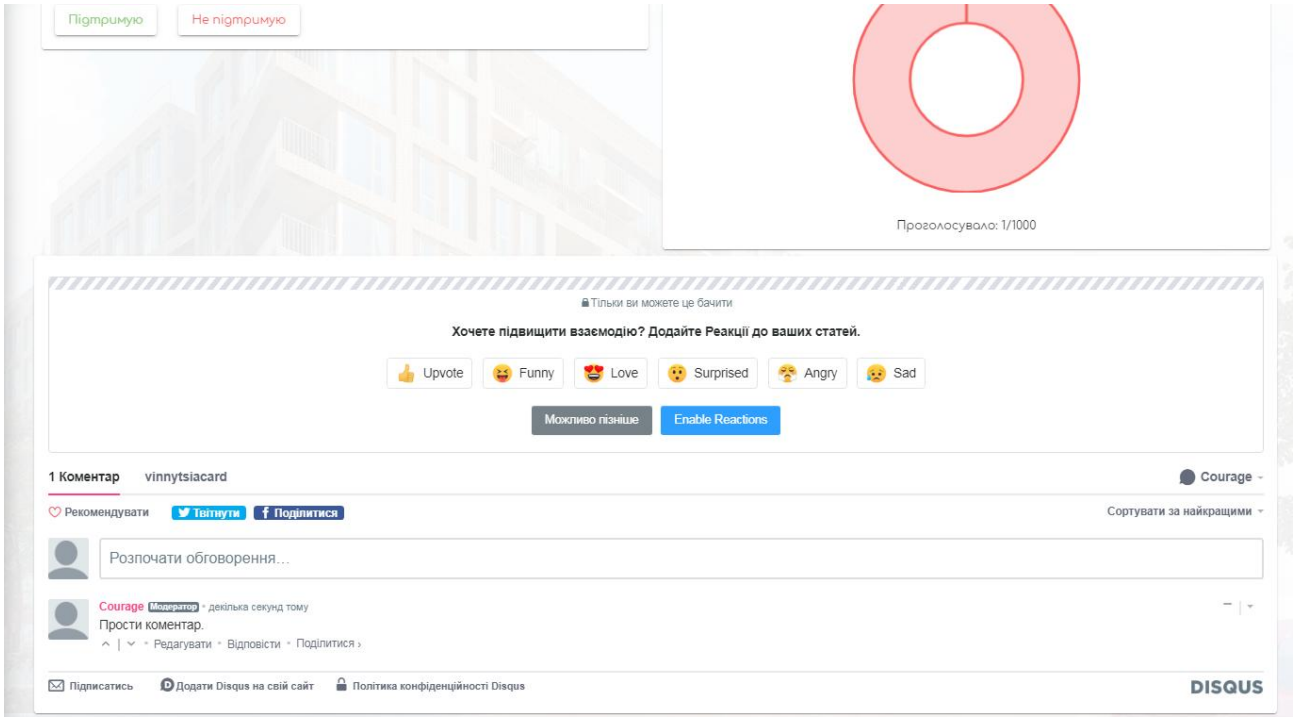


Figure 4.26 – Comments of petition

If user clicks on «Підтримую» or «Не підтримую» button, he will vote for petition. After voting, petition page will not contain these buttons. Web page of petition which has been voted is shown on Figure 4.27.

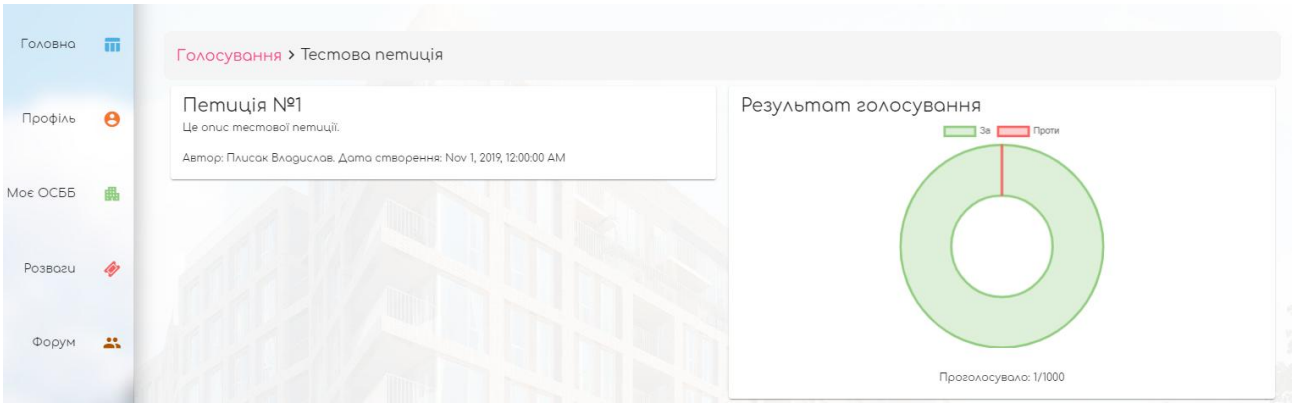


Figure 4.27 – Voted petition

If user clicks on «Статистика» menu item, he will see charts will see charts about diverse topics. Chart of statistic are shown on Figures 4.28 – 4.31.

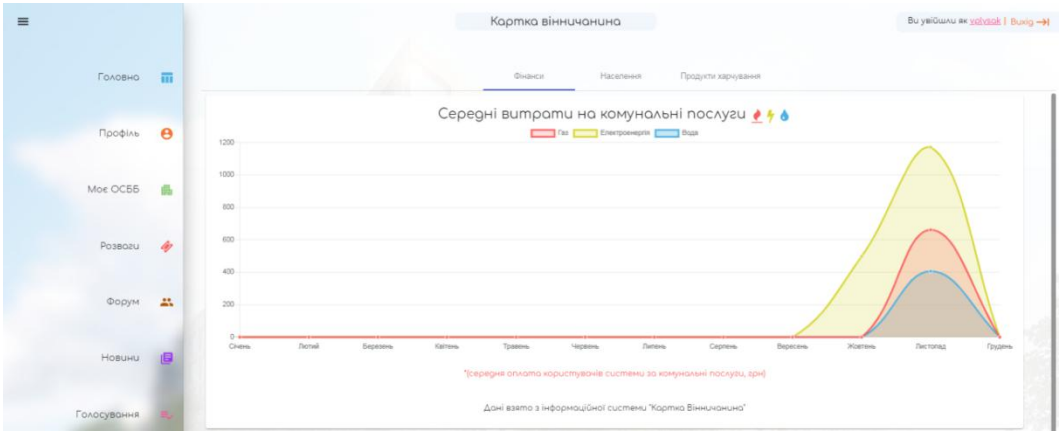


Figure 4.28 – Chart of average main payments

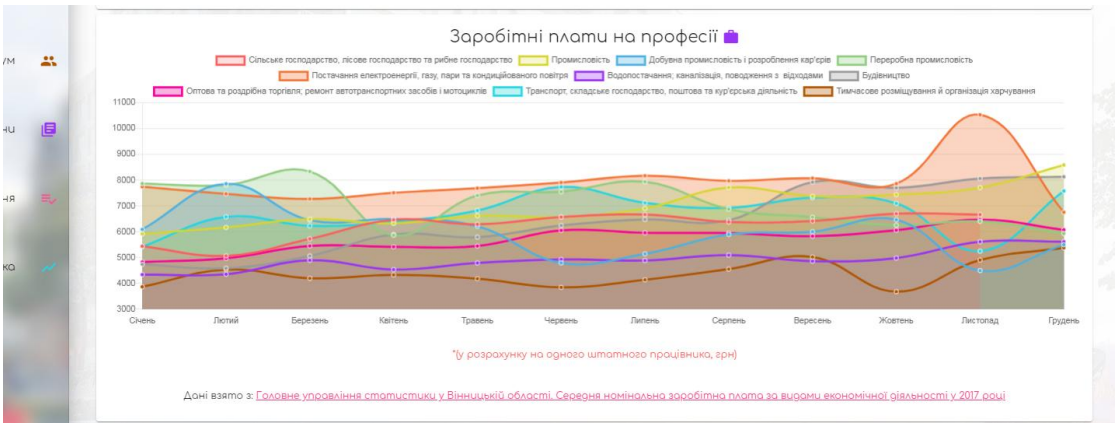


Figure 4.29 – Chart of salaries of each professions types

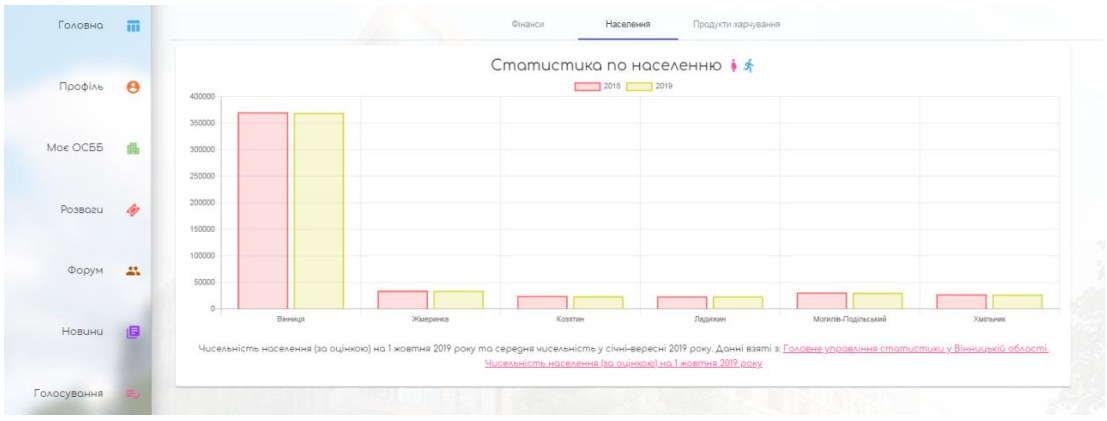


Figure 4.30 – Chart of population

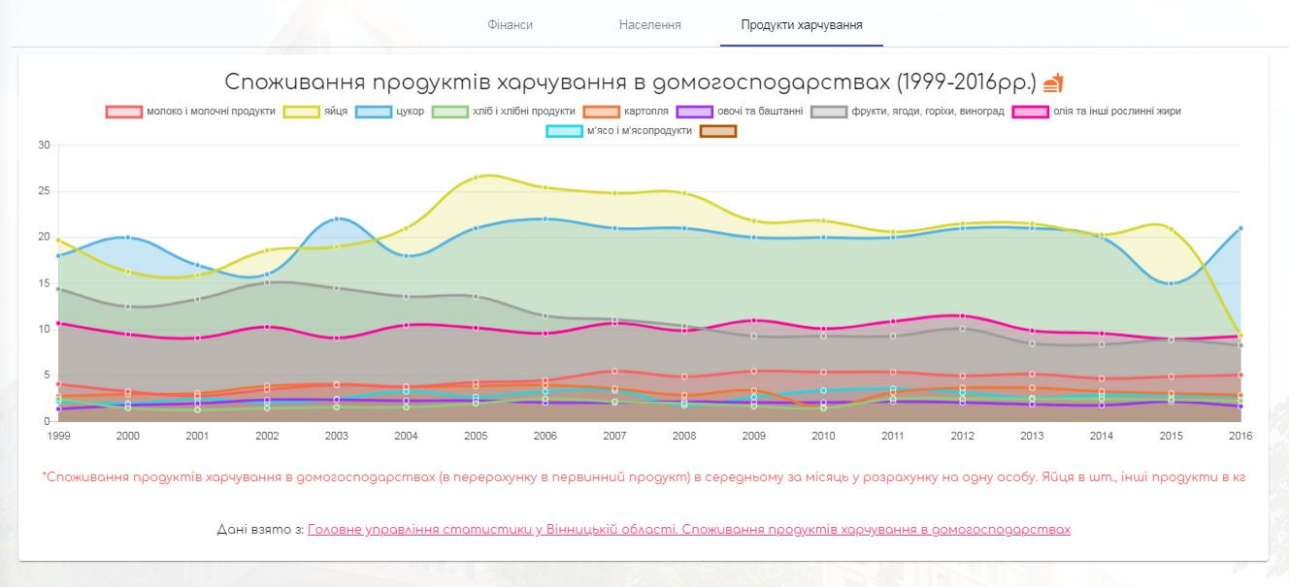


Figure 4.31 – Chart of consumption of food in households

If user clicks on «Форум» menu item, he will see list of topics. Forum page is shown on Figure 4.32.

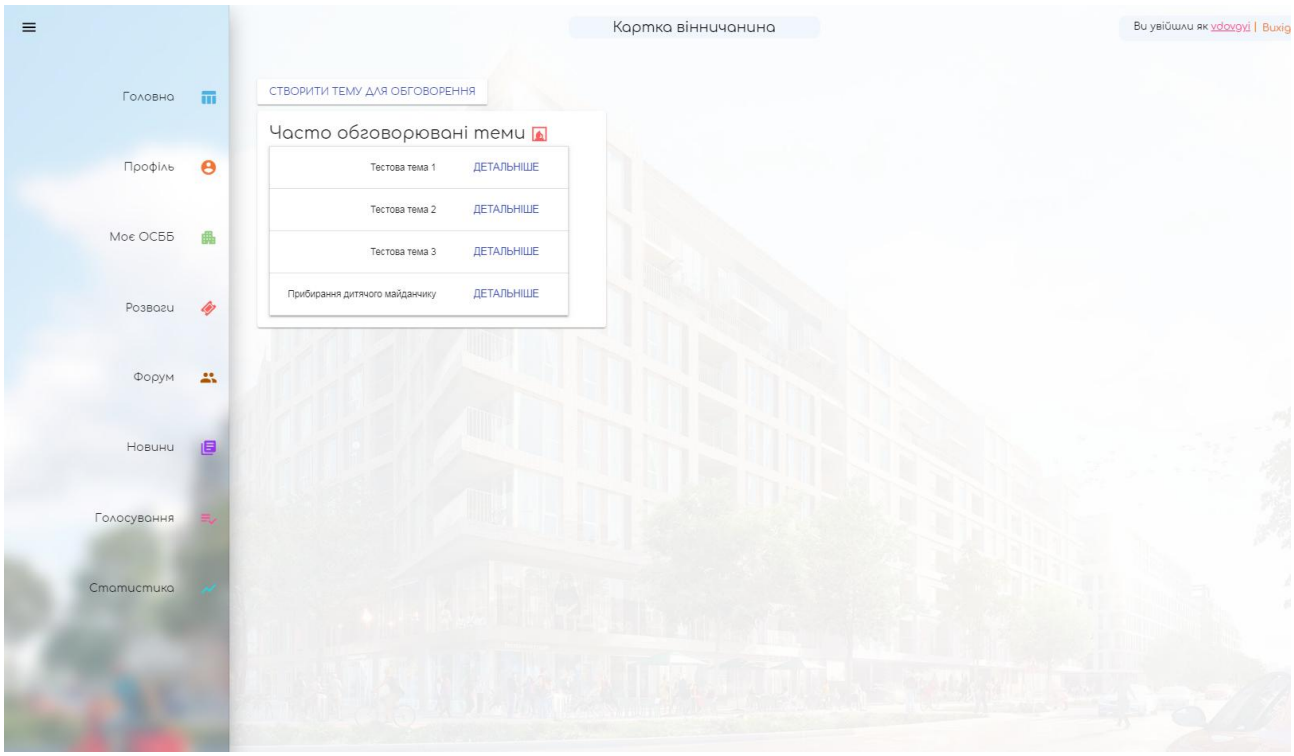


Figure 4.32 – Forum page

If user clicks on topic button «Детальніше», information system will redirect to topic conversation. Topic conversation is shown on Figure 4.33.

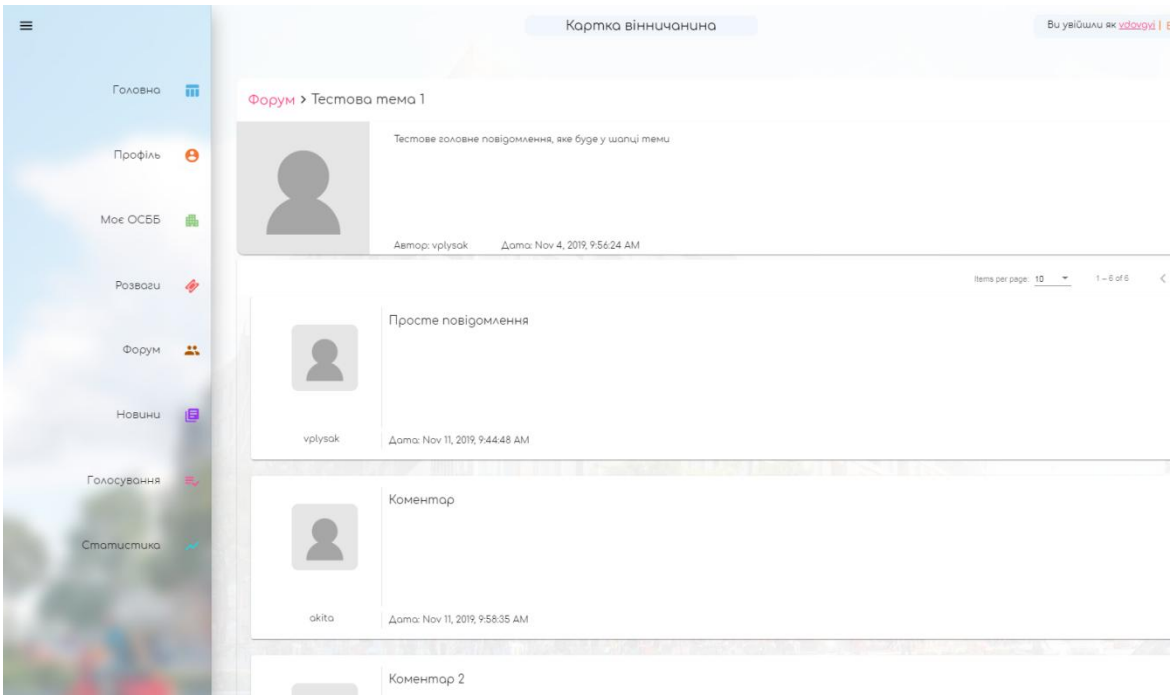


Figure 4.33 – Topic conversation

If user clicks on «Створити тему для обговорення», it will open modal form for creating form. Modal form for creating forum topic is shown on Figure 4.34.

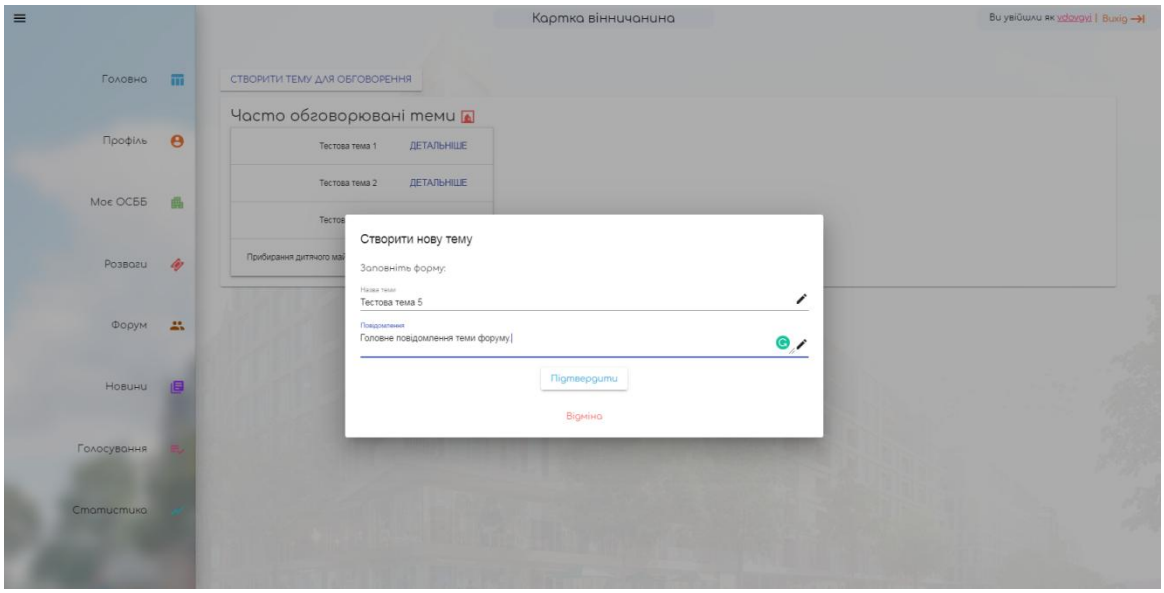


Figure 4.34 – Modal form for creating forum topic

CONCLUSIONS

During the master's work, the development of the information system "Personal office of the resident of Vinnitsa" was developed. Information work has a clear and user-friendly interface and all the necessary functionality for interaction with state and non-state enterprises.

The first section analyzes the information system of the subsystem of interaction of residents in the automated control system of joint owners of an apartment building, analyzes the existing analogues, describes the features of such subsystems and describes the use of data processing tools in the subsystem. The elaborated statement of the problem is developed.

In the second section, the functions of the subsystem "Personal cabinet of the resident of Vinnitsa" to be developed and its modeling were analyzed. On the basis of these data, a structural diagram of the system, a UML sequence diagram, a UML activity diagram was developed.

In the third section, the software was developed in Microsoft Visual Studio Code. The technologies used, their advantages and disadvantages are described.

The fourth section tested the software that showed the correct operation of the information system "Personal office of the resident of Vinnitsa" for users who are connected to the merger and for users who have not yet connected to the merger.

After testing, it was found that the information system meets all requirements and can be used by residents of Vinnitsa.

REFERENCES

1. Aries OI What is an ACS / OI Aries. M .: Science, 1981.
2. Industry-wide guidance methodological materials for the creation of automated systems for managing enterprises and production associations (ACM). M .: Statistics, 1977; Industry-wide guidance methodical materials but the creation of multi-level integrated ACS production association (enterprise). M .: DCST, 1986.
3. A Theory of System Interaction: Components, Interfaces, and Services [Electronic resource]: [Website] - Electronic data. - Access mode: https://link.springer.com/chapter/10.1007/3-540-34874-3_4 – September 2019.
4. M. Abadi, L. Lamport: The Existence of Refinement Mappings. Digital Systems Research Center, SRC Report 29, August 1988.
5. M. Abadi, L. Lamport: Composing Specifications. Digital Systems Research Center, SRC Report 66, October 1990.
6. L. Aceto, M. Hennessy: Adding Action Refinement to a Finite Process Algebra. Proc. ICALP 91, LNCS 510, Springer 1991, 506–519.
7. P. Andrews: An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof. Computer Science and Applied Mathematics. Academic Press 1986.
8. R.J.R. Back: Refinement Calculus, Part I: Sequential Nondeterministic Programs. REX Workshop. In: J. W. deBakker, W.-P. deRoever, G. Rozenberg (eds): Stepwise Refinement of Distributed Systems. LNCS 430, Springer 1989, 42–66 // R.J.R. Back: Refinement Calculus, Part II: Parallel and Reactive Programs. REX Workshop. In: J. W. de Bakker, W.-P. de Roever, G. Rozenberg (eds): Stepwise Refinement of Distributed Systems. LNCS 430, Springer 1989, 67–93.

9. Article about Single Sign-On [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/Single_sign-on – September 2019.
10. Site with information about «KYIV ID» system [Electronic resource]: [Website] - Electronic data. - Access mode: https://www.kyivsmartcity.com/projects/yedinij-oblikovij-zapis-kiyanina/ – September 2019.
11. Site with information about «KYIV ID» system [Electronic resource]: [Website] - Electronic data. - Access mode: https://kiev.vgorode.ua/news/sobytyia/395318-v-dva-klyka-kakye-problemy-mozhno-reshyt-s-pomoschui-Kyiv-ID – September 2019.
12. Site with information about «Особистий кабінет львів'янина» system [Electronic resource]: [Website] - Electronic data. - Access mode: https://city-adm.lviv.ua/policy – September 2019.
13. Article about SharePoint [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/SharePoint – September 2019.
14. ІнфоЛьвів Portal [Electronic resource]: [Website] - Electronic data. - Access mode: https://infolviv.com.ua/ – September 2019.
15. Article about automated control systems [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/Industrial_control_system – September 2019.
16. Identification and modeling of technological objects and control systems: a textbook / VM Dubovoy. - Vinnytsia: VNTU, 2012. – 308 c
17. Article on Automated Control Systems [Electronic resource]: [Website] - Electronic data. - Access mode: https://dic.academic.ru/dic.nsf/ruwiki/312433#.D0.9E.D1.81.D0.BD.D0.BE.D0.B2.D0.BD.D1.8B.D0.B5_.D0.BA.D0.BB.D0.B0.D1.81.D1.81.D0.B8.D1.84.D

0.B8.D0.BA.D0.B0.D1.86.D0.B8.D0.BE.D0.BD.D0.BD.D1.8B.D0.B5_.D0.BF.D1.80.D0.B8.D0.B7.D0.BD.D0.B0.D0.BA.D0.B8 – September 2019.

18. Article on the Functional Structure of ACS [Electronic resource]: [Website] - Electronic data. - Access mode: http://studbooks.net/1160897/informatika/opisanie_funktsionalnoy_struktury – September 2019.
19. ACS Technical Support Article [Electronic resource]: [Website] - Electronic data. - Access mode: http://www.teh-lib.ru/atpip/sostav-asu-tp/Tehnicheskoe-matematicheskoe-i-programmnoe-obespechenie-ASU-TP.html – September 2019.
20. DSTU 2938-94 Information Processing Systems. Basic concepts. [Electronic resource]: [Website] - Electronic data. - Access mode: http://document.ua/sistemi-obroblennja-informaciyi_-osnovni-ponjattja_-termini--std1033.html – September 2019.
21. ISO / IEC25010: 2011 [Electronic resource]: [Website] - Electronic data. - Access mode: https://www.iso.org/standard/35733.html – September 2019.
22. Kudryavtseva SP, Kolos VV, Textbook - K .: Slovo Publishing House, 2005. – 400 pg.
23. Wikipedia article about a three-tier client-server architecture [Online resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/ – September 2019.
24. AN Petrov Computer analysis of text: historiography of the method // Circle of ideas: models and technologies of historical informatics. M., 1996.
25. George A. Numerical solution of large sparse systems of equations. - M .: Mir, 1984. – 287pg.
26. Zenkevich O., Chang I. Finite element method in the theory of structures and in the mechanics of continuous media. - M .: Nedra, 1974. - 240 pg.
27. Figurnov VE "Information Technologies", M, 2000.

28. Wikipedia article about Microsoft SQL Server 2017 [Online resource]: [Website] - Electronic data. - Access mode https://en.wikipedia.org/wiki/Microsoft_SQL_Server – September 2019.
29. Microsoft SQL Server 2017 Official Documentation [Electronic resource]: [Website] - Electronic data. - Access mode: https://docs.microsoft.com/en-us/sql/sql-server/install/planning-a-sql-server-installation?view=sql-server-2017 – September 2019.
30. Transact-SQL Wikipedia article [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/Transact-SQL – September 2019.
31. Microsoft Official Documentation on LINQ to Entities [Web site]: [Web site] - Electronic data. - Access mode: https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/ef/language-reference/linq-to-entities – September 2019.
32. Wikipedia article about UML [Electronic resource]: [Website] - Electronic data. - Access mode: https://ru.wikipedia.org/wiki/UML – September 2019.
33. Dubovoy V.M. Modeling of processes and control systems. Tutorial. / Dubovoy VM, Moskvina SM, Nikitenko OD - Vinnitsa: VNTU, 2009. - 103 p.: Modeling of processes and control systems.
34. Wikipedia article about .NET Framework [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/.NET_Framework – September 2019.
35. Wikipedia article about .NET Core [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/.NET_Core – September 2019.
36. Wikipedia article about TypeScript [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/TypeScript – September 2019.

37. Official documentation for the use of the library Chart.JS [Electronic resource]: [Website] - Electronic data. - Access mode: https://www.chartjs.org/ – September 2019.
38. Official documentation for the use of Material Design Lite [Electronic resource]: [Website] - Electronic data. - Access mode: https://getmdl.io/ – September 2019.
39. UML Use Case Diagrams: Guidelines – MSDN – Microsoft_[Electronic resource]: [Website] - Electronic data. - Access mode: https://msdn.microsoft.com/en-us/library/dd409432.aspx – September 2019.
40. UML recomandation s[Electronic resource]: [Website] - Electronic data. - Access mode: https://msdn.microsoft.com/ru-ru/library/dd409465.aspx – September 2019.
41. Wikipedia article about Data transfer object [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/DTO – November 2019.
42. Wikipedia article about Data transfer object [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/DTO – November 2019.
43.
44. Wikipedia article about .Net Core [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/.NET_Core – November 2019.
45. Landwerth, Immo (4 December 2014). "Introducing .NET Core". .NET Framework Blog. Microsoft. Retrieved 27 February 2015.
46. Wikipedia article about MS SQL Server [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/Microsoft_SQL_Server – November 2019.

47. Entity Framework Core documentation [Electronic resource]: [Website] - Electronic data. - Access mode: https://docs.microsoft.com/ru-ru/ef/core/ – November 2019.
48. Wikipedia article about Entity Framework [Electronic resource]: [Website] - Electronic data. - Access mode: https://ru.wikipedia.org/wiki/ADO.NET_Entity_Framework – November 2019.
49. Wikipedia article about TypeScript [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/TypeScript – November 2019.
50. Wikipedia article about Angular 2+ [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/Angular_(web_framework) – November 2019.
51. Wikipedia article about Material Design [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/Material_Design – November 2019.
52. Wikipedia article about Azure [Electronic resource]: [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/Microsoft_Azure – November 2019.
53. Wikipedia article about stored procedure [Electronic resource]: [Website] - Electronic data. - Access mode: https://en.wikipedia.org/wiki/Stored_procedure – November 2019.
54. Web portal of opened data [Electronic resource]: [Website] - Electronic data. - Access mode: https://opendata.gov.ua/ – November 2019.
55. Wikipedia article about JSON format [Website] - Electronic data. - Access mode: https://uk.wikipedia.org/wiki/JSON – November 2019.
56. Documentation of Stripe [Electronic resource]: [Website] - Electronic data. - Access mode: https://stripe.com/docs – November 2019.

57. Article about Top 10 News Api [Electronic resource]: [Website] - Electronic data. - Access mode: https://medium.com/rakuten-rapidapi/top-10-best-news-apis-google-news-bloomberg-bing-news-and-more-bbf3e6e46af6 – November 2019.
58. Piccoli, Gabriele; Pigni, Federico (July 2018). Information systems for managers: with cases (Edition 4.0 ed.). Prospect Press. p. 28. ISBN 978-1-943153-50-3. Retrieved 25 November 2018.
59. O'Hara, Margaret; Watson, Richard; Cavan, Bruce (1999). "Managing the three levels of change". Information Systems Management. 16
60. Agarwal, R., Sambamurthy, V., & Stair, R. (2000). Research report: The evolving relationship between general and specific computer self-efficacy – An empirical assessment. Infor-mation System Research.
61. Boudreau, M., & Robey, D. (2005). Enacting integrated information technology: A human agency perspective. Organization Science.
62. Chan, S., & Lu, M. (2004). Understanding internet banking: Adoption and use behavior. Journal of Global Information Management.
63. Compeau, D., & Higgings, C. (1995). Computer self-efficacy: Development of a measure and initial test. MIS Quarterly.
64. Compeau, D, Higgings, C., & Huff, S. (1999). Social cognitive theory and individual reactions to computing technology: A longitudinal study. MIS.
65. Davis, F. (1989). Perceived usefulness, perceived ease of use, and user acceptance of informa-tion technology. MIS Quarterly, 13(3), 319-340
66. Thompson, R., Compeau, D., & Higgings, C. (2006). Intentions to use information techno-logies: An integrative model. Journal of Organizational and End-User Computing.

APPENDIXES

Appendix A
(obligatory)

VNTU

APPROVED

Head of the AIIT department,

D. Sc. Prof.

_____R. Kvetny

“ ” 201__

TECHNICAL TASK

for the master's qualification work

«Development of the information system "Personal office of the resident of Vinnitsa". Part 3. Server and tools of subsystem»

08-01.MQW.00.00.001 TT

Supervisor: Phd., associate professor

O. O. Kovaliuk

“___” _____ 2019

Performer: st. of the group 3ACIT -18m

V. O. Plysak

“___” _____ 2019

Vinnytsia 2019

1. Name and field of application

1.1. Title – Development of the information system "Personal office of the resident of Vinnitsa". Part 3. Server and tools of subsystem.

1.2. Field of application – business.

2. The basis for the development.

Theme of master's qualification was approved by order VNTU № 254 from "2" 10 2019

3. Purpose and designation.

Establish an information system that would reduce the time spent on managing money for utilities and other expenses associated with cultural activities. Provide a complete set of tools to improve the interaction of residents with other institutions.

4. Initial data for development.

Master's qualification work is carried out for the first time. During the development, the following documents should be used:

1. ASP.NET Core in Action 1st Edition. Dustin Metzgar, Microsoft 2018.– 288c
2. Ідентифікація та моделювання технологічних об’єктів і систем керування : навчальний посібник / В. М. Дубовой. – Вінниця : ВНТУ, 2012. – 308 с.
3. Закон України «Про об’єднання співвласників багатоквартирного будинку» (Відомості Верховної Ради України (ВВР), 2002, N 10, ст.78) Із змінами, внесеними згідно із Законами N 3053-IV (3053-15) від 03.11.2005, ВВР, 2006, N 4, ст.60 N 2555-VI (2555-17) від 23.09.2010, ВВР, 2011, N 6, ст.41 Кодексом.

5. Requirements for development.

5.1. List of main functions:

- payment for utilities;

- payment for additional services;
- forum;
- signing electronic petitions;
- vote for the introduction of new ideas for the improvement of ACMH users;
- connection of several apartments;
- news;
- view statistics on utility costs;
- view event statistics, cost of spending on the improvement of the city of Vinnitsa;
- visualization of the results.

5.2. Basic technical requirements for development.

5.2.1. Requirements for the software platform:

- VS Code IDE.

5.2.2. Operating system conditions:

- work on standard PCs in premises with standard conditions;
- the possibility of a 24-hour system operation.

6. Stages and stages of development.

6.1 Review of technologies, models and methods of creating the concept of information system 10.10.2019

6.2 Designing and realization of the concept of the information system of the personal cabinet of the resident of Vinnytsia 12.10.2019

6.3 The business model of information system 20.10.2019

6.4 Approval the results of the study 22.11.2019

6.5 Publications 05.12.2019

6.6 Design of explanatory note, graphic material and presentation 05.12.2019

7. The order of control and acceptance.

7.1. The progress of master's qualification work is supervised by the head of the work. The border control should be conducted until November 30, 2019.

7.2. The project certification is carried out on preliminary protection. Preliminary defense of master's qualification work is to be held until December 7, 2019.

7.3. The final decision on the assessment of the quality of the master's qualification work is taken at a meeting of the DEC. The defense of master's qualification work is to hold December 12, 2019.

Appendix B. Listing of code of program

Payment controller:

```
using System.Collections.Generic;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Stripe;
using VinnytsiaCard.API.DTO;
using VinnytsiaCard.API.Models;
using VinnytsiaCard.API.Repositories;
using VinnytsiaCard.Models;

namespace VinnytsiaCard.API.Controllers
{
   [Authorize]
   [Route("api/[controller]")]
   [ApiController]
   public class PaymentController : ControllerBase
   {
      IAuthRepository _authRepository;
      IPaymentRepository _paymentRepository;
      IProfileRepository _profileRepository;
      public PaymentController(IAuthRepository authRepository, IProfileRepository
profileRepository, IPaymentRepository paymentRepository)
      {
         _authRepository = authRepository;
         _profileRepository = profileRepository;
         _paymentRepository = paymentRepository;
      }

      [HttpGet("GetUserPayments/{pageSize}/{pageNumber}")]
      public async Task<IActionResult> GetUserPaymentsAsync(int pageSize, int pageNumber)
      {
         int id = 0;
         string userName = User.Identity.Name;

         if (!string.IsNullOrEmpty(userName))
         {
            User userData = await _profileRepository.GetUserData(userName);
            id = userData.Id;

            PaymentDto userPayments = _paymentRepository.GetUserPaymentsById(id, pageSize,
pageNumber);

            return Ok(userPayments);
         }
```

```
    return BadRequest();
}

[HttpPost("AddPayments")]
public async Task<IActionResult> AddPaymentAsync(Payment payment)
{
    int id = 0;
    string userName = User.Identity.Name;

    if (!string.IsNullOrEmpty(userName))
    {
        User userData = await _profileRepository.GetUserData(userName);
        payment.UserId = userData.Id;

        _paymentRepository.AddNewPayment(payment);

        return Ok($"Payment for a user '{ id }' was added successfully");
    }

    return BadRequest("The payment was not added.");
}

[HttpGet("GetUserMainBills")]
public async Task<IActionResult> GetUserPaymentsAsync()
{
    string userName = User.Identity.Name;

    if (await _authRepository.UserExist(userName))
    {
        List<MainBillDto> userBills = _paymentRepository.GetUserMainBills();

        if (userBills == null)
        {
            return BadRequest($"Can't receive users main bills.");
        }

        return Ok(userBills);
    }

    return BadRequest($"Invalid user");
}

[HttpGet("GetBillsTypes")]
public async Task<IActionResult> GetBillsTypesAsync()
{
    string userName = User.Identity.Name;

    if (await _authRepository.UserExist(userName))
```

```
        {
            List<BillType> billsTypesList = _paymentRepository.GetAllBillsTypes();

            if (billsTypesList == null)
            {
                return BadRequest($"Can't receive bills types.");
            }

            return Ok(billsTypesList);
        }

        return BadRequest($"Invalid user");
    }

    [HttpGet("GetEnterprisesTypes")]
    public async Task<IActionResult> GetEnterprisesTypesAsync()
    {
        string userName = User.Identity.Name;

        if (await _authRepository.UserExist(userName))
        {
            List<EnterpriseType> billsTypesList = _paymentRepository.GetAllEnterprisesTypes();

            if (billsTypesList == null)
            {
                return BadRequest($"Can't receive enterprises types.");
            }

            return Ok(billsTypesList);
        }

        return BadRequest($"Invalid user");
    }

    [HttpGet("GetEnterprisesNames/{enterpriseTypeId}")]
    public async Task<IActionResult> GetEnterprisesNames(int enterpriseTypeId)
    {
        string userName = User.Identity.Name;

        if (await _authRepository.UserExist(userName))
        {
            List<Enterprise> billsTypesList =
_paymentRepository.GetEnterprisesById(enterpriseTypeId);

            if (billsTypesList == null)
            {
                return BadRequest($"Can't receive enterprises name.");
            }
```

```
            return Ok(billsTypesList);
        }

        return BadRequest($"Invalid user");
    }

    [HttpGet("GetEnterpriseServicesNames/{enterpriseId}")]
    public async Task<IActionResult> GetEnterpriseServiceNames(int enterpriseId)
    {
        string userName = User.Identity.Name;

        if (await _authRepository.UserExist(userName))
        {
            List<Service> billsTypesList = await
_paymentRepository.GetEnterpriseServiceByIdAsync(enterpriseId, userName);

            if (billsTypesList == null)
            {
                return BadRequest($"Can't receive enterprises services.");
            }

            return Ok(billsTypesList);
        }

        return BadRequest($"Invalid user");
    }

    [HttpPost("UpdatMainBills")]
    public async Task<IActionResult> UpdateMainBillsAsync(UpdateMainBillsDto bills)
    {
        string userName = User.Identity.Name;

        if (!string.IsNullOrEmpty(userName))
        {
            User userData = await _profileRepository.GetUserData(userName);

            bills = await _paymentRepository.UpdateMainBills(bills, userName);
            if (bills != null)
            {
                return Ok(bills);
            }
        }

        return BadRequest("The bills were not updated.");
    }

    [HttpGet("GetUserOtherBills")]
    public async Task<IActionResult> GetUserOtherPaymentsAsync()
    {
```

```
            string userName = User.Identity.Name;

            if (await _authRepository.UserExist(userName))
            {
                List<UserBillsDto> userBills = _paymentRepository.GetUserOtherBills(userName);

                if (userBills == null)
                {
                    return BadRequest($"Can't receive users bills.");
                }

                return Ok(userBills);
            }

            return BadRequest($"Invalid user");
        }


        [HttpPost("AddUserAdditionalBill")]
        public async Task<IActionResult> AddUserAdditionalBill(AddAdditionalBillDto
addAdditionalBill)
        {
            string userName = User.Identity.Name;

            if (await _authRepository.UserExist(userName))
            {
                UserAdditionalBill userAdditionalBill = await
_paymentRepository.AddUserAdditionalBillAsync(addAdditionalBill, userName);

                if (userAdditionalBill != null)
                {
                    return Ok(true);
                }
            }

            return BadRequest("Invalid transaction");
        }

        [HttpGet("GetUserAdditionalBills")]
        public async Task<IActionResult> GetUserAdditionalBills()
        {
            string userName = User.Identity.Name;

            if (await _authRepository.UserExist(userName))
            {
                AdditionalBillDto additionalBillDto = await
_paymentRepository.GetUserAdditionalBillAsync(userName);

                if (additionalBillDto != null)
```

```
            {
                return Ok(additionalBillDto);
            }
        }

        return BadRequest("Invalid transaction");
    }

    [HttpPost("Pay")]
    public async Task<IActionResult> PayAsync(PayDto pay)
    {
        string userName = User.Identity.Name;

        if (await _authRepository.UserExist(userName))
        {
            Payment payment = await _paymentRepository.PayAsync(pay, userName);

            if (payment != null)
            {
                return Ok(true);
            }
        }

        return BadRequest("Invalid transaction");
    }
  }
}
```

Entertainment module:

```
import { Component, OnInit, ViewChild } from '@angular/core';
import { SmartCinemaRequestPayload } from '../models/smartCinemaRequestPayload.model';
import { SmartCinemaGetMoviesResponse, Soon, Rent } from
'../models/smartCinemaGetMoviesResponse.model';
import { PageEvent, MatPaginator } from '@angular/material';
import { DataService } from '../data.service';
import { EverythingRequest, Languages, SortBys } from '../models/everythingRequest.model';

@Component({
  selector: 'app-entertainment',
  templateUrl: './entertainment.component.html',
  styleUrls: ['./entertainment.component.css']
})

export class EntertainmentComponent implements OnInit {
  innerWidth: number = 0;
  moviesRentLength: number = 0;
  moviesRentPageSize: number = 0;
  moviesSoonLength: number = 0;
```

```
  moviesSoonPageSize: number = 0;
  smartCinemaRequestPayload: SmartCinemaRequestPayload;
  smartCinemaGetMoviesRentResponse: Rent[];
  smartCinemaGetMoviesSoonResponse: Soon[];

  pageEvent: PageEvent;
  pageSizeOptions: number[] = [1, 2, 3, 4, 5];
  @ViewChild('paginatorRent', {static: false}) paginator: MatPaginator;
  @ViewChild('paginatorSoon', {static: false}) paginator2: MatPaginator;

  constructor(private dataService: DataService) {
    this.innerWidth = window.innerWidth;

    if (innerWidth > 1614) {
      this.moviesRentPageSize = 4;
      this.moviesSoonPageSize = 4;
    }
    if (innerWidth < 1614 && innerWidth > 1290) {
      this.moviesRentPageSize = 3;
      this.moviesSoonPageSize = 3;
    }
    if (innerWidth < 1290) {
      this.moviesRentPageSize = 2;
      this.moviesSoonPageSize = 2;
    }

    this.getMoviesRent(0, this.moviesRentPageSize);
    this.getMoviesSoon(0, this.moviesSoonPageSize);
  }

  ngOnInit() {

  }

  ngAfterViewInit() {
    this.paginator.pageIndex = 1;
    this.paginator2.pageIndex = 1;
  }

  getMoviesRent(pageNumber: number, pageSize: number) {
    this.smartCinemaRequestPayload = new SmartCinemaRequestPayload(1);
    this.dataService.getMovies(this.smartCinemaRequestPayload).subscribe((data:
SmartCinemaGetMoviesResponse) => {
      this.smartCinemaGetMoviesRentResponse = data.rent;
      this.paginator.length = this.smartCinemaGetMoviesRentResponse.length;

      this.smartCinemaGetMoviesRentResponse.map(x => {
        x.seo_inline.url = "https://smartcinema.ua" + x.seo_inline.url;
        x.poster = "https://smartcinema.ua" + x.poster;
```

```
      });

      this.smartCinemaGetMoviesRentResponse =
this.smartCinemaGetMoviesRentResponse.slice(pageSize * pageNumber, (pageSize *
pageNumber) + pageSize)
      console.log(this.smartCinemaGetMoviesRentResponse);
    }, err => {
      console.log(err);
    });
  }

  getMoviesSoon(pageNumber: number, pageSize: number) {
    this.smartCinemaRequestPayload = new SmartCinemaRequestPayload(1);
    this.dataService.getMovies(this.smartCinemaRequestPayload).subscribe((data:
SmartCinemaGetMoviesResponse) => {
      this.smartCinemaGetMoviesSoonResponse = data.soon;
      this.paginator2.length = this.smartCinemaGetMoviesSoonResponse.length;
      this.smartCinemaGetMoviesSoonResponse.map(x => {
        x.seo_inline.url = "https://smartcinema.ua" + x.seo_inline.url;
        x.poster = "https://smartcinema.ua" + x.poster;
      });

      this.smartCinemaGetMoviesSoonResponse =
this.smartCinemaGetMoviesSoonResponse.slice(pageSize * pageNumber, (pageSize *
pageNumber) + pageSize)
      console.log(this.smartCinemaGetMoviesSoonResponse);
    }, err => {
      console.log(err);
    });
  }

  paginateMoviesRent(event: PageEvent) {
    console.log(event.pageIndex);
    this.getMoviesRent(event.pageIndex, event.pageSize);
  }

  paginateMoviesSoon(event: PageEvent) {
    console.log(event.pageIndex);
    this.getMoviesSoon(event.pageIndex, event.pageSize);
  }

  setPageSizeOptions(setPageSizeOptionsInput: string) {
    this.pageSizeOptions = setPageSizeOptionsInput.split(',').map(str => +str);
  }
}
```

Appendix C

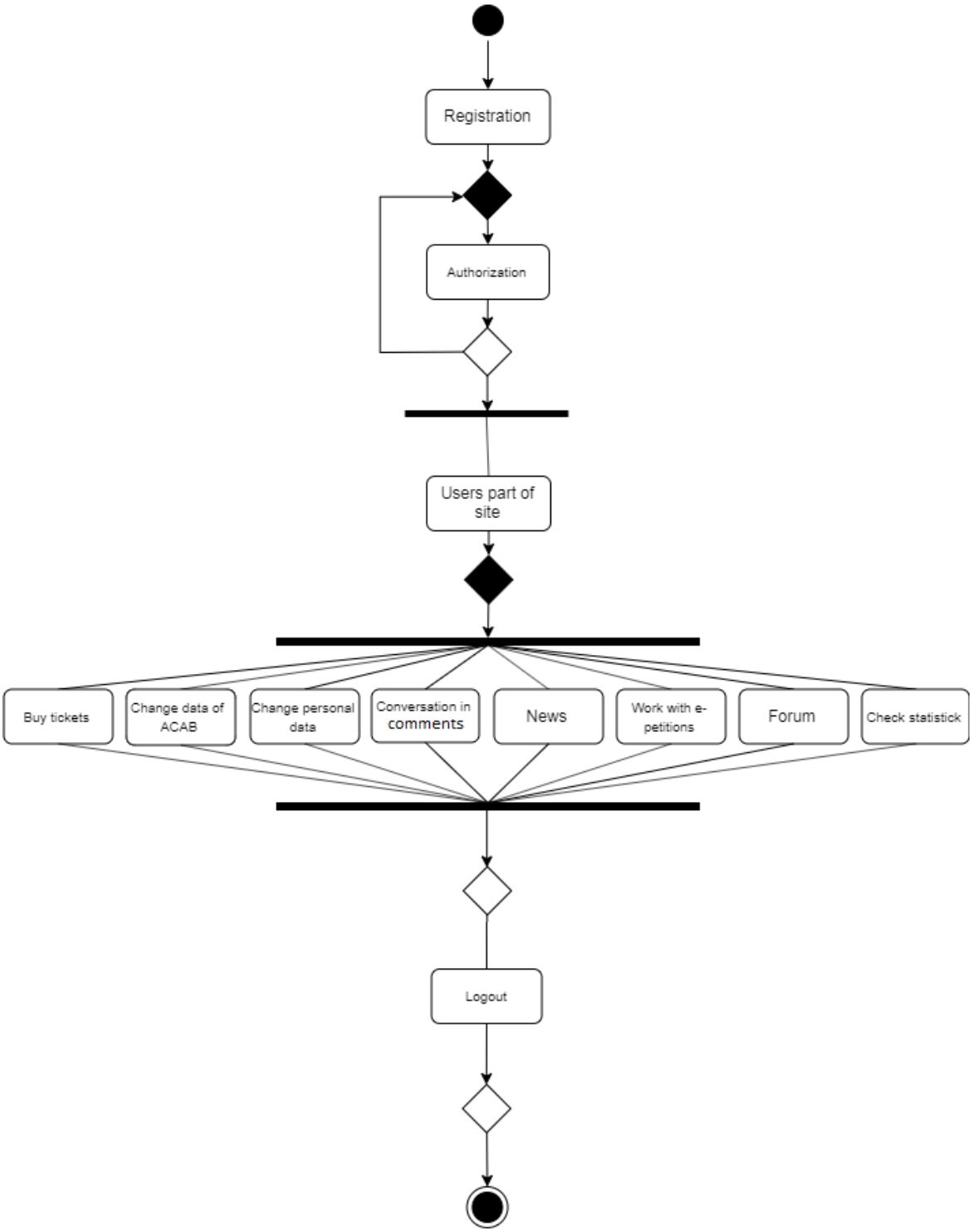
(obligatory)

List of graphical materials

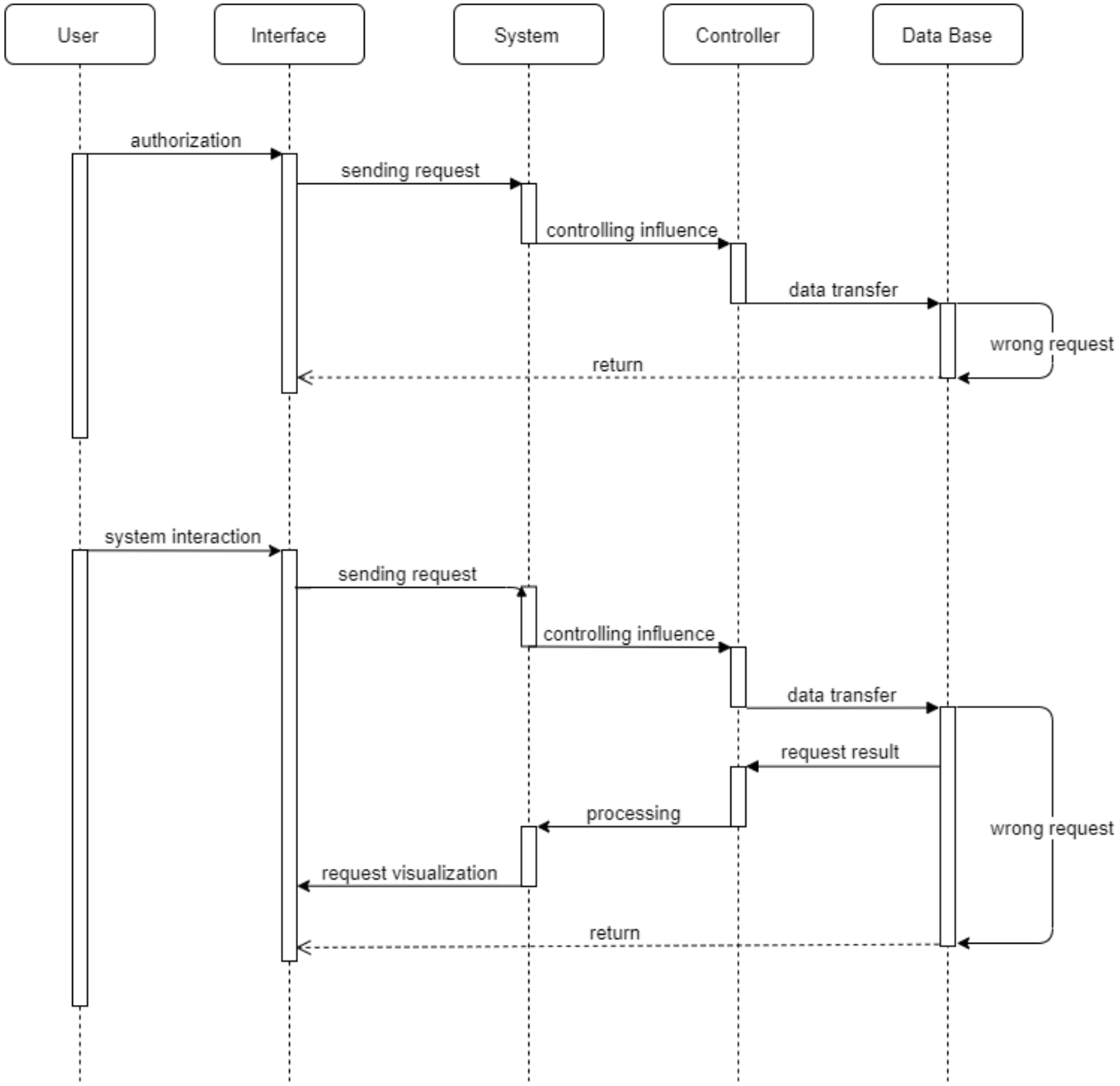
Head of the AIIT department	(signature)	Dr.Sc., Prof R. Kvetny (name, surname, degree, academic status)
Scientific supervisor	(signature)	PhD., As. Prof O. Kovaliuk (name, surname, degree, academic status)
Technical control	(signature)	PhD., As. Prof O. Kovaliuk (name, surname, degree, academic status)
Regulatory control	(signature)	PhD., As.Prof O. Kovaliuk (name, surname, degree, academic status)
Reviewer	(signature)	PhD., As.Prof O. Kryvohubchenko (name, surname, degree, academic status)
Student of the 3ACIT-18m group	(signature)	V. Plysak (name, surname, degree, academic status)

C.1 UML USE CASE DIAGRAM

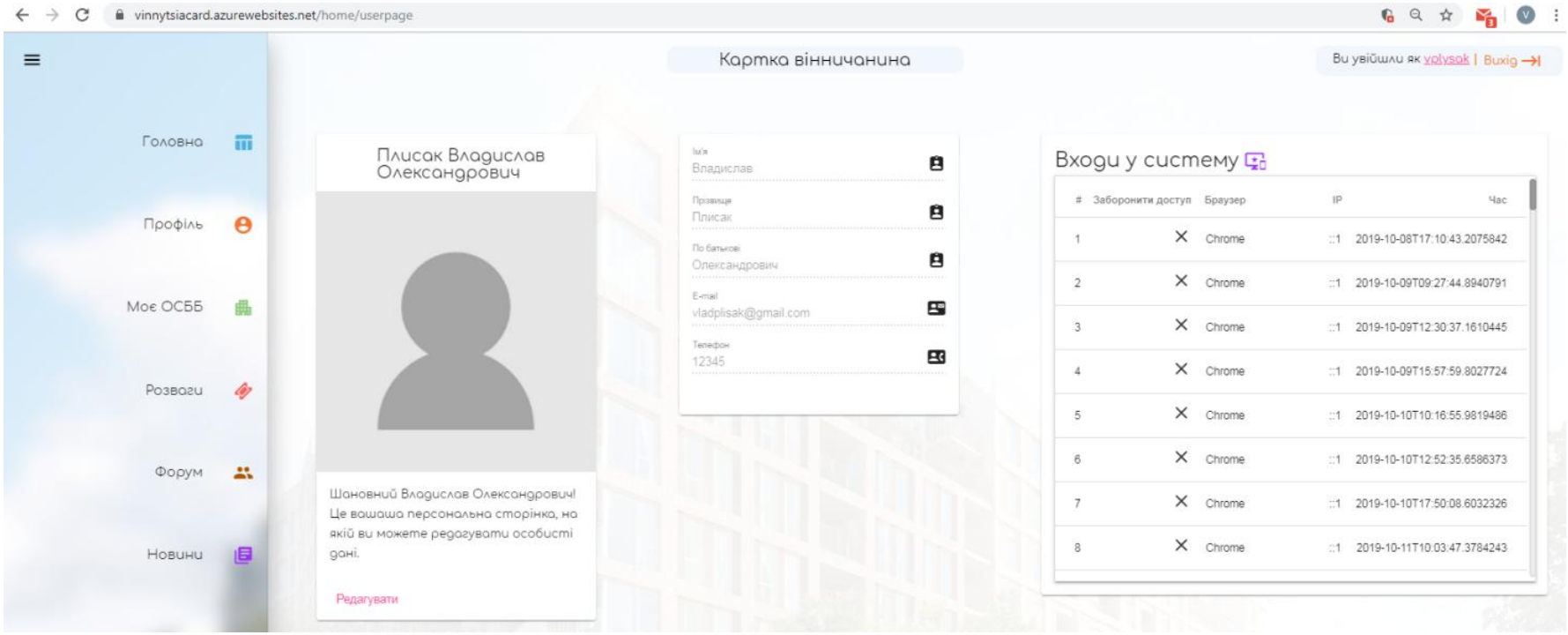
C.2 UML ACTIVITY DIAGRAM



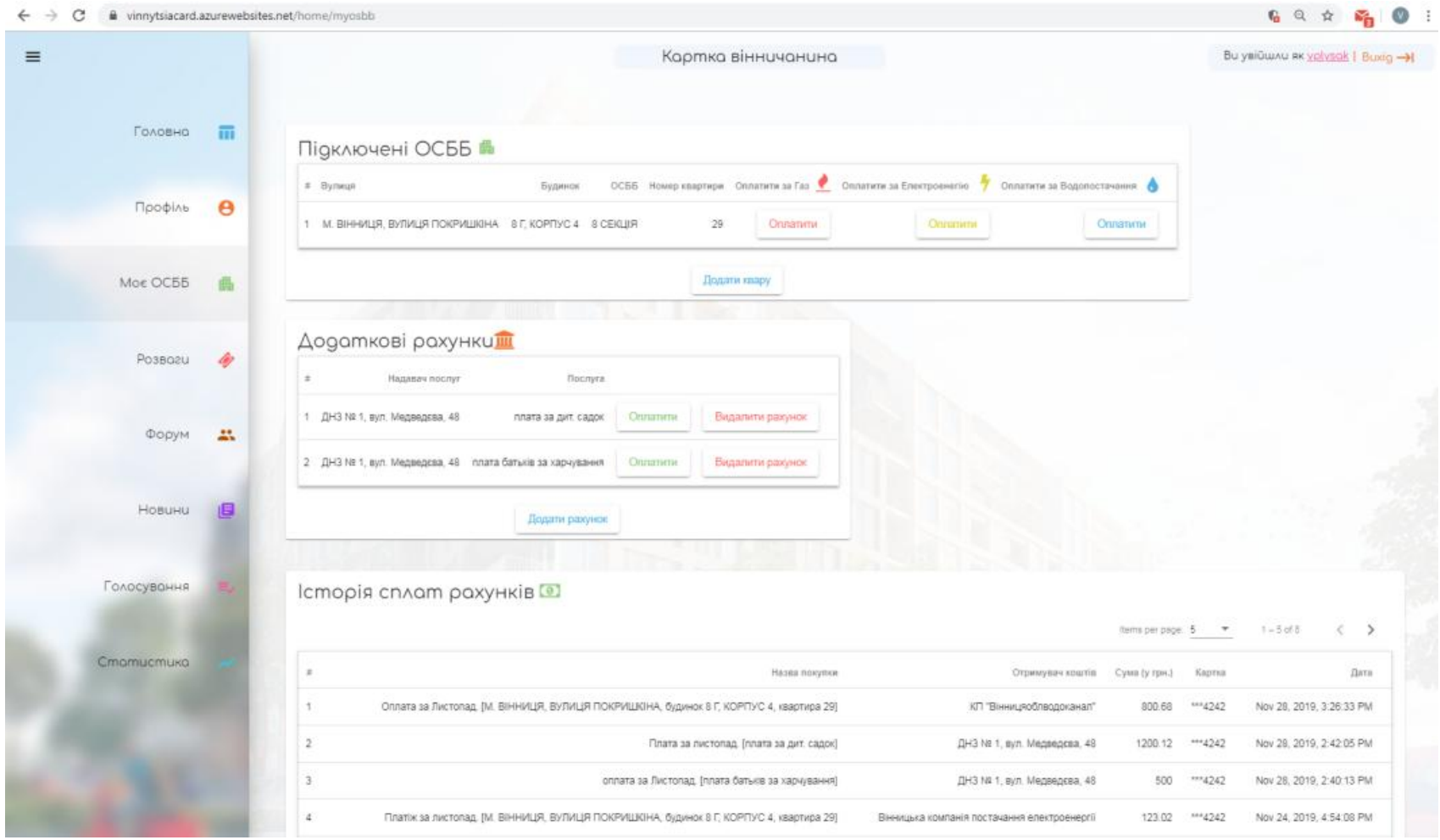
C.3 UML SEQUENCE DIAGRAM



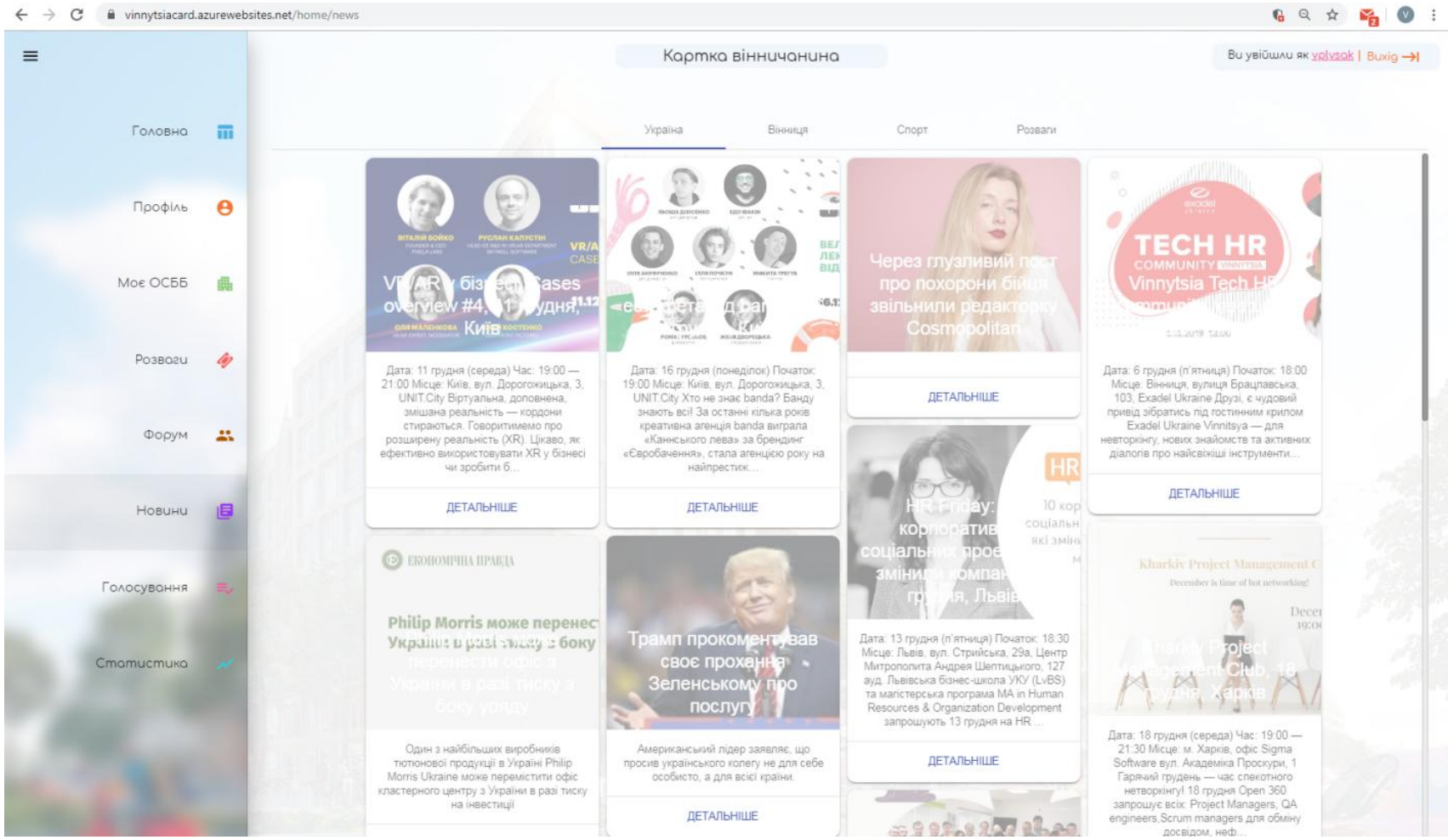
C.4 Screen view of the user's page in information system



C.5 Screen view of the ACAB page in information system



C.6 Screen view of the news page in information system



Presentation

DEVELOPMENT OF THE INFORMATION SYSTEM «PERSONAL ACCOUNT OF A RESIDENT OF VINNYTSIA»

Part 1. Development of the concept of the system

Demian Sembrat

Part 2. Development of data analysis and accounting subsystem

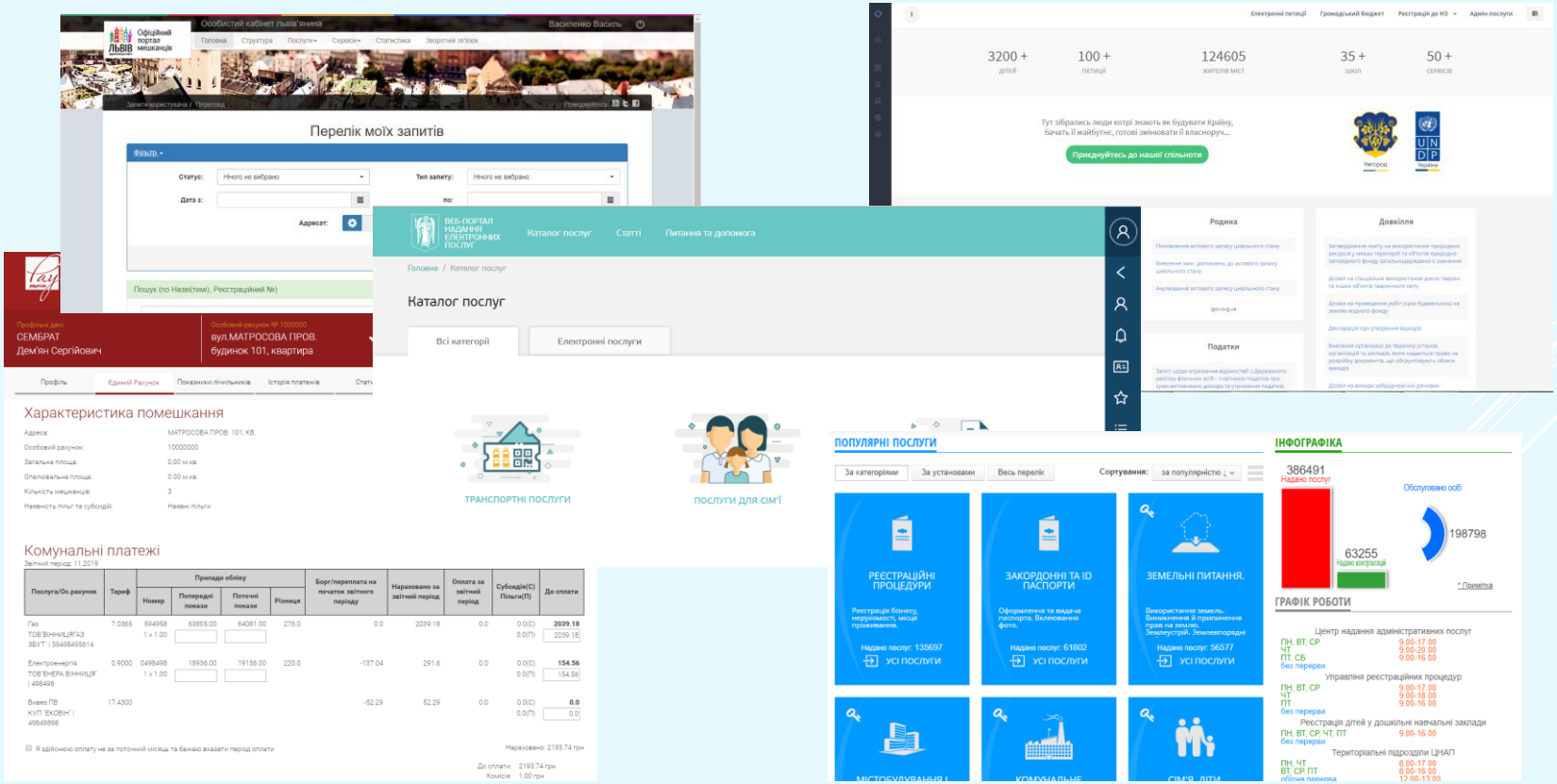
Vladyslav Kyselov

Part 3. Development of services and tools subsystem

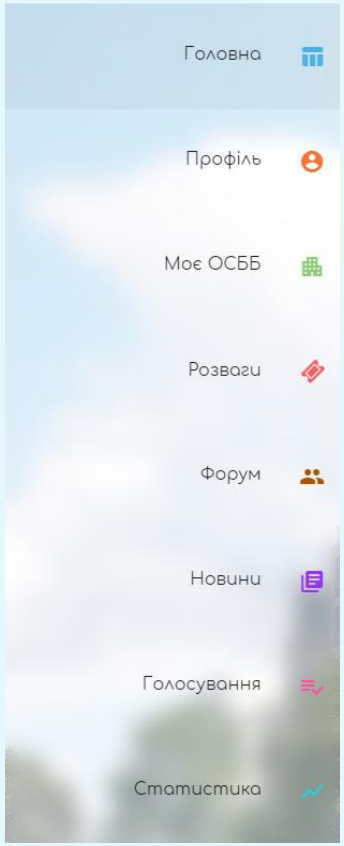
Vladyslav Plysak

Slide about existing information system analysis

EXISTING INFORMATION SYSTEMS ANALYSIS



Slide about main functions of the information system



MAIN FUNCTIONS OF THE INFORMATION SYSTEM

– apartments management;

– payment for utilities;

– forum for communication;

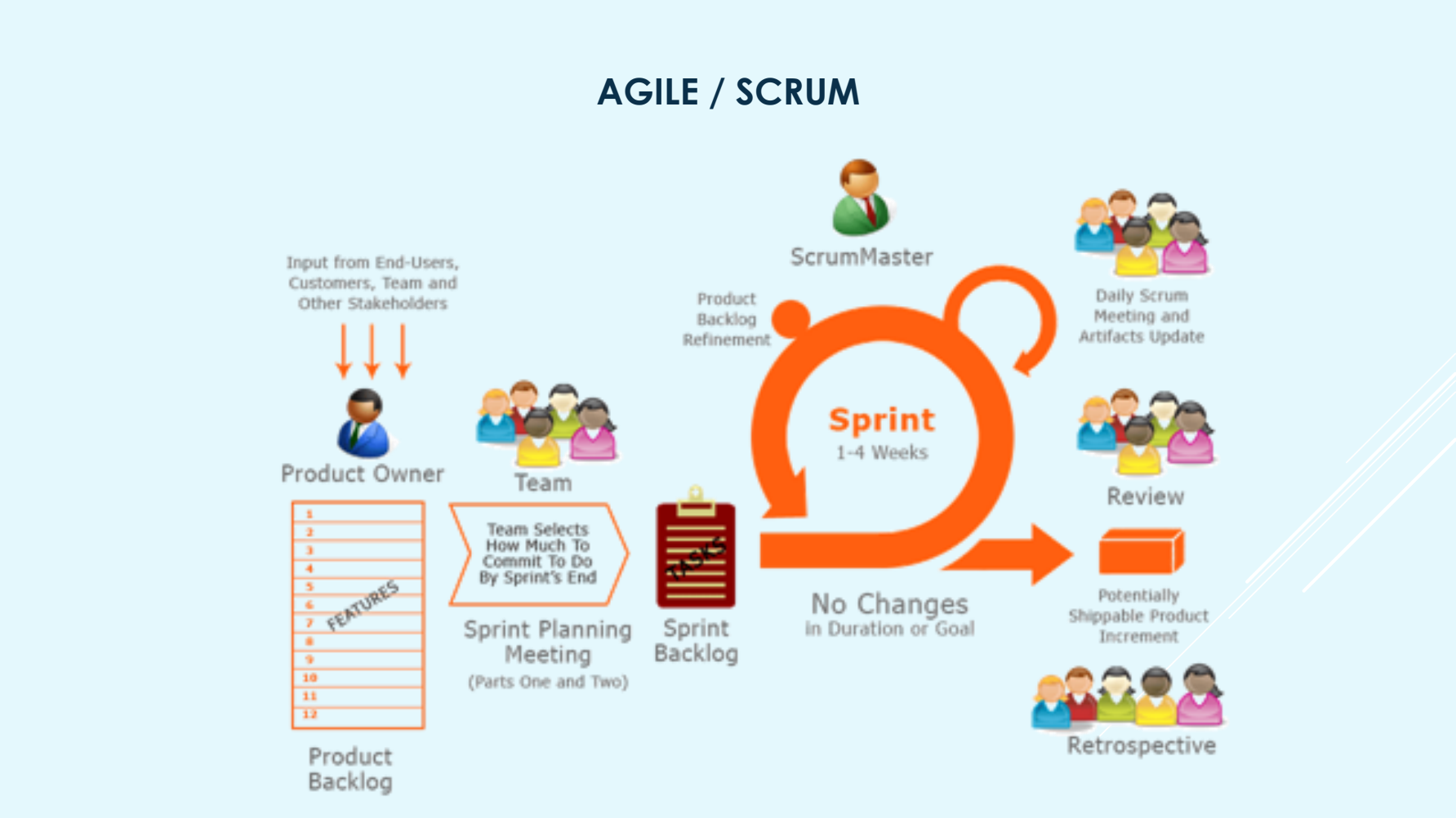
– the ability to submit petitions and votes to introduce new ideas;

– payment of tickets for cultural events;

– statistics;

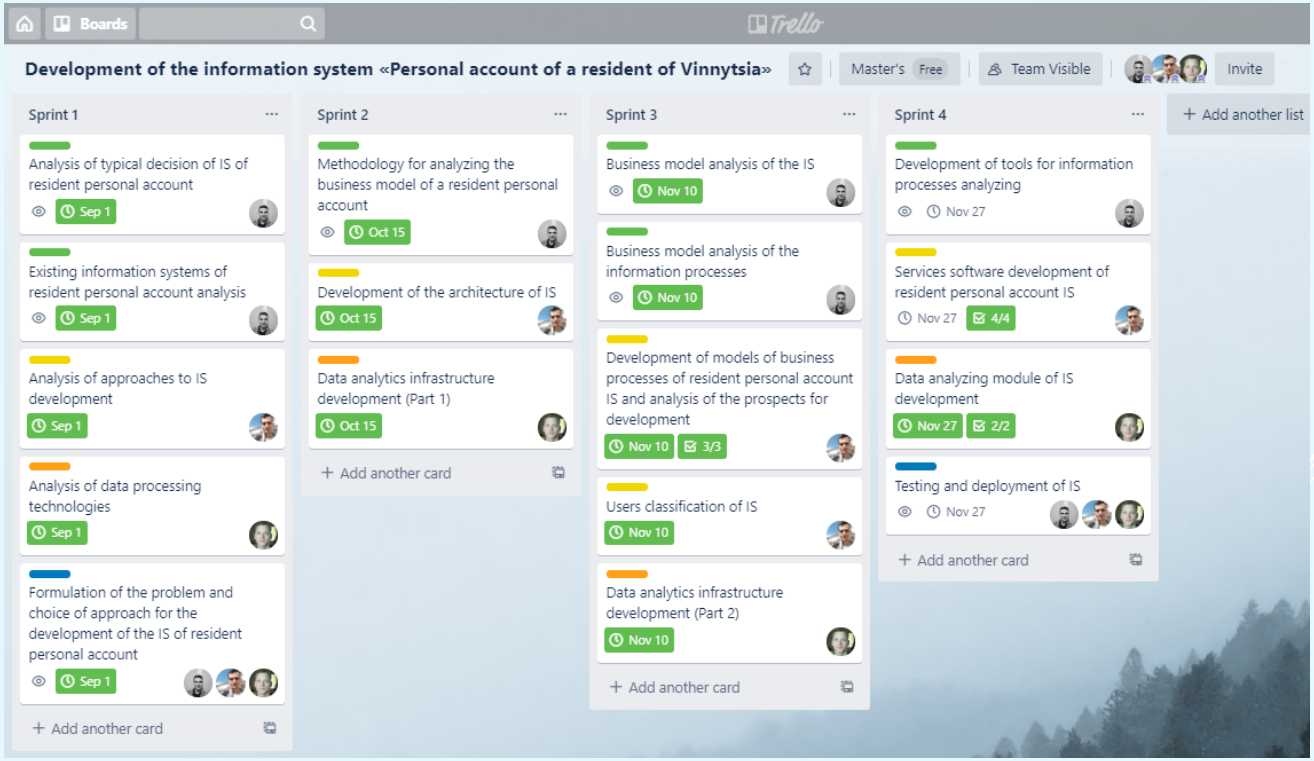
– news.

Slide about AGILE/SCRUM metrology



Slide about project management

PROJECT MANAGEMENT



Slide about innovations and advaneges

DEVELOPMENT OF SERVICES AND TOOLS SUBSYSTEM (VLADYSLAV PLYSAK)

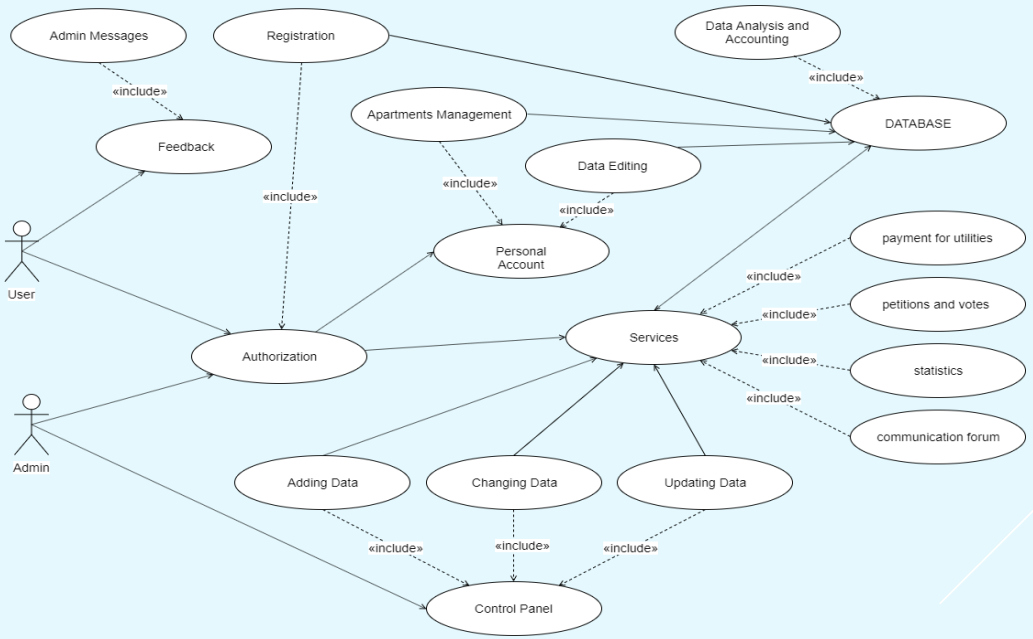
Innovation and Advantages

Implemented services in information system:
1) Payment for utilities.
2) Payment for additional services.
3) Forum.
4) Signing electronic petitions.
5) Vote for the introduction of new ideas for the improvement of ACMH users.
6) Connection of several apartments.
7) News
8) View statistics on utility costs.
9) View event statistics, cost of spending on the improvement of the city of Vinnitsa.

Slide about UML use-case diagram

DEVELOPMENT OF SERVICES AND TOOLS SUBSYSTEM

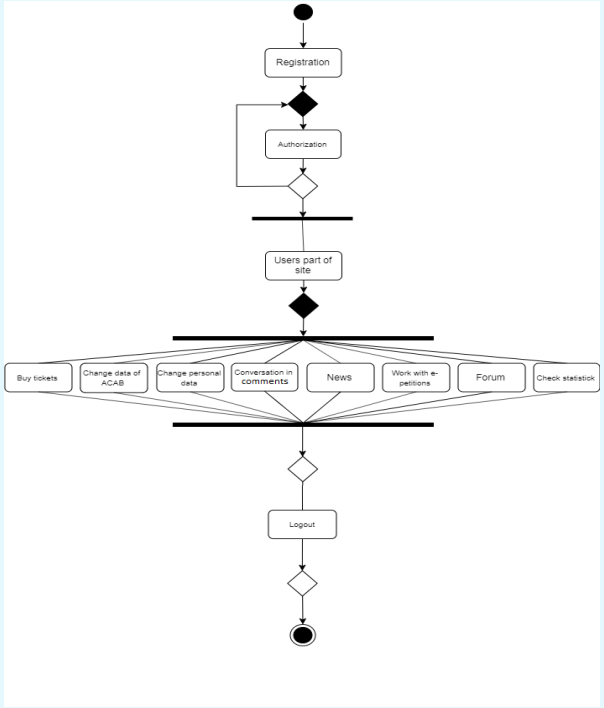
UML Use-Case diagram



Slide about UML activity diagram

DEVELOPMENT OF SERVICES AND TOOLS SUBSYSTEM

UML activity diagram

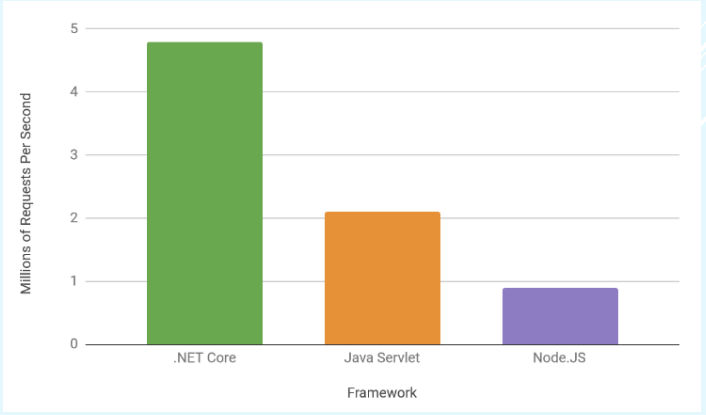


Slide about UML developing of services and tools

DEVELOPMENT OF SERVICES AND TOOLS SUBSYSTEM

Information system consist of:
- .Net Core 2.2 Web Api project (Back-end)
- Angular 8 project (Front-end)

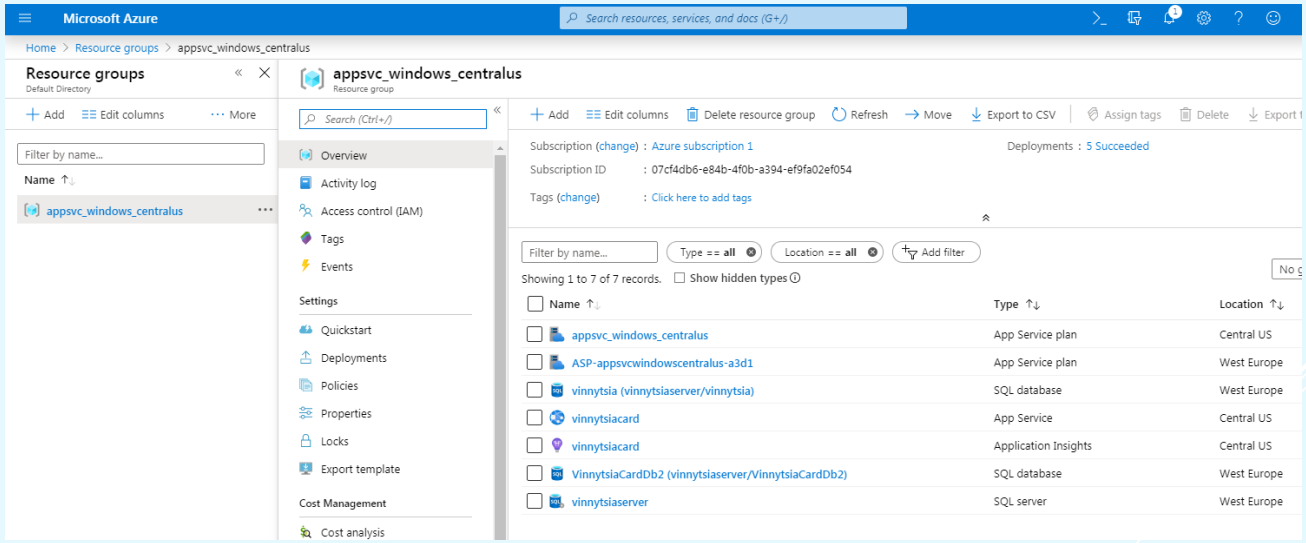
Benefits:
-Cross-platform
-Microsoft Support
-Simple Application Maintenance
-Web API support
-Scaling
-Open Source
-Performance



Slide about information system of Azure

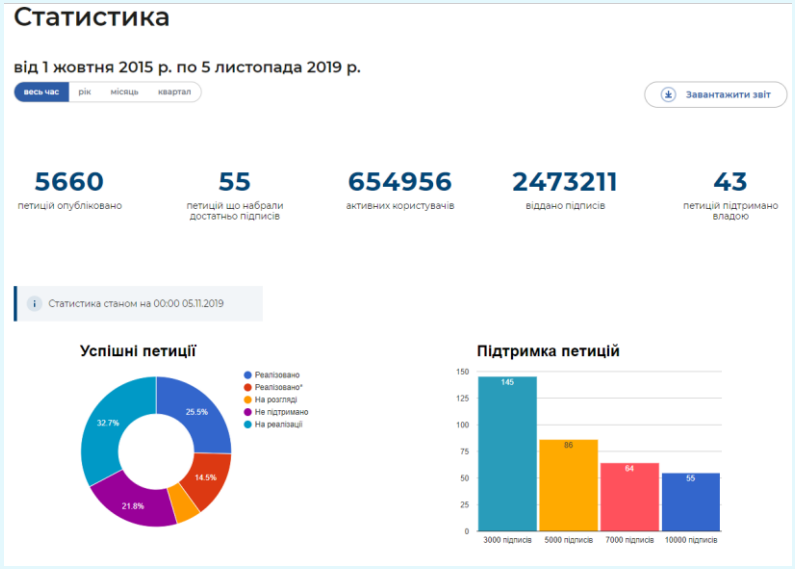
DEVELOPMENT OF SERVICES AND TOOLS SUBSYSTEM

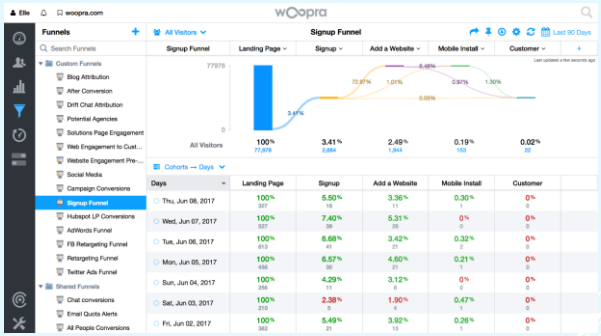
Information system in Azure

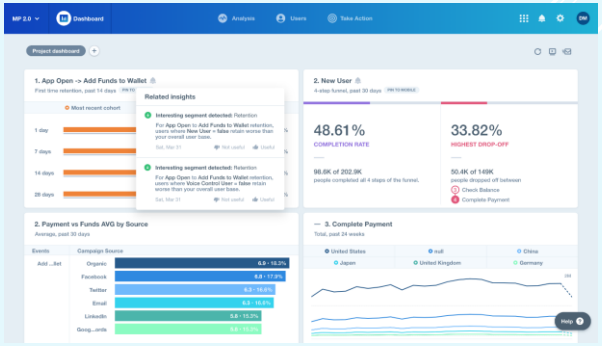


Slide about information system of statistic

DEVELOPMENT OF DATA ANALYSIS AND ACCOUNTING SUBSYSTEM (VLADYSLAV KYSELOV)

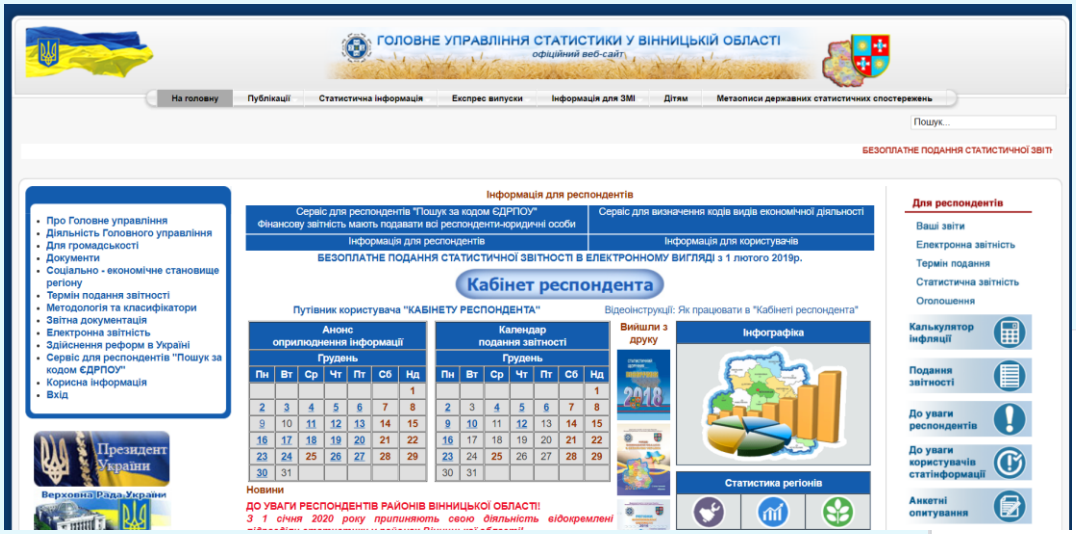


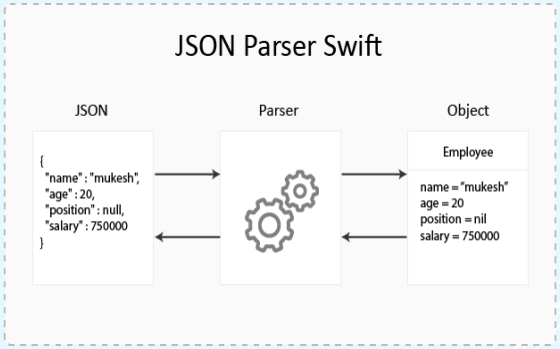


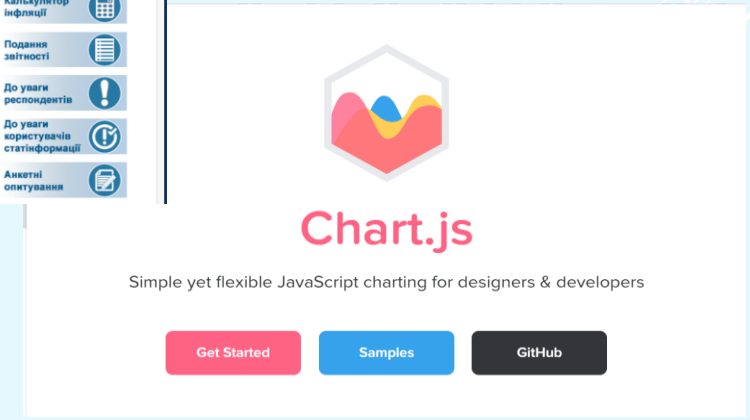


Slide about information system of statistic

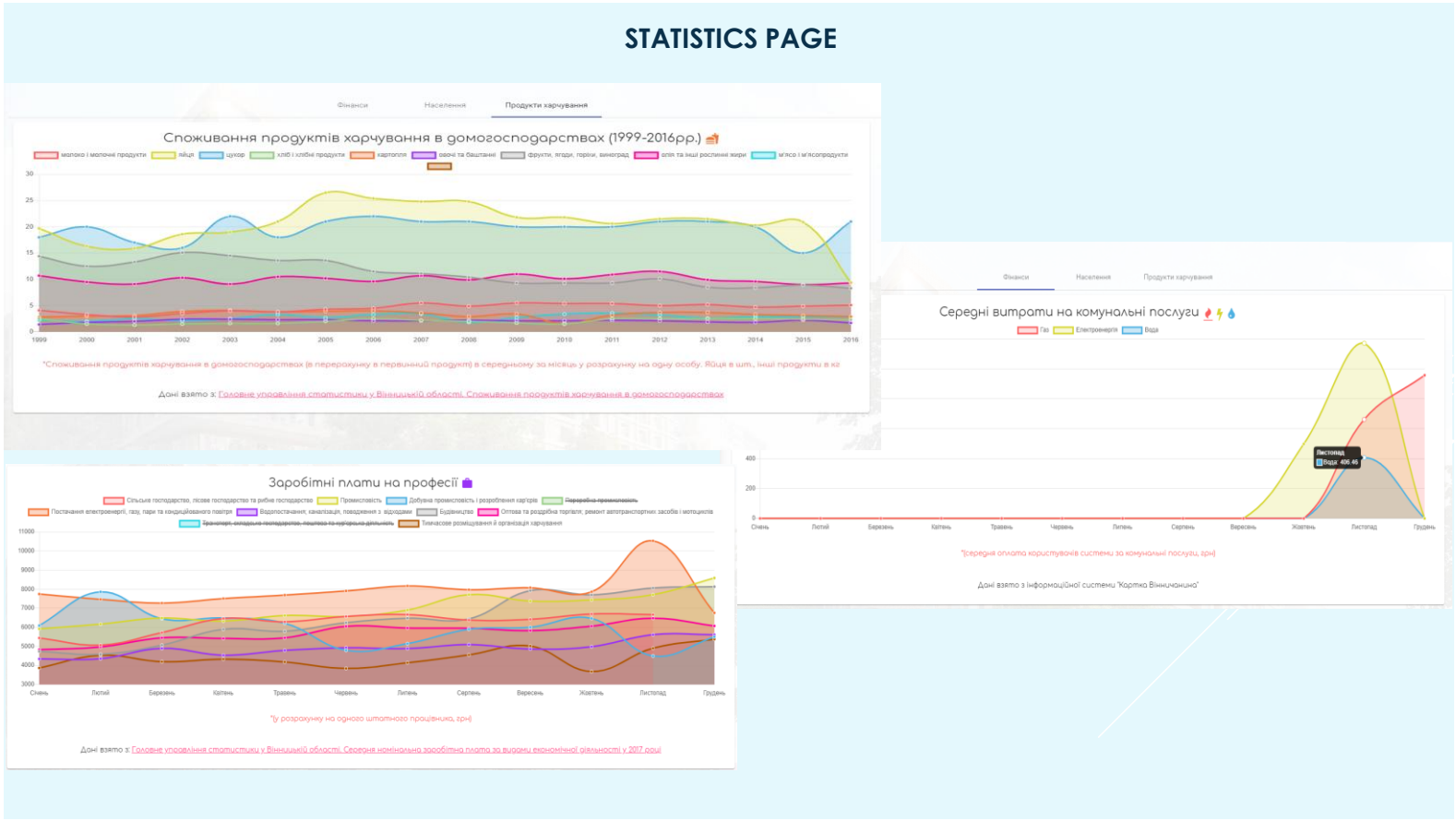
DEVELOPMENT OF DATA ANALYSIS AND ACCOUNTING SUBSYSTEM





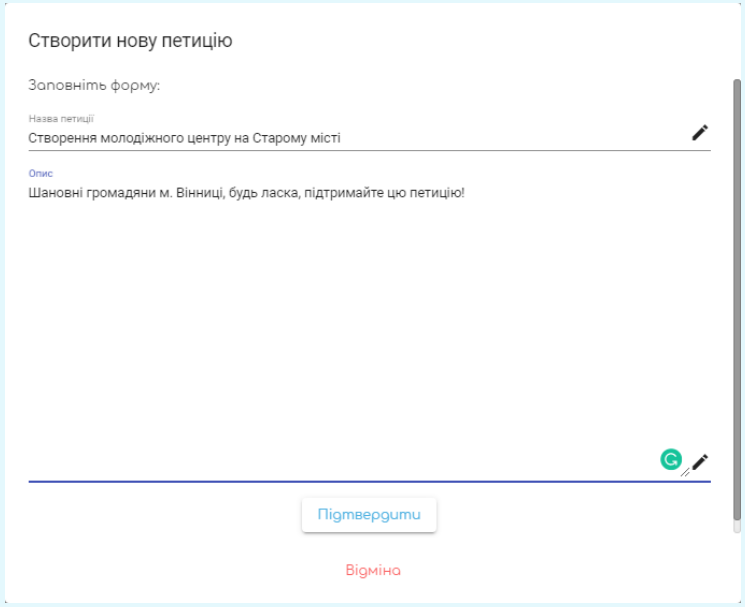


Slide about information system of statistic



Slide about chat bot of information system

TELEGRAM-BOT IMPLEMENTATION



THANK YOU FOR ATTENTION!